Li *et al.*
Chinese Journal of Mechanical Engineering (2022) 35:12
https://doi.org/10.1186/s10033-022-00683-7

Chinese Journal of Mechanical Engineering

ORIGINAL ARTICLE

Open Access

Integrated Production and Transportation Scheduling Method in Hybrid Flow Shop

Wangming Li, Dong Han, Liang Gao, Xinyu Li* and Yang Li

Abstract

The connection between production scheduling and transportation scheduling is getting closer in smart manufacturing system, and both of those problems are summarized as NP-hard problems. However, only a few studies have considered them simultaneously. This paper solves the integrated production and transportation scheduling problem (IPTSP) in hybrid flow shops, which is an extension of the hybrid flow shop scheduling problem (HFSP). In addition to the production scheduling on machines, the transportation scheduling process on automated guided vehicles (AGVs) is considered as another optimization process. In this problem, the transfer tasks of jobs are performed by a certain number of AGVs. To solve it, we make some preparation (including the establishment of task pool, the new solution representation and the new solution evaluation), which can ensure that satisfactory solutions can be found efficiently while appropriately reducing the scale of search space. Then, an effective genetic tabu search algorithm is used to minimize the makespan. Finally, two groups of instances are designed and three types of experiments are conducted to evaluate the performance of the proposed method. The results show that the proposed method is effective to solve the integrated production and transportation scheduling problem.

Keywords: Hybrid flow shop, Integrated scheduling, Task pool, Hybrid algorithm

1 Introduction

Due to the development of industry, scheduling plays a crucial role in modern manufacturing systems. As a branch of flow shop, the hybrid flow shop is widespread in modern industries, including electronics [1], textile [2], steelmaking [3] and petrochemical industries [4].

In mechanical manufacturing, production scheduling and transportation scheduling are two vital parts [5]. Optimizing both of them becomes a core task of advanced manufacturing and modern management, which not only allocates tasks but also affects the utilization level of resources and energy [6].

The importance of production scheduling problem and transportation scheduling problem has been emphasized by many researchers [7–11]. For the literatures of production scheduling problem in hybrid flow shop (also called HFSP), most of the researchers didn't consider the transport procedure between machines or put it as fixed value into the setup time. However, more and more flexible transporters like AGVs are used to perform the transfer tasks in the modern factory, which obviously improves the productivity of manufacturing enterprises [9]. On the other hand, the use of AGVs also brings uncertainty and complexity to the current scheduling scheme. For example, unprocessed jobs can be processed on a set of alternative machines at a specific stage. All of these can cause uncertainty in transfer time during scheduling. As a result, in many manufacturing industries that are sensitive to transport time and limited transport resources, they attach more and more importance to considering the integrated production scheduling and transportation scheduling.

*Correspondence: lixinyu@mail.hust.edu.cn
The State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

Springer Open

© The Author(s) 2022. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

Therefore, this paper integrates the production and transportation scheduling in hybrid flow shops with identical machines, which is an extension of the HFSP. Transfer tasks between machines and machines or machines and warehouse are performed by a certain number of AGVs. And the goal of the IPTSP is to find optimal processing sequences on machines and optimal transport sequences on AGVs simultaneously.

Production scheduling and transportation scheduling are both well-known NP-hard problems [12, 13]. There are several researches that studied the coordination between production scheduling and transportation scheduling. Bilge and Ulusoy [14] assigned the transportation task to AGVs while scheduling the processing sequence of jobs on the machine. Amir and Pedram [15] addressed a permutation flow-shop scheduling problem with a finite number of transporters carrying jobs from each machine to its subsequent machine. Nishi et al. [16] used a bilevel decomposition algorithm to solve the simultaneous scheduling and conflict-free routing problems for AGVs. Elmi et al. [17] addressed the robotic scheduling problem considering multiple part types, unrelated parallel machines, multiple robots in blocking hybrid flow shop. Zabihzadeh et al. [18] used ant colony optimization (ACO) algorithm and genetic algorithm (GA) to solve flexible flow shop scheduling problem with robotic transportation and release time.

However, in all surveys mentioned above, they considered all stages of each job and put them as a long sequence to optimize. When the long sequence was worked as the code in their algorithms, it might make the algorithms search too large solution space and hardly get a satisfactory solution within reasonable computation time. Therefore, this paper proposes a new method for integrated production and transportation scheduling problem in hybrid flow shop environment, including the establishment of task pool, the new solution representation, the new solution evaluation and so on. Based on those, the production scheduling problem and the transportation scheduling problem mentioned above can be treated together. Then, a genetic algorithm with tabu search is applied to solve the integrated scheduling problem.

This paper is organized as follow. Section 2 presents the notation and description of problem. Section 3 introduces some preparation for solving the IPTSP. Section 4 describes the details of the hybrid algorithm. Section 5 shows the experimental design and results. Finally, Section 6 gives the conclusions and future work.

Table 1 Processing time of 3 jobs

Time(job/stage)	Stage 1	Stage 2	Stage 3
Job1	6	8	10
Job2	5	5	4
Job3	6	6	8

2 Problem Description and Formulation

In hybrid flow shop environment, a set of n jobs need to be processed at S stages. Each stage j has M_j identical parallel machines, while $M_j \geq 2$ for at least one stage. Once a job has completed processing at a certain stage, it needs to be transferred to the machine of its next stage by AGV. R identical AGVs are responsible for these transferring tasks. The object of scheduling is to determine the assignment of machines and AGVs at each stage for each job, the sequence of jobs on machines and the sequence of transferring tasks on AGVs, such that the makespan is minimized.

A small example is shown. There are 3 jobs to be processed without preemption on 4 machines. Each job needs to go through 3 processing stages. The number of identical machines in each stage is {1, 2, 1}. The number of identical AGVs is 2. The processing time of each job at each stage is shown in Table 1. The transport time between machines and machines (or a warehouse)

Table 2 Transport time between locations

Time (from/to)	Warehouse	Machine1	Machine2	Machine3	Machine4
Warehouse	0	2	4	6	6
Machine1	2	0	2	4	4
Machine2	4	2	0	2	2
Machine3	6	4	2	0	4
Mahcine4	6	4	2	4	0

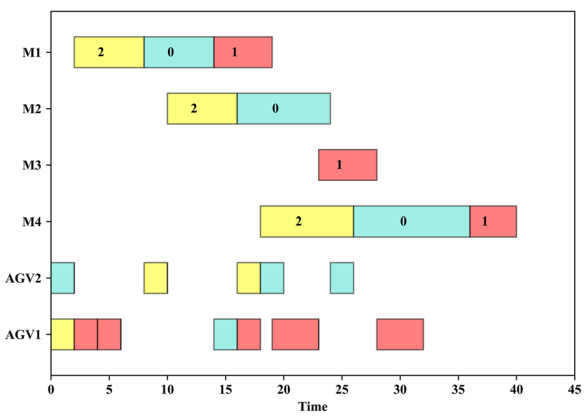


Figure 1 An example for the integrated scheduling problem

is shown in Table 2. Figure 1 shows a Gantt chart of a scheduling scheme with makespan 40.

3 Some Preparation for Solving IPTSP

Under the same scale of problem, it's obvious that the solution space of integrated scheduling problem is much larger than that of production scheduling or transportation scheduling. For getting better algorithm design in next section, some preparations are considered to be completed. Firstly, the task pool is introduced to transform scheduling into task selection and assignment, which can simplify the IPTSP. Secondly, a new way of solution representation according to the proposed task pool is proposed, which provides an encoding method for the IPTSP that can be operated by subsequent algorithms. Finally, the solution evaluation can evaluate the makespan of each code. By this way, all of them can help algorithm efficiently find satisfactory solutions while appropriately limiting the search space.

3.1 Establishment of Task Pool

In the IPTSP, each job at each stage requires a transportation task which takes it from previous machine (or warehouse) to current machine. And the number of existing transportation tasks does not exceed the total number of jobs at the same time. Based on these characteristics, this paper establishes a specific set of transportation tasks, called task pool. Then, the integrated scheduling problem can be solved through the procedure that the AGVs execute the tasks in the task pool in a specific order until the task pool is emptied.

Specifically, a task can be described as:

$$Task = (job\_no, stage\_no, st, from\_location), \quad (1)$$

where *job_no* is the index number of the job involved, *stage_no* is the current stage number of the job involved, *st* is the earliest start time of the task, *from_location* is the starting location of the task (if *stage_no* =1, it represents the warehouse; else, it represents the processing machine of job at stage *stage_no* -1).

Here, the task pool is a collection of all tasks to be scheduled. Initially, in the task pool, the number of tasks is equal to the number of jobs, the *job_no* of all tasks represent all jobs to be scheduled, the *stage_no* of all tasks represent stage one, the *st* of all tasks are zero, the *from_location* of all tasks represent warehouse. Once a task in task pool is completed by an AGV, if the *stage_no* of the task is not equal to the last stage of the job, the completed task ($Task_{old}$) will be removed from the task pool. At the same time, a new task ($Task_{new}$) representing the next stage of the same job is added to the task pool. The old and new tasks satisfy the following relationship:

(1) The job_no_{new} is equal to job_no_{old};
(2) The $stage\_no_{new}$ represents the next stage of $stage\_no_{old}$;
(3) The st_{new} is equal to the completion time of the job i at stage j involved in the $Task_{old}$.
(4) The $from\_location_{new}$ is the processing machine of the job i at stage j involved in the $Task_{old}$.

where, job_no_{new} and $stage\_no_{new}$ is the *job_no* and *stage_no* of $Task_{new}$, job_no_{old} and $stage\_no_{old}$ is the *job_no* and *stage_no* of $Task_{old}$, st_{new} and $from\_location_{new}$ is the *st* and *from_location* of $Task_{new}$. A simple example is shown in Figure 2. Once the old task $Task_2$ is completed and stage one is not the last stage of job_2, a new $Task_2$ which represent the next stage of the job is added in task pool. In the new task, "6" represents the completion time of job_2 on $machine_2$ at stage one (also the earliest start time of the new task). And "$machine_2$" represents the processing machine selected for job_2 at stage one (also the starting location of the new task).

After this, the object of scheduling is to assign the tasks in the task pool to each AGV in a certain order and then to schedule the trip and process, making the makespan smaller. While the task pool is emptied, the scheduling is complete.

3.2 Solution Representation

A new way of solution representation based on the proposed task pool is proposed, which provides an encoding

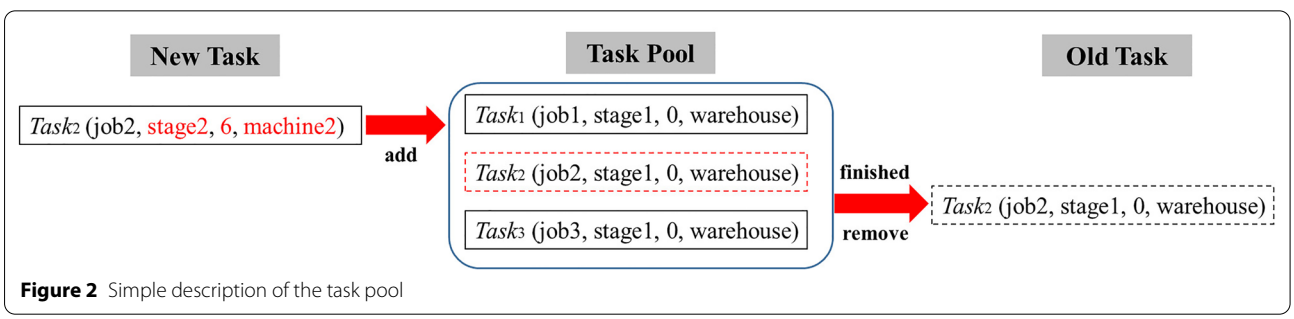


Figure 2 Simple description of the task pool

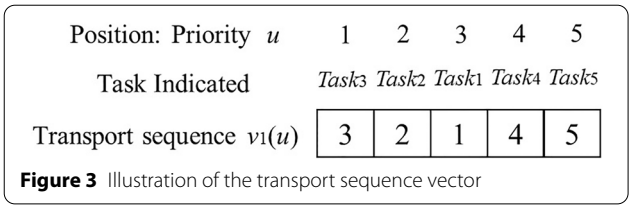


Figure 3 Illustration of the transport sequence vector

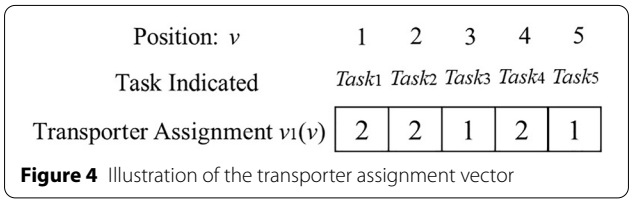


Figure 4 Illustration of the transporter assignment vector

method for the IPTSP that can be operated efficiently by subsequent algorithms.

Under some task selection rules based on time target (like: FCFS, EDT, GWTQ [19]), we found that the assignment of AGVs and the order of tasks transporting at the first stage have a great impact on the makespan, because the earliest start time of tasks in task pool are all zero and starting locations are all warehouses (at the first stage). Therefore, we consider to transform the assignment of AGVs and the order of tasks transporting at the first stage into a coding sequence to optimize. For that a new solution representation is introduced, based on tasks in task pool at the first stage. The new solution representation is composed of two parts:

1. Transport sequence vector (also called v_1);
2. Transporter assignment vector (also called v_2).

Transport sequence vector v_1 represents the order of transport for each task at the first stage. Figure 3 illustrates a transport sequence vector. For example, the transport sequence shown in Figure 3 can be translated into a list of ordered tasks below: $Task_3 \succ Task_2 \succ Task_1 \succ Task_4 \succ Task_5$.

In v_2, $v_2(u)$ represents the AGV selected for the $Task_u$ indicated at position u. Figure 4 illustrates a transporter assignment vector. For example, position 3 indicates $Task_3$, and $v_2(3)$ represents the AGV assigned for $Task_3$.

With the new solution representation, the assignment of AGVs and the sequence of tasks at the first stage have been confirmed. Furthermore, such algorithm applying requires a makespan evaluation procedure, which will be presented in the next section.

3.3 Solution Evaluation

Using the proposed solution representation method, the evaluation of makespan is described as follows:

(1) At stage one, the transport sequence and the assignment of AGVs are determined according to the two-vector representation;
(2) At stage j ($j > 1$), a task is assigned to the earliest idle AGV with a specific task selection rule. Repeat the above process until the task pool is emptied.

Among them, after the task is selected and assigned to an AGV, the processing machine (also the destination of the task) at this stage is selected according to the improved First Available Machine (FAM [20]) rule. When the task is completed, if the stage involved in this task doesn't represent the last stage, the task is removed from the task pool and a new task is added (refer to Section 3.1 for the details).

The steps are shown below:

Step 1. Schedule the tasks in task pool at stage one:

(a) Read the position information in v_1 from left to right, and get the corresponding task $Task_m$ to be scheduled.

(b) Get other information about $Task_m$. For example, we can know the assigned AGV R_v for the task in v_2, the involved job J_i, the processing time p_{ij}, the earliest start time of the task st, the starting location M_k.

(c) Plan the empty trip (from the destination $M_{k'}$ of the previous task to machine M_k) for R_v. The start time of the empty trip is earliest idle time of R_v. The arrival time of the empty trip is calculated as follow:

$$CT'_{i1} = AIT_v + pt_{k'k}, \tag{2}$$

where, $pt_{k'k}$ represents the transport time from location $M_{k'}$ to location M_k, AIT_v is the earliest idle time of R_v.

(d) Select the processing machine for J_i at stage one according the improved FAM rule. The estimated completion time $C_{i1}^{p'}$ of J_i at stage one on each available machine M_p is calculated as follow:

$$C_{i1}^{p'} = \max\left(\max(CT'_{i1}, st) + pt_{kp}, MIT_p\right) + p_{i1}, \tag{3}$$

where, MIT_p represents the earliest idle time of machine M_p, pt_{kp} represents the transport time from location M_k to location M_p, p_{i1} represents the processing time of J_i at stage one.

Then, the machine M_q with the smallest estimated completion time was selected as the processing machine for J_i at stage one.

(e) Plan the loaded trip (from location M_k to location M_q) for R_v. The start time ST_{i1} of the loaded trip is the

maximum between the earliest start time of the task and the arrival time of the empty trip:

$$ST_{i1} = \max(CT'_{i1}, st), \tag{4}$$

And the arrival time of the loaded trip is calculated as follow:

$$CT_{i1} = ST_{i1} + pt_{kq}, \tag{5}$$

where, pt_{kq} represents the transport time from location M_k to location M_q;

(f) Plan J_i to be processed on M_q. The start time S_{i1} of processing on machine M_q is the maximum between the earliest idle time of machine M_q and the arrival time of the loaded trip:

$$S_{i1} = \max(CT_{i1}, MIT_q), \tag{6}$$

where, MIT_q represents the earliest idle time of machine M_q;

(g) Update the earliest idle time of machine M_q and AGV R_v as follow:

$$MIT_q = S_{i1} + p_{i1}, \tag{7}$$

$$AIT_v = CT_{i1}, \tag{8}$$

(h) Update the task pool referred to Section 3.1;

(i) Repeat step1(a)–(h) until the last position of two-vector representation is read.

Step 2. Schedule the tasks in task pool at stage j ($j > 1$):

(a) Select the AGV R_u with the earliest idle time (AIT);

(b) Select a task from task pool as the next task for R_u. According to first come first served (FCFS) rule, the way that selecting the task $Task_m$ with the smallest st is applied;

(c) Plan the empty trip and loaded trip for R_u, select processing machine, plan the job processing and update information. All of these procedures are similar with steps from 1(b) to (h);

(d) Repeat step 2(a)–(c) until the task pool is emptied.

Step 3. Get the makespan of current two-vector solution representation.

3.4 Advantages of the Proposed Solution Representation

The proposed solution representation and solution evaluation do not yield any infeasible solution. We give the theoretical proof:

Proof. The disjunctive graph is used to prove it. According to the Ref. [21], an infeasible solution can be described as the condition that its disjunctive graph contains one or more directed cycles.

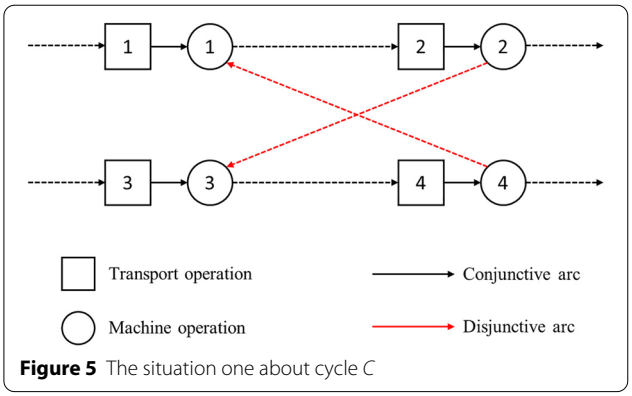


Figure 5 The situation one about cycle *C*

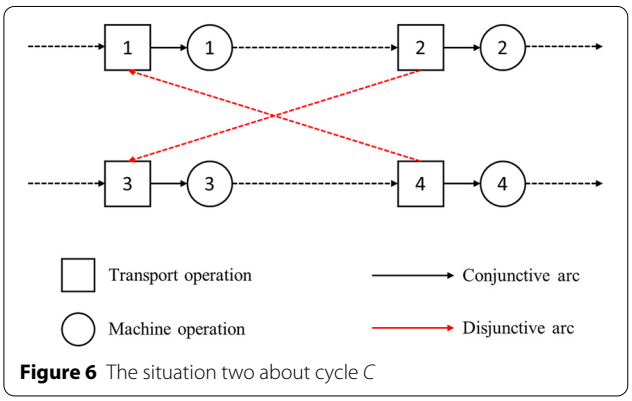


Figure 6 The situation two about cycle *C*

By contradiction: suppose the proposed solution representation and solution evaluation create an infeasible solution P. And the disjunctive graph of P contains a cycle C. There are three possible situations about the cycle C: (The dashed arrows in figure indicate that there may be some unmarked transport or machine operations on them).

(1) As shown in Figure 5, the cycle C is formed due to machine disjunctive arc and machine disjunctive arc.

As shown in Figure 5, the stage of operation 2 is after the stage of operation 1, the stage of operation 2 is after the stage of operation 1. The disjunctive arc between operation 2 and operation 3 indicates that the stage of operation 2 is equal to the stage of operation 1. So, the stage of operation 4 is after the stage of operation 1. There cannot be a disjunctive arc between operation 1 and operation 4, which is contrary to the figure. This proves that situation one cannot exist.

(2) As shown in Figure 6, the cycle C is formed due to transport disjunctive arc and transport disjunctive arc.

As shown in Figure 6, some relationships about the start time of operations can be obtained according to the conjunctive arcs: $ST_1<ST_2$, $ST_3<ST_4$ (ST_x indicates the start time of operation x).

The disjunctive arc between operation 2 and operation 3 indicates $ST_2<ST_3$. It can be inferred that $ST_1<ST_4$. According to the solution evaluation proposed by this paper, if transport operation 1 and transport operation 4 exist in the task pool at the same time, the transport operation 4 must be selected behind transport operation 1. So there can be no disjunctive arc from transport operation 1 to transport operation 4, which is contrary to the figure. This proves that situation two cannot exist.

(3) As shown in Figure 7, the cycle C is formed due to transport disjunctive arc and machine disjunctive arc.

As shown in Figure 7, some relationships about the start time of operations can be obtained according to the conjunctive arcs: $S_1<ST_2$, $ST_3<S_4$ (ST_x indicates the start time of transport operation x; S_y indicates the start time of machine operation y).

The disjunctive arc between operation 4 and operation 1 indicates $ST_4<ST_1$. It can be inferred that $ST_3<ST_2$. According to the solution evaluation proposed by this paper, if transport operation 3 and transport operation 2 exist in the task pool at the same time, the transport operation 2 must be selected behind transport operation 3. So there can be no disjunctive arc from transport operation 2 to transport operation 3, which is contrary to the figure. This proves that situation three cannot exist.

In summary, all of three possible situations about the cycle C cannot exist. This completes the proof.

Meanwhile, the proposed solution representation has major differences with that of the literatures. Refs. [17, 18] represent a solution by three long vectors which considered the sequence of operations, the machine assignment and the transporter assignment for all operations from a global view. They had proved the effectiveness and succeed in many cases. However, when the size of problem becomes larger, it may cause the algorithm to perform a lot of invalid searches and hardly to find an optimal solution.

The proposed solution representation uses two vectors representing the order sequence and assignment of AGVs at the first stage which has a big impact on result. The heuristic rules select tasks for corresponding AGV and select machine for jobs at the other stage. The advantage of the proposed solution representation is that each two-vector representation can be transformed into feasible scheduling scheme. And when applied to algorithms, it can limit the search space within a considerably range. For large-scale problems, a satisfactory solution can be obtained easily within a limited time. Detailed comparative experiments will be presented in Section 5.

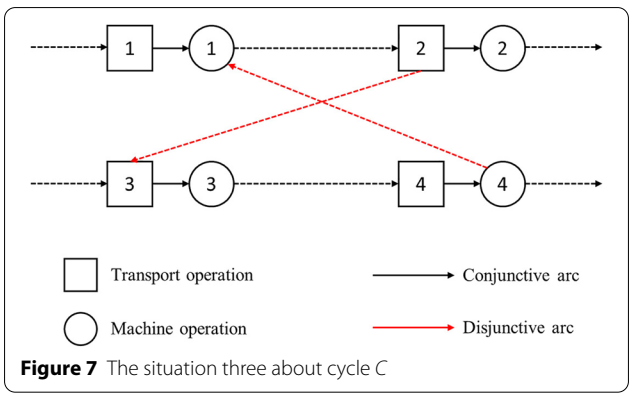


Figure 7 The situation three about cycle C

4 Proposed Genetic Algorithm with Tabu Search for IPTSP

Genetic algorithm (GA) is a well-known meta-heuristic algorithm proposed by Holland inspired by the laws of biological evolution in nature. And tabu search (TS) is a local search algorithm proposed by Glover [22] to simulate human memory function. In this paper, the tabu search algorithm is nested into GA for improving offspring individuals in each generation. The framework of the hybrid method is shown in Figure 8.

4.1 Population Initialization

The chromosomes in GA are corresponding to the solutions or Gantt charts of the integrated scheduling problem. The method of chromosome representation and decoding is as same as that described in Section 3. In order to ensure the diversity of the population, the algorithm initialize the individuals in the initial population randomly.

4.2 Crossover Operator

For GA, the crossover operator determines the way that parents produce new individual, and promotes the algorithm's global search capabilities. In this paper, two crossover operators are adopted.

The first crossover operator is the position-based crossover (PBX) for transport sequence vector (v_1). The basic procedure of PBX is described as follow (two parents are noted as P1 and P2; two offspring are noted as C1 and C2):

Step1. Randomly generate several gene positions, and C1 and C2 respectively inherit the genes of the corresponding gene positions from P1 and P2;

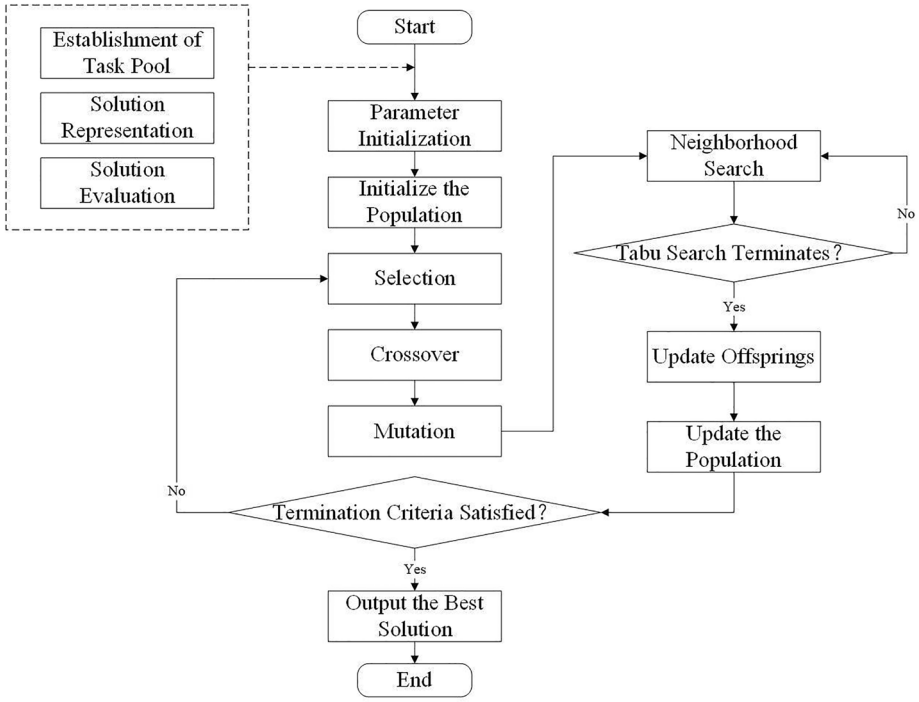


Figure 8 Framework of the method

Step2. C1 and C2 inherit the remaining genes from another parent (P2 and P1) on the unselected gene position.

The second crossover operator is the multi-point crossover (MPX) for transporter assignment vector (v_2). The basic procedure of MPX is described as follow (two parents are noted as P1 and P2; two offspring are noted as C1 and C2):

Step1. Randomly generate a sequence composed of 0 and 1, and the length of the sequence is equal to the length of v_2;

Step2. Select the same genes in P2 and P1 corresponding to position 1 in the sequence, and copy them to C1 and C2, that is, exchange the assigned transporter;

Step3. Keep the remaining genes in P1 and P2 to C1 and C2, thus generating offspring C1 and C2.

4.3 Mutation Operator

For GA, the mutation operator is used to make perturbations on chromosomes in order to maintain the algorithm's local search capabilities. In this paper, two mutation operators are adopted.

The first mutation operator is used for transport sequence vector (v_1), which selects two genes randomly and inserts the back one before the front one or the front one after the back one.

The second mutation operator is used for transporter assignment vector (v_2), which selects one gene randomly and change the value of this selected gene to the other AGV.

4.4 Neighborhood Structure

For TS, neighborhood structure is a mechanism for generating new solutions by making small disturbances to the current solution. In this paper, four ways of neighborhood structure are adopted.

(1) Binary exchange: select two points randomly in chromosome and reverse the order of all genes between these two points.
(2) Two points exchange: select two points randomly in chromosome and exchange the gene value of these two points.
(3) One point insert: select two points randomly and insert the back one before the front one.
(4) Transporter change: select one point randomly and change the AGV assignment of this gene to the other AGV.

4.5 Tabu List

The purpose of the tabu list is to avoid roundabout searches and guide the algorithm to better explore the solution space. The length of the tabu list is the tenure of the subject staying in the tabu list. If the length is L, the tabu list can be expressed as a ring table composed of L subjects. Whenever a new subject is added to the tabu list, it is possible to overwrite one of the oldest elements with this new subject.

4.6 Termination Criterion

The termination criterion determines whether the algorithm should stop. In this paper, the GA with TS terminates when the number of iterations reaches to the maximum iterations (*MaxIter*); TS terminates when the number of iterations reaches to the maximum iterations (*MaxTSIter*).

5 Experiments and Computational Results

Since there are no corresponding benchmarks for the integrated scheduling problem in hybrid flow shop environment, we design two groups of instances in this section. Then we adapt three types of experiments to verify the proposed method. Finally, we make some analysis based on the results. All of the experiments have been coded in C++ and run on Intel Core i5 2.3 GHz PC with 8 GB memory.

5.1 Instances Design

This paper has designed two groups of instances. And all of them can be found at the website [30].

5.1.1 Group 1

Ref. [23] specializes in the benchmark of HFSP. We take some instances in Ref. [23] as a part of input information, and the corresponding transportation time between machines is generated according to number of machines. The instances of proposed benchmark are divided into small-size and large-size according to the number of jobs, as follows:

(1) Small-size: number of jobs: {10, 20, 30}; number of stages: {5, 10}; number of AGVs: {2, 4, 6};

(2) Large-size: number of jobs: {80, 160}; number of stages: {5}; number of AGVs: {4, 6, 8}.

When generating the problems, an important characteristic considered is the relative magnitude of the travel times and the processing times [24]. We consider using a variable α to distinguish different types of instances, and further generate large-α type and small-α type in large/small-size instances. The α represents the ratio between the average machine-to-machine transportation time and the average processing time of the instance. The value of α in each instance is calculated as follows:

$$\alpha = \frac{(\sum\sum p_{ij})/(n*s)}{(\sum\sum pt_{pq})/(m+1)^2}, \tag{9}$$

where, p_{ij} is the processing time of job i at stage j; pt_{pq} is the transportation time of location p to location q; n is the number of jobs; s is the number of stages; m is the number of machines.

5.1.2 Group 2

Zabihzadeh and Rezaeian [18] have researched similar problems to this paper. They gave the parameters which were used to generate the instances, but did not give specific instances. In this group, parameters are generated randomly based on Zabihzadeh and Rezaeian's research [18]. For each job, standard processing time, unloading, transferring and loading time at each stage are generated from the uniform distribution U [10, 100], U [5, 20], U [5, 20], U [5, 20], respectively. The match between the number of jobs, the number of stages, and the number of AGVs still refer to their research.

5.2 Experimental Setup

5.2.1 Experiment 1: Comparison of Solution Representations

Using the same genetic algorithm in Ref. [18], under the same number of iterations and other parameters, this paper solves proposed two groups of instances in three different solution representations (coding and decoding methods), which aims to verify the effectiveness of the proposed one:

(1) Mode1: Use the coding and decoding method of the HFSP in Ref. [28], and whenever the job is scheduled to be processed on the machine, use the AGV assignment rule in Ref. [29] to allocate the appropriate AGV to complete the corresponding transportation task.

(2) Mode2: Use the same coding and decoding method of the integrated scheduling problem with Ref. [18]. Blocking and release time are considered in Ref. [18], both of which are ignored here.

(3) Mode3: Use the coding and decoding method proposed in Section 3.

5.2.2 Experiment 2: Comparison of Selection Rules

According to Section 3.3, the task selection rule has an important impact on solution evaluation. Here, four common and classic rules are used for experiment and comparison, each of which is worked as a part of solution evaluation in the GATS independently [19].

(1) Rule1: First Come First Served (FCFS). Each time, select the task with the minimum earliest start time (*st*);
(2) Rule2: Longest Time Between Arrival (LTBA). Each time, select the task with the minimum difference between the AGV's estimated arrival time and the earliest start time;
(3) Rule3: Shortest Travel Distance Rule (STD). Each time, select the task closest to the current location of the AGV;
(4) Rule4: Greatest Waiting Time in Queue (GWTQ). Each time, select the task with the longest waiting time in the buffer.

5.2.3 Experiment 3: Comparison of Algorithms

In order to verify the effectiveness of the hybrid algorithm, the following five algorithms have been selected and programmed to solve the two groups of instances and compared with the hybrid GATS algorithm. They are ① Genetic algorithm (GA), ② Simulated annealing algorithm (SA), ③ Artificial bee colony algorithm (ABC) [25], ④ Grey wolf optimizer algorithm (GWO) [26] and ⑤ Migrating birds optimization algorithm (MBO) [27].

5.3 The Experimental Results

5.3.1 Results of Experiment 1

For each instance, run it five times and record its best makespan and average makespan. The parameters of GA used here are set as follows: the population size = 20; the crossover probability = 0.9; the mutation probability = 0.5; the maximum number of iterations = 500. The computational results are presented in Tables 3, 4, 5, 6, 7. Among them, "Mode3" represents the solution representation proposed by this paper.

In Tables 3, 4, 5, 6,7, the solution representation (Mode3) proposed by this paper get all optimal solutions in best makespan and average makespan, which is much better than others.

To study the mechanism of the three different solution representations deeply, we analyze lots of the final Gantt charts for different instances (one group of which is shown in Figure 9). Meanwhile, combining with the characteristics of the HFSP, we get some findings: (1) For Mode1, affected by HFSP, processing and transportation of jobs are carried out depending on the stage (the order sequence at the latter stages is determined by the end time of jobs at the previous stage). This will make each AGV only transport jobs with the

Table 3 Comparison on Group 1 with small-size and small-α

Job	Stage	AGV	α	Mode1		Mode2		Mode3	
				Best	Average	Best	Average	Best	Average
10	5	2	0.10	612	622	534	591	424	435
		4		508	512	526	577	423	434
		6		449	458	514	563	423	431
	10	2	0.14	1092	1111	1159	1227	832	854
		4		951	967	1107	1157	830	845
		6		888	905	1085	1145	835	843
20	5	2	0.10	1176	1183	1000	1038	688	707
		4		1058	1068	938	988	686	698
		6		925	949	894	959	688	698
	10	2	0.17	1812	1841	1703	1767	974	1007
		4		1574	1596	1482	1552	871	892
		6		1422	1443	1472	1495	867	885
30	5	2	0.11	1481	1488	1228	1273	742	758
		4		1377	1391	1126	1184	719	729
		6		1276	1291	1156	1178	707	724
	10	2	0.16	2959	2966	2443	2508	1485	1510
		4		2720	2729	2217	2382	1149	1197
		6		2531	2537	2213	2301	1151	1176

Table 4 Comparison on Group 1 with small-size and large-α

Job	Stage	AGV	α	Mode1		Mode2		Mode3	
				Best	Average	Best	Average	Best	Average
10	5	2	0.41	736	757	763	813	611	635
		4		578	589	655	698	471	484
		6		509	520	560	643	469	478
	10	2	0.60	1412	1430	1674	1775	1255	1291
		4		1099	1111	1425	1486	940	961
		6		999	1016	1270	1339	914	930
20	5	2	0.38	1417	1435	1435	1508	1269	1312
		4		1144	1159	1068	1164	797	808
		6		1034	1042	1057	1129	742	759
	10	2	0.72	2610	2646	2974	3061	2505	2540
		4		1849	1867	2105	2222	1461	1492
		6		1614	1634	1921	1989	1136	1174
30	5	2	0.45	2081	2094	2156	2202	1965	2005
		4		1473	1478	1420	1517	1067	1098
		6		1364	1371	1244	1366	811	848
	10	2	0.67	4160	4193	4734	4837	3908	3977
		4		3005	3020	3175	3339	2279	2298
		6		2737	2754	2742	2879	1732	1757

Table 5 Comparison on Group 1 with large-size and small-α

Job	Stage	AGV	α	Mode1		Mode2		Mode3	
				Best	Average	Best	Average	Best	Average
80	5	4	0.10	4449	4474	3246	3321	2066	2085
		6		4337	4347	3122	3255	2014	2053
		8		4214	4240	3110	3254	2054	2068
160	5	4	0.10	9783	9794	7106	7410	4300	4366
		6		9661	9684	7305	7514	4265	4291
		8		9537	9544	7097	7302	4234	4268

Table 6 Comparison on Group 1 with large-size and large-α

Job	Stage	AGV	α	Mode1		Mode2		Mode3	
				Best	Average	Best	Average	Best	Average
80	5	4	0.43	4544	4552	4086	4212	3078	3093
		6		4440	4441	3793	3800	2288	2308
		8		4315	4333	3442	3610	2133	2152
160	5	4	0.40	9859	9876	8435	8545	6441	6463
		6		9753	9773	7687	7737	4871	4926
		8		9648	9653	7540	7700	4534	4567

same stage in a period of time. As a result, the final scheduling schemes are too limited; (2) For Mode2, under the same scale of problem, the coding length is much larger than that of others. On the one hand, it can almost represent all the solutions in the solution space, providing the possibility for the algorithm to

Table 7 Comparison on Group 2

Job	Stage	AGV	α	Mode1		Mode2		Mode3	
				Best	Average	Best	Average	Best	Average
10	2	4	0.80	393	400	448	475	387	397
	4	6	0.73	499	504	656	677	485	493
	6	6	0.71	756	771	1013	1043	728	735
	8	10	0.69	901	911	1273	1288	835	846
	10	10	0.69	1092	1108	1535	1639	1058	1062
20	2	6	0.77	578	584	590	633	496	502
	4	8	0.65	923	930	1081	1129	746	766
	6	12	0.67	1112	1121	1415	1530	895	910
	8	12	0.70	1494	1504	1965	2046	1241	1245
	10	14	0.67	1733	1748	2309	2388	1336	1364

find global optimal solutions. On the other hand, due to the large search range, it is difficult for the algorithm to find a satisfactory solution in a limited time; (3) For Mode3, by optimizing the order sequence at the first stage that has a greater impact on the makespan, and combining with effective heuristic rule, the solution representation performs better.

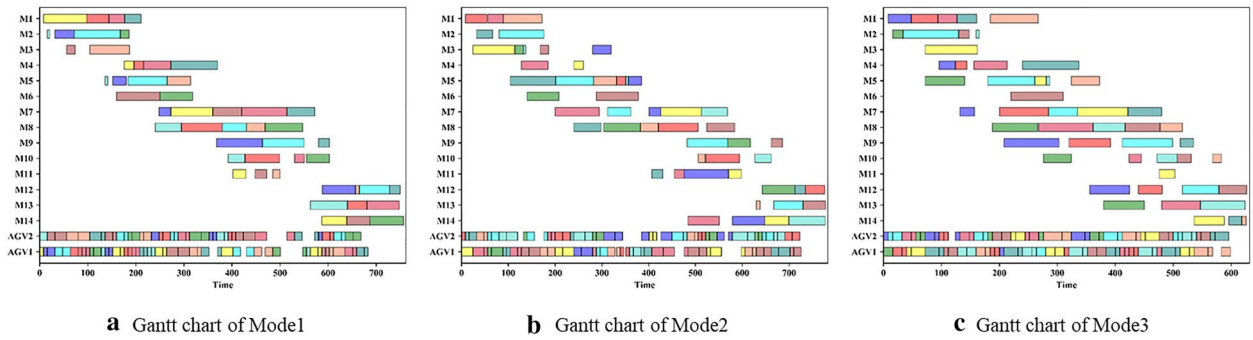
The above results show that the proposed solution representation can help algorithm obtain good results than others, under the same number of iterations and other parameters.

5.3.2 Results of Experiment 2

For each instance, run it five times and record its best makespan and average makespan. The parameters of GATS used here are set as follows: the population size = 20; the crossover probability = 0.9; the mutation

Table 8 Comparison on Group 1 with small-size and small-α

Job	Stage	AGV	α	GATS_FCFS		GATS_LTBA		GATS_STD		GATS_GWTQ	
				Best	Average	Best	Average	Best	Average	Best	Average
10	5	2	0.10	424	426	423	427	456	467	423	424
		4		423	423	422	422	441	455	423	423
		6		423	423	423	423	433	442	423	423
	10	2	0.14	830	833	842	844	948	996	830	845
		4		830	835	830	833	926	967	830	840
		6		830	839	830	830	859	888	830	834
20	5	2	0.10	687	694	689	698	820	837	686	696
		4		681	685	677	682	779	799	675	681
		6		679	685	678	682	780	795	679	682
	10	2	0.17	974	991	1024	1038	1166	1190	975	992
		4		860	861	861	869	1074	1093	851	859
		6		842	855	842	854	1046	1102	848	855
30	5	2	0.11	720	731	729	741	900	918	721	740
		4		696	700	708	709	831	856	697	705
		6		686	699	694	702	838	849	692	702
	10	2	0.16	1485	1531	1590	1603	1500	1602	1510	1532
		4		1161	1167	1155	1187	1448	1486	1135	1154
		6		1115	1123	1132	1144	1385	1395	1111	1127
Number of optimal solutions				10	9	7	6	0	0	10	6



a Gantt chart of Mode1 **b** Gantt chart of Mode2 **c** Gantt chart of Mode3

Figure 9 Gantt charts of three different solution representations

probability = 0.5; the length of tabu list = 10; the maximum iterations of the hybrid GA and TS = 50; the maximum iterations of the TS is 20. The computational results are presented in Tables 8, 9, 10, 11, 12. Among them, "GATS_FCFS" represents the hybrid GA and TS with FCFS rule.

In each type of instances, the statistics of the total number that each rule can achieve optimal result about best makespan and average makespan are shown in the Tables 13, 14. In the 58 instances, the GATS with FCFS rule achieves 30 optimal results about best makespan and 29 optimal results about average makespan, which is better than other algorithms. It is evident that the FCFS rule used in solution evaluation is effective and reliable when solving integrated scheduling problem.

5.3.3 Results of Experiment 3

For each instance, run each algorithm five times and record its best makespan and average makespan. The computational results are presented in Tables 15, 16, 17, 18, 19, 20, 21, 22, 23, 24. And Tables 25, 26, 27 present the comparison of average CPU time on two group of

Table 9 Comparison on Group 1 with small-size and large-α

Job	Stage	AGV	α	GATS_FCFS		GATS_LTBA		GATS_STD		GATS_GWTQ	
				Best	Average	Best	Average	Best	Average	Best	Average
10	5	2	0.41	604	615	615	640	644	664	596	613
		4		469	471	471	475	504	519	467	471
		6		464	465	461	463	492	497	466	467
	10	2	0.60	1250	1278	1330	1345	1416	1451	1276	1302
		4		938	944	950	967	1043	1108	929	938
		6		905	913	911	920	983	1030	900	909
20	5	2	0.38	1269	1293	1333	1350	1245	1307	1277	1303
		4		789	796	810	816	897	922	773	785
		6		728	732	724	736	820	861	728	734
	10	2	0.72	2545	2573	2646	2706	2572	2642	2551	2590
		4		1465	1490	1588	1603	1508	1561	1451	1490
		6		1145	1167	1266	1278	1309	1334	1171	1181
30	5	2	0.45	1929	1969	2038	2057	1813	1877	1919	1952
		4		1084	1118	1159	1177	1088	1108	1094	1108
		6		816	828	864	873	938	974	816	826
	10	2	0.67	3983	4022	4204	4222	3794	3853	3990	4032
		4		2322	2329	2451	2489	2214	2234	2301	2331
		6		1784	1794	1936	1958	1776	1830	1786	1799
Number of optimal solutions				5	8	2	1	5	4	7	8

Table 10 Comparison on Group 1 with large-size and small-α

Job	Stage	AGV	α	GATS_FCFS		GATS_ LTBA		GATS_STD		GATS_GWTQ	
				Best	Average	Best	Average	Best	Average	Best	Average
80	5	4	0.10	2024	2037	1995	2005	2282	2310	1994	2013
		6		1996	2008	1995	2011	2300	2306	2000	2007
		8		1989	1998	1992	2003	2151	2204	1989	2001
160	5	4	0.10	4212	4241	4238	4272	4803	4817	4261	4276
		6		4195	4213	4205	4226	4680	4720	4210	4234
		8		4199	4213	4233	4237	4671	4694	4192	4212
Number of optimal solutions				3	3	1	1	0	0	3	2

Table 11 Comparison on Group 1 with large-size and large-α

Job	Stage	AGV	α	GATS_FCFS		GATS_ LTBA		GATS_STD		GATS_GWTQ	
				Best	Average	Best	Average	Best	Average	Best	Average
80	5	4	0.43	3022	3039	3212	3231	2732	2747	3004	3026
		6		2183	2207	2333	2345	2423	2470	2187	2198
		8		2076	2085	2068	2099	2324	2381	2069	2075
160	5	4	0.40	6380	6414	6771	6774	6450	6453	6444	6446
		6		4681	4701	5071	5072	5002	5027	4736	4763
		8		4373	4416	4385	4453	4902	4924	4433	4451
Number of optimal solutions				4	3	1	0	1	1	0	2

Table 12 Comparison on Group 2

Job	Stage	AGV	α	GATS_FCFS		GATS_LTBA		GATS_STD		GATS_GWTQ	
				Best	Average	Best	Average	Best	Average	Best	Average
10	2	4	0.80	377	380	381	387	379	381	378	381
	4	6	0.73	474	479	491	498	475	486	471	477
	6	6	0.71	712	731	738	755	734	752	717	719
	8	10	0.69	832	838	842	846	846	860	838	843
	10	10	0.69	1038	1050	1045	1056	1084	1106	1050	1052
20	2	6	0.77	465	481	484	489	504	513	473	483
	4	8	0.65	740	755	788	799	767	795	732	754
	6	12	0.67	885	899	915	924	977	994	888	897
	8	12	0.70	1216	1229	1277	1285	1267	1290	1221	1234
	10	14	0.67	1348	1369	1404	1424	1428	1439	1374	1377
Number of optimal solutions				8	6	0	0	0	0	2	4

instances (only the instances with small-α are selected as the representative in the table).

In each type of instances, the statistics of the total number that each algorithm can achieve optimal results about best makespan and average makespan are shown in Tables 28, 29. In the 58 instances, the GATS achieves 40 optimal results about best makespan and 38 optimal results about average makespan, which is better than other five. This means that the hybrid GA and TS has both effectiveness and efficiency for solving the integrated scheduling problem. In most instances, the calculation time of SA is the shortest, but the quality of solutions obtained by SA is relatively poor. The calculation time of GWO is the longest. There is no significant

Table 13 Statistical result of the best makespan

Size	α	Total number	Number of optimal solutions			
			GATS_FCFS	GATS_ LTBA	GATS_STD	GATS_GWTQ
Small	Small	18	10	7	0	10
Small	Large	18	5	2	5	7
Large	Small	6	3	1	0	3
Large	Large	6	4	1	1	0
Group2		10	8	0	0	2
Total		58	30	11	6	22

Table 14 Statistical result of the average makespan

Size	α	Total number	Number of optimal solutions			
			GATS_FCFS	GATS_ LTBA	GATS_STD	GATS_GWTQ
Small	Small	18	9	6	0	6
Small	Large	18	8	1	4	8
Large	Small	6	3	1	0	2
Large	Large	6	3	0	1	2
Group2		10	6	0	0	4
Total		58	29	8	5	22

Table 15 Comparison about best makespan on Group 1 with small-size and small-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	5	2	0.10	424	425	421	423	421	421
		4		423	424	423	423	423	423
		6		423	424	423	423	423	423
	10	2	0.14	832	844	835	842	830	830
		4		830	837	836	834	830	830
		6		835	830	830	834	830	830
20	5	2	0.10	688	707	709	724	681	687
		4		686	699	690	700	682	681
		6		688	685	693	698	677	677
	10	2	0.17	974	996	997	992	948	974
		4		871	876	861	896	862	860
		6		867	877	880	880	850	842
30	5	2	0.11	742	724	766	763	714	720
		4		719	725	734	734	703	696
		6		707	718	718	733	700	686
	10	2	0.16	1485	1497	1522	1531	1459	1485
		4		1149	1194	1201	1200	1152	1161
		6		1151	1159	1186	1178	1137	1115
Number of optimal solutions				3	1	4	2	12	13

Table 16 Comparison about average makespan on Group 1 with small-size and small-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	5	2	0.10	435	433	425	428	422	426
		4		434	428	425	424	425	423
		6		431	427	424	424	424	423
	10	2	0.14	854	848	844	851	837	833
		4		845	844	841	843	838	835
		6		843	840	835	844	833	839
20	5	2	0.10	707	711	714	736	688	694
		4		698	705	700	711	686	685
		6		698	698	702	712	686	685
	10	2	0.17	1007	1009	1011	1017	961	991
		4		892	893	885	906	876	861
		6		885	881	889	897	855	855
30	5	2	0.11	758	752	775	772	733	731
		4		729	730	743	738	711	700
		6		724	728	737	736	707	699
	10	2	0.16	1510	1519	1529	1536	1464	1531
		4		1197	1201	1212	1211	1173	1167
		6		1176	1171	1188	1185	1144	1123
Number of optimal solutions				1	0	0	0	6	13

Table 17 Comparison about best makespan on Group 1 with small-size and large-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	5	2	0.41	611	610	593	613	**577**	593
		4		471	468	478	480	470	468
		6		469	467	466	466	465	464
	10	2	0.60	1255	1264	1242	1259	1226	1250
		4		940	939	934	945	948	934
		6		914	911	911	923	909	905
20	5	2	0.38	1269	1287	1280	1325	1268	1268
		4		797	792	796	802	796	776
		6		742	742	753	763	738	728
	10	2	0.72	2505	2524	2490	2542	2481	2487
		4		1461	1451	1486	1484	1455	1465
		6		1136	1151	1158	1167	1150	1145
30	5	2	0.45	1965	1986	1964	1973	1941	1929
		4		1067	1113	1100	1123	1091	1084
		6		811	850	838	856	845	816
	10	2	0.67	3908	3948	3912	3950	3928	3975
		4		2279	2291	2287	2294	2236	2236
		6		1732	1790	1782	1768	1763	1784
Number of optimal solutions				5	2	1	0	5	9

Table 18 Comparison about average makespan on Group 1 with small-size and large-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	5	2	0.41	635	620	608	633	596	615
		4		484	474	480	485	474	472
		6		478	472	467	469	466	465
	10	2	0.60	1291	1273	1258	1283	1243	1278
		4		961	950	946	961	954	944
		6		930	918	918	930	917	913
20	5	2	0.38	1312	1313	1297	1334	1287	1293
		4		808	802	807	825	801	796
		6		759	753	758	772	746	732
	10	2	0.72	2540	2548	2512	2555	2498	2506
		4		1492	1495	1495	1495	1467	1490
		6		1174	1164	1176	1180	1167	1155
30	5	2	0.45	2005	1992	1972	1986	1964	1963
		4		1098	1124	1114	1125	1104	1118
		6		848	856	852	859	849	828
	10	2	0.67	3977	4000	3960	3980	3979	4022
		4		2298	2309	2297	2308	2267	2255
		6		1757	1795	1788	1783	1773	1794
Number of optimal solutions				2	0	1	0	5	10

Table 19 Comparison about best makespan on Group 1 with large-size and small-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
80	5	4	0.10	2066	2000	2056	2057	2041	2024
		6		2014	2019	2046	2048	2027	1996
		8		2054	2014	2051	2034	2036	1989
160	5	4	0.10	4300	4265	4390	4279	4254	4212
		6		4265	4219	4315	4248	4233	4195
		8		4234	4205	4305	4239	4278	4199
Number of optimal solutions				0	1	0	0	0	5

Table 20 Comparison about average makespan on Group 1 with large-size and small-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
80	5	4	0.10	2085	2037	2070	2068	2061	2035
		6		2053	2022	2057	2066	2039	2008
		8		2068	2018	2059	2057	2050	1998
160	5	4	0.10	4366	4266	4398	4311	4340	4241
		6		4291	4231	4320	4257	4248	4213
		8		4268	4212	4350	4258	4281	4213
Number of optimal solutions				0	1	0	0	0	5

Table 21 Comparison about best makespan on Group 1 with large-size and large-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
80	5	4	0.43	3078	3091	3067	3063	3067	3022
		6		2288	2254	2334	2281	2235	2183
		8		2133	2126	2186	2139	2093	2076
160	5	4	0.40	6441	6498	6485	6465	6501	6380
		6		4871	4799	5008	4772	4792	4681
		8		4534	4416	4659	4367	4518	4373
Number of optimal solutions				0	0	0	1	0	5

Table 22 Comparison about average makespan on Group 1 with large-size and large-α

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
80	5	4	0.43	3093	3093	3077	3079	3074	3039
		6		2308	2287	2342	2301	2283	2207
		8		2152	2131	2198	2167	2121	2085
160	5	4	0.40	6463	6504	6501	6471	6515	6414
		6		4926	4812	5029	4799	4813	4701
		8		4567	4430	4663	4394	4594	4416
Number of optimal solutions				0	0	0	1	0	5

Table 23 Comparison about best makespan on Group 2

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	2	4	0.80	387	384	378	393	381	377
	4	6	0.73	485	479	472	490	479	474
	6	6	0.71	728	728	717	724	718	712
	8	10	0.69	835	832	845	848	834	832
	10	10	0.69	1058	1047	1053	1058	1051	1038
20	2	6	0.77	496	494	497	519	476	465
	4	8	0.65	746	751	751	750	740	740
	6	12	0.67	895	911	906	914	894	885
	8	12	0.70	1241	1230	1222	1240	1203	1216
	10	14	0.67	1336	1365	1362	1363	1336	1336
Number of optimal solutions				1	1	1	0	3	8

difference in the calculation time among the other algorithms. According to the above results, the GATS is effective to solve the IPTSP.

6 Conclusions and Future Work

This paper solves the integrated production and transportation scheduling problem in hybrid flow shop environment. Firstly, some preparation for the problem has been made. Then, a hybrid genetic algorithm with tabu search has been adapted. Finally, two groups of instances have been designed and three types of experiments have been carried out for verifying the effectiveness of the proposed method.

The contributions of this research are as follows.

(1) A new solving method for IPTSP, including the establishment of task pool, the new solution representation and the new solution evaluation, has been proposed for the problem. Taking it as coding and

Table 24 Comparison about average makespan on Group 2

Job	Stage	AGV	α	GA	SA	ABC	GWO	MBO	GATS
10	2	4	0.80	397	389	384	396	383	380
	4	6	0.73	493	482	480	492	481	479
	6	6	0.71	735	733	725	739	722	731
	8	10	0.69	846	843	847	852	837	838
	10	10	0.69	1062	1060	1057	1060	1053	1050
20	2	6	0.77	502	500	502	526	484	481
	4	8	0.65	766	763	762	775	746	755
	6	12	0.67	910	920	920	925	900	899
	8	12	0.70	1245	1238	1233	1247	1214	1229
	10	14	0.67	1364	1371	1364	1372	1344	1369
Number of optimal solutions				0	0	0	0	5	5

Table 25 Comparison about average CPU time on Group 1 with small-size

Job	Stage	AGV	GA(s)	SA(s)	ABC(s)	GWO(s)	MBO(s)	GATS(s)
10	5	2	30.5	28.4	27.0	28.4	15.5	19.1
		4	32.7	23.4	33.7	40.2	22.1	25.2
		6	34.5	30.7	40.8	55.6	30.0	31.3
	10	2	69.3	54.4	56.5	61.9	54.9	57.9
		4	74.0	79.6	72.1	87.2	77.6	70.2
		6	86.0	88.6	84.4	110.6	89.7	81.7
20	5	2	67.5	51.9	53.9	56.4	50.1	55.1
		4	68.1	69.7	68.4	81.9	72.8	66.5
		6	68.3	75.1	84.1	111.5	97.3	77.4
	10	2	114.9	91.0	104.6	109.2	90.0	113.0
		4	136.3	101.2	129.5	159.1	110.7	136.5
		6	151.8	135.2	164.4	227.4	205.2	159.9
30	5	2	61.4	64.8	75.4	78.3	62.6	62.1
		4	95.3	87.0	96.6	114.7	90.8	99.5
		6	108.5	99.7	114.3	149.4	125.0	114.7
	10	2	146.7	144.4	155.6	135.6	126.4	139.4
		4	218.7	138.8	195.3	237.9	192.4	205.3
		6	227.4	189.1	236.3	315.6	252.1	241.1

Table 26 Comparison about average CPU time on Group 1 with large-size

Job	Stage	AGV	GA(s)	SA(s)	ABC(s)	GWO(s)	MBO(s)	GATS(s)
80	5	4	232.3	203.6	286.4	346.7	267.1	268.1
		6	325.5	269.3	348.9	449.4	357.2	351.5
		8	404.0	337.6	426.7	566.8	444.2	402.1
160	5	4	706.3	498.1	717.7	861.6	680.7	770.9
		6	901.0	690.3	862.2	1123	948.4	943.6
		8	1048.8	913.8	1083.9	1464.3	1166.5	1011.6

Table 27 Comparison about average CPU time on Group 2

Job	Stage	AGV	GA(s)	SA(s)	ABC(s)	GWO(s)	MBO(s)	GATS(s)
10	2	4	19.1	9.9	14.1	16.8	14.9	14.8
	4	6	38.0	27.2	35.8	46.2	35.5	33.7
	6	6	57.6	39.8	53.1	68.1	56.1	49.9
	8	10	78.0	81.9	93.4	138.0	105.2	85.3
	10	10	94.5	99.7	116.0	170.5	131.9	106.5
20	2	6	32.5	22.3	29.4	38.1	31.3	30.9
	4	8	68.5	59.7	74.3	101.1	81.9	69.8
	6	12	110.3	131.9	145.3	217.0	176.1	138.5
	8	12	146.1	175.9	197.9	297.1	237.1	196.8
	10	14	181.2	255.6	281.6	418.3	344.0	256.1

Table 28 Statistical result of best makespan

Size	α	Total number	Number of optimal solutions					
			GA	SA	ABC	GWO	MBO	GATS
Small	Small	18	3	1	4	2	12	13
Small	Large	18	5	2	1	0	5	9
Large	Small	6	0	1	0	0	0	5
Large	Large	6	0	0	0	1	0	5
Group 2		10	1	1	1	0	3	8
Total		58	9	5	6	3	20	40

Table 29 Statistical result of average makespan

Size	α	Total number	Number of optimal solutions					
			GA	SA	ABC	GWO	MBO	GATS
Small	Small	18	1	0	0	0	6	13
Small	Large	18	2	0	1	0	5	10
Large	Small	6	0	1	0	0	0	5
Large	Large	6	0	0	0	1	0	5
Group2		10	0	0	0	0	5	5
Total		58	3	1	1	1	16	38

decoding method, the algorithm can search for a satisfactory solution within appropriate time.

(2) A hybrid algorithm which hybridizes the genetic algorithm and tabu search has been proposed to solve the integrated scheduling problem. The hybrid algorithm combines the advantages of the two algorithms. The experimental results show that this algorithm has both effectiveness and efficiency for solving integrated scheduling problem.

Although the method proposed in this paper has achieved good results, there are still some works can be made in the future. Firstly, we can use multiple rules instead of single rule in solution evaluation, which may make the results more stable. Secondly, we can design some efficient methods of population initialization instead of the random one, and design efficient neighborhood structure to improve the hybrid algorithm.

Acknowledgements
Not applicable.

Authors' contributions
XL and WL were in charge of the whole trial; WL wrote the manuscript; DH assisted with problem analyses. All authors read and approved the final manuscript.

Authors' Information
Wangming Li, born in 1997, is currently a master candidate at *State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, China*. He received his bachelor degree from *Huazhong University of Science and Technology, China,* in 2019. His main research interests are scheduling problem and algorithm optimization.
Dong Han, born in 1997, is currently a master candidate at *State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, China*. She received his bachelor degree from *Huazhong University of Science and Technology, China,* in 2019. Her main research interests are deep learning and algorithm optimization.
Liang Gao, born in 1974, is currently a professor at *State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, China*. He received his PhD degree in mechatronic engineering from *Huazhong University of Science and Technology, China,* in 2002. His main research interests are intelligent optimization method and its application in design and manufacturing.
Xinyu Li, born in 1985, is currently a professor at *State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, China*. He received his PhD degree in industrial engineering from *Huazhong University of Science and Technology, China,* in 2009. His main research interests are intelligent manufacturing systems, shop scheduling, intelligent optimization and machine learning.
Yang Li, born in 1996, is currently a PhD candidate at *State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, China*. He received his bachelor degree from *Huazhong University of Science and Technology, China,* in 2018. His main research interests are flow shop scheduling and algorithm optimization.

Funding
Supported by National Key R&D Program of China (Grant No. 2019YFB1704603) and National Natural Science Foundation of China (Grant Nos. U21B2029 and 51825502).

Competing interests
The authors declare no competing financial interests.

Received: 21 February 2021 Revised: 15 December 2021 Accepted: 25 January 2022
Published online: 18 February 2022

References

[1] Wei Qin, Zilong Zhuang, Yang Liu, et al. A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly. *Computers & Industrial Engineering*, 2019: 138.
[2] Leilei Meng, Chaoyong Zhang, Xinyu Shao, et al. More MILP models for hybrid flow shop scheduling problem and its extended problems. *International Journal of Production Research*, 2020, 58(13): 3905–3930.
[3] Quan-ke Pan, Ling Wang, Kun Mao, et al. An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process. *IEEE Transactions on Automation Science and Engineering*, 2013, 10(2): 307–322.
[4] Biao Zhang, Quan-ke Pan, Liang Gao, et al. A three-stage multiobjective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020, 50(12): 4984–4999.
[5] Xingyu Li, Baicun Wang, Chao Liu, et al. Intelligent manufacturing systems in covid-19 pandemic and beyond: Framework and impact assessment. *Chinese Journal of Mechanical Engineering*, 2020, 33: 58.
[6] Ning Zhao, Song Ye, Kaidian Li, et al. Effective iterated greedy algorithm for flow-shop scheduling problems with time lags. *Chinese Journal of Mechanical Engineering*, 2017, 30(3): 652–662.
[7] Lin Gui, Liang Gao, Xinyu Li. Anomalies in special permutation flow shop scheduling problems. *Chinese Journal of Mechanical Engineering*, 2020, 33: 46.
[8] Ning Zhao, Siyu Chen, Yanhua Du. Emergency local searching approach for job shop scheduling. *Chinese Journal of Mechanical Engineering*, 2013, 26(5): 918–927.
[9] Guomin Li, Xinyu Li, Liang Gao, et al. Tasks assigning and sequencing of multiple AGVs based on an improved harmony search algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 2019, 10(11): 4533–4546.
[10] T Miyamoto, K Inoue. Local and random searches for dispatch and conflict-free routing problem of capacitated AGV systems. *Computers & Industrial Engineering*, 2016, 91: 1–9.
[11] Dunbing Tang, Min Dai. Energy-efficient approach to minimizing the energy consumption in an extended job-shop scheduling problem. *Chinese Journal of Mechanical Engineering*, 2015, 28(5): 1048–1055.
[12] MK Marichelvam, T Prabaharan, Yang XS. Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan. *Applied Soft Computing*, 2014, 19: 93–101.
[13] Ghiath Al Aqel, Xinyu Li, Liang Gao. A modified iterated greedy algorithm for flexible job shop scheduling problem. *Chinese Journal of Mechanical Engineering*, 2019, 32: 21.
[14] G Ulusoy, Ü BILGE. Simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 1993, 31(12): 2857–2873.
[15] A Ahmadi-Javid, P Hooshangi-Tabrizi. A mathematical formulation and anarchic society optimisation algorithms for integrated scheduling of processing and transportation operations in a flow-shop environment. *International Journal of Production Research*, 2015, 53(19): 5988–6006.
[16] T Nishi, Y Hiranaka, I E Grossmann. A bilevel decomposition algorithm for simultaneous production scheduling and conflict-free routing for automated guided vehicles. *Computers & Operations Research*, 2011, 38(5): 876–888.
[17] A Elmi, S Topaloglu. A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots. *Computers & Operations Research*, 2013, 40(10): 2543–2555.
[18] S Zabihzadeh, J Rezaeian. Two meta-heuristic algorithms for flexible flow shop scheduling problem with robotic transportation and release time. *Applied Soft Computing*, 2016, 40: 319–330.
[19] Y Ho, H Liu. The performance of load-selection rules and pickup-dispatching rules for multiple-load AGVs. *Journal of Manufacturing Systems*, 2009, 28(1): 1–10.
[20] S A Brah, L L Luan. Heuristics for scheduling in a flow shop with multiple processors. *European Journal of Operational Research*, 1999, 113(1): 113—122.
[21] P Lacomme, M Larabi, N Tchernev. A disjunctive graph for the job-shop with several robots. *MISTA conference*, 2007: 285–92.
[22] F Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 1986, 13: 533–549.
[23] Victor Fernandez-Viagas, Josem Framinan. Design of a testbed for hybrid flow shop scheduling with identical machines. *Computers & Industrial Engineering*, 2020: 141.
[24] G Ulusoy, F Sivrikaya-şerifoğlu, Ü Bilge. A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. *Computers & Operations Research*, 1997, 24(4): 335–351.
[25] Yingli Li, Xinyu Li, Liang Gao, et al. An improved artificial bee colony algorithm for distributed heterogeneous hybrid flowshop scheduling problem with sequence-dependent setup times. *Computers & Industrial Engineering*, 2020: 147.
[26] S Mirjalili, S M Mirjalili, A Lewis. Grey wolf optimizer. *Advances in Engineering Software*, 2014, 69: 46–61.
[27] E Duman, M Uysal, A F Alkaya. Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem. *Information Sciences*, 2012, 217: 65–77.
[28] F S Şerifoğlu, G Ulusoy. Multiprocessor task scheduling in multistage hybrid flowshops: a genetic algorithm approach. *Journal of the Operational Research Society*, 2004, 55(5): 504–512.
[29] Jamest Lin, Chun-chih Chiu, Yu-hsiang Chang. Simulation-based optimization approach for simultaneous scheduling of vehicles and machines with processing time uncertainty in FMS. *Flexible Services and Manufacturing Journal*, 2019, 31(1): 104–141.
[30] Two groups of instances for IPTSP. https://github.com/WaminLee/Instances-For-Integrated-Scheduling-Problem. Accessed 20 February 2021.