Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 418689, 12 pages
http://dx.doi.org/10.1155/2014/418689

Research Article

Double Girder Bridge Crane with Double Cycling: Scheduling Strategy and Performance Evaluation

Dandan Wang,[1,2] Anne Goodchild,[3] Xiaoping Li,[1,2] and Zun Wang[3]

[1] *School of Computer Science and Engineering, Southeast University, Nanjing 211189, China*
[2] *Key Laboratory of Computer Network and Information Integration, Ministry of Education, Nanjing 211189, China*
[3] *Department of Civil and Environmental Engineering, University of Washington, 121E More Hall, P.O. Box 352700, Seattle, WA 98195-2700, USA*

Correspondence should be addressed to Xiaoping Li; xpli@seu.edu.cn

Received 19 February 2014; Accepted 7 August 2014; Published 25 August 2014

Academic Editor: X. Zhang

This paper introduces a novel quay crane design, double girder bridge crane (DGBC). DGBC is capable of handling containers of two adjacent bays simultaneously, avoiding crane collisions, saving travelling and reposition cost, and eventually improving terminal efficiency. This problem is formulated as a resource-constrained project scheduling with objective to minimize the maximum completion time. A two-stage heuristic algorithm is proposed in which an operating sequences on each bay is obtained by double cycling, and the integrated timetable for both bays is constructed by solving resource conflicts using the proposed minimum cost strategy. We examine effectiveness and performance of applying DGBC with double cycling. A case study is presented to illustrate how DGBC works with the two-stage method. Three extreme cases with respective conflict types are investigated to develop the performance bounds of DGBC with double cycling. The results show that DGBC can significantly improve terminal productivity, and outperforms single girder crane in both makespan and the lift operation percentage. The highest DGBC efficiency does not require maximum double cycles in two bay schedules; rather the integrated timetable for two bays is the main contribution to the DGBC performance as it yields better cooperation between two spreaders and the driver.

1. Introduction

The rapid growth in global trade has led to remarkably higher shipping volumes and increased vessel carrying capacity. Technological innovations and high-efficient scheduling strategies are required to meet the demand of increasing throughput in container terminals, especially in managing larger capacity vessels while reducing operating cost and maintaining service reliability. It is important to ensure their operating efficiency by incorporating new technologies and operating strategies when developing infrastructures.

Limited by the current technologies of transportation, the previous work has been mainly focused on the operating strategies for the existing equipment, that is, the traditional single girder quay crane (SG). The crane productivity is greatly improved by those researches; for example, *double cycling* which is well established in Goodchild and Daganzo [1] enables the crane to perform unloading and loading simultaneously. However, SG serves each bay individually, being constrained by safety distance and crane collisions. Usually, cranes are costly as they consume a great deal of power to travel and position between bays which leads to a less economical manner for terminals. If cranes can be deployed in a multiple girders, the efficiency of terminal would be greatly enhanced.

Having this assumption in mind, we propose a new crane-based design method in this paper, double girder bridge crane (DGBC). The availability of double girders would considerably increase the crane's handling capacity while reducing its travelling cost, because this method enables DGBC to serve two adjacent bays at the same time with only one driver. Two girders share the infrastructure, DGBC is therefore operated closer to the economic purpose of the terminals, and its benefits can be obtained with limited investments, such as equipping SG with double girders.

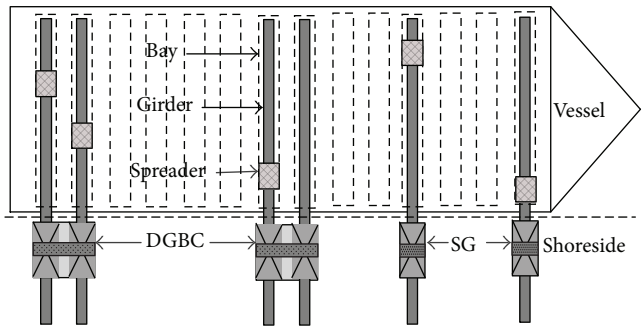


Figure 1: Comparing DGBC with SG.

DGBC can be installed in a terminal as shown in Figure 1. Compared with SG, DGBC is equipped with two girders; each of them has one independent spreader working on the containers of adjacent bays simultaneously with no change to the safety distance constraints. The DGBC-based scheduling problem is described by a directed graph [2], with the objective to minimize the makespan (maximum completion time of two bays) which is measured by processing time. The performance of DGBC is carefully examined in this work. Double cycling reduces the empty movement in each cycle, but increases the processing time of one cycle, since the traverse and hoist have to move more slowly when a spreader carries a full container. Therefore, the effectiveness of applying double cycling on DGBC is also discussed in this paper.

The main contribution of this work is that a DGBC-based scheduling design in shoreside is proposed, which identifies its benefits including the capabilities to serve two bays simultaneously and save crane travel and reposition cost. Based on double cycling, a two-stage heuristic algorithm is developed to demonstrate how DGBC is implemented. It is found from the comparison that DGBC outperforms SG, and double cycling plays less important effect on DGBC than SG.

This paper is organized as follows. Section 2 introduces the related literature. The DGBC framework and its scheduling problem description are given in Section 3. The model is discussed in detail throughout Section 4. Section 5 presents a two-stage heuristic algorithm, including making the operating sequence for each bay and acquiring the integrated timetable for two bays. Section 6 evaluates DGBC performance and double cycling effectiveness by comparing with SG problems. Conclusions are given in Section 7.

2. Literature Review

The quay crane is a key bottleneck for overall terminal efficiency [3]. Daganzo [4] first investigated the quay crane scheduling problem to minimize the total weighted completion time of vessels. An exact method was proposed for a simple static case and an approximate one for the dynamic case. Bierwirth and Meisel [5] classified the literature on quay crane scheduling problems into four classes. The classification scheme is based on the container storage strategy, that is, stacks or area within a bay [6], single bays [7], contiguous groups [8], and each single container [9]. However, there are only a few papers focusing on the problem with each single container [9]. They addressed the internal reshuffling problem for each container. In this paper, we also consider DGBC scheduling on single containers. These problems have much larger problem scale than those addressed using the paradigm [6].

A drawback of the traditional problems mentioned above is that there are many empty movements existing in crane operating cycles due to the use of single cycling, which would significantly affect the crane serving efficiency. However, it is reported that double cycling can reduce empty movements and improve the utilization of quay cranes [1]. On average, double cycling can reduce 40% of operating cycles over single cycling without hatches. Besides, Meisel and Wichmann [9] addressed the internal reshuffles based on double cycling. Zhang and Kim [10] considered double cycling quay crane scheduling problem with hatches, whose results showed that this proposal could get better solutions than human planners.

Many works existed in the literature focusing on the different mathematical formulas for the quay crane scheduling problem. A rich model given by Legato et al. [11] intended to cover practical constraints like service rates, ready times, due dates, and safety requirements. Recent studies focus on cooperation between different facilities. Yuan et al. [12] formulated a mathematical model combining cranes with trucks and then solved it by using a job grouping approach. Chen et al. [13] examined the interactions between crane handling and truck transportation by addressing them simultaneously. Ding et al. [14] used a multidisciplinary variable coupling optimization method to coordinate different equipment. In addition, two cooperating cranes were investigated by Vis and Carlo [15] so as to work on the same stack together. With their method, the container serving time can be reduced, but the large running costs still remain because of two separate cranes. Most research in this area aims at minimizing crane cycles; however, the processing time, which will be discussed in this paper, is of ultimate interest.

3. Problem Description

In order to demonstrate the DGBC-based scheduling problem, we first introduce the framework of DGBC and then give the problem description and settings, as well, and the application of double cycling to DGBC is discussed.

3.1. DGBC. DGBC is a quay crane equipped with twin container spreaders on double girders. Each girder is positioned on one bay with the spreader handling the containers in this area, while another serves the adjacent bay. They are able to work on adjacent bays simultaneously. Both handling concepts are supported by common frame, cable, and power drives. Although this design increases the energy requirements compared with two single girder cranes, the savings on mechanical consumption and maintenance cost are worth more consideration. Furthermore, only one driver is required for two spreaders' operations.

The mechanical structure of DGBC is depicted in Figure 2. Two spreaders, specifically spreader 1(2) works on bay 1(2), are controlled by only one driver. Linkages

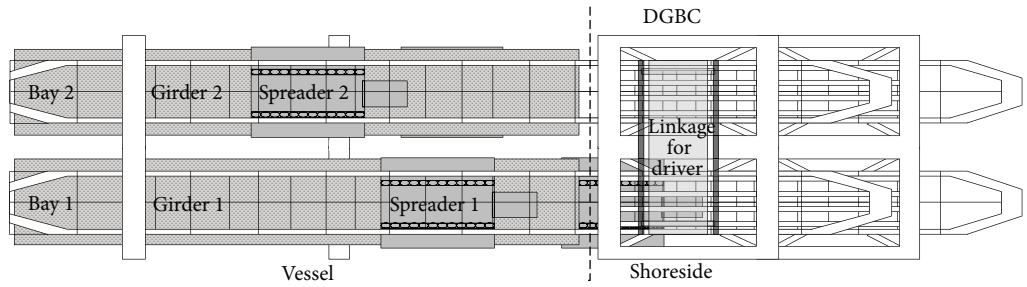


Figure 2: Top view of DGBC.

enable the driver to manage two spreaders to conduct the lifting concept, where the driver controls the spreader to lift containers with different heights and positions, but he/she cannot work until the spreader automatically moves along the girder and arrives at the desired destination. Because the two spreaders perform work simultaneously, the time used for lift is increased compared with the makespan; thus the driver's waiting time is reduced.

3.2. Definition. The problem is described by a directed graph $G(V, E)$ with node set V and edge set E. The node set V corresponds to $|V| = n$ activities. The nodes can be further divided into two subsets V_B; that is, $|V_B| = n_B$ $(B = 1, 2)$, where V_B is the node set of bay B, $B = 1, 2$; then $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$. Each node is one unloading/loading activity which refers to exactly one lift. Edge set $E = \{(i, j) : i, j \in V; i \rightarrow j\}$ represents the temporal precedence constraints between two activities; that is, $i \rightarrow j$ if activity i must finish before activity j starts. According to the physical layout of the vessel stowage plan which satisfies the stability of the vessel, unloading operations of a stack must precede its loading operations. All container-unloading above a hatch must be finished before performing loadings below the hatch. Container-loading above a hatch cannot start until the loading operations below the same hatch are completed. Adjacent lifts are separated by a series of spreader movements. The movement between the lifts i and j is sequence-dependent, denoted by $o_{i,j}$ which is required by the sequentially scheduled activities in the same bay. In addition, dummy activities 0 and $n+1$ with zero duration are added to make sure there are only one starting and finishing node in G.

Take a simple case as an example where the side view of the stowage plan for bays B_1 and B_2 is given in Figure 3. Each bay has one hatch with stacks above and below the deck. Containers are indexed by number and stored one by one as stacks. According to the definition, graph G can be constructed as shown in Figure 4. Each node represents the processing of a container indexed by numbers with the processing time and resource requests (spreader, driver). The precedence relations between activities are characterized by directed edges. Each bay corresponds to a subgraph connected by the dummy nodes 0 and 24.

3.3. Resources. There are three resources: one driver (H) and two spreaders (Q_1 and Q_2). All resources are unary resources

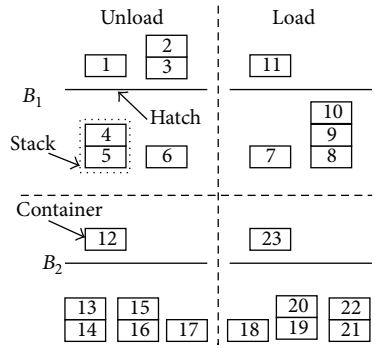


Figure 3: Stowage plan for two bays.

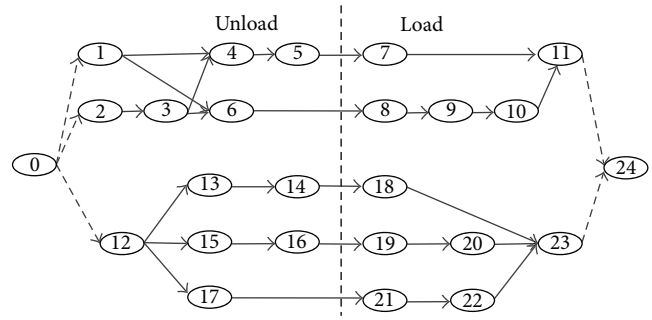


Figure 4: Graph G.

with the available capacity of 1 [16–19]. Generally, one unloading/loading operation has its resource requirement denoted by r_{ik}, where i is the operation and k is the resource. There are two types of operations: movement and lift; the former can be executed automatically by the spreader itself while the latter requires the driver to control the spreader manually. Different from the role of driver, which is the dedicated resource only involved in lifting, spreaders are the allocatable resources in lift and movement; that is, Q_1/Q_2 serves bay B_1/B_2. Specifically, each spreader of DGBC works analogously to one SG, but the lifts of two spreaders must cooperate with each other in the charge of one driver. If the driver has not completed the lift on the current bay, the other spreader has to wait although it has already arrived at the appointed location on the other bay. As a result, there will be blocking time between two sequential activities in different bays.

Table 1: Four modes of movement combination.

o_{ij}	Lift i	Movements	Lift j
O^1	U	$\{VY, \overline{YV}\}$	U
O^2	L	$\{\overline{VY}, YV\}$	L
O^3	U	$\{VY, YV\}$	L
O^4	L	$\{VV\}$	U

3.4. Double Cycling for DGBC. Double cycling is considered in the operating strategy to improve processing efficiency, in which DGBC performs the unloading when the spreader carries an import container from the vessel to the shore and then conducts the loading when the spreader moves from the shore to the vessel with an export container. Then the empty movement in single cycling is replaced by the full movement resulting from double cycling, and the number of operating cycles is reduced. However, the processing time of a double cycle is longer than that of a single cycle as the spreader has to move slower while carrying a full container.

A series of movements must be executed between the adjacent lift activities in the same bay, because the spreader must move to the assigned location before raising or lowering a container. In other words, each $o_{i,j}$ relates to the immediately precedent activity i and the immediately successive j. For this reason, $o_{i,j}$ is sequence-dependent [20] and has four modes ($O^1 \sim O^4$) as listed in Table 1.

Unloading (U)/loading (L) in single or double cycling yields different combinations of the movements in Ψ, which is the set of movement type. The full movement YV denotes that the spreader carrying the container moves from the shore (Y) to the vessel (V) while VY denotes the reverse movement. Then the empty movements $\overline{YV}$ and $\overline{VY}$ imply that only the spreader itself moves between the yard and vessel. VV represents the empty movement within the vessel in the double cycling strategy. Furthermore, one movement conducted by Q_1/Q_2 may overlap with another movement on Q_2/Q_1, because Q_1 and Q_2 can move in parallel.

4. Model

In this section, assumptions and major notations are given, and the DGBC scheduling problem model is presented.

4.1. Assumptions. In order to model the DGBC problem, we make the following assumptions. All the containers can be implemented by DGBC. The processes within shoreside are simplified by ignoring YY, because it relies on the efficiency on the shoreside. For traditional cranes, drivers move with the spreaders; actually they participate in the spreader lift operations. The movement of the driver is neglected here, as we focus on the spreader operations of DGBC. Reshuffles will be unloaded and then reloaded. By choosing the time unit sufficiently small, we can always assume that the processing times are nonnegative integers. Each activity cannot be interrupted until it is completed. Moreover, all activities and resources are available from the start of the project.

4.2. Notations. The major notations used in the remainder of this paper are summarized as follows.

B: bay, $B = 1, 2$

n_B: number of activities in bay B

n: total number of activities $n = n_1 + n_2$

Ψ: set of movement type, $\Psi = \{YV, VY, VV, \overline{VY}, \overline{YV}\}$

Φ: set of lift type, $\Phi = \{U, L\}$

p_l: moving time of type $l \in \Psi$

p_i: processing time of lift i

π: lifts permutation

π^B: permutation of the lifts in bay B

A_t: set of operation in work at time instant t

s_i: start time of the activity i

O: set of movements

V: set of lifts, $|V| = n$

$o_{i,j}$: movement of a spread between lift i and j

E: set of edges in graph G

b_i: blocking time of lift i

$C^B_{\max}$: makespan of bay B

σ: a large number which serves as infinity

R: set of all resources $R = \{Q_1, Q_2, H\}$

r_{ik}: requirement of resource $k \in R$ by operation i.

4.3. Mathematical Model. DGBC cannot finish the project until all the activities on two bays have been completed, so the makespan is the maximum completion time of all activities. The optimization objective is to minimize the makespan, whose function can be expressed as

$$\text{Min} \quad s_{n+1}. \tag{1}$$

The constraints are as follows.

(1) Precedence Constraints. Two lifts cannot be processed by the driver simultaneously, no matter whether they are located on the same bay or two bays, respectively:

$$\sigma\left(1 - Z_{ij}\right) + s_j \geq s_i + p_i + o_{i,j}, \tag{2}$$

where $i, j \in V_B$, $B = 1, 2$. The decision variable Z_{ij} equals 1 if the lift i precedes the lift j; otherwise, $Z_{ij} = 0$.

All activities and resources are available from the start of the project. Consider

$$s_0 = 0. \tag{3}$$

(2) Lift Constraints. Each activity has exactly one lift. Consider

$$\sum_{l \in \Phi} Y_i^l = 1, \quad \forall i \in V, \tag{4}$$

where Y_i^c is the decision variable and $Y_i^c = 1$ if the lift i is of the type $c \in \Phi$; otherwise, $Y_i^c = 0$.

One lift handles one container at a time:

$$\sum_{i \in V} Y_i^l = 1, \quad \forall l \in \Phi. \tag{5}$$

(3) Resource Constraints. Resource limitation should be ensured during each time period, where the requirement for every resource cannot exceed the available capacity:

$$\sum_{i \in A_t} r_{id} \le 1, \quad i \in O \cup V,\ d \in R. \tag{6}$$

The spreader remains occupied from the start of the movement to the end of the lift on bay B, and two spreaders can move in parallel. Consider

$$0 \le r_{iQ_1} + r_{iQ_2} \le 2, \quad i \in O \cap A_t. \tag{7}$$

The lift can only be conducted by the driver controlling the spreader. Consider

$$r_{iQ_B} \ge r_{iH}, \quad i \in V_B \cap A_t. \tag{8}$$

(4) Movement Constraints. Movements are considered between every pair of sequential lifts. As mentioned in Section 3.4, the movement combination is sequence-dependent. Consider

$$o_{i,j} = \sum_{l \in \Psi} p_l X_{ij}^l Z_{ij}, \quad i, j \in V_B,\ B = 1, 2, \tag{9}$$

where X_{ij}^l is the decision variable and $X_{ij}^l = 1$ if the movement between lift i and j is of the type $l \in \Psi$; otherwise, $X_{ij}^l = 0$.

The spreader is blocked after its movement if the driver is still performing another spreader. Consider

$$b_j = \max\left\{0, s_i + p_i - s_j\right\}, \tag{10}$$

where $Z_{ij} = 1$, $i \in V_B$, and $j \in V - V_B$ indicate that the sequential lifts are located on two bays.

Related to constraint (9), the movement type is also sequence-dependent:

$$X_{ij}^{VY} = Z_{ij} Y_i^U Y_j^U, \qquad X_{ij}^{\overline{YV}} = Z_{ij} Y_i^U Y_j^U \tag{11}$$

$$X_{ij}^{YV} = Z_{ij} Y_i^L Y_j^L, \qquad X_{ij}^{\overline{VY}} = Z_{ij} Y_i^L Y_j^L \tag{12}$$

$$X_{ij}^{VY} = Z_{ij} Y_i^U Y_j^L, \qquad X_{ij}^{YV} = Z_{ij} Y_i^U Y_j^L \tag{13}$$

$$X_{ij}^{VV} = Z_{ij} Y_i^L Y_j^U. \tag{14}$$

Specifically, (11)/(12) imply single cycles where the movements VY and $\overline{YV}/YV$ and $\overline{VY}$ occur between two unloading/loading operations. Likewise, (13) and (14) show double cycles in which two full movements VY and YV connect an unloading and a loading operation, and an empty movement VV happens between the loading and unloading operations.

Special movements are defined for the dummy nodes 0 and $n + 1$:

$$\begin{aligned} o_{0,1} &= p_{\overline{YV}} Y_1^U + p_{YV} Y_1^L, \\ o_{n,n+1} &= p_{\overline{VY}} Y_n^L + p_{VY} Y_n^U. \end{aligned} \tag{15}$$

5. Methodology

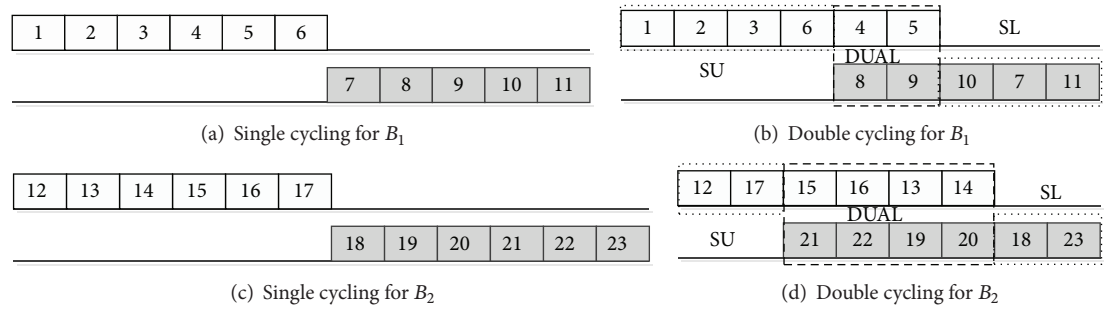
The proposed model cannot obtain the optimum in acceptable time by the existing mathematical programming solver. Naturally, DGBC can be implemented by a two-stage framework, where the first stage problem schedules the containers loading and unloading in each bay, and the second stage handles the coordination between the two spreaders' operation under the driver constraint; thus a two-stage heuristic algorithm is proposed to solve the problem. The proposed approach decomposes the DGBC problem into two stages, which can be solved in sequence.

5.1. Stage 1: Single Bay Scheduling (Double Cycling). First of all, each spreader is regarded as an independent crane to handle each bay. It is a traditional quay crane scheduling problem for single bay. The target of this problem is to find the best (un)loading sequence for each bay with minimum operating cycles. To achieve better crane processing efficiency, double cycling is introduced, which permits a quay crane to perform the unloading and loading in one operating cycle. The objective function of this problem is to minimize cycles required for loading and unloading, and the constraints are corresponding to constraints (2), (3), and (4)–(6) of the DGBC problem.

The double cycling procedure is constructed according to our previous work [21]. A two-stage composite heuristic based on the 2-machine flow shop scheduling problem is presented. In the first stage, stacks in a hatch are scheduled by the Johnson rule but we introduce a gap-shifting strategy. A reconstructive Johnson rule is further applied to the inter-hatch sequencing in the second stage. Then a (un)loading container permutation with minimum operating cycles can be obtained. Since only one bay is considered in that problem, there is no resource (spreader or driver) constraint in the timetabling procedure. Therefore, the permutation can be easily transformed into a processing timetable to perform operations as early as possible.

5.2. Stage 2: Timetable for Two Bays. Lifts require the driver's participation in controlling the spreaders to pick up and drop off containers. However, there is only one driver in charge of two spreaders for DGBC. As a result, there would be resource conflicts between two bays, in which one spreader cannot perform lifting directly after moving and has to wait for the driver to be released from the previous lifting with the other spreader. Based on the single bay scheduling results from Stage 1, a compact timetable for two bays is obtained in which the resource (driver) conflicts are solved by minimum cost strategy.

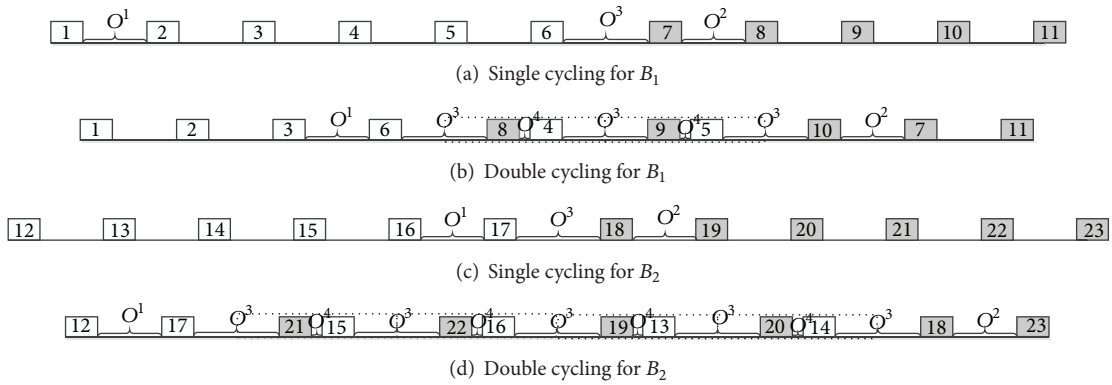
5.2.1. Conflict Type. Although the objective of single bay scheduling is to obtain as many double cycles as possible, there would still exist single cycles. To help distinguish different conflicts, SU is defined hereafter as the interval during which only unloading operations exist, SL is the interval only involving loading operations, and DUAL is the double cycling part. In an operation cycle Gantt chart



(a) Single cycling for B_1 (b) Double cycling for B_1

(c) Single cycling for B_2 (d) Double cycling for B_2

Figure 5: Operation cycle Gantt chart (a)–(d).



(a) Single cycling for B_1

(b) Double cycling for B_1

(c) Single cycling for B_2

(d) Double cycling for B_2

Figure 6: Time Gantt chart (a)–(d).

Table 2: Conflicts type.

Conflict	Single cycle	Double cycle
Single cycle	SS	SD
Double cycle	SD	DD

(OCGC); see Figures 5(b) and 5(d) in Section 5.4; single cycles are included in SU and SL while double cycles exist in DUAL.

The overlap between two lifts on different bays is defined as a conflict. There are three types of conflict between single cycles and double cycles; SS, DD, and SD. As listed in Table 2, we employ SS to represent the conflict between two single cycles, DD to denote the conflict within two double cycles, and SD to show the conflict between a single and a double cycle which is more complicated than SS/ DD.

5.2.2. Minimum Cost Strategy. To remove those three kinds of conflicts, the minimum cost strategy which aims to minimize the increment in the makespan of two bays is developed. In order to settle a conflict, we have two options; that is, either the overlapped lift on B_1 or the one on B_2 is delayed. Obviously, the blocking time (also the delayed time as shown by the grid box in Figure 7) will be directly added to the makespan of the delayed schedule and may change the final makespan of two bays. According to the minimum cost strategy, the bay with the smallest makespan will be chosen to delay. The three conflicts are solved as follows.

(1) SS: take the SS conflict between lifting containers 1 and 12 in Section 4.3 case study (Figure 7) as an example; $C^1_{\max} < C^2_{\max}$ as shown in Figures 6(b) and 6(d). If container 1 is delayed, the completion time of B_1 will be $C^1_{\max} + b_1$, and the final makespan is $\max\{C^1_{\max} + b_1, C^2_{\max}\}$. However, the final makespan is $\max\{C^1_{\max}, C^2_{\max} + b_{12}\}$ when delaying container 12, which is larger than $\max\{C^1_{\max} + b_1, C^2_{\max}\}$ as $b_1 = b_{12}$. Therefore, container 1 in B_1 will be delayed as shown in Figure 7. Because the single cycles have the similar timetable, and all the sequential SS conflicts have the same overlap, the subsequent SS conflicts can therefore be solved at the same time, such as containers 2 and 17.

(2) DD: DD can also be tackled by the minimum cost strategy, for example, containers 8 and 16 in Figure 7. And all the DDs can be removed at once if they are in serial.

(3) SD: SD cannot be resolved by one delaying operator. Due to the different movements required in single/double cycles, SD conflicts can be distinguished by the various overlaps as depicted in Figure 8. For example, the SD conflict (between the single cycle of unloading container 3 and the double cycle consisting of loading container 21 and unloading container 15) is the SD4 type; see Figures 7 and 8(4). The overlap of the succeeding SD conflicts will be changed while solving

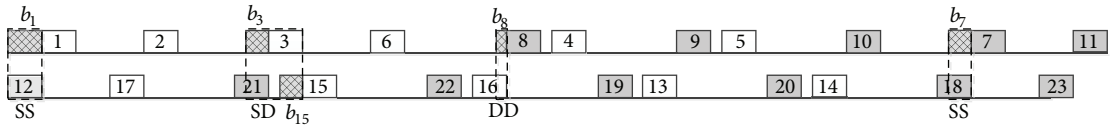


Figure 7: Timetable for two bays.

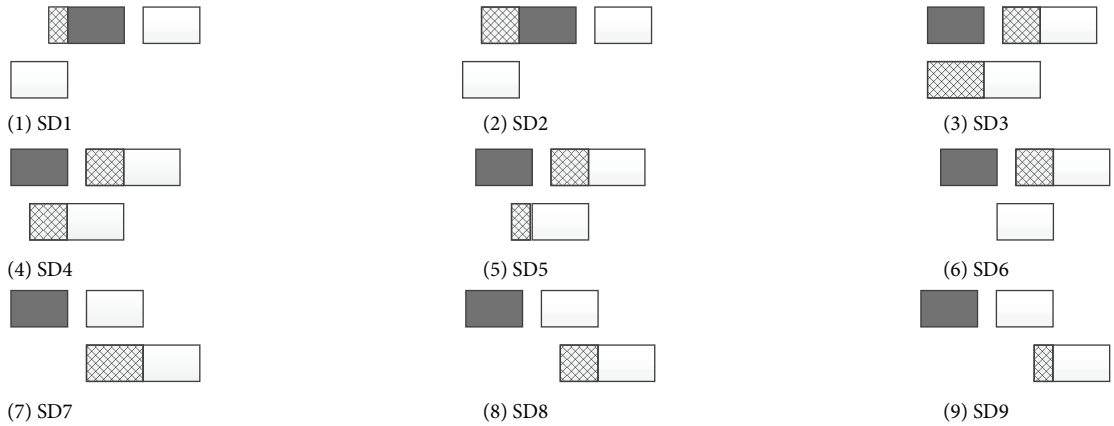


Figure 8: Nine types of SD conflicts.

the current one. Hence, the minimum cost strategy must be applied to each SD case.

5.3. Two-Stage Heuristic Algorithm. A two-stage heuristic algorithm is developed in order to solve the DGBC problem. Firstly, double cycling is used to achieve a better spreader processing schedule for each bay. The critical thing is that the two spreaders cannot be treated as two independent cranes as there is only one driver available. Then, we present a heuristic for two bays to pursue an integrated timetable, in which three types of conflicts are settled by the minimum cost strategy. The two-stage heuristic algorithm is described as in Algorithm 1.

Suppose there are at most A stacks in every hatch and B hatches for bays 1 and 2. According to the result in [21], the time complexity of the double cycling method is $\max\{O(AB\log A), O(B\log B)\}$. Due to the time complexity of timetabling as $O(\max\{n_1, n_2\})$, the final time complexity is $\max\{O(AB\log A), O(B\log B), O(\max\{n_1, n_2\})\}$.

5.4. Case Study. In this section, we provide a case study to illustrate how DGBC works and the performance of the given two-stage heuristic algorithm. Assume the stowage plan is given as the example in Figure 3.

In the first stage, the operating schedule for each bay is obtained independently and depicted by OCGC in Figure 5. Only the lift is shown in OCGC. The unit of the horizontal axis is one cycle. Specifically, white boxes relate to the unloading operations while black ones are loading operations. The number in each box is the index of the processed container.

The solutions obtained by single cycling and double cycling are provided, respectively, to compare their performance. All the cycles in Figures 5(a) and 5(c) are single cycles, because no unloading operation is performed with any loading operation in a single cycle. The results for double cycling on B_1/B_2 are given in Figures 5(b)/5(d). As mentioned above, the single cycles exist in SU and SL, as well as two double cycles shown in DUAL. For example, one double cycle is unloading container 4 and loading container 8, and another is that container 5 is unloaded while container 9 is loaded. Likewise, there are four double cycles for B_2 in Figure 5(d). Comparing the total number of operating cycles for B_1 (see Figures 5(a) and 5(b)) and B_2 (see Figures 5(c) and 5(d)), it is found that double cycling outperforms single cycling with less operating cycles because the empty movement in single cycle is replaced by full movement in double cycle.

However, OCGC cannot describe the exact processing time as movements are not displayed. In this paper, each bay's schedule is represented in time Gantt chart (TGC) (see Figure 6), in which both the lifts and the necessary movements are shown. The horizontal axis represents the time usage of the spreader including the movements and lifts. Apart from the spreader, the lift also requires a driver, which is represented by a box with the processed container index. TGC is assumed to start from a lift for brevity.

Figure 6(a) shows the TGC of B_1 of single cycling. Taking $o_{1,2}$ as an example, the spreader brings container 1, after lifting it from the vessel to the shoreside which is a full movement VY, and then the spreader itself goes back to the vessel, preparing for lifting container 2 which corresponds to the empty movement $\overline{YV}$. The combination of these two movements $\overline{YV}$ and VY belongs to mode O^1. Analogously, the movements required between loading containers 7 and 8 are the full movement YV and the empty one $\overline{VY}$; then $o_{7,8}$ is of mode O^2. Because there are two full movements, that is, YV and VY, between unloading container 6 and loading container 7, $o_{6,7}$ is of mode O^3 as shown in Figure 6(a).

(1) Scheduling B_1 and B_2 in double cycling. Obtain π^B. //single bay scheduling
(2) Compute the timetable and makespan $C^B_{\max}$ for each π^B.
(3) While (there is any activity in π^B) //two bays timetabling
  (3.1) Pick the earliest activities i and j in π^1 and π^2 respectively.
  (3.2) If there is no conflict between activities i and j
    (3.2.1) If $s_i < s_j$
      (3.2.1.1) Add activities (i, j) into π and perform.
      (3.2.1.2) Remove activities i and j from π^B.
    (3.2.2) Else //$s_i \geq s_j$.
      (3.2.2.1) Add activities (j, i) into π and perform.
      (3.2.2.2) Remove activities i and j from π^B.
  (3.3) Else //there exist a conflict between activities i and j
    (3.3.1) Calculate the blocking time b_i/b_j on π^1/π^2. //delay one lifting
    (3.3.2) If $\max\{C^1_{\max} + b_i, C^2_{\max}\} < \max\{C^1_{\max}, C^2_{\max} + b_j\}$ //minimum cost strategy
      (3.3.2.1) Add activity j into π and perform.
      (3.3.2.2) Remove activity j from π^2.
      (3.3.2.3) Right shift activity i and update makespan $C^1_{\max}$.
    (3.3.3) Else
      (3.3.3.1) Add activity i into π and perform.
      (3.3.3.2) Remove activity i from π^1.
      (3.3.3.3) Right shift activity j and update makespan $C^2_{\max}$.
(4) $C = \max\{C^1_{\max}, C^2_{\max}\}$.
(5) Return π.

Algorithm 1

In contrast, all four modes appear in Figure 6(b) when applying double cycling to bay B_1. Modes O^1 and O^2 are related to single cycles such as $o_{3,6}$ and $o_{10,7}$. The mode O^4 is the empty movement within the vessel which may only exist after the mode O^3; for instance, $o_{6,8}$ is followed by $o_{8,4}$. One double cycle includes one unloading lifting, one loading lifting and three movements, that is, YV, VY, and VV. In fact, each movement (YV/VY/VV) is a pair of movement combination of modes O^3 and O^4. For example, a double cycle consisting of loading container 8 and unloading container 4 is identified in Figure 6(b). Specifically, the spreader carries container 8 from the shoreside to the vessel and lays it down under the control of the driver; then the spreader will not go back to the shoreside but move toward container 4 within the vessel, after that the spreader is managed by the driver to pick container 4 up and move back to the shoreside. As well, Figures 6(c) and 6(d) depict the TGC of bay B_2 with both single cycling and double cycling, respectively. Throughout Figure 6, the performance of double cycling on one bay is better than that of single cycling. Besides, the improvements shown in OCGC and TGC are different. The former is characterized by operating cycles while the latter, used in our method, is characterized by time.

In the second stage, the timetable for two bays will be constructed under the driver constraint, since each lift is set to be as early as possible in the obtained single bay schedules (see Figures 6(b) and 6(d)); then the need for a single driver in two bays yields the conflict, as shown in Figure 7, for example, the SS conflicts (unloading containers 1 and 12 and loading containers 7 and 18), the DD conflicts (unloading container 16 and loading container 8), and the SD conflicts (loading container 21 and unloading containers 3 and 15). According to the proposed heuristic algorithm, the conflicts are resolved by the minimum cost strategy. The timetable for two bays is presented in Figure 7 with the blocking time described in the grid boxes.

Table 3: The parameters of processing time.

$c \in \Phi$	U	L	$l \in \Psi$	$\overline{YV}$	$\overline{VY}$	VV	YV	VY
p_c (s)	60	60	p_l	40	40	20	80	80

6. Evaluations and Discussion

This section examines the DGBC performance by testing the problem in three extreme cases. The performance of DGBC improvement can be bounded by those three extreme cases; each of them is compared with the SG problem. The effectiveness of double cycling applied on DGBC is also discussed.

6.1. Evaluation Parameters. In DGBC evaluation, assume there are n activities where $n = n_1 + n_2$ and n_1/n_2 is the number of the processing containers in B_1/B_2. In purposes of DGBC analysis, the parameters are set according to the data of the quay crane found in Stahlbock and Voss [3] which is listed in Table 3.

In the following discussion, SG-SS/SG-SD denotes SG using single cycling/double cycling. The objective of this problem is to minimize the makespan (measured in seconds) of the crane serving two bays. Besides, there are two measures

Table 4: Blocking time of DG-SS, DG-DD, and DG-SD.

Crane	DGBC										
Case	DG-SS	DG-DD	DG-SD								
			SD1	SD2	SD3	SD4	SD5	SD6	SD7	SD8	SD9
Block(s)	60	140	20	40	100	80	60	40	60	40	20

corresponding to makespan. One is makespan percentage (MP) which is the current makespan as a percentage of the SG-SS makespan. MP can be calculated as $100 * M/M_{SG\text{-}SS}\%$, which can quantify the improvement of the case compared to SG-SS, the lower the better. Another is lifting operation percentage (LP) used to represent how much of the makespan is used for lift. It can be computed by $100 * 60 * n/M\%$, in which the lifting time is determined by the lift number. The higher the LP is, the more the lift can be performed efficiently, and the driver's waiting time is consequently reduced.

6.2. Three Extreme Cases. There are three extreme cases with respect to three types of conflicts, respectively; they are considered independently to examine the performance of DGBC.

6.2.1. DG-SS, DG-DD, and DG-SD. Suppose there is only one type of conflicts in Stage 2; the DGBC problem is regarded as an extreme case. There are three extreme cases with SS, DD, and SD conflicts, respectively.

DG-SS is an extreme case which has only SS conflicts in the timetable. If single cycling is applied, the DGBC problem is a DG-SS case.

DG-DD is the one with only DD conflicts. DG-DD is the most effective case with maximum double cycles.

DG-SD is more complicated than the above two cases, because the timetable has various overlaps between the double and single cycles. One SD conflict can be classified into 9 types, as shown in Figure 8. A single cycle is assumed to be an unloading operation, represented by a white box. A double cycle is combined with one loading (a black box) and one unloading (a white box) operation.

6.2.2. Blocking Time. For each extreme case, the blocking time can be determined by solving SS, DD, and SD conflicts separately, as showed in Table 4.

SS conflict occurs between two single cycles, which have the same timetable. Then delaying one lift to remove the first SS conflict in one DG-SS part can solve all the remaining SS conflicts consequently, and the total blocking time is 60 s. Similarly, all the DD conflicts in one DG-DD part can be removed by delaying one lift with the blocking time 140 s.

However, each SD conflict in a DG-SD part must be addressed by the minimum cost strategy individually, and the successive SDs are transferred to another type while dealing with the current SD. All SD conflicts are rescheduled into the feasible timetables with the different blocking time depicted by the grid boxes in Figure 8. For example, both SD1 and SD2 delay the double cycles and transfer into the same timetable, but the blocking time is different, that is, 20 s and 40 s, respectively. Likewise, SD3–SD6 are rescheduled into the same result while SD7–SD9 are modified into another timetable. As listed in Table 4, the blocking time is various; that is, SD3 needs the maximum blocking time (100 s) while SD1 and SD9 have the minimum blocking time (20 s).

Crane	SG	DGBC	
Case	SG-SS	DG-SS	
		$n_1 = n_2$	$n_1 \neq n_2$
Makespan(s)	$n*180$	$n*90 + 60$	$\max\{n_1, n_2\}*180$

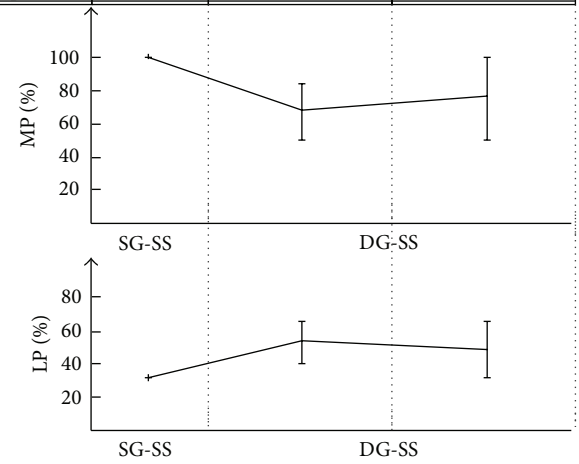


Figure 9: Comparison between SG-SS and DG-SS.

6.3. Comparison Results. Through the two-stage method, all conflicts are examined and addressed sequentially, and the DGBC problem can be separated into several parts each of which is of one extreme case. Generally, the adjacent two bays cannot be served at the same time because SGs have to keep safe distance with others; however, in order to conduct the comparison between DGBC and SG in the same scenario, one SG is assigned to serve two bays sequentially. In this section, both DGBC and SG are evaluated on n activities of two bays, and the three extreme cases of DGBC problem are compared with SG in which SG-SS is set as the reference.

6.3.1. SG. As shown in Figure 9, the makespan of SG-SS is $n * 180$ s, including $n * 120$ s for the spreader movement and $n * 60$ s for the lift. Since SG-SS is the basis of makespan comparison, its MP is 100%, and LP is 33.33% which means the driver has to wait for 66.67% of the completion time.

6.3.2. DG-SS. To make it clear, DG-SS is divided into two scenarios: $n_1 = n_2$ and $n_1 \neq n_2$. According to the presented algorithm, the makespan of DG-SS can be calculated as

Crane	SG	DGBC	
Case	SG-SS	DG-DD	
		$n_1 = n_2$	$n_1 \neq n_2$
Makespan(s)	$n*180$	$n*75 + 140$	$\max\{n_1, n_2\}*150$

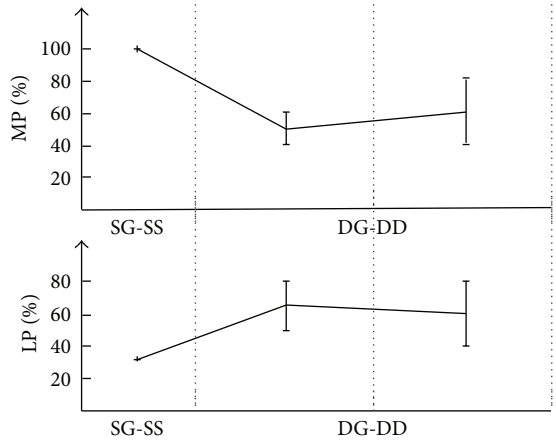


Figure 10: Comparison between SG-SS and DG-DD.

Crane	SG		DGBC		
Case	SG-SS	SG-SD	DG-SD		
			SD1~SD2	SD3~SD6	SD3~SD6
Makespan(s)	540	480	320	340	360

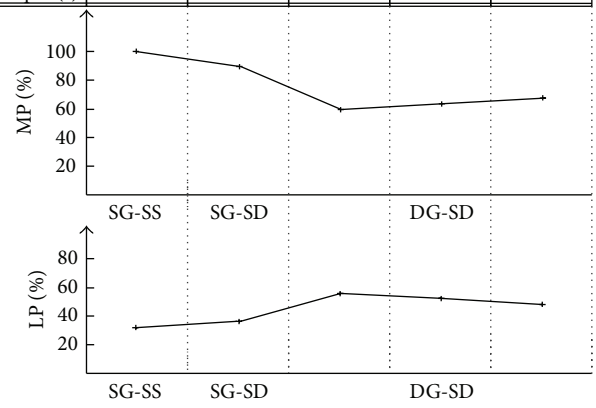


Figure 11: Comparison between SG-SS, SG-SD, and DG-SD.

$\max\{n_1, n_2\} * 180$ s when $n_1 \neq n_2$ or $(n/2) * 180 + 60$ s when $n_1 = n_2$. As illustrated in Figure 9, DG-SS outperforms SG-SS by reducing the makespan to 50% of SG-SS at the best case, and the minimum improvement of MP in $n_1 = n_2$ is 16.67%. Since the lifting time is fixed at $n * 60$ s, then the LP of DG-SS is higher than or equal to that of SG-SS in all cases. In detail, the best LP is as high as 66.67% which is twice that in SG-SS. For $n_1 = n_2$, DG-SS always gets better MP and LP than SG-SS. Figure 9 depicts that DGBC contributes to better makespan and higher LP than SG in the case of single cycling.

6.3.3. DG-DD. DG-DD can also be separated into two scenarios: $n_1 = n_2$ and $n_1 \neq n_2$. The makespan is $\max\{n_1, n_2\} * 150$ s when $n_1 \neq n_2$ and $(n/2) * 150 + 140$ s when $n_1 = n_2$, in which $n_1, n_2 \geq 2$ (to guarantee at least one double cycle). From Figure 10, we can see that DG-DD achieves better makespan than SG-SS. The MP of DG-DD ranges from 41.67% to 61.11% when $n_1 = n_2$ and from 41.67% to 83.33% when $n_1 \neq n_2$. The lifting time remains $n*60$ s; then LP of DG-DD falls within (52.17%, 80%) and (40%, 80%) for $n_1 = n_2$ and $n_1 \neq n_2$, respectively. All the MP and LP values of DG-DD outperform that of SG-SS.

However, DG-DD does not always lead to a better result than DG-SS. For example, the outcomes with DG-DD and DG-SS are overlapped. In other words, more double cycles cannot always result in the better DGBC makespan. The results in Figure 10 describe the improvement of DGBC on makespan and LP. Specifically, DG-DD obtains better efficiency than DG-SS because of the effectiveness of double cycling.

6.3.4. DG-SD. Different from the SS/DD conflicts which can be solved by one time block in one DG-SS/DG-DD part, each SD conflict in a DG-SD part must be handled individually. Therefore, the performance of each DG-SD case with different SD conflict is compared with both SG-SS and SG-SD; see Figure 11.

We assume that there are three lifts in this comparison which simplifies the conflict detection and makes evaluation tractable. For SG-SS, the three lifts are supposed to be processed sequentially in single cycles. Therefore, the makespan is 540 s; the lifting time is 180 s; then the LP is 33.33%.

In contrast, SG performs the three operations sequentially in one single cycle and one double cycle, denoted by SG-SD. The makespan is computed as 480 s, which is 88.89% of that in SG-SS. With the same lifting time 180 s, the LP is improved to 37.5% higher than SG-SS.

As shown in Figure 8, SD1 and SD2 result in the same timetable with the makespan of 320 s which is 59.26% of SG-SS. It is the best makespan among 9 SD types as depicted in Figure 11. SD3–SD6 cases have the same makespan of 340 s, which is as 62.96% as that of SG-SS. Their LP is 52.94% which is lower than SD1 and SD2 by 56.25%. For SD7–SD9, the makespan is 360 s, 66.67% of that in SG-SS. The LP is the lowest among all DG-SD cases which is 50% but still higher than SG-SS and SG-SD. Figure 10 summarizes all the SD cases involved in operating DGBC for three lifts compared with SG in single cycling and double cycling. Then we can quantify the performance of DGBC on the makespan and LP, both of which are significantly improved than SG.

6.4. Makespan Boundary. Although the bounds of the DGBC performance is not quantified, the time complexity of the heuristic is closely related to the conflicts type, number and position. However, the makespan of the DGBC problem can be bounded by comparing three extreme cases against SGs.

In Stage 2 of the proposed algorithm, the timetable of the DGBC problem is splitted into several parts; each of them has one type of conflicts. For example, in Figure 7, part 1 begins from the start of container 12 to the start of container 21. Since there is a SS conflict in part 1, it is related to DG-SS. Part 2 is from the start of container 21 to the start of container 16,

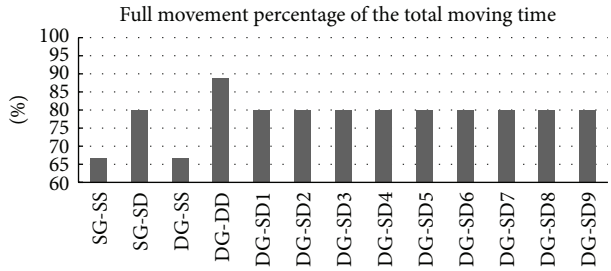


Figure 12: Full movement percentage of double cycling.

which belongs to DG-SD. Part 3 from the start of container 16 to the start of container 18 is related to DG-DD, and part 4 from the start of container 18 to the end corresponds to DG-SD. All the parts can be included in those three extreme cases; for example, both parts 1 and 4 are related to DG-SS. Therefore, the makespan of DGBC can be bounded by the performance of three extreme cases.

The comparisons show that DGBC outperforms SG on the three measures. MP of DGBC can be improved to 41.67% of DG-DD at the best case. The best LP can be obtained for the DG-DD as 80% while the lowest one (33.33%) may come from the DG-SS with $n_1 \neq n_2$. In both MP and LP, the scenario of $n_1 \neq n_2$ yields the larger maximal value and the smaller minimal value than the one with $n_1 = n_2$.

6.5. Double Cycling on DGBC. The effect of double cycling on reducing the empty movement and increasing the crane processing efficiency is well established for SG. As shown in Figure 11, the makespan of SG-SD is 480 s, which is 11.11% better than that of SG-SS with 540 s. Meanwhile, double cycling enhances the DGBC performance. According to Figure 10, the lower bound of the DG-DD makespan domain is less than that of DG-SS. For some instances, DG-DD can obtain better makespan than DG-SS does. In the case of $n_1 \neq n_2$, DG-DD outperforms DG-SS by 16.67% in makespan.

According to the full movement proportion of the total moving time, double cycling and single cycling are compared in Figure 12. Both DG-SS and SG-SS adopt single cycling as the scheduling strategy; the full movement takes 66.67% of the total moving time. Double cycling reduces empty movements and therefore yields more full movements in the crane operations; that is, the full movement proportion is raised from 66.67% (SG-SS and DG-SS) to 88.89% (DG-DD) which is improved by almost 33.3%. Moreover, all the DG-SD cases and SG-SD result in 80% full moving proportion. As shown in Figure 12, double cycling can significantly enhance the performance of DGBC by increasing the full movements.

However, the better makespan does not necessarily include the most double cycles, because the makespan domains of DG-SS and DG-DD are overlapped. Even in the case of the same number of double cycles, the efficiency of the double cycling is varied from the position of double cycles, such as SD1 and SD2 in Figure 11 which lead to the best makespan among all SD types.

Why does double cycling make less effective impact on the DGBC scheduling problem as it does for SG? Because the DGBC problem is sensitive to instances, and the conflicts are propagable. Double cycling only works in Stage 1 to get a good operating sequence for each single bay. An integrated timetable should be constructed for two bays in Stage 2, which has a larger effect on the makespan; a more compact timetable will enable the driver to handle the two spreaders more simultaneously and cooperatively. Double cycling reduces the number of operating cycle for each bay individually; new conflicts would exist for later containers. Therefore, the impact of double cycling is less significant on DGBC.

7. Conclusion

This paper describes how to implement DGBC and examines its performance. In addition to the reduction in cranes collisions, the crane travelling, and reposition cost, the crane serving efficiency can be improved significantly by DGBC with its capability to serve two adjacent bays simultaneously.

Based on the proposed two-stage heuristic algorithm, the makespan of the DGBC problem is bounded by three extreme cases (DG-SS, DG-DD, and DG-SD), and the best makespan takes 41.67% of that for SG-SS. LP is improved from 33.33% (SG-SS) to 66.67% (DG-SS), even 80% (DG-DD). As a result, the driver can perform the lifts more efficiently and productively.

In conjunction with double cycling, the makespan of the DGBC problem can be further improved with the full movement percentages increased to 66.67%, 80%, and 88.89% for DG-SS, DG-SD, and DG-DD, respectively, all of which are better than or equal to SG-SS with 66.67%. Therefore, the effectiveness of double cycling on DGBC is verified throughout the evaluation.

On a more ambitious scale, DGBC can be implemented in the rail-mounted container terminals. The horizontal movement of the driver will be taken into account in future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant 61272377) and the Specialized Research Fund for the Doctoral Program of Higher Education (20120092110027).

References

[1] A. V. Goodchild and C. F. Daganzo, "Double-cycling strategies for container ships and their effect on ship loading and unloading operations," *Transportation Science*, vol. 40, no. 4, pp. 473–483, 2006.

[2] S. Hartmann and D. Briskorn, "A survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 207, no. 1, pp. 1–14, 2010.

[3] R. Stahlbock and S. Voss, "Operations research at container terminals: a literature update," *OR Spectrum*, vol. 30, no. 1, pp. 1–52, 2008.

[4] C. F. Daganzo, "The crane scheduling problem," *Transportation Research Part B*, vol. 23, no. 3, pp. 159–175, 1989.

[5] C. Bierwirth and F. Meisel, "A survey of berth allocation and quay crane scheduling problems in container terminals," *European Journal of Operational Research*, vol. 202, no. 3, pp. 615–627, 2010.

[6] A. Lim, B. Rodrigues, F. Xiao, and Y. Zhu, "Crane scheduling with spatial constraints," *Naval Research Logistics*, vol. 51, no. 3, pp. 386–406, 2004.

[7] D. Lee, H. Q. Wang, and L. Miao, "Quay crane scheduling with non-interference constraints in port container terminals," *Transportation Research Part E: Logistics and Transportation Review*, vol. 44, no. 1, pp. 124–135, 2008.

[8] Z. Lu, X. Han, L. Xi, and A. L. Erera, "A heuristic for the quay crane scheduling problem based on contiguous bay crane operations," *Computers and Operations Research*, vol. 39, no. 12, pp. 2915–2928, 2012.

[9] F. Meisel and M. Wichmann, "Container sequencing for quay cranes with internal reshuffles," *OR Spectrum*, vol. 32, no. 3, pp. 569–591, 2010.

[10] H. Zhang and K. H. Kim, "Maximizing the number of dual-cycle operations of quay cranes in container terminals," *Computers & Industrial Engineering*, vol. 56, no. 3, pp. 979–992, 2009.

[11] P. Legato, R. Trunfio, and F. Meisel, "Modeling and solving rich quay crane scheduling problems," *Computers & Operations Research*, vol. 39, no. 9, pp. 2063–2078, 2012.

[12] S. Yuan, B. T. Skinner, S. Huang et al., "A job grouping approach for planning container transfers at automated seaport container terminals," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 413–426, 2011.

[13] L. Chen, A. Langevin, and Z. Lu, "Integrated scheduling of crane handling and truck transportation in a maritime container terminal," *European Journal of Operational Research*, vol. 225, no. 1, pp. 142–152, 2013.

[14] Y. Ding, T. Y. Gu, G. L. Lin, and C. J. Liang, "The establishment and solution of coupling model on coordinated scheduling of handling facilities in container terminals," *Applied Mathematics & Information Sciences*, vol. 6, no. 3, pp. 915–924, 2012.

[15] I. F. A. Vis and H. J. Carlo, "Sequencing two cooperating automated stacking cranes in a container terminal," *Transportation Science*, vol. 44, no. 2, pp. 169–182, 2010.

[16] L. Bianco, P. Dell'Olmo, and M. G. Speranza, "Heuristics for multimode scheduling problems with dedicated resources," *European Journal of Operational Research*, vol. 107, no. 2, pp. 260–271, 1998.

[17] C. Schwindt and N. Trautmann, "Scheduling the production of rolling ingots: industrial context, model, and solution method," *International Transactions in Operational Research*, vol. 10, no. 6, pp. 547–563, 2003.

[18] R. Čapek, P. Šůcha, and Z. Hanzálek, "Production scheduling with alternative process plans," *European Journal of Operational Research*, vol. 217, no. 2, pp. 300–311, 2012.

[19] U. Dorndorf, E. Pesch, and T. Phan-Huy, "Constraint propagation techniques for the disjunctive scheduling problem," *Artificial Intelligence*, vol. 122, no. 1-2, pp. 189–240, 2000.

[20] M. Mika, G. Waligóra, and J. Weglarz, "Tabu search for multimode resource-constrained project scheduling with schedule-dependent setup times," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1238–1250, 2008.

[21] D. Wang, X. Li, and Q. Wang, "A two-stage composite heuristic for dual cycling quay crane scheduling problem," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '11)*, pp. 1902–1907, Anchorage, Alaska, USA, October 2011.