

Article

A Q-Learning-Based Adaptive NSGA-II for Fuzzy Distributed Assembly Hybrid Flow Shop Scheduling Problem

Rui Wu ¹, Qiang Li ¹, Bin Cheng ^{2,*}, Yanming Chen ³ and Xixing Li ¹

¹ Hubei Key Laboratory of Modern Manufacturing and Quality Engineering, School of Mechanical Engineering, Hubei University of Technology, Wuhan 430068, China; wurui@hbut.edu.cn (R.W.); 102310094@hbut.edu.cn (Q.L.); li_xi_xing@126.com (X.L.)

² College of Computer and Information Engineering, Wuhan Railway Vocational College of Technology, Wuhan 430074, China

³ Hubei Standardization and Quality Institute, Wuhan 430060, China; 2003511510004@alumni.hust.edu.cn

* Correspondence: cbyeah2020@163.com

Abstract

With the growing emphasis on holistic management throughout the entire product lifecycle, multi-stage production models that integrate distributed manufacturing, transportation, and assembly processes have gradually attracted research attention. However, studies in this area remain relatively scarce. This paper addresses the fuzzy distributed assembly hybrid flow shop scheduling problem (FDAHFSP), comprehensively considering the entire production flow from manufacturing and transportation to final assembly. A mathematical model is first established with the objectives of minimizing the fuzzy total weighted earliness/tardiness and the fuzzy total energy consumption. To effectively solve this problem, a Q-learning-based adaptive NSGA-II (Q-ANSGA) is proposed. The algorithm incorporates a hybrid strategy combining multiple rules to enhance the quality of the initial population. Additionally, a Q-learning-based adaptive parameter adjustment mechanism is designed to dynamically optimize genetic algorithm parameters, thereby improving the algorithm's search efficiency and convergence performance. Furthermore, eight neighborhood search operators are developed, and an iterative greedy strategy is integrated to guide the local search process. Finally, comprehensive experiments on 45 test instances are conducted to evaluate the effectiveness of each improvement component and the overall performance of Q-ANSGA. Experimental results demonstrate that the proposed algorithm achieves superior performance in solving the FDAHFSP due to its systematic enhancements.

Keywords: distributed assembly hybrid flow shop; fuzzy scheduling; weighted earliness/tardiness; NSGA-II



Academic Editor: Michael C. Georgiadis

Received: 7 January 2026

Revised: 28 January 2026

Accepted: 29 January 2026

Published: 31 January 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Due to the increasing refinement of product manufacturing and diversification of consumer demands, enterprises find it increasingly difficult to rely solely on single-factory production [1]. In contrast, distributed manufacturing, with its lower risk costs and faster market responsiveness [2,3], has become a crucial pathway for the high-quality development of the manufacturing sector. As one of the most common types of shop scheduling, the distributed hybrid flow shop (DHFSP) is widely applied in semiconductors, textiles, and electronics [4]. In actual production, transportation and assembly are closely linked with the production phase, collectively forming a complete product manufacturing process. Additionally, real-world production involves many unavoidable uncertain factors [5], such

as variations in workers' skill levels, which can cause the actual processing time of jobs to deviate from the planned schedule [6]. To effectively address such uncertainties, the concept of fuzzy time has been introduced into general shop scheduling problems. This approach uses interval values to describe the uncertainties present in the production system. Based on the aforementioned practical significance and important value, this paper investigates the fuzzy distributed assembly hybrid flow shop scheduling problem (FDAHFSP).

As global climate change becomes increasingly severe, the concept of environmental protection has gained widespread acceptance. Energy conservation and emission reduction are not only an individual responsibility but also a long-term development goal that enterprises should adhere to. Over the past few decades, the energy consumption of the manufacturing sector has remained at a high level [7]. Driven by green manufacturing initiatives, more and more companies are actively seeking effective energy-saving approaches. However, developing energy-efficient equipment and products often requires substantial research and development investment, which poses significant risks for most enterprises. Therefore, starting from production scheduling and reducing energy consumption through optimized production planning represents a more feasible path. It is worth noting that while optimizing energy consumption, it often affects conflicting objectives such as earliness/tardiness. For enterprises, on-time delivery is equally critical, as failure to meet delivery deadlines may lead to loss of market share and economic losses. Therefore, this paper takes earliness/tardiness and energy consumption as the optimization objectives to rationally schedule the product manufacturing process.

Meta-heuristic algorithms have been demonstrated to perform effectively in solving shop scheduling problems, with common examples including particle swarm optimization [8], artificial bee colony algorithms [9], and genetic algorithms [10]. Considering that different algorithms possess distinct strengths and limitations, a single algorithm is often insufficient for all shop scheduling scenarios [11]. Therefore, integrating the advantages of specific algorithms is of significant importance for addressing the studied scheduling problems. Among these, the NSGA-II has gained widespread application in combinatorial optimization problems, thanks to its fast non-dominated sorting mechanism that reduces computational complexity and its elitist preservation strategy, which prevents the loss of high-quality individuals during evolution. For this purpose, we adopt NSGA-II as the fundamental algorithmic framework and introduces a series of enhancements based on it.

Reinforcement learning is an important branch of machine learning [12], whose core mechanism lies in the agent's interaction with the environment. Through trial-and-error learning, the agent progressively refines its decision-making policy, ultimately maximizing long-term cumulative reward [13]. In recent years, reinforcement learning has shown outstanding performance in adaptive adjustment of key parameters [14] and dynamic improvement of search strategies [15], leading to widespread attention on its integration with meta-heuristic algorithms. In terms of adaptive parameter acquisition, the Q-learning agent in reinforcement learning can perceive the state features of the algorithm in real time and dynamically adjust parameter values according to the current optimization requirements. To maintain key algorithm parameters in a near-optimal state, Q-learning can be integrated with the global search process of NSGA-II to achieve adaptive parameter optimization. Therefore, this paper introduces Q-learning to guide the algorithm in adaptively selecting crossover and mutation probabilities.

Based on the above analysis, this paper proposes a Q-learning-based adaptive non-dominated sorting genetic algorithm (Q-ANSGA) to solve the FDAHFSP. The main contributions of this work are as follows:

- (1) A FDAHFSP that integrates processing, transportation, and assembly stages is modeled using a mixed-integer linear programming formulation, with objectives to simul-

taneously minimize total fuzzy weighted earliness/tardiness ($\widetilde{TWET}$) and total fuzzy energy consumption ($\widetilde{TEC}$).

- (2) A tailored Q-ANSGA is developed, featuring: (i) a hybrid initialization strategy that combines multiple heuristic rules to generate high-quality initial populations; (ii) an adaptive parameter selection mechanism driven by Q-learning to dynamically adjust crossover and mutation probabilities; (iii) eight problem-specific neighborhood search operators selected via an iterative-greedy strategy to enhance local search efficiency.
- (3) Extensive experiments are conducted, including ablation studies and comparisons with state-of-the-art algorithms. These experiments validate that each component of Q-ANSGA effectively improves optimization performance and demonstrate the overall superiority of the proposed method in solving the FDAHFSP.

The subsequent arrangements are as follows. Section 2 summarizes related problems and existing methods. Section 3 provides the problem description and model formulation. Section 4 elaborates on the algorithmic framework and strategy design of Q-ANSGA. Section 5 validates the effectiveness and superiority of the algorithm through experiments. Section 6 presents the conclusions and future work.

2. Literature Review

2.1. Related Scheduling Problems

The DHFSP constitutes a complex system formed by multiple homogeneous HFSPs, whose solution necessitates the coordinated treatment of three coupled sub-problems: factory assignment, job sequencing, and machine selection [16]. Lu et al. [17] first studied the energy-aware DHFSP and designed a hybrid multi-objective iterated greedy method. Shao et al. [18] proposed an evolutionary algorithm based on multi-neighborhood local search to solve the DHFSP. However, the DAHFSP investigated in this paper further incorporates transportation and assembly stages on the basis of DHFSP, making it more aligned with increasingly complex manufacturing patterns. In this regard, Wang et al. [19] proposed a Q-learning-based artificial bee colony algorithm to address DAHFSP under constraints of factory capacity, transportation resources, and setup time. Luo et al. [20] designed an improved evolutionary algorithm for DAHFSP, considering transportation resources, aiming to simultaneously minimize makespan and energy consumption. Furthermore, the integration of fuzzy theory and shop scheduling has garnered widespread attention in academia, with related studies covering various scheduling scenarios such as fuzzy flexible job shop [21,22], fuzzy hybrid flow shop [23,24], and fuzzy distributed hybrid flow shop [25,26]. In combination with assembly shop scheduling, Zhao et al. [27] proposed an iterative greedy algorithm combined with Q-learning to solve the fuzzy distributed assembly flexible job shop scheduling problem.

Optimization objectives are key factors that determine the direction of scheduling problems. For multi-objective scheduling problems, the selection of objectives should not only exhibit coupling relationships but also align with practical needs. Energy consumption, as an important consideration, is frequently incorporated into scheduling problems. Wang et al. [28] designed a speed-adjustment strategy to regulate machine speeds for balancing energy consumption and makespan. Shi et al. [29] constructed a dynamic shop scheduling model aiming to optimize four objectives: energy consumption, makespan, job tardiness, and machine idle time. Yao et al. [30] proposed a deep Q-network-based memetic algorithm to achieve the minimization of energy consumption and makespan. Moreover, since earliness/tardiness is related to both cost and customer satisfaction, it has also drawn considerable attention in academia. Zhu et al. [31] introduced a discrete fruit fly algorithm for the distributed no-wait flow shop scheduling problem to minimize

weighted earliness/tardiness. Yu et al. [32] proposed a dual-learning-based Jaya algorithm to simultaneously minimize makespan, mean earliness/tardiness, and carbon emissions.

2.2. Related Scheduling Algorithms

In response to diverse and complex shop scheduling problems, the academic community has proposed a variety of distinctive methods. Among them, meta-heuristic algorithms have become indispensable and commonly used tools due to their strong adaptability, excellent search capability, and good convergence performance. However, as research on shop scheduling continues to deepen, conventional meta-heuristics have gradually become insufficient to cope with increasingly complex scheduling demands. Consequently, researchers have begun to focus on targeted improvements of algorithms. Lu et al. [33] designed a hybrid iterated greedy algorithm to address the DHFSP. Lei and Su [34] proposed an algorithm based on a multi-class teaching-learning framework to simultaneously minimize makespan and tardiness. Given the complexity of the FDAHFSP investigated in this paper, the widely adopted NSGA-II for multi-objective combinatorial optimization is employed. Chen et al. [35] developed a knowledge-based NSGA-II that effectively addresses hot-rolling production scheduling by means of an adaptive recombination procedure and a path-related local reinforcement procedure. Lv et al. [36] proposed an NSGA-II algorithm with improved local search for solving production and inventory scheduling problems. Wang et al. [37] designed a correlation entropy-based NSGA-II to tackle the HFSP with variable-speed machines. In recent years, the introduction of reinforcement learning has elevated meta-heuristic algorithms to a new level. The combination of the two further enhances solution efficiency. Zhang et al. [38] proposed a hyper-heuristic evolutionary algorithm based on Q-learning, which employs a reinforcement learning mechanism to drive low-level operators for exploring the solution space. Chen et al. [39] designed a hybrid method that combines an immune algorithm and Q-learning to dynamically select local search strategies.

2.3. Research Gaps

A careful review of the existing literature reveals several unresolved research gaps. From the problem perspective, most studies concentrate on single-stage or conventional shop scheduling problems, while the three-stage FDAHFSP, which jointly considers processing, transportation, and assembly, has received limited attention. In particular, the integration of fuzzy uncertainty into DAHFSP remains insufficiently explored. From the methodological perspective, many existing evolutionary approaches rely on fixed algorithmic parameters, such as crossover and mutation probabilities, typically determined through offline tuning. However, such static settings are inadequate for addressing the highly dynamic and multi-stage nature of FDAHFSP. To bridge these gaps, this study investigates a fuzzy three-stage distributed assembly scheduling framework and develops a reinforcement learning assisted adaptive NSGA-II, enabling dynamic parameter adjustment to better balance exploration and exploitation throughout the optimization process.

3. Problem Statement and Modeling

3.1. Problem Definition

As illustrated in Figure 1, the DAHFSP consists of three main production stages. The first stage is the processing stage, in which the jobs of a batch of products are assigned to different factories F for processing. Each product p is composed of multiple jobs, and each factory can be regarded as a hybrid flow shop. The jobs are processed sequentially according to a predefined order of operations, passing through j operations. Each operation is equipped with m machines. The second stage is the transportation stage, where once

the processing of jobs is completed, transportation vehicles e deliver them to the assembly factory. The final stage is the assembly stage, which begins once all required jobs for a product have arrived and the assembly machines are available. It is noteworthy that the time of all stages is modeled as fuzzy variables to better capture the uncertainty inherent in real-world production systems. Moreover, each product is constrained by a fuzzy earliness time $\tilde{\alpha}_p$ and a fuzzy due time $\tilde{\beta}_p$. Products should be delivered within the interval $[\tilde{\alpha}_p, \tilde{\beta}_p]$; otherwise, earliness or tardiness penalties will be incurred.

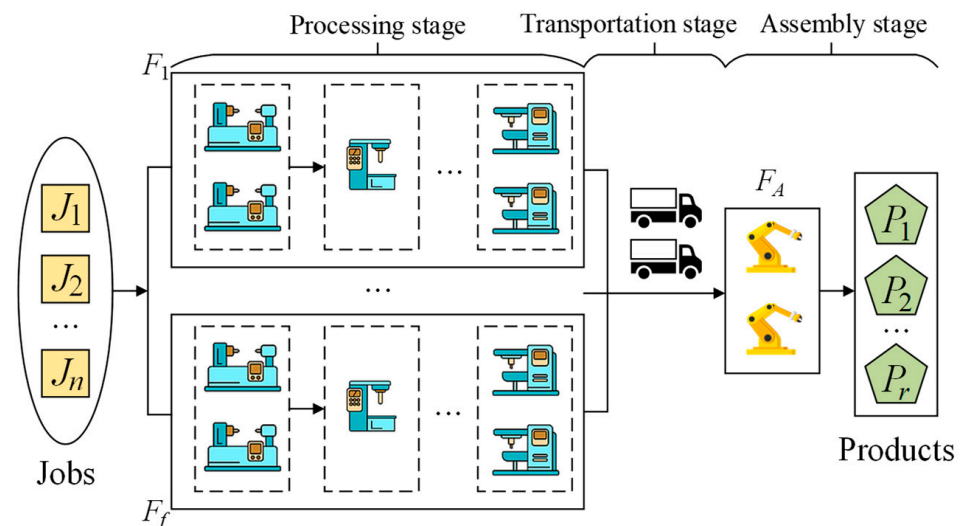


Figure 1. The framework of DAHFSP.

3.2. Problem Model

The formulation of the problem model is based on several assumptions: all jobs and machines are available at zero time; the job cannot change its assigned factory; a transportation vehicle can carry only one job at any given time; all jobs are processed, transported, and assembled without interruption; all components of a product must be fully processed before assembly begins; a machine can only process at most one job or product at the same time, and the job can only be processed on one machine at the same time. The meaning of each symbol is shown in Table 1.

Table 1. Notations and descriptions.

Notations	Descriptions
p, p'	Indexes of products
i, i'	Indexes of jobs
f, s	Indexes of factories and operations
k, e, g	Indexes of machines, transport vehicles and assembly machines
M	Processing machines, $M = \{M_{f,s,1}, M_{f,s,2}, \dots, M_{f,s,k}\}$
AM	Assembly machines, $AM = \{AM_1, AM_2, \dots, AM_d\}$
r	Number of products
n	Number of jobs
h	Number of jobs in product p
w	Number of factories
j	Number of stages
m	Number of machines for operation in factory
v	Number of transport vehicles
d	Number of assembly machines in the assembly shop

Table 1. Cont.

Notations	Descriptions
$M_{f,s,k}$	The k -th machine of operation s in factory f
$\tilde{P}_{i,f,s,k}$	The fuzzy processing time of job i on machine k of operation s in factory f
$\tilde{T}_{i,e}$	The fuzzy transportation time of job i delivered by transport vehicle e
$\tilde{A}_{p,g}$	The fuzzy assembly time of product p
W_p^E	The unit earliness penalty of product p
W_p^T	The unit tardiness penalty of product p
$EO_{f,s,k}$	The processing energy consumption per unit time of machine k for operation s in factory f
$EI_{f,s,k}$	The idle energy consumption per unit time of machine k for operation s in factory f
$\tilde{ST}_{i,s}$	The fuzzy starting time of operation s in job i
$\tilde{FT}_{i,s}$	The fuzzy finish time of operation s in job i
$\tilde{TST}_i$	The fuzzy starting time of transportation for job i
$\tilde{TFT}_i$	The fuzzy finish time of transportation for job i
$\tilde{AST}_p$	The fuzzy starting time of assembly for product p
$\tilde{AFT}_p$	The fuzzy finish time of assembly for product p
$\tilde{CT}_i$	The fuzzy completion time of job i
$\tilde{TCT}_i$	The fuzzy completion time of transportation for job i
$\tilde{ACT}_p$	The fuzzy completion time of assembly for product p
$\tilde{L}$	$\tilde{L} = (L_1, L_2, L_3)$, where L_1, L_2, L_3 are very large positive numbers
$X_{i,f}$	If job i is assigned to factory f , then $X_{i,f} = 1$, otherwise, $X_{i,f} = 0$
$Y_{i,f,s,k}$	If job i of operation s is processed on machine k in factory f , then $Y_{i,f,s,k} = 1$, otherwise, $Y_{i,f,s,k} = 0$
$Z_{i,i',f,s}$	If job i is processed before job i' in operation s of factory f , then $Z_{i,i',f,s} = 1$, otherwise, $Z_{i,i',f,s} = 0$
$N_{i,f}$	If job i of factory f is scheduled to be transported, then $N_{i,f} = 1$, otherwise, $N_{i,f} = 0$
$Q_{i,i',f}$	If product i of factory f is transported before the product i' , then $Q_{i,i',f} = 1$, otherwise, $Q_{i,i',f} = 0$
U_p	If product p is arranged for assembly, then $U_p = 1$, otherwise, $U_p = 0$
$V_{p,p'}$	If product p is assembled before product p' , then $V_{p,p'} = 1$, otherwise, $V_{p,p'} = 0$

This study aims to minimize the $\widetilde{TWET}$ and $\widetilde{TEC}$. Based on the aforementioned mathematical notations and assumptions, the mathematical model is formulated.

Equation (1) defines the expression of $\widetilde{TWET}$, which consists of two components: the weighted earliness and the weighted tardiness.

$$\widetilde{TWET} = \sum_{p=1}^r W_p^E \times \{(\widetilde{\alpha}_p - \widetilde{AFT}_p) \vee (0, 0, 0)\} + \sum_{p=1}^r W_p^T \times \{(\widetilde{AFT}_p - \widetilde{\beta}_p) \vee (0, 0, 0)\} \quad (1)$$

Equations (2)–(5) define the expression of $\widetilde{TEC}$. The energy consumption considered in this study encompasses machine processing energy consumption, machine idle energy consumption, and vehicle transportation energy consumption. Specifically, $\widetilde{EC}_1$, $\widetilde{EC}_2$, $\widetilde{EC}_3$ denote the energy consumption from the processing, transportation, and assembly phases, respectively.

$$\widetilde{TEC} = \widetilde{EC}_1 + \widetilde{EC}_2 + \widetilde{EC}_3 \quad (2)$$

$$\widetilde{EC}_1 = \sum_{f=1}^w \sum_{s=1}^{j-2} \sum_{k=1}^m \left(\sum_{i=1}^n Y_{i,f,s,k} \times \tilde{P}_{i,f,s,k} (EO_{f,s,k} - EI_{f,s,k}) + Y_{i,f,s,k} \times EI_{f,s,k} \times \max_i \{\widetilde{CT}_i\} \right) \quad (3)$$

$$EC_2 = \sum_{f=1}^w \sum_{e=1}^v \sum_{i=1}^n N_{i,f} \times \tilde{T}_i (EO_{f,j-1,e} - EI_{f,j-1,e}) \quad (4)$$

$$EC_3 = \sum_{g=1}^d \left(\sum_{p=1}^r U_p \times \widetilde{A}_p (EO_{f,j,g} - EI_{f,j,g}) + U_p \times EI_{f,j,g} \times (\max_p \{\widetilde{ACT}_p\} - \max_p \{\widetilde{TCT}_p\}) \right) \quad (5)$$

Subject to

$$\sum_{f=1}^w X_{i,f} = 1, \forall i \quad (6)$$

$$\sum_{k=1}^m Y_{i,f,s,k} = X_{i,f}, \forall i, f, s \quad (7)$$

$$\widetilde{ST}_{i,1} \geq \{0, 0, 0\}, \forall i \quad (8)$$

$$\widetilde{ST}_{i,s+1} \geq \widetilde{FT}_{i,s}, \forall i, s \quad (9)$$

$$\widetilde{FT}_{i,s} = \widetilde{ST}_{i,s} + \sum_{f=1}^w \sum_{k=1}^m \widetilde{P}_{i,f,s,k}, \forall i, s \quad (10)$$

$$\widetilde{TST}_i \geq \max_i \{\widetilde{CT}_i\}, \forall p \quad (11)$$

$$\widetilde{TFT}_i = \widetilde{TST}_i + \sum_{e=1}^v \widetilde{T}_{i,e}, \forall p \quad (12)$$

$$\widetilde{AST}_p \geq \widetilde{AFT}_p, \forall p \quad (13)$$

$$\widetilde{AFT}_p = \widetilde{AST}_p + \sum_{g=1}^d \widetilde{A}_{p,g}, \forall p \quad (14)$$

$$Z_{i,i',f,s} + Z_{i',i,f,s} \leq 1, \forall i, i', f, s \quad (15)$$

$$Z_{i,i',f,s} + Z_{i',i,f,s} \geq Y_{i,f,s,k} + Y_{i',f,s,k} - 1, \forall i, i', f, s \quad (16)$$

$$\widetilde{ST}_{i',k} \geq \widetilde{FT}_{i,k} - \widetilde{L} \times (3 - Y_{i,f,s,k} - Y_{i',f,s,k} - Z_{i,i',f,s}), i \neq i', f, s, k \quad (17)$$

$$Q_{i,i',f} + Q_{i',i,f} \leq 1, \forall i, i', f \quad (18)$$

$$Q_{i,i',f} + Q_{i',i,f} \geq N_{i,f} + N_{i',f} - 1, \forall i, i', f \quad (19)$$

$$\widetilde{TST}_{i'} \geq \widetilde{TFT}_i - \widetilde{L} \times (3 - N_{i,f} - N_{i',f} - Q_{i,i',f}), \forall i, i', f \quad (20)$$

$$V_{p,p'} + V_{p',p} \leq 1, \forall p, p' \quad (21)$$

$$V_{p,p'} + V_{p',p} \geq U_p + U_{p'} - 1, \forall p, p' \quad (22)$$

$$\widetilde{AST}_{p'} \geq \widetilde{AFT}_p - \widetilde{L} \times (3 - N_p - N_{p'} - V_{p,p'}), \forall p, p' \quad (23)$$

Equations (6)–(8) specify the fundamental constraints of the scheduling process, including the assignment of each job to a specific factory, the requirement that all processing operations must be completed, and the restriction that processing cannot start before time zero. Equation (9) enforces the technological precedence among operations by ensuring that each operation starts only after the completion of its preceding operation, while Equation (10) further requires that job processing is non-preemptive once initiated. The interdependence between processing, transportation, and assembly stages is captured in Equations (11)–(14), which stipulate that each stage is carried out in turn. Assembly can only begin after all components of the product are delivered, and all processes cannot be interrupted. Finally, Equations (15)–(17), (18)–(20), and (21)–(23) impose capacity constraints

on processing machines, transportation vehicles, and assembly machines, respectively, ensuring that each resource can handle at most one job or product at any given time. The use of $\tilde{L}$ ensures that this constraint is satisfied for all jobs.

3.3. Operations on Fuzzy Numbers

To fully decode the entire scheduling process, it is essential to establish appropriate arithmetic operators for fuzzy time. This paper employs triangular fuzzy numbers (TFNs) to represent fuzzy time [40]. Unlike deterministic models with fixed processing times, triangular fuzzy numbers capture processing time uncertainty and its impact on scheduling decisions. Scheduling is therefore influenced not only by the most likely duration but also by possible deviations, making operations with higher uncertainty more critical. As a result, fuzziness leads to more robust and risk-aware scheduling solutions. Given two TFNs $\tilde{X} = (x_1, x_2, x_3)$ and $\tilde{Y} = (y_1, y_2, y_3)$, the specific arithmetic operators are as follows:

- (1) Addition/subtraction operator: $\tilde{X} \pm \tilde{Y} = (x_1 \pm y_1, x_2 \pm y_2, x_3 \pm y_3)$.
- (2) Ranking operation:
 - (i) If $f_1(\tilde{X}) = (x_1 + 2x_2 + x_3)/4 > f_1(\tilde{Y}) = (y_1 + 2y_2 + y_3)/4$, then $\tilde{X} > \tilde{Y}$;
 - (ii) If $f_1(\tilde{X}) = f_1(\tilde{Y})$ and $f_2(\tilde{X}) = x_2 > f_2(\tilde{Y}) = y_2$, then $\tilde{X} > \tilde{Y}$;
 - (iii) If $f_1(\tilde{X}) = f_1(\tilde{Y})$, $f_2(\tilde{X}) = f_2(\tilde{Y})$ and $f_3(\tilde{X}) = x_3 - x_1 > f_3(\tilde{Y}) = y_3 - y_1$, then $\tilde{X} > \tilde{Y}$.
- (3) Max operation: If $\tilde{X} > \tilde{Y}$, then $\tilde{X} \vee \tilde{Y} = \tilde{X}$; otherwise, $\tilde{X} \vee \tilde{Y} = \tilde{Y}$.
- (4) Since scheduling decisions are made under fuzzy representations, the fuzzy values must be converted into precise ones when evaluating the objective functions. Therefore, a defuzzification method needs to be designed to ensure the integrity of the entire scheduling process. The specific defuzzification formula is given as follows:

$$D_1(\tilde{X}) = \frac{x_1 + 4x_2 + x_3}{6} \quad (24)$$

4. Presented Algorithm

Considering the widespread application of evolutionary algorithms in the scheduling domain and their demonstrated excellent performance, this paper proposes a Q-ANSGA to solve the problem under study. The framework of the algorithm is shown in Figure 2. Hybrid initialization injects diversity and quality into the initial population, providing a promising starting region for the search. The Q-learning continuously adjusts crossover and mutation probabilities in response to the current search state, thereby balancing global exploration and local refinement dynamically. The eight neighborhood operators enable structured local moves tailored to the problem's constraints, while the iterative-greedy selection ensures that the most effective operators are applied more frequently without prematurely discussing less active ones.

Crucially, these components are not merely stacked; they interact synergistically. For instance, the adaptive parameter control ensures that the population maintains sufficient diversity when entering neighborhood search, while the greedy operator selection leverages the Q-learning agent's state-awareness to prioritize operators that best match the current convergence stage. This cohesive design avoids the inefficiencies of isolated mechanisms and has been shown experimentally to produce more stable and closer to true Pareto solutions.

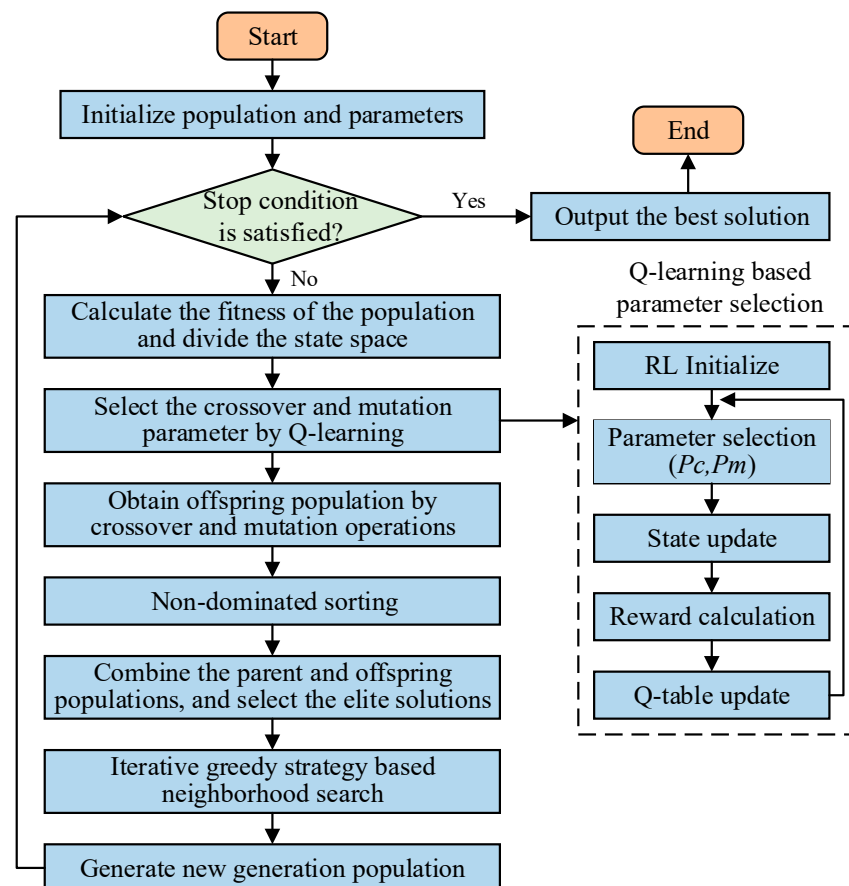


Figure 2. The framework of Q-ANSGA.

4.1. Encoding and Decoding

This study employs a three-layer structure, including the factory layer FS , product layer PS , and job layer JS . The lengths of all three layers are identical and equal to the n . Assume that there are two factories, four products, and ten jobs, where the composition of each product is as follows: $P_1 = \{J_2, J_4, J_5\}$, $P_2 = \{J_1, J_6\}$, $P_3 = \{J_8, J_9\}$, and $P_4 = \{J_3, J_7, J_{10}\}$. If the job sequence layer is encoded as $\{9, 3, 8, 4, 6, 7, 1, 2, 5, 10\}$, the corresponding product sequence layer can be determined as $\{3, 4, 3, 1, 2, 4, 2, 1, 1, 4\}$. As illustrated in Figure 3, Factory 1 is responsible for processing jobs $\{9, 3, 4, 1, 10\}$, while the remaining jobs are processed in Factory 2.

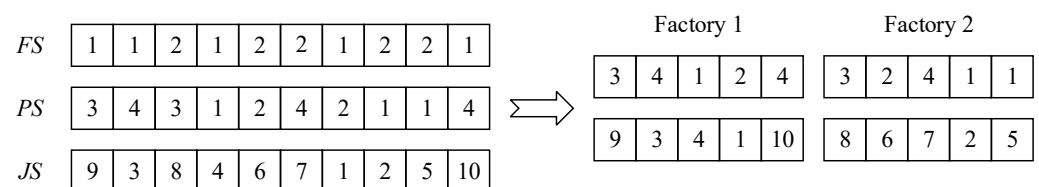


Figure 3. An example of encoding and decoding.

A forward decoding strategy converts encoded information into a feasible schedule. As shown in Figure 4, the subscripts in the figure indicate the job number and its current operation, respectively. For example, J_{2-1} represents the first operation of job 2. Taking Factory 1 as an example, the job order in the encoding determines the processing sequence of the first operation. Subsequently, jobs that finish earlier are prioritized for their next operation. For machine selection, an available machine is assigned when the job is ready; if multiple machines are idle, the one with lower unit energy consumption is preferred.

Transportation vehicles are allocated similarly after processing. Finally, the assembly of a product begins only after all its required jobs have arrived at the assembly shop, at which point an available assembly machine is assigned.

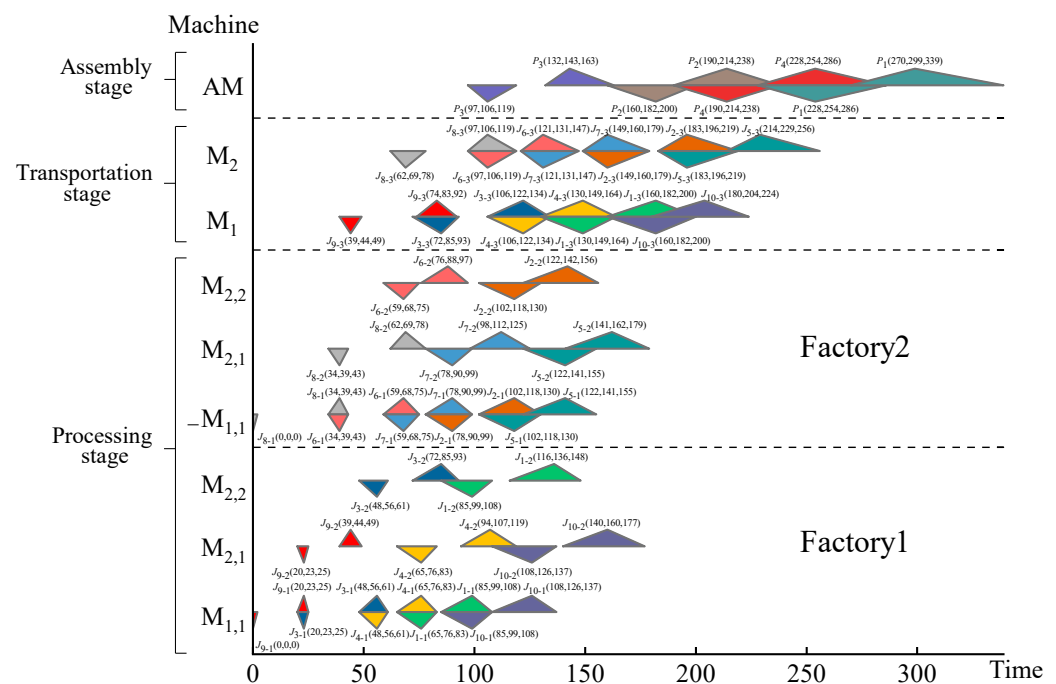


Figure 4. Gantt charts of schedules by forward decoding methods.

The example Gantt chart in Figure 4 verifies both the effectiveness of the encoding decoding scheme and the correctness of the mathematical model in this paper. Analysis of the scheduling solution shown in the figure indicates that the job processing sequence and machine assignment conform to the problem description and are consistent with the actual solution logic.

4.2. Population Initialization

Considering that randomly initialized populations exhibit high uncertainty and often fail to ensure satisfactory initial solution quality, this study designs three heuristic rules to generate an excellent initial population. Meanwhile, to preserve population diversity, individuals generated through these heuristic rules constitute only half of the population size (ps). Specifically, the entire population is divided into four equal parts, among which randomly initialized individuals occupy $ps/2$, while the two heuristic initialization rules each contribute $ps/4$.

(1) Job allocation rule

To facilitate early entry into the final assembly stage, it is crucial to process and transport components of the same product within similar time intervals. Therefore, during factory allocation, components belonging to the same product are preferentially assigned to different factories to enable parallel processing and transportation. This strategy effectively reduces product cycle times and enhances overall system efficiency.

(2) Priority scheduling rule

Since the delivery time window of each product is known in advance, products with earlier delivery dates should be prioritized for processing to ensure on-time delivery. By prioritizing the scheduling of products with earlier delivery requirements, this rule

effectively reduces the likelihood of both earliness and tardiness. Consequently, it enhances the timeliness and stability of the overall scheduling plan.

4.3. Genetic Operation

Genetic operators are commonly employed to achieve global search capability within evolutionary algorithms. By adopting a tournament selection mechanism, the algorithm effectively preserves high-quality individuals, thereby maintaining the overall excellence of the population. To enhance the exploratory capability in the solution space and generate more diverse offspring, this study utilizes single-point crossover and mutation operator [41], as detailed below:

The single-point crossover operator primarily acts on JS . It randomly selects an adjacent gene position within a parent individual, copies all genes before this position directly into the offspring. The remaining genes are then filled sequentially following the original order of the other parent individual. This approach maintains structural feasibility while promoting population diversity. The single-point mutation operator operates on FS by randomly selecting a gene and modifying its value within a feasible range. Figure 5 shows an example of a genetic operation.

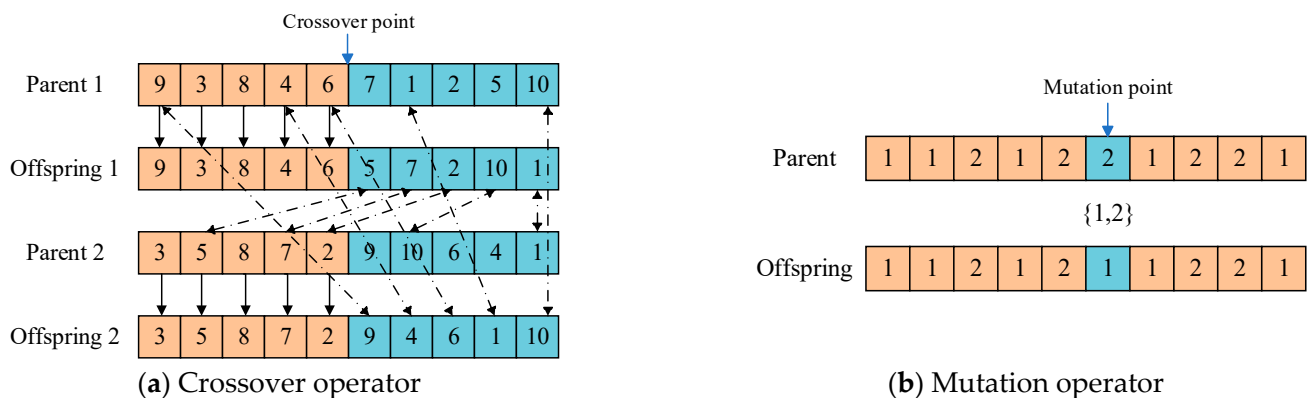


Figure 5. Example of the genetic operators.

4.4. Adaptive Parameter Selection Based on Q-Learning

In the global search process, the crossover rate p_c and mutation rate p_m jointly regulate the trade-off between exploration and exploitation, directly determining convergence efficiency, solution quality, and the ability to attain global extrema. During the initial phase of the algorithm, larger p_c and p_m should be adopted to enable extensive exploration of the solution space and to improve population structure. As the algorithm progresses to later stages, these probabilities should be gradually reduced to focus search attention on high-potential solution regions, thereby enhancing convergence precision and speed [42].

To achieve adaptive selection of crossover and mutation probabilities, this paper introduces the Q-learning algorithm. This paper adopts tabular Q-learning as the adaptive control mechanism, mainly for the following reasons. First, the state space of the problem is a discrete finite set based on regions in the objective space and population diversity indicators. The tabular approach maintains efficiency while avoiding the complexity of function approximation. Second, compared to conventional adaptive methods, the sequential decision-making framework of Q-learning aligns well with the iterative evolutionary process of NSGA-II, enabling operator selection in each generation based on the current state and long-term policy optimization through reward feedback. The update method of the Q-table is shown in Equation (25):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \quad (25)$$

where $Q(s_t, a_t)$ denotes the Q-value corresponding to executing the action a_t in state s_t ; α and γ represent the learning rate and discount factor, respectively, both ranging between 0 and 1; r_{t+1} indicates the reward value obtained after performing action a_t , and $\max Q(s_{t+1}, a_{t+1})$ signifies the maximum Q-value among possible actions in the next state s_{t+1} . The definitions of the key components in the Q-learning algorithm are as follows:

State: The state is constructed directly from the two conflicting optimization objectives— $\widehat{TWET}$ and $\widehat{TEC}$ [43]. For each objective, the maximum, median, and minimum fitness values of the current population are calculated. These six reference points partition the objective space into four regions (φ_1 – φ_4), as shown in Figure 6. The agent's state corresponds to the region in which the current individual resides (e.g., $s = s_1$ for φ_1). This discrete representation directly links environmental feedback to the multi-objective trade-off landscape, enabling focused operator selection during evolution.

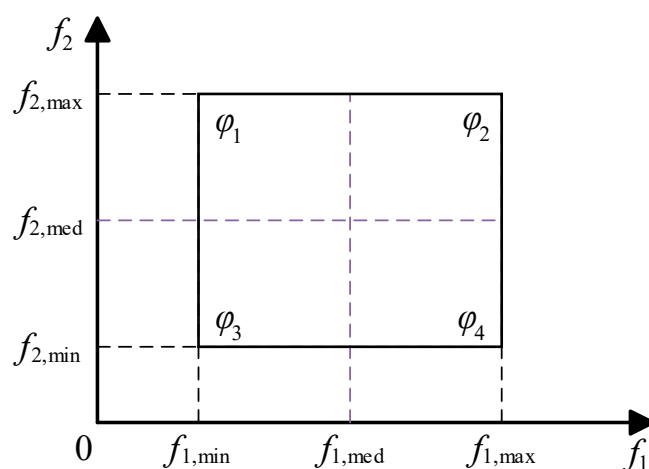


Figure 6. Diagram of state division.

Action: To enable adaptive selection of p_c and p_m , this paper defines a candidate set $\{0.6, 0.7, 0.8, 0.9\}$ for p_c , and a candidate set $\{0.1, 0.2, 0.3, 0.4\}$ for p_m . Each action is formed by combining one value from p_c and one from p_m , resulting in 16 possible action combinations.

Reward: The reward mechanism is initialized to 0, and subsequent cumulative rewards are assigned according to the dominance relationship among solutions, as specified by Equation (26) [44]:

$$R = \begin{cases} 2 + \sum_{c=1}^{obj} \frac{f_{\max}^c - f^c}{f_{\max}^c - f_{\min}^c}, & \text{if } x' \text{ dominates } x \\ 0, & \text{if } x \text{ dominates } x' \\ 1 + \sum_{c=1}^{obj} \frac{f_{\max}^c - f^c}{f_{\max}^c - f_{\min}^c}, & \text{otherwise} \end{cases} \quad (26)$$

where R is the reward value, x' is the new solution, x is the old solution, and $f_{\max}^c, f_{\min}^c, f^c$ represent the maximum, minimum and current value of the objective, respectively. When a better solution is obtained, the reward is accumulated according to the degree of improvement; otherwise, it is correspondingly reduced. This approach effectively enhances the quality and discriminability of the learning signal.

4.5. Neighborhood Search Operators

To prevent the algorithm from converging to local optima, well-designed neighborhood operators are introduced to explore promising regions. Based on key factories and key products, eight neighborhood structures are proposed, all of which enhance solution

quality by adjusting job sequences. Here, the key factory is defined as the one with the highest energy consumption, and the key product as the one with the largest weighted earliness/tardiness. The eight operators are outlined below:

- (1) NS1 (Intra-factory swap): Randomly select a job from the key factory and swap it sequentially with other jobs in the same factory until improvement.
- (2) NS2 (Intra-factory insertion): Randomly select a job from the key factory and insert it before or after other jobs in the same factory until improvement.
- (3) NS3 (Inter-factory swap): Randomly select a job from the key factory and swap it with jobs from a non-key factory until improvement.
- (4) NS4 (Inter-factory insertion): Randomly select a job from the key factory and insert it into feasible positions in a non-key factory until improvement.
- (5) NS5 (Intra-product swap): After identifying the key product, randomly select a job from it and swap with other jobs of the same product until improvement.
- (6) NS6 (Intra-product insertion): Randomly select a job from the key product and insert it among other jobs of the same product until improvement.
- (7) NS7 (Inter-product swap): Randomly select a job from the key product and swap it with a job from another product until improvement.
- (8) NS8 (Inter-product insertion): Randomly select a job from the key product and insert it adjacent to a job of another product until improvement.

4.6. Iterative Greedy Based Neighborhood Structures Selection

When facing multiple neighborhood search operators, the proper selection of suitable operators is crucial for enhancing algorithmic performance. This study adopts a neighborhood search operator selection mechanism based on an iterative greedy strategy (IG), the detailed implementation of which is as follows:

Assume that each neighborhood search operator initially has the same selection probability, denoted as $SP_{1,q}, SP_{2,q}, \dots, SP_{8,q}$, where q represents the number of times the operator has been selected. To prevent one or a few operators from dominating the selection process for an extended period, a random value *num* is uniformly generated from $[0, 1]$ in each iteration and compared with a greedy probability $\varepsilon = 0.2$. If $num < \varepsilon$, a neighborhood search operator is selected randomly; otherwise, the operator is chosen according to the current probabilities SP . The probability of each operator is dynamically updated as the search progresses. As shown in Equation (27), suppose that operator NS1 is selected in the current iteration. The update rule for its counter variable l_1 is defined as follows: When a fully dominating superior solution is generated, l_1 increases by 1; when an incompletely dominating new solution is generated, l_1 increases by 0.5; and when the generated solution cannot dominate any other solution, l_1 remains unchanged. The update mechanisms for the remaining operators are analogous, and all counter variables are initialized to 1.

$$SP_{1,q} = \frac{l_1}{l_1 + l_2 + \dots + l_8} \quad (27)$$

IG-based neighborhood structures selection strategy effectively balances exploration and exploitation. It not only prevents the algorithm from becoming trapped in local optima but also allows promising operators to play a greater role in subsequent iterations, thereby improving the overall search efficiency and solution quality.

5. Experiments and Discussion

All algorithms experimented in this section were implemented by Python 3.9. The programs were executed on Windows 11, utilizing an AMD Ryzen 7 7840H CPU at 3.8 GHz, with 16 GB of RAM.

5.1. Experimental Instance Settings

Based on the benchmark by Hatami et al. [45] and the experimental setup of Yu et al. [46], 45 test instances are generated, including 27 small-to-medium and 18 large-scale cases. For varying scales, the number of operations $s \in \{2, 4, 6\}$ and factories $f \in \{3, 4, 5\}$. Small-to-medium instances have job counts $n \in \{20, 50, 100\}$ and product types $r = 5$; large instances use $n \in \{150, 200\}$ and $r = 10$. Each instance is labeled as “ $n_r_s_f$ ” (e.g., “20_5_2_3” denotes 20 jobs, 5 products, 2 operations, and 3 factories). Jobs per product are randomly assigned, while machines and transport vehicles are randomly chosen from $\{2, 3\}$. To reflect fuzzy time parameters, lower and upper bounds of time-related data are scaled by factors drawn uniformly from $[0.8, 1]$ and $[1, 1.2]$, respectively. Unit-time processing and idle energy rates of machines are uniformly sampled from $[0.5, 1]$ and $[0.2, 0.4]$; transport vehicle energy consumption is taken from $[1, 1.5]$. Product due dates are set reasonably according to total processing time. Penalty weights for earliness and tardiness are uniformly generated from $[1, 3]$ and $[4, 10]$, respectively, reflecting their asymmetric impact.

It should be noted that, as this paper deals with a multi-objective optimization problem, the quality of a solution cannot be judged based on a single objective. Therefore, the experimental results are evaluated using two metrics: hypervolume (HV) and inverted generational distance (IGD). HV is employed to comprehensively assess the algorithm’s performance, reflecting both the convergence and diversity of the solution set; IGD measures how closely the algorithm’s solution set approximates the true Pareto front, thereby indicating the quality of the solutions.

5.2. Parameter Calibration

Since parameters p_c and p_m in the Q-ANSGA algorithm are adaptively determined, this section only requires calibration of the remaining four key parameters: population size $ps = \{40, 60, 80, 100\}$; learning rate $\alpha = \{0.1, 0.2, 0.3, 0.4\}$; discount rate $\gamma = \{0.7, 0.8, 0.9, 1.0\}$; and greedy rate $\varepsilon = \{0.1, 0.2, 0.3, 0.4\}$. To reduce the complexity of parameter calibration, an orthogonal experimental design with four factors and four levels, comprising 16 parameter combinations, was constructed based on the Taguchi method. The benchmark instance “100_5_4_4” was selected for this experimental phase. Furthermore, under each parameter combination, the algorithm was independently executed 20 times, and the mean value of the HV indicator was adopted as the final response value to mitigate the influence of random variations.

Figure 7 illustrates the variation trends of each parameter at different levels. It is important to emphasize that a higher HV indicates better overall performance. As observed from the figure, the algorithm achieves relatively ideal performance when the parameter combination is set to $ps = 80$, $\alpha = 0.3$, $\gamma = 0.8$, and $\varepsilon = 0.2$.

5.3. Effectiveness of Algorithm Designs

Q-ANSGA includes four enhancements: a hybrid initialization strategy, a Q-learning-based adaptive parameter-selection strategy, neighborhood search operators, and an IG-based operator selection mechanism. To assess the contribution of each improvement, Q-ANSGA is compared with four variants (Q-ANSGA-1 to Q-ANSGA-4), which are built cumulatively: Q-ANSGA-1: standard NSGA-II with random initialization; Q-ANSGA-2: adds the hybrid initialization strategy; Q-ANSGA-3: further includes the Q-learning-based adaptive selection; Q-ANSGA-4: additionally introduces random selection of neighborhood operators. This controlled-variable setup allows a clear analysis of each component’s impact on performance. All algorithms were terminated under the same runtime limit, each

instance was executed independently 20 times, and the average values of the evaluation metrics were taken as the experimental results.

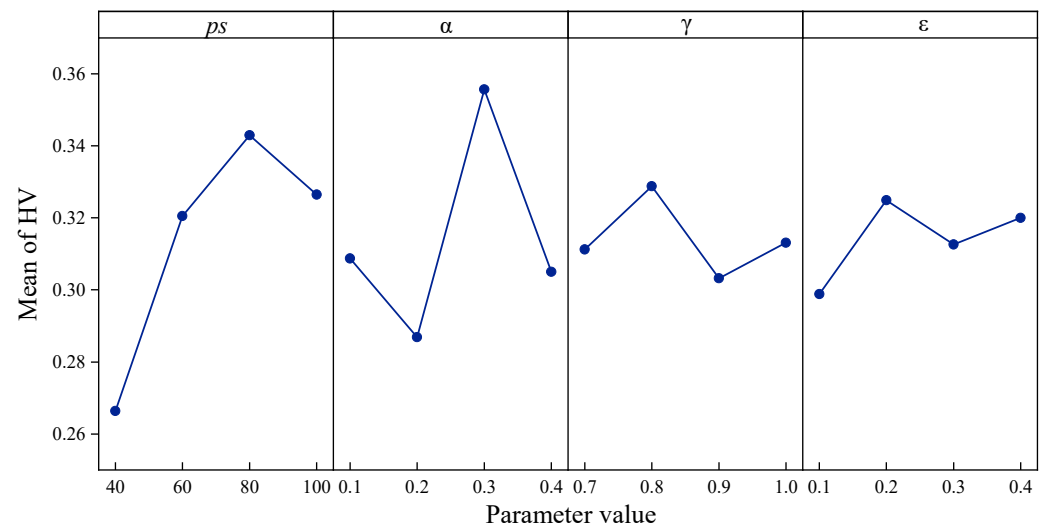


Figure 7. The main effects plot HV index.

Tables 2 and 3 present the average HV and IGD values of Q-ANSGA and its variants, with superior results highlighted in bold. As shown in the tables, Q-ANSGA achieves larger HV values in 33 out of the 45 test instances of different scales. This suggests that, in most cases, Q-ANSGA can explore a broader solution space, and its obtained solution set exhibits excellent distribution characteristics. It can also be observed that in some test instances, the HV value of Q-ANSGA is slightly lower than that of other variant algorithms. This is primarily because the solution set obtained by Q-ANSGA is more concentrated and uniformly distributed, without excessively favoring any single optimization objective. On the other hand, the IGD metric is used to measure the degree of approximation between the solution set obtained by the algorithm and the real Pareto frontier. The experimental results show that Q-ANSGA attains smaller IGD values across all instances, indicating its superiority in convergence compared to the other variants.

Table 2. Average HV values obtained by Q-ANSGA and its variants.

Instance	Q-ANSGA-1	Q-ANSGA-2	Q-ANSGA-3	Q-ANSGA-4	Q-ANSGA
20_5_2_3	0.3289	0.3558	0.3578	0.3622	0.4020
20_5_4_3	0.4039	0.4785	0.4407	0.5025	0.5754
20_5_6_3	0.4108	0.4552	0.4356	0.4778	0.5059
20_5_2_4	0.4137	0.4910	0.5086	0.5444	0.5458
20_5_4_4	0.4602	0.4993	0.4983	0.5405	0.5846
20_5_6_4	0.5090	0.5200	0.5345	0.5570	0.6785
20_5_2_5	0.3468	0.3376	0.3653	0.3645	0.4031
20_5_4_5	0.5441	0.5760	0.6080	0.6162	0.6174
20_5_6_5	0.4255	0.4531	0.4405	0.4522	0.4617
50_5_2_3	0.1913	0.1961	0.2200	0.2855	0.2876
50_5_4_3	0.1880	0.2025	0.1765	0.2257	0.2008
50_5_6_3	0.2159	0.2483	0.2797	0.2583	0.2466

Table 2. Cont.

Instance	Q-ANSGA-1	Q-ANSGA-2	Q-ANSGA-3	Q-ANSGA-4	Q-ANSGA
50_5_2_4	0.2233	0.2409	0.2846	0.3230	0.3410
50_5_4_4	0.1515	0.1728	0.2105	0.2858	0.3265
50_5_6_4	0.2350	0.2942	0.2382	0.2737	0.3424
50_5_2_5	0.2632	0.2779	0.3098	0.3610	0.3621
50_5_4_5	0.2031	0.1756	0.1705	0.1592	0.2105
50_5_6_5	0.2386	0.2433	0.2423	0.2435	0.2432
100_5_2_3	0.3082	0.3256	0.3263	0.3321	0.3522
100_5_4_3	0.3200	0.3122	0.3200	0.3418	0.3182
100_5_6_3	0.2169	0.2464	0.2786	0.3180	0.3417
100_5_2_4	0.3254	0.3325	0.3301	0.3372	0.3611
100_5_4_4	0.2028	0.2350	0.2515	0.2662	0.3437
100_5_6_4	0.3694	0.3918	0.3515	0.3812	0.3746
100_5_2_5	0.3263	0.3379	0.3355	0.3310	0.3909
100_5_4_5	0.3163	0.3168	0.3354	0.3394	0.3481
100_5_6_5	0.2568	0.2678	0.2799	0.2804	0.2855
150_10_2_3	0.3023	0.3114	0.3496	0.3200	0.3203
150_10_4_3	0.3107	0.3121	0.3250	0.3186	0.3354
150_10_6_3	0.3137	0.3217	0.3243	0.3422	0.3512
150_10_2_4	0.3340	0.3295	0.3394	0.3318	0.3277
150_10_4_4	0.3360	0.3323	0.3575	0.3349	0.3262
150_10_6_4	0.3147	0.3181	0.3257	0.3595	0.3437
150_10_2_5	0.3012	0.3080	0.3162	0.3193	0.3747
150_10_4_5	0.3040	0.3199	0.3315	0.3371	0.3943
150_10_6_5	0.3208	0.3462	0.3535	0.3585	0.3555
200_10_2_3	0.2843	0.3114	0.3291	0.3277	0.3672
200_10_4_3	0.2993	0.3065	0.3203	0.3214	0.3218
200_10_6_3	0.3121	0.3129	0.3232	0.3218	0.3266
200_10_2_4	0.3063	0.2982	0.3002	0.3299	0.3101
200_10_4_4	0.2819	0.3128	0.3124	0.3112	0.3107
200_10_6_4	0.3127	0.3139	0.3175	0.3332	0.3405
200_10_2_5	0.3147	0.3504	0.3532	0.3729	0.3837
200_10_4_5	0.2733	0.3026	0.3126	0.3193	0.3283
200_10_6_5	0.3178	0.3262	0.3370	0.3315	0.3453

Table 3. Average IGD values obtained by Q-ANSGA and its variants.

Instance	Q-ANSGA-1	Q-ANSGA-2	Q-ANSGA-3	Q-ANSGA-4	Q-ANSGA
20_5_2_3	0.6195	0.5417	0.5282	0.5180	0.4523
20_5_4_3	0.7097	0.6877	0.6602	0.6330	0.5723

Table 3. Cont.

Instance	Q-ANSGA-1	Q-ANSGA-2	Q-ANSGA-3	Q-ANSGA-4	Q-ANSGA
20_5_6_3	0.5681	0.5606	0.5011	0.5101	0.4774
20_5_2_4	0.3098	0.2605	0.2483	0.2272	0.2166
20_5_4_4	0.2403	0.2428	0.2014	0.2032	0.1626
20_5_6_4	0.1870	0.1802	0.1572	0.1349	0.1173
20_5_2_5	0.2503	0.2431	0.2418	0.2154	0.1929
20_5_4_5	0.2010	0.1913	0.1656	0.1468	0.1173
20_5_6_5	0.1932	0.1852	0.1582	0.1613	0.1166
50_5_2_3	0.6910	0.6682	0.6755	0.6501	0.5113
50_5_4_3	0.8071	0.7883	0.7283	0.7176	0.7171
50_5_6_3	0.5764	0.5121	0.5108	0.4949	0.4110
50_5_2_4	0.3466	0.3767	0.3577	0.3074	0.2405
50_5_4_4	0.5955	0.5718	0.5136	0.5120	0.4818
50_5_6_4	0.5031	0.5063	0.5749	0.5876	0.4867
50_5_2_5	0.4646	0.4183	0.3810	0.3679	0.2737
50_5_4_5	0.2583	0.2402	0.2400	0.2099	0.1979
50_5_6_5	0.2487	0.1998	0.1901	0.1408	0.1263
100_5_2_3	0.5697	0.4899	0.4611	0.4637	0.4374
100_5_4_3	0.9234	1.0028	0.8484	0.8398	0.8000
100_5_6_3	0.4512	0.4028	0.3995	0.3853	0.3374
100_5_2_4	0.7499	0.7356	0.7234	0.7228	0.6945
100_5_4_4	0.4932	0.4654	0.4539	0.4506	0.4131
100_5_6_4	0.4706	0.4618	0.4634	0.4600	0.3277
100_5_2_5	0.5531	0.5479	0.4883	0.4711	0.3716
100_5_4_5	0.3937	0.3925	0.3729	0.3691	0.2626
100_5_6_5	0.5851	0.5297	0.4727	0.4426	0.4157
150_10_2_3	1.0989	0.9884	0.9824	0.9629	0.9600
150_10_4_3	0.5864	0.5402	0.4645	0.4533	0.3932
150_10_6_3	0.7569	0.6758	0.7579	0.7123	0.5336
150_10_2_4	0.6882	0.6548	0.6733	0.6771	0.6373
150_10_4_4	0.7140	0.6895	0.6697	0.6564	0.6217
150_10_6_4	0.6501	0.6769	0.5356	0.5371	0.4696
150_10_2_5	0.6476	0.6023	0.5739	0.5068	0.4950
150_10_4_5	0.3861	0.3691	0.3521	0.3542	0.2510
150_10_6_5	0.5954	0.5630	0.5316	0.5262	0.4519
200_10_2_3	0.4220	0.3940	0.3946	0.3706	0.2522
200_10_4_3	0.7101	0.6788	0.6299	0.6373	0.6172
200_10_6_3	0.5178	0.4931	0.4891	0.4615	0.3619
200_10_2_4	0.6052	0.5631	0.5667	0.5132	0.4836

Table 3. Cont.

Instance	Q-ANSGA-1	Q-ANSGA-2	Q-ANSGA-3	Q-ANSGA-4	Q-ANSGA
200_10_4_4	0.5996	0.6726	0.6375	0.6668	0.5061
200_10_6_4	0.3308	0.3320	0.2950	0.3043	0.2903
200_10_2_5	0.6672	0.5410	0.6010	0.6294	0.5111
200_10_4_5	0.3819	0.3353	0.2855	0.2651	0.2157
200_10_6_5	0.4644	0.5054	0.4786	0.5225	0.3947

The p_c and p_m are key parameters in the algorithm. As illustrated in Figure 8 using the benchmark instance “100_5_4_4”, the adaptive Q-learning mechanism dynamically adjusts the (p_c, p_m) combination from 16 available choices. In the early stage, the values remain relatively high to promote broad exploration of the solution space. As the run progresses, the parameters gradually decrease to refine the search around promising non-dominated individuals. This dynamic adjustment demonstrates the stability and convergence of the Q-learning module, as it consistently transitions from exploration to exploitation according to the search progress. A comparison with the fixed-parameter variant Q-ANSGA-2 confirms the effectiveness of this adaptive strategy.

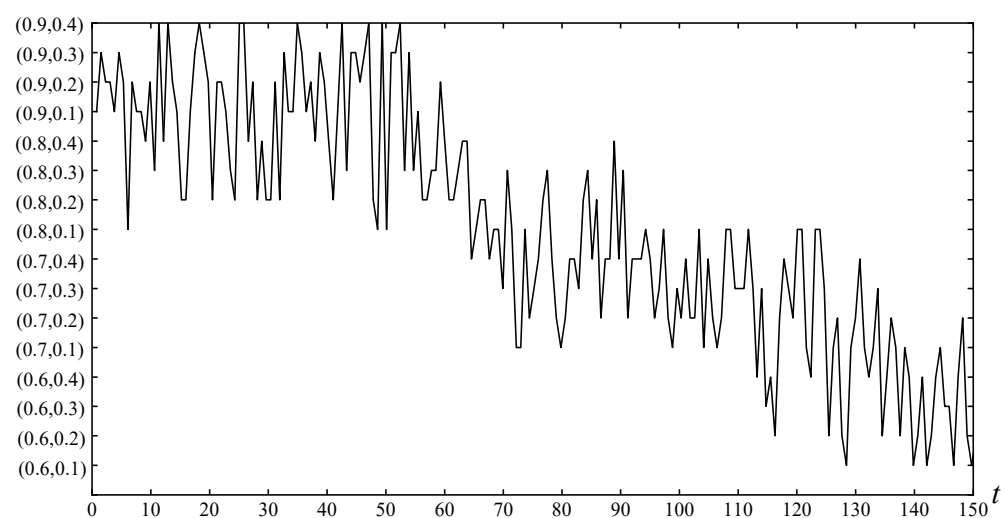


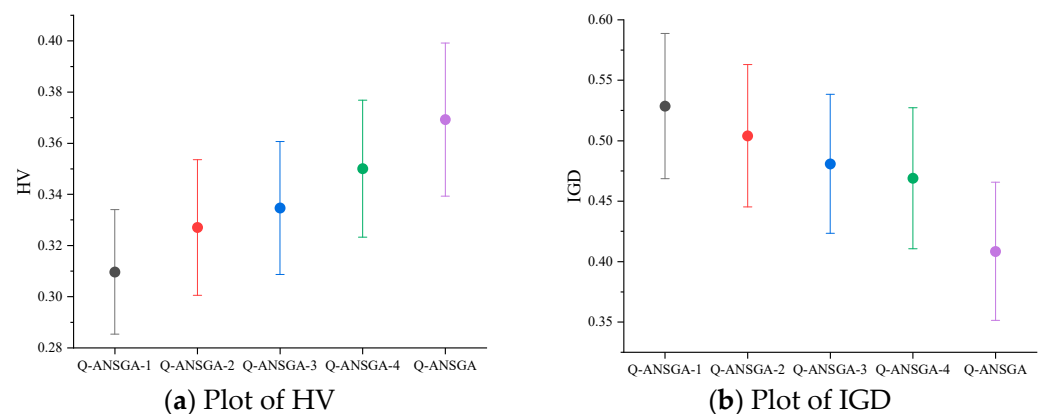
Figure 8. The trend of parameter variation.

To visually demonstrate the progressive performance improvement as the enhancement strategies are successively incorporated, the Friedman test is employed for analysis. As displayed in Table 4, the p -values for HV and IGD metrics are below the significance threshold, confirming that statistically significant differences exist between Q-ANSGA and the algorithmic variants. Furthermore, the ranking of the algorithms on the two evaluation metrics is Q-ANSGA-1 > Q-ANSGA-2 > Q-ANSGA-3 > Q-ANSGA-4 > Q-ANSGA, which reflects that the algorithm's performance is progressively enhanced as the improvements are incrementally added. Figure 9 illustrates the 95% confidence interval plots for Q-ANSGA and its variants. From the interval plots, it can be observed that as the enhancement strategies are sequentially introduced, the confidence intervals of Q-ANSGA corresponding to HV exhibit a gradual upward trend. A similar pattern is also observed for the IGD metric, further indicating that each improvement strategy has an excellent impact on the algorithm optimization. In summary, all four improvement components included in Q-ANSGA effectively enhance the overall performance.

Table 4. Friedman test for Q-ANSGA and its variants (level of significance $\alpha = 0.05$).

Algorithm	HV		IGD	
	Ranking	<i>p</i> -Value	Ranking	<i>p</i> -Value
Q-ANSGA-1	4.622	8.37×10^{-21}	4.644	2.80×10^{-29}
Q-ANSGA-2	3.578		3.889	
Q-ANSGA-3	2.978		3.000	
Q-ANSGA-4	2.222		2.467	
Q-ANSGA	1.600		1.000	

Bold values in the table indicate better experimental results.

**Figure 9.** Interval plot with 95% confidence intervals for the five algorithms.

5.4. Comparison with Other Algorithms

For validating the superiority of the Q-ANSGA, this paper selects four advanced algorithms for comparative analysis: IMOEA/D [20], MOHIG [33], KB-NSGA-II [35], and QLHHEA [38]. These algorithms were chosen as comparison objects based on two primary considerations: first, they incorporate well-designed improvement strategies; second, they have been demonstrated to exhibit excellent performance in solving shop scheduling problems. Table 5 presents the appropriate parameter setting of the comparative algorithms. In the experiments, all algorithms were independently executed 20 times on each test instance, with consistent runtime allocated to ensure fairness. Algorithm performance was evaluated using the average HV and IGD metrics.

Table 5. Parameter settings of the comparison algorithms.

Algorithm	Parameter Setting
IMOEA/D	$PS = 50, MR = 0.1, LSR = 0.2, T = 20$
MOHIG	$PS = 80, d = 4, pc = 0.8, pm = 0.4$
KB-NSGA-II	$pc = 0.6, pm = 0.3, Max_Gen = 250, Np = 0.6 \times n, \alpha = 0.9$
QLHHEA	$PS = 30, \varphi = 0.2, \lambda = 0.5, \gamma = 0.7$

Tables 6 and 7 compare Q-ANSGA with other algorithms on HV and IGD metrics (best means in bold). In Table 6, Q-ANSGA achieves higher HV values in most small-scale instances and in over half of the medium- to large-scale cases, indicating stronger solution-space exploration. Table 7 shows that Q-ANSGA yields lower average IGD across all instances, meaning its solutions are closer to the true Pareto front and exhibit better dominance performance. Overall, Q-ANSGA demonstrates superior performance for the FDAHFSP.

Table 6. Average HV values obtained by Q-ANSGA and its comparison algorithms.

Instance	IMOEa/D	MOHIG	KB-NSGA-II	QLHHEA	Q-ANSGA
20_5_2_3	0.2444	0.2543	0.2922	0.3006	0.3980
20_5_4_3	0.3331	0.3814	0.3556	0.3972	0.5796
20_5_6_3	0.2753	0.4269	0.3754	0.3936	0.5174
20_5_2_4	0.3064	0.3251	0.4394	0.3831	0.5306
20_5_4_4	0.4705	0.3768	0.5256	0.4119	0.5365
20_5_6_4	0.4082	0.3547	0.5076	0.3240	0.6174
20_5_2_5	0.2192	0.2778	0.3346	0.3025	0.4194
20_5_4_5	0.5948	0.5029	0.5146	0.4596	0.6427
20_5_6_5	0.4318	0.4421	0.4499	0.2953	0.4726
50_5_2_3	0.3454	0.3628	0.3640	0.3616	0.3970
50_5_4_3	0.2633	0.2977	0.3092	0.2863	0.2568
50_5_6_3	0.1633	0.2174	0.2270	0.2182	0.2600
50_5_2_4	0.2503	0.2679	0.2945	0.2718	0.3397
50_5_4_4	0.2493	0.2619	0.2932	0.2566	0.2776
50_5_6_4	0.2194	0.2194	0.2116	0.2031	0.2459
50_5_2_5	0.2356	0.3163	0.2512	0.2483	0.2726
50_5_4_5	0.2628	0.2495	0.3020	0.2664	0.2598
50_5_6_5	0.2818	0.2481	0.2176	0.2983	0.3140
100_5_2_3	0.2122	0.2780	0.2244	0.2219	0.2228
100_5_4_3	0.3169	0.3197	0.3339	0.3282	0.3298
100_5_6_3	0.3074	0.3183	0.3217	0.3263	0.3474
100_5_2_4	0.3152	0.3418	0.3300	0.3163	0.3664
100_5_4_4	0.3149	0.3156	0.3056	0.3234	0.3313
100_5_6_4	0.3731	0.3359	0.3506	0.3406	0.3814
100_5_2_5	0.3034	0.3360	0.3294	0.3090	0.3193
100_5_4_5	0.3506	0.3268	0.3620	0.3431	0.3518
100_5_6_5	0.3249	0.3508	0.3465	0.3408	0.3688
150_10_2_3	0.2951	0.3013	0.2924	0.3098	0.3108
150_10_4_3	0.3221	0.3088	0.3188	0.3166	0.3316
150_10_6_3	0.3292	0.3142	0.3435	0.3285	0.3318
150_10_2_4	0.3166	0.3011	0.3288	0.3182	0.3307
150_10_4_4	0.3335	0.3188	0.3248	0.3237	0.3442
150_10_6_4	0.3354	0.3258	0.3413	0.3238	0.3606
150_10_2_5	0.3037	0.3232	0.3156	0.3132	0.3459
150_10_4_5	0.3105	0.3294	0.3321	0.3332	0.3319
150_10_6_5	0.3282	0.3447	0.3325	0.3331	0.3546
200_10_2_3	0.3054	0.3060	0.3114	0.3089	0.3088
200_10_4_3	0.3188	0.3180	0.3370	0.3050	0.3205
200_10_6_3	0.3112	0.3152	0.3246	0.3166	0.3255

Table 6. Cont.

Instance	IMOEAD	MOHIG	KB-NSGA-II	QLHHEA	Q-ANSGA
200_10_2_4	0.3015	0.2991	0.3027	0.3020	0.2964
200_10_4_4	0.3045	0.3083	0.3177	0.3218	0.3112
200_10_6_4	0.3154	0.3205	0.3022	0.3192	0.3272
200_10_2_5	0.2986	0.3059	0.3114	0.3053	0.3096
200_10_4_5	0.3195	0.3206	0.3210	0.3092	0.3471
200_10_6_5	0.3164	0.3213	0.3219	0.3279	0.3310

Bold values in the table indicate better experimental results.

Table 7. Average IGD values obtained by Q-ANSGA and its comparison algorithms.

Instance	IMOEAD	MOHIG	KB-NSGA-II	QLHHEA	Q-ANSGA
20_5_2_3	0.9383	0.9163	0.8831	0.8804	0.5197
20_5_4_3	0.9630	0.9483	0.9223	0.9060	0.6243
20_5_6_3	0.8648	0.7594	0.7550	0.7866	0.4916
20_5_2_4	0.4250	0.4141	0.4518	0.4476	0.2364
20_5_4_4	0.3250	0.3328	0.3221	0.3424	0.1699
20_5_6_4	0.2321	0.1980	0.2558	0.1824	0.1036
20_5_2_5	0.4232	0.4305	0.4590	0.4444	0.1964
20_5_4_5	0.2029	0.1809	0.2486	0.1568	0.1073
20_5_6_5	0.2519	0.2692	0.2510	0.2271	0.1304
50_5_2_3	0.8910	0.8885	0.8929	0.8352	0.4896
50_5_4_3	0.9897	0.9596	0.9284	0.9542	0.6920
50_5_6_3	0.8853	0.8502	0.8305	0.8096	0.4278
50_5_2_4	0.6102	0.6495	0.5907	0.6119	0.2148
50_5_4_4	0.9010	0.8609	0.8778	0.8862	0.6226
50_5_6_4	0.8219	0.8243	0.8433	0.8145	0.4371
50_5_2_5	0.7074	0.6877	0.6936	0.6944	0.2516
50_5_4_5	0.6174	0.6584	0.5500	0.5728	0.2431
50_5_6_5	0.3897	0.3485	0.4012	0.3610	0.1837
100_5_2_3	1.0060	0.9792	0.9499	0.9692	0.4888
100_5_4_3	1.1629	1.1338	1.1345	1.1313	0.7983
100_5_6_3	0.8427	0.8200	0.7982	0.8236	0.3196
100_5_2_4	0.9115	0.9306	0.9059	0.9312	0.6885
100_5_4_4	0.7972	0.7471	0.7567	0.7699	0.3279
100_5_6_4	0.7888	0.7612	0.7746	0.8002	0.4673
100_5_2_5	0.9435	0.8297	0.8573	0.9122	0.5120
100_5_4_5	0.6835	0.6822	0.6852	0.6874	0.2749
100_5_6_5	0.6689	0.6619	0.6329	0.7516	0.4308
150_10_2_3	1.1933	1.1994	1.1594	1.2140	0.9426
150_10_4_3	0.9303	0.9443	0.9190	0.9445	0.4727

Table 7. Cont.

Instance	IMOE/D	MOHIG	KB-NSGA-II	QLHHEA	Q-ANSGA
150_10_6_3	1.0204	1.0164	1.0266	1.0074	0.6088
150_10_2_4	0.9233	0.8700	0.9035	0.8954	0.6167
150_10_4_4	0.8752	0.9009	0.7918	0.8694	0.6001
150_10_6_4	0.9498	0.9389	0.9097	0.9671	0.4117
150_10_2_5	0.8982	0.8501	0.8796	0.8776	0.5163
150_10_4_5	0.8029	0.7877	0.7265	0.7500	0.2902
150_10_6_5	0.9266	0.8541	0.9015	0.8867	0.4622
200_10_2_3	0.9333	0.9578	0.9044	0.9250	0.3896
200_10_4_3	1.0101	0.9950	0.9727	1.0164	0.6018
200_10_6_3	0.9632	0.9165	0.9192	0.9419	0.4570
200_10_2_4	1.0053	0.9981	0.9927	0.9887	0.5755
200_10_4_4	1.0046	0.9997	0.9698	0.9519	0.6723
200_10_6_4	0.7982	0.7610	0.7000	0.7591	0.3221
200_10_2_5	0.9641	0.9278	0.9177	0.9427	0.5668
200_10_4_5	0.7246	0.6848	0.6774	0.6784	0.2277
200_10_6_5	0.8858	0.8471	0.8443	0.8303	0.4493

Bold values in the table indicate better experimental results.

Table 8 presents the Friedman test results for the algorithms. The results show that the significance level is substantially larger than the p -values obtained for both the HV and IGD. This shows that there are obvious differences between the algorithms and confirms that the outcomes of metrics are not coincidental. In terms of two indicators, the proposed Q-ANSGA ranks first, demonstrating performance that is significantly superior to the other comparative algorithms. Furthermore, Figure 10 presents a comparison of the Pareto fronts of the algorithms across different problem scales, visually illustrating that Q-ANSGA possesses distinct advantages in both the quality of non-dominated solutions and distribution uniformity.

Table 8. Friedman test for Q-ANSGA and its competitors (level of significance $\alpha = 0.05$).

Algorithm	HV		IGD	
	Ranking	p -Value	Ranking	p -Value
IMOE/D	4.111	6.71×10^{-14}	4.289	5.30×10^{-22}
MOHIG	3.467		3.311	
KB-NSGA-II	2.467		3.044	
QLHHEA	3.333		3.355	
Q-ANSGA	1.622		1.000	

Bold values in the table indicate better experimental results.

The experimental results above indicate that Q-ANSGA exhibits clear advantages over the comparative algorithms on both evaluation metrics. This can be mainly attributed to the following aspects: first, Q-ANSGA adopts an initialization strategy that integrates three heuristic rules; second, it incorporates a Q-learning-based adaptive selection strategy, effectively balancing exploration and exploitation during different stages of the algo-

rithm; finally, through an iterative greedy based operator selection mechanism, it conducts neighborhood search to identify promising solutions in the vicinity of high-quality individuals. In summary, Q-ANSGA represents an effective and efficient approach for solving the FDAHFSP.

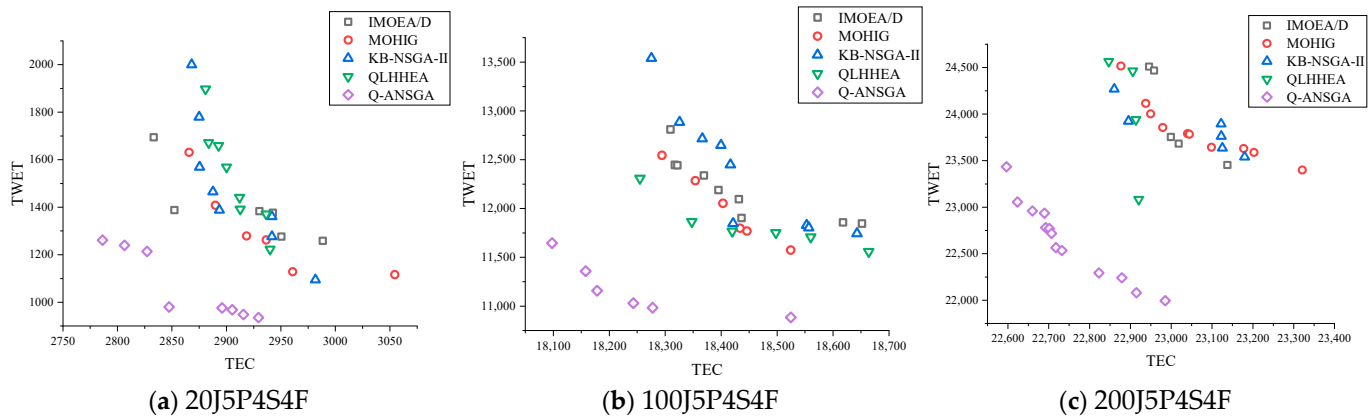


Figure 10. Pareto fronts of the algorithms under different scales.

Figures 11 and 12, respectively, present the scheduling Gantt charts of the best solutions obtained by Q-ANSGA for the two optimization objectives. As can be seen from the figure, the earliness/tardiness-oriented scheduling scheme tends to prioritize tasks with tight due dates and aims to balance the workload across shops. In contrast, the energy-oriented scheduling scheme tends to assign tasks to machines with lower energy consumption and strives to minimize machine idle time, thereby improving overall energy efficiency. Moreover, since product delivery dates are closely linked to the assembly stage, and assembly depends on the completion status of each component, it is essential to rationally arrange the processing sequence of the components belonging to the same product during scheduling to ensure on-time delivery.

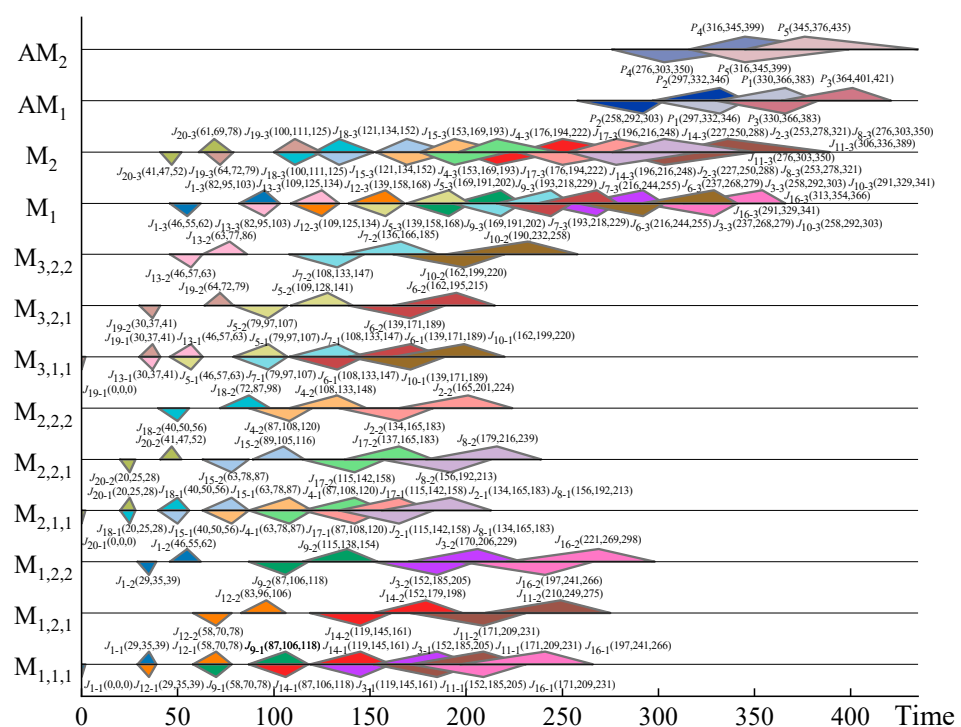


Figure 11. The Gantt chart of the $TWET$ optimal solution obtained by Q-ANSGA.

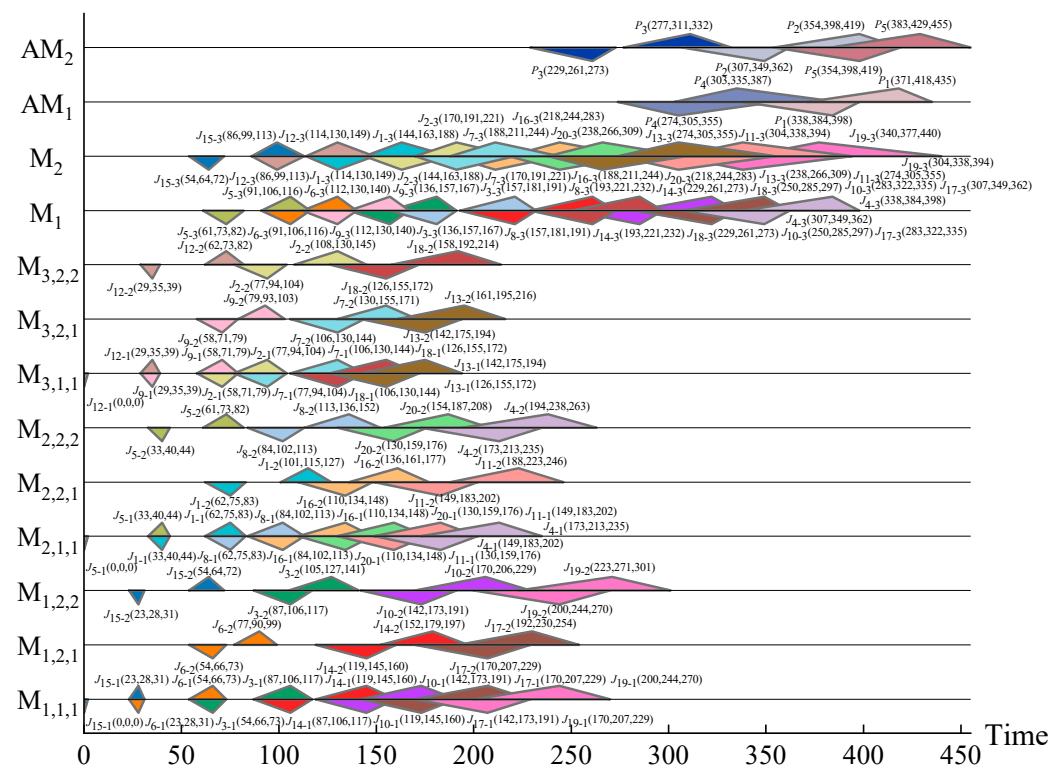


Figure 12. The Gantt chart of the $\widetilde{TEC}$ optimal solution obtained by Q-ANSGA.

6. Conclusions

This paper addresses the FDAHFSP aiming to minimize $\widetilde{TWET}$ and $\widetilde{TEC}$, and proposes a Q-ANSGA as a solution approach. First, a mathematical model of the FDAHFSP is developed. Second, a hybrid initialization method is proposed to improve the quality of initial solutions while preserving population diversity. Then, a Q-learning-based adaptive parameter selection strategy is introduced to dynamically adjust the p_c and p_m , thereby effectively controlling the intensity of global search. Subsequently, eight problem-specific neighborhood search operators are designed, and an iterative greedy-based operator selection mechanism is adopted to strengthen local search capability. The effectiveness of all proposed improvements is validated experimentally, and the results are statistically confirmed through Friedman tests. Finally, Q-ANSGA is compared with four advanced algorithms. The experimental results indicate that Q-ANSGA achieves average improvements of approximately 8.6% in HV and 92.2% in IGD compared to the best-performing baseline algorithm (KB-NSGA-II). These gains demonstrate the superior overall performance of Q-ANSGA in solving the FDAHFSP.

It should be noted that the proposed algorithm also has certain limitations. For example, the incorporation of the Q-learning module introduces additional computational overhead, which may hinder real-time applicability in highly dynamic environments. Future work may be extended along the following lines: (1) extending the problem to dynamic scheduling settings with practical disruptions such as machine failures and resource limits; (2) integrating human factors like worker skill and fatigue into the model, in line with human-centric manufacturing; (3) investigating alternative fuzzy methods (e.g., type-2 fuzzy numbers) to strengthen uncertainty handling.

Author Contributions: Conceptualization, R.W. and Q.L.; methodology, R.W.; software, Q.L. and B.C.; validation, R.W., Q.L. and X.L.; formal analysis, X.L. and Y.C.; investigation, R.W. and Q.L.; resources, B.C.; data curation, R.W.; writing—original draft preparation, R.W., Q.L. and B.C.; writing—review and editing, R.W. and Q.L.; visualization, Q.L.; supervision, B.C. and Y.C.; project administration,

X.L. and Y.C.; funding acquisition, R.W. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was jointly supported by the Scientific Research Foundation for High-level Talents of Hubei University of Technology (GCRC2020009), and the Doctoral Scientific Research Foundation of Hubei University of Technology (BSQD2020007).

Data Availability Statement: The data are contained within the article.

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

Abbreviations

The following abbreviations are used in this manuscript:

FDAHFSF	Fuzzy distributed assembly hybrid flow shop scheduling problem
Q-ANSGA	Q-learning-based adaptive non-dominated sorting genetic algorithm
DHFSP	Distributed hybrid flow shop scheduling problem
IMOEAD	Improved multi-objective evolutionary algorithm
MOHIG	Multi-objective hybrid iterated greedy algorithm
KB-NSGA-II	Knowledge-based non-dominated sorting genetic II algorithm
QLHHEA	Q-learning-based hyper-heuristic evolutionary algorithm

References

- Hou, Y.; Liao, X.; Chen, G.; Chen, Y. Co-Evolutionary NSGA-III with Deep Reinforcement Learning for Multi-Objective Distributed Flexible Job Shop Scheduling. *Comput. Ind. Eng.* **2025**, *203*, 110990. [\[CrossRef\]](#)
- Qin, H.-X.; Han, Y.-Y.; Liu, Y.-P.; Li, J.-Q.; Pan, Q.-K.; Han, X. A Collaborative Iterative Greedy Algorithm for the Scheduling of Distributed Heterogeneous Hybrid Flow Shop with Blocking Constraints. *Expert Syst. Appl.* **2022**, *201*, 117256. [\[CrossRef\]](#)
- Pan, Y.; Gao, K.; Li, Z.; Wu, N. A Novel Evolutionary Algorithm for Scheduling Distributed No-Wait Flow Shop Problems. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *54*, 3694–3704. [\[CrossRef\]](#)
- Chen, X.; Li, Y.; Wang, K.; Wang, L.; Liu, J.; Wang, J.; Wang, X.V. Reinforcement Learning for Distributed Hybrid Flowshop Scheduling Problem with Variable Task Splitting towards Mass Personalized Manufacturing. *J. Manuf. Syst.* **2024**, *76*, 188–206. [\[CrossRef\]](#)
- Li, R.; Gong, W.; Lu, C. Self-Adaptive Multi-Objective Evolutionary Algorithm for Flexible Job Shop Scheduling with Fuzzy Processing Time. *Comput. Ind. Eng.* **2022**, *168*, 108099. [\[CrossRef\]](#)
- Zhang, X.; Liu, S.; Zhao, Z.; Yang, S. A Decomposition-Based Evolutionary Algorithm with Clustering and Hierarchical Estimation for Multi-Objective Fuzzy Flexible Jobshop Scheduling. *IEEE Trans. Evol. Computat.* **2024**, *30*, 2–15. [\[CrossRef\]](#)
- Wang, J.-J.; Wang, L. A Cooperative Memetic Algorithm with Learning-Based Agent for Energy-Aware Distributed Hybrid Flow-Shop Scheduling. *IEEE Trans. Evol. Computat.* **2022**, *26*, 461–475. [\[CrossRef\]](#)
- Zhang, W.; Geng, H.; Li, C.; Gen, M.; Zhang, G.; Deng, M. Q-Learning-Based Multi-Objective Particle Swarm Optimization with Local Search within Factories for Energy-Efficient Distributed Flow-Shop Scheduling Problem. *J. Intell. Manuf.* **2023**, *36*, 185–208. [\[CrossRef\]](#)
- Li, D.; Gao, K.; Duan, P.; Suganthan, P.N.; Wu, N. Reinforcement Learning Assisting Artificial Bee Colony Algorithm for Scheduling Distributed Assembly Flowshops with Batch Delivery. *IEEE Trans. Syst. Man Cybern. Syst.* **2025**, *55*, 9295–9308. [\[CrossRef\]](#)
- Ma, J.; Gao, W.; Tong, W. A Deep Reinforcement Learning Assisted Adaptive Genetic Algorithm for Flexible Job Shop Scheduling. *Eng. Appl. Artif. Intel.* **2025**, *149*, 110447. [\[CrossRef\]](#)
- Zhang, W.; Li, C.; Gen, M.; Yang, W.; Zhang, G. A Multiobjective Memetic Algorithm with Particle Swarm Optimization and Q-Learning-Based Local Search for Energy-Efficient Distributed Heterogeneous Hybrid Flow-Shop Scheduling Problem. *Expert Syst. Appl.* **2024**, *237*, 121570. [\[CrossRef\]](#)
- Wu, X.; Yan, X.; Guan, D.; Wei, M. A Deep Reinforcement Learning Model for Dynamic Job-Shop Scheduling Problem with Uncertain Processing Time. *Eng. Appl. Artif. Intel.* **2024**, *131*, 107790. [\[CrossRef\]](#)
- Kayhan, B.M.; Yildiz, G. Reinforcement Learning Applications to Machine Scheduling Problems: A Comprehensive Literature Review. *J. Intell. Manuf.* **2023**, *34*, 905–929. [\[CrossRef\]](#)
- Wang, H.; Yu, X.; Lu, Y. A Reinforcement Learning-Based Ranking Teaching-Learning-Based Optimization Algorithm for Parameters Estimation of Photovoltaic Models. *Swarm Evol. Comput.* **2025**, *93*, 101844. [\[CrossRef\]](#)

15. Zuo, G.; Jia, Z.; Wu, Z.; Shi, J.; Wang, G. A Q-Learning Guided Dual Population Genetic Algorithm for Distributed Permutation Flow Shop Scheduling Problem with Machine Having Fuzzy Processing Efficiency. *Expert Syst. Appl.* **2025**, *285*, 127882. [[CrossRef](#)]
16. Li, Y.; Li, X.; Gao, L.; Zhang, B.; Pan, Q.-K.; Tasgetiren, M.F.; Meng, L. A Discrete Artificial Bee Colony Algorithm for Distributed Hybrid Flowshop Scheduling Problem with Sequence-Dependent Setup Times. *Int. J. Prod. Res.* **2021**, *59*, 3880–3899. [[CrossRef](#)]
17. Lu, C.; Zhou, J.; Gao, L.; Li, X.; Wang, J. Modeling and Multi-Objective Optimization for Energy-Aware Scheduling of Distributed Hybrid Flow-Shop. *Appl. Soft Comput.* **2024**, *156*, 111508. [[CrossRef](#)]
18. Shao, W.; Shao, Z.; Pi, D. Multi-Objective Evolutionary Algorithm Based on Multiple Neighborhoods Local Search for Multi-Objective Distributed Hybrid Flow Shop Scheduling Problem. *Expert Syst. Appl.* **2021**, *183*, 115453. [[CrossRef](#)]
19. Wang, J.; Tang, H.; Lei, D. A Q-Learning Artificial Bee Colony for Distributed Assembly Flow Shop Scheduling with Factory Eligibility, Transportation Capacity and Setup Time. *Eng. Appl. Artif. Intel.* **2023**, *123*, 106230. [[CrossRef](#)]
20. Luo, Q.; Deng, Q.; Guo, X.; Gong, G.; Zhao, X.; Chen, L. Modelling and Optimization of Distributed Assembly Hybrid Flowshop Scheduling Problem with Transportation Resource Scheduling. *Comput. Ind. Eng.* **2023**, *186*, 109717. [[CrossRef](#)]
21. Sun, L.; Lin, L.; Gen, M.; Li, H. A Hybrid Cooperative Coevolution Algorithm for Fuzzy Flexible Job Shop Scheduling. *IEEE Trans. Fuzzy Syst.* **2019**, *27*, 1008–1022. [[CrossRef](#)]
22. Li, R.; Gong, W.; Lu, C.; Wang, L. A Learning-Based Memetic Algorithm for Energy-Efficient Flexible Job-Shop Scheduling with Type-2 Fuzzy Processing Time. *IEEE Trans. Evol. Computat.* **2023**, *27*, 610–620. [[CrossRef](#)]
23. Wang, J.; Lei, D.; Tang, H. A Multi-Objective Dynamical Artificial Bee Colony for Energy-Efficient Fuzzy Hybrid Flow Shop Scheduling with Batch Processing Machines. *Expert Syst. Appl.* **2025**, *259*, 125244. [[CrossRef](#)]
24. Wang, Y.-J.; Wang, G.-G.; Tian, F.-M.; Gong, D.-W.; Pedrycz, W. Solving Energy-Efficient Fuzzy Hybrid Flow-Shop Scheduling Problem at a Variable Machine Speed Using an Extended NSGA-II. *Eng. Appl. Artif. Intell.* **2023**, *121*, 105977. [[CrossRef](#)]
25. Wang, J.; Lei, D.; Li, D.; Li, X.; Tang, H. A Dynamic Artificial Bee Colony for Fuzzy Distributed Energy-Efficient Hybrid Flow Shop Scheduling with Batch Processing Machines. *J. Manuf. Syst.* **2025**, *78*, 94–108. [[CrossRef](#)]
26. 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*, 125002. [[CrossRef](#)]
27. Zhao, F.; Du, Y.; Zhuang, C.; Wang, L.; Yu, Y. An Iterative Greedy Algorithm for Solving a Multiobjective Distributed Assembly Flexible Job Shop Scheduling Problem with Fuzzy Processing Time. *IEEE Trans. Cybern.* **2025**, *55*, 2302–2315. [[CrossRef](#)]
28. Wang, Y.; Han, Y.; Wang, Y.; Pan, Q.-K.; Wang, L. Sustainable Scheduling of Distributed Flow Shop Group: A Collaborative Multi-Objective Evolutionary Algorithm Driven by Indicators. *IEEE Trans. Evol. Comput.* **2024**, *28*, 1794–1808. [[CrossRef](#)]
29. Shi, Z.; Si, J.; Zhang, J.; Pang, Z.; Chen, H.; Ding, G. A Deep Reinforcement Learning Method Based on Hindsight Experience Replay for Multi-Objective Dynamic Job-Shop Scheduling Problem. *Expert Syst. Appl.* **2025**, *284*, 127989. [[CrossRef](#)]
30. Yao, Y.; Li, X.; Gao, L. A DQN-Based Memetic Algorithm for Energy-Efficient Job Shop Scheduling Problem with Integrated Limited AGVs. *Swarm Evol. Comput.* **2024**, *87*, 101544. [[CrossRef](#)]
31. Zhu, N.; Zhao, F.; Wang, L.; Ding, R.; Xu, T.; Jonrinaldi, J. A Discrete Learning Fruit Fly Algorithm Based on Knowledge for the Distributed No-Wait Flow Shop Scheduling with Due Windows. *Expert Syst. Appl.* **2022**, *198*, 116921. [[CrossRef](#)]
32. Yu, H.; Gao, K.; Li, Z.; Duan, P. Double-Learning-Strategy-Based Evolutionary Algorithm for Scheduling Multiobjective Distributed Assembly Permutation Flowshops with Setup Time. *IEEE Trans. Syst. Man Cybern. Syst.* **2025**, *55*, 925–935. [[CrossRef](#)]
33. Lu, C.; Liu, Q.; Zhang, B.; Yin, L. A Pareto-Based Hybrid Iterated Greedy Algorithm for Energy-Efficient Scheduling of Distributed Hybrid Flowshop. *Expert Syst. Appl.* **2022**, *204*, 117555. [[CrossRef](#)]
34. Lei, D.; Su, B. A Multi-class Teaching-learning-based Optimization for Multi-objective Distributed Hybrid Flow Shop Scheduling. *Knowl.-Based Syst.* **2023**, *263*, 110252. [[CrossRef](#)]
35. Chen, L.; Cao, L.; Wen, Y.; Chen, H.; Jiang, S.-L. A Knowledge-Based NSGA-II Algorithm for Multi-Objective Hot Rolling Production Scheduling Under Flexible Time-of-Use Electricity Pricing. *J. Manuf. Syst.* **2023**, *69*, 255–270. [[CrossRef](#)]
36. Lv, L.; Shen, W. An Improved NSGA-II with Local Search for Multi-Objective Integrated Production and Inventory Scheduling Problem. *J. Manuf. Syst.* **2023**, *68*, 99–116. [[CrossRef](#)]
37. Wang, Y.-J.; Li, J.; Wang, G.-G. Fuzzy Correlation Entropy-Based NSGA-II for Energy-Efficient Hybrid Flow-Shop Scheduling Problem. *Knowl.-Based Syst.* **2023**, *277*, 110808. [[CrossRef](#)]
38. Zhang, Z.-Q.; Qian, B.; Hu, R.; Yang, J.-B. Q-Learning-Based Hyper-Heuristic Evolutionary Algorithm for the Distributed Assembly Blocking Flowshop Scheduling Problem. *Appl. Soft Comput.* **2023**, *146*, 110695. [[CrossRef](#)]
39. Chen, X.; Li, J.; Xu, Y. Q-Learning Based Multi-Objective Immune Algorithm for Fuzzy Flexible Job Shop Scheduling Problem Considering Dynamic Disruptions. *Swarm Evol. Comput.* **2023**, *83*, 101414. [[CrossRef](#)]
40. Wang, G.-G.; Gao, D.; Pedrycz, W. Solving Multiobjective Fuzzy Job-Shop Scheduling Problem by a Hybrid Adaptive Differential Evolution Algorithm. *IEEE Trans. Ind. Inf.* **2022**, *18*, 8519–8528. [[CrossRef](#)]
41. Qiao, Y.; Wu, N.; He, Y.; Li, Z.; Chen, T. Adaptive Genetic Algorithm for Two-Stage Hybrid Flow-Shop Scheduling with Sequence-Independent Setup Time and No-Interruption Requirement. *Expert Syst. Appl.* **2022**, *208*, 118068. [[CrossRef](#)]

42. Zhuang, M.; Zhang, W.; Tang, H.; Li, X.; Wang, K. A Multi-Objective Genetic Algorithm Based on Two-Stage Reinforcement Learning for Green Flexible Shop Scheduling Problem Considering Machine Speed. *Expert Syst. Appl.* **2024**, *258*, 125189. [[CrossRef](#)]
43. Chen, R.; Wu, B.; Wang, H.; Tong, H.; Yan, F. A Q-Learning Based NSGA-II for Dynamic Flexible Job Shop Scheduling with Limited Transportation Resources. *Swarm Evol. Comput.* **2024**, *90*, 101658. [[CrossRef](#)]
44. Yu, H.; Gao, K.; Li, Z.; Suganthan, P.N. Energy-Efficient Multi-Objective Distributed Assembly Permutation Flowshop Scheduling by Q-Learning Based Meta-Heuristics. *Appl. Soft Comput.* **2024**, *166*, 112247. [[CrossRef](#)]
45. Hatami, S.; Ruiz, R.; Andres-Romano, C. The Distributed Assembly Permutation Flowshop Scheduling Problem. *Int. J. Prod. Res.* **2013**, *51*, 5292–5308. [[CrossRef](#)]
46. Yu, F.; Lu, C.; Yin, L.; Zhou, J. Modeling and Optimization Algorithm for Energy-Efficient Distributed Assembly Hybrid Flowshop Scheduling Problem Considering Worker Resources. *J. Ind. Inf. Integr.* **2024**, *40*, 100620. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.