

A Variable Neighbourhood Search for Solving the Yard Crane Scheduling Problem in Container

Saber Ben Zina^{1,*}, Abdelkarim Elloumi¹, and Alain Quilliot²

¹ Department of Quantitative Methods, Faculty of Economics and Management, University of Sfax (FSEGS), Tunisia

² Institute of Advanced Computer Models and Applications, University Clermont Auvergne (LIMOS), France
Email: saber.benzina@fsegs.usf.tn (S.B.Z.); abdelkarim.elloumi@fsegs.usf.tn (A.E.); alain.quilliot@uca.fr (A.Q.)

*Corresponding author

Manuscript received April 1, 2025; accepted May 10, 2025; published May 30, 2025.

Abstract—In this context, the objective of the Gantry Crane Scheduling Problem (GCSP) is to develop a schedule that minimizes the completion times of the RTGCs in the container storage areas. The makespan based on a set of container loading or unloading operations. This heuristic is able to find slot schedules with indicated capacity that are efficient for small instances, their performance decreases as the size increases. In this framework, a Variable Neighborhood Search (VNS) was developed for the GCSP which contains a local search architecture to retain efficient solutions. Computational evaluations show that VNS is able to significantly outperform the results obtained by CPLEX.

Keywords—container terminal, scheduling problem, Rubber Tyred Gantry cranes (RTGC), variable neighbourhood search, mixed integer programming

I. INTRODUCTION

The container terminal is generally operated with three types of equipment, quay cranes which ensure the loading and unloading of ships, yard trucks (auto guided vehicles, straddle carriers, etc.) which ensure the transportation of containers from the quay towards the storage areas and vice versa and yard rubber tire gantry cranes Get Ridder Titan Genesis Corp (RTGC) which ensure the storage of containers in the storage areas. Following the considerable increase in container traffic by sea, handling operations in container terminals have increased significantly in recent years. These studies generally focus on specific problems such as scheduling a type of equipment, assigning ships to docks or managing storage spaces with the aim of maximizing container traffic. For terminal operators, the most important objective is to decrease the total resource processing time in the storage area at the terminal, and positioning problem, Fig. 1 is an illustration of the RTG cranes and the storage containers is an illustration of the containers blocks assignment. Allocation of storage space in rail yards is one of the typical decision-making problems faced by operators. There is a common practice according to which storage strategies depend on the movement of the yard crane and the park is divided into several container blocks Guan *et al.* [1] proposed a new formulation of a time-space network scheme with non-crossing constraints for the problem and applies an exact solution approach. Dafnomilis *et al.* [2] adopted terminal operational strategies, a new mixed-integer linear programming model to examine a problem of deploying container terminal equipment to minimize operational and cost costs. Terminal investment linked to specific equipment. Finally, the results obtained an optimal solution for the objective of minimizing storage costs. Studies concerning the

optimization of handling equipment in a container terminal have also been carried out. Soriguera *et al.* [3] analyzed the operations of handling equipment and models their optimization. The queuing theory method is adopted to achieve better efficiency to minimize the total cost of port handling. The work on the Radioisotope Thermoelectric Generator (RTG) Grand Challenges Scholars Program (GCSP) involves identifying the order of tasks for each quay crane to minimize the total compilation time. At the beginning, the GCSP remained very complex, since the authors tried to represent the realities of the economic context. For example, the interference between the RTG crane like constraint (3) is a security condition. Planning and scheduling of operations at container terminals includes allocation of containers in storage areas to import activities, Scheduling (RTGC), to improve storage space. Efficiency in scheduling crane yards is a fundamental element of terminal management. RTG crane operations are closely linked in storage areas and require good coordination so as not to reduce the performance of these operations. Therefore, in this work we examine the problem of scheduling crane RTGs in storage areas. The scheduling decision of GCSP minimizes total makespan of RTG crane. The contribution of this research concerns the two methods mentioned, the flow scheduling model is deepened and modeled in the form of a Mixed Integer Programming (MILP) as well as a VNS algorithm to optimize the approach optimality at the large instance level of a favorable CPU execution time. This study is organized as follows. First, in Section II that follows, we provide a brief review of related research in the current literature. Then, Section III provides a more detailed explanation of the problem. A mixed integer linear programming model is developed In Section IV; we find a Variable Neighborhood Search (VNS) algorithm as a solution method for (GCSP). The model as well as the algorithm VNS were developed with a view to solving this problem. In Section V, we report the experimental results indicating that the applied algorithm VNS performs well. Finally, the conclusions and felt contribution are presented in Section VI.

II. LITERATURE REVIEW

A. Yard Crane Scheduling Problem

With the development of global trade, the scheduling of RTG crane and container storage operations is essential and a main element in the management of port terminals, has been examined by many authors. Some of the most interesting recent research results for yard crane scheduling problems

GCSP are presented in this section. For an in-depth review of container port operations, Zheng *et al.* [4] proposed of an optimization model for the scheduling of crane yards, by an objective function having minimization of the maximum delay of operations on the containers, a Genetic Algorithm (GA) and heuristic based on the programming model in numbers integers Mixed-Integer Linear Program (MILP). Allocation of storage space in rail yards is one of the typical decision-making problems faced by operators. There is a common practice according to which storage strategies depend on the movement of the yard crane and the park is divided into several container blocks. Liang *et al.* [5] developed a new integer programming model, with multi objective to minimize the total completion time of yards cranes, the search method based on the use of metaheuristics based on a hybrid Genetic Algorithm (GAH) and method pareto is important. The result shows the performance of both methodologies. Lee *et al.* [6] proposed a MILP method which is a heuristic simulated annealing algorithm is used to optimize Youth Community Support Program (YCSP), with the objective of minimizing the total costs, including the cost of the yard crane and the cost of delays in the storage area. Numerical tests were also carried out to test the effectiveness of the simulated annealing algorithm. Jin *et al.* [7] considered an approach to transshipment flows in a container terminal. Adopted the mixed integer programming formulation is it understood the objective of minimizing both the total distance traveled by the total distance flows. An algorithm is designed based on lagrangian relaxation to find solutions close to the optimum in calculation times. This corresponds to the hypothesis retained in this document. They proposed a mathematical model that represents the basis of the measurement model in this study. There is also related research in other areas that applies NSGA-II to address the problem. Petering *et al.* [8] studied a problem in the design of seaport container terminals. Using a fully integrated discrete event-based simulation model. The various tests relate to large instances depending on the amount of equipment of a container terminal, which validates the optimality of the design and the dimensions of the storage block. The results indicate that the performance difference between the best block width. The problem of container relocation represents one of the major issues in port terminals, which would generate very significant savings in operational costs, due to the high number of containers handled each year in the global activity.

Zhang *et al.* [9] proposed new algorithms based on machine learning, which combine optimization methods and intelligent learning techniques to optimize the movement of containers to the storage area in loading or unloading level to await the objective of minimizing the total cost of operations. The results of the two different methods of calculating the lower limits and shows the numerical testing performance of the proposed algorithms (IB&B). Using numerical experiments, shows the performance of the decision process at the storage block level. Li *et al.* [10] proposed an integrated decomposition and dynamic programming algorithm to solve instances adapted to reality in very short times for the problem of scheduling a single yard crane Content Security Policy (CSP), is to minimize container movements and the train parking time in the terminal. More studies have focused

on allocation structures based on different sub-blocks. Yu *et al.* [11] proposed a space management strategy, to the extent that containers relating to different services are assigned to a single container sub-block for the optimization of a more adaptable storage space is necessary. A multi objective to minimize all possible handling in the export yard for the loading process is also focused on the total distance in the container terminal. Specifically for the scheduling of yard cranes, several research works have been carried out. For example, Chang *et al.* [12] developed a new dynamic decision strategy using an integer programming model to optimize the scheduling of yard cranes (deployment). which also applied a genetic algorithm to minimize the total delay of tasks in container blocks. On the other hand, Ben *et al.* [13] assumed that container block dimensions were considered as storage space at the port terminal, where the location of containers is always known during import activity, following the direct storage strategy. Therefore, the location capacity in the container block is compatible with the RTGC working equipment. For this reason, the present document proposes a particularly simple adjustable Hansen and Mladenovic [14] Variable Neighborhood Search (VNS) algorithm that allows specific adjustment of the (MILP) parameters. In this work, a VNS algorithm for the RTG crane GCSP scheduling problem, which integrates a memory architecture to maintain the best quality solutions is proposed. In this context, a VNS algorithm is proposed to solve the GCSP. These metaheuristics are integrated into an efficient framework with the aim of enabling the scheduling of container yards and terminal gates in an import activity. In this paper, a Variable Neighborhood Search (VNS) applied to the problem of gantry crane scheduling is proposed, which integrates a memory structure to maintain solutions close to optimality (elite solutions).

III. MATHEMATICAL MODEL

As shown in Fig. 1, crane RTGs have been used to store containers on both the seaward and landward sides by directing container blocks from the storage area to the terminal, which is more common in export and import terminals in the port of Sfax-Tunisia. The scheduling of RTGCs based on container storage in a single container block is affected by yard cranes, as shown in Fig. 1. In our example, the complexity of scheduling yard cranes or gantry cranes based on containers is explained in order to achieve the objective of minimizing the makespan of the optimal sequence. Suppose the example scenario follows a single container block with 3 RTGCs that process 4 slots, with (5, 20, 10, 18) containers assigned. The number of containers per slot is randomly managed between (5, 30), and the assignment or preparation time per yard crane on a single slot is between (10, 40). As shown in Fig. 2, the number of container storage operations is generally planned as 3 groups, each group working 8 hours a day. When a container arrives at the storage areas, the containers to be unloaded are assigned to container blocks at the terminal. The container storage area at a terminal gate is used to temporarily store imported containers while awaiting transport by trucks to their final destination. A large storage area can be divided into several slots (container locations) that are available in a rectangular container block, as shown in Fig. 1. The block can

have 6 rows of containers positioned side by side, with each row stacked 5 tiers high, and a length of 20 containers in rectangular form to accommodate gantry crane equipment or yard cranes.

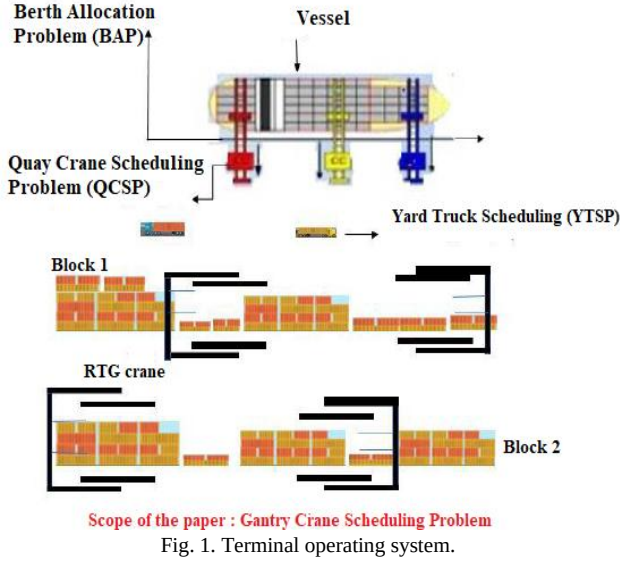


Fig. 1. Terminal operating system.

A. Assumption

In order to clarify the problem, it is necessary to pose the following hypotheses:

- Transport time between slots for the gantry crane is negligible.
- The length of the container block slot is determined by the standard container size (20–40feet).
- Gantry crane work is stopped if no storage location is available for their assigned slot.
- Inter-trucks arrive at the assigned location once the gantry crane has completed loading or unloading.
- There is always an adequate number of inter-trucks available.

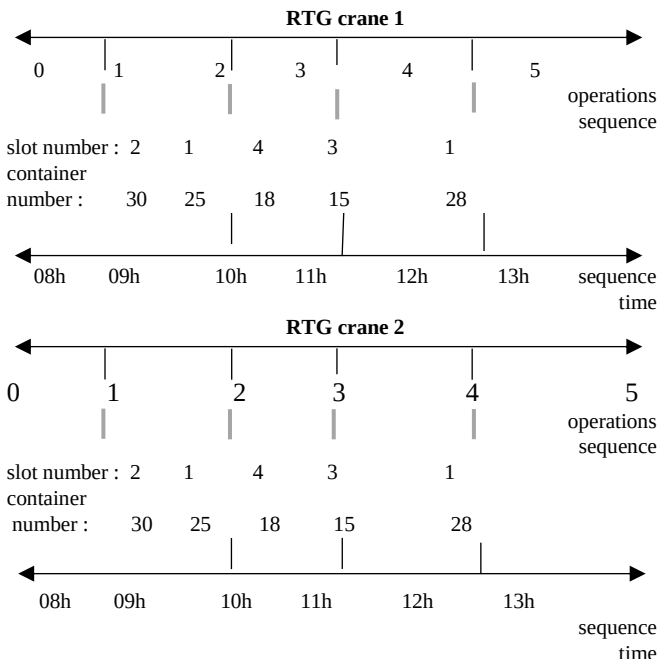


Fig. 2. An Example of a RTG crane schedule.

1) Sets

S : sets of slots (storage location per TEUs)

K : sets of gantry crane

S_i : Sets of stacks in slot i

2) Indices

i, j : index of slots $i, j \in S$

k, g : index of gantry crane $k, g \in K$

s' : index of size stacks $s' \in t_i$

s : indexed by $s \in S_i$

3) Parameters

p_i : The processing time of slot i by a gantry crane

n_s : Number of containers loading or unloading

t_i : a size to the set stack for slot i , indexed ($s'=1$ to t_i)

h : the time taken to assigned containers in slot i

f_i : the storage capacity of slot i per (TEUs)

M : big number integer used to infinity

4) Decision variables

c_i : Integer variable that represents the current completion time of stack s in slot i

$\max c_i$: Makespan

x_{ik} : 1 and only if slot i is assigned to gantry crane k
0 otherwise

y_{ij} : 1 if and only if slot i completes no later than slot j starts
0 otherwise

z_{ijk} : 1 in and only if slot i is assigned to gantry crane k and slot j is assigned to gantry crane g
0 otherwise

5) Objective Function

$$\text{Minimize } \max C_i \quad \forall i \in S \quad (1)$$

6) Subject to

$$\max C_i \geq c_i \quad \forall i \in S \quad (2)$$

$$c_i - p_i \geq 0 \quad \forall i \in S \quad (3)$$

$$\sum_{k=1}^K x_{ik} = 1 \quad \forall i \in S \quad (4)$$

$$p_i \geq h \times n_s, \quad \forall i \in S \quad (5)$$

$$c_i - (c_j - p_i) + y_{ij} M \geq 0 \quad \forall i, j \in S \quad (6)$$

$$c_i - (c_j - p_i) - (1 - y_{ij}) M \leq 0 \quad \forall i, j \in S \quad (7)$$

$$\sum_{s'=1}^{t_i} x_{ik} \sum_{s=1}^{S_i} t_i S_i x_{ik} \leq f_i \quad \forall i \in S, s' \in t_i, s \in S_i, k \in K \quad (8)$$

$$z_{ijk+1} \leq y_{ij} + y_{ji} \quad \forall i, j \in S, k \in K, i \neq j \quad (9)$$

$$z_{ijk} \leq y_{ij} + y_{ji} \quad \forall i, j \in S, k \in K, i \neq j \quad (10)$$

$$x_{ik} \in \{0,1\}, y_{ij} \in \{0,1\}, z_{ijk} \in \{0,1\}, c_i \geq 0; \max C_i \geq 0 \quad \forall i, j \in S, \forall k, g \in K \quad (11)$$

The objective function (1) minimises the maximum of workload of container operations in the storage area, which is

the final completion time from all slots. Constraints (2) and (3) define the value properties the final completion time for variable decision and the completion time at the beginning of the handling operation. Constraint (4) ensures that each slot must be assigned to at least one gantry crane. Constraint (5) ensures that the processing time must be greater than or equal to the workload (workload = number of containers $\times$ time to assign a container in the slot per gantry crane). Constraints (6) and (7) definition of the linear of the decision variable y : constraint (6) shows that $y_{ij} = 1$ if the (laste completion time less than or equal to completion time at the latest – processing time, if only if means that $y_{ij} = 1$ when slot i ends before slot j begins, constraint (7) indicates the contrary. Constraint (8) ensures that each slot capacity i does not exceed the stack tier (height), indicates that the two gantry cranes do not process the same slot at the same time. And finally, constraint (9) and (10) represent the interference constraint, that is, when slot i is assigned to RTG crane k and slot j is assigned to crane g at the same time and under the condition that slot i that the processing time is lower than slot j than the processing time if $i < j$ & $k < g$. Constraint (11) ensure the nature of the used decision variables.

B. Illustration of MILP in Terms of Numbers

For an instance of $S = 10$ slots and $R = 2$ RTG crane, the p_i processing times and S_i the stacks in slots i ($s = 1$ to S_i), of slots n_i number of containers loading or unloading. Numerical resolution with 2 iterations shown in Tables 1 and 2.

Table 1. Iteration 1 of initial RTG crane assignment

Slot i	Prossing time p_i	Assigned to RTG k	Completion time c_i
1	24	1	24
2	28	2	28
3	30	1	54
4	22	2	50
5	36	1	90
6	26	2	76
7	34	1	124
8	42	2	11
9	30	1	154
10	38	2	156

We replace slot $i = 5$ (36) with slot $i = 6$ (26) to see if the lead time decreases then makespan = 156-time units.

TABLE 2. Iteration 2 of adjustment to minimize makespan

Slot i	Prossing time p_i	Assigned to RTG k	Completion time c_i
1	24	1	24
2	28	2	28
3	30	1	54
4	22	2	50
5	36	1	80
6	26	2	86
7	34	1	124
8	42	2	118
9	30	1	154
10	38	2	156

With 2 iterations shown in Table 2 adjustment to reduce makespan = 146-time units, improved on the first iteration (156). Result optimal of makespan found after 2 iterations:

146-time units, then optimal sequence, (RTG crane $1 \rightarrow 3 \rightarrow 6 \rightarrow 7 \rightarrow 9$) and (RTG 2 $2 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 10$).

IV. METAHEURISTIC APPROACH

Variable Neighbor Search (VNS) is a metaheuristic that has been shown to be effective in a large number and wide range of operational domains. This method is based on the systematic variation of numerous neighborhood structures Hansen and Mladenovi'c [14]. In this paper, we construct (VNS) with the aim of solving GCSP problems simultaneously.

A. Variable Neighbourhood Search for Solving the GCSP

We then presented the pseudo code of the VNS proposed for the resolution of the GCSP, it uses two neighborhood structures according to the reintegration movement $L_1(\beta, \alpha)$, which consists of β slot $I = j$ and their α RTGC $k \in K$ and the permutation movement $L_1(\beta)$, which consists of replacing a slot $i \in S$ avec $i = j \in S$ assigned to the gantry crane $k \in K$ by another slot j .

With this in mind, the pooling of neighborhood structures makes it possible to escape from local optima. Unlike other neighborhood-based searches, the VNS algorithm uses a set of neighborhood structures $L_1(k = 1, \dots, k_{max})$. During this research, neighborhood structures $L_2(k = 1, \dots, k_{max})$ are examined during the exploration of the solution space, is constantly remodeled to avoid local optima. The following steps are repeated until the stopping criterion is satisfied; it being specified that it concerns all the neighboring structures. To begin, an initial solution σ is defined and the value of k is set to 1.

Then a neighboring solution σ at the level of the neighborhood hierarchy L_k is determined randomly (Shaking) then an improvement procedure is carried out on the solution β to reach the solution σ (improvement procedure) if $f(\beta'') < f(\beta)$, $\beta = \beta''$ and $k = 1$, otherwise, k is incremented ($k = k + 1$). The steps mentioned above are repeated until $k = k_{max}$. We will find in the algorithm, the coding of the VNS proposed for the GCSP. It is based on two neighborhood structures, such as slot reallocation (S_1) and slot permutation (S_2). The search process is started by creating a starting schedule β by assigning each gantry crane $k \in K$ to the nearest slot $i \in S$. Furthermore, the parameter k is also set to 1.

- 1) Initial Solution: suppose that $slot_{min}$ and $slot_{max}$ represent the slots which have at least one operation, and whose index is minimum and maximum, each of them. The capacity interval of $[slot_{min}, slot_{max}]$ is divided into a set of similar stacks (S_i) and height (t_i).
- 2) Combination Procedure: when two schedules β' and β'' having the same direction of movement are given, it is necessary to use the combination strategy to create a new storage operation schedule in the container block of slot β based on these assignments to the gantry crane in the β' and β'' . Indeed, the variety of two schedules represents the number of slots carried out by gantry cranes independent of each other.
- 3) During each step, ES proposed a combination of schedule β and β' randomly determined so that they were combined and the search started again. In the case of combination, this means that these slots are performed by the same RTG crane and keep the slots

executed by the same RTG crane, while the other slots are assigned randomly.

- 4) Shaking Procedure: In its structure, the VNS offers a stirring step for in this case, it is necessary to randomly choose a solution neighboring the ordering β in the neighborhood structure $L_k(\beta)$.

It should be noted that the shaking function requires reassigning k operations so that they are assigned to neighboring gantry cranes in the slot next. Additionally, operations performed as part of the shaking function are randomly selected from those that were performed less regularly as part of the shaking process. Consequently, there exists, in each case of slot i and gantry crane k operation, a function $f(i, k)$ which corresponds to the number of RTG crane to be performed in the slot. 4. Local Search: In this context, the valorization approach is focused on the consecutive carrying out of two local research projects. Each local search applies a greedy strategy, which allows in-depth observation of the immediate vicinity of the solution. Suppose that β is the current scheduling and that β' is the local optimum reached, after carrying out a local search starting from β using the neighborhood structure $L_1(\beta')$ starting with σ , we perform a new local search using the neighborhood structure $L_2(\beta)$. The local optimum β'' is carried out, as well as the optimized scheduling β'' and permutation by current scheduling β' . 5. Elite Set Updating: The list of elites is updated taking into account effective and diversity criteria. For this reason, only a limited number of local optima are integrated into the set of elites, both those whose values of the objective function in terms of quality are the lowest and those which present a very high diversity in terms of the schedule in the elite set non-common operations affecting the RTGC are defined whose maximum processing time. The RTGC whose total completion time corresponds to the makespan (see Algorithms 1–2).

Algorithm 1: VNS for the solving GCSP

```

 $\beta \leftarrow$  Creating an initial solution
ES  $\leftarrow$  random instance in the course of research.
repeat
     $k \leftarrow 1$  // index of initial neighborhood
    repeat
         $\beta' \leftarrow$  Shake ( $\beta; k$ ) // Random selection
         $\beta'' \leftarrow$  Local Search ( $\beta'$ ) // Neighborhood
        modifications
        Update ES // Random instance in the course of
        research
        if  $f(\beta'') < f(\beta)$  then
             $\beta \leftarrow \beta''$  // Modify the current solution
             $k \leftarrow 1$ 
        else
             $k \leftarrow k + 1$  // next neighborhood
        end if  $k = k_{max}$ 
        ( $\beta', \beta''$ )  $\leftarrow$  Choose your ES schedule
    Until  $\beta \leftarrow$  Combine ( $\beta', \beta''$ ) // randomly assigned RTGC.
Until Stopping Criteria // maximum number of iterations
    
```

Algorithm 2: Local search function with iteration limit

```

local Search ( $\beta, \beta', \alpha$  gantry-Assignments, nbSlots, nbRTGC,
maxIterations) current-Makespan = Makespan(gantry-
    
```

```

Assignments, slots, nbSlots, nbRTGC)
iteration  $\leftarrow 0$ 
while iteration < max- Iterations improved  $\leftarrow 0$ 
for  $i \in$  nbSlots do
    current-RTGC = gantry-Assignments[i]
    for  $k \in$  nbRTGC new-RTGC do
        if new-RTGC  $\neq$  current-RTGC then
            valid-Move  $\leftarrow 1$ 
            for  $j \in$  nbSlots do
                if  $i \neq j$  && gantry-Assignments[j] == new- RTGC &&
                (gantry-Assignments, i, j) then
                    valid-Move = 0
                    break
            if valid-Move then gantry
            Assignments[i]  $\leftarrow$  new-RTGC new
            Makespan = calculate-Makespan (gantry
            assignments, slots, nbSlots, nbRTGC)
            //Calculates the makespan with the new assignment
            if (new-Makespan < current-Makespan) then
                current-Makespan  $\leftarrow$  new-Makespan improved
                 $\leftarrow 1$ 
            else gantry-Assignments[i]  $\leftarrow$  current- RTGC
            end
        end
    end
end if
if  $\neq$  improved then
    break
end if
return iteration
end while
    
```

V. RESULTS AND DISCUSSION

For the computer experiments, the objective is to compare the VNS with CPLEX. To do this we used the instances of MILP proposed by Zina *et al.* [13]. The instances are made up of 20 instances defined in number of slots according to gantry crane with set of stacks combining sets of containers ($S \times Si \times K$) of different sizes. Table 3 shows the number of slots for a size for the stack and gantry crane with the processing time in each operation.

TABLE 3. Parameters for block storage used in test cases

S	p_i	$n_i \times h$	$t_i \times S_i$	f_i (TEUs)
1	24	12	6	30
2	28	14	6	30
3	30	15	5	30
4	22	11	4	30
5	36	18	5	30
6	26	13	6	30
7	34	17	6	30
8	42	22	6	30
9	30	15	5	30
10	38	19	5	30

A. Numerical Experiments

In order to determine the effectiveness of the model and the applied VNS algorithm, experiments were carried out. Firstly, the verification of the effectiveness of the algorithm was carried out using a case study, the effectiveness of the algorithm was verified based on a test of the 20 operations of RTGC assignment to the available slot to store the container. Then, large-scale trials were carried out and cases with different total processing times to reach the objective were examined. The VNS algorithm was programmed in C language. All tests were performed on a computer with Intel

® Pentium ® CPU P6100 @ 2.00 GHz 2. GHz processors, 4 GB RAM, and a 64-bit operating system.

For each example, the average execution times and average variations are shown. The results in Table 4 show that CPLEX and VNS have the same result for small instances. As well as, Both VNS and exact methods aim to obtain unique solutions for the same objective function but (VNS) has lower execution times. For each example instance, however, the (VNS) execution times are quite short. Concerning large instances, the difference between (VNS) and CPLEX is always less than -8% . Indeed, the (VNS) offers a solution with the same objective function and CPU value as CPLEX in 20 instances out of 7. In addition, the execution time applied with the (VNS) is considerably optimized, the execution time employed with the VNS is significantly lower than that of CPLEX for instances set K and S , notice that for these instance sets, CPLEX was greater than the execution time limit for 3 hours.

According to the analysis, CPLEX and VNS obtain equivalent results for instances 1 to 6, demonstrating their ability to find optimal solutions in these cases. However, significant differences appear later on: VNS outperforms CPLEX for instances 14, Increase of 45 units, instances 16, increase of 96 units and 18 increase of 166 units, probably using its heuristic exploration more effectively for these

complex problems, while CPLEX shows some superiority for instances 8, increase of 19, 13 plus 18 and 17 increase of 22 units, reflecting the robustness of its algorithm. These results suggest that the relative performance of the methods depends heavily on specific instance characteristics, with no clear global advantage of one over the other (see Fig. 3).

Table 4. VNS experimental results for cplex and VNS

In. No.	Size Set $K \times S \times t_i$	CPLEX Obj	CPLEX CPU(s)	VNS Obj	VNS CPU(s)	GAP(%)
1	2×3×6	82	0.047	82	0.013	0.00
2	2×4×6	104	0.10	104	0.02	0.00
3	2×6×5	190	0.141	190	0.02	0.00
4	2×7×6	216	0.250	216	0.012	0.00
5	2×8×5	216	0.25	216	0.013	0.00
6	2×7×6	494	8.156	494	0.02	0.00
7	3×9×6	658	17.828	658	0.02	0.00
8	2×10×6	232	49.45	213	0.021	-8.925
9	3×10×6	156	10.53	155	0.0335	-0.645
10	2×11×6	198	120.02	193	0.0352	-2.590
11	3×11×4	176	71.07	179	0.0868	1.675
12	2×12×5	226	392.24	224	0.0337	-0.892
13	3×12×5	188	148.11	170	0.0722	-10.588
14	2×12×5	228	448.47	273	0.0345	16.483
15	3×15×2	206	1230.09	176	0.0329	-17.045
16	2×20×6	187	926.82	283	0.0469	33.922
17	3×20×4	212	363.54	190	0.0212	-11.578
18	3×20×6	131	194.67	297	0.0377	55.892
19	3×20×5	260	569.14	259	0.0613	-0.386
20	2×30×6	1000	13.19	1059	0.04	5.571

$$\text{GAP (\%)} = (\text{VNS objective} - \text{CPLEX objective}) / \text{CPLEX objective} \times 100$$

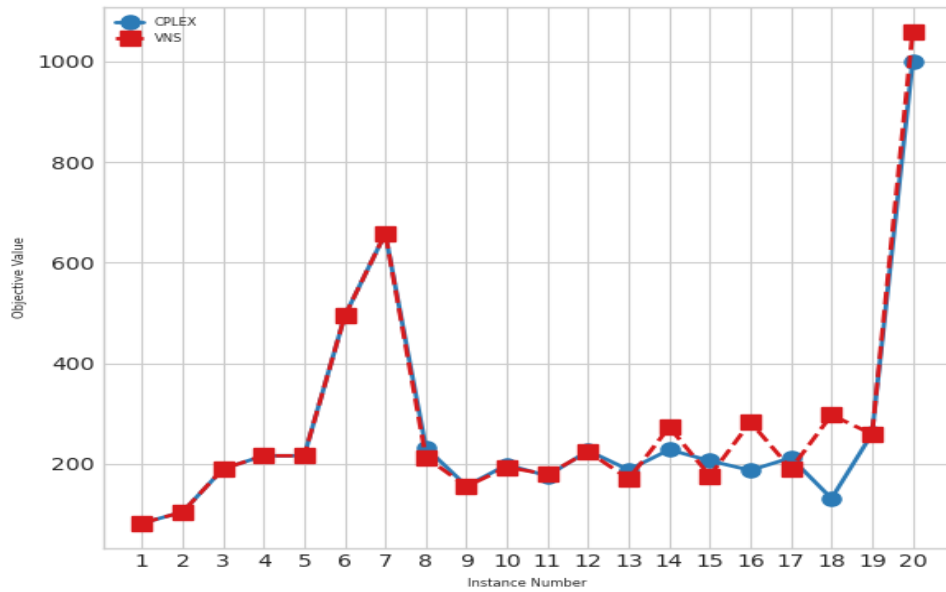


Fig. 3. Comparison of objective function the CPLEX -VNS.

VI. CONCLUSION

This paper deals with Variable Neighbor Search (VNS) for solving the gantry crane scheduling problem in a terminal port for import activity. A list of chosen solutions is created to store data on the studied solution space, with the aim of generating new initial solutions. In this case, most of the solutions obtained by the VNS are comparable to those obtained by the MILP proposed from the publications. We see that this evolution is more marked with the increase in the size of the instances, which shows acceptable search efficiency. In conclusion, the VNS finds solutions approaching CPLEX optimality in two cases. The performance of the proposed VNS is acceptable for solving

large instances. The integration of yard crane scheduling and the scheduling of yard truck is another interesting extension of this research. There are several possibilities for extending the study. On the one hand, more complex metaheuristics can be developed to improve the results obtained. Secondly, the approaches proposed here make it possible to study cases where more than one RTG crane serves a yard block. The comparison of the VNS method with other metaheuristic algorithms, such as simulated annealing (SA) and tabu search (TS), is important. We will take this comparison into account in our future papers in order to better evaluate the relative performance and quality of the solutions of these methods in solving the GCSP.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Saber Ben Zina carried out the problem reduction and developed the programming implementation. Abdelkarim Elloumi contributed to the methodological reduction and provided theoretical insights. Alain Quilliot worked on the problem reduction as well and proposed the metaheuristic approach; all authors had approved the final version.

REFERENCES

- [1] H. Guan, C. Zhang, and Y. Yuan, "The crane scheduling problem: models and solution approaches," *Annals of Operations Research*, vol. 203, pp. 119–139, 2013.
- [2] I. Dafnomilis, M. B. Duinkerken, and M. Junginger, "Optimal equipment deployment for biomass terminal operations," *Transportation Research Part E: Logistics and Transportation Review*, vol. 115, pp. 147–163, 2018.
- [3] F. Soriguera, F. Robuste, and R. Juanola, "Optimization of handling equipment in the container terminal of the port of Barcelona Spain," *Transportation Research Record*, 2006.
- [4] F. Zheng, X. Man, and F. Chu, "Two-yard crane scheduling with dynamic processing time and interference," *IEEE Trans. Intell. Transport Syst.*, vol. 19, pp. 3775–3784, 2018.
- [5] C. J. Liang, M. Chen, and M. Gen, "A multi-objective genetic algorithm for yard crane scheduling problem with multiple work lines," *Journal of Intelligent Manufacturing*, vol. 25, pp. 1013–1024, 2014.
- [6] D. H. Lee, J. G. Jin, and J. H. Chen, "Schedule template design and storage allocation for cyclically visiting feeders in container transshipment hubs," *Transportation Research Record*, pp. 87–95, 2012.
- [7] X. Jin, K. T. Park, and K. H. Kim, "Storage space sharing among container handling companies," *Transportation Research Part E: Logistics and Transportation Review*, vol. 127, pp. 111–131, 2019.
- [8] E. H. M. Petering, "Effect of block width and storage yard layout on marine container terminal performance," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, pp. 591–610, 2009.
- [9] C. Zhang, H. Guan, and Y. Yuan, "Machine learning-driven algorithms for the container relocation problem," *Transportation Research Part B: Methodological*, vol. 139, pp. 102–131, 2020.
- [10] X. Li, A. Otto, and E. Pesch, "Solving the single crane scheduling problem at rail transshipment yards," *Discrete Applied Mathematics*, vol. 264, pp. 134–147, 2019.
- [11] H. Yu, M. Zhang, and J. He, "Choice of loading clusters in container terminals," *Advanced Engineering Informatics*, vol. 46, pp. 101190, 2020.
- [12] D. Chang, Z. Jiang, and W. Yan, "Developing a dynamic rolling-horizon decision strategy for yard crane scheduling," *Advanced Engineering Informatics*, vol. 25, pp. 485–494, 2011.
- [13] S. B. Zina, A. Elloumi, and A. Quilliot, "A rubber tyred gantry crane scheduling in container terminals in port of Sfax-Tunisia," in *Proc. 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2023, pp. 1068–1073.
- [14] P. Hansen, *Variable Neighborhood Search Handbook of Metaheuristics*, Springer New York, 2003.

Copyright © 2025 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).