Scheduling quay cranes and yard trucks for unloading operations in container ports

Lu Zhen[1], Shucheng Yu[1], Shuaian Wang[2*], Zhuo Sun[3]

[1] School of Management, Shanghai University, Shang Da Road 99, Shanghai 200444, China
[2] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
[3] Transportation Management College, Dalian Maritime University, Dalian 116026, China.

Abstract: This paper studies an integrated optimization problem on quay crane and yard truck scheduling in container terminals. A mixed-integer programming model is formulated. For the model, we show the integrated scheduling problem is strongly NP-hard and investigate some properties that can considerably reduce the computational complexity. For solving the proposed model within a reasonable time, a particle swarm optimization based solution method is developed. Numerical experiments are conducted to compare the proposed method with the CPLEX solver and the genetic algorithm. The results validate the effectiveness of the proposed model and the efficiency of the proposed solution method.

Keywords: OR in transportation; scheduling; container port operation; quay cranes; yard trucks.

1. Introduction

During the past decades, the maritime container transportation has developed rapidly. Since 2001 the annual growth rate of the world container port throughput has been up to 10% on average. Some forecasts suggest this trend will continue till 2020. In addition, size of vessels becomes larger and larger. Some mega-vessels can carry up to 19,000 TEUs (Twenty-foot Equivalent Units). Thus ports need to handle a larger number of containers during a limited length of time than before. More and more challenges are brought to port operators for improving the efficiency in their management of various resources and to decrease ship turnaround times in ports, especially some mega-ports such as Shanghai, Singapore, Shenzhen and Hong Kong.

[*] Corresponding author. Email: wangshuaian@gmail.com.

The container terminal operations can be decomposed into several types of sub-problems, such as berth allocation, quay crane (QC) scheduling, yard truck (YT) scheduling, yard crane (YC) scheduling and storage allocation. In fact, these sub-problems are tightly interconnected. Optimizing only one type of sub-problems may not be an overall optimal operation. Therefore, this study combines the QC scheduling problem and the YT scheduling problem as a whole. From an operational point of view, the above two scheduling problems are intertwined. The QC scheduling problem mainly determines the assignment of QCs to ships and the sequence of tasks to be processed by each QC, respectively. The YT scheduling problem is mainly about the transportation of containers between the QC side and the YC side, which significantly affects the efficiency of the whole container terminal operations. The total make-span is highly influenced by the synchronization of QC scheduling and YT scheduling.

This paper studies an integrated optimization problem on QCs and YTs synchronization scheduling in container terminals, considering the coordination of the QCs scheduling and YTs scheduling problem to reduce the idle time between performing two successive tasks. The QCs are in charge of the process of loading and unloading containers; and the YTs are in charge of the process of transporting these containers between quay side and yard side. A good integrated optimization that combines the above two handling processes can avoid efficiency loss due to waiting for each other. The contribution of this study mainly lies in consideration of the following aspects: (1) determining each container's throughput time in the QC scheduling stage that considers precedence, temporal distance between adjacent QCs, non-crossing constraints and some other realistic constraints; (2) the decision of routing and allocation problems about YTs; (3) jointly scheduling the QCs handing, routing and allocation of YTs processes. Additionally, some properties on parameters setting and symmetry issue are also discussed. Then these properties are used to reduce the problem's computational complexity and improve the efficiency of the solving process.

The remainder of this paper is structured as follows: Section 2 reviews the related works. Section 3 addresses the background of the integrated optimization problem, a mixed-integer programming (MIP) model is proposed and meanwhile some propositions are discussed. Section 4 elaborates a particle swarm optimization (PSO) based solution method. Section 5 presents the results of the computational experiments that compare the results obtained by the PSO with the optimal results solved by the CPLEX directly as well as the results obtained by the genetic algorithm (GA). Finally, conclusions and closing remarks are summarized in the last section.

2. Literature review

During the last two decades, the QC scheduling problem has received much attention from academia and practitioners. The QC scheduling problem was first discussed by Daganzo (1989). He proposed two algorithms to minimize all the ships' aggregate cost of delay. Peterkofsky and Daganzo (1990) extended the previous work to a feasible problem, which is allowed to solve larger instances and multiple machines working simultaneously on a single task. An improved branch and bound (B&B) method was proposed. Recently, the complete bay is considered. Liu et al. (2006) proposed a mixed-integer programming (MIP) model to solve a large-scale QC scheduling problem in container terminals, where inbound vessels have different ready times. They divided the proposed model into two sub-models, one is a vessel related model and the other is a berth related model. Two heuristic algorithms were developed to solve the two sub-models. Kim and Park (2004) discussed the QC scheduling problem. A MIP model, which considers various related constraints, was formulated. In that paper a greedy randomized adaptive search procedure (GRASP) was proposed to overcome the computational difficulty of the branch and bound (B&B) method. Tavakkoli-Moghaddam et al. (2009) developed a MIP model for QC scheduling and assignment problem, namely QCSAP. A genetic algorithm (GA) is used to solve the above-mentioned QCSAP for real-world instances. Lim et al. (2004) extended the QC scheduling problem with three spatial and separation constraints: the non-crossing constraint, the neighborhood constraint and the job-separation constraint. They provided three algorithms to find an optimal crane-to-job matching, which could maximize throughput under these constraints. Moreover, a squeaky wheel optimization method with a local search approach was proposed to search high-quality results within short times. Guan et al. (2013) studied the QC scheduling problem for vessels mooring in a terminal. They firstly developed a time-space network flow formulation with non-crossing constraints for the problem. Then the Lagrangian relaxation approach and two heuristics that based on the threshold policy and the worst case bound policy are developed to solve the problem. Legato et al. (2012) considered the unidirectional scheduling issue with a series of practical constraints, such as ready times, due dates for QCs, precedence relations among container groups and crane-individual service rates. They used a B&B scheme and a novel timed Petri net approach in the algorithm design. Numerical experiments showed their proposed method obtained high-quality solutions.

The YT scheduling problem also plays an important role in container terminals. In reality, YT acts as a bridge between QC side and YC side. The efficiency of YT schedule has a significant influence on the performance of the whole container terminal operations. Kim and Kim (1999) discussed a routing problem of straddle carriers during loading operations on export containers. The routing problem was formulated as an integer programming model. Bish (2003) studied a vehicle dispatching problem with considering the QC related activities. The objective is to minimize the maximum time it takes to serve a given set of ships. Bish et al. (2005) extended the vehicle dispatching issue in a mega container terminal in order to minimize the total time it takes to serve a ship. Moreover, some easily implementable heuristic algorithms were developed in this paper. Nguyen and Kim (2009) studied the automated lifting vehicle (ALV) scheduling problem. A heuristic algorithm was developed to solve the model. The effect of dual cycle operation was also analyzed in this paper. Hu et al. (2013) proposed a new automated container terminal (ACT) system which utilizes multistory frame bridges and rail-mounted to transport containers between the quay side and yard side. This new system is different from the traditional equipment that uses YT and automated guided vehicles (AGV). Recently the fuel consumption and emission issue attracted researchers' interest intensively. Du et al. (2011) proposed an elaborate model on berth allocation with considering fuel consumption and emission. Nielsen et al. (2015) took the Singapore port as a case study to consider the emission reduction issue of truck engines in ports. Their study suggested that port operators should schedule YTs' arriving time with an optimal way in order to reduce the YTs' idling time and emotions. Additionally, Li (2012), Wang (2013), Liu (2016) considered the ship and truck routing and scheduling problem.

The scheduling decision of port resources (such as QCs, YTs, YCs) are usually intertwined (Pang et al., 2011; Talley and Ng, 2013; Tran and Haasis, 2015). However, it is a challenging task to propose an integrated model on optimizing some of these intertwined resources. Even the most advanced computer may not easily solve a model that integrates all the resources applied in a large port such as Shanghai port. In recent years, some studies have integrated parts of these terminal operations. For example, Chen et al. (2013) studied the interaction between QC handling and YT transportation in a container terminal. They proposed a three-stage algorithm to solve the YT scheduling problem. In their model a YT can be shared among different ships. This sharing policy reduces empty YT trips in container terminals. He et al. (2015) also considered the integrated scheduling problem that combined the QC scheduling, internal truck scheduling and YC scheduling as a whole. In that paper a GA algorithm was used for global search and a PSO algorithm was used for local search to optimize the total departure delay of all vessels and the total transportation energy consumption of all tasks. Cao et al. (2010) developed a novel integrated model for YT and YC scheduling. A general Benders decomposition based method and a combinatorial Benders decomposition based method were designed to solve the model. Jin et al. (2015) proposed an integrated model to solve the berths and yard spaces problem. Jiang et al. (2012) gave us a framework strategy that integrated space reservation and workload assignment as a whole. Kaveshgar and Huynh (2015) proposed a new mathematical model to reduce vessels' turn time. This paper combines the synergies between QC, YT and YC, which captures the essential characteristic of marine container terminals. Tang et al. (2014) studied a joint QC and YT scheduling problem in container terminals. They considered the coordination of the two types of equipment to reduce the idle time both in the unidirectional and bidirectional flow situations. An improved particle swarm optimization (PSO) algorithm and several valid inequalities were proposed to solve the joint scheduling problem. There are also heuristic approaches used to solve the port resource (such as QCs, YTs, YCs) scheduling problems. The most popular meta-heuristics are GA and PSO algorithm. For example, Bruzzone and Signorile (1998), Tavakkoli-Moghaddam et al. (2009), Yu et al. (2011), Nguyen et al. (2013), and Fu and Tsai (2014) proposed some GA based solution methods in the port related scheduling problems. For the PSO algorithm applied in the port related scheduling problems, we can refer to Wang et al. (2012) , Guo et al. (2014), Yao et al. (2014), Han et al. (2015), etc.

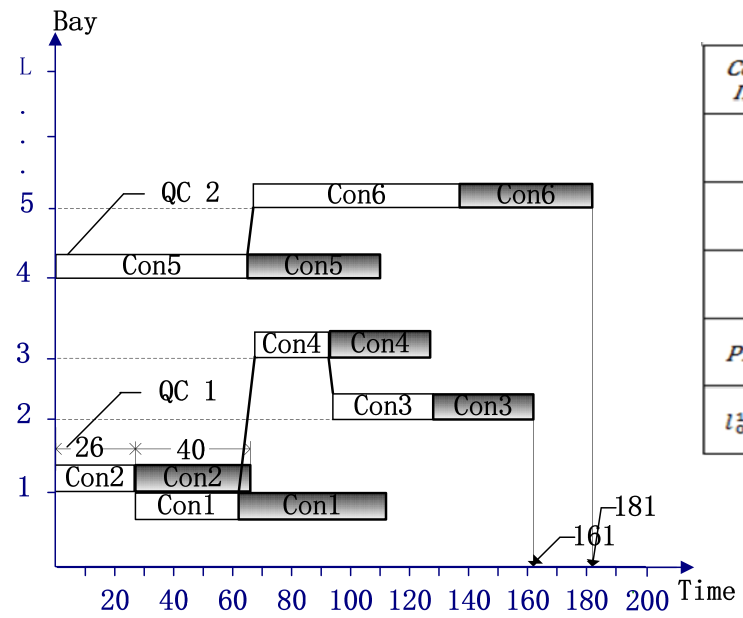
This study developed a mathematical model that is based on the integrated scheduling on QCs and YTs in container terminals. It extends the work of Tang et al. (2014) by enhancing the QC interference constraints. Meanwhile, we improve the work of Kaveshgar and Huynh (2015) by considering the issue that containers in the same bay should be unloaded by the same QC. This study also considers some realistic factors such as QC interference, safety margin, and a sufficiently large temporal distance between the processing of adjacent QCs. We also prove that the integrated scheduling problem is strongly NP-hard. Based on the complexity of the research problem, two proved properties are proposed to reduce the computational complexity and a PSO based solution method is developed.

3. Problem description and model formulation

3.1 Problem description

This study supposes the vessels have L bays, and N tasks need to be handled. There are $|Q|$ QCs and $|T|$ YTs serving the vessels. The $|T|$ YTs are homogenous with respect to their capacity. Figure 1 provides an example with two QCs and four YTs unloading six containers to illustrate the integrated scheduling problem. The corresponding values of parameters are shown in the right table. P_{i1} denotes QC processing time of container i, P_{i2} denotes the round transportation time of container i, l_0^q is the initial position of QC q, f_q is the earliest available time of QC q, g is the QC's unit travel time per bay. The precedence constraint is that container 2 must be handled before container 1.

As shown in Figure 1, we can observe the optimal solution is containers 1, 2, 3, 4 are allocated to QC 1 and the handling sequence is 2→1→4→3; containers 5 and 6 are allocated to QC 2 and the handling sequence is 5→6. We can also see the YTs scheduling plan through the vertical axis direction, e.g., when the time is 100 there are three YTs that are working, which is less than the number of available YTs in the example (i.e., four). By manual calculation, if we evenly distribute the workload (i.e., assigning the first three tasks and two YTs to QC 1, the remaining three task and two YTs to QC 2), the total processing time will be 206, which is much longer than the optimal result shown in Figure 1 (i.e., 181).



Container Index i	*1*	*2*	*3*	*4*	*5*	*6*
P_{i1}	35	26	44	24	65	70
P_{i2}	50	40	30	40	50	45
bay_i	1	1	2	3	4	5
Precedence constraints: $\Phi = \{(2,1)\}$						
$l_0^1 = 1, l_0^2 = 4,\ g = 1, f_1 = f_2 = 0$						

Con i Container i

Pickup time of QC

Transport time of YT

Figure 1: Illustration of QCs and YTs schedule

We define the index bay 1 as the leftmost bay and the bay L as the rightmost bay. Tasks in the same bay must considered the precedence constraint. In our proposed scheduling model, we are given a set of tasks $\Omega = \{1, \cdots, N\}$, a set of parallel identical QCs, $Q = \{1,2, \cdots, |Q|\}$ and a set of YTs, $T = \{1,2, \cdots, |T|\}$. At each stage each task has a certain processing time and the processing time of task i in stage $t (t = 1,2)$ is expressed as p_{it}.

In the previous literature, the objective function of the integrated QC and YT scheduling problem is to minimize the make-span of the two sub-operations, i.e., minimize the sum of the

QC processing time and YT transportation time. However, the processing time of QCs is more important than the transportation time of YTs in reality because the QCs are usually the bottleneck for the port's productivity. Therefore, this study uses a weighted sum of the QC processing time and YT transportation time; and we set the weight of the former one to be larger than the weight of the latter one.

3.2 Model formulation

***Index and sets*:**

i, j	tasks/containers index
Ω	set of tasks/containers to be performed, $\Omega = \{1, \cdots, N\}$; two dummy tasks 0 and F with zero processing time are given to indicate the beginning and the end of whole schedule. Additionally, $\Omega^0 = \Omega \cup \{0\}$, $\Omega^F = \Omega \cup \{F\}$, $\bar{\Omega} = \Omega \cup \{0, F\}$
h	bay index
E	set of bays of the ship to be berthed; $E = \{1,2, \cdots, \|E\|\}$, the bays are numbered sequentially along the quay in the same direction as for the QCs
q	QC index
Q	set of QCs; $Q = \{1,2, \cdots, \|Q\|\}$
k	YT index
T	set of YTs, $T = \{1,2, \cdots, \|T\|\}$
Φ	set of task/container pairs (i, j) in the same bay that i must precede j in unloading process
Ψ	set of all task pairs that cannot be performed simultaneously with considering the safety of QC margin constraint and task precedence constraint, $\Phi \subseteq \Psi$

Input data:

λ	a constant weight that belongs to (0,1) interval
f_q	earliest available time of QC q

g QC's unit travel time between bays

l_0^q initial position (the position is pressed using the bay number) of QC q

L_i bay in which container i is located, $L_i \in E$

M a sufficiently large positive number

P_{j1} QC processing time needed to unload task/container j

P_{j2} time for a YT to transport container j from the QC side to the container's destination in the yard and return

t_{ij} travel time of QC from the bay position of task i to task j

Δ_{ij}^{vs} minimum time span to elapse between task i and task j, if the two tasks processed by QC v and s, respectively

Θ set of all combinations of tasks and QCs that potentially lead to QC interference. It is defined as $\Theta = \{(i,j,v,s) \in \Omega^2 \times Q^2 | i < j \ \ and \ \ \Delta_{ij}^{vs} > 0\}$

Decision variables:

W_{ij} binary variable. It equals 1 if task/container j starts after the completion of task/container i by QC, 0 otherwise

X_{ij}^q binary variable. It equals 1 if QC q performs task/container i just before task/container j, 0 otherwise

Y_{ik} binary variable. It equals 1 if container i is allocated to YT k for transporting, 0 otherwise

Y_{ij}^k binary variable. It equals 1 if YT k performs container i just before container j, 0 otherwise

Z_{iq} binary variable. It equals 1 if task/container i allocated to QC q for unloading, 0 otherwise

C_{i1} complete time of unloading task/container i

C_{i2} complete time of the round trip for transporting container i

Based on the above notations, we formulate the problem as the following mathematical model:

$$Minimize\ max_{i\in\Omega}(C_{i1}+\lambda C_{i2}) \qquad (1)$$

subject to:

$$\sum_{j\in\Omega^F} X_{0j}^{q} = 1 \qquad q\in Q \qquad (2)$$

$$\sum_{j\in\Omega^0} X_{jF}^{q} = 1 \qquad q\in Q \qquad (3)$$

$$\sum_{j\in\Omega^0} X_{ji}^{q} - \sum_{j\in\Omega^F} X_{ij}^{q} = 0 \qquad i\in\Omega, q\in Q \qquad (4)$$

$$Z_{iq} = \sum_{j\in\Omega^0} X_{ji}^{q} \qquad i\in\Omega, q\in Q \qquad (5)$$

$$\sum_{q\in Q} Z_{iq} = 1 \qquad i\in\Omega \qquad (6)$$

$$C_{i1} \le C_{j1} - P_{j1} \qquad (i,j)\in\Phi \qquad (7)$$

$$W_{ij} + W_{ji} = 1 \qquad (i,j)\in\psi \qquad (8)$$

$$C_{i1} + P_{j1} - C_{j1} \le M \times (1 - W_{ij}) \qquad i,j\in\Omega \qquad (9)$$

$$C_{j1} - P_{j1} - C_{i1} \le M \times W_{ij} \qquad i,j\in\Omega \qquad (10)$$

$$\sum_{u\in\Omega^0} X_{ui}^{v} + \sum_{u\in\Omega^0} X_{uj}^{s} \le 1 + W_{ij} + W_{ji} \qquad (i,j,v,s)\in\Theta \qquad (11)$$

$$C_{i1} + \Delta_{ij}^{vs} + P_{j1} - C_{j1} \le M \times \left(3 - W_{ij} - \sum_{u\in\Omega^0} X_{ui}^{v} - \sum_{u\in\Omega^0} X_{uj}^{s}\right) \qquad (i,j,v,s)\in\Theta \qquad (12)$$

$$C_{j1} + \Delta_{ji}^{vs} + P_{i1} - C_{i1} \le M \times \left(3 - W_{ji} - \sum_{u\in\Omega^0} X_{ui}^{v} - \sum_{u\in\Omega^0} X_{uj}^{s}\right) \qquad (i,j,v,s)\in\Theta \qquad (13)$$

$$f_q + g \times \left|l_0^{q} - L_j\right| + P_{j1} - C_{j1} \le M \times (1 - X_{0j}^{q}) \qquad j\in\Omega, q\in Q \qquad (14)$$

$$C_{i1} + g \times |L_i - L_j| + P_{j1} - C_{j1} \leq M \times (1 - X_{ij}^q) \qquad i,j \in \Omega, q \in Q \qquad (15)$$

$$C_{i1} + P_{i2} = C_{i2} \qquad i \in \Omega$$

(16)

$$\sum_{j \in \Omega^F} Y_{0j}^k = 1 \qquad k \in T$$

(17)

$$\sum_{j \in \Omega^0} Y_{jF}^k = 1 \qquad k \in T$$

(18)

$$\sum_{j \in \Omega^0} Y_{ji}^k - \sum_{j \in \Omega^F} Y_{ij}^k = 0 \qquad i \in \Omega, k \in T \qquad (19)$$

$$Y_{ik} = \sum_{j \in \Omega^0} Y_{ji}^k \qquad i \in \Omega, k \in T$$

(20)

$$\sum_{k \in T} Y_{ik} = 1 \qquad i \in \Omega$$

(21)

$$C_{j1} + M(1 - Y_{ij}^k) \geq C_{i2} \qquad i,j \in \Omega, k \in T \qquad (22)$$

$$W_{ij} \in \{0,1\}, W_{ii} = 0 \qquad i,j \in \Omega$$

(23)

$$X_{ij}^q \ , Y_{ij}^k \in \{0,1\} \qquad i,j \in \bar{\Omega}, q \in Q, k \in T$$

(24)

$$X_{ii}^q = 0 \ , Y_{ii}^k = 0 \qquad i \in \bar{\Omega}, q \in Q, k \in T$$

(25)

Objective (1) is to minimize the combined processing time of the two sub-operations. Detail reasons are described in section 3.1. The parameter λ is a fractional constant weight between zero and one. Constraints (2) and (3) guarantee each QC have a dummy task 0 as its initial task and a dummy task F as its final task. Constraints (4) ensure the operation order of the tasks. Each task (except the two dummy tasks) has only one task as its immediate predecessor and one task as its immediate follower. Constraints (5) make sure if a container is unloaded by a QC, it should be the initial task or there is another task before it. Constraints (6) require that each task must be handled by exactly one QC. Constraints (7) guarantee that if task i and j belong to the set $\Phi(i,j)$ in the same bay, task i must precede task j in the unloading process. Constraints (8) ensure if task i and j belong to the set $\psi(i,j)$, the two tasks cannot be performed simultaneously. Constraints (9) and Constraints (10) define the binary variable W_{ij}. W_{ij} is equal to 1 in the case that task j must be handled after the operation of task i and W_{ij} is equal to 0 in the case that task i started after the completion of task j. Constraints (11)-(13) consider the non-crossing constraints and safety margin constraints, when QCs v and s are adjacent and their processing tasks are i and j, respectively. Constraints (11) guarantee that task i and task j are not handled simultaneously by QCs v and s, respectively. If both assignments i and j take place in that way, the left side value is two that is greater than the right side value one. In other words, if QCs v and s are adjacent, either $W_{ij} = 1$ or $W_{ji} = 1$. Constraints (12) prescribe a proper temporal distance calculated by equation (27) which explained later. It applies to the starting time of task i and the completion time of task j if $W_{ij} = 1$. Similarly, Constraints (13) work in the same manner as constraints (12) but for the case $W_{ji} = 1$. Constraints (14) use the processing time of the first task to define the earliest starting operation time of each QC. Constraints (15) define the completion time of each task handled by QC, which equals the completion time of the adjacent tasks (the two tasks handled by the same QC) plus the QC travel time between these two tasks as well as their processing time. Constraints (16) define the relationship between the completion time of unloading and the completion time of transportation. Constraints (16) connect the two types of activities by QC and YT. Constraints (17)-(19) have the same function for YTs as the Constraints (2)-(4) for QCs. Constraints (20) ensure that if container i is transported by a YT, it is either the initial task or there is another task transported just before it. Constraints (21) require that each container should be transported by one YT and only one YT. Constraints (22) ensure that if both container i and j are transported by the same YT and container i is handled first, the completion time of task j

handled by QC should be no earlier than the completion time of task i. Note that in order to understand Constraints (22) one should connect with Constraints (16). Constraints (23)-(25) define the decision variables.

Objective (1) is hard to solve as we should optimize the $Minimize\ max_{i\in\Omega}(C_{i1}+\lambda C_{i2})$, i.e., we first optimize $Max(C_{i1}+\lambda C_{i2}), i\in\Omega$, and then optimize the minimization of $Max(C_{i1}+\lambda C_{i2}), i\in\Omega$. To address the difficulty, we formulate an auxiliary decision variable Λ as an intermediate variable. The auxiliary decision variable Λ is defined as an upper bound of $max_{i\in\Omega}(C_{i1}+\lambda C_{i2})$, i.e.

$$max_{i\in\Omega}(C_{i1}+\lambda C_{i2}) \le \Lambda \tag{26}$$

Then the objective (1) turns to *min* Λ, subjects to constraints: $\Lambda \ge (C_{i1}+\lambda C_{i2}), i\in\Omega$

The parameter Δ_{ij}^{vs} is defined as the minimum temporal distance. It is inserted into the processing time slots of two tasks i and j, which handled by QCs v and s, respectively. The definition of Δ_{ij}^{vs} was proposed by Bierwirth and Meisel (2009) as follows:

$$\Delta_{ij}^{vs}=\begin{cases}(l_i-l_j+\delta_{vs})\cdot g & if\ \ v<s, i\ne j, l_i>l_j-\delta_{vs}\\ (l_j-l_i+\delta_{vs})\cdot g & if\ \ v>s, i\ne j, l_i<l_j+\delta_{vs}\\ 0 & otherwise\end{cases}$$

(27)

Where l_i is the bay position of taks i, the parameter δ_{vs} is the safety margin that must be maintained between the adjacent QCs v and s, $\delta_{vs}=(\delta+1)\cdot|v-s|$ (δ is a contant, example the value of δ can be 2). The set Θ is defined as follows

$$\Theta=\{(i,j,v,s)\epsilon\Omega^2\times Q^2 | i<j, \Delta_{ij}^{vs}>0\} \tag{28}$$

The set Θ defines the combination that the QCs and tasks may lead to potential interference.

Proposition 1: Finding an optimal solution for the model is strongly NP-hard.

Proof: See Appendix A ■

In the traditional model formulation, M is set as a sufficiently large positive number, such as 10,000 or even larger. In the process of model solving, the value of M influences the efficiency of the solving process. Li et al. (1996) and Junior and Lins (2005) have investigated the issue of setting the value of M, and found that a proper value of M could not only reduce the difficulty of understanding the problem but also improve the computational efficiency. Moreover, different problem backgrounds may need different methods to set a proper M value. In this study, we focus on this issue and try to set a proper upper bound value of M for different scale problems. Specifically, we assume there is only one QC (the first QC in the set) and one YT to handle all the tasks. The QC moves from one side to the next, the containers are labeled such that container 1 can be unloaded first, container 2 can be unloaded second, etc. Then we can calculate the time required to complete all of the tasks. So a valid value for M can be equal to the above time. We can see that for different-scale problems the M value is not the same. A proposition about this is stated as follows:

Proposition 2: A formula used to calculate a proper value of M about the integrated QC and YT scheduling problem is as follows:

$$M^{UB} = f_q + g \times |l_0^q - L_1| + P_{11} + \sum_{i \in \Omega \backslash \{1\}} max\{P_{i-1,2},\ g \times |L_i - L_{i-1}| + P_{i1}\} + P_{|\Omega|2} \quad (29)$$

Proof: See Appendix B ■

In realistic port operation, a port operator usually assigns a given set of YTs to each QC. However, it usually will lead to QCs and YTs wait for each other, because these two types of machines lack coordination. Such as if a container is handled by a QC and all the YTs allocated to the QC are occupied, the QC has to hold the container till an empty YT coming. On the contrary, YT sometime needs to wait for the QC that the YT assigned. These waiting activities on the sides of QC and YT reduce the handling efficiency of the port operation. For example, there are two QCs allocated to bay 4 and bay 6. YTs 1, 2 and 3 are allocated to QC 1, and YTs 4, 5 and 6 are allocated to QC 2. The YT 1 is fully loaded, while YTs 2 and 3 are waiting in a queue for QC 1. Meanwhile, YTs 4, 5 and 6 are busy. According to the usual practice, QC 2 needs to hold container and wait for YTs. Based on the above problems, this study considers the YTs sharing among QCs. Then a YT may be dispatched from bay 4 to bay 6. However, as these YTs are homogeneous, dispatching YT 2 to bay 6 has the same effect as dispatching YT 3 to bay

6. In fact, their effects are the same. We call this phenomenon a symmetry problem. When we use mixed-integer linear programming software to solve the scheduling problem, the solver may not distinguish the symmetry phenomenon. A lot of solved solutions have the same effect and the symmetry problem may lead to a time-consuming solving process.

Proposition 3: To reduce the influence of symmetry phenomenon, we propose the following method as follows:

$$\sum_{j\in\Omega^F} j.Y_{0j}^{k} \leq \sum_{j\in\Omega^F} j.Y_{0j}^{k+1} \qquad k \in T/\{|T|\} \qquad (30)$$

Proof: See Appendix C ■

4. Solution method

For small-scale problem instances, the proposed model can be solved directly by the CPLEX solver. However, the CPLEX solver cannot solve large-scale problem instances within a reasonable period of time. Therefore, we have to consider heuristic algorithms. Particle Swarm Optimization (PSO) was first introduced by Eberhart and Kennedy (1995). Compared with the traditional GA algorithm, the PSO algorithm needs fewer parameters to adjust. Nowadays the PSO algorithm has been widely used for solving port optimization problems. Wang et al. (2012) used an improved discrete PSO algorithm to solve a QC scheduling problem in order to reduce the turnaround time of a ship. Ting et al. (2014) used it to solve a berth allocation problem. Guo et al. (2014) also applied the PSO algorithm to optimize a QC scheduling problem. These results demonstrate the PSO algorithm can effectively solve port optimization problems. Thus this study designs a PSO based solution method for solving the integrated QC and YT scheduling problem.

The PSO algorithm is a population-based algorithm that is initialized with a group of random particles. Each particle represents a solution method that has a given position and velocity. The particles search for optimal solutions through updating generations. In each generation, each particle is updated by a new position and velocity. The position reflects the search quality, and velocity determines the direction where the particle would move in the next iteration. The updating formula of velocity and position are presented as follows:

$$v_{mi}^{n+1} = v_{mi}^{n} + c_1 rnd_1 (lpBest_{mi}^{n} - l_{mi}^{n}) + c_2 rnd_2 (lgBest_{m}^{n} - l_{mi}^{n}) \tag{31}$$

$$l_{mi}^{n+1} = l_{mi}^{n} + v_{mi}^{n} \tag{32}$$

In Eqs. (31) and (32), v_{mi}^{n+1} and v_{mi}^{n} represent the current velocity and the previous velocity of particle i on dimension m, respectively; $lpBest_{mi}^{n}$ denotes the best position of particle i on dimension m up to iteration n; $lgBest_{m}^{n}$ denotes the best position of the whole swarm on dimension m until iteration n; l_{mi}^{n+1} and l_{mi}^{n} denote the current and previous position of particle i on dimension m, respectively. Both c_1 and c_2 are acceleration weights, and they determine whether the particle fly to the best position it has reached so far or the best position of the whole swarm. rnd_1 and rnd_2 are two positive random numbers generated within the interval [0, 1].

The standard PSO algorithm may lead the particles to grow unlimitedly, which influences the particles' convergence to the optimal solution. To overcome this problem Shi and Eberhart (1998) proposed a new velocity updating formula which imposes the velocity of particle i an inertia coefficient. The updating formula is as follows:

$$v_{mi}^{n+1} = w^{n} v_{mi}^{n} + c_1 rnd_1 (lpBest_{mi}^{n} - l_{mi}^{n}) + c_2 rnd_2 (lgBest_{m}^{n} - l_{mi}^{n}) \tag{33}$$

In Eq. (33), the inertia coefficient w^{n} is expressed as:

$$w^{n} = \frac{\gamma_{max-n}}{\gamma_{max}} (\beta_{max} - \beta_{min}) + \beta_{min} \tag{34}$$

In Eq. (34), β_{max} and β_{min} represent the maximum and the minimum values of the inertia weight coefficients, respectively. γ_{max} is the maximum number of iteration.

4.1. Solution representation

In this study, the integrated optimization model solves two problems: one is to allocate the tasks to QC and YT, the other is to identify the handling sequence of QC and YT. Thus, the first stage is to find a suitable mapping between the two decision problems and particles. We encode the particles in terms of the QC allocation and the handling sequence of containers, and then decode them to get solutions. Let a $2N$-dimensional vector ($\boldsymbol{X} = \{X_1, \cdots, X_i, \cdots, X_{2N}\}$) represent a solution ($N = |\Omega|$). The first N dimensions in the front part of the vector indicate the QC allocation. And the value of each dimension is a real number (i.e., X_i) within the interval $[0,|Q|]$, where $|Q|$ is the number of QCs. The value $\lceil X_i \rceil$ denotes the index of the QC allocated for Task i. Here $\lceil X_i \rceil$ is the smallest upper integer bound of X_i. The remaining N dimensions in the vector represent the handling sequence of each QC. The value in the dimension $N+i$ is also a real number (i.e., X_{N+i}) within the interval $[0,N]$, and X_{N+i} determines the unloading sequence of container i. The unloading sequence for a certain QC is decided by the ascending order of the values in those dimensions corresponding to the containers allocated to the QC. It should be mentioned that we allocate containers to YT in terms of the ascending order of the QC scheduling sequence. Here we give an example to illustrate it. The example encoding of the solution contains two QCs and five tasks. Table 1 shows the encoding results of this instance.

Table 1: The encoding of the solution

Task No.	X_i	X_{i+N}
1	0.32	2.77
2	0.65	5.99
3	1.40	4.99
4	1.71	2.41
5	1.36	3.87

In Table 1， $X_1 = 0.32 < X_2 = 0.65 < 1, \lceil X_1 \rceil = \lceil X_2 \rceil = 1$. So Task 1 and Task 2 are handled by QC 1. Task 2 is handled later than Task 1, because $X_{2+N} = 5.99 > X_{1+N} = 2.77$. $\lceil X_3 \rceil = \lceil X_4 \rceil = \lceil X_5 \rceil = 2$, so Task 3, Task 4 and Task 5 are handled by QC 2. As for $X_{4+N} = 2.41 < X_{5+N} = 3.87 < X_{3+N} = 4.99$, and the handling sequence is Task 4, Task 5 and Task 3. The corresponding solution after decoding is in Table 2.

Table 2: The decoding of the solution

Task No.	Quay crane No.	Sequence
1	1	1
2	1	2

3	2	3
4	2	1
5	2	2

4.2 The PSO procedure

Based on the above components, the completed PSO procedure for solving the integrated QC and YT scheduling problem is as follows.

Step 1: Initialize K particles as a swarm to obtain the QC allocation and the handling sequence by each QC. Set iteration n =1.

Step 2: For i =*1, 2,* $\cdots$,K, update particles with the probability ε, re-initialize the particles and their best positions ($pBest$).

Step 3: For i =*1, 2,* $\cdots$,K, ensure that containers in the same bay must be handled by the same QC and precedence constraints must be satisfied in the initial solutions. A sub-procedure $\boldsymbol{Adjust}(m)$ is performed to revise the initial particles. Details of the sub-procedure $\boldsymbol{Adjust}(m)$ are addressed in Appendix D.

Step 4: For i =*1, 2,* $\cdots$, K, allocate containers to YTs in terms of the ascending order of the QC scheduling sequence, which determined by the value of X_{N+i} that described in section 4.1.

Step 4.1: For each QC, sort out all the containers based on the ascending order of their completion time.

Step 4.2: Allocate the containers to YTs with the earliest available time criterion and check it. Calculate the completion time of the containers on the allocated YTs. Meanwhile, record the earliest available time of the YTs. Repeat this process until all containers are allocated by YTs.

Step 5: For i =*1, 2,* $\cdots$,K, calculate the fitness value.

Step 6: Update the best position of each particle, $pBest$.

Step 7: Update the best position of the swarm, $gBest$.

Step 8: Update the velocity and the position of each particle.

Step 9: If n reaches the preset maximum iteration, stop; otherwise set n= $n+1$ and go to step 2.

4.3 Modifying the generated particles at each iteration

The initial particles are generated randomly for ensuring the diversity of particles. In this study, we set containers that belong to the same bay must be handled by the same QC. In addition, the containers in the same bay should be followed some certain handling precedence relationship. However, some randomly generated particles may not satisfy these requirements. Thus modifications should be made for these infeasible particles so that they can satisfy the practical constraints. Thus we propose a method as below to modify the infeasible dimension values. Suppose two containers i and j in the same bay. If their QC allocations disobey the constraints in the randomly generated particle process, i.e., the value in dimension i and j, $\lceil X_i \rceil \neq \lceil X_j \rceil$. We will revise the integer part of X_j so that the revised value of X_j satisfy $\lceil X_i \rceil = \lceil X_j \rceil$. If two containers i and j in the same bay, and container i must be handled before container j. But the values of X_{N+i} and X_{N+j} violate the precedence constraints, we would exchange the two values. Table 3 and Table 4 demonstrate an example to modify the random generated particles. In the example, there is a precedence relationship that requires Task 2 must be handled before Task 3. Table 3 shows the modification of the N dimensions in the front part of the vector. Table 4 shows the modification of the N dimensions in the rear part of the vector.

Table 3: An example of modifying an initially generated solution (Part 1)

Task No.	Bay No.	Original X_i	Modified X_i
1	1	0.32	0.32
2	2	0.65	0.65
3	2	1.40	0.40
4	3	1.71	1.71
5	5	1.36	1.36

Table 4: An example of modifying an initially generated solution (Part 2)

Task No.	Bay No.	Original X_{i+N}	Modified X_{i+N}
1	1	2.77	2.77

2	2	5.99	4.99
3	2	4.99	5.99
4	3	2.41	2.41
5	5	3.87	3.87

4.4 Re-updating the particles at each iteration

A demerit of the PSO algorithm is that the solutions may easily fall into the local optima. Though we impose an inertia weight coefficient to balance it, the particles prefer to fly towards the best position that it has ever reached. In order to avoid falling into local optima quickly, we use some randomly generated particles to replace the original particles at each iteration. The probability of replacing the particles by some new ones is set as ε $(0 < \varepsilon < 1)$.

5. Computational experiments

We conduct experiments to assess the solution quality and efficiency of our algorithm. All experiments are performed by CPLEX 11.0 with technology of C# (VS2012) on a PC (Intel Core i3, 2.4 GHz; Memory, 2G).

To validate the performance of the proposed propositions, we compare the results obtained by CPLEX solver (the existing model in which the value of M is set to 10,000) with CPLEX-M (the existing model in which the value of M is set based on Proposition 2) solver and CPLEX-S (the existing model with Proposition 3) solver. For evaluating the performance of the proposed PSO based algorithm. In small-scale problems, the PSO algorithm is compared with the optimal solution obtained by CPLEX-S-M (the existing model with both Proposition 2 and Proposition 3) solver. In large-scale instances, the PSO algorithm is compared with the widely used genetic algorithm (GA). In this paper, the GA is similar to Lee et al. (2008). Some details about the GA we can refer to Golberg (1989) Hartmann (2001), Alp et al. (2003). The chromosome of GA coded and decoded the sequence of containers are the same way as the proposed PSO algorithm, which described in sub-section 4.1. In the GA used in the following experiments, a roulette wheel approach is used as the selection procedure, the chromosomes probability of crossover, the chromosomes mutation probability are set as 0.4 and 1/2 N(2 N is the number of genes). The size of the population is 20, and the algorithm will be stop iterated until the pre-specified generation number 50 is reached. For the PSO algorithm, the procedure termination condition and the population size are set as the same as the GA in this study. Based on the results of test runs, we set the two acceleration weights c_1 and c_2 as c_1=1, c_2=1; the maximum value of the inertia weight coefficient $\beta_{max} = 1.2$, the minimum value of the inertia weight coefficient $\beta_{min} = 0.7$, and the probability of replacing the particles by some new one is set as $\varepsilon = 0.01$.

The constant weight λ is set as 0.4. The experiment data is randomly generated as follows: (1) the time that a QC unloads a container is generated by following a uniform interval [150,190] (including the picking up time, dropping off time and traveling time by QC); (2) the roundtrip time that a YT transports a container is generated by following a uniform interval [50, 90].

5.1 Performance evaluation of Proposition 2

For evaluating the performance of the proposed Proposition 2 that calculate a proper value of M, we compare the results obtained by the existing model with Proposition 2 (CPLEX-M) with the optimal results solved by CPLEX solver (CPLEX). The experiment results are listed in Table 5.

From Table 5, we can observe the CPLEX solver and CPLEX-M solver obtain the same objective results. However, Proposition 2 helps the CPLEX-M solver reduce the computational time significantly. As shown in Table 5, the average computational time required by CPLEX is 515.93(s). While the average computational time of the model with Proposition 2 (CPLEX-M) is 269.44(s), which is 52.22% of the time of the CPLEX solver. We can observe that in most cases of Table 5 (e.g., cases 9, 14 and 16), the CPLEX-M solver considerably outperforms the CPLEX solver.

Table 5: Comparison between the CPLEX solver and the CPLEX-M solver

	Instance	CPLEX		CPLEX-M		Time(s)
ID	#	OBJ	Time(s)	OBJ	Time(s)	Gap
1	5-2-2	816	1.82	816	1.44	20.88%

2	5-3-3	676.5	1.43	676.5	1.43	0.00%
3	6-2-2	927	5.06	927	3.36	33.60%
4	6-3-3	681	1.74	681	1.73	0.57%
5	8-2-4	1137	13.64	1137	14.68	-7.62%
6	8-3-6	879	16.63	879	10.01	39.81%
7	10-2-4	1437	90.50	1437	94.79	-4.74%
8	10-3-6	1176	39.37	1176	27.29	30.68%
9	11-2-4	1599	357.65	1599	166.49	53.45%
10	11-3-6	1386	55.21	1386	69.09	-25.14%
11	11-4-8	1383	150.06	1383	154.25	-2.79%
12	12-2-4	1650	726.81	1650	468.61	35.53%
13	12-3-6	1416	172.41	1416	115.76	32.86%
14	12-4-8	1413	389.69	1413	167.48	57.02%
15	13-3-6	1431	258.07	1431	152.25	41.00%
16	13-4-8	1428	1748.61	1428	416.33	76.19%
17	14-3-6	1431	1138.54	1431	1019.49	10.46%
18	14-4-8	1428	4119.57	1428	1958.32	52.46%
Average(s)		1222.18	515.93	1222.18	269.44	47.78%

Notes: (1) Instance # denotes No. of tasks - No. of QCs - No. of YTs.

(2)Gap= (Time (CPLEX)－ Time (CPLEX-M)) / Time (CPLEX) × 100%.

5.2 Performance evaluation of Proposition 3

In order to evaluate the performance of Proposition 3, we compare the results obtained by the existing model adds the Proposition 3 (CPLEX-S) with the optimal results solved by the CPLEX solver. Table 6 reports the best solutions found by the CPLEX-S solver, the CPLEX solver and the average improvement.

From the results presented by Table 6, we can observe the CPLEX solver and CPLEX-S solver obtained the same objective results. The last column in the table shows the gap of the CPLEX and CPLEX-S solvers with respect to their computation time. It shows the average improvement rate of the CPU time by using Proposition 3 is 3.91%. The results indicate the existing model with Proposition 3 (CPLEX-S) outperforms the existing model solved by the CPLEX solver.

Table 6: Comparison between the CPLEX solver and the CPLEX-S solver

	Instance	CPLEX		CPLEX-S		Time(s)
ID	#	OBJ	Time(s)	OBJ	Time(s)	Gap

1	5-2-2	816	1.82	816	1.45	20.33%
2	5-3-3	676.5	1.43	676.5	1.39	2.80%
3	6-2-2	927	5.06	927	4.91	2.96%
4	6-3-3	681	1.74	681	1.73	-0.57%
5	8-2-4	1137	13.64	1137	13.66	-0.15%
6	8-3-6	879	16.63	879	16.33	1.80%
7	10-2-4	1437	90.50	1437	90.04	0.51%
8	10-3-6	1176	39.37	1176	39.48	-0.28%
9	11-2-4	1599	357.65	1599	344.12	3.78%
10	11-3-6	1386	55.21	1386	55.05	0.29%
11	11-4-8	1383	150.06	1383	148.37	1.13%
12	12-2-4	1650	726.81	1650	191.29	4.89%
13	12-3-6	1416	172.41	1416	170.82	0.29%
14	12-4-8	1413	389.69	1413	385.29	1.13%
15	13-3-6	1431	258.07	1431	263.25	-2.01%
16	13-4-8	1428	1748.61	1428	1647.73	5.77%
17	14-3-6	1431	1138.54	1431	1114.61	2.10%
18	14-4-8	1428	4119.57	1428	3933.95	4.51%
Average(s)		1222.18	515.93	1222.18	495.75	3.91 %

Notes: (1) Instance # denotes No. of tasks - No. of QCs - No. of YTs.

(2)Gap= (Time (CPLEX) − Time (CPLEX-S)) / Time (CPLEX) × 100%.

5.3 Performance of the proposed solution methods

In order to validate the performance of the proposed PSO based solution methods. For small scale problems we compare the results obtained by the PSO based solution methods with the optimal results obtained by CPLEX-S-M (the existing model with both Proposition 2 and Proposition 3) solver. The objective value results obtained by the two methods and their computation time (in seconds) are presented in Table 7.

From Table 7, we can observe the objective value obtained by the PSO based algorithm is close to the optimal results solved by CPLEX-M-S solver. The average optimality gap of the PSO is about 2.23%. Moreover, for some large instances, the CPLEX-M-S solver cannot solve the integrated model directly within a reasonable time, while the PSO based algorithm can solve it within a reasonable time. The results demonstrate the proposed PSO based algorithm is effective and efficient to solve the integrated optimization model.

Table 7: Comparison between the PSO and the CPLEX-M-S for small-scale instances

ID	Instance #	CPLEX-S-M OBJ	CPLEX-S-M Time(s)	PSO OBJ(s)	PSO Time(s)	OBJ Gap
1	8-2-4	1137	13.51	1152	68.86	1.31%
2	8-3-6	879	9.76	879	184.87	0.00%
3	10-2-4	1437	91.05	1437	127.55	0.00%
4	10-3-6	1176	27.52	1182	233.57	0.51%
5	11-2-4	1599	240.17	1629	151.83	1.88%
6	11-4-8	1383	151.89	1398	260.24	1.08%
7	12-2-4	1650	548.43	1704	225.88	3.27%
8	12-4-8	1413	166.43	1458	588.93	3.18%
9	13-2-4	1428	1438.09	1434	612.73	1.49%
10	13-4-8	N.A.	N.A.	2160	554.41	N.A.
11	14-2-4	1428	1121.64	1539	868.39	7.78%
12	14-4-8	N.A.	N.A.	2164.5	342.63	N.A.
13	16-2-4	N.A.	N.A.	1752	1529.41	N.A.
14	16-4-8	N.A.	N.A.	2458.5	426.37	N.A.
15	18-2-4	N.A.	N.A.	1929	1344.37	N.A.
16	18-4-8	N.A.	N.A.	2755.5	724.87	N.A.
Average(s)						2.23%

Notes: (1) Instance # denotes No. of tasks - No. of QCs - No. of YTs.

(2)Gap= (OBJ (CPLEX-M-S) − OBJ (PSO)) / OBJ (CPLEX-M-S) × 100%.

To further evaluate the effectiveness of the proposed PSO based algorithm in large scale instances. We compare the PSO based algorithm with the widely used genetic algorithm (GA). The results are listed in Table 8.

Based on the above results, we can observe that the proposed PSO based algorithm outperforms the widely used GA algorithm for the cases in Table 8. In the process of experiments, the computational time of the GA algorithm significantly increases as the problem size growing. For some large-scale problems such as cases 10-18, the GA based algorithm cannot solve the model within 10,000 seconds, while the proposed PSO based algorithm can solve the model within a reasonable time. Moreover, the average improvement rate of the proposed PSO based algorithm with respect to the objective values is about 14.35% by comparing with the GA based algorithm.

Table 8: Comparison between the PSO and the GA under large-scale instances

ID	Instance #	PSO OBJ	PSO Time(s)	GA OBJ	GA Time(s)	OBJ Gap
1	14-4-8	1689	868.39	1911	1221.99	11.62%
2	16-2-4	2164.5	342.63	2634	717.08	17.82%
3	16-4-8	1752	1579.41	1978.5	1952.23	11.45%
4	18-2-4	2458.5	426.37	2736	888.28	10.14%
5	18-4-8	1929	1344.77	2277	3083.41	15.28%
6	20-2-4	2755.5	724.87	3163.5	1030.49	12.90%
7	20-4-8	2167.5	1975.37	2514	2316.47	13.78%
8	22-2-4	2985	599.86	3483	1240.93	14.30%
9	22-3-6	2493	1202.62	3190.5	2683.70	21.86%
10	22-4-8	2452.5	3088.36	N.A.	>10,000	N.A.
11	24-2-4	3318	1650.27	N.A.	>10,000	N.A.
12	24-3-6	2905.5	1868.55	N.A.	>10,000	N.A.
13	24-4-8	2704.5	4366.14	N.A.	>10,000	N.A.
14	26-2-4	3970.5	1022.08	N.A.	>10,000	N.A.
15	26-3-6	2952	2709.38	N.A.	>10,000	N.A.
16	26-4-8	2884.5	3051.63	N.A.	>10,000	N.A.
17	28-2-4	4180.5	1044.94	N.A.	>10,000	N.A.
18	28-3-6	3219	3135.18	N.A.	>10,000	N.A.
Average						14.35%

Notes: (1) Instance # denotes No. of tasks - No. of QCs - No. of YTs.
(2)Gap= (OBJ (GA) − OBJ (PSO)) / OBJ (GA) × 100%.

It should be noted that the above results cannot rigorously prove the PSO is better than GA in universal context because the above experiments are conducted for some specific problem cases and under some certain parameter settings of PSO and GA. At the same time, the results indeed demonstrate that the proposed PSO based algorithm could be a proper solution method for solving the integrated QC and YT scheduling problem.

6. Conclusions

This paper studies an integrated QC and YT optimization scheduling problem with unidirectional flow in container terminals, a MIP model is formulated. This integrated scheduling problem is proved to be strongly NP-hard. A method is proposed to calculate a proper value of big number M. Moreover, we make some attempts to mitigate the influence of the symmetry issue and a proposition about this is proposed. Computational experiments validate the efficiency of the proposition. A PSO based solution method is developed to solve the problem. Numerical experiments show the relative gap of the objective value obtained by the PSO based solution method from the optimal objective value is 2.23% on average in small scale problems. Some experiments on the large scale instances are also conducted; and results show that the PSO based solution method could be a proper solution method for solving the integrated QC and YT scheduling problem.

For practitioners, the proposed model and method in this study can provide some quantitative decision tools for the decision makers (yard resource planners) to further improve the efficiency of the schedules on trucks and QCs. More specifically, the proposed model and methods can be inbuilt in a TOS (terminal operating system), and be further developed as a DSS (decision support system) for the planners in various departments. The DSS with our proposed technique embedded in its kernel may be much faster than a DSS with the normal CPLEX solver embedded, according to the experimental result that the proposed technique on average can save 47.78% computational time compared with the direct solving mode of the CPLEX solver. In all, the proposed model and method in this study can be potentially useful for enriching the database of the algorithms embedded in some TOSs of the port operators.

This study also contains limitations. For example, the congestions of the vehicles in yard (Zhen, 2016) and some stochastic factors (Zhen, 2015) have not been taken into account. In future research, stochastic influences such as the YT congestion and productivity rate fluctuations for QCs and YTs should be further explored.

Appendices

Appendix A. Proof of Proposition 1

Proposition 1: Finding an optimal solution for the model is strongly NP-hard.

Proof: We can prove the proposition by reducing the problem in polynomial time to the 3-Partition Problem (Garey and Johnson (1979), Liu and Tang (2008)), which is well-known to be strongly NP-hard. Given $3h$ items, $H = \{1,2,\cdots,3h\}$, each item $j \in H$ has a positive integer size a_j satisfying $a/4 < a_j < a/2$, and $\sum_{j=1}^{3h} a_j = ha$, for some integer a. The 3-Partition Problem asks whether there are h disjoint subsets $H_1, H_2, \cdots, H_h$ of H such that $|H_i| = 3$ and $\sum_{j \in H_i} a_j = a$, $i = 1,2,\cdots,h$.

Given any instance of a 3-Partition Problem, consider the following instance that we construct for our problem: number of tasks: $N = 3h$, number of QCs: $K = h$, processing time: $t_j = P_{j1} + P_{j2} = a_j$, $j = 1,2,\cdots 3h$. We will show that there exists a solution to the 3-Partition Problem if and only if there is a feasible solution to our integrated QC and YT scheduling problem. (i) The "only-if" direction: Given a solution to the 3-Partition Problem, $H_1, H_2, \cdots, H_h$, we can simply let tasks in set S_j correspond to the elements of H_j, $1 < j < h$, and construct a schedule to our integrated QC and YT scheduling problem. (ii) The "if" direction: Suppose there exists a schedule for the constructed instance of our integrated QC and YT scheduling problem. Since the total processing time of all tasks is $\sum_{j=1}^{3h} a_j = ha$, a total of h QCs are fully utilized and the completion time of each QC is a. So we know the schedule contains exactly h sets, the total processing time of tasks in each set S_j is a. A partition for the set H is obtained by mapping the elements corresponding to the tasks in set S_j to the elements in the sub-set H_j, $1 < j < h$. Then $\sum_{j \in H_i} a_j = a$, and $|H_i| = 3$ because $a/4 < a_j < a/2$.

It can be thus seen that our integrated QC and YT scheduling problem has a feasible solution if and only if there exists a solution to the 3-Partition Problem. The reduction of the integrated QC and YT scheduling problem to the 3-Partition Problem can be done in polynomial time. Therefore, finding an optimal solution for the model is strongly NP-hard. ■

Appendix B. Proof of Proposition 2

Proposition 2: A formula used to calculate a proper value of M about the integrated QC and YT scheduling problem is as follows:

$$M^{UB} = f_q + g \times |l_0^q - L_1| + P_{11} + \sum_{i\in\Omega\backslash\{1\}} max\{P_{i-1,2}, g \times |L_i - L_{i-1}| + P_{i1}\} + P_{|\Omega|2}$$

Proof: Suppose that (I) there is only one QC (the first QC in the set Q) and one YT; (II) the QC moves from one side to the next; (III) the containers are labeled such that container 1 can be unloaded first, container 2 can be unloaded second, etc. Then we can calculate the time required to complete all of the tasks, i.e., $max_{i\in\Omega} C_{i2}$ as the formula (29).

In formula (29), $q = 1$. The first term is the ready time of the QC. The second term is the moving time of the QC from the initial position to the first task. The third term is the QC time for the first task. In the fourth term, the YT can receive task i from the QC if (I) the YT has completed task $i-1$ (the component $P_{i-1,2}$), and (II) the QC has moved to the bay where task i is located and has unloaded task i (the component $g \times |L_i - L_{i-1}| + P_{i1}$). The fifth term is the transportation time of the last task. Through this formula a valid value for M can be equal to the above time. We can see that for different-scale problems the corresponding data M is not the same. When the problem size is increasing, the value of M is increasing. Compared with the traditional data M that usually is a single and sufficiently large value, the value of M that we calculated by the proposed formula is more reasonable. In our model there are seven constraints that use the value of M, including Constraints (9), (10), (12)-(15), (22). A proper value of M can improve the computational efficiency. Through the computational experiments we can also confirmed the validity ofthe proposed proposition.■

Appendix C. Proof of Proposition 3

Proposition 3: To reduce the influence of symmetry, we propose the following method as follows: $\sum_{j\in\Omega^F} j.Y_{0j}^k \leq \sum_{j\in\Omega^F} j.Y_{0j}^{k+1},\ k \in T/\{|T|\}$.

Proof: We prove in the following that the above constraint removes some feasible solutions and optimal solutions, but at least one optimal solution is not removed. We prove it in two steps.

(i) Evidently, the problem has optimal solutions. We let $\{Y_{ij}^{k^*}\}, (i, j \in \Omega^F, k \in T)$ be an optimal solution that does not satisfy Constraint (30).

Then at least $\exists\ \bar{K} \in T$, such that $\sum_{j\in\Omega^F} j.Y_{0j}^{\bar{k}} \neq 0$. Through the solution $\{Y_{ij}^{k^*}\}$, $(i, j \in \Omega^F, k \in T)$, we will construct a new optimal solution satisfying Constraint (30) below.

Firstly we define a parameter $\pi_k = \sum_{j\in\Omega^F} j \cdot Y_{0j}^{k^*},\ k \in T$. π_k denotes the ID of the first task that YT k transports. Note that if YT k is not used, then $\pi_k = 0$. And then define a function $\{\theta(i)\}$:

$\{\theta(i)\} = \{1,2, \cdots\cdots|T|\}$, $i = 1,2 \cdots\cdots|T|$, and $\pi_{\theta(1)} \le \pi_{\theta(2)} \le \pi_{\theta(3)} \cdots\cdots \le \pi_{\theta(|T|)}$.

That is, we sort the first task's ID in ascending order and $\theta(i)$ denotes the ID of the YT whose first task is the i th smallest.

Let us give an example to illustrate. If YT 1 transports tasks 3, 5, and 9, YT 2 transports task 8, and YT 3 transports tasks 1, 2, 4, 6 and 7, then $\pi_1 = 3$, $\pi_2 = 8$, $\pi_3 = 1$ and $\theta(1) = 3, \theta(2) = 1, \theta(3) = 2$. Thus, we have $\pi_{\theta(1)} = 1 \le$ $\pi_{\theta(2)} = 3 \le$ $\pi_{\theta(3)} = 8$. Now a new solution

$\{\bar{Y}_{ij}^k\} = \{Y_{ij}^{\theta(k)^*}\}, (i, j \in \Omega^F, k \in T)$

is defined. We can observe the new solution $\{\bar{Y}_{ij}^k\}$ is also a feasible solution with the same objective function value as $\{Y_{ij}^{\theta(k)^*}\}$, and satisfy Constraint (30). Therefore, there exists at least one optimal solution that satisfies Constraint (30).

(ii) Now we would prove Proposition 3 could remove a part of the optimal solutions. Suppose $\{\bar{Y}_{ij}^k\}$ is an optimal solution and satisfies Constraint (30). Define $\bar{k}$:

$\bar{k}$=$argmin\{\sum_j \bar{Y}_{0j}^k > 0\}, k \in T, j \in \Omega^F$

If $\bar{k} < |T|$, we exchange $\bar{k}$ and $\bar{k}+1$, and then a new optimal solution to the original model that does not satisfy Constraint (30) is generated. If $\bar{k} = |T|$, we exchange $\bar{k}$ and $\bar{k}-1$, then a new optimal solution that does not satisfy Constraint (30) is generated. In sum, Proposition 3 removes some optimal solutions to the original model.

Proposition 3 can remove some feasible solutions and a part of optimal solutions but do not remove all the optimal solutions. In this way we can reduce the influence of symmetry. ■

Appendix D. The sub-procedure Adjust(m)in in the PSO based method

The sub-procedure $\boldsymbol{Adjust}(m)$

For all the $i, j,\ i, j \in \ \Omega$

For all the $q,\ q \in Q$

Define a particle $Randomposition[\ ]$ // Each particle has $2 \times |\Omega|$ dimensions

Define a set $Collection_{iq}$ // This set is used to denote the number of containers that handled by QC q

For all $m,\ n,\ m$ and n are in the front of $|\Omega|$ dimensional elements of $Randomposition\ [i]$

If $(m, n)\ \in \Phi$

If $Randomposition\ [m] > Randomposition\ [n]$,***Then***

$temp = Randomposition\ [m]$ // the temp is a transit variable

$Randomposition\ [m] = Randomposition\ [n]$

$Randomposition\ [n] = \ temp$ // According to Constraint (7)

End If

End If

End For

For all $u, v,\ u$ and v are the rear part of $|\Omega|$ dimensional elements of $Randomposition\ [i]$

If $X_{uv}^{q} = 1$

If $\lfloor Randomposition\ [u] \rfloor \neq \lfloor Randomposition\ [v] \rfloor$, ***Then***

$Randomposition\ [v] = \lfloor Randomposition\ [u] \rfloor + \ Randomposition\ [v]$ $- \lfloor Randomposition\ [v] \rfloor$ // According to Constraint (5)

End If

End If

End For

End For

End For

Appendix E. The pseudo code of PSO algorithm

PSO algorithm pseudo code

*//*Initialization*// Initialize each particle's position and velocity*

Parameter setting: $numPar \coloneqq 20, t \coloneqq 1, maxStep \coloneqq 200, c_1 := 1.0, c_2 \coloneqq 1.0, w_{max} = 1.2, w_{min} \coloneqq 0.7, \varepsilon \coloneqq 0.01, countGbest \coloneqq 0, count \coloneqq 0$

For *i* = 1 to 150 ***do***

Generate a particle l_{mi}^{n} randomly

// the value of dimension: $l_{mi}^{0}[t] \in [0, |Q|), t \in [1, N]; l_{mi}^{0}[t] \in [0, |N|), t \in [N+1, 2N]$

If (Constraints (5)&& $Constraints(7)$ satisfy) ***then***

Add Particle l_{mi}^{n} to the set of particles $\mathcal{P}$

If (number of particles $\mathcal{P} \geq$ numPar) ***then***

Obtain the initial swarm, and Jump out the ***For*** loop

End If

End If

End For

For *j* = 1 to 20 ***do***

Initialize Particle's velocity randomly

Calculate Particle l_{mj}^{n}'s fitness value $f\left(l_{mj}^{n}\right)$, find out the best particle l_{mj}^{n}, and Set $Pbest^{0} = l_{mj}^{n}, f_{Pbest}^{0} = f\left(l_{mj'}^{n}\right)$ $Gbest = l_{mj'}^{n}, f_{Gbest} = f\left(l_{mj'}^{n}\right)$

End For

While $(t \leq maxStep)$ ***do*** *// Evaluation Loop*

$countGbest = f_{Gbest}$

For $ii = 1$ to 20 ***do***

Update the velocity and position of Particle l_{mii}^{n} according to Formula (33) and (32), replace the original particles at the probability of ε

Calculate Particle l_{mii}^{n}'s fitness value $f(l_{mii}^{n})$

If $(f(l_{mii}^{n}) < f_{Pbest}^{t-1})$ ***then*** Set $Pbest^{t} = l_{mii}^{n}$, $f_{Pbest}^{t} = f(l_{mii}^{n})$

If $(f_{Pbest}^{t} < f_{Gbest})$ ***then*** $Gbest = Pbest^{t}$, $f_{Gbest} = f_{Pbest}^{t}$

End If

End If

End For

Set $t = t + 1$ *//Termination condition*

Global optimal solution unchanged in 10 consecutive iterations, jump out the loop

If $(f_{Gbest} == countGbest)$ ***then***

$count = count + 1$

Else

$count = 0$

End If

If $(count \geq 10)$ ***then*** Jump out While loop

End If

End While

The global optimal particle and fitness is Gbest and f_{Gbest}, respectively.

Appendix F. The pseudo code of GA algorithm

GA algorithm pseudo code

// Initialize each individual
Parameter setting: $num_{Indiv} \coloneqq 20, t \coloneqq 1, maxStep \coloneqq 200,\ mutation_{prob} \coloneqq 1/N,$
$crossover_{prob} \coloneqq 0.40, countGbest \coloneqq 0,\ count \coloneqq 0$
For $i = 1$ to 150 ***do*** Generate one individual l_{mi}^{0} randomly
// the value of dimension: $l_{mi}^{0}[t] \in [0, |Q|), t \in [1, N]; l_{mi}^{0}[t] \in [0, |N|), t \in [N+1, 2N]$
If (Constraints (5) && Constraints (7) satisfy) ***then***
Add individual l_{mi}^{0} to the set of Individuals $\mathcal{P}$
Evaluate the fitness of each individual F_i^0. Find out the best individual Bl_m^0, and the best fitness F_{best}^{0}
If (number of individuals $\geq num\_Indiv$) ***then***
Obtain the initial population, and Jump out the ***For*** loop
End If
End If
End For ***//Evaluation Loop***
While ($t \leq maxStep$) ***do*** $countGbest = F_{best}^{0}$
For $j = 1$ to 20 ***do***
If (rand()<$crossover_{prob}$) Select two parents individuals $p1_j^t$, $p2_j^t$ randomly
Generate one crossover point p, $p2_j^t[i]$ copy to $Child1_j^t[i]$; $p1_j^t$ copy to $Child2_j^t[i]$
For i=1 to p ***do*** $p1_j^t[i]$ *copy to* $Child2_j^t[i]$; $p2_j^t[i]$ *copy to* $Child1_j^t[i]$
End For
For *i= p +1 to 2N* ***do*** $p1_j^t[i]$ *copy to* $Child1_j^t[i]$; $p2_j^t[i]$ *copy to* $Child2_j^t[i]$
End For
End if
For i =1 to 2N do
If rand () < $\boldsymbol{mutation_{prob}}$
do $\boldsymbol{p_j^t[i]} = \boldsymbol{p_j^t}[i] + (maxGene[i] - minGene[i]) * rand()$, $\boldsymbol{Child_j^t[i]} = \boldsymbol{p_j^t[i]}$
End If
End For
End for
Set $t = t + 1$ ***// Termination condition***
If $(\boldsymbol{f_{Gbest}} == countGbest)$ ***then*** $count = count + 1$
Else $count = 0$
End If
If $(count \geq 10)$ ***then*** Jump out **While** loop
End If
End While

Acknowledgements

The authors would like to thank the editors and three anonymous reviewers for their valuable comments and constructive suggestions, which have greatly improved the quality of this paper. This research is supported by the National Natural Science Foundation of China (71422007), Shanghai Social Science Research Program (2014BGL006).

References

Alp, O., Erkut, E., & Drezner, Z. (2003). An efficient genetic algorithm for the p-median problem. *Annals of Operations research*, 122(1-4), 21-42.

Bierwirth, C. and Meisel, F. (2009). A fast heuristic for quay crane scheduling with interference constraints. *Journal of Scheduling* 12, 345-360.

Bish, E. K. (2003). A multiple-crane-constrained scheduling problem in a container terminal. *European Journal of Operational Research* 144, 83-107.

Bish, E. K., Chen, F. Y., Leong, Y. T., Nelson, B. L., Ng, J. W. C. and Simchi-Levi, D. (2005). Dispatching vehicles in a mega container terminal. *OR Spectrum* 27, 491-506.

Bruzzone, A. and Signorile, R. (1998). Simulation and genetic algorithms for ship planning and shipyard layout. *Simulation*, 71, 74-83.

Cao, J. X., Lee, D. H., Chen, J. H. and Shi, Q. (2010). The integrated yard truck and yard crane scheduling problem: Benders' decomposition-based methods. *Transportation Research Part E* 46, 344-353.

Chen, L., Bostel, N., Dejax, P., Cai, J. and Xi, L. (2007). A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. *European Journal of Operational Research* 181, 40-58.

Chen, L., Langevin, A. and Lu, Z. (2013). Integrated scheduling of crane handling and truck transportation in a maritime container terminal. *European Journal of Operational Research* 225, 142-152.

Chipperfield, A., Fleming, P. and Pohlheim, H. (1994) . Genetic Algorithm Toolbox: For Use with MATLAB; User's Guide (version1.2). *University of Sheffield, Department of Automatic Control and Systems Engineering.*

Daganzo, C. F. (1989). The crane scheduling problem. *Transportation Research Part B* 23, 159-175.

Du, Y., Chen, Q., Quan, X., Long, L. and Fung, R. Y. (2011). Berth allocation considering fuel consumption and vessel emissions. *Transportation Research Part E* 47, 1021-1037.

Eberhart, R. C. and Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science* 1, 39-43.

Fu, Y. M., Diabat, A., and Tsai, I. T. (2014). A multi-vessel quay crane assignment and scheduling problem: Formulation and heuristic solution approach. *Expert Systems with Applications*, 41, 6959-6965.

Garey, M.R. and Johnson, D.S. (1979). *Computers and intractability: a guide to the theory of NP-completeness.* W.H. Freeman, New York.

Golberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. *MA: Addion- wesley*, 1989, 102.

Guan, Y., Yang, K. H. and Zhou, Z. (2013). The crane scheduling problem: models and solution approaches. *Annals of Operations Research*, 203, 119-139.

Guo, P., Cheng, W. and Wang, Y. (2014). A modified generalized extremal optimization algorithm for the quay crane scheduling problem with interference constraints. *Engineering Optimization* 46, 1411-1429.

Han, X., Gong, X. and Jo, J. (2015). A new continuous berth allocation and quay crane assignment model in container terminal. *Computers & Industrial Engineering*, 89, 15-22.

Hartmann, S. (2001). Project scheduling with multiple modes: a genetic algorithm. *Annals of Operations Research*, 102(1-4), 111-135.

He, J., Huang, Y., Yan, W. and Wang, S. (2015). Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption. *Expert Systems with Applications* 42, 2464-2487.

Hu, H., Lee, B. K., Huang, Y., Lee, L. H. and Chew, E. P. (2013). Performance analysis on transfer platforms in frame bridge based automated container terminals. *Mathematical Problems in Engineering*.

Jiang, X., Lee, L. H., Chew, E. P., Han, Y. and Tan, K. C. (2012). A container yard storage strategy for improving land utilization and operation efficiency in a transshipment hub port. *European Journal of Operational Research* 221, 64-73.

Jin, J. G., Lee, D. H. and Hu, H. (2015). Tactical berth and yard template design at container transshipment terminals: A column generation based approach. *Transportation Research Part E* 73, 168-184.

Jung, S. H. and Kim, K. H. (2006). Load scheduling for multiple quay cranes in port container terminals. *Journal of Intelligent Manufacturing* 17, 479-492.

Junior, H. V. and Lins, M. P. E. (2005). An improved initial basis for the simplex algorithm. *Computers & Operations Research* 32, 1983-1993.

Kaveshgar, N. and Huynh, N. (2015). Integrated quay crane and yard truck scheduling for unloading inbound containers. *International Journal of Production Economics* 159, 168-177.

Kim, K. H. and Park, Y. M. (2004). A crane scheduling method for port container terminals. *European Journal of Operational Research* 156, 752-768.

Kim, K. Y. and Kim, K. H. (1999). A routing algorithm for a single straddle carrier to load export containers onto a containership. *International Journal of Production Economics* 59, 425-433.

Lee, D. H., Wang, H. Q. and Miao, L. (2008). Quay crane scheduling with non-interference constraints in port container terminals. *Transportation Research Part E 44*, 124-135.

Legato, P., Trunfio, R. and Meisel, F. (2012) Modeling and solving rich quay crane scheduling problems. *Computers & Operations Research* 39, 2063-2078.

Li, H. L. (1996). An efficient method for solving linear goal programming problems. *Journal of Optimization Theory and Applications* 90, 465-469.

Lim, A., Rodrigues, B., Xiao, F. and Zhu, Y. (2004). Crane scheduling with spatial constraints. *Naval Research Logistics* 51, 386-406.

Li, F., Gao, Z., Li, K., & Wang, D. Z. (2012). Train routing model and algorithm combined with train scheduling. *Journal of Transportation Engineering*, 139(1), 81-91.

Liu, J., Wan, Y. W. and Wang, L. (2006). Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures. *Naval Research Logistics* 53, 60-74.

Liu, P.and Tang, L. (2008). The refining scheduling problem with crane non-collision constraint in steelmaking process. *In Automation and Logistics, 2008, IEEE International Conference on, IEEE* 536-541.

Liu, Z., Wang, S., Chen, W., & Zheng, Y. (2016). Willingness to board: A novel concept for modeling queuing up passengers. *Transportation Research Part B*, 90, 70-82.

Nguyen, V. D. and Kim, K. H. (2009). A dispatching method for automated lifting vehicles in automated port container terminals. *Computers & Industrial Engineering* 56, 1002-1020.

Nguyen, S., Zhang, M., Johnston, M. and Tan, K. C. (2013). Hybrid evolutionary computation methods for quay crane scheduling problems. *Computers & Operations Research*, 40, 2083-2093.

Nielsen, I. E., Do, N. A. D., Nguyen, V. D., Nielsen, P. and Michna, Z. (2015). Reducing truck emissions in import operations at container terminal—a case study in a Singaporean port. *Technology Management for Sustainable Production and Logistics*, Springer Berlin Heidelberg, 133-151.

Pang, K.-W., Xu, Z. and Li, C.-L. (2011). Ship routing problem with berthing time clash avoidance constraints. *International Journal of Production Economics* 131, 752-762.

Peterkofsky, R. I. and Daganzo, C. F. (1990). A branch and bound solution method for the crane scheduling problem. *Transportation Research Part B* 24, 159-172.

Sammarra, M., Cordeau, J., Laporte, G.and Monaco, M. (2007). A tabu search heuristic for the quay crane scheduling problem. *Journal of Scheduling* 10, 327-336.

Shi, Y. and Eberhart, R. (1998). A modified particle swarm optimizer. *IEEE World Congress on Computational Intelligence* 69-73.

Stahlbock, R. and Voß, S. (2008). Operations research at container terminals: a literature update. *OR Spectrum* 30, 1-52.

Talley, W.K. and Ng, M.W. (2013). Maritime transport chain choice by carriers, ports and shippers. *International Journal of Production Economics* 142, 311-316.

Tang, L., Zhao, J. and Liu, J. (2014). Modeling and solution of the joint quay crane and truck scheduling problem. *European Journal of Operational Research* 236, 978-990.

Tran, N.K. and Haasis, H.-D. (2015). An empirical study of fleet expansion and growth of ship size in container liner shipping. *International Journal Production Economics* 159, 241-253.

Tavakkoli-Moghaddam, R., Makui, A., Salahi, S., Bazzazi, M. and Taheri, F. (2009). An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. *Computers & Industrial Engineering* 56, 241-248.

Ting, C. J., Wu, K. C. and Chou, H. (2014). Particle swarm optimization algorithm for the berth allocation problem. *Expert Systems with Applications* 41, 1543-1550.

Wang, S., Zheng, J., Zheng, K., Guo, J. and Liu, X. (2012). Multi resource scheduling problem based on an improved discrete particle swarm optimization. *Physics Procedia* 25, 576-582.

Wang, S., Meng, Q., & Liu, Z. (2013). Containership scheduling with transit-time-sensitive container shipment demand. *Transportation Research Part B*, 54, 68-83.

Yao, B., Yu, B., Hu, P., Gao, J. and Zhang, M. (2014). An improved particle swarm optimization for carton heterogeneous vehicle routing problem with a collection depot. *Annals of Operations Research*, 1-18.

Yu, B., Yang, Z., Sun, X., Yao, B., Zeng, Q. and Jeppesen, E. (2011). Parallel genetic algorithm in bus route headway optimization. *Applied Soft Computing*, 11, 5081-5091.

Zhen, L. (2015) Tactical berth allocation under uncertainty. *European Journal of Operational Research*, 247: 928-944.

Zhen, L. (2016) Modeling of yard congestion and optimization of yard template in container ports. *Transportation Research Part B*, 90: 83-104.