Md. Mahfuzur Rahman¹, Md Abrar Jahin¹, Md. Saiful Islam¹, M. F. Mridha², and Jungpil Shin³

¹Department of Industrial Engineering and Management, Khulna University of Engineering & Technology (KUET), Khulna, 9203, Bangladesh
²Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka, 1229, Bangladesh
³Department of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, 965-8580, Japan

April 01, 2024
QCDC-DR-GA: Optimizing Container Loading and Unloading through Dual-Cycling and Dockyard Rehandle Reduction Using a Hybrid Genetic Algorithm

Md. Mahfuzur Rahman, Md Abrar Jahin, Md. Saiful Islam, M. F. Mridha, Senior Member, IEEE, and Jungpil Shin, Senior Member, IEEE

Abstract—This paper addresses the optimization of container unloading and loading operations at ports, integrating quay-crane dual-cycling (QCDC) with dockyard rehandle minimization. We present a unified model encompassing both operations: ship container unloading and loading by quay crane, and the other is reducing dockyard rehandles while loading the ship. We recognize that optimizing one aspect in isolation can lead to suboptimal outcomes due to interdependencies. Specifically, optimizing unloading sequences for minimal operation time may inadvertently increase dockyard rehandles during loading and vice versa. To address this NP-hard problem, we propose a hybrid genetic algorithm (GA) QCDC-DR-GA comprising 1-dimensional and 2-dimensional GA components. Our model, QCDC-DR-GA, consistently outperforms four state-of-the-art methods in maximizing dual cycles and minimizing dockyard rehandles. Compared to those methods, it reduced 15-20% of total operation time for large vessels. Results underscore the inefficiency of separately optimizing QCDC and dockyard rehandles. Fragmented approaches, such as QCDC Scheduling Optimized by bi-level GA and GA-ILSRS (Scenario 2), show limited improvement compared to QCDC-DR-GA. As in GA-ILSRS (Scenario 1), neglecting dual-cycle optimization leads to inferior performance than our proposed QCDC-DR-GA.

Index Terms—Dual Cycling, Quay Crane, Dockyard Rehandles, Genetic Algorithm, 2D Crossover, 2D Mutation

I. INTRODUCTION

Global trade relies heavily on efficient port operations, with shipping containers carrying nearly 80% of the world’s goods. Therefore, countries are competing to have large fleets. The operation to receive these mega-ships needs preparations. The goal and challenge of every port is now to reduce the turnaround time of vessels. The most expensive single container handling equipment unit and the main operational bottleneck at ports are the Quay Cranes (QCs) [1]. Ports can decrease ship turnaround time, increase productivity, and boost freight transportation system throughput by increasing QC efficiency [2]. Our research addresses this key bottleneck to port productivity. The approach taken here is a low-cost method; neither new infrastructure nor technologies are needed. Although our strategy would not fix the capacity issue in the long term, it can be applied more quickly than other approaches and be used in conjunction with other methods.

Traditionally, ports adopt a single cycle approach (see Fig. 1), where QCs handle loading after completing unloading tasks. However, dual cycling presents an advanced strategy allowing simultaneous loading and unloading, thereby reducing empty crane moves and potentially decreasing turnaround times significantly [2] (see Fig. 1).

![Fig. 1: (a) Unloading Using Single Cycling; (b) Unloading and Loading with Double Cycling](image-url)

Despite its benefits, maximizing the no. of dual cycles requires careful consideration of factors such as the unloading sequence of stacks within a ship row. Previous studies have proposed greedy, heuristic, and metaheuristic algorithms, including genetic algorithms (GA), to address the NP-hard nature of optimizing quay crane dual-cycling (QCDC) scheduling [3], [4], [5], [6]. Some of the studies focused on overall handling efficiency and the system’s stability of container terminals with double cycling and other inbound vehicles of the port [7], [8].

Another issue named rehandling of containers arises at the dockyard while loading the ship. Rehandling occurs when the
target container is not on the top of the stack. So, minimizing rehandling in the dockyard, where containers are moved for retrieval or rearrangement, is crucial for efficiency. Numerous works have addressed this issue by creating models and developing solving approaches \cite{9, 10, 11}. This NP-hard problem is also tackled effectively using GA \cite{12}.

Our work builds upon existing research by integrating maximization of the no. of dual cycles and minimization of the no. of dockyard rehandles into a unified model. We introduce the Maximizing Quay Crane Dual Cycles and Minimizing Dockyard Rehandles by GA (QCDC-DR-GA) method to solve this model efficiently.

This study presents seven significant contributions:
1) Empirical validation of the correlation between unloading sequence and dockyard rehandles.
2) Development of a comprehensive model integrating dockyard and QCDC operations.
3) Introduction of a novel hybrid GA approach tailored to container handling optimization.
4) Proposal of specialized GA techniques to address unique challenges.
5) Extensive analysis of computational parameters within the GA framework.
6) Rigorous benchmarking analysis against four state-of-the-art algorithms, demonstrating superior performance and reliability.
7) Statistical validation of the significant performance of QCDC-DR-GA using a two-tailed paired t-test.

This article is structured in the following manner: In section “Problem Description”, we discuss the problem statement. The “Methodology” section covers model formulation (objectives and constraints) and our approach, QCDC-DR-GA, including its workflow, strategies, and parameters. The “Results” section details scenario generation, computational experiments, and result analysis. Finally, the “Conclusions” section summarizes the work, highlights primary contributions, and suggests future directions.

II. Problem Description

Ship or port yard container is arranged in a three-dimensional matrix of rows, bays, and tiers (see Fig. 2). Containers are stacked in rows, with each row spanning the width of the bay or ship. The operating cycle of a Quay Crane (QC) involves (a) Locking and unlocking the trolley with the container, (b) Horizontal movement of the trolley (with container), and (c) Vertical movement of the trolley (with container).

A. Case Consideration

Upon the arrival of a vessel at the port, with containers to be unloaded and a loading plan for other containers, let $U_c$ and $L_c$ represent the numbers of containers to be unloaded and loaded, respectively, for each stack. Fig. 3 illustrates an example used in this work. Let $S$ be the set of stacks in a row. $|S| = N$ denotes the number of stacks in set $S$, and $P$ is a permutation of set $S$ indicating the order of stack handling. The sequence in which stacks within each row are handled affects the total number of cycles, as explained by Goodchild (2006) \cite{2}.

1) Generic Double Cycling Method:
   (i) Select any unloading permutation, $P'$. Unload containers stack by stack.
   (ii) Select a loading permutation, $P$, and load stacks according to it.

2) Number of Rehandles in Dockyard:

   We integrate the nearest lowest stack strategy to address rehandles, as applying the lowest stack strategy alone can be challenging in real-life scenarios. The example in Fig. 3 (b) illustrates a scenario with 3 rehandles under this strategy.

III. Methodology

A. Mathematical Model

The QCDC problem is modeled as a two-machine flow shop problem.

1) Assumptions: We make the following assumptions:
   (i) Containers at the dockside are prepared for loading as needed.
   (ii) Unloaded containers are promptly removed from the area and stored appropriately.
   (iii) Rehandles of containers on the ship are counted during both the unloading and loading processes.
   (iv) Rehandles on the ship are returned to the same stack from which they were taken.
   (v) Rehandles on the ship are considered to move between the vessel and the apron; however, in reality, some may only move between stacks on the vessel.
   (vi) Rehandles in the dockyard are prioritized using the nearest stack strategy, placing containers on the nearest lowest stack.
   (vii) The turnaround time of a vessel, indicative of QC efficiency, is measured by minimizing the total number of
(viii) Unloading and loading of one row are completed before
following:

Fig. 3: Illustration of unloading and loading plan of a ship

row.

single \( (w_s) \) and dual cycles \( (w_d) \) required for unloading
and loading.

(ix) No interruptions occur due to inbound vehicles or cranes.

2) Symbols and Decision Variables: The notations are as
follows:

\( m \): Bay of containers in the yard
\( n \): Stack of containers in the yard
\( o \): Tier of containers in the yard
\( U_c \): Number of containers to unload in stack \( c \in S \)
\( L_c \): Number of containers to load in stack \( c \in S \)
\( T U_c \): Completion time of unloading \( c \in S \)
\( T L_c \): Completion time of loading \( c \in S \)
\( T \): Total completion time of unloading loading
\( R \): Number of rehandles of a row in the dockyard
\( \alpha \): Average completion time of a single cycle
\( \beta \): Average completion time of a double cycle
\( \gamma \): Average time it takes to tackle a rehandle at the dockyard
\( \mu \): Large value
\( H_{mn} \): Highest tier of the yard bay \( m \) and stack \( n \)
\( h_{mn} \): Height of the yard-bay \( m \) and stack \( n \)

The decision variables are as follows:

\( X_{ij} \): binary variable for the sequence of unloading jobs (1 if
\( j \in S \) is loaded after \( i \in S \) and 0 otherwise)
\( Y_{ij} \): binary variable for the sequence of loading jobs (1 if
\( j \in S \) is loaded after \( i \in S \) and 0 otherwise)
\( x_{rmno} \): Equals to 1 if the container \( (m, n, o) \) is loaded onto
the ship-bay and 0 otherwise.

3) Model Establishment: The objective is to minimize the
maximum completion time of all jobs while adhering to
constraints. The completion time \( T \) depends on \( w \) and \( R \), given
by \( T = \alpha w_s + \beta w_b + \gamma R \).

\[
\text{minimize, } T_{\text{max}} \quad (1)
\]
subject to,

\[
T U_c - T U_c \geq L_c \quad \forall c \in S \quad (2)
\]
\[
T U_i - T U_j + \mu X_{ij} \geq U_i \quad \forall j, i \in S \quad (3)
\]
\[
T U_j - T U_i + \mu (1 - X_{ij}) \geq U_j \quad \forall j, i \in S \quad (4)
\]
\[
T L_i - T L_j + \mu Y_{ij} \geq L_i \quad \forall j, i \in S \quad (5)
\]
\[
T L_j - T L_i + \mu (1 - Y_{ij}) \geq L_j \quad \forall j, i \in S \quad (6)
\]
\[
T U_c \geq U_c \quad \forall c \in S \quad (7)
\]
\[
h_{mn} \leq H_{mn} \quad (8)
\]
\[
X_{ij} \in \{1, 0\} \quad \forall j, i \in S \quad (9)
\]
\[
Y_{ij} \in \{1, 0\} \quad \forall j, i \in S \quad (10)
\]
\[
x_{rmno} \in \{1, 0\} \quad (11)
\]

These constraints fully define the model. Constraint (2)
ensures a stack is only loaded after necessary unloading.
Constraints (3), (4), (9), and (10) sequence unloading stacks
and ensure adequate time between them. Constraints (5), (6),
(9), and (10) do the same for loading. Constraint (7) ensures
sufficient time for unloading. Constraint (8) limits stack height.
Constraint (11) enforces binary conditions on flow variables.

B. QCDC-DR-GA

The paper addresses the optimization of unloading se-
quencies and dockyard arrangements for container ships. The
problem complexity is defined by \( S! \times N! \), representing the
permutations of stacks and containers, respectively. The aim
is to maximize dual cycles during unloading and minimize re-
handles in the dockyard, resulting in a complexity of \( S! \times N! \).

The genetic algorithm (GA) is employed as a metaheuristic
approach to finding high-quality solutions. With crossover
and mutation, a mixed GA is developed to handle both one-
dimensional (1-D) unloading sequences and two-dimensional
(2-D) dockyard plans. Key challenges include fitness calcu-
lation and integration of unloading sequences and dockyard
arrangements. The operating flow path of the QCDC-DR-GA
is illustrated in Fig. 4.

1) Set Initial Population: The initial population \( (P) \) com-
prises chromosomes representing unload sequences and dock-
yard plans. Notations include:

\( P = \) population of chromosomes
\( n = \) the number of chromosomes in \( P \)
Fig. 4: Methodological flowchart of the proposed QCDC-DR-GA.

$c_i$ is the $i^{th}$ chromosome in $P$, where $1 \leq i \leq n$

$c_{i\text{us}} = \text{part of chromosome representing unloading sequence}$

$c_{i\text{dp}} = \text{part of chromosome representing dockyard plan}$

$$A_1 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix};$$

$$A_2 = \begin{bmatrix} 3A & 3B & 1C & 1D \\ 1A & 2B \\ 2A & 1E & 1B \\ 3C & 2C & 3E & 3D \\ 2D & 2E & 4A & 4D \\ 4B & 4C & 3C \\ 4E \end{bmatrix};$$

The solution vectors $A_1$ and $A_2$ denote the unloading sequence and dockyard plan, respectively. Each row in $A_2$ represents a dockyard stack, with elements indicating the position of containers on ships. For instance, $3A$ denotes a container on the third stack, the first position of the selected ship row. $A_1$ and $A_2$ correspond to parts of $c_{i\text{us}}$ and $c_{i\text{dp}}$.

2) Crossover: We employ two different methods for crossover to handle the 1D and 2D parts of each chromosome.

### One Dimensional Crossover

We employ the Two-Point Crossover method, a special case of N-Point Crossover. Two random points on the individual chromosomes are selected, and genetic material is exchanged between these points. Common genes are retained, and any dropped genes are appended to the chromosomes (see Fig. 5).

![Fig. 5: 1D Two-Point Crossover technique.](image)

### Two Dimensional Crossover

For the 2D vector, we use the 2D Substring Crossover method. This involves row swap and column swap operations, which is a modified version of the 2D crossover introduced in the aircraft scheduling problem [13].

- **Row-wise operation**: Two random points are selected, and the entire row of the parents between these points is swapped.
- **Column-wise operation**: The column-wise operation is performed on the selected rows using the Two-Point Crossover method previously used for 1D vector crossover.

Repeated items are removed, and any dropped-out items are appended to the offspring (see Fig. 6).

3) **Mutation**: Mutation, akin to biological mutation, maintains genetic diversity between generations. We employed two methods for distinct chromosome parts. The mutation probability is denoted by $P_m$, and it occurs probabilistically, governed by algorithm 1.

#### Algorithm 1: Mutation Algorithm

**Input**: Two 1D vectors after newly crossed child

**Output**: Two 2D vectors as mutated child

1. Generate a random number $R$.
2. if $R > P_m$ then
3. \quad Do not do the mutation operation.
4. else
5. \quad Do mutation operation.

### One Dimensional Mutation

We utilize the Swap Mutation method for the 1D chromosome part, interchanging two selected genes after crossover (see Fig. 7).

### Two Dimensional Mutation

Here, we chose the 2D Two-Point Swapping Mutation method for the 2D part of our chromosome. This is also the modified version of the mutation method introduced by Tsai et al. [13]. The method is described in algorithm 2 (also see Fig. 8).
Notations:

- $R$: the number of rows in the 2D chromosome part.
- $C_{R_i}$: the number of columns in the $i^{th}$ row.

Algorithm 2: 2D Mutation Algorithm

**Input:** A 2D vector from newly crossed children

**Output:** A 2D vector as a mutated child

1. Randomly generate $r_1$ and $r_2$ to select two rows from the 2D vector, where $1 \leq r_1, r_2 \leq R$
2. Generate random integers $c_1$ and $c_2$ to select two points from the selected rows, where $1 \leq c_1 \leq C_{R_1}$ and $1 \leq c_2 \leq C_{R_2}$
3. Interchange the genes between the selected points of the 2D vector

4) Calculate Fitness: Chromosomes are evaluated based on their completion time, which is our objective function. The cost, representing the total completion time, is computed for each chromosome in every generation. This cost is then used to select the fittest chromosomes for the next generation. Details of the cost calculation are provided in algorithms 3 and 4.

5) Selection: We employ the Roulette Wheel selection technique, a probabilistic method favoring individuals with higher fitness. Unlike traditional roulette, our approach employs weighted probabilities based on fitness (Fig. 9). Notations used include $P_E$ for the percentage of elite chromosomes and $E_{rw}$ denoting the end value of the roulette wheel. The elite class, representing the fittest individuals, automatically advances to the next generation (with $P_E$ set at 20%).

Algorithm 5 outlines the steps of the roulette wheel selection process.

6) Termination: The termination condition of the GA determines when the run ends. Initially, the GA progresses quickly, yielding better solutions every few iterations. However, this
progress tends to slow down later, with minimal improvements. To guarantee that our solution approaches optimality, we establish a termination condition as follows: $g_i$ denotes the $i^{th}$ generation, $G$ represents the maximum number of generations, and $N_s$ stands for the number of successive

Algorithm 3: Cost function

**Input:** Loading plan, unloading plan, dockyard container arrangement, maximum dockyard container stack height

**Output:** No. of single cycles, no. of double cycles, no. of dockyard rehandles

1. **Function** `unload_first_stack` (unloading plan, unloading sequence):
   
   for container ∈ the dockyard stack of unloadingSequence do
   
   if the container will not stay on the vessel then
   
   unload the container
   
   no_of_single_cycles += 1

2. **Function** `calculate_rehandles` (target container):
   
   Let, no_of_rehandles ← 0
   
   Let, found_the_container ← false
   
   for $i$ ∈ stacks of dockyard do
   
   for $j$ ∈ containers of current stack do
   
   if $j$ = target container then
   
   found_the_container ← true
   
   while until containers are shifted from the top of the target container one by one do
   
   Shift the container nearest lowest stack
   
   return no_of_rehandles
   
   if found_the_container = false then
   
   Warning! container not found
   
   return 0

3. **Function** `loading_operation` (unloading plan, loading plan, unloading sequence):
   
   if the current loading stack is empty then
   
   if the current unloading stack is empty then
   
   go to the next loading stack
   
   else
   
   return false, 0

   Load the current container from dockyard

   return true, calculate_rehandles (current container to be loaded at dockyard)

   Let, no_of_single_cycles ← 0
   
   Let, no_of_double_cycles ← 0
   
   Let, no_of_rehandles ← 0
   
   unload_first_stack (unloading plan, unloading sequence)

Algorithm 4: Cost function (continued)

32. while until all the stacks are unloaded from ship do

33. for container ∈ current stack do

34. if the container will not stay on the vessel then

35. unload the container

36. if there is any container to load and any stack of the ship is free for loading then

37. flag, rehandles ← loading_operation (unloading plan, loading plan, unloading sequence)

38. no_of_rehandles += rehandles

39. no_of_double_cycles += 1

40. while complete loading the remaining stacks do

41. flag, rehandles ← loading_operation (unloading plan, loading plan, unloading sequence)

42. no_of_rehandles += rehandles

43. no_of_single_cycles += 1

Algorithm 5: Roulette Wheel Selection Algorithm

**Input:** Probability against the fitness value of each chromosome

**Output:** A selected chromosome

1. Define a 1D vector $RW$ of size $n$ for storing the fitness value of each chromosome. The fitness value is stored as a cumulative sum order where $E_{rw}$ is the total sum of all fitness.

2. for $i ← 1$ to $n$ do

3. Generate a random number $r$, where $0 ≤ r ≤ E_{rw}$.

4. Select a chromosome as a parent for crossover.

Fig. 9: Weighted roulette wheel.
generations where the fittest chromosome incurs the same cost. The genetic algorithm (GA) execution concludes according to the criteria specified in Algorithm 6.

Algorithm 6: GA termination algorithm

Input: Number of successive generations in which the cost of the fittest chromosome is the same and iteration number

Output: Boolean value to take termination decision

1. \( N_s \leftarrow \) Number of successive generations in which the fittest chromosome costs the same.
2. if \( g_i = G \) or \( N_s = 100 \) then
3. \( \) Terminate the GA run.
4. else
5. \( \) Continue

7) Parameters: The GA control parameters are shown in Table I. The parameters that best fit our model, such as population size, crossover technique, elite percentage, mutation probability, selection method, etc., are selected. As the solution to our problem is a smooth landscape type and the complexity of our problem is medium, we selected these parameters to fit the situation.

TABLE I: GA control parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>200</td>
</tr>
<tr>
<td>1D crossover strategy</td>
<td>Two-Point Crossover</td>
</tr>
<tr>
<td>2D crossover strategy</td>
<td>2D Substring Crossover</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>0.80</td>
</tr>
<tr>
<td>1D mutation strategy</td>
<td>Swap Mutation</td>
</tr>
<tr>
<td>2D mutation strategy</td>
<td>2D Two-Point Swapping Mutation</td>
</tr>
<tr>
<td>Mutation rate</td>
<td>0.30</td>
</tr>
<tr>
<td>Selection strategy</td>
<td>Roulette wheel</td>
</tr>
<tr>
<td>Elite class</td>
<td>0.20</td>
</tr>
<tr>
<td>Consecutive iterations</td>
<td>100</td>
</tr>
</tbody>
</table>

IV. RESULTS AND DISCUSSION

This section addresses the magnitude of QCDC-DR-GA. We offer tools to translate cycle-based benefits into time equivalents and validate those estimates against real-world double-cycling data. With an eye on the present and future, we analyze the financial impact of double cycling, estimating potential rewards for both existing vessels and those gracing the waves in the years ahead. The results of the experiments were obtained using a computer with 8 gigabytes of RAM, an Ubuntu 22.04 operating system, and an Intel Core i5 8th Gen. The algorithm was implemented using Python libraries- Pandas and NumPy.

A. Performance Comparisons of the Algorithms

We compared our QCDC-DR-GA algorithm with three established methods:

(a) Dual-Cycling Greedy Upper Bound Approach (Greedy UB): This heuristic sorts container stacks in descending order for dual-cycle loading/unloading without considering dockside rehandles [2].

(b) Mixed-Integer Programming Model for QDCS (bi-level GA): This method improves upon the Greedy UB by integrating QDCS optimization within a bi-level genetic algorithm framework [5].

(c) GA-ILSRS: Explored in two scenarios, this approach optimizes dockyard rehandles using a genetic algorithm combined with Iterated Local Search, neglecting loading/unloading in one scenario and focusing solely on dockyard rehandling in the other [12].

While the Greedy-upper-bound focuses solely on dual cycling, the QDCS-bilevel GA enhances it by incorporating QDCS optimization. Conversely, the GA-ILSRS solely optimizes dockyard rehandles. In contrast, our QCDC-DR-GA considers both dual cycling and dockyard rehandles, optimizing them using a sophisticated genetic algorithm approach.

B. Datasets

Six scenarios were created with varying numbers of stacks (5 to 30) and maximum stack heights (4 to 10) for container rows, reflecting typical container ship characteristics. Loading and unloading plans for each scenario, along with dockyard container arrangements, were generated by the program. Configuration details for the six datasets are summarized in Table II. Sample unloading and loading plans for a small ship are presented in Tables III and IV, respectively.

TABLE II: Loading-unloading plan configuration

<table>
<thead>
<tr>
<th>Scenario</th>
<th>No. of stacks</th>
<th>Maximum stack height</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

TABLE III: Unloading plan of a ship’s row

<table>
<thead>
<tr>
<th>Stack No.</th>
<th>Tier No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1A</td>
<td>1B</td>
<td>1C</td>
<td>1D</td>
<td>1E</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2A</td>
<td>2B</td>
<td>2C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>3A</td>
<td>3B</td>
<td>3C</td>
<td>3D</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4A</td>
<td>4B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5A</td>
<td>5B</td>
<td>5C</td>
<td>5D</td>
<td>5E</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>F</td>
<td>6A</td>
<td>6B</td>
<td>6C</td>
<td>6D</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>F</td>
<td>7A</td>
<td>7B</td>
<td>7C</td>
<td>7D</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>F</td>
<td>8A</td>
<td>8B</td>
<td>8C</td>
<td>8D</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>F</td>
<td>9A</td>
<td>9B</td>
<td>9C</td>
<td>9D</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>F</td>
<td>10A</td>
<td>10B</td>
<td>10C</td>
<td>10D</td>
<td></td>
</tr>
</tbody>
</table>

Tables III and IV provide container location information. For instance, 1B indicates a container located in the 1st stack, the 2nd tier of the row, with F indicating the container
remaining on the ship. Assuming a maximum stack height of 6, the dockyard plan is discussed further in Section III-B1 as $A_2$. Detailed generated data is available as supplementary material at “https://dx.doi.org/10.21227/cj08-qn62” due to spatial constraints.

C. Numerical Tests

The six scenarios detailed in subsection IV-B serve as the basis for numerical testing. Processing times for QCs in single and dual cycling, as reported by Goodchild [2], are 90 and 170 seconds, respectively. Container rehandling time by a gantry crane at the dockyard follows a uniform distribution of 60 seconds.

1) Test Results: We thoroughly assess the QCDC-DR-GA algorithm’s performance, comparing it with four established methods for port container handling optimization. The evaluation utilizes the six datasets, representing various scenarios with differing container numbers and ship configurations. Table V presents simulation results from each method on these datasets. Additionally, Fig. 10 visually juxtaposes QCDC-DR-GA’s performance with that of other approaches.

2) Key Findings: The key findings and remarks of the simulation are as follows:

- The proposed QCDC-DR-GA model consistently outperforms other methods by maximizing dual cycles and minimizing container handling. This demonstrates its effectiveness in optimizing the total unloading-loading time.
- Combining QCDC optimization with dockyard rehandle minimization in QCDC-DR-GA yields superior results compared to fragmented approaches like QCDC Scheduling Optimized by bi-level GA and GA-ILSRS (Scenario 2).
- Neglecting dual-cycling in QC operation optimization, as seen in GA-ILSRS (Scenario 1), leads to inferior performance compared to QCDC-DR-GA. This underscores the importance of simultaneous consideration of both aspects for optimal resource utilization.

D. Significance Test

A two-tailed paired t-test compared the operation times given by the QCDC-DR-GA strategy with others for the numerous datasets.

V. CONCLUSIONS

Utilizing a heuristic approach, we proposed a hybrid QCDC-DR-GA algorithm that optimizes ship unloading and loading processes using dual cycling and reduces dockyard rehandles. Our model consistently outperforms existing methods across the six scenarios of datasets that we created from small to large, with particularly notable improvements for large vessels. However, certain limitations warrant consideration. The model assumes immediate loading container availability at the dockside, potentially neglecting pre-staging requirements. Furthermore, it focuses solely on rehandling between the ship and the apron, overlooking potential relocations within ship stacks. Lastly, disruptions from inbound vehicles or cranes are not accounted for, possibly leading to underestimating operational variability. Future research avenues could address these limitations by incorporating pre-staging needs, exploring intra-ship rehandle optimization, and integrating dynamic disruptions to enhance practical applicability.

REFERENCES

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>No. of stacks</th>
<th>Maximum stack height</th>
<th>Strategies</th>
<th>Operation time (min)</th>
<th>1(\beta(0.05) = 2.093)</th>
<th>Improvement of QCDC-DR-GA (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
<td>10</td>
<td>Proposed QCDC-DR-GA</td>
<td>615.25 704.92 667.62</td>
<td>25.29</td>
<td>14.44%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>718.58 825.92 870.33</td>
<td>33.04</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>714.92 811.25 763.00</td>
<td>31.26</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>669.23 764.25 708.95</td>
<td>28.18</td>
<td>8.53%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>820.75 935.95 835.09</td>
<td>9.95</td>
<td>20.35%</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>10</td>
<td>Proposed QCDC-DR-GA</td>
<td>522.42 604.75 559.02</td>
<td>27.24</td>
<td>17.08%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>645.33 710.75 674.20</td>
<td>20.93</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>652.17 722.75 672.10</td>
<td>22.52</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>535.42 631.08 579.37</td>
<td>28.28</td>
<td>3.51%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>666.01 761.00 680.79</td>
<td>8.11</td>
<td>17.89%</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>10</td>
<td>Proposed QCDC-DR-GA</td>
<td>381.42 452.08 417.35</td>
<td>22.53</td>
<td>20.29%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>491.08 563.08 523.60</td>
<td>19.66</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>484.08 563.08 519.85</td>
<td>25.76</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>397.42 465.75 425.77</td>
<td>19.96</td>
<td>1.98%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>492.25 511.50 362.76</td>
<td>4.98</td>
<td>16.99%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Proposed QCDC-DR-GA</td>
<td>201.67 259.00 229.01</td>
<td>18.85</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>8</td>
<td>Proposed QCDC-DR-GA</td>
<td>295.33 358.67 327.83</td>
<td>18.90</td>
<td>30.14%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>295.50 349.33 323.00</td>
<td>17.54</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>291.53 349.33 323.00</td>
<td>17.54</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>211.33 277.61 248.65</td>
<td>17.79</td>
<td>7.90%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>279.00 295.00 266.56</td>
<td>5.52</td>
<td>20.08%</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>5</td>
<td>Proposed QCDC-DR-GA</td>
<td>89.50 138.67 119.40</td>
<td>16.39</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>138.50 186.67 164.44</td>
<td>14.66</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>147.00 183.67 162.13</td>
<td>12.50</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>97.00 147.67 123.13</td>
<td>15.56</td>
<td>3.03%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>137.00 185.00 165.76</td>
<td>4.52</td>
<td>18.09%</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>Proposed QCDC-DR-GA</td>
<td>37.67 70.42 55.43</td>
<td>10.75</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Greedy Upper Bound</td>
<td>48.67 81.42 63.82</td>
<td>12.87</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QCDC-bi-level-GA</td>
<td>47.67 80.42 65.60</td>
<td>10.75</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-1</td>
<td>37.67 70.42 54.48</td>
<td>10.78</td>
<td>13.15%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GA-ILSRS-Scenario-2</td>
<td>55.50 66.00 60.05</td>
<td>3.05</td>
<td>7.07%</td>
</tr>
</tbody>
</table>
Md Abrar Jahin, (Certified Supply Chain Analyst), is an AI researcher and multidisciplinary scientist who obtained his B.Sc. Eng. degree in industrial & production engineering from Khulna University of Engineering & Technology (KUET) in Bangladesh. Abrar’s exceptional leadership skills and innovative thinking led him to win the “Smart Roads – Winter Road Maintenance Hackathon 2021” and subsequently earn a three-week research internship at UIT - The Arctic University of Norway. During this internship, he actively contributed to the DIT4BEARs Smart Roads project and conducted research in the field of Machine Learning. In 2021, he received a fully funded scholarship as a ‘Research Intern’ at Japan’s Okinawa Institute of Science and Technology Graduate University (OIST). Under the supervision of Prof. Miller in the Physics and Biology Unit, he focused on investigating perfectly conserved sequences in humans and other species. Building on his previous achievements, he was offered another opportunity in 2023 as a ‘Visiting Research Student’ at OIST. He has been actively collaborating as a ‘Visiting Researcher’ since 2024 with Prof. Miller, Dr. Biller, Dr. Zifcakova, Mr. Parker, and Dr. Pnini, delving into the evolutionary dynamics of strongly conserved sequences in vertebrates and insects. Mr. Jahin’s research interests span a wide range of disciplines, including Machine Learning, Deep Learning, Quantum Computing, Explainable AI (XAI), Natural Language Processing (NLP), Physics-Informed Neural Networks (PINN), Conformal Prediction, Comparative Genomics, Human-Computer Interaction (HCI), and Supply Chain Analytics. He serves as a peer reviewer for IEEE Access, contributing his expertise to ensure the quality and rigor of scholarly publications in his field.

Md. Saiful Islam has been working as an Assistant Professor in the Department of Industrial Engineering and Management at Khulna University of Engineering & Technology, Khulna, Bangladesh, since 2017. He received his B.Sc. Eng. in industrial & production engineering degree in 2013 and M.Sc. Eng. in Industrial Engineering and Management degree from the Department of Industrial Engineering and Management, Khulna University of Engineering & Technology in 2018. His main research areas are warehouse, inventory, and supply chain management. He has published more than 10 research articles in the corresponding fields.

Dr. M. F. Mridha (Senior Member, IEEE) received a B.Sc. degree in computer science and statistics and an M.Sc. degree in computer science from Pusan National University, South Korea, in 1990 and 1994, respectively, and a Ph.D. degree in computer science and communication engineering from Kyushu University, Japan, in 1999. He was an Associate Professor, a Senior Associate Professor, and a Full Professor at the School of Computer Science and Engineering, The University of Aizu, Japan, in 1999, 2004, and 2019, respectively. He currently working as an Associate Professor in the Department of Computer Science at American International University-Bangladesh (AIUB). Before that, he worked as an Associate Professor and chairman in the Department of Computer Science and Engineering at the Bangladesh University of Business and Technology (BUBT). He also worked as a CSE department faculty member at the University of Asia Pacific and as a graduate head from 2012 to 2019. He received his Ph.D. in AI/ML from Jahangirnagar University in 2017. His research experience in academia and industry has resulted in over 120 journal and conference publications. His research contributed to the reputed Journal of Scientific Reports Nature, Knowledge-Based Systems, Artificial Intelligence Review, IEEE Access, Sensors, Cancers, Biology and Applied Sciences, etc. His research interests include artificial intelligence (AI), machine learning, deep learning, and natural language processing (NLP). For more than 10 (Ten) years, he has been with the masters and undergraduate students as a supervisor of their thesis work. His research interests include artificial intelligence (AI), machine learning, natural language processing (NLP), big data analysis, etc. He has been a program committee member in several international conferences/workshops. He served as an academic editor for several journals, including PLOS ONE Journal. He has served as a reviewer of reputed journals like IEEE Transactions on Neural Networks, IEEE Access, Knowledge-Based Systems, Expert Systems with Applications, Bioinformatics, Springer Nature, MDPI, etc. and conferences like ICCIT, HONET, ICIEV, IJCCI, ICAEE, ICCAIE, ICSIPIA, SCORED, ISIEA, APACE, ICOS, ISCAIE, BIELAC, ISWTA, IC3e, ISWTA, CoAST, icIVPR, ICSCT, 3ICT, DATA21, etc.

Junggil Shin (Senior Member, IEEE) received a B.Sc. degree in computer science and statistics and an M.Sc. degree in computer science from Pusan National University, South Korea, in 1990 and 1994, respectively, and a Ph.D. degree in computer science and communication engineering from Kyushu University, Japan, in 1999. He was an Associate Professor, a Senior Associate Professor, and a Full Professor at the School of Computer Science and Engineering, The University of Aizu, Japan, in 1999, 2004, and 2019, respectively. He currently works as an academic editor for several major IEEE and SCI journals.