Yugoslav Journal of Operations Research
34 (2024), Number 3, 439–456
DOI: https://doi.org/10.2298/YJOR231015043C

A VNS-BASED APPROACH FOR SOLVING THE MANHATTAN METRIC STRADDLE CARRIER ROUTING PROBLEM WITH BUFFER AREAS

Ahmet CÜREBAL
Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, Hamburg 20146, Germany
ahmet.cuerebal@uni-hamburg.de

Nina RADOJIČIĆ
Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, Hamburg 20146, Germany
nina.radojicic.matic@uni-hamburg.de

Leonard HEILIG
Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, Hamburg 20146, Germany
leonard.heilig@uni-hamburg.de

Stefan VOß
Escuela de Ingenieria Industrial, Pontificia Universidad Católica de Valparaíso, Chile, e-mail: stefan.voss@pucv.cl and Institute of Information Systems, University of Hamburg, Von-Melle-Park 5, Hamburg 20146, Germany
stefan.voss@uni-hamburg.de

Received: October 2023 / Accepted: April 2024

Abstract: This paper presents a metaheuristic approach for solving an optimization problem that arises at container terminals where straddle carriers (SCs) transport containers between the stacking areas and the seaside. In such container terminals, operational efficiency depends mainly on SC routing. SCs routes should consider the order in which containers are unloaded and loaded at the quay cranes (QCs), taking into account the limited capacity of the buffer area of each QC where containers are temporarily stored after being handled by a QC or an SC. Besides the precedence relations

(i.e., container sequences) and buffer capacities, the solution framework considers safety constraints. Efficient routing of SCs directly contributes to minimizing the idle time of QCs, thereby improving their overall productivity and minimizing the turnaround time of vessels, which is the objective of the problem. Specifically, we present two different variants of the Variable Neighborhood Search (VNS) algorithm. Each variant is initialized in both a greedy and a random manner. These algorithms address the problem by incorporating four LS operators commonly utilized in vehicle routing problems. We perform a comparative analysis of the results of these four approaches against each other and against solutions generated by an exact solver. Our numerical experiments show that the proposed algorithms perform better than the used solver, especially for bigger instances. A comparison with the results from the literature is also given and shows that the proposed VNS-based approach provides competitive results.

Keywords: VNS, port logistics, container terminals, straddle carrier routing problem, vehicle routing.

MSC: 90C59, 90C27, 90B06.

1. INTRODUCTION

Various problems in port logistics can be modeled as discrete optimization problems, which play a crucial role in efficiently managing the complex operations that occur within seaports. These problems involve taking discrete decisions to optimize various aspects of port operations, such as container handling, vessel scheduling, resource allocation, and transportation. Some common discrete optimization problems encountered in port logistics include: Container Allocation, Container Storage, Vessel Berthing and Scheduling, Quay Crane Scheduling, Rail Yard Scheduling, Inventory Management, etc. Different metaheuristics have been used for solving many of these problems in literature, such as POPMUSIC [1, 2], Variable Neighborhood Search (VNS) [3], a Simulated Annealing-based simulation optimization method [4], etc. In general, various metaheuristic approaches have been developed with the goal of solving a wide range of tasks that often cannot be solved by exact methods due to their complexity. For an overview of metaheuristics, the reader is referred to [5], [6], and [7].

One of the important problems that arises in ports is the scheduling of horizontal transport vehicles such as automated guided vehicles (AGVs), straddle carriers (SCs), reach stackers, etc. In this paper, we focus on SC routing, where the aim is to determine optimal routes for SCs as they transport containers between the quay and the storage yard while adhering to loading/unloading sequences. More specifically, we consider an optimization problem proposed in [8] which is referred to as the Manhattan Metric Straddle Carrier Routing Problem with Buffer Areas (MSCRB). The MSCRB emerges in seaport operations, specifically concerning the movement of containers between large stacking areas (i.e., yard areas) and smaller buffer areas with limited space. These buffer areas are strategically positioned within the operational range of QCs. SCs are responsible for transporting the containers and their routes must adhere to prescribed vessel unloading and loading sequences given by the QCs. The primary goal is to minimize vessel turnaround

times which can be operationalized in various directions, including to minimize the distance of the routes for the horizontal transport vehicles. The MSCRB was shown to be NP-hard in the strong sense in [8], where they also provided a mathematical model of the problem. To address the problem while considering high performance requirements of real-world scenarios, we utilize a VNS approach and achieve competitive results. In addition, we use GUROBI to compare the obtained results with an exact solver.

The remainder of the paper is organized as follows. Section 2 provides the problem description. Section 3 covers the related literature review. Section 4 gives a detailed overview of our VNS approach. Computational results are discussed in Section 5. Finally, the paper concludes in Section 6, summarizing findings and contributions.

2. PROBLEM DESCRIPTION

As previously outlined, we propose a solution framework for an optimization problem in container terminals where SCs are the primary means of horizontal transport, facilitating container movements between large stacking areas and QCs introduced in [8]. In the terminal, when addressing container transports from the stacking areas, including internal movements, to QCs at the quayside, and vice versa, a variety of container flows emerges. The categorization of these container flows is influenced by both their initial location and final destination. SCs are tasked with both transporting related containers from their starting point to their designated destination and managing their stacking (i.e., lifting and dropping) within the terminal. Upon closer examination of container flows, containers situated in the yard area, awaiting transport by an SC to a QC and subsequently onto a vessel by a QC, are termed loading containers (outbound containers). On the other hand, containers located in one of the buffer areas that need to be stacked in the stacking area are referred to as unloading containers. For these unloading containers (inbound containers), after being unloaded by a QC from a vessel, SCs are responsible for transporting them from the buffer area to their designated location within the stacking area and ensuring their precise placement. Lastly, restacking containers represent containers that are moved within the stacking areas. That is, some containers may need to be repositioned within the stacking area to allow SCs to either access a loading container or to facilitate the placement of an unloading container. They have to be processed by SCs before the relevant loading or unloading of the containers.

The scope of this study defines a *job* as the responsibility of an SC to retrieve a container from its initial location and transport it to its destination. This concept of jobs is central to the optimization problem concerning the routing of SCs. Given the aforementioned definition of a job, each container requiring movement by an SC within the terminal corresponds to a distinct job. As such, a container can be referred to as a job, more specifically labeled as a loading, unloading, or restacking job. As previously discussed, regarding the flow of these jobs, QCs are tasked with loading containers onto vessels and unloading them from vessels.

Each QC has a designated area within its range, intended explicitly for container exchange, previously referred to as buffer. Containers that have been processed or are awaiting processing by QCs are temporarily stored in buffer areas. Due to limited space on the operational side and in the terminal as a whole, these areas have a container capacity limit. Each QC possesses a pre-defined, sequenced list of containers to process during the loading or unloading operations, resulting from the vessel stowage planning. Considering the dynamic nature of container terminals, the problem assumes that these container sequences are planned for a relatively short timeframe, facilitating rescheduling with updated information. Thus, throughout a planning period, each QC is specialized in either loading or unloading operations, respectively, termed as loading crane and unloading crane assuming that dual-cycling is not possible, which is usually the case for container terminals.

The problem entails determining routes for the SCs, which includes the allocation of jobs to individual SCs and the subsequent sequencing of these assigned jobs. This must be done subject to the capacity restrictions in the buffer areas and considering the precedence relations since QCs process containers in a sequenced manner. This allocation and sequencing is designed to minimize the idle time of QCs, which, in turn, enhances terminal productivity by directly reducing the duration vessels remain docked, namely the vessel turnaround time. Considering the pivotal role of effective coordination between SCs and QCs, it becomes evident that the routing of SCs plays a critical role in influencing the overall terminal productivity.

In addition to foundational assumptions, such as predefined container sequences processed by QCs and the known initial and final destinations of each container, the problem stipulates that all containers are of a uniform size. Furthermore, all SCs and QCs are assumed to be homogeneous. The problem also outlines specific assumptions related to the beginning of the planning period. Regarding the outset conditions, SCs may be positioned throughout the terminal. Additionally, some SCs and QCs will be engaged in the processing of a container, which will prevent them from performing immediate tasks. Expanding on this, buffer areas may already be occupied by containers awaiting further actions. Pertaining to these containers, those situated within a loading crane's buffer area are not the primary focus of the optimization problem. Their allocation to SCs has been pre-established, and they now stand in readiness to be loaded onto vessels by the corresponding cranes. Nevertheless, they remain a consideration within the scope of the problem due to the impact of their presence on the QC occupancy.

All relevant locations in the terminal are mapped using integer points in a Cartesian place. Distances between points are measured using the Manhattan metric. The time required for an SC to process a container is determined by considering both the lift and drop durations, as well as the transit time from its initial location to the designated destination. An SC is considered unloaded when it is en route to the initial location of its next assigned container, either from the final location of the previously processed container or from its initial position, if it is the first container to be processed in the planning period. This unloaded (or

empty) movement is a primary focus of the optimization. Whereas QCs adhere to a fixed sequence for processing containers, the scenario for SCs offers a degree of flexibility from the standpoint of loading operations. SCs can handle containers in a non-fixed order, ensuring that buffer capacities are not surpassed and the container sequence designated for QCs remains intact. In relation to handling unloading containers, SCs are not bound by a specific sequence. Immediately after containers are unloaded by QCs, SCs have the flexibility to transport them from the buffer area to their designated positions within the stacking areas.

3. LITERATURE REVIEW

In regards to the main logistics processes and operations in container terminals, Steenken et al. [9] provided detailed descriptions and classifications, as well as an overview of methods for optimizing these processes. A subsequent publication of this work appeared in [10]. Recently, Kizilay et al. [11] and Weerasinghe et al. [12] have offered in-depth analyses covering both the landside and quayside aspects of container terminal operations, including their integrated handling. In the existing literature, many survey papers address horizontal transport means relevant to our study as part of more general considerations. However, both Stahlbock and Voß [13] and Carlo et al. [14] provide comprehensive discussions specifically on this subject. Notably, Stahlbock and Voß [13] conducted an extensive review on the operations of container terminals, emphasizing horizontal container movements such as AGVs and double rail mounted gantry cranes. They provide a foundational context on the problem domains of the vehicle routing problem (VRP) and its variants. Further, they describe the structure and operational nuances of seaport container terminals. They further highlight main terminal processes such as berth scheduling, QC scheduling and horizontal transport operations. On the other hand, Carlo et al. [14] offer a comprehensive overview of transport operations, emphasizing the material handling equipment utilized. They highlight industry trends and propose a classification system for both transport operations and associated academic literature. Heilig and Voß [15] offer an annotated bibliography that chronologically reviews research on inter-terminal transportation operations. Their work encompasses the approach, scope, and methods employed in these studies. A bibliometric analysis was recently conducted by [16] regarding container terminal operations from an operations research perspective, with the goal of highlighting influential articles. Focusing on pivotal topics like big data analytics, environmental implications, and the uncertainties prevalent in container terminal operations, Raeesi et al. [17] offer a comprehensive analysis, detailing the significant advancements in the domain.

Many studies within our review highlight specific horizontal transport methods and the unique facets of container terminal operations and layouts that may differ from the primary topic of our investigation. However, due to the evident similarities in the container flow procedures within terminals, as portrayed in the existing literature, the scope of our review has been expanded. We do not restrict our analysis to papers that focus solely on particular types of horizontal transport

methods. As far as buffer areas are concerned, horizontal transport characteristics play a crucial role in determining the flow of containers in terminals. In certain configurations, especially due to stacking capabilities of certain horizontal transport vehicles, QCs can utilize buffer areas for container exchanges, whereas in others, QCs directly handle the loading and unloading to and from the horizontal transport means such as for AGVs not having stacking capabilities.

In container terminal operations, the routing, dispatching, and scheduling of transport vehicles are key research topics that have been extensively explored in the literature. Böse et al. [18] focus on container transportation coordinated by QCs and SCs between vessels and yard areas. In their setting, QCs have a buffer area below them, although the capacity restrictions for these buffer areas are not considered. They investigate two dispatching strategies, also referred to as pooling strategies: assigning a fixed number of SCs to QCs serving a single vessel and assigning a fixed number of SCs to all QCs. The primary objective of their work is to minimize the time in port for the vessel by maximizing QCs productivity. They state that this can be achieved through efficient scheduling and dispatching of the SCs. In terms of methodology, the authors use evolutionary algorithms and propose a genetic algorithm. They combine this with changes to terminal processes for optimization. Kim and Kim [19] present a study on SC routing in a port setting where QCs are designated for loading operations exclusively. In this setup, the SC moves between specific yard-slots and the endpoints of yard-bays to facilitate container transfers to trailers. Their main goal is to minimize the overall distance traveled by the SC. Additionally, the authors determine both the number of containers the SC picks up at each yard-bay and the order in which these yard-bays are visited. The routing issue is modeled as an integer programming model. To solve it, the authors introduce an optimization algorithm and employ a two-step solution method. The first step determines the number of containers to be picked up at each yard-bay during a given tour, while the second step focuses on the sequencing of the yard-bay visits by the SC. Yin et al. [20] focus on the scheduling of integrated QCs and SCs in the context of inbound container operations. They consider real-world constraints like traveling time for QCs, non-interference of QCs, safety requirements, and a specific order in which containers must be handled. Additionally, they apply a buffer capacity rule from [21], a more streamlined version of the one introduced in [8]. In their model, only the QCs have the task of placing containers in the buffer area. In contrast, the setting considered in this paper allows SCs to place containers in the buffer as long as they respect the buffer's capacity and the container processing sequence for QCs. Skaf et al. [22] address the scheduling problem for unloading operations involving one QC and multiple yard trucks at a container port. The problem is modeled as a mixed-integer linear programming model with the objective of minimizing the total completion time for all containers. A genetic algorithm and exact enumerative algorithms are used by the authors in order to find solutions to their model.

In summary, the literature on container terminal operations highlights the significance of efficient routing, dispatching, and scheduling of transport vehicles. No-

tably, while different objectives are presented—such as minimizing the makespan for both horizontal transport vehicles and QCs, reducing the overall distance for horizontal transport routes, and minimization of the total completion time for all containers—all aim towards a common primary goal: enhancing the overall productivity of the terminal. A reduced turnaround time for vessels stands out as a widely recognized metric of this efficiency (see, e.g., [23]).

4. VNS FOR MSCRB

In this research, we explore using a very popular metaheuristic proposed by Mladenović and Hansen [24], namely VNS for solving the optimization problem at hand. Starting with the fundamental form of VNS from [24], known as Basic VNS (BVNS), numerous variations of the VNS method have been developed. Those methods based on the VNS methodology have demonstrated their effectiveness in tackling a range of optimization problems [25], including discrete optimization from location problems (e.g. the bus terminal location problem in [26], the maximal covering location problem with customer preferences [27]) to several routing problems that share similarities with the MSCRB problem. For example, an efficient VNS with tabu shaking was used for solving a class of multi-depot VRPs in [28]. Also, a modified VNS was proposed for solving a practical version of the VRP with cross-docking in [29]. A VNS approach was successfully used for solving the multi-compartment VRP with time windows considering carbon emission in [30].

4.1. VNS algorithm

The BVNS iteratively goes though two key phases: an improvement phase, where local search (LS) techniques are employed, and a shaking phase, aimed at breaking free from local minima. These two phases, along with the neighborhood change step, are performed in a cyclic manner until a predetermined stopping condition is met. Typically, the stopping criterion is defined as the maximum allowable CPU time for the BVNS to run.

The pseudocode for BVNS is given in Algorithm 1 as it was presented in [31]. As input for BVNS, we should provide an initial solution x, the maximum number of neighborhoods k_{max} and the maximum allowable CPU time t_{max}. At the beginning of the algorithm, the current time t is reset to 0. The main loop repeats while t does not reach t_{max}. Within this loop, we firstly set k to 1 in order to start the search within the first neighborhood of solution x. More precisely, we firstly use a shake function to move from the current solution x, obtaining a new solution x', that might be even worse than x. This allows exploring the neighborhood of the solution x' using LS as presented in Algorithm 2, finding the best improvement and returning the best-found solution x' withing the used neigborhoods. Within the BVNS, we denote the solution obtained by *BestImprovementLS* as x''. Finally, using the Neighborhood Change procedure presented in Algorithm 3, we accept x'' as x and reset k to 0 if x'' is better than x; otherwise we just increase k by 1 and continue the search within the k-th neighborhood of x. These three

steps (*Shake*, *BestImprovemenetLS* and *NeighborhoodChange*) are repeated until k reaches k_{max}. As a result of the BVNS algorithm, we have the best found solution x.

Algorithm 1 BVNS

Require: x, k_{max}, t_{max}
Ensure: x
  $t \Leftarrow 0$
  while $t < t_{max}$
    $k \Leftarrow 1$
    repeat
      $x' \Leftarrow Shake(x, k)$
      $x'' \Leftarrow BestImprovemenetLS(x')$
      $x, k \Leftarrow NeighborhoodChange(x, x'', k)$
    until $k == k_{max}$
    $t \Leftarrow CPUTime()$

Algorithm 2 BestImprovementLS

Require: x
Ensure: x
  repeat
    $x' \Leftarrow x$
    $x \Leftarrow argmin_{y \in N(x')} f(y)$
  until $f(x) \geq f(x')$

Algorithm 3 NeighborhoodChange

Require: x, x'', k
Ensure: x, k
  if $f(x'') < f(x)$
    $x \Leftarrow x''$
    $k \Leftarrow 1$
  else
    $k \Leftarrow k + 1$

An important part of each iteration of the BVNS algorithm is the potential change of neighborhoods. This is performed as presented in Algorithm 3.

The Variable Neighborhood Descent (VND) method (presented as Algorithm 4) performs a change of neighborhoods in a deterministic way. This algorithm requires an initial solution x and the maximum number of neighborhoods k_{max} as input and returns the best solution x found. These neighborhoods are denoted as $N_k, k = 1, ..., k_{max}$. Initially, k is set to 1, and we repeat two steps until we have an improvement (i.e. until $f(x) \geq f(x')$). The first step is finding the best

solution in the k-th neighborhood of x and call it x'. The second step is where we perform *NeighborhoodChange* as presented in Algorithm 3.

There are three main variants of the VND method: sequential, nested, and mixed. In this study, we use the sequential VND strategy, in which we return to the initial neighborhood after discovering an improvement, referred to as B-VND [32].

The LS step from BVNS (Algorithm 1) can also be replaced by VND (Algorithm 4) from [31], which leads us to the General VNS (GVNS) presented in Algorithm 5. This algorithm requires the maximum number of neighborhoods in the VND phase l_{max} in addition to the same input as required for the BVNS. Note that different neighborhoods $N_1, ..., N_{l_{max}}$ are used in the VND step in addition to the neighborhoods $N_1, ..., N_{k_{max}}$ applied within the Shake step. The GVNS approach has been successful in the literature for various problems (e.g. the traveling salesman problem [33], the swap-body VRP [34], the capacitated dispersion problem [35]). Therefore, we also implemented it for the MSCRB problem to compare its performance with the BVNS.

Algorithm 4 VND

Require: x, k_{max}
Ensure: x
  $k \Leftarrow 1$
  repeat
    $x' \Leftarrow argmin_{y \in N(x)} f(y)$
    $x, k \Leftarrow NeighborhoodChange(x, x', k)$
  until $f(x) \geq f(x')$

Algorithm 5 GVNS

Require: $x, l_{max}, k_{max}, t_{max}$
Ensure: x
  $t \Leftarrow 0$
  while $t < t_{max}$
    $k \Leftarrow 1$
    repeat
      $x' \Leftarrow Shake(x, k)$
      $x'' \Leftarrow VND(x', l_{max})$
      $x, k \Leftarrow NeighborhoodChange(x, x'', k)$
    until $k == k_{max}$
    $t \Leftarrow CPUTime()$

4.2. Neighborhood Structures

The main concept needed to apply VNS to any problem is to construct neighborhood structures suitable for solving the specific problem. We propose adapting

neighborhood structures commonly employed in VRPs to create suitable neighborhood structures for addressing the MSCRB problem, described as follows.

- An intra-route operator (focused on improvements within a single route): 2-opt;
- Inter-route operators (focused on improvements between pairs of routes): 2-relocate (Figure 1), 2-exchange (Figure 2) and 2-opt for two routes (2-opt*) (Figure 3).

In this implementation, all swaps are performed according to the best-improvement strategy with the successive change of neighborhood structures.

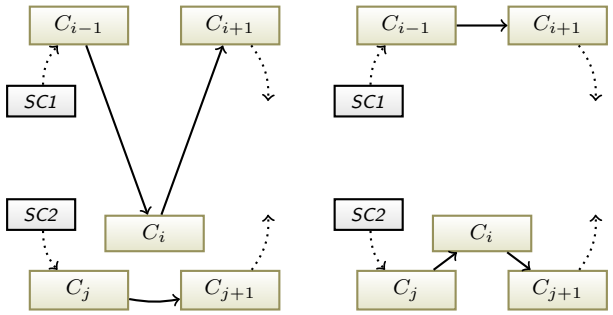

Figure 1: Graphic representation of the 2-relocate LS operator

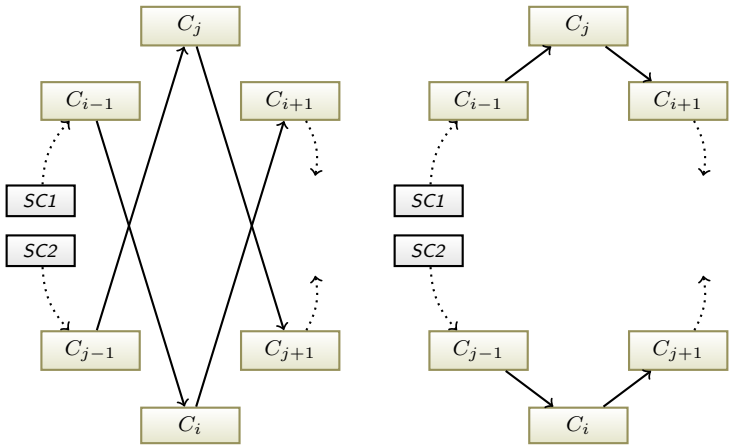

Figure 2: Graphic representation of the 2-exchange LS operator

4.3. Shaking

In addition to the operators used in the LS phase, we propose three different shaking operators:

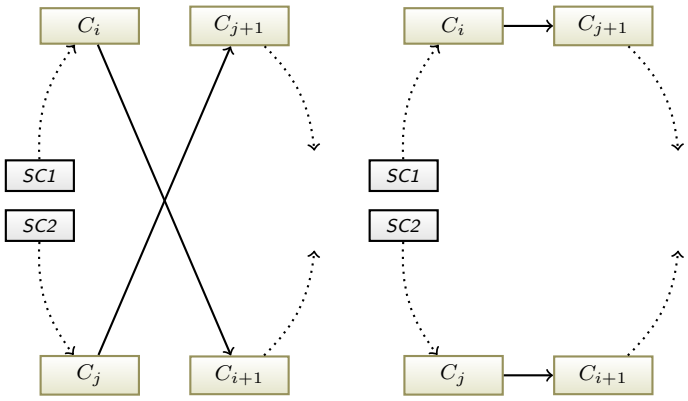

Figure 3: Graphic representation of the 2-opt for two routes LS operator

- The first shaking operator is inspired by the LS operator 2-relocate. We randomly select two SCs. Then we randomly select a container from the first SC route and relocate it to a random location within the route of the second SC.
- The second shaking operator is inspired by the LS operator 2-exchange. We randomly select two SCs. Then we randomly select two containers from each of the selected SC routes and exchange them.
- The third shaking operator is similar to the previous one, except that it works with three randomly selected SCs. We then randomly select one container from each of the selected SC routes and move the first container to the position of the second, which in turn is moved to the position of the third, while the third is moved to the position of the first container.

4.4. Initial solution for MSCRB

In addition to generating a random initial solution, we have developed a greedy procedure for generating an initial solution. In the greedy algorithm, the assignment of jobs to SCs is iterative, taking into account both the distances between the current job and all available SCs and the availability status of the SCs. The order of the jobs is determined by the predefined container orders associated with each crane.

5. COMPUTATIONAL RESULTS

In this section, we present the computational results of the proposed methods implemented in the Python programming language and their comparison with the results from [8]. All our tests for instances with three and four QCs have been carried out on a computer with Intel i5-7300HQ CPU 2.50 GHz with 8 GB RAM memory under the Windows 10 Professional operating system, while a computer

with Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 2.00 GHz with 16 GB RAM memory under Windows 11 Home is used for all instance with five and six QCs.

To evaluate the effectiveness of the proposed methods and compare them with the results of [8], test instances are constructed in the same way as presented in [8]. All relevant locations within a specific problem instance were randomly generated in a Cartesian coordinate system within a two-dimensional plane. The dimensions of this plane were configured to align with real-world data according to [8]. These instances differ in sizes: the number of QCs $(3, 4, 5, 6)$, containers per QC $(10, 14, 18, 22)$, and the ratio of SCs to QCs $(3, 4, 5)$. While each primary instance consistently presents numbers for QCs, containers, and SCs, there are 45 variations of each due to different buffer area capacities (3, 4, 5), and rates of restacking jobs (0.05, 0.1, 0.15) relative to the total number of loading and unloading jobs. Additionally, these instances are affected by five different seed values as they are generated randomly. For the purpose of this study, we narrowed down our scope by selecting a subset of these instances, specifically considering only those with a buffer area capacity of four and a restacking job rate of 0.1. Consequently, each primary instance now comprises five distinct instances, differentiated by their seed values.

Note that in [8], the authors presented the effectiveness of their methods compared with each other and compared with the results obtained with CPLEX. Their computational experiments were executed on a PC with an Intel®CoreTM i7-4770 CPU running at 3.4 GHz and 16 GB RAM running a 64-bit version of Windows 8. A 64-bit version of IBM ILOG CPLEX 12.7 was used. They presented the resulting average quality ratios for each group of instances with an identical number of QCs, containers per crane, and SCs. They also presented the percentage of instances for which CPLEX found a feasible solution within the same time constraint we employed in our experiments, which is 180 seconds.

To compare our results with the results from [8], we first cross-compare the results obtained with the proposed methods with those we obtained with GUROBI 10.0.1. in Table 1 and Table 2 using the following calculations. Given a set of all approaches J, let us denote $\text{obj}(j)$ to represent the objective value obtained by a specific approach $j \in J$. We normalize these values by comparing them to the smallest (i.e., best) objective value across all approaches in J. Let j^* be the approach with the smallest objective value. Thus, $\text{obj}(j^*)$ would represent the smallest objective value across all approaches. To normalize the objective value of each approach j, we calculate: $\text{normalized}(j) = \text{obj}(j)/\text{obj}(j^*)$ for each j in J and for each instance. This way, $\text{normalized}(j)$ would give us a value where the best approach j^* has a normalized value of 1, and all other approaches have values relative to this best value. Finally, we include the average values of instances with the same number of QCs, SCs, and containers per crane in the table.

Table 1 contains the average solution quality ratios for real-world instances, while Table 2 contains the average running times until the best solution is found for each group of instances. In the first three columns of the presented tables, different values of the corresponding values are given: number of cranes (QC), number of containers for each crane to be transported by SCs (CT) and the ratio

of the total number of SCs to QCs (R^*). The next five columns correspond to the best objective values and the time needed to obtain it by five different approaches we have implemented, namely the following methods:

- GVNS initiated in a greedy manner - labeled as GVNS (greedy);
- BVNS initiated in a greedy manner - labeled as BVNS (greedy);
- GVNS initiated in a random manner - labeled as GVNS (random);
- BVNS initiated in a random manner - labeled as BVNS (random);
- GUROBI, used as an exact method.

Moreover, in Table 1, the last column additionally shows the percentage of instances within each group for which GUROBI was able to find a feasible solution within the given time limit.

Let us now consider Table 1, where the best value for the given set of instances (in each row) is indicated by the elements that are highlighted elements. It can be seen that the greedy initial solution instead of the randomly generated one was beneficial for both GVNS and BVNS overall. Moreover, BVNS (greedy) performed the best among all tested methods.

Table 1: Average solution quality ratio for real-world instances

QC	CT	R^*	GVNS(greedy)	GVNS(random)	BVNS(greedy)	BVNS(random)	GUROBI	GUROBI[%]
3	10	3	1.008	1.003	1.007	**1.002**	1.034	100
		4	**1.004**	1.006	1.005	1.006	1.016	100
		5	**1.002**	1.004	1.005	1.005	1.009	100
	14	3	1.011	**1.009**	1.018	1.010	1.092	100
		4	1.014	1.015	**1.007**	**1.007**	1.077	100
		5	1.012	1.014	**1.007**	1.008	1.020	100
	18	3	**1.006**	1.046	1.017	1.007	1.200	20
		4	1.005	1.012	**1.004**	1.006	1.260	100
		5	**1.004**	1.015	1.007	1.013	1.093	80
	22	3	**1.005**	1.010	**1.005**	1.007	-	0
		4	1.008	1.035	**1.006**	1.008	-	0
		5	**1.007**	1.024	1.014	1.022	1.476	80
4	10	3	1.008	1.012	**1.006**	1.007	1.046	100
		4	1.007	1.012	1.012	**1.005**	1.017	100
		5	1.002	1.003	1.002	**1.001**	1.003	100
	14	3	1.014	1.023	1.015	**1.002**	1.121	40
		4	1.016	1.026	**1.006**	1.015	1.127	100
		5	1.010	**1.006**	1.009	**1.006**	1.053	100
	18	3	1.017	1.020	**1.004**	1.024	-	0
		4	1.006	1.026	**1.004**	1.013	2.293	40
		5	1.012	1.015	**1.009**	1.017	1.283	100
	22	3	1.025	1.015	**1.004**	1.006	-	0
		4	1.023	1.034	**1.003**	1.007	-	0
		5	1.009	1.055	**1.005**	1.021	-	0
5	10	3	**1.004**	1.030	1.012	1.018	1.056	100
		4	**1.010**	1.035	1.014	1.024	1.013	100
		5	1.032	1.027	1.024	1.036	**1.000**	100
	14	3	1.025	1.074	**1.008**	1.018	3.791	20
		4	1.009	1.033	**1.006**	1.027	1.117	40
		5	1.010	1.029	**1.004**	1.015	1.041	100
	18	3	1.005	1.060	**1.001**	1.027	-	0
		4	1.014	1.058	**1.000**	1.021	-	0
		5	**1.003**	1.060	1.009	1.041	-	0
	22	3	1.008	1.076	**1.003**	1.025	-	0
		4	1.004	1.056	**1.001**	1.045	-	0
		5	**1.004**	1.077	1.005	1.045	-	0
6	10	3	1.007	1.028	**1.005**	1.024	1.077	100
		4	1.019	1.048	**1.005**	1.047	1.023	100
		5	1.013	1.038	**1.009**	1.021	**1.002**	100
	14	3	1.012	1.050	**1.004**	1.014	-	0
		4	1.014	1.072	**1.004**	1.053	1.298	20
		5	1.006	1.062	**1.001**	1.033	1.206	80
	18	3	1.023	1.098	**1.000**	1.042	-	0
		4	1.011	1.080	**1.001**	1.049	-	0
		5	1.004	1.044	**1.003**	1.056	-	0
	22	3	1.015	1.175	**1.002**	1.018	-	0
		4	**1.001**	1.085	1.005	1.074	-	0
		5	**1.002**	1.062	1.012	1.046	-	0
AVERAGE			1.010	1.040	**1.007**	1.022	1.129	50.417

Table 2: Average times for real-world instances

QC	CT	R^*	GVNS (greedy)	GVNS (random)	BVNS (greedy)	BVNS (random)	GUROBI
3	10	3	73.1	112.1	107.5	106.9	180
		4	40.3	95.8	74.5	97.0	180
		5	55.2	106.7	53.1	57.8	180
	14	3	125.5	88.1	69.0	112.0	180
		4	116.7	92.4	108.1	102.8	180
		5	110.5	134.0	108.6	49.1	180
	18	3	166.5	129.4	108.4	91.6	180
		4	72.4	127.8	58.2	106.1	180
		5	90.2	150.4	109.2	140.0	180
	22	3	134.6	145.5	84.5	144.5	-
		4	91.1	118.4	56.7	117.5	-
		5	122.7	116.7	94.3	102.9	180
4	10	3	153.0	108.6	110.9	77.9	180
		4	85.1	108.5	75.5	101.9	180
		5	83.1	76.9	55.0	85.2	149.84
	14	3	129.4	126.3	66.8	90.5	180
		4	118.9	150.3	123.9	110.2	180
		5	123.4	146.1	134.3	107.6	180
	18	3	149.4	163.0	132.6	83.7	-
		4	102.4	162.6	123.5	128.0	180
		5	140.7	154.4	94.0	84.2	180
	22	3	142.6	168.7	117.2	164.3	-
		4	177.7	183.3	125.6	144.0	-
		5	176.1	180.2	134.6	140.7	-
5	10	3	143.9	165.4	128.8	114.4	180
		4	153.8	133.3	66.9	84.1	180
		5	82.7	142.0	85.4	97.8	151.645
	14	3	181.9	186.5	137.0	156.2	180
		4	181.9	183.1	161.9	178.1	180
		5	164.5	175.8	153.3	167.2	180
	18	3	193.5	191.9	155.7	167.8	-
		4	187.1	191.6	142.3	171.7	-
		5	197.6	194.5	179.3	161.6	-
	22	3	219.3	191.8	125.2	180.8	-
		4	201.1	207.0	160.3	182.5	-
		5	217.3	222.3	151.3	188.1	-
6	10	3	124.5	161.9	133.4	149.5	180
		4	143.7	174.6	135.9	155.9	180
		5	150.3	160.7	123.7	143.2	142.837
	14	3	190.9	189.8	146.5	146.3	-
		4	188.1	185.7	121.2	139.2	180
		5	190.1	190.1	140.2	162.2	180
	18	3	204.8	199.0	175.1	176.8	-
		4	205.7	200.6	162.9	154.1	-
		5	208.4	213.0	140.1	177.2	-
	22	3	227.7	219.9	295.3	307.5	-
		4	231.0	193.9	275.4	270.8	-
		5	200.8	205.0	268.3	304.1	-

Finally, the comparison between the best results of [8] and the results of our BVNS (greedy) implementation is presented in Table 3, using the overall average values of the average solution quality ratio. The instances are grouped by R^* value. We can see that for these test instances generated in the same way as in [8], our BVNS (greedy) outperforms GUROBI more than the best approach from [8] compared with CPLEX. Note that our GUROBI implementation outperforms their CPLEX implementation in terms of the percentage of feasible solutions found. A more detailed comparison was not possible because neither the results on the individual instances nor the instances in [8] were available.

Table 3: Comparison with the best approach from [8]

R^*	3	4	5
Best approach from [8]	1.011	1.013	1.011
CPLEX from [8]	1.046 (29.6)	1.024 (32.9)	1.039 (35.8)
BVNS (greedy)	1.007	1.005	1.008
GUROBI	1.164 (36.25)	1.144 (50)	1.099 (71.25)

6. CONCLUSION

This paper presented solution frameworks for the routing of SCs, taking into account QC operations, buffer area capacities, and the precedence relations set by QCs to handle container movements between the yard area and the QCs.

We presented two different variants of the VNS method and each variant is initialized in both a greedy and a random manner. These algorithms address the problem by incorporating four LS operators and three shaking operators. We performed a comparative analysis of the results of these four approaches against each other and against solutions generated by an exact solver. Our numerical experiments showed that greedy initialization was useful for both VNS approaches. Moreover, the computational results on the tested instances indicated that all our proposed algorithms perform better than the used solver, especially for larger instances. Finally, a comparison with the results from the literature showed that the proposed VNS-based approach is promising for solving the MSCRB problem.

In addressing the objective function from a practical standpoint, the problem focuses on minimizing the time instants when QCs interact with their buffer area, effectively reducing their idle time. By ensuring that the QCs are utilized efficiently, we expedite container loading and unloading processes, leading to vessels spending less time docked at the terminal. This not only enhances the overall productivity of the terminal but also curtails operational costs associated with prolonged vessel docking.

Future work can be performed in various directions, including testing on more instances. Moreover, we want to test other variations of the VNS approach as well as other metaheuristic approaches. In this paper, the problem assumes that all containers, SCs, and QCs are homogeneous. Moving forward, a more realistic setting can be developed by introducing heterogeneous elements. Incorporating SCs with different speeds and container handling capacities can make the model more reflective of real-world scenarios. Furthermore, considering collision avoidance in the yard area would enhance the model's alignment with real-world operational challenges. Incorporating uncertainties, such as the unexpected unavailability of SCs or QCs, can further approximate real-world challenges. As sustainability becomes a central concern, energy consumption could be introduced as an objective to address the environmental impact of terminal operations. There is a potential in integrating machine learning techniques into the optimization processes. Additionally, ensuring the efficacy of these methods on larger datasets would also be a promising avenue for research. In summary, while this study provides a foundational understanding of the problem, there are numerous avenues through which it can be expanded to align more closely with the complex challenges faced in real-world container terminal operations.

Acknowledgments. Ahmet Cürebal was supported by the Study Abroad Postgraduate Education Scholarship (YLSY) awarded by the Republic of Türkiye Ministry of National Education.

Funding. This research was funded by the German Federal Ministry for Digital and Transport (BMDV) with the funding program for Innovative Port Technologies

(IHATEC) within the project TwinSim.

REFERENCES

[1] E. Lalla-Ruiz, and S. Voß, "POPMUSIC as a matheuristic for the berth allocation problem", *Annals of Mathematics and Artificial Intelligence*, vol. 76, 2016, pp. 173-189.

[2] E. Lalla-Ruiz, S. Voß, C. Exposito-Izquierdo, B. Melian-Batista and J.M. Moreno-Vega, "A POPMUSIC-based approach for the berth allocation problem under time-dependent limitations" *Annals of Operations Research*, vol. 253, 2017, pp. 871–897, https://doi.org/10.1007/s10479-015-2055-6.

[3] N. Cheimanoff, F. Frédéric, Kitri, M. Nour and N. Tchernev, "A reduced VNS based approach for the dynamic continuous berth allocation problem in bulk terminals with tidal constraints", *Expert Systems with Applications*, vol. 168, 2021, pp. 114–215.

[4] G. Tasoglu, and G. Yildiz, "Simulated annealing based simulation optimization method for solving integrated berth allocation and quay crane scheduling problems" *Simulation Modelling Practice and Theory*, vol. 97, 2019, pp. 101948.

[5] M. Caserta, S. Voß, ed. Maniezzo, Vittorio, Stützle, Thomas and Voß, Stefan "Metaheuristics: Intelligent Problem Solving", *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, 2010, Springer US, Boston, MA, pp. 1–38.

[6] T. El-Ghazali, Metaheuristics: from design to implementation, John Wiley & Sons, 2009.

[7] C. Blum, and GR. Raidl, Hybrid metaheuristics: powerful tools for optimization, 2016, Springer, 157.

[8] D. Kress, S. Meiswinkel, and E. Pesch, "Straddle carrier routing at seaport container terminals in the presence of short term quay crane buffer areas", *European Journal of Operational Research*, vol. 279, no. 3, 2019, pp. 732–750.

[9] D. Steenken, S. Voß, and R. Stahlbock, "Container terminal operation and operations research-a classification and literature review", *OR Spectrum*, vol. 26, no. 1, 2004, pp. 3–49.

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

[11] D. Kizilay, D. Eliiyi, and D. Türsel, "A comprehensive review of quay crane scheduling, yard operations and integrations thereof in container terminals" *Flexible Services and Manufacturing Journal*, vol. 33, no. 1, 2021, pp. 1–42.

[12] B.A. Weerasinghe, H. Niles Perera, and X. Bai, "Optimizing container terminal operations: a systematic review of operations research applications" *Maritime Economics & Logistics*, vol. 26, 2024, pp. 307–341.

[13] R. Stahlbock, and S. Voß, "Vehicle routing problems and container terminal operations–an update of research", The Vehicle Routing Problem: Latest Advances and new Challenges, 2008, pp. 551–589.

[14] H. J. Carlo, Iris FA Vis, and K.J. Roodbergen, "Transport operations in container terminals: Literature overview, trends, research directions and classification scheme", *European Journal of Operational Research*, vol. 236, no. 1, 2014, pp. 1–13.

[15] L. Heilig, and S. Voß, "Inter-terminal transportation: an annotated bibliography and research agenda", *Flexible Services and Manufacturing Journal*, vol. 29, 2017, pp. 35–63.

[16] B. Dragović, N. Zrnić, E. Tzannatos, N. Kosanić, and A. Dragović, "A bibliometric analysis and assessment of container terminal operations research", *Maritime Business Review*, Emerald Publishing Limited, vol. 8, no. 3, pp. 269–293.

[17] R. Raeesi, N. Sahebjamnia and S. Afshin Mansouri, "The synergistic effect of operational research and big data analytics in greening container terminal operations: A review and future directions", *European Journal of Operational Research*, vol. 310, 2023, pp. 943–973. https://www.sciencedirect.com/science/article/pii/S0377221722009274

[18] J. Böse, T. Reiners, D. Steenken, and S. Voß, "Vehicle dispatching at seaport container terminals using evolutionary algorithms", *Proceedings of the 33rd annual Hawaii International Conference on System Sciences (HICSS)*, 2000, pp. 1–10.

[19] K.Y. Kim, and K.H. Kim, "A routing algorithm for a single straddle carrier to load export containers onto a containership", *International Journal of Production Economics*, vol. 59, no. 1, 1999, pp. 425–433, https://doi.org/10.1016/S0925-5273(98)00108-X

[20] Y.Q. Yin, M. Zhong, X. Wen, and Y.E. Ge, "Scheduling quay cranes and shuttle vehicles simultaneously with limited apron buffer capacity", *Computers & Operations Research*, vol. 151, 2023, pp. 106096.

[21] V.D. Nguyen, and K.H. Kim, "A dispatching method for automated lifting vehicles in automated port container terminals" *Computers & Industrial Engineering*, vol. 56, no. 3, 2009, pp. 1002-1020.

[22] A. Skaf, S. Lamrous, Z. Hammoudan, and M.A. Manier, "Integrated quay crane and yard truck scheduling problem at port of Tripoli-Lebanon", *Computers & Industrial Engineering*, vol. 159, 2021, pp. 107448.

[23] B. Li, Z. Elmi, A. Manske, E. Jacobs, Y.Y. Lau, Chen, Qiong, Dulebenets, and A. Maxim, "Berth allocation and scheduling at marine container terminals: A state-of-the-art review of solution approaches and relevant scheduling attributes", *Journal of Computational Design and Engineering*, vol. 10, no. 4, 2023, pp. 1707–1735.

[24] N. Mladenović, and P. Hansen, "Variable neighborhood search", Computers & Operations Research, vol. 24, no. 11, 1997, pp. 1097–1100.

[25] P. Hansen, N. Mladenović, P. Moreno, and A. Jose, "Variable neighbourhood search: methods and applications", *Annals of Operations Research*, vol. 175, 2010, pp. 367–407.

[26] A. Djenić, N. Radojičić, M. Marić, and M. Mladenović, "Parallel VNS for bus terminal location problem", *Applied Soft Computing*, vol. 42, 2016, pp. 448–458.

[27] L. Mrkela, and Z. Stanimirović, "A Multi-objective variable neighborhood search for the maximal covering location problem with customer preferences", *Cluster Computing*, vol. 25, no. 3, 2022, pp. 1677–1693.

[28] M.E.H. Sadati, B. Çatay, and D. Aksen, "An efficient variable neighborhood search with tabu shaking for a class of multi-depot vehicle routing problems" *Computers & Operations Research*, vol. 133, , 2021, pp. 105–269.

[29] A. Baniamerian, M. Bashiri, and R. Tavakkoli-Moghaddam, "Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking", *Applied Soft Computing*, vol. 75, 2019, pp. 441–460.

[30] J. Chen, B. Dan, and J. Sing, "A variable neighborhood search approach for the multi-compartment vehicle routing problem with time windows considering carbon emission" *Journal of Cleaner Production*, vol. 277, 2020, pp. 123932.

[31] P. Hansen, N. Mladenović, J. Brimberg, J. Pérez, and A. Moreno, "Variable Neighborhood Search", in: Gendreau, Michel and Potvin, Jean-Yves, *Handbook of Metaheuristics*, Springer, Cham, 2019, pp. 57–97.

[32] A. Duarte, N. Mladenovic, J. Sánchez-Oro, and R. Todosijević, "Variable Neighborhood Descent", Handbook of Heuristics, Springer International Publishing, https://uphf.hal.science/hal-03674503, 2018, pp. 341–367.

[33] P. Karakostas, and A. Sifaleras, "A double-adaptive general variable neighborhood search algorithm for the solution of the traveling salesman problem", *Applied Soft Computing*, vol. 121, 2022, 108746.

[34] R. Todosijević, S. Hanafi, D. Urošević, B. Jarboui, and B.Gendron, "A general variable neighborhood search for the swap-body vehicle routing problem", *Computers & Operations Research*, vol. 78, 2017, pp. 468–479.

[35] N. Mladenović, R. Todosijević, D. Urošević, and M. Ratli, "Solving the capacitated dispersion problem with variable neighborhood search approaches: From basic to skewed VNS", *Computers & Operations Research*, vol. 139, 2022, pp. 105622.