IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 15, 2020, accepted August 22, 2020, date of publication August 26, 2020, date of current version September 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3019703

Load-In-Load-Out AGV Route Planning in Automatic Container Terminal

YIXIANG XU[1,2], LIANG QI[1], (Member, IEEE), WENJING LUAN[1], (Member, IEEE), XIWANG GUO[3], (Member, IEEE), AND HUIJUAN MA[4]

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
[2]Department of Information Science, Beijing University of Technology, Beijing 100022, China
[3]College of Computer and Communication Engineering, Liaoning Shihua University, Fushun 113001, China
[4]Qingdao New Qianwan Container Terminal Company Ltd., Qingdao 266500, China

Corresponding author: Liang Qi (qiliang@sdust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61903229 and Grant 61973180, in part by the Natural Science Foundation of Shandong Province under Grant ZR2019BF004 and Grant ZR2019BF041, in part by the Shandong University of Science and Technology Research Fund under Grant 2019TDJH102, in part by the Scientific Research Foundation of the Shandong University of Science and Technology for Recruited Talents under Grant 2019RCJJ012, in part by the Liaoning Revitalization Talents Program under Grant XLYC1907166, in part by the Liaoning Province Department of Education Foundation of China under Grant L2019027, and in part by the Liaoning Province Dr. Research Foundation of China under Grant 20170520135.

ABSTRACT Efficient automatic guided vehicle (AGV) scheduling is the key to increase the throughput of automated container terminals. Traditional transport strategies cannot guarantee that AGVs are fully loaded during their traveling between the dock and the container yard, which leads to the insufficient utilization of AGVs. A load-in-load-out AGV route planning mode provides two-way loading between the dock and the container yard and thus improves the efficiency of container terminals. In this paper, a load-in-load-out AGV route planning model is designed with the help of a buffer zone where an AGV can carry at most two containers. A simulated annealing algorithm is used to solve it. Comparisons with the popular generic algorithm and particle swarm algorithm are made. Simulation results show that the proposed algorithm can effectively solve the problem and realize the bi-directional loading of AGVs, which is of great significance to improve the efficiency of the production of automated container terminals.

INDEX TERMS Load-in-load-out, AGV scheduling, automated container port, simulated annealing algorithm.

I. INTRODUCTION

As one of the most basic logistics and transportation modes, sea transportation has been playing an important role in economic trade and cultural communication among worldwide countries. With the development of the current frequent international trade, container terminals need to use advanced technologies to increase their efficiency and throughput. Efficient automatic guided vehicle (AGV) scheduling strategies are the key to ensure the high efficiency of an automatic container terminal, which has been studied by many scholars. They can support the full use of port resources and realize the efficient operations of the quayside and yard.

Studies about AGV scheduling can be divided into three aspects: path planning, real-time scheduling, and resource allocation optimization. Path planning means scheduling AGV to complete transportation tasks. It can use intelligent optimization algorithms or hybrid algorithms to plan the AGV routes. Mohammad and Saeed propose a mathematical model that is composed of a job shop scheduling problem and a conflict-free routing problem. A two-stage ant colony algorithm is also proposed to solve it [1]. Based on the characteristics of the A* algorithm, Zheng *et al.* optimize a node search method and use an angle evaluation cost function to find the path with the least inflection point [2]. Jerald and Asokan implement a non-traditional optimization technique called the adaptive genetic algorithm to do simultaneous scheduling [3]. Liu proposes an algorithm based on an immune algorithm and constructs AGV path planning by using a bipartite graph [4]. Chen *et al.* propose a new navigation method based on a fuzzy neural network by defining safety measures [5]. Cordeau *et al.*

The associate editor coordinating the review of this manuscript and approving it for publication was Yanbo Chen.

provide a unified tabu search heuristic algorithm for time windows and multi-warehouse vehicle paths [6]. Real-time scheduling is a strategy for avoiding collisions in a planned path. It implements deadlock-free and collision-free transport rules. This will achieve the goal of increasing the efficiency of the container terminals. Ayoub *et al.* propose a hybrid approach to solve the scheduling and path selection of automatically guided vehicles [7]. Nishi *et al.* propose a Petri net decomposition method for scheduling and conflict-free routing simultaneously for two-way automatic vehicle systems in dynamic environments [8]. Dimitri *et al.* propose a new strategy based on time logic to avoid collisions by delaying AGV operations [9]. Shi *et al.* present a two-stage scheduling strategy for offline shortest path library generation and online optimal scheduling scheme generation [10]. Ho develops a strategy that not only can prevent the collision of vehicles but also can avoid the disadvantage of fixed-zone strategies [11]. Gan *et al.* aim to minimize the end time of AGV tasks and the time of invalid operations and optimize the unloading operations of automatic container terminals [12]. Luo and Ni propose a method based on ordinary Petri nets to design the programmable logical controller to prevent collisions between vehicles in the automatic vehicle system [13]. Resource allocation focuses on disposing of containers in the shortest time with limited transportation resources. Singh *et al.* adopt a new scheduling rule and simulate the distribution efficiency and uniformity [14]. Toshiyuki *et al.* study the scheduling and path planning of AGV with a capacity [15]. Hadjar and Soumis consider a widespread branch-and-price approach to solve the multiple depot vehicle scheduling problems with time windows [16]. Confessore *et al.* propose an approach for solving the dispatching problem in an AGV system. The problem is modeled through a network by relying on the formulation of a minimum cost flow problem [17]. Kim and Chung present an analytical model to design a tandem AGV system with two-load AGVs [18]. Hassan and Edward define a scheduling problem for automated guided vehicles in container terminals and formulate as a minimum cost flow model [19]. A delay in one job may affect all the subsequent jobs served by the same quay crane. To handle this assignment problem, Song and Huang propose a hybrid metaheuristic method [20]. However, the aforementioned studies cannot guarantee that AGVs are fully loaded during their traveling between the dock and the container yard, which leads to the insufficient utilization of AGVs.

With the increasing throughput of the port, the traditional operation mode will cause a waste of time and resources. This will affect the whole transport link of the quayside bridge, AGV transport zone, and yard bridge. To the best of our knowledge, the existing big container ports cannot guarantee that AGVs are fully loaded during their traveling between the dock and the container yard, which leads to the insufficient utilization of AGVs. Also, in the existing studies few scholars study the considered fully loaded mode. Thus, this paper proposes a load-in load-out route planning for AGVs providing two-way loading between the dock and the container yard. It is designed with the help of a buffer zone. A simulated annealing algorithm is used to solve it. It can undoubtedly improve the efficiency of port transport. In this paper, we consider the strategy that an AGV can accommodate and carry two containers at the same time. In recent years, abundant achievements in various areas including scheduling in slab yard [21], manufacturing [22]–[25], transportation [26]–[28] and cooperative systems [29], [30] have been made with the help of intelligent optimization algorithms. We use a simulated annealing algorithm to solve the model and adopt a repetitive locations combination strategy to improve the efficiency of the algorithm.

The rest of the paper is organized as follows. Section II presents the formulation of the problem where an AGV can carry two containers. Based on a seaside buffer zone, it designs a load-in-load-out AGV scheduling model. Section III proposes a solution algorithm for the model based on a simulated annealing algorithm. Section IV makes a comparative analysis of the results. Finally, Section V concludes this paper and discusses the future work.

II. PROBLEM FORMULATION

The model studied in this paper is the multi-AGV path optimization model for a single cargo ship. Fig. 1 shows the part of seaside transportation area in Qingdao Qianwan Container Terminal. In this figure, there are some quay cranes, a ship on the left of quay cranes, and some AGVs under the quay cranes. At the seaside of the port, the task of AGVs is to transport imported containers from a quay crane to a yard crane, unload the containers, and return to the quay, or to transport exported containers from a yard crane to a quay crane, unload the containers, and return to the yard. We design a load-in-load-out AGV transportation (LILO) mode in this process. After the imported container is unloaded at a yard, the AGV loads another exported container and returns to the quay as shown in Fig. 2. The goal is to complete all containers transfer tasks between the sea-side and the yard in the load-in-load-out mode while minimizing the total traveling distance of all AGVs. A quayside buffer can ensure loose coupling between quayside work and trolley path planning, and thus the efficiency of port work can be increased. The buffer at

FIGURE 1. Part of seaside transportation area.

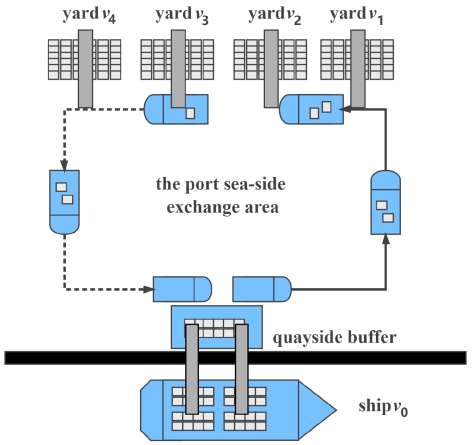

FIGURE 2. The load-in-load-out mode. After an imported container is unloaded at a yard, the AGV loads another exported container and returns to the quay.

the quayside bridge is designed in Fig. 3. Taking the loading process as an example, quayside buffers store containers. Containers are allocated from the bridge crane to the carriage, and the carriage can move from side to side between the tracks so that there is no requirement.

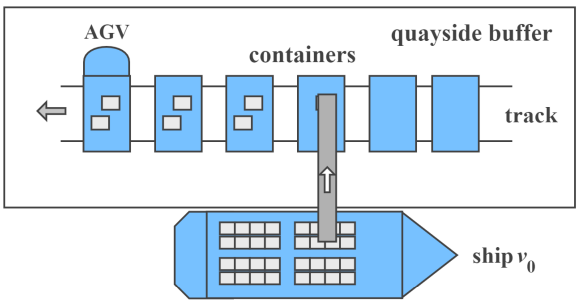

FIGURE 3. Quayside buffer where AGVs load or unload the carriages that are full of containers.

We give some assumptions about the model:

1) There is one ship, several AGVs, and a similar number of containers that need to be transferred from quayside to the yard and from the yard to the quayside.
2) Each AGV can carry two containers at most, and each task needs to transport containers from the quayside to the yard or from the yard to the quayside.
3) The quayside buffer can automatically realize the container loading and unloading of AGVs.

First, we give the notations that are used in the paper below.

1) N: $\mathrm{N} = \{1, 2, \ldots\}$ is the set of natural numbers;
2) N_m: $\mathrm{N}_m = \{1, 2, \ldots, m\}$;
3) S^*: the multiple sets of a set S;
4) V: $V = \{v_0; v_1, v_2, \ldots, v_n\}$ denotes a set of locations where v_0 represents a quayside bridge and v_i represents the ith location in the yard;
5) $V(v_i)$: $V(v_i) \in \{v_i\}^*$ represents a subset of a multiple set consisting of v_i, $i \in \mathrm{N}$;
6) X: $X \in \{V\text{-}\{v_0\}\}^*$ represents a set of locations where each location in the yard needs to receive a container that is unloaded from the ship;
7) Y: $Y \in \{V\text{-}\{v_0\}\}^*$ represents a set of locations where each location in the yard has one container that needs to be transported to the ship;
8) $|S|$: represents the number of elements in set S;
9) c: The maximum number of containers for each AGV;
10) m: $m \in \mathrm{N}$ is the total number of AGVs;
11) k: $k \in \mathrm{N}_m$ represents an index of an AGV;
12) d_{ij}: represents the distance between location v_i to v_j;
13) η^k: $\eta^k = x_1 x_2 \ldots x_j$ represents a sequence of locations that the kth AGV passes through during its transportation, and for the convenience we denote $x_i \in \eta^k$;
14) Q^k: $Q^k = \sum_{i,j} d_{ij}$ represents the distance traveled by the kth AGV;
15) l_i^k: it represents the number of loading containers at v_i in η^k, $l_i^k \in \{0, 1, 2, \ldots, c\}$;
16) u_i^k: it represents the number of unloading containers at v_i in η^k, $u_i^k \in \{0, 1, 2, \ldots, c\}$;
17) n_i^k: it represents the number of the containers carried by the kth AGV before its next loading or unloading task at v_i in η^k, $n_i^k \in \{0, 1, 2, \ldots, c\}$.

According to the LILO mode, the objective function is the shortest distance of AGVs that can accomplish all tasks.

$$\min Z = \sum_{k=1}^{m} Q^k \tag{1}$$

The constraints are as follows.

$$l_0^k = u_0^k = c, \quad \forall v_0 \in \eta^k \tag{2}$$

$$0 \le n_i^k \le c, \quad \forall v_i \in \eta^k \tag{3}$$

$$u_i^k \le n_i^k, \quad \forall v_i \in \eta^k \tag{4}$$

$$n_{i+1}^k = n_i^k - u_i^k + l_i^k, \quad \forall v_i \in \eta^k \tag{5}$$

$$\left(u_i^k\right)^2 + \left(l_i^k\right)^2 \ne 0, \quad \forall v_i \in \eta^k \tag{6}$$

$$\sum_{k=1}^{m} u_i^k = |V(v_i)|, \quad V(v_i) \subseteq Y, \ i \in N_n \tag{7}$$

$$\sum_{k=1}^{m} u_i^k = |V(v_i)|, \quad V(v_i) \subseteq X, \ i \in N_n \tag{8}$$

Constraint (2) means that AGV should fully load into and out of the quayside bridge, and this meets the requirements of the LILO mode. (3) means that the maximum carrying capacity of the AGV is no more than c. (4)-(5) restrict the number of loading and unloading containers for each AGV. (6) prohibits the illegal operation of neither loading nor unloading in the process of transportation at a location in the yard. (7) means that each unloading container at a location can be accomplished. (8) means that each loading container at a location can be accomplished.

III. PROPOSED ALGORITHM

First, we analyze the scale of the solution space of the problem. When $c = 1$ and $X = Y = n$. A basic sequential operation of an AGV is "unloading → loading" denoted by UL for short. Both the numbers of solutions for unloading containers in X and loading containers in Y are $n!$. When we have m AGVs, the solution space is $O(n!^2 \times C_{n-1}^{m-1})$. When $c = 2$ and $X = Y = n$. There are two basic sequential operations, i.e., "unloading → loading → unloading → loading" and "unloading → unloading → loading → loading" that are respectively denoted by $(\mathrm{UL})^2$ and $\mathrm{U}^2\mathrm{L}^2$ for short. We $\frac{2n}{4}$ assume that the number of carriage times by AGV is t, then $t = \frac{2n}{4}$, there are altogether 2^t modes of transport. Both the numbers of solutions for unloading containers in X and loading containers in Y are $n!$. When we have m AGVs, the solution space is $O(2^t \times n!^2 \times C_{n/2-1}^{m-1})$. It can be seen that the time complexity level of the exhaustive method is $O(n!)$. When the scale of the problem becomes larger, the traditional algorithm cannot solve the problem within an acceptable time. Therefore, simulated annealing algorithms for solving the problem are presented in this paper.

We adopt a two-stage coding scheme. It divides the code into two parts. The first part is an integer sequence representing an order of locations. The second part represents the number of locations traversed by AGV. Fig. 4 shows an example of the two-stage coding scheme.

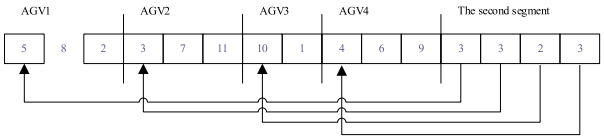

FIGURE 4. A two-stage coding scheme. The first segment represents the locations and the second segment represents the number of locations traversed by AGVs.

Now, we propose a code decomposition and combination (CDC) strategy. In order to make the algorithm have good convergence and avoid falling into local optimum, we screen the same locations in a scheduling process. At the same time, we analyze the condition that an AGV could carry two containers.

A. SIMULATED ANNEALING ALGORITHM BASED ON LILO

A simulated annealing algorithm was proposed by Metropolis *et al.* [31] in 1992. The idea comes from the similarity between the annealing process of solid matter in physics and general combinatorial optimization problems. The steps of the simulated annealing algorithm to solve this problem are as follows: a) Constructing an initial solution; b) Generating new solutions from the random perturbations of the old ones; c) Calculating the difference between the new-generated and the old solutions; d) Updating the solution according to a Metropolis rule; e) Going back to b) until reaching the maximum number of iterations.

B. CODING COMBINATION AND INITIALIZATION

First, we give some symbols that will be used in the following content.

$F(v)$: Mapping of location v to an integer, $F(v) \in \mathrm{N}$;

$F(v)^{-1}$: $F(v)^{-1} = v$ maps an integer to a location, $F(v)^{-1} \in V$;

CX: A set of repeated locations for unloading containers;

CY: A set of repeated locations for loading containers;

C: A set of locations in X and Y after removing duplicate elements;

α: A scheduling sequence;

α_i: The ith segment in an sequence, where $i \in \{1, 2, 3\}$, $\alpha_1 = \beta_1\beta_2 \ldots \beta_j$ and $\beta_i \in \{\beta_1, \beta_2, \ldots, \beta_j\}$ represents the sequence of locations that an AGV passes through during a legal loading and unloading process; α_2 represents the starting point of each car traveling sequence; $\alpha_3 = x_1x_2 \ldots x_j$, $x_i \in \{x_1, x_2, \ldots, x_j\}$, $x_i = 1, 2$, record the loading and unloading sequence combinations for each legitimate transportation, if $x_i = 1$, then β_i represents $\mathrm{U}^2\mathrm{L}^2$, if $x_i = 2$, then β_i represents $(\mathrm{UL})^2$.

Initial Solution Generation: Step1: Combining repeated locations. To reduce the size of the initial solution space, we can combine successive operations in the same yard. If $c = 1$, we select the same locations from X and Y to form C, and remove the locations of X and Y appearing in C to form X' and Y'. If $c = 2$, for every two identical locations in X, both locations are screened out from X and put into CX. For every two identical locations in Y, they are screened out from Y and put into CY. By referring to CX and CY, elements in X and Y are removed to generate X' and Y', the same elements are screened from X' and Y' to form C, and the locations that appear in C are removed.

Step2: Coding initialization. The location v in X', Y', CX, CY and C is mapped to $F(v)$, and $F(v)$ is coded to α. If $c = 1$, then $\alpha = \alpha_1\alpha_2$, else $\alpha = \alpha_1\alpha_2\alpha_3$.

C. ALGORITHM PROCESS

In addition to the symbols defined in the problem model and the initial solution generation process, symbols and formulas in the simulated annealing algorithm are as follows.

T_0: Initial temperature, $T_0 \in \mathrm{N}$;

T_{end}: End temperature, $T_{end} \in \mathrm{N}$;

T_v: Cooling rate, $0 \leq T_v \leq 1$;

L: The number of iterations per temperature, $L \in \mathrm{N}$;

l: Current iteration number, $l \in \mathrm{N}$;

$Pathlength(\alpha)$: $Pathlength(\alpha) = \sum_{i,j} d_{ij}, v_i, v_j \in \alpha_1$, the distance traveled in a sequence.

Input: locations V, set of unloading locations X, set of loading locations Y, the number of AGVs m, the maximum capacity c, the number of iterations per temperature L, the initial temperature T_0, the end temperature T_{end}, cooling rate T_v.

Output: legally scheduled travel distance Z.

Step1: Code split and set the initial value. Elements in α_1 are composed into unloading sequence α_x and loading sequence α_y, we initialize $l = 1$, and $\alpha_{best} = \alpha$;

Step2: Generate new solutions. Two randomly selected elements in α_x and α_y are exchanged to produce a new solution;

Step3: Accept new solutions. We combine α_x and α_y into a new solution α', let $dp = Pathlength(\alpha')$- $Pathlength(\alpha)$. If $dp < 0$, then $\alpha = \alpha'$, otherwise, we accept the α' with a probability of -dp/c. If $\alpha_{best} > \alpha$, $\alpha_{best} = \alpha$;

Step4: If $l < L$, then $l = l + 1$ and return Step2, else enter Step 5;

Step5: Iteration. $T_0 = T_0 * T_v$, if $T_0 < T_{end}$, then return Step1, else end the algorithm and map α_{best} back to the sequence of locations $F(v)^{-1}$, output the travel distance Z.

The flowchart of the algorithm is shown in Fig. 5.

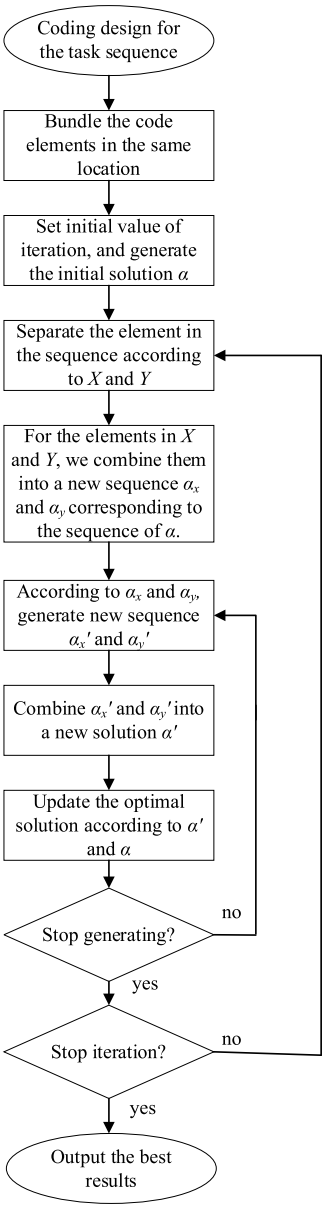

FIGURE 5. The flowchart of the algorithm.

Since the algorithm mutates, crosses, and reverses sequence codes in this algorithm process, there will be illegal solutions after the operation. For example, support $c = 1$, 1 represents the quayside location, 5 and 4 represent unloading locations, 3 and 6 represent the loading location, the sequence is [1 5 3 1 4 6 1] meaning that an AGV loads a container from the quayside and unload at the location 5 in the yard, then loads another container from the location 3 and returns to the quayside. The operation of locations 4 and 6 is the same as described above. In the mutation operation, it selects the 2 and 3 positions in a sequence and the sequence after mutation is [1 3 5 1 4 6 1]. At this time, the sequence [1 3 5 1] is illegal, because it is illegal to load the container first and then unload the container when the load is full. In order to solve the above problems, we adopt a CDC strategy to fix the operation of each position in the sequence. In fact, in the case of $c = 1$, the operation of each position in the sequence is fixed, but the location corresponding to this position is uncertain. In other words, position 2 in the example can only be an unloading operation, but the location of position 2 may be 5 or 4. First, the unloading sequence [5 4] and loading sequence [3 6] are separated according to the sequence. Then, mutation and reversal operations were performed on the two sequences respectively. We assume that the two sequences after the operation are [4 5] and [3 6], we can get a sequence [1 4 3 1 5 6 1]. The above operation solves the problem of generating illegal solutions during mutation or reversal.

In the case that $c = 2$, there are two situations in one carriage: $(UL)^2$ and U^2L^2. At this point, the sequence will also produce a solution that does not conform to LILO when performing cross operations. For example, there are two sequences [1 2 3 4 5 1][1] and [1 2 5 3 4 1][2]. The first section represents the scheduling sequence of the AGV, while the second section 1 represents the transport strategy of the car in this transportation as U^2L^2, 2 means that the transportation strategy is $(UL)^2$. 1 represents quayside location, 2 and 3 represent unloading locations in the yard, 4 and 5 represent loading locations. During the crossover operation, it is assumed that we select the 3rd position in the sequence, and the sequence after the crossover is [1 2 5 4 5 1][1] and [1 2 3 3 4 1][2]. [1 2 5 4 3 1][1] and [1 2 3 5 4 1][2] are generated after the conflict is eliminated by means of partial mapping. Now the first sequence produces an illegal sequence of ULLU. To solve the above situation, we still adopt the CDC strategy. According to the sequence, the unloading sequence [2 3][2 3] and loading [4 5][5 4] were separated, and then the crossover operation was performed for each set of sequences. Assuming that the two sets of sequences after the operation are [2 3][2 3] and [5 4][4 5], then new solutions [1 2 3 5 4 1][1] and [1 2 4 3 5 1][2] are combined according to the transport strategy in the sequence. The above operation solves the problem of illegal solutions when crossing.

IV. SIMULATIONS

The parameters of the simulation, including the size of the AGV transportation area, and the number of storage yards according to the Qingdao New Qianwan Container Terminal, China. The tool used in the simulations is MATLAB R2019a.

To measure the optimization ability of the model, we set $c = 1, 2$, and the initial values $m = 2$, $t_{max} = 100$, $T_0 = 1000$, $T_{end} = 100$, $T_v = 0.9$, and $L = 100$. The results are shown in Tables 1-4. Within the scope of small-scale locations, an optimization precision of simulated annealing algorithm (SA) can reach 100%. With the increase of problem scale, SA can significantly reduce the time required for solving problems. When the scale of locations continues to increase, the results cannot be obtained by exhaustive search (EA) within an acceptable time. At this time, SA algorithm can find a good solution to realize LILO in a very short time, which reflects the good optimization ability of the algorithm.

TABLE 1. Comparison of algorithm results ($c = 1$).

Number of tasks	EA	SA
4	400	400
5	520	520
6	480	480
7	580	580

TABLE 2. Algorithm running time comparison (sec) ($c = 1$).

Number of tasks	EA	SA
4	0.6	1.46
5	1.22	1.67
6	23.73	2.13
7	1479.7	2.17

TABLE 3. Comparison of algorithm results ($c = 2$).

Number of tasks	EA	SA
4	300	300
6	340	340

TABLE 4. Algorithm running time comparison (sec) ($c = 2$).

Number of tasks	EA	SA
4	0.84	0.84
6	536	1.35

According to the LILO mode and algorithm optimization process in this paper, we made a comparison with the other two popular algorithms, i.e., genetic algorithm and mixed particle swarm optimization. When $|X|, |Y| = 16$ and 20, we set the initial parameters as $m = 3$, $c = 2$, $T_{max} = 300$, $T_0 = 2000$, $T_{end} = 0.001$, $T_v = 0.95$, $L = 100$. After running the algorithm multiple times, we adopt the best of them as a final result. The simulated results are shown in Figs. 6. SA has the best effect in finding the optimal solution although the convergence speed is a little bit slower than that of a genetic algorithm.

In order to measure the influence of the increasing number of task points on the convergence performance of the algorithm, the evolutionary process of the three algorithms is compared and analyzed when $|X|$ and $|Y| = 8$, 12, 16, and 20. We set the initial value of parameters as: $m = 2$, $c = 2$, $T_{max} = 300$, $T_0 = 2000$, $T_{end} = 0.001$, $T_v = 0.95$, $L = 100$. Simulated results as shown in Tables 5 and 6 and

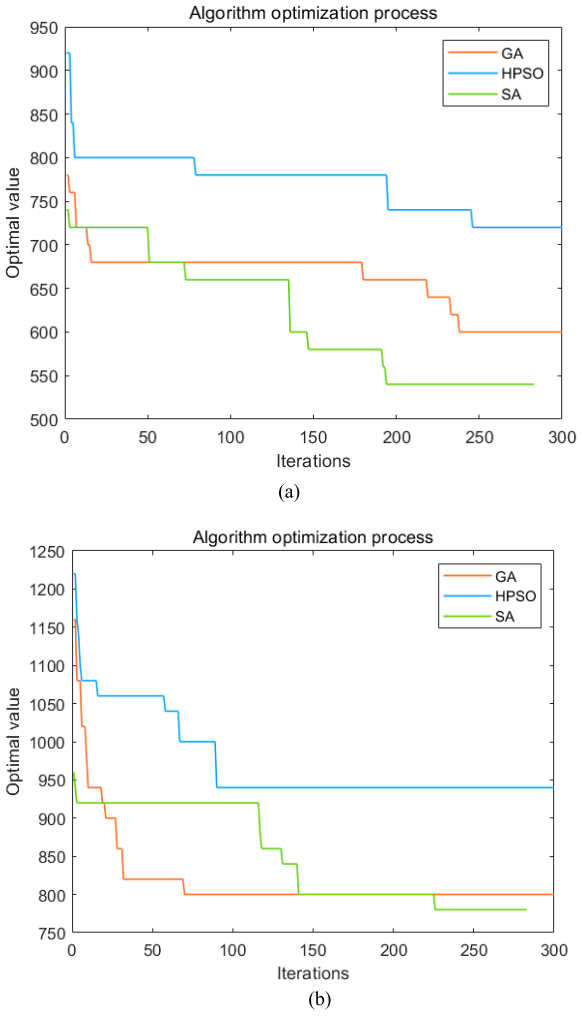

FIGURE 6. Comparison of the three optimization algorithms. The figure (a) is the result of the simulations at X and $Y = 16$. The figure (b) is the result of the simulations at X and $Y = 20$.

TABLE 5. Comparison of simulation results.

Number of tasks	GA	HPSO	SA
16	420	480	420
24	640	700	580
32	920	1080	900
40	980	1180	980

TABLE 6. Running time (sec).

Number of tasks	GA	HPSO	SA
16	2.770627	4.490954	2.590724
24	3.501646	4.713754	3.065533
32	4.318039	5.532288	4.211876
40	4.538350	5.703430	4.398270

Fig. 7, the genetic algorithm converges on 12th, 34th, 75th and 124th generations (Fig. 7 (a)), hybrid particle swarm optimization converges on 63th, 115th, 220th and 203rd generation (Fig. 7 (b)), simulated annealing algorithm converges on 88th, 212th, 219th and 229th generations (Fig. 7 (c)). It can be seen from the simulations that as

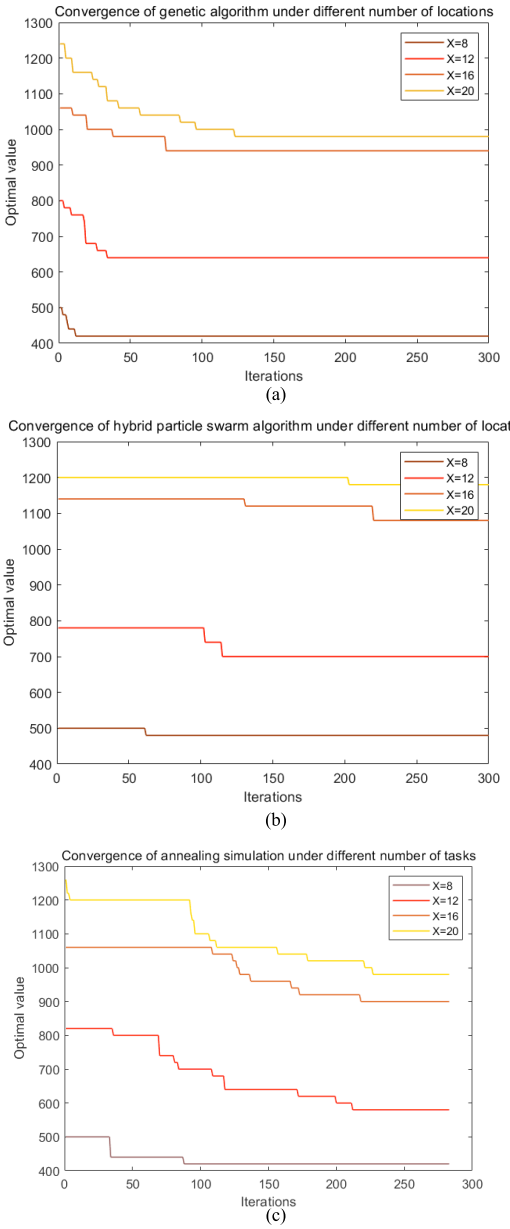

FIGURE 7. Iteration of the algorithm under different tasks.

the number of locations increases, the convergent number of iterations of the three algorithms also increases, but the algorithm can still solve the problem in a short time, which shows the applicability of the algorithm in this problem.

The scenario in this paper concerns a single ship, so the number of containers is not very large. For small-scale problems, we can get the exact optimal solution; and for medium-scale problems, we have compared with classical intelligent algorithms, and get good solutions.

V. CONCLUSION

To improve the working efficiency of port transportation, we propose a new AGV scheduling strategy – load in load out mode. We construct the mathematical model of the mode of LILO and solve the problem based on the simulated annealing algorithm. According to the simulated results, the improved algorithm is effective in solving the problem. To the best of our knowledge this is the first study a load-in-load-out AGV route planning mode providing two-way loading between the dock and the container yard. It can improve the energy-efficiency of AGVs as well as of efficiency of container terminals. The proposed algorithms can be used in automated container ports. A necessary condition is that the numbers of imported and exported containers are similar. With further exploration of the model, we will consider multiple ships, the specific design of the quayside buffer and use a time window to study the task scheduling process. At the same time, when the problem scale becomes very large, the simulated annealing algorithm may not find the optimal results due to the blind search issue, which is also a problem to be solved in our future work. We can also use some novel heuristic algorithms to solve the model to improve the accuracy of the problem. To improve the applicability of the model, we will also consider detecting path conflicts in AGV scheduling.

REFERENCES

[1] M. Saidi-Mehrabad, S. Dehnavi-Arani, F. Evazabadian, and V. Mahmoodian, "An ant colony algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs," *Comput. Ind. Eng.*, vol. 86, pp. 2–13, Aug. 2015.

[2] T. Zheng, Y. Xu, and D. Zheng, "AGV path planning based on improved A-star algorithm," in *Proc. IEEE 3rd Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, Chongqing, China, Oct. 2019, pp. 1534–1538.

[3] J. Jerald, P. Asokan, R. Saravanan, and A. D. C. Rani, "Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm," *Int. J. Adv. Manuf. Technol.*, vol. 29, nos. 5–6, pp. 584–589, Sep. 2006.

[4] S. N. Liu, "Optimization and scheduling of AGV in automated warehouse system based on immune algorithm," in *Proc. Int. Conf. Autom. Control Artif. Intell. (ACAI)*, 2012, pp. 1492–1495.

[5] C. Wuwei, J. K. Mills, and S. Wenwu, "A new navigation method for an automatic guided vehicle," *J. Robotic Syst.*, vol. 21, no. 3, pp. 129–139, 2004.

[6] J.-F. Cordeau, G. Laporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *J. Oper. Res. Soc.*, vol. 52, no. 8, pp. 928–936, Aug. 2001.

[7] A. I. Corréa, A. Langevin, and L.-M. Rousseau, "Scheduling and routing of automated guided vehicles: A hybrid approach," *Comput. Oper. Res.*, vol. 34, no. 6, pp. 1688–1707, Jun. 2007.

[8] T. Nishi and Y. Tanaka, "Petri net decomposition approach for dispatching and conflict-free routing of bidirectional automated guided vehicle systems," *IEEE Trans. Syst., Man, Cybern.-A, Syst. Humans*, vol. 42, no. 5, pp. 1230–1243, Sep. 2012.

[9] D. Antakly, J. J. Loiseau, and R. Abbou, "A temporised conflict-free routing policy for AGVs," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11169–11174, Jul. 2017. [Online]. Available: https://www.sciencedirect.com/journal/ifac-papersonline /issues

[10] Y. Shi, X. Wang, X. Sun, R. Xie, and X. Zheng, "A two-phase strategy with micro genetic algorithm for scheduling Multiple AGVs," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Banfa, BC, Canada, Oct. 2016, pp. 003101–003106.

[11] Y.-C. Ho, "A dynamic-zone strategy for vehicle-collision prevention and load balancing in an AGV system with a single-loop guide path," *Comput. Ind.*, vol. 42, nos. 2–3, pp. 159–176, Jun. 2000.

[12] Y. F. Gan, "Scheduling problems of automated guided vehicles in automated container terminals using a genetic algorithm," in *Proc. IOP Conf., Mater. Sci. Eng.*, Guangzhou, China, 2020.

[13] J. Luo and H. Ni, "Programmable-logical-controllers synthesis for Automated-guided-vehicle systems using ordinary Petri nets," *Asian J. Control*, vol. 16, no. 6, pp. 1760–1770, Nov. 2014.
[14] N. Singh, P. V. Sarngadharan, and P. K. Pal, "AGV scheduling for automated material distribution: A case study," *J. Intell. Manuf.*, vol. 22, no. 2, pp. 219–228, Apr. 2011.
[15] T. Miyamoto and K. Inoue, "Random search for dispatch and conflict-free routing problem of capacitated AGV systems," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Manchester, U.K., Oct. 2013, pp. 1611–1615.
[16] A. Hadjar and F. Soumis, "Dynamic window reduction for the multiple depot vehicle scheduling problem with time windows," *Comput. Oper. Res.*, vol. 36, no. 7, pp. 2160–2172, Jul. 2009.
[17] G. Confessore, M. Fabiano, and G. Liotta, "A network flow based heuristic approach for optimising AGV movements," *J. Intell. Manuf.*, vol. 24, no. 2, pp. 405–419, Apr. 2013.
[18] K. S. Kim and B. D. Chung, "Design for a tandem AGV system with two-load AGVs," *Comput. Ind. Eng.*, vol. 53, no. 2, pp. 247–251, Sep. 2007.
[19] H. Rashidi and E. P. K. Tsang, "A complete and an incomplete algorithm for automated guided vehicle scheduling in container terminals," *Comput. Math. Appl.*, vol. 61, no. 3, pp. 630–641, Feb. 2011.
[20] L. Q. Song and S. Y. Huang, "A hybrid metaheuristic method for dispatching automated guided vehicles in container terminals," in *Proc. IEEE Symp. Comput. Intell. Scheduling (CISched)*, Apr. 2013, pp. 52–59.
[21] X. Wang, M. C. Zhou, Q. H. Zhao, X. W. Guo, and L. Qi, "A branch and price algorithm for crane assignment and scheduling in slab yard," *IEEE Trans. Autom. Sci. Eng.*, early access, Jun. 19, 2020, doi: 10.1109/TASE.2020.2996227.
[22] X. W. Guo, M. C. Zhou, S. X. Liu, and L. Qi, "Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption," *IEEE Trans. Automat. Sci. Eng.*, early access, Jun. 19, 2020, doi: 10.1109/TASE.2020.2992220.
[23] X. Guo, M. Zhou, S. Liu, and L. Qi, "Lexicographic multiobjective scatter search for the optimization of sequence-dependent selective disassembly subject to multiresource constraints," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3307–3317, Jul. 2020.
[24] J. Zhao, S. Liu, M. Zhou, X. Guo, and L. Qi, "Modified cuckoo search algorithm to solve economic power dispatch optimization problems," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 4, pp. 794–806, Jul. 2018.
[25] Y. Fu, M. Zhou, X. Guo, and L. Qi, "Scheduling dual-objective stochastic hybrid flow shop with deteriorating jobs via bi-population evolutionary algorithm," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Apr. 16, 2019, doi: 10.1109/TSMC.2019.2907575.
[26] L. Qi, M. Zhou, and W. Luan, "A two-level traffic light control strategy for preventing incident-based urban traffic congestion," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 13–24, Jan. 2018.
[27] L. Qi, M. Zhou, and W. Luan, "Impact of driving behavior on traffic delay at a congested signalized intersection," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 7, pp. 1882–1893, Jul. 2017.
[28] L. Qi, M. Zhou, and W. Luan, "A dynamic road incident information delivery strategy to reduce urban traffic congestion," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 5, pp. 934–945, Sep. 2018.
[29] L. Qi, W. J. Luan, X. S. Lu, and X. W. Guo, "Shared P-Type Logic Petri Net Composition and Property Analysis: A Vector Computational Method," *IEEE Access*, vol. 8, pp. 36386–36397, 2020.
[30] W. Luan, L. Qi, Z. Zhao, J. Liu, and Y. Du, "Logic Petri net synthesis for cooperative systems," *IEEE Access*, vol. 7, pp. 161937–161948, 2019.
[31] M. Steinbrunn, G. Moerkotte, and A. Kemper, "Heuristic and randomized optimization for the join ordering problem," *VLDB J. Int. J. Very Large Data Bases*, vol. 6, no. 3, pp. 191–208, Aug. 1997.

YIXIANG XU received the B.S. degree in computer science and technology from the Shandong University of Science and Technology, Qingdao, China. He is currently pursuing the M.S. degree with the Beijing University of Technology. His main research interests include optimization algorithms and intelligent logistics.

LIANG QI (Member, IEEE) received the B.S. degree in information and computing science and the M.S. degree in computer software and theory from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2017. From 2015 to 2017, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He is currently with the Shandong University of Science and Technology. He has published over 60 papers in journals and conference proceedings, including the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and the IEEE TRANSACTIONS ON CYBERNETICS. His interests include Petri nets, optimization, cyber-physical systems, and intelligent transportation systems.

WENJING LUAN (Member, IEEE) received the B.S. and M.S. degrees from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2018. From May to July 2017, she was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. She is currently a Lecturer of computer science and technology with the Shandong University of Science and Technology. She has published over 20 papers in journals and conference proceedings, including the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, and the IEEE INTERNET OF THINGS JOURNAL. Her current research interests include optimization algorithms and intelligent transportation systems. She received the Best Student Paper Award-Finalist from the 13th IEEE International Conference on Networking, Sensing, and Control (ICNSC) Conference in 2016.

XIWANG GUO (Member, IEEE) received the B.S. degree in computer science and technology from the Shenyang Institute of Engineering, Shenyang, China, in 2006, the M.S. degree in aeronautics and astronautics manufacturing engineering from Shenyang Aerospace University, Shenyang, in 2009, and the Ph.D. degree in system engineering from Northeastern University, Shenyang, in 2015. From 2016 to 2018, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He is currently an Associate Professor with the College of Computer and Communication Engineering, Liaoning Shihua University, Fushun, China. He has published more than 50 technical papers in journals and conference proceedings, including the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, and the IEEE/CAA JOURNAL OF AUTOMATICA SINICA. His current research interests include Petri nets, remanufacturing, recycling and reuse of automotive, and intelligent optimization algorithms.

HUIJUAN MA received the B.S. and M.S. degrees in computer science and technology from the Shandong University of Science and Technology, Qingdao, China, in 2015 and 2018, respectively. She is currently with Qingdao New Qianwan Container Terminal Company Ltd., Qingdao. Her main research interests include optimization algorithms and intelligent logistics.