TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 15/16 pp467-474
Volume 15, Number 4, August 2010

Integrated Quay Crane and Yard Truck Schedule Problem in Container Terminals

CAO Jinxin (曹瑾鑫)[1,2], SHI Qixin (史其信)[1,**], Der-Horng Lee[3]

1. Department of Civil Engineering, Tsinghua University, Beijing 100084, China;
2. Department of Transportation Engineering, Inner Mongolia University, Hohhot 010070, China;
3. Department of Civil Engineering, National University of Singapore, Singapore 117576, Singapore

Abstract: Quay crane and yard truck scheduling are two important subproblems in container terminal operations which have been studied separately in previous research. This paper proposes a new problem for the integrated quay crane and yard truck scheduling for inbound containers. The problem is formulated as a mixed integer programming (MIP) model. Due to the intractability, a genetic algorithm (GA) and a modified Johnson's Rule-based heuristic algorithm (MJRHA) are used for the problem solution. In addition, two closed form lower bounds are given to evaluate the solution accuracy. Computational experiments show that the solution algorithm can efficiently handle the scheduling problem and that the integrated methods are very useful.

Key words: container terminal; scheduling; mixed integer programming; heuristic algorithm

Introduction

Maritime container transport has developed rapidly during the past several decades. Container handling has grown from 3.3 million twenty-foot equivalent units (TEU) in 1997 to 9.9 million TEU in 2007. In 2008, Singapore, Shanghai, Hong Kong, and Shenzhen were the top 4 busiest container terminals in the world, with throughputs of 29.9 million TEUs, 28.0 million TEUs, 24.2 million TEUs, and 21.4 million TEUs, respectively[1]. Container terminals as hubs of the maritime container transport network are facing strict demands from shipping companies to increase the efficiency of stacking and transport of growing numbers of containers. High productivity and container throughput from quayside to land side and back at low costs are vital for terminal operators to compete with other terminals[2].

Received: 2008-12-22; revised: 2009-11-30

** To whom correspondence should be addressed.
E-mail: dcisqx@tsinghua.edu.cn; Tel: 86-10-62792260

Although container terminal operations are becoming more and more complicated with automated handling machines and transporters, advanced communication technologies and modern decision making methods to facilitate improved efficiency and to reduce costs, in general there are three types of equipment employed in container terminals, namely quay cranes (QCs), transporters, and yard cranes (YCs).

As can be seen in Fig. 1, QCs operate at the quayside for loading containers onto and unloading containers from container vessels. Before arrival, each container vessel sends its loading and unloading plan to the container terminal. Terminal operators will designate a QC split plan which indicates the number of QCs required to serve the ship according to the plan, and which ship bay is serviced by which QC. The operators then make a storage plan based on the container characteristics related to the vessel, i.e., contents, storage period, and next destination that determines the temporary storage location for each container in the terminal. YCs work at the yard side staking containers onto their allocated storage locations and picking up containers from their current storage locations in the yard. Transporters transport containers between the quayside and the yardside. Various transporters are used in container terminals, such as yard trucks (YTs), multi-trailers, straddle carriers, automated guided vehicles (AGVs), and automated lift vehicles (ALVs). This paper considers only YTs as the transporters, since they are most widely used in current container terminals.

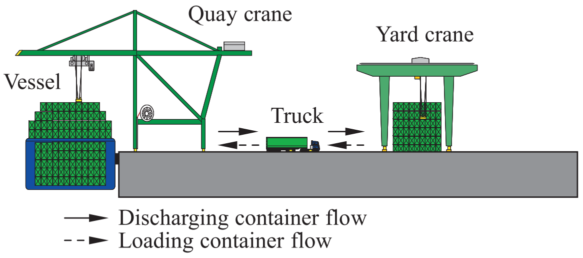

Fig. 1 Typical container flows in container terminals[3]

During the past several decades, container terminal operations have attracted more and more attention from researchers. Container terminals are a complex systems with the operations in the container terminals composed of a series of decision problems, i.e., berth allocation, QC split, QC scheduling, YT scheduling, YC scheduling, and storage allocation. Even today's computational capacities are not sufficient to formulate and optimize container terminal operations as a whole.

Most previous research on container terminal operations has focused on optimizing a single subproblem instead of the whole system. Two of the well-studied subproblems are QC scheduling and YT scheduling.

The QC scheduling problem was first discussed by Daganzo in 1989[4]. He suggested an algorithm for determining the number of cranes to assign to ship-bays of multiple vessels. Peterkofsky and Daganzo[5] provided an algorithm for determining the departure times of multiple vessels and the number of cranes to assign to individual holds of vessels within a specific time segment. The objective function was to minimize the delay costs. Both of these studies assumed one task per ship bay which needed crane operations during a specific length of time. In contrast, Kim and Park[6] extended the QC scheduling problem by assuming that there may be multiple tasks involved in a ship bay, thus determining the QC schedule for each container rather than for each ship bay. The problem was formulated as a mixed integer programming model with a branch and bound algorithm used to solve the model. However, none of these studies related to QC scheduling considered the constraints of yard operations, such as the QC waiting time for the YT. Previous studies all assumed that the YTs were always available under the QCs, which is not necessarily true during peak periods.

A minimum flow algorithm was proposed for scheduling of transporters[7] to determine the minimum number of AGVs needed to complete a given set of delivery tasks without causing a delay in the semi-automated container terminals. Bish et al.[8] suggested a greedy algorithm for dispatching the YTs with a worst-case analysis of the dispatching method. Kim and Bae[9] proposed a look-ahead dispatching method for AGVs. Unlike in previous studies Kim and Bae[9] assumed that the QCs might need to wait for the AGVs with the objective function being to minimize the AGV traveling time as well as the QC waiting time for the AGVs. The problem was formulated as a one-to-one assignment problem and solved with a dedicated heuristic algorithm. The static version of the problem was then extended into a dynamic version by using different step sizes for looking ahead. Ng et al.[10] proposed the problem of scheduling a fleet of trucks to perform a set of transportation jobs with sequence-dependent processing times and different ready times. The ready times for the containers reflect the QC operations and are assumed to be given.

Even though more and more researchers have realized the importance of integrated methods for container terminal operations[2,11], QC scheduling and YT scheduling, two highly related subproblems in container terminal operations, have not been considered simultaneously. This paper presents an integrated quay crane and yard truck scheduling problem (IQYSP) for inbound containers. The problem is formulated as a mixed integer programming (MIP) model. Due to the intractability of the IQYSP, a genetic algorithm (GA) and a modified Johnson's Rule-based heuristic algorithm (MJRHA) was used for the solution. In addition, a job-based lower bound and an equipment-based lower bound were derived to estimate the quality of the solutions obtained by the GA and the MJRHA.

1 Problem Definition and Formulation

In practical operations, a set of QCs is assigned to serve a vessel, with unloading of containers generally preceding loading operations. The ship bays in the vessels are partitioned, with one QC per partition. The input is the set of containers to be unloaded by each QC which is assumed to be given, with the schedule of only one QC considered. A *job* is defined as a container to be discharged. Let i and j be the indices of jobs with $i, j=0, \cdots, N+1$, where N is the number of jobs to be unloaded by the QC, and Job 0 and Job $N+1$ are two dummy jobs related to the original and final states.

In general, a fixed set of YTs is assigned to perform the transportation tasks for a QC. The YTs have two operational strategies, namely single-cycling and dual-cycling. Figure 2 shows the differences between single-cycling and dual-cycling strategies for YT operations. Even though dual-cycling strategies obviously increase the transportation efficiency and reduce the empty YT moves, they are actually only suitable for automated container terminals, since the complicated dispatching strategies are difficult for the drivers to follow. Thus this paper focuses on the single-cycling strategies which are widely used at manned container terminals. Let k be the index for the YTs assigned to the QC, with $k=1, \cdots, K$, where K is the number of YTs assigned to the QC.

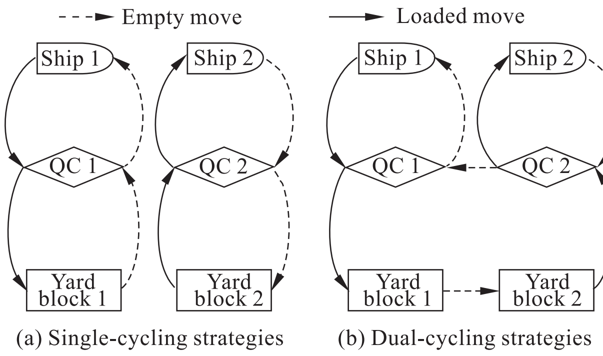

Fig. 2 YT operations strategies

The notation for the ship operations is:

p_i, processing time required to discharge Job i by the QC, $i=1, \cdots, N$.

s_{ij}, setup time for the QC to perform Job j immediately after Job i, $i=0, \cdots, N$ and $j=1, \cdots, N$.

t_i, transportation time required for the YTs to transport Job i to its storage location, $i=1, \cdots, N$.

d, time required to unload a container from the YT by a YC.

M, a large positive number.

The decision variables in the mathematical formulation are:

$X_{ij}=1$, if the QC performs Job j immediately after Job i,
$=0$, otherwise. ($i=0, \cdots, N$ and $j=1, \cdots, N+1$).

$Y_{ik}=1$, if Job i is assigned to YT k,
$=0$, otherwise. ($i=1, \cdots, N$ and $k=1, \cdots, K$).

$Z_{ijk}=1$, if YT k performs Job j immediately after Job i,
$=0$, otherwise. ($i=0, \cdots, N$, $j=1, \cdots, N+1$, and $k=1, \cdots, K$).

r_{i1} = starting time for performing Job i by the QC, $i=1, \cdots, N$.

c_{i1} = time when Job i is ready for moving by the QC, $i=1, \cdots, N$.

r_{i2} = starting time for moving Job i by a YT, $i=0, \cdots, N$.

c_{i2} = completion time of Job i, $i=1, \cdots, N$.

$C_{\max}$ = makespan of all jobs.

Then, the IQYSP can be formulated as:

$$\text{Min } C_{\max} \tag{1}$$

s.t.
$$C_{\max} \geqslant c_{i2}, \text{ for } \forall\, i=1,\cdots,N \tag{2}$$

$$\sum_{i=0}^{N} X_{ij}=1, \text{ for } \forall\, j=1,\cdots,N+1 \tag{3}$$

$$\sum_{j=1}^{N+1} X_{ij}=1, \text{ for } \forall\, i=0,\cdots,N \tag{4}$$

$$r_{j1}+(1-X_{ij})M \geqslant s_{ij}+r_{i2}, \text{ for } \forall\, i=0,\cdots,N \text{ and } j=1,\cdots,N \tag{5}$$

$$c_{i1}=r_{i1}+p_i, \text{ for } \forall\, i=1,\cdots,N \tag{6}$$

$$r_{i2} \geqslant c_{i1}, \text{ for } \forall\, i=1,\cdots,N \tag{7}$$

$$\sum_{k=1}^{K} Y_{ik}=1, \text{ for } \forall\, i=1,\cdots,N \tag{8}$$

$$\sum_{j=1}^{N+1} Z_{0jk}=1, \text{ for } \forall\, k=1,\cdots,K \tag{9}$$

$$\sum_{j=1}^{N+1} Z_{i,N+1,k}=1, \text{ for } \forall\, k=1,\cdots,K \tag{10}$$

$$\sum_{j=1}^{N+1} Z_{ijk}=Y_{ik}, \text{ for } \forall\, i=1,\cdots,N \text{ and } k=1,\cdots,K \tag{11}$$

$$\sum_{i=0}^{N} Z_{ijk}=Y_{jk}, \text{ for } \forall\, j=1,\cdots,N \text{ and } k=1,\cdots,K \tag{12}$$

$$Z_{ijk}+Z_{jik} \leqslant 1, \text{ for } \forall\, i,j=1,\cdots,N \text{ and } k=1,\cdots,K \tag{13}$$

$$Z_{ijk}+Z_{jik} \geqslant Y_{ik}+Y_{jk}-1, \text{ for } \forall\, i,j=1,\cdots,N \text{ and } k=1,\cdots,K \tag{14}$$

$$r_{j2} + (1 - Z_{ijk})M \geqslant c_{i2} + t_i, \text{ for } \forall\, i, j = 1, \cdots, N \text{ and } k = 1, \cdots, K \quad (15)$$

$$r_{j2} + (1 - Z_{0jk})M \geqslant 0, \text{ for } \forall\, i = 1, \cdots, N \text{ and } k = 1, \cdots, K \quad (16)$$

$$r_{i2} + t_i + d \leqslant c_{i2}, \text{ for } \forall\, i = 1, \cdots, N \quad (17)$$

where X_{ii}=0 for all i=1,⋯,N, Z_{iik}=0 for all i=1,⋯,N and $k = 1, \cdots, K$, $r_{02} = 0$ and $Y_{0k} = 1$ for all k=1,⋯,K.

The objective of this problem is to minimize the makespan for dispatching the set of containers allocated to the QC. Constraint (2) is used to calculate the makespan that is the completion time of the last container in the job set. Constraints (3) and (4) force the one-to-one assignment between all jobs in the set to form an unloading sequence of the QC. Constraint (5) implies that if Job j is immediately unloaded by the QC after Job i, the QC needs a setup time of s_{ij} before it starts to unload Job j after Job i has been transported. Constraints (6) represents that the completion time for unloading a container by the QC is equal to the starting time of unloading the job plus the processing time for the container by the QC. Constraint (7) implies that a container can be transported only after the completion of the unloading process by the QC. Constraint (8) guarantees that each container can be assigned to and transported by one and only one YT. Constraints (9) and (10) force the dummy jobs to be the first and last job of each YT. Constraints (11) and (12) guarantee that if a container is allocated to YT k, there must be one and only one succeeding job after and preceding job before this job. Constraints (13) and (14) give the relationship between jobs assigned to the same YT. Constraints (15) and (16) relate the starting and completion times between adjacent jobs for the same YT. Constraint (17) states that a job experiences the transportation and stacking times between the completion time of a job and the starting of the transportation.

The YT scheduling is similar to the multiple traveling salesmen problem (m-TSP). The problem is also analogous to the two-stage flexible flow shop problem (2-FFSP) with sequence-dependent setup times and blocking. All of these reduced problems have been proved to be NP-hard problems, from the computational complexity point of view, which means that there is no efficient algorithm to obtain the exact optimum solution of the problem. However, lower bounds for the problem can be found to evaluate solution quality before designing the solution algorithms.

2 Lower Bounds

This section analyzes the lower bounds for the IQYSP. First define an operator to state the bounds.

Definition Define the operator "$\min^{[k]}$" as the indication of the (k+1)-th solution from the lowest value with $\min^{[0]} \equiv \min$.

For example, given a list of values S={1, 4, 7, 9, 10}, $\min\limits_S^{[2]} = 7$.

The following theorem provides two lower bounds for the IQYSP, namely the job-based lower bound and the equipment-based lower bound.

Theorem The following are lower bounds on any feasible solution to the IQYSP:

$$\mathrm{LB}^{(1)} = \max_{i=1,\cdots,N} \{p_i + \min_{j=0,\cdots,N} s_{ji} + t_i + d\} \quad (18)$$

$$\mathrm{LB}^{(2)} = \min_{i=1,\cdots,N} (p_i + \min_{j=0,\cdots,N} s_{ji}) + \frac{\sum\limits_{i=1,\cdots,N} (2t_i + d)}{K} + \frac{1}{K}\sum_{k=1}^{K-1}[\min_{i=1,\cdots,N}^{[k]}(p_i + \min_{j=0,\cdots,N} s_{ji}) - \min_{i=1,\cdots,N}(p_i + \min_{j=0,\cdots,N} s_{ji})] - \min_{i=1,\cdots,N} t_i \quad (19)$$

Proof $\mathrm{LB}^{(1)}$: This is the job-based lower bound. Every Job i must experience the QC processing time, the YT transportation time, and the stacking time in the yard block. In addition, the QC requires a setup time which is at least the minimum amount of time required to set up Job i. Solutions which are feasible to Constraints (2)-(17) satisfy this condition.

$\mathrm{LB}^{(2)}$: This is the equipment-based lower bound. The first term represents the minimum time required for a job to be ready for transport. The second term assumes that jobs are transported preemptively by the YTs. The third term follows from the observation that the second YT does not work until the second job arrives and so on for all the YTs. The minimum is then the sum of all the times for the second, third, and other jobs to be ready for transport and allocated to the YTs. The first three terms obtain the lower bound of the time when the last job has been finished and the YT returns to the QC. Compared to the makespan definition, the YT return time is subtracted to find the minimum travel time of jobs so as to compute the lower bound for the makespan, which is represented by the last term. Solutions which are feasible to Constraints (2)-(17) to satisfy this condition.

3 Genetic Algorithm

The GA, developed by Holland[11], is one of the nature-inspired meta-heuristics used for handling combinatorial problems. The usual form of the GA was reported by Goldberg[12]. The GA is a stochastic search technique based on the mechanism of natural selection and natural genetics. The GA starts with a set of random solutions called a population. Each individual, named a chromosome, is represented by a string. The chromosomes evolve through successive iterations, called generations. Offspring are generated through crossover and mutation operations on randomly selected chromosomes. A new generation is then selected based on Darwinian evolution by evaluating the fitness. Individuals with better performances will have more probability to be chosen. The GA has been refined by numerous researchers and has become one of the most popular meta-heuristic algorithms for solving facility scheduling and operation problems in container terminals[10].

3.1 Representation

As can be seen in Fig. 3, a feasible solution, namely a chromosome, for the problem can be represented by a string describing the QC dispatching sequence. Each position represents the index of a job. The chromosome is decoded by a YT dispatching rule that assigns the transportation task to the first idle YT. Chen et al.[13] proved that this dispatching rule obtains the optimal solution given a QC schedule.

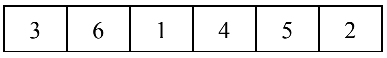

Fig. 3 Illustration of chromosome representation

3.2 Initialization

Initialization is the first stage of a genetic algorithm. For the representation method, a set of feasible solutions is randomly generated to form the first generation. Let pop_size be the population size of the first generation. Thus, pop_size chromosomes are generated, each of which is a string describing the QC unloading sequence.

3.3 Fitness evaluation and selection

The chromosome is evaluated based on:

$$\text{eval}=1/C_{\max} \tag{20}$$

In this paper, a roulette wheel approach is used as the selection procedure using fitness-proportional selection that selects a new population with respect to the probability distribution based on the fitness values[14].

3.4 Crossover

"Ordered crossover"[14] is used for one of the two parts of the chromosomes with a repair procedure to resolve illegitimate offspring. The "order crossover" works as follows.

Step 1 Randomly select a substring in one parent.

Step 2 Produce a proto-child by copying the substring into the corresponding positions in the child.

Step 3 Delete the holds which are already in the substring from the second parent. The resulted sequence of holds contains the holds needed by the proto-child.

Step 4 Place the holds into the unfixed positions of the proto-child from left to right according to the order of the sequence used to produce an offspring.

The procedure is illustrated in Fig. 4 which gives an example making two offsprings from the same parents.

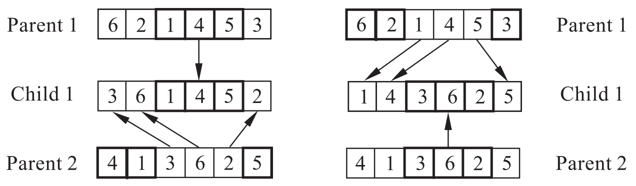

Fig. 4 Illustration of crossover operator

3.5 Mutation

Mutation forces the GA search into new areas to help the GA avoid premature convergence to find the global optimal solution. In general, in mutations all individuals in the population are checked bit by bit and the bit values are randomly reversed according to a pre-specified rate. However, in this paper the mutation process randomly selects chromosomes based on the mutation probability and chooses two positions in the same part of the selected chromosome at random to exchange the values as illustrated in Fig. 5.

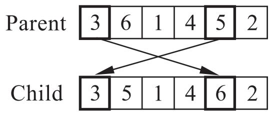

Fig. 5 Illustration of mutation operator

4 Rule-Based Heuristic Algorithm

The GA performance is affected by the algorithm parameters, such as the population size, mutation probability, and crossover probability. This section describes a rule-based heuristic algorithm, which is an extension of Johnson's Rule taking into account the setup times[15]. Johnson's Rule has been used extensively for the flow shop scheduling problem (FSSP) with the objective to minimize the job makespan. However, the original Johnson's Rule is not applicable to the IQYSP due to two assumptions:

(1) Johnson's Rule assumes that the machine environment is a flow shop, which means there is only one machine at each stage;

(2) Johnson's Rule does not take the sequence-dependent setup time into consideration.

Therefore, a heuristic algorithm was developed for the QC and YT scheduling which is a modified MJRHA to overcome these two limitations.

The modified processing for the QC is:

$$\tilde{p}_i^1 = p_i + s_i \tag{21}$$

where

$$s_i = \min_{j=1,\cdots,N,\ j\neq i} s_{ji} \tag{22}$$

The modified processing time incorporates the minimum time to setup Job i into its processing time, which extends the original Johnson's Rule to handle the sequence-dependent setup time.

In addition, the modified transport time is defined as:

$$\tilde{p}_i^2 = \frac{2t_i + d}{K} \tag{23}$$

The modified transport time for a job extends the original Johnson's Rule to a flexible flow shop machine environment.

All jobs can be partitioned into two sets denoted by Ω_1 and Ω_2 based on the modified QC processing time and the modified YT transport time.

$$\Omega_1 = \{i \mid \tilde{p}_i^1 < \tilde{p}_i^2\} \tag{24}$$

$$\Omega_2 = \{i \mid \tilde{p}_i^1 \geqslant \tilde{p}_i^2\} \tag{25}$$

The MJRHA can then be described as:

Step 1 Jobs in Ω_1 precede jobs in Ω_2.

Step 2 Jobs in Ω_1 are performed in increasing order of $\tilde{p}_i^1$ (shortest processing time first, SPT).

Step 3 Jobs in Ω_2 are performed in the nonincreasing order of $\tilde{p}_i^2$ (longest processing time first, LPT).

Step 4 The QC job sequence obtained from Steps 1 to 3 is used to assign each job to the first idle YT.

5 Computational Tests

The performance of the solution algorithms were evaluated based on 20 computational examples generated with random inputs. For smaller scale problems, the exact solution can be obtained by CPLEX within rational computational times. However, with increasing problem scales, the computational times quickly become unacceptably long.

Table 1 lists the makespans for the computational examples obtained by the different solution methods. The second column shows the number of containers and the number of YTs in each example, ranging from 5 containers with 1 YT to 15 containers with 4 YTs. Column 3 shows the lower bound for each example, which is equal to $\max\{\text{LB}^{(1)}, \text{LB}^{(2)}\}$. The results calculated by CPLEX for smaller scale examples are listed in Column 4. The Gap is defined as:

$$\text{Gap} = \frac{\text{makespan obtained by the solution algorithm} - \text{LB}}{\text{LB}} \times 100\% \tag{26}$$

The Gap for CPLEX shows that the lower bound is at most 10% lower than the optimal solution. In most cases, at least for small scale examples, the GA can get the same exact solution as obtained from CPLEX. The Gap for the GA in each example is controlled to within 10%. However, the qualities of solutions obtained from the rule-based heuristic are not as good as those of the GA, though it can be easily implemented. The largest Gap for the MJRHA is 44.56%.

Besides the solution quality, the computational time is always another important factor for evaluating solution algorithms.

As can be seen in Fig. 6, the GA obtains solutions within acceptable computational times ranging from 0.062 s to 0.375 s. The computational times for CPLEX increase unacceptably with a larger number of vehicles (1049.86 s for experiment 9 and 345 600.90 s for experiment 10). For all the examples, the computational times for the MJRHA were less than 0.001 s.

Table 1 Results of computational tests

No. of example	No. of containers × No. of trucks	LB	CPLEX		GA		MJRHA	
		max{LB$^{(1)}$, LB$^{(2)}$}	Makespan (s)	Gap (%)	Makespan (s)	Gap (%)	Makespan (s)	Gap (%)
1	5×1	3018	3203	6.11	3300	9.33	3310	9.66
2	5×2	2372	2608	9.97	2608	9.97	2890	21.86
3	5×3	1868	1991	6.58	1997	6.90	2571	37.62
4	6×1	3784	4100	8.36	4105	8.49	4107	8.54
5	6×2	2816	2922	3.77	2922	3.77	3180	12.94
6	6×3	1959	2120	8.21	2120	8.21	2496	27.40
7	7×1	6948	7220	3.91	7220	3.91	7633	9.85
8	7×2	3407	3650	7.12	3650	7.12	4301	26.22
9	7×3	2441	2626	7.60	2626	7.60	3125	28.04
10	8×1	8515	–	–	9226	8.35	9269	8.85
11	8×2	4326	–	–	4630	7.03	4749	9.78
12	8×3	3271	–	–	3403	4.05	3691	12.85
13	9×1	8455	–	–	9024	6.73	9160	8.34
14	9×2	4308	–	–	4658	8.13	5092	18.21
15	9×3	3042	–	–	3266	7.36	3890	27.87
16	10×3	3232	–	–	3399	5.15	3886	20.22
17	10×4	2569	–	–	2826	9.99	3519	36.96
18	10×5	2362	–	–	2546	7.77	3415	44.56
19	15×3	5373	–	–	5537	3.04	5795	7.85
20	15×4	4140	–	–	4306	4.00	4633	11.90

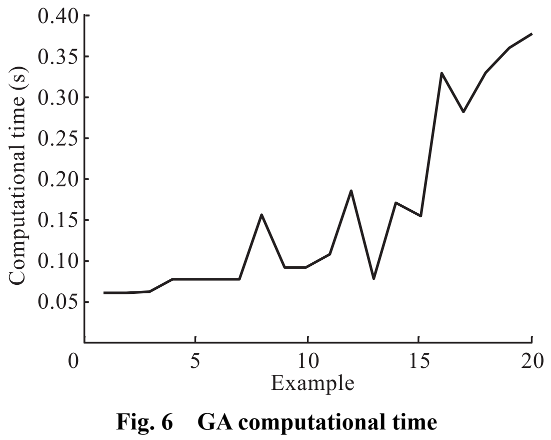

Fig. 6 GA computational time

To test the significance of the integrated model, a benchmark strategy was developed for the IQYSP. The benchmark strategy always chooses the job with the smallest setup time as the immediately succeeding job, which is similar to strategies used in practice today, but this does not necessarily give the best results.

As shown in Fig. 7, compared with the benchmark strategy, the makespan obtained by the proposed GA to the IQYSP can be reduced by a large percentage, ranging from 23% to 115%. Thus, the integrated model can be used to determine significantly better solutions.

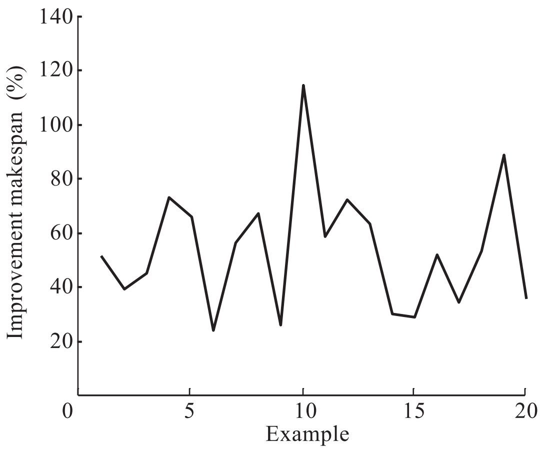

Fig. 7 Improvement of makespan

6 Conclusions and Future Research

This paper describes an IQYSP for discharging in-bound containers at container terminals. The IQYSP determines the QC unloading schedule taking into account the YT operations. The problem is formulated as a two-stage flexible flow shop problem with sequence-dependent setup times and blocking in an MIP. The objective of the problem is to minimize the makespan for discharging all containers assigned to the QC. Due to the computational complexity of the problem, optimal solutions cannot be found through exact algorithms, thus a GA and a MJRHA were used for the problem solution. Two lower bounds were also derived to facilitate evaluations of the solution qualities.

The computational examples showed that:

(1) The exact algorithm cannot solve large scale problems within an acceptable amount of time.

(2) The proposed algorithms can obtain solutions with satisfactory qualities in terms of the Gaps relative to the lower bounds.

(3) The algorithms can obtain solutions for larger scale problems in acceptable computational times.

(4) Compared with the benchmark strategy, the IQYSP provides better solutions reducing the makespan of discharging inbound containers than the benchmark strategy, which would reduce the costs of container terminal operations.

There are also several potential directions of future research work:

(1) Container loading operations could be optimized in a similar way.

(2) The YC operations can also be integrated into the problem instead of using a fixed YC stacking time as in this paper.

(3) Explore efficient formulations and algorithms for the problem.

References

[1] Port of hamburger. http://www.hafen-hamburg.de/content/view/31/33/lang,en/, 2008.

[2] Stahlbock R, Voß S. Operations research at container terminals: A literature update. *OR Spectrum*, 2008, **30**: 1-52.

[3] Ng W C. Crane scheduling in container yards with inter-crane interference. *European Journal of Operational Research*, 2005, **164**: 64-78.

[4] Daganzo C F. The crane scheduling problem. *Transportation Research*, 1989, **23B**(3): 159-175.

[5] Peterkofsky R I, Daganzo C F. A branch and bound solution method for the crane scheduling problem. *Transportation Research*, 1990, **24B**(3): 159-172.

[6] Kim K H, Park Y M. A crane scheduling method for port container terminals. *European Journal of Operational Research*, 2004, **156**: 752-768.

[7] Vis I F A, de Koster R M B M, Savelsbergh M W P. Minimum vehicle fleet size under time-window constraints at a container terminal. *Transportation Science*, 2005, **39**(2): 249-260.

[8] Bish E K, Chen F Y, Leong Y T, et al. Dispatching vehicles in a mega container terminal. *OR Spectrum*, 2005, **27**: 491-506.

[9] Kim K H, Bae J W. A look-ahead dispatching method for automated guided vehicles in automated port container terminals. *Transportation Science*, 2004, **38**(2): 224-234.

[10] Ng W C, Mak K L, Zhang Y X. Scheduling trucks in container terminals using a genetic algorithm. *Engineering Optimization*, 2007, **39**(1): 33-47.

[11] Holland J H. Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press, 1975.

[12] Goldberg D. Genetic Algorithms in Search, Optimization and Machine Leaning. MA: Addison-Wesley, 1989.

[13] Chen Y, Leong T, Ng W C, et al. Dispatching vehicles in a mega container terminal. Working paper, Northwestern University, Evanston, IL, 1998.

[14] Gen M, Cheng R. Genetic Algorithms and Engineering Design. Singapore: John Wiley, 1996.

[15] Johnson S M. Optimal two- and three-stage production schedules with setup time included. *Naval Research Logistics Quarterly*, 1954, **1**: 61-67.