Computers & Operations Research 39 (2012) 2915–2928

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A heuristic for the quay crane scheduling problem based on contiguous bay crane operations

Zhiqiang Lu [a], Xiaole Han [b,*], Lifeng Xi [b], Alan L. Erera [c]

[a] School of Mechanical Engineering, Tongji University, Shanghai 201804, PR China
[b] School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, PR China
[c] School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA

ARTICLE INFO

Available online 18 February 2012

Keywords:
Container terminal operation
Quay crane scheduling

ABSTRACT

Quay cranes (QC) are key resources at container terminals, and the efficiency of QC operations is vital for terminal productivity. The Quay Crane Scheduling Problem (QCSP) is to schedule the work activities for a set of cranes assigned to a single berthed vessel with the objective of minimizing the completion time of all container handling tasks. The problem is complicated by special characteristics of QC operations. Considering QC moving time and interference constraints, the concept of contiguous bay operations is proposed and a heuristic is developed to generate QC schedules with this feature. The heuristic is efficient and effective: it has polynomial computational complexity, and it produces schedules with a completion time objective bounded above by a small increment over the optimal completion time. Importantly, the heuristic guarantees that no quay cranes are idle due to interference. Numerical experiments demonstrate that the optimality gap is small for practical instances.

1. Introduction

Container seaports today are important nodes for global freight transportation. Terminals face increasing demand from container shipping customers, and often operate in fierce competition with alternative terminals. It is vital for a terminal to effectively utilize its resources such as the ship berthing areas, container yard, cranes, and manpower, to achieve high productivity and customer satisfaction. To help in this regard, operations research methods have been widely applied in terminal operation optimization [1,2].

Quay cranes (QC) are critical resources in container terminals, and their operating efficiency has direct and indirect impacts on the throughput of container terminal. Quay cranes are the most expensive equipment at terminals, and often represent the bottleneck on throughput. When a vessel is berthed, a certain number of QCs are designated to serve this vessel, i.e., to perform container discharging and loading operations. Containers to be handled are currently stacked or are to be stacked at different locations on vessel, and QCs move along the quay on rails to convey these containers [3].

The Quay Crane Scheduling Problem (QCSP) is to find a complete schedule for the QCs serving a vessel, and its decisions include assigning container handling tasks to specific QCs, sequencing these tasks on each QC, and specifying each task's operation start time. The QCSP can be viewed as an extension of the single stage parallel machine scheduling problem with positioned jobs (containers) and movable machines (QCs). The processing time of each job is the required container handling time, while the setup time between jobs is sequence-dependent and depends on the time required for the QC to travel from the predecessor job location. Furthermore, jobs have precedence relationships dictated by container stowage locations on the vessel, and machines must respect some spatial interference requirements with each other when performing jobs. In addition to these complexities, the large number of container jobs in practical QCSP instances also increases the difficulty of effectively solving the QCSP.

The rest of this paper is organized as follows: Section 2 reviews relevant literature on the QCSP. Section 3 describes the problem addressed in this paper. Specifically, the concepts of contiguous and non-contiguous QC schedules are defined, which describe different container-to-QC assignment patterns. Additionally, a decomposition of QC operating time into four parts is proposed, in connection with the spatial interference constraints. Based on the concepts defined in Section 3, Section 4 proposes a polynomial time complexity heuristic which generates a contiguous QC schedule. Section 5 proves the existence of a bounded objective function value gap between the generated contiguous schedule

* Corresponding author.
E-mail addresses: zhiqianglu.tju@gmail.com (Z. Lu), hanxl@sjtu.edu.cn (X. Han), lfxi@sjtu.edu.cn (L. Xi), alerera@isye.gatech.edu (A.L. Erera).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2012.02.013

and an optimal schedule. Section 6 demonstrates the size of this gap for a set of practical test instances, and discusses the effectiveness of the heuristic. Section 7 finally presents conclusions.

2. Literature review

Bierwirth and Meisel [4] provide a review and a classification scheme for QCSP, and summarize potential job discretization choices when modeling and solving the QCSP, such as bay area, single bay, container group, stack and single container. As the discretization becomes more detailed, the accuracy of the QC scheduling model increases at the expense of increased complexity and problem scale.

Daganzo [5] first studied the problem of QC assignment among different holds on multiple vessels to minimize total weighted completion time. A principle-based heuristic was proposed for the static problem where vessels arrive at time 0, and is then modified for the dynamic problem. Peterkofsky and Daganzo [6] further studied the static problem with the objective of minimizing total weighted tardiness, and use a branch-and-bound solution approach. These two studies do not consider QC interference, which is an important feature of QC operations. QCs move along shared quayside guiding rails and cannot pass one another (*non-crossing constraint*). Additionally, a safety distance must always be maintained between two adjacent QCs whenever they are handling containers or moving along the guiding rail (*safety distance constraint*).

Steenken et al. [7] included non-crossing constraints when assigning QCs to bays on vessels. By restricting QCs to contiguous bay areas, the problem was converted to a partition problem. Lim et al. [8] further considered the safety distance constraint for QCs and non-simultaneous constraints among bays. Jobs also refer to bay areas, but crane schedules are periodically updated given the dynamically changing environment. Thus for each period, there is no temporal component in their problem; instead the throughput of each possible crane-to-job assignment is given. The resulting crane-to-bay allocation problem is represented by a bipartite-graph matching problem with the objective of maximizing the total throughput. Dynamic programming, tabu search, as well as squeaky wheel optimization heuristics were adopted. Lim et al. [9] augmented this work by relaxing the contiguous bay area restriction and incorporating operation processing time. A job is defined to be all container handling work for a single ship bay, and the objective is to minimize the makespan of all jobs. By considering only non-crossing interference constraints, the problem becomes an NP-hard m-parallel machine scheduling problem. The paper proved that under any given bay-to-crane allocation, a crane schedule following a straightforward unidirectional operation mode is optimal. Thus there exists at least one unidirectional schedule which is globally optimal, and the decision space is reduced to bay-to-crane allocations. Several heuristics for the problem were presented. Lee et al. [10] extended these models by considering a handling priority for each vessel bay, and proposed a genetic algorithm to obtain near-optimal solutions. However, none of these four studies simultaneously considers non-crossing constraints, safety distance constraints, and a QC moving time model.

Liu et al. [11] studied the problem of QCs serving bays on multiple vessels to minimize the maximum relative tardiness of vessel departures. The QCSP was augmented to also select vessel berthing times given fixed berthing locations. The problem was decomposed into two levels. Each lower vessel-level problem is a QCSP with non-crossing constraints, safety distance constraints, and a QC moving time model. The objective of each QCSP is to minimize processing completion time given a fixed number of assigned QCs. The higher berth-level problem determines vessel berthing times and a QC-to-vessel assignment, based on the solutions to QCSPs at vessel-level. The authors proposed two types of QCSP models: one with preemptive operations where a QC can take over for another at a ship bay, and the other non-preemptive where each QC must fully process an assigned bay before moving to another. Computational results demonstrate the improvements possible by preemptive scheduling.

Ak [12] describes various configurations of QC operations when processing multiple vessels, i.e., QCs can be roaming or dedicated, shifting or blocking. A berth and quay crane scheduling problem (BQCSP) with roaming and shifting QCs was solved using a tabu search heuristic.

Ng and Mak [13] refine the job discretization by dividing the containers to be handled for each bay into two groups: discharging and loading. Furthermore, the time horizon is also discretized. The bays on vessel are partitioned into almost non-overlapping zones (only allowed to be overlapped at the boundary bays) which are assigned to different QCs. The unidirectional operation mode is proved to be optimal for any given partition, and dynamic programming is used to search for optimal partitions. However, no safety distance constraint is enforced.

Using instead the more detailed container group discretization, Bierwirth and Meisel [14] addressed some deficiencies in QC interference constraint models in earlier studies [15–17], and proposed a modified model. A fast branch-and-bound algorithm assuming the unidirectional operation mode was developed which outperforms other existing algorithms. Based on this algorithm, Meisel and Bierwirth [18] generated benchmarks for QCSP with various configurations and evaluated the effectiveness of the model assuming different job granularities (container group, bay, and bay area). The results suggest that the reduction in vessel processing time gained when solving more detailed models with smaller granularity might not be worth the increase in required computational effort.

For other discretization choices such as stack [19] and container [20], research focuses on QC scheduling for a single bay. Double cycling is adopted to enhance the handling efficiency.

In this paper, a QC schedule for an entire vessel is constructed with the objective of minimizing the makespan of all container discharging and loading jobs. The workload of each bay may be assigned to multiple QCs, and thus the bay handling operation is preemptive. QC moving time is explicitly modeled, and interference constraints are included. According to the scheme proposed in [4], the QCSP addressed in this paper can be written as $Bay,prmp|move|cross,save|max(compl)$. The purpose of this paper is to develop a method to quickly acquire a feasible QC schedule for such problem with some assured maximum optimality gap.

3. Problem description

Containers to be handled are distributed among different bays on a vessel. Before starting the loading or discharging operation of a given container job, the QC must be aligned to the job's bay location on the vessel, as shown in Fig. 1. Containers in a single bay can be assigned to multiple QCs, but only one QC can handle this bay at any time. By viewing each container as a container group, our problem formulation is similar to the model of [14], except that we ignore QC initial position and ready time. It is assumed in this paper that each QC is available from time 0 and no set-up time is required for its first assigned job, i.e., each QC can be initially aligned to the bay location of its first assigned container at time 0, as long as the safety distance is maintained. Such relaxation retains the major characteristics of QCSP, such as the precedence of container jobs, interference (non-crossing and safety distance) among QCs, sequence-dependent setup time, etc., and the problem remains NP-hard.

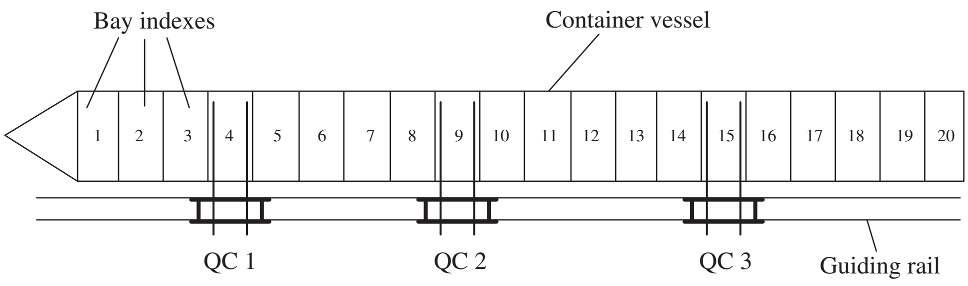


Fig. 1. Three QCs serving a vessel with 20 bays.

As shown in Fig. 1, the bay indexes can be used to indicate the spatial locations of QCs along the quayside. Without loss of generality, QC index increases in the same direction with bay index, which is from bow to stern of the vessel, and is also from left to right in all figures of this paper. The vessel superstructures can be viewed as empty virtual bays. The time of QC traveling one bay is assumed constant and is used as the temporal unit for scheduling. The following two sub-sections further describe the concept of a contiguous bay range and the decomposition of QC operating time, respectively.

3.1. Contiguous and noncontiguous bay ranges

To represent the job assignment among QCs, the following parameters are used:

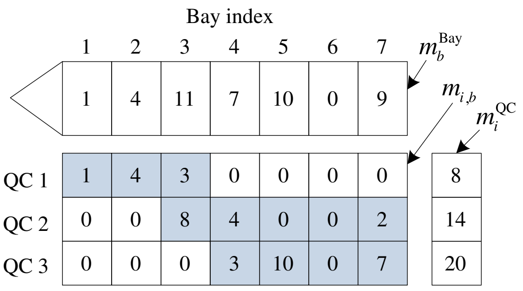
$j \in \mathbf{J} = \{1,\ldots,J\}$	set of container jobs
$i \in \mathbf{Q} = \{1,\ldots,Q\}$	set of QCs
$b \in \mathbf{B} = \{1,\ldots,B\}$	set of bays
B_j	Job j's bay location
$J_b^{\text{Bay}} \subseteq \mathbf{J}$	set of jobs in Bay b
m_b^{Bay}	number of all jobs to be handled in Bay b

Moreover, the following decision variables are used:

$J_{i,b} \subseteq \mathbf{J}$	set of jobs in Bay b that are assigned to QC i. Obviously, $\bigcup_i J_{i,b} = J_b^{\text{Bay}}$
$J_i = \bigcup_b J_{i,b}$	set of jobs that are assigned to QC i
$m_{i,b}$	number of jobs in $J_{i,b}$. Obviously, $\sum_i m_{i,b} = m_b^{\text{Bay}}$
$m_i^{\text{QC}} = \sum_b m_{i,b}$	number of all jobs assigned to QC i
$w_i^{\min} = \operatorname{argmin}_b m_{i,b} > 0$	smallest index of bay with jobs assigned to QC i
$w_i^{\max} = \operatorname{argmax}_b m_{i,b} > 0$	largest index of bay with jobs assigned to QC i
$R_i = \{w_i^{\min},\ldots,w_i^{\max}\}$	**Operating Bay Range** of QC i

Fig. 2 illustrates the $m_{i,b}$, m_i^{QC} and m_b^{Bay} values for a vessel with 7 bays and 3 QCs, and the operating bay ranges are illustrated by shaded grids. For example QC 2 is assigned 8 jobs from Bay 3, 4 jobs from Bay 4 and 2 jobs from Bay 7; 4 and 3 jobs of Bay 4 are assigned to QC 2 and QC 3, respectively; QC 3 is assigned 20 jobs in total; and Bay 7 contains 9 jobs. A bay is called a **Shared Bay** if its jobs are assigned to more than one QC, e.g., Bay 3, 4 and 7 in Fig. 2. $w_i^{\min}$ and $w_i^{\max}$ can be viewed, respectively as the lower and upper index boundary of QC i's operating involved bays. For example, $w_1^{\min} = 1$, $w_1^{\max} = 3$; $R_1 = \{1,2,3\}$, $R_2 = \{3,4,5,6,7\}$ and $R_3 = \{4,5,6,7\}$.

QC operating bay ranges can be categorized into two types: contiguous and noncontiguous ranges. Intuitively, a contiguous range contains no bays from the ranges of other QCs, except for the possibly shared boundary bays. A noncontiguous range contains at least one



Bay index	1	2	3	4	5	6	7	
	1	4	11	7	10	0	9	
QC 1	1	4	3	0	0	0	0	8
QC 2	0	0	8	4	0	0	2	14
QC 3	0	0	0	3	10	0	7	20

Fig. 2. A vessel's job numbers for bays and job assignments for QCs.

bay from the ranges of other QCs, aside from the possibly shared boundary bays. For example, in Fig. 2, R_2 is a noncontiguous range since it contains Bay 4, 5 and 6 which are also bays in R_3. R_3 is noncontiguous since it contains Bay 5 and 6 which are also bays in R_2. However R_1 is contiguous, since although it contains one bay (Bay 3) from another range R_2, this Bay 3 is its upper boundary. The detailed definitions of these concepts are as follows:

Definition 1. For any QC i with operating bay range $R_i = \{w_i^{\min},\ldots,w_i^{\max}\}$, if $\forall i' \in \mathbf{Q}\backslash\{i\}$ $w_i^{\min} \geq w_{i'}^{\max}$ or $w_{i'}^{\min} \geq w_i^{\max}$, R_i is a **Contiguous Range**; otherwise, R_i is a **Noncontiguous Contiguous Range**.

QC operating bay ranges reflect different job-to-QC assignment patterns, based on which the QC schedules can also be categorized into two types, as follows:

Definition 2. For a feasible QC schedule S, if the operating bay range R_i of each QC i is contiguous, then S is a **Contiguous Schedule**; otherwise, S is a **Noncontiguous Schedule.**

3.2. Active and idle QC times

Based on the job assignment among QCs, a complete schedule can be derived by further determining the job sequence and the processing start time on each QC. In other words, for any QC, given the set of its assigned jobs, its total operating time depends not only on its job sequence, but also on the operations conducted by other QCs, due to the existence of QC moving and interference constraints. Based on the characteristics of QC operation, each QC's operating time can be decomposed into two parts: the active time when it is handling some container or moving to change its bay location; and the idle time when it has to wait complying with interference constraints. Each part can be further decomposed into two components. To define these components, we first define the following parameter:

t_j^{fix}	fixed handling time of Container j by QC

Moreover, following decision variables are used:

t_j^{move}	variable lateral moving time of QC before handling Container j

T_i	total operating time of QC i with assigned jobs of J_i
T_i^{fix}	container **handling time** of QC i
T_i^{move}	lateral **moving time** of QC i
T_i^{conf}	**conflict idle time of** QC i
T_i^{bloc}	**blocking idle time** of QC i
T_i^{act}	active time of QC i

We have

$$T_i = T_i^{\text{fix}} + T_i^{\text{move}} + T_i^{\text{conf}} + T_i^{\text{bloc}} \tag{1}$$

The first part is the QC's actual handling time for its assigned containers. t_j^{fix} represents the QC component (including trolley and spreader) moving time between the vessel and internal trucks, in order to convey Container j. During this handling time the QC position on quayside is fixed and the QC does not move along the guiding rail. Thus t_j^{fix} is only determined by Container j's location on vessel and is independent of its operating order on QC i. Hence $T_i^{\text{fix}} = \sum_{j \in J_i} t_j^{\text{fix}}$ and is constant given J_i.

The second part is the QC's moving time between the handling of assigned containers. t_j^{move} represents QC i's lateral traveling time along the guiding rail in order to handle Container j. If the job that immediately precedes j, say, j', is in a different bay from j, then t_j^{move} takes a positive value, otherwise $t_j^{\text{move}} = 0$. Thus t_j^{move} is dependent on Container j's operating order on QC i. Hence $T_i^{\text{move}} = \sum_{j \in J_i} t_j^{\text{move}}$ and is variable under the given J_i.

Handling and moving time comprise QC i's active time under some schedule, i.e., $T_i^{\text{act}} = T_i^{\text{fix}} + T_i^{\text{move}}$. The active time would be the time required by the QC to serve its assigned containers, in the absence of the QC interference constraints.

The latter two parts of (1) comprise QC i's idle time of waiting caused by interference with other QCs. Conflict and blocking idle time are differentiated as they relate to safety distance (denoted by r in this paper, see Section 4) and non-crossing constraints, respectively. On one hand, when Job j is to be processed by QC i, it is possible that some bay indexed between $[B_j - r,\quad B_j + r]$ is being operated by another QC. Since QCs can not simultaneously operate jobs in bays within the safety distance, conflict occurs, and QC i must idle, thus T_i^{conf} is increased. On the other hand, when Job j is to be processed by QC i, it is also possible that some QC is operating a third bay between the current bay of QC i and the bay indexed by B_j. Since QCs cannot cross each other, blocking occurs, and again QC i must idle, thus T_i^{bloc} is increased.

As a simple illustration, two QC operating schedules are acquired from the job assignment in Fig. 2, and their Gantt charts are provided in Fig. 3. In the Gantt charts, each bar corresponds to a grid in Fig. 2 and is labeled with the number of its assigned jobs. Here, we suppose each container job requires the same handling time, which also equals the QC moving time between adjacent bays. The schedule (a) is an infeasible schedule that ignores QC interference constraints, while the schedule (b) is feasible by fixing those QC blocking and conflicts. Schedule (b)'s makespan increases from schedule (a) as the result of respecting such interference

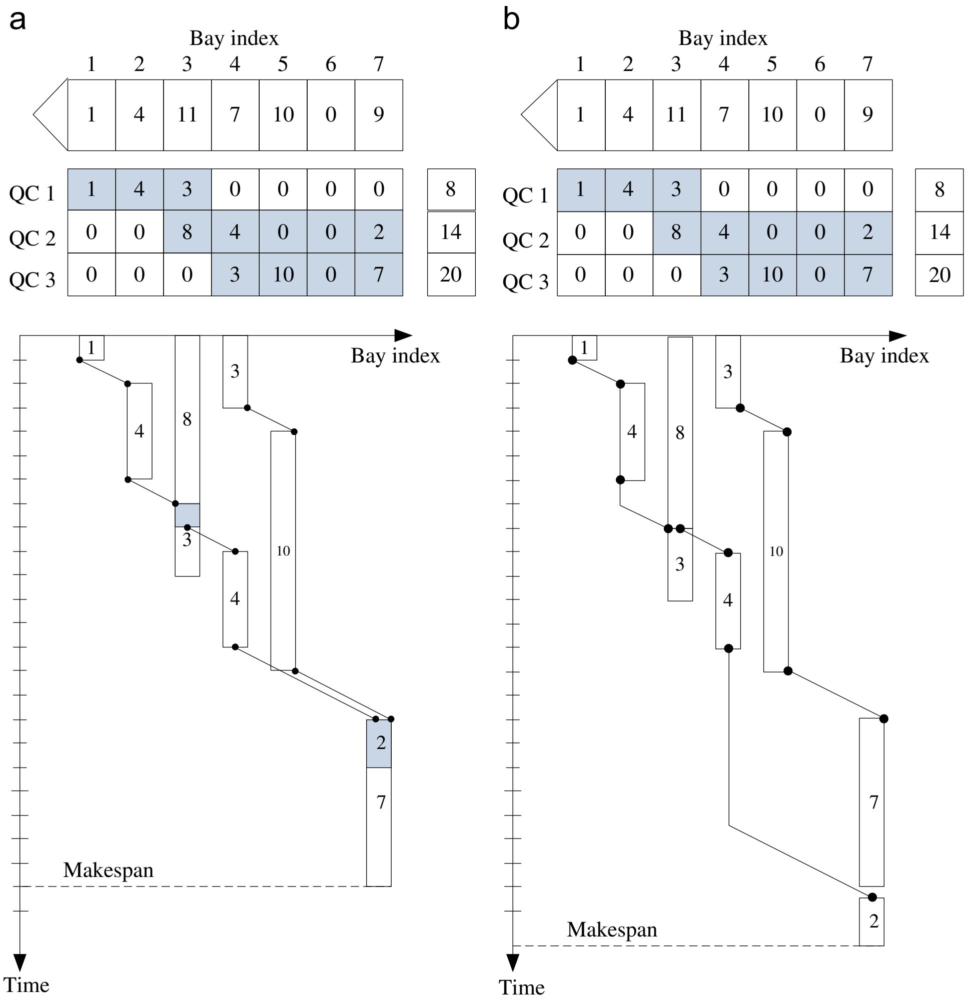


Fig. 3. Gantt charts for two schedules (a) ignoring and (b) considering interference constraints.

constraints. The two kinds of interference constraints have different impact on QCs of different operating range types. A QC with a noncontiguous range may get blocked by other QCs, since its operating range overlaps with other QCs' ranges, while a QC with a contiguous range is free from blocking constraints, since contiguous ranges exclude the necessary condition for QC blocking: the spatial inclusion of another QC's operating bay. Conflict can happen to both QCs with contiguous and noncontiguous ranges. For instance, in Fig. 3(b), QC 2 is blocked by QC 3 after finishing handling Bay 4, and conflicts with QC 3 for handling Bay 7.

4. Heuristic for job assignment and sequencing

In this section, a polynomial time heuristic procedure is proposed to assign and sequence jobs on QCs. The generated schedule S^c is contiguous and follows a unidirectional operating mode. The schedule S^c does not explicitly specify job starting times; thus, it is not initially clear whether any scheduled crane incurs interference time beyond its active time. However it will be proved in Section 5 that QC interference would not exist if each QC independently operates its assigned and sequenced jobs according to the generated S^c.

For the vessel to be handled, the following parameters are defined:

b_L	smallest index of bay with jobs to be handled, i.e., the index of the left-most non-empty bay on vessel. $b_L = \underset{b}{\arg\min}\, m_b^{\text{Bay}} > 0$.
b_R	largest index of bay with jobs to be handled, i.e., the index of the right-most non-empty bay on vessel. $b_R = \underset{b}{\arg\max}\, m_b^{\text{Bay}} > 0$.
σ_b	difference of b and the index of Bay b's nearest left-sided non-empty bay. It equals the required QC moving time from Bay b's nearest left-sided non-empty bay to Bay b. Note that $\sigma_b = 0, \forall b \le b_L$.
σ'_b	the $\sigma_b$value for non-empty Bay b. $\sigma'_b = 0$, $\forall b \mid m_b^{\text{Bay}} = 0$; $\sigma'_b = \sigma_b$, otherwise.
δ	maximum σ_b for all b.
φ	sum of the largest $(Q-1)$ σ_b values among all b.
φ'	sum of the largest $(Q-1)$ σ'_b values among all b.
r	safety distance measured in bays. For two QCs located at Bay b_1 and Bay b_2, such safety distance is maintained if and only if $|b_1 - b_2| \ge r+1$.
$B_{\max}^{r+1}$	maximum active time needed to process any $r+1$ consecutive bays.

To illustrate the defined terms, consider a vessel with 20 bays. The workload distribution among bays is shown as in Fig. 4. According to the definition, $b_L = 3$, $b_R = 19$. Besides, $\{\sigma_b\} = \{0,0,0,1,1,1,2,1,1,1,2,3,4,1,1,2,1,2,3,1\}$, $\{\sigma'_b\} = \{0,0,0,1,1,0,2,1,1,0,0,0,4,1,0,2,0,0,3,0\}$. For instance, $\sigma_6 = 1$ since Bay 6's nearest left-sided non-empty bay is Bay 5; $\sigma_7 = 2$ since Bay 7's nearest left-sided non-empty bay is Bay 5 as well; $\sigma'_6 = 0$ since no workload is required on Bay 6; $\sigma'_7 = \sigma_7 = 2$ since non-zero workload is required on Bay 7. $\delta = \sigma_{13} = 4$, which is the largest among all σ_b values. Thus $\varphi = 4$, 7, 10, 12, 14, and $\varphi' = 4$, 7, 9, 11, 12, when Q=2, 3, 4, 5, 6, respectively. For instance, when Q=4, since the largest three σ_b values are 4, 3 and 3, $\varphi = 4+3+3 = 10$; and the largest three σ'_b values are 4, 3 and 2, $\varphi' = 4+3+2 = 9$.

For any feasible schedule, the following decision variables are defined:

$C_{\max}$	makespan value of vessel processing.
T_{sum}	total operating time of all QCs. $T_{\text{sum}} = \sum_i T_i$.
$T_{\text{sum}}^{\text{act}}$	total active time of QCs. $T_{\text{sum}}^{\text{act}} = \sum_i T_i^{\text{act}}$.
$T_{\text{avg}}^{\text{act}}$	average active time of all QCs. $T_{\text{avg}}^{\text{act}} = T_{\text{sum}}^{\text{act}}/Q$.
$T_{\text{sum}}^{\text{move}}$, $T_{\text{avg}}^{\text{move}}$, $T_{\text{sum}}^{\text{fix}}$, $T_{\text{avg}}^{\text{fix}}$	can be defined similarly. Note that the latter two are constant for any feasible schedule.
$L_{\text{avg}}^{\text{act}}$	minimum required average active time of all QCs.

First, $T_{\text{sum}}^{\text{move}}$, the total moving time of all QCs, has a lower bound of $b_R - b_L - \varphi$. This can be proved through the following Lemmas 1–3.

For any feasible schedule (not necessarily contiguous), suppose the number of utilized QCs is $Q'(Q' \le Q)$. Without loss of generality, re-index these Q' QCs from 1 to Q', in non-decreasing order of $w_i^{\min}, i \in \{1,\dots,Q'\}$. Since QCs can not get across each other, the $w_i^{\max}$ values are thus also non-decreasing ordered for $i \in \{1,\dots,Q'\}$. As a result, $w_1^{\min} \le b_L$ and $w_{Q'}^{\max} \ge b_R$. We have:

Lemma 1. If $\exists i \in \{2,\dots,Q'\}, w_{i-1}^{\max} < w_i^{\min} - 1 \Rightarrow m_b^{\text{Bay}} = 0, \forall b \in \{w_{i-1}^{\max} + 1,\dots,w_i^{\min} - 1\}$

Proof. Suppose given the conditions, there exists some b for which Bay b is non-empty. Then there should be at least one QC, say, QC k, for which $w_k^{\min} \le b \le w_k^{\max}$, so that the tasks in Bay b can be handled. Two cases are differentiated:

Case 1 $k \le i-1$. Since $b \ge w_{i-1}^{\max} + 1 > w_{i-1}^{\max} \ge w_k^{\max}$, such case does not exist.
Case 2 $k \ge i$. Since $b \le w_i^{\min} - 1 < w_i^{\min} \le w_k^{\min}$ such case does not exist either.

As a result, contradiction exists. □

This lemma implies that for any two adjacent QCs, if their bay ranges are not overlapped, then all bays in between their bay ranges'

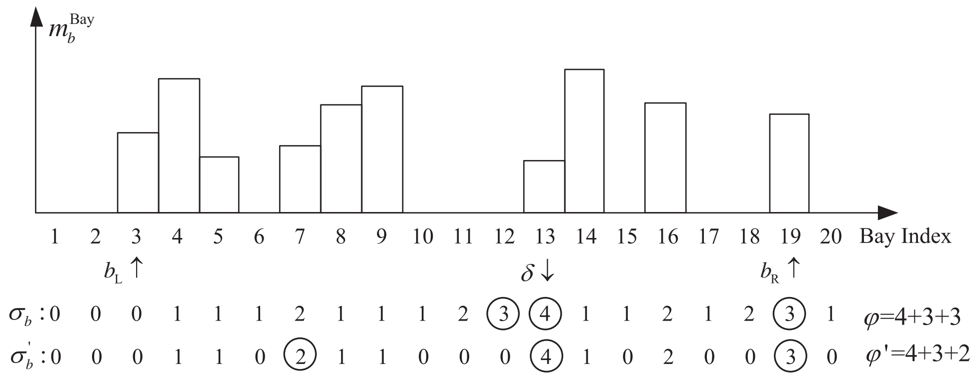


Fig. 4. Illustration of σ_b, δ and φ concepts.

gap should be empty. Note that this is applicable to both contiguous and non-contiguous schedules. This leads to the following:

Lemma 2. $\sum_{i\in\{2,\ldots,Q'\}}(w_i^{\min}-w_{i-1}^{\max})\le\varphi$

Proof. As shown in Fig. 4, whenever $\sigma'_b\ge 2$, it represents the number of some contiguous empty bays, or the length of this empty bay segment. Thus φ' is no smaller than the sum of the $Q-1$ longest empty bay segments on vessel. Additionally, $\varphi'\le\varphi$, since $\sigma'_b\le\sigma_b$.

$\sum_{i\in\{2,\ldots,Q'\}}(w_i^{\min}-w_{i-1}^{\max})$ is the sum of $Q'-1$ bay range gaps, and it is maximized when these $Q'-1$ bay range gaps are exactly the $Q'-1$ longest empty bay segments on vessel. Since$Q'\le Q$, $\sum_{i\in\{2,\ldots,Q'\}}(w_i^{\min}-w_{i-1}^{\max})\le\varphi'\le\varphi$. □

Thus we have

Lemma 3. $T_{\text{sum}}^{\text{move}}\ge b_{\text{R}}-b_{\text{L}}-\varphi$

Proof.

$$T_{\text{sum}}^{\text{move}}=\sum_{i\in\{1,\ldots,Q'\}}T_i^{\text{move}}(J_i)=\sum_{i\in\{1,\ldots,Q'\}}(w_i^{\max}-w_i^{\min})$$
$$=-w_1^{\min}+\sum_{i\in\{2,\ldots,Q'\}}(w_{i-1}^{\max}-w_i^{\min})+w_{Q'}^{\max}\ge -b_{\text{L}}-\varphi+b_{\text{R}}.\quad\square$$

Lemma 3 proves the lower bound of $T_{\text{sum}}^{\text{move}}$ as $b_{\text{R}}-b_{\text{L}}-\varphi$. Consider the example in Fig. 4 when $Q=4$. Since $\varphi=10$, the lower bound is $b_R-b_L-\varphi=19-3-10=6$. This lower bound value can be approached when $R_1=\{3,4,5\}$, $R_2=\{7,8,9\}$, $R_3=\{13,14,15,16\}$ and $R_4=\{19\}$. The moving times for each QC are 2, 2, 3 and 0, respectively.

Consider $L_{\text{avg}}^{\text{act}}$, the minimum required average active time of all QCs. Since $T_{\text{sum}}^{\text{fix}}+(b_{\text{R}}-b_{\text{L}}-\varphi)$ is the minimum required total handling and moving time of all QCs, we have $L_{\text{avg}}^{\text{act}}=T_{\text{avg}}^{\text{fix}}+(b_{\text{R}}-b_{\text{L}}-\varphi)/Q$. $L_{\text{avg}}^{\text{act}}$ is constant for any feasible schedule and is a lower bound of $C_{\max}$. Thus

$$L_{\text{avg}}^{\text{act}}\le C_{\max} \tag{2}$$

Since at any time there can be at most one QC processing any consecutive $r+1$ bays, $B_{\max}^{r+1}$ is another lower bound of $C_{\max}$. Thus

$$B_{\max}^{r+1}\le C_{\max} \tag{3}$$

By using a combined lower bound $LB=\max\{L_{\text{avg}}^{\text{act}},B_{\max}^{r+1}\}$, the detailed heuristic procedure is provided as follows:

```
Heuristic Procedure:
    1. Sequence all jobs in J:
        For each Bay $b$, and any two jobs $j_1$ and $j_2$ in $J_b^{\text{Bay}}$, if
        precedence constraint exists between $j_1$ and $j_2$, arrange $j_1$
        prior to $j_2$, i.e., $j_1\to j_2$; otherwise, ties are broken randomly.
        Denote $Seq_b$ the resulted sequence. Denote $Seq_b^{\text{R}}$ the
        reversed sequence of $Seq_b$.
        Order J as $\{Seq_1^{\text{R}}\to Seq_2^{\text{R}}\cdots\to Seq_b^{\text{R}}\cdots\to Seq_B^{\text{R}}\}$.
    2. Partition J to $J_i$:
        Set $J_i=\emptyset,\forall i=1,\ldots,Q$; Set $i=1, j=1$.
        For $j=1$ to $J$
            IF $i<Q$
                IF $T_i^{\text{act}}(J_i)\le LB+\delta$
                    $J_i=J_i+\{j\}$
                Else
                    $i$++
                    $J_i=J_i+\{j\}$
                EndIF
            Else
                $J_i=J_i+\{j\}$
            End IF
        EndFor
    3. Re-Sequence jobs in each $J_i$:
        Order assigned jobs in each $J_i$ in accordance with
        $Seq_{w_i^{\min}}\to\cdots\to Seq_{w_i^{\max}}$.
```

The main idea of the proposed heuristic procedure is to extract one by one each element (job) from the original job set **J** and to assign it into some QC i's job set J_i, and to determine the operating sequence of jobs assigned to each QC i. First, the sequence of all the jobs in **J** is arranged: for two jobs from different bays, the one with smaller bay index precedes the other one with larger bay index; for two jobs from the same bay, their precedence constraint, if exist, is reversed for subsequent partition sub-procedure. Second, following this sequence, jobs will be taken out of **J** one by one, and appended to the end of some QC's operating sequence. Specifically, jobs taken out of **J** are initially assigned to QC 1, and QC 1's required active time for already assigned jobs is updated every time a new job is appended to its operating sequence. We stop assigning jobs to QC 1 once its active time exceeds $LB+\delta$, and start assigning jobs to QC 2. Such process continues for QC 3, QC 4 … etc, until all jobs are assigned to some QC. In this way, if each QC i handles its assigned jobs following their order of entering J_i, each QC will have a contiguous operating range and moves within this range unidirectionally from bow to stern of the vessel, as illustrated in Fig. 5. However, every inner-bay job precedence constraint will be violated, as in the arranged job sequence of **J**. Hence lastly, after all jobs have been assigned to QCs, each QC's operating sequence is further adjusted to respect inner-bay job precedence constraints. Such inner-bay operating sequence adjustments will not change the required active time of each QC. The complexity of the heuristic procedure is $O(J)$.

Obviously, in the generated S^{c}, QCs' workloads may not be precisely balanced for two reasons: (1) Since after i reaches Q, extracted jobs will all be assigned to QC Q no matter whether $LB+\delta$ is reached or not, it is possible that QC Q is relatively overloaded compared with other QCs. (2) It is also possible that some QCs will be assigned no jobs, since the procedure may terminate before i reaches Q. However, the objective of our procedure is not to optimally balance workloads, but to ensure a bounded gap of makespan between the acquired feasible solution and the theoretic optimal schedule. It will be proved in Section 5 that such a bound exists under all circumstances (Theorem 1). Specifically, each QC's active time is bounded regardless of the workload balance of QCs (Formula (4) and Lemma 4).

Note that the heuristic only explicitly acquires job assignments and sequences on QCs. The detailed job starting times should be further determined considering possible interference among adjacent QCs, as illustrated in Fig. 3. Since in S^{c} all QCs move unidirectionally, whenever two QCs are conflicting with or blocked by each other, the only resolution is to prioritize the job on the right-sided QC, and to add conflict or blocking idle time to the left-sided QC's schedule. However such resolution will be proved unnecessary in Section 5 since no QC interference will

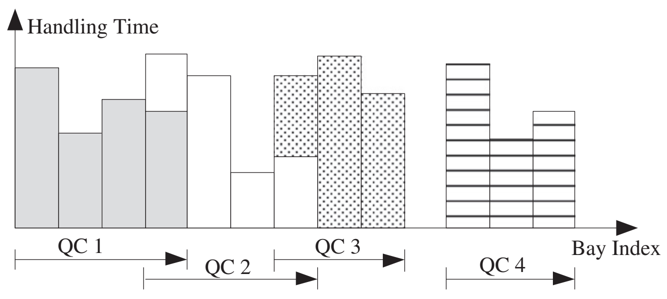


Fig. 5. QC operating ranges and moving directions of S^{c} acquired by heuristic.

occur, given the job assignments and sequences in S^c (Lemma 5–8).

5. Proof of a bounded objective gap

The logic of this section is: Formula (4) and Lemma 4 together ensure that the active time of any QC is bounded. It is further proved by Lemma 5–8 that no blocking or conflict idle time exists, i.e., the makespan value of each QC equals its active time, and thus is bounded as well. Hence Theorem 1 can be derived that the makespan value of vessel processing is bounded.

Denote $Q'(Q' \le Q)$ as the QC index to which the last job of $\mathbf{J}$ is assigned, and thus QC $Q'+1,\ldots,$QC Q are assigned no jobs.

Since $\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}$ is an upper bound of QC active time required for processing any single job j, the heuristic procedure itself ensures that for S^c:

$$LB+\delta < T_i^{\text{act}}(J_i) \le LB+2\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}, \quad \forall i=1,\ldots,Q'-1, \tag{4}$$

Lemma 4. For S^c, $T_{Q'}^{\text{act}}(J_{Q'}) \le LB+2\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}$

Proof. Suppose $T_{Q'}^{\text{act}}(J_{Q'}) > LB+2\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}$, two cases can be differentiated:

Case 1 $Q'<Q$

Denote j the last job in $J_{Q'}$. Given the hypothesis, $T_{Q'}^{\text{act}}(J_{Q'}-\{j\}) > LB+\delta$, thus according to the heuristic procedure, Job j will be assigned to QC $Q'+1$ since $Q'+1 \le Q$. Therefore this case does not exist.

Case 2 $Q'=Q$

We have:

$$\sum_{i=1,\ldots,Q} T_i^{\text{act}}(J_i) = \sum_{i=1,\ldots,Q} [T_i^{\text{fix}}(J_i)+T_i^{\text{move}}(J_i)]$$
$$= \sum_{i=1,\ldots,Q} T_i^{\text{fix}}(J_i)+\sum_{i=1,\ldots,Q} T_i^{\text{move}}(J_i) = T_{\text{sum}}^{\text{fix}}+\sum_{i=1,\ldots,Q} T_i^{\text{move}}(J_i)$$

Given the hypothesis, we also have:

$$\sum_{i=1,\ldots,Q} T_i^{\text{act}}(J_i) > (Q-1)\cdot(LB+\delta)+(LB+2\delta) \ge Q\cdot L_{\text{avg}}^{\text{act}}+(Q+1)\cdot\delta$$
$$= Q\cdot T_{\text{avg}}^{\text{fix}}+(b_R-b_L-\varphi)+(Q+1)\cdot\delta$$
$$= T_{\text{sum}}^{\text{fix}}+(b_R-b_L-\varphi)+(Q+1)\cdot\delta \ge T_{\text{sum}}^{\text{fix}}+b_R-b_L+2\delta$$

Thus $\sum_{i=1,\ldots,Q} T_i^{\text{move}}(J_i) > (b_R-b_L+2\delta)$. But since S^c is contiguous and unidirectional, b_R-b_L is the maximum required total moving time for all QCs in S^c. Therefore a contradiction exists. □

Lemma 5. For S^c, $w_i^{\max}-w_i^{\min} \ge r+1$, $\forall i=1,\ldots,Q'-1$.

Proof. Suppose $w_i^{\max}-w_i^{\min} \le r$ for some $i\in[1,Q'-1]$, thus the number of bays contained in QC i's operating range is no more than $r+1$. Hence $T_i^{\text{act}}(J_i) \le B_{\max}^{r+1} \le LB$ for such $i\in[1,Q'-1]$, which violates (4). □

Lemma 6. For S^c, $w_{i+1}^{\min}-w_i^{\min} \ge r+1$, $\forall i=1,\ldots,Q'-1$.

Proof. $\forall i=1,\ldots,Q'-1$, since $w_{i+1}^{\min} \ge w_i^{\max}$, from Lemma 5, $w_{i+1}^{\min}-w_i^{\min} \ge w_i^{\max}-w_i^{\min} \ge r+1$. □

As assumed in Section 3, at time 0, each QC can reach its initial location which is aligned to the bay location of its first assigned container job, as long as the safety distance is maintained. Lemma 6 suggests that in S^c, such safety distance is ensured, thus QCs can immediately start processing their first assigned container jobs at time 0.

Lemma 7. S^c contains no blocking idle time.

Proof. S^c is a contiguous schedule since each QC i of S^c has contiguous bay range as depicted in Fig. 5. Thus operating time $T_i(J_i)$ contains no blocking idle time, and T_{sum} contains no blocking idle time as well. □

Lemma 8. S^c contains no conflict idle time.

Proof. Each QC i $(i=1,\ldots,Q')$ of S^c has a contiguous bay range with boundary bays that are possibly shared by adjacent QCs: QC $i-1$ and QC $i+1$. Note that for QC 1, its only adjacent QC is QC 2, i.e., the lower boundary bay of QC 1 is not a shared bay. And similarly, for QC Q', its only adjacent QC is QC $Q'-1$, and the upper boundary bay of QC Q' is not a shared bay. So for S^c there can be at most $Q'-1$ shared bays.

Suppose conflict idle time exists in S^c. Since S^c is contiguous and QCs move unidirectionally, such conflict idle time occurs only when QCs are operating near their boundary bays and the minimum safety margin is reached. Precisely, the necessary condition for nonzero conflict idle time is that there exist two adjacent QC $i-1$ and QC i, and QC $i-1$ reaches some Bay b' before QC i reaches Bay $b'+r+1$. Consider the largest $i\in[2,Q']$ for which QC $i-1$ has conflict idle time, thus $T_{i'}^{\text{conf}}(J_{i'})=0$, $\forall i'\in[i,Q']$, i.e., there is no conflict idle time for QC $i,\ldots,Q'$. Depending on different boundary modalities between QC $i-1$ and i, three scenarios are differentiated and discussed, respectively, each of which contains a contradiction and therefore this lemma can be proved.

Scenario 1 $w_{i-1}^{\max}=w_i^{\min}$

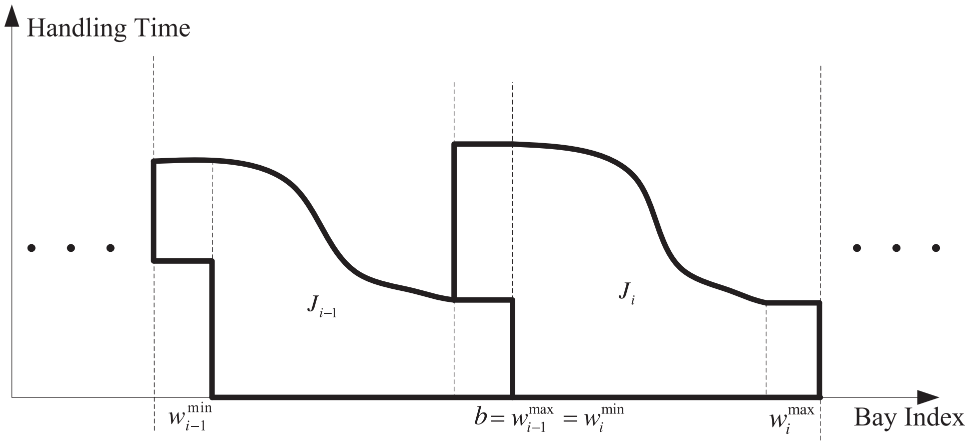


Fig. 6. Illustration of container assignment and boundary modality for Scenario 1.

The container assignment and boundary modality between the two QCs are illustrated in Fig. 6. Bay b is the bay shared by QC $i-1$ and QC i, and $w_{i-1}^{\max}=w_i^{\min}=b$. The safety distance constraint applies only when at least one QC is located within the open interval between Bay $b-r-1$ and Bay $b+r+1$. Precisely, for QC $i-1$ to have conflict idle time, there must exist some $x\in[1,r+2]$ for which QC $i-1$ reaches Bay $b-x+1$ before QC i reaches Bay $b-x+r+2$. Consider the largest $x\in[1,r+2]$ which causes the conflict idle time, thus QC $i-1$ does not need conflict idle time when processing bays indexed from $w_{i-1}^{\min}$ to $b-x$.

If $x=r+2$, given the condition of conflict idle time occurrence, QC $i-1$ has to reach Bay $b-r-1$ before QC i reaches Bay b. Besides, from Lemma 5, $w_{i-1}^{\min}\le w_{i-1}^{\max}-r-1=b-r-1$, which means QC $i-1$ reaches Bay $w_{i-1}^{\min}$ no later than it reaches Bay $b-r-1$. Thus in order to incur conflict, QC $i-1$ has to reach Bay $w_{i-1}^{\min}$ before QC i reaches Bay $w_i^{\min}$, which is infeasible since Lemma 6 ensures that both QC $i-1$ and QC i can reach their first assigned jobs at time 0. Therefore we only need to consider $x\in[1,r+1]$ in the rest part of this Scenario 1.

Based on the x value ($x\in[1,r+1]$), jobs assigned to these two QCs can be partitioned into at most six sets. Fig. 7 illustrates such decomposition of QC $i-1$ and QC i's operating ranges for a given $x\in[1,r+1]$. The six sets are labeled as A–F. A–C comprise jobs assigned to QC $i-1$, and D–F comprise jobs assigned to QC i. Jobs of A are those on bays indexed no more than $b-x$, jobs of B are on bays indexed from $b-x+1$ to $b-1$, jobs of C and D are on the shared Bay b, jobs of E are on bays indexed from $b+1$ to $b-x+r+1$, jobs of F are on bays indexed no less than $b-x+r+2$.

Note that Fig. 7 represents a general decomposition and some set may actually be null or empty. A set is **null** if it does not exist, i.e., the set contains no bays at all. A set is **empty** if it exists but all bays in the set contain no jobs to be handled. By definition, each of Bay $w_{i-1}^{\min}$, Bay $w_i^{\max}$ and Bay b contains at least one job to be handled, and Set C and D are not null or empty.

From Lemma 6, $\forall i=2,\ldots Q'$, $w_{i-1}^{\min}\le w_i^{\min}-r-1=b-r-1\le b-x$, i.e., Set A contains at least one bay (Bay $w_{i-1}^{\min}$) and is not null or empty.

From Lemma 5, $\forall i=2,\ldots Q'-1$, $w_i^{\max}\ge w_i^{\min}+r+1=b+r+1\ge b-x+r+2$, i.e., Set F contains at least one bay (Bay $w_i^{\max}$) and is not null or empty when $i<Q'$. Set F may be null when $i=Q'$, but is not empty as long as it exists.

Set B, E may be null. For example Set B is null when $x=1$, Set E is null when $x=r+1$, and both Set B and E are null when $r=0$. Besides, even though exists, Set B or E may be empty.

From this perspective, six scenarios are further differentiated:

Scenario 1.1 Set F is null and Set E is null

Under this scenario, $i=Q'$, $x=r+1$ and $w_{Q'}^{\max}=b$. Fig. 7 transforms to Fig. 8. Since Bay $w_{Q'-1}^{\min}$ is not empty, $\sigma_{b-r}\ge 1$. Given the condition of conflict idle time occurrence, QC $Q'-1$ reaches Bay $b-r$ before QC Q' reaches Bay $b+1$. we have:

$$T_{Q'-1}^{\text{act}}(A)+\sigma_{b-r}<T_{Q'}^{\text{act}}(D)+1=T^{\text{fix}}(D)+1$$

Note that Bay $b+1$ may be an empty bay on vessel, or even an virtual bay outside the vessel range that is only used to locate QC position. In this way, even after finishing Bay b on vessel, QC Q' continues to move in order to maintain the safety distance from QC $Q'-1$.

If B is null or empty, $\delta\ge\sigma_b=\sigma_{b-r}+r\ge r+1$. We have

$$\begin{aligned}B_{\max}^{r+1}&\ge T_{Q'}^{\text{act}}(B\cup C\cup D)=T^{\text{fix}}(C)+T^{\text{fix}}(D)>T^{\text{fix}}(C)+(T_{Q'-1}^{\text{act}}(A)+\sigma_{b-r}-1)\\&=(T_{Q'-1}^{\text{act}}(A)+T^{\text{fix}}(C)+\sigma_b)-\sigma_b+\sigma_{b-r}-1=T_{Q'-1}^{\text{act}}(A\cup B\cup C)-r-1\\&>(LB+\delta)-r-1\ge(B_{\max}^{r+1}+\delta)-r-1=B_{\max}^{r+1}+(\delta-r-1)\ge B_{\max}^{r+1}\end{aligned}$$

Otherwise, B is not empty, we have

$$\begin{aligned}B_{\max}^{r+1}&\ge T_{Q'}^{\text{act}}(B\cup C\cup D)=T_{Q'}^{\text{act}}(B\cup C)+T^{\text{fix}}(D)=T_{Q'-1}^{\text{act}}(B\cup C)+T^{fix}(D)\\&>T_{Q'-1}^{\text{act}}(B\cup C)+(T_{Q'-1}^{\text{act}}(A)+\sigma_{b-r}-1)\\&=(T_{Q'-1}^{\text{act}}(A)+T_{Q'-1}^{\text{act}}(B\cup C))+\sigma_{b-r}-1\\&\ge(T_{Q'-1}^{\text{act}}(A\cup B\cup C)-\delta)+\sigma_{b-r}-1>(LB+\delta)-\delta+\sigma_{b-r}-1\\&\ge(B_{\max}^{r+1}+\delta)-\delta+\sigma_{b-r}-1=B_{\max}^{r+1}+(\sigma_{b-r}-1)\ge B_{\max}^{r+1}\end{aligned}$$

Contradiction exists in both cases under this scenario.

The proofs of the rest 5 sub-scenarios (Scenario 1.2–1.6) given $w_{i-1}^{\max}=w_i^{\min}$ are provided in Online Appendix. Moreover, the

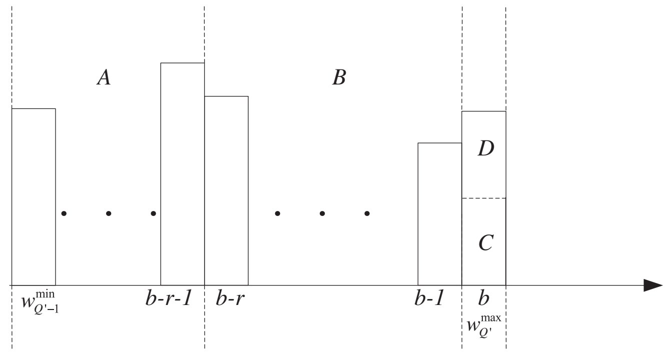


Fig. 8. Decomposition of operating ranges of QC $i-1$ and QC i under Scenario 1.1.

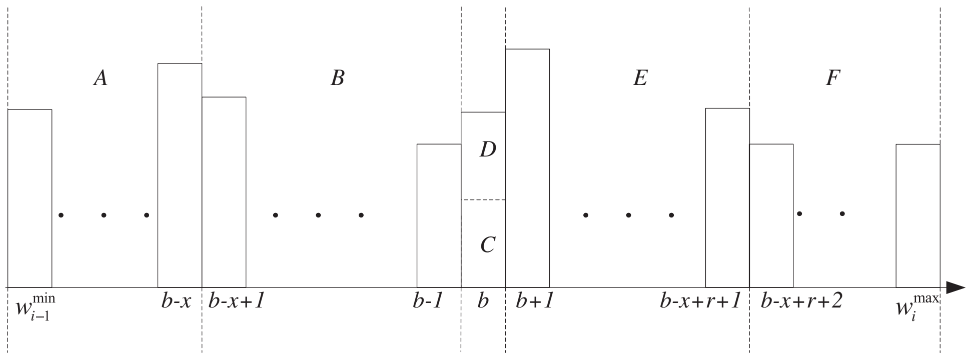


Fig. 7. Decomposition of operating ranges of QC $i-1$ and QC i under Scenario 1.

other two scenarios ($w_{i-1}^{\max} = w_i^{\min} - 1$, $w_{i-1}^{\max} < w_i^{\min} - 1$) can be analyzed for contradiction similarly, where 12 additional sub-scenarios will be differentiated (Scenario 2.1–2.6; Scenario 3.1–3.6). The proofs of these two scenarios (i.e., 12 sub-scenarios) are also provided in Online Appendix. There are similarities between the proofs of some scenarios, and some other scenarios are proved infeasible. Fig. 9 summarizes these internal relations. Finally, after identifying contradiction for each 18 possible scenarios given that QC conflict idle time exists for S^c, Lemma 8 is proved. □

Theorem 1. $C_{\max}(S^c) \le C^*_{\max} + 2\delta + \max_{j \in \mathbf{J}} t_j^{\text{fix}}$, where $C^*_{\max}$ is the makespan of optimal schedule.

Proof. Based on (2), (3), $LB = \max\{L_{\text{avg}}^{\text{act}}, B_{\max}^{r+1}\} \le C^*_{\max}$; based on Lemma 7 and 8, $T_i(J_i) = T_i^{\text{act}}(J_i)$, $\forall i = 1,\ldots,Q'$; then according to (4) and Lemma 4, we have

$$C_{\max}(S^c) = \max_{i \in \mathbf{Q}}\{T_i(J_i)\} = \max_{i=1,\ldots,Q'}\{T_i^{\text{act}}(J_i)\}$$
$$\le LB + 2\delta + \max_{j \in \mathbf{J}} t_j^{\text{fix}} \le C^*_{\max} + 2\delta + \max_{j \in \mathbf{J}} t_j^{\text{fix}} \quad \square$$

Example 1. For this example, we consider a case of scheduling a ship with very large task volume variance between adjacent bays. To simplify the numerical expression, let the fixed handling time of each container equal 1 unit of time, which is also the QC moving time between two adjacent bays. We have:

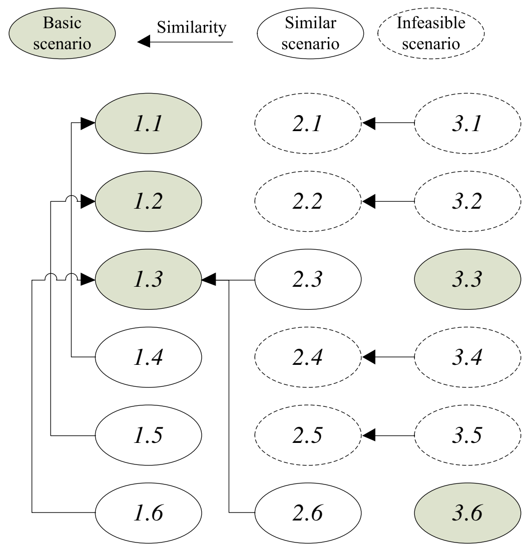


Fig. 9. Internal relations among proofs of different scenarios for Lemma 8.

$B=20$, $J=707$, $Q=5$, $r=1$, $\{m_b^{\text{BAY}}\}=\{100,1,0,100,1,0,100,1,0,100,1,0,100,1,0,100,1,0,100,1\}$, $b_L=1$, $b_R=20$, $\{\sigma_b\}=\{0,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1\}$, $\delta=2$, $\varphi=8$, $B_{\max}^{r+1}=101$, $T_{\text{avg}}^{\text{fix}}=141.4$, $L_{\text{avg}}^{\text{act}}=143.6$, $LB=143.6$, $LB+\delta=145.6$, $LB+2\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}=148.6$.

The $m_{i,b}$ values of the schedule S^c acquired by heuristic are shown in the top 5 rows of Table 1. Bay 4, 7, 13 and 16 are shared bays, so Lemmas 5 and 6 are verified. This schedule is contiguous by definition, so Lemma 7 is verified. Since each container job has fixed handling time of 1, the active time of each QC can be calculated and are listed in the right-most column, so both Formula (4) and Lemma 4 are verified. Besides, $RT_{i,b}$, the time of QC i reaching Bay b are given in the bottom 5 rows of Table 1, and Lemma 8 can be verified. For example, considering QC 4 and QC 5, since $RT_{4,14} > RT_{5,16}$, $RT_{4,15} > RT_{5,17}$, $RT_{4,16} > RT_{5,18}$, QC 4 and QC 5 would not conflict with each other. Finally, the makespan of S^c equals 146, and Theorem 1 is verified.

6. Numerical experiment

6.1. Test instance modification

The test instances for the proposed heuristic are modified from the QCSP test sets provided by [18]. The test instances of [18] adopt the granularity of container group, and for each container group, its handling time equals its number of containers, i.e., the average handling time for a single container is one unit of time. For our experiment, the information about each container j should be further generated, including its handling time t_j^{fix}, stack location S_j and precedence constraints with other containers.

The generated t_j^{fix} should fulfill two requirements: (1) t_j^{fix} is distributed between $[1-\theta,\ 1+\theta]$ units of time, where θ is a parameter indicating the variance range of single container handling time; (2) the sum of t_j^{fix} in each group equals ω, which is the number of containers in this group.

For this purpose, the idea of "group processing time generation" procedure in Section 4.2.2 of [18] is borrowed. This procedure splits the total amount of processing time into a given number of random segments, while the maximum duration of each segment is ensured, and the minimum duration of each segment is set to zero. This procedure is modified to further ensure a minimum duration of $(1-\theta)$ for each segment, so that the two requirements on t_j^{fix} can be satisfied in the generated instance. The modified procedure is:

*Step*1 By using the "group processing time generation" procedure, split $\theta\omega$ into ω segments $\{\tau_j, j=1,\ldots\omega\}$, while ensuring $0 \le \tau_j \le 2\theta$, $\forall j=1,\ldots\omega$.

Table 1
$m_{i,b}$, $T_i^{\text{act}}(J_i)$ and $RT_{i,b}$ values of schedule S^c acquired by heuristic.

i	b																				$T_i^{\text{act}}(J_i)$
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
$m_{1,b}$	100	1	0	42																	146
$m_{2,b}$				58	1	0	84														146
$m_{3,b}$							16	1	0	100	1	0	22								146
$m_{4,b}$													78	1	0	64					146
$m_{5,b}$																36	1	0	100	1	142
$RT_{1,b}$	0	101	103	104																	
$RT_{2,b}$				0	59	61	62														
$RT_{3,b}$							0	17	19	20	121	123	124								
$RT_{4,b}$													0	79	81	82					
$RT_{5,b}$																0	37	39	40	141	

*Step*2 Let $t_j^{\text{fix}} = 1\text{-}\theta + \tau_j, \quad \forall j = 1,\ldots\omega$.
*Step*3 Repeat *Step*1–2 for each group.

After generating t_j^{fix} for each container, its stack location S_j is randomly assigned. As to the precedence constraints, for containers from different groups, the precedence relations of their groups are inherited; for containers in the same group, if they are also in the same stack, a precedence relation is randomly indicated.

6.2. *Test results discussion*

The test results are shown in Table 2. Following the definitions of [18], instance set *A*, *B*, *C* represent test sets of small, media and large vessels, respectively; and notation *n*, *f*, *loc*, *d*, *q* represent number of tasks, handling rate, location parameter, precedence density and number of cranes, respectively. Detailed explanations related to these test sets are given in [18]. In Table 2, *UB* is the proved upper bound for $C_{\max}(S^c)$ and equals $LB+2\delta+\max_{j\in\mathbf{J}} t_j^{\text{fix}}$. Each *LB*, $C_{\max}(S^c)$ and *UB* item takes the sum value of 10 instances with different random seeds, as in [18]. *Gap*-1 is calculated by $(C_{\max}(S^c)/LB-1)*100\%$, and *Gap*-2 is calculated by $(UB/LB-1)*100\%$.

In test set *E*, different *d* values are compared, representing various precedence densities among groups. Without loss of feasibility, we supplement precedence relations to the instances of $d<1$ until $d=1$. Consequently, the results for $d<1$ are the same with results of $d=1$ in test set *E*.

For each instance, $LB < C_{\max}(S^c) < UB$ is verified, as proved by Theorem 1. *Gap*-1 represents the difference between acquired contiguous schedule and optimal schedule, and *Gap*-2 represents the maximum possible *Gap*-1 value as proved. The average of *Gap*-1 is 0.3%, and the average of *Gap*-2 is 0.57%. Both values turn out to be very small under practical problem scales.

Comparing our test results to those reported in [18], a smaller makespan value can be observed in most cases. Denote Z_{MB} the test results reported in [18]. $C_{\max}(S^c) < Z_{\text{MB}}$ is expected since by using lower granularity, better workload balance among QCs can be achieved. Besides, the additional assumption that each QC is available

Table 2
Test results of numerical experiment.

Test set								
A	$n=$	10	15	20	25	30	35	40
	LB	5,032.5	5,035	5,036	5,037	5,038	5,039	5,039.5
	$C_{\max}(S^c)$	5,057.93	5,058.08	5,056.82	5,057.89	5,055.68	5,057.37	5,057.12
	UB	5,086.5	5,083	5,082	5,079	5,078	5,075	5,075.5
	Gap-1(%)	0.51	0.46	0.41	0.41	0.35	0.36	0.35
	Gap-2(%)	1.07	0.95	0.91	0.83	0.79	0.71	0.71
B	$n=$		45	50	55	60	65	70
	LB		7,699.5	7,623.5	7,664.25	7,658.75	7,650	7,553.75
	$C_{\max}(S^c)$		7,721.17	7,645.67	7,684.98	7,676.90	7,668.69	7,573.5
	UB		7,739.5	7,665.5	7,702.25	7,694.75	7,686	7,589.75
	Gap-1(%)		0.28	0.29	0.27	0.24	0.24	0.26
	Gap-2(%)		0.52	0.55	0.50	0.47	0.47	0.48
C	$n=$		75	80	85	90	95	100
	LB		11,382.33	11,107.67	10,790	10,631	10,773	10,785.33
	$C_{\max}(S^c)$		11,402.05	11,129.91	10,812.84	10,651.74	10,792.97	10,806.83
	UB		11,416.33	11,147.67	10,832	10,667	10,807	10,821.33
	Gap-1(%)		0.17	0.20	0.21	0.20	0.19	0.20
	Gap-2(%)		0.30	0.36	0.39	0.34	0.32	0.33
D	$f=$		0.2	0.2	0.2	0.8	0.8	0.8
	$loc=$		*cl*1	*cl*2	*uni*	*cl*1	*cl*2	*uni*
	LB		4,718	3,641.25	3,456.75	12,023.25	12,026.25	12,027
	$C_{\max}(S^c)$		4,742.07	3,665.56	3,478.22	12,042.3	12,045.14	12,047.37
	UB		4,766	3,685.25	3,498.75	12,059.25	12,064.25	12,065
	Gap-1(%)		0.51	0.67	0.62	0.16	0.16	0.17
	Gap-2(%)		1.02	1.21	1.22	0.30	0.32	0.32
E	$d=$			0.8	0.85	0.9	0.95	1
	LB			7,623.5	7,623.5	7,623.5	7,623.5	7,623.5
	$C_{\max}(S^c)$			7,645.67	7,645.67	7,645.67	7,645.67	7,645.67
	UB			7,665.5	7,665.5	7,665.5	7,665.5	7,665.5
	Gap-1(%)			0.29	0.29	0.29	0.29	0.29
	Gap-2(%)			0.55	0.55	0.55	0.55	0.55
F	$q=$			2	3	4	5	6
	LB			15,062.5	10,038.33	7,623.5	7,352	7,352
	$C_{\max}(S^c)$			15,083.65	10,059.91	7,645.67	7,375.01	7,375.01
	UB			15,104.5	10,080.33	7,665.5	7,394	7,394
	Gap-1(%)			0.14	0.21	0.29	0.31	0.31
	Gap-2(%)			0.28	0.42	0.55	0.57	0.57
G	$r=$			0	1	2	3	4
	LB			7,526.25	7,623.5	9,949	12,398	14,054
	$C_{\max}(S^c)$			7,549.98	7,645.67	9,973.27	12,419.64	14,077.55
	UB			7,568.25	7,665.5	9,991	12,440	14,096
	Gap-1(%)			0.32	0.29	0.24	0.17	0.17
	Gap-2(%)			0.56	0.55	0.42	0.34	0.30

from time 0 also improves the solution quality, since it reduces a small amount of set-up time required for each QC's first assigned job. However, there are five cases under which $C_{\max}(S^c)$ values are slightly larger than Z_{MB} values, i.e., the test set *F* with $q=5$, 6 and the test set *G* with $r=2$, 3, 4. This is mainly because the obtained contiguous schedule by the heuristic is not optimal considering single container granularity. As aforementioned, the purpose of this paper is not to find optimal solutions, but to balance between schedule quality and solving speed. For this purpose, this paper actually focuses on only contiguous bay area partition and unidirectional operation manner; additionally, the proposed heuristic is constructed to generate only "conflict-free" schedules. Indeed, there can be some global optimal schedule at single container granularity which are not contiguous bay area partitioned, not unidirectionally operated, or not conflict-free. Nevertheless, although the scope of considered schedules is limited in this paper, the quality of obtained solution is ensured, i.e., its gap above the global optimal solution (*Gap*-1) is proved to be bounded (by *Gap*-2) and quite small in practice (0.3% on average).

Quantitatively, according to (4), a lower bound $LB_{\mathrm{P}}=LB+\delta$ can be identified for the conflict-free contiguous schedule S^c generated by the proposed heuristic. Comparatively, as aforementioned in (2) and (3), *LB* is the lower bound for any feasible QC schedule, no matter whether it is contiguous or noncontiguous, with or without conflict idle time. For the five cases with $C_{\max}(S^c)>Z_{\mathrm{MB}}$, Table 3 summarizes their lower bounds and makespan values, where *Gap*-3 is calculated by $(C_{\max}(S^c)/Z_{\mathrm{MB}}-1)*100\%$. For these cases, although $C_{\max}(S^c)>Z_{\mathrm{MB}}$, the actual difference is quite small (*Gap*-3 is 0.15% on average).This is acceptable in practice.

Note that, LB_{P} is smaller than Z_{MB} for the first case. This implies the possible existence of an optimal conflict-free contiguous schedule which is better than Z_{MB}. This also reveals a limit of the search implemented in the straightforward heuristic. The proposed heuristic first generates a sequence of all containers on vessel by the beginning "sequence" sub-procedure, where ties are broken randomly when no precedence relation exists between two containers. Then the posterior "partition" and "re-sequence" sub-procedures decode this sequence. In this way a conflict-free contiguous schedule is obtained. However, there actually exists a huge number of feasible sequences of containers, in addition to the one randomly generated by the "sequence" sub-procedure; and any feasible sequence can produce a conflict-free contiguous schedule using the same "partition" and "re-sequence" sub-procedures. To improve the search, such polynomial-time "partition" and "re-sequence" sub-procedures in the proposed heuristic can be incorporated into some meta-heuristic algorithm as the decoding and evaluation method, while the meta-heuristic searches within the space of feasible container sequences. In this way, an improved conflict-free contiguous schedule may be found which is better than the reported initial S^c. Nevertheless, such improved $C_{\max}(S^c)$ will not outperform the LB_{P} value, and the tightness between current $C_{\max}(S^c)$ and LB_{P} values (the gap of 0.12% in this case) is likely not worth such computational burden.

Furthermore, note that LB_{P} is larger than Z_{MB} for the rest four cases in Table 3. As aforementioned, this implies the limited scope of considered schedules in this paper. Thus theoretically, in these cases, there will be no optimal conflict-free contiguous schedule with better makespan than Z_{MB}, as long as the partition condition in the heuristic is used. However, since $LB<Z_{\mathrm{MB}}$, a schedule at the granularity of single container with better makespan than Z_{MB} possibly exists. The feasible solution space can be expanded by using other partition conditions, by allowing conflict idle time, or by incorporating noncontiguous schedules. Nevertheless, such improved $C_{\max}(S^c)$ will not outperform the *LB* value eventually, and the small difference between current $C_{\max}(S^c)$ and *LB* values (the average *Gap*-1 of 0.22% in these four cases) indicates little improvement potential.

As a result, although not fully optimized and worse than Z_{MB} in some cases, the obtained contiguous schedule by the heuristic achieves a good trade-off between solution quality and solving speed. Besides, note that the five cases in Table 3 actually reflect relatively congested situations, where many cranes or large safety distances may inhibit an effective utilization of all cranes. Even so, the proposed contiguous schedule and heuristic can still appropriately handle such situations, by simultaneously utilizing two different lower bounds. This issue will be discussed in detail in Section 6.3.

Table 3
Comparison between test results under five cases.

Case	Z_{MB}	Z	LB_{P}	LB	*Gap*-1(%)	*Gap*-3(%)
$q=5$ in test set *F*	7,373	7,375.01	7,366	7,352	0.31	0.03
$q=6$ in test set *F*	7,361	7,375.01	7,366	7,352	0.31	0.19
$r=2$ in test set *G*	9,954	9,973.27	9,963	9,949	0.24	0.19
$r=3$ in test set *G*	12,399	12,419.64	12,412	12,398	0.17	0.17
$r=4$ in test set *G*	14,054	14,077.55	14,068	14,054	0.17	0.17

6.3. Correlation between LB value and QC workload balance

In the heuristic, in order to eliminate the conflict idle time, the *LB* value is set as the maximum between $L^{\mathrm{act}}_{\mathrm{avg}}$ and $B^{r+1}_{\max}$. However, only the larger one of them is active for a given instance and adopted into the partition condition of the heuristic. This property influences the QC workload balance of the obtained contiguous schedule.

Figs. 10–15 illustrate the utilized lower bounds and scheduled QC operating times for different cases. For each case, only the instance with random seed of 1 is selected for illustration.

In Fig. 10, $L^{\mathrm{act}}_{\mathrm{avg}}$ varies little under different cases in test set *A*. This is because by definition, $L^{\mathrm{act}}_{\mathrm{avg}}=T^{\mathrm{fix}}_{\mathrm{avg}}+(b_{\mathrm{R}}-b_{\mathrm{L}}-\varphi)/Q=(T^{\mathrm{fix}}_{\mathrm{sum}}+b_{\mathrm{R}}-b_{\mathrm{L}}-\varphi)/Q$, and thus its value is mainly influenced by the total workload on vessel and the number of deployed QCs, both of which remain unchanged in test set *A*. $B^{r+1}_{\max}$ fluctuates under different cases, because by definition, $B^{r+1}_{\max}$ not only depends on the total workload and available QCs, but also depends on the particular workload distribution among bays. Note that under each case, $L^{\mathrm{act}}_{\mathrm{avg}}$ remains larger than $B^{r+1}_{\max}$, thus by (4), QC 1's operating time, i.e., $T^{\mathrm{act}}_1(J_1)$, is no larger than $(T^{\mathrm{fix}}_{\mathrm{sum}}+b_{\mathrm{R}}-b_{\mathrm{L}}-\varphi)/2+2\delta+\max_{j\in\mathbf{J}} t^{\mathrm{fix}}_j$. This implies that the remaining workload assigned to QC 2 should be closed to $(T^{\mathrm{fix}}_{\mathrm{sum}}+b_{\mathrm{R}}-b_{\mathrm{L}}-\varphi)/2$ as well, which accords with Fig. 10, where operating times of two QCs are well balanced.

In Fig. 11, $L^{\mathrm{act}}_{\mathrm{avg}}$ varies little under different cases in test set *B*, while $B^{r+1}_{\max}$ fluctuates. Operating times of four QCs are also well balanced for most cases (except for $n=60$). The reasons are the same as in test set *A*. For the case of $n=60$, $L^{\mathrm{act}}_{\mathrm{avg}}$ is smaller than

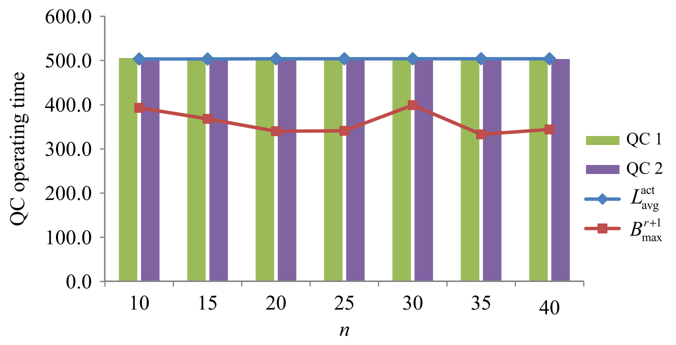


Fig. 10. QC operating times for cases of Set *A* (instance seed=1).

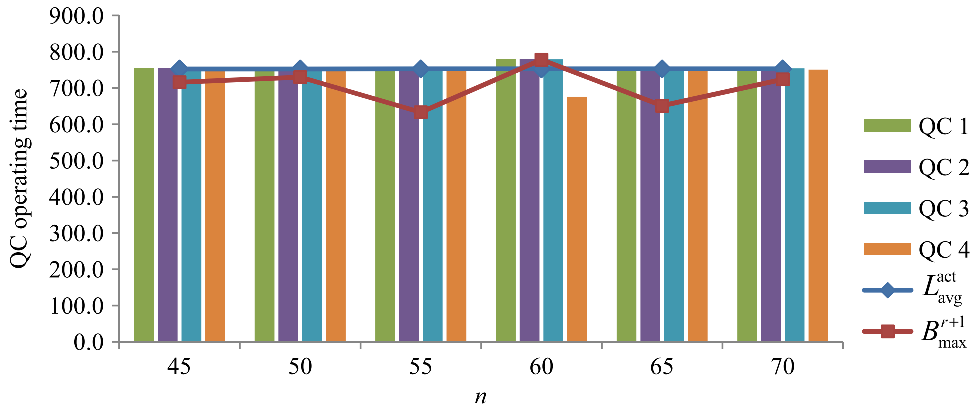


Fig. 11. QC operating times for cases of Set *B* (instance seed = 1).

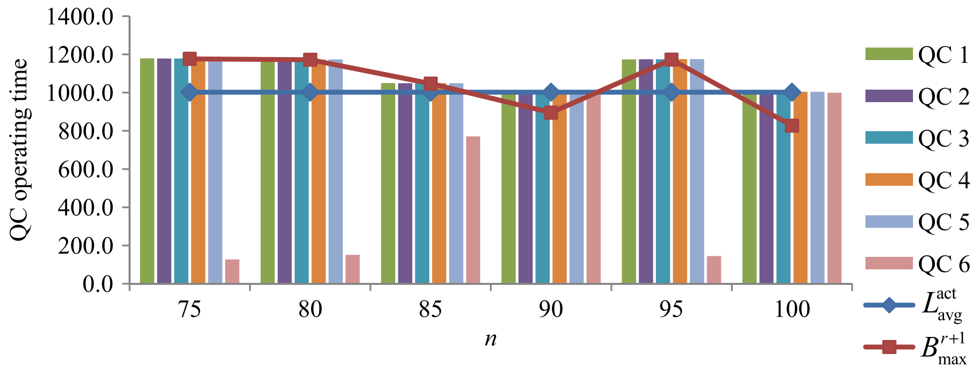


Fig. 12. QC operating times for cases of Set *C* (instance seed = 1).

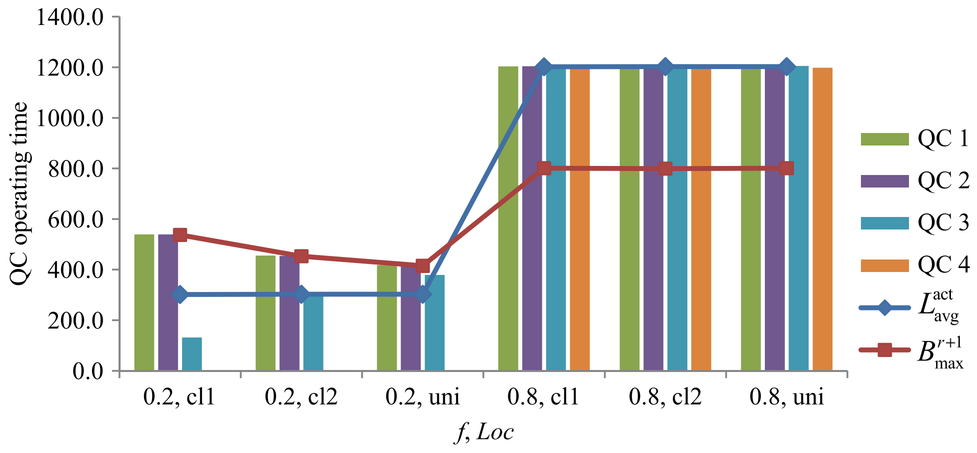


Fig. 13. QC operating times for cases of Set *E* (instance seed = 1).

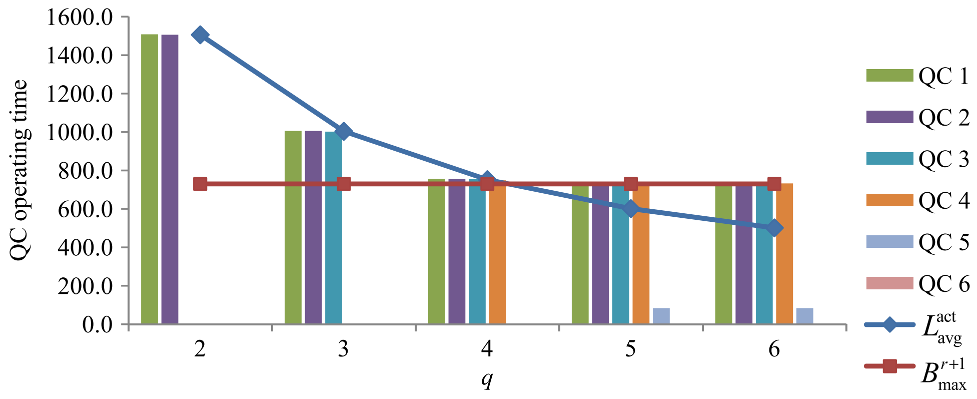


Fig. 14. QC operating times for cases of Set *F* (instance seed = 1).

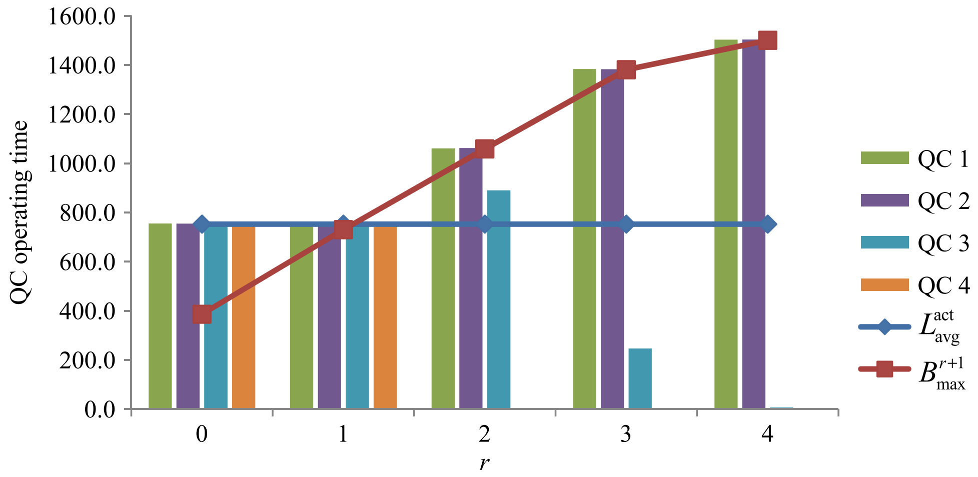


Fig. 15. QC operating times for cases of Set G (instance seed=1).

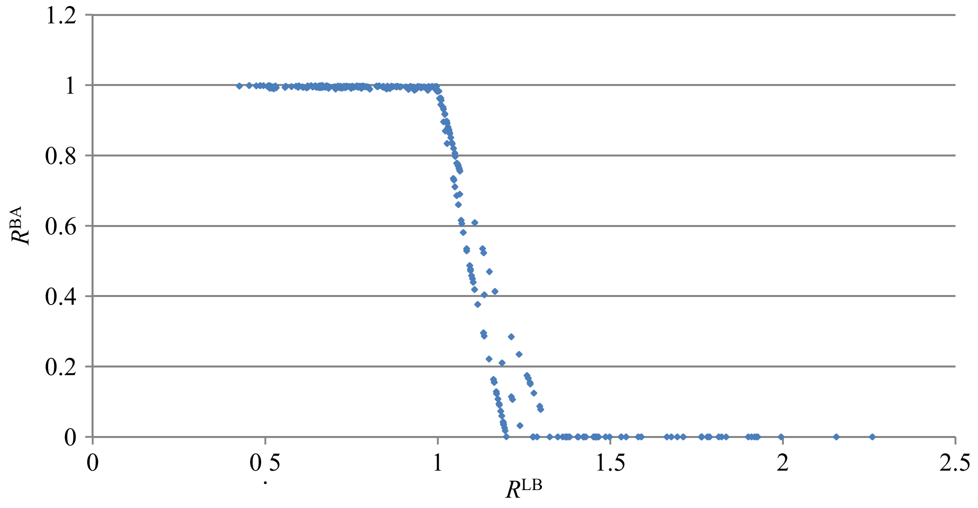


Fig. 16. Correlation between R^{LB} and R^{BA} values for all instances from test sets.

B_{max}^{r+1}, thus by (4), $T_i^{act}(J_i) > B_{max}^{r+1}+\delta > (T_{sum}^{fix}+b_R-b_L-\varphi)/4$, $\forall i=1,2,3$. This implies the remaining workload assigned to QC 4 is smaller than $(T_{sum}^{fix}+b_R-b_L-\varphi)(1-3/4)$, and hence $T_4^{act}(J_4) < B_{max}^{r+1}=LB$. Accordingly in Fig. 11, under this case of $n=60$, the operating times of QC 4 is not balanced with QC 1, 2 and 3.

In Fig. 12, L_{avg}^{act} again varies little under different cases in test set C, while B_{max}^{r+1} fluctuates. For cases of $n=90$ and $n=100$, operating times of six QCs are well balanced. The reasons are also the same as in test set A. For the cases of $n=75$, 80, 85 and 95, since $L_{avg}^{act} < B_{max}^{r+1}=LB$, they are similar to the case of $n=60$ in Fig. 11, only with worse workload balance observed.

In Fig. 13, since the total workload on vessel is proportional to the handling rate in test set E, both L_{avg}^{act} and B_{max}^{r+1} increase with higher handling rate in test set E, and L_{avg}^{act} varies little under the same handling rate. By definition, under the same handling rate, the B_{max}^{r+1} value should be largest with single Gaussian distributed containers ($cl1$), and smallest with uniform distributed containers (uni). This trend is significant when $f=0.2$. However when $f=0.8$, B_{max}^{r+1} values have no significant differences, since each bays' capacity has already been highly occupied. The QC workloads are balanced when $f=0.8$ and $L_{avg}^{act} > B_{max}^{r+1}$, and unbalanced when $f=0.2$ and $B_{max}^{r+1} > L_{avg}^{act}$. Moreover, given the relatively small amount of containers on vessel when $f=0.2$, only three of the available QCs are utilized, i.e., $Q'=3$.

In Fig. 14, with more QCs being deployed in test set F, L_{avg}^{act} decreases since it is approximately inversely proportional to the number of available QCs, whereas B_{max}^{r+1} remains unchanged since it is only related to containers' distribution on vessel and safety distance of QCs. For the cases of $q=2$, 3 and 4, $L_{avg}^{act} > B_{max}^{r+1}$, thus the QC workloads are balanced. For the cases of $q=5$ and 6, $B_{max}^{r+1} > L_{avg}^{act}$, thus the QC workloads are unbalanced. Furthermore, the acquired schedules are the same under these two cases, since the containers' distribution on vessel are the same and LB takes the fixed value of B_{max}^{r+1}. As a result, only five QCs are actually utilized in the case of $q=6$.

In Fig. 15, with larger QC safety distance specified in test set G, B_{max}^{r+1} increases since it is positively correlated with the r value, whereas L_{avg}^{act} remains unchanged since it is unrelated to the r value. For the cases of $r=0$ and 1, $L_{avg}^{act} > B_{max}^{r+1}$, thus the QC workloads are balanced. For the cases of $r=2$, 3 and 4, $B_{max}^{r+1} > L_{avg}^{act}$, thus the QC workloads are unbalanced, and only three QCs are actually utilized, respectively in these cases. Furthermore, the acquired schedules are the same under cases of $r=0$ and 1, since containers' distribution on vessel are the same and LB takes the fixed value of L_{avg}^{act}.

Moreover, instances in Figs. 10–15 also imply that by the heuristic, the relatively larger gap between B_{max}^{r+1} and L_{avg}^{act} can intensify the workload unbalance among QCs. Denote $R^{LB}=B_{max}^{r+1}/L_{avg}^{act}$, which indicates the relative quantity of B_{max}^{r+1} and L_{avg}^{act}, and denote $R^{BA}=\min_{i=1,\dots Q} T_i^{act}(J_i)/C_{max}(S^C)$, which evaluates the balance among QC workloads. To illustrate their correlation, R^{LB} and R^{BA} values of all

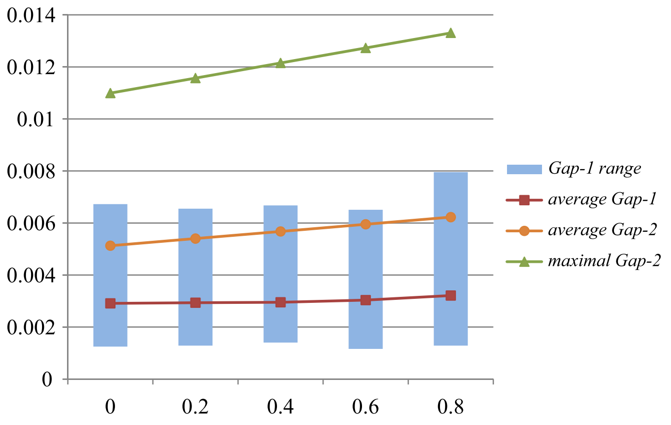


Fig. 17. Sensitive of gaps under different θ values.

tested instances are summarized as the scatter points in Fig. 16. It shows that when $R^{LB} \le 1$, i.e., when $B_{\max}^{r+1} \le L_{\text{avg}}^{\text{act}}$ and $LB = L_{\text{avg}}^{\text{act}}$, R^{BA} values are quite close to 1, meaning that QC workloads are well balanced. However, when $R^{LB} > 1$, i.e., when $B_{\max}^{r+1} > L_{\text{avg}}^{\text{act}}$ and $LB = B_{\max}^{r+1}$, R^{BA} values start declining from 1 to 0, indicating unbalanced QC workloads. For many instances, R^{BA} takes the value of 0, meaning that at least one available QC is not utilized.

Note that $B_{\max}^{r+1}$ remains unchanged as more QCs are assigned to a vessel, whereas $L_{\text{avg}}^{\text{act}}$ decreases, since it measures the average workload of QCs. Thus the value of R^{LB} can be viewed as an estimation of QC congestion relative to vessel workload. Facing higher QC congestion, the proposed heuristic can eliminate the conflict idle time for the generated contiguous schedule, at the cost of compromising some workload balance among QCs. Hence for the terminal operator to properly decide the deployed QC number, it is recommended to choose the q value which makes R^{LB} close to 1. Otherwise, a too large R^{LB} value induces imbalance among QCs and wastes QCs' capacity; whereas a too small R^{LB} value implies the positive marginal utility of QC number and suggests further makespan improvement by adding QC resources. Furthermore, for strategic decisions like quayside resource configuration and equipment parameter selection, a desired combination of q and r can be determined based on historical or predicted vessel data, by making the R^{LB} values clustered near 1 for as many vessels as possible.

6.4. *Sensitivity analysis*

Although the θ value is set to 0.4 in previous tests, the reasoning and implications in this section still applies with other θ values. The influence of practical θ value on *Gap*-1 and *Gap*-2 are illustrated in Fig. 17. With θ increases, the average and maximum values of *Gap*-2 increases proportionally. However the average value of *Gap*-1 and its range do not change much. Thus the heuristic shows a steady performance under different variation degrees of container handling time.

7. Conclusion

This paper studies the QCSP based on single container operation. Ideally, by taking smaller granularity and assigning container jobs in one bay to multiple QCs, the QC workload can be further balanced and the equipment utilization can be enhanced. Mathematically, existing models of QCSP based on container groups are equivalent to the model studied herein, as long as we view each single container as a container group. However, existing algorithms of QCSP based on container groups become much too time-consuming when applied to single container discretizations, due to the explosion of problem scale in both variables and constraints. Based on the analysis of practical characteristics and constraints of QC operations, the proposed heuristic procedure is efficient and effective at the granularity of single container. It uses polynomial time, and is proved to have a bounded optimality gap under all circumstances. In our experiments, such gap is quite small in reality, and the proposed method achieves a good trade-off between solution quality and solving speed. The generated conflict-free contiguous schedule also reduces the operational complexity for QC operators. Sensitivity analysis shows the steady performance of our method. Aside from QC schedule generation, the revealed correlation between *LB* value and QC workload balance is also applicable for the decision making of container terminal operators. For future research, double cycling mode of QCSP should be considered for our method, so that more efficient QC utilization can be achieved.

Acknowledgments

This research is supported by National Natural Science Foundation of China (No. 71171130, 70771065) and National High Technology Research and Development Program of China (863 Program) (No. 2009AA043000, 2009AA043001).

Appendix A. Supplementary Information

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.cor.2012.02.013.

References

[1] Stahlbock R, Voß S. Operations research at container terminals: a literature update. OR Spectrum 2008;30(1):1–52.
[2] Vacca I, Bierlaire M, Salani M. Optimization at container terminals: status, trends and perspectives. Citeseer 2007:12–4.
[3] Vis IFA, de Koster R. Transshipment of containers at a container terminal: an overview. European Journal of Operational Research 2003;147(1):1–16.
[4] Bierwirth C, Meisel F. A survey of berth allocation and quay crane scheduling problems in container terminals. European Journal of Operational Research 2010;202(3):615–27.
[5] Daganzo CF. The crane scheduling problem. Transportation Research Part B. 1989;23(3):159–75.
[6] Peterkofsky RI, Daganzo CF. A branch and bound solution method for the crane scheduling problem. Transportation Research Part B. 1990;24(3):159–72.
[7] Steenken D, Voß S, Stahlbock R. Container terminal operation and operations research – a classification and literature review. OR Spectrum 2004;26(1):3–49.
[8] Lim A, Rodrigues B, Xiao F, Zhu Y. Crane scheduling with spatial constraints. Naval Research Logistics 2004;51(3):386–406.
[9] Lim A, Rodrigues B, Xu ZA. M-parallel crane scheduling problem with a non-crossing constraint. Naval Research Logistics 2007;54(2):115–27.
[10] Lee DH, Wang HQ, Miao L. Quay crane scheduling with handling priority in port container terminals. Engineering Optimization 2008;40(2):179–89.
[11] Liu J, Wan YW, Wang L. Quay crane scheduling at container terminals to minimize the maximum relative tardiness of vessel departures. Naval Research Logistics 2006;53(1):60–74.
[12] Ak A. Berth and quay crane scheduling: problems, models and solution methods. PhD Thesis. Atlanta: Georgia Institute of Technology; 2008.
[13] Ng WC, Mak KL. Quay crane scheduling in container terminals. Engineering Optimization 2006;38(6):723–37.
[14] Bierwirth C, Meisel F. A fast heuristic for quay crane scheduling with interference constraints. Journal of Scheduling 2009;12(4):345–60.
[15] Kim KH, Park YM. A crane scheduling method for port container terminals. European Journal of Operational Research 2004;156(3):752–68.
[16] Moccia L, Cordeau JF, Gaudioso M, Laporte G. A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal. Naval Research Logistics 2006;53(1):45–59.
[17] Sammarra M, Cordeau JF, Laporte G, Monaco MF. A tabu search heuristic for the quay crane scheduling problem. Journal of Scheduling 2007;10(4–5):327–36.
[18] Meisel F, Bierwirth C. A unified approach for the evaluation of quay crane scheduling models and algorithms. Computers & Operations Research 2011;38(3):683–93.
[19] Goodchild AV, Daganzo CF. Double-cycling strategies for container ships and their effect on ship loading and unloading operations. Transportation Science 2006;40(4):473–83.
[20] Meisel F, Wichmann M. Container sequencing for quay cranes with internal reshuffles. OR Spectrum 2010;32(3):569–91.