Ann Oper Res (2016) 243:55–70
DOI 10.1007/s10479-014-1645-z

On double cycling for container port productivity improvement

Dusan Ku · **Tiru S. Arthanari**

Published online: 7 June 2014

Abstract How quay cranes (QC) are scheduled is vital to the productivity of seaside container port operations. Double cycling concept is an operation strategy of loading the containers into ships as they are unloaded, thus improving the efficiency of a QC as well as the container port. Goodchild and Daganzo (Transp Sci 40(4):473–483, 2006) first described QC double cycling problem and solved the problem after formulating it into a two machine flow shop problem. Song (Port Technol Int 36:50–52, 2007) studied the formula to determine the optimal starting sequence for double cycling while reflecting on the practical issue of QC working direction. The above studies focused on a single QC double cycling and their empirical trials showed the double cycling could improve the productivity of each QC approximately by between 10 and 20 %. In Zhang and Kim (Comput Ind Eng 56(3):979–992, 2009), a multiple QC double cycling model was first suggested by formulating a mixed integer programming model to maximise the number of double cycles between multiple QCs. In the present paper we point out a flaw with the existing multiple QC double cycling model that lets cycles that are not implementable. In addition, the paper discusses the need for imposing constraints arising from real world requirements to the formulations aiming at double cycling.

Keywords Container terminal · Quay crane scheduling · Double cycling

1 Introduction

Ports are the gateways in global distribution networks and play a vital role in completing the global supply chains. Considering the growing level of service requirements that shipping

D. Ku · T. S. Arthanari (✉)
Department of Information Systems and Operations Management, Business School,
University of Auckland, Auckland, New Zealand
e-mail: t.arthanari@auckland.ac.nz

D. Ku
e-mail: d.ku@auckland.ac.nz

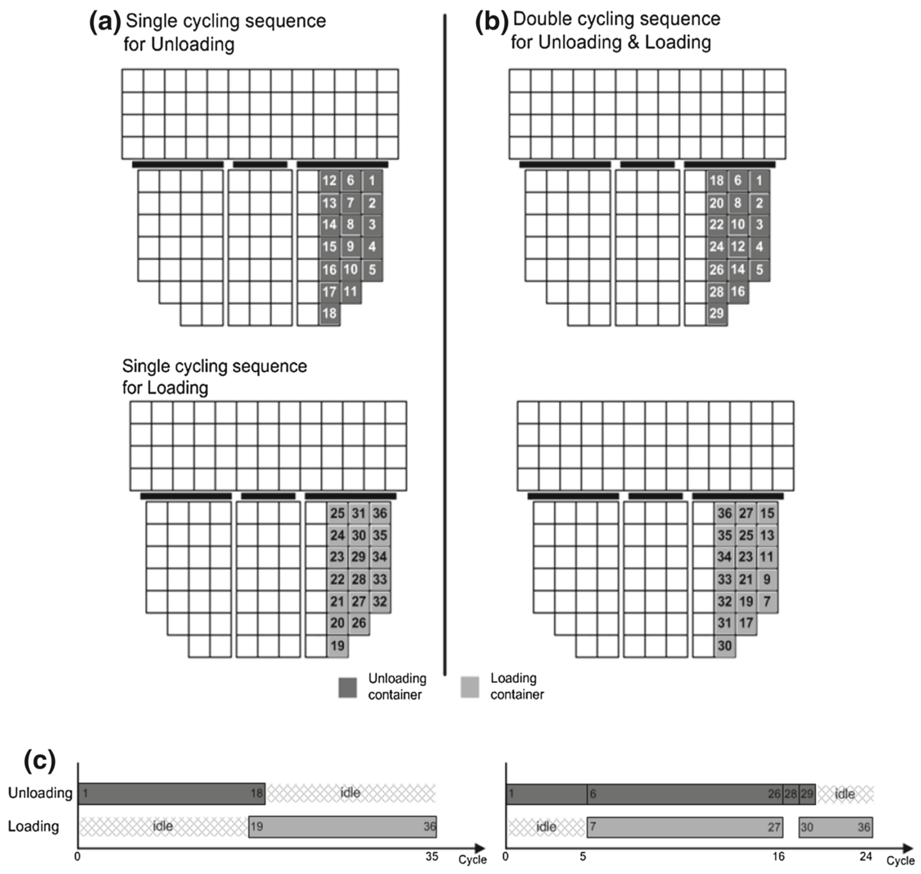

(a) Single cycling sequence for Unloading
Single cycling sequence for Loading
(b) Double cycling sequence for Unloading & Loading
Unloading container
Loading container
(c)
Unloading
Loading
idle
Cycle

Fig. 1 Comparison of sequence order between single and double cycling

lines demand to the terminal operators, the efficiency of terminal operation is becoming a matter of business survival. Although new facility or a new decision support system may increase the terminal productivity, the terminal operators are often reluctant to take this option due to the high investment cost associated with it. In the present paper, we aim to discuss a more viable option to boost the productivity of the vessel operation without incurring a significant amount of investment. In contrast to other measures, double cycling strategy in seaside operation is regarded as a low-cost method to boost the vessel productivity; it does not require new technology or infrastructure except that the vessel planning tool that terminal planners use should be able to cater for double cycling sequencing. Although double cycling will not solve the capacity problem in the long run, it is more quickly implementable than other solutions, and can be used to complement other strategies.

Double (or dual) cycling is a seaside operation strategy by which idle moves of QCs or horizontal transport vehicles, represented by yard tractors (YT) or straddle carriers (SC), for shore-to-yard transport are converted into productive ones. With single cycling, as opposed to double cycling, QCs carry out only either unloading or loading activity in each cycle and horizontal transport vehicles transport containers for only one direction in each cycle. From the QC perspective, a cycle is a crane spreader's round-trip between ship and shore. As illustrated in the Gantt chart of Fig. 1, for example, the double cycling operation requires

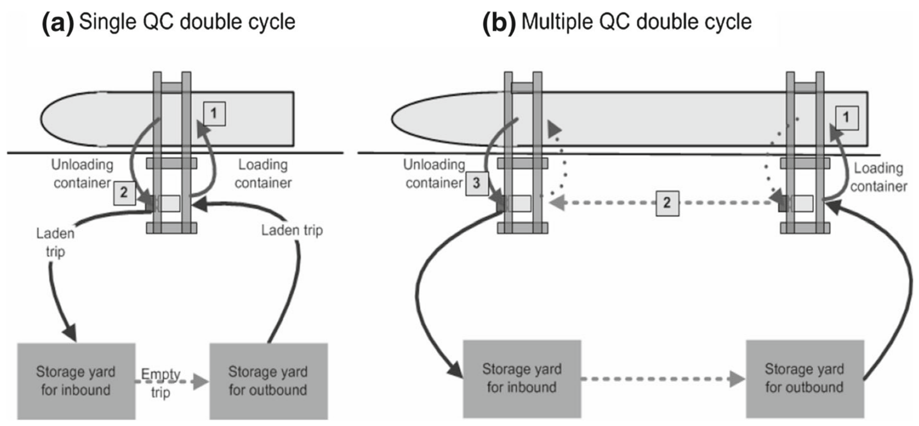


Fig. 2 Single QC double cycling vs Multiple QC double cycling

a total of 24 cycles whereas the single cycling operation requires 35 cycles to complete the same amount of tasks.

There are two types of models with respect to the scope of double cycling. One is single QC double cycling as described in the above example and the other is multiple QC double cycling. In the single QC double cycling, the operation is carried out by a single QC within one hatch. In the multiple QC double cycling, it is carried out between two or more QCs located in separate hatches, first loading by one QC and then unloading by another QC, and the transport of the respective container(s) is handled by the same landside vehicle. The flows of equipment between the two models are illustrated in Fig. 2.

The advantage of single QC double cycling is the reduced number of QC cycles. According to Goodchild and Daganzo (2006), a trial run at the Port of Tacoma, US, showed that the adjusted average time for a single cycle was 1 min 45 s, and for a double cycle it was 2 min 50 s. Thus, double cycling saved 40 s per pair of double-cycled containers and the operation time with the parameters input from the trial run was estimated to be reduced by around 10 %. The advantage of multiple QC double cycling is twofold: First, it reduces the required number of horizontal transport cycles and its direct benefit is less congestion at yard traffic as well as less consumption of fuel. Second, as in single QC double cycling it also implies the potentially reduced number of QC cycles wherever applicable. However, the empirical data of the multiple QC double cycling is yet to come out. For one reason, there is a lack of research into operational methods. For another, it requires a substantial change in work procedure, supported by the enhancement of the terminal operating system (TOS) and much training towards manpower.

Double cycling is not the method that is always applicable during the operation. There are some preconditions to be satisfied. First, the QC should be equipped with a twin-lift spreader. If it is not, the bays dedicated only to either 20- or 40-foot containers are appropriate bays for double cycling. If a bay (i.e. hatch) is mixed with 20- and 40-foot containers, 20-foot containers are typically stowed below 40-foot containers for safety reasons.[1] Engaging double

[1] Inside a ship, containers stacked in the vertical direction are linked together by twist-locks (or cones) on four corners. The four corners of a 40-foot unit on top of two 20-foot units can be adequately linked by twist-locks, but the opposite can fix only two corners of each 20-foot container on either side. This may cause containers to digress while the ship is sailing.

cycling under this condition may easily lead to either an accidental hit by a cell guide in the hold or an excessive QC moving time.

Double cycling strategy can be formulated within the context of the quay crane scheduling problem (**QCSP**). The purpose of the **QCSP** is to find a complete schedule for the QCs working on a vessel with its objective being typically the minimisation of the vessel turnaround time, the decisions including assigning a sequence of tasks to each QC, and the constraints including physical limitations as well as administrative limitations: physical limitations include precedence relationship among tasks, safety distance between QCs, and non-crossing constraint against another QC; administrative limitations include allowed operation time such as vessel arrival time, the deadline for departure time and break-time between shift changes of stevedore.

In this paper we first overview the literature relating to double cycling concepts in container terminal operation. The main contribution of the paper is to highlight the flaws in a recent article formulating the multiple QC double cycle problem. In addition, this paper discusses the need for imposing constraints arising from real world requirements to the formulations aiming at double cycling.

The rest of this paper is organised as follows: Sect. 2 reviews relevant literature on the quay crane scheduling problem (QCSP) and the double cycling quay crane scheduling problem (**DCQCSP**), which is a special case of the **QCSP**. Section 3 describes the **DCQCSP** from the literature. Section 4 notes the issues on the **DCQCSP** from the latest research trends, and discusses considerations for the double cycling to be implementable. Finally, summary of this paper and the future research direction are provided in Sect. 5.

2 Literature review

Each subsystem in container terminal has its unique operational characteristics depending on its configuration, so managing such a system optimally is very complicated. For this reason, much research tackles each subsystem separately and tries to restrict interactions between subsystems although some research tries to integrate the subsystems across terminals (Avriel et al. 1998) or within a terminal. We first introduce the quay crane scheduling problem (**QCSP**) and the relevant literature on this problem.

2.1 Quay crane scheduling problem (QCSP)

Determining the granularity of a task is the first step in modelling the **QCSP** in general. Tasks can be defined on the basis of bay areas, complete bays, groups of containers in the same bay or individual containers (Bierwirth and Meisel 2010). Precedence relationships can be given to the tasks where, for example, unloading precedes loading for the same ship spaces, unloading from the upper slots precedes unloading from the lower slots, and loading on the lower slots precedes loading on the upper slots. By its definition, a task is assumed to be processed without preemption by a QC while a QC can process at most one task at a time. A QC schedule solved by this problem contains a sequence of tasks assigned to each QC. Therefore, assigning a whole ship bay or a bay area, i.e. a series of consecutive bays, to a task is of little practicality because the chunk of a bay or a bay area is too big for a task and it is often in real situations in a ship bay, the workload is split between QCs. On the other hand, applying the finest possible granularity, i.e. a container, to a task makes the problem size significantly large to the point where very few algorithms are available to solve the problem within a reasonable time.

The approach of Kim and Park (2004) is regarded as a breakthrough work for the problem formulation with respect to assuming multiple tasks in a ship bay. They define a task as an unloading or a loading operation for a cluster, which consists of a group of homogeneous containers with, for example, the same destination port. In their mixed integer programming (MIP) model, the authors propose a Branch-and-Bound method and a Greedy Randomized Adaptive Search Procedure (GRASP). Coined by Feo and Resende (1995), GRASP is a metaheuristic algorithm commonly applied to a variety of combinatorial problems. It consists of two phases: the construction phase and the improvement phase. In the construction phase, an element of a solution is added iteratively to the solution set by using a random number and a greedy function. In the improvement phase, the solution is iteratively improved through a local search. The Branch-and-Bound method provides an exact solution very well for instances with two cranes and up to 10–15 tasks while it fails on instances with three cranes and 20–25 tasks. Although Branch-and-Bound method outperforms GRASP in terms of the solution quality, it is inadequate to solve large instances and therefore GRASP is suggested to solve the model.

In subsequent studies following the approaches taken by Kim and Park (2004), Moccia et al. (2006) note the model in Kim and Park (2004) do not avoid crane interference in every case. It only avoids collisions between QCs working on tasks in the same bay by forcing non-simultaneity constraint between two tasks in the same bay, but fails to take the travel time of QCs into account, making the model unable to consider satefy margin of QCs when the tasks are in the neighbouring bays. Therefore, after revising the model of Kim and Park (2004) by incorporating travel times for QCs, they develop a Branch-and-Cut algorithm which finds improved solutions both in terms of solution quality and of computation time for the benchmark suite of problems from Kim and Park (2004). However, the fast growth of solution time as a function of the size of instances is still the major drawback of the Branch-and-Cut method although it is faster than the Branch-and-Bound method. Sammarra et al. (2007) present a Tabu Search heuristic, tackling the **QCSP** as a vehicle routing problem. Developed by Glover (1989), Tabu Search is basically a local search heuristic in which it moves iteratively from a candidate solution to a better solution in a subset of the neighbourhood by applying local changes. It may perform moves that deteriorate the current solution in order to avoid being trapped in a local optimum, with the hope of eventually finding a better one. In their model, a neighbourhood is defined by resequencing the tasks of a crane and by swapping tasks between cranes. Compared to the Branch-and-Cut method of Moccia et al. (2006), it provides a reasonable compromise between computational burden and solution quality, yielding less computation time but slightly weaker solution quality. Bierwirth and Meisel (2009) note that even the model revised by Sammarra et al. (2007) may yield solutions where interference between QCs is not detected. They propose a Branch-and-Bound algorithm for searching a reduced solution space of unidirectional schedules. Although the unidirectional schedule approach can fail in finding the optimal solution, it might be a good strategy for solving the **QCSP** with container groups heuristically when satisfying all requirements of the **QCSP** including the issue of crane interference. Still, this algorithm clearly outperforms the heuristics proposed in all aforementioned literature in terms of solution quality and of computation time.

2.2 Double cycling quay crane scheduling problem (DCQCSP)

The **DCQCSP** is a special case of the **QCSP** in the sense that QCs are operated by double cycling method where applicable.

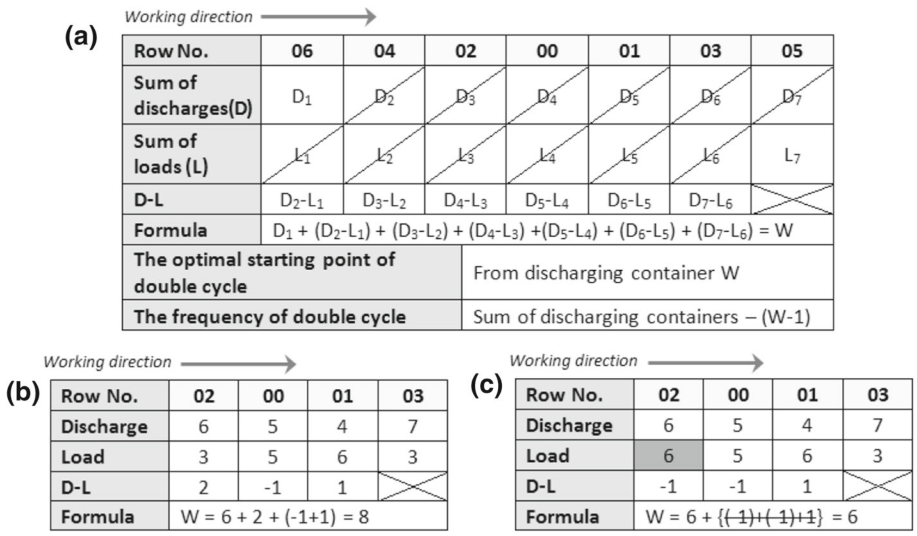
As for the literature on the **DCQCSP**, Goodchild and Daganzo are the first group of authors that formulate the double cycling problem mathematically (Goodchild 2005; Goodchild and Daganzo 2004, 2006). The model in their studies considers a single QC double cycling where a ship arrives in port with a set of containers on board to be unloaded and a loading plan for containers to be loaded. Given are u_c and l_c, the number of containers to be unloaded and loaded, respectively, in each stack labelled c. A key feature of double cycling is that the order in which the stacks within each ship bay are handled; this is represented as permutation functions, one for an unloading permutation and the other for a loading permutation. If we consider the makespan as the time that takes to finish the unloading and loading activities in the assumed ship bay, the proxy for the makespan is the number of cycles required to unload and load containers. Then, the problem of determining the optimal permutation to reduce the makespan can be formulated as a two-machine flow shop scheduling problem where one job corresponds to one stack and each job has two operations: an unloading operation that must be completed first, and a subsequent loading operation. The exact solution for this class of problem is solved via Johnson's rule (Johnson 1954). First, list the tasks of each stack by unloading and loading with its number of containers to be handled, then according to u_c and l_c, select the task with the shortest cycles. If the task is for the first work centre (unloading operation), schedule the task first; if that task is for the second work centre (loading operation), schedule the last task and eliminate the selected task from further consideration. Then, repeat the above steps until all tasks have been scheduled. The output of this procedure is the permutation of handling the stacks to minimise the makespan via single QC double cycling method.

A double cycling's stevedoring direction can be divided into *from starboard to portside*, *from portside to starboard* or *zigzag type*. Song (2007) notes that the third type may maximise the frequency of a double cycling as formulated by Goodchild and Daganzo, but it is difficult for stevedores to understand and follow such work flow. Therefore, it is a practical approach that the stevedoring direction goes in only one way, either *starboard to portside* or *portside to starboard*. Given the constraint of only one direction for double cycling, Song (2007) also notes that the double cycling operation may frequently encounter blocking delays[2] were it not optimally considered how to decide the starting sequence for the double cycling. The point here is once a double cycling begins in a hatch, it is desirable to continue the double cycling without having blocking delays until the unloading tasks are completed, so his study focuses on finding a general rule on the optimal starting point of double cycling.

Figure 3a shows the formula for calculating the optimal starting point of double cycling. In this formula, a careful consideration should be given to the case where the value of the element $(D_i - L_{i-1})$ becomes negative. If it becomes negative, it should be separately treated by being added to the next elements repeatedly until the sum of the added elements becomes nonnegative. If the sum of the added elements is still negative even after the last element has been added, then those elements should be abandoned from the calculation. In Fig. 3b, for example, $(D_3 - L_2)$ became negative by -1, so it has been added to the next element, $(D_4 - L_3)$, and the sum becomes zero, thus being admitted into the calculation. In Fig. 3c, however, the sum of $[(D_2 - L_1) + (D_3 - L_2) + (D_4 - L_3)]$ is still negative, and therefore they are cancelled out from the calculation. The end result is the amount of single unloading cycles after which the double cycling should start given the working direction.

In contrast to a single QC double cycling model, Zhang and Kim (2009) extend the model to the case of multiple hatches. They formulate the problem as an MIP model and decompose the

[2] Loading can begin as soon as at least one stack has been unloaded or is empty. Loading can proceed in any unloaded or empty stack until loading operations are complete. If there is no column available, the loading operations must wait. This is called blocking delay (Goodchild and Daganzo 2004).



(a) Working direction →

Row No.	06	04	02	00	01	03	05
Sum of discharges(D)	D_1	$\cancel{D_2}$	$\cancel{D_3}$	$\cancel{D_4}$	$\cancel{D_5}$	$\cancel{D_6}$	$\cancel{D_7}$
Sum of loads (L)	$\cancel{L_1}$	$\cancel{L_2}$	$\cancel{L_3}$	$\cancel{L_4}$	$\cancel{L_5}$	$\cancel{L_6}$	L_7
D-L	D_2-L_1	D_3-L_2	D_4-L_3	D_5-L_4	D_6-L_5	D_7-L_6	☒
Formula	$D_1 + (D_2\text{-}L_1) + (D_3\text{-}L_2) + (D_4\text{-}L_3) + (D_5\text{-}L_4) + (D_6\text{-}L_5) + (D_7\text{-}L_6) = W$						
The optimal starting point of double cycle	From discharging container W						
The frequency of double cycle	Sum of discharging containers – (W-1)						

(b) Working direction →

Row No.	02	00	01	03
Discharge	6	5	4	7
Load	3	5	6	3
D-L	2	-1	1	☒
Formula	W = 6 + 2 + (-1+1) = 8			

(c) Working direction →

Row No.	02	00	01	03
Discharge	6	5	4	7
Load	6	5	6	3
D-L	-1	-1	1	☒
Formula	W = 6 + {~~(-1)+(-1)+1~~} = 6			

Fig. 3 Double Cycle Formula: from portside to starboard. *Source*: Song (2007)

whole QC scheduling into two parts: intra-stage optimisation (sequencing all the stacks in one hatch) and inter-stage optimisation (sequencing all the hatches). For intra-stage optimisation, they apply Johnson's rule similar to that of Goodchild and Daganzo. As an improvement to this method, Wang et al. (2011) suggest a gap-shifting strategy in their two-stage composite heuristic, in which stacks in a hatch are scheduled by the Johnson rule integrating with the gap-shifting strategy, and the Johnson rule is reconstructed for the inter-hatch sequencing. Note that Song (2007) has already suggested the formula in finding the optimal starting sequence with a gap-shifting concept, and the result of that formula is conceptually the same as the gap-shifting strategy of Wang et al. (2011) except that the stevedoring direction in Song (2007)'s model is only one direction due to practical reasoning. In Wang et al. (2011)'s model shown in Fig. 4, the Gantt charts can be partitioned into three sections.

- SU : the interval with only unloading operations
- SL : the interval with only loading operations
- DUAL : the double cycling interval
- SLOT : the gathered gaps by left shifting

In Fig. 4b, DUAL can be shortened by shifting all gaps before the first operation on the loading QC without changing the makespan of the hatch. Correspondingly the length of SU is increased by the length of SLOT. Therefore, there is a possibility that a better makespan may be yielded when reconnecting hatch sequences with the prolonged SUs.

For inter-stage optimisation, Zhang and Kim (2009) suggest heuristics based on gap-based neighbourhood local search. The experimental results show that the proposed method outperforms the real schedules constructed by human planners. However, their model fails to recognise the cranes' interference constraints and the non-crossing constraints between cranes, thus making it practically contradictory. He et al. (2011) include the cranes' interference constraints in their study on the integrated large-capacity quay crane scheduling but still have the shortcomings of ignoring the non-crossing constraints between the cranes.

Gadeyne and Verhamme (2011) propose two models in the **QCSP**, one for the **QCSP** in general and the other for the **DCQCSP**. In their **DCQCSP** model, they incorporate some

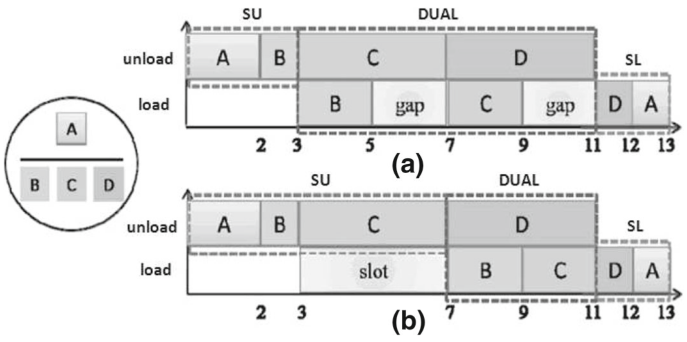


Fig. 4 Gap-shifting strategy for intra-stage optimisation. *Source*: Wang et al. (2011)

Table 1 Time units per QC action

Action	Average	Approximation	Time units
Spreader setup	28 s	30 s	1 (per action)
Single cycle	1 min 45 s	1 min 30 s	3 (per action)
Empty movement	–	30 s	1 (per action)
Double cycle	2 min 50 s	3 min	6 (per action)
QC displacement	–	(Fixed part) 4 min 30 s	9 (per action)
	1 min 00 s	(Variable part) 30 s	1 (per 40ft bay)

Source: Gadeyne and Verhamme (2011)

practical constraints such as sequence-dependent setup times and interference constraints. Also, the discretised time steps (e.g. 30 s in one time unit) are added in their model to visualise and verify the correctness of each constraint, at every time unit. The decision on one time unit is made with approximation from the literature (Goodchild and Daganzo 2006) and their own measurements at a reference port (Port of Zeebrugge), and is summarised in Table 1. Each QC can have only one action status among the five statuses. The application of the time segment concept for measuring the makespan is a more practical approach than just the number of cycles as assumed in the previous literature. Consequently, however, this model is quite detailed, and has too many variables. Therefore, this approach is only applicable to small problems.

Except for the studies mentioned above, it is not easy to find notable literature as to double cycling. As Goodchild opened the floodgate for modelling double cycling and Zhang and Kim extended the scope of the model to multiple hatches, more thorough research is expected to come out in future. As it is expected that much of the future research bases their model on the scenario of multiple hatches following Zhang and Kim's model, it is important to note what the critical shortcomings are in this model, thus motivating the future researchers to tackle these issues first when formulating their models.

3 Multiple QC double cycling

3.1 Problem description

The ultimate goal of a vessel operation is to minimise the makespan of all QC activities on a ship. When attempting to apply multiple QC double cycling method, the problem consists of at least two independent QCs working on different hatches, and where at some moments one

QC loads a container (or twin containers) onto the vessel, another QC unloads a container (or twin containers) from the vessel, and both QC activities are interacted with the same landside vehicle. Regarding the makespan, the following assumptions are made:

1. the operation time for a QC to complete all the discharging and the loading operations in a ship-bay can be expressed as the number of cycles;
2. the operation of QCs is the bottleneck during the vessel operation, so the throughput rate of QCs determines the throughput rate of the integrated handling system consisting QCs, landside transporters and yard cranes.

Regarding landside transport, we assume a closed queueing system in which landside vehicles are the customers shuttling between two types of server stations, i.e. the QC type and the storage yard type. In reality, the QC productivity is significantly affected by the productivity of the landside processing. However, the appropriate number of landside vehicles assigned to a QC or a pool of QCs varies depending on the environment of each terminal. For the simplification of this model, the following assumptions are made:

1. there are enough number of landside vehicles so that the queues at the QC servers are never idle;
2. there are infinitely many servers at the storage yard so that landside vehicles do not have to wait for services at yard.

Regarding hatch covers by which most modern container ships separate the deck from the hold of a ship bay, the following assumptions are made within the range of a hatch:

1. all the unloading tasks under the deck (in the hold) cannot begin until all the unloading tasks on the deck are completed;
2. all the loading activities on the deck cannot begin until all the loading tasks in the hold are completed. These assumptions lead to an additional implicit assumption that if single QC double cycling is applicable to reduce the makespan it will be conducted only within the tasks in the hold.

We revisit Zhang and Kim (2009) to describe a mathematical formulation of this problem. We are given a set of hatches, H, a set of stacks for which unloading operation is planned, S^u, and a set of stacks for which loading operation is planned, S^l. A set $\overline{S}_h^u$ is defined as the set of stacks for which unloading operation is planned in the hold of hatch h, $\overline{S}_h^l$ as the set of stacks for which loading operation is planned in the hold of hatch h, $\underline{S}_h^u$ as the set of stacks for which unloading operation is planned on the deck of hatch h, and $\underline{S}_h^l$ as the set of stacks for which loading operation is planned on the deck of hatch h. Let n_i^u denote the number of containers to unload from stack i, and n_i^l the number of containers to load into stack i. The notations for the completion times are as follows:

- C_i^u: completion time of unloading all containers from stack i
- C_i^l: completion time of loading all containers into stack i
- $\overline{C}_h^u$: completion time of unloading all containers from the hold of hatch h
- $\overline{C}_h^l$: completion time of loading all containers into the hold of hatch h
- $\underline{C}_h^u$: completion time of unloading all containers from the deck of hatch h
- $\underline{C}_h^l$: completion time of loading all containers into the deck of hatch h

The objective is to minimise w, which represents the number of cycles necessary to complete all the QC tasks. It can be constrained by the following inequalities:

$$w \geq \overline{C}_h^u \quad \forall h \in H \tag{1}$$
$$w \geq \overline{C}_h^l \quad \forall h \in H \tag{2}$$
$$w \geq \underline{C}_h^u \quad \forall h \in H \tag{3}$$
$$w \geq \underline{C}_h^l \quad \forall h \in H \tag{4}$$

Constraints regarding hatch covers and the precedence relationship between the loading/unloading activities in the same stack are represented as follows: Constraint (5) ensures that the unloading activities in a hold cannot begin until all the unloading activities on the deck of the same hatch are completed; Constraint (6) ensures that the loading activities in a stack cannot begin until all the unloading activities in the same stack are completed, while constraint (7) ensures that in each hatch the loading activities on deck cannot begin until all the loading activities in hold of the same hatch are completed.

$$C_i^u \geq \underline{C}_h^u + n_i^u \quad \forall i \in \overline{S}_h^u,\ \forall h \in H \tag{5}$$
$$C_i^l \geq C_i^u + n_i^l \quad \forall i \in S^u \cap S^l \tag{6}$$
$$C_i^l \geq \overline{C}_h^l + n_i^l \quad \forall i \in \underline{S}_h^l,\ \forall h \in H \tag{7}$$

X_{ij} and Y_{ij} denote binary decision variables. X_{ij} is equal to 1 iff unloading for stack j is performed immediately after unloading for stack i, and Y_{ij} is equal to 1 iff loading for stack j is performed immediately after loading for stack i. This definition can be represented as the following constraints (8) and (9) where M represents a sufficiently large number:

$$C_j^u - C_i^u + M\left(1 - X_{ij}\right) \geq n_j^u \quad \forall i, j \in S^u \tag{8}$$
$$C_j^l - C_i^l + M\left(1 - Y_{ij}\right) \geq n_j^l \quad \forall i, j \in S^l \tag{9}$$

Apparently, there is no mention of crane interference constraints in this mathematical formulation, let alone the indices for cranes. This has led us to test the validity of this model and our findings will be noted in the next section.

3.2 Two-stage hybrid heuristic

In the hybrid heuristic approach taken by Zhang and Kim (2009) for the **DCQCSP** in multiple hatches, the algorithm is decomposed into two parts: inter-stage sequencing and intra-stage sequencing. If we assume that the intra-stage sequencing has been optimised by the combined contributions from Goodchild (2005), Goodchild and Daganzo (2004), Goodchild and Daganzo (2006), Song (2007), Wang et al. (2011), a simple example of the inter-stage sequencing problem can be illustrated as in Fig. 5.

For a single QC double cycling to be started in a ship hold, at least one stack, usually the first stack to load, should be free of any unloading task. This is evident the first loading stack C in Fig. 5b begins to be filled in, at which point double cycling also begins, after the unloading task in that stack is completed, and the same applies to the first loading stack I in Fig. 5c. This illustration shows that the different hatch working sequence may yield different number of cycles as its makespan, so finding the best hatch working sequence of each QC for minimising the makespan of the vessel operation is the crux of this problem.

Zhang and Kim's approach is to apply a local search heuristics for the inter-stage optimisation. Initially, an arbitrary hatch sequence is given (or chosen), and swapping the hatches has the possibility of yielding the different number of total cycles: see Fig. 5b, c. Then, using a local search heuristic called Gap-based neighbourhood local search, they propose to find the best hatch sequence among the neighbourhood that minimises the total makespan.

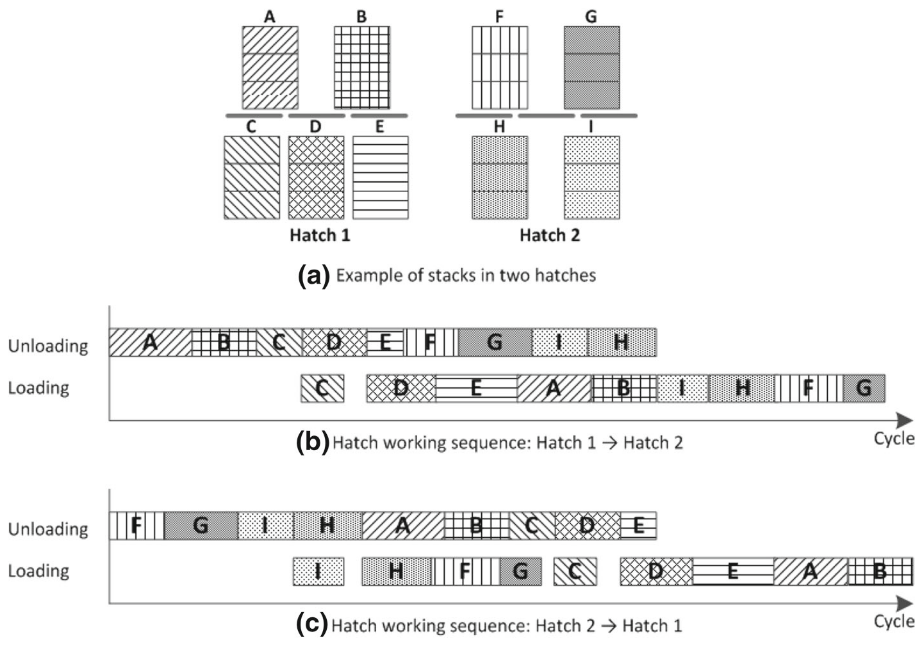

A
B
F
G
C
D
E
H
I
Hatch 1
Hatch 2
(a) Example of stacks in two hatches
Unloading
Loading
A B C D E F G I H
C D E A B I H F G
(b) Hatch working sequence: Hatch 1 → Hatch 2
Cycle
F G I H A B C D E
I H F G C D E A B
(c) Hatch working sequence: Hatch 2 → Hatch 1

Fig. 5 Simple comparison between different hatch sequencing orders

In order to simplify the Gantt chart for processing each hatch, they classify the tasks in one hatch into three parts: head, tail and trunk as in Fig. 6a.

Head part consists of not only all the unloading activities on the deck, but also the first unloading activity in the hold. Tail part consists of not only all the loading activities on the deck, but also some part of loading activity in the hold. Trunk part consists of both loading and unloading activities, which represent dual cycles. It is the ignorable part in the whole schedule during the hatch sequencing, and can be reduced by pushing the gaps as in Fig. 6b. Since the trunk part involves both loading and unloading activities, it cannot affect the makespan of the original problem when swapping hatch sequences during the inter-stage optimisation. Therefore, it can be reformulated as in Fig. 6c by deleting the trunk and linking the head and the tail up together. Then, the reformulation falls into a category of a two-machine flow shop scheduling problem, which again enables the use of Johnson's rule to obtain the optimal hatch working sequence.

4 Issues on the DCQCSP

4.1 Crane interference constraints

Since a QC is a rail-mounted vehicle, crane interference constraints are of such an importance when it comes to the **QCSP** that involves multiple QCs. This is also true for the **DCQCSP** in multiple hatches, which is a special case of the **QCSP**. Two types of crane interference constraints are typically considered:

- Non-crossing constraint: QCs cannot cross each other;
- Safety constraint: There is a safety distance between QCs at any time.

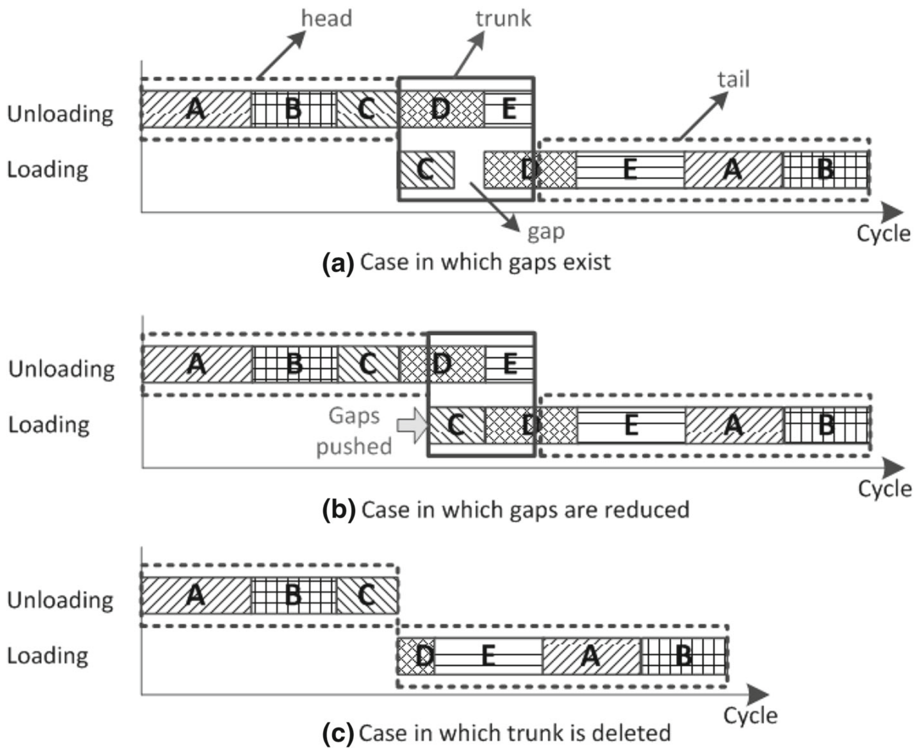

head
trunk
tail
Unloading
Loading
A
B
C
D
E
gap
Cycle
(a) Case in which gaps exist
Gaps pushed
(b) Case in which gaps are reduced
(c) Case in which trunk is deleted

Fig. 6 Abstraction of tasks in one hatch

In the **QCSP**, interference constraints were first included in Kim and Park (2004) as a mathematical form, and thereafter rigorously studied by Moccia et al. (2006), Sammarra et al. (2007), Bierwirth and Meisel (2009), and so on. However, the existing models for multiple QC double cycling do not take these constraints into account. To demonstrate the incorrectness in the model of Zhang and Kim (2009), we consider a sample problem where there are four hatches and two QCs.

Assume that the optimised hatch sequence among the hatches arranged laterally as A, B, C and D is resulted in the sequence of A→D→C→B through the inter-stage optimisation. One example of the Gantt chart of this hatch sequence would be such as in Fig. 7a. We can expect two QCs work simultaneously with double cycling between the hatches of ‘A and D,’ ‘D and C,’ and ‘C and B.’ In this sample schedule, the first hatch to work is A, and therefore we locate one QC (QC1) in this hatch. By the time QC1 has to work on the single loading tasks in the hatch A, another QC (QC2) in the hatch D participates in the single unloading tasks in the hatch D, the combination of which makes QC1 and QC2 perform the double cycling operation between the hatch A (loading) and the hatch D (unloading). Once the tasks in the hatch A are completed, QC1 that was working in the hatch A moves to the hatch C, and starts to perform the single unloading tasks when it is appropriate to engage in the double cycling operation with QC1 in the hatch D.

In this manner, the entire movements of QCs along the hatches can be illustrated in Fig. 7b. Here we can detect a crossover situation between the two QCs when QC2 attempts to work in the hatch B while QC1 is assigned to the hatch C. This means the solution found by this

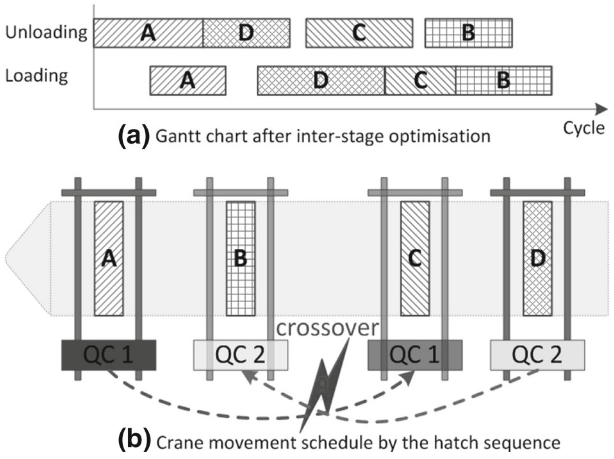

Unloading
Loading
A
D
C
B
Cycle
(a) Gantt chart after inter-stage optimisation
A
B
C
D
crossover
QC 1
QC 2
QC 1
QC 2
(b) Crane movement schedule by the hatch sequence

Fig. 7 Counter example of crane interference

model is infeasible by the non-crossing constraint. Furthermore, it could be that hatches in the neighbouring hatch sequences are located within the safety margin, in which case the solution derived by this model is also infeasible by violating the safety constraint.

4.2 Maximising double cycles by connecting two hatches

The reasoning of Zhang and Kim's approach is that the more double cycles are carried out between groups of two hatches, the less number of operation cycles as the makespan will be yielded. So, in their model, maximising the number of double cycles is equivalently treated as minimising the number of operation cycles. If it were limited to a single QC double cycling, maximising the double cycles would likely lead to the minimisation of overall makespan. However, it is worthwhile to note that maximising the double cycles by connecting two appropriate hatches continuously does not necessarily lead to the minimisation of overall makespan due to the following reasons:

1. At times one QC, which is connected to another QC for double cycling, travels to a hatch and waits there until the scheduled double cycling sequence arrives. Therefore, travel time should be considered and also unnecessary idle time easily arises until a matching pair of double cycle sequences between the two QCs arrive;
2. To mitigate the issue with the above situation, it can sometimes be more efficient that two or more unloading QCs are connected to one loading QC, or vice versa. However, Zhang and Kim's model has no room for this consideration;
3. A delay of a QC resulting from avoiding the interference with another QC should be considered.

4.3 Granularity of a task

The granularity of a task needs to be elaborated for single QC double cycling. In the existing literature on the **DCQCSP**, a task is typically defined on the basis of a stack. As noted earlier, a task is a unit of activities to be processed without preemption by a QC. We can easily encounter a contradiction if we assume a stack as a task in the case of double cycling operation. In double cycling, two stacks are alternatingly involved in the operation, one stack

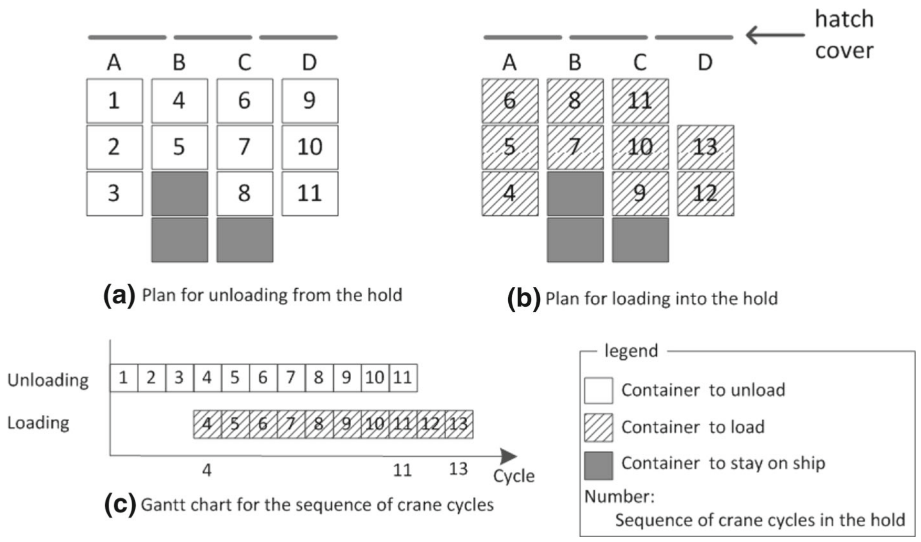

hatch cover
A B C D
(a) Plan for unloading from the hold
(b) Plan for loading into the hold
Unloading
Loading
Cycle
(c) Gantt chart for the sequence of crane cycles
legend
Container to unload
Container to load
Container to stay on ship
Number:
Sequence of crane cycles in the hold

Fig. 8 Sequence for unloading, dual and loading cycles

for unloading and the other stack for loading. However, this does not mean we can isolate these two stacks as one task because the combination of two stacks for double cycling is continuously changing one after the other and therefore there is no way to isolate two specific stacks as a task. As Fig. 8 illustrates, stacks involved with double cycling are continuously changing. Between the fourth and the fifth cycle stacks A and B are involved, for the sixth cycle stacks A and C are involved, and between the seventh and the eighth cycle stacks B and C are involved in the double cycling operation. It is evident that no single stack or two can be isolated to consist of a task in case of double cycling.

Considering the issues with a stack as a task in double cycling model, our question is what could be an alternative unit of a task for future research direction. Obviously, this remains to be explored through further research, so there is neither more efficient nor less efficient answer to this question yet. Our suggestion to consider is as follows:

For single QC double cycling, all activities above the deck have nothing to do with double cycling. So, we can leave the modelling of tasks above the deck to the traditional **QCSP** model. For example, Kim and Park (2004) suggest that tasks be separated by the hatch covers for unloading containers, and by container groups for loading containers. Following this suggestion, there is only one task for containers to unload from above the deck, and are as many tasks as there are distinguishable groups (container size and type, destination and so on) for containers to load into above the deck. However, retaining the stack concept as a task for activities below the deck seems inevitable because double cycling only becomes available as soon as any stack in the hold is free of containers to unload.

Considering the continuously changing combination of stacks for double cycling, we propose to ignore tasks on loading stacks where double cycling takes place. In this proposal, *dual* tasks are additionally categorised on top of the traditional *load* and *unload* tasks. The *unload* tasks are reclassified as *dual* tasks where unloading stacks are simultaneously worked with other loading stacks. Therefore, the tasks in the sample shown in Fig. 8 can be reduced to five tasks—A (unload), B (dual), C (dual), D (dual) and D (load)—instead of having eight tasks (four *unload* tasks and four *load* tasks). The details of reclassified tasks are as follows:

– A(unload) : unload 1–3
– B(dual) : unload 4–5 and load 4–5
– C(dual) : unload 6–8 and load 6–8
– D(dual) : unload 9–11 and load 9–11
– D(load) : load 12–13

5 Summary and future research

The analysis made in this paper discloses some serious weaknesses of the existing model for multiple QC double cycling. First and foremost, they do not take into account the assignment of tasks to each QC properly, and thereby the yielded solution can cause the crane interference, thus making it infeasible. Later work by He et al. (2011) attempt to include the crane interference constraints in the formulation but it only considers the situation where two QCs operate on the same stack simultaneously, which is far from satisfactory for the consideration of the safety distance between QCs, let alone the non-crossing constraint.

Secondly, the objective in the existing model on the multiple QC double cycling is to maximise the frequency of double cycles by connecting two appropriate hatches continuously. However, we have rebutted that doing so does not necessarily result in the minimisation of overall makespan.

In addition, there are some real world requirements like additinoal physical constraints and operational convenience to the formulations aiming at double cycling. To take care of the missing contraints may need a totally different approach than just used by the existing models. Since the existing models do not seem to be scalable enough to include those missing constraints straightaway, our current research effort is on formulating a new model that will include the missing contraints on top of the constraints addressed by the existing models. Some of the approaches we have investigated or plan to investigate are as follows:

– the parameters for QCs need to be included to support the assignment of tasks to QCs;
– the objective function needs to consider not only the efficiency of the seaside, i.e. the number of the required QC cycles, but also the efficiency of the landside, i.e. the number of landside transport cycles;
– a new perspective on the granularity of a task needs to be examined as we have already elaborated in the previous section.

Exploiting double cycling strategy for quayside operation at seaport is a useful option to boost the terminal productivity especially when the infrastructure expansion is limited. A properly implemented double cycling process can improve the QC operation, which is generally regarded as the bottleneck to the entire throughput at seaport, by reducing the required number of QC cycles as well as that of landside transport cycles. Also, this can economically benefit the terminal operators by saving fuel consumption with less travelled yard-to-shore transports.

In contrast to the potential benefits of the double cycling, single cycling method is commonly employed in most container terminals for various reasons. One critical reason, among others, is that the industry practitioners have been used to the current practice for a long time to the point where many rules and systems are already mingled with the current single cycling method. For one, there can be issues with labour environment. A gang is a group of stevedores that work on a QC during the given shift hours. In some cases, the labour agreement even specifies that once a gang is assigned to operate on a ship, the gang will not work on another ship unless paid by an extra amount of money. For another, the tools the terminal operators use often require modifications such as enabling to plan double cycling sequences easily if the current tools they use do not support it yet. For the other, even though most academic research such as this paper assumes the yard condition is so favourable that any internal carrier as well as any container to be loaded can be available whenever needed by the double cycled QC, the reality is often not the case. There can be more and more reasons that hamper the implementation of double cycling. Therefore, the above items, but not limited to, are the agenda that can be addressed in future research direction.

Acknowledgments The authors would like to thank the referees for their valuable comments and suggestions.

References

Avriel, M., Penn, M., Shpirer, N., & Witteboon, S. (1998). Stowage planning for container ships to reduce the number of shifts. *Annals of Operations Research*, *76*, 55–71.

Bierwirth, C., & Meisel, F. (2009). A fast heuristic for quay crane scheduling with interference constraints. *Journal of Scheduling*, *12*(4), 345–360.

Bierwirth, C., & Meisel, F. (2010). A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, *202*(3), 615–627.

Feo, T., & Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*(2), 109–133.

Gadeyne, B., & Verhamme, P. (2011). Optimizing maritime container terminal operations. Master's thesis, Ghent University, Belgium.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on Computing*, *1*(3), 190–206.

Goodchild, A. (2005). Crane double cycling in container ports: algorithms, evaluation, and planning. PhD thesis, University of California, Berkeley.

Goodchild, A., & Daganzo, C. (2004). *Reducing ship turn-around time using double-cycling*. Institute of Transportation Studies, University of California, Berkeley.

Goodchild, A., & Daganzo, C. (2006). Double-cycling strategies for container ships and their effect on ship loading and unloading operations. *Transportation Science*, *40*(4), 473–483.

He, X., Wang, S., & Zheng, J. (2011). A hybrid heuristic algorithm for integrated large-capacity quay crane scheduling problem. In *Computer Research and Development (ICCRD), 2011 3rd international conference on, IEEE*, vol. 1, pp. 309–312.

Johnson, S. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, *1*(1), 61–68.

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

Moccia, L., Cordeau, J., Gaudioso, M., & Laporte, G. (2006). A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal. *Naval Research Logistics (NRL)*, *53*(1), 45–59.

Sammarra, M., Cordeau, J., Laporte, G., & Monaco, M. (2007). A tabu search heuristic for the quay crane scheduling problem. *Journal of Scheduling*, *10*(4), 327–336.

Song, J. (2007). A study for optimization of double cycling in container ports. *Port Technology International*, *36*, 50–52.

Wang, D., Li, X., & Wang, Q. (2011). A two-stage composite heuristic for dual cycling quay crane scheduling problem. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE international conference on, IEEE*, pp. 1902–1907.

Zhang, H., & Kim, K. (2009). Maximizing the number of dual-cycle operations of quay cranes in container terminals. *Computers & Industrial Engineering*, *56*(3), 979–992.