Scheduling of container-handling equipment during the loading process at an automated container terminal

Abstract:

To improve the operational efficiency of container terminals, it is important to consider the coordination of different types of container-handling equipment, which typically include vehicles, yard cranes and quay cranes. This paper addresses the integration of scheduling each constituent of handling equipment in an automated container terminal, in order to minimise the loading element of the ship's berthing time. A mixed-integer programming (MIP) model was developed to mathematically formulate this challenge. Small-sized problems can be solved optimally using existing solver. In order to obtain approximately optimal solutions for large-sized problems, an adaptive heuristic algorithm was created that can adjust the parameters of a genetic algorithm (GA), according to the observed performance. Experiments were carried out for both small-sized and large-sized problems to analyse the impact of equipment used in the loading process on berthing and computation times, as well as to test the efficiency of our proposed adaptive GA in solving this integrated problem.

Keywords: container loading, automated container terminal, container handling equipment, adaptive GA, mixed integer programming

1 Introduction

Containers are large steel boxes of standard sizes, which were first used in the maritime transportation industry during the 1950s. Since then, the container terminal has become an important intermodal interface between marine and land transportation systems. In order to raise productivity, as well as reduce labour costs, many terminals across the world have started to deploy automated equipment to manage the ever-increasing container traffic passing through container terminals, particularly automated container terminals. Since the equipment in such terminals is controlled automatically, it is imperative to establish an efficient scheduling system for determining handling sequences, timings of all types of container handling equipment, and, ultimately, achieve optimal performance from the container terminals.

Among the different types of automated equipment that are used worldwide, automated guided vehicles (AGVs) are the most representative. An AGV is a mobile robot that follows markers or wires on the floor, or uses vision, magnets or lasers for navigation. In this paper, we consider a typical automated container terminal, in which three types of container-handling equipment are involved: quay cranes (QCs), automated guided vehicles (AGVs) and yard cranes (YCs). QCs are equipped at the quayside for handling containers from and onto a ship; YCs are used for retrieving and stacking containers in the storage yard; while AGVs are deployed for transporting containers between the quayside and storage yard.

Typically, container terminal operations consist of (1) unloading operations, during which containers are unloaded from a ship, delivered, and then stacked in storage yards; and (2) loading operations, during which containers are handled in the reverse direction of the unloading operations. (These operations are described in figure 1.) In this paper, we examine loading operations, during which containers are picked up by YCs from the yard, and loaded onto AGVs; AGVs then move containers from the yard to the quayside, where QCs pick up containers from AGVs and load them onto the ship.

While a significant body of research has investigated the scheduling challenges of QCs, AGVs and YCs separately, these problems are in fact interrelated. As stated in L. H. Lee et al. (2010), the efficiency of the transportation between the quayside and the yard-side plays a crucial role in determining the productivity of the terminal, because it might delay the QC/YC operations if vehicles do not arrive in time, or cause traffic congestions if vehicles arrive early; QC/YC operations can also affect vehicle transportation, since vehicles have to wait if QCs/YCs are not available to handle containers when vehicles have arrived at the quayside/yard-side. QCs are the most expensive pieces of handling equipment, and represent the bottleneck resources in container terminals; YCs play an important role in improving the productivity of the loading operations (Wenkai Li et al., 2009) and in determining the overall efficiency of the handling system (L. Chen et al., 2007); Kim and Kim (1999) concluded that the time of loading can be reduced significantly through efficiently sequencing loading operations. Our study therefore investigates the integrated scheduling of all the handling equipment, in order to minimise the loading element of the ship's berthing time, which is the key measurement of operational efficiency in container terminals. The optimal container handling sequences by QCs, AGVs and YCs will be determined by solving the proposed integrated scheduling problem.

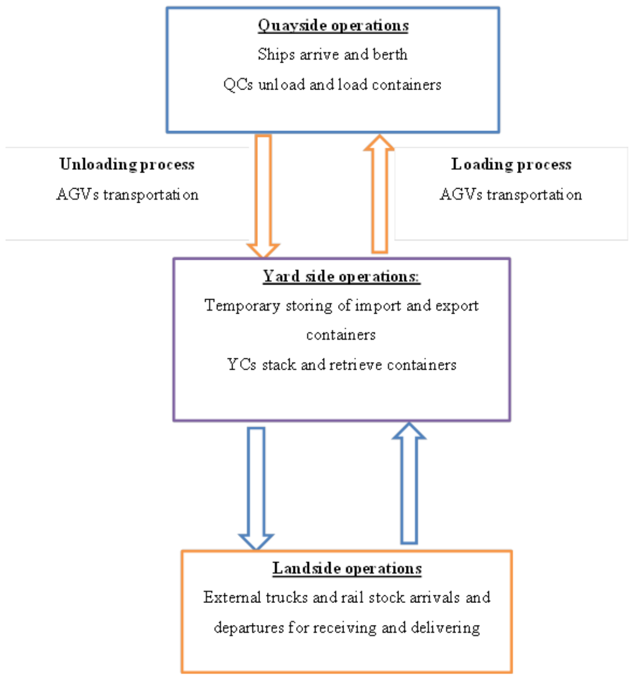
Quayside operations
Ships arrive and berth
QCs unload and load containers
Unloading process
AGVs transportation
Loading process
AGVs transportation
Yard side operations:
Temporary storing of import and export containers
YCs stack and retrieve containers
Landside operations
External trucks and rail stock arrivals and departures for receiving and delivering

Figure 1: Unloading and loading processes in a container terminal

The objectives of our research are twofold: first, in terms of methodology, we provide an integrated modelling approach, by considering the scheduling of QCs, AGVs and YCs in an automated container terminal. Apart from this modelling technique, an efficient adaptive genetic algorithm (GA) has also been developed to solve the NP-hard problem in large sizes with hundreds of containers; second, from a practical point of view, this study would benefit terminal managers when making decisions about coordinating different types of handling equipment, so as to improve the operational efficiency of terminals.

Following this introduction, section 2 reviews the literature on the scheduling problems in container terminals. The proposed integrated problem is described and formulated as a mixed-integer programming (MIP) model in section 3. In section 4, an adaptive GA is developed for problem solutions. Experiments are illustrated in section 5 to demonstrate the performance of the proposed model and algorithm. Section 6 concludes this paper.

2 Literature review

Research on container terminal operations has received significant attention from both academics and practitioners. This section will review the relevant studies of the scheduling challenges of different types of handling equipment in container terminals (i.e. the scheduling of QCs, AGVs and YCs), in addition to the associated integrated problems. The differences between our paper and related works are identified and discussed.

In the area of QC scheduling, the earliest research paper was written by Daganzo (1989), who analysed a situation in which ships were partitioned into bays, and where the movement of containers in a bay was defined as a task. The author suggested an algorithm to determine the number of cranes needing to be assigned to ship-bays of multiple vessels. Kim and Park (2004) assumed that there were unloading and loading tasks within the same ship-bay, and the QC schedule for each container was determined; they also developed an efficient heuristic approach based on a greedy randomised adaptive search procedure to overcome computational obstacles. D. H. Lee, Wang, et al. (2008) proposed a genetic algorithm (GA) to find the optimal handling sequence of holds for QCs assigned to a ship, while considering the case of interferences between different QCs. More recently, Al-Dhaheri et al. (2016) applied a simulation-based GA for the QC scheduling in order to minimise ship handling times. Kasm and Diabat (2019) incorporated both non-crossing and safety clearance constraints into the QC scheduling, and accordingly developed a partitioning heuristic algorithm.

From the perspective of AGV scheduling in automated container terminals, most studies have focused on its dispatching methods, which can be defined as the assignment of AGVs to deliver containers. Kim and Bae (2004) proposed a mixed-integer programming (MIP) model, aiming to minimise both the total travel time of AGVs and a delay in the completion time of QCs. Briskorn et al. (2007) presented an alternative formulation of the AGV assignment problem, based on a process analogous to inventory management, which was solved using an exact algorithm. Angeloudis and Bell (2010) studied an assignment algorithm for AGVs under unreliable conditions, which was suitable for real-time control of AGVs. The developed algorithm was applied to a simulated port environment, where it was found to outperform the well-known heuristics. Choe et al. (2016) proposed an online referencing learning algorithm for scheduling AGVs by adapting the policy dynamically, which recommended the best assignment. Zhong et al. (2020) designed a conflict-free path plan for AGVs, and solved this by a combined GA and particle swarm optimisation method.

In the field of deployment and scheduling of YCs, Zhang et al. (2002) formulated a YC-deployment problem aimed at finding the times and routes of YC movements, in order to minimise the total delayed workload in the yard. The problem was conceived as an MIP model, and solved by Lagrangian relaxation to obtain an optimal solution. Ng and Mak (2005) investigated a novel model to study the YC scheduling problem, with several given ready times to minimise the sum of job-waiting times. A branch-and-bound algorithm was proposed to solve this problem, and a set of tests based on real data was generated to evaluate its efficacy. He et al. (2010) developed a dynamic YC-scheduling model based on a rolling-horizon approach via objective programming, with the aim of minimising the total delayed workloads. A hybrid algorithm, which employed heuristic rules and a parallel genetic algorithm, was used to solve the NP-hard problem. Speer and Fischer (2016) studied the branch-and-bound (B&B) procedure for YC scheduling at seaport terminals in light of crane interferences. Xiaolong Han et al.

(2019) developed an MIP model for scheduling twin-automated YCs to serve storage and retrieval requests, aiming to minimise the makespan of all requests.

Some studies have examined the integrated scheduling of different pieces of handling equipment. Lau and Zhao (2008) investigated an integrated scheduling model of different types of handling equipment at automated container terminals. Their model aimed to minimise the total travel time of YCs and AGVs, and the delays of QC operations. A multi-layer genetic algorithm was generated to obtain a near-optimal solution for the integrated problem. Cao et al. (2010) addressed an integrated problem for yard truck and YC-scheduling during loading operations, which was formulated as a MIP model. Two efficient solution methods, based on Benders' decomposition, were developed for model implementation. Dkhil et al. (2017) developed an integrated model that took account of storage location and vehicle scheduling problems during the unloading process, whose objective was to minimise the total operation costs; a tabu search was used as the solution method. X. Chen et al. (2020) studied the integrated AGV and YC scheduling as a multi-robot coordination and scheduling problem. A real-time scheduling framework was proposed based on the rolling-horizon method.

This study is based on our previous research regarding the integration of vehicle scheduling and container storage allocation in the unloading process (Luo et al., 2016) and dual-cycle process (Luo & Wu, 2015). We further developed an integrated model for the loading process, during which we focused on the scheduling of AGVs, QCs and YCs. Unlike most of the existing literature, which has considered the minimisation of total travel times, waiting times or delay times as the main objectives, our model aims to ensure a minimum berthing time, which is commonly used to evaluate a terminal's operational efficiency. To achieve the capability for solving practical sized problems with hundreds of containers, we developed an adaptive GA, which consists of a novel chromosome design, self-adjusted GA parameter, and works efficiently in all the examples for returning results in a reasonable computation time.

3 Problem formulation

During the loading process, a container that is picked up by a YC is loaded onto an AGV, which will deliver this container to the quayside. At the quayside, a QC picks up the container from the AGV and loads it, in its correct location, onto the ship (see figure 2). In this study, we adopt the pooling strategy for AGVs, which means that AGVs can serve any QC. Congestion among AGVs on the guide path is not considered, since studying the interference of vehicles involves more complex scheduling of detailed movements of vehicles, though this does constitute another important issue associated with the AGV system in automated container terminals (Evers & Koppers, 1996).

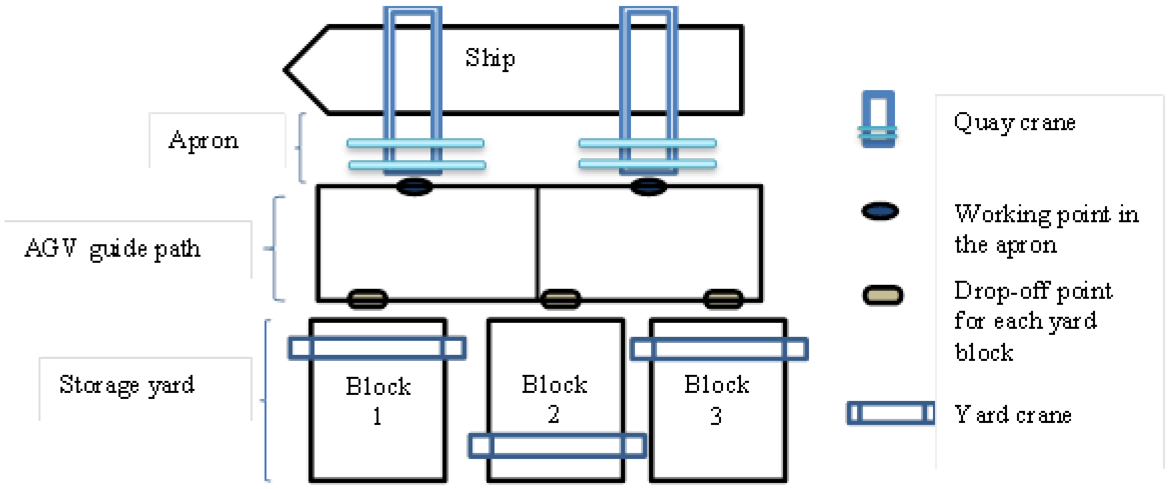

Figure 2: The layout of an automated container terminal being analysed

Each export container will be loaded onto a specified stack on the ship; the time required to load an export container by a QC depends on its stack location on the ship. In addition, within the vessel, ship bays (consisting of several stacks) are partitioned, each of which will be served by one QC (D. H. Lee, Cao, et al., 2008). Interferences among QCs are not considered in this study because, in reality, any two adjacent QCs must be set apart from each other by at least one ship bay to avoid interference. If a container is in a ship bay, which is in the covered area of a QC, the container will be handled by this QC. Accordingly, the destinations of containers (i.e. which QC will handle which container) are known. Usually, containers in the same ship stack will go to the same destination; these containers can thus be loaded in any order within the stacks. In our proposed problem, it is therefore assumed that the containers can be handled by QCs in any order. The same assumption has been made in the work of Cao et al. (2010). The detailed QC schedule will then be determined based on the information of container locations on the ship, as well as synchronisation with other handling equipment.

Export containers usually arrive at a terminal over a period of more than a week before the scheduled loading time, and are assigned to particular slots in the storage yard (Kang et al., 2006). Therefore, in this study, it is assumed that the yard locations of export containers are known. We define the container yard location by the YC's handling time of this container, which is the time the YC requires to handle a container from its yard location to the transfer point in front of the block.

The problem being investigated is how to minimise the loading element of the ship's berthing time, which is the time during which all export containers have been delivered and placed onto the ship. This loading time duration consists of (1) the retrieval time for containers by YCs at the storage yard; (2) the travelling time of containers from the storage yard to the ship by AGVs; and (3) the handling time required for all containers to be placed onto the ship by QCs. Apart from deciding QC schedules, as mentioned above, one of our objectives is to ascertain the sequence and time of containers handled by the AGVs, because the transportation between the yard-side and quayside plays a crucial role in determining the productivity of the terminal (L. H. Lee et al., 2010). The problem will also determine the container-handling sequences of the YCs during the loading operations, which is very important in improving the overall efficiency of the system (L. Chen et al., 2007; W. Li et al., 2009). An integrated model is developed in this section to address these operational decisions at the same time.

From the descriptions above, we summarise the assumptions considered in this study:

(1) The yard storage locations for export containers are given.

(2) The locations of QCs by which containers will be loaded (destinations of containers) are known.

(3) The number of export containers, and number of QCs, YCs and AGVs are all known.

(4) QCs, YCs and AGVs can only take one container at a time.

(5) A pooling policy is applied to the scheduling of AGVs, which means that AGVs are shared among all the QCs.

(6) The travelling times of AGVs between any two processing locations (transfer points of blocks and QCs) are known, e.g. the travelling time between each QC and block, and the travelling time among transfer points between blocks.

(7) Traffic congestion of the AGVs on the path is not considered.

(8) The interference among YCs and the interference among QCs are not considered.

(9) The time needed for YCs dropping off containers onto AGVs and QCs picking up containers from AGVs are not considered.

A mixed-integer programming (MIP) model will be employed for the integrated scheduling problem. As discussed, the objective is to minimise the berthing time of the ship. The main operational decisions seek to determine the schedules, i.e. the times and sequences to handle containers by AGVs, YCs and QCs. The following represent the notations related to this problem.

Sets and parameters

N	set of export containers
Q	set of QCs
W	set of YCs
K	set of AGVs
k	index for AGVs
w	index for YCs
q	index for total number of QCs
i *and* j	index for export containers
m *and* n	index for handling sequences
h_i	QC’s handling time of container i

t_i	AGV's travelling time for taking container *i* from storage yard to quayside
p_i	YC's handling time of container *i*
s_{ij}	AGV's travelling time from the QC, which handles container *i*, to the block which stores container *j*
w_{ij}	YC's travelling time from the yard block, which stores container *i*, to the yard location of container *j*
M	a very large positive number
S	dummy starting container
F	dummy ending container
O_S	the container set which contains all the export containers and the dummy starting container $$O_S = N \cup \{S\}$$
O_F	The container set which contains all the export containers and the dummy ending container $$O_F = N \cup \{F\}$$
O	The container set which contains all the export containers and dummy starting and ending containers $$O = \{S, F\} \cup N.$$

Decision variables

u_i	the time a QC picks up container *i* from an AGV
d_i	the time a YC releases container *i* onto an AGV

In this study, the intention is to decide the sequences and times of containers for QCs, YCs and AGVs to operate. We subsequently introduced the following three variables:

$$x_{ik}^{m} = \begin{cases} 1, if \;\; container\ i \text{ is the } m\text{th container delivered by AGV } k \\ 0\,, otherwise;\ \forall i \in N, \forall m \in N^{+}, \forall k \in K \end{cases}$$

$$y_{iw}^{n} = \begin{cases} 1, if\ container\ i \text{ is the } n\text{th container handled by YC } w \\ 0\,, otherwise;\ \forall i \in N, \forall n \in N^{+}, \forall w \in W \end{cases}$$

$$z_{ij} = \begin{cases} 1, if \;\; container\ j \text{ is handled immediately after conainer } i \text{ by same QC} \\ 0\,, otherwise;\ \forall i \in O_S, \forall j \in O_F \end{cases}$$

Objective: minimise the ship's berthing time

$$Min: Max_i(u_i + h_i)$$

The objective is to minimise the loading element of the ship's berthing time, which is described as the time during which all the export containers have been loaded onto the ship.

Subject to:

Constraint set 1: AGV transport sequences

Constraint (1) implies that container i must be delivered once and only once by an AGV taking it from the storage yard to the quayside. Constraint (2) means that an AGV can only deliver one container at a time. Constraint (3) indicates that containers must be delivered by an AGV sequentially.

$$\sum_{m\in N^+}\sum_{k\in K} x_{ik}^m = 1, \forall i \in N \tag{1}$$

$$\sum_{i\in N} x_{ik}^m \leq 1, \forall m \in N^+, \forall k \in K \tag{2}$$

$$\sum_{i\in N} x_{ik}^m \geq \sum_{i\in N} x_{ik}^{m+1}, \forall m \in N^+, \forall k \in K \tag{3}$$

Constraint set 2: YC handling sequences

Constraint (4) ensures that each container $i \in N$ must be handled once, and only once, by a YC, taking it from its yard location to the transfer point of the yard block. Constraint (5) means that a YC can only carry one container at a time. Constraint (6) guarantees that containers must be handled by a YC sequentially.

$$\sum_{n\in N^+}\sum_{w\in W} y_{iw}^n = 1, \forall i \in N \tag{4}$$

$$\sum_{i\in N} y_{iw}^n \leq 1, \forall n \in N^+, \forall w \in W \tag{5}$$

$$\sum_{i\in N} y_{iw}^n \geq \sum_{i\in N} y_{iw}^{n+1}, \forall n \in N^+, \forall w \in W \tag{6}$$

Constraint set 3: QC handling sequences

Constraint (7) means that for every container $\in N$, there is exactly one container $i \in O_S$ handled before it and by the same QC. Constraint (8) means that for every container $i \in N$, there is exactly one container $j \in O_F$ handled after it, and by the same QC. Constraints (9) and (10) guarantee that the number of QCs employed for handling these containers is exactly q.

$$\sum_{i\in O_S} z_{ij} = 1, \forall j \in N \tag{7}$$

$$\sum_{j\in O_F} z_{ij} = 1, \forall i \in N \tag{8}$$

$$\sum_{i \in N} z_{iF} = q \tag{9}$$

$$\sum_{j \in N} z_{Sj} = q \tag{10}$$

Constraint set 4: Time constraints for containers handled by the same equipment

Constraint (11) implies that if container *i* and container *j* are assigned to the same AGV, and container *j* is delivered immediately after container *i* by this AGV, the time between when this AGV starts handling container *j* from the yard and completes its delivery of container *i* at the quayside must be set apart by a certain travelling time of this AGV. Constraint (12) suggests that if container *i* and container *j* are assigned to the same YC, and container *j* is handled after container *i* by this YC, then the time between when this YC starts handling container *j* and finishes handling container *i* must be set apart by a certain handling time and travelling time. Constraint (13) posits that if container *j* is handled after container *i* by the same QC, then the QC can only start handling container *j* after finishing handling container *i*, i.e. after container *i* has been loaded onto the ship. Constraint (14) states that a QC can only start handling container *i* after container *i* has been delivered from the yard to the quayside.

$$d_j + M\left(2 - x_{ik}^{m} - x_{ik}^{m+1}\right) \geq u_i + s_{ij}, \forall i \in O_S, j \in O_F, \forall m \in N^+, \forall k \in K \tag{11}$$

$$d_j + M\left(2 - y_{iw}^{n} - y_{iw}^{n+1}\right) \geq d_i + w_{ij} + p_j, \forall i \in O_S, j \in O_F, \forall n \in N^+, \forall w \in W \tag{12}$$

$$u_j + M\left(1 - z_{ij}\right) \geq u_i + h_i, \forall i \in O_S, \forall j \in O_F \tag{13}$$

$$u_i \geq d_i + t_i, \forall i \in N \tag{14}$$

Constraint set 5: Binary and non-negative restrictions

Constraints (15)-(16) provide the restrictions of the decision variables, which are binary and non-negative.

$$x_{ik}^{m}, y_{iw}^{n}, z_{ij} \in \{0,1\}, \forall i, j \in O, \forall m, n \in N^+, \forall k \in K, \forall w \in W \tag{15}$$

$$u_i, d_i \geq 0, \forall i \in N \tag{16}$$

The above problem is NP-hard, so it is difficult to solve the problem with hundreds of containers within a few minutes. In the following section, we propose a heuristic method, i.e. an adaptive genetic algorithm, for solving large-sized problems within a reasonable computation time.

4 The proposed adaptive genetic algorithm

The proposed adaptive heuristic algorithm uses all the features of a genetic algorithm (GA), while adopting the self-adjustment of GA parameters to improve the diversity of the population (Prasad et al., 2005). The GA is a well-known heuristic approach, its efficiency verified by the large number of challenges in the field of container terminal operations, and employed to solve large-sized problems with approximately optimal solutions (Bazzazi et al. (2009); Fu et al. (2014); X. Han et al. (2010); L. H. Lee et al. (2010); Tavakkoli-Moghaddam et al. (2009)). Although many new advanced metaheuristic techniques are available, as summarised in a recent paper by Gharehgozli et al. (2016), the proposed adaptive GA is the most apposite for our proposed problem. This is because the decision variables in our study can be easily interpreted by the chromosomes, and such an algorithm is efficient in obtaining solutions in a relatively short computation time. The detailed algorithm design is explained in the following steps:

Chromosome representation and initialisation: The decision variables of the mathematical model are usually described as 'chromosomes' when designing a GA. In this paper, the chromosomes represent the three main decision variables, x_{ik}^{m}, y_{iw}^{n} and z_{ij}, as stated in the model, which define the container handling sequences by AGVs, YCs and QCs respectively. In figure 3, we provide an example of chromosome representation with six containers, three AGVs, three YCs and two QCs. In such a representation, each container, AGV, YC and QC will be assigned a number as a label for easy description in the algorithm.

Figure 3(a) can be explained as follows:

(1) Take AGV 2 as an example: in this figure, AGV 2 will deliver container 1 first, then deliver container 3; mathematically, this means $x_{12}^{1} = 1$ (container 1 is the first container delivered by AGV 2) and $x_{32}^{2} = 1$ (container 3 is the second container delivered by AGV 2).

(2) Now look at YC 3: YC 3 will handle container 4 first then handle container 5; mathematically, this means $y_{43}^{1} = 1$ (container 4 is the first container handled by YC 3) and $y_{53}^{2} = 1$ (container 5 is the second container handled by YC3).

(3) Explanations for other AGVs and YCs follow the above steps. Assigning AGVs and YCs in this manner will ensure that all the containers will be handled by an AGV and a YC once.

Assuming that containers 1, 3 and 4 will be handled by QC 1, and that containers 2, 5 and 6 will be handled by QC 2, figure 3(b) shows one possible solution for the QCs' handling sequences:

(1) QC 1 will handle container 4 first, then container 1 and then container 3, in that sequence; mathematically, this means $z_{41} = 1$ and $z_{13} = 1$.

(2) QC 2 will handle container 2 first, then container 6 and then container 5, in that sequence; mathematically, this means $z_{26} = 1$ and $z_{65} = 1$.

Container	*AGV*	*YC*
1	2	2

2	3	1
3	2	2
4	1	3
5	1	3
6	2	1

(a)

QC 1	***QC 2***
4	2
1	6
3	5

(b)

Figure 3: (a) An illustration of chromosome representation for the AGV and YC handling sequences; (b) An illustration of chromosome representation for the QC handling sequences

Parents' selection strategy: Parents' selection strategy decides how to choose chromosomes in the current population as parents, creating offspring for the next generation. Here, we adopted the most common method, which is the 'roulette wheel' sampling, to select parents for the next generation.

Genetic operators' design: A genetic operator helps to improve the solution gradually in the evolving process, while maintaining the feasibility of the newly generated offspring for the problem. We used the uniform crossover and swap mutation, which are described as follows:

(1) Uniform crossover: this type of crossover operator generates a template binary string of uniformly distributed "1"s and "0"s with the same length as that of the parents. The template string is then mapped to one of the parents, in which the genes that have the same positions with "1"s in the template string given to a child; the remaining empty genes of this child are filled from another parent. This crossover can be directly used for the chromosome of AGV and YC sequences, as illustrated in figure 4. For the QC sequences part, it was necessary to delete the duplicate genes from another parent and then fill the remaining genes to generate a feasible child, which is called the 'uniform order-based crossover'.

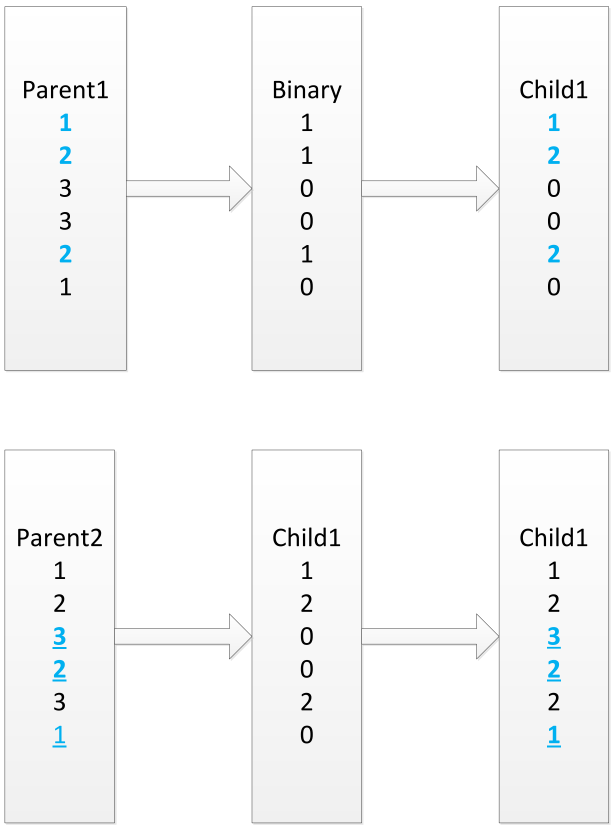

Figure 4: An illustration of uniform crossover for an example of six containers, three AGVs and three YCs

(2) Swap mutation: according to the mutation rate P_m, swap mutation is carried out by choosing two positions of the chromosomes randomly, and then swapping the genes on these positions. Since the selection of values for the mutation rate can vary a great deal and does not always yield the best GA performance (Reardon, 1998), we designed an adaptive process for selecting this value. The mutation rate here is self-adjusted according to the performance of the current generation, in order to help secure a diverse population. The value P_m therefore changes with respect to the standard deviation of the fitness values of all the chromosomes (σ_T) after crossover. In particular, if σ_T is small, which indicates a lower diversity in the population, the value of P_m increases with an increment of a certain value – for example, 0.01 – to ensure that mutation is performed on more individuals. On the other hand, if σ_T is large, we keep the value of P_m at the initial setting.

Offspring acceptance strategy: We used a semi-greedy strategy to accept the offspring created by the GA operators. In this strategy, an offspring is accepted as the new generation only if its objective function value (OFV) is less than the average of the OFVs of its parent(s), because we are aiming to minimise the OFV.

Stopping criterion: We used two criteria as stopping rules: (1) the maximum number of evolving generations M_g allowed for the adaptive GA, which is a common criterion adopted by many GA-based optimisation problems (Bazzazi et al., 2009; Huang et al., 2009; Kozan & Preston, 1999); and (2) the standard deviation of the fitness values of chromosomes (σ_T) in the current generation T is below a small value (Tavakkoli-Moghaddam & Safaei, 2006).

5 Experimental results

The following sets of experiments were carried out to investigate the efficiency of the model and the algorithm developed: (1) cases with a small number of containers; (2) cases with a large number of containers; (3) efficiency analysis; (4) a comparison of computational times.

For small-sized problems, results obtained by the B&B algorithm (implemented by CPLEX in AIMMS 3.11) and the adaptive GA were compared in terms of objective function value (OFV) and the computation time of the model. Due to the exponential increase in the computation time of the B&B algorithm as the problem size grows larger, it is impossible to solve the problem with the B&B algorithm in order to obtain the exact solution for large sizes. Therefore, the adaptive GA is adopted for solving large-sized problems by providing approximately optimal solutions in a reasonable time. The adaptive GA is implemented in MATLAB 7.11. All the experiments were performed on a computer with Intel® Core™ i3 CPU M370@2.40GHz and 4GB RAM under the Windows 7 operating system. For each problem, we used the same initial setting of parameters in order to compare and test the results. Each model was run 20 times by the proposed adaptive GA, and the means of objective function values and computation times are reported.

Model parameter settings

(1) The number of containers varies from 5 to 250, where 5-20 are considered small-sized problems and 20-250 are considered large-sized problems. We also considered that the number of AGVs varies from 2 to 15, the number of YCs varies from 2 to 8, and the number of QCs varies from 2 to 3.

(2) The uniform distribution was assumed for all the operation times. The processing times h_i of each QC on these containers follows uniform distribution U(30, 180)s, and the handling times p_i of each YC from each container's available location to the transfer point of the block, in which this container is located, follows uniform distribution U(60,140)s.

(3) The values of t_i, i.e. AGV's travelling time for taking container i from the storage yard to quayside, can be calculated by the AGV's travelling time from the yard block where container i locates to the quay crane that handles this container; similarly, the values of s_{ij} (AGV's travelling time from the destination QC of container i to the origin block of container j, i.e. the block which stores container j,) and the values of w_{ij} (YC's travelling time from the transfer point of the block which stores container i to the yard location of container j) can also be calculated. An example of these calculations is shown in Appendix 1.

The adaptive GA parameters' settings

GA parameters take the following initial settings based on our preliminary tests: crossover rate P_c is 0.7; the self-adaptive mutation rate P_m starts with 0.01 and the increment value of 0.01; population size *Pop* is 100; and maximum generations M_g is 40.

5.1 Results for small-sized problems

This set of experiments is for small-sized cases, with the number of containers ranging from 5 to 20. An illustrative example is presented in Appendix 1, demonstrating how the model works. These experiments aim to test the efficiency of our model and to compare the results obtained from the B&B by existing software and from our proposed adaptive GA. Table 1 compares results of the B&B algorithm and our proposed GA for small-sized problems. As expected, when the number of containers is increased, i.e. the problem size becomes larger, the objective function value (OFV) increases accordingly. The OFV represents the berthing time as described in the model. The results from the B&B are the optimal solutions; it can be observed that there is little difference between the OFVs obtained by these two algorithms. Compared with the optimal results, the average OFV difference between the B&B and the GA is only 1.5%, which is a very promising result. Regarding the computation time, the proposed GA can obtain approximately optimal solutions in a faster computation speed: the computation time of the GA for small-sized problems ranges from 1.97 seconds to 3.98 seconds, while the computation time of the B&B ranges from 2.82 seconds to 16044.91 seconds for solving the cases up to 10 containers. It should also be noted that, in order to obtain the optimal solutions, more computation time is required alongside the increase in the number of containers. The computation time of the B&B increases exponentially, and is unable to provide solutions with more than 10 containers. Thus, our adaptive GA is adopted for solving large-sized problems in the next section.

Table 1: Results of computational experiments in small sizes

No	**Containers**	**AGVs/QCs/ YCs**	**B&B (MIP)**		**Adaptive GA**		**OFV Gap rate (%)**
			Computation time (s)	OFV (s)	Computation time (s)	OFV (s)	
1	5	2/2/2	2.82	407	1.97	407	0%
2	6	2/2/2	24.37	541	2.88	543	0.3%
3	7	2/2/2	172.02	604	3.06	612	1.3%
4	8	2/2/2	183.69	640	3.44	654	2.2%
5	9	2/2/2	667.09	750	4.03	767	2.3%
6	10	2/2/2	16044.91	1867	3.98	1898	1.7%
7	10	3/2/2	/	/	3.25	1715	/
8	15	3/3/3	/	/	4.11	2108	/
9	15	2/3/3	/	/	4.32	2342	/
10	20	2/3/2	/	/	4.61	2985	/

5.2 Results of large-sized problems

As discussed in the earlier section, due to the complexity of the proposed problem, it is difficult to obtain optimal solutions for the problem in large sizes from the B&B algorithm using AIMMS 3.11; thus, we employed our proposed GA, which is able to obtain near-optimal solutions in reasonable computation times for large-sized problems. We ran a series of large-sized problems, the number of containers varying from 30 to 250, with different combinations in the numbers of AGVs, QCs and YCs. Table 2 shows the computation results from the GA in large-sized cases. After analysing the results obtained from the computational experiments, it can be observed that:

a) Generally, there has been a trend of increasing berthing times.

   As the problem size becomes larger, i.e. more containers, it takes more time to deliver all these containers from the yard and load them onto the ship.

b) The impacts of YCs and AGVs on the OFVs are different.

   As containers are handled by a series of different types of equipment, it is necessary to investigate the effects of the number of different types of handling equipment on the OFV. Comparing case 14 with case 15, the OFV has improved by 15.24% when the number of YCs changes from 2 to 3, with all other variables the same; comparing case 11 with case 12, the OFV has improved by 8.37% when the number of AGVs changes from 3 to 4, with all other variables the same. It can be concluded that YCs are more likely to be the bottleneck resources in container terminal operations, which has a significant impact on the efficiency of the terminal.

c) Our adaptive GA shows convergence behaviour.

   We applied our proposed GA to the problem with 50 containers, five AGVs, three QCs and five YCs with the optimal GA parameter settings identified in previous experiments; the algorithm was run 10 times. The results are illustrated following the pattern shown by the box plot in figure 5. Each box represents the OFVs of the 10 runs in one generation. The central mark is the median of OFVs, the edges of the box are the 25th and 75th percentiles and the whiskers are the most extreme data points. The data revealed that our proposed GA performs in a stable manner in all the experiments, and the OFVs of the best solutions at 40 generations were getting closer to each other.

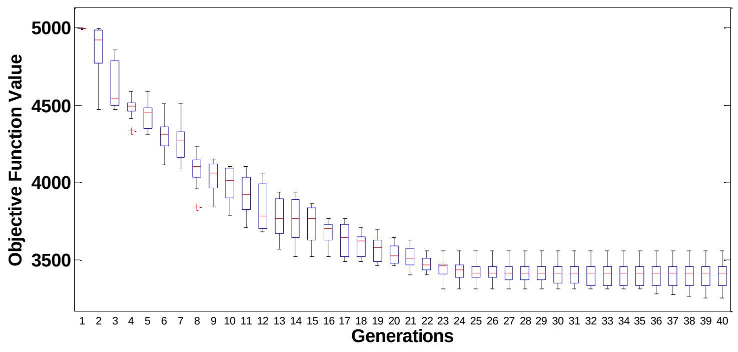

Figure 5: GA performance in 10 runs with $P_c = 0.7$, $initial\ P_m = 0.01$ and $Pop = 100$ for the case with 50 containers, five AGVs, three QCs and five YCs

Table 2: Results of computational experiments in large sizes by the adaptive GA

No	**Containers**	**AGVs/QCs/ YCs**	**Computation time (s)**	**OFV (s)**
11	30	3/3/2	5.95	4773
12	30	4/3/2	7.72	4377
13	30	5/3/2	12.17	4444
14	40	4/3/2	8.28	5327
15	40	4/3/3	10.19	4515
16	40	4/3/4	15.14	3573
17	50	4/2/4	14.71	4089
18	50	5/2/4	7.91	4136
19	50	5/2/5	19.38	3423
20	60	6/3/5	14.70	3938
21	70	6/3/5	23.47	4379
22	80	6/3/5	26.35	4518
23	90	6/3/5	21.51	4863
24	100	6/3/6	27.33	6352
25	100	8/3/6	41.06	6099
26	100	10/3/6	45.05	6128
27	150	10/3/5	74.92	9794
28	150	10/3/6	54.05	8822
29	150	10/3/7	48.17	8338
30	200	10/3/6	89.81	11768
31	200	15/3/6	66.23	10196
32	200	10/3/8	60.73	9716
33	250	10/3/4	134.06	21402
34	250	10/3/6	107.56	15508
35	250	10/3/8	109.05	12284
36	250	15/3/8	131.31	11553

d) The adaptive GA is able to return results for large-sized problems in a very short time.
In most of the experiments shown in table 2, the GA is able to achieve solutions within one minute. Even in the case of 250 containers, 15 AGVs, three QCs and eight YCs, it only takes about two minutes to solve. This indicates that the proposed GA is reliable for different-sized experiments.

5.3 Efficiency analysis for large-sized cases

Efficiency is defined as the ratio between output and input. In our considered cases, the inputs are QCs, AGVs and YCs given the same available space, labour level and relating costs level. The outputs are represented by the number of containers handled per minute, which are obtained based on the calculations from results in table 2. For example, in case 18 of table 2, it takes 4136 seconds (i.e. 68.93 minutes) to handle 50 containers; output is thus calculated as 50/68.93=0.725. It means that in any one minute, there will be 0.725 containers handled for this case. Since we have multiple inputs and one output in each case, the efficiency is calculated by using weights:

$$Efficiency = \frac{weighted\ outputs}{weighted\ inputs}$$

Therefore, efficiency is a value between 0 and 1. For example, the above equation becomes $efficiency = \frac{container\ per\ minute*u}{QC*v1+AGV*v2+YC*v3}$, where v1, v2 and v3 are input weights and u is the output weight; QC, AGV and YC represent the number of QCs, AGVs and YCs. A base case is randomly chosen as highlighted in table 3, with 250 containers, 3 QCs, 10 AGVs and 6 YCs. We aimed to discover the set of weights that maximises the efficiency of this case and therefore compare its efficiencies relative to all other cases. The corresponding linear programming model is presented in Appendix 2. By solving the model, the optimal efficiency of this base case is 81.1%, and the weights are $u = 0.839, v1 = 0.196, v2 = 0, v3 = 0.069$. The weight value (v1) is higher than the weight value (v3), which means that QCs are more critical resources in comparison with YCs. The value of v2 is 0, which means that the AGVs are less critical compared with QCs and YCs. By substituting the same weights to the above formula, we obtained the efficiency scores in each case relative to others. For example, the case before the base case has an efficiency of 68.1%, which means that it is less efficient in relation to the considered base case, which has an efficiency of 81.1%.

Table 3: Efficiency scores for large-sized examples

# of QCs	# of AGVs	# of YCs	Containers/Minute	# of Containers	Efficiency Score
3	3	2	0.377	30	0.436
3	4	2	0.411	30	0.475
3	5	2	0.405	30	0.468
3	4	2	0.451	40	0.521
3	4	3	0.532	40	0.561
3	4	4	0.672	40	0.653
2	4	4	0.734	50	0.923
2	5	4	0.725	50	0.913
2	5	5	0.876	50	1
3	6	5	0.914	60	0.823
3	6	5	0.959	70	0.864
3	6	5	1.062	80	0.957
3	6	5	1.110	90	1
3	6	6	0.945	100	0.792
3	8	6	0.984	100	0.825
3	10	6	0.979	100	0.821
3	10	5	0.919	150	0.828

3	10	6	1.020	150	0.856
3	10	7	1.079	150	0.847
3	10	6	1.020	200	0.855
3	15	6	1.177	200	0.987
3	10	8	1.235	200	1
3	10	4	0.701	250	0.681
3	10	6	0.967	250	0.811
3	10	8	1.221	250	0.901
3	15	8	1.298	250	0.958

Looking at the relationship between the number of containers handled per minute and the relative efficiency scores, as shown in figure 6, it can be observed that these two factors correlate strongly with the correlation coefficient of 0.87. This demonstrates that the more containers that can be handled per unit of time, the more efficient the system becomes.

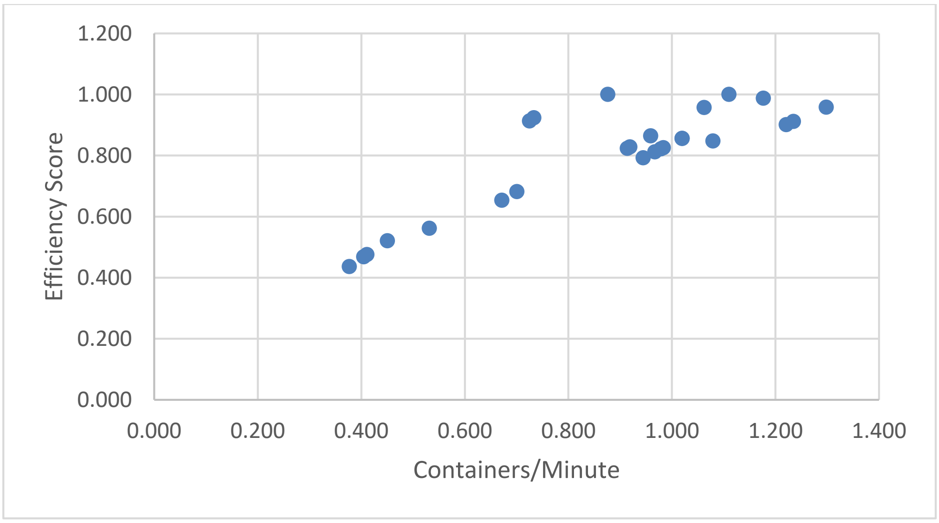

Figure 6: Correlation between efficiency scores and the number of containers handled per minute

5.4 Comparisons of computational times

Single-cycle operations and dual-cycle operations are both commonly used in practice. A single-cycle operation requires an unloading and loading process, which take place separately; while a dual-cycle operation enables simultaneous unloading and loading processes. Based on our previous studies of the unloading process (Luo et al., 2016), dual-cycle process (Luo & Wu, 2015) and the results obtained from this investigation, we were able to compare single-cycle and dual-cycle operations in terms of computational efficiency. Figure 7 shows that our proposed model in this study returns results most rapidly in all the cases for the loading process, compared with the models for the unloading process and dual-cycle process. For example, when handling 150 containers, the computation time for the loading process model is around 50 seconds, the computation time for the unloading process model is above 400 seconds, and the computation time for the dual-cycle process model is nearly 300 seconds.

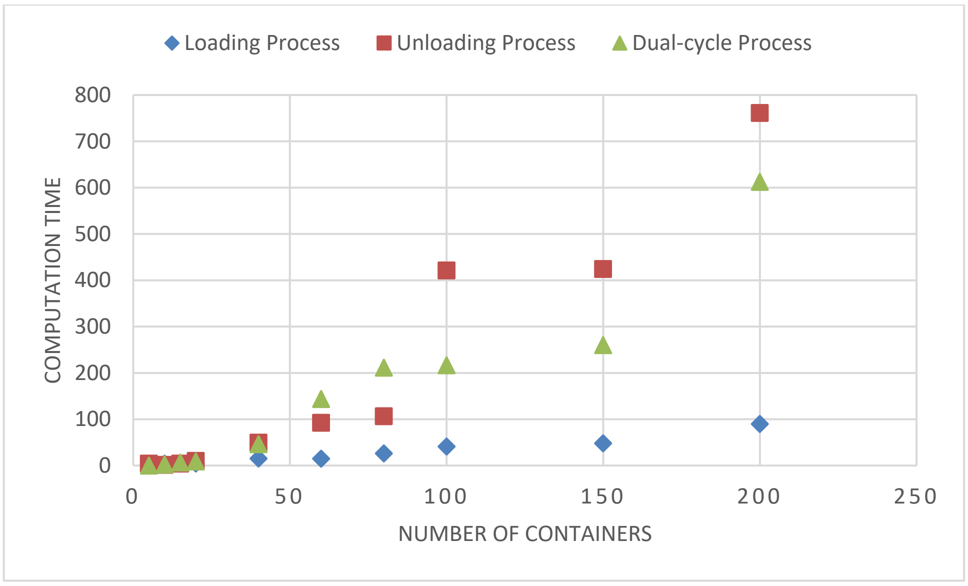

Figure 7: A comparison of computation times (in seconds) for single-cycle and dual-cycle operations

6 Conclusion

In this paper, we have proposed a novel integrated method for scheduling all types of container handling equipment – i.e. QCs, AGVs and YCs – in an automated container terminal. The loading process is considered in this problem, during which containers are handled by YCs first, delivered by AGVs from the yard to quayside, and then loaded onto a container ship by QCs. The aim is to minimise the loading element of the berthing time of the ship. The problem is formulated as a mixed-integer programming (MIP) model, which is known to be NP-hard. Small-sized problems are solved optimally by existing optimisation software (AIMMS 3.11); we also developed an efficient adaptive genetic algorithm to solve the considered problem in large sizes. The main contributions of our work are that we have provided a new integrated modelling approach and have designed an adaptive GA using matrix representation of chromosomes that are well tailored to our model.

We have carried out a series of computational experiments and observed that our proposed integrated modelling approach and the designed adaptive GA are very efficient. This model optimises the container terminal operations by minimising the berthing time by determining the most effective solutions for the schedules of AGVs, YCs and QCs. The findings of this study indicate that it takes more time to handle a greater number of containers, and cranes tend to play a more significant role in the terminal operations compared with AGVs, due to the improvements achieved when increasing the number of YCs in handling the containers. This finding can also be observed from the experiments on efficiency analysis, where the weights for QCs and YCs are higher than the weight for AGVs. In order to test the performance of the proposed GA, we have undertaken comparative experiments in small-sized problems, and have compared the results obtained by the adaptive GA with results obtained by the B&B in terms of OFVs and computation times. It was made clear that the GA proposed in this study can obtain near-optimal solutions for all the cases in small sizes, with the average difference in OFV being just 1.5%. Therefore, the GA was adopted for solving large-sized problems and obtaining the near-optimal results. In all the cases we ran in our experiments, our proposed GA worked effectively in involving approximate optimal solutions, and always demonstrated good convergence behaviour. We further compared the computational efficiency of single-cycle and dual-cycle process models, and found out that the model for the loading process as proposed in this study works the most efficiently.

There are several ways to extend this study, which can be considered as future avenues for research:

- *Development in new advanced heuristics*
  Despite the good performance of our proposed GA in this paper, there are novel ways to develop it further in order to obtain results closer to the optimal values and in a shorter time. This can be done by combining other techniques, such as simulated annealing, particle swarm optimisation, or ant colony optimisation into steps within the GA.
- *Extension into an uncertain environment*
  Uncertainties always exist in the real world, and they also exist in the container terminals. For example, the container handling times by cranes and vehicles are hardly likely to remain at a constant speed, due to the interferences of equipment and traffic conditions on the road. This can be modelled by stochastic programming or fuzzy optimisation. Other types of uncertainties, such as inaccuracy of container information could also be considered in the future.

References

Al-Dhaheri, N., Jebali, A. & Diabat, A. (2016). A simulation-based Genetic Algorithm approach for the quay crane scheduling under uncertainty. Simulation Modelling Practice and Theory, 66, 122-138.

Angeloudis, P. & Bell, M.G.H. (2010). An uncertainty-aware AGV assignment algorithm for automated container terminals. Transportation Research Part E: Logistics and Transportation Review, 46(3), 354-366.

Bazzazi, M., Safaei, N. & Javadian, N. (2009). A genetic algorithm to solve the storage space allocation problem in a container terminal. Computers & Industrial Engineering, 56(1), 44-52.

Briskorn, D., Drexl, A. & Hartmann, S. (2007). Inventory-based dispatching of automated guided vehicles on container terminals. In: K. Kim & H.O. Guenther, Container Terminals and Cargo Systems (pp. 195-214): Springer Berlin Heidelberg.

Cao, J.X., Lee, D.H., Chen, J.H. & Shi, Q. (2010). The integrated yard truck and yard crane scheduling problem: Benders' decomposition-based methods. Transportation Research Part E: Logistics and Transportation Review, 46(3), 344-353.

Chen, L., Bostel, N., Dejax, P., Cai, J. & Xi, L. (2007). A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. European Journal of Operational Research, 181(1), 40-58.

Chen, X., He, S., Zhang, Y., Tong, L.C., Shang, P. & Zhou, X. (2020). Yard crane and AGV scheduling in automated container terminal: A multi-robot task allocation framework. Transportation Research Part C: Emerging Technologies, 114, 241-271.

Choe, R., Kim, J. & Ryu, K.R. (2016). Online preference learning for adaptive dispatching of AGVs in an automated container terminal. Applied Soft Computing, 38, 647-660.

Daganzo, C.F. (1989). The crane scheduling problem. Transportation Research Part B: Methodological, 23(3), 159-175.

Dkhil, H., Yassine, A. & Chabchoub, H. (2017). Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import. Journal of the Operational Research Society, 1-23.

Evers, J.J.M. & Koppers, S.A.J. (1996). Automated guided vehicle traffic control at a container terminal. Transportation Research Part A: Policy and Practice, 30(1), 21-34.

Fu, Y.-M., Diabat, A. & Tsai, I.-T. (2014). A multi-vessel quay crane assignment and scheduling problem: Formulation and heuristic solution approach. Expert Systems with Applications, 41(15), 6959-6965.

Gharehgozli, A.H., Roy, D. & de Koster, R. (2016). Sea container terminals: New technologies and OR models. Maritime Economics & Logistics, 18(2), 103-140.

Han, X., Lu, Z. & Xi, L. (2010). A proactive approach for simultaneous berth and quay crane scheduling problem with stochastic arrival and handling time. European Journal of Operational Research, 207(3), 1327-1340.

Han, X., Wang, Q. & Huang, J. (2019). Scheduling cooperative twin automated stacking cranes in automated container terminals. Computers & Industrial Engineering, 128, 553-558.

He, J., Chang, D., Mi, W. & Yan, W. (2010). A hybrid parallel genetic algorithm for yard crane scheduling. Transportation Research Part E: Logistics and Transportation Review, 46(1), 136-155.

Huang, Y., Liang, C. & Yang, Y. (2009). The optimum route problem by genetic algorithm for loading/unloading of yard crane. Computers & Industrial Engineering, 56(3), 993-1001.

Kang, J., Ryu, K.R. & Kim, K.H. (2006). Deriving stacking strategies for export containers with uncertain weight information. Journal of Intelligent Manufacturing, 17(4), 399-410.

Kasm, O.A. & Diabat, A. (2019). The quay crane scheduling problem with non-crossing and safety clearance constraints: An exact solution approach. Computers & Operations Research.

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

Kim, K.H. & Kim, K.Y. (1999). Routing straddle carriers for the loading operation of containers using a beam search algorithm. Computers & Industrial Engineering, 36(1), 109-136.

Kim, K.H. & Park, Y.M. (2004). A crane scheduling method for port container terminals. European Journal of Operational Research, 156(3), 752-768.

Kozan, E. & Preston, P. (1999). Genetic algorithms to schedule container transfers at multimodal terminals. International Transactions in Operational Research, 6(3), 311-329.

Lau, H.Y.K. & Zhao, Y. (2008). Integrated scheduling of handling equipment at automated container terminals. International Journal of Production Economics, 112(2), 665-682.

Lee, D.H., Cao, J.X. & Shi, Q. (2008). Integrated quay crane and yard truck schedule for inbound containers. In: IEEE International Conference on Industrial Engineering and Engineering Management (pp. 1219-1223). Singapore: IEEE.

Lee, D.H., Wang, H.Q. & Miao, L. (2008). Quay crane scheduling with non-interference constraints in port container terminals. Transportation Research Part E: Logistics and Transportation Review, 44(1), 124-135.

Lee, L.H., Chew, E.P., Tan, K.C. & Wang, Y. (2010). Vehicle dispatching algorithms for container transshipment hubs. OR Spectrum, 32(3), 663-685.

Li, W., Wu, Y., Petering, M., Goh, M. & Souza, R. (2009). Discrete time model and algorithms for container yard crane scheduling. European Journal of Operational Research, 198(1), 165-172.

Li, W., Wu, Y., Petering, M.E., Goh, M. & Souza, R.d. (2009). Discrete time model and algorithms for container yard crane scheduling. European Journal of Operational Research, 198(1), 165-172.

Luo, J. & Wu, Y. (2015). Modelling of dual-cycle strategy for container storage and vehicle scheduling problems at automated container terminals. Transportation Research Part E: Logistics and Transportation Review, 79, 49-64.

Luo, J., Wu, Y. & Mendes, A.B. (2016). Modelling of integrated vehicle scheduling and container storage problems in unloading process at an automated container terminal. Computers & Industrial Engineering, 94, 32-44.

Ng, W. & Mak, K. (2005). Yard crane scheduling in port container terminals. Applied Mathematical Modelling, 29(3), 263-276.

Prasad, K., Ranjan, R., Sahoo, N. & Chaturvedi, A. (2005). Optimal reconfiguration of radial distribution systems using a fuzzy mutated genetic algorithm. IEEE Transactions on Power Delivery, 20(2), 1211-1213.

Reardon, B.J. (1998). Fuzzy logic versus niched Pareto multiobjective genetic algorithm optimization. Modelling and Simulation in Materials Science and Engineering, 6(6), 717.

Speer, U. & Fischer, K. (2016). Scheduling of Different Automated Yard Crane Systems at Container Terminals. Transportation Science, 51(1), 305-324.

Tavakkoli-Moghaddam, R., Makui, A., Salahi, S., Bazzazi, M. & Taheri, F. (2009). An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. Computers & Industrial Engineering, 56(1), 241-248.

Tavakkoli-Moghaddam, R. & Safaei, N. (2006). An evolutionary algorithm for a single-item resource-constrained aggregate production planning problem. In: IEEE Conference on Evolutionary Computation (CEC) (pp. 2851-2858). Vancouver, B.C., Canada: IEEE.

Zhang, C.Q., Wan, Y.W., Liu, J.Y. & Linn, R.J. (2002). Dynamic crane deployment in container storage yards. Transportation Research Part B-Methodological, 36(6), 537-555.

Zhong, M., Yang, Y., Dessouky, Y. & Postolache, O. (2020). Multi-AGV scheduling for conflict-free path planning in automated container terminals. Computers & Industrial Engineering, 142, 106371.

Appendix 1: An illustration example

The operating environment considered is as follows: the layout of the automated container terminal is shown in figure 2. There are two QCs working at the quayside for loading containers onto the ship. In the yard, three YCs are working to handle containers within three blocks and to load them onto the AGVs. Between the quayside and yard-side, three AGVs are travelling for the purpose of delivering containers. Because the destinations of containers are known and the AGVs travel at constant speed, the time AGV 1 spends on delivering one container from the yard to its destination QC is the same with the time taken by AGV 2. We give an example of nine containers to be loaded onto the ship. The destination QC and the yard storage block are given in table A1; for example, as in table A1, container 1 which is located in block 1 will be loaded by QC 1. Container 4 located in block 2 will be loaded onto the ship by QC 1. The AGV travel time from each block to each QC is given in table A2; for example, as in table A2, the AGV travel time from block 2 to QC 1 is 105 seconds, and the travel time from block 1 to QC 2 is 53 seconds.

Table A1: The destination QC and yard storage location information in an example of nine export containers

Container	Destination QC	Storage block
1	QC 1	Block 1
2	QC 1	Block 1
3	QC 1	Block 2
4	QC 1	Block 2
5	QC 1	Block 3
6	QC 2	Block 3
7	QC 2	Block 3
8	QC 2	Block 3
9	QC 2	Block 3

Table A2: AGV travel time between each block and each QC for the case of two QCs and three blocks (in seconds)

QCs	Block 1	Block 2	Block 3
QC 1	101	105	61
QC 2	53	49	50

According to tables A1 and A2, the values of t_i (AGV travel time for each container from the storage block, where it is located, to its destination QC) and s_{ij} (the empty-loaded travel time of AGVs between handling any two containers) can be calculated. Take container 4 as an example: t_4 will be the AGV travel time from block 2 (where container 4 is located) to QC 1 (container 4's destination QC), which is 105 seconds. If an AGV delivers container 3 and then goes to pick up container 5, then the empty loaded travel from the destination of container 3 (QC 1) to the yard location of container 5 (block 3) s_{35} is 61 seconds. These values are calculated in the above ways and shown in table A3 and table A4.

Table A3: The values of t_i-the AGV travel time for each container from storage block to QC

Containers	**1**	**2**	**3**	**4**	**5**
t_i **(sec)**	101	101	105	105	61
Containers	**6**	**7**	**8**	**9**	
t_i **(sec)**	50	50	50	50	

Table A4: The values of s_{ij}-every combination of the empty loaded travel time of AGV between any two containers (in seconds)

j / **i**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	N/A	101	105	105	61	61	61	61	61
2	101	N/A	105	105	61	61	61	61	61
3	101	101	N/A	105	61	61	61	61	61
4	101	101	105	N/A	61	61	61	61	61
5	101	101	105	105	N/A	61	61	61	61
6	53	53	49	49	50	N/A	50	50	50
7	53	53	49	49	50	50	N/A	50	50
8	53	53	49	49	50	50	50	N/A	50
9	53	53	49	49	50	50	50	50	N/A

The handling time of each container by a QC is the time duration from picking up the container from an AGV until it is placed on its location on the ship. The handling time of each container by the YC is determined by the container's location in the yard; it is the time duration between retrieving the container in the storage yard and placing it onto the AGV at the working points in front of the block. Assuming containers are evenly distributed both on the ship and in the yard, the handle time of QC for each container - h_i and handle time of YC for each container - p_i are generated from uniform distribution U(30, 180)s and U(60, 140)s, respectively. An example of these values are shown in table A5.

Table A5: The values of QC handling times h_i and YC serving times p_i (in seconds)

Containers	h_i	p_i
1	80	104
2	73	132
3	128	107
4	87	84
5	74	122
6	133	98
7	175	110
8	132	138
9	180	100

The handling time of each container by a YC is defined as the time from each container's yard location to the transfer point in front of the block. As the YC travels at constant speed, there is no difference in the speed of loaded move and empty-loaded move. Let the YC travel time between the transfer points of any two adjacent blocks be 40 seconds. There are two conditions when calculating the values of a YC's empty-loaded travel time w_{ij} for any two successive containers handled by the same YC:

(1) Containers in the same block. For any two consecutive containers located within the same block and to be handled by the same YC, the empty-loaded travel time from dropping off the previous container onto an AGV at the transfer point to the target container, is the same as the process time of the target container because of the same travel distance. For example, consider that container 6 and container 7 will be performed by the same YC. After the YC releases container 6 onto an AGV at the transfer point of block 3, this YC moves to the location of container 6 without carrying containers, i.e. empty-loaded. Thus this empty-loaded travel time is the process time of container 7, which, as shown in table A5, which is 110 seconds.

(2) Containers in different blocks. For calculating the empty-loaded travel times between any two consecutive containers in different blocks and to be handled by the same YC, additional YC travel time between blocks should be to the process time. Let the travel time of YCs between the transfer points at block 1 and block 2 be 40 seconds. For example, if container 2 and container 3 are processed by the same YC, then after an AGV collects container 2 at the transfer point of block 1, YC will take 40 seconds to move to the transfer point at block 2; then it goes to pick up container 3. Thus the empty travel time is 40+107=147 seconds, where 107 is the process time of container 3, as in table A5.

Therefore, the values of w_{ij} could be calculated followed by the above discussion: we list the results in table A6. In the same way, the values of parameters can be calculated accordingly for different sized problems.

Table A6: The empty-loaded travel time between any two successive containers by the same YC

j i	1	2	3	4	5	6	7	8	9
1	N/A	132	147	164	202	178	290	218	180
2	104	N/A	147	164	202	178	290	218	180
3	144	172	N/A	124	162	138	250	178	140
4	144	172	107	N/A	162	138	250	178	140
5	184	212	147	164	N/A	98	110	138	100
6	184	212	147	164	122	N/A	110	138	100
7	184	212	147	164	122	98	N/A	138	100
8	184	212	147	164	122	98	110	N/A	100
9	184	212	147	164	122	98	110	138	N/A

The solver takes 1393.31 seconds to solve the model and the obtained optimal berth time is 648 seconds. The values of decision variables with respect to the schedules of AGVs, YCs and QCs are given in table A7, A8 and A9 respectively. For example, from table A7, we can see that AGV1 will deliver container 3 first, then container 1, followed by container 5; from table A8, it can be observed that YC2 will handle container 2 first, then container 1, followed by container 4; from table A9, it can be found that QC 2 will handle container 8, container 6, container 9 and container 7 in sequence.

Table A7: The values of the decision variables x_{ik}^{m}

Sequences	AGV1	AGV2	AGV3
1st	$x_{31}^{1}=1$ container 3	$x_{22}^{1}=1$ container 2	$x_{83}^{1}=1$ container 8
2nd	$x_{11}^{2}=1$ container 1	$x_{92}^{2}=1$ container 9	$x_{73}^{2}=1$ container 7
3rd	$x_{51}^{3}=1$ container 5	$x_{42}^{3}=1$ container 4	$x_{63}^{3}=1$ container 6

Table A8: The values of the decision variables y_{iw}^{n}

Sequences	YC1	YC2	YC3
1st	$y_{81}^{1}=1$ container 8	$y_{22}^{1}=1$ container 2	$y_{33}^{1}=1$ container 3
2nd	$y_{71}^{2}=1$ container 7	$y_{12}^{2}=1$ container 1	$y_{93}^{2}=1$ container 9
3rd	$y_{51}^{3}=1$ container 5	$y_{42}^{3}=1$ container 4	$y_{63}^{3}=1$ container 6

Table A9: The values of the decision variables z_{ij}

Sequences	QC1	QC2
1st	Container 2 $z_{23}=1$	Container 8 $z_{86}=1$
2nd	Container 3 $z_{31}=1$	Container 6 $z_{69}=1$
3rd	Container 1 $z_{15}=1$	Container 9 $z_{97}=1$
4th	Container 5 $z_{54}=1$	Container 7
5th	Container 4	

Appendix 2: model for calculating the efficiency score

Objective: maximise the efficiency score for the base case highlighted in table 3

$$efficiency = \frac{container\ per\ minute * u}{QC * v1 + AGV * v2 + YC * v3}$$

Subject to

$$\frac{container\ per\ minute * u}{QC * v1 + AGV * v2 + YC * v3} \leq 1 \text{ for each case as listed in table 3}$$

Weights ≥ 0

Applying to our cases in table 3 to the above model, we have:

Maximise: efficiency=$\frac{0.967*u}{3*v1+10*v2+6*v3}$

Subject to

$$\frac{0.377 * u}{3 * v1 + 3 * v2 + 2 * v3} \leq 1$$

$$\frac{0.411 * u}{3 * v1 + 4 * v2 + 2 * v3} \leq 1$$

$$\frac{0.405 * u}{3 * v1 + 5 * v2 + 2 * v3} \leq 1$$

$$\dots$$

$$\frac{1.298 * u}{3 * v1 + 15 * v2 + 8 * v3} \leq 1$$

$$u, v1, v2, v3 \geq 0$$

Converting this model into a linear model, we have

Maximise: 0.967*u

Subject to

$$3 * v1 + 10 * v2 + 6 * v3 = 1$$

$$0.377 * u - 3 * v1 - 3 * v2 - 2 * v3 \leq 0$$

$$0.411 * u - 3 * v1 - 4 * v2 - 2 * v3 \leq 0$$

$$0.405 * u - 3 * v1 - 5 * v2 - 2 * v3 \leq 0$$

$$\dots$$

$$1.298 * u - 3 * v1 - 15 * v2 - 8 * v3 \leq 0$$

$$u, v1, v2, v3 \geq 0$$

This model is solved by Excel Solver and the optimal efficiency for the base case is 81.1%, and the optimal solutions for the weights obtained are $u = 0.839, v1 = 0.196, v2 = 0, v3 = 0.069$.