Received February 2, 2021, accepted February 17, 2021, date of publication February 23, 2021, date of current version March 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3061288

Improved Particle Swarm Optimization Algorithm for AGV Path Planning

TAO QIUYUN, SANG HONGYAN, (Member, IEEE), GUO HENGWEI, AND WANG PING
School of Computer Science, Liaocheng University, Liaocheng 252059, China

Corresponding author: Sang Hongyan (sanghongyan@lcu-cs.com)

This research was partially supported by the National Natural Science Foundation of China (61773192 and 61803192), Special fund plan for local science and technology development lead by central authority, Research project of Liaocheng University (318011922).

ABSTRACT In smart manufacturing workshops, automated guided vehicles (AGVs) are increasingly used to transport materials required for machine tools. This paper studies the AGV path planning problem of a one-line production line in the workshop, establishes a mathematical model with the shortest transportation time as the objective function, and proposes an improved particle swarm optimization(IPSO) algorithm to obtain an optimal path. In order to be suitable for solving the path planning problem, we propose a new coding method based on this algorithm, design a crossover operation to update the particle position, and adopt a mutation mechanism to avoid the algorithm from falling into the local optimum. By calculating the shortest transportation time obtained, the improved algorithm is compared with other intelligent optimization algorithms. The experimental results show that the algorithm can improve the efficiency of AGV in material transportation and verify the effectiveness of related improvement mechanisms.

INDEX TERMS Automated guided vehicle, improved particle swarm optimization algorithm, scheduling optimization, routing plan.

I. INTRODUCTION

With the continuous development of society, intelligent manufacturing has been extensively developed. Material transportation in the manufacturing process is an indispensable part of the workshop production process and an important part of the workshop job scheduling problem [1]. Improving material handling management and the automation of material storage processes can not only improve work efficiency and speed up material flow, but also reduce production costs, shorten production cycles and increase economic efficiency, so AGV came into being. The use of AGV in the intelligent manufacturing workshop to solve the problem of machine tool scheduling, thereby obtaining the best solution, will help the company develop into a manufacturing giant [2], [3]. In the intelligent production workshop, AGV is used to solve two problems. One is the task scheduling problem. The other is the path planning problem. Both are important parts that affect workshop performance. This paper focuses on the path planning part and aims to find the optimal transportation path for the machine tool that calls materials. This plays an important role in shortening the transportation distance, saving material transportation time, reducing logistics costs and improving operation efficiency. So AGV has great application value. Regarding the path planning of the AGV [4], it mainly includes three issues: 1) Whether the running path is conflict-free and deadlock-free (this is only limited to the case of multiple AGVs, and the single AGV system does not need to consider this issue); 2) The planned path should optimize the operating efficiency of the entire workshop; 3) Whether there is a feasible path between the two points (for example, there are obstacles that involve impassable conditions).

This article aims at the one-line production line problem in the production workshop, and establishes a mathematical model with the shortest transportation time as the goal, and finally obtains an optimal path. In the production workshop, it usually includes three parts: AGV, machine tool and warehouse. The layout of the one-line production line is shown in Fig.1. There are n machine tools in the production line in total, with $n/2$ machines on each side. The AGV travels back and forth on the middle road to transport materials for the machine tool. The initial location of the AGV is in the warehouse, and the material is transported to one of the machine tools according to the planned route by AGV.

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

Then the AGV directly transports the material to the next machine tool without returning to the warehouse until all tasks are completed.

In the past research on the problem of AGV path planning, scholars mostly use dynamic programming algorithms, heuristic algorithms or intelligent optimization algorithms, such as GA algorithm [5], ant colony optimization (ACO) [6], [7] algorithm and particle swarm optimization(PSO) [8] algorithm. Although the dynamic programming algorithm can get the best solution to this problem, it usually has the disadvantages of time-consuming and low efficiency. Based on the above, this article uses the IPSO algorithm to solve the path planning problem of the one-line production line. The novelty of the IPSO algorithm is: 1) A coding method based on path planning problem is proposed. 2) The crossover operation is designed to realize position update. 3) The mutation operation is proposed to help jump out of the local optimum.

The rest of this article is organized as follows. Section 2 introduces the related work of AGV and the PSO algorithm. Section 3 describes the establishment of the model. Section 4 introduces the improvement of the IPSO algorithm. In section 5, the improved algorithm is evaluated through experiments to verify its effectiveness. The sixth section puts forward conclusions and looks forward to the future development direction of this research.

II. LITERATURE REVIEW

A. AGV PATH PLANNING AND SCHEDULING PROBLEMS IN THE WORKSHOP

AGV is currently widely used in the production workshop, mainly to select the best transportation path. The goal is to meet the minimum distance between the starting point and the end point, or to avoid collisions with obstacles. Path planning can be transformed into an optimization problem under certain goals, such as the shortest transportation time. At the same time, certain constraints, such as time windows and no collisions, can be restricted. Luo [9] presented a single AGV scheduling mathematical model to ensure that the total time completing the task is the shortest. Minis and Tatarakis [10] proposed a dynamic programming algorithm to minimize the routing cost of the single vehicle delivering random products and picking up items. Du *et al.* [5] created a feasible route topology according to the specific workshop layout and established a route planning model for multi-load AGV. Using the model, the correct and effective path can be got, and the efficiency of machine use can be improved. Guo *et al.* [11] and others considered AGVs with limited power supply capacity and established a single AGV energy-saving path planning model with transportation path and transportation energy consumption as the optimization goal. Ding *et al.* [8] found an optimal path to avoid collisions by considering the number of turns and turn time of the AGV, and introducing GA to compensate the deficiency of the PSO algorithm. Smolic-Rocak *et al* [12] proposed a dynamic routing method to make the determined shortest feasible path by extending the time window, so that there are no problems such as conflicts in the system. He and Mao [7] used the ACO algorithm to construct a wireless grid route and obtained an optimal distance route with obstacle avoidance ability. Most of the studies mentioned above did not consider the problem of inconsistent machine tool calling material time, but this problem exists in the production workshop. On this basis, this article considers the *RTS* (a fixed time period, this article is set to 1000s). In RTS, the number of machine tools that need to distribute materials is different, and the time point is different. At any point in this period, any machine tools may call for materials. According to the time for the machine tool to call for materials, an optimal path is made so that the time for the AGV to complete the task is the shortest.

B. PARTICLE SWARM OPTIMIZATION ALGORITHM

The PSO algorithm is a swarm intelligence algorithm [13] proposed by Dr. Eberhart and Dr. Kennedy in 1995. It modifies individual action strategies through group information sharing and the summary of individual's own experience, so as to determine how to adjust and change the direction and speed of the flight in the next iteration. The speed and position solution formulas are important for the PSO algorithm. The formulas are defined as follows:

$$v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1 \left(p_{id}^{k} - x_{id}^{k}\right) + c_2 r_2 \left(p_{gd}^{k} - x_{id}^{k}\right) \quad (1)$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \quad (2)$$

where d represents the dimension of the search space. w is the inertia weight factor, reflecting the particle's ability to inherit the speed of the previous iteration. $d = 1,2,\ldots, D$; $i = 1, 2,\ldots, N$. N represents the number of particles. k is the current number of iterations. v_{id} is the velocity of the particles. c_1 and c_2 are non-negative constants, called acceleration factors. r_1 and r_2 are random numbers.

The PSO algorithm is a random search algorithm based on mutual cooperation between groups. It was developed by simulating the foraging behavior of birds. Because of its simple operation and fast convergence speed, it has been widely used in many fields such as scheduling problem [14]–[16], neural networks [17], power systems [18], biology [19], and combined double auctions [20], etc.

Li *et al.* [21] proposed an improved PSO algorithm embedded with a local search strategy (PSO-LS) to solve the single machine scheduling problem with energy consumption constraints to achieve the goal of minimizing the total weighted delay. The article designs a coding method suitable for single-machine scheduling problems and the PSO-LS framework. Experimental results show that the improved algorithm has certain advantages for solving large-scale problems.

The paper by Ding and Gu [22] presented a hybrid HLO-PSO algorithm to solve the flexible job shop scheduling problem (FJSP). With the support of the HLO algorithm, the PSO algorithm has stronger individual learning capabilities and local search capabilities. The comparison between

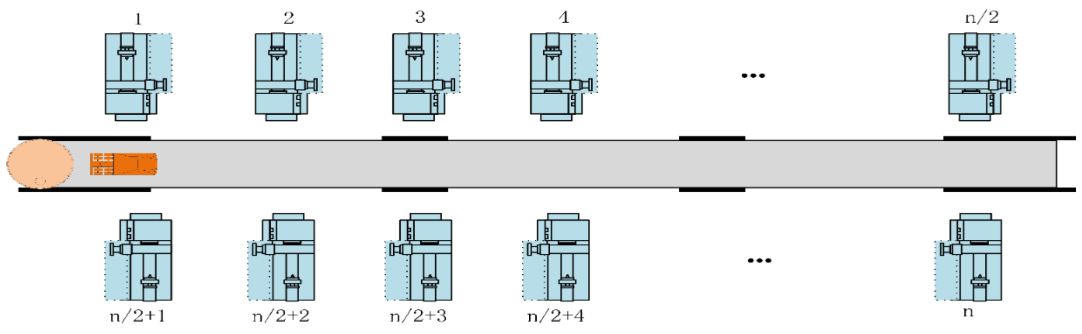


FIGURE 1. Schematic diagram of one-line workshop production line.

experimental results proves that the improved algorithm proposed is more suitable for FJSP.

Zhang *et al.* [15] proposed a new bare-bones multi-objective PSO algorithm based on the environmental/ economic scheduling problem. The algorithm does not need to adjust the control parameters. In algorithm, the mutation operator can be dynamically changed over time to improve the search ability. The global particle leader can be updated based on the particle diversity. Experimental results show that the algorithm can produce an excellent approximation of the real pareto front.

Through literature analysis, it can be concluded that the basic PSO algorithm has certain shortcomings. For example, the basic PSO algorithm is only suitable for solving continuous problems, and has certain shortcomings in dealing with combinatorial optimization problems, and it is easy to fall into local optimality. For the different problems studied, the PSO algorithm adopts different optimization strategies to improve the performance of the PSO algorithm, such as modifying the inertia weight coefficient [23], [24], updating the speed and position formula [25]–[27], and enhancing the local search [28], or combined with other intelligent algorithms [29]. These conversions can help the PSO algorithm find high-quality solutions. However, for the problem of processing path planning, the continuous solution is likely to cause a large number of invalid searches and affect efficiency. Therefore, this paper proposes an integer coding method suitable for path planning problems, designs crossover operations to update particle positions, and uses a mutation mechanism to help particles jump out of local optimal positions to enhance spatial search capabilities. Through the above improvements, the method is more suitable for solving the problem of AGV path planning, so as to ensure that the AGV can complete the task in the shortest time.

III. PROBLEM DESCRIPTION AND MODELING

The problem of AGV path optimization is usually described as: In the workshop, there is an AGV and a one-line production line containing n machine tools. However, not all machine tools call AGV for material transportation. Assume that the number of machine tools calling for materials is m, which means, during *RTS*, there are m ($m \leq n$) machine tools that need AGV to transport materials. The AGV needs to start from the warehouse, and select the best path to transport materials according to the time and distance of the machine tools calling materials.

In view of the problem shown in Fig.1, the following assumptions are made:

(1) AGV speed is consistent during operation.
(2) All equipment and machines are available.
(3) After the AGV is started, the problems of insufficient power and sudden failure are ignored.
(4) The distance between adjacent machine tools is equal.
(5) The material types required by each machine tool are the same.
(6) AGV does not need to return to the warehouse after performing tasks, but directly executes the next task in the queue, and does not return to the warehouse until all tasks are completed.

Assume that production line in the workshop has 30 machine tools in total, that is, $n = 30$. The number m of machine tools calling materials is uncertain. As mentioned above, how to transport materials to the machine tool in the shortest time is our first consideration. Specifically, we propose the following mathematical model:

T_{sl}: Time for AGV to load material to the machine tool

T_i: The material transportation completion time of the machine tool represented by the i-th task

T_i': The material calling time of the machine tool represented by the i-th task

C_i: The Time to start transferring materials to the machine tool represented by the i-th task

m: Number of machines tools calling for material

V: The running speed of AGV

pos (x_i): The position of the machine tool represented by the i-th task

pos (x_0): The initial position of the AGV

$d_{i-1,i}$: The distance between the two machine tools represented by the i-1th and i-th tasks

$t_{i-1,i}$: The time difference between the task completion time of the two machine tools represented by the i-1 and i-th tasks

$x_{i-1,i}$: decision variable representing whether operated

$$x_{i-1,i} = \begin{cases} 0, & T_i' < T_{i-1} \\ 1, & else \end{cases}$$

TABLE 1. **Coding method.**

	0	1	2	3	4	5	6	7	8	9
no:	9	2	7	11	18	8	10	3	14	17
time:	0	120	200	250	320	500	650	720	800	950

Taking the shortest transportation time as the objective function, the formula is as follows:

$$F = min\left\{\sum_{i=1}^{m} t_{i-1,i}\right\} \tag{3}$$

subject to:

$$d_{i-1,i} = |pos\,(x_{i-1}) - pos(x_i)| \tag{4}$$

$$t_{i-1,i} = d_{i-1,i}/V + T_{sl} + x_{i-1,i}|T_i' - T_{i-1}| \tag{5}$$

$$pos(x_0) = pos\,(AGV) \tag{6}$$

$$T_i = T_{i-1} + t_{i-1,i} \tag{7}$$

$$T_{i-1} < C_i \tag{8}$$

V represents the running speed of AGV. T_{sl} is the time for AGV to load the material to the machine. The objective function (3) is to find an optimal path so that the AGV spends the shortest time after completing the distribution tasks of all machine tools. Equation (4) represents the actual distance between two adjacent distribution tasks of machine tools, and Equation (5) represents the time required between the two machine tools, including three parts: the actual distance required time, loading time and waiting time. Equation (6) represents the initial position of the AGV, which is the warehouse. Equation (7) represents the time to complete the material transportation of the machine tool represented by the i-th task. Equation (8) expresses a constraint that the current task must be completed before the next transportation task can be carried out. It is worth noting that the position of the AGV needs to be updated at any time according to the number of the material being transported to the machine tool.

IV. IMPROVED PSO ALGORITHM

Although the PSO algorithm shows excellent performance in solving various optimization problems, there is no unified algorithm that can solve all optimization problems. In order to improve the performance of the PSO algorithm for solving discrete problems, a coding method suitable for path planning is proposed, and crossover and mutation operations are designed to update particle position, enhance local search, and greatly improve search efficiency.

A. ENCODING OF PARTICLES

Since AGV scheduling is a discrete optimization problem, it should be coded using a discrete coding method, which can greatly reduce the search range of the solution space. We use the following vector to represent the i-th particle in the IPSO, where X_i represents the i-th particle in the IPSO, m represents the length of the particle, that is, the total number of machine tools that call the material.

Because the machine and corresponding time of the calling material are different, in order to make the coding more convenient, this article adopts an integer coding method, which uses the serial number of the transported material to the machine tool to encode the particles. This method directly reflects the AGV transporting the material to the machine tool. Sequence can be expressed as $\{0, 1, 2, \ldots, m\text{-}1\}$. Each element in the particle includes the number of the machine tool and the calling time of the machine tool. Each particle represents a routing plan. According to the number of the calling material and the calling time of the machine tool, a particle is randomly generated. Therefore, the code can be displayed in Table 1, where *no* represents the machine number of the calling material, and the *time* corresponds to the specific time when the machine calls the material. For example, the i-th particle is $X_i = (3,1,0,5,9,8,6,7,2,4)$, which means that the order of AGV transportation material is (11,2,9,8,17,14,10,3,7,18).

B. INITIALIZATION OF PARTICLE

For the evolution and convergence of the algorithm, a well-structured initial population is very important. The algorithm in this paper takes the population size as 100, and the process of generating the initial population is as follows:

Step 1: Determine the particle length as m according to the number of machine tools calling the material.

Step 2: Use the method of creating arbitrary discrete random populations to generate random numbers r_k, $r_k \in [0, m\text{-}1]$ for particles, and generate m times in total.

Step 3: Confirm the generated particles and ensure that all the machine tools that call materials are included.

Step 4: Repeat the above steps 100 times to generate an initial population of 100 individuals.

C. CROSSOVER OPERATION

In the IPSO algorithm, a crossover operation is designed to update the particle position. In this article, set the cross-probability $G = 1$, that is, all particles must update the position through the cross operation.

1) Randomly select the segment $S = \{S_1, S_2\}$ of individual optimal or global optimal solution P.
2) Insert the segment after the element e of the particle X_i (e $\notin S$, and the distance between the machine tool represented by the e element and the machine tool represented by S_1 is the closest).
3) Delete S_1, S_2 in particle X_i.

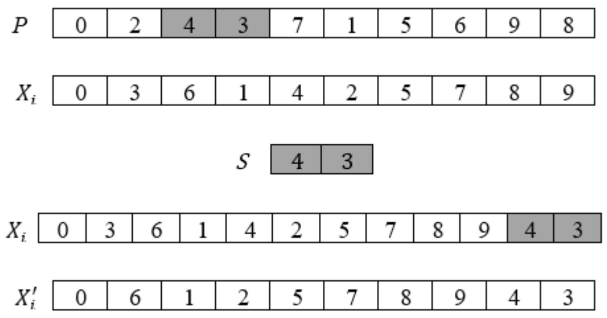


FIGURE 2. Cross operation of particle.

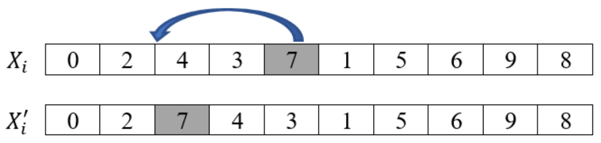


FIGURE 3. Insertion operation.

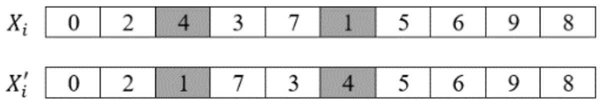


FIGURE 4. Reverse sequence mutation.

The example is as follows: Assuming that the selected segment $S = \{4,3\}$ in P, the closest distance to the machine tool 18 represented by $S_1 = 4$ is the machine tool 17 represented by the element 9, and then the S segment is inserted into the element 9 of the particle X_i after that. The duplicate elements in particle X_i are then deleted, and a new particle is obtained. The specific process is shown in Fig. 2.

D. MUTATION OPERATION

In the later iteration of the PSO algorithm, the algorithm has the disadvantage of easily falling into the local optimum, the convergence speed is relatively slow, and the suboptimal solution is easily obtained. When it is detected that the global optimal value three times has not been updated, the algorithm is considered to fall into the local optimal value. To help jump out of the local optimum, a mutation operation is proposed to enhance its local search ability. In this paper, we set two mutation operations, the mutation probability is $Q = 0.2$, and the two mutation operations each account for 0.1. The two mutation operations are as follows:

(1) Insertion operation. Randomly select a position in the particle X_i, and insert the element at that position in front of its neighboring elements, as shown in Fig.3.

(2) Reverse sequence mutation. Randomly select two elements in the particle X_i, and then reverse the part in between, as shown in Fig.4.

E. ITERATION TERMINATION CONDITION

The termination condition adopted in this paper is that the algorithm stops running after H iterations. Record the total travel time of the AGV conveying material sequence obtained when iterating 20 times, 40 times, 60 times, 80 times, 100 times, 120 times, ..., to 200 times. Choosing an appropriate number of iterations can avoid unnecessary waste of program running time.

F. PROCEDURE OF THE ALGORITHM

According to the above improvement strategy, the complete IPSO algorithm process is as follows:

Step1: Initialize all particles randomly.

Step2: Calculate the fitness of all particles. Save the optimal solution of individual particles and the optimal solution of the group.

Step3: Perform crossover operation according to probability G.

Step4: Perform mutation operation with probability Q.

Step5: Calculate the fitness of the new generation of individuals, the optimal solution of the new generation of individuals and the overall best solution.

Step6: Determine whether the iterative conditions are met, if yes, output the result; if not, return to Step 3.

V. EXPERIMENTAL RESULTS AND ANALYSIS

A. EXPERIMENTAL SETTING

The experimental environment is the windows 10 operating system, the processor frequency is 2.5GHz, and it is implemented using the C++ programming language, and the Matlab software is used to draw pictures for the result analysis. To verify the effectiveness of IPSO, the experimental results are compared with the PSO algorithm (Compared with the IPSO algorithm in this paper, no mutation operation) algorithm, GA algorithm [5], ACO algorithm [7]. The experimental data of this paper refer to these three documents. We get the practical example from the actual factory (Foxconn Technology Group). The production line used in our experiments includes 30 machine tools and one AGV. Based on the number of machine tools that call materials, the experiment in this article compares two cases, namely 10 machine tools and 25 machine tools, representing small-scale and large-scale situations, to verify the effectiveness of the IPSO algorithm. The time when each machine tool calls materials is automatically generated within *RTS*.

1) Case 1: There are 10 machine tools in the *RTS* to call materials, as shown in Table 2.
2) Case 2: There are 25 machine tools in the *RTS* to call materials, as shown in Table 3.

B. PARAMETER SETTING

Tables 2 and 3 list the parameter values and experimental values of the AGV workshop. To improve accuracy, each algorithm was run 25 times. The comparison settings of IPSO algorithm parameters and other algorithms are shown in Table 4 and Table 5 below.

In Table 5, *PopSize*: population size, *G*: Crossover probability, *Q*: Mutation probability, *Rho*: Pheromone evaporation coefficient.

In the experiment, we use relative percentage increase (*RPI*) as the response variable, and its calculation formula

TABLE 2. Case 1 data.

no:	1	5	14	18	20	9	12	24	2	11
time:	10	110	120	240	450	560	640	780	820	960

TABLE 3. Case 2 data.

no:	8	18	25	6	12	20	24	3	30	26	15	2	1
time:	10	50	86	110	146	187	210	257	299	326	375	399	421
no:	29	22	14	9	26	10	23	16	5	7	4	17	
time:	451	482	522	546	588	637	678	746	799	825	868	980	

TABLE 4. Parameter settings of experiments.

Items	Values
Distance between two machines	5.5(m)
V	0.45(m/s)
RTS	1000(s)
T_{sl}	30(s)

TABLE 5. Parameter settings of algorithms.

Algorithm	Parameters
PSO	$PopSize$=100,$G = 1$
IPSO	$PopSize$=100,$G = 1$,$Q = 0.2$
GA	$PopSize = 100$, G=0.8, $Q = 0.2$
ACO	$PopSize = 100$, Rho=0.5

is as follows: $RPI = (F - F_b)/F_b*100\%$. Here, F represents the average value of the transportation time generated by each algorithm running 25 times in the experiment, and F_b represents the shortest transportation time generated by all the algorithms in the experiment. Obviously, in contrast, a lower RPI value is preferred.

C. RESULTS ANALYSIS

To test the efficiency of this algorithm, the IPSO algorithm was compared with the PSO (no mutation operation) algorithm, ACO algorithm and the GA algorithm for Cases 1-2. The results obtained are presented in the form of figures or tables to see the performance of each algorithm, thereby proving the effectiveness of the improved algorithm.

1) COMPARISON OF THE IPSO AND PSO ALGORITHMS

Firstly, in order to prove the influence of mutation operation on PSO algorithm, IPSO algorithm is compared with PSO (no mutation operation) algorithm. The comparison results are shown in table 6.

Obviously, as shown in Table 6, the mutation operation can help the PSO algorithm to enhance the search ability, thereby obtaining a good quality solution.

TABLE 6. Computational results for the RPI values of IPSO AND PSO algorithms.

Case	IPSO	PSO
1	**3.43**	8.27
2	**3.16**	5.37
Mean	**3.30**	6.82

The plot (a) and plot (b) of Fig. 5 show the convergence behavior of the IPSO and PSO algorithms' shortest transportation time in Case 1 and Case 2, respectively, as the number of iterations increases. Both situations reflect that in the early iterations, both have a faster convergence rate, and the result of the improved algorithm is always better than the PSO algorithm, and neither falls into a local optimum. However, it can be seen from both plot (a) and plot (b) that as the number of iterations increases, the PSO algorithm exhibits limitations. It converges prematurely around 140 iterations, and the optimization effect is poor. The final result is far worse than the IPSO algorithm. The IPSO algorithm has always had strong search ability. Due to the design of the mutation mechanism, the local search ability has been greatly enhanced, helping to jump out of the local optimum, and avoiding premature convergence, obtaining a final result superior to the PSO algorithm. This phenomenon shows that the improvement strategy proposed by IPSO is effective.

2) COMPARISON OF THE IPSO, ACO AND GA ALGORITHMS

The comparison results of RPI values obtained by IPSO, GA and ACO algorithms are shown in table 7. It can be seen that the IPSO algorithm can obtain better quality solutions. Through the calculation of case1 and case2, the average RPI generated by the IPSO algorithm is the smallest, which is 3.3%. The average RPI value of this algorithm is smaller than the ACO algorithm (6.98%) and GA algorithm (6.04%). Therefore, it can be concluded that the IPSO algorithm is effective for solving path planning problems.

The plot (a) and plot (b) of Fig. 6 show the convergence behavior of the IPSO, GA and ACO algorithms' shortest transportation time in Case 1 and Case 2, respectively, as the number of iterations increases. We can conclude that due to the influence of the positive feedback mechanism, the ACO algorithm has a faster convergence rate in the early stage of the iteration. However, in the later iterations, especially when

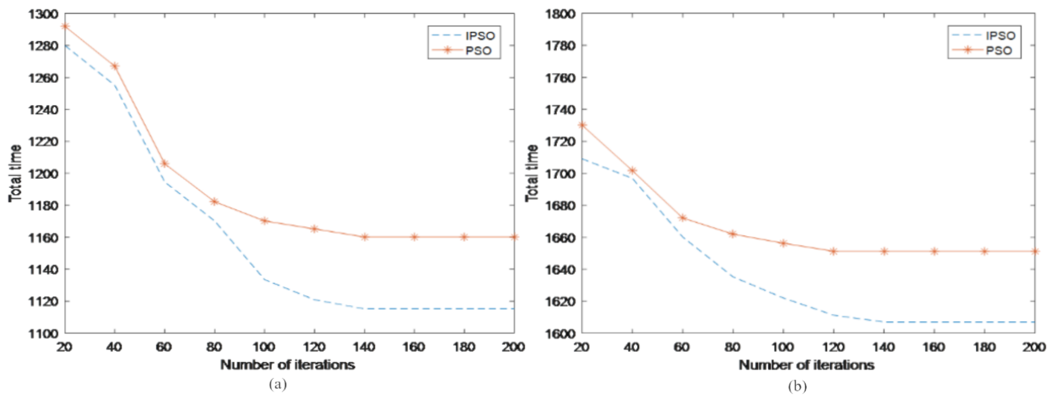


FIGURE 5. The convergence curve between the IPSO algorithm and the PSO algorithm in case 1 and case 2. plot(a) shows the change of the convergence curve of the PSO and IPSO algorithms when n = 10 in Case 1. plot(b) shows the change of the convergence curve of the PSO and IPSO algorithms when n = 25 in case 2.

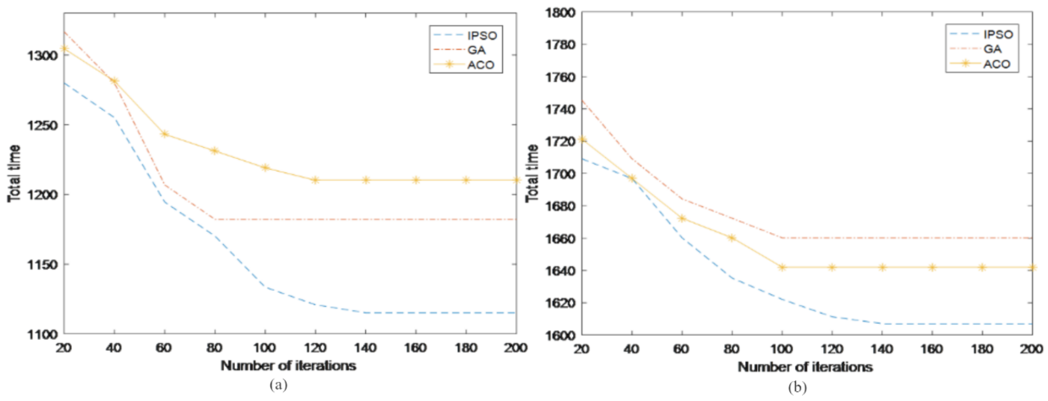


FIGURE 6. The convergence curve between the IPSO algorithm, ACO algorithm and GA algorithm in case 1 and case 2. plot(a) shows the change of the convergence curve of the PSO, ACO, GA algorithms when n = 10 in case 1. plot(b) shows the change of the convergence curve of the IPSO, ACO, GA algorithms when n = 25 in case 2.

TABLE 7. Computational results for the RPI values of IPSO and PSO algorithms.

Case	IPSO	GA	ACO
1	**3.43**	7.19	8.68
2	**3.16**	4.89	5.27
Mean	**3.30**	6.04	6.98

the number of iterations is 80-100, the convergence rate is significantly reduced, and it falls into a local optimum. It has converged prematurely. Similarly, as the number of iterations increases, the GA algorithm also shows its limitations. When the number of iterations is 100, it converges prematurely. And its ability to converge to the global optimal solution is far worse than the IPSO algorithm. Observing the IPSO algorithm, because its crossover operation is based on selecting the closest machine tool, its results are better than the other two algorithms in most iteration. Due to the design of the mutation mechanism, it also quickly jumps out of the local optimum and has a strong local mining capacity, avoiding the condition of premature convergence. The final result is also better than the GA algorithm and ACO algorithm. The experimental results further prove the effectiveness of the algorithm.

3) COMPARISON OF THE IPSO, PSO, ACO AND GA ALGORITHMS

Table 8 shows the shortest transportation time (per run), average transportation time and standard deviation of each algorithm when each algorithm is run 25 times. It can be seen that the results obtained by the IPSO algorithm are more clustered, and most of the results are smaller than other algorithms. The difference between the maximum and minimum values of this algorithm is also the smallest compared to other algorithms.

The average value obtained by the IPSO algorithm is the best and the standard deviation is the smallest. It shows that the calculation result of the improved algorithm is relatively stable and the fluctuation is small. The experimental results show that the IPSO algorithm can better deal with the problem of AGV path planning.

The bar graph of the shortest AGV driving distance obtained after running each algorithm 25 times is shown in Fig. 7. When 10 machine tools call AGV, the distance obtained by the IPSO algorithm is the shortest. The same

TABLE 8. **Results of 25 times of various algorithms.**

time	Case 1				Case 2			
	IPSO	PSO	ACO	GA	IPSO	PSO	ACO	GA
1	1156.9	1198.6	1162.8	1239.9	1691.7	1728.7	1656.6	1677
2	1139.8	**1160.2**	1156.4	1196.4	1686.9	1661.8	1727.2	1698.8
3	**1115.2**	1165.5	1187.7	**1160.6**	1679.5	1657.2	**1641.7**	1681.7
4	1148.7	1189.6	1216.9	1229.0	1655.1	1701.9	1711.6	1670.7
5	1187.2	1177.7	1247.5	1218.5	1646.7	1697.1	1731.5	1683.1
6	1153.9	1217.2	1257.0	1166.1	1687.1	1659.7	1658.8	1741.7
7	1134.7	1176.7	1251.5	1193.2	1639.2	1707.1	1717.0	1671.5
8	1179.1	1190.5	1208.7	1183.4	1651.9	1697.8	1651.5	1767.1
9	1126.5	1231.2	1170.5	1190.0	1680.5	1720.5	1721.6	1676.6
10	1162.4	1238.7	1178.3	1217.0	1656.7	1676.5	1646.9	1676.6
11	1145.2	1228.0	1190.0	1214.7	1666.1	1710.7	1706.8	1668.9
12	1156.9	1237.6	1240.8	1189.2	1691.7	1721.8	1661.7	1697.8
13	1159.2	1216.2	1236.7	1187.6	1659.4	**1651.5**	1742.1	1679.8
14	1176.6	1185.2	1180.2	1182.6	1642.0	1679.3	1650.2	**1660.2**
15	1136.2	1247.0	1218.3	1180.6	1684.2	1726.8	1735.9	1711.9
16	1127.2	1209.4	1200.3	1190.6	1671.6	1672.0	1758.1	1684.8
17	1123.2	1180.7	1197.4	1194.8	1687.6	1681.7	1751.7	1666.7
18	1161.7	1197.1	1241.0	1199.1	1632.1	1664.7	1691.7	1704.7
19	1123.0	1220.2	1226.2	1225.2	1676.9	1731.6	1675.8	1661.7
20	1164.6	1211.9	1239.8	1222.2	1681.1	1674.7	1746.7	1686.7
21	1175.2	1192.7	1215.8	1235.6	1663.6	1717.1	1689.7	1698.5
22	1168.7	1233.1	1236.2	1175.8	**1608.8**	1738.7	1644.2	1669.4
23	1140.7	1237.6	1234.0	1179.7	1667.2	1723.8	1681.1	1664.2
24	1182.6	1216.7	**1150.2**	1212.2	1663.9	1711.9	1671.2	1695.0
25	1190.5	1225.7	1255	1230.9	1671.0	1667.2	1668.6	1693.6
AVG	**1153.44**	1207.40	1211.97	1200.60	**1665.70**	1695.27	1693.60	1687.55
SD	**21.79**	24.85	32.74	22.33	**20.76**	27.30	38.23	24.73

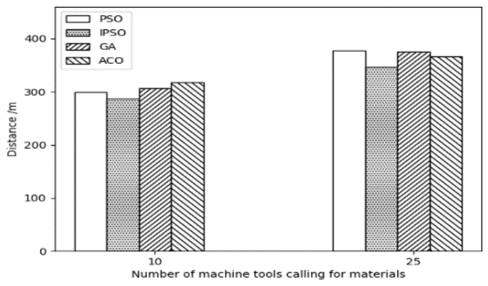


FIGURE 7. **Comparison of the optimal results of each algorithm.**

result is also true for 25 machines. It can be concluded from the data that the IPSO algorithm can always achieve excellent results. Experimental data proves the effectiveness of the improved algorithm.

Table 9 shows the shortest material transportation time obtained by each algorithm when the number of machine tools requiring materials is 10 and 25, and the running time of the algorithm. In order to improve the reliability, each algorithm is run 25 times, and the shortest transportation time result is selected as the experimental result for comparison. It can be seen that in terms of the shortest material transportation time, whether it is Case1 or Case2, the results obtained by the IPSO algorithm are always better than the other three algorithms. Although there is no advantage in the running time of the algorithm, for the actual production workshop, the running time of the algorithm can be ignored. Therefore, the IPSO algorithm is very effective for solving the AGV path planning problem.

Finally, we examine the effectiveness of IPSO algorithm using statistical analysis. We compare the experimental results through analysis of variance (ANOVA). Fig.8 shows the least significant difference (LSD) intervals (at the 95% confidence level) for the four algorithms. In Fig.8, by comparing the results, it can be seen that the performance of IPSO is statistically significantly better than other algorithms. Therefore, it can be proved that in this experiment, IPSO is very effective in solving the problem of AGV path planning.

TABLE 9. Comparison of results of various algorithms.

Algorithms	Statistics category	Number of tasks	
		10 machine tools	25 machine tools
GA	Transportation time	1160.6	1660.2
	Calculating time	1.58	2.30
ACO	Transportation time	1150.2	1641.7
	Calculating time	2.34	2.56
PSO	Transportation time	1160.2	1651.5
	Calculating time	1.76	2.05
IPSO	Transportation time	**1115.2**	**1608.8**
	Calculating time	**2.37**	**2.54**

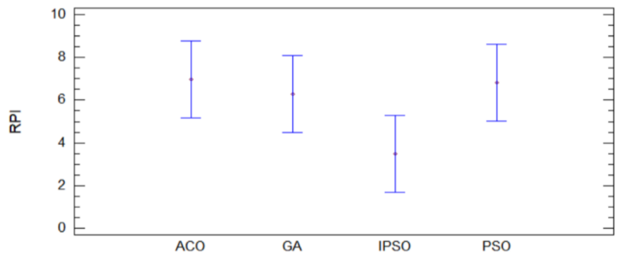


FIGURE 8. Means plots of 95% LSD confidence intervals for the algorithms.

TABLE 10. The optimization results.

Items	Best sequences
Case 1	1,5,18,14,20,9,12,2,24,11
Case 2	8,25,3,24,6,18,1 ,15,12,20,26,9 ,2,29,14,16,7,23,4,10,5,22,30,26,17

4) THE OPTIMAL PATH OF THE IPSO ALGORITHM

The two cases of Case1 and Case2 are run 25 times respectively, and the IPSO algorithm can obtain the optimal sequence of AGV transporting materials to the machine tool. The results are shown in Table 10.

From the above figures or tables, we can see that the results obtained by the IPSO algorithm are significantly better than the PSO, GA and ACO algorithms. Compared with the other three algorithms, this improved algorithm is not easy to fall into the local optimum, and can obtain the shortest transportation time. Therefore, the IPSO algorithm is effective for improving the efficiency of workshop material distribution.

VI. CONCLUSION

This paper proposes an effective model with the shortest running time of AGV as the objective function. Then the IPSO algorithm is used to solve the workshop AGV path planning problem. The IPSO algorithm has the following improvements:

1) According to the characteristics of the AGV problem, a new coding method is proposed.

2) Realize particle position update based on the crossover operation.

3) Propose mutation operation to enhance local search ability and help jump out of local optimum.

The experimental results show that the algorithm can effectively solve the AGV path optimization problem, with fast convergence speed and not easy to fall into the local optimum.

In fact, in the actual production workshop, there will be multiple AGVs transporting materials for multiple production lines at the same time, and the specific number of AGVs and energy consumption will also affect scheduling. The scheduling and configuration of multiple AGVs is worthy of further study.

REFERENCES

[1] J. Li, Z.-M. Liu, C. Li, and Z. Zheng, "Improved artificial immune system algorithm for Type-2 fuzzy flexible job shop scheduling problem," *IEEE Trans. Fuzzy Syst.*, early access, Aug. 12, 2020, doi: 10.1109/TFUZZ.2020.3016225.

[2] W.-M. Chow, "Development of an automated storage and retrieval system for manufacturing assembly lines," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Apr. 1986, pp. 490–495.

[3] C. Oboth, R. Batta, and M. Karwan, "Dynamic conflict-free routing of automated guided vehicles," *Int. J. Prod. Res.*, vol. 37, no. 9, pp. 2003–2030, Jun. 1999.

[4] L. Qiu, W.-J. Hsu, S.-Y. Huang, and H. Wang, "Scheduling and routing algorithms for AGVs: A survey," *Int. J. Prod. Res.*, vol. 40, no. 3, pp. 745–760, Jan. 2002.

[5] L. Z. Du, S. F. Ke, Z. Wang, J. Tao, L. Q. Yu, and H. J. Li, "Research on multi-load AGV path planning of weaving workshop based on time priority," *Math. Biosci. Eng*, vol. 16, no. 4, pp. 2277–2292, 2019.

[6] J. Z. Huang and Y. W. Cen, "A path-planning algorithm for AGV based on the combination between ant colony algorithm and immune regulation," in *Proc. Adv. Mater. Res.*, vol. 422. Cham, Switzerland: Trans Tech Publications, 2012, pp. 3–9.

[7] C. He and J. Mao, "AGV optimal path planning based on improved ant colony algorithm," in *Proc. MATEC Web Conf.*, 2018, vol. 232, no. 35, Art. no. 03052.

[8] C. J. Ding, X. Wang, and Y. B. Feng, "AGV path planning based on PSO algorithm," *Transducer Microsyst. Technol.*, vol. 39, no. 8, pp. 123–126, 2020.

[9] J. Luo, C. Wu, B. Li, H. Yin, and Q. Zhang, "Modeling and optimization of RGV system based on improved QPSO," *Comput. Integr. Manuf. Syst.*, vol. 17, no. 2, p. 321, 2011.

[10] I. Minis and A. Tatarakis, "Stochastic single vehicle routing problem with delivery and pick up and a predefined customer sequence," *Eur. J. Oper. Res.*, vol. 213, no. 1, pp. 37–51, Aug. 2011.

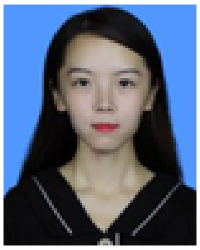
[11] Y. M. Guo, Z. Y. Wu, Z. W. Zhang, L. H. Wu, and B. H. Zhang, "Energy-efficient path planning for single agv in flexible manufacturing workshop," *Modular Mach. Tool Autom. Manuf. Techn.*, no. 10, pp. 181–184, Oct. 2020.

[12] N. Smolic-Rocak, S. Bogdan, Z. Kovacic, and T. Petrovic, "Time windows based dynamic routing in multi-AGV systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 1, pp. 151–155, Jan. 2010.
[13] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw. (ICNN)*, vol. 4, 1995, pp. 1942–1948.
[14] Q.-K. Pan, L. Wang, M. F. Tasgetiren, and B.-H. Zhao, "A hybrid discrete particle swarm optimization algorithm for the no-wait flow shop scheduling problem with makespan criterion," *Int. J. Adv. Manuf. Technol.*, vol. 38, nos. 3–4, pp. 337–347, Aug. 2008.
[15] Y. Zhang, D.-W. Gong, and Z. Ding, "A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch," *Inf. Sci.*, vol. 192, pp. 213–227, Jun. 2012.
[16] J. Li, Q. Pan, and K. Mao, "Hybrid particle swarm optimization for hybrid flowshop scheduling problem with maintenance activities," *Sci. World J.*, vol. 2014, Jan. 2014, Art. no. 596850.
[17] S. Chatterjee, S. Sarkar, S. Hore, N. Dey, A. S. Ashour, and V. E. Balas, "Particle swarm optimization trained neural network for structural failure prediction of multistoried RC buildings," *Neural Comput. Appl.*, vol. 28, no. 8, pp. 2005–2016, Aug. 2017.
[18] T. Yuejun, "Improvement of particle swarm algorithm and its application in power system," *Electron. Test*, no. 19, pp. 47–48, Oct. 2015.
[19] J. Li and F. Xie, "Self-organized criticality of molecular biology and thermodynamic analysis of life system based on optimized particle swarm algorithm," *Cellular Mol. Biol.*, vol. 66, no. 2, pp. 177–192, 2020.
[20] F.-S. Hsieh and Y.-H. Guo, "A discrete cooperatively coevolving particle swarm optimization algorithm for combinatorial double auctions," *Int. J. Speech Technol.*, vol. 49, no. 11, pp. 3845–3863, Nov. 2019.
[21] Y. Li, H. Soleimani, and M. Zohal, "An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives," *J. Cleaner Prod.*, vol. 227, pp. 1161–1172, Aug. 2019.
[22] H. Ding and X. Gu, "Hybrid of human learning optimization algorithm and particle swarm optimization algorithm with scheduling strategies for the flexible job-shop scheduling problem," *Neurocomputing*, vol. 414, pp. 313–332, Nov. 2020.
[23] W. Jun-Wei and W. Ding-Wei, "Experiments and analysis on inertia weight in particle swarm optimization," *J. Syst. Eng.*, vol. 20, no. 2, pp. 194–198, Apr. 2005.
[24] Q. Zhenkun, "A new method of weighted traveling salesman problem based on discrete particle swarm optimization," *Comput. Appl. Softw.*, vol. 36, no. 1, pp. 127–131, 2019.
[25] F. Guojiang and Wang, "An improved velocity mutation particle swarm optimizer," *Comput. Eng. Appl.*, no. 13, pp. 48–50, May 2006.
[26] Z. Jian-Ke, L. San-Yang, and Z. Xiao-Qing, "Improved particle swarm optimization," *Comput. Eng. Des.*, vol. 28, no. 17, pp. 4215–4219, Sep. 2007.
[27] W. Zhi-Gang, "A modified particle swarm optimization," *J. Harbin Univ. Commerce, Natural Sci. Ed.*, vol. 15, no. 2, pp. 31–34, Apr. 2010.
[28] Y. Mingrang, C. Yun, and Z. Zhigang, "A discrete version of particle swarm optimization for multi-objective flexible job-shop scheduling problems," *Manuf. Technol. Mach. Tool*, no. 1, pp. 159–165, Jan. 2019.
[29] D. Wei-Lin and H. U. Gui-Wu, "A particle swarm algorithm for discrete optimization problem," *Comput. Technol. Develop.*, vol. 22, no. 5, pp. 116–119, 2012.

TAO QIUYUN was born in Liaocheng, Shandong, China, in 1995. She received the bachelor's degree from the School of Computer Science, Liaocheng University, in 2014. She is currently pursuing the master's degree with the Software Engineering Department, Liaocheng University. Her main research interests include intelligent optimization and scheduling.

SANG HONGYAN (Member, IEEE) received the M.S. degree from the School of Computer Science, Liaocheng University, Liaocheng, China, in 2010, and the Ph.D. degree in industrial engineering from the Huazhong University of Science Technology, Wuhan, China, in 2013. Since 2003, she has been with the School of Computer Science, Liaocheng University, where she became an Associate Professor, in 2016. She has authored more than 60 refereed articles. Her current research interests include intelligent optimization and scheduling.

GUO HENGWEI received the bachelor's and master's degrees in intelligent optimization algorithms and applications from the School of Computer Science, Liaocheng University, in 2015.



WANG PING was born in Liaocheng, Shandong, China in 1994. She is currently pursuing the M.S. degree in computer science with Liaocheng University. She has published articles. Her main research interest includes intelligent optimization. She participated in the 2019 Intelligent Simulation Optimization and Scheduling Academic Conference, in 2019, and received the third prize for symposium papers. She also participated in the 2019 5th International Conference on Control Science and Systems Engineering (ICCSSE) international conferences.