Marine Quay Crane Scheduling Using a Combined Modified Genetic Algorithm and Priority Rules Approach

Vi H. Nguyen and Duc T. Nguyen

Abstract—**Quay crane scheduling problem (QCSP) is the problem of the allocation of quay cranes to handle the unloading and loading of containers at seaport container terminals and defining the service sequence of vessel bays of each quay crane. The treatment of crane interference constraints and the increased in vessel size make the problem difficult to solve. Due to the growing interest in applied research for this problem, many researchers have used different algorithms and methods to obtain some solutions. This paper will propose a modified genetic algorithm combined with priority rules to deal with it. The advantage of the proposed algorithm comes from the fact that crane interference constraints, non-simultaneous constraints and precedence constraints can all be handled in simple ways and is to provide practical solutions. The effectiveness and reliability of the approach are tested using several benchmark instances proposed by Meisel and Bierwirth (2011). A comparison with the current best-known solutions reveals that the proposed method is capable of finding the optimal or near-optimal solution in reasonable computing time.**

Index Terms—**Crane scheduling, genetic algorithm, optimization, priority rules.**

I. Introduction

Containers are large boxes that are used to transport goods from one destination to another by several transportation systems. Transport containers oversea are carried out by vessels. Container terminals are the place at which vessels are charged and discharged. To perform container discharging and charging operations from a vessel when it is berthed, quay cranes (QCs) are critical resources in container terminals. The QCs move along the quay on rails to take/put containers off/on the deck and holds. They are the most expensive equipment at terminals and can be used both at an automated and a manned terminal. Nowadays due to the large growth rates of sea transportation, terminals are faced with more and more containers to be handled. Moreover, because of the high competition between container terminals, improving QCs operating efficiency is very important to achieve high productivity (short completion time at low cost) and customer satisfaction. Fig. 1 shows a picture of quay cranes operating to load or unload containers to and from the vessel.

QCSP refers to finding an optimal schedule for QCs serving a vessel in which the allocation of quay cranes to handle the unloading and loading tasks, sequencing these tasks on each QC, and task's operation start time is specified to get the scheduling targets. It was firstly described by Daganzo (1989). He stated that the QCSP seems related to the machine scheduling field, parallel machine scheduling problem, and a resource-constrained project scheduling problem with precedence constraints but more complicated. His paper examined crane scheduling for ports, studied the problem of QC assignment among different holds on multiple vessels to minimize the ships' delay at berth. As a result, crane idle time is minimized and berth throughput maximized, and this reduces queuing delay as well [2]. In a related study, Peterkofsky and Daganzo (1990) proposed a branch and bound algorithm to deal with the QCSP problem with the objective of minimizing total weighted tardiness. However, in these two studies, QCs interference and safety distance between two adjacent QCs which are important features of QC operation are not considered [3].

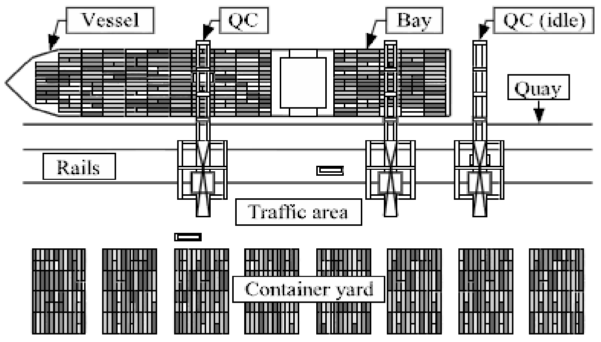


Fig. 1. A drawing of quay cranes working on vessel [1].

Lim *et al.* (2004) further considered spatial constraints which related to the relative positions of cranes and jobs in QCSP. Three types of spatial constraints are identified which are non-crossing constraints (or inference constraint), neighborhood constraint (or safety distance constraint) and job-separation constraint (or non-simultaneous constraint). They adopted dynamic programming, tabu search, as well as squeaky wheel optimization heuristics to solve the problems with both separately and simultaneously constraints cases [4]. Kim and Park (2004) formulated the quay crane scheduling problem and proposed a branch and bound method and a heuristic algorithm, Greedy randomized adaptive search procedure for solving the QCSP. In the formulation, the QCSP with multiple tasks involved in a ship-bay is considered with crane interference, safety distance constraints as well as non-simultaneous and precedence relations among tasks [5]. Moccia *et al.* (2006) revised Kim and Park's model by incorporating travel times for the quay cranes that subsequently process tasks in the same bay. They proposed a branch-and-cut algorithm as the

Manuscript received September 9, 2018; revised January 15, 2019.

Vi H. Nguyen is with Faculty of Engineering, Vietnamese-German University, Binh Duong, Vietnam (e-mail: vi.nh@vgu.edu.vn).

Duc T. Nguyen is with Civil and Environmental Engineering (CEE) Department, Old Dominion University, Virginia, USA (e-mail: dnguyen@odu.edu).

solution methodology and applied to solve the same benchmark instances used by Kim and Park (2004). Their approach yielded better solutions than Kim and Park (2004) and is able to handle the bigger size of practical problems [6]. Later on, Sammarra *et al.* (2007) solved the QCSP, which have been studied by Kim and Park (2004) and Moccia *et al.* (2006) under the same assumption made, by separating the problem into two parts, routing problem and the scheduling problem. They proposed a local search algorithm for the scheduling sub-problem and a Tabu Search (TS) algorithm for routing problem. Their methods can obtain almost optimal solutions in the medium-size problem with reasonable computation time [7]. Bierwirth and Meisel (2009) revised the QCSP model proposed by Sammarra *et al.* (2007) and developed a heuristic based on the branch-and-bound algorithm. In comparison with existing methods, their proposed heuristic outperformed in terms of both the objective function value and the computation time [8], [9]. Meisel and Bierwirth (2011) proposed a unified approach for evaluating the performance of different model classes and solution procedures. In their paper, they also proposed a set of benchmark instances with a computational results document for each instance [1].

S.H. Chung and K.L. Choy proposed a modified genetic algorithm to deal with the QCSP. Their model is based on the model developed by Kim and Park (2004) and the modification made by Moccia *et al.* (2006). They used a new approach for defining the chromosomes: a chromosome to have two parts. In the first part, from left to right, genes represent the tasks and their performance sequence whereas in the second part, they represent the quay cranes assigned to each task of the first part. They adopted order crossover proposed by Gen and Cheng (1996) and swapping mutation in their proposed algorithm. Then set of well-known benchmarking data obtained from Kim and Park (2004) for testing and comparing. They yielded that the proposed algorithm performs as good as many existing algorithms and it is much faster than the existing approaches [10]. Narges Kaveshgar *et al.* (2012) extends the research in the QCSP area by utilizing the GA that is available in the latest version of Global Optimization Toolbox in MATLAB 7.13 to facilitate development. The mathematical formulation used for the QCSP in their paper is based on the one developed by Kim and Park (2004). By using an initial solution based on the S-LOAD rule developed by Sammarra, Cordeau, Laporte, and Monaco (2007), using a new approach for defining the chromosomes (i.e., solution representation) to reduce the number of decision variables, and using new procedures for calculating tighter lower and upper bounds for the decision variables, they improved the efficiency of the GA search. Their proposed GA is capable of finding the optimal or near-optimal solution in a significantly shorter time when comparing to the current best-known solutions of benchmark instances proposed by Meisel and Bierwirth (2011) [11].

All of the above algorithms require the users and readers to have a strong background in mathematics and programming. With the goal to develop a simple and effective method to solve the QCSP, this paper will propose a modified generic algorithm combined with simple priority rules which do not require mathematical formulation. By testing benchmark instances which are downloaded from [15], and comparing our results with the current best-known solutions, it can be shown that the proposed method can get to the optimal or near-optimal solutions in reasonable computing time.

II. Problem Description

In the QCSP for container groups, there is a set of tasks Ω = {1, 2… n} and a set of QCs Q = {1, 2. . . q}. Each task i ∈ Ω represents a loading or discharging operation of a certain container group. The tasks have individual processing times and bay positions. They must be processed by a QC without preemption. Pairs of tasks which are located within the same bay may have precedence relation among tasks: when discharging and loading operations must be performed at the same ship-bay, the discharging operation must precede the loading operation. When discharging operation is performed in a ship-bay, tasks on the deck must be performed before tasks in the hull of the same ship-bay are performed. Also, the loading operation in a hold must precede the loading operation on the deck of the same ship-bay [3]. Crane interference constraints are involved in the QCSP because QCs are rail mounted: All QCs can move between two adjacent bays in an identical travel time t> 0. They are not allowed to cross each other and have to keep a safety margin δ, measured in units of bays at any time. Certain tasks cannot be performed simultaneously because, at the same time, no two QCs can operate at the same bay or at two bays that do not satisfy the safety distance requirement.

The objective of QCSP is to determine the allocation of quay cranes to handle the unloading and loading tasks, sequencing these tasks on each QC, and task's operation start time so that the completion time of a ship operation is minimized.

TABLE I: A SMALL QCSP INSTANCE [9].

Task index i	1	2	3	4	5	6	7	8	9
Processing time	22	46	8	70	10	38	40	16	12
Bay position	1	1	2	3	5	5	7	9	11
Precedence-constrained task	$\phi = \{(1,2),(5,6)\}$								
Non-simultaneous tasks	$\psi = \{(1,2),(1,3),(2,3),(3,4),(5,6)\}$								
QC1	$l_0^1 = 1, r^1 = 0$								
QC2	$l_0^2 = 4, r^2 = 0$								
QC travel time	t=1								
Safety margin	$\delta = 1$								

Table 1 shows the data for a small QCSP instance which is used in the paper "A fast heuristic for quay crane scheduling with interference constraints" by Christian Bierwith and Frank Meisel to illustrate the proposed solution procedure. The problem contains nine container groups placed in a vessel with eleven bays. Two QCs are assigned to this vessel.

The interpretation of the above restriction is as follow

$\phi = \{(1,2),(5,6)\}$: Task 1 must precede task 2; Task 5 must precede task 6.

$\psi = \{(1,2),(1,3),(2,3),(3,4),(5,6)\}$: Task 1 and 2 cannot be performed simultaneously, etc.

QC1 $l_0^1 = 1, r^1 = 0$: Location of QC1 at time 0 is 1, i.e. bay 1. Ready time of QC1 is 0

QC2 $l_0^2 = 4, r^2 = 0$: Location of QC2 at time 0 is 4, i.e. bay 4. Ready time of QC2 is 0

t=1: 1 time unit to travel from bay to bay

$\delta = 1$: Safety margin between any 2QC's is 1 bay

III. Combine Modified Generic Algorithm and Priority Rules

Genetic algorithm (GA) is a stochastic search method that mimics the metaphor of natural biological evolution. Genetic algorithm starts with no knowledge of the correct solution and depends completely on responses from its environment and evolution operators (i.e. reproduction, crossover, and mutation) to arrive at the best solution [12]. In this paper, traditional GA will be modified then combine with the priority rule to solve the QCSP.

A. Chromosome Representation

GA starts with a random population of individuals which are called chromosomes. Each chromosome represents a possible solution. A chromosome of the modified GA represents a sequence handling of tasks in which a gene is a task number.

Chromosome	2	3	1	6	7	8	4	9	5
Handling Sequence	1	2	3	4	5	6	7	8	9

Fig. 2. A sample chromosome.

B. Validation of Chromosome with Precedence-Constrained

Check Chromosome to make sure that it satisfies the precedence constraint. If it does not satisfy, the constraint then swap the task positions to make it satisfactory.

In the example, $\phi = \{(1,2),(5,6)\}$ means task 1 must be processed before task 2 and task 5 must be handled before task 6. Therefore, the sample chromosome is modified as follow Fig. 4.

Non-interference constraints and non-simultaneously constraints will be handled when calculating objective function.

C. Assigning QCs and Calculating Objective Function by Applying Priority Rules

Based on the handling sequence of tasks represented by the chromosome, the QCs objective function, which is makespan, can be calculated by using the following procedure.

Step 1: Determine the ready time and current location of each quay crane, the checking time is the minimum time of all the ready times.

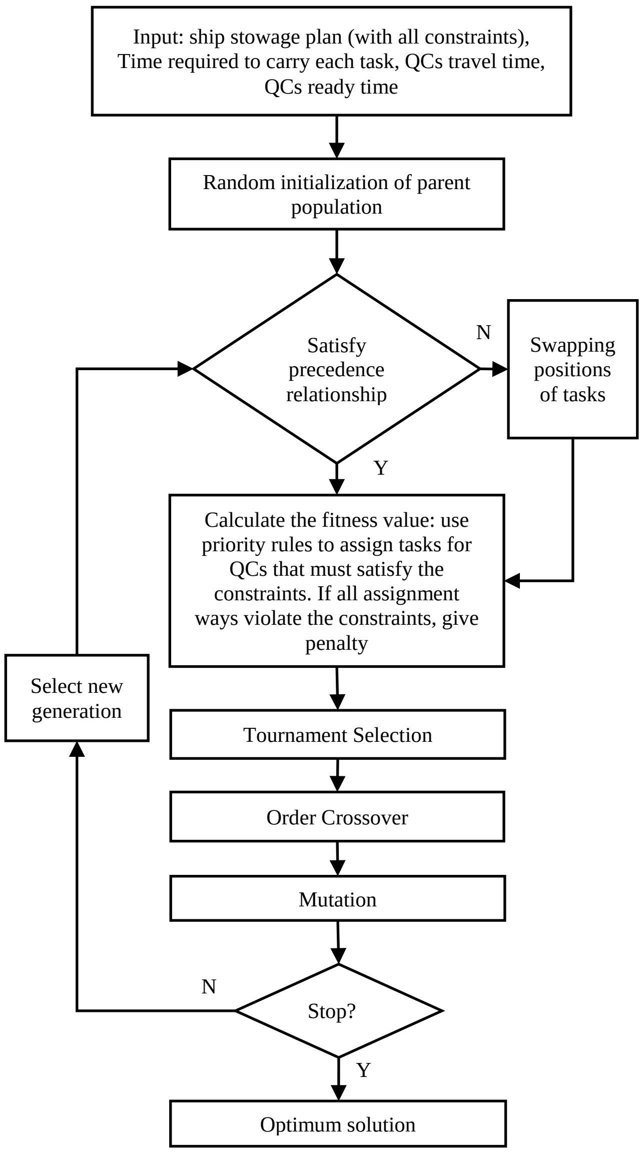

Input: ship stowage plan (with all constraints), Time required to carry each task, QCs travel time, QCs ready time
Random initialization of parent population
Satisfy precedence relationship
N
Swapping positions of tasks
Y
Calculate the fitness value: use priority rules to assign tasks for QCs that must satisfy the constraints. If all assignment ways violate the constraints, give penalty
Select new generation
Tournament Selection
Order Crossover
Mutation
N
Stop?
Y
Optimum solution

Fig. 3. Flowchart of methodology using modified GA combined priority rules.

Modified chromosome	1	3	2	5	7	8	4	9	6
Sequence handling	1	2	3	4	5	6	7	8	9

Fig. 4. Modified the sample chromosome in Fig. 3.

Step 2: At the checking time, determine which quay cranes can handle the first unassigned task indicated in the chromosome that does not interfere with the other quay cranes and also satisfy the safety constraint between adjacent cranes as well satisfy the non-simultaneous clause.

No interference with the other cranes means that the crane in question in trying to reach a target location cannot cross over the other cranes. The safety constraint implies that the minimum physical distance between any two cranes must equal the safety distance value. The non-simultaneous clause requires some tasks designated to be placed in some bays not be executed at the same time. There are three possible cases at each checking time.

Case 1: There is one or more cranes available at checking-time and can handle target job: move to step 3: Apply the assignment priority rules.

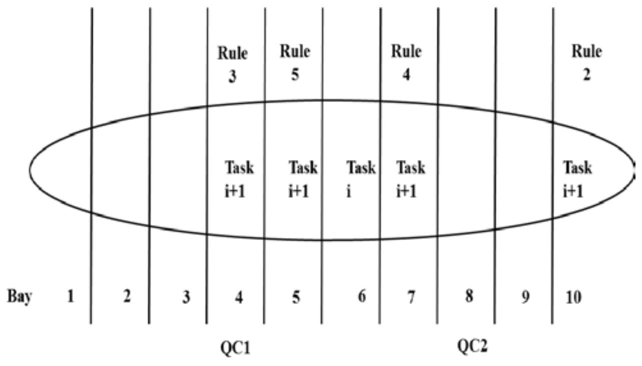


Fig. 5. Illustration of priority rules for assigning QC's to task.

Case 2: There is one or more cranes can handle target job but they are not available at checking time. They should remain in place and continue to work on their assignment. For cranes which have finished their jobs will remain idle. The checking time is increased to the next minimum cranes 'ready time. The longer idle time of cranes makes the makespan value worse. Therefore, an optimal makespan will have the minimum quay crane's idle time. Update the current position, the completion time, and the ready time of the quay crane at the new checking time (To track the current locations of quay cranes exactly with the checking time). Repeat step 1 and step 2.

Case 3: There is no crane that is able to handle the target job: add a penalty to the fitness value of the solution.

Step 3: Priority rules for assigning QC's to task

Two assumptions are made in this paper:

1 - The bays are identified in increasing order from left to right, i.e. bay 1, bay 2, bay 3, etc.

2 - The QC's are identified in increasing order from left to right, i.e. QC1, QC2, QC3, etc.

Case 1: Task i is at a location non-equidistant from either crane, then it will be assigned to the closest crane (rule 1).

Case 2: Task i is at a location equidistant from either crane. The allocation of task I follows from the following rules:

Rule 2: If the location ID of the next task, task i+1, is higher than or equal to the location ID of the crane with higher ID, i.e. QC2, then assign task i to the crane with the lower ID, i.e. QC1.

Rule 3: If the location ID of the next task, task i+1, is lower than or equal to the location ID of task i and that of the crane with higher ID, i.e. QC2, then assign task I to the crane with the higher ID, i.e. QC1.

TABLE II: AN ILLUSTRATION OF THE APPLICATION OF THE PRIORITY RULES

Bay location	1	2	3	4	5	6	7	8	9	10	11
Initial QC location	QC1			QC2							
Task-Bay requirements	1&2	3	4		5&6		7		8		9
Feasible sequence of task (chromosome)	1	3	2	5	7	8	4	9	6		
Bay requirement	1	2	1	5	7	9	3	11	5		

		QC1 ready time	QC1 location	QC2 ready time	QC2 location	Checking time
Initial state		0	1	0	4	0
Iter 1	Rule 1: Task 1 to QC1	22	1	0	4	0
Iter 2	Case 2: only crane 1 can handle task 3 but not available	22	1	22	4	22
Iter 3	Rule 1: Task 3 to QC1	31	2	22	4	22
Iter 4	Case 2: only crane 1 can handle task 2 but not available	31	2	31	4	31
Iter 5	Rule 1: Task 2 to QC1	78	1	31	4	31
Iter 6	Rule 1: Task 5 to QC2	78	1	42	5	42
Iter 7	Rule 1: Task 7 to QC2	78	1	84	7	78
Iter 8	Case 2: only crane 2 can handle task 8 but not available	84	1	84	7	84
Iter 9	Rule 1: Task 8 to QC2	84	1	102	9	84
Iter 10	Rule 1: Task 4 to QC1	156	3	102	9	102
Iter 11	Rule 1: Task 9 to QC2	156	3	116	11	116
Iter 12	Only crane 2 available at checking time and can handle task 6, then it to QC2	156	3	160	5	156

Rule 4: If the location ID of the next task, task i+1, is between the location ID of task i and that of the crane with higher ID, i.e. QC2, then assign task i to the crane with the lower ID, i.e. QC1.

Rule 5: If the location ID of the next task, task i+1, is between the location ID of task i and that of the crane with lower ID, i.e. QC1, then assign task i to the crane with the higher ID, i.e. QC2.

The start time of the target task is made equal to the ready time of the assigned quay crane. Update the current crane position, its completion time, and its ready time.

Step 4: Step 1-3 are repeated for the next task in the sequence.

The following illustration is to show how to calculate the objective function value for the example shown in Table 1. Table 2 provides an illustration of the application of the priority rules elaborated above.

Finally, makespan = 160.

IV. Tournament Selection

From the population, random choose *S* competitors, with *S* being the tournament size, then figure out the winner of the tournament which is the individual with the highest fitness of the *S* tournament competitors. The winner is then inserted into the mating pool. Tournament winners make the mating pool have higher average fitness than the average population fitness. Then from the mating pool, the parents are selected for mating to create the succeeding generation [13].

V. Order Crossover (OX)

Then from the mating pool, the parents are randomly selected for mating by crossover to create the succeeding generation. Order crossover proposed by Gen and Cheng (1997) [14] is applied.

1) From the mating pool, two parent chromosomes are given. Two random crossover points are selected from one parent, partitioning it into a left, middle and right substring.
2) Produce the first offspring chromosome by copying the middle substring of the first parent into the corresponding positions.
3) Delete the tasks which exist in the substring of the offspring from the second parent. The remaining tasks in the second parent are copied to the child from left to right according to the sequence in the second parent.
4) The second offspring copies the middle substring of the second parent into the corresponding positions. Delete the tasks which are already in the middle substring from the first parent. The remaining tasks of the first parent are placed into the second child from left to right to the order of the sequence.

VI. Reverse Sequence Mutation (RSM)

Reverse sequence mutation is used to mutate offspring chromosomes. In the reverse sequence mutation operator, we take a sequence S limited by two random positions. The gene order in this sequence will be reversed.

Parent	7	9	**3**	**4**	**5**	**6**	**1**	2	8
Child	7	9	**1**	**6**	**5**	**4**	**3**	2	8

Fig. 6. Illustration of inverse mutation.

The number of offspring chromosomes that have mutations in a population is determined by the mutation rate parameter.

TABLE III: Comparison of the Proposed Methodology with Current Best Solution as Reported in Meisel and Bierwirth (2011) and in Narges Kaveshgar (2012)

Ex. no.	Size (cranes x tasks)	Meisel and Bierwirth (2011)	Our modified GA combined priority rule	Developed GA by Narges Kaveshgar - 2012	Gap between our GA and Meisel
1	2x10	520	520	520	0
2		508	508	508	0
3		513	513	513	0
4		510	510	510	0
5		515	515	515	0
6		513	513	513	0
7		511	511	511	0
8		513	513	513	0
9		512	512	512	0
10		549	549	549	0
1	2x15	514	514	514	0
2		507	507	507	0
3		515	515	515	0
4		513	514	516	0.01
5		507	507	507	0
6		508	511	513	0.03
7		507	507	508	0
8		508	508	513	0
9		507	507	507	0
10		513	513	514	0
1	2x20	508	508	509	0
2		509	509	514	0
3		509	509	509	0
4		509	509	513	0
5		506	507	507	0.01
6		508	508	508	0
7		507	507	507	0
8		510	510	510	0
9		508	508	508	0
10		507	507	511	0
1	2x25	508	510	513	0.02
2		507	507	513	0
3		507	507	507	0
4		507	507	507	0
5		507	507	507	0
6		507	507	507	0
7		508	508	508	0
8		507	507	507	0
9		506	506	507	0
10		506	510	513	0.04
1	4x50	774	786	784	0.12
2		771	790	782	0.19
3		772	798	784	0.26
4		765	808	803	0.43
5		762	786	792	0.24
6		765	798	769	0.33
7		782	800	782	0.18
8		761	810	781	0.49
9		798	810	860	0.12
10		759	802	792	0.43

VII. New Generation Selection

In this paper, we apply the Steady-State method for the new generation selection: *n* worst old parents are deleted and are replaced them with *n* best new offspring. *n* is a parameter to be experimented with.

VIII. Experimental Results and Discussion

Fig. 7 shows the grant chart of the small QCSP instance in Table 1. It specifies the allocation of quay cranes to handle tasks, sequencing these tasks on each QC, and the operation start-time and complete time of tasks. Optimum makespan of this instance is 145 (days).

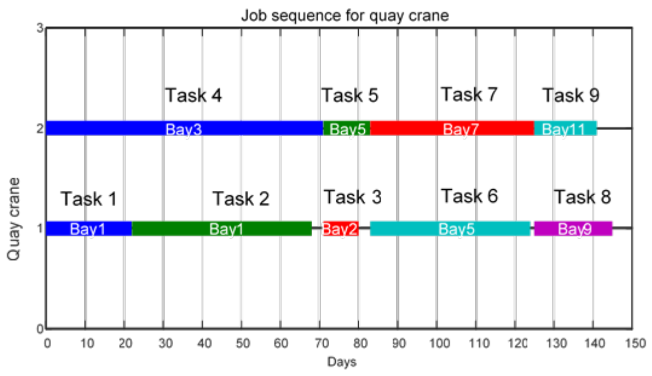


Fig. 7. Grant chart for optimum solution of the small QCSP instance in Table 1.

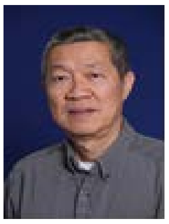
To evaluate the effectiveness of the proposed methodology, the experiments used benchmark data provided by Meisel and Bierwirth (2011) which can be downloaded from [15] and compare its performance with the best-known solutions as reported in Meisel and Bierwirth (2011). The parameters used in the proposed algorithm are: population size = 2x number of crane x number of tasks (individuals), crossover rate = 0.9, the mutation probability = 0.1, and the tournament size = 4. The new generation is created by 50% of best parents combined with 50% of best offsprings. The number of evolution = 10 x population size. The algorithm is programmed by using Matlab Language and performed on an Intel ® Core ™2, CPU 2.13 GHz, RAM 4.00 GB, 64-bit system.

Based on the compared results in Table 3, the modified GA combined priority rule is capable to find the optimum or very close optimum solution for all the benchmark instances although it is very simple to apply. The precedence constraints are handled effectively by the method to build chromosome and to validate them. The priority rules help to assign QCs to do jobs efficiently and avoid interface. The computation time of this method is affected by the size of the problem. It is increased when the size of the problem increases. The processing time to solve these instances ranges from 3 to 60 seconds which is reasonable and acceptable. Overall, the modified GA combined priority rule proposed a simple, effective and efficient way to solve the difficult quay crane scheduling problem with crane interference constraints. Comparing with other algorithms or methods which require the users and readers to have a strong background in mathematics and programming, the proposed modified generic algorithm combined with simple priority rules does not require mathematical formulation but still is capable of finding the optimal or near-optimal solution in reasonable computing time.

References

[1] F. Meisel and C. Bierwirth, “A unified approach for the evaluation of quay crane scheduling models and algorithms,” *Computers & Operations Research*, vol. 38, pp. 683-693, 2011.
[2] C. F. Daganzo, “The crane scheduling problem,” *Transportation Research Part B*, vol. 23, no. 3, pp. 159-175, 1989.
[3] R. I. Peterkofsky and C. F. Daganzo, “A branch and bound solution method for the crane scheduling problem,” *Transportation Research Part B: Methodological*, vol. 24, no. 3, pp. 159-172, 1990.
[4] A. Lim, B. Rodrigues, F. Xiao, and Y. Zhu, “Crane scheduling with spatial constraints,” *Naval Research Logistics*, vol. 51, pp. 386-406, 2004.
[5] K. H. Kim and Y. M. Park, “A crane scheduling method for port container terminals,” *European Journal of Operational Research*, vol. 156, no. 3, pp. 752-768, 2004.
[6] L. Moccia, J. F. Cordeau, M. Gaudioso, and G. Laporte, “A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal,” *Naval Research Logistics*, vol. 53, no. 1, pp. 45-59, 2006.
[7] Sammarra *et al.*, “A tabu search heuristic for the quay crane scheduling problem,” *Journal of Scheduling*, vol. 10, no. 4-5, pp. 327-336, 2007.
[8] C. Bierwirth and F. Meisel, “A fast heuristic for quay crane scheduling with interference constraint,” *Journal of Scheduling*, vol. 12, pp. 345-360, 2009.
[9] C. Bierwirth and F. Meisel, “A survey of berth allocation and quay crane scheduling problems in container terminals,” *European Journal of Operational Research*, vol. 202, pp. 615-627, 2010.
[10] S. H. Chung and K. L. Choy, “A modified genetic algorithm for quay crane scheduling operations,” *Expert System with Applications*, vol. 39, pp. 4213-4221, 2012.
[11] N. Kaveshgar, N. Huynh, and S. K. Rahimian, “An efficient genetic algorithm for solving the quay crane scheduling problem,” *Expert System with Applications*, vol. 39, pp. 13108-13117, 2012.
[12] K.-H. Choi, T. T. Tran, and D.-S. Kim, “A new approach for intelligent control system design using the modified genetic algorithm,” *International Journal Intelligent Systems Technologies and Applications*, vol. 9, no. 3-4, 2010.
[13] V. Nguyen and H. P. Bao, “An efficient solution to the mixed shop scheduling problem using a modified genetic algorithm,” *Journal of Procedia Computer Science*, vol. 95, pp. 475-482, 2016.
[14] M. Gen and R. Cheng, *Genetic Algorithms and Engineering*, 1st ed. New Jersey: John Wiley & Sons, 1997.
[15] Martin-Luther-Universität Halle-Wittenberg. QCSPgen-A benchmark generator for quay crane scheduling problems. [Online]. Available: http://prodlog.wiwi.uni-halle.de/forschung/research_data/qcspgen/

Vi H. Nguyen is currently a lecturer in Sustainable Product Development Global Production Engineering Management (GPEM), Faculty of Engineering, Vietnamese-German University. She got the full-scholarship from Vietnam Education Foundation for 5-Year Combined Masters-PhD Program in the USA. She obtained her out-standing student award and her Ph.D. degree in design and manufacturing from Mechanical Engineering Department, Old Dominion University in 2017. Her research interests are sustainable manufacturing, engineering design optimization, scheduling and planning optimization.



Duc T. Nguyen obtained his B.S. (Northeastern University, 1974), M.S. (U.C. Berkeley, 1976), and Ph.D (University of Iowa, 1982) degrees in civil/structural engineering. He has been with Civil Engineering Faculty at ODU since 1985. His teaching activities (including 3 published textbooks, 1 co-edited book), research works (with 53 journal papers, 81 conference proceeding papers, 53 technical reports, 22 abstracts, and 62 seminars) and 36 funded projects (Totally = over $4,000,000) in large-scale parallel computational mechanics have led to several international/national/regional awards.