Computers & Operations Research 36 (2009) 1935 – 1944

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/cor

Integrating simulation and optimization to schedule loading operations in container terminals

Qingcheng Zeng, Zhongzhen Yang*

Transport Management College, Dalian Maritime University, 1#, Linghai Road, Dalian, Liaoning 116026, China

ARTICLE INFO

Available online 24 June 2008

Keywords:
Container terminal
Genetic algorithm
Hybrid flow shop problem
Simulation optimization

ABSTRACT

In this paper, a simulation optimization method for scheduling loading operations in container terminals is developed. The method integrates the intelligent decision mechanism of optimization algorithm and evaluation function of simulation model, its procedures are: initializing container sequence according to certain dispatching rule, then improving the sequence through genetic algorithm, using simulation model to evaluate objective function of a given scheduling scheme. Meanwhile, a surrogate model based on neural network is designed to predict objective function and filter out potentially bad solutions, thus to decrease the times of running simulation model. Numerical tests show that simulation optimization method can solve the scheduling problem of container terminals efficiently. And the surrogate model can improve the computation efficiency of simulation optimization.

1. Introduction

With the rapid development of container transport, container terminals have become the important nodes of transport network, which serve as hubs for the transshipment of containerized goods from ship to ship or from ship to other transport modes. As the container transport system is capital intensive, the turnaround time of ships at container terminals is an important factor for liner shipping companies to decrease their cost. The turnaround time includes berthing, unloading, loading and departure, therefore, the reasonable scheduling of loading and unloading operations is critical to the efficiency of container transport system. In addition, the rising competition among ports has compelled them to improve their service, which makes the efficiency of port operation an important factor to succeed in the fierce competition.

For most container terminals, there are mainly three types of equipments involved in the loading and unloading, i.e., quay cranes (QCs), yard trailers (YTs) and yard cranes (YCs). Upon a ship's arrival, QCs unload containers from or load containers onto the ship, and YTs move containers from quayside to storage yard and vice versa. At the storage yard, YCs perform the loading and unloading for YTs.

Scheduling the operation order of QCs, dispatching YTs to containers, allocating the optimal storage location for each container, and dispatching YCs to YTs in storage yard are major problems for the optimization of loading or unloading process in the container terminals. These problems are interrelated, and the efficiency of container terminal operations depends on the coordination of different types of equipments. However, most of the existing researches mainly focus on a special sub-process. In addition, operation of container terminal has the feature of uncertainty, multi-objectives and complexity, therefore, the optimization of the operation is too complex to be solved by mathematical programming model alone.

This paper will develop a simulation optimization method to schedule loading or unloading containers in container terminals. The optimization algorithm is used to search the solution space; and the simulation model is used to evaluate the solutions generated by the optimization module. Thus the intelligent decision mechanism of optimization algorithm and the evaluation function of simulation method are integrated.

This paper is organized as follows. In Section 2, a brief review of previous works is given. An optimization model based on hybrid flow shop problem is formulated in Section 3. The framework and procedure of simulation optimization method is developed in Section 4. A procedure for calculating makespan lower bounds is developed in Section 5. Numerical examples are used to test the performance of the proposed method in Section 6. And conclusions are given in Section 7.

2. Literature review

Issues related to container terminal operations have gained attention and have been extensively studied recently due to the

* Corresponding author. Tel./fax: +86 411 8472 6756.
E-mail addresses: zqcheng2000@tom.com (Q. Zeng), yangzhongzhen@263.net (Z. Yang).

0305-0548/$ - see front matter Crown Copyright © 2008 Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2008.06.010

increased importance of marine transport systems. Here, we provide a brief review of existing studies related to the operation scheduling problem in container terminals.

The first kind of studies are mathematical programming models, researchers developed optimization models and algorithms for different sub-processes of the container terminal operation system, such as QCs scheduling, YCs allocation and scheduling model, storage optimization, and YTs routing model, etc.

QCs are the main bottleneck of the efficiency of the container terminals, and their operation plan determines the turnaround time of a ship in the terminals. Daganzo [1], suggested an algorithm for determining the number of QCs assigned to ship-bays of multiple vessels. Kim [2] developed a mixed-integer programming model considering various constraints related to the operation of QCs, and proposed a heuristic search algorithm to solve the problem. Lee et al. [3,4] provided a mixed integer programming model for the considered QC scheduling problem, and developed a genetic-based algorithm to solve the model Goodchild and Daganzo [5] studied the double cycling method of QCs, and the performance of the double cycling method was evaluated.

Yard vehicles are used to transfer containers between the quay and the yard. Most of studies about routing problem in container terminals are focused on automated guided vehicle (AGV) and straddle carrier. Evers and Koppers [6] developed a hierarchical AGV control system by using semaphores. Liu and Ioannou [7] compared different AGV dispatching rules in container terminals. Vis [8] develop a heuristic based on the maximum flow problem to determine the fleet size of AGVs. Kim et al. [9,10] developed models and algorithms to optimize the routing of straddle carrier. Considering YTs, Nishimura et al. [11] proposed dynamic routing trailer assignment method, and developed a heuristic algorithm to solve the problem.

The scheduling of YCs determines the terminal efficiency to a great extent. Research focused on scheduling of YCs has been conducted widely. Zhang et al. [12] formulated the dynamic crane deployment problem as a mixed integer programming model and solved it by Lagrangean relaxation. Linna et al. [13] proposed an algorithm and a mathematical model for the optimal YC deployment problem. Kim et al. [14] developed a dynamic programming model to optimize the receiving and delivery operations of outside trucks, and derived the decision rule by learning-based method. Ng [15] examined the problem of scheduling multiple YCs to perform a given set of jobs with different ready times in a yard zone with only one bi-directional traveling lane.

The operations in terminals have the features of multi-objectives, uncertainty and complexity. Most of the existing researches focus on a special sub-process. Very little of the existing literature focuses on the integrated scheduling problem of various types of handling equipment used in container terminals. Bish [16] provided models and algorithms to integrate several sub-processes; the problem is (1) to determine a storage location for each unloaded container, (2) to dispatch vehicles to containers, and (3) to schedule the loading and unloading operations on the cranes, so as to minimize the maximum time it takes to serve a given set of ships. Chen et al. [17] proposed an integrated model to schedule different equipments in container terminals, and tabu search algorithm was designed to solve the model. These models and algorithms improve the coordination and integration of the operation scheduling in container terminals. However, how to tackle the complex constraints and interrelation, how to improve the computation efficiency, and how to realize coordination among different sub-processes are the problems that have not been solved well.

The scheduling problem of container terminals involves numerous variables and constraints. Therefore, when tackling complexion of model and computation, especially, considering the uncertain and stochastic factors, analytic models often confront either the problem that model is too simplify, or the problem that the computation is too complex. Therefore, simulation is wildly used in scheduling problem of container terminals recently.

Shabayek and Yeung [18] developed an application of a simulation model using Witness software to simulate Kwai Chung container terminals. Won et al. [19] proposed a simulation model for container terminal system analysis. The simulation model was developed using an object-oriented approach, and using SIMPLE + +, object-oriented simulation software. Maurizio et al. [20] outlined a container terminal simulation model and gave component architecture that was implemented with Java.

Simulation model can be used to evaluate the scheduling scheme; however, as a test and validation tool, it can only evaluate a given design, not provide more assistant decision making function. Recently, simulation optimization method is proposed to overcome these limitations, e.g., April et al. [21], Allaoui and Artiba [22], Guo et al. [23], Julian [24]. Combining the simulation analysis and the optimal decision-making mechanism, the simulation optimization method cannot only enhance intelligent decision making of the simulation, but also build the complex system model easily that is more difficult by traditional optimization methods. In this paper, simulation optimization method for scheduling problem in container terminals is proposed. Meanwhile, methods to improve the computation efficiency of simulation optimization are designed.

3. Integrating scheduling model for containers terminals

In this section, an integrating model is developed in order to (1) coordinate the scheduling of various types of equipments simultaneously to achieve a high level integrated optimization and (2) minimize the makespan or time being taken to load or unload a given set of outbound containers. Firstly, we make a brief description of the hybrid flow shop scheduling (HFSS) problem; then the scheduling model for loading outbound containers is formulated based on HFSS.

3.1. Hybrid flow shop scheduling problem

The HFSS problem can be stated as follows. Consider a set $J = \{1, 2, \ldots, n\}$ of n jobs, each is to be processed in S consecutive stages. Stage s has a set of $M(s)$ identical machines, with $m_s = |M(s)|$, $s = 1, 2, \ldots, S$. In each stage s, there are $m_s \geqslant 1$ parallel identical machines, with $m_s \geqslant 2$ for at least one stage, $s = 1, 2, \ldots, S$.

Let p_{is} be the processing time of job i at stage s. Each machine can process only one job once. Since all machines at each stage are identical and preemptions are not allowed, to define a schedule, it suffices to specify the completion times for all tasks. Let C_{is} be the ending time of the sth stage of job i. Therefore, the HFSS is to find a schedule to minimize the maximum completion time $C_{\max}$, with $C_{\max} = \max C_{is}$.

The HFSS has recently received attention because of its importance from both theoretical and practical point of views. Lin and Liao [25] presented a case study in a two-stage hybrid flow shop with sequence-dependent setup time and dedicated machines. Kurz and Askin [26] explored three kinds of heuristics for flexible flow lines with sequence-dependent setup times. Jin et al. [27] discussed the three stages of HFSS; and proposed two metaheuristic algorithms first sequence and then allocate jobs to machines based on a particular partition of the shop. Chen [17] developed an integrated model to schedule different equipments in container terminals based on HFSS. In his model, the loading and unloading operations are considered simultaneously. However, at present, most of the container terminals used the method “scheduling loading and unloading, respectively”. In this method, when a ship berths terminal, loading is processed when all the unloading operation is finished. If loading and unloading operations are processed simultaneously in the same ship bay, it

is denoted "crane double cycling problem" (Goodchild and Daganzo [5]). The scheduling model for "crane double cycling problem" is different from models for current "scheduling loading and unloading simultaneously" problem. And the "crane double cycling" method is only in experiment. The model developed by Chen is for "scheduling loading and unloading, respectively" problem, and it is difficult to be used to realize loading and unloading simultaneously in present scheduling practice.

The objective of our paper is to develop a new method (simulation optimization method) to obtain the optimal scheme for operation scheduling in container terminals. And considering the loading operation or unloading operations, respectively, is in accord with present scheduling practice. Although we focus on the loading operation; the proposed method can also be used to unloading operation. And further study is to use the proposed method to "crane double cycling problem" which considers loading and unloading operations.

3.2. Scheduling problem for loading outbound containers

Container terminal operations can be divided into two parts: loading outbound containers and unloading inbound ones. E.g., the process of loading outbound containers involves three stages: YCs pick up the desired containers from yard blocks and load them onto the yard trailers, then YTs transport the containers to QCs, finally the QCs load the containers onto the vessels.

The loading process in container terminals is similar to HFSS problem. For loading outbound containers, each container must undergo three handling operations: a transfer operation within storage yard, a transfer operation of a container onto the ship and a transfer operation between QCs and YCs. And there are three different sets of machines: QCs, YCs and YTs. Thus, a job can be defined as a complete loading process for a container. While, comparing with the classical HFSS problem, our problem has several unique characteristics as follows.

3.2.1. Job precedence constraints

E.g. for loading, containers in the hold must precede the containers on the deck of the same vessel.

3.2.2. Blocking

There is limited or no buffer between two successive machines, therefore, blocking happens when the buffer is full. E.g. when the YT carries a container to a QC that is handling another container, the YT has to wait for the QC.

3.2.3. Setup times

In container terminals, there is empty movement when a crane or YT moves between two jobs. For example, once a YT carries an outbound container to a QC, it has to make an empty trip to the storage yard in order to proceed next container. We denote it as *setup time*.

Based on above analysis, the scheduling problem for loading outbound containers can be formulated as HFSS problem. The objective is (1) to assign each operation to a machine and (2) to sequence the assigned operations on each machine, thus to minimize the makespan of the loading operations.

3.3. Model formulation

In order to formulate the scheduling problem for loading containers, the following parameters and decision variables are defined:

Problem parameters:

N	the set of all containers (jobs)
n	the number of containers
i,j	container index
s	stage index, $s=1,2,3$
m	machine index
m_s	the number of machines at stage s
M_{is}	the set of machines to process container i at stage s
E_m	the set of containers that might be processed on machine m
B	the set of pairs of containers between which there is precedence relationship, when container i must precedence container j
p_{is}	processing time of container i at stage s
w_{ijs}	setup time between container i and j at stage s
G	a sufficiently large constant

Decision variables:

x_{ism}=1, if operation of container i at stage s is assigned to machine m; 0, otherwise;

$y_{ijsm}=1$, if operation of container i and j at stage s are assigned to the same machine m; 0, otherwise;

$z_{ijsm}=1$, if operation of container i immediately precedes j on machine m at stage s;

t_{is} is the starting time of container i at stage s;

$C_{\max}$ is the completion time of the last container.

The scheduling model can be formulated as follows:

$$\text{Min} \quad C_{\max}=\max_{i,s}(t_{is}+p_{is}) \tag{1}$$

$$\text{s.t.} \quad \sum_{m\in M_{is}} x_{ism}=1, \quad \forall i\in N,\ \forall s\in\{1,2,3\} \tag{2}$$

$$t_{is}\geqslant 0, \quad \forall i\in N,\ \forall s\in\{1,2,3\} \tag{3}$$

$$t_{is}+p_{is}\leqslant t_{i(s+1)}, \quad \forall i\in N,\ \forall s\in\{1,2,3\} \tag{4}$$

$$y_{ijsm}=y_{jism}, \quad \forall i,j\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{5}$$

$$y_{ijsm}\leqslant 0.5(x_{ism}+x_{jsm})\leqslant y_{ijsm}+0.5, \quad \forall i,j\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{ij} \tag{6}$$

$$\sum_{j\in E_m} z_{ijsm}\leqslant 1, \quad \forall i\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{7}$$

$$\sum_{j\in E_m} z_{jism}\leqslant 1, \quad \forall i\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{8}$$

$$z_{ijsm}+z_{jism}\leqslant 1, \quad \forall i,j\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{9}$$

$$x_{ism}-0.5\leqslant 0.5(z_{ijsm}+z_{jism})\leqslant x_{ism}, \quad \forall i,j\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{10}$$

$$t_{i(s+1)}+w_{ijs}\leqslant t_{js}+H(1-z_{ijsm}), \quad \forall i,j\in E_m,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{11}$$

$$t_{is}\leqslant t_{js}, \quad \forall i,j\in B,\ \forall s\in\{1,2,3\} \tag{12}$$

$$x_{ism},y_{ijsm},z_{ijsm}=0 \text{ or } 1,\ \forall i,j\in N,\ \forall s\in\{1,2,3\},\ \forall m\in M_{is} \tag{13}$$

The objective function (1) is to minimize the makespan. Constraints (2) guarantee that each operation must be processed by exactly one machine. Constraints (3) ensure that each operation begins after time zero. Constraints (4) ensure that the order of operation for each container is respected. Constraints (5) and (6) ensure that $y_{ijsm}=y_{jism}=1$ when $x_{ism}=x_{jsm}=1$. Constraints (7) and (8) ensure that each operation has at most one predecessor and successor on machine m. Constraints (9) ensure that z_{ijsm} and z_{jism} cannot equal to 1 simultaneously. Constraints (10) ensure that $x_{ism}=1$ when $z_{ijsm}+z_{jism}=1$. Constraints (11) determine the starting time of the operation and define the blocking constraints. Constraints (12) determine the set of pairs of jobs between which there is precedence relationship. Constraints (13) are binary constraints.

It is well known that the HFSS problem is NP-hard, thus, scheduling problem for loading or unloading containers is also NP-hard. It is doomed unable to obtain optimal solutions for large-scale problems.

Hence, heuristic algorithms are wildly used to obtain near-optimal solutions efficiently. However, because of the numerous constraints, it is difficult to evaluate a scheduling scheme in the process of heuristic algorithms. In addition, because of the blocking factor, it is essential to calculate the waiting and delay time for QCs, YTs and YCs, which is time consuming.

Meanwhile, although many constraints are considered in above scheduling model, it is too complex to model all the constraints analytically. Also uncertainty is difficult to tackle by analytical model alone. Therefore, this paper handled these problems through the integration of simulation and optimization.

4. Simulation optimization

4.1. Framework of simulation optimization

Many complex systems such as manufacturing, supply chain, and container terminals are too complex to be modeled analytically. Discrete event simulation has been a useful tool for evaluating the performance of such systems. However, simulation can only evaluate a given design, not provide more optimization function. Therefore, the integration of simulation and optimization is needed. Simulation optimization is the process of finding the best values of some decision variables for a system where the performance is evaluated based on the output of a simulation model of this system (Olafsson [28]).

As shown in Fig. 1, the simulation optimization for scheduling loading containers consists of two relative unattached modules, namely, optimization module and simulation module. This integrating method realizes the separation of optimization algorithm from model, which cannot only improve the universalization of optimization algorithm, but also help for the software integration.

When the loading operations need to be optimized, simulation module must be built first, and then the optimization module is created to optimize the parameters of simulation module. The setup of input and output variables, such as initial solution, constraints of decision variables, objective function, and repetition time of simulation in simulation module is done in optimization module. As optimization module running, optimization algorithm would create a set of feasible design variables, and transfer them to the simulation module by data interface; then the simulation module applies these variables to reconfigure the simulation parameters in real time, and run the simulation, then simulation results are returned to the optimization module; according to the results, the algorithm adjusts the direction of optimal solution searching and creates a new set of feasible solutions. The process would repeat until the stop criterion is satisfied. Finally, optimized design variables and best scheduling scheme are output.

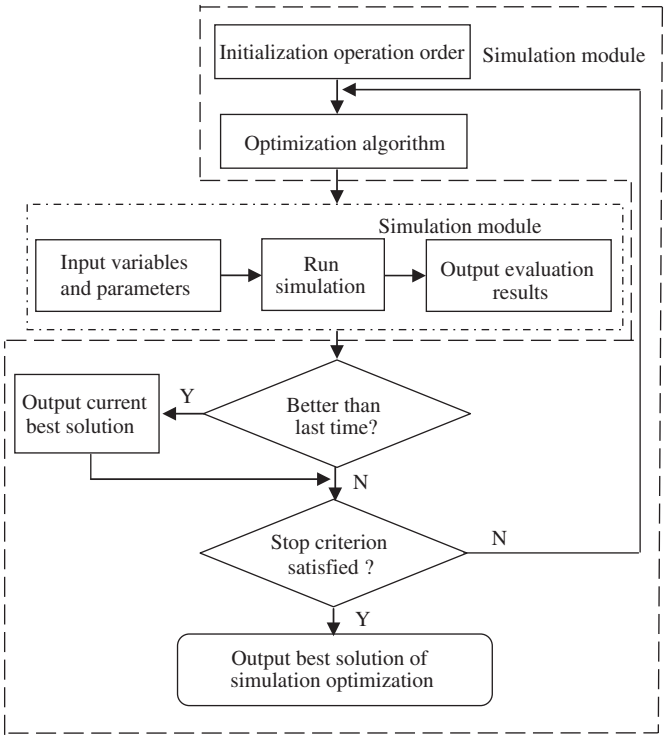

Fig. 1. Framework of simulation optimization.

4.2. Integration environment

An integration environment is needed to integrate simulation module and optimization module. In this paper, Visual Basic is used as integration environment and Arena7.0 is used as simulation platform. Arena combines the simplification of high layer simulator and flexibility of simulation language. Also, it can integrate with Visual Basic and Visual C++, which enhances the modeling capability. The optimization algorithm is coded with Visual Basic 6.0. The controlling of Arena simulation module is realized by Visual Basic, which can modify simulation parameter when Arena is operating, and control the running of Arena module. The process is shown in Fig. 2.

Step 1: Define an Arena application and activate it.

Step 2: Open an Arena project file, instantiate it in Visual Basic program.

Step 3: Find the required control module, instantiate and operate them in program, and return the results to Arena model.

Step 4: Run the Arena model.

Step 5: Deal with the results in Visual Basic program.

4.3. Simulation module

Simulation module is used to (1) evaluate the schedule obtained by the optimization module and (2) or to construct a schedule considering all the constraints.

In simulation, parts of the parameters of simulation model are controlled by VB program, e.g., the attributes of each container (storage location, transport time, container types), the loading or unloading sequence of each containers, and the running and stopping criterion of simulation model. Other attributes such as the operation efficiency of QCs, YCs, and YTs are pre-determined by defining attributes of each module when developing simulation model.

Arena7.0 is used to develop simulation model. To facilitate the modeling process, the modeling tools of Arena provide abundant panel for users, and many modules are incorporated in the panels. Modules are the basic components of Arena model, which can be classified into flowchart module and data module. We use flowchart module to describe the dynamic process of an entity from start-point to end-point; and use data module to define the attribute of each entity, variables and expressions of the simulation model.

Two-dimension simulation model is developed by Arena. To improve the running speed of simulation, the entities in model are not endowed with pictures; and the simulation interface is not displayed when running simulation model. Meanwhile, the logic relations among module are simplified as possible.

In simulation module, three rules, namely FAM (first available machine), LFM (last available machine), and FCFS (first come first served) can be used. According to the FAM rule, the container is assigned to the first available machine, i.e., the machine that finishes the previous job first. According to the LFM rule, the container is assigned to the last available machine, i.e., the machine with the smallest idle time among all available machines. The rationality of LFM

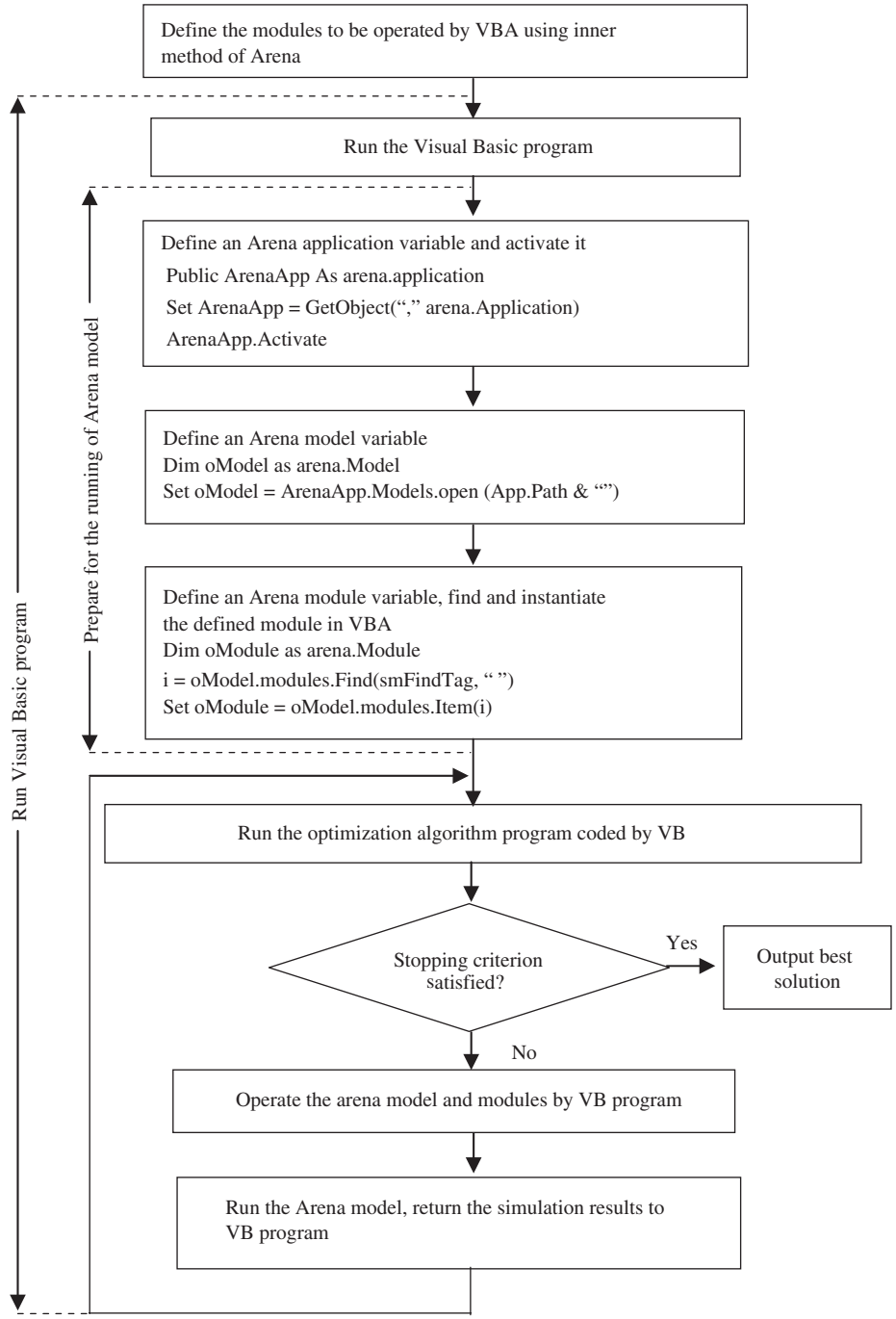

Fig. 2. Integration of Arena simulation model and optimization algorithm by VB.

rule is to improve the future assignment of the currently unscheduled jobs when two or more machines provide the same length of partial schedules. Because our scheduling problem has no intermediate storage, those containers whose handling operations finished earlier will be delivered to the next stage as soon as possible. The FCFS rule is applied when building the container sequences in the stages other than the first stage. Therefore, the two strategies to select dispatching rules for different stages are: (1) FAM–FCFS–FCFS and (2) LFM–FCFS–FCFS. After comparison by simulation tests we find that LFM–FCFS–FCFS is better. Therefore, the dispatching of the three stages are LFM, FCFS and FCFS, respectively.

4.4. Simulation optimization algorithm

Five available approaches are being used to simulation optimization, namely, gradient-based approach, response surface methodology, random search, statistical selection and metaheuristic algorithms. Among them, metaheuristic algorithms develop rapidly. Various metaheuristic algorithms have been used to simulation optimization, such as genetic algorithms (GAs), simulated annealing, tabu search and neural networks (NNs). Although these methods are generally designed for combinatorial optimization in deterministic context and may not guarantee convergence, they have been quite

successful when applied to simulation optimization, especially in some commercial simulation software.

GA searches the solution space by building and evolving a population of solutions. The evolution is achieved by means of creating new solutions from two or more solutions in current population. The main advantage of genetic algorithm over those based on sampling the neighborhood of a single solution (e.g., simulated annealing and tabu search) is that it can explore a larger area of solution space with smaller number of objective function evaluations. In simulation optimization method, evaluating the objective function needs running the simulation model, thus finding good solutions early in the search is important. Therefore, in this paper, the simulation optimization is implemented based on GA. And to improve the computation efficiency, a hybrid algorithm based on GA and NN is designed in Section 5.

5. A hybrid algorithm for simulation optimization

5.1. Outline of solution procedure

To improve the computation efficiency of simulation optimization, we design surrogate model based on NN to filter out obvious potential bad solutions. Running simulation module is computationally expensive in simulation optimization method, therefore, by surrogate model, the running time of simulation model may decrease, thus the computation efficiency of simulation optimization will be improved. E.g., container vessels are typically divided longitudinally into holds that open to the deck through a hatch. In practice, to improve the operation efficiency of QCs, a QC can move to another hold until it completes the current one. Therefore, if two containers in different holds are continuous in QC operation sequence, it is a potential bad solution. The potential bad solution can be filtered out by trained NN.

The process is shown as Fig. 3. At the beginning of the simulation optimization, there are no data available to train the NN; however, with the search processing, data become available because new trial solutions are evaluated by running the simulation model. If the NN is trained enough, the prediction function of NN is triggered.

Let $f(x)$ be the objective function associated with solution x. Also let $f'(x)$ be the predicted objective function value for a solution x. x^* is the best solution found so far in the search. If $f'(x) - f(x^*) > d$, discard x; else, run simulation model to obtain objective function $f(x)$. d is a parameter to trade off speed and accuracy of the search. With the increase of d, the convergence of the algorithm accelerates, but the accuracy decreases.

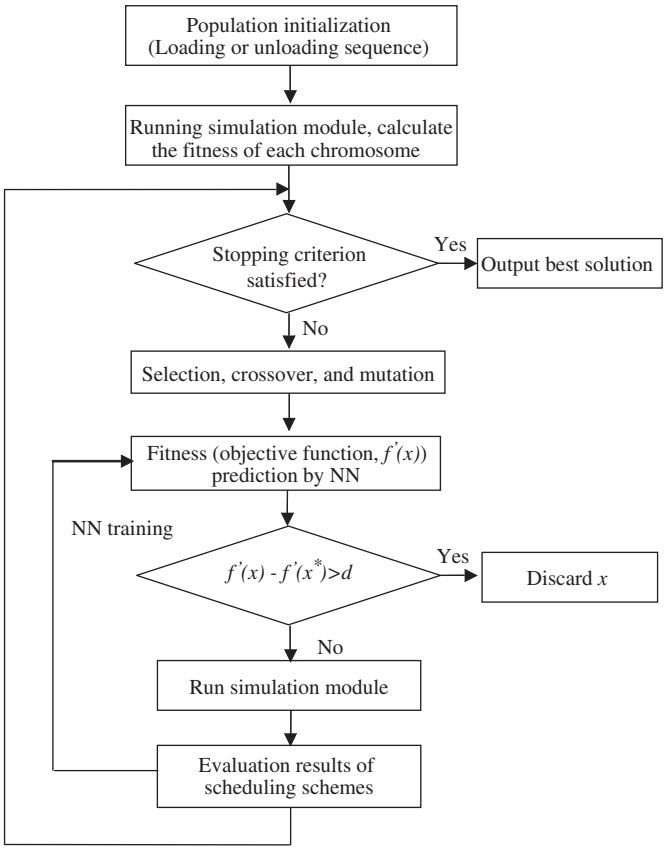

Fig. 3. Simulation optimization method of NN-based surrogate model.

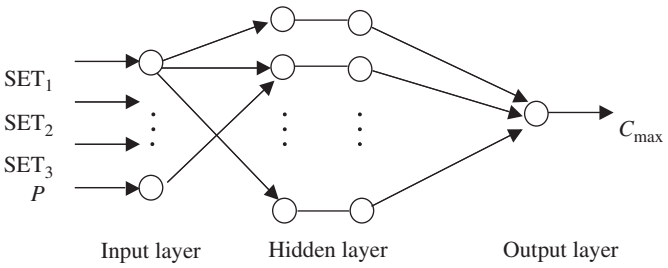

Fig. 4. Structure of BP neural network.

5.2. NN design

NNs are powerful tools for approximation of unknown nonlinear functions and have gained wide applications in a variety of fields (Lawrence [29]). It is able to learn underlying relationship from a collection of training samples. In this paper, error backpropagation (BP) NN is used (Fig. 4), which includes input layer, hidden layer and output layer. The learning process includes forward and backward propagation. Training the BP network is to minimize the error function by determining the weights and neuron thresholds. Once the NN has been trained, it can be used to predict the unknown output of some input.

To determine input neurons, we first define the process time and setup time first. Let P denote the total process time of all containers; w_{ijs} denote setup time between container i and j at stage s; SET_s denote the total setup time at stage s

$$P = \sum_{i=1}^{N} \sum_{s=1}^{3} p_{is} \tag{14}$$

$$\mathrm{SET}_s = \sum_{i,j \in N} w_{ijs}, \quad i<j, \ \ s = 1,2,3 \tag{15}$$

In loading and unloading process, the total operation time is mainly influenced by total process time of different equipment and setup time. Therefore, the setup time at each stage and the process time are treated as input neuron, namely, SET_1, SET_2, SET_3 and P.

The output neuron is $C_{\max}$; the number of neuron in hidden layer is seven; the learning rate parameter is 0.1, and the accuracy parameter is 0.02; Levenberg–Marquardt backward propagation function is used to train the NN. After 2000 iterations training by 60 sets data, the prediction and filtering function of NN is triggered.

5.3. Encoding and initialization of GA

Instead of using the classical binary bit string representation, the chromosomes are represented as character strings. In this representation the chromosomes are a string, length equal to the number of

containers, of integers with each number occurring only once, i.e., each chromosome contains all integers from one to the number of containers exactly once. If a string is 3–5–6–1–7–9–2–8–10–4, this means that the loading sequence is 3, 5, 6, 1, 7, 9, 2, 8, 10, 4.

Initial solution could be a solution obtained from a heuristic or generated randomly. Since a random solution may not satisfy good performances, two heuristic algorithms are used to generate the initial solutions for the genetic algorithm. These heuristics are: LPT and SPT. SPT rule sequences the jobs in non-decreasing order of the total processing times. And LPT rule sequences the jobs in decreasing order of the total processing times. Firstly, an initial solution is obtained by LPT or SPT rule, then, select two containers randomly and swap their position in chromosome strings. By this method, an initial population with M individuals is obtained.

5.4. Calculation of fitness value

In the simulation optimization process, GA is used to create loading order of outbound containers first. And then the order is inputted in simulation model. In simulation model YCs, YTs and QCs are dispatched to container according to certain rules (the dispatching of the three stages are LFM, FCFS and FCFS, respectively). Finally, the loading time can be obtained by running simulation model. Therefore, simulation cannot only be used to evaluate the loading order obtained by GA, but also be used to construct a schedule considering all the constraints.

The container loading problem is a minimization problem, thus the smaller the objective function value, the higher the fitness value must be. To maintain the variety of chromosome by crossover, the exponential and sigmoid functions can be used. We define the fitness function as

$$f(x) = 1/(1 + \exp(y(x)/1000)) \tag{16}$$

5.5. Genetic algorithm operators

The underlying fundamental mechanism of GA consists of three main operators: reproduction, crossover and mutation.

5.5.1. Reproduction

Reproduction is a process in which individual chromosomes are copied according to their scaled fitness function values. Chromosomes with higher fitness value would be selected with higher probabilities. The selection probability is as following formulation:

$$p(x_i) = \frac{f(x_i)}{\sum_{i=1}^{M} f(x_i)}, \quad i = 1, 2, \ldots, M \tag{17}$$

5.5.2. Crossover operator

A crossover operator called single point with repairs is designed. This operator first chooses a point of one parent and inserting the segment to the left of this into the first offspring in the same order. The remaining positions are filled from the other parent, select the gene from left to right of the chromosome following the rule that gene cannot be identical. Fig. 5 shows an example where new children are created by crossover. The crossover point is chosen at 5. The first five genes of offstring 1 are inherited from parent 1; and remaining genes are inherited from parent 2.

5.5.3. Mutation

Mutation introduces random changes to the chromosomes by alter the value to a gene with a user-specified probability p_m called mutation rate. In our application, we generate two random numbers between 1 and the string length. The values of a gene at these two positions are interchanged.

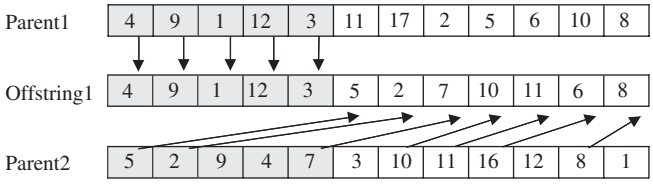

Fig. 5. Example of crossover processing.

5.6. Stopping criterion

To compare the computation efficiency of different algorithms, different stopping times are set as stopping criterion. After a predetermined stopping time, the algorithm ends.

6. Lower bound

To determine the effectiveness of the developed method, we need to compare the makespan obtained by the proposed metaheuristics algorithms with the optimal one obtained by the solution of above mixed-integer program. However, finding the optimal makespan requires the solution of a mixed-integer program, which can be quite time consuming even for medium-sized problems. Therefore, the optimal solution is measured against well-defined lower bounds.

Lower bound problem is widely discussed in HFSS problem. Here, we derive the lower bound on the optimal makespan based on the method proposed by Stantos (1995) and the method designed by Chen (2007).

Let $\mathrm{LS}(i,s)$ denote the left-hand side sum of processing time from stage 1 to $s-1$ for job i, and $\mathrm{RS}(i,s)$ the right-hand side sum of processing time from stage $s+1$ to S for job i. $\mathrm{LS}(i,s)$ and $\mathrm{RS}(i,s)$ are given by

$$\mathrm{LS}(i,s) = \begin{cases} \sum_{l=1}^{s-1} p(i,l), & s > 1 \\ 0, & s = 1 \end{cases} \tag{18}$$

$$\mathrm{RS}(i,s) = \begin{cases} \sum_{l=s+1}^{S} p(i,l), & s < S \\ 0, & s = S \end{cases} \tag{19}$$

$\mathrm{JL}(k,s)$ is the kth value in the ascending order list of $\mathrm{LS}(i,s)$ for all jobs in stage s, and $\mathrm{RL}(k,s)$ is the kth value in the ascending order list of $\mathrm{RS}(i,s)$ for all jobs in stage s. Based on these denotations, Stantos proposed stage-based lower bounds, lb_s and global lower bound, glb for HFSS problem as follows:

$$\mathrm{lb}_s = \frac{1}{m_s}\left[\sum_{i=1}^{m_s} \mathrm{LS}(i,s) + \sum_{i=1}^{n} p(i,s) + \sum_{i=1}^{m_s} \mathrm{RS}(i,s)\right], \quad s = 1, 2, \ldots, S \tag{20}$$

$$\mathrm{glb} = \max_{0 \leqslant s \leqslant S} \{\mathrm{lb}_s\} \tag{21}$$

During the container loading process, a setup time is needed after each operation is finished. Therefore, the makespan lower bound must account for these setup times. Considering that the container sequence for each machine and the exact values of the setup times are not known, we develop a notation $\mathrm{SET}_s^{\mathrm{L}}(i)$ to obtain the minimum possible setup time for each container.

For each container i, let $\mathrm{SET}_s^{\mathrm{L}}(i)$ denote the minimum time required to set up the container immediately preceding container i at stage s, w_{ijs} denote setup time between container i and container j at stage s.

$$\mathrm{SET}_s^{\mathrm{L}}(i) = \min_j w_{ijs} \tag{22}$$

Therefore, the stage-based lower bounds, LB_s and global lower bound, LB for scheduling problem for loading outbound containers can be formulated as follows:

$$LB_s = \frac{1}{m_s}\left[\sum_{i=1}^{m_s} LS(i,s) + \sum_{i=1}^{n} p(i,s) + \sum_{i=1}^{m_s} RS(i,s) + \sum_{i=1}^{n-m_s} SET_s^L(i)\right], \quad s = 1,2,3 \tag{23}$$

$$LB = \max_{1 \leqslant s \leqslant 3}\{LB_s\} \tag{24}$$

7. Numerical experiments

7.1. Validity of the simulation optimization method

Numerical test are given to assess the solution quality and efficiency of the developed simulation optimization method. And two scheduling method, namely "single-crane oriented" and "multi-crane oriented", are compared.

In "single-crane oriented" method, a set of YTs is usually assigned to a specific QC until the work is completed, the YTs return to the QC directly after finishing a delivery task. And the YTs cannot be shared by different QCs. "Single-crane oriented" method is simple to be implemented, but it leads to the increment of empty travel of YTs. In "multi-crane oriented" method, instead of being assigned to a specific QC, YTs are shared by different QCs, thus can be dispatched accord to the real-time operation task of QCs. Therefore, the "multi-crane oriented" method can reduce the needed trailers without increasing the loading and unloading time of the vessel.

Firstly, "single-crane oriented" case is used to test the validity of the developed simulation optimization method. Data generator is developed to produce instance sets with specific characteristics. The parameters for terminal layout are based on the data received from Port of Dalian. Details are as follows:

- Loading process is considered.
- The available QCs are 3; each QC dispatched 3 YTs and 2 YCs.
- The processing time of QCs are generated from uniform distribution of $U(100,150)$ s, and the processing time of YCs follows the uniform distribution of $U(260,320)$ s.
- Storage location in yard for each container is selected randomly. The storage location determines the processing time for YTs.
- The setup times in stages 1 and 2 are determined by the storage location for each container In stage 1, the setup time is obtained by calculating the move time of YC between the locations of two containers In stage 2, the setup time of two containers is obtained by calculating the travel time of YT from the shipside to the location of the second container in container yard after it has transported the first container to shipside. The setup times in stage 3 depend on the ship stowage plan. It can be obtained by calculating the move time of QC between the locations of two containers on ship.

Four GA-based simulation optimization algorithms are compared in terms of both solution quality and efficiency which can be defined as

(1) GN-LPT: Initialize container sequence according to LPT rule, improve the sequence by GA, and predict objective function and filter out bad solution by NN.
(2) GN-SPT: Initialize container sequence according to SPT rule, improve the sequence by GA, and predict objective function and filter out bad solution by NN.
(4) GA-LPT: Initialize container sequence according to LPT rule, improve the sequence by GA.

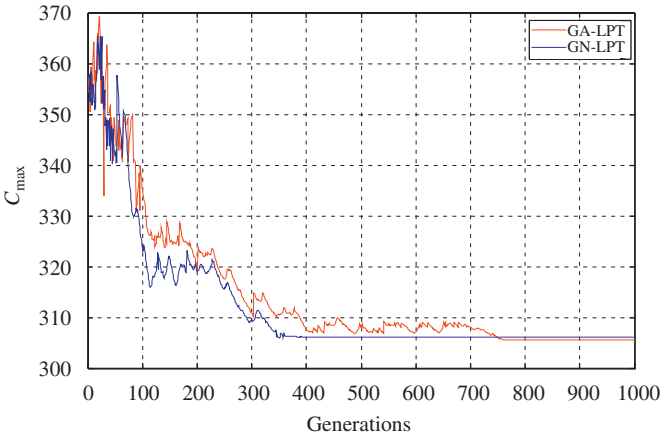

Fig. 6. Convergence of the proposed algorithms.

(5) GA-SPT: Initialize container sequence according to SPT rule, improve the sequence by GA.

The makespan obtained by simulation optimization method is compared with the global lower bound. The relative deviation (RD) is calculated by the following formula:

$$RD = \frac{C_{max} - LB}{LB} \times 100 \tag{25}$$

where C_{max} is the makespan obtained by the developed algorithms and LB is its lower bound.

The number of containers is 400. In the test, the population size is 50. The algorithms and modules are run on personal computer with Pentium IV 1.7 GHZ processor and 512 MB random access memory. To examine the convergence speed of the proposed algorithms, we run the four algorithms 10 times, respectively. Fig. 6 reveals the average values of the total loading times changing with the generation.

From Fig. 6 we can find that both of GN-LPT and GA-LPT can reach convergence. However, the needed generation to convergence of GN-LPT is less than that of GA-LPT. GN-LPT can reach convergence within 350 generations, while GA-LPT reached convergence in 750 generations. Meanwhile, the solution quality of GA-LPT is better than GN-LPT. This is because that in GA-LPT, NN filter out potential bad solution which reduces the search space of GA, thus accelerate the convergence speed. And the filtering of potential bad solutions may also filter out potential good solutions.

Then, the genetic search procedure is run for fixed time periods of 60, 90, 120 min, respectively, and Table 1 summarizes the results.

Table 1 show that all of the four methods can improve the initial solutions greatly. When the search time is fixed to 60 or 90 min, the makspans obtained by GN-LPT and GN-SPT are less than that of GA-LPT and GA-SPT. And when the search time is fixed to 120 min, there is no obvious difference between GN-LPT and GA-LPT. This is because that the prediction and filter function of NN is incorporated into GN-LPT and GN-SPT to decrease the time to calculate objective function and the computation time, thus the convergence times of GN-LPT and GN-SPT is less than that of GA-LPT and GA-SPT.

As to RD, there is no obvious difference between GN-LPT, GN-SPT and GA-LPT, GA-SPT. This indicates that the incorporation of NN does not influence the accuracy. Therefore, the proposed hybrid algorithm can greatly decrease the computation time without decreasing the solution accuracy. In addition, GN-LPT performs better than GN-SPT, and GA-LPT performs better than GA-SPT, which indicates that LPT heuristics performs better than SPT heuristics.

Furthermore, six scenarios are designed according to the number of being handled containers. Using these scenarios, GN-LPT and GN-

Table 1
Results obtained by different simulation optimization algorithms.

Optimization algorithms	Initial solution (C_{max})	Solution obtained by simulation optimization			RD (%)
		Search time 60 min	Search time 90 min	Search time 120 min	
GN-LPT	357.16	306.40	306.14	306.09	4.19
GN-SPT	368.83	307.27	306.93	306.84	4.45
GA-LPT	357.16	326.74	310.84	305.57	4.01
GA-SPT	368.83	331.56	314.02	306.48	4.32

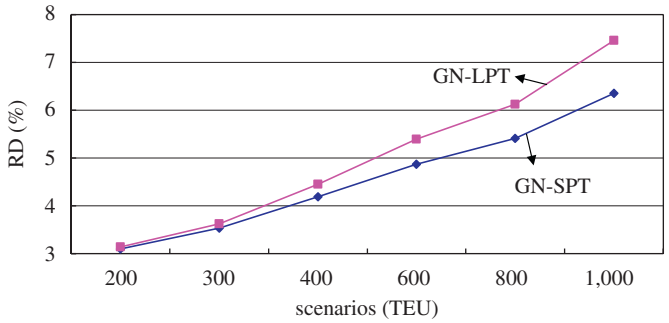

Fig. 7. Relative deviation (RD) obtained by GN-LPT and GN-SPT methods.

Table 2
Comparison of "single-crane oriented" with "multi-crane oriented" method.

Scheduling method	Makespan (C_{max})	QCs utilization (%)	YTs utilization (%)	Convergence time (min)
Single-crane oriented	306.40	90.40	72.87	60
Multi-crane oriented	290.18	95.33	83.54	110

SPT are compared. The running time of simulation optimization is fixed to 90 min. Results (Fig. 7) indicates that with the increment of containers, the difference of RD between GN-LPT and GN-SPT increases. It indicates that initial solution becomes more important as the problem size increment.

7.2. Comparing "single-crane oriented" method with "multi-crane oriented" one

"Single-crane oriented" and "multi-crane oriented" scheduling methods are compared. Suppose that GNLPT method is used and the number of containers is 400. Results are given in Table 2.

Table 2 indicates that "multi-crane oriented" scheduling method can decrease the makespan and increase the YTs utilization. This is because that YTs can be shared by different QCs in "multi-crane oriented", thus it can decrease the waiting time of QCs and YTs. The comparison of these two policies has been studied also in Chen's paper (named static and dynamic) and similar conclusions were already presented. This also indicates the our simulation optimization can be used to solve the two scheduling problems.

In addition, the convergence time of "multi-crane oriented" method is more than that of "single-crane oriented" method, which indicates that "multi-crane oriented" method is more complicated than "single-crane oriented" method.

7.3. Comparing simulation optimization method with existing method

Presently, two methods, namely optimization and simulation, are widely used in scheduling problem of container terminals. Optimization method aims to obtain optimal scheduling scheme by search the solution space. Simulation model can be used to evaluation certain scheduling scheme or dispatching rules. And method proposed in this paper integrates optimization method with simulation. Therefore, to demonstrate the benefits, we compare our method with two existing methods.

Table 3
Comparison of simulation optimization with method proposed by Chen [16].

n	Problem size			RD (%)		CPU times (s)	
	QC	YC	YT	GN-LPT	TA1	GN-LPT	TA1
40	2	6	8	0.92	0.38	65.0	46.7
50	2	6	8	1.20	0.48	103.2	95.2
80	2	6	8	1.49	1.35	268.4	176.0
100	2	6	10	1.74	2.14	423.1	387.2
200	4	10	16	2.63	4.01	1053.6	635.4
400	4	10	16	3.35	6.17	2321.7	1764.2
500	4	10	16	4.07	8.04	2930.4	2846.5

Firstly, experiments are conducted to compare our method GN-LPT with method proposed by Chen [17]. Chen proposed two algorithms based on tabu search, namely, tabu search algorithm with MIH_ND initial solution heuristic (TA1), and tabu search algorithm with MIH_MET initial solution heuristic (TA2). In this paper, we compare our method with TA1. Results are given in Table 3.

From Table 3, we can find that RD obtained by GN-LPT is more than that of TA1 when the container number is small (40, 50, 80), and when the container number is large, RD obtained by GN-LPT is less than that of TA1. This is because that TA1 method tries to optimize the operation order of all three stages; but GN-LPT method only optimizes the loading order, the operation order in each stage is determined by dispatching rules. Therefore, with the increase of container number, the solution quality of TA1 decreases. Moreover, simulation optimization method considers the uncertainty factor in operation process, thus the results of simulation optimization method is more accurate. On the other hand, the computation time of our method is more than that of TA1. Although our method does not intend to find the optimal operation sequence of each machine, the running of simulation is computationally expensive which increase the computation time of simulation optimization method.

From the seven data sets in Table 3, we can calculate that the average ratio of time by optimization and simulation is 13.5% and 86.5%, respectively. The number of iterations between optimization module and simulation module of the seven data sets is 160, 200, 250, 250, 300, 350 and 400, respectively.

Furthermore, we obtain near optimal scheduling scheme by optimization algorithm designed by Chen ([17], TA1), and then input the scheme to simulation model. By running simulation model, the accurate loading or unloading times of a scheduling scheme can be obtained. We compare these results with those of our proposed method. Results are given in Table 4.

From Table 4, the loading or unloading time obtained by optimization algorithm (e.g., TA1) are more that of simulation algorithm. For three-stage hybrid flow shop problem, evaluating of loading or unloading time is difficult due to the complex constraints and uncertain factors, thus in the process of optimization algorithm, the objective function is obtained by approximation calculation without considering the blockage and setup time and the scheme obtained by optimization algorithm (TA1) cannot ensure the minimization of

Table 4
Comparison of loading/unloading time obtained by different method.

n	Problem size			The deviation of loading/unloading time with lower bound (%)	
	QC	YC	YT	Optimization algorithm (TA1)	Simulation optimization (GN-LPT)
40	2	6	8	1.24	0.92
50	2	6	8	1.41	1.20
80	2	6	8	1.53	1.49
100	2	6	10	2.29	1.74
200	4	10	16	4.37	2.63
400	4	10	16	5.96	3.35
500	4	10	16	7.96	4.07

loading or unloading time. In the process of simulation optimization, the accurate loading or unloading time of a scheme can be obtained by simulation model. Therefore, the simulation algorithm can improve the solution quality comparing to single optimization algorithm or simulation.

8. Conclusions

In this paper, scheduling problem for loading or unloading containers in container terminals was discussed. Combining the simulation analysis and the optimal decision-making mechanism, a simulation optimization method was proposed, this method integrates optimization algorithms, simulation model, and job partition rule. Meanwhile, to improve the computation efficiency of simulation optimization, an NN-based surrogate model was designed. Numerical tests were provided to illustrate the efficiency of the proposed method. Numerical tests show that simulation optimization method can solve the scheduling problem of container terminals efficiently. And NN-based surrogate model can improve the computation efficiency of simulation optimization. In addition, initialization methods (such as LPT and SPT) are important for the scheduling problem and helpful for the improvement of the solution.

The proposed model involves the problem of determining the loading or unloading sequence, scheduling and dispatching various kind of equipment (QCs, YCs, YTs) simultaneously. Therefore, comparing with models optimizing different equipments, respectively, it can improve the coordination among different equipments and enhance the integration of operation scheduling in container terminals. Simulation optimization method considers the uncertain and stochastic factor in operation process; also it can deal with complex constraints in scheduling model. Therefore, simulation optimization method can tackle the practical scheduling problem efficiently. The main disadvantage of the simulation optimization is the long computation time.

References

[1] Daganzo CF. The crane scheduling problem. Transportation Research Part B 1989;23:159–75.
[2] Kim KH. A crane scheduling method for port container terminals. European Journal of Operational Research 2004;156:752–68.
[3] Lee D-H, Cao Z, Meng Q. Scheduling of two-transtainer systems for loading outbound containers in port container terminals with simulated annealing algorithm. International Journal of Production Economics 2007;107:115–24.
[4] Lee D-H, Wang HQ, Miao L. et al. Quay crane scheduling with non-interference constraints in port container terminals. Transportation Research Part E 2008;44:124–35.
[5] Goodchild AV, Daganzo CF. Crane double cycling in container ports: planning methods and evaluation. Transportation Research Part B 2007;41:875–91.
[6] Evers JM, Koppers AJ. Automated guided vehicle traffic control at a container terminal. Transportation Research Part A 1996;30:21–34.
[7] Liu C-I, Ioannou PA. A comparison of different AGV dispatching rules in an automated container terminal. In: The IEEE fifth conference on intelligent transportation systems, Singapore; 2002. p. 880–5.
[8] Vis IFA. Minimum vehicle fleet size under time-window constraints at a container terminal. Transportation Science 2005;39(2):249–60.
[9] Kim KY, Kim KH. A routing algorithm for a single straddle carrier to load export containers onto a containership. International Journal of Production Economics 1999;59:425–33.
[10] Kim KH, Kim KY. Routing straddle carriers for the loading operation of containers using a beam search algorithm. Computers & Industrial Engineering 1999;36:109–13.
[11] Nishimura E, Imai A, Papadimitriou S. Yard trailer routing at a maritime container terminal. Transportation Research Part E 2005;41:53–76.
[12] Zhang C, Wan Y-W, Liu J. Dynamic crane deployment in container storage yards. Transportation Research Part B 2002;36:537–55.
[13] Linna R, Liu J-Y, Wan Y-W. Rubber tired gantry crane deployment for container yard operation. Computers & Industrial Engineering 2003;45:429–42.
[14] Kim KH, Lee KM, Hwang H. Sequencing delivery and receiving operations for yard cranes in port container terminals. International Journal of Production Economics 2003;84:283–92.
[15] Ng WC. Crane scheduling in container yards with inter-crane interference. European Journal of Operational Research 2005;164:64–78.
[16] Bish EK. A multiple-crane-constrained scheduling problem in a container terminal. European Journal of Operational Research 2003;144:83–107.
[17] Chen L, Bostel N, Dejax P. et al. A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. European Journal of Operational Research 2007;181:40–58.
[18] Shabayek AA, Yeung WW. A simulation model for the Kwai Chung container terminals in Hong Kong. European Journal of Operational Research 2002;140:1–11.
[19] Won YY, Yong SC. A simulation model for container-terminal operation analysis using an object-oriented approach. International Journal of Production Economics 1999;59:221–30.
[20] Maurizio B, Azedine B, Mohamed R. Object oriented model for container terminal distributed simulation. European Journal of Operational Research 2006;175:1731–51.
[21] April J, Glove F, Kelly JP, Laguna M. Practical introduction to simulation optimization. In: Proceedings of the 2003 winter simulation conference. 2003. p. 71–8.
[22] Allaoui H, Artiba A. Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. Computers & Industrial Engineering 2004;47:431–50.
[23] Guo Y, Liao W, Cheng X, Liu L. SimOpt: a new simulation optimization system based virtual simulation for manufacturing system. Simulation Modelling Practice and Theory 2006;14:577–85.
[24] Julian SY. Solid waste planning under uncertainty using evolutionary simulation-optimization. Socio-Economic Planning Sciences 2007;41:38–40.
[25] Lin H-T, Liao C-J. A case study in a two-stage hybrid flow shop with setup time and dedicated machines. International Journal of Production Economics 2003;86(2):133–43.
[26] Kurz ME, Askin RG. Comparing scheduling rules for flexible flow lines. International Journal of Production Economics 2003;85(3):371–88.
[27] Jin Z, Yang Z, Ito T. Metaheuristic algorithms for the multistage hybrid flow shop scheduling problem. International Journal of Production Economics 2006;100:322–34.
[28] Olafsson S, Kim J. Simulation optimization. In: Proceedings of the 2002 winter simulation conference. 2002. p. 79–84.
[29] Lawrence J. Introduction to neural networks: design, theory, and applications. Nevada, CA: California Scientific Software; 1994.