Research Article

A CEGA-Based Optimization Approach for Integrated Designing of a Unidirectional Guide-Path Network and Scheduling of AGVs

Haining Xiao 1, Xing Wu 2, Yong Zeng 1, and Jingjing Zhai 2

1College of Mechanical Engineering, Yancheng Institute of Technology, Yancheng 224051, China
2College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

Correspondence should be addressed to Xing Wu; wustar5353@nuaa.edu.cn

Received 30 October 2019; Revised 31 December 2019; Accepted 7 January 2020; Published 7 February 2020

Copyright © 2020 Haining Xiao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the current industrial fields, automatic guided vehicles (AGVs) are widely employed to constitute the flexible manufacturing system (FMS), owing to their great advantages of routing flexibility and high efficiency. However, one main challenge lies in the coupling process of the design problem of the unidirectional guide-path network (UGN) and the task scheduling problem of AGVs. To reduce the complexity, most pertinent literatures only handle these problems one by one, based on the stepwise design methods, thereby neglecting the constraint conditions and the optimization objectives caused by the FMS environment. The motivation of the paper is to bring the coupling factors into the integrated design and solution process. Firstly, an integrated design model of designing UGN and scheduling AGVs simultaneously is proposed, with the objective of minimizing the makespan (i.e., the maximum completion time of all handling tasks), in the consideration of the practical constraints, e.g., the job handling and processing sequence constraints and the AGV number constraint. Secondly, a dual-population collaborative evolutionary genetic algorithm (CEGA) is developed to solve the problems of designing and scheduling in a parallel way. The solutions of the integrated model, i.e., the potential strongly connected UGN and the feasible processing and handling sequence, are, respectively, coded as two different subpopulations with independent and concurrent evolution processes. The neighbourhood search operation, the niche technique, and the elitism strategy are combined to improve the convergence speed and maintain the population diversity. The experimental results show that the integrated design model can formulate the problem more accurately, and the CEGA algorithm is computationally efficient with high solution quality.

1. Introduction

As a flexible and high-efficiency logistics system, an automatic-guided vehicle system (AGVS) has broad application prospects in fields such as port transportation, warehousing and distribution, and manufacturing [1]. The unidirectional guide-path network (UGN) design and the AGVS task scheduling [2–4] are the two main issues that should be addressed when the AGVS is adopted. Many design models and algorithms for these two problems have been proposed in the literature. For example, aiming at the UGN design problem, Kaspi et al. [5] proposed a branch and bound method to minimize the travel distance of loaded vehicles, which can obtain the optimal solution more stably. Since the time complexity of the algorithm increases exponentially with the scale of the network, the method is only suitable for small-scale design problems. To decrease the design time, some heuristic algorithms have been developed. For example, Ko and Egbelu [6] proposed a multistage design method that determines the direction of the path sequentially, according to the traffic flow, with the objective of minimizing the travel distance of loaded vehicles. However, this algorithm does not take into account the empty travel distance and the result may substantially differ from the actual situation. Seo et al. [7] solved the UGN design problem by using a tabu search (TS) algorithm but without
the consideration of the empty travel distance. Besides, the algorithm has the comparatively strong dependence on the initial status, thereby being prone to fall into the local optimum. Guan et al. [8] proposed an improved electromagnetism-like mechanism algorithm to minimize both loaded and empty travel distance. Functionally similar approaches also include the genetic algorithm [9, 10] and the meta-heuristics algorithm [11].

In recent years, many mathematical models and algorithms have been developed for the AGVs task scheduling problem in the manufacturing environment. For example, aiming at the logistics dynamic scheduling problem in an intelligent manufacturing workshop (IMW), Xu et al. [12] proposed an intelligent logistics scheduling model and response method with Automated-Guided Vehicles (AGVs) based on the mode of "request-scheduling-response." Umar et al. [13] investigated the simultaneous scheduling of machines and AGVs in an FMS environment. A mixed-integer nonlinear programming model was formulated firstly, and a hybrid multi-objective genetic algorithm was then proposed to optimize the makespan, AGV travel time, and penalty cost. Kumar et al. [14] addressed simultaneous scheduling of both machines and material handling systems with alternative machines for the makespan minimizing objective. A machine selection heuristic and a vehicle assignment heuristic were incorporated in the differential evolution approach to assign tasks to appropriate machines and vehicles and to minimize cycle time. Similar research works can be found in Nouri et al. [15], Ahmadi-Javid and Hooshangi-Tabrizi [16], Lin et al. [17], Lacomme et al. [18], and Zhang and Li et al. [19]. Ahmadi-Javid and Hooshangi-Tabrizi [20] addressed a ternary integration scheduling problem, which incorporated employee timetabling into the schedules of machines and transportation systems in a job shop environment. A timed coloured Petri net (TCPN) approach with two performance objectives was devised in [21] for the simultaneous scheduling problem of machines and AGVs. Moreover, two heuristics (A* and ALS [22]) were developed to partially expose the graph. Lyu et al. [23] focused 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, they proposed a genetic algorithm combined with the Dijkstra algorithm that is based on a time window. Li et al. [24] proposed a mathematical model and an improved harmony search algorithm for material transfer in a real-world manufacturing system.

In fact, the design process of the UGN is potentially correlated with the task scheduling problem of AGVs in an FMS environment [25]. The mutual coupling relationship is mainly reflected in the following two aspects. On the one hand, the UGN determines the job transportation time among different workstations [26], while the transportation time is an important factor that must be considered in solving the AGVs task scheduling problem [27–29]. On the other hand, the AGVs task scheduling scheme influences the traffic flow [30] between each two workstations, which provides one main reference for the UGN design process. In this sense, these two problems are highly interacted with each other.

However, the mutual coupling relationship is often neglected in the pertinent literature, in order to decrease the complexity of the problem. The stepwise design method, which separates the UGN design from the AGV task scheduling, is mostly employed in the current design process. Firstly, the loaded traffic flow matrix among all workstations is generated on the basis of the job technological route information. Secondly, an optimal UGN is designed by optimizing only one objective, i.e., the loaded and empty travel distance. Other objectives and constraints in the FMS environment, e.g., the job handling and processing sequence constraints introduced by the job technological route, are neglected in this design stage. Finally, the simultaneous scheduling of machines and AGVs is addressed by taking into account the practical constraints and other objectives in the industrial field, e.g., minimizing the makespan. It is obvious that the stepwise design method reduces the complexity of the problem significantly. However, the solution quality of UGN cannot be always guaranteed due to the model infidelity without the coupling relationship.

The motivation behind this article lies in developing an integrated design model and a parallel optimization approach of the UGN design and AGV task scheduling, by considering the practical constraints imposed by the FMS environments. Our main contributions are twofold. On the one hand, an integrated design model of the UGN design and AGV scheduling in an FMS environment is constructed with the makespan minimization objective. The practical constraints in the industrial field, e.g., the job handling and processing sequence constraints, and the AGV number constraint, are considered in the model. On the other hand, a dual-population collaborative evolutionary genetic algorithm (CEGA) is proposed to solve the problems of designing and scheduling in a parallel way. Two kinds of chromosome are used, respectively, for coding the strongly connected UGN and the feasible processing and handling sequence as two different subpopulations. They evolve independently and concurrently, only having an information exchange when an individual need to be evaluated. Moreover, the neighbourhood search operation, the niche technique, and the elite retention strategy are combined to improve the convergence speed and maintain the population diversity.

The rest of this paper is organised as follows. The problem statement is described in detail in Section 2 and a new integrated model of simultaneously designing a UGN and scheduling AGVs in an FMS is formulated. In Section 3, a CEGA approach with specific evolutionary operators is proposed to solve the integrated model. In Section 4, computational experiments are carried out and experimental results are analysed and compared. Finally, conclusions and recommendations for future research are given in Section 5.
2. Problem Formulation

Figure 1 shows a typical FMS layout and an undirected guided-path network of AGVS. In the figure, $M_0$ and $M_8$ represent the input workstation and output workstation, respectively. $M_1$ to $M_8$ are the processing workstations, and each workstation $M_m$ is composed of an input buffer, a processing machine, and an output buffer. The input workstation only has an output buffer, and the output workstation has only an input buffer. There is an undirected guided-path network $G = (V, E)$ in the middle, where $V$ is the node set, which is composed of the loading points, unloading points, and other path intersection points. The edge (path segment) set is $E$, and the length of the edge $e_{i,j}$ is $d_{i,j}$. There are $N_J$ kinds of jobs $J = \{J_j|j = 1, 2, ..., N_J\}$ to be processed in this workshop. As shown in Figure 2, each job $J_j$ is composed of $n_j$ processing tasks and $n_j + 1$ handling tasks. All handling tasks are performed by $N_G$ AGVs. The processing workstation and the time of each processing task are known. The questions needed to be settled are as follows: the direction of each guided-path section, the execution AGV of each handling task, and the execution sequence of all processing and handling tasks. First, some assumptions are given as follows:

1. A processing machine can only perform one processing task at a time. Once a processing task is started, it cannot be interrupted. Each processing machine processes the jobs in the input buffer based on the first-come-first-served rule.

2. When the FMS starts to run, each job is located at the output buffer of the input workstation and is waiting to be transported to the first processing workstation according to its technological route. The job will enter the input buffer when it arrives at the unloading point of the workstation. After the job completes a processing task by the processing machine, it will enter the output buffer of the corresponding workstation and wait to be transported to the next workstation. After completing all processing tasks, the job will leave the system via the output workstation.

3. When the FMS starts to run, all AGVs are located at the loading point of the input workstation. They will return to the loading point of the input workstation after completing all handling tasks.

4. The capacity of all buffers in the system is sufficiently large, so deadlock problems triggered by the insufficient buffer capacity are not considered.

5. An AGV can only carry one job at a time. The loading and unloading time are not taken into account.

6. An idle AGV will stay at the unloading points of its last handling task but does not affect the operation of other AGVs.

(7) Routing deadlocks caused by path conflicts between AGVs are not considered.

2.1. Symbol Definitions

$N_M$ is the number of processing workstations.

$\Omega_M = \{M_m|m \in N \text{ and } 0 \leq m \leq N_M + 1\}$, the set of workstations. $M_0$ to $M_{N_M}$ are the processing workstations. $M_0$ and $M_{N_M + 1}$ represent the input workstation and output workstation, respectively.

$N_J$ is the number of job types in the system.

$\Omega_I = \{J_j|j = 1, 2, ..., N_J\}$, the set of jobs.

$n_j$ is the total number of processing tasks for the job $J_j$.

$\Omega_O = \{O_{i,j}|i \in N \text{ and } 1 \leq i \leq n_j\}$, the set of all processing tasks for the job $J_j$.

$\Omega_G = \{AGV_k|k \in N \text{ and } 1 \leq k \leq N_G\}$, the set of AGVs, and $N_G$ represents the number of AGVs in the system.

$b_{i,j}$ is the direction of the path segment $e_{i,j}$. If the value is 1, the direction is from node $v_i$ to $v_j$; the reverse direction occurs, if the value is 0.

$N_p$ is the number of loading points.

$N_D$ is the number of unloading points.

$N_T$ is the number of handling tasks.

$X^T_{i,j,k}$ is the assignment relationship between the handling tasks $T_{i,j}$ and the vehicle AGV $k$. The value of 1 means task $T_{i,j}$ will be executed by AGV $k$; otherwise, the value is 0.

$[X^T_{i,j,k}]_{N_D \times N_G}$ is the assignment matrix between all the AGVs and the handling tasks.

$[D^{LP}]_{N_D \times N_D}$ is the matrix of the shortest distance among all loading and unloading points.

$d^{LP}_{u,v}$ is the element located at the $u$-th row and $v$-th column in $[D^{LP}]_{N_D \times N_D}$, representing the shortest distance from the loading point $P_u$ to the unloading point $P_v$.

$[D^{LP}]_{N_D \times N_D}$ is the matrix of the shortest logistic distance among all unloading and loading points.

$d^{LP}_{u,v}$ is the element located at the $u$-th row and $v$-th column in $[D^{LP}]_{N_D \times N_D}$, representing the shortest logistic distance from the unloading point $D_u$ to the loading point $P_v$.

$C_k$ is the completion time for the AGV $k$ as it returns to the loading point of the input workstation after finishing all handling tasks.

$S^T_{i,j,k,m}$ is the starting time of handling task $T_{i,j}$, which is the $m$-th handling task of vehicle AGV $k$.

$E^T_{i,j,k,a}$ is the completion time of handling task $T_{i,j}$, which is the $a$-th handling task of the vehicle AGV $k$.

$S^O_{i,j,m,n}$ is the starting time of processing task $O_{i,j}$, which is the $n$-th processing task of workstation $M_{i,j}$. 

2.4. Constraint Conditions

2.4.1. Decision Variable Constraints. The decision variable is a 0-1 integer, that is,

\[ \forall e_{i,j} \in E \text{ has } b_{i,j} \in \{0, 1\}. \]  \hspace{1cm} (2)

The direction of each path segment is unique:

\[ \forall e_{i,j} \in E \text{ has } b_{i,j} + b_{j,i} = 1. \]  \hspace{1cm} (3)

2.4.2. Auxiliary Decision Variable Constraints. The auxiliary decision variable is a 0-1 integer, that is,

\[ \forall X^T_{i,j,k} \text{ has } X^T_{i,j,k} \in \{0, 1\}. \]  \hspace{1cm} (4)

Each handling task can only be assigned to one AGV:

\[ \forall T_{i,j} \text{ has } \sum_{k=1}^{N_v} X^T_{i,j,k} = 1. \]  \hspace{1cm} (5)

2.4.3. The UGN Must Be Strongly Connected. Each node contains at least one outdegree (i.e., the number of edges starting from this node) and one indegree (i.e., the number of edges ending at this node), which is a necessary condition for a strongly connected UGN:

\[ \forall e_{i,j} \in E \text{ has } \sum_{j=1}^{N_v} b_{i,j} \geq 1 \text{ and } \sum_{j=1}^{N_v} b_{j,i} \geq 1. \]  \hspace{1cm} (6)

There is a route to link any pair of unloading point and loading point, so it can be written as follows:

\[ \forall d^E_{u,w} \in [D^{DP}]_{N_v \times N_r} \text{ has } d^E_{u,w} < \infty. \]  \hspace{1cm} (7)

There is a route to link any pair of loading point and unloading point, which can be expressed as follows:

\[ \forall d^L_{u,w} \in [D^{PD}]_{N_r \times N_v} \text{ has } d^L_{u,w} < \infty. \]  \hspace{1cm} (8)
2.4.4. Processing and Handling Sequence Constraints. Handling task $T_{i,j}$ can be executed only after the processing task $O_{i,j}$ is completed:

$$E_{i,j,w,n}^{O} \leq S_{i,j,w,n}^{T}$$ (9)

Handling task $T_{i,j}$ can be started only after the AGV $k$ has completed the previous handling task $T_{c,d}$ and arrived at the loading point $P_u$ of the task $T_{i,j}$:

$$E_{c,d,k,(m-1)}^{T} + \frac{d_{u,w}^{E}}{v} \leq S_{i,j,k,m}^{T}$$ (10)

Only after the job arrives at the current workstation, the current processing task $O_{i,j}$ can start, which can be expressed by

$$E_{i,j,w,n}^{O} \leq S_{i,j,w,n}^{T}$$ (11)

Each processing machine can only perform one processing task at a time. So, only after the previous processing task $O_{p,h}$ is finished, the next processing task $O_{i,j}$ can start, which can be expressed by

$$E_{p,h,u,w,(n-1)}^{O} \leq S_{i,j,w,n}^{O}$$ (12)

As said before, the UGN design problem is potentially correlated with the AGVs task scheduling problem. The mutual coupling relationship is mainly reflected in the following two aspects. On the one hand, it can be seen from constraints (10) and (11) that the shortest distances among all loading or unloading points ($d_{w,u}^{E}$ and $d_{u,w}^{E}$) are the important factors to determine the beginning time of all processing and handling tasks, while the shortest distance between all loading points depends on the decision variable of the traditional UGN design problem. On the other hand, only when the decision variables of the AGVs scheduling problem are known, the completion time $C_k$ can be determined by constraints (9) to (12), so as to evaluate the quality of the designed UGN.

As both the UGN design and the AGVs scheduling problem in the FMS environment are nondeterministic polynomial time- (NP-) hard problems [31, 32], the integration of these two problems will become a more complex NP-hard problem. As a high-performance multipopulation heuristic intelligent search algorithm, the collaborative evolutionary genetic algorithm (CEGA) [33] divides the system decision variables into several variable subgroups. Each variable subgroup can select appropriate encoding scheme to obtain a subpopulation. Moreover, appropriate genetic operators can be designed for each subpopulation according to its encoding characteristics. The CEGA is very suitable for solving the problem of multivariable combinatorial optimization [34, 35]. Based on the characteristics of the abovementioned mathematical model, a dual-population CEGA involving the collaborative evolution of the UGN subpopulation and the task sequence subpopulation is proposed.

3. Optimization Algorithm

Drawing upon the idea of collaborative evolution, two subpopulations are adopted to represent the UGN and the feasible task sequence, respectively. The UGN subpopulation is labelled as P1, in which each chromosome represents a strongly connected UGN. The task sequence subpopulation is labelled as P2, in which each chromosome corresponds to a feasible handling and processing task sequence. In the evolutionary process, the two subpopulations independently and simultaneously evolve. Information exchange between the subpopulations occurs only during the process of fitness calculation. The algorithm mainly includes chromosome encoding, fitness calculation, subpopulations initialization, and the genetic evolutionary operations. The detailed methods are listed as follows.

3.1. Chromosome Representation and Encoding

3.1.1. Encoding of the UGN. To reduce the complexity of the problem, the guide-path network is preprocessed before encoding. The node with only two path segments connected has special significance. Taking the path network shown in Figure 1, for example, edge (1, 2) should have the same direction as edge (2, 3), in order to make node 2 satisfy the constraint shown in formula (6). Therefore, in the design process, edges (24, 1), (1, 2), (2, 3), (3, 4), (4, 5), and (5, 6) can be merged into one combined edge (24, 6). According to this preprocessing principle, the guide-path network shown in Figure 1 has nine combined edges. All combined edges and serial numbers are as shown in Table 1. The UGN is encoded with a binary chromosome code $B = [b_{N_1} \ldots b_{N_b}]$. Each bit in the code corresponds to the direction of a combined edge. The chromosome code length $N_b$ represents the number of combined edges after the preprocessing.

3.1.2. Encoding of the Task Sequence. As shown in Figure 2, a corresponding handling task occurs before each processing task. Therefore, the task sequence is coded based on the processing task sequence. Each gene in the task sequence chromosome denotes a processing task, which is expressed by the job type number. The length of code is equal to the total number of processing tasks of all jobs put together. For instance, when there are three jobs need to be processed and each job requires three processing tasks, there can be merged into one combined edge (24, 6). According to this preprocessing principle, the guide-path network shown in Figure 1 has nine combined edges. All combined edges and serial numbers are as shown in Table 1. The UGN is encoded with a binary chromosome code $B = [b_{N_1} \ldots b_{N_b}]$. Each bit in the code corresponds to the direction of a combined edge. The chromosome code length $N_b$ represents the number of combined edges after the preprocessing.

3.2. Fitness Calculation. Since the individual in each subpopulation represents only partial optimization problem, it cannot be evaluated independently. Therefore, if an individual in a subpopulation need to be evaluated, it has to combine with an individual in the other subpopulation to form a complete chromosome. The fitness calculation method for a complete chromosome is described as follows.
A complete chromosome is composed of two parts. Take the complete chromosome of [1111111100][413421322123414132431234] as an example. The first half is the UGN code. Referring to Table 1, the corresponding UGN is constructed, as shown in Figure 3. And the shortest logistic distance between all the loading and unloading points can be obtained by using the Floyd algorithm [36], as shown in Tables 2 and 3. If the AGV travelling speed is given, the logistics transportation times between all the loading and unloading points can be calculated. Therefore, the UGN determines the logistic time between workstations.

The last half part of the complete chromosome is the task sequence code. First, the task sequence code is converted to a processing task sequence. Second, according to the fitness calculation flow shown in Figure 4, the machine tool and the AGV are scheduled according to the task sequence to calculate the fitness. Assuming that the technological route information of the job set is provided, as shown in Table 4, the number of AGVs in the system is 4, the AGV speed is 1 m/s. The Gantt chart obtained by the fitness calculation flow is shown in Figure 5, and the makespan is 1,260 seconds.

### 3.3. Subpopulation Initialization
The methods of the initial subpopulation generation for the two subpopulations are discussed as follows.

#### 3.3.1. UGN Subpopulation
The size of the UGN subpopulation is $N_{P1}$. The procedure for generating the initial UGN subpopulation is as follows. First, generate a random UGN chromosome code and assign each bit of the code a random value of 0 or 1. Then, decode this code and check whether the corresponding UGN is strongly connected; if yes, keep the code; otherwise, generate a new code. Repeat the above steps until $N_{P1}$ feasible codes are obtained. These feasible codes constitute the initial UGN subpopulation $G_{P1}$.

#### 3.3.2. Task Sequence Subpopulation
The size of the task sequence subpopulation is $N_{P2}$. $N_{P2}$ distinct chromosome individuals are generated randomly, and these individuals constitute the initial task sequence subpopulation $G_{P2}$.

### 3.4. Evolutionary Operations for the UGN Subpopulation
The evolutionary algorithm flow for the UGN subpopulation is shown in Figure 6. The key steps include selection, crossover, mutation, individual evaluation, neighbourhood search, and niche elimination operations. The specific operations are described as follows.

#### 3.4.1. UGN Selection
Firstly, all individuals in the parent subpopulation $G_{P1}$ are sorted from large to small according to their fitness. Then, two different individuals are selected to enter the crossover operation by using the roulette selection method. The probability of an individual being selected is proportional to its fitness.

#### 3.4.2. UGN Crossover
This operator is implemented using the double-point crossover method. A crossover operation example is shown in Figure 7. Firstly, two crossover positions are randomly generated. Then, the genes between the two positions of the parent individual 1 are replaced by the genes at the corresponding positions of the parent individual 2. The changed parent individual 1 becomes a new child individual.

#### 3.4.3. UGN Mutation Operation
The two-point mutation is employed, and a mutation operation example is shown in Figure 7. Firstly, two mutation positions are randomly generated. Then, the genes at the two positions are reversed. Finally, the strongly connectivity of the UGN that corresponds to the code is verified. If the UGN is strongly connected, evaluate the individual; otherwise, repeat the steps of selection, crossover, and mutation operations again.

#### 3.4.4. UGN Individual Evaluation
Since the UGN individual represents only partial optimization problem, it cannot be evaluated directly. Therefore, in order to evaluate an UGN individual, several individuals are randomly selected from the task sequence subpopulation by using the roulette method. Multiple complete chromosomes are formed by combining these task sequence individuals with the UGN chromosome code. The fitness of each complete chromosome is separately calculated, and the greatest fitness is selected as the individual fitness for the UGN chromosome. To improve the convergence speed of the algorithm, the complete chromosome with the highest fitness will be used by the neighbourhood search operation.

#### 3.4.5. Neighborhood Search
As already mentioned, a complete chromosome is composed of the UGN code and the task sequence code. A neighbourhood solution will be obtained by reversing a certain bit of the UGN code, while other bits are unchanged. For an original complete chromosome with a UGN code length of $N_{B}$, there are $N_{B}$ neighbourhood solutions. First, the strong connectivity of the UGN that corresponds to each neighbourhood solution is verified. If the UGN is strongly connected, calculate its fitness value; otherwise, set the fitness value to 0. Then, find

### Table 1: The combined edges after preprocessing.

| Edge no. | Combined edge | Edge no. | Combined edge | Edge no. | Combined edge |
|----------|---------------|----------|---------------|----------|---------------|
| 1        | (24, 6)       | 4        | (12, 18)      | 7        | (24, 6)       |
| 2        | (6, 9)        | 5        | (18, 21)      | 8        | (9, 21)       |
| 3        | (9, 12)       | 6        | (21, 24)      | 9        | (18, 12)      |

...
In the formula, $b^i_k$ is the value of the $k^{th}$ position in the UGN individual code $B^i$.

To avoid the early maturity in the evolution process of the subpopulation, it is necessary to maintain genetic diversity among the individuals in the subpopulation. Firstly, the $N_{p1}$ offspring individuals are merged with all $N_{p1}$ parent individuals to obtain $2N_{p1}$ individuals as the candidate subpopulation. Then, calculate the Hamming distance $H(B^i_i, B^j_i)$ between any two chromosomes. If the calculated $H(B^i_i, B^j_i)$ value is smaller than $H_{min}$ (which means they are similar chromosomes), multiply the fitness of the poor individual by an elimination factor $\eta < 1$. Finally, these $2N_{p1}$ individuals in the candidate subpopulation are sorted according to the adjusted fitness, and the optimal $N_{p1}$ individuals are selected to the next generation.

3.5. Evolutionary Operations for the Task Sequence Subpopulation. The flowchart of the evolutionary algorithm for the task sequence subpopulation is shown in Figure 9. The key steps include the selection, crossover, mutation, and individual evaluation operations. The details are provided as follows.

3.5.1. Task Sequence Selection. Using the roulette selection method, two different individuals from $G_{p2}^q$ are selected to enter the crossover operation step.

3.5.2. Task Sequence Crossover. This operator is implemented by two-point crossover. An example of crossover operation is shown in Figure 10. Assuming that the two parent individuals are Parent1 and Parent2, the crossover operation is described as follows:

(i) Step 1: randomly select two crossover positions Pos1 and Pos2

(ii) Step 2: translate the parent individual code into the processing task sequence
(iii) Step 3: rearrange the processing tasks of Parent1 between the two crossover positions according to their order in Parent2
(iv) Step 4: copy the rearranged Parent1 to the child individual
(v) Step 5: convert the child processing task sequence to the task sequence code.

3.5.3. Task Sequence Mutation. An example of the task sequence mutation operation is shown in Figure 11. Randomly select two crossover positions Pos1 and Pos2. If the job types are different at these two positions, then the codes of the two positions are exchanged. Whether the mutated individuals have appeared in the parent subpopulation needs to be verified. If this individual has appeared in the parent population, then the selection, crossover, and mutation operations will be repeated until a different individual is generated.

3.5.4. Task Sequence Evaluation. Firstly, the roulette method is used to select multiple individuals from the UGN subpopulation. Secondly, the task sequence individual is merged with these UGN individuals to form multiple complete chromosomes. Finally, calculate the fitness of each complete chromosome and select the largest fitness as the evaluated individual fitness.

3.5.5. Task Sequence Elitism Strategy. New \( N_{P2} \) offspring individuals are merged with \( N_{P2} \) parent individuals. These \( 2N_{P2} \) individuals constitute the candidate population. \( N_{P2} \) optimal individuals of the candidate population are selected to constitute a new generation task sequence subpopulation.

4. Experiments and Analysis

4.1. UGN Design Experiment. The layout of an example FMS is shown in Figure 1. The AGV speed is set at 1 m/s. The technological information of the job set is shown in Table 4. The integrated design method proposed in this paper is compared with the traditional stepwise design methods, which can be implemented as follows: firstly, the technological route information is converted into the loaded flow matrix among all loading and unloading points (Table 5). Secondly, with the objective of minimizing both loaded and empty travel distance, the method proposed by GUAN and DAI [8] is used to design the UGN. The results obtained by the method is the optimal UGN code [111111100], which corresponds to the UGN shown in Figure 3. The best empty flow matrix among all unloading and loading points is shown in Table 6. The total loaded and empty travel distance...
is  $J = J^L + J^E = 1920 + 800 = 2720$, which is calculated by formula (1) in reference [8]. Finally, with the makespan minimization objective, the simultaneous scheduling of machines and AGVs are addressed under this UGN by taking into account the constraints of the FMS application environment, e.g., the job handling and processing sequence constraint, as shown by formula (9)–(12), and the number of AGVs constraint, as shown in formula (5). The optimal results of the traditional methods and our approach for an AGVS with different numbers of AGVs and jobs are compared in Table 7.

As shown in Table 7, the number of AGVs and the task sequence constraints introduced by the job technological route are all neglected by the traditional design methods based on the loaded flow matrix. Since the loaded flow matrix that corresponds to the technological routes is unchanged, the optimal UGN results obtained by the traditional method remain the same in all cases. However, when taking into account the constraints of the FMS application environment and the makespan minimization objective, the integrated design model proposed in this paper can adjust the optimal UGN according to the increase number of AGVs, and it can obtain a better solution than the traditional design model. In addition, as shown in Table 7, when both the number of jobs and AGVs is 4, the traditional stepwise design method gets a solution with the makespan of 1260 and the actual travel distance of 2720. The corresponding Gantt chart is shown in Figure 5. On the other hand, the integrated design method can get a better solution with the makespan of 1260 and the actual travel distance of 3490. This indicates that the total loaded and empty travel distance is not directly related to the makespan. However, the traditional UGN design method can only optimize the UGN by minimizing the total loaded and empty travel distance and reducing the UGN quality of the FMS application environment inevitably. This experiment proves the superiority of the integrated design model.

4.2 Optimization Algorithm Experiment. In order to test the performance of the proposed CEGA algorithm, a comparative experiment of four algorithms is carried out. The four algorithms are as follows:

(1) TS: A tabu search algorithm. The neighbour definition of an UGN solution is consistent with reference [7] defined by Seo and Lee. The tabu size of the UGN solution is set as 20. The neighbours of a
task sequence solution are generated by exchanging the codes of the two positions where the job types are different. The tabu size of the task sequence solution is set at 100.

(ii) CEGA: The CEGA proposed in this paper. The parameters of the CEGA are set as follows. The population size of the UGN subpopulation $N_{P1}$ is set to be 20. The population size of task sequence subpopulation $N_{P2}$ is set at 100. Because the CEGA adopts elitist retention strategy to ensure the stability of the algorithm, so, we set the crossover rate $P_c = 1$ and the mutation rate $P_m = 1$.

(iii) TGA: among the CEGA, if without incorporating the niche technology and the neighbourhood search strategy, it becomes the traditional collaborative evolution genetic algorithm (TGA).

(iv) NGA: adding the niche strategy to the TGA will transform it into the niche collaborative evolution genetic algorithm (NGA). The parameters of the TGA and the NGA are consistent with that of the CEGA.

All programs are written in Plant Simulation 15.0 from Siemens Ltd. The program interface is shown in Figure 12. The running environment is Windows 10 with Intel Xeon E3-1535M V6 3.1 GHz, 32 GB RAM. Each algorithm is executed ten times using the same calculation example (both the number of jobs and AGVs is 4). The optimization process obtained by each algorithm is recorded every 0.4 minutes. The average makespan and the standard deviation performance of all the algorithms are shown in Figures 13 and 14.

As shown in Figure 13, the TS algorithm has a fast convergence speed in the early stage, when the search time is less than 0.8 minutes, it can get the local optimal solution as the best solution, but due to the lack of global search ability, the optimal result of the TS mainly depends on the initial
The i-th generation population \( G_i \)

Candidate population

The (i + 1)-th generation population \( G_{i+1} \)

Selection operation

Task sequence crossover operation

Task sequence mutation operation

Task sequence individual evaluation

Start

Generate initial task sequence population \( G_0 \)

\( n_p \) Parent individuals generation

\( n_p \) new individuals

\( n_p \) optimal individuals

Candidate population

\( n_p \) optimal individuals

Is the number of evolution generations reached?

No

Replace old population with new population

End

Interpopulation interaction

The i-th UGN population \( G_{p1} \)

NP1 Parent individuals generation

NP2 new individuals

NP2 optimal individuals

Generate initial task sequence population \( G_0 \)

Figure 9: Collaborative evolutionary algorithm for the task sequence subpopulation.

Before mutation

After mutation

Figure 10: Example of task sequence crossover operation.

Before mutation

After mutation

Figure 11: Example of task sequence mutation operation.
random solution, which has greater randomness. As shown in Figure 14, the standard deviation of the TS is always greater than 180, which indicates that the stability of the algorithm is poor. Different from TS algorithm, the standard deviation of the three genetic algorithms decreases with the progress of the search process. In the later stage of the search, the standard deviation of the three genetic algorithms can be reduced to less than 75, which shows that the stability of genetic algorithm is much higher than the TS. Among the three genetic algorithms, the TGA has the good initial convergence effect. As the diversity among populations cannot be guaranteed, the TGA is easily trapped into the

| Job | Mixing ratio (%) | Processing station Mm and processing time (s) |
|-----|------------------|---------------------------------------------|
| J1  | 25               | O1,1 M1(150) O2,2 M2(70) O3,3 M3(100) O4,4 M4(50) O5,5 M5(0) |
| J2  | 25               | M1(100) M2(50) M3(75) M4(150) M5(0) |
| J3  | 25               | M1(150) M2(75) M3(125) M4(60) M5(0) |
| J4  | 25               | M1(150) M2(150) M3(50) M4(50) M5(0) |

| Table 4: Technological route information of job set 1. |

| Job | Mm and processing time (s) |
|-----|-----------------------------|
| J1  | M1(150) M2(70) M3(100) M4(50) M5(0) |
| J2  | M1(100) M2(50) M3(75) M4(150) M5(0) |
| J3  | M1(150) M2(75) M3(125) M4(60) M5(0) |
| J4  | M1(150) M2(150) M3(50) M4(50) M5(0) |

| Table 5: Loaded flow matrix among all loading and unloading points. |

| From | To | P0 | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 |
|------|----|----|----|----|----|----|----|----|----|----|
| D1   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D2   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D3   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D4   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D5   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D6   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D7   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D8   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| D9   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |

| Table 6: Optimal empty flow matrix among all unloading and loading points. |

| From | To | P0 | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 |
|------|----|----|----|----|----|----|----|----|----|----|
| D1   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D2   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D3   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D4   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D5   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D6   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D7   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D8   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| D9   | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |

| Table 7: UGN design results. |

| Number of jobs | Number of AGVs | Traditional stepwise design method | Integrated design method |
|----------------|----------------|-----------------------------------|--------------------------|
|                |                | Makespan (s) Actual travel distance J (m) | The optimal UGN | Makespan (s) Actual travel distance J (m) | The optimal UGN |
| 4              | 2              | 1730 3290 [111111110] 1705 3090 [111111110] |
| 4              | 3              | 1265 2970 [111111110] 1265 3090 [111111110] |
| 4              | 4              | 1260 2720 [111111110] 1200 3490 [111111111] |
| 8              | 2              | 3120 6060 [111111110] 3060 5950 [111111111] |
| 8              | 3              | 2120 6130 [111111110] 2090 5880 [111111110] |
| 8              | 4              | 1660 5990 [111111110] 1650 6550 [111111110] |
| 8              | 5              | 1475 5990 [111111110] 1430 5930 [111111110] |
| 8              | 6              | 1410 6190 [111111110] 1395 6530 [111111110] |
local optimum. Although the addition of niche technology causes a relatively poor initial convergence of the NGA, it can maintain the diversity of the populations. Hence, the later search performance of the NGA is better than that of the TGA. The CEGA combines both benefits of the niche technique and the neighbourhood search strategy, which is computationally efficient with high solution quality. As shown in Table 7, when both the number of jobs and AGVs is four, the optimal makespan is 1,200. The proposed CEGA can stably obtain the global optimal solution after 3.6 minutes during each of ten runs, exhibiting the excellent overall performance.

5. Conclusions

A mutual coupling relationship exists between the UGN design process and the AGV task scheduling problem in the FMS environments. However, the coupling factors are often
neglected by using the stepwise design method in the pertinent literature, in order to reduce the complexity of the problem at the cost of the solution quality of UGN. The motivation of the paper is to bring the coupling factors into the integrated design process. On the one hand, an integrated design model of designing a UGN and scheduling AGVs simultaneously is proposed, with the objective of minimizing the makespan, in the consideration of the practical constraints. On the other hand, a dual-population collaborative evolutionary genetic algorithm (CEGA) is developed to solve the problems of designing and scheduling in a parallel way. The experimental results show the satisfactory efficiency and performances of our model and approach. It is noteworthy that the AGVs in this research work still pertain to the ordinary single-load type. However, multiple-load AGVs are becoming increasingly popular in modern factories, owing to their less traffic congestion, smaller fleet size, and increased system throughput [37, 38]. Hence, the integrated design and solution process oriented to multiple-load AGVs will attract great interest in the future research work.

Data Availability
The data used to support the findings of this study are included within the article.

Conflicts of Interest
The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant nos. 61973154, 61503319, and 51605414), National Defence Basic Scientific Research Program of China (Grant no. JCKY2018605C004), and Natural Science Research Project of Jiangsu Higher Education Institutions (Grant nos. 19KJB510013 and 18KJA460009).

References

[1] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Ensemble coordination approach in multi-agv systems applied to industrial warehouses," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 922–934, 2015.
[2] T. Le-Anh and M. B. M. De Koster, "A review of design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 171, no. 1, pp. 1–23, 2006.
[3] A. A. Faieza, R. T. Johari, A. M. Anuar, M. H. A. Rahman, and A. Johar, "Review on issues related to material handling using automated guided vehicle," *Advances in Robotics and Automation*, vol. 5, p. 140, 2016.
[4] I. F. A. Vis, "Survey of research in the design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 170, no. 3, pp. 677–709, 2006.
[5] M. Kaspi, U. Kesselman, and J. M. A. Tanchoco, "Optimal solution for the flow path design problem of a balanced unidirectional AGV system," *International Journal of Production Research*, vol. 40, no. 2, pp. 389–401, 2002.
[6] K.-C. Ko and P. J. Egbelu, "Unidirectional AGV guidepath network design: a heuristic algorithm," *International Journal of Production Research*, vol. 41, no. 10, pp. 2325–2343, 2003.
[7] Y. Seo, C. Lee, and C. Moon, “Tabu search algorithm for flexible flow path design of unidirectional automated-guided vehicle systems,” *OR Spectrum*, vol. 29, no. 3, pp. 471–487, 2007.

[8] X. Guan, X. Dai, and J. Li, “Revised electromagnetism-like mechanism for flow path design of unidirectional AGV systems,” *International Journal of Production Research*, vol. 49, no. 2, pp. 401–429, 2011.

[9] J. Rubaszewski, A. Yalaoui, L. Amodeo, and S. Fuchs, “Efficient genetic algorithm for unidirectional flow path design,” *IFAC Proceedings Volumes*, vol. 45, no. 6, pp. 883–888, 2012.

[10] I. K. Singgh, S. Hong, and K. H. Kim, “Flow path design for automated transport systems in container terminals considering traffic congestion,” *Industrial Engineering and Management Systems*, vol. 15, no. 1, pp. 19–31, 2016.

[11] J. Rubaszewski, A. Yalaoui, L. Amodeo, and S. Fuchs, “Extensions of the unidirectional flow path design problem solved by efficient metaheuristics,” *IFAC Proceedings Volumes*, vol. 46, no. 9, pp. 784–789, 2013.

[12] W. Xu, S. Guo, X. Li, C. Guo, R. Wu, and Z. Peng, “A dynamic scheduling method for logistics tasks oriented to intelligent manufacturing workshop,” *Mathematical Problems in Engineering*, vol. 2019, Article ID 7237459, 18 pages, 2019.

[13] 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,” *The International Journal of Advanced Manufacturing Technology*, vol. 81, no. 9–12, pp. 2123–2141, 2015.

[14] M. V. S. Kumar, R. Janardhana, and C. S. P. Rao, “Simultaneous scheduling of machines and vehicles in an FMS environment with alternative routing,” *The International Journal of Advanced Manufacturing Technology*, vol. 53, no. 1–4, pp. 339–351, 2011.

[15] H. E. Nouri, O. Belkhala Driss, and K. Ghédira, “Simultaneous scheduling of machines and transport robots in flexible job shop environment using hybrid metaheuristics based on clustered holonic multiagent model,” *Computers and Industrial Engineering*, vol. 102, pp. 488–501, 2016.

[16] A. Ahmadi-Javid and P. Hooshanghi-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*, vol. 53, no. 19, pp. 5988–6006, 2015.

[17] J. T. Lin, C.-C. Chiu, E. Huang, and H.-M. Chen, “A multi-fidelity model approach for simultaneous scheduling of machines and vehicles in flexible manufacturing systems,” *Asia-Pacific Journal of Operational Research*, vol. 35, no. 1, p. 1850005, 2018.

[18] P. Lacomme, M. Larabi, and N. Tchernev, “Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles,” *International Journal of Production Economics*, vol. 143, no. 1, pp. 24–34, 2013.

[19] F. Zhang and J. Li, “An improved particle swarm optimization algorithm for integrated scheduling model in AGV-served manufacturing systems,” *Journal of Advanced Manufacturing Systems*, vol. 17, no. 3, pp. 375–390, 2018.

[20] A. Ahmadi-Javid and P. Hooshanghi-Tabrizi, “Integrating employee timetabling with scheduling of machines and transporters in a job-shop environment: a mathematical formulation and an anarchic society optimization algorithm,” *Computers and Operations Research*, vol. 84, pp. 73–91, 2017.

[21] O. T. Baruwa and M. A. Piera, “A coloured Petri net-based hybrid heuristic search approach to simultaneous scheduling of machines and automated guided vehicles,” *International Journal of Production Research*, vol. 54, no. 16, pp. 4773–4792, 2016.

[22] O. T. Baruwa and M. A. Piera, “Identifying FMS repetitive patterns for efficient search-based scheduling algorithms: a colored Petri net approach,” *Journal of Manufacturing Systems*, vol. 35, pp. 120–135, 2015.

[23] X. Lyu, Y. Song, C. He, Q. Lei, and W. Guo, “Approach to integrated scheduling problems considering optimal number of automated guided vehicles and conflict-free routing in flexible manufacturing systems,” *IEEE Access*, vol. 7, pp. 74909–74924, 2019.

[24] G. Li, B. Zeng, W. Liao, X. Li, and L. Gao, “A new AGV scheduling algorithm based on harmony search for material transfer in a real-world manufacturing system,” *Advances in Mechanical Engineering*, vol. 10, no. 3, pp. 1–13, 2018.

[25] V. Chawla, A. Chanda, and S. Angra, “Automatic guided vehicle systems in flexible manufacturing system—A review,” *International Journal of Industrial Engineering: Theory, Applications and Practice*, vol. 26, pp. 737–765, 2019.

[26] P. R.utta, V. S. Chinthala, R. V. Manchoju, V. C. MVN, and R. Purohit, “A Review on facility layout design of an automated guided vehicle in flexible manufacturing system,” *Materials Today: Proceedings*, vol. 6, no. 2, pp. 3981–3986, 2018.

[27] M. Dai, D. Tang, A. Giret, and M. A. Salido, “Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints,” *Robotics and Computer-Integrated Manufacturing*, vol. 59, pp. 143–157, 2019.

[28] D. B. M. M. Fontes and S. M. Homayouni, “Joint production and transportation scheduling in flexible manufacturing systems,” *Journal of Global Optimization*, vol. 74, no. 4, pp. 879–908, 2019.

[29] Q.-V. Dang, C. T. Nguyen, and H. Rudová, “Scheduling of mobile robots for transportation and manufacturing tasks,” *Journal of Heuristics*, vol. 25, no. 2, pp. 175–213, 2019.

[30] M. Saidi-Mehrabad, S. Dehnavi-Arani, F. Erazbadian, and V. Mahmodian, “An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs,” *Computers and Industrial Engineering*, vol. 86, pp. 2–13, 2015.

[31] M. Gen, W. Zhang, and L. Y. Yun, “Recent advances in hybrid evolutionary algorithms for multi-objective manufacturing scheduling,” *Computers and Industrial Engineering*, vol. 112, pp. 616–633, 2017.

[32] M. K. Amjad, S. I. Butt, R. Kousar et al., “Recent research trends in genetic algorithm based flexible job shop scheduling problems,” *Mathematical Problems in Engineering*, vol. 2018, pp. 1–32, 2018.

[33] R. Kala, “Multi-robot path planning using co-evolutionary genetic programming,” *Expert Systems with Applications*, vol. 39, no. 3, pp. 3817–3831, 2012.

[34] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, “Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming,” *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 193–208, 2014.

[35] J. Gu, M. Gu, C. Cao, and X. Gu, “A novel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem,” *Computers and Operations Research*, vol. 37, no. 5, pp. 927–937, 2010.
[36] Q. Cheng, B. Sun, Z. Liu, Q. Feng, and P. Gu, “Geometric error compensation method based on Floyd algorithm and product of exponential screw theory,” Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, vol. 232, no. 7, pp. 1156–1171, 2018.

[37] Y.-C. Ho, H.-C. Liu, and Y. Yih, “A multiple-attribute method for concurrently solving the pickup-dispatching problem and the load-selection problem of multiple-load AGVs,” Journal of Manufacturing Systems, vol. 31, no. 3, pp. 288–300, 2012.

[38] V. K. Chawla, A. K. Chanda, and S. Angra, “Multi-load AGVs scheduling by application of modified memetic particle swarm optimization algorithm,” Journal of the Brazilian Society of Mechanical Sciences and Engineering, vol. 40, no. 9, p. 436, 2018.