Research on Intelligent Book Sorting AGV Task Scheduling Based on Firefly Algorithm

Yijun Tang, Hongyuan Wang*

(School of Business Administration, Liaoning University of Engineering and Technology, Huludao, Liaoning 125100, China)
2250752366@qq.com, 26605805652@qq.com*

Abstract. Addressing the challenges of task scheduling algorithms for book sorting AGVs in vast and intricate settings, a refined task scheduling model tailored for book sorting AGV systems has been devised, leveraging the colored traveling salesman problem (CTSP). Additionally, a firefly algorithm is introduced for seamless coordination. Initially, the problem is outlined as a CTSP. Subsequently, the firefly algorithm is employed to tackle it effectively. The strategy is then validated and analyzed through rigorous case studies, ten in total. The findings reveal that this strategy excels in efficiency, accuracy, and consistently provides optimized solutions across varying task scales.

Keywords: Firefly algorithm; Task scheduling; Book sorting; Colored traveling salesman

1 INTRODUCTION

Foreign libraries, such as the University of Hamburg in Germany and Osaka City University in Japan, have employed automated guided vehicles (AGVs) for book sorting. The Hamburg University Library installed an induction strip rail system in its reading room, facilitating AGV loading/unloading at designated points and navigation between floors via elevator entrances[1]. Osaka City University Library, alternatively, used magnetic guidance lines on the ground for AGV navigation[2]. Both universities achieved efficient book sorting with AGVs[3].

Despite a late start, China's AGV sorting technology has experienced rapid development. In 2019, Shenzhen Bao'an Library introduced 32 AGV systems, 28 for sorting and 4 for transportation, achieving a remarkable sorting speed of 2,000 books per hour, surpassing manual efficiency[4]. Since then, the technology has gained widespread adoption, with libraries in Tianjin Binhai, Guangzhou Nansha, Guizhou, Shantou, and the China-Singapore Tianjin Eco-City Library also adopting intelligent sorting AGVs[5].

In the context of domestic libraries' sorting procedures, current AGV task scheduling primarily aims to enhance individual task efficiency, neglecting the broader impact on system-level efficiency as task volumes escalate[6]. This article addresses this challenge by introducing a novel approach combining the firefly algorithm with a climbing operation to model and solve the colored traveling salesman problem[7]. This method aims to enhance algorithm performance[8]. When compared to various alternative algorithms, the proposed method exhibits superior performance in terms of time consumption, average path, and total path[9], validating its practical efficacy in book sorting AGV scheduling[10].

2 SORTING AGV WORKING MODE

Libraries employ AGVs for book sorting, incorporating functions like identification, gripping, sorting, and transportation. Nowadays, this technology is gradually being integrated into library operations. The workflow, depicted in Figure 1, primarily encompasses the following steps:

Step1： Assign one or more tasks to a sorting AGV based on task requirements.

Step 2: The sorting AGV autonomously plans the route based on the task sequence, efficiently retrieves or replaces books on the shelf, and proceeds to the next shelf.

Step 3: The sorting AGV efficiently completes tasks in the queue, adhering strictly to the pre-determined task sequence.

Step 4: Upon task completion, the sorting AGV moves to the sorting table, removes all books, and either places them on the table or loads the next batch destined for the bookshelf.

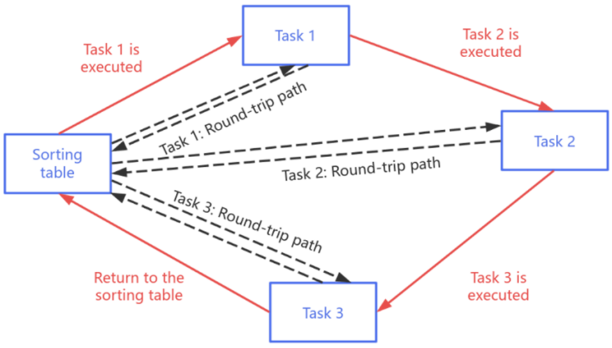

Fig. 1. Task sequence diagram of the sorting AGV

3 COLOURING OF TRAVELER ISSUES

Assuming that there are m travelers and n cities, $m,n \in N, m < n,$ can model this problem using the complete graph $G(V,E)$. In this graph, $V = \{0,1,\ldots,n-1\}$ is the set of cities, E is the set of edges, W_{ij} denotes the edge $\langle i,j \rangle$ weights,

$Z_m = \{1,2,\ldots,m\}$ is the set of travelers, $X_{ijk}\left(i \neq j, j \in V, k \in Z_m\right)$ denotes a Boolean variable indicating whether the k traveler passes through the edge $\langle i, j\rangle$. U is the set of shared cities and V_k is the set of exclusive cities for the first k traveler is also defined separately.

The objective function of the colouring traveler problem is defined as follows:

$$\min f = \sum_{k=1}^{m}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} W_{ij} X_{ijk} \tag{1}$$

Restrictions on colouring traveler issues include:

$$\sum_{i=1}^{n-1} X_{0ik} = \sum_{i=1}^{n=1} X_{i0k} = 1, k \in Z_m \tag{2}$$

$$\sum_{i}\sum_{j} X_{ijk} = \sum_{i}\sum_{j} X_{jik} = 0, i \in V_k, j \in V \setminus \{U \cup V_k\}, k \in Z_m \tag{3}$$

$$\sum_{i}\sum_{j} X_{ijl} = \sum_{i}\sum_{j} X_{jil} = 0, i \neq j, k \neq l, i \in V_k, j \in V, l \in Z_m \tag{4}$$

$$\sum_{j=0}^{n-1}\sum_{k=1}^{m} X_{ijk} = \sum_{j=0}^{n-1}\sum_{k=1}^{m} X_{jik} = 1, j \neq i, i \in V \setminus \{0\} \tag{5}$$

$$\sum_{l} X_{jlk} = \sum_{i} X_{ijk}, i \neq j \neq l, j \in U, i, j \in \left(V_k \cup U\right) \tag{6}$$

where Eq. (2) indicates that the starting and ending cities of each traveler must be the starting cities of their respective choices; Eq. (3) indicates that the k th traveller cannot visit the exclusive city of another traveller or depart from the exclusive city of another traveller; Eq. (4) indicates that no other traveller can visit the exclusive city of the k th traveller or depart from the exclusive city of the k th traveller; Eq. (5) indicates that cities other than the starting city can only be visited once; Equation (6) indicates that each traveller must visit and exit the shared set of cities the same number of times.

In a real library setting, the challenge of scheduling multiple AGVs involves efficiently allocating various sorting tasks to these vehicles while determining the order and quantity of tasks for each. This becomes especially intricate when dealing with a significant number of tasks and AGVs. To address this, the scheduling issue can be transformed into a full-colour traveller problem and analyzed using a model, as depicted in Fig. 2.

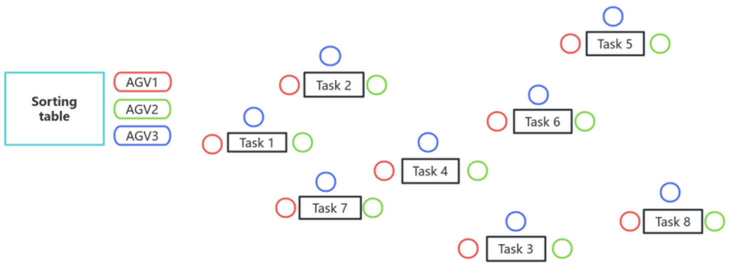

Fig. 2. Colour representation of the colouring traveller model for sorting individuals and objects

4 SCHEDULING MODEL SOLUTION BASED ON FIREFLY ALGORITHM

4.1 Firefly Algorithm-Based Solution

Coding Design. Fig. 3 demonstrates dual chromosome coding in the colouring traveler model with two examples. Task chromosome sequences indicate task order, while sorting AGV sequences represent the AGV accessing each task. Each AGV's first three nodes represent its starting task, which must be added to its path. In Examples 1 and 2, AGV1 follows 1-9-7-1, AGV2 follows 2-6-8-2, and AGV3 follows 3-5-4-3. Despite potential similarities in encoded paths, crossover and mutation require reverification of task-encoding and travelling salesman-encoding relationships, making this coding scheme less efficient.

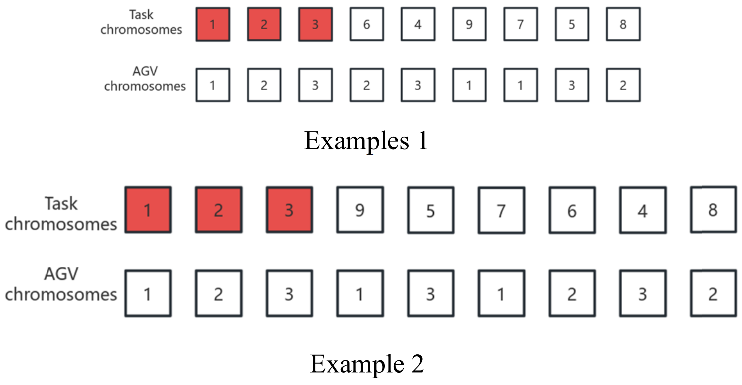

Fig. 3. Dual chromosome coding

In direct path coding, each AGV is assigned a unique task access sequence. By appending the corresponding starting task to each AGV's sequence, we derive the paths as depicted earlier. Precisely, See Figure 4AGV1 follows 1-9-7-1, AGV2 traverses 2-6-8-2, and AGV3 adheres to 3-5-4-3. Given the one-to-one correspondence between encoding and decoding, this method is employed in this paper.

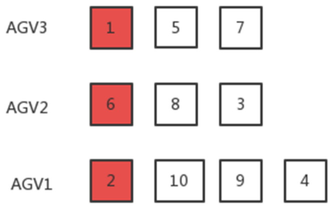

Fig. 4. Direct encoding

Brightness and Attractiveness Calculations. The objective of this paper is to minimize the overall path traversed by sorting AGVs between the sorting table and designated tasks. The attraction between fireflies is directly proportional to the luminance of the brighter firefly and the distance separating them. In the firefly algorithm, fitness function values are reflected by firefly luminance. Consequently, the fitness function of the algorithm can be represented by equation (7):

$$F = 1/(1 + f_{\max}),\ f_{\max}(f_1, f_2, \ldots, f_m) \tag{7}$$

where $f_{\max}$ represents the largest traveller path value, f_i represents the path value of traveller i and $i \in Z$. The larger adaptation value F indicates the better quality of the solution, which indicates the better quality of the solution, and also F represents the brightness of the firefly.

To calculate the distance between fireflies, equation (8) is used:

$$r = 10 * d / n \tag{8}$$

where d denotes the number of edges where the two firefly path species differ, and n denotes the number of tasks.

Finally, Eq. (9) was used to calculate the attraction between two fireflies at a distance of r In this equation, β_0 is the brightness value of the brighter firefly and γ is the light intensity absorption coefficient.

$$\beta(\mathrm{r}) = \beta_0 e^{-\gamma r^2} \tag{9}$$

Update Operations. The firefly renewal operation involves attracting and relocating fireflies. This paper employed a reverse mutation strategy to update fireflies, as depicted in Fig. 5. Initially, directly encoded task sequences of each AGV are combined to form an overall task sequence. Subsequently, two endpoints are randomly chosen, and an inversion operation is performed on the task sequence between these endpoints. Lastly, it's crucial to guarantee that the color information of the task sequence is simplified to the directly encoded individual.

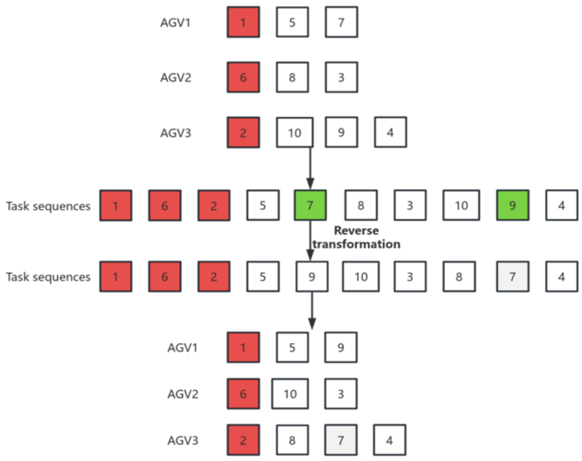

Fig. 5. Reverse mutation operation

Climbing Operations. Enhanced in this paper, the firefly algorithm globally searches while locally, hill-climbing boosts its search capabilities.See Figure 6 ,this hybrid approach ensures efficient local searches on all individuals once the next generation population is saturated, thus ensuring rapid convergence and averting premature local optimization. Hill-climbing randomly selects two tasks in a sorting AGV sequence, swaps them, and checks for path optimization. Negative optimization leads to swapping back tasks, restoring the original sequence.

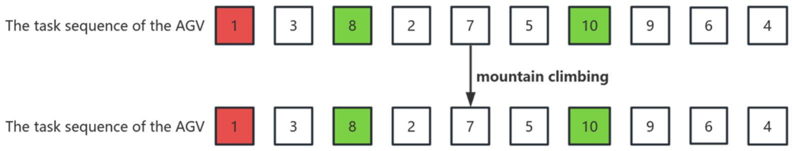

Fig. 6. Climbing operation

4.2 Solution Flow Based on Firefly Algorithm

Step 1: Optimize various parameters of the firefly algorithm and construct the initial population based on the sorting AGV and task information.

Step 2: Iteratively operate on the current population until the termination condition is met. In each iteration, preferentially select the brightest firefly and carry it over to the next generation.

Step 3: Assess brightness of individuals and their attraction to the brightest individual. If attraction surpasses individual brightness, fixed-point reverse mutation guides the individual towards the brightest, ensuring its inclusion in the next generation. Conversely, if attraction falls below brightness, random reverse mutation occurs, greedily selecting both original and mutated individuals for the next generation.

Step 4: Optimize each AGV task sequence for each individual in the updated population through hill-climbing techniques.

Step 5: Once the termination condition is met, present the optimal individuals from the final generation as the algorithm's conclusive solution.

5 EXPERIMENTS AND ANALYSIS OF RESULTS

5.1 Experimental Setup

Selected from the TSPLIB problem library, public cases were obtained from (https://pan.baidu.com/s/10hV2u5cdGLqTiFGh0Gd0fQ%2C) including task location and color assignment information files. Conducted in a C++ environment, the experiments utilized VS2013 as the running platform. To replicate real-world conditions, all test cases were randomly generated across a 130m x 246m rectangular raster library electronic map, with tasks randomly assigned to sorting-capable AGVs.

5.2 Comparison of Algorithms in Sorting Scenarios

To assess the Firefly algorithm's feasibility and superiority in addressing the coloring traveler problem, we plan a series of experiments comparing it with the algorithm presented in this paper. Such as Table 1 and Table 2.We aim to comprehensively explore the impact of two distinct scenarios on time consumption, average path length, and overall path length.

Small-Scale Sorting AGV Task Scheduling.

Table 1. Experimental results of task scheduling for small-scale sorting robots

case (law)	Number of machines	Number of tasks	GA			FA		
			total path	Average path	Time Consumption (ms)	total path	Average path	Time Consumption (ms)
1	2	20	1372	686	175	1360	680	55
2	4	40	2976	744	304	2802	700.5	105
3	7	70	5174	739.14	502	4696	670.86	185
4	10	100	8314	831.4	705	7160	716	260
5	15	150	12024	801.6	1080	10766	717.73	405
6	20	200	15692	784.6	1493	14162	708.1	544
7	25	250	19006	760.24	1936	17758	710.32	706
8	30	300	24356	811.87	2663	20840	694.67	871
9	40	400	31758	793.95	3418	28666	716.65	1224
10	50	500	40236	804.72	4642	35514	710.28	1547

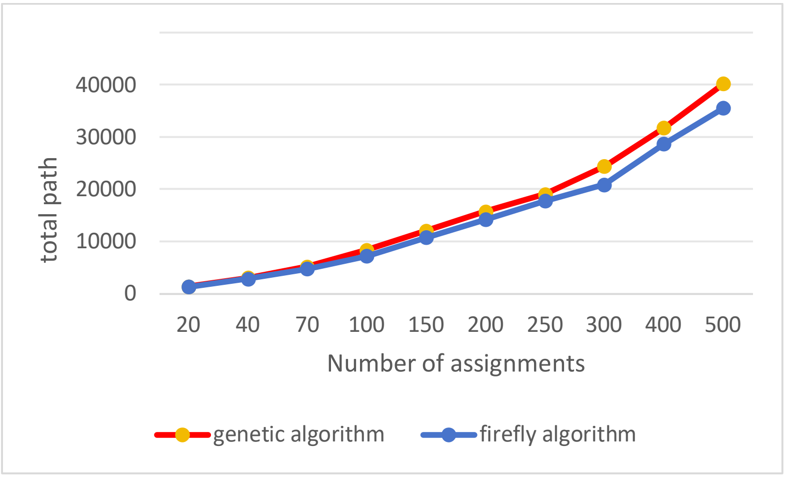

Fig. 7. Comparison of total paths

Fig. 7 reveals that in the small-scale sorting scenario with 10 cases, the firefly algorithm generates significantly fewer AGV paths compared to the genetic algorithm.

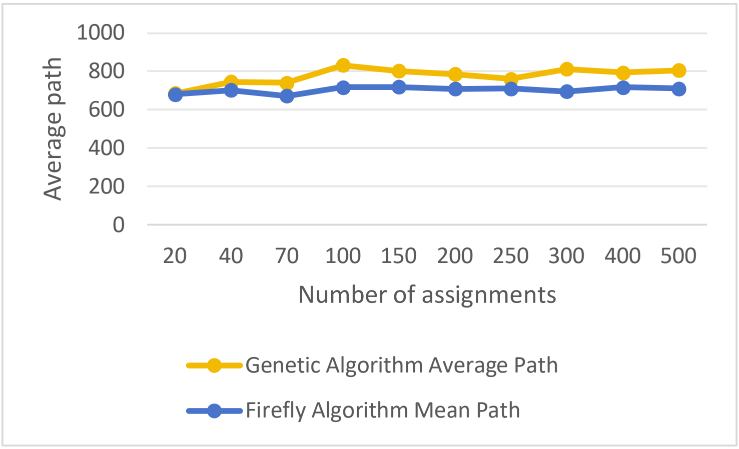

Fig. 8. Comparison of average paths

Figure 8 compares the average paths of sorting AGVs in 10 cases, using both the firefly and genetic algorithms. The firefly algorithm significantly outperforms the genetic algorithm in terms of average path length.

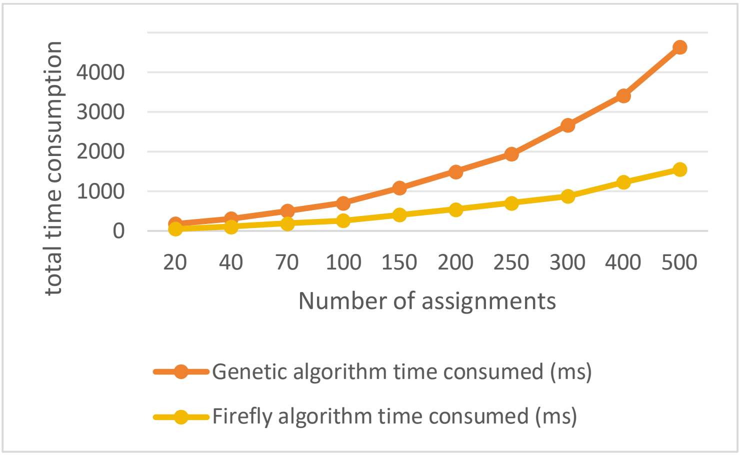

Fig. 9. Comparison of total elapsed time

Figure 9 compares the total running time of two algorithms across 10 cases. Evidently, the firefly algorithm boasts the shortest simulation time. As tasks and sorting AGVs multiply, both algorithms' running time rises, but the genetic algorithm consistently requires significantly more time than the firefly algorithm.

Large-Scale Sorting AGV Task Scheduling.

Table 2. Experimental results of task scheduling for large-scale sorting AGVs

case (law)	Number of machines	Number of assignments	GA			FA		
			total path	Average path	Time Consumption (ms)	total path	Average path	Time Consumption (ms)
1	50	500	40130	802.6	4798	35716	714.32	1617
2	100	1000	79388	793.88	9805	71132	711.32	3222
3	150	1500	119102	794.01	14681	106760	711.73	4776
4	200	2000	158902	794.51	20309	141936	709.68	6504
5	250	2500	198794	795.18	25294	176678	706.71	8377
6	300	3000	240208	800.69	29905	212756	709.19	9992
7	350	3500	279292	797.98	34699	248266	709.33	11718
8	400	4000	320380	800.95	39501	283494	708.74	13277
9	450	4500	359316	798.48	44650	318614	708.03	14868
10	500	5000	399318	798.64	49637	353726	707.45	16495

To enhance computational efficiency, a group parallelism strategy assigns 500 sorting tasks to 50 AGVs simultaneously. This problem can be formulated as a full-color traveler problem with 500 cities and 50 travelers.Such as Table 2.

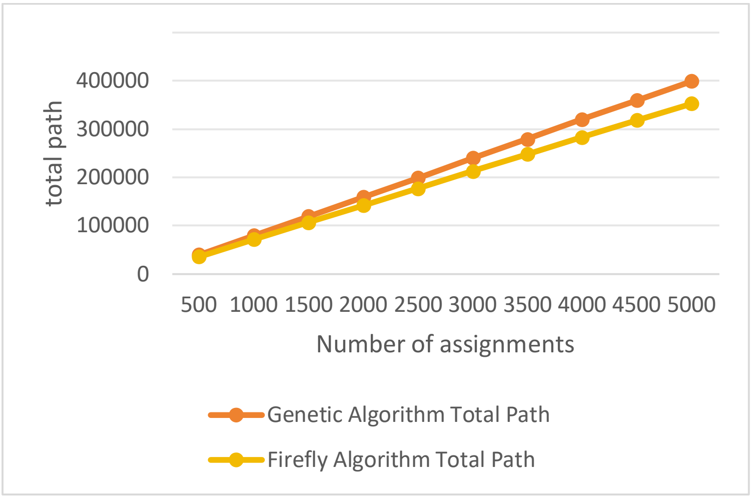

Fig. 10. Comparison of total paths

Figure 10 reveals that the firefly algorithm maintains stability in large-scale sorting, whereas the genetic algorithm experiences significant fluctuations. This underscores the firefly algorithm's superior robustness and stability in tackling large-scale sorting challenges.

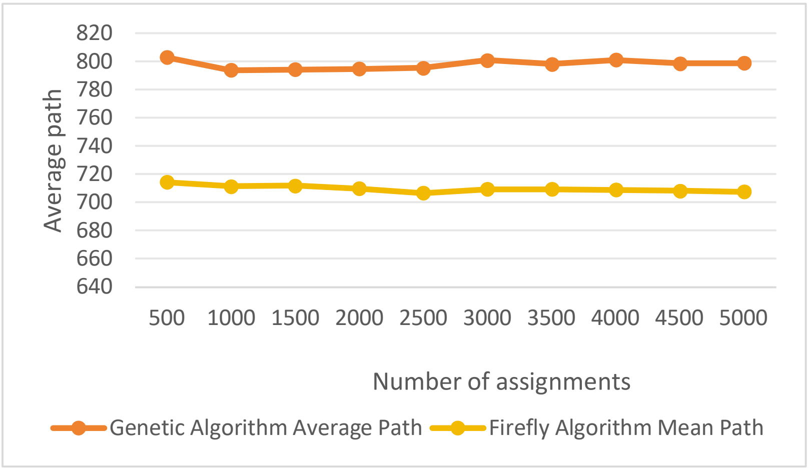

Fig. 11. Comparison of average paths

In Fig. 11, the average AGV paths for both the genetic and firefly algorithms remain stable across various task sizes in large-scale sorting scenarios. Notably, the firefly algorithm outperforms the genetic algorithm in terms of shorter average sorting AGV paths.

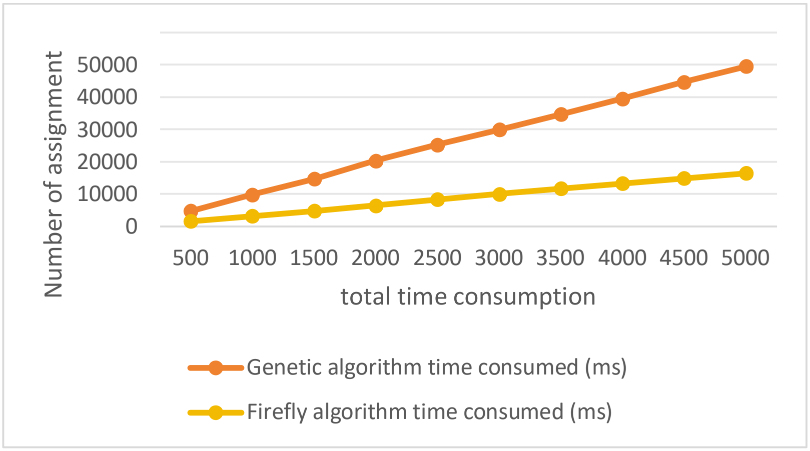

Fig. 12. Comparison of total time spent

Fig. 12 reveals that in large-scale sorting scenarios, the AGVs utilizing the firefly algorithm achieve the fastest total runtime. Conversely, the execution time of AGVs employing both the genetic and firefly algorithms increases proportionally with the number of sorting AGVs.

6 CONCLUSION

This paper introduces an enhanced task scheduling algorithm tailored for library sorting AGVs. By prioritizing total path minimization, the algorithm meticulously models sorting AGVs with task data, from small to large-scale, maintaining the integrity of the task sequence. Leveraging the coloring traveler's proficiency in handling dense, high-volume tasks, the firefly algorithm incorporates group scheduling and hill-climbing techniques, significantly enhancing convergence speed and solution quality. To assess its performance, the algorithm was tested against 10 cases and benchmarked against other methods. Results demonstrate superior time efficiency, average path length, and overall path optimization, significantly improving AGV sorting efficiency.

AUTHOR INTRODUCTION

Author Resume: Yijun Tang(1974-), Male, Master, Associate Professor, Master Supervisor, with research interests in production modelling and simulation, production scheduling, etc.2250752366@qq.com;

HongyuanWang (1999-), Female, Master Candidate of Class 2022, with research interests in production modelling and simulation.26605805652@qq.com.

REFERENCES

1. Li J ,Meng X ,Dai X .Collision-free Scheduling of Multi-bridge Machining Systems: a Colored Travelling Salesman Problem-based Approach[J].IEEE/CAA Journal of Automatica Sinica,2018,5(01):139-147.
2. Dong Guoming,She Chunhua. Greedy dual-chromosome genetic algorithm for coordinated scheduling of multiple robots[J/OL]. Mechanical Design and Manufacturing:1-8[2023-11-05].https://doi.org/10.19356/j.cnki.1001-3997.20230724.025.
3. TAN Heyi, ZHANG Wei, MIN Bingyuan. Task scheduling of multi-sorting robots based on a hybrid tenno whisker-particle swarm algorithm[J]. Mechanical Design and Research,2022,38(03):56-59.DOI:10.13952/j.cnki.jofmdr.2022.0133.
4. Zhang Jinghui. Composition Analysis and Planning Design of Library AGV Intelligent Sorting System--Taking Shantou Library as an Example[J]. Research on Librarianship,2021(04):23-28.DOI:10.15941/j.cnki.issn1001-0424.2021.04.004.
5. Wang Jingfeng.Research on the application of AGV sorting technology in libraries[J]. Library Journal,2020,39(12):66-70.DOI:10.13663/j.cnki.lj.2020.12.009.
6. Wang YD. Application of automatic book sorting system in libraries--Taking Guangzhou Library as an example[J]. Library Forum,2015,35(05):84-88.
7. Nana Yu,Tieke Li,Wenxin Zhang et al. Multi-load AGV scheduling and path planning algorithms in automatic sorting warehouses[J/OL]. Computer Integrated Manufacturing Systems:1-22[2023-11-05].http://kns.cnki.net/kcms/detail/11.5946.TP.20220927.1021.002.html.
8. Liu Zhe. Research on the new generation of intelligent storage system for library documents under the view of intelligent library--Taking the North Library of Shenzhen Library as an example[J]. Library,2023(02):26-32.
9. LI Zhengfeng,ZHANG Dongfang,DING Qicong et al. Research on green scheduling of job shop considering AGV transport and machine speed[J/OL]. Electromechanical Engineering:1-12[2023-11-05].http://kns.cnki.net/kcms/detail/33.1088.TH.20231017.0946.004.html .
10. Tian Shuaihui,Huangfu Chengyang,Deng Xueping.A review of application scenarios, key factors and research methods for AGV scheduling optimisation[J/OL]. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition):1-12[2023-11-05].http://kns.cnki.net/kcms/detail/50.1181.N.20230608.1228.004.html .