Bi-objective optimization for the container terminal integrated planning

Ming Liu [a], Chung-Yee Lee [b,*], Zizhen Zhang [c], Chengbin Chu [a,d]

[a] School of Economics & Management, Tongji University, Shanghai 200092, P.R. China
[b] Department of Industrial Engineering & Logistics Management, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
[c] School of data & computer science, Sun Yat-sen University, Gangzhou 510275, P.R. China
[d] Laboratoire Génie Industriel, Centrale Supélec, Université Paris-Saclay, Grande Voie des Vignes, 92290 Châtenay-Malabry, France

A B S T R A C T

In this paper, we study the joint optimization of the tactical berth allocation and the tactical yard allocation in container terminals, which typically consist of berth side and yard side operations. The studied two objectives are: (i) the minimization of the violation of the vessels' expected turnaround time windows with the purpose of meeting the timetables published by shipping liners, and (ii) the minimization of the total yard transportation distance with the aim to lower terminal operational cost. We propose a bi-objective integer program which can comprehensively address the import, export and transshipment tasks in port daily practice. Traditionally, a container transshipment task is performed as a couple of import and export tasks, called *indirect-transshipment mode*, in which the transit container are needed to be temporally stored in the yard. As the way of transferring containers directly from the incoming vessel to the outgoing vessel, called *direct-transshipment mode*, has potential to save yard storage resources, the proposed model also incorporates both indirect- and direct-transshipment modes. To produce Pareto solutions efficiently, we devise heuristic approaches. Numerical experiments have been conducted to demonstrate the efficiency of the approaches.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

World container port throughput increased by an estimated 3.8% to 601.8 million 20-foot equivalent units (TEUs) in 2012. The world fleet has more than doubled since 2001, reaching 1.63 billion deadweight tons in 2013. In 2013, the worldwide container trade accounts for about 22% of the 6.7 billion tons of dry-cargo trade, and all loads are being transported by vessels via container terminals. Recent statistics show that total container trade volumes reached 160 million TEUs in 2013 with a growth of 4.6% (c.f., UNCTAD (2015)). In line with the increase of global economy and containerized trade, one of general trends is that seaport experiences a strong growth in transit container volume, which may spur greater inter-port competition and increased port performance. Port managers have been making continuous effect on increasing resource utilization and reducing operational cost, to maintain margins as well as meet the timetables published by shipping companies.

---

* Corresponding author.
  E-mail address: cylee@ust.hk (C.-Y. Lee).

In world-class ports, common container activities include import, export and transshipment container tasks. Even though container ports can be generally divided into gateway ports and transshipment hubs, but also the boundary between these two kinds of ports is fuzzy. This is because (i) even in the worldwide largest transshipment hubs, such as Singapore Port, the import and export tasks also pose around 20% among all tasks, and (ii) in the gateway ports, such as Shanghai Port, the transshipment tasks occupy about 20%. Therefore, useful management systems or operational optimization softwares for container ports should be capable to handle all the three kinds of container tasks.

In recent trends of container port development, the optimization of transshipment modes receives much attention. Transshipment activities can be performed in two ways: (i) direct-transshipment mode, which transfers transit containers from one vessel to the connecting vessel directly without yard storage but requiring contingent vessel berthing times (see Fig. 3a), and (ii) indirect-transshipment mode, which temporally stores the transit containers in the yard and then load them to the connecting vessels. For comparison, direct-transshipment mode could save yard resources but require contingent berthing time of vessels, whereas indirect-transshipment mode could relieve the contingent vessel berthing time restriction but consume yard storage and incur yard transportation cost. Port managers prefer performing a transshipment task via direct-transshipment mode, as it requires no yard storage and thus helps relieve port congestions and reduce yard transportation cost, at the price of violating timetables published by shipping liners. According to our interviews with Shanghai Port's managers, direct-transshipment mode is applied in port daily practice and it possesses a relatively small percentage of all transit activities. Therefore, a better container terminal planning system should well balance the violation of timetables of shipping liners and the cost of yard transportation distance.

Generally speaking, the terminal operations planning system can be classified into two levels: tactical and operational (Moorthy and Teo (2006)). Tactical problems include the tactical berth allocation problem (TBAP) and the tactical yard allocation problem (TYAP). In the TBAP, the terminal managers try to satisfy the expected turnaround time windows or intervals of vessels by allocating berths and quay cranes. With the consideration of transshipment modes, port operators also need to determine the mode (i.e., direct-transshipment or indirect-transshipment mode) for each transshipment activity. In the TYAP, which is also known as yard template planning (Zhen et al. (2011)), the consignment strategy is widely utilized and only indirect-transshipment mode is adopted. This strategy stores transit containers to be loaded on a specific vessel at the dedicated storage locations allocated for that vessel. In the TYAP, terminal managers attempt to minimize the yard transportation distance.

The TYAP and the TBAP are intertwined. On one hand, the yard storage allocated in the TYAP affects the best berthing positions for vessels; on the other hand, the berthing positions allocated in the TBAP impact the assignment of yard storage locations to vessels. In particular, the direct-transshipment decisions must be made under the integration framework of the TYAP and the TBAP, because if contingent berthing times can be allocated for two vessels connected by a transshipment container task (in the TBAP) the direct-transshipment mode can be applied which could significantly reduce yard transportation distance (in the TYAP); otherwise, the terminal operators must assign some appropriate yard storage for transit containers which must incur the increase of yard transportation distance (in the TYAP).

Motivated by the need of world-class container ports, i.e., to balance the violation of timetables of shipping liners and the yard transportation distance, this work investigates the joint planning of tactical berth allocation and tactical yard allocation, and solve the integration problem via bi-objective optimization approaches. On one hand, port operators make schedules, greatly conforming with timetables of shipping liners, to increase their satisfaction. On the other hand, port managers attempt to lower port operational cost. These two aims are measured in dimensions of (timetable violation) times, and (truck traveling) distances, respectively, according to our interviews with Shanghai Port's managers. To balance these two objective values, bi-objective optimization techniques are employed in this work.

To the best of our knowledge, we are the first to establish a bi-objective mathematical model

(i) to address import, export and transshipment tasks simultaneously,
(ii) to comprise both the direct- and indirect-transshipment modes,
(iii) to balance dissatisfaction of shipping companies and operational cost of container terminals.

These three features make the container terminal integrated planning even more complicated. To solve the problem efficiently, we devise bi-objective methods to obtain approximate Pareto front solutions. Numerical experiments have been conducted to demonstrate the efficiency of the proposed approaches.

The remainder of the paper is organized as follows. In Section 2, a brief literature review is given. Section 3 states the studied problem, and a mathematical formulation is proposed in Section 4. In Section 5, we devise a promising heuristic solution approach. Numerical experiments have been conducted in Section 6. We conclude the work and indicate further research directions in Section 7.

2. Literature review

There is a vast number of literature dedicated to the study of maritime logistics, as it is vital to the world trade and economy. The related literature mainly includes container assignment problems (e.g., Bell et al. (2011); Bell et al. (2013); Wang et al. (2015)), liner shipping problems (e.g., Liu et al. (2014); Meng et al. (2015)) and terminal operation planning problems (e.g., Lee et al. (2014); Tao and Lee (2015)). Our work falls in the group of terminal operation planning, where the integrated planning problems have been receiving more and more investigation recently. For details, interested readers

may refer to surveys such as Stahlbock and Voß (2008), Bierwirth and Meisel (2010), and Bierwirth and Meisel (2015). Recent trends of maritime logistics can be found in Lee and Meng (2015). Below we briefly review the integration studies of container port logistics, which can be roughly divided into two categories: operational-level integration and tactical-level integration. For container terminal managers, decisions made on the tactical level are regarded as the parameters or input for operational-level problems.

2.1. Operational-level integration

Chen et al. (2007) investigate a joint optimization of operations of quay cranes, yard cranes and yard vehicles, as these equipment operations are strongly interactive. They treat the integration problem of three kinds of equipments at container terminal as a three-stage hybrid flow shop scheduling problem with blocking and sequence-dependent setup times. Their model is based on an assumption that quay cranes assigned to a ship can not move to another ship's area. Tabu search heuristic is employed to obtain efficient solutions.

Cao et al. (2010) consider an integrated yard truck and yard crane scheduling problem. They consider loading operations for outbound containers, and assume that multiple yard cranes do not interfere with each other and the containers can be handled in any order. They view the problem as a two-stage flexible flowshop scheduling problem, and propose a mixed integer programming formulation. They apply a benders' cut-based method to efficiently solve the problem.

Choo et al. (2010) study the integrated optimization of quay crane scheduling and yard traffic-control management, with the purpose to prevent landside congestion and other operational inefficiency. Concerning that the yard congestion constraint is the only connecting constraint among all ships, lagrangian relaxation is applied on this constraint to decompose the problem into several vessel-independent subproblems. As for the single vessel related subproblem, a generalized set-covering formulation is developed, with the idea of covering each bays an amount of times by a minimum number of position-to-bay assignments of all cranes, and a branch-and-price algorithm is designed.

Chen et al. (2013) address the problem with interactions between crane handling and truck transportation in a maritime container terminal simultaneously. They formulate the problem as a constraint programming model, and devise a three-stage heuristic algorithm. At the first stage crane schedules are generated first by a heuristic. At the second stage the multiple-truck routing problem is solved based on the outcomes of the first stage. At the last stage a complete solution is constructed iteratively. Their computational results show that the proposed three-stage algorithm is effective.

Other operational-level integration studies may include Alessandri et al. (2007), Nabais et al. (2012), Xin et al. (2015), Li et al. (2015), etc.

2.2. Tactical-level integration

In the last decade, the integrated planning of tactical berth allocation and yard allocation is emerging.

2.2.1. Tactical berth allocation

Park and Kim (2003) present a pioneering work which addresses the integrated planning of berth allocation and quay crane assignment. They propose a two-phase solution procedure. The first phase determines berthing positions, berthing time and the number of quay cranes assigned to each vessel at each time-step. The second phase makes detailed decision for each quay crane based on outcomes of the first phase. This is a classic hierarchical approach. Their work starts the track of investigation on integrated planning of multiple operations.

Giallombardo et al. (2010) investigate the integrated planning of berth allocation and quay crane assignment. They propose the concept of quay crane assignment profile (QC-profile) to simply the QC assignment. They formulate a mathematical model, and present an optimization-based heuristic method. Later, Vacca et al. (2013) employ a branch-and-price algorithm to solve the set-partitioning reformulation of Giallombardo et al. (2010)'s model. QC-profile is qualified to handle the quay crane decreasing marginal productivity and does not involve crane idleness. Besides, this concept, capturing the major characteristics of quay crane schedules for vessels, facilitates quay crane assignment.

Other studies on tactical berth allocation include Meisel and Bierwirth (2013), Vacca et al. (2013), Imai et al. (2014), Turkogullari et al. (2014), Iris et al. (2015), etc.

2.2.2. Tactical yard allocation

Zhen et al. (2016) study a tactical-level yard management problem in container ports. For vessels visiting the port periodically, the tasks are to allocate yard spaces for vessels. The transportation cost of moving containers around the yard is minimized. With the consideration of yard traffic congestion, they propose a mixed integer program for multi-period yard template planning. To solve the problem in practice, they develop a local branching method and a Particle Swarm Optimization method. Numerical experiments show, compared with the classic First-Come-First-Serve strategy, their solution approaches are very efficient.

Zhen (2016) investigates yard truck congestions in a tactical level. A combination of probabilistic and physics-based models is proposed for measuring yard truck interruptions. Based on the linking travel times evaluated, a mixed integer program is proposed to minimize the expected travel time of moving containers around the yard. To solve the problem efficiently, a Squeaky Wheel Optimization is developed.

2.2.3. Integrated tactical berth and yard allocation

Zhen et al. (2011) is the first to study the joint planning of tactical berth allocation and tactical yard allocation, by observing that the yard transportation cost is jointly determined by vessel berthing positions and yard stowage locations. Adopting the QC-profile concept, they formulate a very huge mixed integer program for the joint problem, aiming to attain efficient use of resources and cost reduction. Due to their problem complexity, a phase-separation-based heuristic (i.e., hierarchical approach) is devised and shown very efficient.

Hendriks et al. (2013) present a simultaneous berth allocation and yard planning problem at tactical level. The problem is solved by an alternating berth and yard planning heuristic approach. A real size case study provided by PSA Antwerp shows that the proposed approach to simultaneously solve both problems might reduce the total straddle carrier travel distance considerably as compared with a representative allocation.

Lee and Jin (2013) study a joint optimization problem, consisting of determining preferred berthing positions and service time for cyclical visiting feeders, and allocating storage yard space to the transshipment flows between mother vessels and feeders. They consider these tactical decision problems simultaneously for a container transshipment terminal. The purpose is to relieve quayside congestion and reduce the housekeeping cost of container movements. The integration problem is formulated as a mixed integer programming model and solved by a memetic heuristic approach. Computational experiments have been conducted to show the effectiveness of the heuristic. Jin et al. (2015) study the same problem and present a set covering formulation. In their work, heuristic methods based on column generation are developed to obtain near-optimal solutions. Computational experiments demonstrate the efficiency.

Summing up, in the literature (i) only transshipment tasks have been addressed, or to say, import and export tasks have been ignored, (ii) direct-transshipment mode has not been formally considered, and (iii) the proposed single objective is the weighted sum of berth and yard cost (see Table 1). Therefore, to improve the literature studies and make the solution approach applicable in port daily operations, we consider (i) all import, export and transshipment tasks, (ii) both direct- and indirect-transshipment modes, and (iii) two objectives to balance the needs of shipping liners and container terminals.

3. Problem statement

In this section, we introduce the background of the considered integration problem, i.e., the integrated planning of TBAP (at quay side) and TYAP (at land side).

3.1. Tactical berth allocation problem

In the TBAP, the primal goal is to conform with the timetables published by shipping liners. In other words, to minimize the violation of all vessels' expected turnaround time windows. At quayside, each arriving vessel must be assigned to one specific berthing position along the continuous quay wharf, as well as a berthing time window over the planning horizon. Container unloading and loading operations are performed in parallel by quay cranes. A typical quay crane has a width of 25.8 m, and its usual performance is 30 containers per hour. Typically, up to six quay cranes can be allocated to a large vessel. Running on tracks parallel to the quay wharf, quay cranes move horizontally and can not cross over each other. Each vessel must be served in a continuous time duration which may comprises several work-shifts or time-steps, each spreading 4 h (c.f. Zhen et al. (2011)), and in each time-step a number of quay cranes must be determined to serve the vessel.

For each vessel, the terminal managers try their best to complete service of each vessel within its expected turnaround time intervals $[a^e, b^e]$, i.e., to accord to timetables published by shipping liners. If some vessel's berthing schedule violates its expected arrival or departure time, it must cause vessel acceleration in the voyage from its previous port or next port. Besides, the violation of the expected time window cannot exceed a certain degree. For example, if a vessel sequentially visits Ningbo Port and Shanghai Port (both in China), for Shanghai Port, the violation of the expected arrival time cannot exceed 8 h (i.e., two time-steps), since otherwise the vessel cannot be completed at Ningbo port. Similarly, the violation of expected departure time cannot exceed a certain level. Thus, a feasible time window $[a^M, b^M]$ must be guaranteed by the terminal managers. The expected time window is nested in the feasible time window, i.e., $a^M \leq a^e$ and $b^e \leq b^M$. For each vessel, the berthing time on the planning horizon and berthing position on the quay wharf are to be made, subject to available berth line (e.g., 2600 m at Shanghai Port). The vessel's handling or turnaround time is affected by the quay crane assignment, which is determined for that vessel by the terminal managers. Fig. 1 illustrates a real berth allocation plan at Shanghai Port. In this figure, the lateral axis (or *x*-axis) denotes the length of quay wharf, and the vertical axis indicates planning time horizon. The unit of *x*-axis is one meter, and the unit of *y*-axis is a time-step which covers 4 h. The entire quay wharf is nominally divided into seven berths (for ease of management at Shanghai Port), however, the berth allocation on the quay wharf at Shanghai Port is continuous, not discrete. For example, vessel SAJIR occupies the right part of Berth 4 and the left part of Berth 5. This kind of figures is also called *time-space graph*. In this figure, each rectangle illustrates a vessel's information. The upper-case letters on the top of each rectangle represents the name of a vessel. The numbers on the bottom of each rectangle indicate the berthing start and end positions for that vessel. The width of a rectangle denotes the physical length of that vessel. The height of a rectangle illustrates the handling time of that vessel.

For example, the expected time window for vessel BREVIK B is time-step duration [7, 11] (which contains five time-steps, i.e., 7, 8, 9, 10, and 11), and the feasible time window is time-step duration [5, 12]. Note that a time-step duration [*a*, *b*] contains $b - a + 1$ time-steps. In the depicted berth allocation plan (see Fig. 1), this vessel is served within [7, 10], thus its

Table 1
Comparison on studies of the joint planning of TBAP and TYAP .

| Literature	Berth and yard features	Container tasks and modes	Objectives	Decisions	Approaches
Zhen et al. (2011)	Continuous berth Cyclical vessel arrivals Yard subblocks Yard traffic congestion	Transshipment tasks (Indirect-transshipment mode)	1. Total cost (Berth and yard)	Berthing positions Berthing time windows QC profiles Yard subblocks	Decomposition
Hendriks et al. (2013)	Continuous berth Cyclical vessel arrivals Yard blocks	Transshipment tasks (Indirect-transshipment mode)	1. Yard cost	Berthing positions, Yard blocks	Decomposition
Lee and Jin (2013), Jin et al. (2015)	Discrete berth Cyclical vessel arrivals Yard blocks	Transshipment tasks (Indirect-transshipment mode)	1. Total cost (Berth and yard)	Berths Yard blocks	Memetic heuristic Column generation
This paper	Continuous berth Cyclical vessel arrivals Yard subblocks Yard traffic congestion	Transshipment tasks (Direct-transshipment and indirect-transshipment modes) Import and export tasks	1. Violation of vessel turnaround time windows 2. Yard vehicle transportation distance	Berthing positions Berthing time windows QC profiles Yard subblocks	NSGA-II
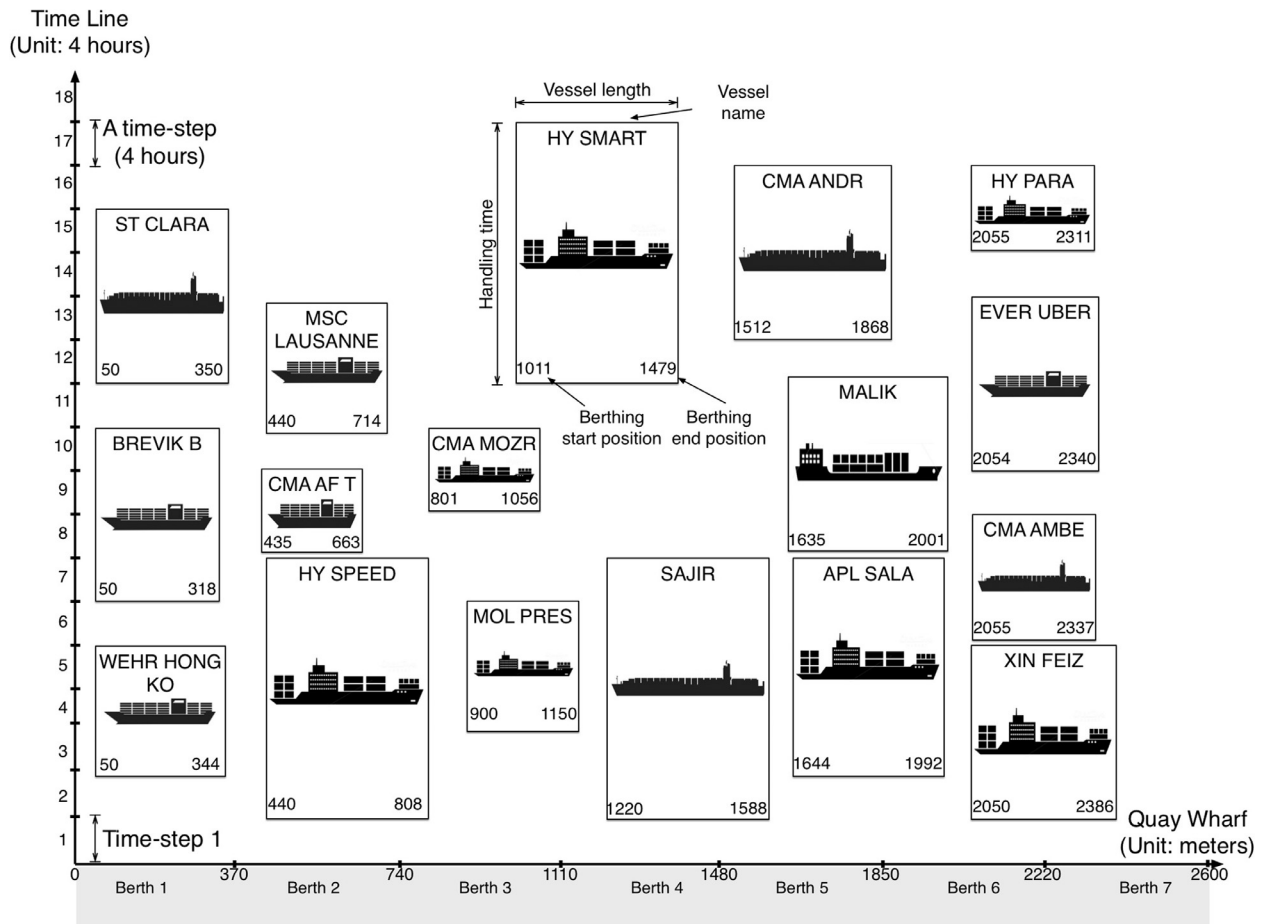
Fig. 1. Illustration of time-space graph for berth allocation (Source: Shanghai Port).

service quality is guaranteed. If vessel BREVIK B is served within [6, 9] (which is still feasible compared with its feasible interval [5, 12]), then the vessel's speedup from the previous port is derived and this plan causes a certain violation of the timetable published by the shipping liner. Similarly, if vessel BREVIK B is handled within [9, 12], then a certain violation is also incurred by the terminal. The consideration of the minimization of violation of vessel's expected time window is important for benefits of shipping companies, since "*low berth productivity costs shipping billions*" (Port Technology (2015)).

In the TBAP, the handling time of a vessel is controllable in a certain degree via quay crane assignment decisions. We introduce a concept called *QC-time-step*, which denotes the number of loading or unloading container operations a quay crane can complete within a time-step of 4 h. Typically a modern quay crane can handle 30 loading or unloading operations in an hour. Thus, a QC-time-step corresponds to 120 loading or unloading container operations. For example, the container tasks of vessel BREVIK B comprises: 237 import containers, 275 transit containers, and 456 export containers. Measured by QC-time-steps, the workload of vessel BREVIK B is eight QC-time-steps, which comprises four QC-time-steps of unloading operations (two for import tasks and two for transshipment tasks), and four QC-time-steps of loading operations. We borrow a concept of QC-profile, which has been widely adopted in the literature (c.f. Giallombardo et al. (2010); Zhen et al. (2011), Vacca et al. (2013)). A *QC-profile*, dedicated to a vessel, denotes the number of QCs assigned to the vessel in each time-step. Fig. 2 illustrates an example of the set of QC-profiles. In this figure, four QC-profiles are depicted in details. For example, the number of QCs can be assigned to vessel BREVIK B falls in the range [2, 3]. The first QC-profile takes four time-steps, and the second one consumes only three time-steps. In particular, in the second QC-profile, three quay cranes are deployed in the first handling time-step, used for unloading operations. In the next time-step, one quay crane are assigned to BREVIK B's neighbor vessel, and two quay cranes are left for remanding tasks: one for unloading and one for loading. In the last time-step, three quay cranes works parallel for loading operations. In each QC-profile, eight QC-time-steps are completed. On one side, the terminal operators prefer to assign QC-profile 1 or 3 to vessel BREVIK B, to spare one quay crane for other usage; on the other hand, if BREVIK B's expected departure time is close, a high concentration of quay cranes should be deployed to that vessel, and QC-profile 2 or 4 is a good option to meet the vessel's expected departure time. QC-profiles have been widely recognized as convenient ways to manage the dynamics of QC assignment for vessels.
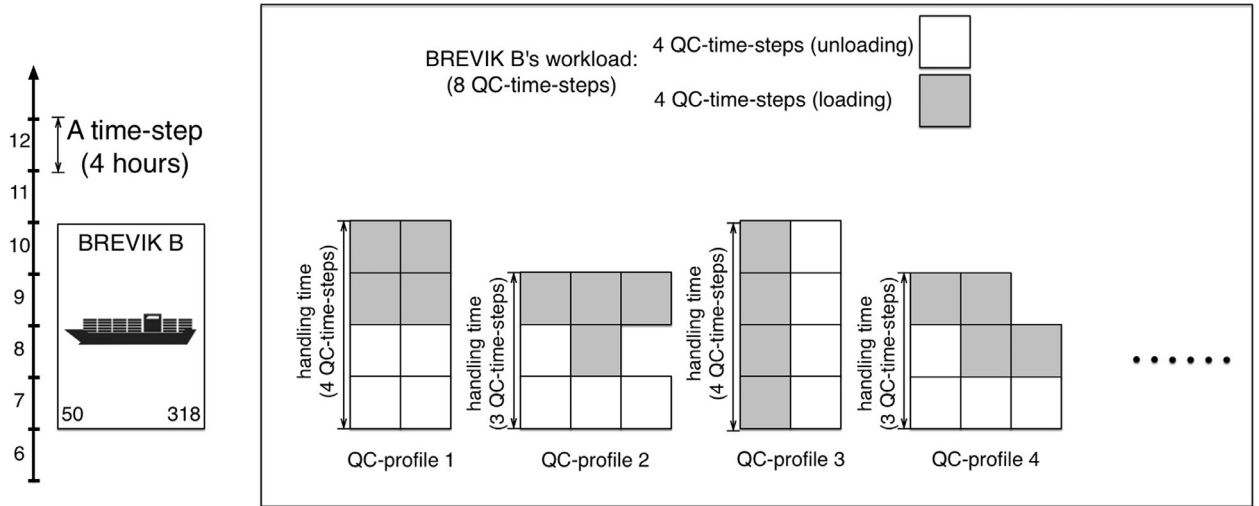
Fig. 2. An example of the set of QC-profiles for vessel BREVIK B.

When generating the possible QC-profiles for a vessel, the total number of required QC-time-steps must be satisfied. Besides, some practical rules in terminal daily operations must be obeyed. First, the minimum and maximum number of QCs can be assigned to the vessel simultaneously, where the minimum number of QCs is required by shipping companies in the negotiation with the terminal managers, and the maximum number is confined by the terminal managers considering crane safety margin and efficiency in parallel working. Second, in a time-step the number of QCs is fixed and the allowed variation of the number of QCs between two adjacent time-steps is one, which keeps the distribution of QCs along the quay wharf as regular as possible. As the allowed maximum number of QCs guarantee the crane productivity loss (incurred by QC interference) falls in a very small range, we ignore this operational-level consideration in our tactical-level planning problem. When a vessel arrives at a container port, normally the container unloading operations are performed first.

As shipping companies design shipping line service on a weekly basis, the cyclical vessel arrival issues introduce additional complexity into the TBAP (Moorthy and Teo (2006)). To handle cyclical issues, the original planning horizon H must be enlarged to further cover an additional planning time E which is larger than the maximum handling time for the vessels. For each berth segment (covering 50 m on the quay wharf) b, A usual way is to introduce two time-steps o_b and d_b, which are, respectively, defined as the first (or start) and last (or end) time-steps that berth segment b is occupied. To ensure the schedule of berth segment b is wrapped around within the planning time H, the number of time-steps in the range $[o_b, d_b]$ cannot be larger than H. In the context of cyclical arrivals of vessels, the QC capacity constraint requires that in each time-step $t \in \{E+1, E+2, \ldots, H\}$, the number of working QCs cannot exceed its available number in time-step t, and in each time-step $t \in \{1, 2, \ldots, E\}$ the total number of used QCs in both time-steps t and $t+H$ cannot exceed the number of available QCs in time-step t. The numbers of available QCs in different time-steps may differ a little, due to maintenance activities.

3.2. Tactical yard allocation problem

In the TYAP, the primary target is to minimize total vehicle transportation distance (the unit of which is 1 km). On tactical level, landside activities mainly include yard storage allocation. Together with the allocated berthing positions of vessels, the yard allocation decisions affect the vehicle transportation distance between berth and yard for containers. For example, the yard container transportation is carried out by internal trucks (which are not allowed to work outside container ports) at Shanghai Port.

Specifically, for transshipment tasks, the yard transportation distance is also affected by transshipment modes. There are two scenarios: (i) if direct-transshipment mode is applied, the distance is calculated between the berthing positions of two vessels, (ii) if indirect-transshipment mode is applied, the distance is measured between the berthing position and the yard storage location (like a couple of import and export tasks). Please see Fig. 3 for illustration.

Yard storage space is separated into yard blocks. Each yard storage block, consisting of five subblocks, is a rectangular region, comprising six to eight rows for storing containers in stacks. Please see Fig. 4 for illustration.

Under the *separation-based consignment strategy* utilized at Shanghai Port, the import, export and transit containers are stored in separated areas, i.e., import area, export area, and transshipment area. In each specific yard area, some dedicated subblocks are reserved for each vessel. In the TYAP, terminal managers need to assign subblocks to vessels. The assigned subblocks are dedicated for each vessel. For an arriving vessel j, (i) the import containers, which will be taken away by external trucks (which belong to logistics companies or customers outside the container port, in contrast to internal trucks
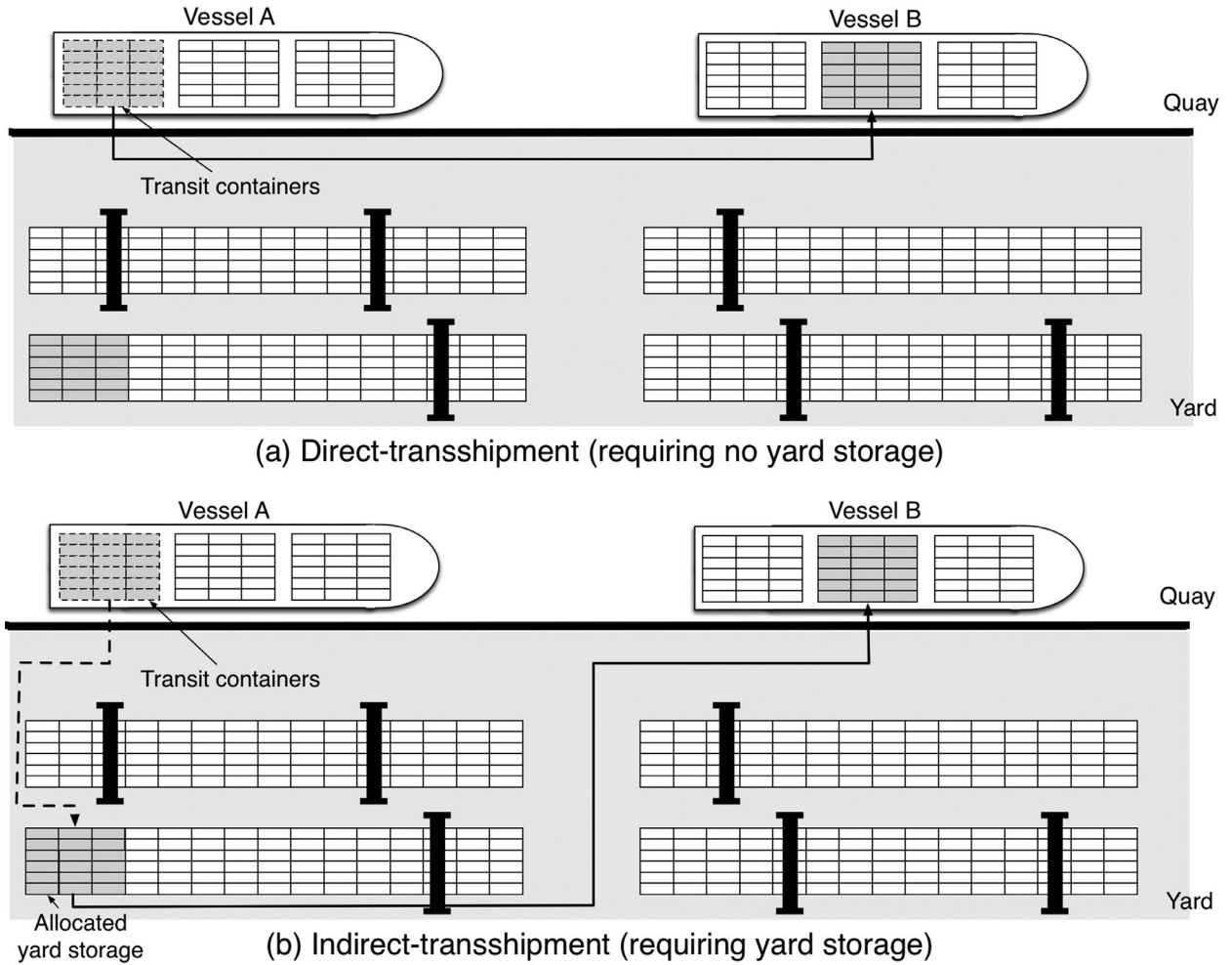
Fig. 3. Direct-transshipment vs. Indirect-transshipment.

which are hold by the port), are discharged from vessel j and stored in the vessel j's dedicated subblocks in the import area, (ii) the export containers, which are to be loaded on vessel j, are stored in vessel j's dedicated subblocks in the export area, (iii) the transit containers to be loaded on vessel j, should be discharged from some other vessel i and stored in vessel j's dedicated subblocks in the transshipment area. This strategy could reduce the number of container reshuffles in the yard.

In the yard management of container ports, the loading process is crucial for berth productivity. To guarantee the efficiency of quay cranes at quayside, the land side activities must keep in the same speed. However, in the ship loading process, the yard cranes working in a certain yard block must provide the required container sequence for quay cranes. (The container sequence is generated from the stowage plan of the vessel, which is out of the scope of this paper.) The expected departure time of each vessel is given, and the service of each ship should be completed in a short time. In contrast, in the unloading process, the requirement of container positions in subblocks is not strict and the terminal managers has some flexibility in this process. The major concern in the tactical yard management is the traffic congestion regulation, because when a subblock is in loading process, the yard crane and trucks are under high workload, and thus the traffic near the subblock is very heavy. To ensure a smooth traffic flow, (i) there should not be two or more neighbor subblocks which shares a same truck path (e.g., subblock pairs (1F, 2F) and (1F, 1G) in Fig. 4) that have loading activities at the same time-step, and (ii) at most one among the five subblocks in each block is reserved for each vessel (c.f. Zhen et al. (2011)). In the TYAP, the terminal managers try their best to minimize the route length of all container transportation.

4. Mathematical formulation

As the considered problem contains two criteria, in this section, we propose a bi-objective integer program for the container terminal integration planning (CTIP). The two objectives are described as OBJ_1 and OBJ_2 in the following, which are to be minimized simultaneously. That is to say, our task is to find the true Pareto front or an approximate Pareto front for
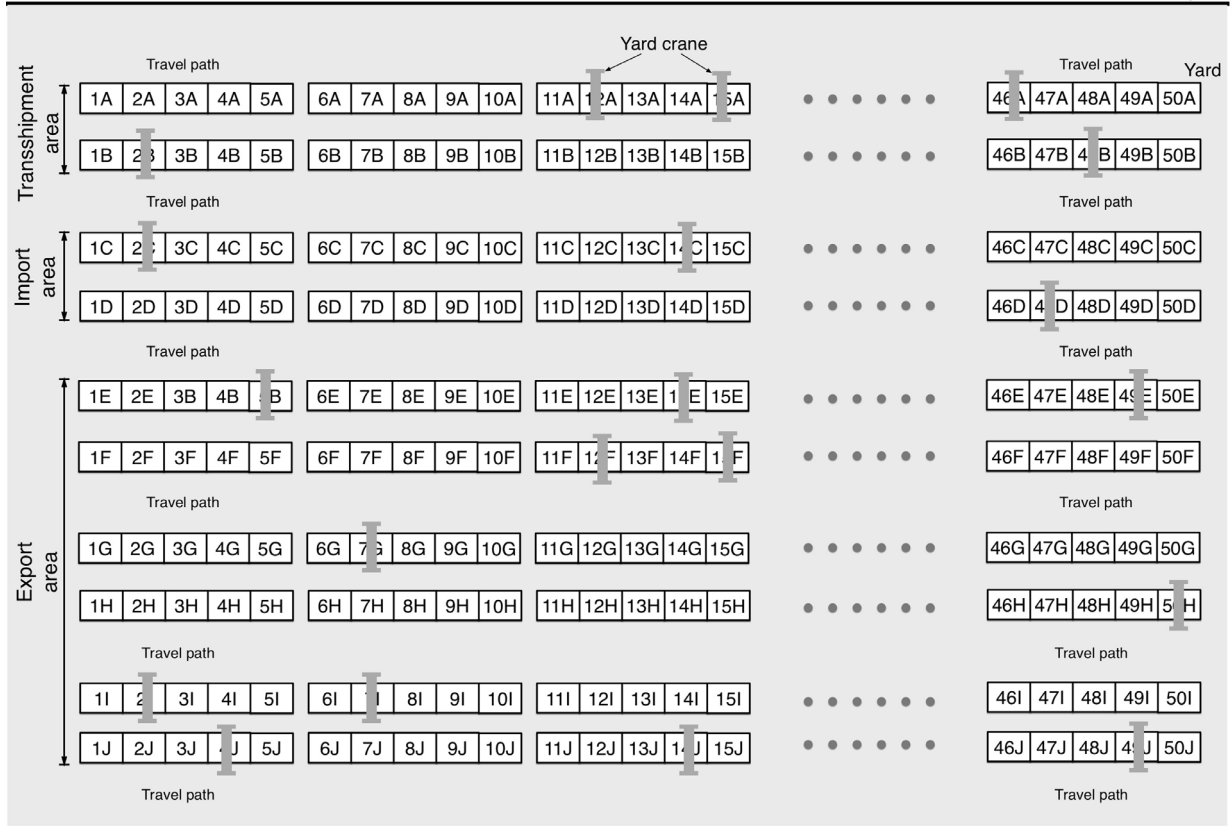
Fig. 4. Yard layout at Shanghai Port.

the CTIP, which may provide a trade-off scheme between the two objectives. A classic exact approach, called ϵ-*constraint method*, is widely used to obtain the true Pareto front for bi-objective optimization problems. The main idea of this method is to transform one objective into a constraint added to the formulation with a gradual reduction of value ϵ (where the reduction step can be set as the minimum unit of that objective value), and then to solve a series of single objective (i.e., the other objective) optimization problems to obtain a set of non-inferior solutions. If we plan to apply ϵ-constraint method to find the true Pareto front, we need to formulate a bi-objective integer program first, to describe the CTIP in mathematical programming language.

In Section 4.1, we give the definitions of problem parameters and decision variables, where some notations are borrowed from Zhen et al. (2011). In Section 4.2, we present a bi-objective integer program for the CTIP.

4.1. Notation

Indices:

i, j: indices of vessels, $i \neq j$;
k: index of subblocks;
t: index of abstract time-steps;
m: index of relative time-steps;
p: index of QC-profiles;
b: index of berth segments;
n: index of adjacent subblock pairs that are neighbors in the transshipment and export areas, e.g., (1F, 2F) and (1F, 1G) in Fig. 4;
g: index of blocks in the transshipment and export areas, each of which consists of five subblocks, e.g., (1A, 2A, 3A, 4A, 5A) in Fig. 4.

Problem parameters:

V: set of vessels;

B: set of berth segments;

K^I: set of available subblocks in yard import area;

K^E: set of available subblocks in yard export area;

K^T: set of available subblocks in yard transshipment area;

s: length of a berth segment (e.g., $s = 50$ m at Shanghai Port);

L: length of the quay wharf (e.g., $L = 2600$ m at Shanghai Port);

H: number of time-steps in a planning horizon;

P_i: set of QC-profiles for vessel i, where $i \in V$;

h_{ip}: handling time of vessel i using QC-profile p, where $i \in V$ and $p \in P_i$;

f: the maximum number of time-steps allowed for the difference of start time-steps of two vessels utilizing direct-transshipment mode, e.g., $f = 6$ at Shanghai Port;

E: maximum handling time among all vessels, i.e., $E = \max_{i \in V, p \in P_i}\{h_{ip}\}$;

T: set of time-steps under consideration, $E = \{1, 2, \ldots, H + E\}$;

$[a_i^M, b_i^M]$: feasible turnaround time window or interval for vessel i, where $i \in V$ and $a_i^M, b_i^M \in T$;

$[a_i^e, b_i^e]$: expected turnaround time window or interval for vessel i, where $i \in V$ and $a_i^e, b_i^e \in T$;

N: set of all the adjacent subblock pairs in the transshipment and export areas;

G: set of all the blocks in yard transshipment and export areas;

u_i: physical length of vessel i (For ease of handling, we also calculate a safety distance between two adjacent berthed vessels in this notation u. The safety distance about 10 m must be kept in berth allocation, to avoid collision of vessels. The safety distance is relatively very small, compared with vessel lengths, especially for mega ships);

Q_t: number of available QCs in time-step t, where $t \in \{1, 2, \ldots, H\}$;

l_{ipm}: $l_{ipm} = 1$, means vessel i performs loading activities in the m-th time-step using QC-profile p; $l_{ipm} = 0$, otherwise, where $i \in V$, $p \in P_i$ and $m \in \{1, 2, \ldots, h_{ip}\}$;

q_{ipm}: number of utilized QCs in the m-th time-step if vessel i is served with QC-profile p, where $i \in V$, $p \in P_i$ and $m \in \{1, 2, \ldots, h_{ip}\}$;

r_i^I: number of subblocks in the import area should be reserved for vessel i, where $i \in V$;

r_i^E: number of subblocks in the export area should be reserved for vessel i, where $i \in V$;

r_i^T: number of subblocks in the transshipment area should be reserved for vessel i, where the transit containers stored in these subblocks are to be loaded on vessel i, where $i \in V$;

R_i^I: $R_i^I = 1$ if $r_i^I > 0$; 0 otherwise, where $i \in V$;

R_i^E: $R_i^E = 1$ if $r_i^E > 0$; 0 otherwise, where $i \in V$;

R_i^T: $R_i^T = 1$ if $r_i^T > 0$; 0 otherwise, where $i \in V$;

c_i^I: number of import containers of vessel i, where $i \in V$;

c_i^E: number of export containers of vessel i, where $i \in V$;

c_{ij}^T: number of transit containers to be transshipped from vessel i to vessel j, where $i, j \in V$ and $i \neq j$;

A_{ij}^T: $A_{ij}^T = 1$ if there exists a transshipment activity from vessel i to vessel j; 0 otherwise, where $i, j \in V$ and $i \neq j$;

D_{bk}^U: length of unloading route (including a full-load trip and an empty trip) from berth segment b to subblock k, where $b \in B$ and $k \in K^I \bigcup K^T$;

D_{kb}^L: length of loading route (including a full-load trip and an empty trip) from subblock k to berth segment b, where $b \in B$ and $k \in K^E \bigcup K^T$;

$D_{b_1 b_2}^D$: length of direct-transshipment route (including a full-load trip and an empty trip) from berth segment b_1 to berth segment b_2, where $b_1, b_2 \in B$;

w_i^a: the weight (per time-step) assigned to the earliness for vessel i in the objective, where $i \in V$;

w_i^b: the weight (per time-step) assigned to the tardiness for vessel i in the objective, where $i \in V$;

M: a sufficiently large integer.

Decision variables:

β_i: berthing position for vessel i, according to the vessel's middle point (along x-axis in Fig. 1), where $i \in V$ and $\beta_i \in [0, L]$. As the unit of quay wharf is one meter, this variable is integral;

ω_{ib}: equal to 1 if the middle point of vessel i locates in berth segment b; 0 otherwise, where $i \in V$ and $b \in B$;

φ_{ik}: equal to 1 if subblock k is reserved for vessel i; 0 otherwise, where $i \in V$ and $k \in K^I \bigcup K^E \bigcup K^T$;

ξ_{ikt}: equal to 1 if subblock k is reserved for vessel i and vessel i is in loading process or activity in time-step t, where $i \in V$, $k \in K^E \bigcup K^T$ and $t \in T$;

γ_{ip}: equal to 1 if vessel i is served with QC-profile p; 0 otherwise, where $i \in V$ and $p \in P_i$;

η_{ipt}: equal to 1 if vessel i is served with QC-profile p and starts handling in time-step t, where $i \in V$, $p \in P_i$ and $t \in T$;

ε_i: the start time-step of vessel i, where $i \in V$;

π_{it}: equal to 1 if vessel i's start time-step ε_i is t; 0 otherwise, where $i \in V$ and $t \in T$;

σ_i: the end time-step of vessel i, where $i \in V$;

δ_{ij}^x: equal to 1 if vessel i is berthed on the leftside of vessel j along the quay wharf (along x-axis in Fig. 1); 0 otherwise, where $i, j \in V$ and $i \neq j$;

δ_{ij}^y: equal to 1 if vessel i is completed before vessel j starts service (along y-axis in Fig. 1); 0 otherwise, where $i, j \in V$ and $i \neq j$;

θ_{it}: equal to 1 if vessel is in loading activity in time-step t, where $i \in V$ and $t \in T$;

ζ_{ib}: equal to 1 if vessel i occupies berth segment b, where $i \in V$ and $b \in B$;

o_b: the start time-step of berth segment b, where $b \in B$;

d_b: the end time-step of berth segment b, where $b \in B$;

ρ_t: the number of used QCs in time-step t, where $t \in T$;

υ_{ij}^D: equal to 1 if direct-transshipment mode is applied for the transshipment task or activity from vessels i to j, 0 otherwise, where $i, j \in V$ and $i \neq j$;

υ_{ij}^I equal to 1 if indirect-transshipment mode is applied for the transshipment task or activity from vessels i to j, 0 otherwise, where $i, j \in V$ and $i \neq j$.

λ_i^I: the average distance of unloading route between the berthing position of vessel i and all the subblocks in the yard import area reserved for vessel i, where $i \in V$;

λ_i^E:e the average distance of loading route between the berthing position of vessel i and all the subblocks in the yard export area reserved for vessel i, where $i \in V$;

λ_{ij}^{TU}: the average distance of unloading route between the berthing position of vessel i and all the subblocks in the transshipment area reserved for vessel j, where $i, j \in V$ and $i \neq j$;

λ_j^{TL}: the average distance of loading route between all the subblocks in the yard transshipment area reserved for vessel j and the berthing position of vessel j, where $j \in V$;

λ_{ij}^T: the average distance of route between vessels i and j. That is, (i) if direct-transshipment mode is applied (i.e., $\upsilon^D = 1$), then λ_{ij}^T is the distance between berthing positions occupied by vessels i and j, and (ii) if indirect-transshipment mode is applied (i.e., $\upsilon^I = 1$), then yard storage is involved and λ_{ij}^T is equal to $\lambda_{ij}^{TU} + \lambda_j^{TL}$, where $i, j \in V$ and $i \neq j$;

τ_i^a: earliness of vessel i, i.e., $\tau_i^a = \max\{a_i^e - \varepsilon_i, 0\}$, where $i \in V$;

τ_i^b: tardiness of vessel i, i.e., $\tau_i^b = \max\{\sigma_i - b_i^e, 0\}$, where $i \in V$;

ψ_{jkib}: equal to 1 if $\varphi_{jk} = 1$ and $\omega_{ib} = 1$; 0 otherwise, where $i, j \in V$, $k \in K^T$, $b \in B$ and $i \neq j$;

κ_{jkb}: equal to 1 if $\varphi_{jk} = 1$ and $\omega_{jb} = 1$; 0 otherwise, where $j \in V$, $k \in K^I \bigcup K^E \bigcup K^T$ and $b \in B$;

$\varpi_{ib_1 jb_2}$: equal to 1 if $\omega_{ib_1} = 1$ and $\omega_{jb_2} = 1$; 0 otherwise, where $i, j \in V$, $b_1, b_2 \in B$, $i \neq j$ and $b_1 \neq b_2$;

ϑ_{ib}^L: equal to 1 if berth segment b is on the left side of vessel i (along x-axis in Fig. 1), i.e., $\beta_i \geq s \cdot b$; 0 otherwise, where $i \in V$ and $b \in B$;

ϑ_{ib}^R: equal to 1 if berth segment b is on the right side of vessel i (along x-axis in Fig. 1), i.e., $\beta_i + u_i \leq s \cdot (b - 1) - 1$; 0 otherwise, where $i \in V$ and $b \in B$;

ι_{ij}^L: equal to 1 if $\varepsilon_j \geq \varepsilon_i$; 0 otherwise, where $i, j \in V$ and $i \neq j$;

ι_{ij}^R: equal to 1 if $\varepsilon_j \leq \varepsilon_i + f - 1$; 0 otherwise, where $i, j \in V$ and $i \neq j$;

4.2. The formulation of the CTIP

The two objectives are described as OBJ_1 and OBJ_2 below. The first objective is to minimize the yard transportation distance, and the second one is to minimize the violation of vessels' expected turnaround time windows. In Section 4.2.1, we describe the two objective functions. In Sections 4.2.2–4.2.9, we present the main constraints should be subjected to by the formulation of CTIP. Ranges of decision variables are specified in Section 4.2.10.

4.2.1. Two objective functions

In the minimization item of OBJ_1, as there are import, export and transit containers, we calculate all three kinds of transportation distances for all vessels accordingly, and then sum them up.

$$OBJ_1 = \min \left(\sum_{i \in V} c_i^I \lambda_i^I + \sum_{i \in V} c_i^E \lambda_i^E + \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij}^T \lambda_{ij}^T \right). \tag{1}$$

In the minimization item of OBJ_2, we calculate the weighted earliness and weighted tardiness for each vessel and then sum them up for all vessels.

$$OBJ_2 = \min \sum_{i \in V} (w_i^a \tau_i^a + w_i^b \tau_i^b). \tag{2}$$

These two objectives are to be minimized simultaneously, and the desired output is a solution set (i.e., Pareto front).

4.2.2. Yard storage contraints

In the yard storage allocation, some constraints should be satisfied and we present them below. Each yard subblock k should be reserved for at most one vessel.

$$\sum_{i \in V} \varphi_{ik} \leq 1, \quad k \in K^I \bigcup K^E \bigcup K^T. \tag{3}$$

The number of reserved subblocks in yard import (resp. export) area for vessel i should satisfy the vessel's required number r_i^I (resp. r_i^E).

$$\sum_{k \in K^I} \varphi_{ik} = r_i^I, \quad i \in V. \tag{4}$$

$$\sum_{k \in K^E} \varphi_{ik} = r_i^E, \quad i \in V. \tag{5}$$

If a transshipment task exists, or to say, there are transit containers (i.e., $A_{ij}^T = 1$), then this task must be performed in either direct-transshipment mode or indirect-transshipment mode.

$$v_{ij}^D + v_{ij}^I = A_{ij}^T, \quad i, j \in V, i \neq j. \tag{6}$$

If indirect-transshipment mode is applied to vessel j, then a number of subblocks in yard transshipment area should be reserved for that vessel. The transshipment activity from (or to) one vessel happens at most once in practice, which means we have $\sum_{i \in V, i \neq j} A_{ij}^T \leq 1$ and thus $\sum_{i \in V, i \neq j} v_{ij}^I \leq 1$. The following two constraints guarantee that if indirect-transshipment mode is applied (i.e., $\sum_{i \in V, i \neq j} v_{ij}^I = 1$), then some subblocks in transshipment area should be reserved for the outgoing vessel (i.e., $\sum_{k \in K^T} \varphi_{jk} = r_j^T$, another group of subblock constraints).

$$\sum_{k \in K^T} \varphi_{jk} \geq r_j^T - \left(1 - \sum_{i \in V, i \neq j} v_{ij}^I \right) M, \quad j \in V. \tag{7}$$

$$\sum_{k \in K^T} \varphi_{jk} \leq r_j^T + \left(1 - \sum_{i \in V, i \neq j} v_{ij}^I \right) M, \quad j \in V. \tag{8}$$

4.2.3. Vessel handling constraints

During the service of vessels, some constraints should be satisfied and we present them below. Each vessel must be served by exact one QC-profile.

$$\sum_{p \in P_i} \gamma_{ip} = 1, \quad i \in V. \tag{9}$$

Each vessel starts handling in exact one time-step.

$$\sum_{t \in T} \pi_{it} = 1, \quad i \in V. \tag{10}$$

The following constraint links the start time-step ε and 0–1 variable π for each vessel i.

$$\varepsilon_i = \sum_{t \in T} t \cdot \pi_{it}, \quad i \in V. \tag{11}$$

The following constraint links the start time-step and the end time-step of one vessel, whose service cannot be interrupted. In this constraint, the handling time of a vessel is expressed as $\sum_{p \in P_i} \gamma_{ip} \cdot h_{ip}$, i.e., if QC-profile p is selected for vessel (i.e., $\gamma_{ip} = 1$) then vessel i's handling time is h_{ip}. For example, if a vessel starts at time-step 3 (which occupies the time segment from time point 3 to 4), and its handling time is 3 time-steps (which occupies the time segment from time point 3 to 6), then its end time-step should be $3 + 3 - 1 = 5$ (which occupies the time segment from time point 5 to 6). Note that here a time-step is a time interval or duration, not a specific time point in the planning horizon.

$$\varepsilon_i + \sum_{p \in P_i} \gamma_{ip} \cdot h_{ip} - 1 = \sigma_i, \quad i \in V. \tag{12}$$

The feasible time window of each vessel must be satisfied.

$$\sigma_i \leq b_i^M, \quad i \in V. \tag{13}$$

$$\varepsilon_i \geq a_i^M, \quad i \in V. \tag{14}$$

The definition of earliness τ^a and tardiness τ^b are described below.

$$\tau_i^a \geq a_i^e - \varepsilon_i, \quad i \in V. \tag{15}$$

$$\tau_i^b \geq \sigma_i - b_i^e, \quad i \in V. \tag{16}$$

4.2.4. Berth allocation constraints

The middle point of each vessel is located within exact one berth segment.

$$\sum_{b \in B} \omega_{ib} = 1, \quad i \in V. \tag{17}$$

The berthing position of vessel i's middle point β_i, should be located within the range of berth segment b, if $\omega_{ib} = 1$. A berth segment b covers the range $[s \cdot (b-1), s \cdot b - 1]$ on the quay wharf.

$$s \sum_{b \in B} (b-1) \cdot \omega_{ib} \le \beta_i, \quad i \in V. \tag{18}$$

$$\beta_i \le s \sum_{b \in B} b \cdot \omega_{ib} - 1, \quad i \in V. \tag{19}$$

For any two vessels i and j, if $\delta_{ij}^x = 1$, then the rightmost berthing position (i.e., $\beta_i + u_i/2$) of vessel i is not larger than the leftmost berthing position of vessel j (i.e., $\beta_j - u_j/2$).

$$\beta_i + u_i/2 \le \beta_j - u_j/2 + (1 - \delta_{ij}^x)M, \quad i, j \in V, i \ne j. \tag{20}$$

For any two vessels i and j, if $\delta_{ij}^y = 1$, then the start time-step of vessel j (i.e., ε_j) is not smaller than the end time-step of vessel i (i.e., σ_i).

$$\sigma_i \le \varepsilon_j + (1 - \delta_{ij}^y)M, \quad i, j \in V, i \ne j. \tag{21}$$

For any two different vessels i and j, each representing a rectangle in the 2-dimension graph (see Fig. 1), they are not allowed to overlap in the time-space graph.

$$\delta_{ij}^x + \delta_{ji}^x + \delta_{ij}^y + \delta_{ji}^y \ge 1, \quad i, j \in V, i \ne j. \tag{22}$$

The rightmost berthing position (i.e., $\beta_i + u_i/2$) cannot exceed the maximum berthing position.

$$\beta_i + u_i/2 \le L, \quad i \in V. \tag{23}$$

4.2.5. QC assignment constraints

The capacity of available QCs at each time-step can not be violated. To express the consumed QC resource in each time-step, we need an assistant variable η. The following constraint links η to γ and π. The meaning of this constraint is that if vessel i is assigned with QC-profile p (i.e., $\gamma_{ip} = 1$) and vessel i starts service at time-step t (i.e., $\pi_{it} = 1$) then $\eta_{ipt} = 1$, or to say, we linearize the expression $\eta_{ipt} = \gamma_{ip} \cdot \pi_{it}$.

$$\eta_{ipt} \ge \gamma_{ip} + \pi_{it} - 1, \quad i \in V, p \in P_i, t \in T. \tag{24}$$

The number of QCs utilized by all vessels at a time-step (i.e., ρ_t) can be expressed as follows via η.

$$\rho_t = \sum_{i \in V} \sum_{p \in P_i} \sum_{h=\max\{1, t-h_{ip}+1\}}^{t} q_{i,p,t-h+1} \cdot \eta_{iph}, \quad t \in T. \tag{25}$$

In the extended planning horizon, the constraint on the maximum number of working QCs can be described as follows via ρ.

$$\rho_t + \rho_{t+H} \le Q_t, \quad t \in \{1, \ldots, E\}. \tag{26}$$

$$\rho_t \le Q_t, \quad t \in \{E+1, \ldots, H\}. \tag{27}$$

4.2.6. Yard traffic constraints

To regulate the yard traffic, some constraints should be satisfied by the relevant subblocks which are in loading activity (i.e., loading containers to vessels) in the same time-step. Via assistant variable θ_{it}, we can depict when vessel i is in loading activity (or performing loading operations).

The following constraint links variable θ to variable η. If vessel i is assigned with QC-profile p and starts service in abstract time-step t (i.e., $\eta_{ipt} = 1$), and the m-th relative time-step of QC-profile p is loading (i.e., $l_{imp} = 1$), then vessel i must be in loading activity in abstract time-step $t + m - 1$ (i.e., $\theta_{i,t+m-1} = 1$). For example, if abstract time-step $t = 2$ and relative time-step $m = 1$, then abstract time-step $t + m - 1 = 2$.

$$\theta_{i,t+m-1} \ge \eta_{ipt} \cdot l_{ipm}, \quad i \in V, p \in P_i, m \in \{1, \ldots, h_{ip}\}, t \in T. \tag{28}$$

The following constraint links variable ξ to φ and θ, in yard export area. The meaning of this constraint is that if an export area subblock k is assigned to vessel i (i.e., $\varphi_{ik} = 1$) and this vessel is loading in time-step t (i.e., $\theta_{it} = 1$) then this subblock k is in loading activity in time-step t (i.e., $\xi_{ikt} = 1$), or to say, we linearize the expression $\xi_{ikt} = \varphi_{ik} \cdot \theta_{it}$.

$$\xi_{ikt} \ge \varphi_{ik} + \theta_{it} - 1, \quad i \in V, k \in K^E, t \in T. \tag{29}$$

In yard transshipment area, if vessel j is served in indirect-transshipment mode (i.e., $\sum_{i\in V, i\neq j} v_{ij}^I = 1$), then the following constraint links variable ξ to φ, θ and v. The meaning of this constraint is similar to the above one.

$$\xi_{jkt} \geq \varphi_{jk} + \theta_{jt} + \sum_{i\in V, i\neq j} v_{ij}^I - 2, \quad j \in V, k \in K^T, t \in T. \tag{30}$$

In yard export area and transshipment area, in each time-step, at most one subblock in each adjacent subblock pairs is allowed to be in loading activity (i.e., loading containers to vessels).

$$\sum_{k\in n}\sum_{i\in V} \xi_{ikt} + \sum_{k\in n}\sum_{i\in V} \xi_{i,k,t+H} \leq 1, \quad t \in \{1, \ldots, E\}, n \in N. \tag{31}$$

$$\sum_{k\in n}\sum_{i\in V} \xi_{ikt} \leq 1, \quad t \in \{E+1, \ldots, H\}, n \in N. \tag{32}$$

In yard export area and transshipment area, in each time-step, at most one subblock in each block is allowed to be in loading activity (i.e., loading containers to vessels).

$$\sum_{k\in g}\sum_{i\in V} \xi_{ikt} + \sum_{k\in g}\sum_{i\in V} \xi_{i,k,t+H} \leq 1, \quad t \in \{1, \ldots, E\}, g \in G. \tag{33}$$

$$\sum_{k\in g}\sum_{i\in V} \xi_{ikt} \leq 1, \quad t \in \{E+1, \ldots, H\}, g \in G. \tag{34}$$

4.2.7. Distance calculation constraints

The average distance of route of import containers is calculated as follows. If $R_j^I = 0$ (which indicates that $r_j^I = 0$), then $\lambda_j^I = 0$. If $R_j^I = 1$, then $\lambda_j^I \geq \frac{\sum_{k\in K^I}\sum_{b\in B} \kappa_{jkb} \cdot D_{bk}^U}{r_j^I}$. The purpose of introducing parameter R_j^I is to rule out the case when $r_j^I = 0$ and the expression $\frac{\sum_{k\in K^I}\sum_{b\in B} \kappa_{jkb} \cdot D_{bk}^U}{r_j^I}$ has no meanings.

$$\lambda_j^I \leq R_j^I \cdot M, \quad j \in V. \tag{35}$$

$$\lambda_j^I \geq \frac{\sum_{k\in K^I}\sum_{b\in B} \kappa_{jkb} \cdot D_{bk}^U}{R_j^I - 1 + r_j^I}, \quad j \in V. \tag{36}$$

The average distance of route for export containers is calculated as follows.

$$\lambda_j^E \leq R_j^E \cdot M, \quad j \in V. \tag{37}$$

$$\lambda_j^E \geq \frac{\sum_{k\in K^E}\sum_{b\in B} \kappa_{jkb} \cdot D_{kb}^L}{R_j^E - 1 + r_j^E}, \quad j \in V. \tag{38}$$

The average distance of unloading route of transit containers in indirect-transshipment mode is computed as follows.

$$\lambda_{ij}^{TU} \leq R_j^T \cdot M, \quad i, j \in V, i \neq j. \tag{39}$$

$$\lambda_{ij}^{TU} \geq \frac{\sum_{k\in K^T}\sum_{b\in B} \psi_{jkib} \cdot D_{bk}^U}{R_j^T - 1 + r_j^T}, \quad i, j \in V, i \neq j. \tag{40}$$

Relation between ψ, φ and ω is established as follows. The meaning of this constraint is that if vessel j is allocated with subblock k (i.e., $\varphi_{jk} = 1$) and the middle point of vessel i is assigned within berth segment b (i.e., $\omega_{ib} = 1$) then $\psi_{jkib} = 1$, or to say, we linearize the expression $\psi_{jkib} = \varphi_{jk} \cdot \omega_{ib}$.

$$\psi_{jkib} \geq \varphi_{jk} + \omega_{ib} - 1, \quad i, j \in V, i \neq j, k \in K^T, b \in B. \tag{41}$$

The average distance of loading route of transit containers in indirect-transshipment mode is computed as follows.

$$\lambda_j^{TL} \leq R_j^T \cdot M, \quad j \in V. \tag{42}$$

$$\lambda_j^{TL} \geq \frac{\sum_{k\in K^T}\sum_{b\in B} \kappa_{jkb} \cdot D_{kb}^L}{R_j^T - 1 + r_j^T}, \quad j \in V. \tag{43}$$

Relation between κ, φ and ω is established as follows. The meaning of this constraint is that if vessel j is allocated with subblock k (i.e., $\varphi_{jk} = 1$) and the middle point of vessel j falls in the range of berth segment b (i.e., $\omega_{jb} = 1$) then $\kappa_{jkb} = 1$, or to say, we linearize the expression $\kappa_{jkb} = \varphi_{jk} \cdot \omega_{jb}$.

$$\kappa_{jkb} \geq \varphi_{jk} + \omega_{jb} - 1, \quad j \in V, k \in K^I \bigcup K^E \bigcup K^T, b \in B. \tag{44}$$

If indirect-transshipment mode is applied for transit containers from vessel i to vessel j (i.e., $\nu_{ij}^I = 1$), then the average distance of route of the transit containers is expressed as follows. This is because in indirect-transshipment mode, a transshipment task must be first unloaded from the incoming vessel to the reserved subblocks and then loaded to the outgoing vessel. That is, the distance of entire route is equal to the sum of an unloading distance and a loading distance (i.e., $\lambda_{ij}^T \geq \lambda_{ij}^{TU} + \lambda_j^{TL}$).

$$\lambda_{ij}^T \geq \lambda_{ij}^{TU} + \lambda_j^{TL} - (1 - \nu_{ij}^I)M, \quad i, j \in V, i \neq j. \tag{45}$$

If direct-transshipment mode is applied for transit containers from vessel i to vessel j (i.e., $\nu_{ij}^D = 1$), then the average distance of route of transit containers is expressed as follows.

$$\lambda_{ij}^T \geq \varpi_{ib_1 jb_2} \cdot D_{b_1 b_2}^D - (1 - \nu_{ij}^D)M, \quad i, j \in V, i \neq j, b_1, b_2 \in B, b_1 \neq b_2. \tag{46}$$

The relation between ϖ and ω is described as follows. The meaning of this is that if vessel i occupies berth segment b_1 (i.e., $\omega_{ib_1} = 1$) and vessel j occupies berth segment b_2 (i.e., $\omega_{jb_2} = 1$) then $\varpi_{ib_1 jb_2} = 1$, or to say, we linearize the expression $\varpi_{ib_1 jb_2} = \omega_{ib_1} \cdot \omega_{jb_2}$

$$\varpi_{ib_1 jb_2} \geq \omega_{ib_1} + \omega_{jb_2} - 1, \quad i, j \in V, i \neq j, b_1, b_2 \in B, b_1 \neq b_2. \tag{47}$$

4.2.8. Cyclical arrival constraints

To address cyclical vessel arrival (or periodicity) issues, the original planning horizon has been enlarged from H to $H + E$. However, for each berth segment, the difference of its latest busy time-step (i.e., the end time-step) and its earliest busy time-step (i.e., the start time-step) should not violate the original planning horizon length. The assistant variable ζ_{ib} helps to indicate whether vessel i occupies berth segment b or not. Via ζ_{ib}, we can express the start and end time-steps for each berth segment.

The following three constraints show the relation between ζ_{ib} and β_i. Recall that a berth segment b covers the range $[s \cdot (b-1), s \cdot b - 1]$ on the quay wharf. Among the following five constraints, the first two constraints express the relation that if and only if $\beta_i \geq s \cdot b$ then $\vartheta_{ib}^L = 1$ (i.e., the vessel moors on the right side of this segment), the next two constraints impose that if and only if $\beta_i + u_i \leq s \cdot (b-1) - 1$ then $\vartheta_{ib}^R = 1$ (i.e., the vessel moors on the left side of this segment), and the last one guarantees that if both $\vartheta_{ib}^L = 0$ and $\vartheta_{ib}^R = 0$, then $\zeta_{ib} = 1$. There exists the relation that $\vartheta_{ib}^L + \vartheta_{ib}^R \leq 1$ by their definitions.

$$\beta_i \leq s \cdot b - 1 + \vartheta_{ib}^L M, \quad i \in V, b \in B. \tag{48}$$

$$\beta_i \geq s \cdot b - (1 - \vartheta_{ib}^L)M, \quad i \in V, b \in B. \tag{49}$$

$$\beta_i + u_i \geq s \cdot (b-1) - \vartheta_{ib}^R M, \quad i \in V, b \in B. \tag{50}$$

$$\beta_i + u_i \leq s \cdot (b-1) - 1 + (1 - \vartheta_{ib}^R)M, \quad i \in V, b \in B. \tag{51}$$

$$\zeta_{ib} \geq 1 - \vartheta_{ib}^L - \vartheta_{ib}^R, \quad i \in V, b \in B. \tag{52}$$

The start and end time-steps of a berth segment can be guaranteed by the following two constraints. If vessel i occupies berth segment b (i.e., $\zeta_{ib} = 1$), then the start time of berth segment b is not larger than vessel i's start time-step ε_i.

$$o_b \leq \varepsilon_i + (1 - \zeta_{ib})M, \quad i \in V, b \in B. \tag{53}$$

If vessel i occupies berth segment b (i.e., $\zeta_{ib} = 1$), then the end time-step of berth segment b is not smaller than vessel i's end time-step σ_i.

$$d_b \geq \sigma_i - (1 - \zeta_{ib})M, \quad i \in V, b \in B. \tag{54}$$

The duration from the start time-step to the end time-step of each berth segment should be smaller than H, which is required by considering cyclical arrivals of vessels. For example, if $d_b = 4$ and $o_b = 2$, then the busy time-steps for berth segment b is {2, 3, 4} and the number of busy time-steps is $d_b - o_b + 1 = 3$.

$$d_b - o_b + 1 \leq H, \quad b \in B. \tag{55}$$

4.2.9. Indirect-transshipment constraints

For transshipment activity from vessel i to vessel j, the following three constraints guarantee if vessel j's start time-step does not fall into the range $[\varepsilon_i, \varepsilon_i + f - 1]$ (i.e., $\iota_{ij}^L + \iota_{ij}^R = 1$, not both ι_{ij}^L and ι_{ij}^R equal 1), then indirect-transshipment mode must be applied (i.e., $v_{ij}^I = 1$). Among these constraints, the first one ensures that if $\varepsilon_j \geq \varepsilon_i$, then $\iota_{ij}^L = 1$. the second one means that if $\varepsilon_j \leq \varepsilon_i + f - 1$, then $\iota_{ij}^R = 1$. The third one imposes that if $A_{ij}^T = 1$, and $\iota_{ij}^L + \iota_{ij}^R = 1$, then $v_{ij}^I = 1$.

$$\varepsilon_j \geq \varepsilon_i - (1 - \iota_{ij}^L)M, \quad i, j \in V, i \neq j. \tag{56}$$

$$\varepsilon_j \leq \varepsilon_i + f - 1 + (1 - \iota_{ij}^R)M, \quad i, j \in V, i \neq j. \tag{57}$$

$$v_{ij}^I \geq A_{ij}^T \cdot (2 - \iota_{ij}^L - \iota_{ij}^R), \quad i, j \in V, i \neq j. \tag{58}$$

4.2.10. Variable ranges

The range of decision variables are given below.

$$\beta_i, \varepsilon_i, \sigma_i, \tau_i^a, \tau_i^b, o_b, d_b, \rho_t \in \mathbb{Z}_+, \quad i \in V, b \in B, t \in T. \tag{59}$$

$$\omega_{ib}, \varphi_{ik}, \xi_{ikt}, \zeta_{ib} \in \{0, 1\}, \quad i \in V, b \in B, k \in K^I \bigcup K^E \bigcup K^T, t \in T. \tag{60}$$

$$\gamma_{ip}, \eta_{ipt}, \pi_{it}, \theta_{it} \in \{0, 1\}, \quad i \in V, p \in P_i, t \in T. \tag{61}$$

$$\delta_{ij}^x, \delta_{ij}^y, v_{ij}^D, v_{ij}^I, \lambda_i^I, \lambda_i^E, \lambda^T U_{ij}, \lambda_j^{TL}, \lambda_{ij}^T \in \{0, 1\}, \quad i, j \in V, i \neq j. \tag{62}$$

$$\psi_{jkib} \in \{0, 1\}, \quad i, j \in V, i \neq j, k \in K^T, b \in B. \tag{63}$$

$$\kappa_{jkb} \in \{0, 1\}, \quad j \in V, k \in K^I \bigcup K^E \bigcup K^T, b \in B. \tag{64}$$

$$\varpi_{ib_1 jb_2} \in \{0, 1\}, \quad i, j \in V, i \neq j, b_1, b_2 \in B, b_1 \neq b_2. \tag{65}$$

$$\theta_{ib}^L, \theta_{ib}^R, \iota_{ij}^L, \iota_{ij}^R \in \{0, 1\}, \quad i, j \in V, i \neq j, b \in B. \tag{66}$$

For the CTIP, we first attempt to obtain the true Pareto front with ϵ-constraint method, which consists of a series of iterations. In the application of ϵ-constraint method, we transform OBJ_2 into a constraint with a gradual reduction value ϵ in each iteration, and set OBJ_1 as the single objective. In particular, the values of ϵ can be chosen from an appropriate set, i.e., the range of OBJ_2 values.

To obtain a lower bound on OBJ_2, denoted by LB_{OBJ_2}, we simply need to omit objective OBJ_1 and solve the single-objective (i.e., only OBJ_2) integer program with CPLEX. Clearly, LB_{OBJ_2} is an ideal value for OBJ_2.

To generate an upper bound on OBJ_2, denoted by UB_{OBJ_2}, we first find the lower bound on OBJ_1 (i.e., omit OBJ_2 and solve the single-objective (i.e., OBJ_1) integer program with CPLEX), denoted by LB_{OBJ_1}, then we transform the first objective OBJ_1 into a constraint such that $\sum_{i \in V} c_i^I \lambda_i^I + \sum_{i \in V} c_i^E \lambda_i^E + \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij}^T \lambda_{ij}^T = LB_{OBJ_1}$ and solve the single-objective (i.e., OBJ_2) integer program with CPLEX. The meaning of these operations is that to achieve the ideal value LB_{OBJ_1} for OBJ_1 how much cost for OBJ_2 should be paid.

Now we have the range of OBJ_2, i.e., $[LB_{OBJ_2}, UB_{OBJ_2}]$. Applying ϵ-constraint method, we omit OBJ_2 and iteratively add (or update) a constraint $\sum_{i \in V} (w_i^a \tau_i^a + w_i^b \tau_i^b) \leq \epsilon$ for $\epsilon = UB_{OBJ_2}, UB_{OBJ_2} - \Delta, UB_{OBJ_2} - 2\Delta, \ldots, LB_{OBJ_2}$, where Δ denotes the minimum unit of the value for OBJ_2.

Following these specifications of ϵ-constraint method, we apply CPLEX to solve the resultant single-objective integer programs. The experiment environment is specified in Section 6. However, even for a very small scale instance with six vessels and quay wharf of 500 m, CPLEX takes several hours for one iteration in ϵ-constraint method. Thus, exact solution approaches, such as ϵ-constraint method, are not applicable for instances of the CTIP in practice. Therefore, we need to design efficient heuristic approaches to generate an approximate Pareto front for large scale instances.

5. Bi-objective solution approach

In this section, to solve the CTIP, we devise a variant of non-dominated sorting genetic algorithm II (NSGA II for short), which is proposed by Deb et al. (2002). As NSGA II is an elitist multi-objective evolutionary algorithm and a powerful yet simple tool to implement, this algorithm has been widely employed in the literature to solve theoretical and practical multi-objective problems. In the following, we first introduce the framework of our proposed approach, and then present algorithm details.

5.1. Introduction of NSGA II

We adopt the framework of NSGA II to find an approximate Pareto front solutions for the CTIP. NSGA II is a widely used population-based multi-objective genetic algorithm. The framework of NSGA II is given in Algorithm 1 .

Algorithm 1: Framework of NSGA II for the CTIP.

Data: Problem parameters.
Result: Pareto front solution set.
pop = 100 (% Population size);
gen = 20 (%Generation number);
P = INITIALIZE_POPULATION();
P = NON_DOMINATION_SORT(P);
for *i = 1: gen* **do**
    pool = gen / 2 (% Mating pool size);
    tour = 2 (% Tournament size);
    parent_P = TOURNAMENT_SELECTION(P, pool, tour);
    offspring_P = CROSSOVER_MUTATION(parent_P);
    intermediate_P = parent_P $\bigcup$ offspring_P;
    F = NON_DOMINATION_SORT(intermediate_P);
    P = ∅;
    i = 1 (%Non-domination level index);
    while $|P| + |F_i| \leq pop$ **do**
        P = P $\bigcup F_i$;
        i = i + 1;
    end
    K = pop - |P| (%Number of individuals to be further added);
    P = P $\bigcup$ SELECTION(P, F_i, K);
end

In Algorithm 1, the procedure INITIALIZE_POPULATION() randomly generates a number of feasible solutions, which also incorporates a feasibility check process (which will be introduced later). The initial population is sorted by NON_DOMINATION_SORT() procedure. For each generation, the offspring solutions are generated by genetic algorithm operators defined in CROSSOVER_MUTATION() function. Next, a recombination process is needed. Specifically, the routine NON_DOMINATION_SORT() sorts the members in set P into several non-domination levels $\{F_1, F_2, \ldots, F_i, \ldots\}$. Then the members of levels $F_1, \ldots, F_{i-1}$ are included in set P. We also need K members from the next non-selected level F_i to meet the requirement of *pop*, which is realized by SELECTION(P, F_i, K) using a binary tournament selection policy. For more details of NSGA II, interested readers may refer to Deb et al. (2002).

5.2. Solution representation

A solution S is associated with four decision sets: (i) the berthing position $\beta_i \in [0, L]$ for each vessel $i \in V$, (ii) the start time-step $\varepsilon_i \in [1, H]$ for each vessel $i \in V$, (iii) the QC-profile selected for each vessel $i \in V$, say $p_i \in [1, |P_i|]$, and (iv) the assignments of each yard subblock to each vessel $\varphi_{ik} \in [0, 1]$, where $i \in V, k \in K^E \bigcup K^I \bigcup K^E$. The above four sets of decision variables are all integral.

Observe that once berthing position β_i is determined, the berth segments occupied by a vessel i (i.e., variables ω_{ib}) can also be determined. After we choose the start time-step ε_i and the serving QC-profile p_i, the end time-step of vessel σ_i is also determined. At the same time, we can also calculate the number of used QCs in each time-step in the planning horizon. Most importantly, we can obtain the mode of transshipment for two vessels with transshipment activity, i.e., if the duration between the start time-steps of these two vessels is not larger than six time-steps, the transshipment activity is performed in direct-transshipment mode; otherwise, indirect-transshipment mode is applied. Once the assignments of subblocks to vessels (i.e., variables φ_{ik}) are known, together with the information of QC-profile assignments and vessel start time-steps, we can obtain the information when a subblock is in loading activity. Besides, the average lengths of unloading route and loading route can also be determined with the above information.

5.3. Feasibility check

The feasibility check procedure, denoted by FEASIBILITY_CHECK(), is to guarantee feasible solutions for the CTIP. After generating a possible solution S, the feasibility check must be performed. The feasibility check includes multiple checking procedures: (i) the selected berthing position should be within the quay wharf, (ii) the time-window (determined by the

start time-step and QC-profile used in service) assigned to the vessel should be within the feasibility range $[a^M, b^M]$, (iii) the number of assigned yard import, export, and transshipment subblocks for a vessel should be sufficient for import, export, and transit containers, respectively, (iv) there should be no overlap among rectangles in the berth plan, (v) the number of used QCs at each time-step should not be larger than the total number of available QCs, (vi) in each time-step, at most one subblock in the yard neighborhood pairs is in loading activity, (vii) in each time-step, at most one subblock among each yard block is in loading activity. If the feasibility check is not passed for a possible solution S, then a very large penalty is added to both its corresponding objective values. The purpose is to make this infeasible solution to be discarded in the NON_DOMINATION_SORT() procedure according the fitness values, where the fitness criteria are represented by the two objective values.

5.4. Population initialization

In NSGA II, the initial population consists of a number of *pop* feasible initial solutions. We employ random-selection-based procedure to generate each individual solution (see Algorithm 2).

5.5. Non-domination sort

The initialized population is sorted based on non-domination, where a fast sorting algorithm (Deb et al. (2002)) is widely utilized. We use this classic sorting method, which is described in Algorithm 3.

5.6. Crowding distance

The crowding distance is assigned once the non-domination sort is completed. According to NSGA II, the individuals are selected based on their ranks and crowding distances. Crowding distance is assigned front-wise, which is calculated with the method described in Algorithm 4 (c.f. Deb et al. (2002)). The basic idea is to find the Euclidian distance between each individual in a front based on their m objective values in the m dimensional hyper space. For the CTIP, the number of objectives is $m = 2$.

5.7. Selection

Once the individuals are sorted by NON_DOMINATION_SORT() and assigned with crowding distances, the selection is implemented with a *crowded-comparison-operator* ($\prec_n$). The comparison is implemented in Algorithm 5 (c.f. Deb et al. (2002)).

5.8. Genetic operators

We implement a read-coded genetic algorithm, which uses simulated binary crossover operator and polynomial mutation.
The crossover procedure is commonly used in genetic algorithm for producing offspring solutions from parent solutions. Generally, two parent solutions are randomly selected to generate two offsprings after mating selection, with a specified crossover probability P_r. Simulated binary crossover simulates the binary crossover, which is given below (Deb et al. (2002))

$$c_{1,k} = \frac{(1-\beta_k)p_{1k} + (1+\beta_k)p_{2k}}{2}, \quad c_{2,k} = \frac{(1+\beta_k)p_{1k} + (1-\beta_k)p_{2k}}{2},$$

where c_{ik} is the i-th child with k-th component, p_{ik} is the selected parent and $\beta_k \geq 0$ is a sample from a random number generated having the density:

$$p(\beta) = \begin{cases} \left(\frac{\eta_c + 1}{2}\right)\beta^{\eta_c}, & \text{if } 0 \leq \beta \leq 1; \\ \left(\frac{\eta_c + 1}{2}\right)\frac{1}{\beta^{\eta_c + 2}}, & \text{if } \beta > 1, \end{cases}$$

where η_c is the distribution index for crossover.

After performing the crossover operator, a mutation procedure is conventionally invoked to help evolutionary algorithms escape from local optima. With mutation probability P_m, we implement a polynomial mutation, which can be realized as follows

$$c_k = p_k + (p_k^u - p_k^l)\delta_k,$$

where c_k is the child, p_k is the parent with p_k^u (resp. p_k^l) being the upper (resp. lower) bound on the parent component, and δ_k is the small variation which is calculated from a polynomial distribution:

Algorithm 2: Framework of initial solution generation.

Data: Problem parameters and population size *pop*.
Result: Initial solution set.
$S_{ini} = \emptyset$;
for *pop_iter = 1 : pop* **do**
    for *iter_1 = 1: iter_max* **do**
        for *i = 1 : |V|* **do**
            beta = randi(0, L) (%Generate berthing position);
            varepsilon = randi(a^M, b^M) (%Generate start time-step);
            gamma = randi(1, |P_i|) (%Generate QC-profile);
            Randomly assign subblocks *varphi*;
        end
        if [*beta, varepsilon, gamma, varphi*] *is feasible* **then**
            break;
            found_feasible_solution = 1;
        end
    end
    if *found_feasible_solution is equal to zero* **then**
        while *true* **do**
            Randomly construct a *vessel_list*, not violating the vessel transshipment (or precedence)
            relations;
            gamma = randi(1, |P_i|) (%Randomly assign QC-profiles);
            vessel_row = 1 (%Row number of vessels on the berth plan);
            berth_position = 0 (%Berth length used in a row);
            while *vessel_list is not empty* **do**
                if *berth_position + vessel_length ≤ total_berth_length* **then**
                    Take the first vessel from *vessel_list*;
                    Set the vessel's start time-step as *(vessel_row − 1) * largest_handling_time + 1*;
                    Update *berth_position = berth_position + vessel_length*;
                    Delete this vessel from *vessel_list*;
                else
                    Update *vessel_row = vessel_row + 1*;
                    Update *berth_position = 0*;
                end
                Randomly assign subblocks *varphi*;
            end
            if [*beta, varepsilon, gamma, varphi*] *is feasible* **then**
                break;
            end
        end
    end
    Update $S_{ini} = S_{ini} \bigcup \{S\}$;
end

$$\delta_k = \begin{cases} (2r_k)^{\frac{1}{\eta_m+1}} - 1, & \text{if } r_k < 0.5; \\ 1 - [2(1 - r_k)]^{\frac{1}{\eta_m+1}}, & \text{if } r_k \geq 0.5, \end{cases}$$

where r_k is an uniformly sampled random number between 0 and 1, and η_m is mutation distribution index (c.f. Deb et al. (2002)).

After crossover and mutation, the offspring solutions may be infeasible due to (i) non-integral and (ii) infeasible for the CTIP, or to say, not passing FEASIBILITY_CHECK(). To overcome this difficulty, once crossover and mutation are completed, we first round each component coded in each solution to the nearer integer, and then we apply feasibility check procedure. If a solution does not pass the feasibility check, we add a very large integer or penalty to its both objective values. The purpose is to make it to be discarded in future selection process.

Algorithm 3: Framework of non-dominated sort.

---

for $p = 1 : pop$ **do**
    Initialize $S_p = \emptyset$ (%It contains individuals dominated by p);
    Initialize $n_p = 0$ (%The number of individuals dominating p);
    for $q = 1 : pop$ **do**
        if *p dominates q* **then**
            $S_p = S_p \bigcup \{q\}$;
        else
            if *q dominates p* **then**
                $n_p = n_p + 1$;
            end
        end
    end
    if *n_p is equal to zero* **then**
        $p_{rank} = 1$ (%The rank of individual p);
        $F_1 = F_1 \bigcup \{p\}$ (%First front set);
    end
end
Initialize $i = 1$ (%The front counter);
while *F_i is not empty* **do**
    $Q = ($% The set stores the individuals in the $(i + 1)$-th front);
    for *p in front F_i* **do**
        for *q in S_p* **do**
            $n_q = n_q - 1$;
            if $n_q == 0$ **then**
                $q_{rank} = i + 1$;
                $Q = Q \bigcup \{q\}$;
            end
        end
    end
    $i = i + 1$;
    $F_i = Q$;
end

Algorithm 4: Framework of crowding distance calculation.

Initialize i = 1 (% The front counter);
while *F_i is not empty* **do**
    Initialize the distance to be zero for all individuals, i.e., $F_i(d_j) = 0$, where j corresponds to the j-th individual in front F_i;
    $m_{max} = 2$ (% The number of objectives);
    for $m = 1 : m_{max}$ **do**
        Sort the individuals in front F_i based on objective m, i.e., $I = sort(F_i, m)$;
        Assign infinite distance to boundary values for each individual in F_i, i.e., $I(d_i) = \infty$ and $I(d_n) = \infty$, where n is the number of individuals in this front;
        for $k = 2 : (n - 1)$ **do**
            $I(d_k) = I(d_k) + \frac{I(k+1).m - I(k-1).m}{f_m^{max} - f_m^{min}}$, where $I(k).m$ is the value of the m-th objective function of the k-th individual in I, f_m^{max} is the maximum value of the m-th objective, and f_m^{min} is the minimum value of the m-th objective;
        end
    end
    $i = i + 1$;
end

Algorithm 5: Framework of selection.

Initialize $i = 1$ (% The front counter);
while *Front F_i is not empty* **do**
  Assign each individual p in F_i with rank $p_{rank} = i$;
  $i = i + 1$;
end
for $p = 1 : pop$ **do**
  for $q = 1 : pop$ **do**
    if $p_{rank} < q_{rank}$ **then**
      $p \prec_n q$;
    end
    if *p and q belong to the same front F_i and $F_i(d_p) > F_i(d_q)$* **then**
      $p \prec_n q$;
    end
  end
end

Table 2
Parameters for instance class generation .

Class	Vessels	Berth (m)	Total QCs	Import subblocks	Export subblocks	Transshipment subblocks
Super_Small	6	500	6–8	5	15	5
Small_1	10	1000	14–16	10	30	10
Small_2	20	1400	18–20	20	60	20
Medium_1	30	1800	22–24	30	90	30
Medium_2	40	2200	26–28	40	120	40
Large_1	50	2600	30–32	50	150	50
Large_2	60	3000	34–36	60	180	60

5.9. Recombination

According to Deb et al. (2002), the selection is performed to choose the individuals from the combined population for the next generation, where the combined population is constructed from offspring solutions and the population in current generation. Then non-domination sort is performed for the combined population, which generates a number of fronts. The new generation is filled by each front subsequently until the population size exceeds the current population size. If by adding all the individuals in front F_i the population exceeds *pop* then individuals in front F_i are selected based on their crowding distances in the descending order until the population size *pop* is reached.

6. Simulation study

In this section, we evaluate and analyze the performance of the proposed solution approach on a series of instances. Our algorithm NSGA II for the CTIP is implemented in Matlab R2014. Computational experiments have been conducted on a Mac computer with 2.4 GHz Core i7 and 8 GB Memory. CPLEX 12.6 is used and called in Matlab to solve the CTIP model with a weighted sum of two objectives (employed by Weighted Method which will be introduced later). The computation time is calculated in CPU seconds on this testing computer.

6.1. Instance generation

The experiments have been carried out on 60 instances with different berth and yard structures. These instances are generated based on the layout of Shanghai Port. The settings of instances are detailed as follows.

- The planning horizon is one week, i.e., $H = 42$ time-steps, each lasting for 4 h. This is a typical setting in Zhen et al. (2011).
- The maximum vessel handling time is $E = 6$ time-steps.
- The enlarged planning horizon is $|T| = H + E = 48$ time-steps.

In this study, we mainly generate and test six classes of instances (see Table 2). In the Small_1, Small_2, Medium_1, Medium_2, Large_1, and Large_2 instance classes, we test 10 × 1000, 20 × 1400, 30 × 1800, 40 × 2200, 50 × 2600, and 60 × 3000, respectively, where ($|V| \times L$) denotes a number of vessels $|V|$ and quay wharf length L. In particular, to test the performance of Weighted Method (which is based on the integer program in Section 4), we design a Super_Small class,

Table 3
Parameters for vessel and QC-profile generation.

Class	Percentage	Vessel length (m)	Used QCs	Handing time	Workload	Import (%)	Export (%)	Transit (%)
Feeder	1/3	100–200	1–3	2–4	2–5	20	60	20
Medium	1/3	200–300	2–4	3–5	6–14	20	60	20
Jumbo	1/3	300–400	3–6	4–6	15–20	20	60	20

because, even for instance class Small_1, CPLEX (employed by Weighted Method) is out of memory in the solution process. In all the instance classes, the number of QC-profiles for each vessel is set to be 10. The number of available QCs in each time-step is randomly generated, and these informations are summarized in Table 2). The difference of maximum and minimum numbers of QCs is two, which conforms with the investigation of Shanghai Port, since in one time-step, at most two QCs are in maintenance activity. At Shanghai Port, the yard layout is divided into import area (possessing 20% in all the yard storage), export area (60%), and transshipment area (20%). From the quay wharf to the inland, the deployment of storage is that transshipment blocks, followed by import blocks and then by export blocks (see Fig. 4). According to Shanghai Port's daily practice, the current settings conform with instance class Large_1, i.e., quay wharf of 2600 m and 50 vessels per week on average. For example, in Small_1 class, each problem instance is associated with 10 vessels, berth length of 1000 m, from 14 to 16 QCs in each time-step, 10 subblocks in yard import area, 30 subblocks in yard export area, and 10 subblocks in yard transshipment area.

For vessels, in line with the vessel generation method utilized in Zhen et al. (2011), we distinguish between three vessel classes: Feeder, Medium and Jumbo. In generating test instances, a vessel is randomly selected to be a Feeder, Medium, or Jumbo with equal possibility. The expected start time-steps of vessels are randomly distributed along the planning horizon with H time-steps. A QC-time-step is set to accommodate $30 \times 4 = 120$ containers, as a quay crane performs 30 operations an hour on average and a time-step comprises 4 h. According to Table 3, the average lengths of the three classes are 150 m, 250 m, and 350 m, respectively. The average handling times of the three classes are 3, 4, and 5, respectively. The average workloads of the three classes are 3.5, 10, and 17.5 QC-time-steps, respectively. Based on the characteristics of Shanghai Port, the import containers occupy about 20%, the export containers hold about 60%, and the transit containers possess about 20% for each vessel on average. The information of vessel generation details is summarized in Table 3.

For the QC-profile generation, the variance between QC numbers in adjacent two time-steps is set to be one (c.f. Vacca et al. (2013)). QC-profiles are generated in the following way. Based on the range of each type of vessels specified in Table 3, the number of required QC-time-steps is generated randomly. Then a pool of all feasible QC-profiles can be obtained by enumeration method. We randomly select 10 profiles from the pool for each vessel as the set of all possible QC-profiles.

According to Shanghai Port's yard layout, the width, height and length of a container subblock on the yard are six, four, and five containers, respectively. Thus, a container subblock can typically hold one QC-time-steps, i.e., 120 containers. The planning of yard storage allocation is based on subblock level. For the width of travel paths in the yard, we set it to be 30 m. The width of two adjacent rows of blocks is set as 35 m. The length of a subblock is 50 m and thus a yard storage block lasts for 25 m. Between two blocks in the same row, the blank takes 50 m for yard cranes turn-directions. The width between the neatest row of blocks to the quay wharf is set to be 50 m. With these specifications, the lengths of loading and unloading routes can be calculated precisely.

For parameters w_i^a and w_i^b, i.e., the unit time-step weights assigned to earliness and tardiness respectively, we set them to be one for all vessels. We assume that all shipping companies have the same priority.

For generating the expected time windows (i.e., $[a^e, b^e]$) and the feasible time-windows (i.e., $[a^M, b^M]$) for vessels, they are randomly distributed along the planning horizon. For each vessel, the length of the expected time-window is about the same length of its average handling time, and the length of the feasible time window is five times of the length of its average handling time.

To generate the number of transit containers c_{ij}^T from vessel i to vessel j, we first sort the vessels in an increasing order of their expected arrival times, resulting in a vessel list. Then we set that the transit container flows come from the first half (vessels) in the list and go to the last half (vessels) in the list. The number of transit containers is generated randomly based on the minimum capacity of the two associated vessels. Each vessel is associated with at most one transshipment task. This number should not be larger than half the workload of the capacity, because the workload of a vessel is the capacity of total unloading and loading tasks. After all the c^T generations are completed, then the number of import containers c^I can be generated. In our experiments, we set this number as the half of workload minus the capacity of transshipment tasks, if any. The number of export containers c^E should be set as the half of the workload. We attempt to guarantee the export containers occupy about 60% of the vessel workload.

For r^I, r^E and r^T, i.e., the number of required subblocks in the import, export and transshipment areas respectively, their generation are naturally based on the number of containers c^I, c^E and c^T. As one subblock can typically hold one QC-time-step workload, we set r^I, r^E and r^T as the ceilings of $c^I/120$, $c^E/120$ and $c^T/120$, respectively. For berth segment length s, it typically takes 50 m in the literature (c.f. Zhen et al. (2011)).

6.2. Performance measurement

A promising bi-objective algorithm lies in two aspects: (i) the algorithm can find a set of solutions that are close to the true Pareto front, and (ii) the algorithm can find a set of solutions with a large diversity. In order to measure the performance of the proposed algorithm and other possible algorithms (for which we then design Weighted Method based on integer program), the following indicators are introduced: (i) *inverted generational distance* indicator, or called *the distance from reference set* (Coello and Cortés (2005)), which could measure how "far" an approximate Pareto front is from the true Pareto front, (ii) *set coverage* indicator (Zitzler et al. (2003)) which illustrates the dominance relations between two sets of non-dominated solutions, and (iii) *maximum spread* indicator (Zitzler et al. (2000)) which evaluates the spread of a non-dominated solution set.

6.2.1. Inverted generational distance

Inverted generational distance is the most important indicator among the three indicators, which could measure how "far" an approximate Pareto front is from the true Pareto front. The inverted generational distance indicator (I_D) is used to measure the performance of both convergence and spread. Given a non-dominated solution set A and a reference set P^*, I_D is defined as

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

where $d(x, y)$ indicates the distance between the solutions x and y, which is computed based on the relative Euclidean distance

$$d(x, y) = \sqrt{\sum_{i=1}^{M} \left(\frac{f_i(x) - f_i(y)}{f_i^{max} - f_i^{min}} \right)^2},$$

where f_i^{max} (resp. f_i^{min}) is the maximum (resp. minimum) value of the i-th objective (M in total, in our context $M = 2$) among the solutions in P^*.

The reference set P^* should ideally be the true Pareto front, obtained by exact methods such as ϵ-constraint method. However, as we say in Section 4, it is difficult to obtain the true Pareto front due to computational complexity of the CTIP. We will confirm this fact in the following experiments (by using Weighted Method to be introduced later). In our experiments, P^* is set as the Pareto front of all the solutions obtained by all algorithms in comparison. A small $I_D(A, P^*)$ means that front A is a good approximation of the reference set P^*. An efficient and promising algorithm should have a small $I_D(A, P^*)$ value.

6.2.2. Set coverage

Suppose that A and B are two sets of non-dominated solutions, the set coverage indicator $C(A, B)$ is defined as

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

where $y \succeq x$ indicates that solution y dominates solution x. $C(A, B) = 1$ if every solution in B is dominated by some solution in A, and $C(A, B) = 0$ if no solution in B is dominated by some solution in A. Both $C(A, B)$ and $C(B, A)$ should be taken into consideration to compare sets A and B. If $C(A, B) > C(B, A)$, A is said to be better than B in terms of their dominance relations. Note that $C(A, B) + C(B, A) \leq 1$.

6.2.3. Maximum spread

Maximum spread (MS) indicator measures the distance between the boundary solutions in a non-dominated solution set. For the non-dominated solution set A, $MS(A)$ is defined as

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

where f_i^{max}, f_i^{min} and M are defined as that in the definition of I_D. A large $MS(A)$ (MS for short) means that a wide range of objective values are covered by the non-dominated solutions in A.

6.3. Compared algorithms in computational experiments

The compared algorithms include: (i) NSGA II, (ii) Random List, and (iii) integer-program-based Weighted Method. In Section 6.3.1, we mainly present the parameters used in NSGA II method. Section 6.3.2 and 6.3.3 give the details for Random List method and Weighted Method, respectively.

Table 4
Parameters for NSGA II.

Parameter	Value
Population size (pop)	100
Generation number (gen)	20
Crossover probability (P_r)	0.9
Mutation probability (P_m)	0.1
Tournament size	$0.5 \times pop$

6.3.1. NSGA II method

For NSGA II, we set the parameters of the algorithm according to Table 4. The population size is set to be 100 by a rule of thumb. In our preliminary experiments on the instances in Small_1 class, the generation number of 20 is suitable for convergence. We also tried to tune crossover probability P_r and mutation probability P_m, and the values are suitable to be set as 0.9 and 0.1, respectively. In our experiments, tournament size is set to be $0.5 \times pop$, in line with the classic NSGA II.

6.3.2. Random list method

For the Random List approach, we randomly attempt a number of times to generate feasible solutions. Then we select the non-dominated solutions. The details are described in Algorithm 6 . The purpose of setting the maximum iteration to be 500

Algorithm 6: Scheme of random list method.

Data: Problem parameters.
Result: Pareto front solution set.
Iteration_max = 500;
$S_{total} = \emptyset$;
for $iter = 1 : Iteration\_max$ **do**
   while *true* **do**
      Randomly construct a *vessel_list*, conforming with vessel transshipment precedences;
      $gamma = randi(1, |P_i|)$ (%Randomly assign QC-profiles);
      $vessel\_row = 1$; $berth\_position = 0$;
      while *vessel_list is not empty* **do**
         if $berth\_position + vessel\_length \leq berth\_length$ **then**
            Take the first vessel from *vessel_list*;
            Set the vessel's start time-step as $(vessel\_row - 1) * largest\_handling\_time + 1$;
            Update $berth\_position = berth\_position + vessel\_length$;
            Delete this vessel from *vessel_list*;
         else
            Update $vessel\_row = vessel\_row + 1$; $berth\_position = 0$;
         end
         for *i = 1 to V* **do**
            $temp\_KT = randperm(1 : |K^T|)$;
            Set the first r_i^T variables equal to one in $temp\_KT$;
            $temp\_KI = randperm(1 : |K^I|)$;
            Set the first r_i^T variables equal to one in $temp\_KI$;
            $temp\_KE = randperm(1 : |K^E|)$;
            Set the first r_i^T variables equal to one in $temp\_KE$;
            $varphi(i) = [temp\_KT, temp\_KI, temp\_KE]$ (%Randomly assign subblocks);
         end
      end
      $S = [beta, varepsilon, gamma, varphi(1), ..., varphi(|V|)]$;
      If FEASIBILITY_CHECK(S) succeeds, then break;
   end
   Update $S_{total} = S_{total} \bigcup \{S\}$;
end
S_{total} = NON_DOMINATION_SORT(S_{total});

in Random List is that the running times of NSGA II and Random List algorithms are roughly the same in the experiments. Random List involves relatively more randomness compared with NSGA II.

6.3.3. Weighted method

Weighted Method is based on integer program proposed in Section 4. We describe this approach in Algorithm 7 . In

---

Algorithm 7: Scheme of weighted method.

Data: Problem parameters.
Result: Pareto solution set.
$S = \emptyset$ (%Solution set);
$i = 1$ (%The iteration counter);
for $w_1 = 0 : 0.05 : 1$ **do**
　　$w_2 = 1 - w_1$;
　　Solve with CPLEX within 600 seconds the following integer program: $S_i = \{\min w_1 \cdot OBJ_1 + w_2 \cdot OBJ_2$: subject to constraints (3) - (66)\};
　　$S = S \bigcup S_i$;
end

---

each single run of CPLEX, we set a time limit of 600 s. This is because CPLEX solver cannot generate an optimal solution within 600 s. Therefore, we let CPLEX stop in such an amount of time in each iteration. As Weighted Method consists of 21 iterations, each consuming 600 s, and thus one run of Weighted Method consumes 12,600 s (about 4 h) in total.

In order to test the effectiveness of our proposed algorithm, the true Pareto front should be needed. As mentioned above, the true Pareto front cannot be obtained in a reasonable time, and thus it is impossible to directly measure the performance of the approach. Therefore, we compare our approach with this commonly used Weighted Method for very small scale instances.

6.4. Computational results and analysis

We conduct experiments for all the proposed algorithms, according to the settings in Section 6.1. We then apply the three indicators to test the performance of the algorithms. For each of Super_Small instances, 10 independent runs are carried out for NSGA II, and one run is conducted for Weighted Method as it is based on integer program and extremely time-consuming. For each of Small, Medium, and Large instances, 10 independent runs are carried out for both NSGA II and Random List methods. Then all the results are collected for analysis.

For Super_Small instances, we compare NSGA II and Weighted Method in Table 5. For NSGA II, its average values are reported in the table. The average values for these two methods are also reported in the bottom line of this table. In Table 5, the instance names are listed in the column "Instance". The columns under NSGA II respectively represent the average values of I_D, $C(A, B)$ (or $C(B, A)$), MS and computation time for this algorithm, because for each instance NSGA II have been run 10 times. The columns under Weighted Method report the value of I_D, $C(A, B)$ (or $C(B, A)$), MS and computation time for this algorithm in one single run. The unit of computational time is 1 s.

In terms of running time, NSGA II has a very great superiority. For each instance, NSGA II takes about 4 s, however, Weighted Method consumes about 4 h. In terms of the indicator I_D, NSGA II achieves an average value of 0.52 for all the instances, and this value for Weighted Method is zero. This results suggest that, for NSGA II, the distance from reference set

Table 5
Computational results for Super_Small instances .

Instance	NSGA II (referred to as A)				Weighted method (referred to as B)			
	I_D	$C(A, B)$	MS	Time	I_D	$C(B, A)$	MS	Time
6_1	0.51	0	0	4	0	1	1	12,600
6_2	0.51	0	0	4	0	1	1	12,600
6_3	0.52	0	0	4	0	1	1	12,600
6_4	0.55	0	0	4	0	1	1	12,600
6_5	0.49	0	0	4	0	1	1	12,600
6_6	0.55	0	0	4	0	1	1	12,600
6_7	0.48	0	0	4	0	1	1	12,600
6_8	0.55	0	0	4	0	1	1	12,600
6_9	0.48	0	0	4	0	1	1	12,600
6_10	0.54	0	0	4	0	1	1	12,600
Average	0.52	0	0	4	0	1	1	12,600

Table 6
Computational results for Small_1 and Small_2 instances .

Instance	NSGA II (referred to as *A*)				Random list (referred to as *B*)			
	I_D	$C(A, B)$	MS	Time	I_D	$C(B, A)$	MS	Time
10_1	0.05	0.01	1.07	8	0.77	0.42	0.55	10
10_2	0.06	0.32	1.08	8	0.77	0.23	0.56	10
10_3	0.07	0.37	1.15	8	0.78	0.34	0.63	10
10_4	0.10	0.25	1.40	8	0.82	0.40	0.88	10
10_5	0.04	0.11	0.95	8	0.76	0.36	0.43	10
10_6	0.09	0.10	1.35	8	0.81	0.38	0.83	10
10_7	0.03	0.38	0.86	8	0.75	0.29	0.34	10
10_8	0.09	0.08	1.34	8	0.81	0.42	0.82	10
10_9	0.02	0.04	0.83	8	0.74	0.38	0.30	10
10_10	0.08	0.34	1.28	8	0.80	0.34	0.76	10
Average	**0.06**	0.20	**1.13**	8	0.78	**0.35**	0.61	10
20_1	0.17	0.01	1.05	20	0.22	0.86	0.89	18
20_2	0.17	0.19	1.07	20	0.22	0.50	0.90	18
20_3	0.18	0.22	1.13	20	0.25	0.69	0.97	18
20_4	0.22	0.15	1.38	20	0.37	0.81	1.22	18
20_5	0.16	0.07	0.94	19	0.16	0.73	0.77	18
20_6	0.21	0.06	1.34	20	0.35	0.77	1.18	18
20_7	0.15	0.23	0.85	20	0.12	0.61	0.69	18
20_8	0.21	0.05	1.33	20	0.34	0.85	1.16	18
20_9	0.14	0.03	0.81	20	0.10	0.77	0.65	18
20_10	0.20	0.21	1.27	20	0.32	0.71	1.10	18
Average	**0.18**	0.12	**1.12**	20	0.25	**0.73**	0.95	18

is not far. In terms of $C(A, B)$ indicator, we observe that for such super small instances, the solution set of Weighted Method dominates the solution set of NSGA II. In terms of *MS* indicator, Weighted Method achieves larger *MS* values, and this means that a wide range of objective values are covered by the non-dominated solution set of Weighted Method. Clearly, using about 4 h in computation, the fact that Weighted Method performs better than NSGA II is reasonable.

For Small_1 and Small_2 instances, we attempt to apply Weighted Method. However, this method cannot solve any instances due to "Out of Memory" on the testing computer. Therefore, we only implement and compare the performance of NSGA II and Random List methods in Table 6 for both Small_1 and Small_2 instances. For each instance, 10 independent runs have been conducted for each algorithm. We mark the average value of I_D in bold if it is smaller. We also mark the average values of $C(A, B)$ and *MS* in bold if they are larger. In terms of I_D indicator, the proposed NSGA II performs better than Random List in 17 among 20 instances in Small_1 and Small_2 classes. On average, NSGA II is superior to Random List with regards of I_D values. These results reflect that NSGA II can produce efficient and promising non-dominated solutions. NSGA II performs slightly worse than Random List in terms of set coverage indicator $C(A, B)$. It is reasonable because more randomness is adopted in Random List method. This fact also illustrate the limitation of NSGA II in solving bi-objective optimization problems. In terms of *MS* indicator, as we can see, NSGA II achieves larger values on average. These results show that NSGA II spreads over a wider extent, compared with Random List. Decision maker may have more flexibility in the trade-off of the two objective values among different solutions.

We implement and compare the performance of NSGA II and Random List methods for Medium_1 and Medium_2 instances. For each instance, 10 independent runs have been conducted for each algorithm, and the average values are reported in Table 7. In this table, we also mark the value of I_D in bold if it is smaller, and mark the values of $C(A, B)$ and *MS* in bold if they are larger. In terms of I_D indicator, clearly the proposed NSGA II outperforms Random List on average for both Medium_1 and Medium_2 instances. Among 20 instances, NSGA II generates smaller I_D values than Random List in 18 instances. These results indicate that NSGA II could produce a very promising Pareto front. For Medium_1 instances, NSGA II performs slightly worse than Random List on average in terms of $C(A, B)$ indicator, and for Medium_2 instances, NSGA II outperforms Random List on average in terms of $C(A, B)$ indicator. These figures reflect that on the perspective of set coverage, NSGA II has a little limitation in solving Medium_1 instances for the CTIP. In terms of indicator *MS*, NSGA II outperforms Random List approach for both Medium_1 and Medium_2 instance sets. This fact demonstrates that the solution set generated by NSGA II spread relatively wider.

For each instance in Large_1 and Large_2 classes, 10 independent runs have been carried out for both NSGA II and Random List. The average values are reported in Table 8. In this table, bold figures in columns "I_D", "$C(A, B)$" and "*MS*" imply these ones are superior in comparison between these two approaches. In terms of I_D indicator, the devised NSGA II performs better than Random List for both instance sets. Thus, NSGA II is demonstrated advantageous to generate efficient non-dominated solution set. In terms of $C(A, B)$ indicator, we observe that NSGA II outperforms Random List in both Large_1 and Large_2 instances. These results mean that NSGA II is much better than Random List with regard to their dominance

Table 7
Computational results for Medium_1 and Medium_2 instances .

Instance	NSGA II (referred to as A)				Random list (referred to as B)			
	I_D	C(A, B)	MS	Time	I_D	C(B, A)	MS	Time
30_1	0.20	0.01	0.80	62	0.20	0.77	0.57	48
30_2	0.19	0.25	0.82	62	0.21	0.50	0.58	48
30_3	0.17	0.30	0.94	62	0.27	0.64	0.64	49
30_4	0.09	0.20	1.35	61	0.47	0.73	0.85	48
30_5	0.24	0.09	0.61	61	0.11	0.67	0.47	48
30_6	0.10	0.08	1.28	61	0.43	0.70	0.81	48
30_7	0.27	0.30	0.46	62	0.04	0.58	0.40	48
30_8	0.11	0.07	1.26	62	0.42	0.76	0.80	49
30_9	0.28	0.04	0.40	62	0.10	0.70	0.37	48
30_10	0.13	0.27	1.16	62	0.37	0.65	0.75	48
Average	**0.18**	0.16	**0.91**	62	0.26	**0.67**	0.62	48
40_1	0.05	0.43	1.16	123	0.19	0.11	0.89	152
40_2	0.05	0.84	1.17	123	0.20	0.02	0.90	153
40_3	0.06	0.92	1.23	123	0.22	0.07	0.96	153
40_4	0.10	0.75	1.44	122	0.30	0.10	1.16	153
40_5	0.03	0.57	1.07	123	0.16	0.08	0.79	153
40_6	0.10	0.55	1.40	123	0.29	0.09	1.13	153
40_7	0.01	0.93	0.99	122	0.13	0.05	0.72	153
40_8	0.09	0.53	1.39	123	0.29	0.11	1.12	153
40_9	0.01	0.47	0.96	123	0.11	0.09	0.69	153
40_10	0.08	0.88	1.34	123	0.27	0.07	1.07	153
Average	**0.06**	**0.69**	**1.21**	123	0.22	0.08	0.94	153

Table 8
Computational results for Large_1 and Large_2 instances .

Instance	NSGA II (referred to as A)				Random list (referred to as B)			
	I_D	C(A, B)	MS	Time	I_D	C(B, A)	MS	Time
50_1	0.02	0.57	0.96	238	0.13	0.11	1.14	218
50_2	0.03	0.87	0.97	240	0.13	0.01	1.13	222
50_3	0.03	0.93	1.04	238	0.15	0.06	1.18	230
50_4	0.05	0.81	1.29	238	0.20	0.09	1.35	210
50_5	0.02	0.67	0.84	236	0.10	0.07	1.05	254
50_6	0.04	0.66	1.24	243	0.19	0.08	1.32	250
50_7	0.01	0.94	0.75	230	0.08	0.04	0.99	152
50_8	0.04	0.64	1.23	233	0.19	0.10	1.31	220
50_9	0.01	0.60	1.17	246	0.07	0.08	0.97	220
50_10	0.04	0.90	0.71	239	0.18	0.07	1.27	219
Average	**0.03**	**0.76**	1.02	238	0.14	0.07	**1.17**	220
60_1	0.01	0.58	0.95	421	0.16	0.11	1.06	553
60_2	0.01	0.88	0.96	425	0.16	0.02	1.07	387
60_3	0.02	0.94	1.03	391	0.17	0.07	1.12	329
60_4	0.03	0.82	1.28	410	0.22	0.10	1.28	382
60_5	0.00	0.68	0.83	393	0.13	0.08	0.98	335
60_6	0.03	0.67	1.24	407	0.21	0.09	1.25	514
60_7	0.00	0.95	0.75	409	0.11	0.05	0.93	318
60_8	0.03	0.65	1.22	436	0.21	0.11	1.24	464
60_9	0.00	0.61	0.71	391	0.11	0.09	0.90	444
60_10	0.03	0.91	1.16	415	0.20	0.07	1.20	512
Average	**0.02**	**0.77**	1.01	410	0.17	0.06	**1.10**	422

relations. NSGA II executes slightly worse than Random List in terms of *MS* indicator. These results demonstrate that the solution set of Random List spreads slightly wider than that of NSGA II and also illustrate a limitation of NSGA II.

In Fig. 5, for illustrative purpose, we plot the non-dominated solutions produced by NSGA II and Random List methods in two instances: (i) instance 50_1 in Large_1 class, and (ii) instance 60_1 in Large_2 class. The non-dominated solutions are collected and plotted with different markers. In this figure, the blue marker "∗" represents the solution set produced by NSGA II and the brown marker "+" illustrates the Pareto front obtained by Random List. In this figure, two Cartesian coordinate systems are depicted. In each of the system, the *x*-axis denotes the value of objective function OBJ_1 (proposed
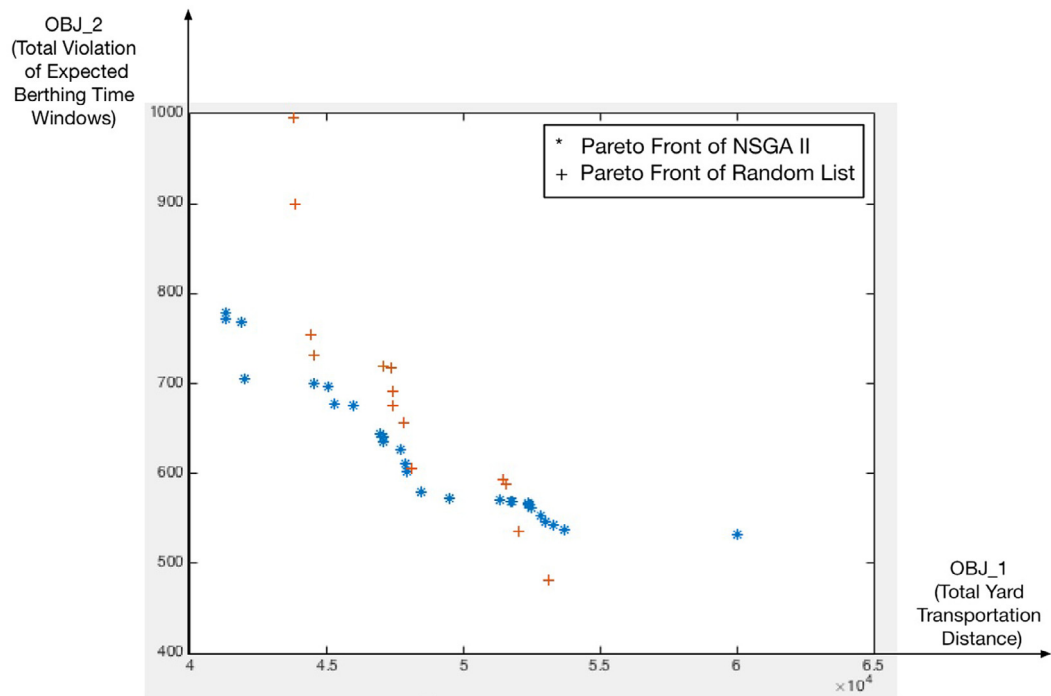
(a) Instance 50_1
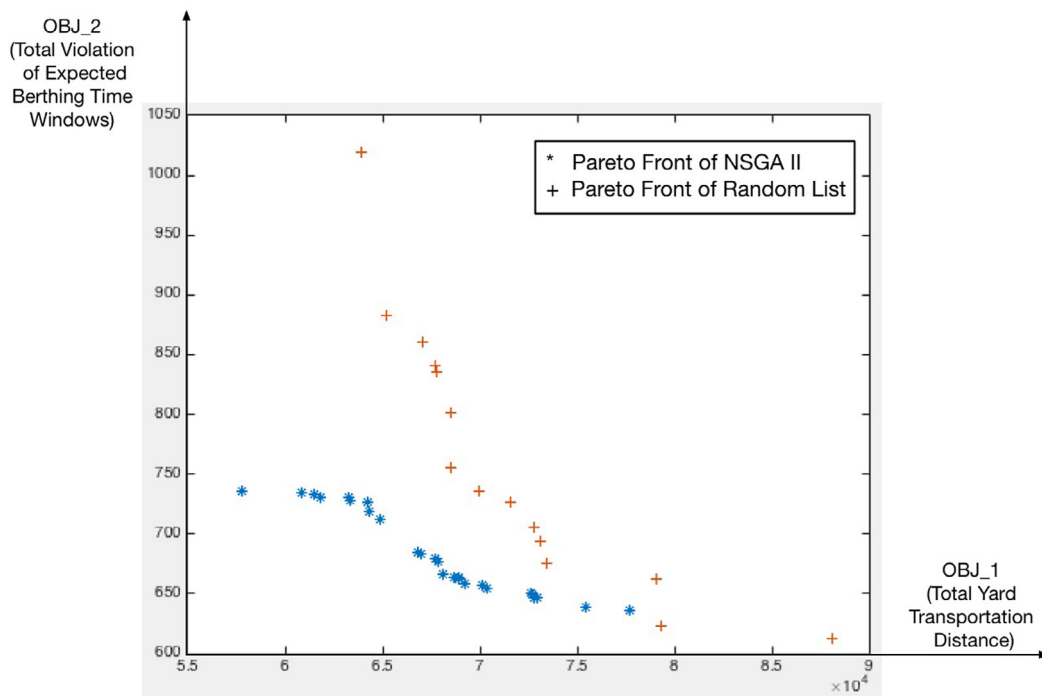
(b) Instance 60_1

Fig. 5. Non-dominated solutions produced by NSGA II and Random List.

for yard transportation distance minimization, in unit of one kilometer meter), and the *y*-axis indicates the value of objective function OBJ_2 (proposed for berthing time windows violation minimization, in unit of one). For example, in Fig. 5(a), we observe that at the cost of increasing yard transportation distance (i.e., for OBJ_1), the total violation of berthing time windows decreases (i.e., for OBJ_2), and vice versa. This figure well illustrates the trade-off relation between the two studied objectives. Besides, as we can see, the graphical results agree with the above analysis.

According to the above computational results, NSGA II is better than Random List. In terms of I_D indicator, NSGA II is strictly better than Random List in instances of all six classes. In terms of $C(A, B)$ indicator, NSGA II and Random List are roughly the same, considering all the instances. In terms of MS indicator, NSGA II outperforms Random List in general, in all the instances. We also observe that in Medium_2 instances, NSGA II is strictly better than Random List in terms of all three indicators. Summing up, these results confirm that NSGA II can produce a set of non-dominated solutions with excellent quality and satisfactory diversity. We observe that even in problem instances of Large_2 class where we test 60 vessels and quay wharf of 3000 m, NSGA II can generate solution sets in a short time about 410 s. This fact demonstrates NSGA II can be applied in practice to fast address large scale problems. Therefore, the proposed NSGA II is suggested to be used in practice for solving the CTIP.

7. Conclusion

This paper considers the joint optimization of the tactical berth allocation and the tactical yard allocation in container terminals. The berth side and yard side operations are simultaneously addressed. To balance the dissatisfaction of shipping liners and the cost of container terminal, we study two objectives: (i) the minimization of the violation of the vessels' turnaround time-windows, and (ii) the minimization of the total yard transportation distance. To describe this complex system, we propose a comprehensive mathematical model. With this model, we can simultaneously address the import, export and transshipment container tasks in port daily practice, we can optimally decide which transshipment modes should be applied for transit containers. To produce Pareto solutions, we devise three methods. The first one is an evolutionary algorithm, which is based on the framework of non-dominated sorting genetic algorithm II (i.e., NSGA II). The second is a weighted method based on integer program and solved with CPLEX. The third is a constructive heuristic method, which is based on List scheduling heuristic and vessel lists are generated randomly in multiple iterations. Numerical experiments have been conducted and computational results demonstrate the efficiency of our proposed NSGA II.

Future research directions may include (i) development of more efficient bi-objective algorithm, and (ii) consideration of the container port integrated planning in stochastic settings, such as stochastic arrival or delay of vessels, export container tasks' arrival under uncertainty. In the stochastic situations, a robust integration plan is very valuable.

Acknowledgment

References

Alessandri, A., Sacone, S., Siri, S., 2007. Modelling and optimal receding-horizon control of maritime container terminals. J. Math. Model. Alg. 6 (1), 109–133.
Bell, M.G.H., Liu, X., Angeloudis, P., Fonzone, A., Hosseinloo, S.H., 2011. A frequency-based maritime container assignment model. Transport. Res. Part B 45 (8), 1152–1161.
Bell, M.G.H., Liu, X., Rioult, J., Angeloudis, P., 2013. A cost-based maritime container assignment model. Transp. Res. Part B 58, 58–70.
Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. Eur. J. Oper. Res. 202 (3), 615–627.
Bierwirth, C., Meisel, F., 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. Eur. J. Oper. Res. 244 (3), 675–689.
Cao, J.X., Lee, D.H., Chen, J.H., Shi, Q.X., 2010. The integrated yard truck and yard crane scheduling: Benders' decomposition-based methods. Transp. Res. Part E 46 (3), 344–353.
Chen, L, Bostel, N., Dejax, P., Cai, J.G., Xi, L.F., 2007. A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. Eur. J. Oper. Res. 181, 40–58.
Chen, L., Langevin, A., Lu, Z.Q., 2013. Integrated scheduling of crane handling and truck transportation in a maritime container terminal. Eur. J. Oper. Res. 225 (1), 142–152.
Choo, S., Klabjan, D., Simchi-Levi, D., 2010. Multiship crane sequencing with yard congestion constraints. Transp. Sci. 44 (1), 98–115.
Coello, C.A.C., Cortés, N.C., 2005. Solving multiobjective optimization problems using an artificial immune system. Genet. Program. & Evolv. Mach. 6 (2), 163–190.
Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (2), 182–197.
Giallombardo, G., Moccia, L., Salani, M., Vacca, I., 2010. Modeling and solving the tactical berth allocation problem. Transp. Res. Part B 44 (2), 232–245.
Hendriks, M.P.M., Lefeber, E., Udding, J.T., 2013. Simultaneous berth allocation and yard planning at tactical level. OR Spectr. 35 (2), 501–517.
Imai, A., Yamakawa, Y., Huang, K., 2014. The strategic berth template problem. Transp. Res. Part E 2014, 77–100.
Iris, C., Pacino, D., Ropke, S., Larsen, A., 2015. Integrated berth allocation and quay crane assignment problem: set partitioning models and computational results. Transp. Res. Part E 81, 75–97.
Jin, J.G., Lee, D.H., Hu, H., 2015. Tactical berth and yard template design at container transshipment terminals: a column generation based approach. Transp. Res. Part E 73, 168–184.
Lee, C.Y., Liu, M., Chu, C.B., 2014. Optimal algorithm for general quay crane double-cycling problem. Transp. Sci. 49 (4), 957–967.
Lee, C.Y., Meng, Q., 2015. Handbook of ocean container transport logistics: making global supply chains effective. In: International Series in Operations Research & Mangagement Science, vol. 220. Springer, p. 2015.

Lee, D.H., Jin, J.G., 2013. Feeder vessel management at container transshipment terminals. Transp. Res. Part E 49, 201–216.

Li, L., Negenborn, R.R., De Schutter, B., 2015. Intermodal freight transport planning - a receding horizon control approach. Transp. Res. Part C 60, 77–95.

Liu, Z., Meng, Q., Wang, S., Sun, Z., 2014. Global intermodal liner shipping network design. Transp. Res. Part E 61, 28–39.

Meisel, F., Bierwirth, C., 2013. A framework for integrated berth allocation and crane operations planning in seaport container terminals. Transp. Sci. 47 (2), 131–147.

Meng, Q., Wang, S., Lee, C.Y., 2015. A tailored branch-and-price approach for a joint tramp ship routing and bunkering problem. Transp. Res. Part B 72, 1–19.

Moorthy, R., Teo, C.P., 2006. Berth management in container terminal: the template design problem. OR Spectr. 28 (4), 495–518.

Nabais, J.L., Negenborn, R.R., Botto, M.A., 2012. A novel predictive control based framework for optimizing intermodal container terminal operations. In: Proceedings of the 3rd International Conference on Computational Logistics (ICCL 2012), Shanghai, China, pp. 53–71.

Park, Y.M., Kim, K.H., 2003. A scheduling method for berth and quay cranes. OR Spectr. 25 (1), 1–13.

Port Technology, 2015. Low berth productivity costs shipping billions, 21 Dec 2015. https://www.porttechnology.org/news.

Stahlbock, R., Voß, S., 2008. Operations research at container terminals: a literature update. OR Spectr. 30, 1–52.

Tao, Y., Lee, C.Y., 2015. Integrated planning of berth and yard allocation in transshipment terminals using multi-cluster stacking strategy. Transp. Res. Part E 83, 34–50.

Turkogullari, Y.B., Caner, T.Z., Aras, N., Kuban, A.I., 2014. Optimal berth allocation and time-invariant quay crane assignment in container terminals. Eur. J. Oper. Res. 235 (1), 88–101.

UNCTAD, 2015. Review of maritime transport: 2014. ISBN 9789211128109.

Vacca, I., Salani, M., Bierlaire, M., 2013. An exact algorithm for the integrated planning of berth allocation and quay crane assignment. Transp. Sci. 47 (2), 148–161.

Wang, S., Liu, Z., Bell, M.G.H., 2015. Profit-based maritime container assignment models for liner shipping networks. Transp. Res. Part B 72, 59–76.

Xin, J., Negenborn, R.R., Corman, F., Lodewijks, G., 2015. Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance. Transp. Res. Part C 60, 377–396.

Zhen, L., 2016. Modeling of yard congestion and optimization of yard template in container ports. Transp. Res. Part B 90, 83–104.

Zhen, L., Chew, E.P., Lee, L.H., 2011. An integrated model for berth template and yard template planning in transshipment hub. Transp. Sci. 45 (4), 483–504.

Zhen, L., Xu, Z., Wang, K., Ding, Y., 2016. Multi-period yard template planning in container terminals. Transp. Res. Part B doi:10.1016/j.trb.2015.12.006.

Zitzler, E., Deb, K., Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. 8 (2), 173–195.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G., 2003. Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. 7 (2), 117–132.