OR Spectrum (2010) 32:633–661
DOI 10.1007/s00291-009-0183-6

REGULAR ARTICLE

Development and simulation analysis of real-time, dual-load yard truck control systems for seaport container transshipment terminals

Matthew E. H. Petering

Published online: 28 October 2009

Abstract We show how a seaport container terminal's long-run average quay crane rate depends on the system that automatically assigns yard trucks to container transportation jobs in the terminal in real time. Several real-time, dual-load yard truck control systems are proposed and evaluated by a fully-integrated, discrete event simulation model of a vessel-to-vessel transshipment terminal. The model is designed to reproduce the microscopic, stochastic, real-time environment at a multiple-berth facility. Overall, the literature still lacks a comprehensive analysis that (1) considers different methods for controlling dual-load vehicles in real time within a fully-integrated, stochastic container terminal environment and (2) compares them in terms of an absolute global performance measure such as average quay crane rate. This paper provides such an analysis.

Keywords Seaport container terminal · Real-time control · Dynamic vehicle routing · Gross crane rate · Simulation

1 Introduction

Today, most overseas shipping of furniture, toys, footwear, clothing, auto parts, bananas, and electronics components is done via standardized 20′, 40′, and 45′ long steel containers aboard deep-sea container vessels. With today's just-in-time global supply chain, improving the efficiency of container shipping processes is more important than ever. This paper focuses on operational control problems at seaport container terminals. Container terminals are the places in seaports where container vessels are

M. E. H. Petering (✉)
Department of Industrial and Manufacturing Engineering,
University of Wisconsin-Milwaukee, P.O. Box 784, Milwaukee, WI 53201, USA
e-mail: mattpete@uwm.edu

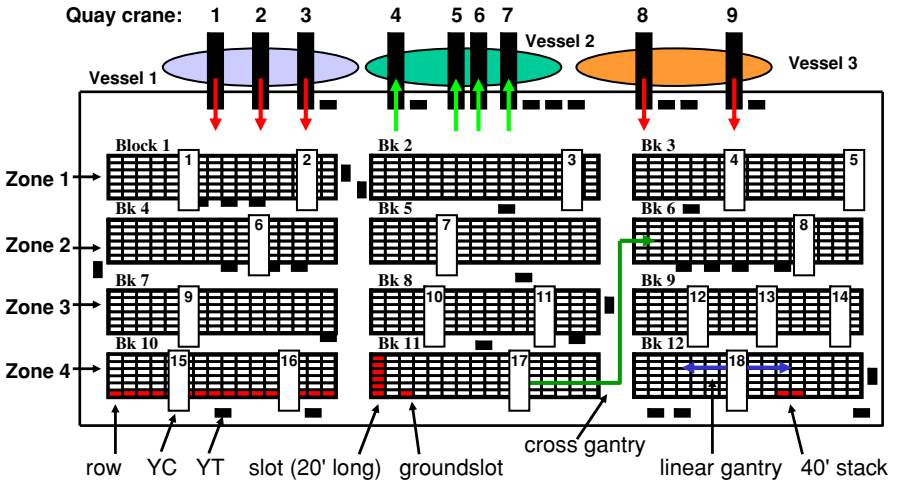

Fig. 1 Bird's eye view of a transshipment container terminal

loaded and unloaded, and where containerized cargo is temporarily stored while awaiting a future journey.

In this paper, we consider a land-scarce terminal in which containers are stacked high and equipment is manually controlled by human operators. This kind of terminal is found throughout Asia and other crowded regions of the world. In addition, we consider a pure transshipment terminal in which all containers are transshipped from vessel to vessel. That is, all cargo enters the terminal by vessel and departs by vessel. In 2007, the proportion of transshipment cargo at the world's busiest container port, Singapore, was about 80%. Transshipment cargo also had a 50% share at two other ports in the top 10—Dubai and Kaohsiung.

Figure 1 shows the general layout of a land-scarce, transshipment terminal. QCs (quay cranes) unload and load container vessels. They transfer cargo between vessels and YTs (yard trucks). YTs transport cargo between the shore and a storage yard (yard). YCs (yard cranes) transfer containers between YTs and stacks in the yard. Containers may spend 0–7 days in the yard before they are loaded onto a vessel.

The yard is divided into rectangular regions called *blocks*. The width of a block is typically divided into 7 *rows*—6 for stacks of containers and the seventh for YTs to interact with YCs. Traffic lanes for YTs occupy the spaces between blocks. Blocks are divided along their length into 20′ sections called *slots*. The region occupied by a stack of 20′ (40′) containers is a *groundslot* (40′ stack). 40′ stacks occupy two adjacent groundslots in the same row. In each stack, containers are stored one on top of the other 3 to 6 *tiers* high.

Yard cranes straddle the blocks beneath them and move along the lengths of the blocks. A *zone* is a sequence of blocks that together form a single lane for YC movement. In Fig. 1, blocks 1–3 are in zone 1, blocks 4–6 are in zone 2, and so on. YCs move easily within a block and from block to block within a zone; such movement is called *linear gantrying*. To move between zones, YCs must spend at least 15 min executing a maneuver called a *cross gantry*.

YTs carry one 20′, one 40′, one 45′, or two 20′ containers at a time. QCs load and unload containers according predetermined *job sequences* that have limited flexibility for changing the order in which lifts are performed. *In this study, we assume all QCs handle one container at a time*. A QC's top speed is typically 40 *lifts* (moves between vessel and shore) per hour if it does not have to wait for YTs to appear beneath it. However, QCs at most terminals average about 25 lifts/h. The average number of lifts achieved at a terminal per QC working hour is known as the GCR (gross crane rate, QC rate). In this paper, we use GCR as the measure of performance to maximize over the long run.

Maximizing GCR is important for several reasons. For the terminal operator, a higher GCR results in greater business turnover using the same equipment and labor force. For the vessel operator (shipping line), a higher GCR leads to higher vessel utilization as vessels spend less time at port and more time at sea. With annual revenues for the largest container terminal (vessel) operating companies at around USD $3 ($30) billion, there is significant potential for financial gain via GCR improvement. Maximizing GCR also helps the environment. Indeed, if existing terminals and vessels operate at higher efficiencies, the need for additional terminals and vessels is diminished.

This paper is organized as follows. Section 2 discusses the problem in detail. Section 3 summarizes the literature. Section 4 describes the YT control system developed in this research. Section 5 describes the simulation model used for experimentation. Section 6 describes the experiments, presents the results, and discusses their significance. We conclude in Sect. 7.

2 Problem description

2.1 Container terminal operations

In this paper, we investigate how the settings and logic within a real-time yard truck (YT) control system affect the long-run performance of a container terminal as measured in terms of GCR. Container terminal operations exhibit several features that are relevant to YT control. First, there is a lot of stochasticity in the terminal environment. For example, YT (YC) traveling (gantrying) speed varies with operator style, operator error, and real-time traffic conditions in the terminal. Also, QCs and YCs have highly variable container handling times. According to a major terminal operator, the time (in minutes) taken by a YC to handle a single container follows a triangular (1.2, 2.0, 3.4) distribution. The high variance is due to many factors including (1) the trial-and-error method by which YC operators place containers precisely onto stacks; (2) the varying skill levels of YC operators; (3) the varying skill levels of YT operators with whom YCs interact; and (4) real-time wind/weather conditions. Secondly, most terminals never close; the workload is processed 24 hours per day, 365 days per year. Thirdly, there is an uneven distribution of workload over time. Fourthly, QCs load and unload containers according predetermined job sequences that have limited flexibility for changing the order in which lifts are performed. Thus, a poorly-sequenced arrival of YTs beneath a QC lowers terminal productivity. Fifthly, the maximum handling

speed of a YC is roughly 25 lifts/h, much slower than a QC's 40 lifts/h. Finally, YTs and YCs move great distances while QCs are virtually immobile. Hence, at least 5 YTs and 2–3 YCs are typically needed per QC to keep the QCs busy.

2.2 Four experiments on real-time, dual-load yard truck control

Within the above environment, the YT control system is responsible for determining the sequence of journeys and idling operations undertaken by all YTs at all times. YT control involves the following problems. When a YT becomes free, what should it do next? When should a YT transport two 20′ containers simultaneously? If a YT is instructed to haul two 20′ containers, which container should be picked up/dropped off first? In other words, when and how should a YT's *dual-load capability* be used?

Figure 2 illustrates many important YT control issues. In this figure, the YT in the center has just become free, has an empty trailer, and needs to be given new instructions. The arrows beside the YT represent alternatives for what the YT could do next. Assuming the YT is not instructed to remain idle, the YT's next job will correspond to a container in one of the QC job sequences (Fig. 2, top). Each QC job sequence consists of container lifts (jobs) that have not yet been completed and are still pending. Each lift is listed as "Q" followed by a coded string *A-BCD*. "Q" indicates the job belongs to a QC; *A* gives the QC number; *B* indicates the job's position in the sequence; *C* specifies whether it is an unloading (U) or loading (L) job; and *D* gives the length of the container being handled (20′, 40′, or 45′). In this study, QCs handle one container at a time, so $D = 20$ means the QC is lifting a single 20′ container. In the figure, the first four such jobs are listed for each QC. Due to real-time circumstances, it is highly probable that many jobs (shown in bold) have already been assigned to a YT during previous executions of the YT control logic. Thus, the YT in the center of the figure must be assigned to one or more of the other, non-bold jobs.

Note in the figure that the number of YTs queuing beneath each QC is often less than the number of jobs at the QC that have already been assigned to YTs. This is true for three reasons. First, it is very likely that many YTs that have been assigned to QC jobs are still making their way to the quay. Moreover, those YTs assigned to loading jobs may still be traveling to, or waiting for service in, the yard. Finally, some YTs may be assigned to two (20′) QC jobs.

In this study, we conduct four experiments to evaluate the performance of different YT control systems. In Experiment 1, we compare the performance of YT control systems that *pool* YTs at the QC, vessel, and terminal levels. We also compare *due-time-based* and *QC-starvation-based* (inventory-based) YT job assignment systems. The latter systems assign free YTs to jobs at the QCs with the smallest "inventory" (greatest starvation). QC inventory equals the number of pending jobs at the QC that have already been assigned to a YT, multiplied by a *loading preference multiplier* (LPM) in the case of an unloading QC. The LPM indicates the YT control system's preference level for assigning free YTs to loading versus unloading jobs (Fig. 2, bottom). For the QC-starvation-based system, we also investigate the impact of the value of LPM on GCR.

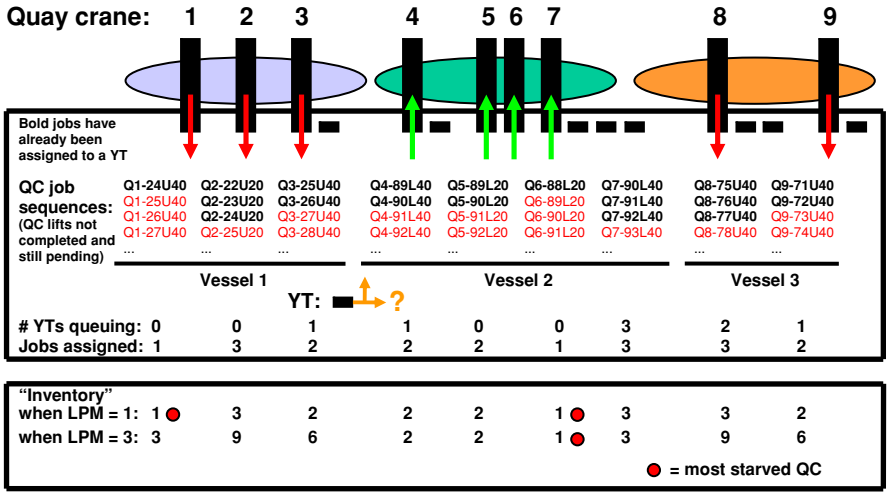

Fig. 2 An important aspect of YT control is the assignment of YTs to QC jobs in real time (*top*). One strategy is to assign free YTs to jobs at the QCs with the smallest "inventory" (i.e. greatest "starvation"). QC inventory equals the number of pending jobs at the QC that have already been assigned to a YT, multiplied by a loading preference multiplier (LPM) in the case of an unloading QC (*bottom*)

In Experiment 2, we explore whether GCR can be improved by allowing free YTs to select from a greater number of jobs than before so that they are assigned to closer (but possibly less urgent) jobs. In Experiment 3, we show the effect of dual loading on GCR. Experiment 4 explores whether GCR can be improved by assigning multiple YTs to multiple jobs simultaneously ($m - n$ assignment) instead of assigning YTs to jobs one at a time ($1 - n$ assignment). We also identify the optimal size of the critical mass for two different group assignment systems—a *batch-size-triggered* system and a *time-triggered* system (Fig. 3). The former system keeps free YTs idle until a batch of trucks of specified size is formed; the latter system keeps free YTs idle until the next periodic dispatching epoch is encountered. In both cases, once the threshold is reached, an $m \times n$ assignment problem is solved in real time by the Hungarian method, and the trucks are instantly dispatched according to the optimal solution.

3 Literature review

Excellent surveys of the container terminal literature have been done by Stahlbock and Voß (2008), Steenken et al. (2004), and Vis and de Koster (2003). A good overview of container terminal operations is provided by Günther and Kim (2006). Murty et al. (2005a,b) and Monaco et al. (2009) discuss the various operational decisions made in container terminals.

Very few articles consider multi-load vehicles for container transport or provide a numerical comparison of alternatives for real-time control of container terminal vehicles. Saanen (2004) considers many issues of real-time vehicle control but provides only limited numerical results. Kim and Bae (2004) show how the performance of

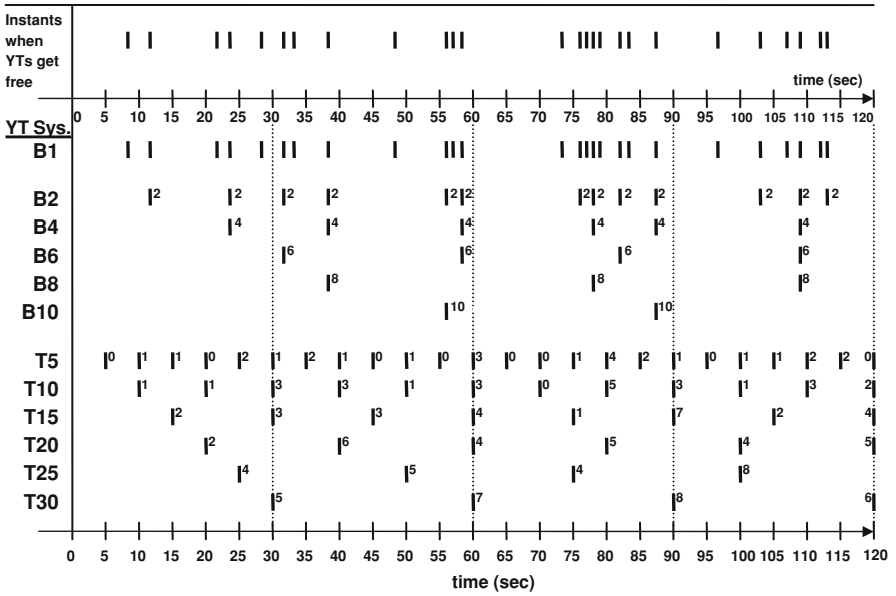

Fig. 3 Tick marks indicate when the YT job assignment routine is triggered for batch-size-triggered (B1–B10), and time-triggered (T5–T30) YT control systems given the instants when YTs become free (*top*). The number beside a tick mark indicates how many free YTs are simultaneously assigned to jobs

a single-load AGV (automated guided vehicle) dispatching system depends on the number of future jobs known for each QC and the variability in QC operating time. Nishimura et al. (2005) propose an IP model for scheduling multi-load YTs but do not consider real-time YT control. They conclude that a "dynamic" YT scheduling policy that allows YTs to serve multiple QCs is superior to a "static" policy that assigns each to YT to a single QC. The current study arrives at the same conclusion.

Grunow et al. (2004, 2006) study the assignment of dual-load AGVs to transportation jobs in real time at an automated container terminal. After establishing the ideas of full and partial AGV availability, the authors develop a dispatching algorithm that is initiated whenever a new transportation order appears within a look-ahead time window. For each partially available AGV, the algorithm generates up to three possible tours for the AGV, each one corresponding to a different ordering of the pick-up/drop-off operations of the new 20′ container with respect to the drop-off of the 20′ container already on the AGV. For each fully available AGV, only one tour is generated. Among all possible tours for all available AGVs, the algorithm chooses the vehicle–tour combination that handles the transportation order with minimum lateness. One noteworthy finding is that the benefits of dual loading diminish as the terminal becomes larger. This trend is also observed in the current study.

Briskorn et al. (2006) have performed what is perhaps until now the most comprehensive analysis of real-time vehicle dispatching systems in container terminals. They focus on the assignment of transportation jobs to single-load AGVs in real time at an automated container terminal. They (A) compare due-time-based and inventory-based AGV dispatching systems; (B) allow each AGV to *dual-cycle*, i.e. to serve more than

one QC; (C) introduce the *phase factor* concept to balance the dispatching system's preference for assigning AGVs to loading versus unloading jobs; (D) consider systems that find optimal solutions to $n \times n$ assignment problems and implement them in real time; and (E) show how the level of flexibility for re-sequencing jobs in the QC job sequences affects overall terminal performance. Performance is measured in relative, not absolute, terms. In the current study, we use different terminology but consider many of the same issues in the context of a real-time, *dual-load* YT dispatching system. Regarding (A), we compare due-time-based and QC-starvation-based dispatching systems. For (B), we compare systems in which YTs are pooled at the QC, vessel, and terminal levels. Regarding (C), we show how the value of the loading preference multiplier (LPM, phase factor) affects GCR. For (D), we investigate how the size of the assignment problem affects long-run GCR. The results in the current manuscript generally agree with those in Briskorn et al. (2006).

Fifty container terminal simulation models were found in the literature. Some of the recent models include: Briskorn et al. (2006), Dekker et al. (2006), Grunow et al. (2004, 2006), Kim et al. (2006), Kozan (1997), Legato et al. (2009), Liu et al. (2002, 2004), Parola and Sciomachen (2005), Petering (2007, 2009), Petering and Murty (2009), Petering et al. (2009), Saanen (2004), and Vis (2006). Only a handful of models—including those by Saanen, Petering, Liu, and Vis—show how different values of an input parameter (e.g. vehicle fleet size, terminal layout) affect *overall* terminal performance as measured by absolute GCR or average vessel turnaround time. In addition, only Saanen's and Petering's models consider microscopic operations at a multiple-berth terminal where yard equipment (YTs, YCs) serves multiple vessels simultaneously. Saanen (2004) describes several sophisticated simulation models in rough terms but presents relatively few numerical results. The papers involving Petering study different problems using the same simulation model described in the current manuscript.

In summary, the current study combines the scopes of Briskorn et al. (2006) and Grunow et al. (2004, 2006) to consider how various real-time, dual-load YT control systems affect container terminal performance. Unlike the previous studies, we measure performance in terms of an absolute global performance metric—the average GCR that each system can maintain in the long run—among other indicators. Furthermore, we consider a fully-integrated container terminal operation; describe the YT control system in detail; and add to the existing quantity of numerical results on YT control by several fold.

4 YT control system

4.1 YT states

The real-time YT control system developed in this research allows each YT to be in one of ten *states* at a given time. Figure 4 shows the ten possible states for YTs and how the status of a YT may change as operations unfold. The letters U, S, R, and L signify (quayside) unloading, (yard) storage, (yard) retrieval, and (quayside) loading respectively. Note that every container passing through a vessel-to-vessel transshipment

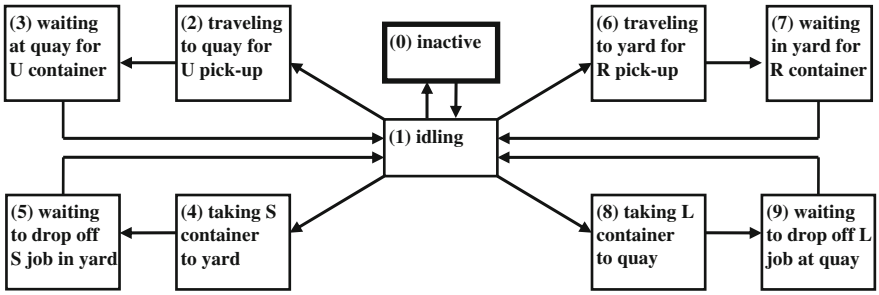

Fig. 4 The YT control system allows a YT to be in one of ten states at a given time

terminal undergoes these operations in the order $U - S - R - L$. Typically, several days elapse between operations S and R. YTs may instantaneously pass through any state 1–9 if conditions are right. State 0 occurs when a YT is inactive and parked for an extended time outside the cargo handling area.

When a YT is deployed within the cargo handling area, it is in states 1–9. An idle YT that is awaiting instructions is in state 1. Consider an idle YT with an empty trailer. Once this YT receives instructions from the control system, it enters into state 2 (6) if the instruction is to receive a container from a QC (YC) (alternatively, the YT may enter into state 0). The YT is in state 2 (6) as it proceeds to the quay (yard). After arriving at the appropriate QC (yard slot), it may enter state 3 (7) if the QC (YC) is not ready to interact with it. Upon receiving the container from the crane, the YT returns to state 1 and the system determines if the YT should receive a second container (20′ only) or not. If so, the YT again passes through states 2 and 3 (6 and 7) before returning to state 1. If the QC (yard slot) for the second container is the same as the first, state 2 (6) is passed through instantly the second time around. After all pick-up operations are completed, the YT passes once through the state sequence 4-5-1 (8-9-1) for each container it is hauling. Jobs can only be assigned to YTs when they are in state 1.

4.2 YT task sequences

Unlike an AGV control system, the control system for a fleet of manually-operated YTs should not redirect YTs that are already en route as this is likely to lower driver morale. Moreover, the driver may become confused if several redirections are made within a short period. These observations lead to an importation limitation within our YT control system: once begun, YT journeys cannot be changed. Moreover, YT job assignments cannot be changed once they are made. Given the stochasticity in the container terminal environment, it also makes sense not to assign a YT to too many future jobs simultaneously. Thus, the YT control system limits the number of QC jobs that can be associated with (assigned to) a YT at any given time to either one 40′/45′ job or one or two 20′ jobs—i.e. enough QC jobs to fill up its trailer. In other words, "chains" of more than one 40′/45′ job or two 20′ jobs are never created for individual vehicles. This restriction allows the control system to make decisions at the

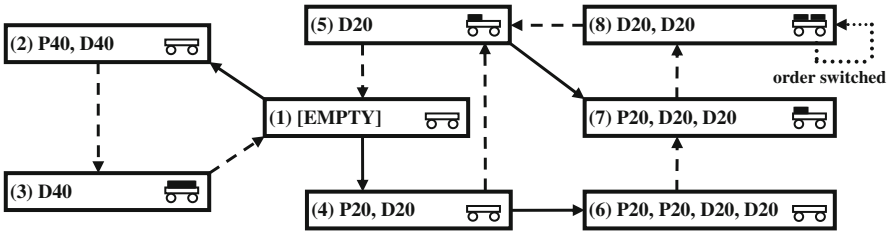

Fig. 5 A YT's task sequence has eight possible arrangements

latest possible instant based upon real-time information coming out of the terminal operating system (TOS).

Each YT performs *tasks* according to a sequence (YT task sequence) that is managed by the YT control system. Each task corresponds to the transfer of a single container to or from the YT. There are four main YT tasks: 40′/45′ pick-ups (P40), 40′/45′ drop-offs (D40), 20′ pick-ups (P20), and 20′ drop-offs (D20). As the preceding paragraph indicates, a maximum of four future tasks can be assigned to a YT at any given time. Tasks are added to YT task sequences in pairs and deleted individually. When two tasks are added to a sequence, they are always the pick-up and drop-off of the same container.

Figure 5 shows the eight possible arrangements of a YT's task sequence and how the sequence may change as operations unfold. The physical status of the YT trailer is shown to the right of each arrangement. Arrangement 1 is empty. In arrangement 2 (4), the YT has been instructed to haul a 40′/45′ (20′) container but has not yet picked up the container. Arrangement 3 (5) arises after the YT has picked up the 40′/45′ 20′ container but before it has dropped it off. In arrangement 6, the YT has been instructed to haul two 20′ containers but has not yet picked up either container. Arrangement 7 (8) arises after the YT has picked up one (both) of the containers. The dotted arrow beside arrangement 8 indicates that the YT control system has a subroutine that switches the order of two 20′ drop-offs whenever such a switch results in less overall travel distance for the two drop-offs. In the figure, solid (dashed) arrows indicate additions to (deletions from) a YT's task sequence. Additions are driven by decision making processes in the YT control system and are only executed when the YT is in state 1. Deletions correspond to the completion of physical processes.

We now consider the relationship between Figs. 2, 4, and 5. Consider a YT that passes through states in the order 1-6-7-1-6-7-1-8-9-1-8-9-1-2-3-1-4-5-1 (Fig. 4). This might happen if, for example, the YT in the center of Fig. 2 is first assigned to jobs "Q6-89L20" and "Q6-90L20" and then is assigned to the job "Q1-34U40" upon completing these first two jobs. In this case, its task sequence would pass through the following arrangements: 1-4-6-7-8-5-1-2-3-1.

4.3 Two algorithms for controlling the YT fleet

The YT control system's main responsibility is to (1) decide when and how to add tasks to a YT's task sequence, (2) decide when and how to re-sequence tasks already in a

YT's task sequence, and (3) avoid operational deadlocks. Two algorithms—the workcenter YT assignment algorithm and the individual YT dispatching algorithm—work together to control the YT fleet.

The workcenter YT assignment algorithm performs two main functions. First, it awakens inactive YTs—moving them from state 0 to state 1—and assigns them to new QC jobs whenever a new vessel begins to moor alongside the terminal. Second, it uses the Hungarian method to solve $m \times n$ assignment problems when the batch-size-triggered or time-triggered YT control systems are used (Fig. 3). This algorithm is triggered either when a new vessel begins to dock at the terminal or when the batch size or time threshold for $m - n$ YT job assignment has been reached. The algorithm can cause the task sequences of one or more YTs to transition from arrangement 1 to arrangements 2, 4, or 6.

The individual YT dispatching algorithm adds tasks to, and re-sequences tasks already in, a YT's task sequence. It considers a single YT and is activated whenever a YT finishes a task, i.e. whenever a task is deleted from a YT's task sequence. This algorithm can make the following transitions in a YT's task sequence: $1 \rightarrow 2$, $1 \rightarrow 4$, $1 \rightarrow 4 \rightarrow 6$, $5 \rightarrow 7$, and $8 \rightarrow 8$. It may also instruct a YT to remain idle in state 1 if the batch size or time threshold for group dispatching has not been reached. The $8 \rightarrow 8$ transition occurs when the order of two drop-off tasks is switched. The algorithm's job assignment routine is essentially a $1 \times n$ selection routine; one YT and many jobs are considered, and a maximum of two (20′) jobs are assigned to the YT. Both the workcenter and individual YT algorithms can deactivate idle YTs—moving them from state 1 to state 0—when there is no more work to be done.

We mentioned above that YT job assignments cannot be changed once they are made. However, there is one exception to this rule that resolves situations in which two YTs with empty trailers are heading towards the same location (QC or yard slot) but arrive in a different order than was initially expected. In such instances, the YT control system allows these two YTs to instantaneously swap job assignments so that the YT that arrives first can be served first. In other words, YTs with empty trailers heading towards the same location (QC or yard slot) are fully substitutable. This feature of the control system improves terminal productivity.

4.4 Job assignment and deadlocking

Several restrictions are placed on the YT control system to guarantee that operational deadlocks are avoided on 100% of occasions. First, the first job assigned to an idle and empty YT is always the earliest unassigned job in a QC job sequence. Without such a stipulation, it is theoretically possible for the YT control system to keep assigning free YTs to later jobs until all YTs are "stuck" and there are no more YTs to assign to the earliest job. If the first job assigned to a YT is a 20′ container, the system determines if the next job in the same QC's job sequence is also a 20′ container. If so, this second container is eligible to be assigned to the YT. If not, no additional job is assigned to the YT. In the context of Fig. 2, the above limitations mean that there are exactly nine options—one for each QC—for the first job that is assigned to a YT. In addition, there is only one option for the second job if the first job is a 20′ container.

Even with the above stipulations, the YT fleet may still be susceptible to deadlocking owing to the YTs' dual-load capability and the relatively inflexible QC job sequences. To prevent such deadlocks, the YT control system does not allow YTs to consecutively (A) pick up two 20′ containers from different locations (QCs or yard stacks) or (B) drop off two 20′ containers at different QCs. However, the system does allow YTs to consecutively drop off two 20′ containers at different yard stacks. Although they seem severe, these restrictions still allow for a substantial number of dual loading opportunities in the yard and at the quay. Indeed, regarding the yard, many containers that are stored in the same yard stack are also loaded consecutively by the same QC. Regarding the quay, the 20′-20′ twin-lift capability of QCs at many terminals (but not those considered in this study) means that virtually all YT dual loading at the quay involves two containers coming from the same QC. Thus, the restrictions on dual loading are reasonable and hardly draconian. The above limitations mean that the two pick-ups in arrangement 6 in Fig. 5 must be made at the same QC or yard stack.

4.5 YT control parameters

The YT control system has 6 parameters that control the YT fleet. These parameters are listed and briefly described in Table 1. In the experiments, we show how these parameter values affect the long-run average GCR at a multiple-berth container terminal. Horizontal lines indicate which parameter values are modified in each of the four experiments. We now discuss each parameter in detail.

The parameter "Pool" refers to the level of YT pooling that is done. Many container terminals allocate labor and equipment based on the number of QCs expected to be operating during each 8-hour work shift. For each operational QC, one *gang* is deployed. A gang consists of 1 QC and a certain number, say 9, of supporting YTs.

Table 1 Parameters in the YT control system

Parameter	Possible values	Explanation
Pool	Q, V, T	Indicates whether YTs are pooled at the QC (Q), vessel (V), or terminal (T) level
Type	DT, QS	Type of job assignment system used: due-time-based (DT) or QC-starvation-based (QS)
LPM	1, 1.5, 2, 2.5, 3, 4, 5, 6, 8, 10	Loading preference multiplier (QC-starvation-based system only)
StarvProx	0, 1, 2, 3, 4	Proximity to most starved QC that is allowed when assigning jobs to YTs (QC-starvation-based system only)
DualLoad	Yes, No	Indicates whether system allows YTs to haul two 20′ containers simultaneously
Trig	B-X, T-X	Indicates if YT job assignment algorithm is triggered when a batch of X idle and empty YTs accumulates (B-X), or when the end of a time interval of length X seconds is reached (T-X)

At most terminals, each YT exclusively serves the QC in its gang. We refer to such an operational mode as "pooling at the QC level" (Pool = Q). However, other operational modes might yield a higher GCR. For example, YT resources could be "pooled at the vessel level," whereby each YT may serve any QC that is working on a particular vessel (Pool = V). Alternatively, YTs could be "pooled at the terminal level" so that each YT may serve any QC in the terminal (Pool = T).

The parameter "Type" refers to the type of job assignment performed. In a due-time-based system (Type = DT), a due time is attached to each QC job. The due time is the TOS's current best estimate for when the QC will begin lifting that container, assuming the QC is not delayed and does not have to wait for YTs in the interim. The due times of all pending jobs at a QC are updated every time the QC begins a new lift. If the new lift begins earlier (later) than expected, the due times of all pending jobs are adjusted earlier (later). Note that a job always has a later due time than all of its predecessors. In this paper, the due-time-based YT control system assigns a free YT to the QC job with the earliest due time that has not yet been assigned to a YT.

A QC-starvation-based job assignment system (Type = QS) makes use of information regarding the number of pending QC jobs that have already been assigned to YTs in real time. This system focuses on filling gaps in the queues beneath the QCs that are closest to being starved based on real-time information. Free YTs are sent to serve the "most idle" QCs first, i.e. those QCs with the lowest inventories. The idea is identical to the inventory-based AGV dispatching approach that Briskorn et al. (2006) found to be effective.

In QC-starvation-based job assignment, the proper measurement of QC inventory (starvation) is very important. For example, the number of YTs queuing beneath each QC (Fig. 2, top) is not a good measure of QC inventory because it does not account for YTs that are traveling towards QCs or for YTs that have been assigned to two 20′ containers (two lifts worth of cargo if QCs lift one container at a time). A better measure of inventory is the number of lifts worth of cargo (i.e. jobs) at each QC that have already been assigned to YTs (Fig. 2, middle). However, this measure is still insufficient because it does not distinguish between unloading and loading jobs. Indeed, YTs assigned to unloading jobs pass through only two states (2 and 3) prior to interacting with a QC, but YTs assigned to loading jobs pass through four states (6, 7, 8, and 9) prior to QC interaction (Fig. 4). A well-balanced container terminal operation should have an even distribution of YTs in the state spaces (2 or 3), (4 or 5), (6 or 7), (8 or 9) at any given time. Thus, there should be roughly twice as many YTs in states 6–9 versus states 2–3. When computing inventory, the YT control system should therefore exhibit a "loading preference" so the loading QCs can have more of their pending jobs assigned to YTs than the unloading QCs in real time. Such a loading preference is quantified in the YT control system through the parameter LPM, which stands for "loading preference multiplier." QC inventory is computed as the number of pending jobs at the QC that have already been assigned to a YT, multiplied by LPM in the case of an unloading QC (Fig. 2, bottom).

The parameter LPM only applies to the QC-starvation-based job assignment system (Type = QS). This parameter is called a "phase factor" in Briskorn et al. (2006). It indicates the control system's preference level for assigning free YTs to loading versus unloading jobs. The preceding discussion indicates that LPM should be greater than 1.

Now consider Fig. 2. When LPM = 1, the control system is indifferent to whether a QC is loading or unloading when it computes QC inventory. Thus, when Pool = T and LPM = 1, QCs 1 and 6 tie for being the most starved QCs and the control system assigns the next free YT to either job Q1-25U40 or job Q6-89L20. The final decision depends on which job's pick-up location is closer to the YT. When LPM = 3, however, QC 6 is uniquely the most starved QC. In this case, the YT control system assigns the next free YT to job Q6-89L20.

The parameter "StarvProx" is only relevant when Type = QS and Trig = B1. "StarvProx" is short for "starvation proximity." StarvProx has a default value of 0. StarvProx = 0 means that a free YT is always assigned to the earliest unassigned job in the job sequence belonging to the most starved QC (or one of the most starved QCs in case of a tie). StarvProx = X means that a free YT can be assigned to the earliest unassigned job belonging to any QC whose inventory is no more than X greater than that of the most starved QC. The final decision depends on which job's pick-up location is closest to the YT. Consider Fig. 2. When Pool = T, LPM = 3, and StarvProx = 0, a free YT is assigned to job Q6-89L20. However, if StarvProx = 1, QCs 4, 5, and 6 all qualify as "starved" and a free YT is assigned to the job with the closest pick-up location among jobs Q4-91L40, Q5-91L20, and Q6-89L20.

The parameter "DualLoad" indicates whether the YT control system allows YTs to haul two 20′ containers simultaneously. DualLoad = Yes (No) means that dual loading is allowed (forbidden). Consider Fig. 2. If Pool = T, Type = QS, LPM = 3, StarvProx = 0, and DualLoad = No, the YT control system assigns a free YT to job Q6-89L20. However, if DualLoad = Yes, the YT control system assigns a free YT to jobs Q6-89L20 and Q6-90L20 if these jobs have the same pick-up stack in the yard; otherwise, the YT is only assigned to job Q6-89L20.

The parameter "Trig" governs the triggering of the YT job assignment routine. This parameter's value consists of a letter—either B or T—followed by a number X. "B" stands for batch-size triggering and "T" stands for time triggering. If the letter is B, YTs are assigned to jobs in batches of constant size X (Figure 3). For example, if Trig = B6, YTs that become free (with empty trailers) remain idle until a total of 6 YTs are free. At the moment a sixth YT becomes free, all 6 YTs are simultaneously assigned to jobs. If the letter is T, the YT job assignment routine is triggered every X seconds. For example, if Trig = T20, YTs that become free remain idle until the end of the current 20-s time interval. Once this interval expires, all YTs that became free in the intervening interval are simultaneously assigned to jobs. A new interval then begins. The Hungarian method optimally solves the assignment problems for the batch-size-triggered and time-triggered YT control systems. "Trig" has a default value of B1. Pool = T and Type = QS whenever Trig $\neq$ B1.

The idea of a group assignment system is to delay giving instructions to one or more YTs until additional real-time information—identifying the locations of additional YTs that have just become free—becomes available that allows for a more "intelligent" assignment of YTs to jobs. The downside of this system is that YTs waste time waiting for instructions. The benefit is that YTs make shorter journeys to their next container pick-up locations once they receive the instructions. While this "delay and group" vehicle dispatching strategy clearly holds promise for a large taxicab company—that might deploy 1000 taxis over a large region such as Los Angeles

County—no academic study has considered whether it might be suitable for a non-automated container terminal.

We now consider two examples to illustrate how the group assignment system works. First, assume that Pool = T, Type = QS, LPM = 3, DualLoad = No, Trig = B6, and that 6 YTs are to be assigned to jobs in the terminal depicted in Fig. 2. In this case, the workcenter YT assignment algorithm (Sect. 4.3) formulates and solves an assignment problem of size 6 trucks by n jobs. These n jobs are the first jobs in a special list of unassigned QC jobs ordered from most to least urgent based on QC starvation: (Q6-89L20, Q4-91L40, Q5-91L20, Q6-90L20, Q1-25U40, Q4-92L40, Q5-92L20, Q6-91L20, Q7-93L40, Q4-93L40, …). Importantly, the QC inventories are re-computed each time an item is added to the list. The inventories associated with the above jobs are (1, 2, 2, 2, 3, 3, 3, 3, 3, 4,...). Note that job Q1-26U40 cannot be listed tenth in the sequence because LPM = 3, so the inventory of QC 1 goes from $3 \rightarrow 6$, not $3 \rightarrow 4$, after job Q1-25U40 is added to the list. n is the smallest number ≥ 6 such that the inventory for the nth job is strictly less than that for the $(n+1)^{st}$ job in the list. Here, $n = 9$, so a 6×9 assignment problem is formed. For each truck i and job j, the cost c_{ij} of assigning i to j is the expected travel time between truck i's current location and job j's pick-up location. The 6×9 matrix is augmented with three rows of zeros, representing three fictitious trucks, to obtain a square 9×9 matrix. Next, a large penalty cost P ($P >> \max(c_{ij})$) is added to certain matrix cells to ensure that the most urgent jobs are not assigned to fictitious trucks. The most urgent jobs are the first u jobs in the list where u is the largest number $\leqslant 6$ such that the inventory for the uth job is strictly less than that for the $(u+1)^{st}$ job in the list. Here, $u = 4$. Thus, P is added to all cells (i, j) such that $i \geq 7$ and $j \leq 4$ to guarantee that jobs 1–4 are not assigned to fictitious trucks. The optimal solution is therefore the least cost assignment of 9 trucks (6 real, 3 fictitious) to 9 jobs such that the first 4 jobs are assigned to real trucks. The three jobs not assigned to real trucks remain unassigned until the next execution of the routine.

Now consider the above situation except that DualLoad = Yes and the first two unassigned (20′) jobs at QC 6 can be hauled by the same YT. In this case, the YT control system forms the same job and inventory lists as above except that now the second 20′ container is deleted from both lists, leaving them in the form (Q6-89L20, Q4-91L40, Q5-91L20, Q1-25U40, Q4-92L40, Q5-92L20, Q6-91L20, Q7-93L40, Q4-93L40, …) and (1, 2, 2, 3, 3, 3, 3, 3, 4, …). A 6×8 assignment problem is then generated, and the penalty factor P is again employed to guarantee that the first three jobs are assigned to real trucks. One of these "jobs" consists of two 20′ containers.

5 Discrete event simulation model of a seaport container terminal

5.1 Introduction

The simulation model used for experimentation has been developed based on the author's discussions with managers at several container terminals. The model simulates the activities of individual containers, vessels, QCs, YCs, YTs, and groundslots over an arbitrarily long, user-defined time period.

Table 2 Discrete events in the simulation model

Primary events	Secondary events
Vessel arrives	Vessel starts berthing
Vessel finishes berthing	Vessel starts un-berthing
Vessel finishes un-berthing	QC job sequences generated
All QC job sequences scanned	QC job sequences erased
QC finishes handling	QC starts handling
Multiple YC deployment algorithm called	General storage and retrieval algorithm called
YC finishes cross gantry	Real-time container storage algorithm called
YC finishes linear gantry	YC cross gantry scoring re-computed
YC finishes handling	Individual YC dispatching algorithm called
Workcenter YT assignment algorithm called	Individual on-the-fly YC dispatching algorithm called
YT finishes journey	YC starts cross gantry
Check terminal status consistency	YC starts linear gantry
Start data collection	YC starts handling
	Individual YT dispatching algorithm called
	YT starts journey

Table 2 shows the events in the simulation model. There are two kinds of events: primary and secondary. A primary event may (1) cause an immediate change in the state of the terminal, (2) trigger other (primary and/or secondary) events that occur together with the primary event, and/or (3) cause other primary events to be placed in the future event list. Note that two events in Table 2 are calls to the workcenter YT assignment and individual YT dispatching algorithms. Several other events are calls to algorithms that make container storage, YC deployment, and YC dispatching decisions in real time.

5.2 Main features and limitations of simulation model

We now discuss some features of the simulation model. More extensive descriptions of the model are given in Petering (2007, 2009), Petering and Murty (2009), and Petering et al. (2009).

The model accommodates two container sizes (20′ and 40′). The traveling and handling times for all machines (QCs, YTs, YCs) follow user-defined probability distributions. QCs handle one container at a time. Each QC handles containers according to a sequence with limited flexibility for changing the order in which lifts are performed. Thus, a poorly sequenced arrival of YTs beneath a loading QC can significantly harm GCR as measured by the simulation model.

YTs may haul two 20′ containers simultaneously, i.e. YTs have dual-load capability. Although the total number of YTs is held constant throughout each simulation run, the number of active YTs (in states 1–9) is a fixed multiple of the number of vessels present at all times. Inactive YTs may only be reactivated when a new vessel starts to dock at the terminal.

The model tracks the groundslots where individual containers are stored in the yard. Containers are assigned storage locations immediately after being placed onto YTs during unloading. Each container that is stored in the yard is retrieved from the same location before being loaded onto a vessel. Vessels may not leave the terminal unless all of their containers have been unloaded and loaded.

The model's main limitations are as follows. Firstly, only 20′ and 40′ standard dry containers are considered; 45-foot long, refrigerated, dangerous goods (DG), and other containers that make up about 15% of overall cargo volume are ignored. Secondly, YT congestion on roads and at handling points is not explicitly modeled. Finally, the model considers a pure transshipment terminal. That is, all containers enter and leave the terminal by vessel. The pure transshipment assumption means that there is no gate or rail yard, and there are no external trucks or trains.

6 Experiments, results, and discussion

6.1 General setup for experiments

The simulation model was written and compiled using the Professional Edition of Microsoft Visual C++ 6.0. Experiments were run in Windows XP using a 2.4 GHz computer with 2 GB of RAM. Each simulation terminates after three weeks worth of vessels are fully processed at the terminal. This instant may fall within week 4 or later if significant backlogging has occurred. Data collection starts at the beginning of week 2.

The experiments consider two terminals—a small and large terminal—and two yard fleet sizes for each terminal—"less equipment" and "more equipment." Thus, four terminal *scenarios* are considered. For the sake of brevity, we refer to these as the "SL," "SM," "LL," and "LM" scenarios, where the first letter (S, L) stands for terminal size and the second letter (L, M) stands for the fleet size. Table 3 gives the main specifications of these scenarios. The small terminal has 5 yard zones, 4 blocks per zone, 42 slots per block, and 6 rows per block (Fig. 6). The large terminal has 10 yard zones, 9 blocks per zone, 42 slots per block, and 6 rows per block. As Fig. 6 indicates, YT travel is bidirectional on all vertical avenues and on the two horizontal avenues at the top and bottom of each terminal. It is unidirectional along the horizontal avenues in the middle of each terminal. Each horizontal avenue in the middle of the terminal consists of two lanes running in the same direction—a handling lane for YT–YC interaction and a bypass lane for through traffic. The bypass lane allows multiple YTs to simultaneously reach multiple YCs in the same block without creating a traffic jam. Within each scenario, all vessels are the same size and have equal expected cargo throughput each week, but the throughput associated with a particular vessel in a given week is a stochastic quantity. Four QCs are assigned to each berth in each scenario.

In all experiments, GCR is measured as follows:

$$\text{GCR} = (\text{total number of QC lifts})/(\text{total number of QC hours beside a busy berth}).$$

Table 3 Container terminal scenarios considered in the experiments

	Small terminal less equipment (SL)	Small terminal more equipment (SM)	Large terminal less equipment (LL)	Large terminal more equipment (LM)
Expected vessel calls per week	10	10	90	90
Expected QC lifts per vessel	3,600	3,600	1,920	1,920
Expected QC lifts per week	36,000	36,000	172,800	172,800
Berths	2	2	9	9
QCs	8	8	36	36
Groundslots in yard	5,040	5,040	22,680	22,680
Yard zones	5	5	10	10
Max. cont. stacking height in yard	6	6	5	5
Ratio of 20′ to 40′ conts.	2:1	2:1	3:2	3:2
YCs	20	25	90	110
YCs per zone	4	5	9	11
YTs	40	72	180	324
Active YTs per vessel present	20	36	20	36

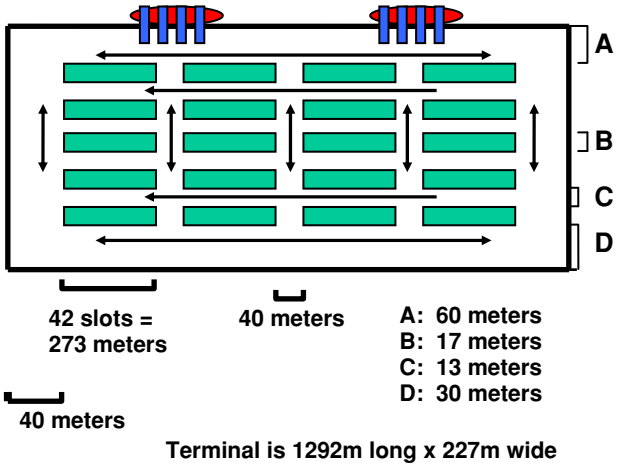

Fig. 6 Layout of the small container terminal considered in the experiments

A busy berth is a berth with a ship alongside such that at least one QC is still moving containers between ship and shore. GCR is inversely proportional to average vessel turnaround time.

6.2 Vessel, QC, YT, YC, and container storage settings

The values of the model's input parameters (except for the real-time algorithm settings) were chosen based upon information received from a major container terminal operator. In all experiments, vessels arrive between 2 h prior and 12 h after their scheduled arrival times. The time taken by a QC to handle a single container is triangularly distributed with parameters (1.0, 1.5, 2.0) minutes. This distribution has mean 1.5, so the maximum possible GCR, if QCs are never starved of YTs, is 40 lifts per hour.

YTs travel 40 (25) km/h on average when empty (carrying one or more containers). They spend an average of 10 seconds making a turn. Their actual travel time for a given journey ranges from 30% below to 30% above the expected time. The number of active YTs is a fixed multiple of the number of vessels present (Table 3, bottom row). Inactive YTs can only be reactivated when a new vessel starts to dock at the terminal.

YCs gantry at an average rate of one slot every four seconds. Their actual gantry time for a given journey ranges from 30% below to 30% above the expected time. The time taken by a YC to handle a single container is triangularly distributed with parameters (1.2, 2.0, 3.4) minutes. YCs are not allowed to make cross-gantry moves like that indicated by the arrow to the right of YC 17 in Fig. 1. YCs in the same block remain separated by at least 170 feet at all times. YCs are allowed to move between blocks in the same zone. However, such movement is restricted by a "restrictive" YC deployment system which sets a minimum and maximum number of YCs that must be present in each block at all times (Petering and Murty 2009). In the less equipment scenarios, exactly 1 YC must be present in each storage block at all times. In the more equipment scenarios, a minimum of 1 and maximum of 2 YCs must be present in each storage block at all times.

The YC control system works as follows. Whenever a YC becomes free, the system identifies the set of jobs (container moves) that have been assigned to YTs that are already waiting near the YC or are soon expected to appear near the YC. Among these jobs, the system assigns to the YC the *retrieval* (*R*) job that is most urgent from the point of view of the QC loading sequences. If there are no retrieval jobs, the system assigns to the YC the *storage* (*S*) job nearest to the YC. See Petering et al. (2009) for more details.

The container storage location assignment system is triggered every time a QC unloads a container and places it onto a YT. If there is a yard stack, with available capacity, that already stores containers in the same group to which the container belongs, the container is assigned to one such stack. Otherwise, the container is assigned to an empty stack in the block with the lowest combined (1) number of YTs heading to or waiting at the block and (2) number of container retrievals whose timing is expected to coincide with (clash with) the retrieval of the current container. The final decision is made by weighting these two measures.

We now discuss the four experiments. For the sake of continuity, we present the setup, results, and discussion for each experiment before moving on to the next experiment. In each experiment, we use the Welch confidence interval method—with confidence levels of 90, 95, 98, and 99%—to statistically analyze the results. If the X% confidence interval (CI) for the GCR *difference* between two systems does not include

the value 0, we say that "the GCR difference is statistically significant at the X% confidence level." If the 90% CI for the GCR difference includes 0, we say that "the GCR difference is not statistically significant."

6.3 Experiment 1: pooling, two dispatching paradigms, loading preference multipliers

In Experiment 1, we explore how the parameters "Pool," "Type," and "LPM" affect GCR when the other parameters are held constant at the following values: StarvProx = 0, DualLoad = Yes, Trig = B1 (Table 1). Table 4 displays the setups considered and experimental results. Note that when Pool = Q, free YTs have only one possible job assignment, so the values of "Type" and "LPM" are irrelevant. Also, when Type = DT, "LPM" is irrelevant. Thus, the experiments consider every possible combination of values for these three parameters (Table 1) in each of the four terminal scenarios (Table 3). For the small (large) terminal scenarios, each data point is the average GCR from ten (six) independent simulation replications of 3 weeks each. Thus, Table 4 displays the results from a total of $23*2*10 + 23*2*6 = 736$ runs. Each run for scenario (SL, SM, LL, LM) required less than (5, 4, 29, 10) minutes of CPU time. The highest GCR value in each scenario is displayed in bold. The average GCR across all four scenarios is displayed in the final column. The highest such average GCR (34.32) is highlighted and corresponds to Pool = T, Type = QS, and LPM = 5. This, the best performing system overall, is the default system in all future experiments and is indicated by two asterisks (**). The default system was statistically compared to each system indicated by a single asterisk (*). For scenarios SL, LL, and LM, the GCR difference between the default system (**) and each of the other systems (*) is statistically significant at the 99% confidence level. For scenario SM, the difference, going down the table from top to bottom, is not significant; not significant; significant at the 95% level; and significant at the 99% level.

The results in Table 4 strongly indicate that a QC-starvation-based YT control system (Type = QS) is superior to a due-time-based system (Type = DT). In fact, even the worst system with Type = QS (GCR = 33.34) performs better than the best system with Type = DT (GCR = 32.91) (Table 4, final column). More importantly, the best system with Type = QS (GCR = 34.32) performs 4.3% better than the best system with Type = DT. These results agree with Briskorn et al. (2006) who find that inventory-based YT control systems outperform due-time-based systems by about 5%. Overall, the high stochasticity of a container terminal operation makes future due time predictions highly unreliable. Dispatching based on such unreliable information is therefore inferior to dispatching based on hard facts: the real-time QC inventories.

The results also demonstrate that productivity can be increased when YTs are pooled at higher levels. Indeed, ignoring the DT systems, we see that GCR generally increases when we move from Pool = Q to Pool = V to Pool = T (Table 4, final column). For example, the best system with Pool = V (GCR = 33.66) performs 1.3% better than the Pool = Q system (GCR = 33.23). Also, the best system with Pool = T (GCR = 34.32) performs 2% better than the best system with Pool = V. The aforementioned statistical tests show these differences are statistically significant. Thus, it is best to pool YT

Table 4 GCR results from experiment 1 (QC lifts/h)

YT control system	Small term less equipment (SL)	Small term more equipment (SM)	Large term less equipment (LL)	Large term more equipment (LM)	Average
Pool = Q*	31.10	37.55	27.66	36.60	33.23
Pool = V, QS, LPM = 1	31.89	37.26	28.50	36.80	33.61
Pool = V, QS, LPM = 1.5	31.86	37.35	28.47	36.78	33.61
Pool = V, QS, LPM = 2	31.79	37.16	28.42	36.76	33.53
Pool = V, QS, LPM = 2.5*	31.81	37.57	28.50	36.78	33.66
Pool = V, QS, LPM = 3	31.53	37.37	28.43	36.78	33.53
Pool = V, QS, LPM = 4	31.99	37.38	28.49	36.77	33.66
Pool = V, QS, LPM = 5	31.86	37.43	28.42	36.77	33.62
Pool = V, QS, LPM = 6	31.92	37.29	28.44	36.75	33.60
Pool = V, QS, LPM = 8	31.84	37.26	28.35	36.78	33.56
Pool = V, QS, LPM = 10	31.99	37.51	28.28	36.68	33.62
Pool = V, DT*	30.59	37.10	27.26	36.68	32.91
Pool = T, QS, LPM = 1	31.22	37.27	28.31	36.58	33.34
Pool = T, QS, LPM = 1.5	31.73	37.47	29.08	37.04	33.83
Pool = T, QS, LPM = 2	31.77	37.35	29.38	37.20	33.93
Pool = T, QS, LPM = 2.5	31.94	37.35	29.58	37.28	34.04
Pool = T, QS, LPM = 3	32.03	37.48	29.82	37.38	34.18
Pool = T, QS, LPM = 4	32.16	37.53	**29.89**	**37.45**	34.26
Pool = T, QS, LPM = 5**	**32.41**	37.65	29.86	37.36	**34.32**
Pool = T, QS, LPM = 6	32.32	37.83	29.65	37.31	34.28
Pool = T, QS, LPM = 8	31.99	**37.91**	29.21	37.03	34.04
Pool = T, QS, LPM = 10	32.14	37.57	29.12	36.94	33.94
Pool = T, DT*	30.33	36.96	26.89	36.61	32.70

resources at the terminal level instead of dedicating them to particular vessels or QCs. These observations agree with the findings of Murty et al. (2005a,b) and Nishimura et al. (2005). However, terminal-wide YT pooling is only possible at terminals with advanced information and/or operating systems. Furthermore, Table 4 shows that terminal-wide pooling is only effective when there is a loading preference, i.e. when LPM is at least 1.5. Otherwise, terminal-wide pooling is worse than pooling at the vessel level.

Figure 7 shows the dependence of GCR on the value of LPM. This figure is a graphical display of the GCR values in the final column of Table 4 when Pool = T and Type = QS. Figure 7 shows that GCR is a concave function of LPM and that GCR is greatest when LPM = 5. This is a much higher LPM than might have been expected. Briskorn et al. (2006) used LPM = 1.6 in their study. However, upon further reflection, LPM = 5 seems reasonable. First, as argued before, LPM = 2 is reasonable because YTs assigned to loading jobs pass through twice as many states (6, 7, 8, and 9) prior

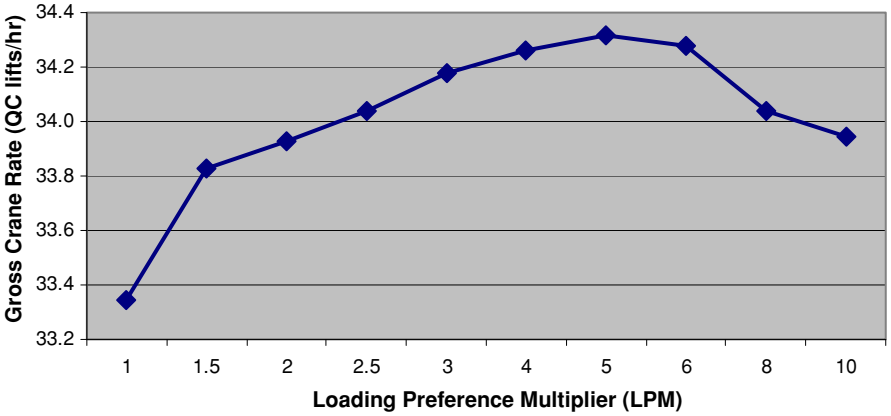

Fig. 7 Dependence of GCR on the loading preference multiplier when Pool = T (average results from all four terminal scenarios)

to interacting with a QC compared to YTs assigned to unloading jobs (states 2 and 3). Secondly, a loading QC with inventory 0 is more starved than an unloading QC with inventory 0 because the former QC cannot begin any part of its operations without a YT, whereas the unloading QC can commence discharging the container from a vessel even without the presence of a YT. Thus, the optimal value of LPM should be greater than 2. Finally, the optimal value of LPM is probably highly sensitive to the exact manner and timing by which the TOS deletes completed jobs from the list of pending QC jobs. This helps to explain the disparity between our results and those in Briskorn et al. (2006).

Table 5 provides more detailed results for selected YT control systems in scenario LL. Each row in the table corresponds to a single GCR value in Table 4. Only two systems with Pool = V are included in this table. These are (1) the best performing system (with LPM = 2.5) when Pool = V and Type = QS and (2) the system with Pool = V and Type = DT. Twenty performance measures are included in the table. A key to these measures is provided beneath the table. The highest GCR is shown in bold.

Table 5 shows that "Pool" and "LPM" have a major impact on several performance indicators. Regarding pooling, the "YT State1" column shows that YT idling decreases when YTs are pooled at higher levels. Indeed, YTs spend roughly 6, 3, and 0% of their time idling when Pool = Q, V, and T respectively. Furthermore, the "YT State2" and "YT State6" columns reveal that YT travel time (distance) to container pick-up locations also decreases when YTs are pooled at higher levels. Thus, pooling at a higher level yields two benefits. First, it gives YTs that finish working on a QC/vessel the opportunity to help out at other QCs/vessels instead of remaining idle. Second, it increases the frequency of YT "dual cycling" whereby YTs consecutively drop-off and then pick-up a container in the same area (yard or quay); this reduces empty YT travel time. Several trends show that the parameter LPM is working as intended. First, note that the values in the "QC WaitU" ("QC WaitL") column increase (decrease) as LPM increases. Thus, unloading (loading) QCs wait more (less) for YTs as the loading preference increases. Also, the values in the "YT State3" ("YT State9") column decrease

Table 5 Detailed results for selected YT control systems in experiment 1 (large terminal less equipment)

YT System	GCR	Berth Occ	BOA Rate	%vol	%area	QCLifts	QC WaitU	QC WaitL	QC Wait	YC Prod	YT Prod	YT State1	YT State2	YT State3	YT State6	YT State9	Avg YT-Q	Avg L YT-Q	Avg U YT-Q	Avg YT-Y
Q	27.66	0.99	0.07	57	85	361,332	0.102	0.157	0.259	10.51	5.33	0.059	0.079	0.082	0.094	0.080	69.0	40.0	29.0	98.1
V,LPM2.5	28.50	0.98	0.08	57	85	362,306	0.087	0.157	0.244	10.77	5.48	0.032	0.078	0.064	0.096	0.092	68.4	42.8	25.6	102.5
V,DT	27.26	0.99	0.07	56	83	363,830	0.080	0.208	0.288	10.39	5.25	0.032	0.080	0.066	0.094	0.129	75.1	48.7	26.4	97.1
T,LPM1	28.31	0.98	0.07	56	84	363,102	0.001	0.240	0.241	10.70	5.45	0.000	0.046	0.177	0.072	0.059	76.7	36.7	40.0	100.1
T,LPM1.5	29.08	0.98	0.11	56	84	363,193	0.002	0.220	0.223	10.95	5.59	0.000	0.055	0.130	0.077	0.066	71.8	38.6	33.3	104.4
T,LPM2	29.38	0.98	0.14	56	84	363,301	0.006	0.210	0.216	11.04	5.65	0.000	0.052	0.110	0.076	0.074	69.5	40.2	29.3	106.5
T,LPM2.5	29.58	0.97	0.13	56	84	361,862	0.010	0.201	0.211	11.08	5.68	0.000	0.057	0.091	0.079	0.078	67.7	41.1	26.6	107.8
T,LPM3	29.82	0.97	0.19	56	84	361,540	0.011	0.194	0.204	11.12	5.73	0.000	0.055	0.086	0.077	0.080	66.8	41.5	25.3	107.9
T,LPM4	**29.89**	0.97	0.17	56	84	362,416	0.019	0.183	0.202	11.15	5.74	0.000	0.051	0.082	0.075	0.083	65.9	42.1	23.8	108.8
T,LPM5	29.86	0.97	0.19	57	84	362,466	0.044	0.160	0.203	11.13	5.73	0.000	0.051	0.071	0.079	0.094	66.0	44.0	21.9	108.8
T,LPM6	29.65	0.97	0.15	57	84	362,949	0.069	0.141	0.209	11.09	5.70	0.000	0.054	0.060	0.079	0.108	67.0	46.5	20.5	108.1
T,LPM8	29.21	0.98	0.11	57	84	362,786	0.089	0.134	0.223	10.98	5.61	0.000	0.058	0.053	0.079	0.123	68.9	48.9	20.0	107.1
T,LPM10	29.12	0.98	0.13	57	84	361,648	0.095	0.130	0.225	10.96	5.60	0.000	0.059	0.052	0.079	0.130	70.0	50.0	20.0	106.3
T,DT	26.89	0.98	0.07	55	82	357,513	0.064	0.229	0.293	10.20	5.18	0.001	0.072	0.066	0.096	0.168	80.5	55.6	25.0	96.5

GCR = gross crane rate (total QC lifts / total hours of QC time beside a busy berth), *Berth Occ* = berth occupancy (average fraction of berths occupied at any instant), *BOA Rate* = the berth-on-arrival rate, i.e. the fraction of vessels that are able to berth upon their arrival, *% vol* = average percentage of storage volume occupied at any instant, *% area* = average percentage of storage area occupied at any instant, *QCLifts* = total number of QC lifts made, *QC WaitU* = fraction of time that an average QC spends waiting for YTs to appear during unloading (averaged over the entire simulation run), *QC WaitL* = fraction of time that an average QC spends waiting for YTs to appear during loading (averaged over the entire simulation run), *QC Wait* = QC WaitU + QC WaitL, *YC Prod* = average yard crane productivity (total YC lifts/total hours of YC time), *YT Prod* = average yard truck productivity (total number of containers placed on YT trailers/total hours of active YT time), *YT StateX* = fraction of time that an average YT spends in state *X* (averaged over the entire simulation run) (see Fig. 4), *Avg YT-Q* = time-averaged number of YTs heading towards or waiting at the quay (averaged over the entire simulation run), *Avg L YT-Q* = time-averaged number of laden YTs heading towards or waiting at QCs that are loading (averaged over the entire simulation run), *Avg U YT-Q* = time-averaged number of empty YTs heading towards or waiting at QCs that are unloading (averaged over the entire simulation run), *Avg YT-Y* = time-averaged number of YTs heading towards or waiting at the yard (averaged over the entire simulation run)

(increase) as LPM increases. Thus, YTs arriving beneath unloading (loading) QCs are waiting less (more) as LPM increases. Finally, columns "Avg L YT-Q" and "Avg U YT-Q" reveal that the time-averaged number of laden (empty) YTs heading towards or waiting at loading (unloading) QCs increases (decreases) as LPM increases. Thus, the value of LPM dramatically affects the equilibrium ratio of YTs whose next interaction is with loading versus unloading QCs. The optimal value of this ratio is about 2 for this scenario.

Table 5 also provides insights into container terminal operations in general. First, note that the numbers in the "QC WaitL" column are larger than those in the "QC WaitU" column for all control systems. This indicates that most QC delays occur during loading, not unloading. This is not surprising given the fact that YTs bringing containers to the quay during loading are typically not substitutable, whereas YTs bringing empty trailers to the quay during unloading are substitutable. This phenomenon, observed by container terminal personnel known to the author, helps to validate the simulation model. Also, note the high negative correlation between GCR and the terms in the "QC Wait" column. In particular, GCR is the highest (at 29.89 QC lifts/h) precisely when the fraction of total QC time spent waiting for YTs is the lowest (0.202).

6.4 Experiment 2: sending YTs to closer, less urgent jobs

We now explore how the parameter "StarvProx" affects GCR when the other parameters take their default values: Pool = T, Type = QS, LPM = 5, DualLoad = Yes, Trig = B1. Table 6 displays the experimental results. The experiments consider StarvProx = 0, 1, 2, 3, and 4 in all four terminal scenarios. Two asterisks (**) denote the default YT control system where StarvProx = 0. For the small (large) terminal scenarios, each data point is the average GCR from ten (six) replications. Thus, the table displays the results from a total of 160 runs. The results for StarvProx = 0 are copied from Table 4. Each run for scenario (SL, SM, LL, LM) required less than (6, 5, 14, 9) minutes of CPU time. The highest GCR value for each scenario is displayed in bold. The average GCR across all scenarios is displayed in the final column. The highest such average GCR (34.32) is highlighted and corresponds to the default system. For scenarios SM, LL, and LM, the GCR difference between the default system (**) and the best of the other systems (*) is not statistically significant. For scenario SL, the difference is significant at the 98% confidence level.

Table 6 GCR results from experiment 2 (QC lifts/h)

YT control system	Small term less equipment	Small term more equipment	Large term less equipment	Large term more equipment	Average
StarvProx = 0**	**32.41**	37.65	**29.86**	37.36	**34.32**
StarvProx = 1	31.99*	37.39	29.85*	**37.40***	34.16
StarvProx = 2	31.71	37.44	29.59	37.36	34.02
StarvProx = 3	31.73	**37.74***	29.39	37.21	34.02
StarvProx = 4	31.36	37.35	29.29	37.23	33.81

Table 7 GCR results from experiment 3 (QC lifts/h)

YT control system	Small term less equipment	Small term more equipment	Large term less equipment	Large term more equipment	Average
DualLoad = No	31.47*	36.81*	29.27*	37.26*	33.70
DualLoad = Yes**	**32.41**	**37.65**	**29.86**	**37.36**	**34.32**

The interpretation of Table 6 is straightforward. At the SL and LL terminals, GCR is decreasing in the value of StarvProx and the highest GCR occurs when StarvProx = 0. At the SM and LM terminals, GCR is roughly concave in the value of StarvProx and the highest GCR occurs when StarvProx $\geq$ 1. In other words, allowing free YTs to be assigned to less starved QCs (i.e. giving free YTs more jobs to choose from) is somewhat harmful when equipment is scarce and somewhat beneficial when equipment is plentiful.

A closer analysis of additional output data (not shown) has revealed that YTs are indeed being assigned to closer jobs as StarvProx increases. However, these YT travel time savings are accompanied by additional QC, YT, and container time spent waiting at the quay during loading. In other words, the YTs are less well-dispersed among the loading QCs as StarvProx increases. Thus, a higher StarvProx value causes too many YTs to be assigned to some loading QCs and too few YTs to be assigned to others. This gives rise to more QC starvation during loading than when StarvProx = 0. Overall, StarvProx = 0 remains a good option.

6.5 Experiment 3: dual loading

We now explore how the parameter "DualLoad" affects GCR when the other parameters take their default values: Pool = T, Type = QS, LPM = 5, StarvProx = 0, Trig = B1. Table 7 displays the experimental results. Two asterisks (**) denote the default YT control system where DualLoad = Yes. For the small (large) terminal scenarios, each data point is the average GCR from ten (six) replications. Thus, the table displays the results from 64 runs. The results for DualLoad = Yes are copied from Table 4. Each run for scenario (SL, SM, LL, LM) required less than (6, 5, 19, 10) minutes of CPU time. The highest GCR value for each scenario is displayed in bold. The average GCR across all scenarios is displayed in the final column. The highest such average GCR (34.32) is highlighted and corresponds to the default system. For scenarios SL, SM, and LL, the GCR difference between the default system (**) and the system with DualLoad = No (*) is statistically significant at the 99% confidence level. For scenario LM, the difference is not significant.

Table 7 shows that dual loading improves container terminal productivity by a statistically significant amount even when all cranes handle only one container at a time. Indeed, the GCR when DualLoad = Yes is higher than the GCR when DualLoad = No for every terminal scenario. Surprisingly, the GCR difference (34.32 versus 33.7) is only 1.8%. However, this is reasonable because (A) only about 60% of containers are

20′ long (Table 3) and (B) QCs and YCs only handle one container at a time in the simulation model. This latter aspect makes dual loading much less attractive than if QCs or YCs could handle two 20′ containers simultaneously. The GCR improvement when going from DualLoad = No to DualLoad = Yes in scenarios SL, SM, LL, and LM is 3.0, 2.3, 2.0, and 0.3% respectively. Thus, the benefits of dual loading diminish as the terminal becomes larger and as more YTs are deployed. The former result agrees with Grunow et al. (2004, 2006).

6.6 Experiment 4: simultaneous assignment of multiple YTs to multiple jobs

We now explore how the parameter "Trig" affects GCR when the other parameters take their default values: Pool = T, Type = QS, LPM = 5, StarvProx = 0, DualLoad = Yes. Table 8 displays the experimental results. The experiments consider 12 different values of "Trig" in each of the four terminal scenarios. Two asterisks (**) denote the default YT control system (Trig = B1). For the small (large) terminal scenarios, each GCR value is the average GCR from ten (six) replications. Thus, the table displays the results from 384 runs. The results for Trig = B1 are copied from Table 4. Each run for scenario (SL, SM, LL, LM) required less than (6, 4, 84, 26) minutes of CPU time. The highest GCR value for each type of control system (B or T) in each scenario is highlighted. The average GCR across all four scenarios is displayed in the final column. The highest average GCR values for the "B" and "T" systems (34.32, 34.32) are highlighted. For scenario LL (LM), the GCR difference between the default system (**) and the best of each of the "B" and "T" systems (*) is statistically significant at the 99% confidence level (is not significant). Seven non-GCR performance measures are included in the table.

Figure 8 is a graphical display of the GCR values in Table 8. This figure shows the dependence of GCR on the value of "Trig" for each scenario. In the figure, each point corresponds to a single GCR value in Table 8. Overall, the B2-B10 and T5-T30 systems perform modestly better than the default system (Trig = B1) in scenarios LL and LM but generally worse than the default system in scenarios SL and SM. In the LL (LM) scenario, the best group assignment system overall (Trig = T15) performs 1.1% (0.2%) better than the default system. These modest gains are relatively close to the gains of 1.4% observed by Briskorn et al. (2006) for a slightly different AGV control system.

Notice that the performance of the batch-size-triggered system is strikingly different at the small and large terminals. GCR drops off rapidly as the batch size increases in scenarios SL and SM, but GCR initially increases and appears to be concave in the batch size in scenarios LL and LM. This behavior can be explained as follows. In scenarios SL and SM, the YT fleet is small and the terminal occupies less area. Thus, a batch of n YTs ($2 \leq n \leq 10$) represents a significant fraction of the entire YT fleet, so it is too costly to keep $n-1$ YTs idle, especially when distances are relatively short and there is little to gain from minimizing empty YT travel distance. In scenarios LL and LM, on the other hand, the terminal occupies more area and the YT fleet is larger. The larger terminal means there is more at stake in the quest to minimize empty travel distance. Furthermore, the larger YT fleet means that a batch of n YTs constitutes

Table 8 Detailed results for experiment 4

Terminal	YT system	GCR	YC Prod	YT Prod	#YT Disp	YT State1	YT State2	YT State6	YT State126	Terminal	YT system	GCR	YC Prod	YT Prod	#YT Disp	YT State1	YT State2	YT State6	YT State126	Overall average GCR
	B1**	**32.41**	10.54	6.24	1.01	.003	.038	.044	.085		B1**	29.86	11.13	5.73	1.00	.000	.051	.079	.129	B1 = **34.32**
	B2	32.13	10.50	6.19	2.01	.015	.036	.042	.093		B2	30.05	11.16	5.77	2.00	.003	.048	.074	.125	B2 = 34.29
	B4	31.87	10.52	6.14	4.02	.039	.035	.040	.114		B4	30.13	11.17	5.78	4.00	.008	.044	.070	.122	B4 = 34.24
	B6	31.18	10.47	6.02	6.02	.064	.034	.039	.136		B6	30.12	11.19	5.78	6.00	.014	.042	.066	.123	B6 = 34.05
Small	B8	30.89	10.46	5.96	8.02	.087	.033	.038	.158	Large	B8*	**30.14**	11.18	5.79	7.99	.019	.039	.063	.122	B8 = 33.91
term	B10	30.15	10.49	5.83	10.02	.112	.032	.038	.182	term	B10	29.96	11.14	5.75	9.97	.025	.041	.062	.127	B10 = 33.66
less	T5	**32.33**	10.57	6.23	1.12	.006	.038	.044	.087	less	T5	30.02	11.16	5.76	1.71	.003	.048	.074	.125	T5 = 34.31
equip	T10	32.23	10.53	6.21	1.24	.008	.037	.043	.089	equip	T10	30.06	11.18	5.77	2.63	.007	.046	.070	.123	T10 = **34.32**
(SL)	T15	32.24	10.55	6.21	1.37	.011	.037	.043	.091	(LL)	T15*	**30.18**	11.17	5.79	3.69	.010	.044	.068	.122	T15 = **34.32**
	T20	31.95	10.56	6.16	1.50	.015	.037	.042	.094		T20	30.14	11.17	5.78	4.83	.013	.043	.066	.123	T20 = 34.31
	T25	31.80	10.52	6.13	1.65	.018	.037	.042	.096		T25	30.12	11.18	5.78	6.01	.017	.040	.064	.121	T25 = 34.21
	T30	32.03	10.55	6.17	1.80	.018	.036	.042	.097		T30	29.97	11.15	5.75	7.20	.020	.041	.064	.124	T30 = 34.23
	B1**	**37.65**	8.44	3.99	1.01	.007	.021	.025	.053		B1**	37.36	9.22	3.94	1.02	.000	.032	.046	.078	
	B2	37.59	8.47	3.99	2.02	.014	.021	.024	.059		B2	37.38	9.25	3.94	2.04	.002	.030	.043	.074	
	B4	37.54	8.50	3.99	4.05	.026	.020	.023	.069		B4*	**37.44**	9.25	3.95	4.07	.005	.027	.040	.071	
	B6	37.50	8.49	3.98	6.06	.040	.020	.022	.082		B6	37.40	9.21	3.95	6.10	.008	.025	.038	.071	
Small	B8	37.21	8.56	3.95	8.08	.052	.020	.022	.093	Large	B8	37.41	9.22	3.95	8.13	.011	.025	.037	.073	
term	B10	37.13	8.51	3.95	10.10	.064	.019	.022	.105	term	B10	37.37	9.23	3.94	10.16	.014	.023	.036	.073	
more	T5	37.47	8.47	3.97	1.14	.010	.021	.024	.056	more	T5	37.42	9.24	3.95	1.77	.002	.030	.043	.074	
equip	T10	37.54	8.50	3.99	1.27	.011	.021	.024	.056	equip	T10	37.44	9.22	3.95	2.74	.004	.028	.041	.072	
(SM)	T15	37.40	8.44	3.97	1.42	.014	.021	.024	.058	(LM)	T15*	**37.45**	9.21	3.95	3.83	.005	.026	.039	.071	
	T20	**37.72**	8.37	4.00	1.56	.013	.021	.024	.057		T20	**37.45**	9.25	3.95	4.98	.007	.025	.038	.071	
	T25	37.53	8.53	3.99	1.72	.015	.021	.024	.060		T25	37.40	9.22	3.95	6.19	.009	.025	.037	.072	
	T30	37.48	8.48	3.98	1.89	.016	.021	.023	.060		T30	37.44	9.24	3.95	7.41	.011	.025	.037	.073	

#YT Disp = avg. # YTs si multaneously dispatched per execution of YT dispatching logic (total # YTs dispatched)/(total # instants when ≥ 1 YT is dispatched)
YT State126 = fraction of time that an average YT spends in st ates 1, 2, and 6 combined (averaged over the entire simulation ru n) (Figure 4)

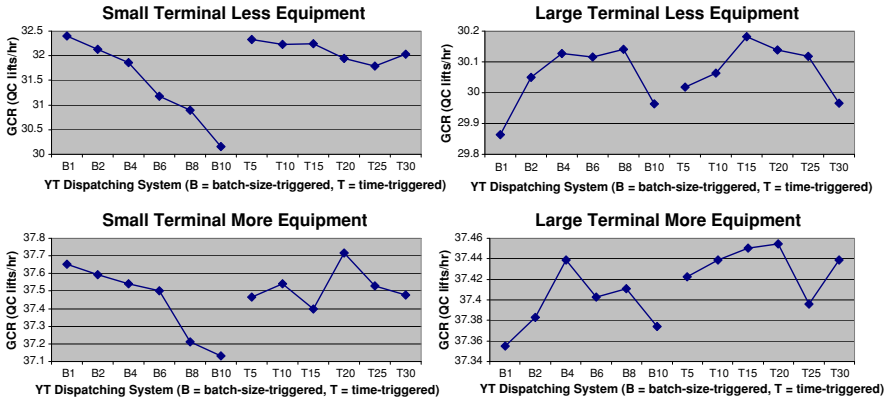

Fig. 8 GCR results from experiment 4

an insignificant portion of the entire YT fleet. Thus, the total percentage of YT time spent waiting for instructions is much less at the large terminal than the small terminal. The results in Table 8 support this thesis. In this table, the YT idle times (in the "YT State1" column) rapidly increase in the batch size in scenarios SL and SM. However, these values slowly increase in the batch size in scenarios LL and LM. Furthermore, the empty travel times ("YT State2" and "YT State6" columns) decrease only slightly as the batch size increases in scenarios SL and SM but decrease more notably as the batch size increases in scenarios LL and LM. To summarize, the risks involved in adopting a batch-size-triggered system at the large terminal are much smaller, and the potential rewards much greater, than at the small terminal.

We now discuss the time-triggered YT control system. This system performs generally worse than the default system (Trig = B1) at the small terminal but better than the default system at the large terminal. At the large terminal, GCR is roughly concave in the length of the time interval. The explanation of this phenomenon follows the logic of the preceding paragraph with one exception. As before, the difference in terminal size means there is less potential to reduce empty YT travel distance (YT State2, YT State6) at the small terminal than at the large terminal. However, the risks regarding YT idle time (i.e. the total fraction of TY time spent idling) are now the same for the two terminals because the system is time-triggered. Overall, the large terminal once again stands to benefit more than the small terminal from a time-triggered YT control system. The results in Table 8 also support this thesis.

The preceding discussion regarding YT idle time and empty travel time is particularly important because the sum of these two quantities is highly correlated with GCR. Table 8 shows this correlation. Indeed, the maximum GCR value and minimum "YT State126" value come from the same "Trig" value in 6 of the 8 portions of this table. Despite the above trends, we should note that the default YT control system performs very well, in fact as well across all four scenarios as any other system from a GCR standpoint. However, from a GCR and environmental standpoint, the T15 system is probably best because it achieves the same GCR with less fuel consumption (i.e. YT travel distance).

On a more general note, notice in the "YT Prod" and "YC Prod" columns in Table 8 that YT productivity is highly correlated with GCR; YC productivity, on the other hand, is somewhat less correlated with GCR. This phenomenon, which has been observed by container terminal personnel known to the author, provides additional validation of the simulation model.

7 Conclusion

In this study, we have shown how the long-run average quay crane (QC) rate at a seaport container terminal depends on the system that automatically assigns yard trucks (YTs) to container transportation jobs in the terminal in real time. Several real-time, dual-load YT control systems have been evaluated by a fully-integrated, discrete event simulation model of a vessel-to-vessel transshipment terminal. The model is designed to reproduce the microscopic, stochastic, real-time environment at a multiple-berth facility. The experiments have considered four terminal scenarios and 39 YT control system settings for each scenario. Six to ten 3-week simulation runs were performed for each of these combinations. Overall, more than 276 million QC lifts worth of activity was simulated.

Results show that a YT control system based on QC starvation yields a higher gross crane rate (GCR) than a system based on job due times. Also, GCR improves when all YTs are combined into a single, terminal-wide dispatching pool. However, such pooling should be accompanied by a loading preference multiplier (LPM) that indicates the system's preference for assigning free YTs to loading versus unloading jobs. YT dual loading improves GCR by a small but statistically significant amount even when all cranes handle only one container at a time. Finally, GCR at large terminals can be improved by about 1% if a group YT assignment system, which simultaneously assigns multiple YTs to multiple jobs, is used instead of an individual YT assignment system. Our comparison of many-to-many and 1-to-many dispatching systems is a significant contribution to the growing body of research on dynamic vehicle routing. This paper has also stressed the need for fully-integrated simulation models of container terminals that can test the viability of proposed algorithms/models within a real-time environment and evaluate performance in terms of a global metric such as GCR.

References

Briskorn D, Drexl A, Hartmann S (2006) Inventory-based dispatching of automated guided vehicles on container terminals. OR Spectrum 28:611–630

Dekker R, Voogd P, van Asperen E (2006) Advanced methods for container stacking. OR Spectrum 28:563–586

Grunow M, Günther H-O, Lehmann M (2004) Dispatching multi-load AGVs in highly automated seaport container terminals. OR Spectrum 26:211–235

Grunow M, Günther H-O, Lehmann M (2006) Strategies for dispatching AGVs at automated seaport container terminals. OR Spectrum 28:587–610

Günther H-O, Kim K-H (2006) Container terminals and terminal operations. OR Spectrum 28:437–445

Kim KH, Bae JW (2004) A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Transp Sci 38:224–234

Kim KH, Jeon SM, Ryu KR (2006) Deadlock prevention for automated guided vehicles in automated container terminals. OR Spectrum 28:659–679
Kozan E (1997) Comparison of analytical and simulation planning models of seaport container terminals. Transp Plan Technol 20:235–248
Legato P, Canonaco P, Mazza RM (2009) Yard crane management by simulation and optimisation. Maritime Econ Logist 11:36–57
Liu CI, Jula H, Ioannou PA (2002) Design, simulation, and evaluation of automated container terminals. IEEE Trans Intell Transp Syst 3:12–26
Liu C-I, Jula H, Vukadinovic K, Ioannou P (2004) Automated guided vehicle system for two container yard layouts. Transp Res Part C: Emerging Technol 12:349–368
Monaco MF, Moccia L, Sammarra M (2009) Operations research for the management of a transhipment container terminal: the Gioia Tauro case. Maritime Econ Logist 11:7–35
Murty KG, Liu J, Wan Y-W, Linn R (2005a) A decision support system for operations in a container terminal. Decis Support Syst 39:309–332
Murty KG, Wan Y-W, Liu J, Tseng MM, Leung E, Lai K-K, Chiu HWC (2005b) Hongkong International Terminals gains elastic capacity using a data-intensive decision-support system. Interfaces 35:61–75
Nishimura E, Imai A, Papadimitriou S (2005) Yard trailer routing at a maritime container terminal. Transp Res Part E: Logist Transp Rev 41:53–76
Parola F, Sciomachen A (2005) Intermodal container flows in a port system network: analysis of possible growths via simulation models. Int J Prod Econ 97:75–88
Petering MEH (2007) Design, analysis, and real-time control of seaport container transshipment terminals. PhD dissertation, University of Michigan, Ann Arbor, Michigan
Petering MEH (2009) Effect of block width and storage yard layout on marine container terminal performance. Transp Res E 45:591–610
Petering MEH, Murty KG (2009) Effect of block length and yard crane deployment systems on overall performance at a seaport container transshipment terminal. Comput Oper Res 36:1711–1725
Petering MEH, Wu Y, Li W, Goh M, de Souza R (2009) Development and simulation analysis of real-time yard crane control systems for seaport container transshipment terminals. OR Spectrum 31:801–835
Saanen YA (2004) An approach for designing robotized marine container terminals. PhD dissertation, Technical University Delft, Delft, Netherlands
Stahlbock R, Voß S (2008) Operations research at container terminals: a literature update. OR Spectrum 30:1–52
Steenken D, Voß S, Stahlbock R (2004) Container terminal operation and operations research - a classification and literature review. OR Spectrum 26:3–49
Vis IFA (2006) A comparative analysis of storage and retrieval equipment at a container terminal. Int J Prod Econ 103:680–693
Vis IFA, de Koster R (2003) Transshipment of containers at a container terminal: an overview. Eur J Oper Res 147:1–16