A Rapid Search Method for Scheduling Delivery of Multiple Automated Guided Vehicles

Koki Meno (meno-k@roboken.cs.tsukuba.ac.jp)
University of Tsukuba: Tsukuba Daigaku https://orcid.org/0000-0002-5424-4460
Ayanori Yorozu
University of Tsukuba: Tsukuba Daigaku
Akihisa Ohya
University of Tsukuba: Tsukuba Daigaku

Research Article

Keywords: multiple AGV, scheduling, optimization

Posted Date: January 10th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1217880/v1

RESEARCH ARTICLE

A rapid search method for scheduling delivery of multiple automated guided vehicles

Koki Meno*, Ayanori Yorozu and Akihisa Ohya

Abstract

In this study, a method was developed to address the automated guided vehicle (AGV) transportation scheduling problem. For deliveries in factories and warehouses, it is necessary to quickly plan a feasible transportation schedule without delay within a specified time. This study focused on obtaining a transport schedule without delay from the specified time while maintaining the search for a better solution during the execution of the transport task. Accordingly, a method was developed for constructing a solution with a two-dimensional array of delivery tasks for each AGV, arranged in the order in which they are executed, as well as for searching for a schedule by performing exchange and insertion operations. For the exchange and insertion, a method that considers the connectivity between the end point of a task and the start point of the next task was adopted. To verify the effectiveness of the proposed method, numerical simulations were performed assuming an actual transportation task.

Keywords: multiple AGV; scheduling; optimization

Introduction

Recently, automated guided vehicles (AGVs) have gradually become essential in warehouses and industry[1]. The introduction of automatic guided vehicles (AGVs) for material handling and visual grasping can alleviate labor shortages and improve the efficiency of product transportation[2].

In warehouses and factories, AGVs are required by the control system to perform tasks and transport all kinds of raw materials and finished products according to specified requirements. In several cases, a transportation requirement period is specified, and the materials at the specified location must be transported to the specified location during that period. It is necessary for delivery scheduling to efficiently operate multiple AGVs while satisfying other requirements such as the transfer request period. In actual workplaces, tasks are added or changed owing to several issues or delays in the production line. Therefore, it is necessary to determine a faster finishing schedule with no delay when additions or changes occur. This study focuses on obtaining a transportation schedule that does not delay the specified time and determining a solution that completes the transportation faster.

In the delivery scheduling of multiple AGVs envisioned in this study, tasks are planned to be assigned to AGVs and in the order they are to be executed. This problem is NP-hard, as the pattern of schedules increases exponentially with the number of AGVs and tasks. Consequently, a method that adopts genetic algorithm, a meta-heuristic algorithm, has been proposed. This method that uses genetic algorithms was adopted by Lyu et al[3]. In Lyu's method, the solution was constructed as illustrated in Fig. 1, and the schedule was determined using a genetic algorithm. Lyu constructed the solution, as illustrated in Fig. 1, and adopted a genetic algorithm to determine the schedule. Each number in the task sequence column represents a task number, thus implying that the tasks are executed from the left. The number in the AGV assignment column represents the number of the AGV, which implies that the AGV is responsible for the task with the number presented above. The solution in Fig. 1 can be converted to an actual schedule, as illustrated in Fig. 2. However, the actual execution order of the tasks may differ from the execution order represented by the solution. For example, Task 8 is supposed to be executed fourth; however, it is actually executed last, and although Task 1 should be executed sixth, it is executed first. Therefore, this study considers that it is difficult to search efficiently with a solution structure similar to the one illustrated Fig. 1.

In this study, the solution is structured as a two-dimensional array of transport tasks arranged in the

*Correspondence: meno-k@roboken.cs.tsukuba.ac.jp
Graduate School of Information Science and Engineering, University of Tsukuba, 1-1-1, Tennodai, Tsukuba-shi, 305-8577 Ibaraki, Japan

order in which they are executed for each AGV, which is close to how they are actually executed. Then, a method is proposed to determine the schedule by exchanging and inserting the task positions. Accordingly, the difference between the solution and the actually executed schedule is eliminated, and a faster and more efficient schedule can be determined. Because the proposed solution structure can represent the execution order of tasks of each AGV, it is possible to consider the distance it takes to physically empty the trip between tasks. Therefore, to shorten the travel distance for transfer, we define connectivity as the proximity of the travel distance from the end point of a task to the start point of the next task. The proposed method can be used to search for a schedule more quickly and efficiently while maintaining the sequence of tasks with short turnaround time. This method is evaluated via numerical simulations.

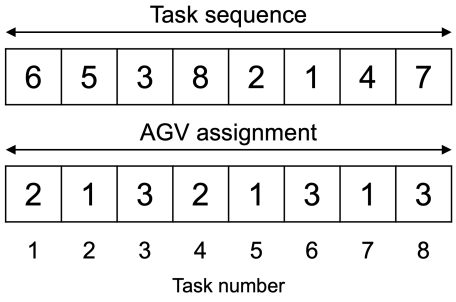


Fig. 1: Example of the structure of a Lyu[3] solution

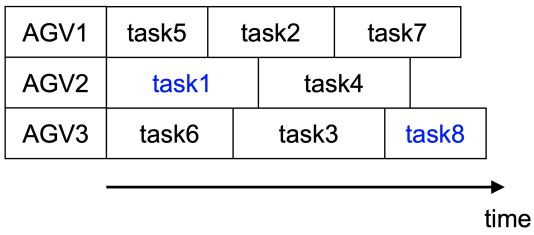


Fig. 2: The schedule represented by the solution in Fig. 1

Related work

The scheduling problem for multiple AGVs is NP-hard because the number of feasible schedules increases exponentially as the number of transport tasks, number of AGVs, and size of the warehouse increase.[4] Consequently, it is difficult to obtain an exact solution; hence, approximate solutions have been studied. A method using mathematical programming has been adopted to obtain this approximate solution.

Chen et al.[5] defined a mixed integer programming model for the transport assignment problem to minimize the total transportation cost of AGVs, and proposed a method to decompose the problem via Lagrangian relaxation and solve the problem efficiently using dynamic programming. Corréa et al.[6] proposed a hybrid method of constraint and mixed integer programming to address the problem of assigning conveyances and solving path planning without collisions between AGVs, to minimize the delay time of AGV conveyances.

However, these mathematical programming methods require a huge amounts of computation as the scale of the problem increases; hence, heuristic methods have been studied as a faster approach to obtaining approximate solutions. Deroussi et al. [7] proposed a hybrid method for iterative local search and a simulated annealing method for scheduling, considering the transfer of tasks between neighboring manufacturing machines. Saidi-Mehrabad et al. [8] proposed a two-stage ant colony algorithm to solve the job shop scheduling problem, considering the transportation times of the jobs from one machine to another. Miyamoto et al. [9] proposed the local/random search methods to address the dispatch and conflict free routing problem of capacitated AGV systems. Li et al. [10] adopted harmony search for scheduling, to minimize the standard deviation of the waiting time of the computer numerical control material buffer and the total travel distance of the AGV. Zou et al. [11] proposed a discrete artificial bee colony algorithm to minimize the standard deviation of cell waiting time and total distance traveled by AGVs.

In addition to these metaheuristics studies, several methods using genetic algorithms have been proposed. [3, 12, 13, 14, 15, 16, 17, 18, 19] Ulusoy et al. [12] proposed a genetic algorithm that simultaneously solves the machine and AGV scheduling problems. Abdelmaguid et al. [13] proposed genetic and heuristic algorithms for machine and AGV scheduling, respectively. Zhang et al. [14] proposed a genetic algorithm that employs a tabu search procedure to minimize the makespan and storage. Lyu et al. [3] used a genetic algorithm to schedule the delivery of machines and AGVs in a manufacturing system to minimize the makespan. They also employed the Dijkstra method based on time window constraint (Time Window) to plan the route, considering the collision between AGVs. Umar et al. [15] proposed a hybrid multi-objective genetic algorithm with three objectives: minimizing makespan, AGV travel time, and penalty cost. Mousavi et al. [16] adopted a genetic algorithm combined with a particle swarm optimization method to

achieve scheduling in a manufacturing system with battery constraints. Xu et al.[17] used a hybrid method combining genetic algorithm and an ant colony optimization method to schedule AGV deliveries and ultimately minimize the makespan. Liu et al. [18] proposed a scheduling method that attempts to optimize AGV battery charging, AGV speed, and makespan using a genetic algorithm. Another study that adopts a solution structure similar to the one proposed in this study is the method presented by Zou et al[19]. They arranged the tasks in order of execution, as illustrated in Fig. 3, and represented the solution as a column separated by zero for each AGV to be executed. They also proposed a search method that employs local search operations such as exchange and insertion, in addition to genetic algorithm operations such as crossover and mutation. In this study, The proposed method is faster and requires fewer operations than the search method proposed by Zou et al. The proposed method was compared via simulation with the method proposed by Lyu et al. and Zu et al.

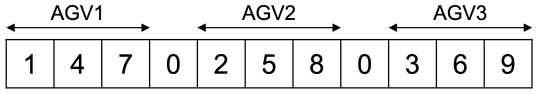


Fig. 3: Example of the structure of a a Zou et al.[19] solution

Problem description

This section describes the problem setting assumed in this study. The environment assumed in this study is illustrated in Fig. 4. All goods are stored on shelves, and the shelves are placed in one of the shelf storage spaces and carry-in/out ports presented in Fig. 4. There are three types of transportations: transportation from the carrying-in/out port to the shelf storage spaces, transportation from the shelf storage space to the carrying-in/out port, and transportation from the shelf storage space to the shelf storage space. The transportation from the carrying-in/out port to the shelf storage space is the transportation of goods brought in by trucks, and the transportation from the shelf storage space to the carrying-in/out port is the transportation of materials to be carried out. The transportation from the shelf storage space to the shelf storage space is the transportation of goods used for work and the transportation of empty shelves for loading goods.

The operator requests the transfer by specifying the shelf to be transferred, destination, and time frame for the transfer (transfer request period). This request is called a transfer task. The location where the specified shelf is placed is the start point of the task, and

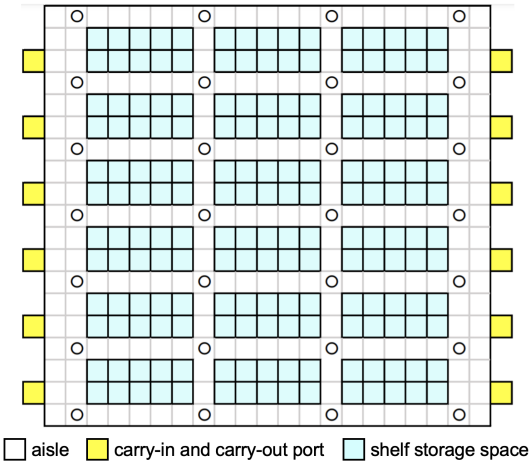


Fig. 4: Assumed environment

the destination is the end point of the task. The AGV is required to transport the shelves to the destination during the transfer request period according to the assigned transfer task. Tasks include transporting products to be carried out by the same truck, or transporting materials used for the same task, and they are divided into groups for each type of transportation. If even one task of each group is delayed, it will affect the subsequent work. Accordingly, a transportation requirement period is given to each group.

In addition to these problem settings, the following conditions are assumed:

1. The aisle is wide enough for easy passage, and no collision occurs between AGVs.
2. It takes 1 s for the AGV to move one cell, and the speed is always constant.
3. The AGV moves along the shortest path from the start point to the end point to complete the transportation.
4. The AGV starts the next transfer task from the point where it finished the previous one.
5. It takes 10 s each for the AGV to load and unload the rack.
6. AGVs can only carry one shelf at a time.
7. AGVs can wait at the shelf storage area and carry-in/out port.
8. AGV battery charging is not considered.

In this study, an evaluation function was designed to determine a schedule that can complete the transfer more quickly within the transfer requirement period. The evaluation function is presented in the Methods section.

Methods

Here, the structure of the solution proposed in this study and the search algorithm adopted with this solution structure are described. In this study, a method is proposed to quickly search for a schedule that can complete the delivery quicker within the delivery request period. To quickly search for a schedule, the solution is constructed in a way that is close to the form in which it will be executed, and the search is performed by repeating the exchange and insertion operations, considering connectivity. The solution is structured as a two-dimensional array of transport tasks arranged in the order in which they are executed for each AGV, which is close to how they are actually executed. To search while maintaining a sequence of tasks with short inter-task turnover, we consider the connectivity between task end points and task start points when performing exchange and insertion operations. First, the structure of the proposed solution is presented, then the flow of the search algorithm is shown. Subsequently, the exchange and insertion operations are explained, and finally, we describe how to convert the solution into a schedule, including how to evaluate and select the solution.

Structure of the solution

In this study, the solution is structured as a two-dimensional array of AGVs arranged in the order in which they perform their transport tasks. By constructing the solution accordingly, we believe that we can determine a schedule while considering the order in which the tasks are actually executed. An example of assigning six transport tasks to three AGVs is illustrated in Fig. 5. In this example, AGV1 executes tasks in the order 7, 4, 9, 6, AGV2 executes tasks in the order 3, 1, while AGV3 executes tasks in the order 5, 2, 8.

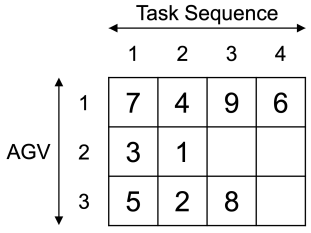


Fig. 5: Structure of the proposed solution

Flow of the search algorithm

The flow of the proposed search algorithm is presented in Fig. 6. First, it takes the input of the task and randomly generated multiple solutions to create an initial set of solutions. In this case, each solution is constructed using the solution structure presented in Fig. 5. The next step is to change the position of the solution task in the set using the exchange and insertion methods to create a new solution. At this time, the replacement or insertion is completed, considering the connectivity between the task to be changed and the neighboring task. Then, the evaluation value of each created solution is calculated using the evaluation function. The evaluation function is designed, such that the evaluation value reduces when the transfer can be completed quickly within the transfer request period. Finally, a solution that constitutes the next set is selected based on the evaluation value of each solution from the set of the generated solutions and the solutions of the set. This process is repeated to determine the solution with the best evaluation value.

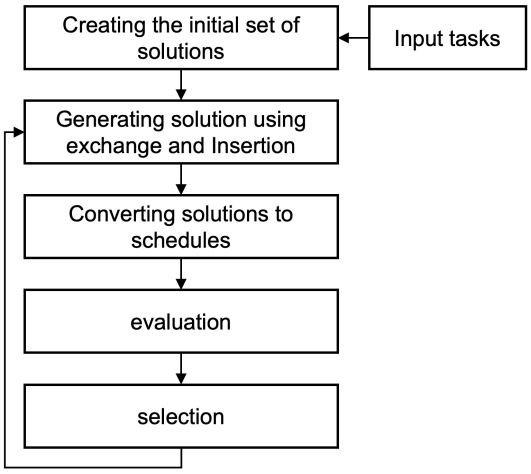


Fig. 6: Flow of the proposed algorithm

Generating solutions using exchange and insertion

In this section, algorithms for generating solutions via exchange and insertion operations are described. First, two points are randomly selected for replacement or insertion from each task, including the points before and after it. If two tasks are selected, an operation is performed to exchange the positions of the two tasks. If a point between the tasks is selected, an operation is performed to insert the selected task between the tasks. An example of selecting two points is shown in Fig. 7. In the example on the left side of Fig. 7, two tasks are selected, and the positions of these two tasks are exchanged. In the example on the right side of Fig. 7, one task and a point between the tasks are selected, and the selected task is inserted into the point between the tasks.

Next, a method that considers the connectivity with the previous and next tasks when performing exchange

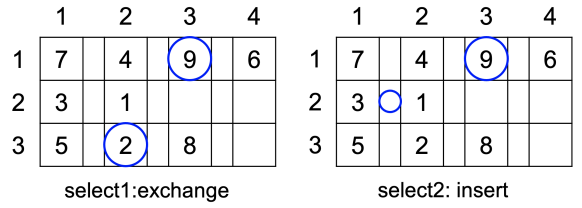


Fig. 7: Example of selection

and insertion operations is described. The degree of connectivity with the previous and subsequent tasks is expressed as the degree of coupling, and the idea is to move the previous and subsequent tasks together based on the degree of coupling. The shorter the empty trip time from the end point of the previous task to the storage location of the next task, the shorter the execution time of the next task. Therefore, the brevity of this time is expressed as the coupling degree. Here, the transfer task numbers are i and j, the transfer task i is T_i, and the start and end times of the transfer request period for T_i are S_i and E_i. The travel time from the shelf of T_i to the destination of T_i is D'_i, the travel time from the shelf location of T_i to the destination of T_i is D'_i, while the total time for T_j when T_j is executed after T_i is $TD_{ij}(TD_{ij} = D_{ij} + D'_j)$. Let p be the parameter that changes the magnitude of the coupling, and let M be the maximum time for transportation. The coupling degree C_{ij} of T_i and T_j is given by the Eq. (1).

$$C_{ij} = \begin{cases} 0 & if\ S_i + TD_{ij} > E_j \\ \left[1 - \frac{D_{ij} + \{S_j - (E_i + TD_{ij})\}}{M}\right] p & if\ S_j > E_i + TD_{ij} \\ (1 - \frac{D_{ij}}{M})p & otherwise \end{cases} \tag{1}$$

By dividing D_{ij} by M, the length of time is normalized and subtracted from 1 to express the brevity of the transmission between T_i and T_j as being between zero and one. If $S_i + TD_{ij}$ is larger than E_j, the task T_j will always exceed the end time E_j of the transportation request period, and it is not desirable to execute the tasks in this order; hence, the coupling degree is set to zero. If S_j is larger than $E_i + TD_{ij}$, AGV always waits more than $\{S_j - (E_i + TD_{ij})\}$ after finishing task i before starting task j; hence, it is calculated by adding $\{S_j - (E_i + TD_{ij})\}$ to D_{ij}.

The selected task adopts Eq. (1) to calculate the degree of coupling with the tasks before and after it, and employs the degree of coupling as a probability to decide whether to move together. In the case of swapping together, we calculate the degree of coupling between the task before and after the task that is also swapped together, and then recursively make the decision to move together.

The last step is to move the position of the task. An example of an exchange is presented in Fig. 8, and an example of an insertion is illustrated in Fig. 9. Fig. 8 calculates the degree of coupling between the selected task and the tasks before and after it, and identifies Tasks 9 and 6 and Tasks 5 and 2 as Tasks to be exchanged together. In Fig. 9, Tasks 9 and 6 are inserted into the selected points.

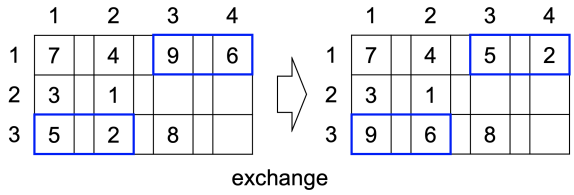


Fig. 8: Example of exchange

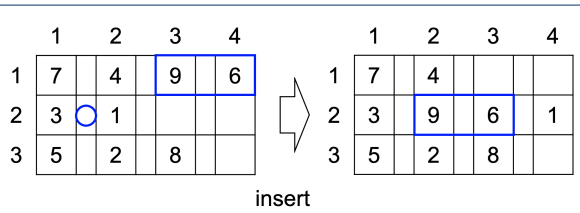


Fig. 9: Example of insertion

Converting a solution to a schedule

In this section, we describe how to convert an array into a schedule. The start and end times of the tasks are calculated in order, starting from the first task, and assigned in order to the last task in each AGV; subsequently, the schedule is obtained from the two-dimensional array. Suppose T_i is assigned to the rth of AGV, a and T_j is assigned to the $r+1$th. The start time of T_i is ST_i and the end time of T_i is ET_i; hence, start time of T_j, ST_j, is given by Eq. (2).

$$ST_j = \begin{cases} 0 & if\ r = 0 \\ S_j - TD_{ij} & if\ S_j > ET_i + TD_{ij} \\ ET_i & otherwise \end{cases} \tag{2}$$

If $x = 0$, it implies that T_j is the first task of AGV a, and $ST_j = 0$. If $S_j > ET_i + TD_{ij}$, and T_j commences immediately after T_i, it means that it is transported before the transport request period. Therefore, it starts T_j at $S_j - TD_{ij}$, and then arrives at exactly

the start time of the transport request period. In other cases, the start time of each task is the same as the end time of the previous task.

The end time ET_j of T_j is given by the Eq. (3).

$$ET_j = ST_j + TD_{ij} \tag{3}$$

The end time is the sum of the start time and the execution time.

Evaluation and selection

In this section, the methods for calculating and selecting the evaluation value of a solution are described. The evaluation value of the solution is calculated by the delay time from the end requirement period for each group to be carried out, while the sum is the evaluation value. Because the group cannot depart until all the supplies are in place, the delay from the required finish time for each group is given by the delay of the task with the slowest finish time in the group. The end time of the end request period given by the group s is TE_s, and the set of task numbers that transport the materials in the group s is X_s. The number of tasks that transports the materials in the group s is $x_s(x_s \in X_s)$. The delay time DT_s from the end of the end requirement period of group s is given by the Eq. (4).

$$DT_s = \begin{cases} Max(ET_{x_s}) - TE_s & if\ Max(ET_{x_s}) - TE_s \geq 0 \\ 0 & otherwise \end{cases} \tag{4}$$

If the delay time from the end requirement period is zero, we attempt to obtain a schedule that completes the transportation earlier. Therefore, the sum of the difference between the end of the end-request period and the time when all goods in the group have been transferred is adopted as the evaluation value. The number of groups is N and the evaluated value of the solution E is given by Eq. (5)

$$E = \begin{cases} \sum_{k=1}^{N}(TE_k - Max(ET_{x_k})) & if\ \sum_{k=1}^{N} DT_k = 0 \\ \sum_{k=1}^{N} DT_k & otherwise \end{cases} \tag{5}$$

In the selection process, according to the evaluation values given by these equations, the next set of solutions is selected from the set of solutions left over from the previous generation and the set of solutions generated, in the order of decreasing evaluation values.

Simulation results and discussion

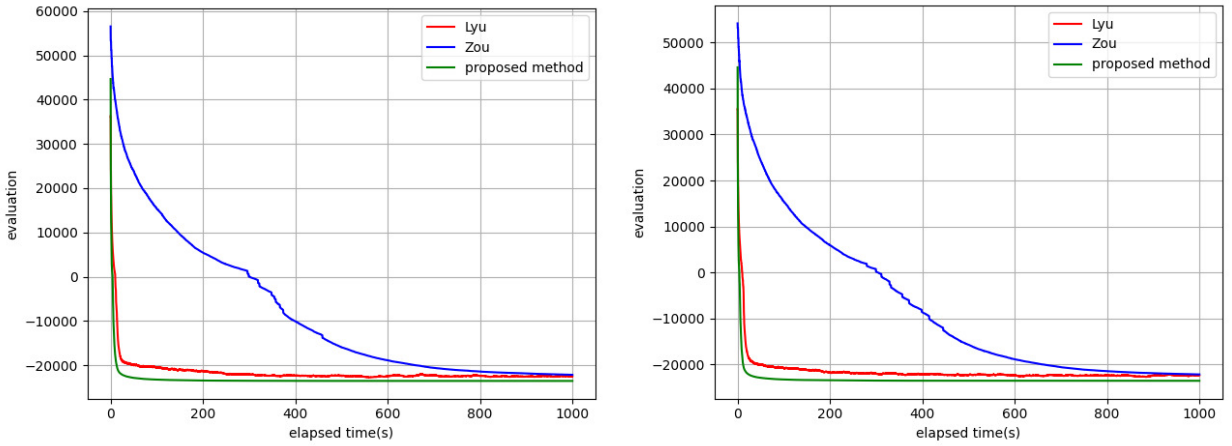
Numerical simulations are conducted to verify the effectiveness of the proposed method. The first step is to compare the search speed with that of previous studies, which will be referred to as Verification 1. In Verification 1, the methods proposed by Lyu et al.[3]'s and Zou et al.[19]'s are executed, and obtained results are compared. Next, the effect of considering connectivity is verified, and this is referred to as Verification 2. In Verification 2, three methods are compared: one with connectivity, one without, and one that switches between with and without connectivity. Finally, the trend of the search speed when the problem size is altered is checked, and this is verified as Verification 3. In Verification 3, the results of applying the proposed method to problems with different number of tasks and AGVs are compared.

Simulation conditions

The conditions for the numerical simulation are presented below. The environment of Fig. 4 is assumed in the numerical simulation. The location and destination of the shelf to be transported for each task are given by randomly selecting one of the shelf storage locations and the carry-in/out ports. A group of tasks comprises 10 tasks. It is assumed that each task of truck carried out with people is divided into 5 time periods, which are 0-20 min, 10-30 min, 20-40 min, 30-50 min, and 40-60 min, respectively. The transport request period for a group of tasks is given in one of these time segments. In this study, the number of AGVs required increases as the number of tasks increases, and one AGV is assumed to be added for every 50 tasks. Under these conditions, the tasks are given to each method and the results are compared. It is important to determine a feasible solution (a solution with no delay) as quickly as possible, and to obtain a solution that can complete the transportation as soon as possible. Therefore, in each numerical simulation, the time taken to determine a feasible solution is compared with the evaluation value at the end of the search. All algorithms are coded in C, and numerical simulations are performed on a PC with Intel Core i7-10510 1.80GHz and 16 GB of memory.

Validation 1. Comparison of search speed with previous studies

First, to verify the effectiveness of the proposed method, the results are compared with the methods proposed by Lyu et al.[3] and Zou et al.[19]. In the proposed method, the connectivity parameter p is set to one until a feasible solution is found, and zero thereafter. In Lyu et al.'s research, simultaneous scheduling of machine and AGV transfers in a manufacturing system is performed; hence the structure of the solution



(a) Run 10 times for the same 1 question **(b)** Run once for each of the 10 different questions.

Fig. 10: Verification 1: Transition of the mean value of the evaluation

Table 1: Verification 1: Elapsed time when a viable solution is obtained

(a) Run 10 times for the same 1 question

method	mean±SD	min
Lyu	11.97±1.64	10.08
Zou	350.37±44.63	297.2
proposed method	5.01±0.51	4.26

(b) Run once for each of the 10 different questions.

method	mean±SD	min
Lyu	12.63±1.49	9.45
Zou	354.54±50.71	280.42
proposed method	4.83±0.70	3.74

Table 2: Verification 1: Mean, standard deviation, and minimum values of the evaluation values for each elapsed time

(a) Run 10 times for the same 1 question

	100s		1000s	
	mean±SD	min	mean±SD	min
Lyu	-20596.7±271.0	-20935	-22545.1±243.2	-22915
Zou	15373.8±2197.0	11975	-22136.9±518.6	-22496
proposed method	-23220.6±75.7	-23305	-23523.7±49.3	-23591

(b) Run once for each of the 10 different questions.

	100s		1000s	
	mean±SD	min	mean±SD	min
Lyu	-20830.1±273.6	-21296	-22392.6±366.0	-22914
Zou	15406.6±1944.4	11856	-22208.0±339.0	-22565
proposed method	-23310.8±102.7	-23456	-23595.1±97.3	-23741

actually comprises three columns: the order of execution of tasks, allocation of tasks to machines, and allocation of transfers to AGVs. Therefore, to apply this method to the problem considered in this study, a method in which the solution comprises two columns of assignments to AGVs as the columns of assignments to machines is adopted in the experiment. In addition, the route is given as the shortest path; hence, route planning is not considered. In their study, Zou et al. adopted a problem-specific creation method to create the initial set of solutions and a restart strategy. In the verification, the initial set of solutions is created randomly and no restart strategy is adopted because only the performance of the search is to be compared. In both the methods proposed by Lyu et al. and Zou et al., the number of initial sets is 80, and the crossover and mutation rates are 0.8 and 0.2, respectively. In this verification, the proposed method and the methods proposed by Lyu et al. and Zou et al. are applied to a problem with 250 tasks and 5 AGVs, and a search is performed for 1000 s. Because the proposed method is a probabilistic search algorithm, it is expected that there will be differences in the results depending on the problem and random numbers. Therefore, the results of 10 problems for each condition are compared with the results of 10 runs for each problem.

The transitions of the mean evaluation values are illustrated in Fig. 10, the elapsed time for obtain-

ing a feasible solution is presented in Table 1, and the mean, standard deviation, and minimum values of the evaluation values for each elapsed time are presented in Table 2. Table 1 shows that the proposed method can determine a feasible solution faster than the methods proposed by Lyu et al. and Zou et al. In addition, Fig. 10 and Table 2 demonstrate that the proposed method determines the solution with a better evaluation value faster than the methods proposed by Lyu et al. and Zou et al. This indicates that the search using the proposed solution structure (Fig. 5) is more effective than the search using the Lyu et al. solution structure (Fig. 1). Furthermore, the proposed method is faster than the method proposed by Zou et al. because it requires fewer operations. Table 2 shows that the standard deviation of the proposed method is smaller than that of other methods; hence, it is less affected by problems and random numbers, and can perform stable search. This could be attributed to the fact that the proposed method selects the solutions in order of the best ones, while the methods by Lyu et al. and Zou et al. adopt the tournament selection of the genetic algorithm to select from the generated solutions. Therefore, the proposed method is effective for the scheduling problem of multiple AGVs.

Verification 2: Verification of the effectiveness of connectivity

To test the effect of considering connectivity, the proposed method is compared by altering the connectivity parameter p. There are three methods for the comparison: the method with $p = 0$, the method with $p = 1$, and the search method with 1 until a feasible solution is determined and then switched to 0. These 3 methods are applied to a problem with 500 tasks and 10 AGVs, and the search is performed for 600 s. In this verification, 10 problems are created in the same way, and the results are run one at a time and 10 times for each problem.

Similar to the previous section, the transitions of the mean evaluation values are presented in Fig. 11, the elapsed time when a feasible solution is obtained is shown in Table 3, and the mean, standard deviation, and minimum values of the evaluation values for each elapsed time are presented in Table 4. Fig. 11 comparing the search when the evaluation value is greater than zero; it can be observed that $p = 1$ and the method of swich decreases the evaluation value faster. In addition, from Table 3, it can be observed that the $p = 1$ and switch methods can determine a feasible solution faster than the $p = 0$ method. This could be attributed to the fact that the evaluation value decreases more quickly than with the method of moving tasks successively because it can move multiple tasks with optimal connectivity. From Fig. 11, it can be observed that $p = 0$ in the search for evaluation values below zero, and the switch method is faster in decreasing the evaluation value. Table 4 shows that the method with $p = 1$ has the largest difference between the evaluation value after 100 s and the evaluation value after 600 s, thus indicating a slow decrease in the evaluation value. This could be attributed to the fact that the amount of tasks allocated to AGVs that move multiple tasks is biased, because the feasible solution allocates tasks to AGVs quite evenly. From these results, it can be inferred that it is effective to search by switching p.

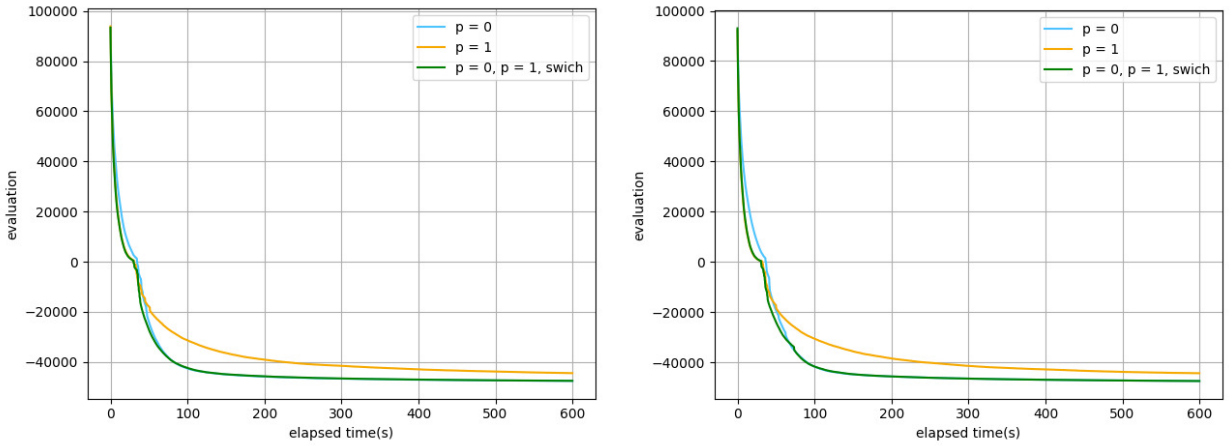
Verification 3: Problem size and search speed of the proposed method

Finally, the effect of the proposed method on the search speed is examined when the size of the problem is altered. In the proposed method, the connectivity parameter p is set to one until a feasible solution is determined, and zero thereafter. The obtained results are compared with the previous results for 250 tasks with 5 AGVs and 500 tasks with 10 AGVs. Because the evaluation criterion changes when the scale of the problem changes, the time elapsed when a feasible solution is obtained is compared.

The elapsed time when a feasible solution is obtained is presented in Table 5. Table 5 demonstrates that a feasible solution is obtained in 880 s on average, thus indicating that the proposed method is valid for a problem with 1500 tasks and 30 AGVs. Table 5 shows that the increase in elapsed time is larger than the increase in tasks. Therefore, for large-scale problems such as 100 AGVs, it is necessary to devise a solution for the initial set to speed up convergence, or divide the problem by time.

Conclusions

To address the problem of scheduling the delivery of multiple AGVs, this study proposes a method for constructing a solution using a two-dimensional array of AGVs arranged in the order in which they perform their transport tasks, and for determining the schedule by performing exchange and insertion operations. Initially, an initial set of solutions was created randomly, and for each solution, task assignments and execution orders were exchanged and inserted, and then evaluation and selection were repeatedly performed to determine solutions with optimal evaluation values. In the exchange and insertion operations, the shortness of the path from the destination of the previous task to the source of the next task was expressed as the degree of coupling, to consider the connectivity between the endpoints of the tasks. To evaluate the generated solutions, the evaluation function was designed so that



(c) Run 10 times for the same 1 question **(d)** Run once for each of the 10 different questions.

Fig. 11: Verification 2: Transition of the mean value of the evaluation

Table 3: Verification 2: Elapsed time when a viable solution is obtained

(a) Run 10 times for the same 1 question

method	mean±SD	min
p = 0	39.67±4.33	34.23
p = 1	37.51±6.06	30.36
p = 0, p = 1, swich	35.12±2.60	29.95

(b) Run once for each of the 10 different questions.

method	mean±SD	min
p = 0	43.06±7.99	36.31
p = 1	37.47±4.95	32.07
p = 0, p = 1, swich	38.79±11.87	30.6

Table 4: Verification 1: Mean, standard deviation, and minimum values of the evaluation values for each elapsed time

(a) Run 10 times for the same 1 question

	100s		600s	
	mean±SD	min	mean±SD	min
p = 0	-42582.2±412.6	-43115	-47548.8±95.2	-47653
p = 1	-31335.4±1152.9	-33187	-44444.4±338.7	-44922
p = 0, p = 1, swich	-42336.9±581.2	-43490	-47501.8±87.3	-47659

(b) Run once for each of the 10 different questions.

	100s		600s	
	mean±SD	min	mean±SD	min
p = 0	-41703.9±1581.0	-43346	-47484.8±194.7	-47708
p = 1	-30597.8±1476.8	-32645	-44378.7±513.2	-45143
p = 0, p = 1, swich	-41692.8±2759.8	-43306	-47489.7±134.4	-47647

Table 5: Verification 3: Elapsed time when a viable solution is obtained

(a) Run 10 times for the same 1 question

condition		mean±SD	min
Number of tasks	Number of AGVs		
task250	AGV5	5.01±0.51	4.26
task500	AGV10	35.12±2.60	29.95
task1500	AGV30	880.59±240.71	738.58

(b) Run once for each of the 10 different questions.

condition		mean±SD	min
Number of tasks	Number of AGVs		
task250	AGV5	4.83±0.70	3.74
task500	AGV10	38.79±11.87	30.6
task1500	AGV30	962.91±318.08	706.39

the evaluation value of a solution with a delay in even one of the transport tasks for obtaining a feasible solution would be the sum of the delay times of each task. This evaluation function was employed to evaluate the generated solutions, and the search was conducted while selecting and retaining solutions with low evaluation values. By comparing the proposed method with the methods of previous studies, it was confirmed that the proposed method can determine a feasible solution faster and obtain a schedule that completes the transportation quicker at the end of the search. This indicates that the proposed method can determine a feasible solution faster than the conventional methods. It was determined that the search for a value greater than or equal to 0 was more rapid when connectivity was taken into account, and the search for a value less than or equal to zero was faster when connectivity was not considered. In addition, better results were obtained by exploring with connectivity until a feasible solution was obtained, and then in the case of not considering connectivity once it was obtained. In examining the effect of the proposed method when the size of the problem is varied, it is inferred that the increase in task and search speed is greater than linear. Therefore, for large-scale problems such as 100 AGVs, it is necessary to devise a solution for the initial set to speed up convergence, or to divide the problem by time. In this study, all tasks were given at once and scheduled statically; however in practice, tasks are frequently added or altered. Hence, scheduling that can be added and modified dynamically is required. Therefore, we plan to study and implement a method that can handle dynamic scheduling in the future.

Acknowledgements
The authors are thankful for the generous support from AISIN AW Co.,Ltd. for this study. The authors also thank the Intelligent Robotics Laboratory of the Graduate School of the University of Tsukuba for their support.

Authors' contributions
KM proposed the method described in this paper, implemented all the programing, conducted some of the experiments, and drafted the manuscript. AY and AO provided the inspiration for this study, provided advice, and checked and corrected the manuscript. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
The analysis and experimental results during the current research are available from the corresponding author on reasonable request.

Declarations

Competing interests
The authors declare that they have no competing interests.

References

1. Yu, R., Zhao, H., Zhen, S., Huang, K., Chen, X., Sun, H., Zhang, K.: A novel trajectory tracking control of agv based on udwadia-kalaba approach. IEEE/CAA Journal of Automatica Sinica, 1–13 (2016)
2. Zhao, Y., Liu, X., Wang, G., Wu, S., Han, S.: Dynamic resource reservation based collision and deadlock prevention for multi-agvs. IEEE Access **8**, 82120–82130 (2020)
3. Lyu, X., Song, Y., He, C., Lei, Q., Guo, W.: Approach to Integrated Scheduling Problems Considering Optimal Number of Automated Guided Vehicles and Conflict-Free Routing in Flexible Manufacturing Systems. IEEE Access **7**, 74909–74924 (2019)
4. Zou, W.Q., Pan, Q.K., Meng, T., Gao, L., Wang, Y.L.: An effective discrete artificial bee colony algorithm for multi-AGVs dispatching problem in a matrix manufacturing workshop. Expert Systems with Applications **161**, 113675 (2020)
5. Chen, M.: A mathematical programming model for agvs planning and control in manufacturing systems. Computers and Industrial Engineering **30**, 647–658 (1996)
6. Corréa, A.I., Langevin, A., Rousseau, L.M.: Dispatching and conflict-free routing of automated guided vehicles: A hybrid approach combining constraint programming and mixed integer programming. In: Régin, J.-C., Rueher, M. (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, pp. 370–379. Springer, ??? (2004)
7. Deroussi, L., Gourgand, M., Tchernev, N.: A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles. International Journal of Production Research **46**, 2143–2164 (2008)
8. Saidi-Mehrabad, M., Dehnavi-Arani, S., Evazabadian, F., Mahmoodian, V.: An ant colony algorithm (aca) for solving the new integrated model of job shop scheduling and conflict-free routing of agvs. Computers and Industrial Engineering **86**, 2–13 (2015)
9. Miyamoto, T., Inoue, K.: Local and random searches for dispatch and conflict-free routing problem of capacitated AGV systems. Computers and Industrial Engineering **91**, 1–9 (2016)
10. Li, G., Zeng, B., Liao, W., Li, X., Gao, L.: A new AGV scheduling algorithm based on harmony search for material transfer in a real-world manufacturing system. Advances in Mechanical Engineering **10**, 1–13 (2018)
11. Zou, W.Q., Pan, Q.K., Tasgetiren, M.F.: An effective discrete artificial bee colony algorithm for scheduling an automatic-guided-vehicle in a linear manufacturing workshop. IEEE Access **8**, 35063–35076 (2020)
12. Ulusoy, G., Sivrikaya-Şerifoğlu, F., Bilge, Ü.: A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. Computers and Operations Research **24**, 335–351 (1997)
13. Abdelmaguid, T.F., Nassef, A.O., Kamal, B.A., Hassan, M.F.: A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles. International Journal of Production Research **42**, 267–281 (2004)
14. Zhang, Q., Manier, H., Manier, M.A.: A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times. Computers and Operations Research **39**, 1713–1723 (2012)
15. Umar, U.A., Ariffin, M.K.A., Ismail, N., Tang, S.H.: Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (agv) in flexible manufacturing systems (fms) environment. International Journal of Advanced Manufacturing Technology **81**, 2123–2141 (2015)
16. Mousavi, M., Yap, H.J., Musa, S.N., Tahriri, F., Md Dawal, S.Z.: Multi-objective agv scheduling in an fms using a hybrid of genetic algorithm and particle swarm optimization. PLoS ONE **12**, 1–24 (2017)
17. Xu, W., Guo, S., Li, X., Guo, C., Wu, R., Peng, Z.: A dynamic scheduling method for logistics tasks oriented to intelligent manufacturing workshop. Mathematical Problems in Engineering **2019** (2019). doi:10.1155/2019/7237459
18. Liu, Y., Ji, S., Su, Z., Guo, D.: Multi-objective agv scheduling in an automatic sorting system of an unmanned (intelligent) warehouse by using two adaptive genetic algorithms and a multi-adaptive genetic algorithm. PLoS ONE **14**, 1–21 (2019)
19. Zou, W.Q., Pan, Q.K., Wang, L.: An effective multi-objective evolutionary algorithm for solving the AGV scheduling problem with pickup and delivery. Knowledge-Based Systems **218**, 106881 (2021)