

A hybrid algorithm for routing optimization of AGVs with multi-task assignment

Xiaolei Liang^{1,3*}, Yuan Sun^{1,3} and Xiao He²

¹ School of Automobile and Traffic Engineering, Wuhan University of Science and Technology, Wuhan, Hubei, 430081, China

² Wuhan Zhong Yuan Electronics Group CO., LTD, Wuhan, Hubei, 430081, China

³ Institute of Intelligent Driving and Smart Transportation, Wuhan University of Science and Technology, Wuhan, Hubei, 430081, China

*Corresponding author's e-mail: liangxiaolei@wust.edu.cn

Abstract. To solve the multi-task assignment and routing optimizing problem of AGVs, this paper proposed a hybrid algorithm. Based on analysis of a practice case, it divided the working process of AGVs into two stages: loading and unloading. In the first stage, the shortest paths between loading site and unloading site were obtained via Dijkstra method. Then in next stage, a pre-processing method was applied to simplify the path network. Then the unloading travel distance was optimized by Hungarian algorithm. Through the two-section processing method, the optimal of routing design could be obtained. At last, the result of a practice case showed that the method could be used to solve the routing optimization of AGVs with multi-task assignment problems.

1. Introduction

Automated Guided Vehicle (AGV) is a kind of automatic transportation equipment, which is a branch of mobile robot science. With excellent performance on mobility, flexibility and intelligence, it can dock manufacture system easily to fulfil the need of material transportation. Thus, AGV has been applied to heavy industry, manufacture, terminal, *et. al.*, and has very bright developing prospects. Nowadays, many researchers have focused on the development, routing design, assignment, scheduling, and so on.

Guan *et al.* [1] proposed a variable neighborhood niche genetic algorithm (VNS/NGA) by studying and analysing the existing path design methods. Xia *et al.* [2] built a model based on the viewable link map in a static environment, and used the GA to carry out path planning and improvement on AGV, which can quickly search the optimal path of AGV. Xiao *et al.* [3] adopted a multi-attribute task scheduling principle, considered the traffic subsystem and the processing subsystem separately, and proposed a strategy of avoiding deadlock. In order to minimize the total distance of no load and combine the characteristics of flexible manufacturing system (FMS), Fazlollahatabar H *et al.* [4] considered time window constraints and AGV loading, and provided a genetic algorithm to solve AGV path and scheduling problems. Song *et al.* [5] put forward an optimization plan for the distribution system by using industrial engineering method and simulated the logistics distribution system of the engine assembly line. Based on the characteristics of simulation analysis and logistics distribution system, Wang *et al.* [6], combined with heuristic algorithm to obtain a solution that is suitable for the actual needs of the assembly line, at the same time, they proposed a vehicle dynamic scheduling method through adjusting the control strategy and dynamic simulation of decision parameters.



Through AGVs system is flexible and organizational, some important problems like: hardly routing optimization and high no-load ratio still effect the application of AGVs. In modern logistics system, it very common that many transportation requirements occur simultaneously and each AGV need cooperate with others so as to completing the tasks. For example, in a common situation of a factory, several machine tools need different materials simultaneously. But with the capacity limits of AGV and incompatibility of materials, we cannot use one AGV to complete the task. Several AGV are required to work together. Thus, under this situation of multiple tasks occurring, how to make a reasonable schedule of AGVs to short their transportation time and improve the working efficiency is an important problem which also is the aim of this work.

This paper proposes a hybrid algorithm to solve multiple tasks assign and rout optimization problems of a group of AGVs with the aim to improve their utilization.

2. Problem assumption and analysis

2.1. Assumption

The real AGV schedule system is very complex, thus we do some assumptions as follows:

- After unloading, materials enter into the input buffer of a workstation and wait for the processing of the machine; and enter the output buffer of the workstation after finishing each processing procedure.
- The materials in the input buffer are waiting for processing under the rule of FIFO, and the materials in the output buffer can leave the workstation in any order.
- AGV can only load one unit of material at a time, and moves uniformly along a planned path.
- Regardless of loading and unloading time; if multiple AGVs are at the same point, they must queue up.
- AGVs can automatically avoid obstacles during operation.
- Idle AGV stays at an unloading point, which will not affect other AGVs.

2.2. Analysis

In the automatic guided vehicle system, it is assumed that the AGV will park at the unloading point after finishing task. Considering the dynamic change of the position of the idle AGVs at each time, and the number of tasks and idle AGVs are not equal often. Therefore, we divide the AGV scheduling process into two stages. In the first stage, the tasks are assigned to the idle AGVs closest to the task according to the unloading point location D_s and the loading point location P_i of the transport tasks, and the AGVs no-load running path is designed to make the no-load distance shortest; in another stage, according to the task information from the loading point P_i to the unloading point D_j , we design the load running path of AGV to ensure the total load distance to be the shortest.

For the problem of AGV path network design, it is generally assumed that the system path network is given but the direction is unknown. It means that the network is undirected. In this paper, the path of AGV is designed based on the network shown in figure 1[1]. There are four workstations and nine intermediate nodes **M**. each workstation has one loading point **P** and one unloading point **D**. The number values on the sides of each connection indicates the distance between two nodes.

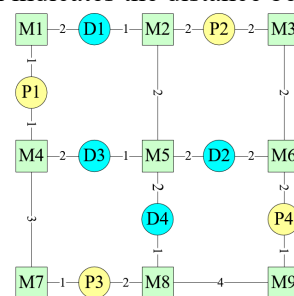


Figure 1. Undirect network.

The load flow between workstations is shown in table 1 below. Each non-zero number represents a task. There are 6 tasks to be assigned at this time. For example, 835 means transporting 835 units of material from loading point P_2 to unloading point D_1 .

Table 1. Load flow between workstations.

No.	1	2	3	4
1	0	0	777	0
2	835	0	0	545
3	0	780	0	558
4	389	0	0	0

Since the loading points and unloading points are given by the tasks, the shortest path of each task can be obtained by Dijkstra method. But in the no-load stage, the parking location and number of idle AGV are constantly changing. In this paper, we improve the efficiency of the handling system by optimizing the path of the no-load stage.

3. Model

For the path network $G = (V, E)$, where the node and edge set is V and E respectively, and direction of the initial edge is uncertain. Suppose the manufacturing system has n processing site, forming a set W . Each processing site has a loading point $W_i \in P$ and an unloading point $W_j \in D$, where P and D is the loading point set and the unloading point set respectively. The load flow matrix between the workstations is f . the parameter and variables are listed as follows:

L_{ij} : the shortest path length from task load node i to unload node j ; t_{ij} : the total running time of AGV; t_{Uij} : the no-load running time; t_{BL} : the remaining blocking time of task loading workbench; t_{FR} : the remaining idle time of task unloading workbench; c_{ij} : the carrying cost per unit distance of AGV load unit flow running; while w_{ij} : the carrying cost per unit distance of AGV no-load running; c_1 , c_2 : the punishment cost due to machine tool blockage and due to machine tool idle; C : the total cost of processing subsystem and transport subsystem. N : The total number of AGVs in the workshop.

m : the number of tasks arrives; N_j : the number of idle cars at the unloading point j ; x_{si} : the number of free cars from the unload point D_j to the task load point P_i ; x_{sk} : the s task is performed by the k unit AGV.

Thus, the objective can be defined as following:

$$\min C = \sum_{i=1}^n \sum_{j=1}^n c_{ij} f_{ij} L_{ij} + \sum_{j=1}^n \sum_{i=1}^n w_{ij} L_{ij} + c_1 \sum_{s=1}^m \max[t_{Uij} - t_{BL}, 0] + c_2 \sum_{s=1}^m \max[t_{ij} - t_{FR}, 0] \quad (1)$$

s. t. :

$$f_{ij} \leq q \quad (2)$$

$$\sum_{s=1}^m x_{sk} = 1 \quad k = 1, 2, 3, \dots, N \quad (3)$$

$$\sum_{k=1}^N x_{sk} = 1 \quad s = 1, 2, 3, \dots, m \quad (4)$$

$$x_{sk} \in \{0, 1\} \quad (5)$$

$$\sum_{s=1}^m x_{sj} \leq N_j \quad (6)$$

$$t_{Uij} \leq t_{BL} \quad (7)$$

$$t_{ij} \leq t_{FR} \quad (8)$$

Equation (2) is the capacity of AGVs and equation (3), (4) and (5) limit a AGV car can only complete a task, and one task can only be performed by a AGV. Equation (6) represents that the number of tasks from the unloading point D_j is no more than the number of idle AGV parking at D_j . Equation (7) shows that no-load running time of AGV cannot exceed the remaining block time of the loading workbench. Equation (8) said total run time of AGV cannot exceed the remaining free time of the unloading workbench.

4. Case Study

4.1. Routing network pre-processing

Taking the practice case in figure 1 as an example, the pre-process operation is carried out as follows: For the path $P_1-M_1-D_1$, the inter node M_1 is only connected to one loading node and one unloading node, and the distances are 1 and 2, respectively. When the node M_1 is removed, the distance between P_1 and D_1 is modified to 3, which will not affect the entire network. Similarly, for the two adjacent unloading nodes D_1 and D_3 , there is only one path without passing through other loading and unloading nodes, namely $D_1-M_2-M_5-D_3$. This path has two intermediate nodes, M_2 and M_5 . As the path is unique, removing the two intermediate nodes, which will not affect the entire path network too.

In this network, the paths of adjacent loading and unloading nodes are not unique, e. g., there are two paths from P_2 to D_2 : $P_2-M_5-D_2$ and $P_2-M_3-M_6-D_2$. The length of these two paths is same. Therefore, we can choose any path and remove the intermediate node. In the case that two adjacent nodes have two or more paths and the path length is different, only the shortest path length is kept, and the length is taken as the distance between the two points. Through the above pre-processing, the results of network in figure 1 is shown in figure 2 as below.

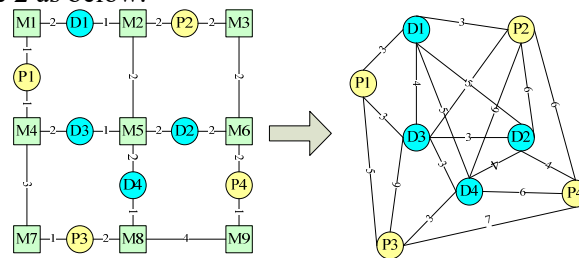


Figure 2. Result of routing network pre-processing

The pre-processed path network obtained from figure 2 can obtain the initial path matrix \mathbf{W} of any two loading and unloading nodes as shown below.

$$\mathbf{W} = \begin{bmatrix} D1 & D2 & D3 & D4 & P1 & P2 & P3 & P4 \\ 0 & 5 & 4 & 5 & 3 & 3 & \infty & \infty & D1 \\ 5 & 0 & 3 & 4 & \infty & 6 & \infty & 4 & D2 \\ 4 & 3 & 0 & 3 & 3 & 5 & 6 & \infty & D3 \\ 5 & 4 & 3 & 0 & \infty & 6 & 3 & 6 & D4 \\ 3 & \infty & 3 & \infty & 0 & \infty & 5 & \infty & P1 \\ 3 & 6 & 5 & 6 & \infty & 0 & \infty & 6 & P2 \\ \infty & \infty & 6 & 3 & 5 & \infty & 0 & 7 & P3 \\ \infty & 4 & \infty & 6 & \infty & 6 & 7 & 0 & P4 \end{bmatrix}$$

4.2. Un-load/load shortest path calculating

The initial path matrix **W** obtained above is loaded into MATLAB, and then the program of AGV path optimization is written. With applying Dijkstra algorithm iteratively, the path set of all no-load stage **Path1**, the path set of load stage **Path2** and the shortest distance matrix **L₁** and **L₂** between each load unloading point can be obtained, as shown below:

$$\text{Path1} = \begin{bmatrix} D1 \rightarrow P1 & D1 \rightarrow P2 & D1 \rightarrow P1 \rightarrow P3 & D1 \rightarrow P2 \rightarrow P4 \\ D2 \rightarrow D3 \rightarrow P1 & D2 \rightarrow P2 & D2 \rightarrow D4 \rightarrow P3 & D2 \rightarrow P4 \\ D3 \rightarrow P1 & D3 \rightarrow P2 & D3 \rightarrow P3 & D3 \rightarrow D2 \rightarrow P4 \\ D4 \rightarrow D3 \rightarrow P1 & D4 \rightarrow P2 & D4 \rightarrow P3 & D4 \rightarrow P4 \end{bmatrix}$$

$$\text{Path2} = \begin{bmatrix} P1 \rightarrow D1 & P1 \rightarrow D3 \rightarrow D2 & P1 \rightarrow D3 & P1 \rightarrow D3 \rightarrow D4 \\ P2 \rightarrow D1 & P2 \rightarrow D2 & P2 \rightarrow D3 & P2 \rightarrow D4 \\ P3 \rightarrow D4 \rightarrow D1 & P3 \rightarrow D4 \rightarrow D2 & P3 \rightarrow D3 & P3 \rightarrow D4 \\ P4 \rightarrow D2 \rightarrow D1 & P4 \rightarrow D2 & P4 \rightarrow D2 \rightarrow D3 & P4 \rightarrow D4 \end{bmatrix}$$

$$\mathbf{L1} = \begin{bmatrix} 3 & 3 & 8 & 9 \\ 6 & 6 & 7 & 4 \\ 3 & 5 & 6 & 7 \\ 6 & 6 & 3 & 6 \end{bmatrix} \quad \mathbf{L2} = \begin{bmatrix} 3 & 6 & 3 & 6 \\ 3 & 6 & 5 & 6 \\ 8 & 7 & 6 & 3 \\ 9 & 4 & 7 & 6 \end{bmatrix}$$

4.3. Multi-task assignment solution

Based on the w task set **Y_s** and the idle AGV set **Y_v**, which are finally determined after the task scheduling process, the task assignment can be calculated according to the Hungarian algorithm according. It is assumed that there are four free AGVs at four different unloading sites of four workstations, and the loading sites on four workstations have exactly four handling tasks. Therefore, we only need to set the no-load shortest path matrix **L₁** obtained above as the coefficient matrix in the assignment problem, and use Hungarian algorithm to obtain the task assignment. The result matrix **S** is as follows:

$$\mathbf{S} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

For other situations, that is, the number of free AGVs is greater than the number of tasks, or multiple different transport tasks are at the same loading point, or multiple free AGVs are at the same unloading point, we just change the coefficient matrix according to the actual situation.

In this practice case shown in figure 1, there are six tasks: $P_1 \rightarrow D_3$, $P_2 \rightarrow D_1$, $P_2 \rightarrow D_4$, $P_3 \rightarrow D_2$, $P_3 \rightarrow D_4$ and $P_4 \rightarrow D_1$. Actually, the parking location of the free AGV is determined by the transport task at the previous stage, that is, the current location of the free AGV is unknown. It can be assumed that at the current time, there are six idle AGVs in the system, two at the unloading point D_1 and D_4 , and one at the unloading point D_2 and one at D_3 . The coefficient matrix **L''** for multi-task assignment at this moment can be obtained, and also the assign matrix **S'** can be calculated.

$$\mathbf{L''} = \begin{bmatrix} 3 & 3 & 3 & 8 & 8 & 9 \\ 3 & 3 & 3 & 8 & 8 & 9 \\ 6 & 6 & 6 & 7 & 7 & 4 \\ 3 & 5 & 5 & 6 & 6 & 7 \\ 6 & 6 & 6 & 3 & 3 & 6 \\ 6 & 6 & 6 & 3 & 3 & 6 \end{bmatrix} \quad \mathbf{S'} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Table 2. The result of routing with multi-task assignment AGV

Task	Routing of AGVs	Shortest distance
$P_1 \rightarrow D_3$	$D_3 \rightarrow M_4 \rightarrow P_1 \rightarrow M_4 \rightarrow D_3$	6
$P_2 \rightarrow D_1$	$D_1 \rightarrow M_2 \rightarrow P_2 \rightarrow M_2 \rightarrow D_1$	6
$P_2 \rightarrow D_4$	$D_1 \rightarrow M_2 \rightarrow P_2 \rightarrow M_2 \rightarrow M_5 \rightarrow D_4$	9
$P_3 \rightarrow D_2$	$D_4 \rightarrow M_8 \rightarrow P_3 \rightarrow M_8 \rightarrow D_4 \rightarrow M_5 \rightarrow D_2$	10
$P_3 \rightarrow D_4$	$D_4 \rightarrow M_8 \rightarrow P_3 \rightarrow M_8 \rightarrow D_4$	6
$P_4 \rightarrow D_1$	$D_2 \rightarrow M_6 \rightarrow P_4 \rightarrow M_6 \rightarrow D_2 \rightarrow M_5 \rightarrow M_2 \rightarrow D_1$	13

5. Conclusion

This paper provides a multiple transport task assignment and routing optimization method of AGVs under multi-task. In order to solve the schedule problem, we divide the scheduling process into two stages: no-load and load. In the load-travel stage, the loading and unloading point are given by the task, and the shortest path of each task can be obtained by Dijkstra method. In the second stage, we improve the transport system efficiency via optimizing the routing of AGVs. Considering the constraints of the processing system, a model is built to minimize the total cost of the transportation system and the processing system. Finally, we provide a hybrid algorithm combining Dijkstra algorithm and Hungarian algorithm to obtain the shortest path matrix of no load and load firstly, and then to get assignment of AGVs to the transport tasks according to the urgent task set. The practical case result also shows that this algorithm is available and efficient.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (No. 61603280) and Foundation of WUST Fund for Young Teachers (No. 2016xz029).

References

- [1] Guang, X.P., Dai, X., Li, J. (2009). Variable neighborhood niche genetic algorithm based AGV flow path design method. *China Mechanical Engineering*, 20(21), 2581-2586.
- [2] Xia, Q., Ye, X. (2008) The application of genetic algorithm in global optimal path searching of AGV. *Journal of Sichuan University (Natural Science Edition)*, 45(5):1129-1136
- [3] Xiao, H. N., Lou, P. H., Man, Z. G. (2012). Real-time multi-attribute dispatching method for automatic guided vehicle system. *Computer Integrated Manufacturing Systems*, 18(10), 2224-2230.
- [4] Fazlollahtabar, H., Saidi-Mehrabad, M. (2015) Methodologies to Optimize Automated Guided Vehicle Scheduling and Routing Problems: A Review Study. *Journal of Intelligent & Robotic Systems*, 77(3-4): 525-545
- [5] Song, L., Jin, S., Tang, P. (2016) Simulation and Optimization of Logistics Distribution for an Engine Production Line. *Journal of Industrial Engineering and Management*, 9(1):59-72
- [6] Wang, C., Guan, Z., Shao, X. (2014) Simulation-based optimisation of logistics distribution system for an assembly line with path constraints, *International Journal of Production Research*, 52(12): 3538-35