Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

YARD CRANE DISPATCHING TO MINIMIZE VESSEL TURNAROUND TIMES IN CONTAINER TERMINALS

Shell Ying Huang
Ya Li
Xi Guo

School of Computer Engineering
Nanyang Technological University
SINGAPORE 639798

ABSTRACT

Yard crane (YC) dispatching in the operational planning of container terminals usually aims to minimize makespan of YC operations or waiting time of vehicles. We propose that minimizing the maximum tardiness of vehicle jobs at yard blocks will minimize the operational delay of the longest quay crane (QC). This will minimize vessel turnaround time which is one of the most important objectives of container terminals. A provably optimal algorithm, MMT-RBA* to minimize maximum job tardiness, is presented to sequence the YC jobs. Jobs requiring reshuffling of other containers, often ignored in other studies, are handled by embedded simulation in our optimization algorithms. Another provably optimal algorithm, MMS-RBA* to minimize makespan, is also presented. Simulation experiments confirm that MMT-RBA* significantly outperforms the optimal algorithm RBA* to minimize vehicle waiting time from earlier studies and MMS-RBA* to minimize makespan in minimizing vessel turnaround time.

1 INTRODUCTION

In container terminal operations, one of the most important objectives is to reduce vessel turnaround time (Steenken *et al*. 2004). Li et al. (2009) pointed out that yard crane (YC) operations are of great importance and likely to be a potential bottleneck to the overall terminal performance. This is because when vehicles are delayed in the storage yard, they will not be able to reach their quay cranes (QCs) on time. It follows that QCs' loading/unloading operations will be delayed and vessel turnaround time lengthened. In YC operation management there are two main problems: (i) deciding job sequence for an YC which we refer to as the YC dispatching problem; (ii) allocating YCs to different parts of the yard which we refer to as the YC deployment problem. We study the YC dispatching problem in this paper.

The storage yard of a container terminal is organized in a number of yard blocks. Containers are arranged in a number of rows and slots in a yard block as shown in Figure 1. In many container terminals, vehicles travel along lanes to load/unload containers at the side of a yard block. When multiple vessels are loading and unloading at the same time, vehicles serving different quay cranes will arrive at different slot locations of a yard block for loading and/or unloading jobs. External vehicles may also arrive at any time at pre-determined slot locations to take import containers or deliver export containers. As a result, YCs need to move among different slot locations to serve vehicle jobs. When a YC is busy serving other vehicle(s), a vehicle needs to wait for it. A vehicle may also need to wait for the YC to move to its job location.

In many YC dispatching works presented in the past, the objective is to minimize the total (average) vehicle waiting time (Ng and Mak 2005a and 2005b; Kumar and Omkar 2008; Guo et al. 2011); or to minimize the makespan (Jung and Kim 2006; Lee et al. 2007), that is, the total time taken to finish a set of jobs by the YC. Minimizing vehicle waiting times helps vehicles to return to the QCs as soon as possible after they arrive at the yard blocks. This often reduces the QC waiting time for vehicles thus reduce the vessel turnaround time. However, minimization of QC waiting time for vehicles is not guaranteed. Minimizing vehicle waiting times may result in some vehicles getting to the quayside earlier than they are needed while others are late for the QCs because they arrive at the yard block later. Minimizing makespan for a YC enables the YC to finish a set of jobs as soon as possible. This optimizes the YC productivity. However, vehicles coming to a YC may deliver containers to different QCs, possibly for different vessels. Optimal productivity of a YC is often achieved with minimum gantry movements to reduce the YC unproductive times. This does not necessarily minimize vehicle delays in reaching the QCs.

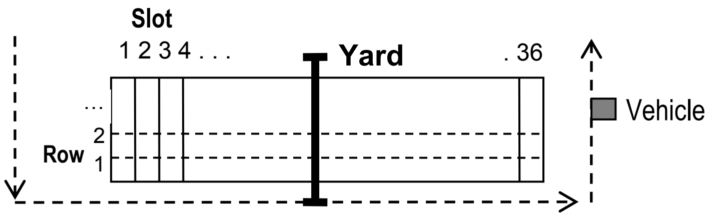

Figure 1: A yard Block with Slots (yard bays) & Rows.

So the question is what objective function a YC dispatching algorithm should have. It is noted that vessel turnaround time is determined by the longest crane serving the vessel, that is, the crane that completes its loading/unloading operations last. Under the assumption that the workload of the QCs serving a vessel is evenly distributed, a QC that experiences the longest delay in its operations due to waiting time for a vehicle will lengthen the vessel's turnaround time. Therefore minimizing the maximum tardiness among YC jobs is an effective way to help reduce vessel delay. We propose that minimizing the maximum tardiness among a set of jobs is most effective in minimizing the operational delay of the longest QC.

For a loading job, based on *the time a QC needs the vehicle at the quayside*, the time this vehicle should leave the yard block with the container to travel to the quayside (the deadline of this vehicle job) can be derived assuming no traffic congestions. The deadline of the vehicle job at the YC is the time the QC needs the vehicle minus the expected travel time from the yard block to the QC. Once a vehicle is assigned to this loading job, *the vehicle's arrival time at the yard block* can be derived or predicted based on the expected travel time of the vehicle and when the vehicle is expected to move towards this block.

Even though an unloading job is less critical to the QCs, longer times needed to store a container in the storage yard also directly lead to interruptions of QCs' unloading process (Kemme, 2010). A deadline set for the vehicle will help it finish the current job and return to the QC on time to get the next container. It may not matter which vehicle returns to the QC first, but it is still important to have a stream of vehicles arriving at the QC continuously but with some intervals in between. In this way when a vessel is being unloaded, quayside will not be crowded with early arriving vehicles but the QC does not need to wait for vehicles either.

For external vehicles carrying export or import containers, a deadline for a storing or retrieving job at the yard block will allow the terminal operator to guarantee a quality of service to the external truck companies.

Given the deadlines of the vehicle jobs and their predicted arrival times at the yard block, the YC dispatching algorithm computes its serving sequence with the ultimate objective of minimizing the delay to the longest QC. We propose optimal algorithm MMT-RBA* to minimize maximum job tardiness as the most suitable algorithm for YC dispatching. An optimal algorithm MMS-RBA* to minimize makespan for the YC dispatching problem is also proposed for the purpose of evaluation. In our performance comparison, we also include the optimal algorithm to minimize total vehicle waiting time (Guo et al. 2011). We show that MMT-RBA* is most effective in reducing operational delays of the longest QC of vessels in each planning window.

The rest of the paper is structured as follows. Firstly, we review the related studies in Section 2. A formal description of the YC dispatching problem and a discussion on the different objective functions for YC dispatching are presented in Section 3. Then the optimal algorithms are proposed in Section 4. The experimental evaluations are presented in Section 5. Conclusion is drawn in the last section.

2 RELATED WORK

The YC dispatching problem was studied by Kim and Kim (1999) where they considered the loading operations only for a single YC with a given load plan and a given bay plan. A Mixed Integer Programming (MIP) model is proposed to minimize the total gantry time of the YC. The solutions focus on the sequence of bay visits and the number of containers to be picked up at each bay while the individual container pick-up sequence within a specific bay is left to the crane operator. Later, Kim and Kim (2003) and Kim et al. (2004) extended the study of this problem by comparing exact optimization, a beam search heuristic and a Genetic Algorithm (GA).

Ng and Mak (2005a, 2005b) developed branch-and-bound heuristics to schedule the single YC for a given set of loading and unloading jobs with different ready times. Their objective is to minimize the total job waiting time. Kumar and Omkar (2008) used particle swarm optimization with genetic algorithm operators to handle YC jobs with different ready times to minimize total job waiting time. It is known that for large problems, the MIP model has limited applicability due to the excessive computational times. On the other hand, heuristics cannot guarantee optimal solutions. Guo et al. (2011) applied A* search to compute optimal single YC dispatching sequence based on vehicle arrival times to minimize vehicle waiting times.

When designing a YC dispatching algorithm, many previous work assumes that the YC service times for vehicle jobs are constant (Jung and Kim 2006, Lee et al. 2007, Cao et al. 2008, Guo et al. 2011). This is correct when the container to be retrieved/stored is on the top of the container stack in the yard. Sometimes the containers to be retrieved/stored are not on top of the stack. The YCs take time to reshuffle the containers above these containers. For an algorithm computing the optimal job sequence, ignoring such difference in YC service times will not guarantee a real optimal solution. Huang et al. (2012) argued that YC service time is sequence dependent and embed the simulation of YC job sequence in the dispatching algorithms. The issue of reducing the computational costs of embedded simulation in the algorithms was studied in Huang and Guo (2013).

Several works studied the problem with 2 or more YCs. Due to the problem complexity, MIP models were commonly employed just to formulate yard related problems while heuristic methods were proposed to find near-optimal solutions. Jung and Kim (2006) considered 2 YCs working in one shared zone to support vessels loadings with a GA and a Simulated Annealing (SA) algorithm to minimize the makespan, i.e. the period between the starting time of the first YC operation and the finishing time of the last YC operation. Lee et al. (2007) considered 2 YCs working in 2 non-overlapping zones with a SA algorithm to minimize the makespan. Ng (2005) studied the problem of scheduling multiple YCs to handle jobs with different ready times within a yard zone with MIP and heuristics. Guo and Huang (2012) proposed space and time partitioning methods to manage the workload among multiple YCs working in a row of yard blocks.

Cao et al. (2008) considered Double-Rail-Mounted gantry (DRMG) crane systems where two YCs can pass through each other along a row of blocks with a combined greedy and SA algorithm to minimize the loading time of containers.

For automated container terminals, loading and unloading operations are done at the two ends of a storage block. There are usually two automated rail-mounted gantry cranes in a block. Stahlbock and Voss (2010) evaluated different online algorithms for sequencing and scheduling of jobs for automated DRMGs serving a yard block. They showed that under high workload, the SA approach performed better than the priority rule-based heuristics. Park *et al.* (2010) studied heuristic methods and local search methods for scheduling twin rail-mounted-gantry (RMG) cranes in an automated container terminal. Different from others, they considered the need to reshuffle containers when a container to be retrieved is not on top of stack. The reshuffling work was treated as independent jobs. This works for their schemes but is not suitable for us. We evaluate different job sequences in the search for an optimal solution and the amount of reshuffling is sequence dependent.

3 THE FORMULATION OF THE YC DISPATCHING PROBLEM

The following assumptions are made in the YC dispatching model:

1) The YC dispatching algorithm is executed to plan the operations within a time window for each YC.
2) The deadlines of vehicle jobs are given at the beginning of each planning window.
3) The vehicle arrival times can be predicted for a relatively short planning window, e.g. 30 minutes. These predicted arrival times are given at the beginning of each planning window.
4) YC gantry time between two job positions could be predicted with high accuracy as gantry speed is usually quite consistent.
5) The slot location (yard bay), the row and tier numbers of the container in each vehicle job are given.
6) The yard block status (how many containers are stored in each stack) is given.

In our formulation, the following notations are used:

$J_y = \{1, 2, \ldots, n_y\}$,	the set of job IDs in the working zone of YC y for a planning window
a_{yi}	the arrival time of job i at the yard block of YC y.
p_{yi}	the process time of job i by a YC y.
d_{yi}	the deadline of job i at YC y.
m_{yij}	the time for YC y to move from the position of job i to that of job j.
S_{yi}	the time YC y starts processing job i.
C_{yi}	the time YC y completes processing job i.

J_y is the set of jobs to be sequenced for the YC y. For each planning window in the operation control of a container terminal, there are M sets of J_y (y = 1, 2, ..., M) where M is the number of YCs operating in the current shift of a day. We assume that there is one YC working in a yard block which has storing or retrieving jobs. When discussing the solution of the general YC dispatching problem for one YC, we drop the subscript y in the notation. m_{0j} is the YC gantry time from its position at the start of the time window to the position of job j. C_0 is the time the YC is available to start moving to the position of its first job in the YC dispatching sequence.

The completion time for job i is equal to its start time + process time, that is, $C_i=S_i+p_i$. When vehicle arrivals can be predicted and the next job is decided, an YC is able to start moving towards the next job location before the actual vehicle arrival. Therefore job starting time is as shown in (1):

$$S_j = max(C_i + m_{ij}, a_j). \tag{1}$$

Job process time p_i is the YC service time for the job. It is a sequence dependent variable which cannot be pre-determined. We will embed simulation to compute p_i dynamically during the planning of the YC dispatching sequence.

Consider a set J of n vehicle jobs with predicted job arrival times a_i ($i = 1, 2, \ldots, n$), deadline d_i ($i = 1, 2, \ldots, n$) and YC gantry times m_{ij} ($i = 0, 2, \ldots, n; j = 1, 2, \ldots, n$), the tardiness of a job J_i is defined as $T_i = max\ (0,\ C_i - d_i\ )$. The waiting time of a job J_i is defined as $W_i = S_i - a_i$. The makespan of the sequence is $C_{last} - C_0$ where *last* is the last job in the dispatched job sequence. In many works presented in the past, the objective of the YC dispatching algorithm is to minimize the total (average) vehicle waiting time:

$$\min_{job\ sequences}\left(\textstyle\sum_i W_i\right) \tag{2}$$

or to minimize the makespan:

$$\min_{job\ sequences}\left(C_{last} - C_0\right). \tag{3}$$

We propose that the YC dispatching problem should have the objective function:

$$\min_{job\ sequences}\left(\max_i(T_i)\right). \tag{4}$$

Consider a sequence of l vehicles scheduled to come to one QC to deliver containers. Each vehicle has a scheduled time to reach its QC in order to keep the QC continuously working. Let $\{d_1, d_2, ..., d_l\}$ be the tardiness of the vehicles in reaching the QC (some tardiness may be zero when a vehicle is not late). The delay to the QC in completing this sequence of jobs is $\max(d_1, d_2, ..., d_l)$. For example, three vehicle jobs to be served by a YC are scheduled to reach a QC. Suppose vehicles V1, V2 and V3 are delayed 1, 3, 2 time units by the YC respectively. V2 reaches the QC three time units later than scheduled but the QC will only wait for this vehicle for 2 time units, because the QC finishes V1's job one time unit later than scheduled. The QC is not delayed by V3 even though it is late by 2 time units because it finishes V2's job 3 time units later than scheduled. The delay to the QC to complete the three vehicles' jobs is max(1, 3, 2). If V1, V2 and V3 are to reach QC1, QC2 and QC3 of a vessel, the delay to the vessel by these three jobs is also max(1, 3, 2) where QC2 is the longest crane as far as these three jobs are concerned. Therefore minimizing the maximum tardiness among the jobs in a sequence will minimize the delay to vessel operations.

```
YCDispatching (J)      // J = {J_1, J_2, ... , J_n}
 {
1    newJ = ϕ;  optimalJ = ϕ;  CurSmallest = ∞;
2    sort J into a certain order where this sequence will give a good solution;
3    RBAframe(J, newJ);
}
RBAframe(J, newJ)
{
1    FOR each job $J_i \in J$    // Select J_i as the job to serve after jobs in newJ;
2        Remove  J_i from J and append  J_i  to newJ;
3        Simulation to get the value of  g(J_i) from start to this job J_i;
4        Estimate lower bound cost h(J_i)  from this job ;
5        IF f( J_i ) based on g(J_i) and h(J_i) is smaller than CurSmallest
6            IF J is not empty
7                RBAframe(J, newJ);
8            ELSE  // f( J_i ) is the real cost and is smaller than CurSmallest
9                Update CurSmallest as  f( J_i ) ;
10               Update optimalJ = newJ;  //Store Optimal List
 }
```

Figure 2: Pseudocode of the RBA* framework.

4 RECURSIVE BACKTRACKING ALGORITHMS WITH AN A* HEURISTIC

Given an YC dispatching problem of *n* jobs, there are *n!* possible dispatching solutions in total. As the dispatching problem is strongly NP-hard (Narasimhan and Palekar 2002), exhaustive search that guarantees optimality would be time-consuming to perform. We propose our optimal algorithms using the recursive backtracking with an A* heuristic approach as in (Guo et al. 2011). Figure 2 presents the framework of this approach. The optimal algorithms for different objective functions (for (2), (3) and (4)) will have different evaluation function *f(x)*.

4.1 MMT-RBA*- Dispatching Yard Crane to Minimize Maximum Tardiness

The algorithm takes as input the predicted vehicle job arrival times and their deadlines for jobs expected in the YC's working zone in the planning window. It also takes in the time the YC is available to start the first job in the planning window and its initial location. This will be the time and position of the YC when this YC finishes its last job in the previous planning window.

The objective of our tree search is to find a path from the start node to a leaf node (a dispatching sequence of *n* jobs) with minimum *f(x)* (i.e. *f(x)* is the maximum job tardiness). The edge weight from node *i* to node *j* in the tree is the tardiness of job *j* if the YC is to do job *j* immediately after finishing job *i*. This edge weight is given by

$$W_{ij} = max\left(0, max\left(C_i + m_{ij}, a_j\right) + p_j - d_j\right). \quad (5)$$

The evaluation function *f(x)* in line 5 of **RBAframe()** in Figure 2 for MMT-RBA* will be

$$f(x) = max(g(x), h(x)). \quad (6)$$

g(x) will be the maximum job tardiness among the jobs already planned in the partial sequence from start node to node *x* (line 3 of **RBAframe()** in Figure 2). So

$$g(x) = \max(W_{12}, W_{23,\ldots,} W_{x-1,x}) \quad (7)$$

where edge weight W_{ij} is defined by (5). The way to compute p_j in W_{ij} dynamically by simulation is presented in Section 4.3. *h(x)* will be the estimated lower bound of the maximum job tardiness among the jobs not planned yet. For these jobs, the minimum tardiness will be the job tardiness if the job is done immediately after the current job *x*. In addition, the lower bound of p_j is the minimum processing time T_p which happens when the container is on the top of the stack. In other words, we assume that containers to be retrieved from the yard during loading are on top of the storage yard, and containers to be stored in the yard during unloading are just placed on top of the proper slot. In this case, process time p_j is the time YC makes one container move, T_p. So the lower bound of maximum tardiness (line 4 of **RBAframe()** in Figure 2) will be computed by

$$LBW_j = \max(0, \max\,(C_x + m_{xj}, a_j) + T_p - d_j), \qquad J_j \in \text{set of jobs not planned yet}, \quad (8)$$

$$h(x) = \max(LBW_j), \qquad J_j \in \text{set of jobs not planned yet}. \quad (9)$$

Proof: *h(x)* is admissible

The tardiness LBW_j as expressed by (8) is the very minimum for each job that is not planned yet. It is the job tardiness if it were the first job to be served after job *x* and the minimum job processing time by the YC is incurred. It follows that *h(x)* as computed by (9) will be the lower bound of the maximum job tardiness among the jobs not planned yet.

Thus *h(x)* can never overestimate the cost from node *x* to the leaf node and is hence admissible. This guarantees that MMT-RBA* will find the job sequence that minimizes the maximum tardiness of jobs among all sequences of the *n* jobs.

To accelerate the search process of the MMT-RBA* algorithm, we propose a prioritized search order which is the ascending order of the job deadlines. Intuitively, if the YC serves a job with an earlier deadline first, it will contribute to the minimization of the tardiness of this job. Therefore in line 2 of **YCDispatching()** in Figure 2 we sort the job list J into ascending order of deadlines in MMT-RBA*.

4.2 MMS-RBA*- Dispatching Yard Crane to Minimize Makespan

The input to this algorithm is the same as those for MMT-RBA* except that the algorithm will not consider the deadlines of the jobs. The objective of our tree search is to find a path from the start node to a leaf node (a dispatching sequence of *n* jobs) which minimizes the completion time of the last job in the dispatched sequence. Therefore *f(x)* will be the completion time of job *x*. The edge weight from node *i* to node *j* in the tree is the completion time of job j if the YC is to do job *j* immediately after finishing job *i*. This edge weight is given by

$$W_{ij} = \max\,(C_i + m_{ij}, a_j) + p_j. \tag{10}$$

g(x) (line 3 of **RBAframe()** in Figure 2) is the makespan of the partially dispatched job sequence up to job *x* so it is the completion time of job *x*. It can be computed by (10) where C_i is the completion time of the previous job in the sequence. The way to compute p_j in W_{ij} dynamically by simulation is presented in Section 4.3.

h(x) (line 4 of **RBAframe()** in Figure 2) is the lower bound of the completion of the last job among all possible job sequences where the partially dispatched sequence is the prefix. We estimate *h(x)* by

$$\begin{aligned} h(x) &= \max(h_1(x), h_2(x)), \\ h_1(x) &= C_x + r^*\, T_p + r^* \min_{jk}(m_{jk}), \\ h_2(x) &= \max(a_l) + T_p \end{aligned} \tag{11}$$

where $j,\ k \in \{x\} \cup$ set of unplanned jobs and $l \in$ set of unplanned jobs. So a_l is the last arrival time among the unplanned jobs. *r* is the number of jobs not planned yet and T_p is the minimum job processing time.

So $h_1(x)$ is the minimum total job processing time plus the minimum total gantry time after the completion of job *x*. $h_2(x)$ is the time of the last arrival plus the minimum job processing time. Obviously the makespan of the complete job sequence cannot be smaller than both $h_1(x)$ and $h_2(x)$. That is, *h(x)* can never overestimate.

The evaluation function *f(x)* in line 5 of **RBAframe()** in Figure 2 will be

$$f(x) = \max(g(x), h(x)). \tag{12}$$

To accelerate the search process of the MMS-RBA* algorithm, in line 2 of **YCDispatching**() in Figure 2 we arrange the job list J into an order by SCJF (Smallest Completion time Job First). Intuitively, if the YC serves the job with the smallest completion time first, it is likely to help in completing all the jobs in the shortest time.

If optimizing YC productivity is the main concern of a terminal, MMS-RBA* will be able to find the job sequence for the optimal productivity of individual cranes.

4.3 Computing Job Processing Times p_j for a Partial Sequence by Simulation

Whether a wanted container in a retrieval operation is at the top or not is sequence dependent. It cannot be pre-determined in the search for the optimal job sequence. Simulation is employed to manage the reshuffling and compute each job's processing time dynamically according to the current job sequence. Based on the processing time of each job, its completion time C_x and tardiness C_x - d_x can be decided. Embedding simulation into a YC dispatching algorithm is proposed in Huang et al. (2012) and Figure 3 shows the outline of the method.

```
sequenceTardiness(JobList)
// compute completion time $C_x$ and tardiness $C_x$ - $d_x$ for each job $x$ in JobList
{  ReInititalizeYardBlock;                    //start simulation with initial yard block status
   FOR each job $j$ in JobList
        serveJob(JobList, $j$);
}
serveJob(JobList, k)     // called from sequenceTardiness()
{   YC moves to JobList[k].slot;   // YC gantry move
    IF container for JobList[k] is not at top tier
        Move the blocking container(s) to suitable neighbouring stacks and update yard block status
   Job service time $p_k$ = number of containers moved * $T_p$;    // $p_k$ in (5)
   Calculate job_tardiness according to (5);
}
```

Figure 3: Outline of pseudocode for simulation to compute C_x and C_x - d_x for each job in a Job List.

5 PERFORMANCE EVALUATION

5.1 Design of Simulation Experiments

To evaluate the performance of the proposed YC dispatching algorithms, simulation experiments were carried out. The YC dispatching models are programmed in C++ language under Microsoft Visual Studio 2010 using Dell Precision T3500, Windows 7 64-bit OS, Intel(R) Xeon(R) CPU with 3.2GHz and 6GB RAM. Parameter settings in the experiments were obtained from real world terminal models as in past projects (Guo et al. 2007). The linear gantry speed of an YC is 7.8km/hour. A yard block has a size of 36 slots.

We simulate the yard crane operations in a terminal with high volumes for transhipment containers. Since the import containers are usually delivered to a yard block designated specifically to import containers, the majority of the vehicles coming to a non-import block are serving vessel loading/unloading operations. We will consider such vehicle jobs only. We approximate real operation environment where q QCs are loading/unloading containers from/to b yard blocks. We consider the cases where the q QCs are serving one vessel to investigate how minimizing maximum job tardiness help in minimizing this vessel's turnaround time. In each planning window there are n jobs from various QCs to each of the q yard block. Each yard block has one YC to serve the vehicles. In all our experiments, we set n to be 10. In other words, each planning window will plan for a job sequence of 10 jobs for each YC. We do not experiment with a job sequence longer than 10 jobs. This is because it is impractical to assume the availability of accurate information about job arrival times for long job sequences in the dynamic operating conditions. A sequence of 10 job arrivals covers a time period of 20-30 minutes.

We reproduce the following operation characteristics in transhipment-intensive terminals when generating the arrival patterns of vehicle jobs to the yard blocks in the simulation experiments.

(1) In a transhipment-intensive terminal, the majority of containers unloaded from one vessel to the yard will be loaded onto a number of second carriers. The containers it will load from the yard are from a number of first carriers. Containers for the same second carrier will be stored in a number of clusters in the yard depending on their destination port, weight, class and type. Therefore within a planning window, each QC will load/unload from a small number of yard blocks (YCs) when serving a vessel. Based on this, we decide where the jobs from a QC will go in the storage yard. We set $q = 4$ and $b = 8$. Each QC has its containers to/from 2 yard blocks.

(2) In the continuous operation of a QC, it is common to see a QC rate of 20-35 containers per hour. This is not the peak rate that a QC can work but it is a realistic rate. Based on this, we decide the job deadlines for a sequence of jobs from a QC, with a small amount of randomness.

(3) Vehicles are assigned to transport containers to meet the QC's deadlines for jobs. With randomness, most vehicles will arrive at the yard block with enough time to spend at the yard block and meet QC's deadline. But a small number of vehicles may be late. So the vehicles arrival times at the yard blocks are set to the job deadline minus a random variable from a uniform distribution U(x, y).

Based on the above, The distribution of jobs in the yard, the job deadlines and arrival time information in our experiments are as given in Table 1. Two job deadline patterns are used in experiment sets 1 and 2 respectively to simulate two QC operating rates.

Table 1: Setting of experiment set C.

QC	Working with	Vehicle job deadlines	Job arrival time
QC1	Yard blocks 1, 2, 3, 4	First job: 360 + a random variable from uniform distribution in [0, 60] from planning time; The other 19 job's due time in Set 1 is a random variable from uniform distribution U(108, 132) seconds after the due time of its previous job. It is U(90, 110) for Set 2.	Each job's arrival time will be a random value from a uniform distribution U(due time – 360, due time – 240).
QC2	Yard blocks 5, 6, 7, 8	As QC1's	
QC3	Yard blocks 3, 4, 1, 2	As QC1's	
QC4	Yard blocks 7, 8, 5, 6	As QC1's	

The jobs that come to a yard block will have a randomly generated slot and row number. Other recent studies using randomized container locations include, for example, Zeng and Yang (2009). The tier numbers of the jobs are distributed according to Table 2. When a container is not at the top tier, reshuffling may be needed depending on the job sequence as discussed in Section 4.3.

We compare three YC dispatching algorithms: (1) RBA*: the algorithm (Guo et al. 2011) minimizes total job waiting time; (2) MMT-RBA*: the algorithm minimizes maximum job tardiness; (3) MMS-RBA*: the algorithm minimizes makespan of jobs. QCs generate jobs according to the setup in the respective tables for a planning window. The generated jobs are distributed into the job list for each yard block (YC). In each yard block, the YC dispatching algorithm plans the job sequence for the job list of the planning window. We compare the maximum tardiness of the jobs for each QC between RBA* and MMT-RBA* and between MMS-RBA* and MMT-RBA* respectively at the end of the planning window. We also compare the maximum tardiness for the vessel at the end of the planning window between RBA* and MMT-RBA* and between MMS-RBA* and MMT-RBA* respectively. The maximum tardiness for the vessel is the maximum tardiness of the longest QC serving the vessel. For each experimental setting, 50 independent runs are conducted and the results of the paired-t comparisons are reported.

Table 2: Distribution of job tier numbers.

Tier number of job	Percentage
Top tier	75%
2nd top tier	15%
3rd top tier	10%

5.2 Results and Discussions

A QC that experiences the longest delay in its operations due to waiting time for a vehicle will lengthen the vessel's turnaround time. Therefore the performance of various algorithms in YC dispatching is measured by the maximum delay among the QCs serving the vessel. The delay for a QC i under algorithm x from simulation run r is the maximum tardiness of the jobs for the QC,

$$D_{x,r,i} = \max(d_1, d_2, ..., d_l) \quad (13)$$

where $d_1, d_2, ..., d_l$ are the tardiness of the sequence of jobs of this QC in the planning window.

The delay to a vessel under algorithm x from simulation run r is defined by the tardiness of the vessel:

$$VT_{x,r} = \max_{i \epsilon QCs\ serving\ the\ vessel}\left(D_{x,r,i}\right). \quad (14)$$

We have explained in Section 3.2 that MMT-RBA* produces the minimum vessel delay. Therefore we conduct the paired-t comparison between RBA* and MMT-RBA* and between MMS-RBA* and MMT-RBA*. The comparison is done for both QC delays and vessel delays. The result of the paired-t comparison of QC delay between algorithm x and MMT-RBA* is presented by

$$\frac{1}{50}\sum_{r=1}^{50}\left(D_{x,r,i} - D_{MMT-RBA*,r,i}\right) \pm t_{49,0.95}\frac{QS_{x-MMT-RBA*}}{\sqrt{50}}. \quad (15)$$

$QS_{x\text{-}MMT\text{-}RBA*}$ is the standard deviation of the differences of QC delay between x and MMT-RBA*. This gives the individual 95% confidence interval of the difference in the delay of QC i as shown in the columns under QC1 – QC4 in Tables 3 and 4.

The result of the paired-t comparison of vessel delay between algorithm x and MMT-RBA* is presented by

$$\frac{1}{50}\sum_{r=1}^{50}\left(VT_{x,r} - VT_{MMT-RBA*,r}\right) \pm t_{49,0.975}\frac{VS_{x-MMT-RBA*}}{\sqrt{50}}. \quad (16)$$

$VS_{x\text{-}MMT\text{-}RBA*}$ is the standard deviation of the differences of vessel delay between x and MMT-RBA*. This gives the individual 97.5% confidence interval of the difference in the vessel delay as shown in the last column of Tables 3 and 4. The confidence level of the conclusions about vessel delays of the two algorithms, RBA* and MMS-RBA*, against MMT-RBA* is 95%.

From tables 3-4, we can see that MMT-RBA* produces lower average delays for all the QCs when compared with RBA* and when compared with MMS-RBA*. MMT-RBA* also produces minimum average delays for the vessel when compared with RBA* and when compared with MMS-RBA*.

Table 3: Set 1: Paired-t comparison of QC delay and vessel delay (seconds).

	QC1	QC2	QC3	QC4	Vessel
RBA*- MMT-BA*	37.86±27.32	61.79±40.32	56.27±35.74	51.20±46.32	148.58±67.63
MMS-RBA* - MMT-RBA*	222.08±75.75	224.48±63.30	149.62±66.66	289.89±85.10	523.89±93.85

Table 4: Set 2: Paired-t comparison of QC delay and vessel delay (seconds).

	QC1	QC2	QC3	QC4	Vessel
RBA*- MMT-BA*	329.81±91.39	347.22±95.17	346.11±88.76	309.22±95.83	660.21±137.21
MMS-RBA* - MMT-RBA*	331.50±82.58	322.79±78.87	275.01±65.35	302.48±69.14	562.66±114.76

If we take the lower boundary of the confidence interval of the difference between RBA* and MMT-RBA* (e.g. 148.58-67.63 in row 2 of Table 3) and divide by the upper boundary of the confidence interval of the vessel delay by MMT-RBA*, we get the lower bound of the percentage improvement of MMT-RBA* over RBA*. Similarly, the upper boundary of the confidence interval of the difference between RBA* and MMT-RBA* (e.g. 148.58+67.63 in row 2 of Table 3) divided by the lower boundary of the confidence interval of the vessel delay by MMT-RBA* returns the upper bound of the percentage improvement of MMT-RBA* over RBA*. The same can be calculated for the percentage improvement of MMT-RBA* over MMS-

RBA*. Table 5 shows the results. For example, MMT-RBA* produces 36.79% - 126.93% less vessel delay than RBA* in experiment Set 1. All the improvements are significant.

Table 5: percentage improvement of vessel delay by MMT-RBA* over RBA* and MMS-RBA*.

	Over RBA*		Over MMS-RBA*	
	Lower boundary	Upper boundary	Lower boundary	Upper boundary
% improvement (Set 1)	36.79%	126.93%	195.44%	362.68%
% improvement (Set 2)	348.95%	655.57%	298.84%	556.91%

6. CONCLUSIONS

We propose to minimize the maximum tardiness among the vehicle jobs for solving the YC dispatching problem. We present optimal algorithm MMT-RBA* to minimize maximum job tardiness for the YC dispatching problem. We also present optimal algorithm MMS-RBA* to minimize makespan in order to evaluate MMT-RBA*.

In real operations, containers involved in a YC job may not be on top of the stack. To handle such jobs, some containers have to be moved first. Such scenarios are very often ignored in some other studies. In order to consider such jobs in the planning of YC dispatching sequence, simulation methods in Huang et al. (2012) and Huang and Guo (2013) are used in our optimization algorithms to help provide accurate YC service times. This results in a more accurate evaluation of job tardiness.

Simulation experiments are conducted to evaluate the algorithms proposed, together with the optimal algorithm RBA* for minimizing total job waiting time. Our results show that MMT-RBA* is the best algorithm to minimize vessel tardiness in completing its loading and unloading operations.

Future work includes investigations into dynamic optimization algorithms that are effective under uncertainty: when predicted vehicle arrival times have noise.

REFERENCES

Cao Z., D-H Lee and Q. Meng. 2008. "Deployment strategies of double-rail-mounted gantry crane systems for loading outbound containers in container terminals." *International Journal of Production Economics* 115: 221–228.

Guo, X., S. Y. Huang, W,J. Hsu, M.Y.H. Low, T.H. Chan and J.H. Liu. 2007. "Vehicle Dispatching with real time location information in container terminals." In *Proceedings of the European Modeling and Simulation Symposium 2007*, edited by A. G. Bruzzone, F. Longo, Y. Merkuryev and M. A. Piera, 346-352. Bergeggi, Italy: Curran Associates, Inc.

Guo X., S.Y. Huang, W.J. Hsu and M.L.H. Low. 2011. "Dynamic yard crane dispatching in container terminals with predicted vehicle arrival information." *Advanced Engineering Informatics* 25(3): 472–484.

Guo X. and S.Y. Huang. 2012. "Dynamic space and time partitioning for yard crane workload management in container terminals." *Transportation Science* 46(1): 134-148.

Huang, S.Y. and X. Guo. 2013. "Reducing Simulation Costs of Embedded Simulation in Yard Crane Dispatching in Container Terminals." In *Proceedings of the 2013 ACM SIGSIM* Principles of Advanced Discrete Simulation, edited by M. L. Loper and G. A. Wainer, 305-314. Montréal, Québec, Canada: ACM.

Huang, S.Y., X. Guo, W.J. Hsu and W.L. Lim. 2012. "Embedding simulation in yard crane dispatching to minimize job tardiness in container terminals." In *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, 1646-1656. Berlin, Germany: Institute of Electrical and Electronics Engineers, Inc.

Jung S.H. and K.H. Kim. 2006. "Load Scheduling for Multiple Quay Cranes in Port Container Terminals." *Journal of Intelligent Manufacturing* 17: 479–492.

Kim K.Y. and K.H. Kim. 2003. "Heuristic Algorithms for Routing Yard-Side Equipment for Minimizing Loading Times in Container Terminals." *Naval Research Logistics* 50: 498–514.

Kim K.H., J.S. Kang and K.R. Ryu. 2004. "A Beam Search Algorithm for the Load Sequencing of Outbound Containers in Port Container Terminals." *OR Spectrum* 26(1): 93–116.

Kim, K. M. and K. Y. Kim. 1999. "An optimal routing algorithm for a transfer crane in port container terminals." *Transportation Science* 33(1): 17–33.

Kumar, M. M. and S.N. Omkar. (2008), "Optimization of yard crane scheduling using particle swarm optimizartion with genetic algorithm operators (psogao)." *Journal of scientific & industrial research* 67: 335-339.

Lee D-H., Z. Cao and Q. Meng. 2007. "Scheduling of Two-Transtainer Systems for Loading Outbound Containers in Port Container Terminals with Simulated Annealing Algorithm." *International Journal of Production Economics* 107: 115–124.

Li W., Y. Wu, M. Petering, M. Goh and R. d. Souza. 2009. "Discrete time model and algorithms for container yard crane scheduling." *European Journal of Operational Research* 198: 165–172.

Narasimhan A. and U.S. Palekar. 2002. "Analysis and Algorithm for the Transtainer Routing Problem in Container Port Operation." *Transportation Science* 36(1): 63–78.

Ng W.C. 2005. "Crane Scheduling in Container Yards with Intercrane Interference." *European Journal of Operational Research* 164: 64–78.

Ng W.C. and K.L. Mak. 2005a. "Yard crane scheduling in port container terminals." *Applied Mathematical Modelling* 29: 263-276.

Ng W.C. and K.L. Mak. 2005b. "An effective heuristic for scheduling a yard crane to handle jobs with different ready times." *Engineering optimization* 37(8): 867-877.

Park, T., R. Choe. S.M. Ok and K.R. Ryu. 2010. "Real-time scheduling for twin RMGs in an automated container yard." *OR Spectrum* 32: 593-615.

Stahlbock, R. and S. Voss. 2010. "Efficiency consideration for sequencing and scheduling of double-rail-mounted gantry cranes at maritime container terminals." *International Journal of Shipping and Transport Logistics* 2(1): 95-123.

Steenken, D., S. Voβ and R. Stahlbock. 2004. "Container terminal operation and operations research – a classification and literature review." *OR Spectrum* 26: 3-49.

Zeng, Q. and Z. Yang. 2009. "Integrating simulation and optimization to schedule loading operations in container terminals." *Computers & Operations Research* 36(6): 1935–1944.

AUTHOR BIOGRAPHIES

SHELLYING HUANG is a senior lecturer in School of Computer Engineering at Nanyang Technological University (NTU), Singapore. Her research interests are in intelligent decision support systems, simulation optimization, heuristics and logistic systems. Her email address is ASSYHUANG@ntu.edu.sg.

YA LI is a research associate in School of Computer Engineering at Nanyang Technological University (NTU), Singapore. She obtained her Master of Science from School of Computer Engineering, NTU, Singapore. Her research interests include simulation-based optimization, dispatching and scheduling problems. Her email address is LIYA@ntu.edu.sg.

XI GUO obtained her Ph.D. from School of Computer Engineering, NTU, Singapore. Her research interests include real time control, artificial intelligence, and efficiency enhancement techniques for simulation-based optimization. She is now with Murex (Singapore). Her email address is guox0006@ntu.edu.sg.