applied sciences

MDPI

Article

Research on Flexible Job Shop Scheduling Problem with Handling and Setup Time Based on Improved Discrete Particle Swarm Algorithm

Jili Kong and Zhen Wang *

School of Modern Post, Beijing University of Posts and Telecommunications, Haidian District, Beijing 100876, China; kongjili1026@163.com
* Correspondence: wangz1210@126.com

Abstract: With the gradual emergence of customized manufacturing, intelligent manufacturing systems have experienced widespread adoption, leading to a surge in research interests in the associated problem of intelligent scheduling. In this paper, we study the flexible job shop scheduling problem (FJSP) with setup time, handling time, and processing time in a multi-equipment work center production environment oriented toward smart manufacturing and make-to-order requirements. A mathematical model with the optimization objectives of minimizing the maximum completion time, the total number of machine adjustments, the total number of workpieces handled and the total load of the machine is constructed, and an improved discrete particle swarm algorithm based on Pareto optimization and a nonlinear adaptive inertia weighting strategy is proposed to solve the model. By integrating the model characteristics and algorithm features, a hybrid initialization method is designed to generate a higher-quality initialized population. Next, three cross-variance operators are used to implement particle position updates to maintain information sharing among particles. Then, the performance effectiveness of this algorithm is verified by testing and analyzing 15 FJSP test instances. Finally, the feasibility and effectiveness of the designed algorithm for solving multi-objective FJSPs are verified by designing an FJSP test example that includes processing time, setup time and handling time.

Keywords: multi-equipment work center; flexible job shop scheduling problem; setup time; handling time; multi-objective optimization; improved discrete particle swarm optimization

Citation: Kong, J.; Wang, Z. Research on Flexible Job Shop Scheduling Problem with Handling and Setup Time Based on Improved Discrete Particle Swarm Algorithm. *Appl. Sci.* **2024**, *14*, 2586. https://doi.org/10.3390/app14062586

Academic Editor: Arkadiusz Gola

Received: 2 February 2024
Revised: 13 March 2024
Accepted: 14 March 2024
Published: 20 March 2024

1. Introduction

With the intensification of market competition and the acceleration of product updates, the modern manufacturing industry is facing challenges such as single-piece, small-batch, and personalized production [1]. In order to adapt to market changes, companies need to quickly adjust workshop layout and production plans to meet demand [2]. Therefore, the production mode of make-to-order (MTO) with multi-variety and small-batch demand is increasingly popular, and flexible job shop scheduling has become the focus of current research, especially for intelligent manufacturing systems with MTO requirements. The production systems of MTO with multi-variety and small-batch demand often employ multi-equipment work centers to maintain process flexibility and separate operation time for fine management. These are difficult and complicated to schedule and optimize for the production system with multi-equipment work centers due to external factors such as the various types of products ordered by customers, small batches, short lead times, and internal factors such as limited production resources. The scheduling problem of the production system with multi-equipment work centers introduces two different types of work centers, the processing equipment work center and the handling work center, and configures multiple pieces of processing equipment and handling equipment with the same function for them, respectively, so as to complete the production tasks of the MTO system

faster and more efficiently to adapt to changes in market demand. This kind of problem is an extension problem based on the FJSP, and it is also an NP-hard problem. Therefore, it can be extended on the basis of the FJSP, using the advantages of multi-equipment work centers and the mutual cooperation and resource sharing in each multi-equipment work center to adapt to the production needs of different products or orders, so as to improve the response speed and flexibility of the MTO system.

FJSPs for intelligent manufacturing and MTO have great application value in actual production. In industries like aerospace, airframe, shipbuilding, and machinery, orders are often unique and low in quantity, requiring customization based on specific customer needs and specifications [3]. Many scholars have researched the FJSP from two aspects. Some scholars have built different mathematical models to study the actual production problems of enterprises, while others have improved the algorithm, aiming to obtain an effective solution. The early workshop scheduling theory only considered the machine processing time factor and only took makespan as the optimization goal, but recently, various multi-objective scheduling optimization studies that are more in line with actual factory needs have been developed. Li et al. [4] established a flexible job shop scheduling model with the optimization objectives of minimizing the maximum completion time, minimizing total carbon emissions, and minimizing the machine load. Wei et al. [5] developed a flexible job shop scheduling problem with the goals of minimizing non-processing energy consumption, total weighted delay, delivery time, and maximum completion time. Nevertheless, the scheduling schemes formulated in the literature only take into account the machine processing time factor and overlook the impact of auxiliary time on scheduling.

With the in-depth study of the theory of shop scheduling, the FJSP should not only consider the workpiece processing time but also incorporate some necessary auxiliary time factors such as the workpiece handling time and the adjustment time of the processing machine. In this regard, some studies have been carried out. Zhang et al. [6] considered an FJSP with shipping and setup times while optimizing the maximum completion time, total energy consumption, critical machine workload, and early/late penalties, and established a multi-objective optimization mathematical model. Pal et al. [7] proposed a multi-agent approach to address the FJSP with setup and transportation times, aiming to minimize makespan. Zhang et al. [8] established a mixed-integer linear programming model, aiming at minimizing makespan and total energy consumption for the FJSP with setup and transportation time. Considering the transportation and sequence-based setup time constraints, Li et al. [9] established an integer programming model that optimized both energy consumption and makespan. However, in the process of research, the above literature only considers the situation in which the workpiece is single and different, and rarely considers the production demand for different parts in batches. For the scheduling of small-batch workpieces based on MTO requirements, there will be continuous processing of the same type of workpieces on the same machine, which will eliminate the interference of machine adjustment factors (tools and fixtures replacement, workpiece positioning, tool alignment measurement, program adjustment, etc.). In addition, some enterprises use the actual production of the work center as a unit to arrange production; at this time, the workpieces in the same multi-equipment work center can be transferred, ignoring the handling time. For example, Kubiak et al. [10] set up a dual-center flexible job shop-scheduling model with one machine in the first center and k parallel machines in the second center to solve the dual-center flexible job shop scheduling problem with the goal of minimizing the maximum completion time.

In FJSP solving, an efficient algorithm is the key to solving the problem model effectively. In recent years, many methods have been applied to solving the FJSP. For example, some studies have proposed precise algorithms, including mathematical programming and mixed-integer target programming, such as that by Choi et al. [11]. However, with the increase in the number of batch workpieces and the number of machines, the mathematical model has become extremely complex, the problem-solving difficulty has increased, and the traditional accurate algorithm has been unable to meet the solving needs. Therefore, many

scholars have begun to study intelligent optimization algorithms to obtain near-optimal solutions. In the field of evolutionary algorithms (EAs), a large number of researchers have made many improvements to the evolutionary algorithm to improve the effectiveness of the FJSP solution. Dai et al. [12] established a multi-objective optimization model, aiming at minimizing energy consumption and completion time for flexible job shop scheduling problems with transportation constraints, and proposed an improved genetic algorithm to solve this problem. Li and Lei [13] consider the energy-efficient flexible job shop scheduling problem with transportation and sequence-dependent setup times, and an imperialist competitive algorithm with feedback (FICA) is developed to minimize makespan, total tardiness, and total energy consumption simultaneously. On the basis of considering factors such as adjustment time and handling time, Feng and Kong [14] proposed an NSGA-II-V algorithm to solve the multi-equipment work center scheduling problem with the optimization objective of minimizing the maximum completion time and handling times. In addition, in the swarm intelligence (SI) field, the results of algorithm research and improvement are more outstanding. In solving the FJSP, combining two or more SI algorithms or criteria leads to better results than those obtained when using a single algorithm or criterion alone [15]. Thevenin and Zufferey [16] combined variable neighborhood search (VNS) and the ant algorithm (AA), and proposed a new general meta-heuristic algorithm, namely learning variable neighborhood search (LVNS), to solve job shop scheduling problems. Han et al. [17] developed a dynamic neighborhood search-based multi-objective artificial bee colony algorithm (MOABC) to solve the two-stage assembly flexible job scheduling problem with transportation and assembly time considerations. Sun et al. [18] proposed a hybrid multi-objective evolutionary algorithm (HMEA) to solve the multi-objective flexible job shop scheduling problem with transportation and setup time. Rossi [19] solved the flexible job shop scheduling problem with separable transport and sequence-dependent assembly time by using ant colony optimization (ACO) based on the sequence of ant colony pheromone relationships. Due to its simplicity and ease of implementation in comparison with other swarm intelligent optimization algorithms, PSO is a preferred option to be combined with other heuristic algorithms or criteria to solve multi-objective FJSPs [20].

To sum up, in the research of FJSPs for MTO requirements, there is relatively little research on equipment resource scheduling considering time factors such as processing, handling, and machine adjustment, and there are still some problems. On the one hand, the influence of the same type of workpiece on machine adjustment in the MTO order is not taken into account (this includes the replacement of tools and fixtures, workpiece positioning, tool alignment measurement, program adjustment, and more). On the other hand, when some enterprises use multi-equipment work centers for production arrangements in actual production, the workpiece handling time in the same work center can be ignored. Therefore, based on the background described above, aiming at fulfilling the MTO requirements of multi-variety and small-batch workpiece processing tasks, the FJSP with multi-equipment work centers as the production unit is studied with the following objectives achieved:

- To meet the customers' demands for multi-variety and small-batch production, a mathematical model of the multi-equipment work center scheduling problem with setup, handling, and processing time is established;
- An improved discrete particle swarm algorithm (IDPSO) based on Pareto meritocracy and a nonlinear adaptive inertia weighting strategy is proposed to solve the model;
- By using three improved crossover and variational operators to update the particle positions, information sharing among particles is achieved, and thus the search capability of the algorithm is improved.

The rest of this article is structured as follows: Section 2 provides a description of the FJSP_SHP problem; Section 3 outlines the mathematical modeling process; and Section 4 introduces an improved discrete particle swarm optimization algorithm. In Section 5, an example simulation verifies the effectiveness of the model algorithm, and Section 6 concludes the paper and outlines future research plans.

2. FJSP_SHP Description

2.1. Problem Description

The FJSP-containing setup, handling and processing time (FJSP_SHP) are formally formulated as follows: In the actual production process, factories often encounter multiple varieties and small batches of order-processing tasks. There are g types of batch jobs to be processed in the workshop, and the number of jobs of type i is n_i, the set of jobs is $\boldsymbol{J} = \{J_{ij} | 1 \le i \le g, 1 \le j \le n_i\}$, The k-th process of job J_{ij} is O_{ijk}, the total number of operations of the j-th job in the i-th job-type is n_{ij}, and the set of operations is $\boldsymbol{O} = \left\{ O_{ijk} \middle| 1 \le i \le g, 1 \le j \le n_i, 1 \le k \le n_{ij} \right\}$; the process sequence of the same type of jobs is certain, but the process sequences of different types of jobs are different. There are N_{WC} work centers, and each work center contains multiple machines and pieces of equipment that can process jobs. Each kind of job has its own set processing route. Each processing route contains multiple processing operations. Each operation can select multiple machines in a processing equipment work center. The processing time of the job on different machines is different. The handling equipment is utilized to transport workpieces between different processing equipment work centers, and there is a handling time associated with the handling process. The equipment within each processing equipment work center requires an adjustment time before processing some specific workpieces. This article only considers the adjustment time related to these types of workpieces, such as changing tooling and fixtures, workpiece positioning, and tool setting. If the processing equipment consecutively processes workpieces of the same type, the adjustment time is zero. Conversely, if the workpiece type changes, the adjustment time is determined by the inherent adjustment time specific to that particular workpiece on the equipment. Each machine can only process one job at a time. The processing time, t^p_{ijkml}, of O_{ijk} in each process on the l-th equipment, M_{ml}, in the m-th work center, WC_m, is known, the setup time, t^s_{ijkml}, before processing and the handling time, t^h_{ijkml}, after processing to the next work center are also known.

The research background of the problem is based on the MTO requirement of multi-variety and small-batch workpiece processing tasks, for which three subscripts are assigned in each process. For example, "O_{ijk}" indicates the k process of the j-th workpiece of type i. This can be a good response to the MTO characteristics of multiple varieties and small batches. In addition, introducing the workpiece-type subscript information also serves two purposes. Firstly, it serves to differentiate information among various types of workpieces and aids in calculating the number of machine adjustments. Conversely, disregarding the types of workpieces would result in an increase in the total number of machine adjustments for the scheduling plan. Secondly, it also helps in examining the quantity variations across different workpiece batches and tackling the challenge of dynamically scheduling the insertion sequence of workpieces in future studies.

2.2. Assumptions

For the FJSP, the general assumptions are as follows [21]: (1)~(5). In addition, in view of the actual production situation of the multi-equipment processing work center, assumptions (6)~(9) of processing, machine adjustment and handling equipment capacity should also be considered.

(1) All jobs and equipment are ready at zero time.
(2) Different jobs have the same level of priority. The processes of the same jobs are processed in strict accordance with the processing sequence. Each process can only be performed after the previous process is completed.
(3) Only one machine can be selected for processing in each operation.
(4) Disruption of jobs during processing is not considered.
(5) One machine can only process one operation at a time.

(6) The processing equipment requires adjustment before processing different types of workpieces; conversely, if processing the same type of workpiece, no readjustment is needed.
(7) The carrying capacity of each handling equipment is not limited, and the number of handling equipment is not limited.
(8) The transportation time is not considered after the last working procedure of the job is completed.
(9) When the adjacent processes of the same job are processed in the same work center, the handling time does not need to be considered. When the adjacent processes of the same job are processed in different work centers, the handling time of the job is determined.

Figure 1 is a schematic diagram of multi-equipment work centers; each manufacturing work center contains at least one piece of equipment, and each processing equipment work center can complete at least different processes for any workpiece in the MTO. Multiple pieces of equipment in the processing equipment work center can be completely related (that is, they complete all processes of the same work piece) or unrelated (that is, the work piece is not necessarily completed by the current multi-equipment work center) [14]. The workpiece needs to be transferred between different processing equipment work centers by means of the equipment of the handling work center, which can include vehicles, handling robots, etc., and the number of workpieces to be transported by handling equipment is greater than or equal to 1. The manufacturer will then arrange a reasonable production scheduling plan in accordance with the customer's order delivery time and the enterprise's efficiency, so as to shorten the manufacturing cycle, improve the equipment utilization rate, and deliver the goods on time.

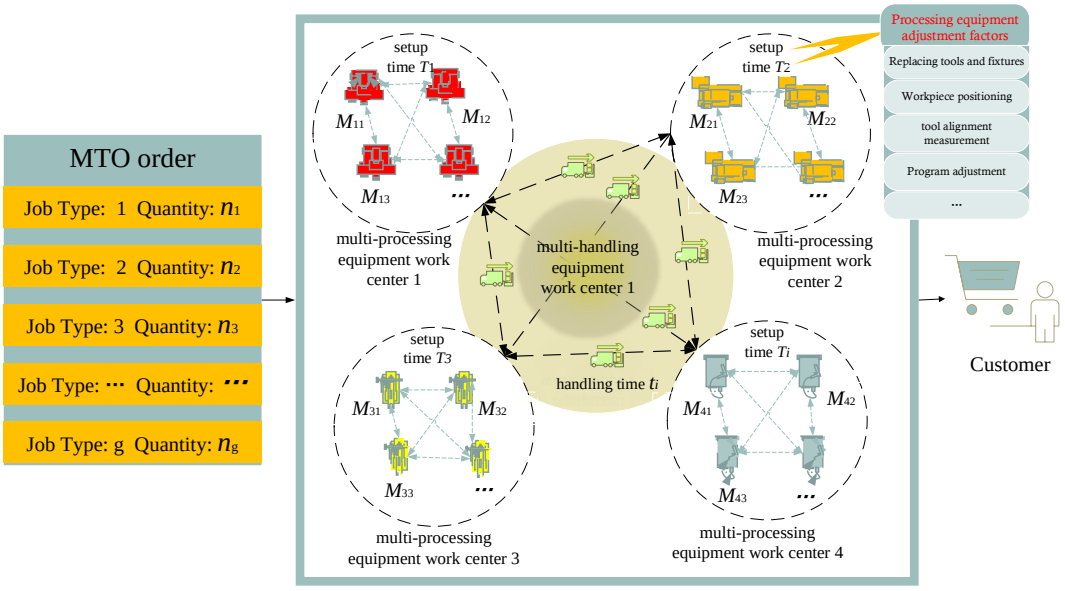

Figure 1. A schematic diagram of multi-equipment work centers.

This paper aims to establish an efficient processing and production plan for a multi-variety of small-batch job orders. Specifically, the focus is on determining the optimal sequence for job processing, arranging equipment resources effectively, and planning the specific operation processing time, equipment setup time, and job handling time. Under the premise of ensuring the optimal maximum completion time of the order, the total number of machine adjustments, the total number of handling tasks and the total load of the machine are reduced as much as possible, so as to improve the economic efficiency of the entire workshop.

3. Mathematical Model

3.1. Parameter and Variable Definitions

The relevant variables and parameters of the model are shown in Table 1.

Table 1. Parameter variable.

Variables	Definitions
i	job type number
j	job number
k, k'	job process number
WC_m	m-th work center
N_{WC}	total number of work centers
M_{ml}	the l-th processing equipment in the m-th work center, WC_m
g	total number of job types
n_i	total number of batch jobs of type i jobs
n_{WC_m}	total number of machines and equipment in the WC_m of the work center
n_{ij}	total number of operations of the j-th job in the i-th job type
O_{ijk}	k-th operation of the j-th job in the i-th job type
$t^p_{(i,j,k,m,l)}$	processing time of the process, O_{ijk}, on the equipment, M_{ml}
$t^p_{(i,j,k,m,l)Bp}, t^p_{(i,j,k,m,l)Ep}$	start time and end time of the process, O_{ijk}, on the equipment, M_{ml}
t^s_{ijkml}	setup time of process, O_{ijk}, before processing on equipment M_{ml}
$t^s_{(i,j,k,m,l)Bp}, t^s_{(i,j,k,m,l)Ep}$	starting and ending setup time of the process, O_{ijk}, on the equipment, M_{ml}
t^h_{ijkml}	handling time after the process, O_{ijk}, is processed on the equipment, M_{ml}
$t^h_{(i,j,k,m,l)Bp}, t^h_{(i,j,k,m,l)Ep}$	start time and end time of job handling after the completion of the process, O_{ijk}
L	an infinite positive number
C_{ij}	completion time of the j-th job in the i-th job type
C_{ijk}	completion time of the k-th operation of the j-th job in the i-th job type
C_{max}	maximum completion time
F_r	function of the r-th decision objective
T_{ml}	total processing time of the l-th processing equipment in the m-th work center, WC_m
T_{load}	total machine load
NOM_{ml}	amount of downtime of the l-th processing equipment in the m-th work center, WC_m
NOC_{ijk}	number of handling tasks of the job after the k operation of the j-th job of type i is completed

The decision variables in this paper are shown below.

$$x_{ijkml} = \begin{cases} 1, \text{ if the process } O_{ijk} \text{ is processed on} \\ \quad \text{the processing equipment } M_{ml}; \\ 0, \text{ otherwise;} \end{cases}$$

$$y_{ijkpqk'} = \begin{cases} 1, \text{ if the process } O_{ijk} \text{ is processed on the processing} \\ \quad \text{equipment } M_{ml} \text{ before process } O_{pqk'}; \\ 0, \text{ otherwise;} \end{cases}$$

$$z_{ijk} = \begin{cases} 1, \text{ if the process } O_{ijk} \text{ is processed on equipment } M_{ml1}, \text{ and} \\ \quad \text{process } O_{ij(k+1)} \text{ is processed on equipment } M_{ml2}; \\ 0, \text{ otherwise.} \end{cases}$$

$$\sigma_{ijk} = \begin{cases} 1, \text{ if the process } O_{ijk} \text{ on the equipment } M_{ml} \text{ processes the} \\ \text{same type of workpiece as the pre} - \text{processing process;} \\ 0, \text{ otherwise.} \end{cases}$$

3.2. Objective Functions

In this paper, the maximum completion time of batch jobs, the total number of machine adjustments, the total number of workpiece handlings, and the total machine load are

considered objectives through which to optimize the optimal production scheduling scheme. The following are the designed decision-making objective functions:

(1) Minimize the maximum completion time.

$$min\ F_1 = min\ C_{max} = min\ max\{C_{ij} | 1 \le i \le g, 1 \le j \le n_i\} \quad (1)$$

Shortening the maximum completion time of an order batch task and ensuring punctual delivery comprise the most fundamental optimization index in scheduling research.

(2) Minimize the total number of machine adjustments.

$$min\ F_2 = min \sum_{m=1}^{N_{WC}} \sum_{l=1}^{n_{WC_m}} NOM_{ml} \quad (2)$$

Minimizing the total number of adjustments to the processing equipment can reduce the number of equipment interruptions in production, decrease the likelihood of human intervention and enhance the quality stability of continuous processing operations, and the utilization rate of the machines. Additionally, this also helps to reduce the energy consumption of the equipment. In flexible job shop scheduling, one machine may process different workpieces, so adjustments need to be made. In accordance with the final production scheduling result, the total number of machine adjustments to the processing equipment is calculated.

(3) Minimize the total number of workpiece handlings.

$$min\ F_3 = min \sum_{i=1}^{g} \sum_{j=1}^{n_i} \sum_{k=1}^{n_j - 1} NOC_{ijk} \quad (3)$$

Minimizing the total number of workpiece handling tasks can reduce the number of handling equipment inputs and the number of handling workpieces, and improve the utilization rate of handling equipment, which can also indirectly affect the cost related to workpiece handling tasks. According to the final production scheduling result, the total number of workpieces handled is calculated.

(4) Minimize the total load of the machine.

$$minF_4 = minT_{load} = min \sum_{m=1}^{N_{WC}} \sum_{l=1}^{n_{WC_m}} T_{ml} \quad (4)$$

In production and processing, we should not only minimize the maximum completion time but also consider the total load of the machine. By minimizing the total processing load of the machine, we can increase the service life of the machine and indirectly reduce the production cost.

In order to facilitate the calculation of the fitness of the objective function in the model, the reciprocals of the values of the above four objective functions are taken as the fitness functions, as shown in Formula (5):

$$f_r = 1/F_r (r = 1, 2, 3, 4) \quad (5)$$

3.3. Constraints of Parallel Movement Mode

In this study, in accordance with the actual production situation, the constraints of the designed FJSP_SHP problem are as follows:

(1) Sequence constraints of any process, O_{ijk}, in the workpiece, J_{ij}.

$$t^p_{(i,j,k,m,l)Bp} + x_{ijkml} \cdot t^p_{(i,j,k,m,l)} \le C_{ijk}\ \forall i, j, k, m, l \quad (6)$$

$$C_{ijk} \le t^p_{(i,j,k+1,m,l)Ep} \qquad \forall i, j, k, m, l \quad (7)$$

(2) One machine can only process one operation at a time.

$$t^p_{(i,j,k,m,l)Bp} + t^p_{(i,j,k,m,l)} \le t^p_{(p,q,k',m,l)Bp} + L(1 - y_{ijkpqk'})\ \forall i, j, k, p, q, k', m, l \quad (8)$$

(3) A certain process of each job can only be processed on one machine of a work center.

$$\sum_{l=1}^{n_{WC_m}} x_{ijkml} = 1 \quad \forall i, j, k, m, l \tag{9}$$

(4) The start time of an operation of a job is not less than the sum of the completion time of the previous operation and the setup time of this operation.

$$t^p_{(i,j,k,m_1,l_1)Ep} + t^s_{(i,j,k+1,m_2,l_2)}(1-\sigma_{ijk}) \le t^p_{(i,j,k+1,m_2l_2)Bp} \ \forall i, j, k, m_1, m_2, l_1, l_2 \tag{10}$$

(5) The start time of an operation of a job is not less than the sum of the completion time and handling time of the previous operation.

$$t^p_{(i,j,k,m_1,l_1)Ep} + t^h_{(i,j,k,m_1,l_1)}\left(1-z_{ijk}\right) \le t^p_{(i,j,k+1,m_2l_2)Bp} \ \forall i, j, k, m_1, m_2, l_1, l_2 \tag{11}$$

(6) The start processing time and end processing time of any operation, O_{ijk}, shall not be less than 0.

$$t^p_{(i,j,k,m,l)Bp} \ge 0\,, \ \ t^p_{(i,j,k,m,l)Ep} \ge 0 \quad \forall i, j, k, m, l \tag{12}$$

(7) The processing time relationship of any process, O_{ijk}, of a job on the processing equipment, M_{ml}.

$$t^p_{(i,j,k,m,l)Ep} = t^p_{(i,j,k,m,l)Bp} + t^p_{ijkml} \ \ \forall i, j, k, m, l \tag{13}$$

$$t^p_{(i,j,k,m,l_1)Ep} \le t^p_{(i,j,k+1,m,l_2)Bp} + L(1-z_{ijk}) \ \forall i, j, k, m, l_1, l_2 \tag{14}$$

(8) The time relationship on the processing equipment, M_{ml}, must be adjusted before any one process, O_{ijk}, of a job starts processing.

$$t^s_{(i,j,k,m,l)Ep} = t^s_{(i,j,k,m,l)Bp} + t^s_{ijkml}\left(1-\sigma_{ijk}\right) \ \forall i, j, k, m, l \tag{15}$$

(9) The time relationship of a job must be handled after processing is completed in any one process, O_{ijk}.

$$t^h_{(i,j,k,m,l)Ep} - t^h_{(i,j,k,m,l)Bp} = \left(1-z_{\,ijk}\right)t^h_{ijkml} \ \forall i, j, k, m, l \tag{16}$$

(10) After any process, O_{ijk}, in which the processing of a job is completed, the handling time relationship of the next process is determined under different processing equipment.

$$t^p_{(i,j,k,m_1,l_1)\mathrm{Ep}} \le t^h_{(i,j,k,m_2l_2)Bp} \ \forall i, j, k, m_1, m_2, l_1, l_2 \text{ and } l_1 \ne l_2 \tag{17}$$

4. IDPSO Algorithm Analysis and Design

Particle swarm optimization (PSO) is an evolutionary computing technique derived from the study of bird predation behavior. The basic idea of the particle swarm optimization algorithm is to find the optimal solution through cooperation and information sharing among individuals in the group. As shown in Table 2, the parameters of six commonly used algorithms are counted. Compared with the other five algorithms, PSO has the fewest number of parameters, which is only four. In addition, the literature [20] points out that the advantage of PSOs is that they are simple and easy to implement and do not have many parameters to adjust.

Table 2. A statistical table of six algorithms' parameters.

Algorithm	Special Parameters	General Parameters	Parameter Number
MOABC [17]	Maximum evaluation times, weighted value, maximum searches limit, search threshold.	Population size, maximum iteration number.	6
ACO [19]	Pheromone factor, heuristic information factor, evaporation coefficient, pheromone increment constant, initial pheromone concentration.		7
PSO [20]	Inertia factor, learning factors.		4
IGA [22]	Number of objective divisions, mutation rate, crossover rate, selection rate.		6
NSGA-II [23]	Mutation rate, crossover rate, selection rate.		5
DQNSGA [24]	Number of objective divisions, mutation rate, crossover rate.		5

(1) Basic idea

PSO simulates birds in a flock by designing a massless particle with only two properties: speed, which represents how fast it moves, and position, which represents the direction of movement. Each particle separately searches for the optimal solution in the search space, and records it as the current individual extreme value, shares the individual extreme value with other particles in the whole particle swarm, and finds the optimal individual extreme value as the current global optimal solution of the whole particle swarm. All the particles in the swarm adjust their speed and position according to the current individual extreme value they find and the current global optimal solution shared by the whole swarm.

(2) Update rule

PSO is initialized to a group of random particles (random solutions). Then, the optimal solution is found through iteration. In each iteration, the particle updates itself by tracking two extreme values (*pbest*; *gbest*). After finding these two optimal values, the particle updates its velocity and position by using the following formula.

$$v_{i+1} = wv_i + c_1 r(pbest_i - x_i) + c_2 r(gbest - x_i) \quad (18)$$

$$x_{i+1} = x_i + v_{i+1} \quad (19)$$

where i = 1, 2, 3…, N. N is t he total number of particles in this group. W is the inertia factor, and c_1 and c_2 are learning factors. W keeps the particles moving inertly, makes them have the trend of expanding the search space, and gives them the ability to explore new areas. c_1 and c_2 represent the weights of the statistical accelerators that push each particle into $pbest_i$ and $gbest$ positions; $pbest_i$ is the individual optimal location searched by the current i-th particle, and $gbest$ is the global optimal location searched by the group particle.

4.1. Encoding and Decoding

As encoding and decoding involve two aspects of process arrangement and machine selection, the two-layer coding method of process and machine is adopted here, and the length of the coding in the process layer and the machine layer is equal, while the positions are one-to-one. Table 3 shows an example of the FJSP. The first number, 111, in the operation column represents the first operation of the first job in the first type, namely O_{111}, the second number, 112, represents the second operation of the first job in the first type, namely O_{112}, the third number, 211, represents the first operation of the first job in the second type, namely O_{211}, and so on. Therefore, the operation code uses the type of job number in the operation arrangement, and the number of occurrences indicates the

processing sequence of the job operation. As shown in Figure 2, if an operation is arranged as $O_{111} - O_{211} - O_{221} - O_{112} - O_{311} - O_{212} - O_{222} - O_{312} - O_{213} - O_{223}$, its operation code can be expressed as [11 21 22 11 31 21 22 31 21 22].

Table 3. A flexible job shop scheduling problem.

Job Type	Job	Operation Number	OS	Machine and Processing Time/min				
				Work Center 1	Work Center 2		Work Center 3	
				WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}
Type 1	Job 1	1	111	2	3	3	-	-
		2	112	3	-	-	2	2
Type 2	Job 1	1	211	-	1	1	3	3
		2	212	2	-	-	2	2
		3	213	3	3	3	-	-
	Job 2	1	221	-	1	1	3	3
		2	222	2	-	-	2	2
		3	223	3	3	3	-	-
Type 3	Job 1	1	311	-	2	2	3	3
		2	312	1	2	2	-	-

Note: "-" indicates that the operation cannot be processed on the machine of the work center.

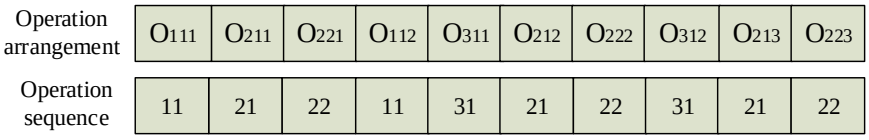

Figure 2. Operation code.

Before coding the machine, the machines WC_{11}, WC_{21}, WC_{22}, WC_{31} and WC_{32} are numbered 1, 2, 3, 4 and 5 in sequence; then, the machine number can be selected according to the operation in the operation sequence (OS). As shown in Figure 3, according to the OS, a machine is successively found from the optional machining machine set in each process, and then one of the machine sequences (MS) is [1 2 3 4 2 5 1 3 2 1]. The first element, 1, in the coding sequence means that O_{111} chooses to process on the first equipment on the first work center with the number of 1, and the second element, 2, means that O_{211} chooses to process on the first equipment on the second work center with the number 2, and so on.

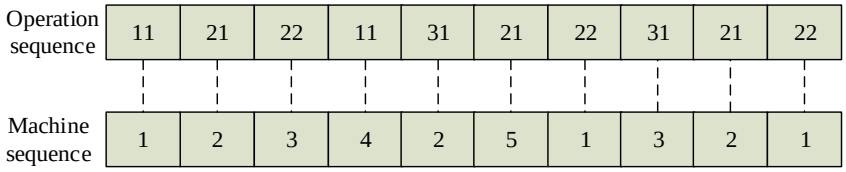

Figure 3. Machine code.

In this paper, the multi-factor flexible job shop scheduling problem under parallel multi-equipment is studied, including machine setup time, job handling time, and machine processing time, for multi-equipment production systems with multi-variety and small-batch job processing task orders.

4.2. Population Initialization

The quality of the initial population has a great influence on the quality of the solution and the convergence speed of the optimization algorithm. Solving the FJSP requires selecting the best machine for the operation sequence and each operation, and the operation sequence affects the selection of machines.

In the initialization process, we use different initialization methods based on two-level coding to generate the initial population and maintain particle diversity. In order to achieve a better scheduling effect, we use the random initialization method in the process selection part. Because the processing time of the job on the machine directly affects the completion time of the operation, we can use a mixed-initialization method to select the machine. At the same time, the SPT (shortest processing time) rule is to preferentially select the machine with the shortest processing time, which can reduce the time it takes to complete an order. Compared with other rules, SPT rules can directly reduce customer waiting times, thereby improving workshop productivity and customer satisfaction. The specific operation steps are as follows: for the processing of 60% particles, we randomly select a machine in the machine selection set; for the processing of 40% particles, we choose the machine with the shortest processing time. This approach balances randomness with the need to consider the SPT to achieve better scheduling results.

4.3. Update of Particle Position

For the FJSP, as a typical discrete model optimization problem, the research shows that standard PSO is poor. Therefore, Ding et al. [25] introduced three operators in the updating process of particles to update the position of particles and designed DPSO. Experiments show that DPSO is efficient in solving FJSP problems and can quickly search for the near-optimal solution. The particle position updating method is shown in Formula (20):

$$X_i^{t+1} = \omega \otimes op_1\left(X_i^t\right) + c_1 \otimes op_2\left(X_i^t, Pb_i^t\right) + c_2 \otimes op_3\left(X_i^t, Gb^t\right) \tag{20}$$

The values of ω, c_1 and c_2 are between (0,1), representing the probability that the particle is affected by the current solution, the individual historical optimal solution and the global optimal solution, respectively. t represents the current algebra for particle swarm optimization, X_i^t represents the current position of particle i, Pb_i^t represents the historically optimal position of particle i after t generation evolution, and Gb^t represents the optimal position of all particles in the t generation. The "$\otimes$" operator indicates that the operation on the right side of "$\otimes$" should be performed when the probability on the left side of "$\otimes$" is satisfied, and the "+" operator indicates that the operation on the right side of "+" should be performed after the operation on the left side of "+" is completed.

In this paper, according to the characteristics of the problem, a coding method containing information about the type and batch of the workpiece is proposed, and three new operators are designed. In order to improve problems such as slow convergence, low stability, and the tendency to fall into local optimality in the solving process of PSO, a new nonlinear adaptive ω variation strategy is proposed by improving the inertia weight, ω, as shown in Formula (21). The proposed method can be adaptively adjusted according to particle velocity and position, and the inertia weight of the improved algorithm shows a nonlinearly decreasing state, which has a trend of expanding space searching, enlarging the particle search area, avoiding falling into local optimal, ensuring the high behavioral divergence of particles, and thus improving the search efficiency of the algorithm.

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min}) \tan\left(\frac{t\pi}{4T}\right) \tag{21}$$

where w is the inertia weight, t is the number of current iterations, and T is the total number of iterations.

The op_1 operator represents the inertial motion of particles affected by the current position, which is reflected in DPSO as particle mutation. The specific mutation operation method is as follows: exchange two position elements randomly in the OS sequence of particles, and adjust the corresponding machine according to the sequence of each process after the exchange; at the corresponding adjustment position in the MS sequence, randomly select a machine from the set of optional machines for its operation to be replaced. The op_1 operator is shown in Figure 4.

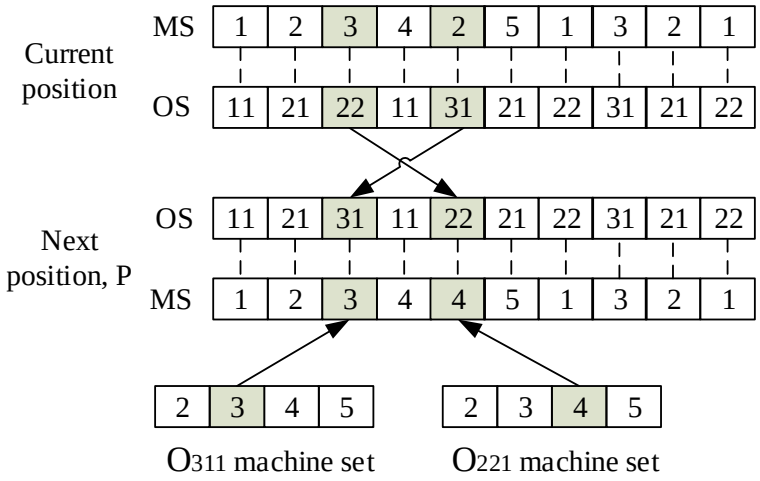

Figure 4. op_1 operator.

The op_2 operator represents the process of the current particle learning from its individual optimal position. It is improved using the POX cross-strategy described in the literature [26]. The operation process is as follows:

(1) All job numbers are randomly divided into two non-empty sets, S_1 and S_2, for example, $S_1 = \{J_{11}, J_{21}\}$, $S_2 = \{J_{22}, J_{31}\}$.
(2) Two empty sub generations, O_1 and O_2, are produced, and the process code belonging to S_1 in P (P_b) is copied to the corresponding position in $O_1(O_2)$. P is a particle of the current population, and P_b is the historical optimal location of the individual particle.
(3) Then, the process coding sequence belonging to S_2 in P (P_b) is inserted into $O_2(O_1)$.
(4) Then, the particles with higher fitness are screened from O_1 and O_2.

The op_2 operator is shown in Figure 5.

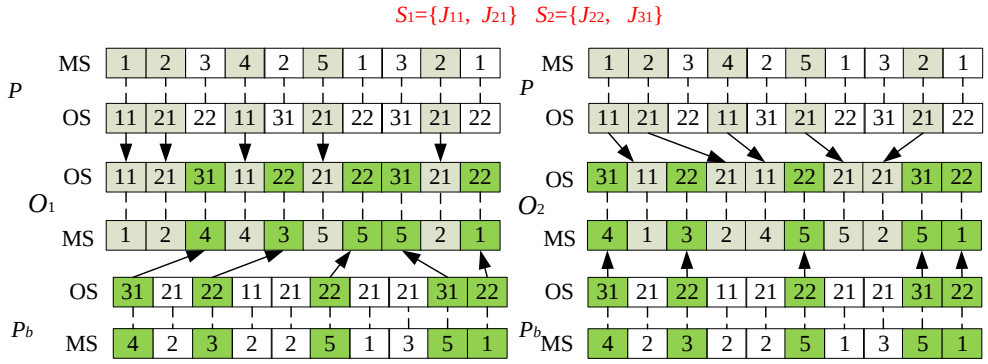

Figure 5. op_2 operator.

The op_3 operator represents the particle motion affected by the optimal position of the current global particle. The op_3 operator is the same as the op_2 operator above in principle. First, replace P_b with G_b to complete the op_2 operation, then select particles with better target values from O_1 and O_2, and adjust their MS sequence. In the MS sequence, a position is randomly selected, and the contents of the position are randomly selected again according to the set of machines available for the corresponding process. The adjustment strategy of the op_3 operator is shown in Figure 6. Suppose that the particle with a larger fitness screened from O_1 and O_2 is the O_2 particle, and the new particle position E is obtained after adjusting the MS sequence through the op_3 operator.

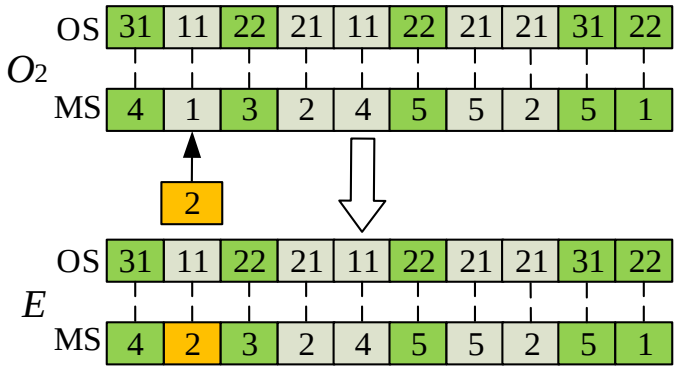

Figure 6. op_3 operator MS sequence adjustment.

4.4. Update of Pbest Selection Method

Let X_i be the current position of the particle of generation t, and let its corresponding individual optimal position be Pb_i^t; the particle position in the next iteration is X_{i+1}, and the corresponding individual optimal position is Pb_{i+1}^{t+1}. When solving a multi-objective FJSP, the *pbest* selection method is as follows: if $Pb_i^t < Pb_{i+1}^{t+1}$, then $pbest = X_{i+1}$; If $Pb_{i+1}^{t+1} < Pb_i^t$, and $pbest = X_i$; If $Pb_i^t \sim Pb_{i+1}^{t+1}$; this means that the two do not dominate each other. At this time, it is necessary to make a secondary selection, taking into account order delivery satisfaction and product quality. The maximum completion time, f_1, the total number of machine adjustments, f_2, and the total number of workpiece handlings, f_3, are compared in order to select the particle *pbest*. Then, secondary screening is required to select particles with a high fitness of f_1; if the f_1 of two particles is equal, the particle with a large fitness of f_2 is selected, and if the f_1 and f_2 of the two particles are equal, the particle with a high fitness of the f_3 target is selected.

4.5. Pareto Optimal Solution Set and gbest Selection

In one update process of particle swarm optimization, there may be multiple particles in the population that are not dominated by other particles, and these particles constitute the Pareto solution set of this iteration. In this paper, the method in the literature [27] is used to construct the Pareto solution set, and its Pareto level 1 solution set is selected as the Pareto solution set in this algorithm.

In each iteration of PDPSO, the Pareto non-inferior solution set is updated and maintained through an optimal set, $\boldsymbol{S}$, so that the saved solutions are gradually closer to the real Pareto front and distributed as evenly as possible. In this algorithm, the optimal set, $\boldsymbol{S}$, and *gbest* are selected at the same time so as to ensure that individuals in the optimal set, $\boldsymbol{S}$, have at least one particle *gbest*, so that the algorithm can search all Pareto fronts as far as possible and avoid local convergence. In the optimal set, $\boldsymbol{S}$, each particle is a Pareto boundary-efficient solution, and particles do not dominate each other. Therefore, a *gbest* selection strategy needs to be developed. The method is as follows: First, select particles with high fitness according to target f_1. Secondly, if the number of particles chosen by f_1 is greater than one, choose particles with high fitness according to target f_2. Finally, if the number of particles chosen by f_1 and f_2 is greater than one, choose a particle with high fitness according to target f_3.

4.6. Improve the Flow of Hybrid Discrete Particle Swarm Optimization

The IDPSO algorithm flow for solving multi-objective FJSPs is shown in Figure 7. The specific steps are as follows:

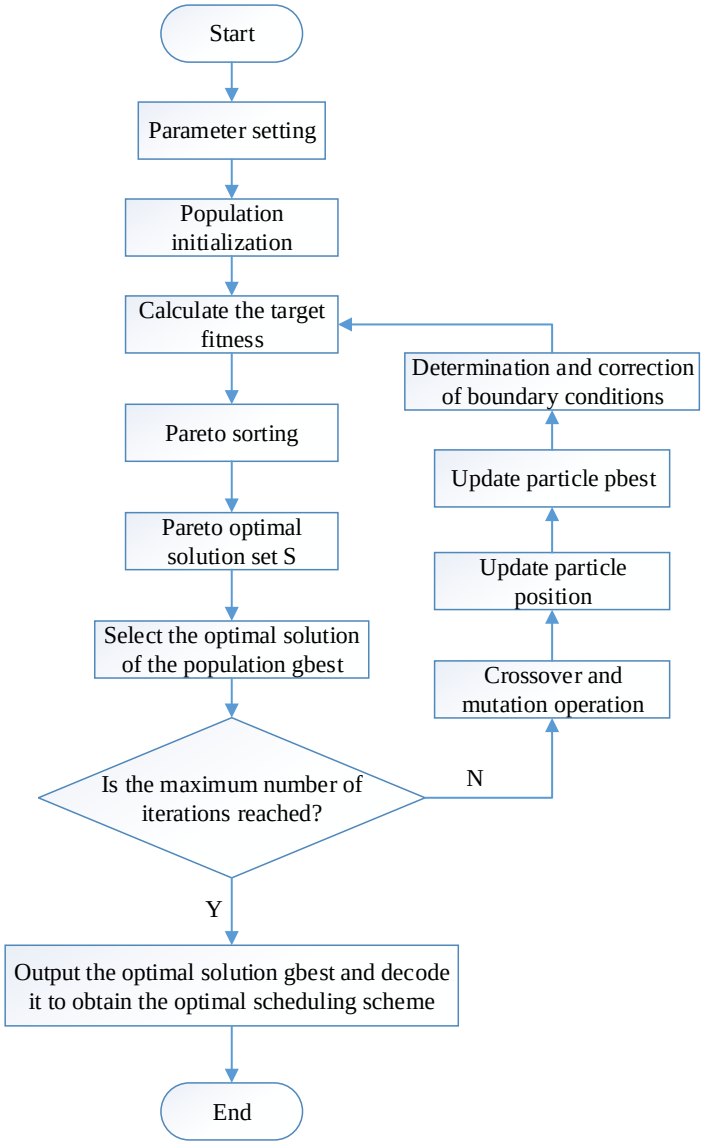

Figure 7. Flow chart of IDPSO algorithm solving multi-objective FJSP.

Step 1: Determine parameters. Set the population size, P, iteration times, t, adaptive inertia weight, w, and learning factors, c_1 and c_2.

Step 2: Population initialization. When each particle is initialized, the process sequencing part adopts the random initialization method, and the machine selection part adopts the mixed initialization method. The position, $\{X_1, X_2, X_3, \cdots, X_p\}$, of the initialized particle is obtained according to the OScode of each particle and the corresponding MScode.

Step 3: Calculate the fitness value of each particle. Calculate the fitness of the initial particle swarm according to the initial position of the initial particle swarm, and record the initial individual optimal value, $Pb_i^t = X_i$.

Step 4: The initial particle population is non-dominated, and the Pareto frontier set, $\boldsymbol{S}$, the global optimal particle *gbest* and global optimal value, Gb^t, are obtained.

Step 5: Update particle positions in accordance with Formula (20).

Step 6: Determine the boundary conditions.

Step 7: Solve the fitness of the current particle swarm, and update the optimal location of each particle in accordance with the *pbest* selection method.

Step 8: Sort the non-dominating current particle swarm, update the optimal set, $\boldsymbol{S}$, according to the sorting result, and select and update the optimal particle *gbest* and global optimal value, Gb^t, from the optimal set, $\boldsymbol{S}$.

Step 9: Determine and correct the convergence condition. When the number of iterations meets the set value, the algorithm ends. Otherwise, continue iteration optimization in step 3.

Step 10: At the end of the cycle, the optimal Pareto solution set ***S*** is obtained, and the global optimal particle *gbest* is output.

5. Simulation Experiment

In order to verify the effectiveness of the proposed algorithm, this paper first uses the international general FJSP example set, the Kacem dataset [28] and the Brandimarte dataset [29] to conduct simulation tests on the IDPSO algorithm; then, a flexible job shop example (AT01) with setup time and handling time is designed, and the IDPSO algorithm is applied to the AT01 example to solve the multi-objective flexible job shop scheduling problem with multi-variety, small-batch and multi-work centers. MATLAB R2021b is used to implement the above algorithm. The operating environment is i5-5200 CPU manufactured by Intel, headquartered in Santa Clara, CA, USA, with a main frequency of 2.20 GHz, and 12 GB memory.

5.1. Parameter Setting Experiment

Configuring reasonable parameters is crucial for the performance of the algorithm. The key parameters that affect the performance of the DPSO algorithm include the population size (*Popsize*), the range of adaptive inertia constants ($w_{\min}$ and $w_{\max}$) and the learning factors (c_1 and c_2). In this paper, the orthogonal experimental design method in reference [6] was used to determine the optimal combination of parameters, and these parameters were set to 5 factors, each of which had 4 levels. Orthogonal experiments based on the FJSP medium-scale test instance MK07 are conducted to adjust the parameters. Each set of parameter combinations is run 10 times, and the average *makespan* is calculated as the evaluation criterion. The orthogonal experiment table for L_{16} (4^5) and its experimental results are shown in Table 4.

Table 4. Table of orthogonal experimental results.

Test Number	Factor					*Makespan*
	Popsize	$w_{\max}$	$w_{\min}$	c_1	c_2	
1	100(1)	0.7(1)	0.2(1)	0.6(1)	0.7(1)	189.2
2	100(1)	0.8(2)	0.3(2)	0.7(2)	0.8(2)	163.1
3	100(1)	0.9(3)	0.4(3)	0.8(3)	0.9(3)	150.6
4	100(1)	1.0(4)	0.5(4)	0.9(4)	1.0(4)	174.8
5	200(2)	0.7(1)	0.3(2)	0.8(3)	1.0(4)	165.7
6	200(2)	0.8(2)	0.2(1)	0.9(4)	0.9(3)	179.1
7	200(2)	0.9(3)	0.5(4)	0.6(1)	0.8(2)	168
8	200(2)	1.0(4)	0.4(3)	0.7(2)	0.7(1)	166.2
9	300(3)	0.7(1)	0.4(3)	0.9(4)	0.8(2)	164.7
10	300(3)	0.8(2)	0.5(4)	0.8(3)	0.7(1)	170.8
11	300(3)	0.9(3)	0.2(1)	0.7(2)	1.0(4)	168.1
12	300(3)	1.0(4)	0.3(2)	0.6(1)	0.9(3)	165.3
13	400(4)	0.7(1)	0.5(4)	0.7(2)	0.9(3)	162.2
14	400(4)	0.8(2)	0.4(3)	0.6(1)	1.0(4)	177.7
15	400(4)	0.9(3)	0.3(2)	0.9(4)	0.7(1)	169.3
16	400(4)	1.0(4)	0.2(1)	0.8(3)	0.8(2)	183.4
k_1	169.425	170.45	179.95	175.05	173.875	
k_2	169.75	172.675	165.85	**164.9**	169.8	
k_3	**167.225**	**164**	**164.8**	167.625	**164.3**	
k_4	173.15	172.425	168.95	171.975	171.575	
R	5.925	8.675	15.15	10.15	9.575	

Note: The position in bold is the number of levels of the parameter.

As shown in Table 4, according to extreme values and ANOVA of parametric orthogonal experiments, the influence degree of five factors on algorithm performance is $w_{\min} > c_1 > c_2 > w_{\max} > Popsize$. The k_i (i = 1, 2, 3, 4) value represents the average of the four experimental results for this parameter at the corresponding i level. As the mean value of *makespan* is selected as the evaluation index, the smaller the mean value of *makespan*, the better the running result of the current example. Therefore, the smallest k_i value of the parameter corresponding to each column indicates that the i level parameter corresponding to this factor has a better value. As can be seen in Figure 8, the optimal parameters of the algorithm in this paper are set as $Popsize$ = 300, $w_{\max}$ = 0.9, $w_{\min}$ = 0.4 and c_1 = 0.7, c_2 = 0.9. Other parameters are set as follows: maximum number of iterations: T = 100.

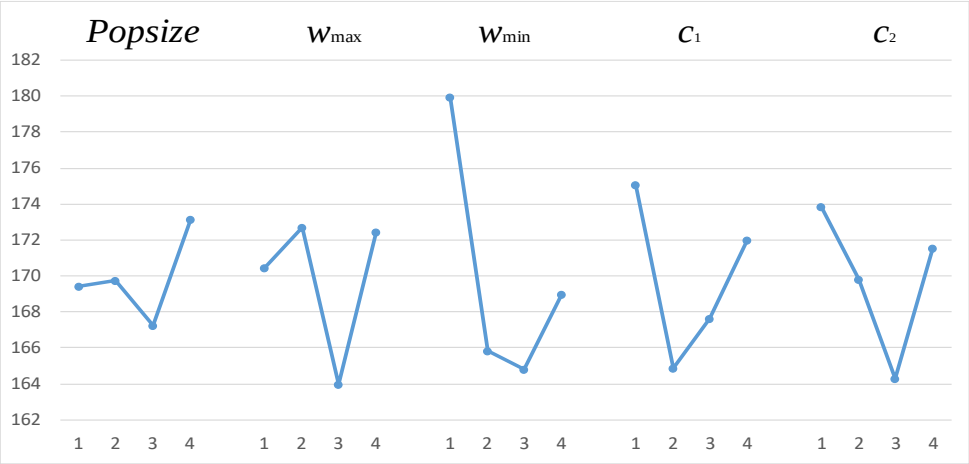

Figure 8. Parameter factor mean effect graph.

5.2. Performance Comparison Test

In order to enhance the stability and reliability of the experimental results, 30 repeated experiments were conducted to mitigate the influence of random factors and improve the accuracy and reproducibility of the experiment. Meanwhile, to verify the effectiveness of the proposed mathematical model, the IBM ILOG CPLEX 12.9.0 solver was utilized to solve exactly 15 FJSP test instances. During the solving process, the maximum computation time for CPLEX was set to 3600 s, and the exact solutions found within the specified time limit were used as reference solutions. The test results of the 15 instances are shown in Table 5, in which [30] the known optimal solution, B_{KS}, of each example is obtained from the literature, t_B is the optimal value of the completion time in the 20 solution experiment results of the example, t_a is the average value of the 20 solution experiment results of the example, and t_{av} is the average calculation solution time of the t_B value.

From Table 5, it can be observed that CPLEX is able to obtain known optimal solutions for small-scale instances (Kacem01–03; MK01–02). However, for larger instances, it fails to reach the optimal solution within the given time limit. Additionally, the algorithm in this paper can obtain the best result when the scale of the example is small. The optimal scheduling Gantt charts for Kacem01 and Kacem03 solved by the algorithm in this article are shown in Figures 9 and 10, respectively. Because this paper is based on a multi-objective mathematical model, when the scale of the example is large, the optimal result obtained is inferior to the optimal solution of the known single-objective model. Figure 11 shows the Gantt chart of the optimal *makespan* of 327 for the larger-scale instance, MK09, solved by the algorithm in this article. However, in all the examples, the average value of the optimal solution of 30 experiments can smoothly approach the optimal result, which shows the effectiveness and reliability of the IDPSO algorithm.

Table 5. Solution results of test example.

Example	Size	B_{KS}	Algorithm in This Paper			CPLEX	
			t_B	t_a	t_{av}/s	t_B	t_{av}/s
Kacem01	4 × 5	11	11	11	8.1	11	20.28
Kacem02	8 × 8	14	14	14	59.2	14	242.34
Kacem03	10 × 7	11	11	11	171.6	11	2468.32
Kacem04	10 × 10	7	7	7.5	258.6	8	3600
Kacem05	15 × 10	11	12	12.6	394.8	13	3600
MK01	10 × 6	40	40	40	178.8	40	2541.24
MK02	10 × 6	26	26	26	181.2	26	2677.51
MK03	15 × 8	204	204	206.2	333.6	243	3600
MK04	15 × 8	60	60	61.5	320.4	82	3600
MK05	15 × 4	172	174	175.1	352.2	195	3600
MK06	10 × 15	57	61	61.67	417.4	76	3600
MK07	20 × 5	139	146	147.2	517.8	184	3600
MK08	20 × 10	523	536	539.4	879.4	652	3600
MK09	20 × 10	307	327	332.7	957.6	378	3600
MK10	20 × 15	198	238	247.4	1177.2	307	3600

To further demonstrate the solution effectiveness of this algorithm, it is compared with the discrete particle swarm optimization (DPSO) algorithm proposed by Ding et al. [25], the efficient heuristic algorithm based on the construction process proposed by Ziaee [30], the improved discrete moth fire-fighting optimization (IDMFO) algorithm proposed by Tao et al. [31], and the hybrid gray wolf optimization (HGWO) algorithm proposed by Jiang [32]. We compare and analyze from the perspective of relative error, and calculate the relative error, e, of the t_B value and the B_{KS} value. The calculation formula is shown in Equation (22):

$$e = (t_B - B_{KS})/B_{KS} \times 100 \quad (22)$$

Then, for each algorithm experiment, we compute the average value MRE of the relative error, e, of the 15 FJSP test instances and count the number of times, N_B, that the optimal value was obtained.

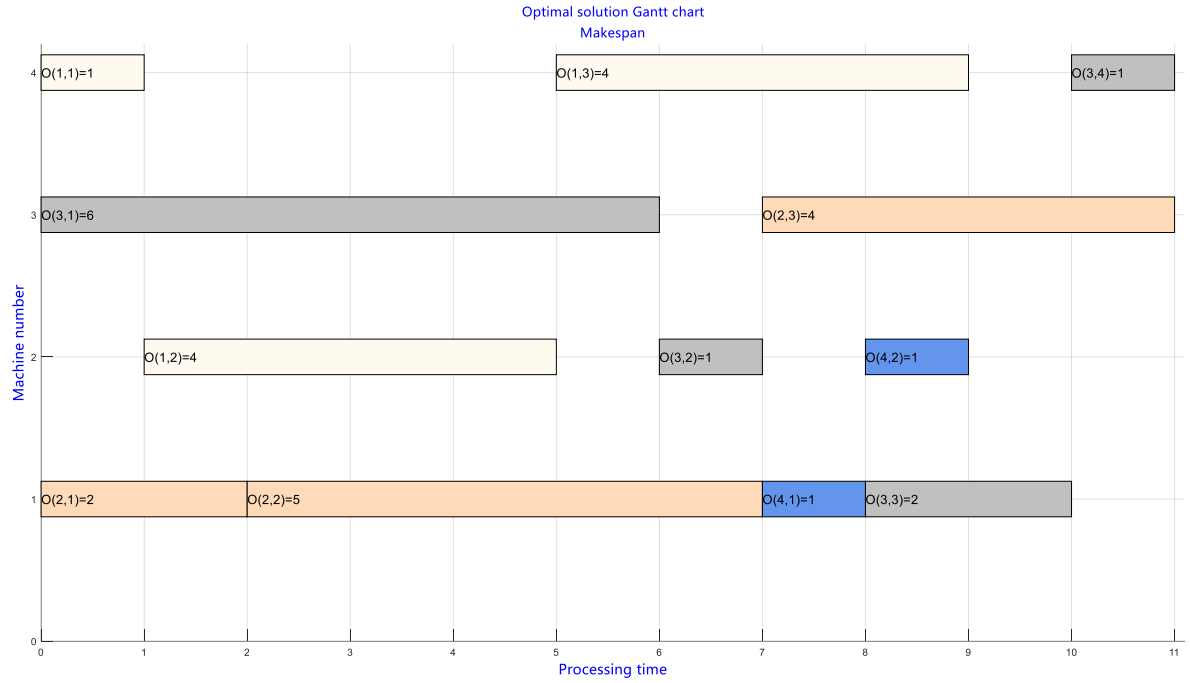

Figure 9. Kacem01 instance optimal scheduling Gantt chart.

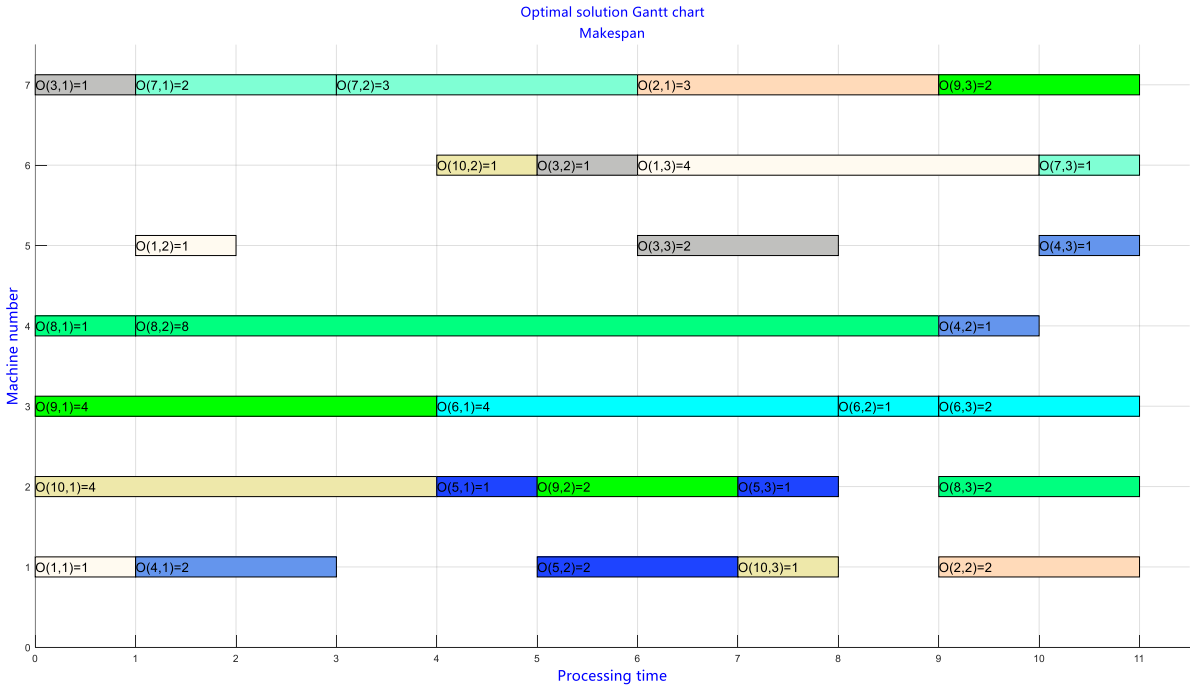

Figure 10. Kacem03 instance optimal scheduling Gantt chart.

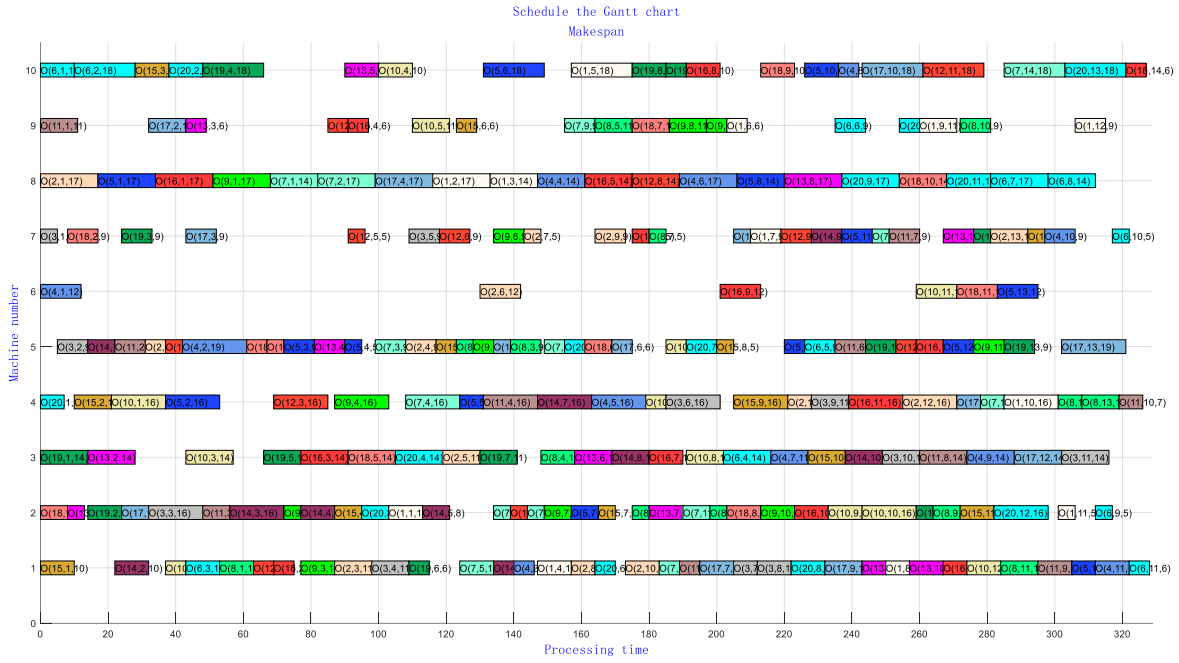

Figure 11. MK09 instance optimal scheduling Gantt chart.

The data in Table 6 show that, when compared with the other four algorithms from the literature, the average relative error, MRE, of this algorithm in 15 standard instances is 3.43, and the number of times the optimal value, N_B, is taken is 7. From the results, the relative error of the IDPSO algorithm is the smallest, and the number of known optimal solutions is the largest, which again proves the effectiveness and excellent stability of the algorithm in this paper.

Table 6. Algorithm performance comparison.

Example	Size	B_{KS}	Algorithm in This Paper		DPSO [25]	Heuristic [30]	IDMFO [31]	HGWO [32]
			t_B	e	t_B	t_B	t_B	t_B
Kacem01	4 × 5	11	**11**	0	**11**	**11**	-	**11**
Kacem02	8 × 8	14	**14**	0	15	15	15	**14**
Kacem03	10 × 7	11	**11**	0	12	13	-	**11**
Kacem04	10 × 10	7	**7**	0	**7**	**7**	**7**	**7**
Kacem05	15 × 10	11	12	9.09	11	12	13	13
MK01	10 × 6	40	**40**	0	42	42	**40**	**40**
MK02	10 × 6	26	**26**	0	32	28	27	29
MK03	15 × 8	204	**204**	0	**204**	**204**	**204**	**204**
MK04	15 × 8	60	**60**	0	80	75	62	65
MK05	15 × 4	172	174	1.16	173	179	174	175
MK06	10 × 15	57	61	7.02	66	69	67	79
MK07	20 × 5	139	146	5.04	145	149	144	149
MK08	20 × 10	523	535	2.48	524	555	523	523
MK09	20 × 10	307	328	6.52	364	342	314	325
MK10	20 × 15	198	238	20.2	252	242	235	253
MRE	-	-	-	**3.43**	**9.01**	**8.39**	**5.83**	**7.95**
N_B	-	-	**7**	-	**4**	**3**	**4**	**7**

Note: The bold is the preferred value for each instance run.

5.3. Example Application

In order to better fit the actual operation of the factory workshop, and further expand the time components of the production cycle on the basis of considering the parallel mobile operation mode, this paper designs a FJSP test example, AT01, containing processing time, setup time and handling time for the production system with multi-equipment work centers, facing the requirements of multi-variety and small-batch job processing task orders. The data of the AT01 example are shown in Tables 7–9.

Table 7. Job processing schedule of multi-equipment work centers.

Job Type	Job	Operation Number	OS	Machine and Processing Time/min					
				Work Center 1	Work Center 2		Work Center 3		Work Center 4
				WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}	WC_{41}
Type 1	Job 1	1	111	4	5	5	-	-	3
		2	112	3	-	-	2	2	4
Type 2	Job 1	1	211	-	3	3	4	4	-
		2	212	2	-	-	2	2	3
		3	213	4	3	3	-	-	5
	Job 2	1	221	-	3	3	4	4	-
		2	222	2	-	-	2	2	3
		3	223	4	3	3	-	-	5
Type 3	Job 1	1	311	-	4	4	3	3	4
		2	312	3	4	4	5	5	3
	Job 2	1	321	-	4	4	3	3	4
		2	322	3	4	4	5	5	3
	Job 3	1	331	-	4	4	3	3	4
		2	332	3	4	4	5	5	3

Table 7. *Cont.*

Job Type	Job	Operation Number	OS	Machine and Processing Time/min					
				Work Center 1	Work Center 2		Work Center 3		Work Center 4
				WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}	WC_{41}
Type 4	Job 1	1	411	8	5	8	-	-	3
		2	412	9	3	6	1	2	6
		3	413	7	-	-	5	4	9
	Job 2	1	421	8	5	8	-	-	3
		2	422	9	3	6	1	2	6
		3	423	7	-	-	5	4	9
	Job 3	1	431	8	5	8	-	-	3
		2	432	9	3	6	1	2	6
		3	433	7	-	-	5	4	9
Type 5	Job 1	1	511	6	2	5	4	1	2
		2	512	8	5	7	4	1	2
		3	513	9	6	2	4	5	-
	Job 2	1	521	6	2	5	4	1	2
		2	522	8	5	7	4	1	2
		3	523	9	6	2	4	5	-
	Job 3	1	531	6	2	5	4	1	2
		2	532	8	5	7	4	1	2
		3	533	9	6	2	4	5	-

Note: "-" indicates that the operation cannot be processed on the machine of the work center.

Table 8. The machine adjustment schedule of multi-equipment work centers.

Job Type	Job	Operation Number	OS	Machine and Processing Time/min					
				Work Center 1	Work Center 2		Work Center 3		Work Center 4
				WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}	WC_{41}
Type 1	Job 1	1	111	0.5	2	2	1	2	1
		2	112	1	3	3	2	1	1
Type 2	Job 1	1	211	3	2	1	2	1	1.5
		2	212	0.5	1	3	1	2	2
		3	213	1	1	3	1.5	2	2
	Job 2	1	221	3	2	1	2	1	1.5
		2	222	0.5	1	3	1	2	2
		3	223	1	1	3	1.5	2	2
Type 3	Job 1	1	311	1	2	1.5	3	2	3
		2	312	3	1	2	2	1	1
	Job 2	1	321	1	2	1.5	3	2	3
		2	322	3	1	2	2	1	1
	Job 3	1	331	1	2	1.5	3	2	3
		2	332	3	1	2	2	1	1

Table 8. *Cont.*

Job Type	Job	Operation Number	OS	Machine and Processing Time/min					
				Work Center 1	Work Center 2		Work Center 3		Work Center 4
				WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}	WC_{41}
Type 4	Job 1	1	411	1	1.5	1	3	2	2
		2	412	1	2	2	3	1	2
		3	413	1	1	1	1	3	2
	Job 2	1	421	1	1.5	1	3	2	2
		2	422	1	2	2	3	1	2
		3	423	1	1	1	1	3	2
	Job 3	1	431	1	1.5	1	3	2	2
		2	432	1	2	2	3	1	2
		3	433	1	1	1	1	3	2
Type 5	Job 1	1	511	1.5	3	2	1	3	2
		2	512	2	1	2	1	1	2
		3	513	3	1	1	2	2	2
	Job 2	1	521	1.5	3	2	1	3	2
		2	522	2	1	2	1	1	2
		3	523	3	1	1	2	2	2
	Job 3	1	531	1.5	3	2	1	3	2
		2	532	2	1	2	1	1	2
		3	533	3	1	1	2	2	2

Note: "-" indicates that the operation cannot be processed on the machine of the work center.

Table 9. Jobs handling schedules between different equipment in the production system with multi-equipment work centers.

Handling Time of Different Equipment in the Work Center/min							
Different Equipment in the Work Center		Work Center 1	Work Center 2		Work Center 3		Work Center 4
		WC_{11}	WC_{21}	WC_{22}	WC_{31}	WC_{32}	WC_{41}
Work Center 1	WC_{11}	0	1	1	1.5	1.5	2
Work Center 2	WC_{21}	1	0	0	0.5	0.5	1
	WC_{22}	1	0	0	0.5	0.5	1
Work Center 3	WC_{31}	1.5	0.5	0.5	0	0	1
	WC_{32}	1.5	0.5	0.5	0	0	1
Work Center 4	WC_{41}	2	1	1	1	1	0

As shown in Table 7, the designed calculation example includes 6 jobs of 3 types, which are processed on 6 equipment at 4 work centers. It is specified that different processes have different setup times on equipment at different work centers, as shown in Table 8. The handling time required for job processing at different work centers is different, as shown in Table 9.

For the calculation example designed in this paper, the scheduling and optimization model of processing and manufacturing equipment, handling equipment and other resources is studied and designed. The simulation analysis of the calculation example is carried out through MATLAB software, and the evolution process curve under each scheduling goal is obtained. The population evolution diagram of the AT01 calculation example is shown in Figure 12.

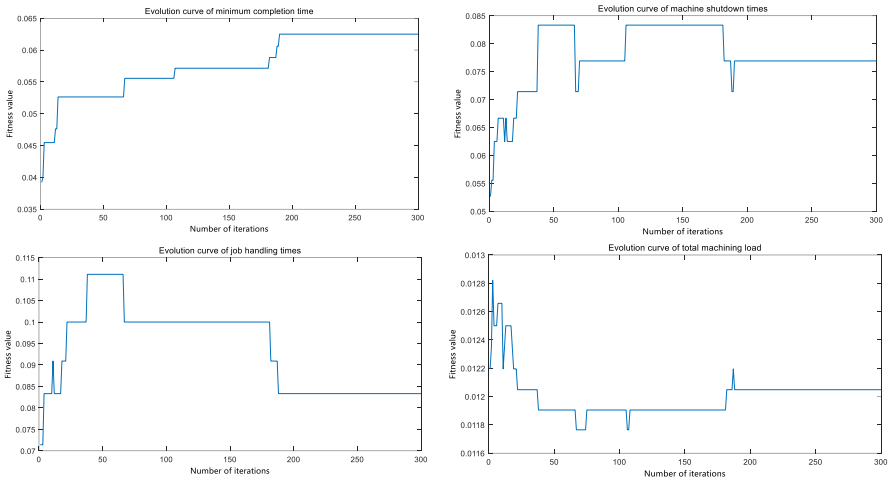

Figure 12. Evolutionary chart of *gbest* individual fitness based on Pareto selection strategy.

As shown in Figure 12, with the increasing number of iterations, each scheduling objective is continuously optimized, and has good convergence; that is, the maximum completion time, the amount of machine downtime, the number of job handling times and the total load of the machine are constantly optimized, and they are nearly stable after 190 iterations. According to the Pareto optimal solution set and *gbest* selection strategy in this paper, the job completion time is taken as the first priority of Pareto selection, and the fitness of other objectives evolves with the fitness of the job completion time. Therefore, it is reasonable that the fitness of other targets may decrease suddenly. The resulting optimal scheduling Gantt chart is shown in Figure 13, where the minimum maximum completion time is 16 min, the total number of machine adjustments is 7, the number of job handling times is 12 and the minimum total machine load is 83 min. In Figure 13, "O" in O (11, 1, 4) represents the processing procedure of the job, "11" represents the first piece of the first type of the job, "1" represents the first process of the job, and "4" represents the processing time of the processing procedure on the equipment; in A (21, 3, 1), "A" refers to the adjustment process of the job before the process to be processed, "21" refers to the first piece of the second job, "3" refers to the third process of the job, and "1" refers to the setup time of the processing process on the equipment; in T (21, 2, 1), "T" refers to the handling process of the job before the process to be processed, "21" refers to the first piece of the second type of the job, "2" refers to the second process of the job, and "1" refers to the job handling time on different work center equipment.

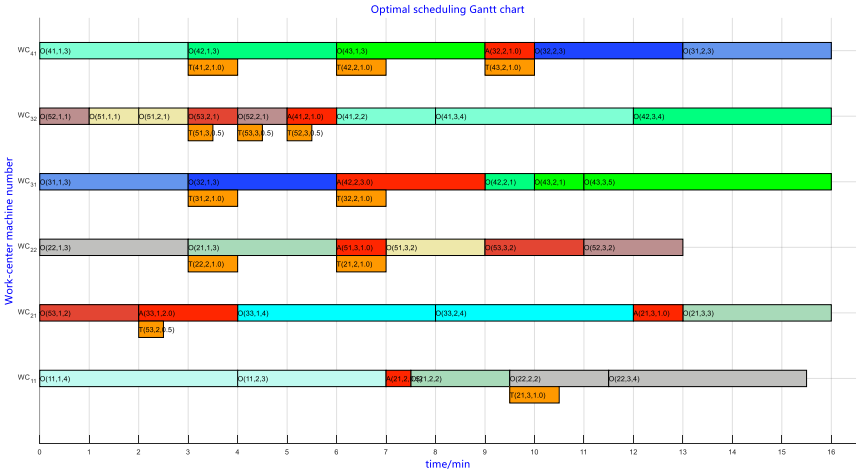

Figure 13. Gantt chart of optimal scheduling with the AT01 example.

6. Conclusions

In this paper, the multi-factor flexible job shop scheduling problem under parallel multi-equipment is studied, including machine setup time, job handling time and machine processing time, for multi-equipment production systems with multi-variety and small-batch job processing task orders. A multi-objective optimization model based on the maximum completion time, total number of machine adjustments, total number of workpiece handlings and total machine load was constructed. An improved discrete particle swarm optimization algorithm based on Pareto selection and an adaptive nonlinear inertia weight change strategy was proposed. Firstly, a hybrid initialization method combining process coding and machine coding was used to generate the initial population in the initialization phase to maintain the diversity of particles and improve the quality of the initial population. Second, an improved cross-variance operator was used to update the positions of the particles and promote good information sharing among the particles. Then, algorithm simulation experiments of different scales were conducted on Kacem and Brandimarte standard datasets and compared with other algorithms to verify the effectiveness of the IDPSO algorithm results. Finally, a multi-factor FJSP example including setup, handling and processing time is designed to verify the feasibility and effectiveness of the IDPSO algorithm in solving multi-objective FJSPs with good practical value through multiple simulations.

However, this study has three limitations. First, this paper studies the static job shop scheduling problem for multi-variety and small-batch processing tasks only. Second, the algorithm in this paper improves the local search capability of the algorithm by changing the adaptive nonlinear inertia weights. The optimal solution can be obtained when the instance size is small, but it can only be close to the known optimal solution in large-scale instances. Finally, in the algorithm process of selecting individual "*gbest*" particles in multi-objective decision-making, the fourth optimization objective, namely minimizing the total load of the machine, is chosen as a secondary target. The key to this objective lies in effectively allocating machine loads, reasonably distributing tasks, managing operation times and optimizing equipment utilization. However, in some actual production workshops, it may not be necessary to consider minimizing the total load of the machine because the complexity of the objective function may not warrant this goal. In subsequent research, the IDPSO algorithm can be combined with other local search algorithms to form a hybrid algorithm to further improve the local search capability of the algorithm and apply it to more complex job shop scheduling problems (such as dynamic planning and variable-batch problems). In addition, using green scheduling and production cost targets as the optimization objectives of this algorithm to actively respond to government policies of green and smart manufacturing in China can further enrich the application scope of the algorithm model.

Author Contributions: Conceptualization, Z.W. and J.K.; methodology, Z.W. and J.K.; validation, Z.W. and J.K.; formal analysis, Z.W.; investigation, Z.W.; resources, J.K.; writing—original draft preparation, Z.W. and J.K.; writing—review and editing, Z.W. and J.K.; supervision, J.K.; project administration, J.K. and Z.W.; funding acquisition, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Humanities and Social Science Youth foundation of the Ministry of Education of China, grant number 20YJC630054.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Garmdare, H.S.; Lotfi, M.M.; Honarvar, M. Integrated model for pricing, delivery time setting, and scheduling in make-to-order environments. *J. Ind. Eng. Int.* **2017**, *14*, 55–64. [CrossRef]
2. Li, Y.; Fadda, E.; Manerba, D.; Tadei, R.; Terzo, O. Reinforcement learning algorithms for online single-machine scheduling. In Proceedings of the 2020 15th Conference on Computer Science and Information Systems (FedCSIS), Sofia, Bulgaria, 6–9 September 2020.
3. Li, H.; Womer, K. Optimizing the supply chain configuration for make-to-order manufacturing. *Eur. J. Oper. Res.* **2012**, *221*, 118–128. [CrossRef]
4. Li, Y.; Huang, W.; Wu, R.; Guo, K. An improved artificial bee colony algorithm for solving multi-objective low-carbon flexible job shop scheduling problem. *Appl. Soft Comput.* **2020**, *95*, 106544–106558. [CrossRef]
5. Wei, H.; Li, S.; Quan, H.; Liu, D.; Rao, S.; Li, C.; Hu, J. Unified multi-objective genetic algorithm for energy efficient job shop scheduling. *IEEE Access* **2021**, *9*, 54542–54557. [CrossRef]
6. Zhang, G.; Yan, S.; Song, X.; Zhang, D.; Guo, S. Evolutionary algorithm incorporating reinforcement learning for energy-conscious flexible job-shop scheduling problem with transportation and setup times. *Eng. Appl. Artif. Intell.* **2024**, *133*, 107974. [CrossRef]
7. Pal, M.; Mittal, M.L.; Soni, G.; Chouhan, S.S.; Kumar, M. A multi-agent system for FJSP with setup and transportation times. *Expert Syst. Appl.* **2023**, *216*, 119474–119487. [CrossRef]
8. Zhang, H.; Xu, G.; Pan, R.; Ge, H. A novel heuristic method for the energy-efficient flexible job-shop scheduling problem with sequence-dependent set-up and transportation time. *Eng. Optim.* **2022**, *54*, 1646–1667. [CrossRef]
9. Li, J.; Deng, J.; Li, C.; Han, Y.Y.; Tian, J.; Zhang, B.; Wang, C.G. An improved Jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times. *Knowl.-Based Syst.* **2020**, *200*, 106032–106045. [CrossRef]
10. Kubiak, W.; Feng, Y.; Li, G.; Sethi, S.P.; Sriskandarajah, C. Efficient algorithms for flexible job shop scheduling with parallel machines. *Nav. Res. Logist.* **2020**, *67*, 272–288. [CrossRef]
11. Choi, I.C.; Choi, D.S. A local search algorithm for job shop scheduling problems with alternative operations and sequence -dependent setups. *Comput. Ind. Eng.* **2002**, *42*, 43–58. [CrossRef]
12. Dai, M.; Tang, D.; Giret, A.; Salido, M.A. Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robot. Comput.-Integr. Manuf.* **2019**, *59*, 143–157. [CrossRef]
13. Li, M.; Lei, D. An imperialist competitive algorithm with feedback for energy-efficient flexible job shop scheduling with transportation and sequence-dependent setup times. *Eng. Appl. Artif. Intell.* **2021**, *103*, 104307–104320. [CrossRef]
14. Feng, Y.J.; Kong, J.L. Multi-Objective Hybrid Flow Shop Scheduling in Parallel Sequential Mode While Considering Handling Time and Setup Time. *Appl. Sci.* **2023**, *13*, 3563. [CrossRef]
15. Defraeye, M.; Van Nieuwenhuyse, I. Staffing and scheduling under nonstationary demand for service: A literature review. *Omega* **2016**, *58*, 4–25. [CrossRef]
16. Thevenin, S.; Zufferey, N. Learning variable neighborhood search for a scheduling problem with time windows and rejections. *Discret. Appl. Math.* **2019**, *261*, 344–353. [CrossRef]
17. Han, R.; Li, J.; Xiao, X. Multi-Objective Artificial Bee Colony for Assembly Flexible Job Shop with Transportation and Setup Times. In Proceedings of the Genetic and EvolutionaryComputation Conference, Lisbon, Portugal, 15–19 July 2023.
18. Sun, J.; Zhang, G.; Lu, J.; Zhang, W. A hybrid many-objective evolutionary algorithm for flexible job-shop scheduling problem with transportation and setup times. *Comput. Oper. Res.* **2021**, *132*, 105263–105278. [CrossRef]
19. Rossi, A. Flexible job shop scheduling with sequence-dependent setup and transportation times by ant colony with reinforced pheromone relationships. *Int. J. Prod. Econ.* **2014**, *153*, 253–267. [CrossRef]
20. Ding, H.; Gu, X. Improved particle swarm optimization algorithm based novel encoding and decoding schemes for flexible job shop scheduling problem. *Comput. Oper. Res.* **2020**, *121*, 104951–104966. [CrossRef]
21. Qu, X.H.; Wang, J.; Ding, B.R.; Meng, G.J. Genetic algorithm of greedy initial population for flexible job shop scheduling. *J. Hefei Univ. Technol. (Nat. Sci. Ed.)* **2021**, *44*, 1153–1156+1171.
22. Zhang, G.; Hu, Y.; Sun, J.; Zhang, W. An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints. *Swarm Evol. Comput.* **2020**, *54*, 100664. [CrossRef]
23. Yuan, M.; Li, Y.; Zhang, L.; Pei, F. Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm. *Robot. Comput.-Integr. Manuf.* **2021**, *71*, 102141. [CrossRef]
24. Tang, H.; Xiao, Y.; Zhang, W.; Lei, D.; Wang, J.; Xu, T. A DQL-NSGA-III algorithm for solving the flexible job shop dynamic scheduling problem. *Expert Syst. Appl.* **2024**, *237*, 121723. [CrossRef]
25. Ding, S.Y.; Bing, L.I.; Shi, H.B. Study on Flexible Job-shop Scheduling Problem Based on Improved Discrete Particle Swarm Optimization Algorithm. *Comput. Sci.* **2018**, *45*, 233–239+256.
26. Shi, J.; Chen, M.; Ma, Y.; Qiao, F. A new boredom-aware dual-resource constrained flexible job shop scheduling problem using a two-stage multi-objective particle swarm optimization algorithm. *Inf. Sci.* **2023**, *643*, 119141–119170. [CrossRef]
27. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T.A.M.T. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
28. Kacem, I.; Hammadi, S.; Borne, P. Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2002**, *32*, 1–13. [CrossRef]
29. Brandimarte, P. Routing and scheduling in a flexible job shop by tabu search. *Ann. Oper. Res.* **1993**, *41*, 157–183. [CrossRef]

30. Ziaee, M. A heuristic algorithm for solving flexible job shop scheduling problem. *Int. J. Adv. Manuf. Technol.* **2014**, *71*, 519–528. [CrossRef]
31. Tao, T.T.; Song, Y.C.; Wang, J.C. Solving flexible job shop scheduling problem with improved discrete moth to put out fire optimization algorithm. *Mech. Eng.* **2020**, *11*, 25–29+33.
32. Jiang, T.H. Hybrid Grey Wolf Optimization Algorithm for Flexible Job Shop Scheduling Problem. *Control Decis. Mak.* **2018**, *33*, 503–508.