IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 26, 2019, accepted May 15, 2019, date of publication May 27, 2019, date of current version June 20, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2919109

Approach to Integrated Scheduling Problems Considering Optimal Number of Automated Guided Vehicles and Conflict-Free Routing in Flexible Manufacturing Systems

XIANGFEI LYU, YUCHUAN SONG, CHANGZHENG HE, QI LEI, AND WEIFEI GUO
The State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing, China

Corresponding authors: Xiangfei Lyu (xiangfei113072@163.com) and Yuchuan Song (syc@cqu.edu.cn)

This work was supported in part by the Ministry of Industry and Information of China under Grant CCLS-JB-002, in part by the National Natural Science Foundation of China under Grant 51205429, and in part by the Ministry of Education of China under Grant IRT_15R64.

ABSTRACT Usually, machine and automated guided vehicle (AGV) scheduling are studied simultaneously. However, previous studies often used a fixed number of AGVs or did not consider routing problems and transportation time. This paper focuses on the machine and AGV scheduling problem in a flexible manufacturing system by simultaneously considering the optimal number of AGVs, the shortest transportation time, a path planning problem, and a conflict-free routing problem (CFRP). To study these problems simultaneously, we propose a genetic algorithm combined with the Dijkstra algorithm that is based on a time window. The tri-string chromosome coding method is designed to ensure that the solutions are feasible after the genetic operator has been applied. Global, local, and random searches are adopted in reasonable proportions to improve the quality and diversity of the initial population. The Dijkstra algorithm based on the time window is embedded into the genetic algorithm to search for the shortest route, detect collisions for multiple vehicles simultaneously, and finally, solve the shortest CFRP. The objective is to minimize the makespan while considering the influence of the number of AGVs. Increasing the number of AGVs has a significant impact on the makespan in the initial stage. However, the makespan tends to stabilize as the number of AGVs reaches some threshold. To balance the relationship between the minimum makespan and the optimal number of AGVs, we set the minimum decrease rate to 5% when determining the minimum makespan and confirming the corresponding number of AGVs to be the optimal value. In this paper, to verify the effectiveness of our approach, we propose two sets of computational experiments. The first set of results shows that the proposed algorithm is as efficient and effective at solving the scheduling problem as the benchmark approaches. The second set of computational experiments indicates that the proposed approach is applicable for solving integrated scheduling problems in flexible manufacturing systems.

INDEX TERMS Conflict-free routing problem, genetic algorithm, Dijkstra algorithm, optimal number of AGVs, time window.

I. INTRODUCTION

A flexible manufacturing system (FMS) is an intelligent, highly integrated and cooperative production system. FMSs are widely used in automobile manufacturing, traditional machining and other industries. An FMS generally consists of two basic parts: a material handling system and a manufacturing system. Multifunctional machines are responsible for manufacturing various types of products, and automated guided vehicles (AGVs) are responsible for transporting materials to the machines. Moreover, an FMS may process one operation on multiple machines, which is a more complex operation than the one-operation-one-machine relationship and causes the scheduling when considering the AGV dispatching problem to be more complicated.

Several researchers have studied job-shop scheduling problems of machines and AGVs. Ulusoy and Bilge [1]

The associate editor coordinating the review of this manuscript and approving it for publication was Le Hoang Son.

proposed a scheduling algorithm to integrate AGV scheduling with overall activity scheduling in an FMS environment and exploit the interactions between the two aspects. Ulusoy and Bilge [2] assumed the absence of vehicle collisions, formulated a nonlinear mixed integer-programming model, and proposed a heuristic algorithm based on a time window to exploit the interactions between the machine scheduling and material handling system. Ulusoy*et al.* [3] proposed a genetic algorithm to solve the machine scheduling and AGV scheduling problem simultaneously. Abdelmaguid *et al.* [4] proposed a genetic algorithm for machine scheduling and a heuristic algorithm for AGV scheduling to solve the two scheduling problems simultaneously. Subsequently, hybrid algorithms have replaced simple algorithms for the simultaneous scheduling of machines and AGVs in an FMS. Hou *et al.* [5] adopted Pareto-optimization via genetic algorithm and develop a chromosome that can describe a feasible schedule such that meta-heuristics can be applied. Deroussi *et al.* [6] proposed the following three metaheuristics for efficient neighborhood scheduling, processing, and transportation: iterated local search, simulated annealing and hybridization of the two. Yang *et al.* [7] reviewed the high-volume low-mix process and high-volume high-mix process by focusing on the quantity production for manufacturing system scheduling. Vilcot and Billaut [8] proposed using a tabu search and a genetic algorithm based on NSGA-II to solve job-shop scheduling under multiple constraints. Fermin Montane and Galvao [9] proposed a tabu search for the vehicle routing problem with simultaneous pick-up and delivery (VRP_SPD) and used three types of movements in their algorithm: relocation, interchange, and crossover. Their results exhibited an improvement with respect to the tested data. Chang *et al.* [10] used a Taguchi-genetic algorithm to solve the flexible job-shop scheduling problem with makespan optimization.

However, the above studies focused primarily on machines and AGV scheduling while ignoring vehicle collisions and routing problems.

To obtain more realistic makespan results, some researchers have focused on the routing problem while considering the transportation time. Kim and Tanchoco [11] addressed the conflict-free routing problem (CFRP) and proposed an efficient algorithm based on the Dijkstra algorithm to determine the conflict-free, shortest-time route for AGVs moving in a bidirectional flow-path network. Wu and Zhou [12] introduced a system modeling with Petri nets and a deadlock avoidance policy to find the shortest conflict-free routes. Gen *et al.* [13] investigated the possibility of using genetic algorithms to solve the shortest-path problem and proposed a priority-based encoding method to present all possible paths. Xing *et al.* [14] embedded the optimal deadlock avoidance policy into the genetic algorithm and developed a novel deadlock-free genetic scheduling algorithm for AMSs. The author used the one-step look-ahead control policy to check the feasibility of the chromosome. Reveliotis [15] proposed a strategy that both ensures conflict resolution and maintains the operational flexibility provided by a free vehicle travelling on an arbitrarily structured guide-path network. Qiu and Hsu [16] presented a bidirectional path layout and a routing algorithm for AGVs. Li *et al.* [17] introduced time window constrains and a collision resolution mechanism to address both job scheduling and collision resolution issues of multi-bridge machining systems. Krishnamurthy *et al.* [18] solved the makespan problem in a bidirectional network in a conflict-free manner via column generation. Möhring and Köhler *et al.* [19] proposed an algorithm that avoids collisions, deadlocks, and livelocks prior to the computation for CFRPs. Gen [20] combined various hybrid genetic algorithms to address a wide range of problems, such as the logistics network model, vehicle routing problem and AGV dispatch problem. Mareda *et al.* [21] assessed the optimal spatial distribution of renewable units through a parameterized optimization method based on a genetic algorithm. Desaulniers *et al.* [22] proposed a column generation approach to solve dispatching and CFRPs simultaneously in FMS. Miyamoto *et al.* [23] proposed a cooperative algorithm that obtains efficient and deadlock-free routes despite the small buffer capacity of autonomous distributed manufacturing systems. Subsequently, Miyamoto and Inoue [24] formulated the dispatching problem and the CFRP for a capacitated AGV system as an integer program and proposed local/random search methods to solve the CFRP. Nishi and Tanaka [25] proposed a Petri net decomposition approach for simultaneous dispatching and the CFRP for dynamic situations. Novas and Henning [26] dealt with several critical features with the following sub-problems: machine loading, manufacturing activities scheduling, part routing, machine buffer scheduling, tool planning, allocation, and AGV scheduling. A constrain programming was tested and various examples demonstrated the importance. Fazlollahtabar and Hassanli [27] investigated the simultaneous scheduling and routing problem for autonomous guided vehicles. Kaboudani *et al.* [28] investigated a vehicle routing problem in a distribution network with a cross-docking center with minimum transport cost by considering both forward and reverse logistics in an integrated model. Meng *et al.* [29] compared a variety of Petri net-based deadlock prevention policies in terms of structural complexity, behavior permissiveness and computational complexity to facilitate engineers in choosing a suited method. Fazlollahtabar *et al.* [30] proposed a mathematical program to minimize penalizing earliness and tardiness for conflict-free and just-in-time production. Maza and Castagna [31] proposed the following two classes of routing algorithms: optimized preplanning algorithms and real-time routing algorithms. A preplanning algorithm was advantageous for conflict-free routes but not for improving vehicle delays and failures; real-time algorithms have the advantage of being reactive but are non-optimal. The above studies mainly focused on the vehicle routing problem (VRP) and the CFRP and largely ignored job-shop scheduling problems, or they assumed the scheduling problem to be optimal. However, the optimal path planning may not match

the optimal scheduling and may not result in an optimum makespan value.

As the transportation resource, there are three reasons to optimize the number of AGVs: reduce transportation equipment costs, reduce congestion caused by a large number of AGVs in the production system and increase AGV utilization. Mahadevan and Narendran [32] developed an analytical model to estimate the number of AGVs that considered both AGV utilization and production volume. Kasilingam and Gobal [33] proposed a cost model that determined the number of AGVs based on the idle-time costs of vehicles, machines, and delays while waiting for parts. Vis *et al.* [34] proposed a minimum flow algorithm to determine the number of AGVs required in a containerized transportation problem. They developed a strong polynomial-time algorithm to solve the case when containers are available for transport at known time instants. Li *et al.* [35] proposed an improved population-based incremental learning algorithm to plan collision-free cutting paths of multi-bridge water-jet cutting processes for solving both the interference and routing problems. Vivaldini *et al.* [36] presented a methodology for estimating the minimum number of AGVs. They considered the CFRP but did not mention the scheduling problem and manufacturing system. Mousavi *et al.* [37] developed a mathematical model that integrated an evolutionary algorithm, particle swarm optimization, and a hybrid to optimize the task scheduling of AGVs by minimizing the makespan and the number of AGVs while considering the AGVs' battery charge. However, they did not consider the CFRP. An example assessment demonstrated the makespan minimization and the selection of the number of AGVs, and a comparison of the optimal results provided the mean AGV operational efficiency of the three proposed algorithms.

Some researchers have investigated the AGV/machine scheduling problem and the CFRP simultaneously. Correa *et al.* [38] proposed a hybrid method to solve dispatching and the CFRP in FMS by using constraint programming to solve the scheduling master problem and a mixed integer-programming solver to address the routing problem. Khayat *et al.* [39] proposed an integrated formulation for machine and material handling systems and considered machines and vehicles as constrained resources. However, Khayat, Langevin and Riopel did not thoroughly study the influence of the number of vehicles; thus, computational experiments may yield better results. Nishi *et al.* [40] proposed a bilevel decomposition algorithm to solve the scheduling problem and the CFRP simultaneously through decomposition of a mixed integer formulation. Saidi-Mehrabad *et al.* [41] proposed an ant colony algorithm to solve an integrated model of job-shop scheduling by considering the transportation time and the CFRP. Umar *et al.* [42] proposed a hybrid genetic algorithm to integrate scheduling, dispatching, and conflict-free routing in an FMS environment. In their pseudocode, machine/AGV scheduling and path planning are performed first, followed by the detection and avoidance of route conflicts; however, calculating each cycle in this approach requires a considerable amount of time. Fu *et al.* [43] studied production scheduling and vehicle routing with job splitting and a delivery time window. The jobs were processed on unrelated parallel machines. They developed a two-phase iterative heuristic to solve the integrated scheduling problem and evaluated the benefits.

Most previous studies considered machine/AGV scheduling in the following forms:

1. The shortest route is assumed to be known, and path planning is absent, which is a reasonable approach for a system with a single AGV. However, when two or more vehicles are included in the production system, collision problems during transit must be considered. Therefore, this theory lacks generality.
2. AGV path planning and the CFRP are considered after AGV/machine scheduling. This strategy divides the solution into two steps. First, the optimal solution for AGV/machine scheduling is determined; then, the vehicle routing problem is solved. However, this method cannot ensure the optimality of both the scheduling solutions and the path planning.
3. AGV path planning is considered simultaneously with AGV/machine scheduling, but with a fixed number of AGVs. However, a fixed number of AGVs may waste transportation resources or be unable to satisfy the required delivery time. Thus, the obtained solutions may be applicable only for the current constraint, and the use of one more or one less vehicle might yield better results.

In this study, we focus on machine and AGV scheduling considering the optimal number of AGVs, the CFRP and the transportation time in FMS. To solve these problems simultaneously, we propose using a genetic algorithm and a Dijkstra algorithm based on a time window to obtain an approximately optimal solution. To obtain a good initial population, we propose using global, local and random search strategies in the genetic algorithm. The Dijkstra algorithm is used to search the globally shortest route of the AGVs to determine the shortest transportation time for the current feasible scheduling scheme. The time window is used to dynamically detect the available vehicle routes. To verify the effectiveness of our approach, we test a set of benchmark problems and compare the results. To study the influence of the number of AGVs on the makespan, we use a set of computational experiments to determine the optimal number of AGV for different production systems sizes. Finally, when the makespan tendency becomes stable according to the decrease rate as the number of AGVs in the system is increasing, the optimal number of AGVs is determined.

The remainder of this paper is organized as follows: Section II describes the proposed problem and constructs the mathematical model. Section III presents the proposed method, and Section IV reports the results of two sets of computational experiments. Finally, Section V concludes this paper.

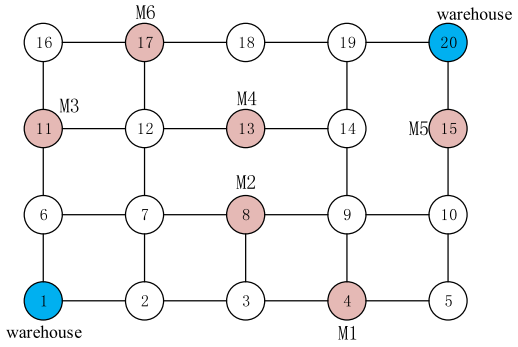


FIGURE 1. Example production system network.

II. PROBLEM DEFINITION AND MATHEMATICAL MODEL

In an intelligent production system, machines are scattered around the production network, and AGVs are responsible for transporting materials to machines and warehouses. The objective of the scheduling problem is to assign tasks to suitable machines and AGVs to suitable tasks. However, a global optimal machine scheduling may not have an optimal shortest route planning, or the optimal shortest route planning may have many collisions and cost extra time. In addition, the number of AGVs affects the quality of the solution. When the production system includes only one vehicle, there is no need to consider the collision problem, and the utilization rate is always close to 100%. When the system includes more than one vehicle, the possibility of collisions occurs, and the utilization rate decreases. The makespan decreases and tends to become stable as the number of AGVs increases. Therefore, it is difficult to obtain the globally optimal solution when considering the number of AGVs, especially for complex production systems. However, we can find an approximately optimal solution for the problem.

In an FMS environment, the layout is represented by a bidirectional network, but only one vehicle at a time can transit a segment, as shown in Figure 1. We assume that there are m machines located in the network to service n jobs, and each job includes more than one operation. Each operation can be processed on a few different machines but with different processing times, and each machine can perform a few different operations. Figure 1 shows the layout of the FMS used in this study. Six machines are located at points $N_4, N_8, N_{11}, N_{13}, N_{15}$ and N_{17}. The roughcast warehouse is located at N_1. The finished products warehouse is located at point N_{20}. Three vehicles are initially located at the roughcast warehouse. All the finished products are transported to the finished products warehouse after the final operation. The vehicles are identical and are responsible for transporting the jobs from the roughcast warehouse at N_1 to the assigned machines and finally to the finished products warehouse at N_{20}.

Based on the above characteristics, we make the following assumptions in our study:

1. Each machine can process only one operation at a time.
2. Each operation can be processed by one machine.
3. Loading and unloading times are considered in the operation time.
4. AGVs move at a constant speed.
5. Each AGV is responsible for one task at a time.
6. All the AGVs are always available, and battery charging is not considered.
7. There are no sequence constraints between different jobs.
8. Interruptions are not allowed during a process or during transport.

The following notation and variables are used in the mathematical model:

- J_i : set of jobs, $i \in (1, 2, \ldots, n)$, $n =$ number of jobs;
- M_k : set of machines, $k \in (1, 2, \ldots, m)$, $m =$ number of machines;
- K_v : set of vehicles, $v \in (1, 2, \ldots, w)$, $w =$ number of vehicles;
- O_{ij} : indicates the jth operation of J_i, $j \in (1, 2, \ldots, u_i)$, $u_i =$ number of operations of J_i;
- C_i : indicates the completion time of J_i;
- S_{ijk} : indicates the start time of O_{ij} on machine k;
- t_{ijk} : indicates the processing time of O_{ij} on machine k;
- C_{ijk} : indicates the completion time of O_{ij} on machine k;
- T_{ijv} : indicates the transit task of delivering O_{ij} to the corresponding machine by vehicle v;
- $t_{kk'}$: indicates the transit time between machine k and machine k';
- LT_{ijv} : indicates the starting location of a machine for transit task T_{ijv};
- $S'T_{ijv}$: indicates the start time of an empty trip for transit task T_{ijv};
- $C'T_{ijv}$: indicates the completion time of an empty trip for transit task T_{ijv};
- ST_{ijv} : indicates the start time of a loaded trip for transit task T_{ijv}
- CT_{ijv} : indicates the completion time of a loaded trip for transit task T_{ijv};
- N_s : indicates the set of points $s \in (1, 2, \ldots, p)$, where p is the number of points in the graph network
- M: indicates a sufficiently large positive number
- $\alpha_{ijk} = \begin{cases} 1 & \text{if } O_{ij} \text{ is operated on machine } k \\ 0 & \text{otherwise} \end{cases}$
- $\beta_{ijpqk} = \begin{cases} 1 & \text{if } O_{ij} \text{ is operated before } O_{pq} \\ & \text{on the machine } k \\ 0 & \text{otherwise} \end{cases}$
- $x_{ijv} = \begin{cases} 1 & \text{if vehicle } k_v \text{ is responsible for transit task } T_{ijv} \\ 0 & \text{otherwise} \end{cases}$
- $y_{ijv} = \begin{cases} 1 & \text{if vehicle } k_v \text{ is } S'T_{i'j'v} \geq CT_{ijv} \\ & \forall i, i' \in \{1, 2, \ldots, n\}, \quad \forall j, j' \in \{1, 2, \ldots, P_i\}, \\ & \forall v \in \{1, 2, \ldots, w\} \\ & \text{responsible for transit task} T_{ijv} \\ 0 & \text{otherwise} \end{cases}$

- $z_{vst} = \begin{cases} 1 & \text{if vehicle } k_v \text{ is on the node } N_s \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$
- $A_{s_1 s_2 t} = \begin{cases} 1 & \text{if feasible path from node } s_1 \\ & \text{to the adjacent node } s_2 \text{ at time } t,\ s_1 \neq s_2 \\ 0 & \text{if not feasible} \end{cases}$
- $V_{s_1 s_2 vt} = \begin{cases} 1 & \text{if vehicle } v \text{ is on the way from node } s_1 \text{ to} \\ & \text{the adjacent node } s_2 \text{ at time } t,\ s_1 \neq s_2 \\ 0 & \text{if another vehicle is traveling on the} \\ & \text{same path from the opposite direction.} \end{cases}$

To ensure that all the machine scheduling rules and vehicle dispatching rules are satisfied, we establish a set of mathematical models. The optimization objective is to minimize the makespan C_{max}.

The mathematical model is as follows:

$$C = \min(C_{\max}) = \min(\max_{i=1}^{n}(C_i)) \tag{1}$$

Subject to

$$\sum_{k=1}^{m} \alpha_{ijk} = 1, \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\} \tag{2}$$

$$S_{pqk} + M(1 - \beta_{ijpqk}) \geq C_{ijk} \quad \forall i, p \in \{1, 2, \ldots, n\},\ \forall j, q \in \{1, 2, \ldots, P_i\},\ \forall k \in \{1, 2, \ldots, m\} \tag{3}$$

$$\sum_{v=1}^{w} x_{ijv} = 1, \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\} \tag{4}$$

$$C_{ijk} = S_{ijk} + t_{ijk} \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\},\ \forall k \in \{1, 2, \ldots, m\} \tag{5}$$

$$S^{'} T_{i'j'v} \geq CT_{ijv} \quad \forall i, i' \in \{1, 2, \ldots, n\},\ \forall j, j' \in \{1, 2, \ldots, P_i\},\ \forall v \in \{1, 2, \ldots, w\} \tag{6}$$

$$S^{'} T_{ijv'} \geq CT_{i(j-1)v} + \sum_{k=1}^{m} \alpha_{i(j-1)k} \cdot t_{i(j-1)k} \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\},\ \forall v, v' \in \{1, 2, \ldots, w\},\ \forall k \in \{1, 2, \ldots, w\} \tag{7}$$

$$CT_{ijv} \geq ST_{ijv} + \sum_{k=1}^{m} \sum_{k'=1}^{m} \alpha_{i(j-1)k'} \cdot \alpha_{ijk} \cdot t_{kk'} \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\},\ \forall k, k^{'} \in \{1, 2, \ldots, m\},\ \forall v \in \{1, 2, \ldots, w\} \tag{8}$$

$$ST_{ijv} \geq \max(C^{'} T_{ijv}, C_{i(j-1)}),\ \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\} \tag{9}$$

$$S_{ijk} \geq \max\{CT_{ijv}, \max\{C_{pqk} \mid \forall p = 1, 2, \ldots, n, q \neq j\}\} \quad \forall i \in \{1, 2, \ldots, n\},\ \forall j \in \{1, 2, \ldots, P_i\},\ \forall k \in \{1, 2, \ldots, m\},\ \forall v \in \{1, 2, \ldots, w\} \tag{10}$$

$$\sum_{v=1}^{w} z_{vst} \leq 1, \quad \forall s \in \{1, 2, \ldots, p\},\ \forall t \in C \tag{11}$$

The goal of the objective function (1) is to minimize the makespan. Constraint (2) ensures that one machine can process only one operation at a time. Constraint (3) ensures that the start time of an operation is later than the completion time of the preceding operation; this ensures that no time collision will occur between two adjacent operations on the same machine. Constraint (4) ensures that one operation is assigned to only one AGV. Constraint (5) ensures that no interruption occurs during the process. Constraints (6)–(9) ensure that no time conflicts occur during the scheduling. Constraint (6) ensures that a transit task commences immediately after the completion of the previous transit task finished. Constraint (7) ensures that the start times of empty trips are greater than or equal to the completion time of the preceding loaded trip and its operation time, which means that an empty trip for the succeeding operation starts only after the current operation completes. Constraint (8) ensures that the completion time of the empty trip is greater than or equal to the trip start time plus the transit time between the two related machines. Constraints (9) and (10) ensure that the sequence of operations and transit tasks is feasible. Constraint (9) ensures that the transit task of a subsequent operation starts only after the preceding operation is complete and the corresponding vehicle has arrived, and constraint (10) ensures that the operation starts only after the transit task is completed. Constraint (11) ensures that a node can be occupied by only one vehicle at a given time.

During production, if an empty trip vehicle arrives at the assigned machine before the current task is completed ($C_{i(j+1)k} > C^{'} T_{ijv}$) or a machine completes its current task before the arrival of the corresponding vehicle $S_{pqk} \cdot \beta_{pqijk} > CT_{ijv}$, these two situations are called ''invalid waiting time (IWT),'' and they ultimately reduce the utilization ratio of the AGVs and increase the makespan. In constraint (12), IWT denotes the sum of the invalid waiting times of all vehicles and machines in the production system:

$$IWT = \sum_{i,p=1}^{n} \sum_{j,n=1}^{u} \left\{ \left(C_{i(j+1)k} - C^{'} T_{ijv} \right) + \left(S_{pqk} \cdot \beta_{pqijk} - CT_{ijv} \right) \right\} \tag{12}$$

III. GENETIC ALGORITHM FOR SCHEDULING PROBLEM

Genetic algorithms are widely used to solve scheduling problems because of their good robustness and extensibility. This paper embeds a Dijkstra algorithm based on a time window into the genetic algorithm to solve the scheduling and routing problems simultaneously. Compared with hybrid-PSO (Particle Swarm Optimization), the genetic algorithm includes genetic operators: a mutation operator and a crossover operator, which help ensure population diversity. The genetic algorithm has a mature method of convergence analysis to avoid premature convergence. Tabu search simulates the human memory process and has a strong capability in local development but a weak capability in global development. Therefore, tabu search usually converges quickly but is more likely than a genetic algorithm to obtain a local optimal solution.

Essentially, a collision is a partial or total overlap of two or more vehicles transiting the same route section within the same time period. The Dijkstra algorithm is used to obtain the global shortest route, and the time window is used to

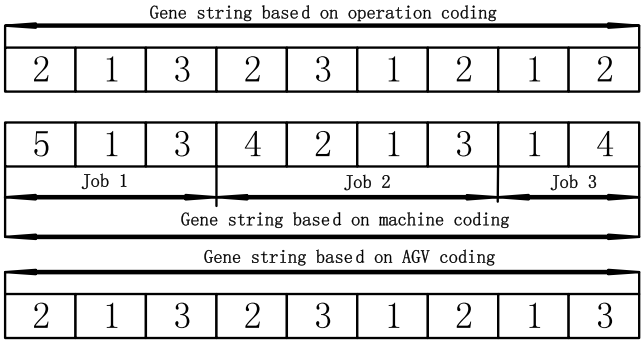


FIGURE 2. Example of the tri-string chromosome coding method.

detect whether a route is available. The search efficiency of Dijkstra is lower than those of heuristic approaches, such as the A $*$ algorithm, simulated annealing algorithm and similar algorithms. However, the A $*$ algorithm may result in more collisions due to is directional search strategy, while the Dijkstra algorithm always finds the global shortest routing while considering the collision problem. However, the Dijkstra algorithm suffers from an exponential increase in the computation time as the problem scale increases. Therefore, the Dijkstra algorithm is suitable for only medium and small-sized problems.

A. CODING

Based on the above assumptions, a layered coding method based on the operations is proposed. The coding consists of three parts: coding based on operations, coding based on machines and coding based on AGVs, as shown in Figure 2. The same location of operation gene string and AGV gene string indicate a corresponding relationship in which the AGV is responsible for the transit task of the operation in that location. We divide the machine gene string into sections according to the number of jobs and operations. A value in the gene string denotes each machine to which the operation is assigned. Therefore, the proposed coding method always yields feasible solutions for machine/AGV scheduling in an FMS environment. For example, the first column in Figure 2 indicates that the No. 2 vehicle is responsible for the transit task of O_{21} and the machine 5 process O_{11}. To find a better initial population and achieve fast convergence, global, local and random search strategies are used: 70% of the population adopt a global search strategy, 20% of the population adopt a local search strategy, and 10% of the population adopt a random search strategy.

B. INITIAL POPULATION

To balance the load and ensure the machine utilization ratio, two search methods, global search and local search, are adopted in this paper. To ensure diversity, a random search is adopted in a small proportion of the population.

Global search strategy: Create an array with the same length as the total number of machines and the same operational time sequence as the assigned machines. Select a job randomly from the first operation and add the operation time of the alternative machines to the number in the array based on the machine sequence. Select the minimum sum as the assigned machine and update the array. Then, select the next operation and perform the same steps until all the jobs are selected. This ensures that all operations are selected and that the machines with the minimum operational time are selected to maintain the machine load balance. Table 3 presents an example.

Step 1: Initialization

Array of operation time chromosome of operations

[0 0 0 0 0 0] [0 0 0 0 0 0 . . .]

Step 2: Choose job 3 randomly, select O_{31}, the operation time and the alternative machines.

M1 M3 [5 6] [**5** 0 **6** 0 0 0]

Select machine 1 and update the array.

[5 0 0 0 0 0] [1 0 0 0 0 0 . . .]

Step 3: Select O_{32}, the operation time and the alternative machines.

M2 M5 [8 7] [5 **8** 0 0 **7** 0]

Select machine 5 and update the array.

[5 0 0 0 7 0] [1 5 0 0 0 0 . . .]

Step 4: Select O_{33}, the operation time and the alternative machines.

M1 M6 [10 9] [**15** 0 0 0 7 **9**]

Select machine 6 and update the array.

[5 0 0 0 7 9] [1 5 6 0 0 0 . . .]

Step 5: Select O_{34}, the operation time and the alternative machines.

M2 M4 [12 9] [5 **12** 0 **9** 7 9]

Select machine 4 and update the array.

[5 0 0 9 7 9] [1 5 6 4 0 0 . . .]

Step 6: Select O_{35}, the operation time and the alternative machines.

M3 [7] [5 0 **7** 9 7 9]

Select machine 3 and update the array.

[5 0 7 9 7 9] [1 5 6 4 3 0 . . .]

Step 7: Choose job 1 randomly, select O_{11},the operation time and the alternative machines.

M1 M2 M3 [6 7 10] [**11** **7** **17** 9 7 9]

Select machine 2 and update the array.

[5 7 7 9 7 9] [1 5 6 4 3 2 . . .]

Step 8: Repeat the preceding steps until all the jobs have been selected.

Step 9: Initialize the array, and perform the next iteration.

Local search: the principle of the local search is similar to that of the global search, but the operation time array

initialization is performed after all the job operations have been selected.

Step 1: Initialization

Array of operation time chromosome of operations

[0 0 0 0 0 0] [0 0 0 0 0 0 . . .]

Step 2: Choose job 3 randomly, select O_{31}, the operation time and the alternative machines.

M1 M3 [5 6] [**5** 0 **6** 0 0 0]

Select machine 1 and update the array.

[5 0 0 0 0 0] [1 0 0 0 0 0 . . .]

Step 3: Select O_{32}, the operation time and the alternative machines.

M2 M5 [8 7] [5 **8** 0 0 **7** 0]

Select machine 5 and update the array.

[5 0 0 0 7 0] [1 5 0 0 0 0 . . .]

Step 4: Select O_{33},the operation time and the alternative machines.

M1 M6 [10 9] [**15** 0 0 0 7 **9**]

Select machine 6 and update the array.

[5 0 0 0 7 9] [1 5 6 0 0 0 . . .]

Step 5: Select O_{34}, the operation time and the alternative machines.

M2 M4 [12 9] [5 **12** 0 **9** 7 9]

Select machine 4 and update the array.

[5 0 0 9 7 9] [1 5 6 4 0 0 . . .]

Step 6: Select O_{35},the operation time and the alternative machines.

M3 [7] [5 0 **7** 9 7 9]

Select machine 3 and update the array.

[5 0 7 9 7 9] [1 5 6 4 3 0 . . .]

Step 7: Choose job 1 randomly, select O_{11}, the operation time and the alternative machines, and initialize the operation time array.

M1 M2 M3 [6 7 10] [**6** **7** **10** 0 0 0]

Select machine 2 and update the array.

[6 0 0 0 0 0] [1 5 6 4 3 1 . . .]

Step 8: Repeat the above steps until all the jobs have been selected.

In a flexible manufacturing system, operations always have alternative machines, and a single searching method may lead to a local optimal solution. A scheduling scheme with an unbalanced machine load cannot be the optimal solution. Global search and local search consider the machine load balance and thus improve the quality of the initial population and achieve a fast convergence speed that most likely yields the optimal solution. To escape from local optimal solutions, the random search is used to increase the diversity of the initial population.

C. CROSSOVER OPERATOR

Based on coding characteristics, Zhang *et al.* [44] proposed the following two types of crossover operators: the improved precedence operation crossover (IPOX) (the X represents the ''crossover'') for the processing sequence and AGV chains to ensure an adequate number of vehicles and the multipoint preservative crossover (MPX) for the machine coding.

D. MUTATION OPERATOR

The mutation operator simulates the variability of biological evolution and increases population diversity. Thus, it is used to avoid premature convergence to a local optimal solution. In this paper, we propose two types of mutation operators: operation mutation (OM) and machine mutation (MM). The OM operator randomly selects two operation codes from the chromosome and exchanges their locations. To fix the assigned AGV, the same mutation operator is adopted for the AGV chain. The MM operator selects an alternative machine from the alternative machine set for a corresponding operation, replacing the current machine.

E. DECODING PROCESS FOR THE SCHEDULING AND CONFLICT-FREE ROUTING PROBLEM

Based on the chromosome coding approach, it is possible to obtain all the information of each gene for each chromosome; this includes the operation O_{ij}, the adopted machine number M_k, the assigned vehicle number K_v for the transit task, the machine location l, and the locations of the starting points S and T. We take the obtained information as the input criterion of the Dijkstra algorithm to achieve feasible conflict-free path and an acceptable AGV travel time through the conflict detection strategy. Finally, the chromosome fitness value is calculated based on the traveling and operation times. The steps are shown below.

Step 1: Convert the gene-string-based operation coding from the chromosome to the operation string.

Step 2: Read the gene in the operation string from left to right. For example, if the corresponding gene is operation O_{ij}, then we know the assigned machine M_k for O_{ij}, the operation time t_{ijk}, the completion time $C_{i(j-1)}$ for the preceding operation $O_{i(j-1)}$, the vehicle K_v to be used for the transit task, the location of machine M_k, and the start time $S^{'}T_{ij}$ of the empty trip.

Step 3: Plan the route for the empty trip. We use the Dijkstra algorithm to calculate the next-shortest path node N according to the matrix $A_{dj}M$, with the start point l, end point S, vehicle number K_v, and available start time $S^{'}T_{ij}$ for the empty trip. Then, execution skips at step 6.

Step 4: Plan the route for a loaded trip. We use the Dijkstra algorithm based on the time window to find an available route for an empty trip from l to S and calculate the completion time $C^{'}T_{ij}$ of the empty trip after a vehicle arrives at node S. We compare the sizes of $C_{i(j-1)}$ and $C^{'}T_{ij}$. If $C_{i(j-1)} > C^{'}T_{ij}$, then the start time ST_{ij} of the loaded trip for the transit task by vehicle K_v is equal to $C_{i(j-1)}$; otherwise, $ST_{ij} = C^{'}T_{ij}$.

Step 5: We use the Dijkstra algorithm to calculate the next-shortest path node N according to the matrix $A_{dj}M$, with the start point S, end point T, vehicle number K_v, and start time ST_{ij} for a loaded trip.

Step 6: Detect collisions. We test whether the route is feasible based on the time window. If so, execution goes to step 7; otherwise, execution skips to step 8.

Step 7: Update the time window of the network. For an empty trip, if $N = S$, we end the step, output the shortest path and the time window of the network, and execution returns to step 4. For a loaded trip, if $N = T$, we end the step, output the shortest path and time window of the network, and execution returns to step 10. If N is neither S nor T, execution returns to step 6 to find the next-shortest path node N.

Step 8: Detect the collision type. The conflict type is detected according to the time window. Three types of conflicts are considered in this study: point, opposite direction, and road junction. For a point conflict, when two vehicles arrive at a corner at the same time, one vehicle is selected randomly to pause for a brief time to avoid a point conflict. In actual situations, the probability of a point conflict is very low. For an opposite direction conflict, when one vehicle occupies a section of road, another vehicle cannot traverse that road from the opposite direction, which immediately turns into a road junction conflict. For road junction conflicts, we use a simple function to determine whether a vehicle should wait until the path is clear or seek another (suboptimal) route. We assume that t_1 is the primary time for the optimal route when a route conflict is not considered, t_2 is the time for the suboptimal route, and Δt is the waiting time of the optimal route when route conflict is considered:

$$t_1 + \Delta t > t_2. \tag{13}$$

When constraint (13) is satisfied, the vehicle will select a suboptimal route; otherwise, the vehicle waits until the original route is available. Figure 3 illustrates the two types of collisions. Figure 3(a) shows the situation that can be solved by (13). Figure 3(b) shows another situation in which a collision cannot be avoided. In this case, the vehicle that arrives at the assigned point earlier selects a suboptimal route, while the later-arriving vehicle continues on the original route. Constraint (14) ensures that at least one path is feasible when two vehicles moves toward the same junction point at the same time:

$$\begin{aligned} &if\ V_{s_1s_2v_1t} \cdot V_{s_3s_2v_2t} = 1, \quad \forall v_1, v_2 \in (1, 2, \ldots, w) \\ &\quad \forall s_1, s_2, s_3 \in (1, 2, \ldots, p), \quad \forall t \in C \\ &\quad then\ A_{s_2s_1t+1} + A_{s_2s_3t+1} \geq 1 \end{aligned} \tag{14}$$

Step 9: Update the time window of the network. For an empty trip, when $N = S$, we end the step, output the shortest path and time window of the network, and return to step 4. For a loaded trip, when $N = T$, we end the step, output the shortest path and time window of the network, and return to step 10. Otherwise, we select an alternative suboptimal route and return to step 6.

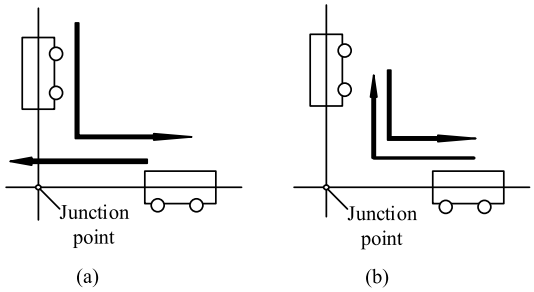


FIGURE 3. Junction collision situation.

Step 10: Calculate the start and end times of the transit and processing tasks for the current operation according to the optimal route and operation time.

Step 11: Repeat steps 2–10 until all tasks are completed and then obtain the detailed start and completion times for all operations.

F. COMPUTATIONAL COMPLEXITY

The main computational burden of the algorithm is dominated by the step 8 solution of the collision types. The conflict-free shortest time procedure proposed by Vilcot and J. Billaut [8] has a polynomial computation complexity o $\left(V^4N^2\right)$, where V is total number of AGVs and N is the number of the node in the network. Mareda *et al.* [21] proposed a robust AGV routing by delaying some AGVs following the late AGV based on the method by Vilcot and J. Billaut [8] and obtained a computational complexity o $\left(\eta v^3\right)$, where η is the number of nodes in the guide-path and v is the number of times that one node can be crossed by vehicles.

In this paper, let ϑ be the total number of operations in the system, NL is the number of collisions that one vehicle may encounter during the transportation task for an operation, which means that the outer-loop has to execute detection procedure the equal number of times. Thus, the algorithm has the complexity of o($\vartheta \cdot$ NL). Suppose that there are δ vehicles in the system, $(\delta - 1)$ collisions can be encountered during the path for one vehicle in the worst case. If each collision is the type that vehicle has to compare the waiting time and the transport time of other feasible path, in the worst case, the other feasible path encounter $(\delta - 2)$ identical type of collisions and so on. So $\delta\,(\delta - 1)/2$ collisions can be encountered in one collision, and there are $(\delta - 1)$ collisions in the worst case. The number of NL is equal to $\delta\,(\delta - 1)^2/2 \approx \delta^3$. As a conclusion, the computational complexity is o $\left(\vartheta\delta^3\right)$. The computational complexity denote that the number of AGVs is much more sensitive than the size of network.

IV. COMPUTATIONAL EXPERIMENT

To verify the effectiveness of the proposed algorithm, we conduct two sets of experiments. The first set validates the effectiveness of the algorithm for the machine and AGV scheduling problem. We take the number of AGVs as the decision variable, which is consistent with the benchmark

TABLE 1. Results comparison when t/p > 0.25.

Prob.no	LB by Ulusoy	LB by Zheng	STW	UGA	AGA	RGA	ZGA	PGA	PGA-B
EX11	72	72	96	96	96	96	96	96	96
EX21	86	86	105	104	102	100	100	100	100
EX31	81	81	105	105	99	99	99	99	99
EX41	62	76	118	116	112	112	112	112	112
EX51	60	60	89	87	87	87	87	87	79*
EX61	96	96	120	121	118	118	118	118	118
EX71	76	76	119	118	115	111	111	111	111
EX81	146	146	169	152*	161	161	161	161	161
EX91	93	93	120	117	118	116	116	116	112*
EX101	124	124	153	150	147	147	146*	150	150
EX12	66	68	82	82	82	82	82	82	69*
EX22	76	76	80	76	76	76	76	76	76
EX32	75	75	88	85	85	85	85	85	82*
EX42	60	64	93	88	88	87	87	87	83*
EX52	54	59	69	69	69	69	69	69	58*
EX62	86	86	100	98	98	98	98	98	98
EX72	74	74	90	85	79	79	79	79	79
EX82	140	140	151	142*	151	151	151	151	151
EX92	91	91	104	102	104	102	102	102	102
EX102	114	114	139	137	136	135	135	135	135
EX13	64	66	84	84	84	84	84	84	69*
EX23	82	82	86	86	86	86	86	86	84*
EX33	77	77	86	86	86	86	86	86	80*
EX43	58	66	95	91	89	89	89	89	78*
EX53	52	57	76	75	74	74	74	74	58*
EX63	88	88	104	104	104	103	103	103	97*
EX73	76	76	91	88	86	83	83	83	77*
EX83	142	142	153	143*	153	153	153	153	153
EX93	93	93	110	105	106	105	105	105	93*
EX103	116	116	143	143	141	139	137	139	134*
EX14	68	68	108	103	103	103	103	103	88*
EX24	84	84	116	113	108	108	108	108	88*
EX34	81	84	116	113	111	111	111	111	95*
EX44	62	76	126	126	126	126	121	126	107*
EX54	56	56	99	97	96	96	96	96	81*
EX64	90	90	120	123	120	120	120	120	112*
EX74	76	76	136	128	127	126	126	127	109*
EX84	148	148	163	163	163	163	163	163	161*
EX94	91	91	125	123	122	122	120	122	117*
EX104	120	120	171	164	159	158	157	158	146*

Notes: LB by Ulusoy: Lower bound by Ulusoy, SIlvrikaya-Serifoglu and Bilge [3]; LB by Zheng: Lower bound by Zheng et al. [42]; STW: Bilge and Ulusoy [2]; UGA: Ulusoy, SIlvrikaya-Serifoglu and Bilge [3]; AGA: Abdelmaguid et al. [4]; RGA: Montane and Galvao [9]; ZGA: Zheng et al. [45].

TABLE 2. Results comparison when t/p < 0.25.

Prob.no	LB by Ulusoy	LB by Zheng	STW	UGA	AGA	RGA	ZGA	PGA	PGA-B
EX110	126	126	126	126	126	126	126	126	126
EX210	148	148	148	148	148	148	148	148	148
EX310	138	138	150	**148***	150	150	150	150	150
EX410	112	112	121	119	119	119	119	119	119
EX510	102	102	102	102	102	102	102	102	102
EX610	163	163	186	186	186	186	186	196	196
EX710	137	137	137	137	137	137	137	137	137
EX810	271	271	292	**271***	292	292	292	292	292
EX910	150	150	176	176	176	176	176	176	176
EX1010	218	218	238	**236***	238	238	238	242	242
EX120	123	123	123	123	123	123	123	123	123
EX220	143	143	143	143	143	143	143	143	143
EX320	135	135	148	145	145	145	145	145	145
EX420	111	111	116	114	114	114	114	116	116
EX520	99	99	100	100	100	100	100	100	**99***
EX620	160	160	183	181	181	181	181	187	187
EX720	136	136	136	136	136	136	136	136	136
EX820	268	268	287	**268***	287	287	287	287	287
EX920	150	150	174	173	173	173	173	179	179
EX1020	216	**213**	236	238	236	236	236	236	236
EX130	122	122	122	122	122	122	122	122	122
EX230	146	146	146	146	146	146	146	146	146
EX330	136	136	149	146	146	146	146	146	**145***
EX430	110	110	116	114	114	114	114	114	114
EX530	98	98	99	99	99	99	99	99	**98***
EX630	161	161	184	182	182	182	182	182	182
EX730	137	137	137	137	137	137	137	137	**134***
EX830	269	269	288	**270***	288	288	288	288	288
EX930	151	151	176	174	174	174	174	177	**173***
EX1030	217	**214**	237	241	237	237	237	237	237
EX140	124	124	124	124	124	124	124	124	124
EX241	217	217	217	217	217	217	217	217	**214***
EX340	138	138	151	151	151	151	151	151	151
EX341	203	203	222	221	221	221	221	221	**220***
EX441	166	166	179	172	172	172	172	172	**168***
EX541	148	148	154	148	148	148	148	148	148
EX640	161	161	185	184	184	184	184	192	192
EX740	137	137	138	137	137	137	137	137	137
EX741	203	203	203	203	203	203	203	203	203
EX840	272	272	293	273*	293	293	293	292	292
EX940	149	149	177	175	175	175	175	175	**174***
EX1040	219	**216**	240	244	240	240	240	240	240

Notes: LB by Ulusoy: Lower bound by Ulusoy, Sivrikaya-Serifoglu, and Bilge [3]; LB by Zheng; Lower bound by Zheng et al. [42]; STW: Bilge and Ulusoy [2]; UGA: Ulusoy, Sivrikaya-Serifoglu, and Bilge [3]; AGA: Abdelmaguid et al. [4]; RGA: Montane and Galvao [9]; ZGA: Zheng et al. [45]; PGA: proposed algorithm; PGA-B: bidirectional layout solution for the proposed algorithm.

problems and the reference literature. The second example considers the number of AGVs and the CFRP in FMS and analyzes their influences on the makespan.

A. ILLUSTRATIVE EXAMPLE 1

The benchmark problems are tested according to the four layouts and 10 job sets proposed by Ulusoy and Bilge [1]. We use two vehicles in the 82 test problems and compare the results obtained by the proposed algorithm with those obtained in the referenced literature under the same constraints. The four layouts are constructed using single-direction routes and machine locations. Here ''PGA'' (the proposed genetic algorithm) denotes the results obtained using the original layout, while ''PGA-B'' (proposed genetic algorithm with bidirectional layout) denotes the results obtained using the layouts with bidirectional route. We revised a value in the fourth layout because of the irregular transport time (the travel time from M2 to M1 should be 12). The layouts and production times are as reported by [1] and listed in Appendices 1 and 2.

This paper compares the results of 40 test problems with $t/p > 0.25$ and 42 test problems with $t/p < 0.25$ in Tables 1 and 2. Generally, the obtained results agree well with the published literature and tend to approach the lower bound (LB) value given by Ulusoy *et al.* [3] and Zheng *et al.* [45].

The LB is the theoretical true optimal solution. In Table 1, when $t/p > 0.25$, two results obtained by PA-B (EX22 and EX93) and one result obtained by PGA (EX22) reach the LB provided by Ulusoy *et al.* [3] and Zheng *et al.* [45]. PGA-B obtained 25 results that are better than the results obtained by other strategies, while the remainder of the PGA and PGA-B results agree well with those obtained by the other algorithms or methods. In Table 2, when $t/p < 0.25$, 16 of the results obtained by PGA and 17 of the results obtained by PGA-B reach the LB provided by Ulusoy *et al.* [3] and Zheng *et al.* [45]. Moreover, two of the results obtained by PGA-B (EX730 and EX241) are smaller than the LB provided by Ulusoy *et al.* [3] and Zheng *et al.* [45]. The remainder of the results obtained by PGA and PGA-B agree well with those obtained by the other algorithms or methods.

Based on our results, the proposed approach solves the machine and AGV scheduling problems efficiently.

TABLE 3. Machine scheduling for operation time (unit/min).

Job	operation	M1	M2	M3	M4	M5	M6
Job 1	1	6	7	10	—	—	—
	2	—	—	—	5	7	—
	3	6	—	—	—	—	7
	4	—	—	5	—	7	—
	5	—	—	—	4	—	8
Job 2	1	8	—	5	—	6	—
	2	—	5	—	—	—	—
	3	—	—	—	6	—	9
	4	9	—	10	—	11	—
Job 3	1	5	—	6	—	—	—
	2	—	8	—	—	7	—
	3	10	—	—	—	—	9
	4	—	12	—	9	—	—
	5	—	—	7	—	—	—
Job 4	1	—	—	—	5	—	7
	2	—	8	—	—	—	—
	3	—	—	10	—	11	—
	4	13	—	—	—	—	9
	5	—	—	—	12	—	—
	6	—	13	—	—	—	—

Notes: A line ("—") denotes that the operation cannot be processed.

TABLE 4. Transportation time of the sample layout (unit/min).

Point	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	2	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	0	2	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	2	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	2	0	0	0	2	0	0	0	0	0	0	0	0	0
7	0	2	0	0	0	2	0	1	0	0	0	1	0	0	0	0	0	0	0	0
8	0	0	3	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	1	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	0
10	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
11	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	1	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	2	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	2	0	0	0	3	0	0	0	0	1	0	0
15	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2
16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	1	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	0	1
20	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0

B. ILLUSTRATIVE EXAMPLE 2

In an FMS environment, each operation can be performed by alternative machines, which lead to alternative routes and different transit times. Additionally, different machines require different operation times to perform the same operation; thus, different operation and machine combinations yield different results. We use the following dataset from a discrete manufacturing enterprise: there are six machines in the system; four jobs that require five, four, five, and six operations; and several identical AGVs that can serve any machine. This model differs from the previous scheduling model because the proposed model is a complete scheduling process that considers both incoming and outgoing warehouse materials. Table 3 lists the operation times at different machines and corresponds to Figure 1.

The adjacent matrix in Table 4 lists the transit times between any two points in the network.

The parameters of the computational experiment are set the same as those in the preceding test example. The range

TABLE 5. Results of the computational experiment.

	AGV number	Mini makespan	Average makespan	Empty trip time	Vehicle idle rate	Invalid waiting time
Job	1	129	135	48	0.78%	1
	2	92.5	96.5	57.5	18.11%	33.5
	3	94	91	48.5	38.45%	101.5
	4	84	84.5	65	48.48%	159
	5	82	84	113	45.85%	188
	6	82	83	45.5	78.15%	348.5

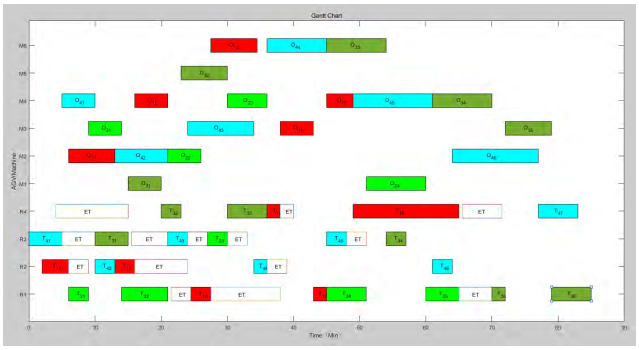


FIGURE 4. Gantt chart for the scheduling scheme.

of feasible solutions is relatively large compared with other benchmark problems; therefore, we set the initial population to 80, and each experiment was performed 10 times. We varied the number of vehicles to determine the optimal number for job-shop scheduling. Table 5 presents the experimental results.

When we employed one vehicle in the network, the average makespan was 135 min, and the minimum makespan was 129 min. When we employed two vehicles, the average makespan decreased to 92.5 min, and the minimum makespan decreased to 92.5 min. When we employed three vehicles, the average makespan decreased to 91 min, and the minimum makespan decreased to 88 min. When we employed four vehicles, the average makespan decreased to 84.5 min, and the minimum makespan decreased to 82 min. The minimum makespan continued to decrease as we employed more vehicles; however, the vehicle idle rate and invalid waiting time trend upward. This result occurs because increasing the number of vehicles increases the number of collisions on roads and vehicle waiting time. The vehicles are not beneficial during invalid waiting times, but they occupy transportation resources, which is an indirect cost.

Ultimately, we employed four vehicles in the calculation experiment to approximate the optimal scheduling scheme, as shown in Figure 4, where T_{16}, T_{25}, T_{36}, and T_{47} denote the operations in which jobs are delivered to the finished parts warehouse, and ET denotes an empty trip.

Figure 5 shows the search process for the optimal and average fitness of the proposed algorithm. To ensure that the optimal result is obtained, we set the algorithm to perform 100 iterations and executed 10 different runs to determine an

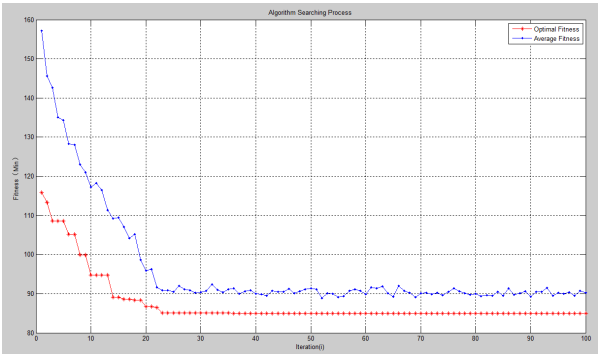


FIGURE 5. Search process of the proposed hybrid genetic algorithm.

optimal result and an average result. The makespan clearly decreases and converges quickly, as shown in Figure 5. The convergence tends to become stable after the 23rd iteration during the search process, which indicates that the algorithm finds the current optimal solution quickly and efficiently. Therefore, the algorithm is advantageous for solving the integrated scheduling problem.

Based on the experimental results, our main conclusion is that the proposed hybrid algorithm is an efficient method for solving the machine and AGV scheduling problems when considering the CFRP and transportation time.

C. ANALYSIS OF THE NUMBER OF AGVs

Montane and Galvao [9] proposed a tabu search algorithm to solve this problem and found that it obtained better makespan results and used fewer AGVs. However, the paper did not reveal the relationship between layout and the number of machines; thus, the paper simply improved the algorithm and did not analyze the effect of the number of AGVs. Corréa *et al.* [38] studied the effect of three elements on the values of objective function: the number of AGV, the number of jobs and the layout of the network, but did not consider the influence of the number of AGVs. Therefore, its results, which used a fixed number of AGVs, may not be optimal. Khayat *et al.* [39] also used a fixed number of AGVs throughout their computational experiments. In this paper, to study the effect of the number of AGVs on the objective function, we conducted a set of test experiments with diverse layouts and varying job numbers to verify the effectiveness of the optimal number of AGVs. The details of the test layouts and job sets are provided in Appendices A and B, respectively.

As shown in Table 6, we selected 14 test problems. Each test problem involved 3–5 experiments using different numbers of vehicles. We executed each test experiment 5–10 times to obtain minimum and average makespan values. The optimal number of AGVs was determined according to the decrease rate of the minimum makespan. We set 5% as the minimum makespan decrease rate to determine the optimal number of AGVs. As shown in (15), C_{iv} denotes the completion time of J_i with v vehicles:

$$\frac{C_{iv} - C_{i(v+1)}}{C_{iv}} \leq 5\%. \tag{15}$$

TABLE 6. Results of the computational experiments.

TP NO.	J/M/L	v	$C_{iv}/meanC_{iv}$	IWT	URV	ONV	CT
1	3/3/1	1	44/48.8	0.2	99.59%	2	30.164
		2	41/41.5	15.9	80.84%		
		3	41/41	24	80.49%		
2	4/4/2	1	75/76.8	0	100%	3	112.55
		2	51/52.2	19.4	91.42%		
		3	47/50	55.86	62.76%		
		4	47/47.9	78.4	59.02%		
3	5/4/2	1	89/91	1.2	98.7%	3	130.61
		2	52/56	11.3	89.91%		
		3	47/49	34.5	76.53%		
		4	47/47.6	64.2	65.85%		
4	5/5/3	1	99/99.25	0.5	99.5%	3	143.78
		2	61/63.25	22.63	82.11%		
		3	54/55.42	45.8	72.45%		
		4	54/55	78.9	64.13%		
5	6/5/3	1	119/121.25	0.5	99.59%	4	177.22
		2	68/70.875	15.5	89.07%		
		3	57/60	33.3	81.5%		
		4	52/54.4	58.3	73.2%		
		5	51/53.8	89.1	66.88%		
6	6/6/4	1	131/134.4	1	99.26%	3	228.03
		2	82/85.3	30.7	82%		
		3	75/76.3	65.58	71.35%		
		4	75/76.6	121.7	60.28%		
7	7/6/4	1	150/153.8	1.6	99.9%	4	252.62
		2	90.5/93.8	18.5	90.14%		
		3	77/78.9	62.4	73.64%		
		4	72/74	86.25	70.80%		
		5	72/73.6	149	59.51%		
8	7/7/5	2	107/116	25.1	89.18%	4	317.46
		3	92/96.3	64.5	77.67%		
		4	84/87.17	107	69.31%		
		5	83/87	189.3	56.48%		
9	8/6/4	2	116.5/120.9	16.5	93.18%	4	347.30
		3	100/103	65.4	78.84%		
		4	95/96.67	97.75	71.75%		
		5	94/94	156.5	66.7%		
10	8/7/5	3	75/78.2	54.3	76.85%	5	287.89
		4	70/73.2	67.6	76.91%		
		5	67/71	116.8	67.1%		
		6	67/69.4	183.13	56.02%		
11	8/8/6	2	107/109.83	24.5	88.85%	4	315.11
		3	95/96	65.17	77.37%		
12	9/7/5	3	80/83.75	30.125	88.01%	6	332.54
		4	71.5/76.75	60.875	80.17%		
		5	67.5/70.3	82.5	76.53%		
		6	64/67.6	140.125	65.45%		
		7	63/66.6	192	58.81%		
13	9/8/6	3	105/109.67	76	76.9%	5	380.15
		4	100.5/105.5	118.9	71.82%		
		5	97/100.2	169.1	66.25%		
		6	96/98.92	226.5	61.84%		
		7	96/97	310.5	54.27%		
14	10/8/6	4	118.5/123.8	120.3	75.71%	6	480.59
		5	111.5/115.6	160.1	72.3%		
		6	102/106.8	234.67	63.38%		
		7	102/105.4	259.5	57.83%		

Notes: TPNO.-test problem number; J/M/L-job/machine/layout; v -automated guided vehicle number; IWT-invalid waiting time; URV-utilization rate of the vehicles in the layout; ONV-optimal number of vehicles; CT-CPU time.

The makespan decreases and tends to stabilize as the number of AGVs increases. When (15) is satisfied, we take C_{iv} as the minimum makespan (which may not be the optimal scheduling solution) and the corresponding number of AGVs v as the optimal value. The invalid waiting time increases sharply due to the addition of more vehicles, which result in more collisions but fewer transfer tasks assigned to each vehicle. Therefore, the utilization rate of vehicles decreases as the number of AGVs increases. The vehicle utilization rate is calculated by (16):

$$URV = \frac{\sum_{i}^{n}\sum_{j}^{u_i}\left(C_{i(j+1)k} - C'T_{ijv}\right)}{v \cdot C_{\max}} \tag{16}$$

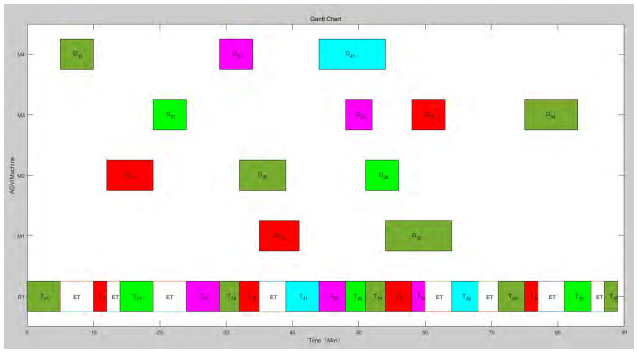


FIGURE 6. Gantt chart of machine and AGV scheduling with 1 vehicle.

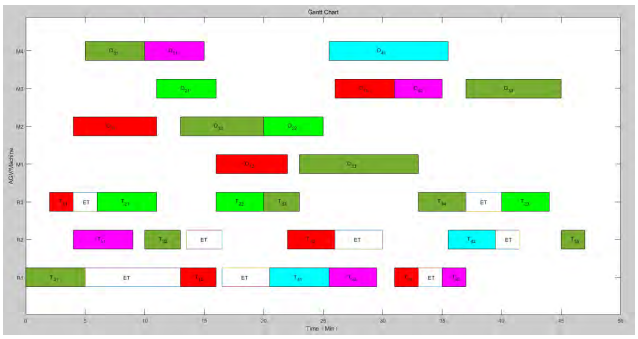


FIGURE 7. Gantt chart of machine and AGV scheduling with 3 vehicles.

To illustrate the influence of the number of AGVs on the makespan and on invalid waiting time, Gantt charts of the No. 3 test problem are given in Figures 6-8:

Vehicles usually have two trip states: empty and loaded. There are two types of gaps in the Gantt charts: gaps between empty trips and loaded trips and gaps between loaded trips. The first type of gap indicates that an empty trip vehicle arrives at its assigned machine before the machine's operation is completed; the second type of gap indicates that the destination of the previous transit task and the start point of the next transit task are located at the same machine and that vehicles arrived at the assigned machines before the machine operations were completed. In this paper, both these gap types are considered "invalid waiting time," accrued by vehicles while performing transport tasks. As shown in Figure 7, when the completion time of an empty trip occurs after the completion time of the previous operation, that time difference constitutes another invalid waiting time.

In Figure 6, only one vehicle is employed and no invalid waiting time is found in the production system because the single vehicle is responsible for all the operations. The transit tasks are transported individually, and no invalid waiting time occurs. However, the vehicle's workload is heavy, and the makespan value is large. If the vehicle happens to be out of service, the entire production system might have to stop. In Figure 7, the makespan decreases when three vehicles are employed in the production system. The transport efficiency increases as more vehicles are assigned, and the

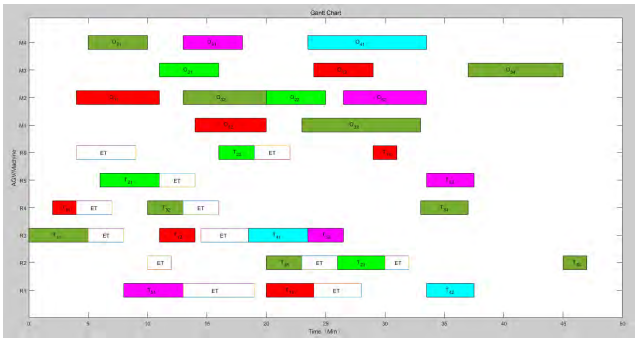


FIGURE 8. Gantt chart of machine and AGV scheduling with 6 vehicles.

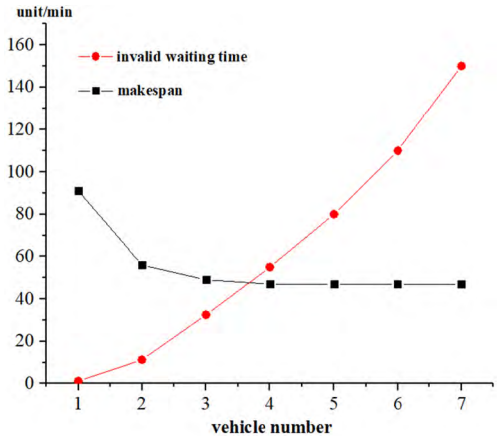


FIGURE 9. The influence of the number of AGVs.

feasible operations can be transported to assigned machines in a timely manner. However, when too many vehicles are employed in the production system, the operations assigned to each vehicle decrease and the vehicle utilization rate also decreases.

As illustrated in Figure 7 and Figure 8, the makespan value stabilizes as the number of AGVs increases past three.

The relationship between the makespan, number of AGVs and invalid waiting time is depicted in Figure 9 using an example of test problem No. 3.

V. CONCLUSION

In this paper, an approach for machine/AGV scheduling problems that considers both the number of AGVs and the CFRP is proposed. First, we construct a mathematical model to define the constraints of integrated scheduling. The model simultaneously considers machine scheduling, AGV scheduling problems and the CFRP. The objective is to minimize the makespan. Second, the network graph is constructed to illustrate the production system. Third, a genetic algorithm combined with the Dijkstra algorithm based on the time window is proposed to solve the problems. The tri-string chromosome coding method is proposed for the genetic algorithm to combine the three constituent elements: job, machine and AGV together and finally realize the integrated scheduling. The Dijkstra algorithm based on the time window is used to find

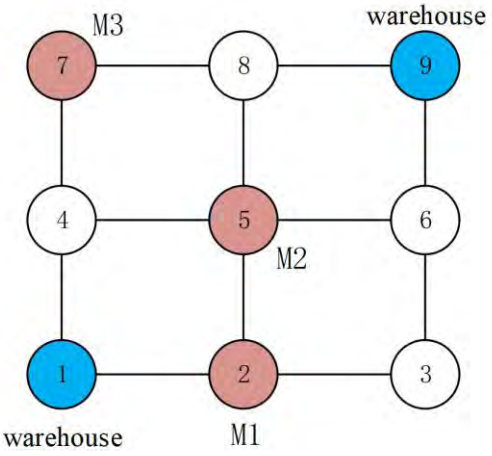


FIGURE 10. Layout 1.

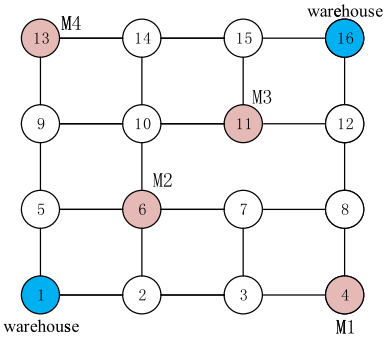


FIGURE 11. Layout 2.

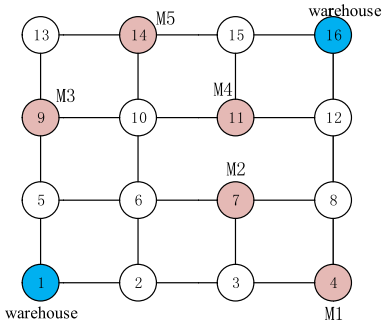


FIGURE 12. Layout 3.

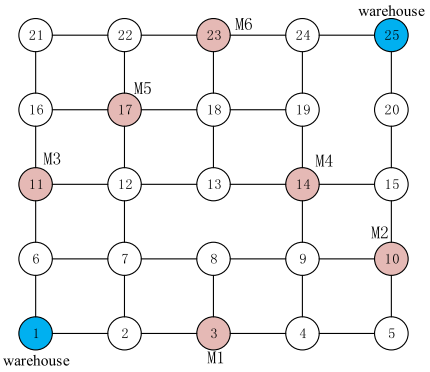


FIGURE 13. Layout 4.

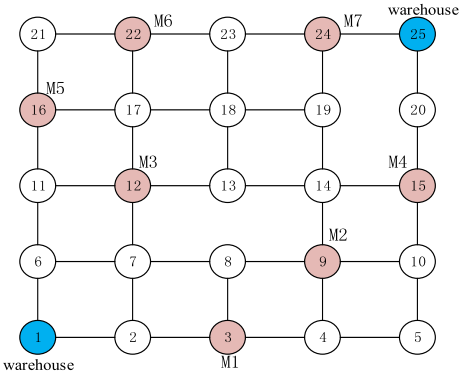


FIGURE 14. Layout 5.

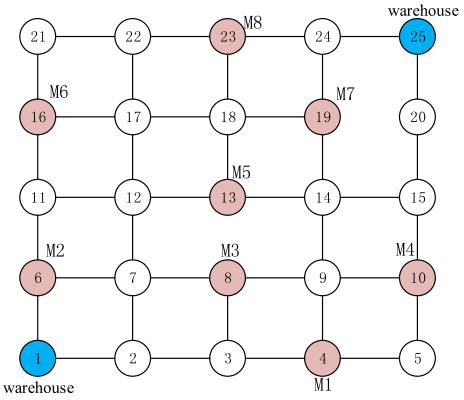


FIGURE 15. Layout 6.

the shortest route and detect collision simultaneously. Finally, two sets of computational experiments were performed to verify the efficiency of the proposed approach. The results of the first set of experiments indicated that the proposed coding strategy for the genetic algorithm solves the traditional problems efficiently compared with the best results in the existing literature. The second set of experiments involved the CFRP and the impact of number of AGVs. The paper set 5% as the makespan minimum decrease rate to determine the optimal number of AGVs. The results of this experiment highlighted the importance of selecting an appropriate number of AGVs. Using a fixed number of AGV in the production system may waste transportation resources and cost the enterprise unnecessarily or result in inadequate transportation resources to obtain the minimum makespan and satisfy the required delivery time.

Future research will involve more constraints, such as job sequencing and dynamic scheduling problems.

APPENDIX A

Layouts of the computational experiments See Figures 10–15.

APPENDIX B

See Table 7.

TABLE 7. Data for job sets used in the second set of computational experiments.

Test prob No.	Job No.	Sequence and operation time	Related layout Test AGV number
1	1	M2(7)/M3(10)-M1(6)-M3(5)	Layout 1 1/2/3
	2	M1(8)/M3(5)-M2(10)	
	3	M1(5)/M3(6)-M2(7)-M1(10)-M3(8)	
2	1	M3(4)-M2(4)/M4(5)-M1(6)/M3(8)-M2(5)	Layout 2 1/2/3/4
	2	M1(9)/M3(10)	
	3	M2(7)-M1(10)-M2(9)/M4(12)-M3(8)	
	4	M4(8)-M1(12)/M3(10)	
3	1	M2(7)/M3(10)-M1(6)-M3(5)	Layout 2 1/2/3/4
	2	M1(8)/M3(5)-M2(5)	
	3	M4(5)/M3(6)-M2(7)-M1(10)-M3(8)	
	4	M4(10)	
	5	M4(5)-M2(7)/M3(4)	
4	1	M3(10)-M1(6)/M2(5)	Layout 3 1/2/3/4
	2	M5(7) -M2(5)-M3(9)/M4(10)	
	3	M4(5)/M3(6)	
	4	M3(8)-M1(6)-M5(5)/M2(7)-M3(6)-M4(8)	
	5	M2(7)-M5(10)/M3(13)-M1(6)	
5	1	M5(13)/M3(11)-M1(4)-M3(7)	Layout 3 1/2/3/4/5
	2	M1(6)-M5(5)-M4(12)/M3(10)	
	3	M4(5)/M5(6)-M1(13)	
	4	M2(9)-M4(5)-M1(4)/M3(7)-M2(5)-M5(7)	
	5	M3(5)-M4(9)/M2(11)	
	6	M2(7)/M5(6)-M3(10)/M1(12)	
6	1	M1(10)/M6(7)	Layout 4 1/2/3/4
	2	M2(5)/M3(7)-M4(5)-M1(9)/M2(10)	
	3	M5(7)/M4(6)-M3(9)-M1(12)-M4(8)	
	4	M3(6)-M2(8)/M5(9)	
	5	M1(10)/M5(8)/M6(12)-M2(7)	
	6	M2(8)-M1(7)-M5(5)/M3(4)-M1(13)-M4(10)-M6(9)	
7	1	M3(10)-M1(6)-M4(5)/M6(7)	Layout 4 1/2/3/4/5
	2	M6(8)-M4(5)-M1(9)/M3(12)	
	3	M1(5)/M3(6)/M4(8)	
	4	M2(10)/M3(12)-M5(14)	
	5	M2(8)-M6(10)/M4(11)-M1(9)	
	6	M4(8)-M2(10)-M1(11)/M6(13)-M5(7)-M3(14)	
	7	M3(5)-M4(9)	
8	1	M2(12)/M6(10)-M1(14)-M3(5)	Layout 4 2/3/4/5
	2	M1(8)-M2(5)-M1(9)/M4(10)	
	3	M1(5)/M3(6)/M2(7)-M4(10)-M3(8)/M5(11)	
	4	M4(13)-M6(10)/M5(12)-M1(12)	
	5	M3(15)/M4(17)-M5(4)	
	6	M6(9)/M4(12)-M3(5)	
	7	M5(15)-M6(5)-M2(7)/M1(10)-M3(12)-M4(9)-M2(6)-M6(13)/M1(9)	
	8	M1(14)/M5(12)/M3(10)	
9	1	M1(14)/M3(10)-M4(6)	Layout 5 2/3/4/5
	2	M2(8)/M3(12)/M6(13)	
	3	M4(5)/M3(6)-M7(11)-M1(10)-M3(8)	
	4	M7(12)/M4(15)-M5(9)-M6(4)	
	5	M5(13)-M6(6)-M4(8)/M1(9)-M5(7)/M7(9)	
	6	M6(15)-M7(5)-M1(8)	
	7	M5(10)-M3(4)/M2(6)-M4(9)-M6(14)-M7(15)/M3(12)-M5(10)	
10	1	M7(12)/M3(15)/M1(16)-M6(5)	Layout 5 3/4/5/6
	2	M4(8)/M3(6)-M7(11)	
	3	M5(5)/M3(6)-M2(7)-M1(10)/M3(12)	
	4	M3(13)-M4(5)-M2(8)-M7(10)-M6(12)/M3(10)	
	5	M2(9)-M4(10)/M1(7)	
	6	M5(10)-M7(12)/M3(15)-M4(4)	
	7	M6(7)-M2(10)-M4(5)/M7(8)	
	8	M2(12)/M5(10)-M4(3)	

TABLE 7. *(Continued.)* Data for job sets used in the second set of computational experiments.

11	1	M1(9)-M3(5)/M5(7)	Layout 5 2/3/4/5
	2	M2(12)/M3(15)-M4(7)/M7(9)-M6(4)/M1(10)	
	3	M4(7)/M1(10)/M3(8)-M2(14)/M7(16)	
	4	M5(8)/M7(11)/M1(12)-M4(10)	
	5	M2(12)/M3(14)	
	6	M1(5)/M7(6)-M4(8)-M6(12)	
	7	M3(15)/M5(17)-M7(4)/M4(6)	
	8	M6(4)-M1(15)-M3(6)/M2(8)-M7(4)	
	9	M7(13)-M3(10)/M4(7)-M1(5)-M5(6)	
12	1	M1(7)/M3(5)-M2(6)/M6(5)-M8(15)	Layout 6 3/4/5/6/7
	2	M4(8)/M3(5)/M2(9)-M1(9)/M6(10)	
	3	M2(5)/M3(6)/M5(8)	
	4	M3(7)-M4(10)/M8(14)	
	5	M6(10)-M2(9)-M3(6)-M5(12)-M7(7)/M3(5)	
	6	M8(13)-M1(4)/M4(6)/M7(5)-M3(10)/M5(8)	
	7	M5(8)/M6(7)-M1(17)/M3(15)	
	8	M2(9)/M5(11)-M1(8)-M8(12)-M5(12)/M7(14)-M4(16)-M6(5)/M7(7)	
13	1	M3(10)-M6(12)/M8(15)-M3(5)	Layout 6 3/4/5/6/7
	2	M6(8)/M3(5)-M2(5)-M8(15)/M1(13)/M5(12)	
	3	M1(15)/M3(16)/M2(17)-M7(10)-M5(8)	
	4	M2(4)-M4(10)/M5(13)	
	5	M4(14)/M7(17)/M8(19)	
	6	M5(4)-M2(8)/M1(9)-M3(12)/M4(15)-M6(9)	
	7	M3(10)-M1(8)/M4(12)-M8(16)-M7(14)/M2(15)	
	8	M2(14)/M7(11)-M3(8)/M4(10)	
	9	M8(12)-M6(8)-M3(17)-M1(4)-M2(8)/M5(10)-M4(11)-M7(7)/M3(9)	
14	1	M4(12)/M3(10)-M1(6)/M7(5)-M5(16)	Layout 6 4/5/6/7
	2	M8(14)/M3(12)-M2(18)-M1(9)/M4(10)	
	3	M7(5)/M3(6)-M2(7)-M1(10)-M3(8)	
	4	M1(12)/M6(15)	
	5	M2(6)/M3(8)-M8(12)-M6(9)	
	6	M5(4)/M7(6)-M4(12)	
	7	M3(8)-M8(12)-M7(5)-M1(9)/M2(11)	
	8	M3(16)-M8(11)-M1(5)/M4(6)/M6(7)-M5(12)-M2(8)/M7(10)-M4(13)	
	9	M6(7)-M2(10)/M4(13)/M8(14)	
	10	M1(11)-M8(7)-M3(17)/M7(14)-M6(5)-M2(9)-M6(10)/M5(8)	

ACKNOWLEDGEMENT

I would like to express my sincere gratitude towards my tutor—Yu-Chuan Song, who has provided me with very valuable suggestions and encouraged me all the time. I am truly thankful for his patience and honored to be his student.

REFERENCES

[1] G. Ulusoy and Ü. Bilge, "Simultaneous scheduling of machines and automated guided vehicles," *Int. J. Prod. Res.*, vol. 31, no. 12, pp. 2857–2873, 1993.

[2] Ü. Bilge and G. Ulusoy, "A time window approach to simultaneous scheduling of machines and material handling system in an FMS," *Oper. Res.*, vol. 43, no. 6, pp. 1058–1070, 1995.

[3] G. Ulusoy, F. Sivrikaya-Şerifoğlu, and Ü. Bilge, "A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles," *Comput. Oper. Res.*, vol. 24, no. 4, pp. 335–351, 1997.

[4] T. F. Abdelmaguid, A. O. Nassef, B. A. Kamal, and M. F. Hassan, "A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles," *Int. J. Prod. Res.*, vol. 42, no. 2, pp. 267–281, 2004.

[5] Y. Hou, N. Wu, M. Zhou, and Z. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 517–530, Mar. 2017.

[6] L. Deroussi, M. Gourgand, and N. Tchernev, "A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles," *Int. J. Prod. Res.*, vol. 46, no. 8, pp. 2143–2164. 2008.

[7] F. Yang, K. Gao, I. W. Simon, Y. Zhu, and R. Su, "Decomposition methods for manufacturing system scheduling: A survey," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 389–400, Mar. 2018.

[8] G. Vilcot and J.-C. Billaut, "A tabu search and a genetic algorithm for solving a bicriteria general job shop scheduling problem," *Eur. J. Oper. Res.*, vol. 190, no. 2, pp. 398–411, 2008.

[9] F. A. T. Montané and R. D. Galvão, "A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service," *Comput. Oper. Res.*, vol. 33, no. 3, pp. 595–619, 2006.

[10] H.-C. Chang, Y.-P. Chen, T.-K. Liu, and J.-H. Chou, "Solving the flexible job shop scheduling problem with makespan optimization by using a hybrid Taguchi-genetic algorithm," *IEEE Access*, vol. 3, pp. 1740–1754, 2015.

[11] C. W. Kim and J. M. A. Tanchoco, "Conflict-free shortest-time bidirectional AGV routeing," *Int. J. Prod. Res.*, vol. 29, no. 2, pp. 2377–2391, 1991.

[12] N. Wu and M. Zhou, "Shortest routing of bidirectional automated guided vehicles avoiding deadlock and blocking," *IEEE/ASME Trans. Mechatronics*, vol. 12, no. 1, pp. 63–72, Feb. 2007.

[13] M. Gen, R. Cheng, and Q. Wang, "Genetic algorithms for solving shortest path problems," in *Proc. IEEE Int. Conf. Evol. Comput.*, Apr. 1997, pp. 401–406.

[14] K. Xing, L. Han, M. C. Zhou, and F. Wang, "Deadlock-free genetic scheduling algorithm for automated manufacturing systems based on deadlock control policy," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 42, no. 3, pp. 603–615, Jun. 2012.

[15] S. A. Reveliotis, "Conflict resolution in AGV systems," *IIE Trans.*, vol. 32, no. 7, pp. 647–659, 2000.

[16] L. Qiu and W.-J. Hsu, "A bi-directional path layout for conflict-free routing of AGVs," *Int. J. Prod. Res.*, vol. 39, no. 10, pp. 2177–2195, 2001.

[17] J. Li, X. H. Meng, and X. Dai, "Collision-free scheduling of multi-bridge machining systems: A colored traveling salesman problem-based approach," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 139–147, Jan. 2018.

[18] N. N. Krishnamurthy, R. Batta, and M. H. Karwan, "Developing conflict-free routes for automated guided vehicles," *Oper. Res.*, vol. 41, no. 6, pp. 1077–1090, 1993.

[19] R. H. Möhring, E. Köhler, E. Gawrilow, and B. Stenzel, "Conflict-free real-time AGV routing," in *Proc. 3rd Int. Conf. Appl. Infrastruct. Res.*, Berlin, Germany, Oct. 2004, pp. 661–675.

[20] M. Gen, J. Gao, and L. Lin, "Multistage-based genetic algorithm for flexible job-shop scheduling problem," in *Intelligent and Evolutionary Systems*. Berlin, Germany: Springer, 2009, pp. 183–196.

[21] T. Mareda, L. Gaudard, and F. Romerio, "A parametric genetic algorithm approach to assess complementary options of large scale windsolar coupling," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 2, pp. 260–272, Apr. 2017.

[22] G. Desaulniers, A. Langevin, D. Riopel, and B. Villeneuve, "Dispatching and conflict-free routing of automated guided vehicles: An exact approach," *Int. J. Flexible Manuf. Syst.*, vol. 15, no. 4, pp. 309–331, 2003.

[23] T. Miyamoto, N. Tsujimoto, and S. Kumagai, "A cooperative algorithm for autonomous distributed vehicle systems with finite buffer capacity," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E88-A, no. 11, pp. 3036–3044, Nov. 2005.

[24] T. Miyamoto and K. Inoue, "Local and random searches for dispatch and conflict-free routing problem of capacitated AGV systems," *Comput. Ind. Eng.*, vol. 91, pp. 1–9, Jan. 2016.

[25] T. Nishi and Y. Tanaka, "Petri net decomposition approach for dispatching and conflict-free routing of bidirectional automated guided vehicle systems," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 42, no. 5, pp. 1230–1243, Sep. 2012.

[26] J. M. Novas and G. P. Henning, "Integrated scheduling of resource-constrained flexible manufacturing systems using constraint programming," *Expert Syst. Appl.*, vol. 41, no. 5, pp. 2286–2299, 2014.

[27] H. Fazlollahtabar and S. Hassanli, "Hybrid cost and time path planning for multiple autonomous guided vehicles," *Appl. Intell.*, vol. 48, no. 2, pp. 482–498, 2017.

[28] Y. Kaboudani, S. H. Ghodsypour, H. Kia, and A. Shahmardan, "Vehicle routing and scheduling in cross docks with forward and reverse logistics," *Oper. Res.*, pp. 1–34, Mar. 2018.

[29] X. Meng, J. Li, M. Zhou, and X. Dai, "Improved population-based incremental learning algorithm for scheduling multi-bridge waterjet cutting processes," in *Proc. 11th IEEE Int. Conf. Netw., Sens. Control (ICNSC)*, Miami, FL, USA, Apr. 2014, pp. 496–500.

[30] H. Fazlollahtabar, M. Saidi-Mehrabad, and E. Masehian, "Mathematical model for deadlock resolution in multiple AGV scheduling and routing network: A case study," *Ind. Robot, Int. J.*, vol. 42, no. 3, pp. 252–263, 2015.

[31] S. Maza and P. Castagna, "A performance-based structural policy for conflict-free routing of bi-directional automated guided vehicles," *Comput. Ind.*, vol. 56, no. 7, pp. 719–733, 2005.

[32] B. Mahadevan and T. T. Narendran, "Estimation of number of AGVs for an FMS: An analytical model," *Int. J. Prod. Res.*, vol. 31, no. 7, pp. 1655–1670, 1993.

[33] R. G. Kasilingam and S. L. Gobal, "Vehicle requirements model for automated guided vehicle systems," *Int. J. Adv. Manuf. Technol.*, vol. 12, no. 4, pp. 276–279, 1996.

[34] I. F. A. Vis, R. de Koster, K. J. Roodbergen, and L. W. P. Peeters, "Determination of the number of automated guided vehicles required at a semi-automated container terminal," *J. Oper. Res. Soc.*, vol. 52, no. 4, pp. 409–417, 2001.

[35] Z. Li, M. Zhou, and N. Wu, "A survey and comparison of Petri net-based deadlock prevention policies for flexible manufacturing systems," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 173–188, Mar. 2008.

[36] K. Vivaldini, L. F. Rocha, N. J. Martarelli, M. Becker, and A. P. Moreira, "Integrated tasks assignment and routing for the estimation of the optimal number of AGVS," *Int. J. Adv. Manuf. Technol.*, vol. 82, nos. 1–4, pp. 719–736, 2016.

[37] M. Mousavi, H. J. Yap, S. N. Musa, F. Tahriri, and S. Z. M. Dawal, "Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization," *PLoS ONE*, vol. 12, no. 3, 2017, Art. no. e0169817.

[38] A. I. Corréa, A. Langevin, and L.-M. Rousseau, "Scheduling and routing of automated guided vehicles: A hybrid approach," *Comput. Oper. Res.*, vol. 34, no. 6, pp. 1688–1707, 2007.

[39] G. El Khayat, A. Langevin, and D. Riopel, "Integrated production and material handling scheduling using mathematical programming and constraint programming," *Eur. J. Oper. Res.*, vol. 175, no. 3, pp. 1818–1832, 2006.

[40] T. Nishi, Y. Hiranaka, and I. E. Grossmann, "A bilevel decomposition algorithm for simultaneous production scheduling and conflict-free routing for automated guided vehicles," *Comput. Oper. Res.*, vol. 38, no. 5, pp. 876–888, 2011.

[41] M. Saidi-Mehrabad, S. Dehnavi-Arani, F. Evazabadian, and V. Mahmoodian, "An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs," *Comput. Ind. Eng.*, vol. 86, pp. 2–13, Aug. 2015.

[42] U. A. Umar, M. K. A. Ariffin, N. Ismail, and S. H. Tang, "Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (AGV) in flexible manufacturing systems (FMS) environment," *Int. J. Adv. Manuf. Technol.*, vol. 81, nos. 9–12, pp. 2123–2141, 2015.

[43] L.-L. Fu, M. A. Aloulou, and C. Triki, "Integrated production scheduling and vehicle routing problem with job splitting and delivery time windows," *Int. J. Prod. Res.*, vol. 55, no. 20, pp. 5942–5957, 2017.

[44] C. Y. Zhang, X. Dong, X. J. Wang, X. Y. Li, and Q. Liu, "Improved NSGA-II for the multi-objective flexible job-shop scheduling problem," *Chin. J. Mech. Eng.*, vol. 46, no. 11, pp. 156–164, 2010.

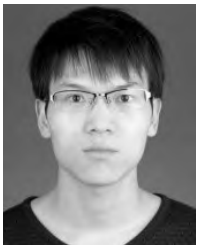
[45] Y. Zheng, Y. Xiao, and Y. Seo, "A tabu search algorithm for simultaneous machine/AGV scheduling problem," *Int. J. Prod. Res.*, vol. 52, no. 19, pp. 5748–5763, 2014.

XIANGFEI LYU received the M.S. degree from the College of Mechanical Engineering, Chongqing University, Chongqing, China, in 2012, where he is currently pursuing the Ph.D. degree in mechanical engineering. His research interests include manufacturing information and networked manufacturing.

YUCHUAN SONG received the B.S., M.S. and Ph.D. degree from the College of Mechanical Engineering, Chongqing University, in 1995, 1998, and 2001, respectively, where he is currently a Professor of Mechanical Engineering. He has led several research projects supported by the Natural National Science Foundation of China, the National High Technology Research and Development Program of China (863 Program), and the Chongqing Science and Technology Commission of China. His work has resulted in over 100 journals and conference proceeding articles. His research interests include green manufacturing, advanced manufacturing technology, intelligent manufacturing, and production and operation management.

CHANGZHENG HE received the B.S. degree from the Henan University of Science and Technology, Henan, China, in 2015, and the M.S. degree from the College of Mechanical Engineering, Chongqing University, in 2018.

QI LEI received the B.S., M.S., and Ph.D. degree from the College of Mechanical Engineering, Chongqing University, in 1999, 2002, and 2006, respectively, where she is currently an Associate Professor of mechanical engineering. She has led several research projects supported by the Natural National Science Foundation of China, the National Science and Technology Pillar Program during the 12-th Five-Year Plan Period of China, and the National High Technology Research and Development Program of China (863 Program). Her work has resulted in over 20 journals and conference proceeding articles. Her research interests include green manufacturing, advanced manufacturing technology, production planning and control, and manufacturing systems engineering.



WEIFEI GUO received the B.S. degree from the School of Energy Science and Engineering, Henan Polytechnic University, Henan, China, in 2010, and the M.S. degree from the College of Mechanical Engineering, Chongqing University, in 2017, where he is currently pursuing the Ph.D. degree in mechanical engineering.