Mathematical Biosciences and Engineering

MBE, 17 (5): 5150–5172.
DOI: 10.3934/mbe.2020279
Received: 26 April 2020
Accepted: 13 July 2020
Published: 31 July 2020

http://www.aimspress.com/journal/MBE

Research article

Multi-AGV dispatching and routing problem based on a three-stage decomposition method

Yejun Hu *, Liangcai Dong and Lei Xu

College of Logistics Engineering, Shanghai Maritime University, Shanghai, 201306, China.

***Correspondence:** Email: Anthony19770909@hotmail.com.

Abstract: Automatic guided vehicle (AGV) is a device for horizontal transportation between quay cranes and yard cranes in an automated container terminal. In which dispatching and routing problem (DRP) of the AGV system is a vital as well as basic issue. In the application of the actual AGV system, several practical factors including avoiding conflicts, path smoothness, difficulty in adjusting routes and anti-interference must be considered. The present study establishes the model with the goal of minimizing AGV travel distance, reducing operation time and response time. Furthermore, a three-stage decomposition solution to the problem was proposed by combining the advantages of pre-planning algorithm and real-time planning algorithm, which combines A* algorithm with the principle of time window to plan the path of each AGV in time order. Finally, the effectiveness of this method in path search and time optimization is illustrated and the system efficiency is improved by comparing and analyzing the calculation examples of different scales.

Keywords: Multi-AGV systems; dispatching and routing problem; three-stage decomposition; no conflict

1. Introduction

As is shown in Figure 1, the classic structure of automated container terminal (ACT) is comprised of three types of equipment, including automatic quay bridge (AQC), automatic yard crane (AYC) and trucks used to transport containers outside the yard and terminal. Moreover, the AQC and AYC are large-sized and small-sized wharf gantry cranes used for loading and unloading containers on ships and yards, respectively [1,2]. In the horizontal transportation area of the ACT, the three types of

equipment, AQC, AGV and AYC, cooperate with each other closely to complete the dispatch of containers between the shore and the yard. Serving as a carrier of transportation, the inefficiency of the scheduling and path planning of AGV will directly reduce the efficiency of the entire ACT. Efficient multi-AGV system could significantly improve the efficiency of container handling in ACT accordingly [3–5]. Therefore, the multi-AGV system has become a vital part of ACT and warehousing facilities due to its ability to transport various goods without manual intervention continuously, safely and efficiently. However, numerous uncertain factors in the horizontal transportation of ACTs still exist, which require highly flexible route planning. Under the multiple influences of continuous increase in labor cost, large-scale ships, and intelligent port, the efficiencies of AGV operations are not only affected by the terminal environment, but also by path planning, equipment scheduling and other factors. Additionally, the issues of equipment waiting, conflicts, deadlocks and so forth during its operation are also becoming increasingly prominent, which has turned into an urgent difficulty to be solved at the current stage of automated terminals [6].

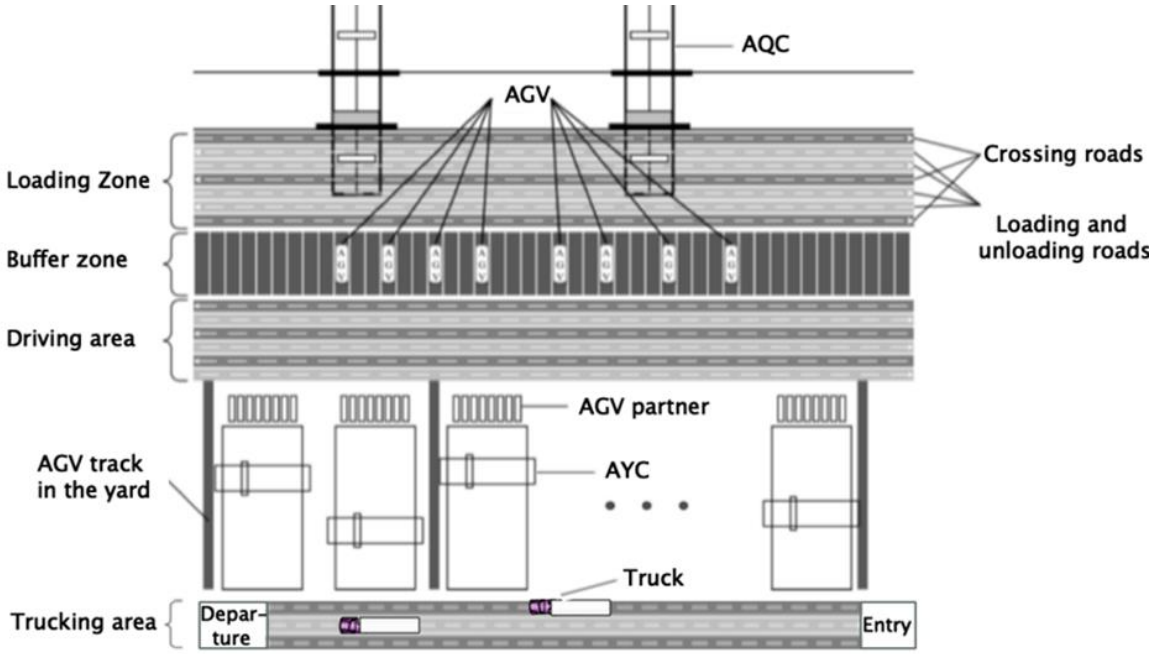

Figure 1. The layout of automated container terminals.

Over the few decades, scholars at home and abroad have performed plenty of studies about the path planning of AGV, including static and dynamic plannings. Ghasemzadeh et al. [7] established a multi-AGV static path model with a time window which targeted at minimizing the maximum task volume passed by each edge in the simulation graph. Oboth et al. [8] proposed a simple and practical continuous trajectory generation algorithm, which could automatically generate the shortest collision free pathway for multi-AGV system. The above methods effectively reduce the conflicts, which were only for static path planning and possess poor dynamic adaptability as well as failing to consider the operation time and path length. In order to overcome the shortcomings of static path planning, Chen et al. [9] adopted the dynamic path planning and proposed a conflict-free path approach for shortest

path planning based on the time window model, but with too long total travel time. Gong et al. [10] also planned the running path and state scheduling of each AGV based on the time window model. However, this method focused on the prevention of collision and failed to consider the real time obstacle avoidance and shortcomings such as low system efficiency exist when continuous collisions occur [11].

Scholars also carried out a quantity of investigations on the path algorithm, mainly including classic methods such as Dijkstra algorithm, A* algorithm, and intelligent algorithm such as genetic algorithm. In the static environment, A* algorithm [12,13] is characterized by optimality, completeness and efficiency, and is widely used in path planning. A large number of scholars have acquired improvements on the shortcomings of A* algorithm and explored more efficient algorithms. Du F and Xin Yu et al. [14,15] reduced the turning point of the final search path and avoided excessive increase of the path length as well as additional AGV delivery time by changing the search direction in the path search, such as octagonal position, sixteen azimuth or even infinite azimuth search. LIN M et al. [16] introduced the real-value coefficient of heuristic function to improve the real-time performance of A* algorithm. These methods focused on the improvements of the heuristic function of A* algorithm and the search efficiency, but these studies on AGV failed to fully consider the conflict factor, and most of the algorithms adopted returned to the task scheduling or path planning when solving the multi-AGV path conflict problem, which increases the complexity of the research and the difficulty of problem solving [17].

As can be seen from the previous part, the current research on the AGV system is not comprehensive enough to meet the actual work requirements. There are no comprehensive investigations that take time and path lengths together, the method of path planning is highly efficient but the planned paths are too tortuous, and more focus on collision prevention rather than real-time obstacle avoidance when investigating system conflicts. Therefore, a clustered AGV system (AGVs) composed of multiple AGV should comprehensively consider the following items in order to meet the operational requirements of actual ACT:

1) Research on coordination mechanism of multi-AGV scheduling. It can shorten the length and the average time for AGVs to respond to tasks, and ensure the efficient flow of containers and Just In Time (JIT) supply.

2) Research on path planning. The method of path planning should be efficiency as well as combine AGV's own conditions and environmental conditions. The AGVs in ATC are very long and difficult to turn, so the planned path needs to reduce the number of turns. What's more, when obstacles are generated in the environment, the planned path needs to avoid obstacles to reduce the possibility of conflict.

3) Research on conflict-free operation mechanism. The scheduling strategy and path planning are formulated according to the production requirements to avoid system collision and deadlock which further realize the conflict-free and efficient operation of AGVs.

4) Research on rescheduling problem, which reflects the study of the real-time dynamics of the system. In case of system emergency, equipment failure, receiving urgent orders or plug-in production, it is necessary to terminate the original plan and task, and quickly re-complete the formulation of AGVs scheduling policy and its path planning.

Therefore, to solve the above issues above, a three-stage scheduling and path planning strategy

was proposed to establish a multi-AGV scheduling model based on the existing researches on AGV scheduling and path planning. In general, the task process is divided into three stages: task assignment based on quick response, path planning based on shortest path, and path re-planning based on both conflict type and processing time. Moreover, the shortest path with the least number of turns is planned based on the improved A* algorithm. The AGV collisions in the path will be identified by simulating the time window, and the time to deal with collisions is carefully investigated. Finally, the path with the least time to deal with conflicts is also selected. This method will effectively improve the horizontal transportation efficiency of the automated container terminal and reduce the time of container transportation in the port.

2. Mathematical model

2.1. Problem description and analysis

As shown in Figure 2, the horizontal transportation area of ACT can be subdivided into four parts: quay crane operation area, which are responsible for interacting with the quay cranes to complete the loading or unloading of the container; horizontal buffer zone, where AGV will dock to if no new task is temporarily assigned after completing the current task; high-speed driving area, where multiple one-way lanes; yard interaction area, which interacts with the shore and the yard and is usually equipped with AGV partner. When AGV lines to the partner, it will set up the AGV to complete the subsequent docking with the yard crane. In the horizontal transportation area, three types of equipment, AQC, AGV and AYC, cooperate closely each other to complete the dispatching of containers between shore ships and yards. The main steps of the operation of multi-AGV system is as follows [18]:

1) Receive container operation instruction;

2) Assign an AGV for the container and plan a route for the AGV to get to the position of the container;

3) Load the container to the AGV by joint operation with loading and discharging equipment (QC/YC);

4) Plan a reasonable driving path and transport the container to the discharging position；

5) Cooperate with the loading and discharging equipment (YC/QC) to remove the container from the AGV and this completes the loading and discharging of the container once;

6) Choose one position in the buffer area and plan a reasonable path for the AGV to get there;

7) Wait for receiving the next operation instruction and repeat the above process.

In the above steps, AGV quantity configuration, task allocation and path planning are three indispensable parts. Among of them, the study on the effects of different AGV configurations quantities on the actual efficiency are analyzed by the comparative experiment of the actual case in the fourth part of the article. And the fixed value is set in the model and algorithm solution. Task allocation is a key part of the AGV scheduling system, and served as a role to achieve the best match between transportation tasks and vehicles. In AGV path planning, an urgent issue that needs to be solved is how to find the most reasonable path, which means it is necessary to comprehensively consider the three factors of path length, time cost and the number of turns in path planning.

At the same time, the average response time of the task reflects the waiting time of the quay cranes and the yard cranes. Shortening the waiting time of the equipment can effectively improve the turnover

rate of the container. Therefore, the consideration of time needs to think about the average response time of the task and whole time of the task.

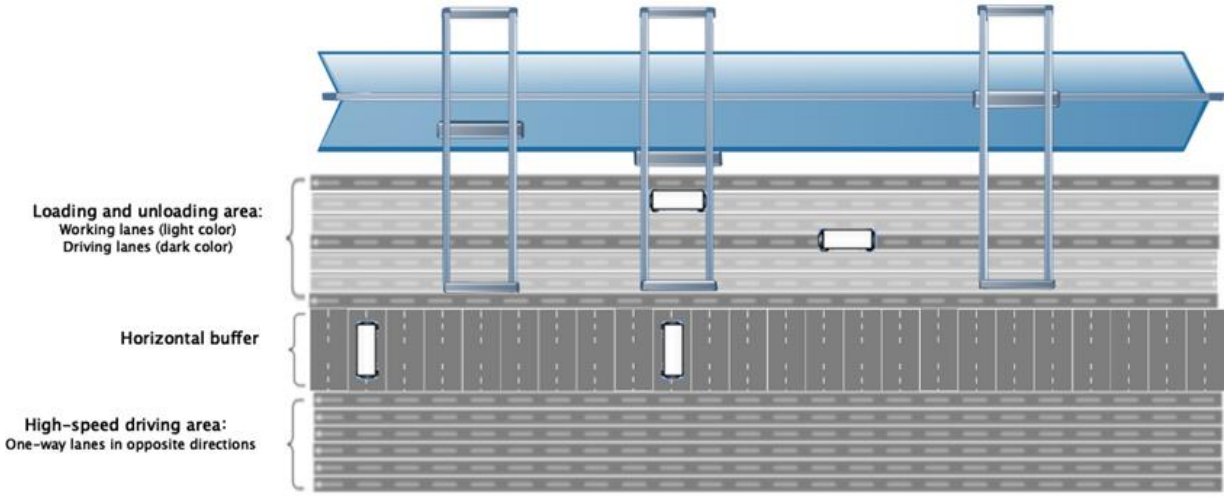

Figure 2. The layout of the horizontal transportation area of the automated terminal.

2.2. Assumptions and variable descriptions

2.2.1. Assumptions

In order to facilitate the study of the AGV path, the path network (See section 3.1.1 for details) and AGV of the ACT need to be specified as follows:

1) Multiple AGVs correspond to multiple quay cranes and do not serve for a fixed QC;

2) Each task is completed by one AGV. Each AGV can only hold one container on each service, and the battery of AGV is enough;

3) Each quay cranes serves for multiple YCs. The positions of the QCs and YCs remain known;

4) When the AGV arrives at the loading or discharging position and waits for YCs or QCs to load or discharge the container, the waiting time is subject to an integer random distribution of (60s, 100s);

5) AGV is driving at a constant speed, without considering the effect of turning and weight of the container on AGV speed;

6) The width of the path can only accommodate one AGV at same time.

2.2.2 Parametric symbols and decision variables

This section explains the parameter symbols, given constants, decision variables, and auxiliary decision variables involved in the model to facilitate the understanding of objective functions and constraints.

1) Parameter collection

Table 1. Description for parameter.

Constants		Constants	
M	set of tasks， M={1,2,...,m}	N_1	a collection of nodes in the buffer area
V	set of AGVs	N	the set of all nodes in the path network
T	set of time, T = {T_1, T_2, ...}	N_i^c	set of nodes adjacent to i
T_n'	set of call time for task n, $T_n' \in \{T'_1, T'_2, ..., T'_m\}$	i, j, k	represent nodes, i, j, k∈N
n	a task, n∈M	(x_i, y_i)	the position coordinate of node i
α,β	α ≠ β, α， β∈V	(i_n, j_n)	the nodes of the starting and ending point, position of task n， $i_n, j_n \in N_0$
T_θ	$T_\theta \in T$	L	AGV's minimum turning radius
v	average speed of AGV	W_{ij}	number of turns between node i and nodej, where one node apart in i and j
N_0	loading and unloading nodes of AGVs		

2) Decision variables and intermediate variables

Table 2. Description for intermediate variable and decision variables.

Intermediate variables		Decision variables	
k_α	the position of AGVα, $k_\alpha \in N$	$X_{n\alpha}$	$X_{n\alpha}$=1， task n is assigned to AGVα. 0，others
d_{ij}	the distance from node i to node j		
t_{ij}	time spent from node i to node j		
Da_i	time of the AGV stays at node i	$Y_{\alpha T\theta}$	$Y_{\alpha T\theta}$=1, AGVα has a task at T_θ 0，others
da_i	time point of reaching node i， $da_i \in T$		
dd_i	time point of leaving node i， $dd_i \in T$		

2.3. Step-by-Step Model

Length of path, number of turns and consumption of time are the three key performance measures to judge the quality of planned path. This model subdivides the process of each task of the AGV operation into two stages: the first stage from the position of the AGV to the starting point of the task when the task is called at no load (referred to as the response stage); the second stage from the start to the end of the task during load operation (referred to as the task stage). Firstly, assigning a applicable AGV for the calling task. Then, planning the route in each stage with the consideration of the length and the number of turns so that to select the shortest path with the least number of turns as few as possible to form a collection of path alternative. Finally, calculating the operation time of each stage through conflict analysis, and choosing the path with the time as least as possible in the path alternatives.

$$F = \begin{cases} \min \sum_{\alpha \in V} X_{n\alpha} * d_{k_\alpha i_n} & (1) \\ \min \sum_{\alpha \in V} X_{n\alpha} * d_{i_n j_n} & (2) \\ \min \left(\sum_{\alpha \in V} \sum_{n \in M} X_{n\alpha} * t_{k_\alpha i_n} \right) / m & (3) \\ \min \left(\sum_{\alpha \in V} \sum_{n \in M} X_{n\alpha} * t_{i_n j_n} \right) / m & (4) \end{cases}$$

Equations (1) – (4) represent the objective function, which examines the path length and time consumption of each stage. Equations (1) – (2) mean that from the current position to the task position and from the task starting point to the ending point, the path length of the assigned AGV is as short as possible for each task. Equations (3) – (4) examine the time consumed and indicate the average response time of the task and the average job task time should be as little as possible.

$$\sum_{\alpha \in V} X_{n\alpha} = 1, \forall n \in M \quad (5)$$

$$\sum_{\forall n \in M} Y_\alpha T_\theta * X_{n\alpha} = 1, \forall T_\theta \in T, \forall \alpha \in V \quad (6)$$

$$\sum_{\alpha \in V} \sum_{n \in M} X_{n\alpha} = m \quad (7)$$

Equations (5) – (7) are some basic constraints of the AGV scheduling system. Equation (5) guarantees that a task has only one AGV to operate; Equation (6) guarantees that an AGV can process at most one task one time; Equation (7) guarantees that all tasks in task set M are completed.

$$\sum_{\alpha \in V} X_{n\alpha} * Y_{\alpha T_n} * \left| x_{k_\alpha} - x_{i_n} \right| \geq L, \forall n \in M \quad (8)$$

Equation (8) is the horizontal distance constraint for AGV dispatching. As is shown in Figure 3, the position of the red dot is where the task is calling, and the turning radius of the AGV should be considered in the actual dispatching. That means the AGV in the red area in the figure does not meet the above requirements.

$$d_{ij} = \left| x_i - x_j \right| + \left| y_i - y_j \right|, j \in N_i^c, i, j \in N \quad (9)$$

$$\begin{cases} d_{ij} = d_{i(i+1)} + d_{i(i+1)(i+2)} + \ldots + d_{(j-1)j}, j \notin N_i^c, i, j \in N \\ \ln(i, i+1, i+2, \ldots, j-1, j), i+1 \in N_i^c, i+2 \in N_{i+1}^c, \ldots, j \in N_{j-1}^c \end{cases} \quad (10)$$

Equations (9) – (10) represent the calculation method of the distance cost between two points by using Manhattan distance. Equation (9) is the calculation method when i and j are directly connected; equation (10) sets the distance cost between two points when i and j are not directly connected. Finding out the middle point to form a set (i, i1, i2, i3,…, j), where the nodes in the set are connected from left to right in sequence. The selection of the connected path is based on the selection of the set Ni^c of

adjacent points starting from i, and the search for adjacent points continued until the ending point j is reached. It can be seen from the objective functions (1) – (2) that the connected path should be as short as possible. Searching for the connected path for one more time to find out short paths as many as possible for AGV$_\alpha$ dispatched to task n, which constitutes an alternative path scheme.

$$x_i - x_j \neq 0, y_i - y_j \neq 0, k \in N_k^c, i, k, j \in N \tag{11}$$

$$\min \sum_i^{j-2} W_{k(k+2)}, k \in (i, i+1, i+2, i = 3, ..., j-2) \tag{12}$$

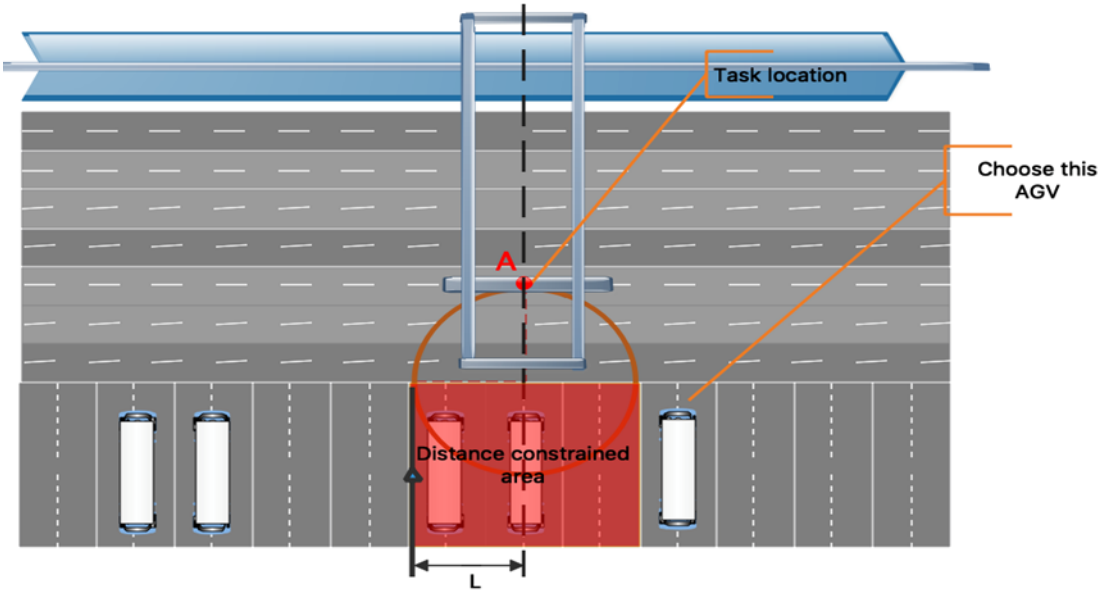

Figure 3. Horizontal distance constraint area for scheduling.

If the three points (i, k, j) is satisfied with the expression (11), it means that a turn occurs at node k from node i to node j via node k, and it is recorded as $W_{ij} = 1$, otherwise it is 0. In the above alternative routes, the number of turns in the route is calculated according to equation (12), and the alternative route is further screened to select fewer turns.

$$t_{ij} = 1 + D\alpha_{ij}, j \in N_i^c, \ \ i, j \in N \tag{13}$$

$$\begin{cases} t_{ij} = t_{i(i+1)} + t_{i(i+1)(i+2)} + ... + t_{(j-1)j}, j \notin N_i^c, i, j \in N \\ \ln(i, i+1, i+2, ..., j-1, j), i+1 \in N_i^c, i+2 \in N_{i+1}^c, ..., j \in N_{j-1}^c \end{cases} \tag{14}$$

Equations (13) – (14) represent the calculation method of the time cost between two points. Equation (13) sets when the node i is connected to the node j, the time required for the AGV is 1 unit of time plus the waiting time that may occur in the process; equation (14) sets the time when the node i and j is not directly connected, it is calculated after forming a connected path by searching for an intermediate connection point.

Table 3. Table for path and time by AGVα.

Time	Task number	Current point	Next point	Waiting time
1		805	(Null)	(Null)
2	29	320	319	(Null)
3	29	319	419	(Null)
4	29	419	581	(Null)
5	29	581	819	(Null)
...	...	...	...	...

The above alternative path meets the shortest possible path and the minimum number of turns. Based on this, the time point when the AGV reaches each node in the path is calculated for the alternative path according to the principles of equations (11) – (12). In the total task, the initial task fails to consider the conflict handling time, and just chooses the shortest path as the final route, and the time of the final path as well as the corresponding node is saved for judging the conflict of the subsequent AGV path (Table. 3). If there is an ongoing task before the task assigned by the AGV, then the calculation of the path time needs to consider the possibility of conflict with other AGVs in the environment.

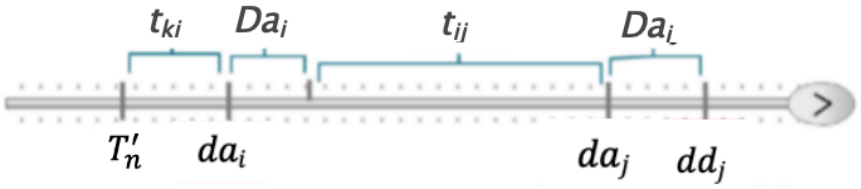

Figure 4. Time division in AGV tasks.

$$d\alpha_k = T_n^{'} + t_{ki} \tag{15}$$

$$d\alpha_j = d\alpha_i + D\alpha_i + t_{ij}, D\alpha_k \in N, D\alpha_k \in (60,100) \tag{16}$$

$$dd_j = d\alpha_j + D\alpha_j, D\alpha_j \in N, D\alpha_k \in (60,100) \tag{17}$$

As is shown in Figure 4, the time axis of the task AGV$_\alpha$ is set since the task call according to equations (15) – (17)starts. When task n is assigned to AGV$_\alpha$, the node where the cart is located, the task starting point and task ending point are i, k, j, respectively. Equation (15) determines the time point of AGV$_\alpha$ from point i to point k; Equation (16) determines the time point that the trolley reaches the end of the task, which includes the time for the waiting equipment at the start of the task to put the box and the time from k to j of the road segment; Equation (17) determines the time when the trolley leaves the ending point j, including the time when the equipment extracts the container from the trolley. The determinations of these time points are very important for the matching of the time and position of the AGV.

3. Solution approach

By analyzing the characteristics and functional requirements of the AGV system scheduling model, the above model can be transformed into independent optimization problems of multiple subsystems. And comprehensive coordination between the systems can be completed based on this. Firstly, the AGV scheduling system is decomposed into three functional subsystems: task scheduling, path planning, and pre-collision analysis and processing. Three-stage control strategy is adopted, namely task allocation strategy, path planning method library based on A* algorithm, and time window-based re-adjust the path after conflict prediction. At the same time, the time-based heuristic path algorithm greatly improves the flexibility of path planning and adjustment so that the system can information in real time and make corresponding adjustments. The solution framework of the three-stage scheduling is shown in Figure 5.

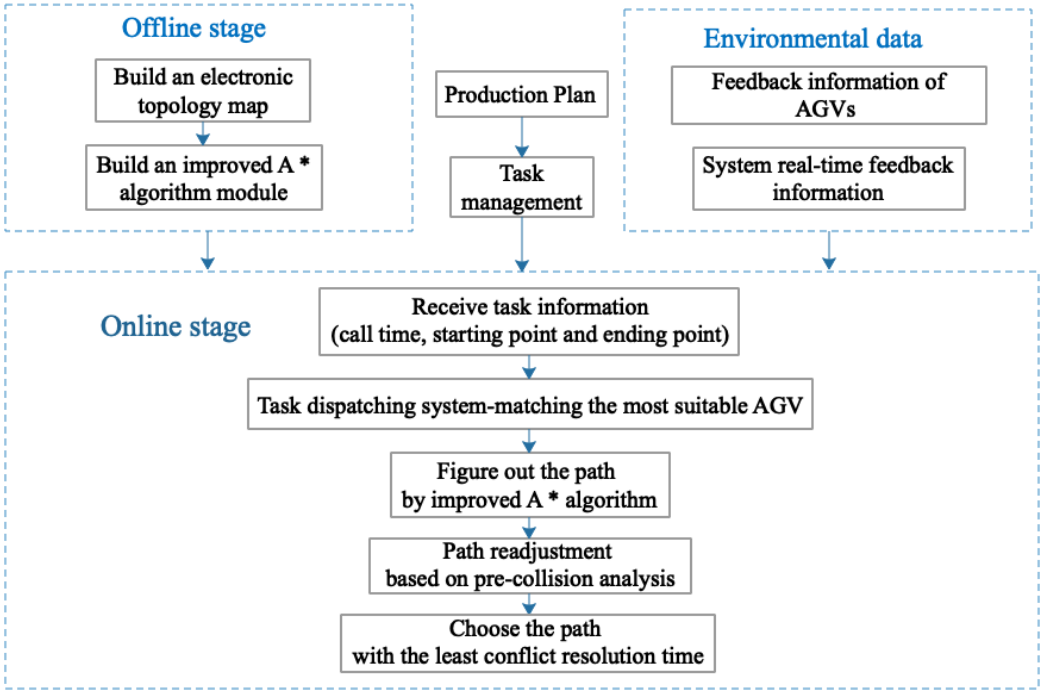

Figure 5. Three-stage scheduling system solution.

The core content of the algorithmic solution of the model includes:

(1) This paper improves the search rules of the traditional A* algorithm to make the search results more flexible. Multiple shortest paths are generated based on the topological graph, where paths with fewer number of turns are selected as alternative paths.

(2) In the process of task assignment, according to the above path planning idea, calculating the time that each AGV arrives at the starting of the task and choosing the AGV that can respond the fastest.

(3) According to the task call time and the AGV speed, the equations (15)-(17) are combined to predict the time when the AGV arrives at each node in the paths generated by the path planning algorithm. Comparing the node time of this AGV with other AGVs which have a determined path and readjusting the path after conflict prediction.

3.1. Analysis of path algorithm

3.1.1. Map modeling

The topological map represents the current environmental information with a series of points and connecting lines, and finally shows the connectivity of an environment through a graph. The topology structure based on environmental information is not only convenient for modeling, but also has lower storage space and calculation time requirements, which can effectively maintain the real-time nature of the computer. Therefore, this paper chooses the topological map as the environment modeling method in order to improve system efficiency.

At present, the storage methods of environmental information mainly include adjacency matrix and adjacency table and so forth. In this paper, the adjacency matrix method is adopted, which uses a one-dimensional array to store node information and a two-dimensional array to store edge information. As is shown in Figure 7, in a directed connected network G = (V, E), V represents a set of nodes including the two-dimension coordinates of the location, and E represents a set of edges connecting the points in V. Each edge can be expressed as an ordered pair of two nodes [19]. Each side of it has a weight value, and in this paper, the weight value is set as the Manhattan distance between two nodes. In this way, a digital topological map containing a set of points and edges can be expressed, and during the running of program, the N_i^c of each node is easy to get. When describing the running path of a vehicle, it can be represented by an ordered set of nodes, and the order of the nodes indicates the running path of the AGV. According to the road distribution map of the horizontal area of the automated terminal (Figure 2), a topology map is drawn, which was shown in Figure 6. Additionally, before path planning, the environment information is received to judge the obstacles in the current environment by which the map is adjusted. This improves the real-time and dynamic nature of path planning, which can also cope with roadblocks caused by some unexpected situations in the system.

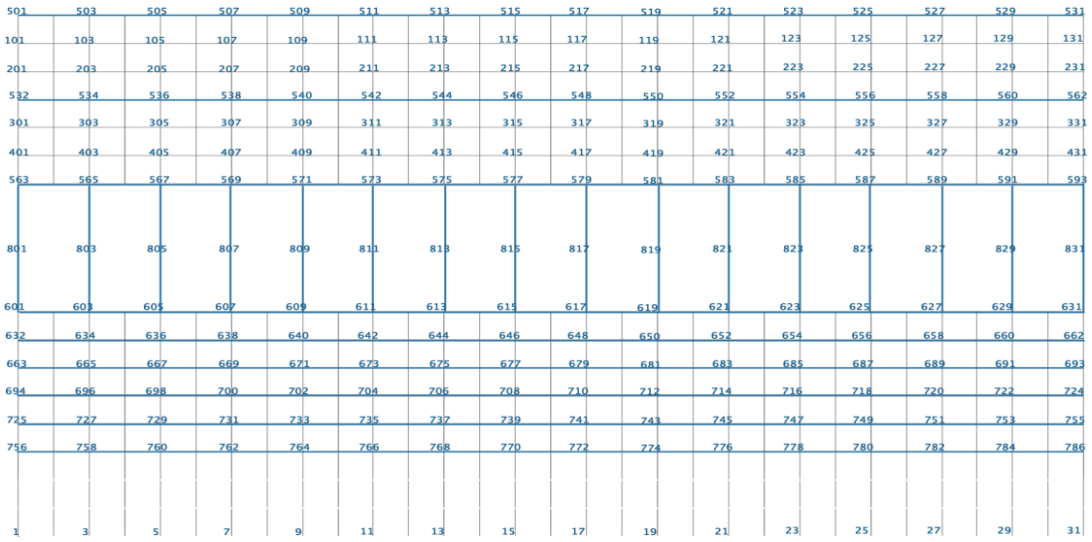

Figure 6. Topology diagram of AGV horizontal transportation path planning based on improved A* algorithm.

3.1.2. Path planning based on improved A* algorithm

The A* algorithm is a heuristic search algorithm, which uses a heuristic function to evaluate the cost of a node (described as the current node) to the target node on the way, and then selects the node with the maximum or minimum cost as the starting point for the next search according to the needs until the target node is found. Since the topological graph extends the child nodes of the current point in up to four directions, this paper uses a heuristic function based on the Manhattan distance to calculate the cost of the node and the target point. In the horizontal transportation of ACTs, due to the large number of AGVs and the sharing of route resources, applying such traditional A* algorithm directly in such a complex environment may result in many conflicts between AGVs and excessive turns. This paper draws on the heuristic function of the A* algorithm and the directionality of the search to eliminate the drawbacks of the traditional A* algorithm, and proposes an improved A* algorithm.

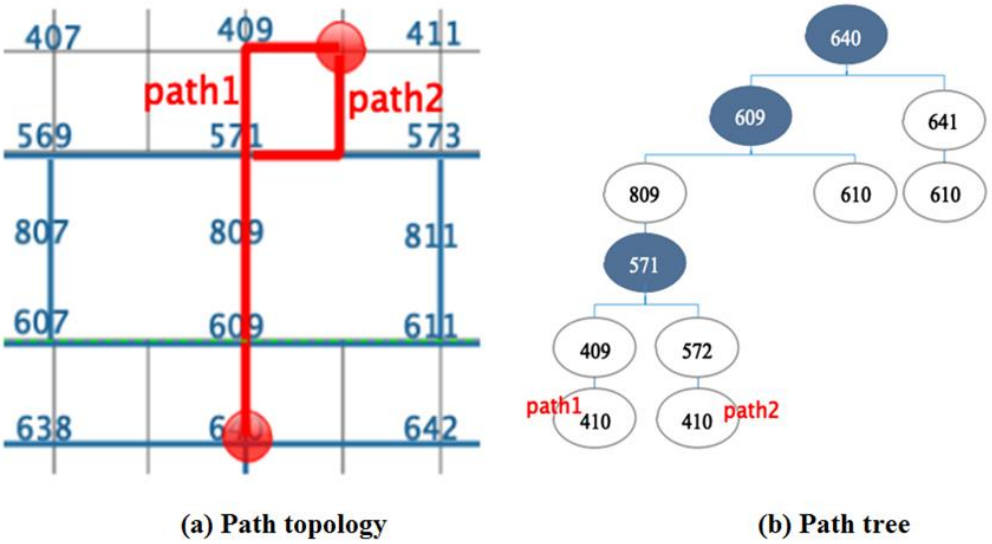

Figure 7. Tree_node collection example.

The algorithm flow is shown in Figure 8, where the current point is called the parent node, and the adjacent point of the parent node is called the child node; v0 represents the current point (also called the parent node), and the starting point and ending point of the task are s_1 and s_t respectively. The set of path records every point selected until the end point is selected, and the points in the path constitute a path from the starting point to the ending point in sequence. In the algorithm flow chart, the tree_node set is used to store bifurcated nodes and their branch nodes in the planned path. Taking Figure 7 as an example, the starting point and the end point are 640 and 410 marked in red in the Figure 7 (a) and Figure 7 (b) is a path tree diagram based on the direction principle of the heuristic function in the A * algorithm. 640, 609 and 571 are all bifurcated nodes, then tree_node is presented in the form of {640: {609, 641}, 609: {809, 610}, 571: {409, 572}}. In the actual program, in order to improve the efficiency of path searching and reduce the repeated calculation of proven invalid nodes, the branch point in tree_node will be pruned in real time. For example, the 641 node is cut out at the bifurcated point of 640 in tree_node, which is not described in detail in Figure 8.

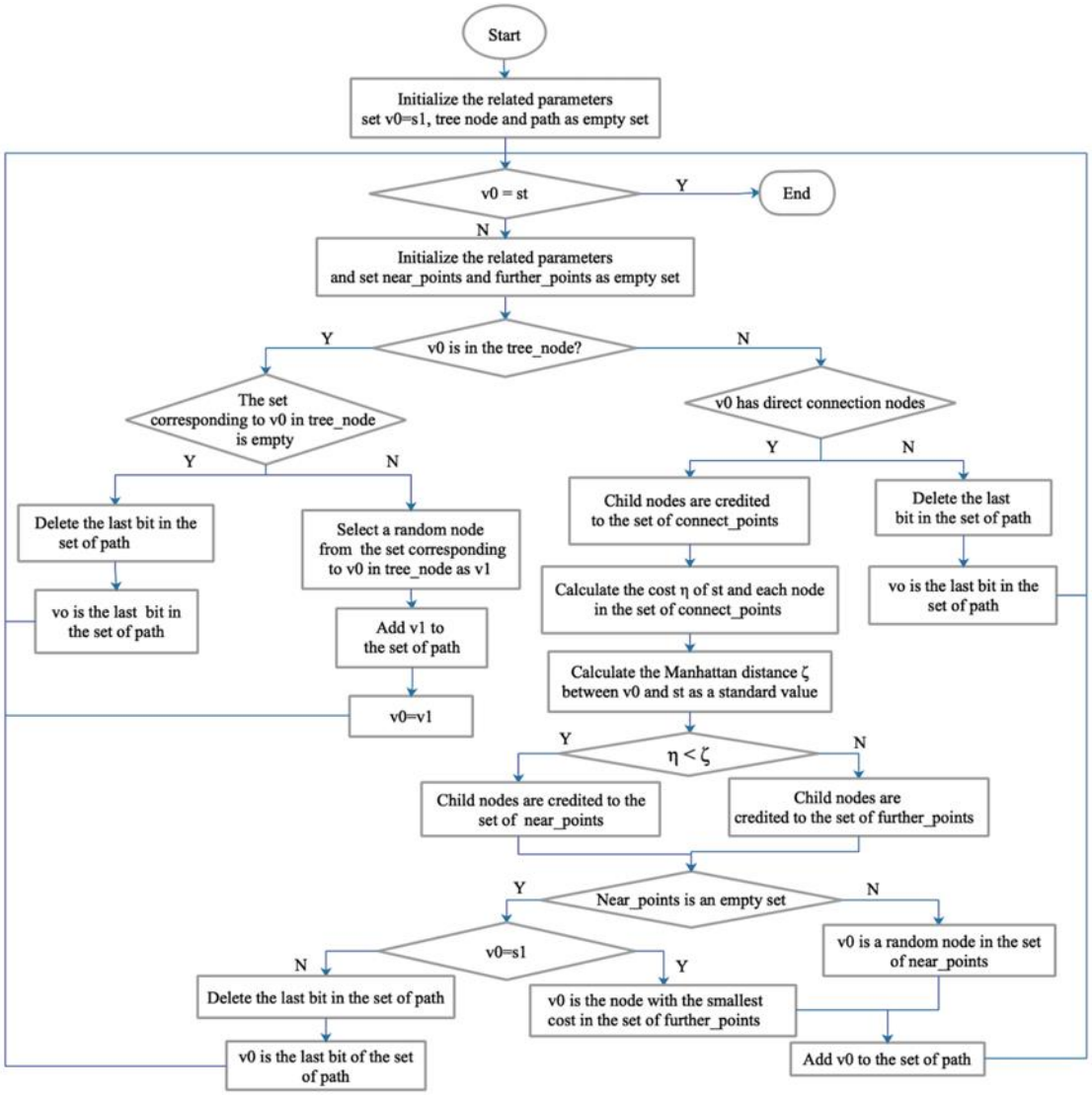

Figure 8. Improved A * algorithm.

Step1: Initialize related parameters

Initially empty: v0, the set of tree_node, the set of path []. Skip to Step2.

Step2: Initialize related parameters

Initially empty: the set of near points, which stores the points that connected with v0 and meet the directional principle; the set of further points storing the points that connected with v0 and do not meet the directional principle. Skip to Step3.

Step3: Expand child nodes

If v0 is already stored in tree_node and the set corresponding to v0 is not empty, any point in the set of parent nodes in tree_node is directly taken as the next starting point for path planning, update the node to v0, and skip to Step6; if v0 of the corresponding set is empty, the current point is not suitable, update the previous point of the parent node in the path to v0, delete the end from the path, and restart Step3.

If there is no v0 in tree_node, v0 has directly connected child nodes, firstly calculating the cost between v0 and the target node (as shown in equation (3.1)) as a standard value ζ, and then calculating the cost of all the child nodes to the target node st (shown in equation (3.1)), skip to Step4; if there is no child node, delete v0 in the path, update the last node in the path to v0, and restart Step3.

Step4: Compare and determine near points collection

Comparing the costs corresponding to all child nodes of v0 with ζ, the generation value greater than the standard value (indicating that the child node is farther from the parent node than the parent node) is filled in the further_points set, and the generation value is less than the standard value ζ (The child node is closer to the target point than the parent node) is filled into the near_point set. Skip to Step5.

Step5: Choose

When the near_points set is not empty, any point is v0; when the near_point set is empty, it means that v0 has no child nodes closer to the target point than it is. If v0 is already the starting point, select the point with the smallest value in the further_point set, add the point to the path and update it to v0; if v0 is not the starting point, delete v0 from the path, and then assign the end to v0. Skip to Step6.

Step6: Repeat the process from step3 to step5 until the target node is searched.

Step7: Repeat the process from step3 to step6, search for multiple routes, calculate the number of turns for each route, select the route with the smallest number of turns as the final search route, and the other route with the smallest number of turns as the alternative route.

3.2. Conflict strategy

In actual wharf operations, putting aside the consideration of the conflict problem to study the path planning problem of AGV is not enough to meet the actual operational needs. Conflict analysis mainly includes conflict prediction and conflict handling. Conflict prediction includes: allocating and comparing path node time to find conflicts and determine the types of conflicts. The node time of the alternative path is calculated according to the task call time and AGV speed. Based on real-time environmental information in the calculation of the time point, making the computer quickly simulate the planned path again and calculating the time cost of each path in the alternative path. By comparing each step with other AGVs with a determined path, it is determined whether the AGV will conflict with other AGVs at the next node. If no conflict occurs, the time point when the AGV reaches the next node is recorded according to equations (14) and (16). If conflicts occur, conflict handling is required.

As is shown in Figure 9, the five major types of conflicts include chasing conflicts, intersection conflicts, opposite conflicts, path conflicts, and deadlock issues. The different handling methods for the five major conflicts in terminal horizontal transportation are shown in Table 4. After the conflict prediction for the alternative path is completed, the path with the smallest conflict processing time is selected as the final planned path.

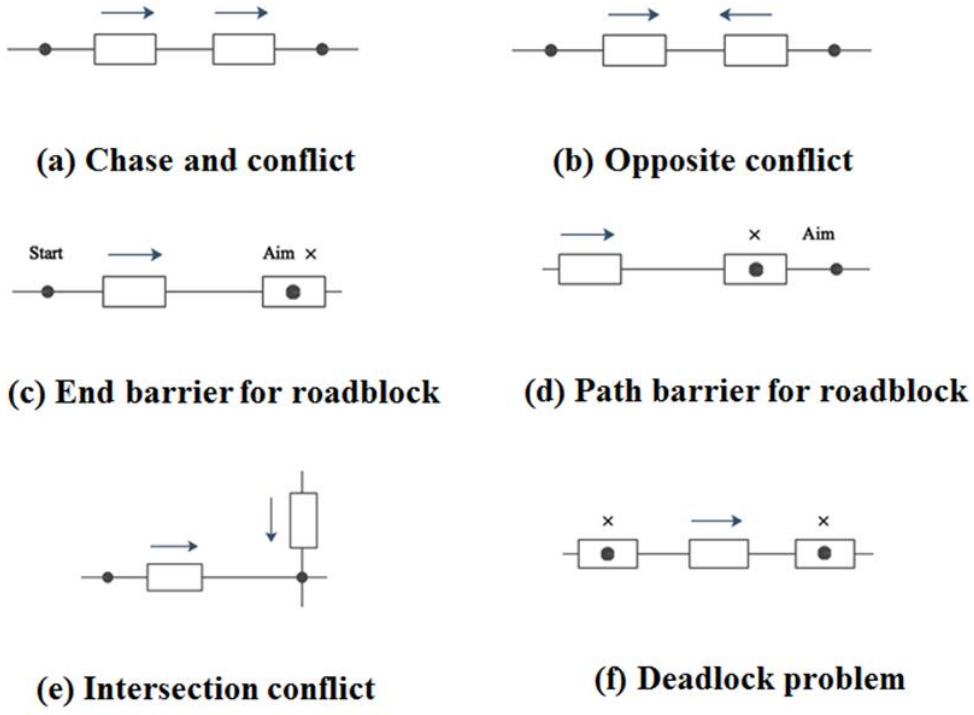

Figure 9. Classification of conflicts.

Table 4. Table of classification and handling of conflict.

Classification	Description	Solution
Chase and conflict	The two AGVs traveled in the same direction, and the backward AGV overtaken due to the speed difference.	Eliminate the occurrence of such conflicts by assuming a constant speed
Opposite conflict	The two AGVs are traveling in opposite directions on the same road.	Adjust the path to avoid obstacles
End barrier for roadblock	There are other AGVs docked at the end of the running AGV, hindering the progress of AGVs.	Wait to avoid obstacles
Path barrier for roadblock	There are other AGVs docked on the AGV's path, hindering the progress of AGVs.	Adjust the path to avoid obstacles
Intersection conflict	The two AGVs crash when they compete for intersection resources.	Wait to avoid obstacles
Deadlock problem	It often occurs when the other problems mentioned above are not resolved.	After solving the above types of conflicts, it can be effectively avoided.

3.3. Task assignment

The module flow of task scheduling is shown in Figure 10. In actual task allocation, AGVs are divided into two categories: idle AGVs and AGVs that are performing tasks. The idle AGV can be subdivided into two categories, one is a car parked in the buffer area, and the other is a car that is just

returning to the buffer area after completing the task. AGVs parked in the buffer area firstly need the reachability of the car to satisfy the turning radius and the deceleration distance, which meets the constraint on the horizontal distance in Equation (8). On this basis, combined with 3.1 and 3.2 to calculate the AGV response time. For the AGV that is returning to the buffer area, it is necessary to directly plan the route to the starting point of the task with the location of the call time AGV as the starting point, and then calculating the calling time. For an AGV that is executing a task, its call time is the sum of the remaining time of the task and the time from the ending point of the previous task to the call point of the task. Finally, comparing the response time of each AGV to the task, and assigning the AGV that can match the task fastest to the task.

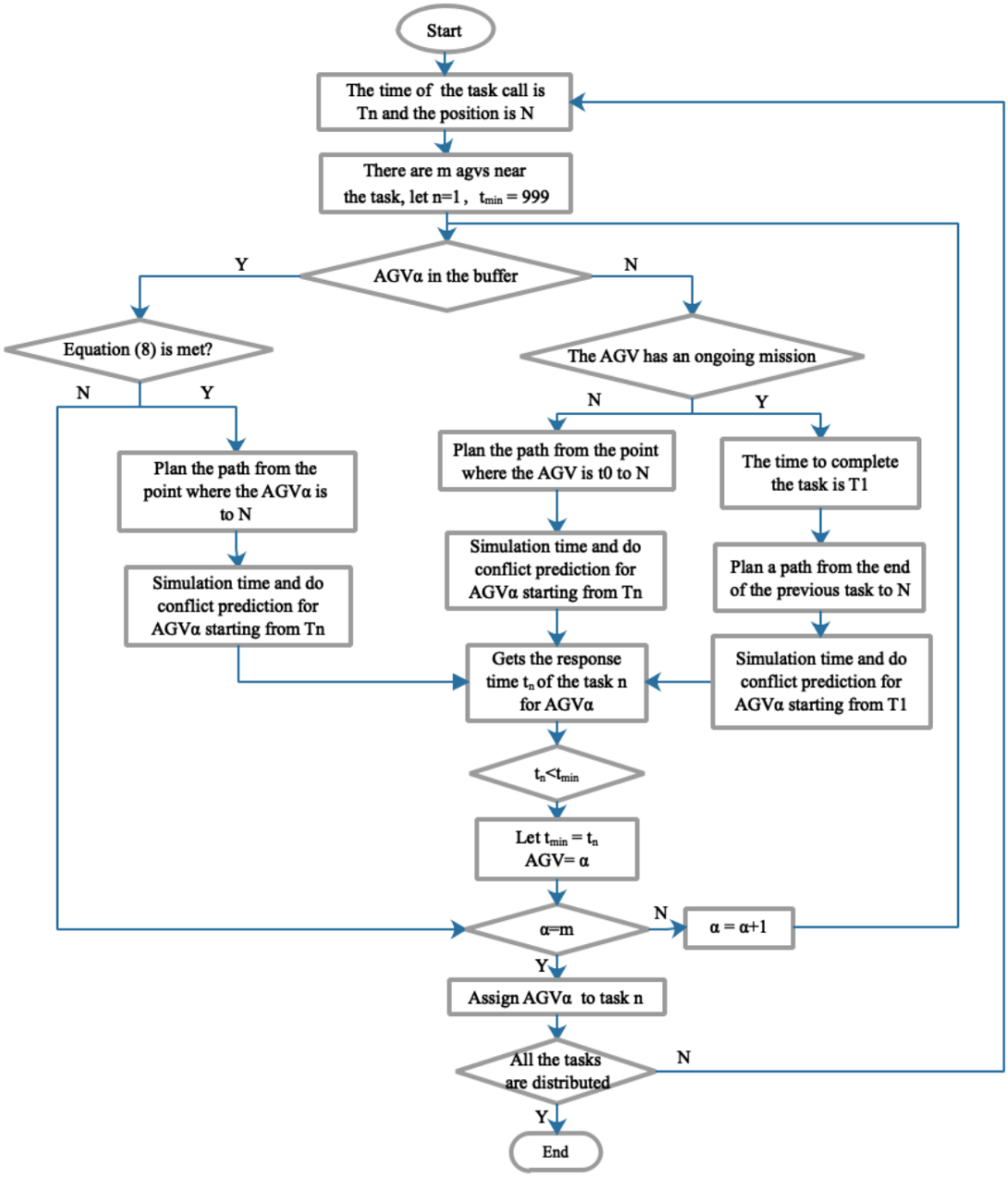

Figure 10. Flow chart of task allocation.

4. Analysis of examples

4.1. Analysis of path algorithm

This section sets two groups of starting points and ending points, and then generates obstacle points randomly to test the effectiveness and flexibility of the improved A* algorithm in 3.1. The effectiveness and flexibility of the improved A* algorithm is tested based on the indicators of the path length, number of turns and calculation efficiency. Improved A* algorithm and the original A* algorithm is used to obtain the results of two groups of path solutions shown in Fig11, and the three indicators are compared in Table 2. In order to further analyze the planning performance of this algorithm, the genetic algorithm [20] and Dijkstra [21] algorithm is used to plan the AGV path. The comparison data is shown in Table 3. The GA parameters include crossover rate pc: 0.9, mutation rate pm: 0.1, Ps: 150, and Mg: 50 [20].

It can be seen from the analysis in Table 5 that by pruning invalid points to avoid repeating calculation, even if the improved A* algorithm runs multiple times (set to 150 times) to get more choices and select the best, its efficiency is still as high as the original A* algorithm. Actually, when the program is running, the original A* algorithm may fail to run successfully because the path reaches a dead end. The improved A * algorithm ensures the success of path planning through pruning. At the same time, the path length calculated by the improved A* algorithm is 7.3% less than the A * algorithm, and the number of turns is reduced by 57.2%, which means that, under the same environment, the improved A * algorithm has obvious advantages over the original A* algorithm in terms of length, number of turns and computational efficiency. Furthermore, Table 6 shows that the genetic algorithm is slightly higher than the improved A* algorithm in path length and number of turns, and the operation efficiency is much lower. The Dijkstra algorithm can get the shortest path and the operation efficiency is high, but it has more number of turns than the improved A * algorithm.

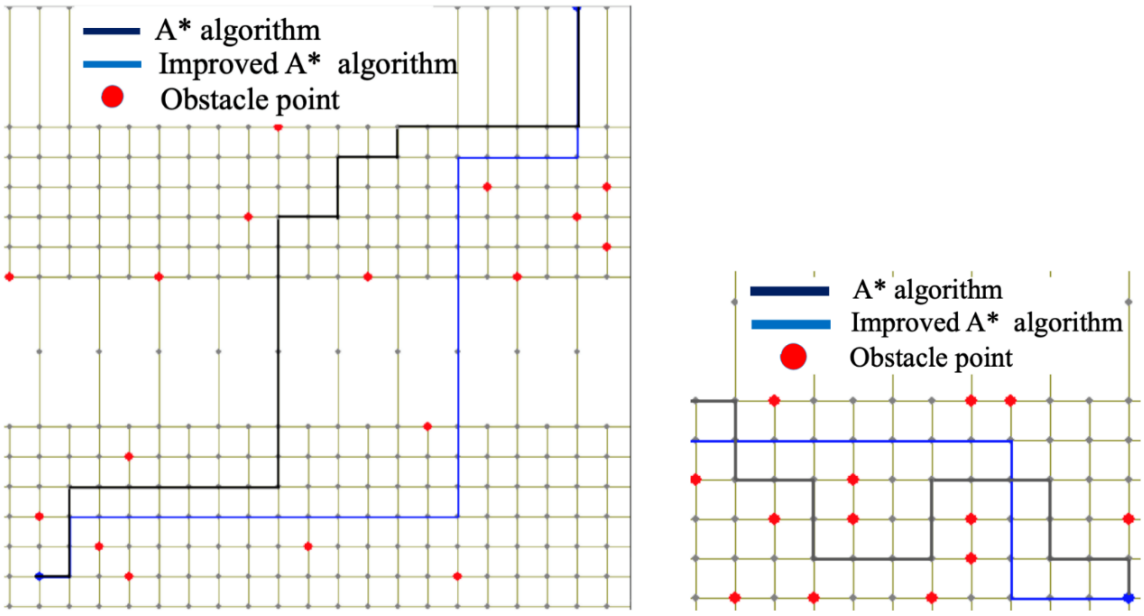

Figure 11. Schematic diagram of the path.

Table 5. Comparison between improved algorithm and original algorithm.

Algorithm	Start and end	Length/m	Number of turns	Operation time/s
A*	21→103	148	9	<0.5
	117→805	110	10	<0.5
Improved A*	21→103	148	5	<0.5
	117→805	94	3	<0.5
Average rate of change/%		-7.3%	-57.2%	\

Table 6. Comparison between the improved A * algorithm and other algorithms.

Algorithm	Start and end	Length/m	Number of turns	Operation time/s
Genetic algorithm	21→103	160	7	8.2
	117→805	94	8	6.3
Dijkstra	21→103	148	11	<0.5
	117→805	94	9	<0.5
Improved A*	21→103	148	5	<0.5
	117→805	94	3	<0.5

From the above analysis, it can be seen the path generated by the improved A* algorithm has the shortest length and a small number of turns while bypassing obstacles, which proves the effectiveness of the path planning algorithm in terms of length, path smoothness and operation efficiency. At the same time, each time before planning the route, it will first receive the environmental information of the system to survey obstacles, such as AGVs remaining in the operation area of QC and the buffer zone of AGV, sudden roadblocks, etc. Then, the map will be disconnected at nodes where the obstacle is located, which also enhances the dynamics of the traditional A * algorithm.

4.2. The solution of the example

In order to verify the feasibility of the model and algorithm proposed in this paper, numerical examples of different situations and scales are designed. The time when the quay cranes starts to work is different in distinct calculation examples, and the safe distance of one bay is stored between the adjacent quayside bridges. In this part, the experiments under 8 different parameter settings are considered respectively: Calculation examples 1 to 3 consider that the number of containers is 60 and the quay cranes is 3, where the AGV numbers are 8, 10, and 12, respectively; Cases 4 to 6 keep the same number of quay cranes and AGVs as Case 1, which increases the number of containers to 80, 120 and 180, respectively; Cases 7 and 8 keep the same number of quay crane and AGV as Case 1, which increases the number of unloading boxes to 250 and 300 respectively. The experimental results are shown in table 7 (Y means consider reducing conflict and N means not)， which can be seen that in the case of the same number of unloaded boxes, the greater number of AGVs, the less average completion time of the task. When the number of quay cranes, AGV and box area is the same, the more number of unloaded boxes, the longer total task will be completed.

Table 7. Results of cases.

Example number	Containers number	AQC-AGV number	Average response time(N/Y)	Average task time(N/Y)	Total collision time(N/Y)
1	60	3–8	15.6/15.2	15.0/14.0	6/2
2	60	3–10	14.0/13.9	15.0/14.0	10/4
3	60	3–12	13.6/13.2	15.0/14.0	16/8
4	80	3–10	15.6/15.3	15.1/15.1	25/6
5	120	3–10	17.1/16.8	15.3/15.0	65/31
6	180	3–10	18.3/17.6	15.0/14.8	127/61
7	250	3–10	16.7/16.2	14.9/14.6	169/72
8	300	3–10	16.4/15.8	14.9/14.7	220/90

In order to observe the advantages of the three-stage method more intuitively, the same unloading quantity, different AGV quantity, same AGV quantity and different unloading quantity are compared with and without considering the system conflict time.

It can be seen in Figure 12 that the average task completion time without considering the system conflict time is significantly higher than that of considering the system conflict time. When the number of quay cranes and AGV is fixed at 3 and 10, the comparison results with and without consideration of system conflict time when the number of containers is 60, 250 and 330 are shown in Figure 13. It can also be seen that the average task completion time without considering the system conflict time is more than that of considering the system conflict time.

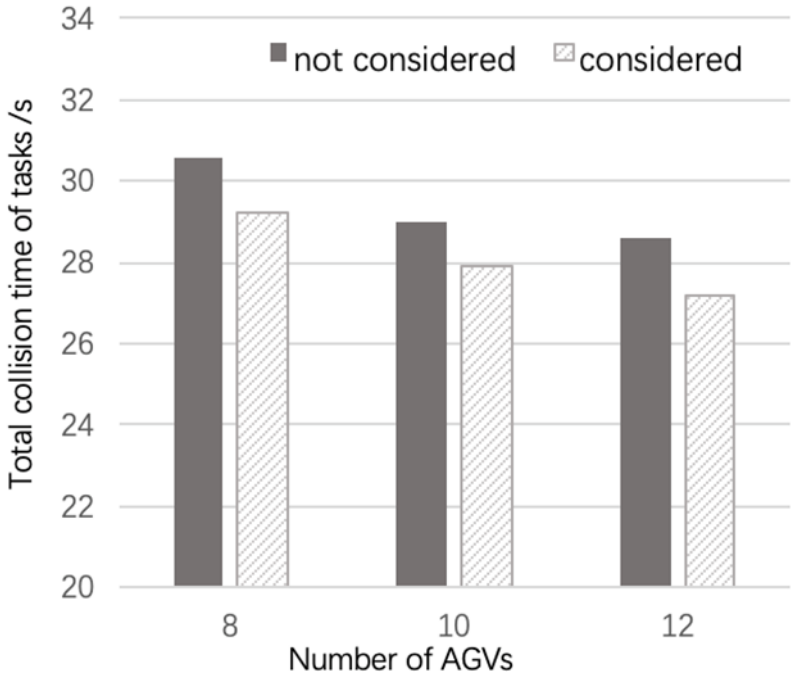

Figure 12. The same number of containers.

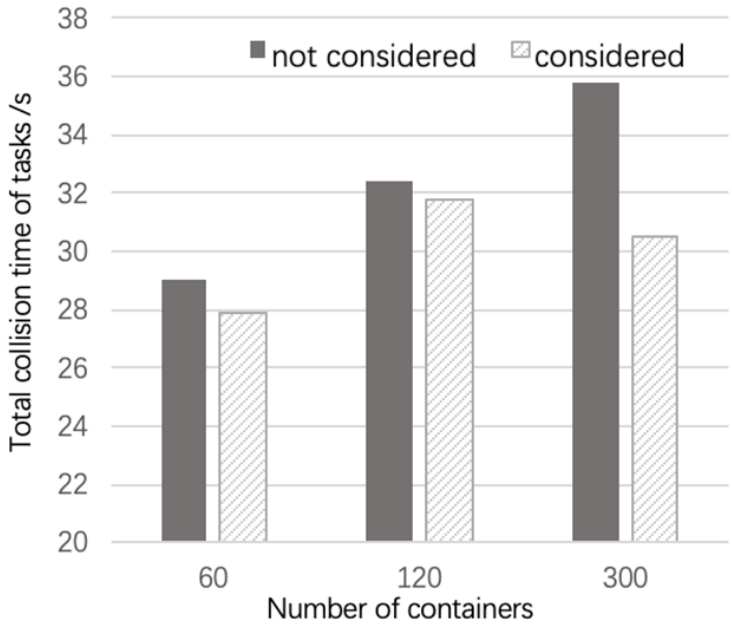

Figure 13. The same number of AGVs.

At the same time, Figure 14 shows the collision time diagram of different AGV quantities when the number of unloaded boxes is 60 and the number of quay cranes and containers is 3 and 6, respectively. Figure 15 shows the collision time diagram of different unloaded containers quantities when the number of quay cranes – AGV – container is 3-15-6. The experimental results show that when the number of boxes unloaded and AGV increases, the probability of collision is greater. After considering the collision avoidance, the waiting time of AGVs at the intersection is saved, and the scheduling result is more optimized.

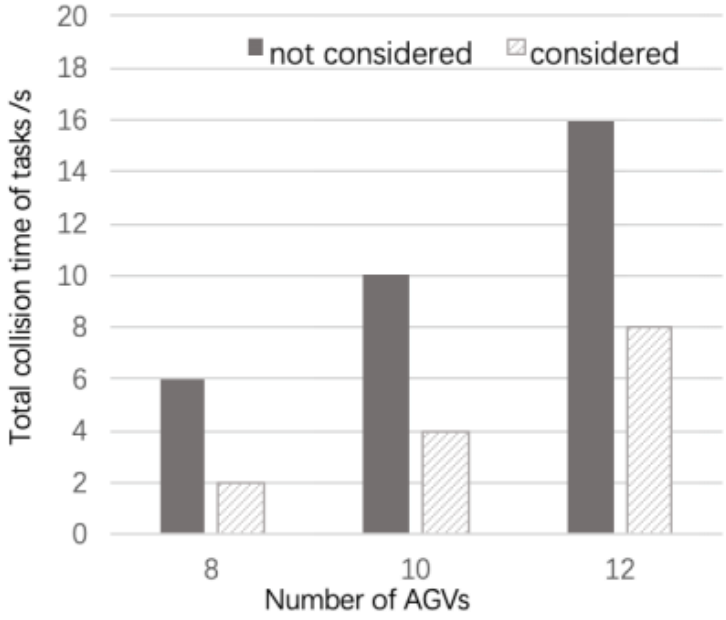

Figure 14. The same number of containers.

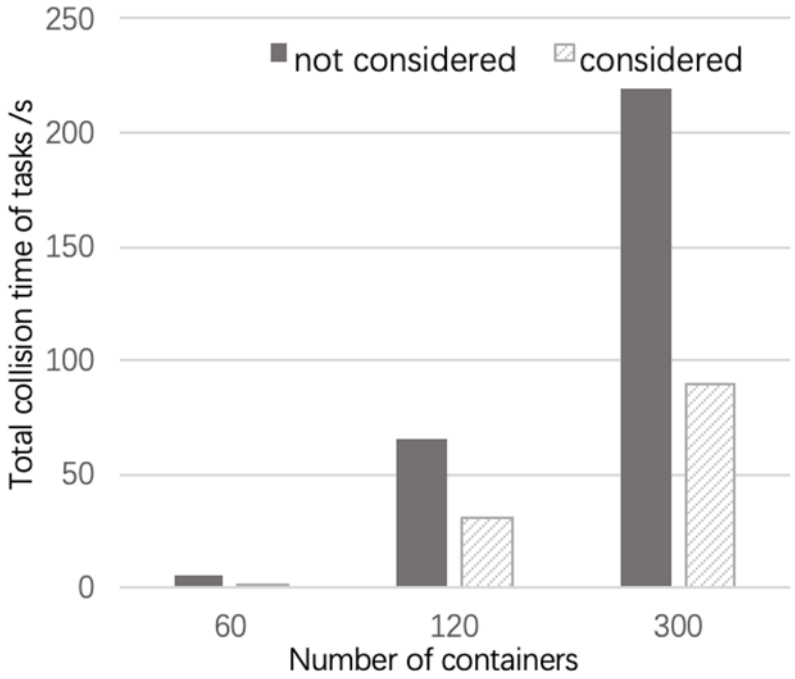

Figure 15. The same number of AGVs.

In the examples 8, the reduction rate of the average total task completion time after considering collision avoidance rules is shown in Table 8, which improves the efficiency of the whole unloading process. The results of the example show that the solution under collision avoidance rules is better than that under no collision avoidance rules.

Table 8. Reduction rate of total task completion time after considering conflict strategies.

Example number	containers number	AQC-AGV number	Average reduction rate of total task time /%
1	60	3–8	0.22
2	60	3–10	0.34
3	60	3–12	0.47
4	80	3–10	0.78
5	120	3–10	0.87
6	180	3–10	1.10
7	250	3–10	1.22
8	300	3–10	1.38

Through the analysis of the above cases, the conclusions are as follows:

1） When the number of AGV, quay cranes and containers are the same, the longer waiting time of AGV collision is, the higher probability of collision is, which means that the probability of conflict and the total task time both increases.

2） When the number of unloading boxes, the number of quay cranes and box areas are consistent, the number of AGVs increases, the longer waiting time for AGV collisions, the greater probability of collisions, and the total task time all decreases.

3） The collision time under the collision avoidance rule is obviously less than that without the collision avoidance rule

4） Considering that the total task completion time under collision avoidance rules is less than that under no collision avoidance rules.

5） The total dock operation efficiency under the consideration of collision avoidance rules is obviously higher than that without considering collision avoidance.

5. Conclusions and prospects

In present research, we successfully established a model for dispatching and routing the problems of AGV system in the horizontal transportation area of the automated container terminal, and proposed a three-stage solution to solve previously existed problems. Combining the improved A* algorithm with the conflict strategy based on the principle of time window, python was used to solve the examples of different scales initially and then the results were compared and analyzed. The experimental results showed that the improved A* algorithm for path planning was quite efficient. At the same time, after considering the strategy of AGV conflict, the average response time and completion time of the task were significantly reduced, which improved the efficiency of the horizontal area. Increasing the AGV's carrying capacity may be considered in future research so that the AGV can simultaneously carry two 20-feet containers or a 40-feet container, making it more in line with the actual situation of the container terminal.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. C. L. Liu, H. Jula, P. A. Loannou, Design, simulation, and evaluation of automated container terminals, *IEEE Trans. Intell. Transp. Syst.*, **3** (2002), 12–26.
2. H. T. Hu, X. Chen, T. Wang, Y. Zhang, A three-stage decomposition method for the joint vehicle dispatching and storage allocation problem in automated container terminals, *Comput. Ind. Eng.*, **129** (2019), 90–101.
3. H. Fazlollahtabar, M. Saidi-Mehrabad, Autonomous guided vehicles: Methods and models for optimal path planning, *Studies in Systems, Decision and Control,* **20** (2015).
4. M. S. Zhong, Y. S. Yang, Y. M. Zhou, Free-conflict AGV path planning in automated terminals based on speed control, *Comp. Sci.*, 2019. DOI: 10.11896/j.issn.1002-137x.2019.07.047.
5. I. Draganjac, T. Petrovic, D. Miklic, Z. Kovacic, J. Orsulic, Highly-scalable traffic management of autonomous industrial transportation systems, *Robot Cim-Int. Manuf.*, **63** (2020), 101915.
6. J. B. Xin, R. R. Negenborn, F. Corman, G. Lodewijks, Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance, *Transp. Res. Part C Emerg. Technol.*, **60** (2015), 377–396.
7. E. Gawrilow, M. Klimm, Conflict-free vehicle routing, *Euro. J. Transp. Logist.*, **1** (2012), 87–111.
8. C. Oboth, R. Batta, M. Karwan, Dynamic conflict-free routing of automated guided vehicles, *Int. J. Prod. Res.*, **37** (1999), 2003–2030.

9. T. J. Chen, Y. Sun, W. Dai, On the Shortest and Conflict-Free Path Planning of Multi-AGV System Based on Dijkstra Algorithm and the Dynamic Time-Window Method, *Adv. Mater. Res.*, **645** (2013), 267–271.
10. L. H. He, P. H. Lou, X. M. Qian, Conflict-free automated guided vehicles routing based on time window, *Comput. Integr. Manuf.*, **16** (2010), 2630–2634.
11. X. Y. Ma, Y. M. Bian, F. Gao, An improved shuffled frog leaping algorithm for multiload AGV dispatching in automated container terminals, *Math. Probl. Eng.*, **2020** (2020), 1–13.
12. P. E. Hart, N. J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, *IEEE Trans. Syst. enc. Cybern.*, **4** (1972), 28–29.
13. K. Daniel, A. Nash, S. Koenig, A. Felner: Any-angle path planning on grids, *J. Artif. Intell. Res.*, **39** (2010), 533–579.
14. Y. Xin, H. Liang, M. Du, An improved A* algorithm for searching infinite neighbourhoods, *Robot*, 2014. DOI: 10.13973/j.cnki.robot.2014.0627.
15. D. Frantiek, A. Babinec, M. Kajan, Path planning with modified a Star Algorithm for a mobile robot, *Proced. Eng.*, **96** (2014), 59–69.
16. M. X. Lin, K. Yuan, C. Shi, Y. Wang, Path planning of mobile robot based on improved A∗ algorithm, *IEEE Control Syst.*, DOI: 2017. 10.1109/CCDC.2017.7979125.
17. U. A. Umar, M. K. A. Ariffin, N. Ismail, Priority-based genetic algorithm for conflict-free automated guided vehicle routing, *Proced. Eng.*, **50** (2012), DOI: 10.1016/j.proeng.2012.10.080.
18. J. Luo, Y. Wu, Modelling of dual-cycle strategy for container storage and vehicle scheduling problems at automated container terminals, *Transp. Res. Part E.*, **79** (2015), 49–64.
19. S. Maza, P. Castagna, P, Conflict-free AGV routing in bi-directional network. Emerging Technologies and Factory Automation, *IEEE Symp. Emerg. Technol. Fact. Autom. ETFA*, DOI: 2001. 10.1109/ETFA.2001.997777.
20. M. S. Zhong, Y. S. Yang, Multi-AGV scheduling for conflict-free path planning in automated container terminals, *Comput. Ind. Eng.*, **142** (2020), 106371.
21. G. Qing, Z. Zheng, X. Yue, Path-planning of automated guided vehicle based on improved Dijkstra algorithm, *Control Decision Conference, IEEE*, **7** (2017), 138–143.

AIMS Press