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 DEPLOYMENT IN CONTAINER TERMINALS

Shell Ying Huang
Ya Li
Meimei Lau
Teck Chin Tay

School of Computer Engineering
Nanyang Technological University
639798, SINGAPORE

ABSTRACT

A three-level, hierarchical system for yard crane (YC) management in container terminals and the algorithms for the bottom two levels were proposed in previous studies. The bottom two levels are responsible for YC job sequencing and intra-row YC deployment. This paper presents YC deployment strategies for inter-row YC deployment. The objectives are to minimize vehicle waiting times and the number of overflow jobs. We show by realistic simulation experiments that (1) when the number of yard cranes is less than the number of yard blocks, deploying YCs in proportion to the number of jobs in each row (3L-Pro-Jobs) is the best; (2) when the number of yard cranes is equal to or more than the number of yard blocks, the apparent workload approach, 3L-AW, performs best.

1 INTRODUCTION

Previous studies have pointed out that yard crane (YC) operations are of great importance and likely to be a potential bottleneck to the overall container terminal performance (Li et al. 2009). In the majority of the container terminals in the world, the container storage yard is still manually operated with Rubber Tyred Gantry Cranes (RTGs). Figure 1 shows a large part of a typical layout of such a container terminal. In such terminals, the storage yard is often divided into several tens of yard blocks in a number of rows parallel to the quay. Each yard block may have more than 30 slots (yard bays) of containers stored in length. Vehicles travel along lanes to the side of a yard block to transfer containers between quayside and yard side. For transshipment-intensive terminals, most of the yard activities are for storing containers unloaded from vessels and retrieving containers to load onto vessels. If multiple vessels are loading and unloading at the same time, in-terminal vehicles may arrive at different slot locations (bays) of a yard block for storing and retrieving containers. Yard cranes are the interface between vehicles and container stacks in the storage yard.

Minimizing vessel turnaround time is one of the most important objectives of container terminals. This means to keep a continuous flow of vehicles at the quayside to support non-stop Quay Crane (QC) operations. At the same time, it is also important to maintain high productivity of vehicles and reduce the waiting time of the vehicles at the quayside and at the yard blocks. Therefore YCs should try to minimize the waiting times of the vehicles to free them in a timely manner to satisfy the demand for them at the quayside. This paper studies the YC deployment problem, that is, where to send the YCs in the storage yard in order to minimize the average vehicle job waiting time.

In a high throughput terminal, multiple berths with vessels of different sizes are often in operation simultaneously. When a vessel finishes its loading/unloading operations, it will leave and another vessel will soon come in. So the number of vessels berthed and thus the number of quay cranes working at

978-1-4799-7486-3/14/$31.00 ©2014 IEEE

instant changes dynamically over time. It follows that the total amount of work to the entire terminal yard changes dynamically over time. Even during a period with the same set of vessels, the level of workload in different parts of the yard may vary greatly. This is because containers unloaded from a vessel will be carried to import container blocks or different parts of the yard depending on their respective second carriers. Similarly, containers to be loaded onto a vessel may also come from various parts of the yard in specific orders matching the vessel's downstream port calls in its voyage. Therefore workload distribution in the yard is uneven and changes dynamically over time. The important task of YC deployment is to assign YCs to various parts of the yard at proper time moments to continuously match the changing distribution of workload.

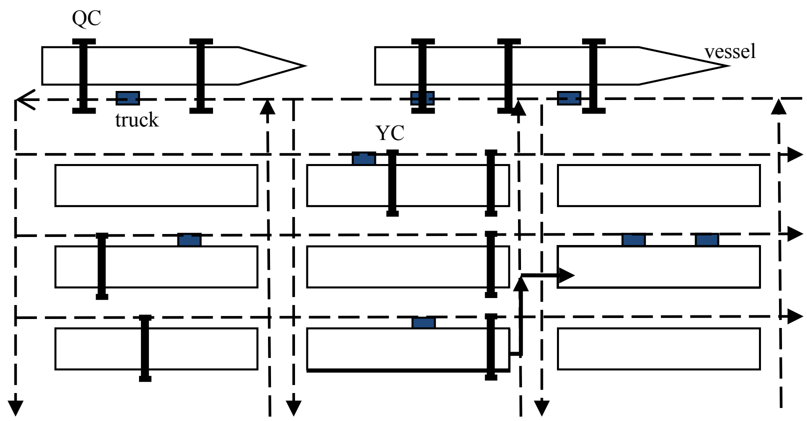

Figure 1: Part of a typical layout in a RTG yard.

To match the changing workload distribution, YCs need to move from time to time. The time taken by an YC to move from one location to another is referred to as gantry time. The movements of a Rubber Tyred Gantry (RTG) Crane include linear gantry and cross gantry. Linear gantries are movements along the lane in the same row of yard blocks with a similar speed to intra-block linear gantry. Cross gantries are movements from one block to another in a different row. Figure 1 shows the trajectory of a YC moving from the second block in Row 3 to the third block in Row 2. A YC doing such a cross gantry has to make two 90° turns which take longer than a linear gantry and may delay the vehicle movements by blocking the lanes. Both YC gantry times and YC service times contribute to vehicle waiting times. YC gantry times, if not properly managed, may become a significant part of an YC's busy time and reduce YC productivity.

After equipment ordering at the beginning of a shift, a common practice (Cheung et al. 2002; Linn and Zhang 2003; Ng 2005; Zhang et al. 2002) is to initially assign YCs to various yard blocks. Then a re-distribution of YCs among the yard blocks is done at the beginning of the subsequent planning windows to match the dynamically changing workload. Each planning window is usually one hour, two hours or even four hours. Under this practice some blocks may have no YCs assigned to them for the entire planning window. This will cause very long vehicle waiting times in these blocks. Since the distribution of YCs does not change within each planning window, there is no way to dynamically respond to changes in workload distribution in the storage yard before the end of the planning window. If the length of the planning window is longer than the durations of relatively constant workload, this will lead to the situation where vehicles are deprived from prompt YC service in some parts of the yard and YCs are under-utilized in other parts of the yard. On the other hand, if the length of the planning window is short it may cause frequent cross gantry movements of yard cranes with substantial loss of productive working time. Under this practice, it may also deploy more than one YC to some blocks. This means carefully synchronized YC operations are needed to avoid YC clashes. Every time YCs block each other, waiting, thus loss of productivity, is unavoidable.

In previous works (Cao et al. 2008; Cheung et al. 2002; Jung and Kim 2006; Kim and Kim 2003; Kim et al. 2004; Lee et al. 2007; Linn and Zhang 2003; Ng 2005; Ng and Mak 2005a, 2005b; Zhang et al. 2002) on YC deployment, the workload of a block is commonly represented by the number of arriving jobs expected in a planning window or a quantity proportional to this number. The number of jobs is only one factor contributing to the busy time a YC will have. For the same number of jobs, one YC may be very idle and another very busy. The difference is because of the different dispatching sequences and the locations of the jobs which results in different inter-job gantry time.

Level 1	Distribution of YCs to different rows for each planning window in a shift
Level 2	Time Partition of a planning window into sub-planning windows
	Space Partition of each row into YC working zones for each sub-planning window
Level 3	YC dispatching in individual working zones (job sequencing)

Figure 2: Hierarchical organization of the management functions of YC operations.

In our earlier studies, a hierarchical system for YC operation management was proposed. The system is organized into three levels as shown in Figure 2 (Guo and Huang 2012). Level 1 distributes YCs among different rows of yard blocks at suitable times based on predicted future jobs in the yard. Level 2**,** which is the focus of the paper by Guo and Huang (2012), dynamically assigns YCs in each row to non-overlapping zones. There will be no part of a yard block that does not belong to a working zone of some YC. So no job will be deprived of YC service in the deployment plan in the system. Working zones are separated by safe distances so YCs will not block each other in their operations. Level 3 (Guo et al. 2011) determines the serving sequences of vehicle jobs for an YC in a service zone which minimizes average vehicle waiting time. Time-consuming YC cross gantry movements are limited to Level 1 and therefore are performed with relatively low frequency. Quick responses to the changes in workload distribution are supported by dynamic deployment plans in Level 2. Level 3 is also used by Level 2 in generating deployment plans that minimize average vehicle waiting times in the entire rows of yard blocks.

Inter-block YC deployment problem has been proved to be NP-hard in the strong sense (Cheung et al. 2002). This paper proposes heuristics to solve the YC deployment problem at Level 1. The strategies proposed and evaluated are (1) YC deployment based on the number of jobs, (2) YC deployment using the least cost approach, (3) YC deployment using the apparent workload approach. The method to select the YCs to perform cross gantry to a different row that will increase the utilization of YC idle time at the end of planning windows and reduce YC congestion and delay to vehicle traffic is also proposed. The YC deployment schemes are evaluated by experiments simulating scenarios where workload in the storage yard is changing dynamically both in time and in space.

In the rest of the paper, we first briefly review the related work in literature in Section 2. Then we propose our YC deployment schemes in Section 3. Section 4 presents the simulation experiments and the evaluation results. Section 5 concludes the paper.

2 RELATED WORK

Zhang et al. (2002) studied the problem of YCs deployment among yard blocks. It is formulated as a MIP model to minimize total unfinished workload at the end of each planning period and is solved by a modified Lagrangian relaxation method. However, only one transfer per YC is allowed in the 4-hour planning period, which may be insufficient to match the changing workload distribution. Cheung et al. (2002) also studied the problem with a Lagrangian decomposition solution and a successive piecewise-linear approximation approach. Their computational experiments show that successive piecewise-linear approximation method is both effective and efficient in managing 24 yard blocks, 24 YCs for an eight hour shift. The workload in a yard block in these two papers is based on the number of container moves (equivalent to the number of YC jobs). Because YCs need to move from one slot position (yard bay) to another in between jobs, a YC's busy time greatly depends on the actual sequence in which the YC serves the jobs. The waiting times of the vehicles are also very much affected by the sequence in which the YC serves their jobs. So the number of container moves is at best a coarse estimation of a YC's workload. In the two papers the amount of work done in a container block per planning period is assumed to be proportional to the number of cranes in the block during that period. This is not very realistic because not all cranes can work simultaneously at all times due to crane interference.

Ng (2005) considered the problem of multiple YCs sharing a single bi-directional lane and modeled it as an IP to minimize total job completion time. A two-phase algorithm is proposed. The first phase uses a dynamic programming approach to compute the best workload partition. The workload for each YC is roughly estimated by a simple greedy heuristic. The greedy heuristic works as follows: Among all the jobs not yet scheduled, compute the completion time for each of these jobs and select the job with the earliest job completion time as the one to be handled next. The second phase exchanges jobs between neighboring YCs to further improve the performance. However, YC deployment among various rows was not mentioned and safety constraints of YC separation by a minimal distance were not considered. Petering et al. (2009) claimed that when YCs are in charge of overlapping zones, YCs could only schedule for at most 1.5 container jobs to avoid deadlocks. This reinforces our belief that non-overlapping YC zones should be formed in our YC dispatching schemes to achieve high performance while maintaining safety constraints and avoiding deadlocks in real-time settings. The three-level hierarchical YC management scheme (Guo and Huang 2012) partitions a row of yard blocks into a number of non-overlapping zones dynamically.

A Least Cost Heuristic (LCH) was proposed by Linn and Zhang (2003) to offer a computationally less expensive solution than that in Zhang et al. (2002). The basic idea of LCH is to deploy YCs to blocks such that the total amount of remaining work in the yard at the end of each planning period is minimized. When estimating the amount of unfinished work, LCH takes into consideration the time needed by YCs in their gantry movements from one block to another. Their experiments are conducted in a yard with five rows and two columns of yard blocks. Their results show that when the vehicle arrivals to the yard follow a uniform distribution, the amount of overflow work produced by LCH was shown to be 3-6% higher than the solution by the MIP model. The computational time needed by LCH is less than 1 second in all their tested scenarios where the MIP solution takes up to 440 seconds. This shows that LCH is very efficient in computational time.

An improved version of the LCH algorithm (ILCH) was proposed by Huang and Guo (2011). Average vehicle waiting times are reduced significantly in the experiments of a yard of four rows of five blocks each.

3 YARD CRANE DEPLOYMENT AMONG DIFFERENT ROWS

The objective of the Level 1 module in the hierarchical YC management system is to deploy YCs among different rows to achieve high service quality in the whole storage yard. This means job waiting times and the number of overflow jobs at the end of the planning windows should be minimized. An overflow job is one which arrives at a yard block within a planning window of Level 1 module but is not served by a YC in the planning window. Since dynamic balancing of workload among YCs in each row in terms of job waiting times is done at Level 2 for each planning window (Guo and Huang 2012), the length of the planning window at Level 1 is kept at a constant length. In other words, re-distribution of YCs among different rows is done periodically based on an estimation of the workload in each row in the next deployment period. In deciding the length of the planning windows at Level 1, two criteria need to be considered. First is the length of the planning window has to be short enough so that the information of the jobs that will arrive in the storage yard within the planning window can be estimated. This information includes their arrival times in addition to their locations. Second is the length of the planning window has to be long enough so that the optimization at level 2 and 3 is effective. Note that it is impractical to predict accurately how many jobs may be expected in each row of yard blocks for a long time period. Therefore the planning intervals at Level 1 are set at half-hour frequency. We investigate a number of strategies for YC deployment at Level 1.

3.1 Deploying YCs based on the number of jobs in the rows

This is to allocate a number of YCs to each row in proportion to the number of jobs the row expects in the planning window. The advantage of this strategy is the more jobs a row expects in a planning window, the more YCs will work to meet the demand on YC service. The algorithm is given in Figure 3. It first computes the number of YCs needed in each row R as the biggest integer smaller than or equal to $\frac{number\ of\ jobs\ in\ R}{total\ number\ of\ jobs} \times m$ where m is the total number of YCs. If there are remaining YCs they will be allocated to rows in descending order of the fractional part of $\frac{number\ of\ jobs\ in\ R}{total\ number\ of\ jobs} \times m$.

```
m = total number of YCs;
Rem = m;
For each row R in the yard
    N_i = ⌊(number of jobs in R / total number of jobs) × m⌋; // ⌊ ⌋ returns the integral part of a real number
    Rem = Rem – N_i;
If Rem > 0
    Sort the rows in descending order of ((number of jobs in R / total number of jobs) × m − N_i);
    Add 1 YC to each row in this order and update N_i until Rem = 0;
```

Figure 3: Proportional Distribution Algorithm.

```
For each row i starting from the first row to the last row
    If (N_i < the current number of YCs in the row)     // this row has surplus YCs
    For each surplus YC in row i
        Destination row = search for a row that needs more YCs starting from Row 1
        YC to move = the YC which will arrive in the destination row the earliest by taking
            the fastest path including the possible delay due to other YC
            movements already scheduled
        Schedule the cross gantry moves for the YC to move
```

Figure 4: Deciding on YC movements between rows.

The Level 1 module also decides which YC should be moved if a re-distribution of YCs among different rows is required. After deciding on the number of YCs each row should have for the next planning window, the next decision is to decide on the movement of YCs. In choosing which YCs should move to which row, a few factors are taken into consideration. First, it should minimize the total distance of YC moves in the redistribution of them among the rows. Second, it should minimize the number of 90-degree turns YCs have to make because 90-degree turns are costly in time. Third, the arrival times of YCs in their destination rows should be as early as possible, that is, YCs which finish their work before the end of the current planning window should be chosen to move. Fourth, if multiple YCs need to move, YC clashes or congestions should be avoided.

The algorithm is given in Figure 4. The algorithm is guaranteed to produce the minimum YC moves because no two YCs going from their source row to destination row will cross each other. Only the YCs in a row with surplus YCs are involved in the YC moves. For example, if Row 3 has one surplus YC and

Row 1 needs one, a YC will move from Row 3 to Row 1. The system will not move one YC from Row 3 to Row 2 and move another YC from Row 2 to Row 1. Even though it is possible that these two YCs move at the same time (if they finish their operations around the same time), both YCs need to make two 90-degree turns. The total time the two YCs spent on cross gantry movement will be significantly longer than if one YC moves from Row 3 to Row 1, with the likely consequence of more delays to the YC operations. Cross gantry movements by two YCs will also block the vehicle paths for a longer time, increasing chances of YC congestions and vehicle congestions. For each row that has surplus YCs, it will select the YC among the ones in the row that will arrive in the destination row the earliest, considering when the operations of each YC will end in the source row in the current planning window, the path each YC may take thus the time for its cross gantry moves and the possible blocking by another YC's move already scheduled.

3.2 Deploying YCs using the least cost approach

The second strategy is adapted from the idea of the least cost heuristic (Linn and Zhang, 2003). At the beginning of each deployment period, each row of yard blocks in the storage yard is classified into one of three classes: demand row, supply row or satisfied row. A demand row is one where the workload in the row is more than the total capacity of the existing YCs in the row and the number of YCs is less than a predefined maximum number of YCs allowed in a row. A supply row is one where the workload in the row is less than the total capacity of the existing YCs in the row. A row is satisfied if it is neither a demand row nor a supply row. A crane in a supply row may either stay in its original row or move to a demand row based on the evaluation of the cost in terms of the remaining work in supply rows and demand rows. The outline of the algorithm is given in Figure 5.

```
At the beginning of each deployment period
    Identify supply rows and demand rows;
    Construct the cost matrix;
    While there are both supply row(s) and demand row(s)
    For each supply row i
        For each demand row j
            Compute the cost of moving one YC from i to j;
    Make YC deployment decision by choosing the least cost element in the cost matrix;
    Update the status of the rows affected by the YC re-deployment;
```

Figure 5: Least cost distribution method.

In essence, this is the LCH applied to rows of yard blocks. The same logic of computing the total capacity of the existing YCs as in Linn and Zhang (2003) is used. That is,

Total capacity = (the number of cranes in the row) x (60 minutes).

The workload of a block in Linn and Zhang (2003) is defined as

Workload = (the number of jobs) x (average YC service time per job).

We use a more accurate model of workload for a row of yard blocks which includes the estimated YC gantry time between jobs:

Workload = (the number of jobs) x (average YC service time per job + average gantry time between jobs)

where the average gantry time between jobs in a row with multiple YCs is estimated as

average gantry time between jobs = ((length of row / number of YCs) / 2) / YC gantry speed.

After the decisions of how many YCs should be re-deployed to a different row from their current row, the algorithm in Figure 4 is used to decide which YC in the supply row to move and which path to take for the cross gantry.

3.3 Deploying YCs using the apparent workload approach

We propose an Apparent Workload-based (AW) deployment rule to decide on which row YCs should be deployed. The objective is to consider multiple aspects of the workload in a row. We expect three main factors that would contribute to a heavier workload in a row: (1) a higher number of jobs expected in the next planning window in a row, (2) a higher number of blocks these jobs spread to, (3) a longer expected inter-job gantry time. Therefore more YCs should be sent to such a row. The apparent workload of a row is represented by an index value which is the sum of three terms. They are the J(obs) term, the B(locks) term and the G(antry) term. The deployment algorithm starts with one YC in each row. Then the index value of each row is calculated based on the current number of YCs in the row, the number of jobs in the row, the number of working blocks in the row and the estimated YC gantry distance. The next YC will be deployed to the row with the highest index value. The deployment process is repeated until all the YCs are deployed.

The J term for a row i is

$$\mathrm{J}(i) = exp\left(w_1 \frac{number\ of\ jobs\ per\ YC\ in\ row\ i}{average\ number\ of\ jobs\ per\ YC\ for\ all\ rows}\right)$$

where w_1 is the weight of the term and $number\ of\ jobs\ per\ YC\ in\ row\ i = \frac{number\ of\ jobs\ in\ row\ i}{number\ of\ YCs\ in\ row\ i}$.

The B term for a row i is

$$\mathrm{B}(i) = exp\left(w_2 \frac{number\ of\ working\ blocks\ per\ YC\ in\ row\ i}{average\ number\ of\ working\ blocks\ per\ YC\ for\ all\ rows}\right)$$

where w_2 is the weight of the term and $number\ of\ working\ blocks\ per\ YC\ in\ row\ i = \frac{number\ of\ working\ blocks\ in\ row\ i}{number\ of\ YCs\ in\ row\ i}$.

The G term for row i is

$$\mathrm{G}(i) = exp\left(w_3 \frac{average\ gantry\ distance\ per\ YC\ in\ row\ i}{average\ of\ the\ average\ gantry\ distance\ per\ YC\ for\ all\ rows}\right)$$

where w_3 is the weight of the term and

$$average\ gantry\ distance\ per\ YC\ in\ row\ i = \frac{total\ inter-job\ gantry\ distance\ in\ row\ i}{number\ of\ YCs\ in\ row\ i}.$$

The *total inter-job gantry distance* is estimated by adding up the inter-job gantry distance in the order of the job arrivals to the row.

Again, the algorithm in Figure 4 is used to decide which YC to move and which path to take for the cross gantry.

4 PERFORMANCE EVALUATION

The YC deployment algorithms proposed for Level 1 of the hierarchical YC management system are evaluated by simulation experiments. Five deployment schemes for the hierarchical management system and one yard block based deployment scheme are compared. In the following, the first is yard block based and the rest are under the hierarchical YC management system.

1) ILCH. The Improved least cost heuristic proposed by Huang and Guo (2011).
2) 3L-Pro-Jobs. Deploying YCs based on the number of jobs in the rows.
3) 3L-LCH. Deploying YCs using the Least Cost Approach.

4) 3L-AW. Deploying YCs using the apparent workload approach.
5) 3L-Eq. Static equal distribution where equal number of YCs are distributed across the rows with no YC re-deployment to other rows.
6) 3L-SCJF. This is the 3-level hierarchical YC management scheme where Level 3's optimal YC job sequencing algorithm is replaced by the Smallest Completion Time Job First heuristic (SCJF). At Level 1, deploying YCs based on the number of jobs in the rows is used. SCJF is also used in ILCH to sequence jobs in individual yard blocks. 3L-SCJF is used to remove the advantage of the optimal algorithm in Level 3 of the hierarchical scheme over ILCH in the comparison with ILCH.

4.1 Experimental Design

The terminal in the performance evaluation has a layout similar to Figure 1 with 4 rows of blocks in the yard and 5 yard blocks in each row. Each block has 36 yard slots (bays). 16, 20, or 24 yard cranes are used in the different sets of experiments. It is assumed in the simulation experiments that an YC takes 2.3 minutes to make a linear gantry movement from a block to a neighbouring block in the same row. This is calculated based on realistic YC linear gantry speed. It is also assumed that an YC takes 15 minutes to execute a cross gantry movement from one row to a neighbouring row which includes two 90-degree turns. Each additional row to cross needs an additional 2.3 minutes.

As reshuffling operation of containers in the yard is commonly done in the lull periods of yard operations, containers to be retrieved are assumed to be already on the top of the stacks and containers to be stored in yard will be placed on top of their stacks. The YC process time is thus assumed to be 120s for each container job (loaded or empty), same as in Jung and Kim (2006). The usage of constant YC process time is also seen in Lee et al. (2007). The simulation model is developed under Visual C++ compiler.

It is not the focus of this paper to study how to predict vehicle arrivals by real time data driven simulation or other methods. So vehicle inter-arrival times (IAT) follow an exponential distribution for a mixture of storing/retrieval jobs for multiple vessels. The dynamically changing workload in the container storage yard both in time and space is simulated by two processes within a 24-hour period.

Table 1: Dynamically changing workload to the yard

Time	Space
Sub-period1 U(120,360) minutes, IAT has exponential distribution with mean = U(15, 25)s in each sub-period1	Sub-period2 U(30,90) minutes, Different probability (between 10% and 40% of total workload) to each row in each sub-period2

First, a 24-hour period consists of a number of sub-periods. Each of these sub-periods has a different mean job arrival rates to the entire storage yard. This presents different total workloads to the entire terminal yard in the various sub-periods. The varying total workload is due to the changing number of simultaneously working quay cranes caused by vessel arrivals and departures. The length of each such period is a random value generated from uniform distribution U(120, 360) minutes. This means after a period which lasts between 2 and 6 hours, the total workload to the yard will change. We call each of such periods a sub-period1. For each sub-period1, the mean job inter-arrival time (IAT) is a random value generated from uniform distribution U(15, 25) seconds. When the mean IAT is 15 seconds, it represents an average of 240 containers per hour to be loaded or unloaded from/to the yard. This is the workload put to the storage yard by 8 quay cranes working continuously at the full speed of 30 moves per hour. It should be noted that quay cranes do not work at full speed all the time. When it finishes the loading or unloading operations at one vessel bay, it moves to another bay before continuing its operation. Sometimes a quay crane also needs to wait due to crane clashes or other problems.

In the second process, each sub-period1 consists of a number of time periods (sub-period2) where the space distribution of jobs changes dynamically. This represents the situation where even with the same total workload for the whole storage yard, container job locations change when a quay is handling containers going for different destination ports or different second carriers. When simulating the dynamically changing space distributions of yard jobs, the percentages of jobs in the various rows in the yard are changed from one sub-period2 to the next. The length of each sub-period2 is a random value from U(30, 90) minutes.

The job distribution to the various rows in the simulation starts from 25% of the total workload to each row of the 4 rows of yard blocks in the yard. At the end of each sub-period2, a random value from U(1%, 5%) is generated, say x%. Then 2 rows are chosen randomly as source rows where workload will be reduced by x% and 2 other rows are chosen randomly as target rows where workload will be increased by x%. It may happen sometimes that the 2 source rows chosen randomly happen to be the same row or the 2 target rows chosen randomly happen to be the same row. We consider the scenario where a row has less than 10% or more than 40% of the whole yard's workload as extreme conditions. Our simulation will not study such a scenario. So if a row is expected to reach such a condition after a reduction or increase of workload, a different row will be chosen.

The dynamically changing workload to the yard is summarized in Table 1. Each setting has 30 independent replications of simulation runs and the results are summarized in the next section.

Table 2: Average vehicle waiting time (seconds)

	QCs	ILCH	3L-Eq	3L-Pro-Jobs	3L-LCH	3L-AW	3L-SCJF
average	16	2677.56	70.79	65.27	82.77	66.98	65.7
	20	684.76	38.06	36.92	38.11	35.15	37.19
	24	376.78	25.97	25.16	34.29	24.27	25.22
90^{th} percentile	16	3985.93	91.81	82.59	102.65	86.85	83.89
	20	1168.14	48.93	46.43	48.97	43.51	46.67
	24	583.93	33.39	30.57	55.54	30.33	30.52

Table 3a: Paired-t comparison of the average vehicle waiting time against 3L-Pro-Jobs with 16 QCs

	ILCH	3L-Eq	3L-LCH	3L-AW	3L-SCJF
average±half width of C.I	2612.29±437.04	5.52±2.27	17.50±4.60	1.70±1.49	0.43±0.42

Table 3b: Paired-t comparison of the average vehicle waiting time against 3L-AW (20 & 24 QCs)

	QCs	ILCH	3L-Eq	3L-Pro-Jobs	3L-LCH	3L-SCJF
average±half width of C.I	20	649.62±204.65	2.91±0.78	1.77±0.69	2.97±0.79	2.04±0.71
	24	352.51±145.08	1.70±0.59	0.89±0.44	10.03±4.24	0.95±0.44

4.2 Evaluation Results

In three sets of experiments, 16, 20 or 24 yard cranes respectively are deployed in the yard of 20 yard blocks with job arrival rate matching that of up to 8 quay cranes working simultaneously. Table 2 presents the average and the 90^{th} percentile of the vehicle waiting times under the various YC deployment algorithms. Table 4 presents the average number and the 90^{th} percentile of overflow jobs under these algorithms. Tables 3a and 5a show the paired-t comparison of the performance of the five algorithms against 3L-Pro-Jobs with 16 QCs computed by

$$mean_{pairs\ of\ results}(Performance\ of\ \ x\ -\ Performance\ of\ 3L-Pro-Jobs) \pm t_{29,0.01}\frac{Stdev}{\sqrt{30}}$$

since 3L-Pro-Jobs has the lowest average job waiting time. *Stdev* is the standard deviation of the differences of the 30 paired results of the two algorithms from the simulation runs. For each algorithm, this gives the 98% confidence interval of the difference between x and 3L-Pro-Jobs. The overall confidence on the differences between the five algorithms and 3L-Pro-Jobs is 90%. Tables 3b and 5b show the paired-t comparison of the performance of the five algorithms against 3L-AW with 20 and 24 QCs computed by

$$mean_{pairs\ of\ results}(Performance\ of\ x\ -\ Performance\ of\ 3L-AW) \pm t_{29,0.01}\frac{Stdev}{\sqrt{30}}$$

since 3L-AW has the lowest average job waiting time.

All tables clearly show that the 3-level yard crane deployment scheme is much better than the yard block based ILCH (improved least cost heuristic) for all scenarios. The main factor for the good performance of the 3-level scheme is that cross gantry of yard cranes are restricted to Level 1 and the entire row of yard blocks is considered as one unit of workload. Therefore the ups and downs in the individual block's workload will have less chance of resulting in an expensive cross gantry movement as in ILCH.

Table 4: Average number of overflow jobs

QCs	ILCH	3L-Eq	3L-Pro-Jobs	3L-LCH	3L-AW	3L-SCJF
16	220.02	4.46	3.7	5.45	4.04	3.78
20	47.99	2.07	1.89	2.07	1.85	1.91
24	23.68	1.36	1.32	2.04	1.27	1.32

Table 5a: Paired-t comparison of the average number of overflow jobs against 3L-Pro-Jobs with 16 QCs

	ILCH	3L-Eq	3L-LCH	3L-AW	3L-SCJF
average±half width of C.I	216.32±35.97	0.75±0.24	1.75±0.41	0.33±0.15	0.07±0.04

Table 5b: Paired-t comparison of the average number of overflow jobs against 3L-AW (20 & 24 QCs)

	QCs	ILCH	3L-Eq	3L-Pro-Jobs	3L-LCH	3L-SCJF
average±half width of C.I	20	46.14±14.55	0.23±0.10	0.04±0.07	0.22±0.10	0.06±0.07
	24	22.42±9.33	0.09±0.05	0.06±0.04	0.77±0.39	0.06±0.04

Among the 3-level systems with different components, Tables 2-5 show some interesting phenomena.

1) When the number of YCs is 16, Tables 3a and 5a show that 3L-Pro-Jobs performs best, both in terms of vehicle waiting times and the number of overflow jobs. This suggests that when the number of YCs is less than the number of yard blocks, deploying YCs in proportion to the number of jobs in rows is the most suitable method. From Table 2's average job waiting times and Table 4's average number of overflow jobs, it can be seen that 3L-SCJF is the second best. 3L-SCJF uses the same method as 3L-Pro-Jobs in Level 1. Therefore the difference between this two methods is due to the optimal job sequencing algorithm at Level 3. 3L-AW is the next, less than 3% worse than 3L-Pro-Jobs on average and less than 6% for the 90th percentile in terms of vehicle waiting time. From Table 4's average number of overflow jobs, it can be calculated that 3L-AW is about 9% worse than 3L-Pro-Jobs. 3L-LCH and 3L-Eq are far worse than 3L-Pro-Jobs.
2) When the number of YCs is 20, Table 3b shows that 3L-AW produces the smallest average job waiting time, better than all other methods. Table 5b shows that 3L-AW has smaller average

number of overflow jobs than ILCH, 3L-Eq and 3L-LCH. However, there is no significant difference between 3L-AW and 3L-Pro-Jobs and between 3L-AW and 3L-SCJF in the average number of overflow jobs. Combining the two performance metrics, 3L-AW is the best.

3) When the number of YCs is 24, Tables 3b and 5b show that 3L-AW is superior to all the other methods, both in terms of vehicle waiting times and the number of overflow jobs. From Table 2's average job waiting times, 3L-AW is 3.6% better than the second best 3L-Pro-jobs. From Table 4's average number of overflow jobs, 3L-AW is about 4.7% better than the closest competitor, 3L-Pro-Jobs and 3L-SCJF.

5 CONCLUSIONS

We propose several algorithms for deploying yard cranes among the rows of yard blocks in a container storage yard. This works as the Level 1 component in the 3-level hierarchical yard crane management system proposed in earlier works. Our evaluation experiments using realistic simulation of container job arrivals show that the hierarchical system performs much better than the block based YC deployment scheme ILCH in the minimization of vehicle waiting times and the number of overflow jobs. Among the various schemes investigated, if the number of YCs is less than the number of yard blocks, deploying YCs in proportion to the number of jobs in each row is the best. When the number of YCs is equal to or more than the number of yard blocks, deploying YCs using the apparent workload approach will be the best. This is because the apparent workload approach considers not only the number of jobs but several factors including the YC gantry movement times.

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.

Cheung R.K., C-L. Li and W. Lin. 2002. "Interblock Crane Deployment at Container Terminals." *Transportation Science* 36(1), 79-93.

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. 2011. "An Improved Least Cost Heuristic for Dynamic Yard Crane Deployment in Container Terminals." *7th Annual IEEE Conference on Automation Science and Engineering*.

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.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.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.

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.

Linn R.J. and C. Zhang. 2003. "A heuristic for dynamic yard crane deployment in a container terminal." *IIE Transactions*, 35(2), 161-174.

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.

Petering M., Y. Wu, W. Li, M. Goh and R. d. Souza. 2009. "Development and Simulation Analysis of Real-Time Yard Crane Control Systems for Seaport Container Transshipment Terminals." *OR Spectrum,* 31(4), 801-835.

Zhang C., Y-W. Wan, J. Liu and R.J. Linn. 2002. "Dynamic Crane Deployment in Container Storage Yard." *Transportation Research Part B: Methodological,* 36(6), 537-555.

AUTHOR BIOGRAPHIES

SHELL YING 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.

MEIMEI LAU is a final year undergraduate students in School of Computer Engineering at NTU, Singapore. Her email address is mlau@ntu.edu.sg.

TECK CHIN TAY is a final year undergraduate students in School of Computer Engineering at NTU, Singapore. His email address is tctay@ntu.edu.sg.