Hindawi
Advances in Multimedia
Volume 2022, Article ID 7641084, 12 pages
https://doi.org/10.1155/2022/7641084

Research Article

Scheduling Optimization for Twin ASC in an Automated Container Terminal Based on Graph Theory

Yifeng Xu and Jin Zhu

Institute of Logistics Science and Engineering, Shanghai Maritime University, Shanghai 201306, China

Correspondence should be addressed to Jin Zhu; jinzhu@shmtu.edu.cn

Received 5 February 2022; Revised 22 February 2022; Accepted 8 March 2022; Published 23 March 2022

Academic Editor: Qiangyi Li

Aiming at the twin automated stacking cranes (ASCs) scheduling problem in a single block of an automated container terminal, resolving conflicts between ASCs is an important point that needs to be considered. To solve this problem, a two-stage adaptive genetic algorithm (AGA) based on a graph theory model is proposed. The first stage of the algorithm reduced conflicts as much as possible by adjusting the order of container operations. In the second stage, when conflicts are unavoidable, conflicts are resolved by transforming conflicts into obstacle diagrams. Solving the completion time is to find the shortest distance in the diagram. The effectiveness of the algorithm is verified by applying the proposed algorithm to different scales of container numbers. Compared with the traditional seaside priority strategy, the graph theory model can shorten the completion time to varying degrees. For the strategy of setting the handshake area to reduce conflicts, the results show that the graph theory model makes ASC more efficient.

1. Introduction and Problem Description

In the past few decades, container ports have developed rapidly and the throughput of each container port has continued to grow rapidly, ships as a transportation unit have rapidly increased in size [1]. To increase the competitiveness of the terminal, terminal operators have made a lot of efforts to realize the automation of container terminals. The yard is an important part of terminal operations, efficient storage operations are required to ensure that ships leave the port on time, and the efficiency of retrieval operations is improved to reduce the waiting time of container trucks. Therefore, choosing an appropriate scheduling method to improve the efficiency of the yard is a hot issue to be solved urgently in the container terminal.

In the European layout mentioned by Carlo [2], the container communicates between the sea side and the land side through quay cranes (QCs), automated guided vehicles (AGVs), and automated stacking cranes (ASCs) as shown in Figure 1. In this typical automated container terminal layout, the yard consists of many blocks, ASCs can only perform handover operations with AGVs or trucks at both ends of the block. Transportation requests arise at the seaside I/O point and landside I/O point and must be handled by the ASC, each request has a clear origin and destination. The considered requests either originate from the yard or end up at the yard. To distinguish these requests, we call them storage tasks and retrieval tasks, respectively. The storage task is executed by the seaside ASC and the retrieval task is executed by the landside ASC. The operation process is shown in Figure 2.

Since the dual ASCs studied in this article cannot pass through each other, ASCs will inevitably interfere during the execution of the task, and interference can be effectively reduced by adjusting the task sequence of ASCs. However, once the number of tasks is large, interference cannot be avoided. As shown in Figure 3, the x-axis is set as the time axis and the y-axis is set as the bay position in the Cartesian coordinate system, and the trajectory of the ASCs can be obtained, once the two ASCs conflict in operation, one of the ASCs needs to wait. From the figure, we can see that different ASCs wait to deal with the conflict, which will affect the makespan. Therefore, choosing a suitable waiting strategy to shorten the completion time is the main research content of this article.

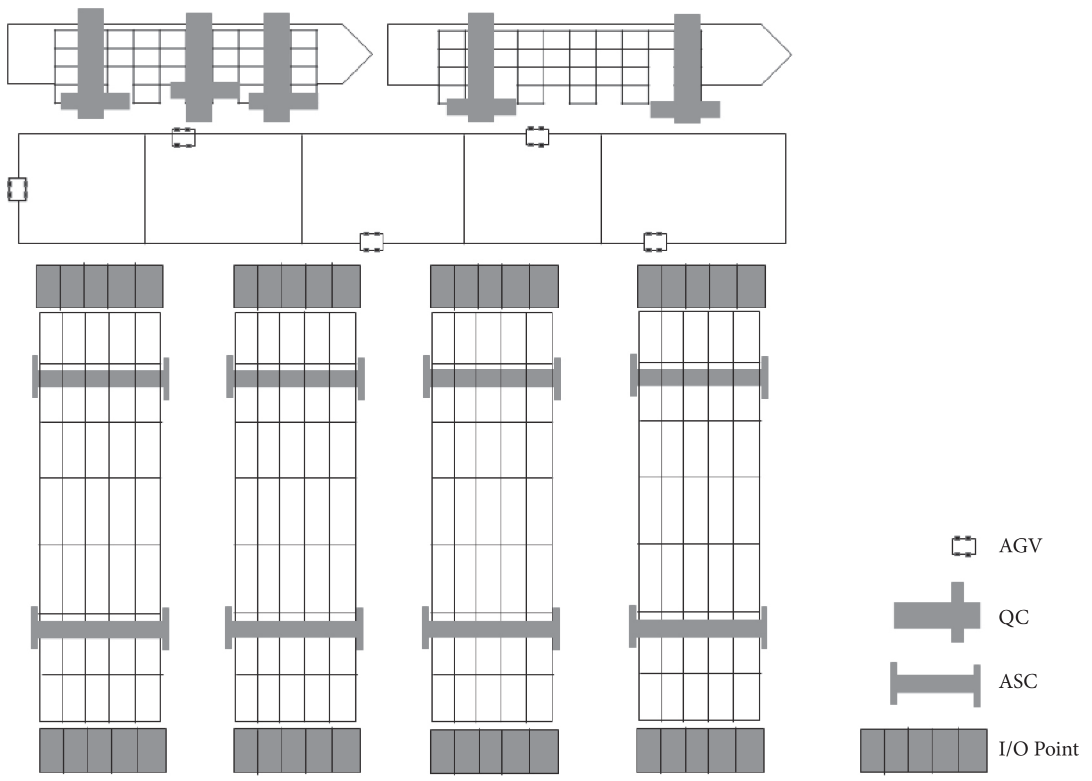


Figure 1: Automated container terminal layout.

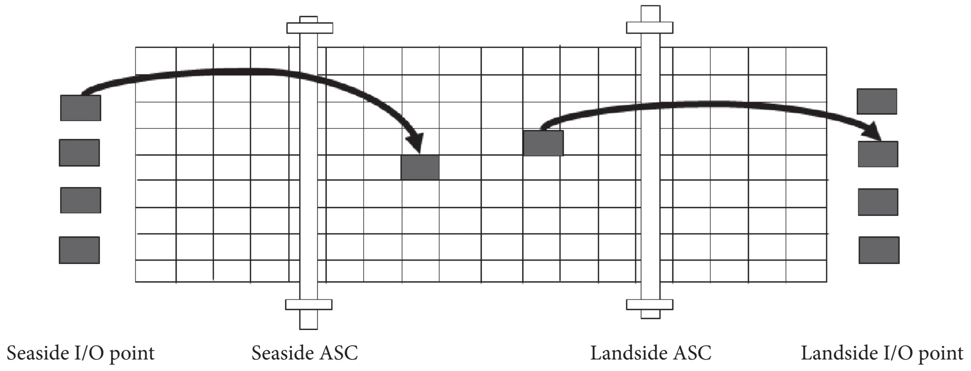


Figure 2: Twin-ASCs workflow.

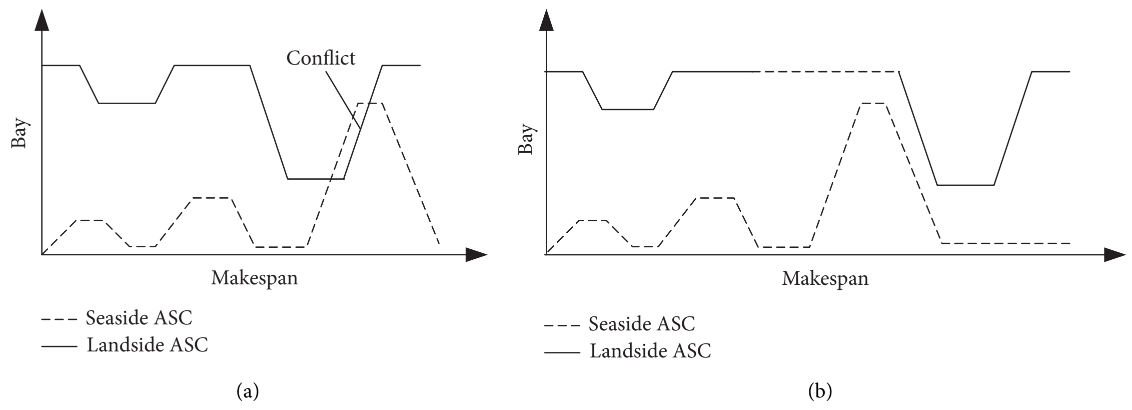
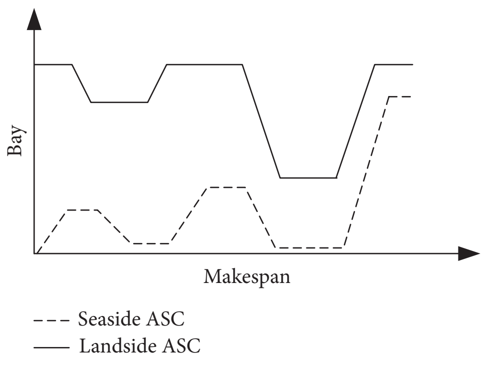


Figure 3: Continued.



(c)

Figure 3: ASC operation conflict and handling diagram.

2. Literature Review

The scheduling of terminal equipment directly affected the operation efficiency of the terminal. To improve the production efficiency of these equipment, a lot of research has been carried out. For the scheduling problem of QC, people mainly consider berth allocation [3–5]. The scheduling problem and path planning problem of AGV have also been widely studied [6–8]. Of course, there are also more and more scholars studying the integrated scheduling of multiple equipment [9, 10].

Compared with the scheduling of ASCs in the automated container terminal, more research in the past is conducted on the equipment scheduling of the traditional container terminal. Narasimhan et al. [11] proved that the path selection problem of a single crane to support the loading operation of a ship is an NP-Hard problem. Ng et al. [12] proposed a branch and bound algorithm, which is suitable for a single yard crane operation at different preparation times. Aiming at the same problem, Guo et al. [13] proposed two new algorithms using different scheduling rules to effectively calculate the yard crane scheduling sequence. Regarding the problem of multiple yard crane scheduling with interference, Ng et al. [14] first studied the interference constraint problem of two yard cranes in a block, proposed an integer programming model and a heuristic algorithm based on dynamic programming. The interference between cranes is eliminated, but the safety distance requirements are not considered, and storage and retrieval tasks are not distinguished. Li et al. [15] developed an effective yard crane scheduling model that includes safe distance, interference, and simultaneous storage and retrieval, then designed a heuristics and rolling-horizon algorithm to quickly solve the model. Chu et al. [16] studied the situation of three yard cranes in two adjacent container blocks and considered the processing time of each container, designed a fast heuristic algorithm and improved the genetic algorithm to solve.

There is no doubt that configuring two ASCs in a block can improve operation efficiency, but solving the interference between two ASCs is a difficult point in scheduling. Saini et al. [17] constructed a two-level random model to analyze the performance of two cranes. It was found that the interference of the cranes will reduce its work efficiency. Gharehgozli et al. [18] studied the twin ASCs scheduling problem and regarded the problem as a multiasymmetric-generalized traveling salesman problem, constructed a mixed integer model with minimized the makespan, and adopted an adaptive neighborhood search heuristic algorithm to quickly obtain the approximate optimal solution. Carlo et al. [19] proposes some priority rules for interference between ASCs. Hu et al. [20] simplified the control and coordination of ASC and designed an accurate algorithm and genetic algorithm to solve the ASC scheduling scheme with the goal of minimizing the minimum time interval between tasks. Briskorn et al. [21] predefined the ASCs evasion rules when interference occurs, by constructed an obstacle graph, the dual ASC scheduling problem is transformed into a job shop problem and a strong polynomial algorithm is developed to solve the problem. Lu et al. [22] established the interference model of the dual ASC system by analyzing the time overlap between tasks and then considering the impact of AGV transportation time, established a scheduling model with the goal of minimizing job waiting time and makespan. To solve this problem, a particle swarm optimization algorithm is designed.

To reduce the conflict between the twin ASCs, setting the middle handshake area is a more commonly used method. Since the ASCs are unable to pass each other, Han et al. [23] set up a handshake area so that two ASCs can be handed over to work. Four interference modes are proposed, and genetic algorithms are designed to solve large-scale problems. Gharehgozli et al. [24] studied the influence of the handshake area on the operating efficiency of dual ASCs in a block with I/O point on both sides of the seaside and landside. It is concluded that the use of the handshake area is helpful for the decoupling of the land and sea, and still has great application and research value. Jaehn et al. [25] studied the benefits of using the handshake area in the dual ASC system, the study showed that it is beneficial to allow ASCs to cooperate in this way, especially when there are no containers in the block, introduced the lower bound method and heuristic method to study whether this positive effect can be sustained. Kress et al. [26] studied the setting of a handshake area in the dual ASC system, proposed a dynamic programming algorithm, which can quickly and effectively solve this problem.

From previous studies, it was found that the response strategy after interference between ASCs has not been fully explored. Due to the high cost of waiting for ships on the seaside, the ASC on the land side passively caters to the work of the ASC on the seaside. Once operational interference occurs, the ASC on the seaside always has a higher priority. This will lead to unbalanced work on both sides, resulting in a longer makespan.

Therefore, this paper focuses on how to quickly obtain a sequence with less interference under a given task sequence and how to balance the operations of the ASC on both sides to get a shorter makespan when the interference cannot be avoided. Combining previous studies on this problem, this paper adopts a two-stage genetic algorithm based on the graph theory, which uses the genetic algorithm to quickly obtain the task sequence with as few interferences as possible, and then obtains the makespan of the task sequence in the graph theory model.

3. Model Construct

3.1. Mathematical Model. In this section, we consider constraints such as the safety distance between ASCs and construct a 0–1 integer programming model with the goal of minimizing the makespan of ASCs. Generally, our model is based on the following assumptions: (1) The I/O point capacity on both sides is sufficient, ASC does not need to wait for loading and unloading at I/O point. (2) Two ASCs travel at a constant speed, ignore acceleration and deceleration. (3) The picking and dropping of ASC take the same time.

3.1.1. Parameters and Decision Variables. 3.1.2. Objective Function and Constraints. Parameter notations have been described in Table 1 above and the twin-ASC scheduling model to minimize the makespan is proposed below.

$$\min W|W = \max\{\mathrm{t_i}\}, \tag{1}$$

$$\sum_{j\in J_k} x^k_{0_k,j} = 1, \quad \forall k \in K, \tag{2}$$

$$\sum_{k\in K}\sum_{i\in J} x^k_{i,j} = 1, \quad \forall j \in J, i \neq j, \tag{3}$$

$$t_{i,j} = x^k_{i,j}\cdot \left|o_j - d_i\right|, k \in K, \tag{4}$$

$$\sum_{j\in J_k} x^k_{i,j} = \sum_{j\in J_k} x^k_{j,i}, \quad \forall i \in J,\ k \in K, \tag{5}$$

$$t_i \geq s_i + 2\tau + t_{i,i}, \quad \forall i \in J, \tag{6}$$

$$s_j \geq t_i + t_{i,j} + M\left(1 - x^k_{i,j}\right), \quad \forall i, j \in J_k, \tag{7}$$

$$\sum_{i\in J_k} x^k_{i,j} + u^k_j = 1, \quad \forall j \in J_k, \tag{8}$$

$$\sum_{i\in J_k} x^k_{i,j} + v^k_i = 1, \quad \forall i \in J_k, \tag{9}$$

$$\sum_{b=1}^{B} by^2_{t,b} - \sum_{b=1}^{B} by^1_{t,b} > \sigma, \tag{10}$$

Equation (1) is the objective function, which means minimizing the makespan of the ASCs, and we do not need to consider the delay time of the job. Constraint (2) ensure ASC1 and ASC2 start to work at the initial position, that is two side of the block. Constraint (3) ensure each job is executed only once by one ASC. Equation (4) calculates the travel time from job i to job j, job i, and job j and are continuous jobs executed by the same ASC. Constraint (5) ensures ASC execute the next job after completing the current job thus guarantee the continuity of jobs. Constraint (6) indicates the relationship between the same task start and end times. Constraint (7) shows the relationship between the start time and end time of two consecutive jobs. Equations (8) and (9) ensure that each ASC can only operate one job at a time. Constraint (10) ensures a safe distance between the two ASCs when working.

3.2. Graph Theoretic Model. Briskorn [21] designed a graph theory model for the ASC scheduling problem under the given task sequence. The model is based on the following two assumptions：If there are multiple cranes, we need to decide which crane completes which task and for given the work distribution of the cranes, we need to determine the work order of each crane.

Among the problem discussed in our paper, the task on the seaside is handed over to ASC1, and the task on the landside is handed over to ASC2, so satisfy the first assumption. Moreover, for the second assumption, the task sequence of each ASC is continuously updated by the genetic algorithm. For the resulting task sequence, at some point in time, a conflict is inevitable. In the event of a conflict, we use the graph theory model to determine which ASC has priority.

When the task sequence is determined, the ASC executes the tasks in sequence, and the movement of twin-ASC needs to fulfil in each time period:

(1) The position of ASC1 is bay r_1 and the position of ASC1 is bay r_2 with $\mathrm{r_2} \geq \mathrm{r_1}$;
(2) The position of ASC1 is bay r_1 and ASC2 is from bay r_2 to bay r_2' with $\min(\mathrm{r_2}, \mathrm{r_2'}) \geq \mathrm{r_1} + 1$;
(3) The position of ASC2 is bay r_2 and ASC1 is from bay r_1 to bay r_1' with $\max(\mathrm{r_2}, \mathrm{r_2'}) \leq \mathrm{r_2} - 1$;
(4) ASC1 is from bay $\mathrm{r_1}$ to bay r_1' and ASC2 is from bay r_2 to bay r_2' with $\mathrm{r_2} \geq \mathrm{r_1} + 1$ and $\mathrm{r_2'} \geq \mathrm{r_1'} + 1$.

Through the proposed graph theoretic theory model, we can convert the twin-ASC scheduling problem under given task sequences into a Job-shop problem, and we develop an obstacle graph to represent the interference between twin ASCs. The problem of seeking the makespan of ASCs

Table 1: Parameter notations.

Parameters	
K	Set of ASCs with $k \in K = \{1, 2\}$, and ASC1 represents the seaside ASC, ASC2 represents the landside ASC
J	set of all jobs with $i \in J = \{1, 2 \ldots, m\}$, and m represents the total number of jobs
J_k	Job set of ASC k
o_j	The origin position of job $j \in J$
d_i	The destination position of job $i \in J$
B	set of all bays in a single block with $o_i, d_i, b \in B = \{0,1, \ldots, 41\}$
$t_{i,i}$	The travel time from the origin position of job i to the destination position of job i, and $t_{i,i} = \lvert o_i - d_i \rvert$
$t_{i,j}$	The travel time from the destination position of job i to the origin position of job j
τ	The picking and dropping time of ASC
σ	The safety distance between two ASCs
0_k	Represent a virtual job in the initial position of ASC,
M	A very large number
Decision variables:	
W	Makespan of the ASCs
$x_{i,j}^k \in \{0, 1\}$	When jobs i, j are executed by the same ASC and job j is executed after job i, $x_{i,j}^k = 1$, where $i, j \in J, i \neq j$, and $k \in K$, otherwise $x_{i,j}^k = 0$
$u_j \in \{0, 1\}$	When ASC starts to perform job j, $u_j = 1$, otherwise $u_j = 0$
$v_i \in \{0, 1\}$	When ASC finish executing job i, $v_i = 1$, otherwise $v_i = 0$
$y_{tb}^k \in \{0, 1\}$	When ASC k is in bay b at the time t, $y_{tb}^k = 1$, otherwise $y_{tb}^k = 0$

translates into finding the shortest path from the lower left corner to the upper right corner in the obstacle graph.

For part of a given task sequence as shown in Figure 4, it shows the position of two ASCs over time. ASC1 starts working from bay 0, and move to bay 5 execute unloading operation, after that back to bay 0 in the time 12. ASC2 starts working from bay 9, and move to bay 3 execute loading operation, after that move to bay 7 in the time 12.

Obviously, there is interference in this operation which will inevitably cause the delay. What we have to do is to minimize the delay time in order to get the minimized makespan. This problem has been solved by a strong polynomial algorithm. In order to understand this problem more deeply, we can refer to the paper of Briskorn and Angeloudis [21].

For the example in Figure 4, we can get the obstacle diagram shown in Figure 5. The horizontal axis and the vertical axis represent the running time of ASC1 and ASC2, respectively, and each unit scale is a unit of time. The obstacle is composed of nine parts, and part C indicates the conflicts of two target bays. Part L represents ASC1 is operating at bay 3 and ASC2 moves closer to the bay 3, the length of the other side of L is the distance between the two ASCs operating bays plus one. Similarly, for ASC1 operating and ASC2 leave the bay 3, we can get the part R. Similarly, for switching ASC1 and 2, we can get the part T and B. By analyzing the conflict caused by the simultaneous movement of ASC1 and ASC2, we can easily get the part TL, TR, BL, and BR.

For the solution of the graph theory model, it is actually the problem of finding the shortest path in the graph and solving it using the strong polynomial algorithm in literature studies [21]. A feasible path can only be composed of horizontal, vertical, and diagonal segments, and cannot pass through obstacles. For a feasible path, we need to calculate

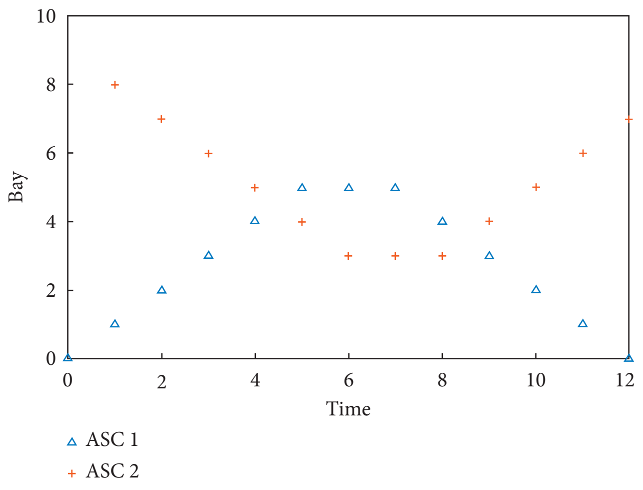


Figure 4: Movement graph for nondelay schedule of twin cranes.

the projection length of the diagonal on the horizontal axis, the length of the vertical and horizontal segments, then the makespan corresponding to this feasible path is equal to the sum of these three. As shown in Figure 5, there are two feasible paths, one pass through points (0, 0), (3, 3), (7, 3), and (12, 8), the other passes through points (0, 0), (2, 2), (2, 8), (6,12), and (12, 12). The length of the first path is 16 and the length of the second path is 18. In each path, the diagonal line represents that there is no interference in the operation of ASC1 and ASC2.

In summary, we get the task sequence through genetic algorithm, some task sequences with interference, we adopt the seaside ASC priority strategy and the equal priority strategy to resolve the conflict. Using the graph theory model, the same priority strategy is considered. Then, the makespan is calculated.

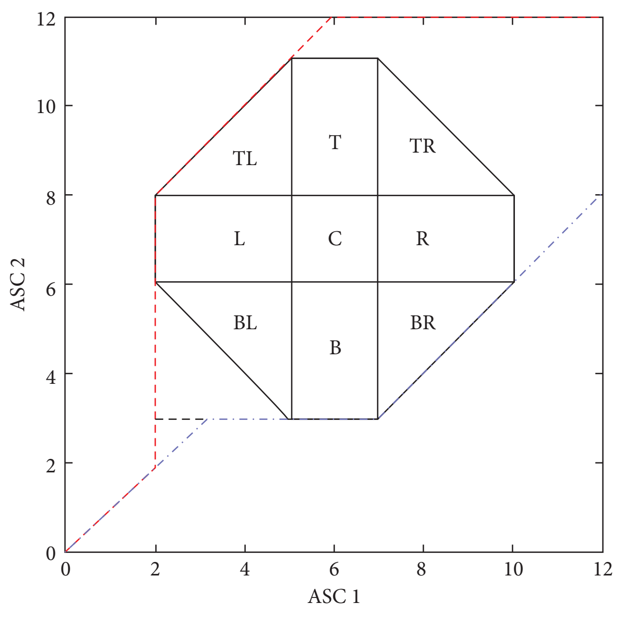


Figure 5: Twin ASCs obstacle diagram.

4. Modified Adaptive Genetic Algorithm (MAGA)

The twin-ASC scheduling problem is an NP-hard problem, so it is difficult to solve directly. Genetic algorithm has a good global search ability. It uses iteration to find the global optimum by imitating the mechanism of natural selection and heredity and uses it inherent parallelism, convenient distributed computing to speed up the solution speed, and has good adaptability to job scheduling problems. In the iterative process of the genetic algorithm, the probability of chromosome crossover and mutation has a great influence on the result of problem solving. If the two are too large, it is easy to destroy the chromosomal individuals with high fitness in the early stage of the iteration; if the two are too small, it is difficult to generate new individuals in the later stage of the iteration, which makes the algorithm fall into a local optimal solution. Therefore, this paper uses an improved adaptive genetic algorithm to solve the problem and combines the graph theory model. The probability of crossover and mutation is obtained by the following two formulas:

$$P_C = \begin{cases} P_{C\max} - \dfrac{\left(P_{C\max} - P_{C\min}\right)\left(f' - f_{\text{avg}}\right)}{f_{\max} - f_{\text{avg}}}, f' \geq f_{\text{avg}} \\ \\ P'_{C\max}, f' < f_{\text{avg}} \end{cases},$$

$$P_m = \begin{cases} P_{m\max} - \dfrac{\left(P_{m\max} - P_{m\min}\right)\left(f - f_{\text{avg}}\right)}{f_{\max} - f_{\text{avg}}}, f \geq f_{\text{avg}} \\ \\ P'_{m\max}, f < f_{\text{avg}} \end{cases}, \tag{11}$$

where $f_{\max}$ represents the maximum fitness value of the population individual, f_{avg} represents the average fitness value of the population, f' represents the larger fitness value of the two individuals performing the crossover operation, f represents the individual fitness value for mutation operation, and $P_{C\max}$ and $P_{C\min}$ represent the maximum and minimum crossover probability. $P_{m\max}$ and $P_{mm\min}$ represent the maximum and minimum mutation probability.

4.1. Chromosome Encoding and Decoding. This paper adopts a double-layer real number coding method, where the upper layer is the number of container tasks, the jobs are executed in order of number, and the lower layer is the ASC number. For example, the task group shown in Table 2, where ASC1 performs the job (1, 4, 5, 7), and ASC2 performs the job (2, 3, 6, 8). In the decoding process, the tasks on the corresponding chromosome positions are first assigned to the ASC to which they belong and then sequentially decoded according to the task sequence on the gene segment to obtain the container task sequence corresponding to the ASC. As shown in the chromosome in Figure 6, the task sequence of decoding ASC1 is (1, 4, 5, 7), the task sequence of decoding ASC2 is (2, 3, 6, 8).

4.2. Fitness Evaluation. The fitness function is taken as $f(u) = 1/C\max$, where $C\max$ is the minimum value of the current ASC makespan. Considering the priority of the seaside ASC, first calculate the operation time of the seaside ASC to complete the job sequence corresponding to the current chromosome, determine its operation track, and then calculate the time required for the landside ASC to complete its corresponding job sequence under the constraint of maintaining a safe distance. Take the larger value of the time required for the two ASCs to complete their respective jobs as the maximum completion time corresponding to the current chromosome. Considering that two ASCs have the same priority, referring to the graph theory model proposed in Section 3.2, the maximum completion time corresponding to the current chromosome is obtained through the graph theory model through the job sequence corresponding to the current chromosome.

4.3. Selection. We use the classic roulette method to select individuals. Through the different fitness of each individual, the probability of different individuals being selected is calculated. Individuals with high fitness are more likely to be selected, which evolves from generation to generation.

4.4. Crossover. In this paper, the PMX crossover method is used to randomly select two genes at the same position on the two parent chromosomes, and use the middle part as the crossover part, as shown in Figure 7. After the crossover, there may be missing or duplicate task numbers in the offspring chromosomes obtained from the crossover. Therefore, the offspring chromosomes must be repaired. Through the mapping relationship, the conflicting genes are

Table 2: Task group example.

Task sequence	1	2	3	4	5	6	7	8
Origin	0	17	27	0	0	20	0	31
Destination	10	40	40	26	35	40	28	40

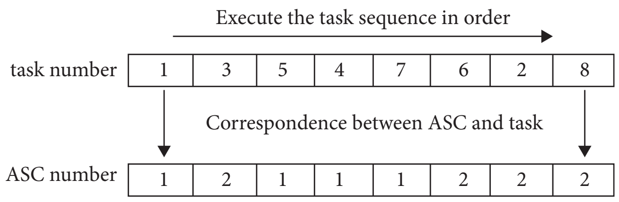

Execute the task sequence in order
task number
1
3
5
4
7
6
2
8
Correspondence between ASC and task
ASC number
1
2
1
1
1
2
2
2

Figure 6: Chromosome encoding and decoding diagram.

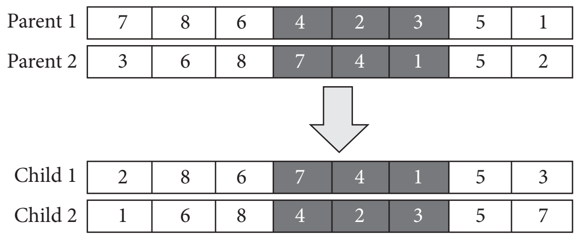


Figure 7: Chromosome swap operation.

repaired one by one, and the repaired offspring chromosomes are obtained.

4.5. Mutation. If only one gene on the chromosome is selected during the mutation operation, it needs to be repaired after the mutation, and the number of the gene does not change before and after the repair, so a certain gene of the mutated chromosome is meaningless. Therefore, two genes on a chromosome are randomly selected, and whether they are mutated according to the mutation factor. Figure 8 demonstrates the method of mutation, by exchanging the corresponding genes to obtain the offspring chromosomes.

5. Numerical Experiment

5.1. Initial Settings. A certain block in a new ACT contains 41 bays. The seaside I/O point is located at 0 bay and the landside I/O point is located at 41 bay. Thus, we set $l = 41$. According to Javanshir [27], The ASCs move at a uniform speed $v = 8$ s/bay and we set the time of ASCs to move one bay need a time unit, picking up or dropping a container takes 240s which is 30 time units. The origin position of the storage task is 0 bay and the destination position generate uniformly by randomly selecting an integer from (1, 40). The destination position of the retrieval task is 41 bay and the origin position generate uniformly by randomly selecting an integer from (1,40). The twin ASCs directly need to maintain a safe distance of one bay.

To ensure the performance of the algorithm and balance the solution speed and the quality of the solution, the maximum number of iterations of the algorithm is finally set to 300 generations, the population size is 200, the maximum mutation probability is 0.1, the minimum mutation probability is 0.01, and the maximum crossover probability is 0.9. The minimum crossover probability is 0.6. Each set of experiments was run ten times and the results were averaged. The described solution method was implemented in *Python* 3.8 under Windows 10 and was run on an Intel Core i5-10400, 2.9 GHz PC with 16 GB RAM.

5.2. Algorithm Performance. Figure 9 shows the convergence of the three algorithms under the same set of tasks at different scales, all of which can converge quickly, and the final results are slightly different. The upper figure shows the average fitness value under different iteration times, the lower figure shows the minimum fitness value under different iteration times. Figures 10–12 are the ASC trajectory diagram corresponding to the solutions obtained by the three methods for the same set of tasks, the x-axis is time, and the y-axis is the bay position of the ASC. It can be found from the figure that the proposed algorithm can maintain the work balance between the two ASCs, thereby shortening the makespan. For setting up the handshake area, some tasks that cross the handshake area will be divided into two, which will increase the loading and unloading time. This reduces interference, but it may increase completion time. Next, we will discuss this in more depth through experiments on different scales.

5.3. Simulation Comparison. In the following sections, (A1), (A2), (A3), and (M) given below represent the adaptive genetic algorithm solution under seaside priority strategy, the adaptive genetic algorithm solution based on graph theory model, the adaptive genetic algorithm solution for setting the middle handshake area, and the equation solution. W is the makespan obtained by the respective algorithm, ASC1 and ASC2 represent the time spent by the seaside and landside ASC to complete their respective tasks.

5.3.1. Comparison with CPLEX Optimizer. To verify the effectiveness of the solution method of adaptive genetic algorithm based on graph theory model designed in this paper, the algorithm program is developed with Python, the mixed integer programming model is solved with CPLEX, and the solution results are compared and analyzed. The first set of experiments is carried out in Table 3, we calculated the results for the instances with $n = 4,8,12,16,20$.

The gap1 is defined by

$$Gap1 = \frac{W_{A2} - W_M}{W_M} \times 100\%. \tag{12}$$

It can be found from Table 3 that when the number of tasks is less than 20, (A2) can obtain a solution in a short time, and the error between the obtained function value and the optimal solution obtained by CPLEX is not greater than 2.61%, proved the effectiveness of the algorithm. After the number of tasks exceeds 20, CPLEX cannot find the optimal solution in a limited time, and (A2) shows its fast solution performance for larger-scale tasks.

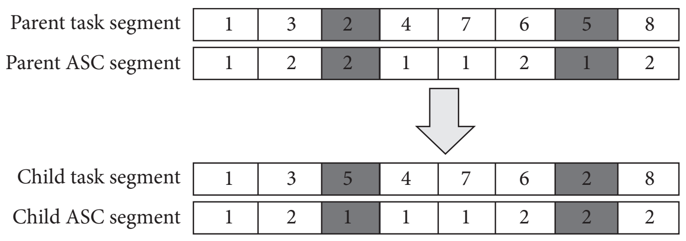


FIGURE 8: Chromosome mutation operation.

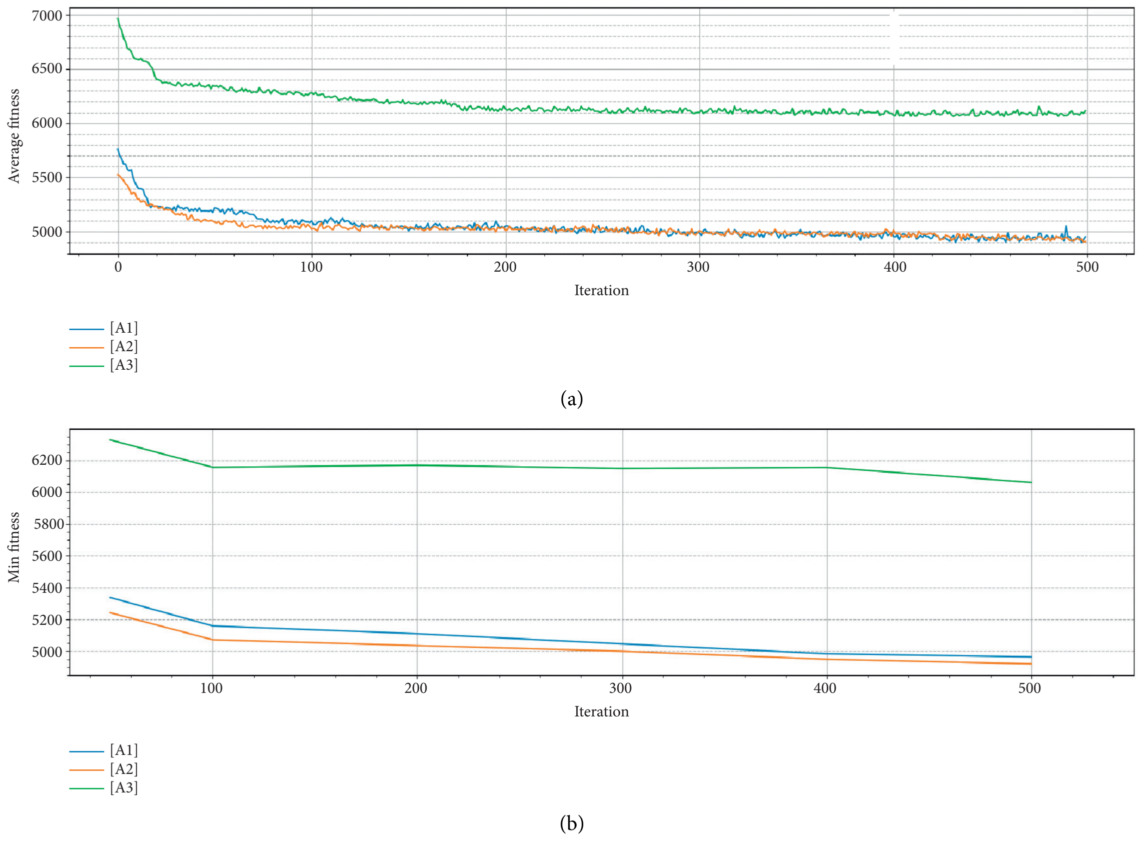


FIGURE 9: Algorithm convergence graph.

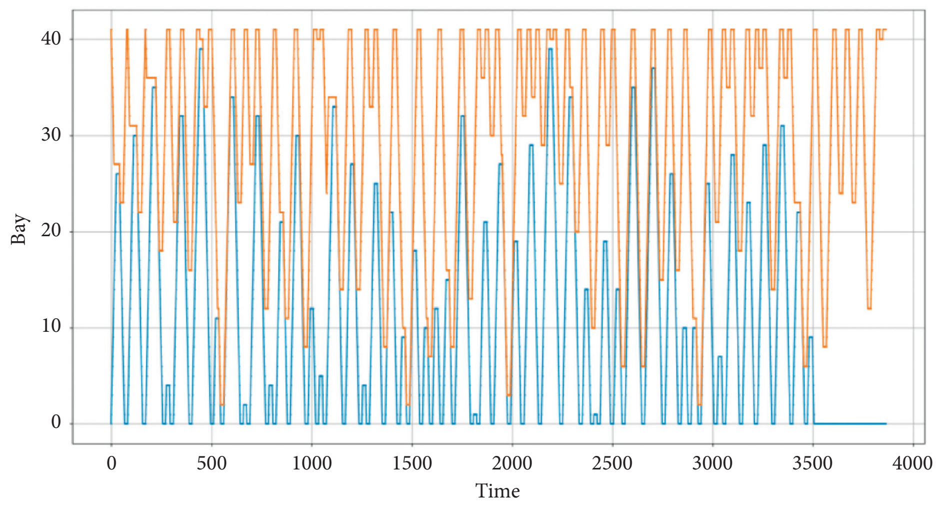


FIGURE 10: Tracks of the twin ASCs under seaside priority strategy.

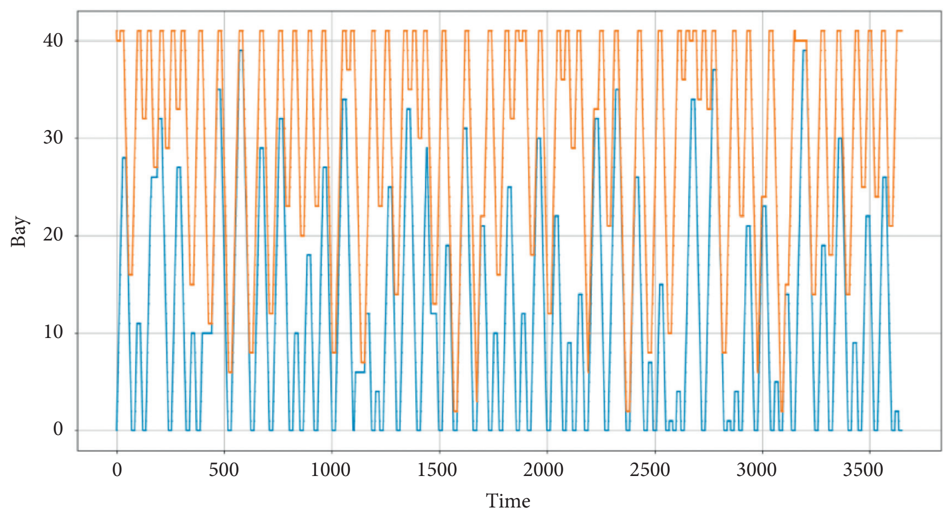


Figure 11: Tracks of the twin ASCs obtained using the algorithm based on graph theory.

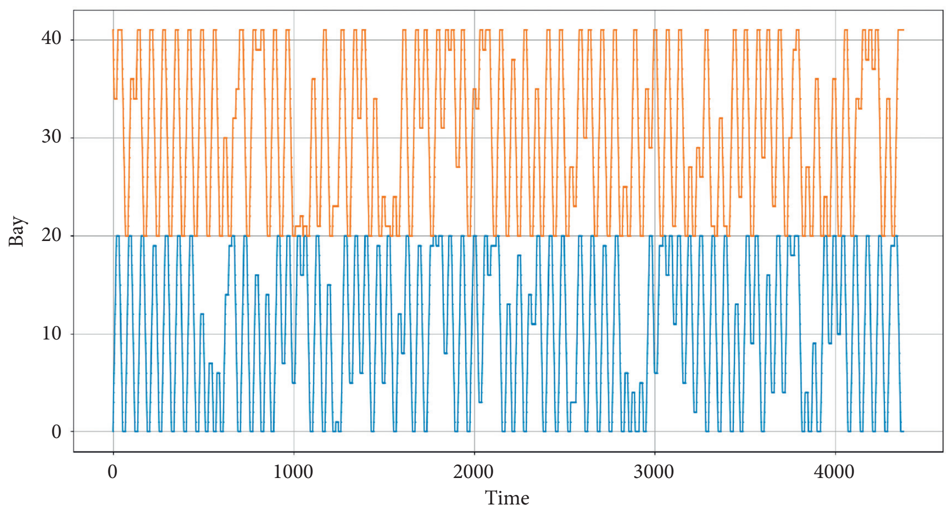


Figure 12: Tracks of the twin ASCs of setting the handshake area.

Table 3: Comparisons among (M) and (A2).

Number of tasks	(M)		(A2)		Gap1
	W_M	Time/s	W_{A2}	Time/s	
4	200	0.67	200	4.18	0.00%
8	377	21.67	384	7.56	1.86%
12	639	259.38	650	13.51	1.75%
16	806	675.69	827	17.36	2.61%
20	1098	1247.56	1125	25.49	2.46%

5.3.2. Comparison with Seaside Priority Strategy. In a further series of tests, we compare the results of our solution method with heuristic proposed by Javanshir [27] for a similar problem setting and expanded the number of tasks. In addition to calculating the makespan, we also obtained the respective completion times of two ASCs.

The Gap2 is defined by

$$Gap2 = \frac{W_{A1} - W_{A2}}{W_{A2}} \times 100\%. \tag{13}$$

Table 4 shows the completion time of two ASCs for the two algorithms under different mission scales. It can be

Table 4: Comparisons among (A1) and (A2).

Number of tasks	(A1) ASC1	(A1) ASC2	(A1) W_{A1}	(A2) ASC1	(A2) ASC2	(A2) W_{A2}	Gap2
30	1472	1615	1615	1553	1608	1608	0.44%
50	2437	2770	2770	2601	2606	2606	6.29%
80	3842	4291	4291	4187	4162	4187	2.48%
100	4845	5106	5106	5032	5071	5071	0.69%
150	7567	8133	8133	8081	8043	8043	1.11%
200	10060	11717	11717	10931	10956	10956	6.95%
300	14785	16731	16731	16172	16217	16217	3.17%

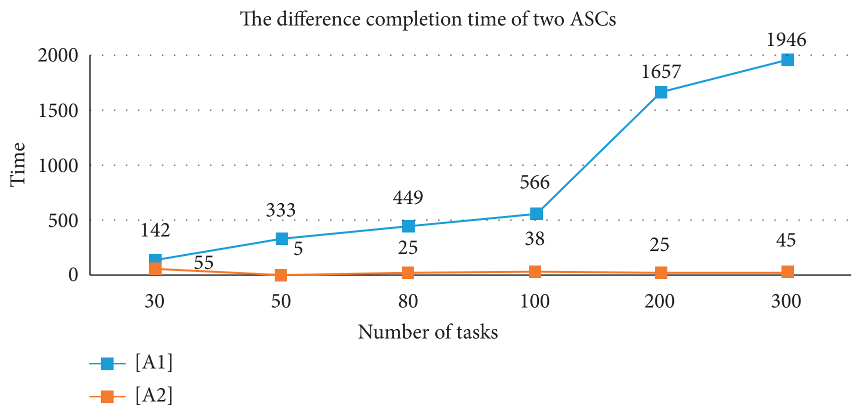


Figure 13: Difference completion time of two ASCs.

Table 5: Sensitivity test data.

Storage task ratio (%)	(A1) ASC1	(A1) ASC2	(A1) W'_{A1}	(A2) ASC1	(A2) ASC2	(A2) W'_{A2}	Gap3
10	1114	8806	8806	1264	8806	8806	—
20	1804	7629	7629	2102	7629	7629	—
30	2893	7147	7147	3588	7081	7081	0.93%
40	3994	6493	6493	5116	6295	6295	3.15%
50	4869	5411	5411	5188	5214	5214	3.78%
60	5717	4944	5717	5717	4944	5717	—
70	6847	4374	6847	6847	4374	6847	—
80	7577	2414	7577	7577	2414	7577	—
90	8793	1259	8793	8793	1259	8793	—

found that the algorithm we designed can always get better results than that obtained under the seaside priority strategy. However, due to the random nature of the tasks, there will be different effects of shortening the completion time under different tasks.

Figure 13 shows the difference between the completion time of the landside ASC and the seaside ASC of the two algorithms. From Figure 13, we can clearly see that the algorithm we designed effectively shortens the completion time difference between the two ASCs, thereby shortened the makespan. Moreover, with the expansion of the task scale, the effect is better.

For tasks of the same scale, the different proportions of storage tasks and retrieval tasks will also affect the results. Then we set the proportion of storage tasks to increase continuously when the task size $N = 100$, and record the completion time of the two ASCs in Table 5. We can analyze from the table that the algorithm we designed is only effective when the storage task ratio is 30%, 40%, and 50%.The reason is that when the storage task ratio is too small, by adjusting the task operation sequence, you can always get a set of tasks for the landside ASC to execute without conflict. Moreover, landside missions are more, the makespan of the landside ASC is the total completion time of the task. When the tasks on both sides are relatively balanced, conflict is inevitable, and the algorithm we designed has played a role. When the storage task ratio is greater than 50%, the makespan of the seaside ASC is longer than the makespan of the landside ASC. Once a conflict occurs, both algorithms will choose to let the landside ASC wait, so the same result will be obtained.

The Gap3 is defined by the formula as follows:

Table 6: Comparisons among (A2) and (A3).

Number of tasks	(A2) ASC1	ASC2	W_{A2}	(A3) ASC1	ASC2	W_{A3}	Gap4
30	1553	1608	1608	1999	2022	2022	25.75%
50	2601	2606	2606	3344	3385	3385	29.89%
80	4187	4162	4187	5573	5116	5573	33.10%
100	5032	5071	5071	6390	6126	6390	26.01%
150	8081	8043	8043	9902	10016	10016	24.53%
200	10931	10956	10956	13145	13827	13827	26.20%
300	16172	16217	16217	19931	19800	19931	22.90%

$$\mathrm{Gap3} = \frac{W'_{A1} - W'_{A2}}{W'_{A2}} \times 100\%. \quad (14)$$

5.3.3. Comparison with Setting the Handshake Area. After that, we compared the algorithm with the method in the Han [13] and set the handshake area as the middle position, that is the 20 bay position as the handshake area, regardless of the capacity of the relay area. As is shown in Table 6, the data obtained from the simulation indicate that because the task of crossing the handshake area is divided into two, the loading and unloading time is increased twice, and the makespan is also extended. Relative to setting the handshake area, the algorithm we designed can shorten the completion time by more than 20% under tasks of different scales. However, setting a handshake area can greatly reduce the number of possible conflicts between two ASCs.

The gap4 is defined by the formula as follows:

$$Gap4 = \frac{W_{A3} - W_{A2}}{W_{A2}} \times 100\%. \quad (15)$$

6. Conclusion

The interference between ASCs is the key to solve the scheduling problem. In this paper, the conflicts between ASCs were abstracted into an obstacle graph, and the conflicts were solved by avoiding obstacles in the graph theory model, a two-stage adaptive genetic algorithm based on graph theory was proposed to minimize the makespan of all tasks. The simulation experiment compared different algorithms to verify the speed and effectiveness of the designed algorithm. Comparative simulations were carried out under different number of containers, different task ratios and whether to set up the handshake area. The results shown that the proposed algorithm can effectively shorten the completion time between two ASCs, thereby shortening the makespan.

The scheduling of two ASCs in a block was considered in the current work. In the future, the impact of the choice of the size and location of the handshake area on scheduling is also a research point. Moreover, it is also possible to consider the scheduling optimization problem of two crossover ASCs in a block by transforming the conflict between the crossover ASCs into the form of an obstacle graph.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to acknowledge the Key Research Project Fund of Shanghai Maritime University: the Shanghai Pujiang Program under Grant 16PJC043.

References

[1] N. K. Park and S. C. Suh, “Tendency toward mega containerships and the constraints of container terminals,” *Journal of Marine Science and Engineering*, vol. 7, no. 5, p. 131, 2019.

[2] H. J. Carlo, I. F. A. Vis, and K. J. Roodbergen, “Storage yard operations in container terminals: literature overview, trends, and research directions,” *European Journal of Operational Research*, vol. 235, no. 2, pp. 412–430, 2014.

[3] E. Thanos, T. Toffolo, H. G. Santos, W. Vancroonenburg, and G. V. Berghe, “The tactical berth allocation problem with time-variant specific quay crane assignments,” *Computers & Industrial Engineering*, vol. 155, Article ID 107168, 2021.

[4] B. Liu, Z. C. Li, Y. Wang, and D. Sheng, “Short-term berth planning and ship scheduling for a busy seaport with channel restrictions,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 154, Article ID 102467, 2021.

[5] B. Liu, Z.-C. Li, D. Sheng, and Y. Wang, “Integrated planning of berth allocation and vessel sequencing in a seaport with one-way navigation channel,” *Transportation Research Part B: Methodological*, vol. 143, pp. 23–47, 2021.

[6] C. Chen, Z.-H. Hu, and L. Wang, “Scheduling of AGVs in automated container terminal based on the deep deterministic policy gradient (DDPG) using the convolutional neural network (CNN),” *Journal of Marine Science and Engineering*, vol. 9, no. 12, p. 1439, 2021.

[7] M. Zhong, Y. Yang, Y. Dessouky, and O. Postolache, “Multi-AGV scheduling for conflict-free path planning in automated container terminals,” *Computers & Industrial Engineering*, vol. 142, Article ID 106371, 2020.

[8] K. Guo, J. Zhu, and L. Shen, “An improved acceleration method based on multi-agent system for AGVs conflict-free path planning in automated terminals,” *IEEE Access*, vol. 9, pp. 3326–3338, 2020.

[9] H.-P. Hsu, C.-N. Wang, H.-P. Fu, and T.-T. Dang, "Joint scheduling of yard crane, yard truck, and quay crane for container terminal considering vessel stowage plan: an integrated simulation-based optimization approach," *Mathematics*, vol. 9, no. 18, p. 2236, 2021.

[10] J. Li, J. Yang, B. Xu, Y. Yang, F. Wen, and H. Song, "Hybrid scheduling for multi-equipment at U-shape trafficked automated terminal based on chaos particle swarm optimization," *Journal of Marine Science and Engineering*, vol. 9, no. 10, p. 1080, 2021.

[11] A. Narasimhan and U. S. Palekar, "Analysis and algorithms for the transtainer routing problem in container port operations," *Transportation Science*, vol. 36, no. 1, pp. 63–78, 2002.

[12] W. C. Ng and K. L. Mak, "Yard crane scheduling in port container terminals," *Applied Mathematical Modelling*, vol. 29, no. 3, pp. 263–276, 2005.

[13] X. Guo, S. Y. Huang, W. J. Hsu, and M. Y. H. Low, "Dynamic yard crane dispatching in container terminals with predicted vehicle arrival information," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 472–484, 2011.

[14] W. C. Ng, "Crane scheduling in container yards with intercrane interference," *European Journal of Operational Research*, vol. 164, no. 1, pp. 64–78, 2005.

[15] W. Li, Y. Wu, M. E. H. Petering, M. Goh, and R. d. Souza, "Discrete time model and algorithms for container yard crane scheduling," *European Journal of Operational Research*, vol. 198, no. 1, pp. 165–172, 2009.

[16] F. Chu, J. He, F. Zheng, and M. Liu, "Scheduling multiple yard cranes in two adjacent container blocks with position-dependent processing times," *Computers & Industrial Engineering*, vol. 136, pp. 355–365, 2019.

[17] S. Saini, D. Roy, and R. de Koster, "A stochastic model for the throughput analysis of passing dual yard cranes," *Computers & Operations Research*, vol. 87, pp. 40–51, 2017.

[18] A. H. Gharehgozli, G. Laporte, Y. Yu, and R. De Koster, "Scheduling twin yard cranes in a container block," *Transportation Science*, vol. 49, no. 3, pp. 686–705, 2015.

[19] H. J. Carlo and F. L. Martínez-Acevedo, "Priority rules for twin automated stacking cranes that collaborate," *Computers & Industrial Engineering*, vol. 89, pp. 23–33, 2015.

[20] Z.-H. Hu, J.-B. Sheu, and J. X. Luo, "Sequencing twin automated stacking cranes in a block at automated container terminal," *Transportation Research Part C: Emerging Technologies*, vol. 69, pp. 208–227, 2016.

[21] D. Briskorn and P. Angeloudis, "Scheduling co-operating stacking cranes with predetermined container sequences," *Discrete Applied Mathematics*, vol. 201, pp. 70–85, 2016.

[22] H. Lu and S. Wang, "A study on multi-ASC scheduling method of automated container terminals based on graph theory," *Computers & Industrial Engineering*, vol. 129, pp. 404–416, 2019.

[23] X. Han, Q. Wang, and J. Huang, "Scheduling cooperative twin automated stacking cranes in automated container terminals," *Computers & Industrial Engineering*, vol. 128, pp. 553–558, 2019.

[24] A. H. Gharehgozli, F. G. Vernooij, and N. Zaerpour, "A simulation study of the performance of twin automated stacking cranes at a seaport container terminal," *European Journal of Operational Research*, vol. 261, no. 1, pp. 108–128, 2017.

[25] F. Jaehn and D. Kress, "Scheduling cooperative gantry cranes with seaside and landside jobs," *Discrete Applied Mathematics*, vol. 242, pp. 53–68, 2018.

[26] D. Kress, J. Dornseifer, and F. Jaehn, "An exact solution approach for scheduling cooperative gantry cranes," *European Journal of Operational Research*, vol. 273, no. 1, pp. 82–101, 2019.

[27] H. Javanshir and A. G. S. Seyed, "Yard crane scheduling in port container terminals using genetic algorithm," *Journal of Industrial Engineering International*, vol. 6, pp. 39–50, 2010.