Journal of Electronic Research and Application, 2024, Volume 8, Issue 3
http://ojs.bbwpublisher.com/index.php/JERA
ISSN Online: 2208-3510
ISSN Print: 2208-3502

Research on Shortest Path BFS Strategy in Multi-AGV Scheduling System

Shi Deng[1], **Di Wu**[2]*

[1]Beijing Huahang Weishi Robot Technology Co., Ltd., Shanghai Branch, Shanghai 201700, China
[2]Gu Fang (Shanghai) Software Technology Studio (Limited Partnership), Shanghai 211206, China

***Corresponding author:** Di Wu, dsyeah@yeah.net

Abstract: With the increasing maturity of automated guided vehicles (AGV) technology and the widespread application of flexible manufacturing systems, enhancing the efficiency of AGVs in complex environments has become crucial. This paper analyzes the challenges of path planning and scheduling in multi-AGV systems, introduces a map-based path search algorithm, and proposes the BFS algorithm for shortest path planning. Through optimization using the breadth-first search (BFS) algorithm, efficient scheduling of multiple AGVs in complex environments is achieved. In addition, this paper validated the effectiveness of the proposed method in a production workshop experiment. The experimental results show that the BFS algorithm can quickly search for the shortest path, reduce the running time of AGVs, and significantly improve the performance of multi-AGV scheduling systems.

Keywords: AGV; Path planning; AGV scheduling system; BFS algorithm

Online publication: June 14, 2024

1. Introduction

The "14th Five-Year Plan" national strategy emphasizes accelerating the advancement of modern manufacturing. In recent years, in the field of intelligent manufacturing, automated guided vehicles (AGVs) have become a common transportation option due to their reliability and safety, leading to their widespread application in intelligent manufacturing workshops, flexible manufacturing systems, and many other industrial manufacturing environments [1].

To address the efficiency issues of AGV operations, Dijkstra initially proposed research related to the shortest paths of AGVs [2]. Subsequently, various path algorithms for AGVs have been continuously explored, and domestic researchers have put forward some theoretical viewpoints on path algorithms. Among them, Chen [3] studied the parallelism and integration of neural networks, developing an AGV path-planning algorithm based on neural networks. Xiao [4] utilized fuzzy logic to merge ultrasonic and visual sensor data, allowing AGVs to navigate autonomously in uncertain environments. Li *et al.* [5] proposed an adaptive AGV trajectory tracking and control method. This paper optimizes the route problem of multiple AGV vehicles based on the BFS algorithm's path planning method, achieving efficient scheduling of multiple AGVs in complex environments. Experimental results demonstrate that the optimized BFS algorithm can effectively reduce the average completion time of AGVs and the total system operation time, enhancing the performance of the scheduling system.

2. AGV scheduling system

An AGV scheduling system is a comprehensive system platform used for managing and scheduling AGVs. It manages multiple AGV devices operating simultaneously in the same working environment, ensuring the completion of scheduling tasks and the efficient operation of the system [6]. AGV scheduling systems cover functions such as homepage, map management, vehicle management, task management, task logs, system settings, robot communication, robot control, and robot scheduling. The system offers a powerful map design tool on the browser side for offline design and editing of 2D map layouts for AGV carts. Through vehicle and homepage management, real-time monitoring and management of various AGV operational statuses are made possible. Map management allows for map creation and design, with settings for map properties such as name, address, running speed of vehicles on the map, acceleration, obstacle avoidance distance, etc. Task management involves task issuance viewing for the scheduling system, displaying task ID, task status, issuance time, issuance result, etc., and determining the specific AGV to execute tasks and priority level. Robot scheduling applies relevant algorithms to find the optimal path for AGV travel based on constraints of the map environment model and performance evaluation indicators, aiming to minimize completion time or reduce operating costs. Finally, the maintenance and monitoring of the scheduling system involves the operation status transition mechanism of AGV in the scheduling system and task scheduling. The AGV scheduling system records all tasks executed by the carts, task execution status, and task execution logs to ensure the successful completion of scheduled tasks by AGVs.

The scheduling of a multi-AGV system is based on the tasks and the conditions of the AGV vehicles. By using scheduling algorithms, AGVs are allocated and arranged to achieve the best match between AGV vehicles and tasks, thereby enhancing the overall efficiency of the multi-AGV system. Typically, in multi-AGV path planning, after assigning a task to an AGV, the system searches for an optimal path based on factors such as the shortest distance and the fewest turns. The principle of the shortest path dictates that when tasks occur, the scheduling system plans out the running routes for all tasks, calculates the travel path length for each AGV performing a single task, selects the AGV with the shortest distance to execute the task until all AGVs have tasks assigned or task distribution is completed, ultimately minimizing the running distance of the AGV performing tasks [7].

3. Methods

3.1. Map-based path search algorithms

Based on known information such as docking areas, roads, obstacles, etc., map-based path search algorithms aim to find the optimal path between two points. Commonly used algorithms include depth-first search (DFS), breadth-first search (BFS), Dijkstra algorithm, A* algorithm, etc [8].

(1) Depth-first search algorithm [9]: Follows the principle of “going all the way down a path,” starting from the root node of the tree, traversing and marking nodes by depth until all nodes have been visited.

(2) Breadth-first search (BFS) algorithm [10]: Starts from the root node of the tree and searches layer by layer according to the width of the tree, following the principle of “exploring every path,” until no

path remains to explore.

(3) Dijkstra algorithm [11]: Integrates the greedy approach into the breadth-first algorithm. It calculates the distances from surrounding points to the initial point, saves the shortest paths from the initial point to the surrounding points in the shortest path set R, recalculates the distances from the current node's surrounding points to the starting point until the shortest path set R includes the endpoint.

(4) A* Algorithm [12-15]: A heuristic-based path planning algorithm. During pathfinding, it considers the distance from each intermediate point to the starting point and also takes into account the estimated distance from the intermediate point to the endpoint.

3.2. Shortest path BFS algorithm

Global path planning in navigation tasks is a single-sequence planning problem (one-query planning), aiming to search for a feasible path from the robot's workspace to ensure that local planning algorithms can be executed quickly and easily. Based on the single sequence planning algorithm, BFS is a search algorithm implemented using a queue, commonly used to solve graph theory problems and maze problems.

In a tree, the path from one node to another is unique, and there may be multiple paths between nodes. The shortest path problem is a typical application of BFS, and its method is closely related to level-order traversal. In a binary tree, BFS can traverse layer by layer. Starting from the source node, BFS first traverses the first layer of nodes at a distance of 1 from the source, then moves on to the second layer of nodes at a distance of 2 from the source, and so on, progressing through each layer. Using the BFS algorithm, nodes closer to the source are traversed first, enabling the identification of the shortest path to reach a specific node. In practice, level-order traversal, BFS traversal, and the shortest path are progressive relations. On the basis of BFS traversal, distinguishing each layer of traversal leads to level-order traversal. Recording the number of layers in level-order traversal results in the shortest path.

4. Experiment

The experimental scenario for the shortest BFS algorithm includes small-scale production areas and warehousing areas with a spatial area of less than 800 m^2. The experimental code implements a path planning function based on the BFS algorithm to find the shortest path from the starting point to the endpoint. During the search process, it traverses neighboring nodes and implements a BFS using a queue until the endpoint is found or all feasible nodes are searched. Part of the code is expressed as follows:

```
for (int i = 0; i < 4; i++) {
    int tx = temp.x;
    int ty = temp.y;
    int t_go = map_point_bfs[temp.x][temp.y][i+1];
while (t_go > 0) {
    t_go = map_point_bfs[tx][ty][ngo[i]];
    tx = tx + nx[i];
    ty = ty + ny[i];
if(tx < 0 || ty < 0 || t_go < 1){
     break; }
```

(1) In the search, there is a loop that attempts to find a path to the endpoint by traversing four possible directions (up, down, left, right). This process is achieved by applying offsets (nx[i], ny[i]) to the current node ("temp.x," "temp.y").

(2) Updating the coordinates of the next node: Initially setting “tx” and “ty” as the current node’s coordinates, then obtaining the number of steps to move in the current direction from “map_point_bfs[temp.x][temp.y][i+1].” In the “while” loop, “tx” and “ty” are updated based on “nx[i]” and “ny[i]” to attempt movement in that direction.

(3) Checking if it can continue forward: The loop will break if the updated coordinates “tx” and “ty” exceed the boundaries or if “t_go” is less than 1, indicating that moving in that direction is not possible.

(4) Checking if the endpoint is reached: In the process of moving in a direction, each time the coordinates “tx” and “ty” are updated, it checks if these new coordinates match the endpoint (“end_x,” “end_y”). If they do, it means a path from the starting point to the endpoint is found, and the function returns 1 to indicate a successful pathfinding.

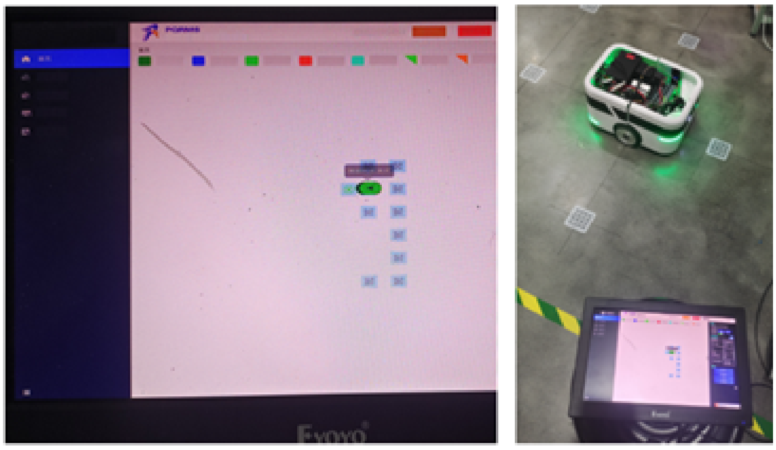

Figure 1. Experimental test

Experimental verification shows that the results of running the BFS algorithm code demonstrate its ability to find the shortest and least turning path in a short amount of time, with good performance. Simultaneously, the AGV scheduling system can operate effectively within the work area, efficiently managing multiple AGVs to complete tasks.

5. Conclusion

With the continuous advancement of industrial automation technology and the increasing demand for intelligent manufacturing, optimizing multi-AGV scheduling systems will become crucial for enhancing production efficiency and reducing operational costs. The BFS algorithm used in this study is suitable for path planning on known maps, offering the shortest path with fewer turns. In practical operations, reducing the number of AGV turns and stops significantly enhances work efficiency. While the shortest path principle focuses on minimizing travel distance, it overlooks conflicts between AGV routes, which may result in extended scheduling execution times. Future work should further explore the combination of BFS strategy with other intelligent algorithms to achieve more flexible and efficient multi-AGV scheduling solutions that meet evolving production demands and work scenarios.

Disclosure statement

The authors declare no conflict of interest.

References

[1] Chen H, 2022, Research on Task Scheduling and Path Planning of Multi-AGV Scheduling System, dissertation, Hefei University of Technology.

[2] Zhan Y, 2020, Research on Task Scheduling of Multi-AGV Systems, dissertation, Hubei University of Technology.

[3] Shi T, 2015, Research and Implementation of Path Planning for Single AGV in Automated Stereoscopic Warehouse, dissertation, Anhui University of Science and Technology.

[4] Xiao B, Wu X, Lou T, et al., 2004, Study on AGV Navigation Strategy Based on Fuzzy Data Fusion. Systems Simulation Journal, 16(1): 148–151.

[5] Li J, Zhuang J, Wang S, 2005, Adaptive Trajectory Tracking Control of Wheeled Mobile Robots Based on Comprehensive Guidance. Journal of Xi'an Jiaotong University, 39(3): 252–255.

[6] Chen Z, 2021, Research on Path Planning Optimization and Communication Adaptation Technology of AGV Scheduling System, dissertation, Institute of Mechanical Science Research, Beijing.

[7] Wei X, 2021, Research on Efficient Task Allocation Algorithm for AGV Scheduling System, dissertation, Zhejiang University.

[8] Chu J, 2021, Research on AGV Path Planning and Dynamic Scheduling in Intelligent Packaging Workshop, dissertation, Harbin Institute of Technology.

[9] Tarjan R, 1972, Depth-First Search and Linear Graph Algorithms. SIAM J Comput, 1: 146–160.

[10] Bundy A, Wallen L, 1984, Breadth-First Search. Proceedings of Catalogue of Artificial Intelligence Tools, 13

[11] Karaman S, 2012, Sampling-based Algorithms for Optimal Path Planning Problems, dissertation, Massachusetts Institute of Technology, Cambridge.

[12] Wang X, Lu X, Feng C, et al., 2018, Improvement and Verification of A-Star Algorithm for AGV Path Planning. Computer Engineering and Applications, 54(21): 217–223.

[13] Wang X, Lu W, Feng C, et al., 2019, Path Planning Research Based on an Improved A* Algorithm from Obilerobot. IOP Conference Series: Materials Science and Engineering, 569(5): 52044.

[14] Lao C, Li P, Feng Y, 2021, Greenhouse Robot Path Planning Based on Improved A* and DWA Algorithm Fusion. Journal of Agricultural Machinery, 52(1): 14–22.

[15] Zong C, Lu L, Yu X, et al., 2017, A Spatial Multi-Degree-of-Freedom Robotic Arm Path Planning Method Based on A* Algorithm. Journal of Hefei University of Technology (Natural Science Edition), 40(2): 164–168.