Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.079917

ARTICLE

Hybrid Flow Shop Rescheduling Approach Based on Hybrid-Driven Mechanism and Improved Multi-Objective WOA

Feng Lv*, Xin Xu, Cheng Yang and Yixuan Tang

School of Mechatronics Engineering, Henan University of Science and Technology, Luoyang, China

*Corresponding Author: Feng Lv. Email: lvfeng1980@haust.edu.cn

Received: 30 January 2026; Accepted: 09 April 2026; Published: 08 May 2026

ABSTRACT: To ensure an effective disturbance response and maintain continuous production in hybrid flow shops, this paper focuses on the design of a rescheduling method. A rescheduling model is constructed that minimizes the makespan, total tardiness, and scheme deviation degree. A hybrid rescheduling driving mechanism based on the latest completion time is designed to effectively trigger rescheduling. The Whale Optimization Algorithm (WOA) is improved by integrating the good point set theory, nonlinear control parameter strategy, and Differential Evolution (DE) algorithm. Moreover, non-dominated sorting and a dynamic external archive mechanism based on crowding distance are introduced to make it suitable for multi-objective optimization problems. The superiority of the Improved Multi-objective Whale Optimization Algorithm (IMOWOA) and the effectiveness of the improved mechanisms are verified through comparative experiments and ablation experiments. Taking the final assembly production line of an agricultural machinery equipment enterprise as an example, a rescheduling scheme is generated based on the practical production requirements, which verifies the feasibility and effectiveness of the proposed method.

KEYWORDS: Hybrid flow shop; production disturbance; production rescheduling; rescheduling driving mechanism; improved multi-objective whale optimization algorithm

1 Introduction

The production system, due to its inherent complexity, is prone to being affected by various disturbances such as order changes and equipment failures, often resulting in the failure of the initial scheduling plan and causing great pressure on manufacturing enterprises. Traditional workshops usually convert dynamic scheduling problems into static ones for handling. This passive response approach has limitations such as low resource utilization and insufficient response capability, often leading to production capacity loss and quality decline. Therefore, establishing and implementing effective rescheduling strategies is crucial for manufacturing enterprises to restore production stability and ensure order delivery.

The core of the rescheduling problem lies in the optimization and reconfiguration of production resources after disturbances in a dynamic environment. In recent years, the research on generating rescheduling schemes for workshops has mainly focused on two aspects: the rescheduling driving mechanism and the rescheduling solution methods.

The rescheduling driving mechanism serves to determine the initiation time of rescheduling when the system encounters disturbances. With deepening research, scholars have progressively shifted their focus toward the quantitative discrimination of production disturbances based on identifying the intrinsic characteristics of various driving mechanisms, thereby facilitating the construction of novel rescheduling

driving mechanisms. Zhou et al. [1] quantified dynamic production states and proposed dynamic scheduling decision-making methods to address machine idling and operational deviations caused by hidden disturbances. Lü et al. [2] designed a dynamic decision-making mechanism for disturbance events and established a multi-objective rescheduling model. Deng et al. [3] argued that quality control is the final stage of shop operations and proposed a dynamic decision-driven memetic algorithm for the order insertion problem. Zheng et al. [4] put forward a rescheduling driving method considering efficiency and stability indicators and demonstrated its advantages in reducing the number of rescheduling triggers and improving rescheduling performance.

Rescheduling solution methods refer to techniques that generate optimized rescheduling schemes meeting specified objectives. Current research predominantly employs metaheuristic algorithms to address optimization requirements. Mainstream approaches include genetic algorithms and swarm intelligence algorithms, among others. An et al. [5] addressed concurrent order insertion and equipment maintenance by developing an improved non-dominated sorting genetic algorithm with adaptive reference vectors (NSGA-ARV). Song et al. [6] put forward a rescheduling decision model based on the Whale Optimization Algorithm and Support Vector Machine (WOA-SVM), employing three rescheduling strategies—complete rescheduling, partial rescheduling, and right-shift rescheduling—for solution generation. Gao et al. [7] proposed an improved Jaya algorithm to address the flexible job shop rescheduling problem with machine failures. Fan et al. [8] presented an enhanced Tuna Swarm Optimization Algorithm to solve the rescheduling problem of flexible job shops with random machine failures. Ali et al. [9] utilized a hybrid genetic algorithm to manage rescheduling triggered by new job arrivals in dynamic scheduling environments. He et al. [10] constructed a model for the flexible job shop scheduling problem with dual resource constraints and solved it using an improved African Vulture Optimization Algorithm. Schworm et al. [11] proposed a multi-objective optimization algorithm based on quantum annealing to shorten delivery time while maintaining production continuity. Sun et al. [12] designed a hybrid framework integrating NSGA with tabu search, later advancing to an enhanced NSGA-III/ARV algorithm for rescheduling. Tang et al. [13] developed a multi-objective rescheduling model addressed by a refined imperialist competitive algorithm for order insertion scenarios. Zhang et al. [14] constructed a multi-objective mixed-integer programming model optimized through improved grey wolf optimization. Saophan et al. [15] designed a fast production scheduling framework for flow shops and proposed a disturbed population genetic algorithm to solve the rescheduling problem under machine failure conditions.

Existing research has extensively analyzed and applied workshop rescheduling under disturbances, providing valuable foundations for this paper. However, there remain areas for improvement in the research on rescheduling problems under disturbances. In terms of disturbance response, existing achievements mainly focus on the improvement, innovation of rescheduling optimization algorithms and the expansion of application scenarios, with less research on rescheduling driving mechanisms. Moreover, existing driving mechanisms are mostly limited to the quantitative discrimination of a single disturbance factor, and rarely consider the tolerance of the production process to disturbance degrees, resulting in insufficient flexibility in rescheduling triggering. Thus, it is necessary to establish an effective rescheduling driving mechanism. In terms of modeling and solving rescheduling problems, the applicability and efficiency of solutions can be improved based on enterprise preferences and production demands. Meanwhile, the solving performance of multi-objective optimization algorithms in rescheduling scenarios still has room for improvement. Therefore, the main contributions of this paper are as follows:

(1) Based on the actual situation in hybrid flow shops, a multi-objective rescheduling model is constructed, with the objectives of minimizing the makespan, total tardiness, and scheme deviation degree.

This model balances the production actual demands of enterprises, ensuring order delivery and production stability.

(2) A hybrid rescheduling driving mechanism based on the latest completion time is established to realize the effective triggering of rescheduling.

(3) An Improved Multi-Objective Whale Optimization Algorithm (IMOWOA) is proposed. Based on the original Whale Optimization Algorithm, improvements are made by integrating the good point set theory, nonlinear control parameter strategy, and Differential Evolution algorithm. In addition, non-dominated sorting and a dynamic external archive mechanism based on crowding distance are introduced to make it suitable for multi-objective optimization requirements. The good performance of the algorithm is verified through experimental simulations.

The remainder of this paper is structured as follows. Section 2 formulates the problem and constructs the mathematical model. Section 3 develops a hybrid rescheduling driving mechanism based on the latest completion time. Section 4 details the design of an improved multi-objective whale optimization algorithm (IMOWOA). Section 5 presents experiments and case analysis, evaluates the algorithm's performance using benchmark test functions, and verifies the method's effectiveness through practical applications. Finally, Section 6 concludes the paper.

2 Problem Description and Modeling

2.1 Problem Description

This paper focuses on the scheduling problems in hybrid flow shops featuring related parallel machines with skippable operations. The problem can be described as follows: The shop comprises V processing operations, where each operation v is equipped with Q_v parallel machines. There are U batches of jobs to be processed, which need to go through some or all of the processing operations in a known technological sequence. The technological paths of different jobs may vary, and at least one processing operation has two or more machines. Machines within the same processing operation are related parallel machines. Each job is produced according to the initial scheduling scheme. At a certain moment, a disturbance is detected. Through analyzing the disturbance, it is determined whether to trigger rescheduling, and the optimal rescheduling scheme is sought.

Considering the complexity of the rescheduling problem, the following assumptions are made in this paper:

(1) The process sequence of each job is fixed, and some processing operations can be skipped according to process requirements.

(2) There are no direct sequential relationships between operations of different jobs.

(3) Each batch of jobs in the same processing operation can only be processed by one machine, and once processing starts, it is not allowed to be interrupted (except for equipment failures).

(4) For any job on the same machine, processing can only start after the previous job has been completely processed.

(5) Processing priorities are the same between the operations of different jobs.

(6) For a batch of jobs in a specific processing operation, the processing time is the same across different machines.

(7) The same batch of jobs cannot be divided into multiple sub-batches, and jobs of different tasks cannot be processed in a mixed manner.

(8) The movement time of tasks is not considered, and the setup times for each operation are included in the processing time.

(9) Disturbances occur randomly.

2.2 Modeling

2.2.1 Parameters

The relevant parameter symbols and their meanings in the model are shown in Table 1.

Table 1: Meaning of the symbol.

Symbol	Meaning
U	Set of jobs
u	Index of job, where $u = 1, 2, \ldots, N$
Vu	Set of processing operations for job u
v	Index of processing operation, where $v = 1, 2, \ldots, V$ V is the total number of operations
Qv	Set of available machines for processing operation v
q	Index of machine, where $q = 1, 2, \ldots, Q$ Q is the total number of machines
C_u	Completion time of job u in the initial scheduling
C_u^r	Completion time of job u after rescheduling
Du	Delivery date of job u
$\mu_{u,v,q}$	0–1 variable: 1 if job u is assigned to machine q for operation v in the initial scheduling, 0 otherwise
$\mu_{u,v,q}^r$	0–1 variable: 1 if job u is assigned to machine q for operation v after rescheduling, 0 otherwise
$PT_{u,v}$	Processing time of job u at operation v
$BT_{u,v}$	Start time of job u at operation v in the initial scheduling
$FT_{u,v}$	Finish time of job u at operation v in the initial scheduling
$BT_{u,v,q}^r$	Start time of job u on machine q at operation v after rescheduling
$FT_{u,v,q}^r$	Finish time of job u on machine q at operation v after rescheduling
MR_q^r	Release time of machine q after rescheduling
AR_u^r	Release time of job u after rescheduling
$\eta_{u,v}$	0–1 variable: 1 if job u goes through operation v ($v \in Vu$), 0 otherwise
NG	Set of jobs to be rescheduled

2.2.2 Rescheduling Model

The multi-objective mathematical model constructed in this paper is as follows:

(1) Makespan

The Makespan refers to the completion time of the last operation among all jobs in the scheduling process, which is a key indicator to measure the overall production efficiency or project progress. Its calculation formula is as follows:

$$f_1 = \max\left(C_u{}^r\right) \tag{1}$$

(2) Total tardiness

Total tardiness of jobs represents the sum of delays in completion times of each job in the rescheduling scheme compared to their due dates in the initial schedule. It reflects the degree of adjustment of the rescheduling scheme to the production plan and the degree of satisfaction with order owing dates. Its calculation formula is as follows:

$$f_2 = \sum_{u=1}^{N} \max(0, C_u^r - D_u) \tag{2}$$

(3) Scheme deviation degree

The scheme deviation degree refers to the deviation in machine allocation between the rescheduling and the initial scheduling schemes, reflecting the degree of change in the scheduling scheme. Its calculation formula is as follows:

$$f_3 = \sum_{u=1}^{N}\sum_{v=1}^{V}\sum_{q=1}^{Q}\left|\mu_{u,v,q} - \mu_{u,v,q}^{r}\right| \tag{3}$$

Based on the above, the optimization objectives are to minimize the makespan, minimize the total tardiness, and minimize the scheme deviation degree, as shown in the objective function of Eq. (4).

$$F = (\min f_1, \min f_2, \min f_3) \tag{4}$$

2.3 Constraints

Based on the characteristics of the hybrid flow shops, the following constraints are established as Eqs. (5)–(13).

(1) Each job can only select one machine for processing in any required operation.

$$\sum_{q=1}^{Q}\mu_{u,v,q} = 1, \forall u \in U, v \in V_u \tag{5}$$

(2) Any machine can only process one job at a time.

$$\sum_{u=1}^{N}\sum_{v=1}^{V}\mu_{u,v,q} = 1, \forall q \in Q_v \tag{6}$$

(3) A job can only be transferred to the next operation after completing processing in the current operation and must follow the technological process.

$$BT_{u,v+1} \geq FT_{u,v}, \ \forall u \in U, \ v \in V_u - 1 \tag{7}$$

(4) The finish time of a job at a certain operation is equal to the sum of its start time and processing time at that operation.

$$FT_{u,v} = BT_{u,v} + PT_{u,v}, \ \forall u \in U, \ v \in V_u \tag{8}$$

(5) The relationship between the finish time of job u at a certain operation and its finish time at the previous operation.

$$\begin{aligned} &FT_{u,v} = FT_{u,v-1} + PT_{u,v-1} \times \eta_{u,v} \\ &\forall u \in U, v \in V_u, v-1 \in V_u \end{aligned} \tag{9}$$

(6) The completion time of a job is equal to its completion time at the last operation V.

$$C_u = FT_{u,V}, \ \forall u \in U \tag{10}$$

(7) Non-negativity constraint on processing time: the processing time of a job at a certain operation is greater than zero when the job is processed at that operation; otherwise, it is equal to zero.

$$PT_{u,v} \geq 0, \ \forall u \in U, \ v \in V_u \tag{11}$$

(8) After rescheduling, the start time of any job processed by each machine must be greater than or equal to the release time of machine.

$$BT^{r}_{u,v,q} \geq MR^{r}_{q}, \forall u \in NG,\ v \in V_u,\ q \in Q_v \tag{12}$$

(9) In the rescheduling scheme, the start time of a job must be greater than or equal to the release time of job.

$$BT^{r}_{u,v,q} \geq AR^{r}_{u}, \forall u \in NG,\ v \in V_u,\ q \in Q_v \tag{13}$$

3 Rescheduling Driving Mechanism

3.1 Hybrid Rescheduling Driving Mechanism Based on the Latest Completion Time

The core function of the rescheduling driving mechanism is to accurately determine the scheduling timing, and its rationality directly affects production efficiency and plan stability. Existing studies have identified some driving mechanisms for rescheduling, including period-driven, event-driven, and hybrid-driven [16].

The event-driven mechanism can respond quickly to sudden disturbances, but it is overly sensitive to minor disturbances, resulting in frequent rescheduling and increased costs. The cycle-driven mechanism triggers at fixed intervals, providing good planning stability, but it has a delayed response to major disturbances within the interval and may miss the optimal adjustment opportunity. The hybrid-driven mechanism combines the two, aiming to balance production flexibility and stability. However, it needs to make decisions at each cycle node or when a disturbance occurs, and the system's computational load is large. Frequent triggering may still affect stability. Therefore, this paper proposes a hybrid rescheduling-driven mechanism based on the latest completion time of the process. Its process is shown in Fig. 1. This mechanism uses the latest completion time as a quantitative benchmark and analyzes the disturbance under the cycle-triggering condition: if the disturbance causes the completion time of the process to exceed its latest completion time, it immediately triggers rescheduling; otherwise, it continues to execute the original plan.

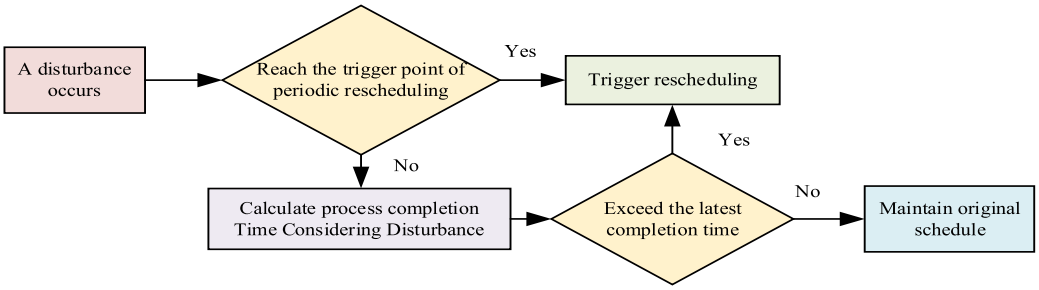


Figure 1: Flowchart of the rescheduling driving process.

3.2 Model of the Latest Completion Time

Each processing operation of a job involves two dimensions: job and machine. The minimum value of the delayed completion time of the operation in the job dimension and machine dimension is determined. Combined with the dynamic correlation between operations, the allowable delay time of each operation is calculated, and the corresponding latest completion time of each operation is determined, which provides a basis for the production system to respond to disturbance events promptly.

$$\sigma_{u,v} = \begin{cases} 1, & \exists O_{u,v+1} \\ 0, & \nexists O_{u,v+1} \end{cases} \forall u \in U, v \in V_u \tag{14}$$

$$\partial_{u,v} = \begin{cases} 1, & \exists O_{u',v'} \\ 0, & \nexists O_{u',v'} \end{cases} \forall u' \in U, v' \in V_u \tag{15}$$

$$\begin{aligned} Y^A_{u,v} &= \sigma_{u,v} \times \left(Y_{u,v+1} + BT_{u,v+1} - FT_{u,v}\right) \\ &\quad + \left(1 - \sigma_{u,v}\right) \times \left(D_u - FT_{u,v}\right) \\ &\quad u \in U, v \in V_u \end{aligned} \tag{16}$$

$$\begin{aligned} Y^M_{u,v} &= \partial_{u,v} \times \left(Y_{u',v'} + BT_{u',v'} - FT_{u,v}\right) \\ &\quad + \left(1 - \partial_{u,v}\right) \times \left(D_u - FT_{u,v}\right) \\ &\quad u \in U, v \in V_u \end{aligned} \tag{17}$$

$$Y_{u,v} = min\left\{Y^A_{u,v}, Y^M_{u,v}\right\} \tag{18}$$

$$W_{u,v} = FT_{u,v} + Y_{u,v} \tag{19}$$

$Q_{u,v}$ denotes operation v of job u, and $Y_{u,v}$ denotes the allowable delay time of operation $Q_{u,v}$. U' and v' denote other jobs and operations, respectively, where $u \neq u'$ and $v \neq v'$. In Eq. (14), $\sigma_{u,v}$ is a 0–1 variable; $\sigma_{u,v} = 1$ if the next operation $O_{u,v} + 1$ of operation $O_{u,v}$ in the job dimension exists, otherwise $\sigma_{u,v} = 0$. In Eq. (15), $\partial_{u,v}$ is a 0–1 variable; $\partial_{u,v} = 1$ if the next operation $O_{u',v'}$ of operation $O_{u,v}$ in the machine dimension exists, otherwise $\partial_{u,v} = 0$. In Eqs. (16) and (17), ${Y^A}_{u,v}$ and ${Y^M}_{u,v}$ denote the allowable delay times of operation $O_{u,v}$ in the job dimension and machine dimension, respectively, and different situations are defined using 0–1 variables σ and ∂. In Eq. (18), $Y_{u,v}$ denotes the allowable delay time of operation $O_{u,v}$ which is the minimum value of ${Y^A}_{u,v}$ and ${Y^M}_{u,v}$. In Eq. (19), $W_{u,v}$ denotes the latest completion time of operation $O_{u,v}$.

The allowable delay time $Y_{u,v}$ of operation $O_{u,v}$ can only be calculated when the allowable delay times $Y_{u,v} + 1$ and $Y_{u',v'}$ of the next operations in its two dimensions are known. Therefore, calculating the allowable delay times of all operations is a circular reverse traversal process.

The core of this mechanism is to construct a calculation model for the latest completion time of operations based on processing time fluctuations caused by disturbances. The critical time threshold affecting production continuity is dynamically calculated and determined by integrating the maximum tolerable delay times of each operation in the job and machine dimensions. This mechanism can ensure immediate responses to key disturbances affecting the overall production. Furthermore, it can prevent invalid responses and excessive adjustments to non-critical fluctuations. Consequently, this can result in a reduction in response costs.

4 Algorithm Design

4.1 Encoding and Decoding

This paper adopts a two-stage encoding method, which consists of two parts: Machine Selection (MS) and Operation sequencing (OS), forming a complete encoding. The MS part represents the processing machines of operations, and the OS part represents the processing sequence between operations. Both parts are represented by vectors with a length equal to the total number of operations. The vector elements in the MS part represent the serial numbers of the processing machines selected for the corresponding operations. The OS part adopts an operation-based encoding method, where each operation is represented by the corresponding job number in the encoding. If a job has V_u operations, it will appear V_u times in the OS part. During decoding, the OS part is compiled from left to right. The number of occurrences of a job number represents the sequence number of the corresponding operation of that job, and the order of appearance of operations in the OS part represents their processing priority in the scheduling scheme. Taking the encoding of 3 jobs with 3 operations as an example, as shown in Fig. 2.

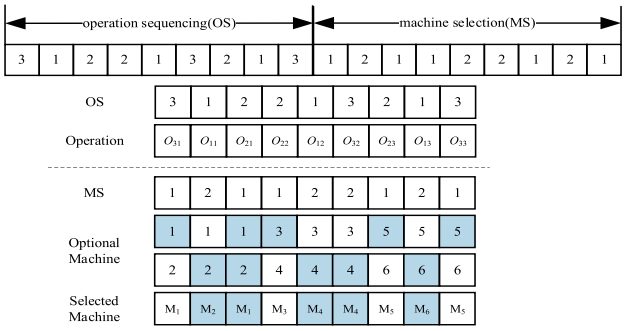


Figure 2: Two-stage encoding example.

As can be seen from the figure, the operation code is $\{3, 1, 2, 2, 1, 3, 2, 1, 3\}$, and the machine code is $\{1, 2, 1, 1, 2, 2, 1, 2, 1\}$, which represent the operation sequence as $\{O_{3,1}, O_{1,1}, O_{2,1}, O_{2,2}, O_{1,2}, O_{3,2}, O_{2,3}, O_{1,3}, O_{3,3}\}$, respectively. The machine numbers corresponding to each operation are $\{M_1, M_2, M_1, M_3, M_4, M_4, M_5, M_6, M_5\}$.

During decoding, the processing information of each job is read from the OS and MS parts, and two issues need to be considered simultaneously: processing sequence and machine selection. According to the encoding method, the genes in the operation sequencing part are read from left to right to determine the order of processing for all operations. In the first stage, the job processing sequence and machine allocation are carried out following the encoding. In the subsequent stages, the processing sequence of jobs in that stage is arranged based on the principle of minimizing the completion time of the shortest processing time in the stage. Here, machines are divided into two categories:

(1) If there are machines not assigned to any job, the machines are allocated according to the job processing sequence.
(2) If all machines have been assigned jobs, the remaining jobs are allocated to the machine with the earliest processing completion time according to the processing information of each job, and this process is repeated until all operations are scheduled.

4.2 IMOWOA

The Whale Optimization Algorithm (WOA) was proposed by Mirjalili and Lewis [17] and is based on the behavior of whales encircling prey. The algorithm has a simple structure and is easy to implement; compared with other algorithms, its simple yet powerful search mechanism can quickly find the optimal solution [18,19].

WOA demonstrates notable adaptability in addressing hybrid flow production rescheduling challenges as a fundamental algorithm. The hybrid flow scheduling studied in this paper involves two sub-problems: The key concepts here are “machine selection” and “operation sequencing”, which can essentially be viewed as discrete combinatorial optimization problems. WOA can flexibly handle the search in discrete solution spaces by simulating behaviors such as “shrinking encirclement” and “spiral updating”. Its mechanism for balancing global exploration and local exploitation can cover machine allocation schemes and optimize operation sequences. In contrast, WOA has a limited number of parameters and requires minimal implementation complexity. In comparison with genetic algorithms and other methods, the linear complexity of WOA renders it particularly well-suited to large-scale operation scheduling. This capability ensures the generation of feasible solutions promptly, thereby facilitating rapid responses to disturbances on the shop floor.

However, WOA also has problems such as being prone to falling into local optima, parameter sensitivity, and limited global exploration capability. This paper further improves its performance and application range by introducing a good point set, adopting nonlinear control parameters, and integrating the differential

evolution algorithm. In addition, the fitness evaluation mechanism of the WOA algorithm is difficult to be directly applied to multi-objective problems. Therefore, on the basis of improving WOA, a multi-objective optimization mechanism is introduced, and an Improved Multi-objective Whale Optimization Algorithm (IMOWOA) suitable for multi-objective optimization problems is proposed by combining non-dominated sorting and a dynamic external archive mechanism based on crowding distance.

4.2.1 Good Point Set

The good point set is an effective experimental method that can uniformly select points. To address the issues of uneven population distribution and insufficient coverage of the solution space caused by random initialization in the original WOA, a good point set is introduced to generate the initial population, making the population distribution more uniform, covering a wider search space, and preventing the algorithm from falling into local optima due to low-quality starting points.

The good point set theory was proposed by Chinese mathematician Loo-Keng Hua and others and can be used for approximate calculations in high-dimensional spaces. Its basic definition is as follows:

Let be a unit cube in the δ-dimensional Euclidean space; assuming, then the good point set can be expressed as:

$$P_g(y) = \left\{\left(\left\{h_1^{(g)} \cdot y\right\}, \left\{h_2^{(g)} \cdot y\right\}, \cdots, \left\{h_\delta^{(g)} \cdot y\right\}\right), 1 \le y \le g\right\}$$

Its discrepancy $\varphi(g)$ satisfies: $\varphi(g) = C(h,\varepsilon)\, g^{(-1+\varepsilon)}$

$$h = \left[2\cos\left(\frac{2\pi y}{\vartheta}\right)\right], 1 \le y \le g, \frac{(\vartheta - 3)}{2} \ge \delta \tag{20}$$

where $P_g(y)$ is the good point set, h is the good point, $C(h,\varepsilon)$ is a constant dependent only on h and ε ($\varepsilon > 0$), $\left\{h_\delta^{(g)} \cdot y\right\}$ denotes the fractional part, g is the number of points and ϑ is the smallest prime number satisfying $(\vartheta - 3)/2 \ge \delta$. Map it to the search space, and the expression is shown in Eq. (21).

$$f_z(y) = \left(z^{ub} - z^{lb}\right)\left[h_z^{(g)} \cdot y\right] + z^{lb} \tag{21}$$

where z^{ub} and z^{lb} are the upper and lower bounds of the z-th dimension.

4.2.2 Nonlinear Control Parameters

WOA mainly regulates the balance between global search and local exploitation by controlling the value of the control coefficient vector $\vec{A}$, with the control parameter a linearly decreasing from 2 to 0 [20]. However, this simple linear parameter decreasing strategy fails to help the algorithm effectively adjust the global search process. Specifically, it causes the value of a to decrease too rapidly in the early iteration stage, which narrows the scope of global exploration and may lead to missing the optimal solution; in the late iteration stage, the excessively slow decrease of a reduces the precision of local exploitation, resulting in a lag in convergence speed. To address this, this paper proposes a two-stage nonlinear convergence factor strategy, as shown in Eq. (22).

$$a = \begin{cases} 2 - 0.7\dfrac{t}{t_{\max}}, & 0 \le t \le \dfrac{t_{\max}}{3} \\ \left(2 - \dfrac{7}{30} \times \dfrac{t}{t_{\max}}\right) \times e^{-10\left(\frac{t}{T} - \frac{1}{3}\right)}, & \dfrac{t_{\max}}{3} \le t \le t_{\max} \end{cases} \tag{22}$$

This strategy ensures that the value of a remains at a relatively high level in the early iteration stage, decreasing at a relatively slow rate. The algorithm focuses on global optimization. In the middle and late iteration stages, the value of a decreases rapidly, accelerating the convergence speed and meeting the precision of local search.

4.2.3 Differential Evolution Algorithm

Differential Evolution (DE) was first proposed by Storn and Price in 1995, which is a population-based adaptive global optimization algorithm [21]. The spiral update mechanism of WOA relies on the distance between individuals and the optimal solution, which easily leads to the loss of population diversity. Especially in the late iteration stage, individuals excessively gather around the local optimal solution, resulting in falling into a local optimum. To remedy the problem of insufficient population diversity in WOA, this paper adopts the DE/best/1 mutation strategy of the differential evolution algorithm to generate new individuals [22]. The mutation operation expands the search range by selecting better whale individuals in the current population for mutation.

For each target individual (i = 1, 2, …, N), mutation is performed using the optimal individual of the current generation and 2 randomly selected distinct whale individuals in the population, as shown in Eq. (23).

$$V_{i,t} = X_{best,t} + \chi \times \left(X_{i1,t} - X_{i2,t}\right) \tag{23}$$

where $V_{i,t}$ is the intermediate individual generated after mutation; t is the current iteration number; i is the current individual; i_1 and i_2 are random mutually exclusive integers in the range [1, N] and not equal to i, where N is the population size; χ is the scaling factor, a random number in the range [0, 2], which is used to control the possible impact of the difference vector in the mutation strategy on the population state and directly affects the degree of expansion or contraction of the difference vector in the above equation.

4.2.4 Non-Dominated Sorting

The original WOA is used for single-objective optimization, and its fitness evaluation mechanism cannot directly handle the multi-objective rescheduling model. Therefore, it is necessary to introduce a non-dominated sorting mechanism, which realizes the hierarchical screening of solution sets through Pareto dominance relations, providing a clear direction for population evolution.

In multi-objective optimization problems, if solution 1 is not inferior to solution 2 in all objective functions and is superior to solution 2 in at least one objective function, then solution 1 is said to dominate solution 2. Non-dominated sorting is a method of stratifying solution sets based on Pareto dominance relations. Its core logic is to compare the dominance relations between solutions; the solution set is divided into multiple non-dominated layers. The earlier the layer, the higher the quality of the solutions, which effectively guides the algorithm to search for high-quality solution regions. The specific steps are as follows: Firstly, record the number of solutions dominated by each solution in the population and the number of solutions that dominate each solution; classify the solutions with a count of 0 as the first non-dominated layer; then, reduce the count by 1 for the individuals dominated by the solutions in the first layer; ignore the

already stratified solutions and classify the remaining solutions with a count of 0 into the next layer; repeat this process until all solutions are stratified.

4.2.5 Dynamic External Archive Mechanism Based on Crowding Distance

After non-dominated sorting, non-dominated solutions at the same level may show a clustered distribution in the objective space, causing the algorithm to fall into local optima. Further screening is needed to improve the quality of the solution set. Crowding distance measures the distribution density of solutions by calculating the neighborhood spacing of individuals in each objective space, as shown in Eq. (24).

$$d_i = \sum_{z=1}^{\delta} \frac{|f_z(i+1) - f_z(i-1)|}{f_z^{\max} - f_z^{\min}} \tag{24}$$

where d_i is the crowding distance of the i-th individual; the crowding distance of individuals at the edge of the layer is infinity; $f_z(i+1)$ and $f_z(i-1)$ are the values of the two individuals adjacent to individual i in the z-th objective function; $f_z^{\max}$ and $f_z^{\min}$ are the maximum and minimum values of individuals in the population for the z-th objective function.

On this basis, a dynamic external archive mechanism is established to store non-dominated solutions generated during iterations. By dynamically maintaining the sparse distribution characteristics of solutions, it effectively balances the algorithm's convergence and the diversity of the solution set. The specific implementation steps of the dynamic external archive mechanism based on crowding distance are as follows:

Step 1. Initialize an empty external archive and set its maximum capacity.

Step 2. Perform non-dominated sorting on the current population.

Step 3. Starting from the first non-dominated layer, sequentially place all solutions in each layer into the external archive.

Step 4. If the number of solutions in the external archive exceeds the maximum capacity after adding a certain layer, calculate the crowding distance for individuals in that layer.

Step 5. Sort individuals by the size of their crowding distance and delete the individual with the smallest crowding distance.

Step 6. Recalculate the dynamic crowding distance of the remaining individuals. Repeat Step 5 until the number of non-dominated solution individuals reaches the maximum capacity.

Therefore, the IMOWOA method proposed in this paper retains the simple structure advantages of WOA while making targeted designs in three core aspects: initial population optimization, search strategy improvement, and multi-objective solution set screening, in order to overcome its limitations in the application of complex rescheduling problems. The flowchart of the obtained IMOWOA algorithm is shown in Fig. 3.

5 Experiments and Case Analysis

5.1 Algorithm Performance Analysis

5.1.1 Comparison Experiment

To evaluate the performance of the proposed IMOWOA, simulation tests were conducted using benchmark functions from the Zitzler-Deb-Thiele (ZDT) and Deb-Thiele-Laumanns-Zitzler (DTLZ) series [23–25]. These function series encompass diverse scenarios, including continuous or discontinuous, linear, convex, or concave Pareto fronts, and are among the most widely utilized multi-objective benchmark

sets. Solving these standard functions provides a clear demonstration of performance differences between algorithms. In this study, the number of objectives for the DTLZ functions was set to 3, consistent with the settings used herein. Inverted Generational Distance (IGD) and Hypervolume (HV) were employed as metrics to assess the quality of the Pareto fronts obtained by the compared algorithms. The algorithm proposed in this paper is implemented in MATLAB R2022b and runs on the Windows 11 operating system, which is equipped with an Intel (R) Core (TM) i9-13900HX processor (2.20 GHz) and 16 GB RAM.

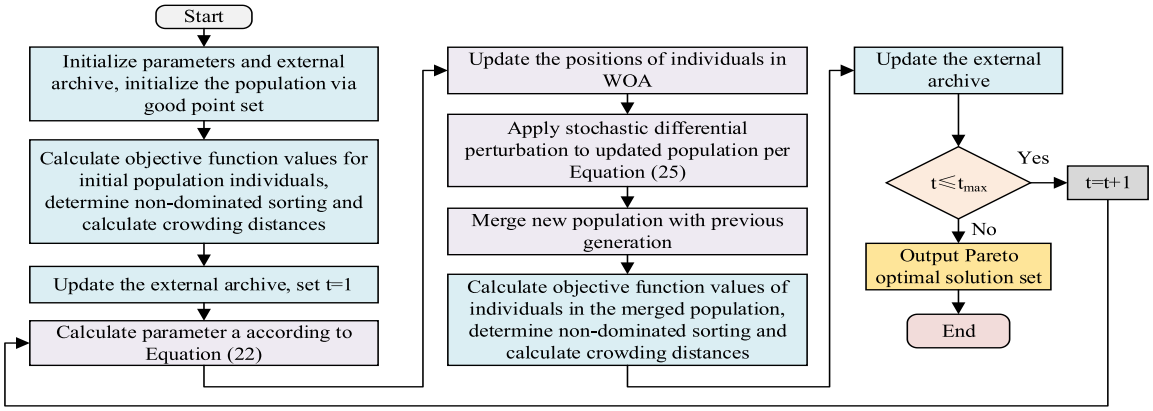


Figure 3: Flowchart of IMOWOA.

Three classic algorithms for multi-objective scheduling problems are selected for comparison: Non-dominated Sorting Genetic Algorithm (NSGA-II) [26,27]. Multi-Objective Particle Swarm Optimization (MOPSO) [28,29], and Multi-objective Artificial Bee Colony Algorithm (MOABC) [30,31], which are compared with the IMOWOA proposed in this paper. The parameter settings of all comparison algorithms follow the corresponding papers; the population size of the algorithms involved in the comparison is set to 100, the maximum number of iterations is uniformly 300, and the external archive capacity is set to 100. Among them, the crossover rate and mutation rate of NSGA-II are 0.8 and 0.1, respectively; the inertia weight of MOPSO decreases linearly from 0.9 to 0.4; the food source update step size of MOABC is 0.01, and the learning factor is 2; the scaling factor χ of IMOWOA is set to the classic 0.5. Each algorithm is run independently 30 times based on each function, and the average values of each metric are calculated as shown in Table 2, where the optimal results are presented in bold numbers. The solution results of each algorithm are plotted into boxplots, as shown in Fig. 4.

Table 2: The average value of the evaluation metric of the algorithm.

Function	Metric	NSGA-II	MOPSO	MOABC	IMOWOA
ZDT1	IGD	4.80E−03	7.07E−03	3.99E−03	2.07E−03
	HV	1.96E−01	2.30E−01	2.25E−01	2.50E−01
	CPU (s)	28.3	21.5	24.7	18.2
ZDT2	IGD	9.03E−02	8.15E−02	1.09E−01	5.74E−02
	HV	1.05E−01	1.27E−01	1.20E−01	1.36E−01
	CPU (s)	27.9	21.0	24.2	17.9
ZDT3	IGD	1.29E−02	8.95E−03	3.71E−03	5.44E−03
	HV	1.34E−01	1.33E−01	1.61E−01	1.78E−01

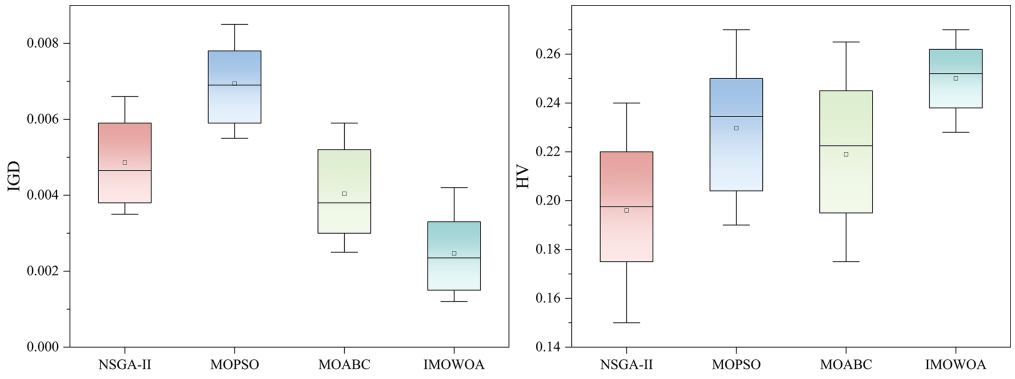
(Continued)

Table 2 (continued)

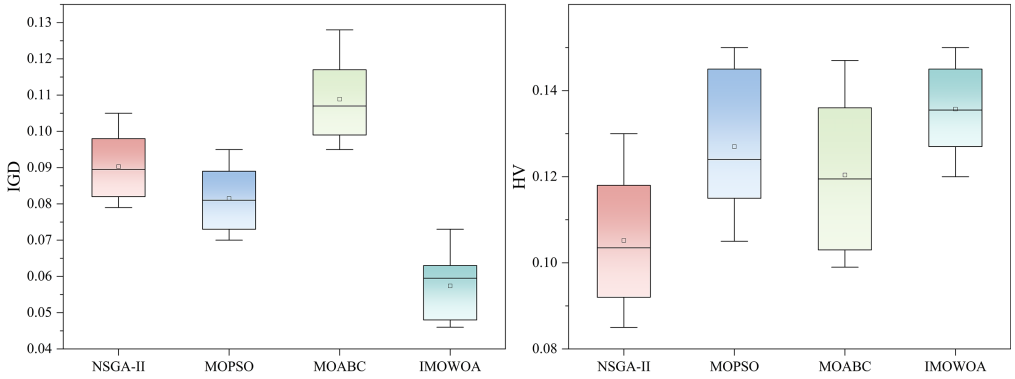
Function	Metric	NSGA-II	MOPSO	MOABC	IMOWOA
	CPU (s)	29.1	22.1	25.0	18.5
ZDT4	IGD	8.22E−03	2.42E−02	3.77E−02	5.42E−03
	HV	2.54E−01	1.81E−01	2.14E−01	2.44E−01
	CPU (s)	30.2	23.4	26.3	19.0
ZDT6	IGD	1.96E−02	2.51E−02	1.65E−02	7.22E−03
	HV	3.30E−01	2.79E−01	4.05E−01	5.05E−01
	CPU (s)	29.8	22.7	25.8	18.8
DTLZ1	IGD	2.27E−01	4.49E−01	3.87E−01	2.55E−01
	HV	6.46E−01	5.08E−01	4.54E−01	7.48E−01
	CPU (s)	34.5	26.5	30.1	22.3
DTLZ2	IGD	3.13E−01	3.75E−01	3.40E−01	2.47E−01
	HV	3.70E−01	3.50E−01	4.15E−01	4.43E−01
	CPU (s)	33.6	25.8	29.3	21.6
DTLZ7	IGD	2.76E−01	2.75E−01	2.36E−01	1.82E−01
	HV	2.65E−01	2.35E−01	1.52E−01	2.90E−01
	CPU (s)	34.2	26.1	29.8	22.0

In rescheduling problems, the time required to obtain a high-quality solution is as critical as the solution quality itself, especially when decisions must be made under strict time constraints in practical environments. To evaluate the real-time performance of the proposed IMOWOA, we recorded the CPU time consumed by each algorithm during the optimization process. The average CPU time over 30 independent runs for each test function is reported in Table 2, alongside the IGD and HV metrics.

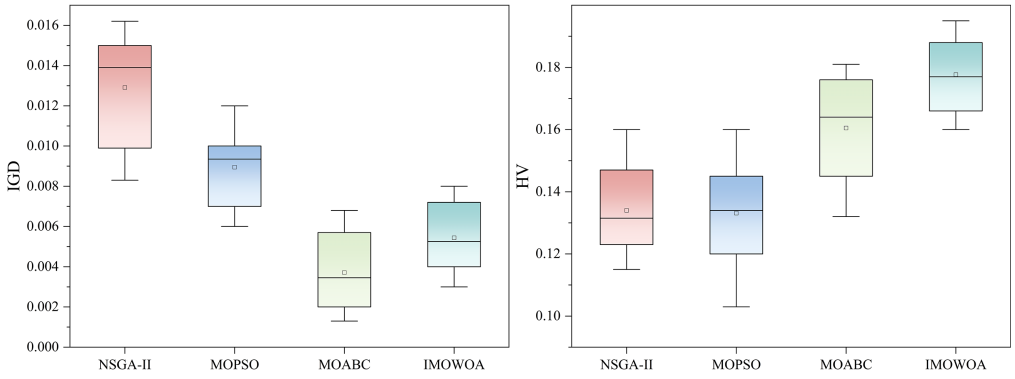
As can be seen from Table 2 and Fig. 4, in terms of the IGD metric, IMOWOA achieves the best performance on 6 test functions, including ZDT1, ZDT2, and ZDT4. This indicates that the obtained Pareto optimal solution set is closest to the true Pareto front, i.e., the algorithm has strong convergence and high solution accuracy. For the HV metric, IMOWOA performs optimally on 7 test functions such as ZDT1, ZDT2, and ZDT3, demonstrating that the Pareto optimal solution set obtained by the algorithm is uniformly distributed and relatively stable. In addition, compared with other algorithms, the IGD and HV metric values of IMOWOA's solutions are concentrated in distribution, indicating good dispersion, which suggests that it has high stability and reliability in solving multi-objective optimization problems. Although IMOWOA integrates a good point set initialization, nonlinear control parameters, and differential evolution mutation, its average CPU time is still lower than that of the comparison algorithms. IMOWOA has a simple structure, and the computational load per generation is lower than that of MOABC, which requires complex neighborhood search, and NSGA-II, which requires global sorting. The good point set initialization is executed only once, and the time it takes is negligible. The nonlinear control parameters enable the algorithm to conduct strong global exploration in the early stage and rapid local exploitation in the later stage, thereby reducing ineffective search and increasing the proportion of effective search, indirectly shortening the total time. Therefore, IMOWOA achieves the highest computational efficiency while ensuring the quality of the solution, and has been verified in practical cases.



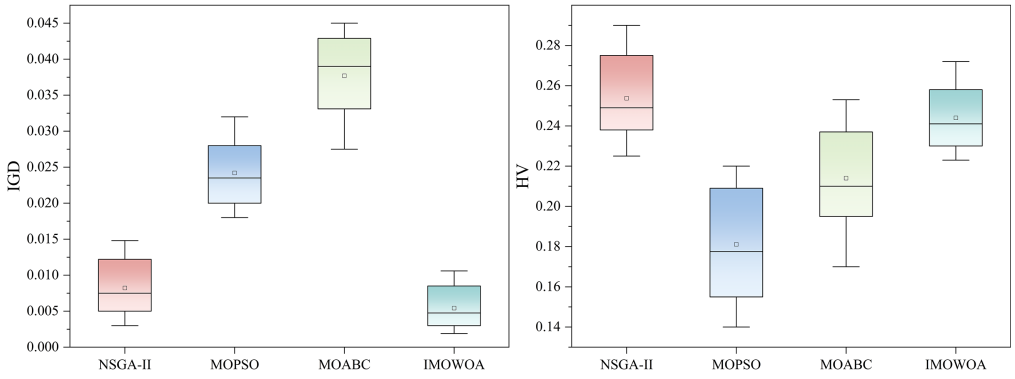
(a) Solution results of each algorithm for ZDT1 function



(b) Solution results of each algorithm for ZDT2 function

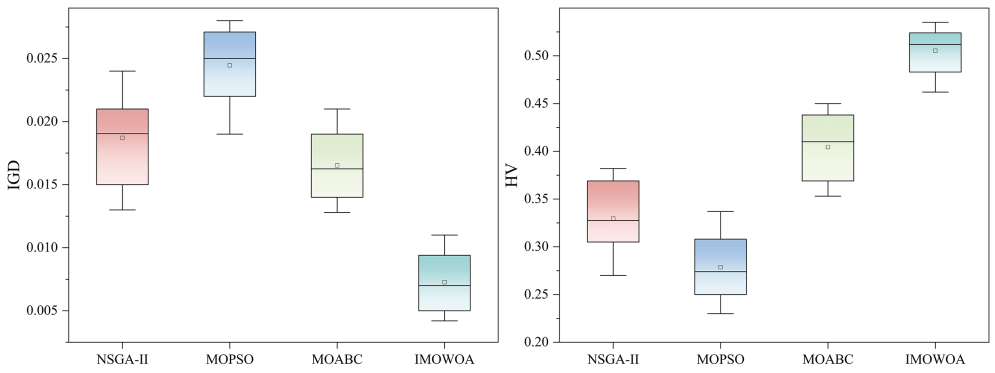


(c) Solution results of each algorithm for ZDT3 function

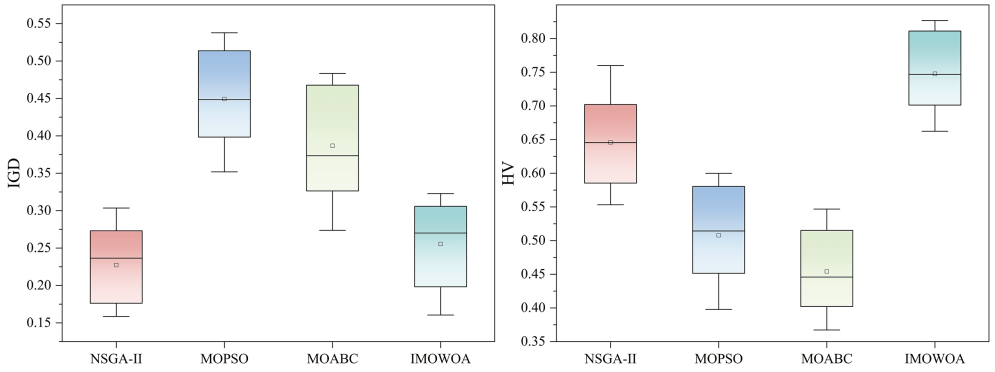


(d) Solution results of each algorithm for ZDT4 function

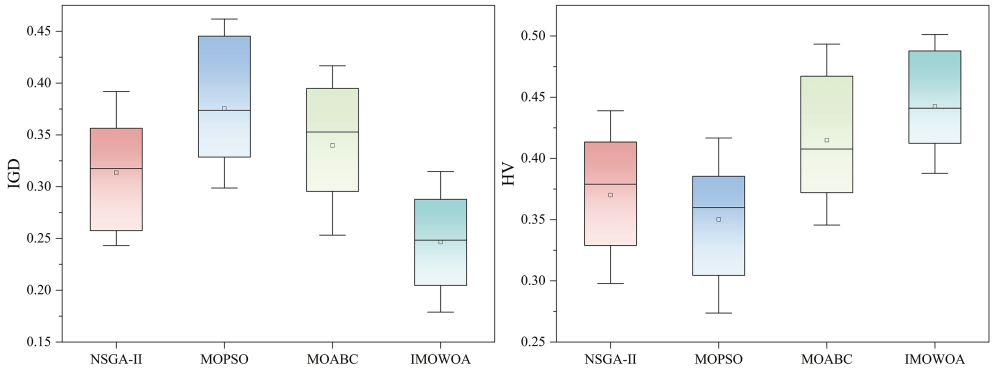
Figure 4: (Continued)



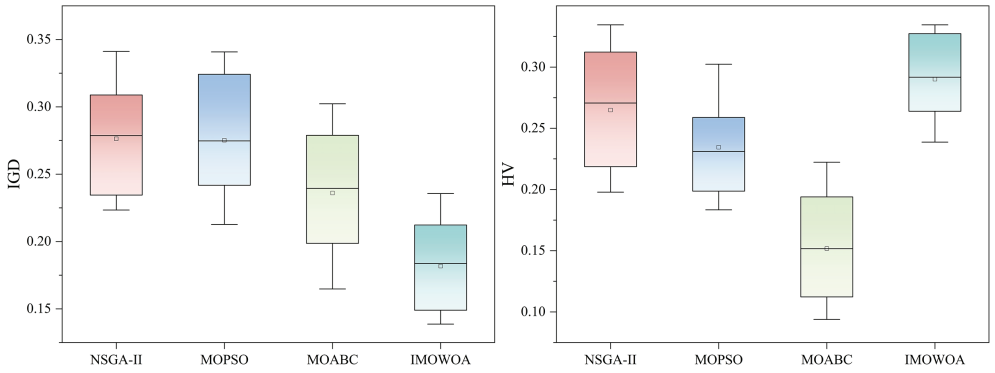
(e) Solution results of each algorithm for ZDT6 function



(f) Solution results of each algorithm for DLTZ1 function



(g) Solution results of each algorithm for DLTZ2 function



(h) Solution results of each algorithm for DLTZ7 function

Figure 4: Box diagrams of the solution results for (**a**) ZDT1, (**b**) ZDT2, (**c**) ZDT3, (**d**) ZDT4, (**e**) ZDT6, (**f**) DTLZ1, (**g**) DTLZ2, and (**h**) DTLZ7.

To more intuitively demonstrate the performance of different algorithms, taking the DTLZ2 function as an example, representative solution sets where both IGD and HV values are close to the average are selected from the results of 30 independent runs of each algorithm. These are then compared with the true Pareto front. The comparison results of each algorithm are shown in Fig. 5.

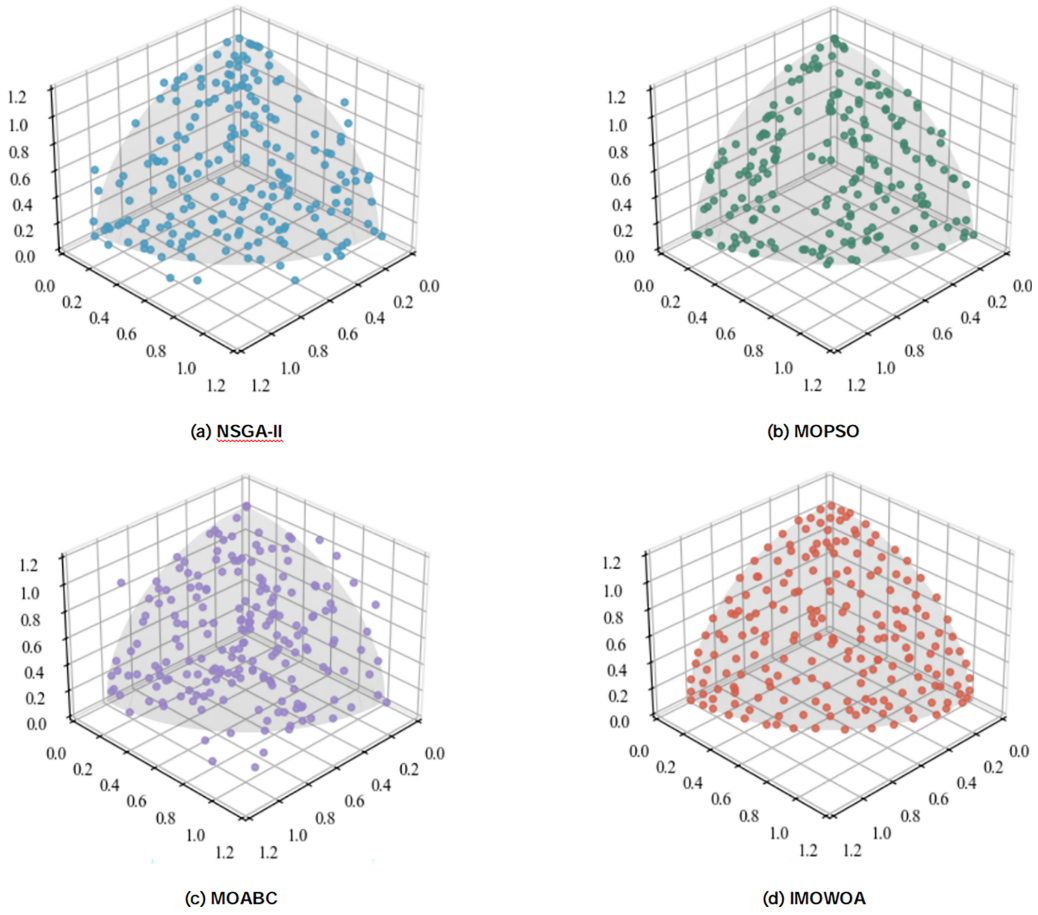


Figure 5: Solution sets of different algorithms on DTLZ2 test functions for (**a**) NSGA-II, (**b**) MOPSO, (**c**) MOABC, and (**d**) IMOWOA.

5.1.2 Validation of the Effectiveness of Improved Strategies

To verify the independent contribution of each improvement strategy to the performance of IMOWOA, this paper designs ablation experiments. Considering computational efficiency, four typical functions, namely ZDT1, ZDT3, DTLZ2, and DTLZ7, are selected from the test function set to cover common scenarios in multi-objective optimization. Under the premise of keeping other modules unchanged, each single improvement module is removed in sequence and compared with the complete IMOWOA. The population size is 100, the maximum number of iterations is 300, and each algorithm is independently run 30 times to calculate the mean values of IGD and HV. Among them: IMW denotes the complete IMOWOA algorithm; IMW1 removes the good point set strategy; IMW2 removes the nonlinear control parameter; IMW3 removes the differential evolution mutation strategy; IMW4 removes the dynamic external archive based on crowding degree. The results are shown in Table 3.

Table 3: Comparison of metrics in ablation experiments.

Function	Metric	IMW	IMW1	IMW2	IMW3	IMW4
ZDT1	IGD	2.07E−03	5.12E−03	3.89E−03	4.26E−03	3.58E−03
	HV	2.50E−01	1.98E−01	2.15E−01	2.02E−01	2.20E−01
ZDT3	IGD	5.44E−03	9.87E−03	7.62E−03	8.13E−03	6.85E−03
	HV	1.78E−01	1.42E−01	1.56E−01	1.48E−01	1.65E−01
DTLZ2	IGD	2.47E−01	3.68E−01	3.05E−01	3.29E−01	2.83E−01
	HV	4.43E−01	3.62E−01	3.98E−01	3.75E−01	4.12E−01
DTLZ7	IGD	1.82E−01	2.75E−01	2.36E−01	2.51E−01	2.15E−01
	HV	2.90E−01	2.32E−01	2.58E−01	2.41E−01	2.68E−01

From the results in Table 3, the complete algorithm IMW exhibits the smallest average IGD values and the largest average HV values across the four test functions, significantly outperforming all variant algorithms. This fully verifies the effectiveness of each improved strategy.

Specifically: On ZDT1, the IGD of IMW1 was 147% higher and the HV was 20.8% lower than that of IMW, indicating that the good solution set provides high-quality initial solutions through uniform initialization, laying the foundation for global exploration. On ZDT3, the IGD of IMW3 increased by 49.4% and the HV decreased by 16.9%, suggesting that the DE mutation perturbation helps to cross discontinuous regions and maintain diversity. On DTLZ2, the IGD of IMW2 increased by 23.5% and the HV decreased by 10.1%, verifying the ability of the two-stage nonlinear parameters to balance exploration and exploitation in the high-dimensional space. On DTLZ7, the IGD of IMW4 increased by 18.1% and the HV decreased by 7.6%, proving that the archive mechanism based on crowding avoids solution aggregation and ensures uniform distribution. Therefore, through the progressive collaboration of these improved strategies, they collectively support the comprehensive advantages of the proposed IMOWOA in convergence, distribution, and stability, verifying the effectiveness of the improved mechanisms.

5.1.3 Parameter Sensitivity Analysis Experiment

This article introduces several new parameters. We conduct a sensitivity analysis on the key parameters, mainly focusing on the scaling factor χ in the differential evolution strategy. This parameter is one of the new key parameters introduced in IMOWOA and has potential influence on the algorithm performance. We analyzed the impact of χ on the algorithm performance through experiments to verify that it can provide parameter setting suggestions for practical applications.

Selecting ZDT1, ZDT3 and DTLZ2 as representatives, these functions cover different characteristics of the Pareto front. Parameter values: The scaling factor χ is set to 0.1, 0.3, 0.5, 0.7, 0.9, 1.1, and 1.5, respectively. The remaining parameters are consistent with those in Section 5.1.1 of the paper: population size 100, maximum number of iterations 300, external archive capacity 100, and each configuration is run independently 30 times. Evaluation indicators: The mean and standard deviation of IGD and HV are used to measure the convergence and distribution of the solution set.

The experimental results are shown in Table 4 (for simplicity, only the IGD values are listed, and the HV trend is similar). The bolded part indicates the optimal IGD value on this function. From the table, it can be seen that when $\chi \in [0.3, 0.7]$, IMOWOA can obtain stable and superior IGD values on all test functions, indicating that the algorithm is insensitive to χ within this range and has good robustness. When $\chi < 0.3$, the

IGD value significantly increases, indicating that a too small scaling factor weakens the exploration ability of the differential mutation, causing the population to converge prematurely to the local optimum. When $\chi > 0.9$, the IGD value also increases, especially at $\chi = 1.5$, where the performance drops significantly. This is because an excessively large scaling factor makes the mutation step size too large, disrupting the structure of the excellent individuals and reducing the convergence accuracy. Near the default value $\chi = 0.5$, the algorithm achieves the best or second-best performance, which is consistent with the performance of IMW in the ablation experiments, verifying the rationality of this parameter setting.

Table 4: The mean (standard deviation) of IGD of IMOWOA on the test function under different χ values.

χ	ZDT1 (IGD)	ZDT3 (IGD)	DTLZ2 (IGD)
0.1	5.21E−3 (1.2E−3)	1.02E−2 (2.1E−3)	3.58E−1 (4.3E−2)
0.3	**2.15E−3 (0.3E−3)**	5.62E−3 (0.8E−3)	2.55E−1 (2.1E−2)
0.5	2.07E−3 (0.2E−3)	**5.44E−3 (0.6E−3)**	**2.47E−1 (1.9E−2)**
0.7	2.21E−3 (0.3E−3)	5.81E−3 (0.9E−3)	2.60E−1 (2.5E−2)
0.9	3.48E−3 (0.7E−3)	7.33E−3 (1.4E−3)	3.02E−1 (3.6E−2)
1.1	4.76E−3 (1.0E−3)	9.15E−3 (1.8E−3)	3.47E−1 (4.0E−2)
1.5	6.83E−3 (1.5E−3)	1.28E−2 (2.5E−3)	4.21E−1 (5.1E−2)

5.1.4 Wilcoxon Rank-Sum Test

To further verify whether the performance improvement of IMOWOA in terms of IGD and HV indicators is statistically significant, this paper employs the Wilcoxon rank sum test to conduct pairwise comparisons of the IGD and HV values obtained from 30 independent runs of IMOWOA and the other three comparison algorithms (NSGA-II, MOPSO, and MOABC).

The bolded values indicate $p < 0.05$, meaning that IMOWOA outperforms the comparison algorithm significantly in this test function and metric. From the Table 5, it can be seen that in the vast majority of cases (out of 42 comparisons, 37 had $p < 0.05$), the performance improvement of IMOWOA is statistically significant. A few insignificant cases (such as IGD of ZDT3 vs. NSGA-II, HV of ZDT4 vs. NSGA-II, and IGD of DTLZ1 vs. NSGA-II) may be due to the fact that these functions have complex Pareto frontiers, resulting in smaller performance differences between the algorithms, but IMOWOA still outperforms the comparison algorithms on average. Overall, the results of the Wilcoxon test further confirm the superiority of IMOWOA in terms of convergence, distribution, and stability.

Table 5: Wilcoxon rank sum test p-value.

Function	Metric	IMOWOA vs. NSGA-II	IMOWOA vs. MOPSO	IMOWOA vs. MOABC
ZDT1	IGD	**0.003**	**0.008**	**0.012**
	HV	**0.002**	**0.005**	**0.009**
ZDT2	IGD	**0.001**	**0.004**	**0.007**
	HV	**0.003**	**0.006**	**0.011**
ZDT3	IGD	0.067	**0.021**	**0.015**
	HV	**0.009**	**0.013**	**0.008**

(Continued)

Table 5 (continued)

Function	Metric	IMOWOA vs. NSGA-II	IMOWOA vs. MOPSO	IMOWOA vs. MOABC
ZDT4	IGD	**0.004**	**0.010**	**0.018**
	HV	0.052	**0.024**	**0.031**
ZDT6	IGD	**0.002**	**0.007**	**0.009**
	HV	**0.001**	**0.005**	**0.003**
DTLZ1	IGD	0.120	**0.034**	**0.045**
	HV	**0.008**	**0.012**	**0.007**
DTLZ2	IGD	**0.005**	**0.009**	**0.014**
	HV	**0.004**	**0.011**	**0.006**
DTLZ7	IGD	**0.003**	**0.008**	**0.013**
	HV	**0.002**	**0.007**	**0.010**

5.2 Practical Case Solution

5.2.1 Basic Data

This paper takes the final assembly production line of an agricultural machinery equipment enterprise as the research object. Due to the numerous and complex processes in the entire tractor production process, to simplify the model, the following modeling is only carried out for 5 key processes of the pre-painting line and 5 key processes of the post-painting line in the final assembly production line, combined with the actual production process, as shown in Fig. 6.

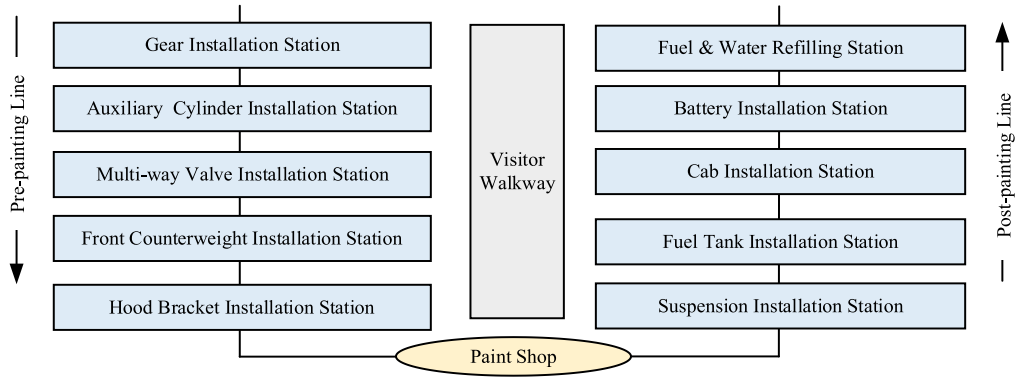


Figure 6: Simplified model of tractor assembly line.

This example production line has 10 processing operations, with each processing operation equipped with two identical parallel machines. These workstations need to process 3 order tasks of different models (OA, OB, OC) simultaneously. The workshop adopts a batch production mode in advance, dividing the three orders into 6 job tasks for batch production. Relevant processing information is shown in Tables 6–8 below, including processing equipment, processing time, job batch size, etc., where "—" indicates that the job does not go through this processing operation.

Table 6: Process operation information.

Operation	Machine	Number
Gear Installation	M1, M2	2
Auxiliary Cylinder Installation	M3, M4	2
Multi-Way Valve Installation	M5, M6	2
Front Counterweight Installation	M7, M8	2
Hood Bracket Installation	M9, M10	2
Suspension Installation	M11, M12	2
Fuel Tank Installation	M13, M14	2
Cab Installation	M15, M16	2
Battery Installation	M17, M18	2
Fuel & Water Refilling	M19, M20	2

Table 7: Job processing information.

Job	Model	Lot Size/Pcs
1	LX1804 (OA)	100
2	LX1804 (OA)	100
3	LX1804 (OA)	100
4	LF-C (OB)	100
5	80/LY (OC)	100
6	80/LY (OC)	100

Table 8: Job processing information.

Installation Operation	Processing Time per Piece/min					
	Job1	Job2	Job3	Job4	Job5	Job6
Gear Installation	3	3	3	2.5	2	2
Auxiliary Cylinder	2.5	2.5	2.5	2	2	2
Multi-Way Valve	3	3	3	2	2	2
Front Counterweight	3	3	3	1.5	1.5	1.5
Hood Bracket	1.5	1.5	1.5	2	—	—
Suspension	2.5	2.5	2.5	2	2	2
Fuel Tank	3	3	3	2.5	—	—
Cab	3	3	3	2.5	2	2
Battery	3	3	3	2	1.5	1.5
Fuel & Water	2	2	2	1.5	1.5	1.5

The Gantt chart of the initial scheduling scheme is shown in Fig. 7. In the chart, different workpieces are distinguished by different colors. For ease of understanding, jobs are represented by the first digit, and operation processes by the second digit. For example, 4-1 represents the first operation process of job 4.

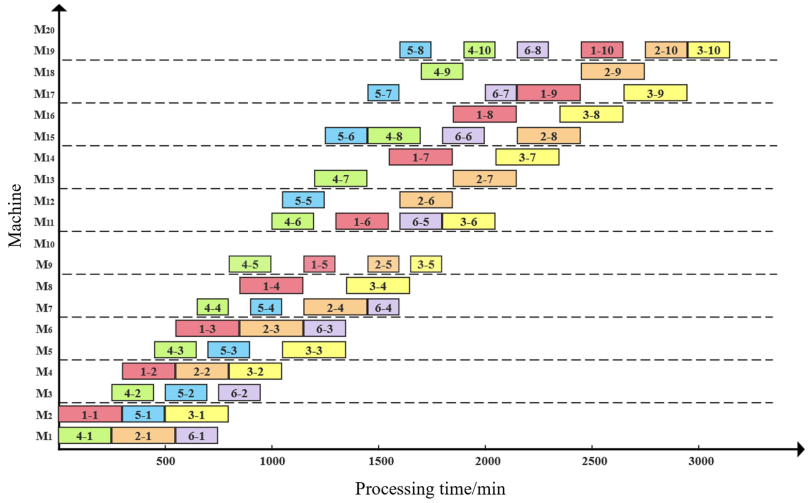


Figure 7: Gantt chart of the initial schedule scheme.

5.2.2 Calculation of Latest Completion Time

As can be seen from the initial scheduling information, the delivery dates of each job are as follows: D1 = D2 = D3 = 3150 min, D4 = 2050 min, D5 = D6 = 2300 min. Based on the calculation model for the latest completion time, the latest completion time of each operation is obtained as shown in Table 9.

Table 9: Latest completion time.

Operation	Latest Completion Time/min					
	Job1	**Job2**	**Job3**	**Job4**	**Job5**	**Job6**
Operation 1	300	550	800	250	500	1050
Operation 2	550	800	1050	450	700	1250
Operation 3	850	1150	1350	650	900	1450
Operation 4	1200	1450	1650	800	1050	1600
Operation 5	1350	1600	1800	1000	1250	1800
Operation 6	1600	1850	2050	1200	1450	2000
Operation 7	1950	2150	2350	1450	1750	2150
Operation 8	2250	2450	2650	1700	1900	2300
Operation 9	2550	2750	2950	1900	—	—
Operation 10	2750	2950	3150	2050	—	—

5.2.3 Solution of Rescheduling Scheme

This study takes a single-machine failure as the disturbance case. The reasons are as follows: Machine failure is the most typical and destructive disturbance event in a hybrid flow shop, directly causing resource interruption and spreading delays through process and machine dimensions. It can comprehensively test the quantitative discrimination ability of the "hybrid-driven mechanism based on the latest completion time" proposed in this paper for the degree of disturbance, as well as the rescheduling optimization performance of the IMOWOA algorithm under strong constraints. Compared with order changes, the failure involves complex constraints such as interruption of remaining time calculation and update of machine release time.

Successful solution can provide strong evidence for the application of the method to other disturbance types. The aim of this study is to propose a general rescheduling framework that is independent of the type of disturbance. The mathematical model and IMOWOA algorithm do not rely on specific disturbance types. The machine failure is only used as a verification example.

Assume that at the 1400th min of processing, machine M11 fails, and expert assessment indicates that the fault repair will take 10 h. As shown in Table 9, the latest completion time of operation O1,6 is 1600 min, while the fault repair completion time of M11 far exceeds 1600 min. Therefore, rescheduling is triggered immediately.

The operation sets in the initial scheduling scheme are partitioned as follows:

Completed Operation Set: These operations are already processed and do not require rescheduling.

In-Progress Operation Set: For operations directly affected by the disturbance, potential machine reallocation needs to be evaluated based on specific conditions; other operations in this set continue according to the original schedule without rescheduling.

Unprocessed Operations and Unscheduled Operations: These operations require rescheduling.

At the rescheduling trigger time, no new orders arrive (i.e., there are no unscheduled operations). The specific classification of operation sets is shown in Table 10.

Table 10: Latest completion time.

Set	Operations
Completed operations	$O_{1,1}$, $O_{1,2}$, $O_{1,3}$, $O_{1,4}$, $O_{1,5}$, $O_{2,1}$, $O_{2,2}$, $O_{2,3}$, $O_{3,1}$, $O_{3,2}$, $O_{3,3}$, $O_{4,1}$, $O_{4,2}$, $O_{4,3}$, $O_{4,4}$, $O_{4,5}$, $O_{4,6}$, $O_{5,1}$, $O_{5,2}$, $O_{5,3}$, $O_{5,4}$, $O_{5,5}$, $O_{6,1}$, $O_{6,2}$, $O_{6,3}$
In-Progress operations	$O_{1,6}$, $O_{2,4}$, $O_{3,4}$, $O_{4,7}$, $O_{5,6}$
Unprocessed operations	$O_{1,7}$, $O_{1,8}$, $O_{1,9}$, $O_{1,10}$, $O_{2,5}$, $O_{2,6}$, $O_{2,7}$, $O_{2,8}$, $O_{2,9}$, $O_{2,10}$, $O_{3,5}$, $O_{3,6}$, $O_{3,7}$, $O_{3,8}$, $O_{3,9}$, $O_{3,10}$, $O_{4,8}$, $O_{4,9}$, $O_{4,10}$, $O_{5,7}$, $O_{5,8}$, $O_{6,4}$, $O_{6,5}$, $O_{6,6}$, $O_{6,7}$, $O_{6,8}$, $O_{6,9}$, $O_{6,10}$
Unscheduled operations	—

The release times of jobs and machines are determined by their processing status at the rescheduling trigger time. When the fault occurs, apart from the faulty machine M11, which is processing operation $O_{1,6}$, machines M7, M8, M13, and M15 are also in operation, while all other machines are idle. Therefore, the release time of machine M11 is the finish time of machine repair; the release times of machines M7, M8, M13, and M15 are the completion times of their currently processing operations; and the release times of the remaining machines are the start time of rescheduling.

For each job, at the moment of the machine fault, operations $O_{1,6}$, $O_{2,4}$, $O_{3,4}$, $O_{4,7}$, and $O_{5,6}$ are being processed. Thus, the release times of jobs 2, 3, 4, and 5 are the completion times of their currently processing operations. Since operation $O_{1,6}$ is being processed on the faulty machine at the time of the fault, the release time of job 1 is the current rescheduling moment. For other jobs that are not being processed at the time of the machine fault, their release times are the rescheduling moment.

In addition, due to the processing interruption of operation $O_{1,6}$, and as part of the batch processing tasks have been completed by the time of the fault, it is necessary to calculate its remaining processing time and update it in the operation processing time information; the processing times of other operations to be rescheduled remain unchanged. The release times of each job and machine are shown in Tables 11 and 12.

Table 11: Release time of jobs.

Job	Release Time/min
Job 1	1400
Job 2	1450
Job 3	1650
Job 4	1450

Table 12: Release time of machines.

Machine	Release Time/min	Machine	Release Time/min
M_1	1400	M_{11}	2000
M_2	1400	M_{12}	1400
M_3	1400	M_{13}	1450
M_4	1400	M_{14}	1400
M_5	1400	M_{15}	1450
M_6	1400	M_{16}	1400
M_7	1450	M_{17}	1400
M_8	1650	M_{18}	1400
M_9	1400	M_{19}	1400
M_{10}	1400	M_{20}	1400

To visually demonstrate the optimization behavior and solution efficiency of IMOWOA, a convergence curve is introduced. Fig. 8 presents the convergence curves of four algorithms (IMOWOA, NSGA-II, MOPSO, MOABC) in 300 iterations for solving the mixed flow-shop rescheduling problem. The vertical axis represents the three optimization objectives, and the curves represent the average of the optimal values from 30 independent runs. The results show that IMOWOA converges faster in the early iterations of the three objectives, and the final solution quality is significantly superior, indicating that the initialization of good points, nonlinear control parameters, and differential evolution mutation effectively enhance the global exploration and local exploitation capabilities. In contrast, NSGA-II and MOABC tend to stagnate in the later stages, and MOPSO has moderate performance but a lower convergence speed and accuracy compared to IMOWOA.

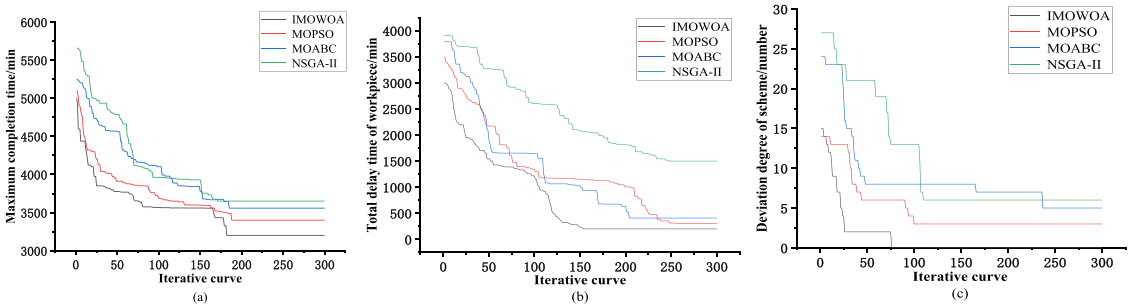
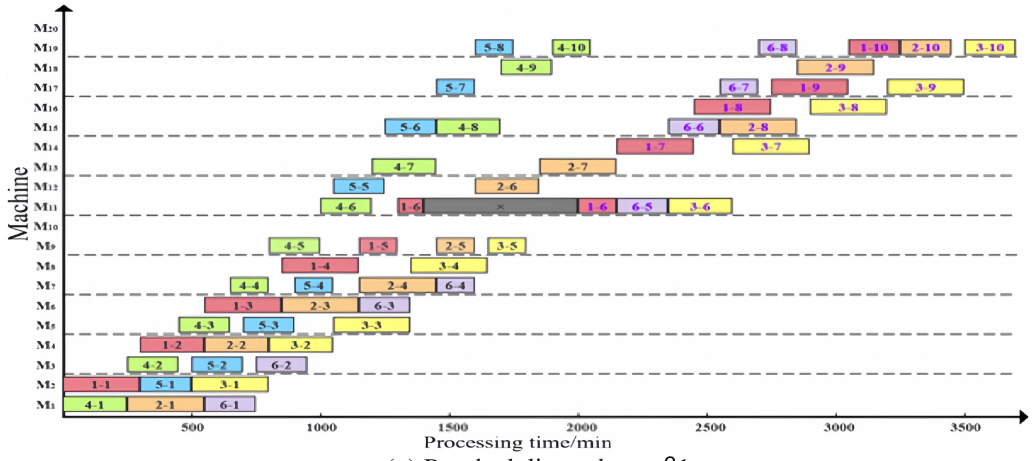


Figure 8: Convergence curves of the objective function values for (**a**) the maximum completion time, (**b**) total delay duration of the workpiece, and (**c**) the degree of deviation of the solution.

The convergence behavior is consistent with the CPU time results in Table 2: Although IMOWOA introduced mechanisms such as non-dominated sorting and dynamic archiving, its average CPU time is still the lowest, indicating that the improvement strategies did not reduce the computational efficiency but achieved more precise and efficient search. Therefore, the low computational cost and rapid convergence jointly verified the effectiveness and practicality of IMOWOA in solving the disturbed multi-objective mixed flow-shop rescheduling problem.

The constraints such as task processing and machine availability time were incorporated into the solution model, and the IMOWOA algorithm was used to solve this model. If the machine fails, the calculation is performed based on the basic data of the rescheduling decision selection model. Three representative schemes were selected from the Pareto set obtained through the calculation. The Gantt chart is shown in Fig. 9. In the figure, the gray shaded area represents the maintenance period of machine M11.

From Fig. 9, it can be concluded that the three typical rescheduling schemes generated based on the IMOWOA algorithm represent different scheduling strategies with different target preferences within the Pareto optimal solution set. Scheme β1 has the optimal deviation from the plan, scheme β2 prioritizes the total delay duration of the workpieces, and scheme β3 prioritizes the maximum completion time. Decision-makers can balance among the three schemes based on the actual production status to achieve lean rescheduling decisions.



(a) Rescheduling scheme β1

Figure 9: (Continued)

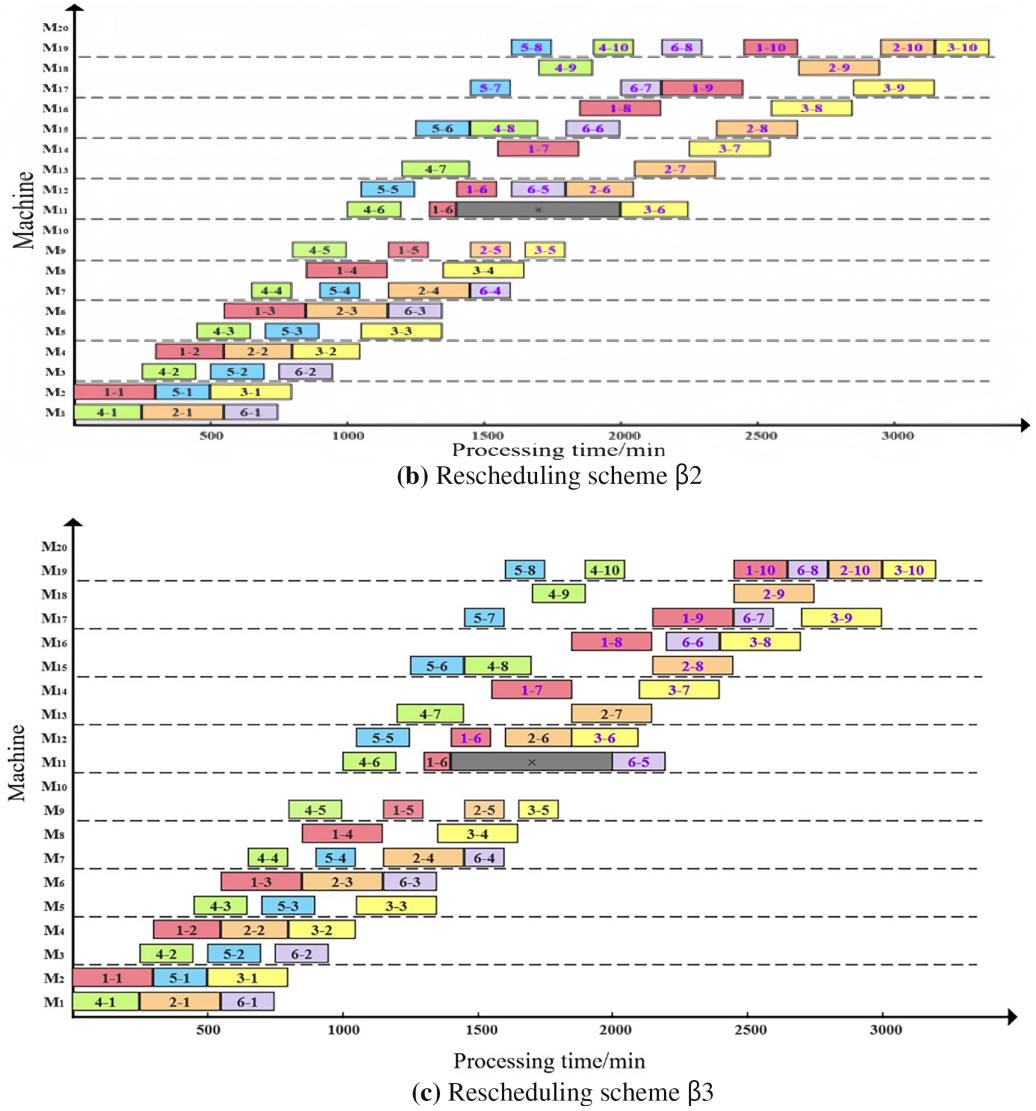


Figure 9: Gantt chart of the rescheduling scheme (**a**) β1, (**b**) β2, and (**c**) β3.

6 Conclusion and Future Works

This paper addresses the problem of disturbance response in a mixed flow shop and proposes a re-scheduling method. A rescheduling mathematical model with the optimization goals of minimizing the maximum completion time, total delay time, and scheme deviation degree is established. A mixed rescheduling driving mechanism based on the latest completion time is constructed. By integrating the good point set theory, nonlinear control parameter strategy, and differential evolution algorithm to improve the whale optimization algorithm, and introducing non-dominated sorting and a dynamic external archive mechanism based on crowding degree, it makes it applicable to multi-objective optimization. Comparative experiments and ablation experiments on ZDT and DTLZ benchmark functions show that the proposed IMOWOA algorithm has good solution performance and convergence. The effectiveness of each improvement mechanism is verified. Taking the assembly line of a certain agricultural machinery manufacturing enterprise as an example, a rescheduling scheme under disturbance is generated based on actual requirements, verifying the feasibility and effectiveness of the proposed method. This paper provides

theoretical and methodological support for disturbance response in mixed flow shops such as agricultural machinery manufacturing. The driving mechanism based on the latest completion time helps to respond quickly to critical disturbances while ensuring the stability of the plan, reducing ineffective scheduling and production interruptions, and has practical significance for improving the on-time delivery rate of orders and reducing delay costs.

However, this research still has certain limitations. The proposed IMOWOA algorithm may face challenges in computational efficiency and real-time response capability when solving ultra-large-scale production scheduling problems or dealing with high-frequency disturbance scenarios. Meanwhile, the current model mainly considers processing time fluctuation under disturbances, and its comprehensive handling capability for complex concurrent disturbance events requires further verification.

Meanwhile, the current mathematical model is based on several idealized assumptions. It is also important to set the time in the actual hybrid flow shop. Therefore, in future research, transportation time and preparation time can be included as additional processing stages for consideration.

Future research will aim to further optimize the computational efficiency and robustness of the multi-objective whale algorithm, exploring the application potential of parallel computing or distributed architectures. Additionally, the model will be expanded to accommodate more types of concurrent disturbance events and complex constraints, enhancing the universality and practicality of the method in complex industrial scenarios.

Acknowledgement: Not applicable.

Funding Statement: This work was funded by National Key Research and Development Program Projects of China under Grant No. 2020YFB1713500.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Feng Lv and Xin Xu; methodology, Xin Xu; software, Xin Xu; validation, Xin Xu; formal analysis, Xin Xu; investigation, Feng Lv and Xin Xu; data curation, Xin Xu; writing—original draft preparation, Xin Xu; writing—review and editing, Feng Lv and Xin Xu; visualization, Xin Xu; supervision, Feng Lv, Cheng Yang and Yixuan Tang; project administration, Feng Lv and Xin Xu; funding acquisition, Feng Lv. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are available within the article.

Ethics Approval: Not applicable.

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

References

1. Zhou G, Chen Z, Zhang C, Chang F. An adaptive ensemble deep forest based dynamic scheduling strategy for low carbon flexible job shop under recessive disturbance. J Clean Prod. 2022;337(1):130541. doi:10.1016/j.jclepro.2022.130541.
2. Lü Y, Xu ZJ, Li CB, Li LL, Yang M. Comprehensive energy saving optimization of processing parameters and job shop dynamic scheduling considering disturbance events. J Mech Eng. 2022;58(19):242–55. (In Chinese).
3. Deng L, Qiu Y, Gong W, Di Y, Li C. A dynamic decision-driven memetic algorithm for fuzzy distributed hybrid flow shop rescheduling considering quality control. Expert Syst Appl. 2024;257(5):125002. doi:10.1016/j.eswa.2024.125002.

4. Zheng P, Wang J, Zhang J, Yang C, Jin Y. An adaptive CGAN/IRF-based rescheduling strategy for aircraft parts remanufacturing system under dynamic environment. Robot Comput Integr Manuf. 2019;58:230–8. doi:10.1016/j.rcim.2019.02.008.
5. An Y, Chen X, Gao K, Li Y, Zhang L. Multiobjective flexible job-shop rescheduling with new job insertion and machine preventive maintenance. IEEE Trans Cybern. 2023;53(5):3101–13. doi:10.1109/tcyb.2022.3151855.
6. Song L, Xu Z, Wang C, Su J. A new decision method of flexible job shop rescheduling based on WOA-SVM. Systems. 2023;11(2):59. doi:10.3390/systems11020059.
7. Gao K, Yang F, Li J, Sang H, Luo J. Improved Jaya algorithm for flexible job shop rescheduling problem. IEEE Access. 2020;8:86915–22. doi:10.1109/access.2020.2992478.
8. Fan C, Wang W, Tian J. Flexible job shop scheduling with stochastic machine breakdowns by an improved tuna swarm optimization algorithm. J Manuf Syst. 2024;74(3):180–97. doi:10.1016/j.jmsy.2024.03.002.
9. Ali KB, Bechikh S, Louati A, Louati H, Kariri E. Dynamic job shop scheduling problem with new job arrivals using hybrid genetic algorithm. IEEE Access. 2024;12(1):85338–54. doi:10.1109/access.2024.3401080.
10. He Z, Tang B, Luan F. An improved African vulture optimization algorithm for dual-resource constrained multi-objective flexible job shop scheduling problems. Sensors. 2022;23(1):90. doi:10.3390/s23010090.
11. Schworm P, Wu X, Klar M, Aurich JC. Multi-objective rescheduling of job shop scheduling problems in manufacturing using Quantum Annealing. Manuf Lett. 2024;42(22):5–10. doi:10.1016/j.mfglet.2024.09.066.
12. Sun J, Zhang Z, Zhang G, Huang Z. Multi-objective evolutionary algorithm based flexible assembly job-shop rescheduling with component sharing for order insertion. Comput Oper Res. 2024;169(5):106744. doi:10.1016/j.cor.2024.106744.
13. Tang L, Cheng F, Ji WX, Jin ZB. Improved ICA for rush order insertion rescheduling problem under flexible job shops. Comput Eng Appl. 2023;59(21):303–11. (In Chinese).
14. Zhang W, Zheng Y, Ahmad R. An energy-efficient multi-objective scheduling for flexible job-shop-type remanufacturing system. J Manuf Syst. 2023;66(2):211–32. doi:10.1016/j.jmsy.2022.12.008.
15. Saophan P, Pannakkong W, Singhaphandu R, Huynh VN. Rapid production rescheduling for flow shop under machine failure disturbance using hybrid perturbation population genetic algorithm-artificial neural networks (PPGA-ANNs). IEEE Access. 2023;11:75794–817. doi:10.1109/access.2023.3294573.
16. Wang J, Liu Y, Ren S, Wang C, Wang W. Evolutionary game based real-time scheduling for energy-efficient distributed and flexible job shop. J Clean Prod. 2021;293(11):126093. doi:10.1016/j.jclepro.2021.126093.
17. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95(12):51–67. doi:10.1016/j.advengsoft.2016.01.008.
18. Uzer MS, Inan O. Application of improved hybrid whale optimization algorithm to optimization problems. Neural Comput Appl. 2023;35(17):12433–51. doi:10.1007/s00521-023-08370-x.
19. Nadimi-Shahraki MH, Zamani H, Asghari Varzaneh Z, Mirjalili S. A systematic review of the whale optimization algorithm: theoretical foundation, improvements, and hybridizations. Arch Comput Meth Eng. 2023;30(7):4113–59. doi:10.1007/s11831-023-09928-7.
20. Wang Z, Li Y, Wu L, Guo Q. A nonlinear adaptive weight-based mutated whale optimization algorithm and its application for solving engineering problems. IEEE Access. 2024;12(4):40225–54. doi:10.1109/access.2024.3350336.
21. Priyadarshi N, Bhaskar MS, Almakhles D. A novel hybrid whale optimization algorithm differential evolution algorithm-based maximum power point tracking employed wind energy conversion systems for water pumping applications: practical realization. IEEE Trans Ind Electron. 2024;71(2):1641–52. doi:10.1109/tie.2023.3260345.
22. Shen Y, Wu J, Ma M, Du X, Wu H, Fei X, et al. Improved differential evolution algorithm based on cooperative multi-population. Eng Appl Artif Intell. 2024;133(1):108149. doi:10.1016/j.engappai.2024.108149.
23. Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput. 2000;8(2):173–95. doi:10.1162/106365600568202.
24. Li Q, Zeng X, Wei W. Multi-objective particle swarm optimization algorithm using Cauchy mutation and improved crowding distance. Int J Intell Comput Cybern. 2023;16(2):250–76. doi:10.1108/ijicc-04-2022-0118.
25. Long Q, Li G, Jiang L. A novel solver for multi-objective optimization: dynamic non-dominated sorting genetic algorithm (DNSGA). Soft Comput. 2022;26(2):725–47. doi:10.1007/s00500-021-06223-0.

26. Liu Y, Wang X, Zhang Y, Liu L. An integrated flow shop scheduling problem of preventive maintenance and degradation with an improved NSGA-II algorithm. IEEE Access. 2023;11:3525–44. doi:10.1109/access.2023.3234428.
27. Ma H, Zhang Y, Sun S, Liu T, Shan Y. A comprehensive survey on NSGA-II for multi-objective optimization and applications. Artif Intell Rev. 2023;56(12):15217–70. doi:10.1007/s10462-023-10526-z.
28. Ghafour K. Multi-objective continuous review inventory policy using MOPSO and TOPSIS methods. Comput Oper Res. 2024;163(1–3):106512. doi:10.1016/j.cor.2023.106512.
29. Li Y, Zhang Y, Hu W. Adaptive multi-objective particle swarm optimization based on virtual Pareto front. Inf Sci. 2023;625:206–36. doi:10.1016/j.ins.2022.12.079.
30. Sassi J, Alaya I, Borne P, Tagina M. A decomposition-based artificial bee colony algorithm for the multi-objective flexible jobshop scheduling problem. Eng Optim. 2022;54(3):524–38. doi:10.1080/0305215X.2021.1884243.
31. Cui Q, Liu P, Du H, Wang H, Ma X. Improved multi-objective artificial bee colony algorithm-based path planning for mobile robots. Front Neurorobot. 2023;17:1196683. doi:10.3389/fnbot.2023.1196683.