

Article

Joint Optimization of Spare Part Manufacturing and Maintenance Workforce Scheduling Under Heterogeneous In-Warranty and Out-of-Warranty Demands

Yinwen Ma ^{1,2}, Qianwang Deng ^{1,2,*}, Juan Zhou ^{1,2} and Jingxing Zhang ^{1,2}

¹ School of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China; elk1319@hnu.edu.cn (Y.M.)

² State Key Laboratory of Advanced Design and Manufacturing Technology for Vehicle, Hunan University, Changsha 410082, China

* Correspondence: deng_arbeit@163.com; Tel.: +86-13319561736

Abstract

The efficient operation of the maintenance service system is key to achieving sustainable operations, with its core lying in the coordinated scheduling of spare parts production and maintenance personnel, as well as the holistic management of in-warranty and out-of-warranty demands. This approach optimizes resource allocation and enhances long-term service value. This paper investigates the integrated scheduling of distributed spare parts production and maintenance personnel with differentiated in-warranty and out-of-warranty demands (ISSPD). To solve the ISSPD, an improved non-dominated sorting genetic algorithm-II that uses Q-learning to adaptively select local search strategies (QLNSGA) is proposed, which incorporates a decoding strategy for differentiated order types, eight knowledge-driven local search strategies, and a Q-learning mechanism for the adaptive selection of key local search operators. Compared to random local search operators, the Q-learning mechanism achieves a 55% decrease in IGD metric and a 65% increase in HV metric. Through comparative experiments with four mainstream algorithms, QLNSGA outperforms RIPG by 58% in terms of the IGD index, and its HV index is generally superior to that of comparative algorithms such as MOEA/D. This indicates that QLNSGA exhibits superior performance in both computational efficiency and solution quality, effectively enhancing service levels and significantly reducing operational costs, thereby providing scientific decision support for service-oriented manufacturing enterprises.

Keywords: distributed permutation flowshop scheduling; integrated scheduling of dual resources; warranty period; differentiated requirements; Q-learning

Academic Editor: Koichi Murata

Received: 12 January 2026

Revised: 12 February 2026

Accepted: 13 February 2026

Published: 17 February 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

As the manufacturing industry rapidly advances toward intelligentization and digitalization [1,2], servitization has become a critical pathway for companies to build new competitive advantages and move up the global value chain. In this context, after-sales service has evolved from a traditional product appendage into a key factor influencing customer loyalty and brand reputation. High-quality after-sales service not only strengthens customer trust and increases repurchase intention but also helps companies tap into

new market opportunities through positive word-of-mouth. Therefore, within the broader trend of servitization in manufacturing, comprehensively improving the quality and efficiency of after-sales service has emerged as a core issue directly related to the sustainable development of enterprises.

The allocation and scheduling of after-sales service resources are central to this endeavor. Manufacturing enterprises must achieve precise and efficient configuration of resources to ensure the continuous operation of customer equipment. This involves organizing timely spare parts production and logistics, while also scheduling skilled maintenance personnel for pre-failure replacements. However, a common challenge is the contradiction between limited resources (e.g., production capacity and workforce) and geographically dispersed customer demands. Therefore, collaboratively optimizing spare parts production and service personnel scheduling under resource constraints is key to enhancing service quality, response efficiency, and thus core competitiveness.

Customer demands in after-sales service are diverse and typically categorized by warranty status: in-warranty and out-of-warranty [3]. For in-warranty demands, manufacturers must fulfill service obligations free of charge as per agreements [4], where service timeliness and customer satisfaction are paramount as they directly impact brand trust. Conversely, out-of-warranty services are customer-paid, shifting the focus to profitability and efficient resource utilization to maximize returns. In practice, requests from both categories often occur simultaneously. Yet, few studies concurrently address the scheduling challenges arising from this warranty-based demand differentiation while integrating the planning of two core resources: spare parts production and maintenance personnel dispatch.

Regarding solution methodologies, metaheuristic algorithms like the iterated greedy (IG) [5], artificial bee colony algorithm (ABC) [6], and non-dominated sorting genetic algorithm II (NSGA-II) [7] have shown excellent performance in shop scheduling due to NSGA-II's flexible structure and stability. However, traditional metaheuristics often rely on fixed rules or random search, lacking adaptability in sequential decision-making. This makes them prone to local optima and low convergence efficiency when facing complex, multi-stage problems like the Integrated Spare Parts and Service Personnel Scheduling (ISSPD) problem, which involves differentiated warranty demands, production planning, and personnel scheduling. Recently, reinforcement learning, especially Q-learning, has offered a new approach by enabling autonomous strategy learning through interaction. For instance, Q-learning has been employed to allocate adaptive individuals to sub-populations [8], dynamically adjust algorithm parameters or optimize strategy selection [9,10], and assist in selecting optimal activity times in multi-mode scenarios [11]. However, existing Q-learning-enhanced approaches still fall short for our problem. They often focus on inventory control or resource allocation in isolation, fail to unify spare parts production and personnel dispatch into one framework, and crucially, do not consider the differentiated demands and decision logic imposed by mixed warranty scenarios.

Existing models are inadequate for effectively addressing the coordinated optimization of production and service domains under mixed warranty periods in this study, due to incompatibilities in objective functions, misalignment in decision priorities and time scales, and the failure of state and reward mechanisms to distinguish the fundamental differences between warranty-covered and out-of-warranty requests. Based on the aforementioned research background, this paper addresses the diverse after-sales service demands faced by manufacturing enterprises by proposing an integrated scheduling of distributed spare parts production and maintenance personnel with differentiated in-warranty and out-of-warranty demands (ISSPD). Given the widespread application of permutation flowshops in spare parts production, the production environment in this study is established as a distributed permutation flowshop scheduling problem (DPFSP). To

accurately characterize this problem, a mixed-integer programming model is formulated with the optimization objective of simultaneously maximizing customer satisfaction during the warranty period and service revenue from out-of-warranty services. Regarding the solution methodology, considering the demonstrated effectiveness of NSGA-II in solving DPFSP, this paper proposes a Q-learning-based optimization framework for coordinated scheduling across production and service domains. By integrating composite state design, cross-domain operator coordination, and a multi-objective-driven decision-coupling mechanism, it achieves dynamic harmonization of resource competition and objective conflicts, addressing the complex constraints and multi-objective nature of ISSPD. Extensive experiments validate the algorithm's superior performance. The main contributions are:

- Investigating a DPFSP that integrates differentiated warranty demands, spare parts production, and maintenance personnel scheduling into a multi-objective model.
- Designing a problem-specific re-decoding method for different warranty types, eight knowledge-driven local search strategies, and a Q-learning mechanism to dynamically guide strategy selection.
- Validating the proposed method's effectiveness and superior performance against mainstream algorithms through extensive computational experiments.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature. Section 3 defines the problem and develops the model. Section 4 details the algorithm design. Section 5 presents and analyzes experimental results. Section 6 concludes the study.

2. Literature Review

This study investigates the integrated scheduling of spare parts production and maintenance personnel with differentiated in-warranty and out-of-warranty customer demands in a distributed permutation flowshop environment, and proposes a corresponding solution. A literature review is then conducted on three research areas relevant to this work: (1) distributed permutation flowshop scheduling, (2) multi-resource coordinated scheduling, and (3) differentiated in-warranty/out-of-warranty services, along with (4) Q-learning enhanced metaheuristics.

2.1. Distributed Permutation Flowshop Scheduling Problem

With the increasing networking of global manufacturing, the rise of personalized demand, and the trends toward economic globalization and decentralization, multi-factory environments have become increasingly common, gradually replacing traditional centralized manufacturing. Consequently, distributed scheduling problems have garnered growing attention [12]. The distributed permutation flowshop scheduling problem (DPFSP) stands as a crucial branch of distributed flowshop scheduling. In recent years, research on DPFSP has primarily focused on two aspects: model extensions and the design of efficient solution algorithms. At the modeling level, researchers have progressively incorporated more realistic production constraints—such as no-idle, no-wait, sequence-dependent setup times, batch processing, assembly constraints, transportation times, and multi-objective requirements—leading to the emergence of various complex variants. For instance, some studies have integrated due window constraints, tardiness costs, and order priorities into the objective function, making the problem more aligned with order-driven manufacturing scenarios. Variable neighborhood search (VNS), tabu search (TS), genetic algorithms (GA), and hybrid swarm intelligence algorithms have been widely applied to DPFSP. Numerous studies have enhanced solution quality and stability through algorithm hybridization and adaptive strategies [13,14]. Existing research has made significant

progress in both model extensions and algorithm design for DPFSP. However, most studies remain focused on internal production system optimization, failing to adequately address the integrated scheduling with after-sales service resources (such as maintenance support and spare parts supply) or account for the challenges posed by differentiated service demands and scheduling priorities arising from varying customer order warranty periods. Therefore, further investigation into DPFSP is not only crucial for enhancing multi-factory resource allocation efficiency and reducing total production costs, but also represents an essential requirement for advancing the deep integration of manufacturing and service sectors and achieving global supply chain optimization, holding substantial theoretical significance and promising application prospects.

2.2. Multi-Resource Collaborative Scheduling

With the rapid advancement of informatization, manufacturing systems are increasingly evolving toward multi-resource coordinated scheduling, multi-resource coordinated scheduling approaches all resources as an integrated organic system. Through unified planning and real-time coordination, these approaches significantly enhance overall system efficiency, flexibility, and responsiveness, making them crucial for implementing smart manufacturing and strengthening core competitiveness.

Sleptchenko, Rahimi-Ghahroodi et al. [15,16] investigated the joint optimization of spare parts inventory and service personnel numbers. Bajestani et al. [17] addressed integrated maintenance and production scheduling for multi-machine degraded production systems across multiple cycles. Building upon the concept of average machine utilization, Liao et al. [18] incorporated machine degradation to construct a more realistic job processing time model for manufacturing scenarios; Zhang et al. [19] proposed a collaborative scheduling algorithm to support real-time production and machine maintenance in highly dynamic CMfg environments, ensuring agility and effectiveness under a service-oriented paradigm; Miao et al. [20] analyzed the joint scheduling problem of spare parts production and service personnel driven by distributed maintenance demands. By implementing just-in-time production for direct delivery from factories to maintenance sites, they simultaneously optimized customer satisfaction and total service costs.

It is noteworthy that most of the aforementioned studies adopt a manufacturer (supply-side) perspective. It is evident that the existing literature predominantly focuses on integrated scheduling of “production and machine maintenance” within the manufacturing phase. However, research remains insufficient regarding the incorporation of “maintenance personnel” as a critical resource into the coordinated scheduling framework, indicating that related mechanisms require further in-depth exploration.

2.3. Differentiated In-Warranty and Out-of-Warranty Services

In the existing research on after-sales services, scholars have predominantly explored service quality improvement from perspectives such as maintenance strategy optimization and resource scheduling. However, there has been limited systematic investigation into differentiated service standards for distinct customer segments.

As highlighted by Vanlerberghe et al. [21], service differentiation can be fundamentally defined as a broad category of resource allocation optimization problems. Against the backdrop of manufacturing servitization, after-sales service models have exhibited increasingly refined classification trends to enhance product added value. The Extended Maintenance Service (EMS) framework introduces a service classification strategy based on product warranty status, categorizing customers into in-warranty and out-of-warranty groups and providing corresponding maintenance services accordingly. Yeh et al. [22] noted that for in-warranty customers, EMS typically provides necessary repairs (such as spare part replacements) upon request without charging the customers. Conversely, Jung

et al. [23] emphasized that for out-of-warranty customers, manufacturers can offer paid repair services to address their needs. Such services not only resolve practical issues but also enhance customer loyalty and satisfaction, thereby implicitly fulfilling deeper customer needs. Due to the fundamental differences in service types, customers' evaluation criteria for service quality also vary significantly. In practice, under resource constraints, manufacturers typically prioritize fulfilling in-warranty orders as stipulated in contracts before allocating remaining resources to out-of-warranty orders, aiming to maximize overall customer satisfaction.

Therefore, in after-sales service resource scheduling, systematically considering the differentiated demands of in-warranty and out-of-warranty customers is crucial for achieving global resource optimization and maximizing customer value in service-oriented manufacturing.

2.4. Meta-Heuristic Algorithm Based on Q-Learning

In existing research, meta-heuristic algorithms have been widely applied to distributed shop scheduling problems due to their excellent solution performance [24–26]. Q-learning, a reinforcement learning method that learns optimal policies through iterative updates of the Q-value function, has recently emerged as a research hotspot in the scheduling field. It is widely employed as an autonomous decision-making mechanism and is often combined with meta-heuristic algorithms to solve various complex production scheduling optimization problems. Primarily focusing on two areas: first, selecting high-quality strategies; and second, dynamically calibrating algorithm parameters. This demonstrates its broad applicability in enhancing search efficiency and solution quality.

In practical applications, Jia et al. [8] proposed an MPMA-QL algorithm based on Q-learning to solve scheduling problems for distributed assembly lines, accounting for machine performance degradation, maintenance, transportation, and the complex flexibility of assembly lines. Q-learning was used to assign adaptive individuals to each subpopulation, enhancing algorithmic stability when solving complex combinatorial optimization problems. Similarly, Li et al. [9] combined Q-learning with a non-dominated sorting genetic algorithm (NSGA-II) driven by a universal variable neighborhood search, proposing the QVNS-NSGA-II algorithm. This approach addresses energy-efficient hybrid flowshop scheduling problems aimed at minimizing total delay, total energy cost, and carbon trading cost. It leverages Q-learning to select high-quality strategies during global and local search processes, thereby enhancing the algorithm's search capability. Chen et al. [10] introduced a Q-learning mechanism into a two-stage improved spider monkey optimization algorithm. Targeting the minimization of completion time, total energy consumption, and total delay time, they addressed scheduling problems in hybrid flowshops with batch flows. Q-learning dynamically adjusted algorithm parameters, guiding the search toward the desired solution space and improving both convergence speed and solution quality. Zuo et al. [27] proposed a Q-learning-based dual-population genetic algorithm (QGGA) for distributed flowshop scheduling. By establishing a cooperative pool, non-dominated solution sharing between dual populations was enabled while maintaining their evolutionary independence. Q-learning facilitated dynamic selection of local search strategies to promote synergistic population optimization. The NSGA-II genetic algorithm, based on crossover, mutation, and the Pareto front, has garnered significant attention and widespread application due to its high adaptability and extensibility [28,29]. For this study, a non-dominated sorting genetic algorithm (NSGA-II) based on the Q-learning mechanism was designed to dynamically select knowledge-driven local search strategies.

3. Problem Description and Formulation

3.1. Problem Description

This paper proposes an after-sales service model designed to address the differentiated demands of in-warranty and out-of-warranty customers. The model centers on an Integrated Scheduling of Spare parts production and maintenance Personnel with Differentiated warranty demands (ISSPD), which aims to simultaneously maximize customer satisfaction for in-warranty services and maintenance revenue from out-of-warranty services. We strictly treat the two objectives as non-dominated relationships and employ a multi-objective optimization framework based on NSGA-II to solve for the Pareto optimal solution set. We chose this approach to avoid pre-setting subjective weights, thereby objectively revealing all possible trade-offs between the two objectives.

Manufacturers produce machinery and equipment, along with their components, and provide after-sales maintenance services for sold products. During the warranty period, manufacturers offer customers complimentary replacement parts and repair services. Once products are out of warranty, maintenance becomes a paid service, requiring customers to cover the costs of both parts and labor. As shown in Figure 1, A manufacturer currently operates F permutation flowshops distributed across different geographical locations within its service area. Each line can process T types of spare parts and is equipped with m machines. Processing times for identical spare parts vary across workshops and machines. These F workshops collectively support D ($D \geq F$) repair demand points, meaning each production site may supply spare parts to multiple demand points. Delivery times from different production sites to the same demand point also vary. The status of sold products changes dynamically over time, generating two distinct types of service orders at the demand points: in-warranty and out-of-warranty orders. In-warranty orders entitle customers to complimentary parts and repair services from the manufacturer, whereas out-of-warranty orders require customers to bear the corresponding parts and labor costs. Furthermore, the manufacturer operates a central service station within the region, staffed by M repair technicians with equivalent skill levels. This team of technicians is shared among all D service points. Technicians depart from the service station for their first assigned task and subsequently travel between different service points according to the schedule. At any given time, the manufacturer receives n repair orders from the service points, comprising v in-warranty orders and $(n-v)$ out-of-warranty orders. The required spare parts are delivered directly to each service point upon completion of production. Each repair order requires J_i spare parts of a specific type and must be assigned to a single technician.

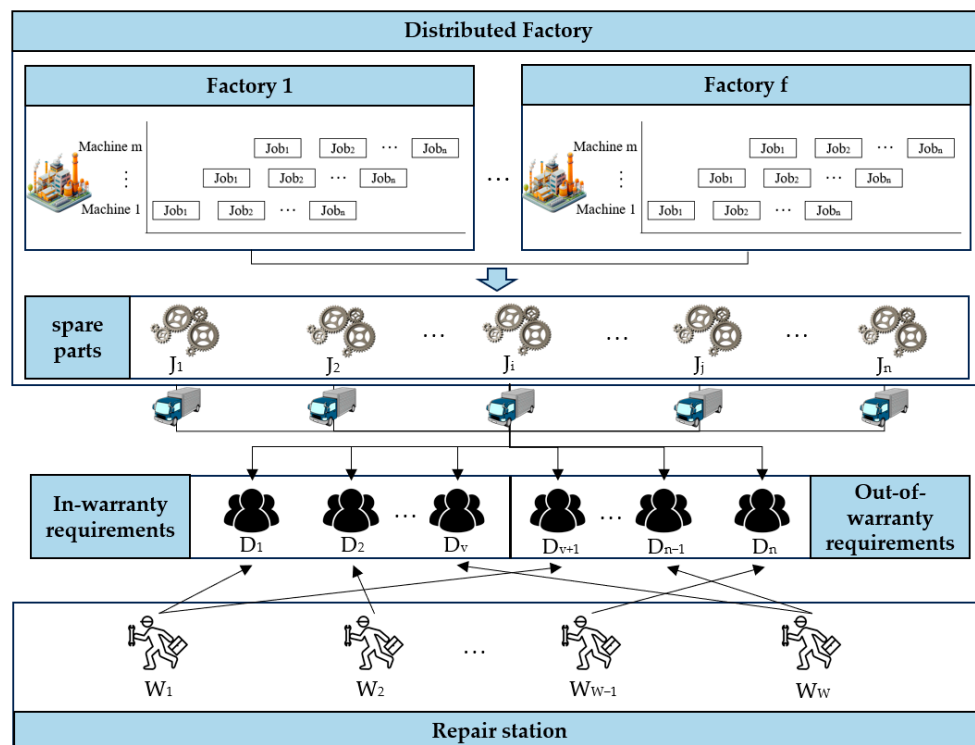


Figure 1. Integrated scheduling scenario for spare parts production and maintenance personnel considering differentiated customer needs.

Each order i corresponds to an ideal delivery time window $[D_i^-, D_i^+]$. The maximum acceptable waiting time for spare part delivery for this order is D_i^+ . Each order can only commence repair after both spare parts and workers arrive and must be completed after D_i^- . If an order's delivery time exceeds the maximum acceptable wait time D_i^+ , the customer satisfaction for in-warranty orders and the profit for out-of-warranty orders will decline. Should the delay exceed the customer's maximum waiting time tolerance WT_i , the out-of-warranty order is considered lost, resulting in corresponding profit loss. In-warranty orders, while not lost, experience a continuous decline in customer satisfaction. The relationship between satisfaction, total profit, and time is shown in Figure 2a, and Figure 2b, respectively. To differentiate order importance, this study assumes that even when orders require identical spare parts, the profit generated from completing repairs may vary, reflecting differences in service priority.

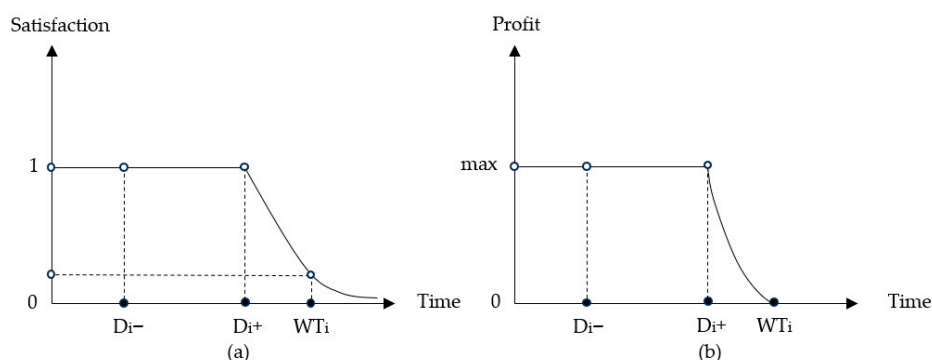


Figure 2. The relationship between target value and service time. (a) Relationship between customer satisfaction with orders during the warranty period and service time. (b) Relationship between total profit from orders outside the warranty period and service time.

3.2. Mathematical Model

Mathematical description:

To better describe the ISSPD problem, we have established a mixed-integer programming (MIP) model.

The metrics, parameters, and decision variables used in this model are detailed below:

- n : Total number of orders/Total number of demand locations.
- f : Total number of factories.
- m : Total number of machines.
- j : Total number of local service providers.
- w : Total number of service workers.
- r : Total number of spare part types.
- D : Total number of required venues.
- N : Collection of orders, $N = 1, \dots, n$.
- F : Collection of factories, $F = 1, \dots, f$.
- W : Collection of workers, $W = 1, \dots, w$.
- i, c : Index of orders, $i, c = 1, 2, \dots, v, v + 1, \dots, n$; the first v represent orders within the warranty period. The following $n - v$ represent orders outside the warranty period.
- J_i : The i -th order.
- SV : Index of orders within the warranty period, $SV = 1, 2, \dots, v$.
- SW : Index of orders outside the warranty period, $SW = v + 1, v + 2, \dots, n$.
- k : Index of factories, $k = 1, 2, \dots, f$.
- g : Index of service workers, $g = 1, 2, \dots, w$.
- t : Index of Models, $t = 1, 2, \dots, r$.
- q : The index of each factory location (process), $q = 1, 2, \dots, n$.
- l : Index of machines in each factory, $l = 1, 2, \dots, m$.
- J : Index of local service providers, $J = 1$.
- p : Sequence index for service personnel performing maintenance tasks, $p = 1, 2, \dots, n$.
- T_i : The type t to which order J_i belongs.
- P_{it} : Processing time for order J_i on machine l .
- PT_{it} : Maintenance time required for technicians to replace t -type spare parts for order J_i .
- TF_{ki} : The time it takes for order J_i to be transported from factory k to demand point i .
- TB_{ji} : Worker transfer time between local service provider J and demand location i .
- TBB_{ic} : Worker transfer time between demand field point i and demand field point c , primarily determined by the geographical distance between orders and the mode of transfer.
- $[D_i^-, D_i^+]$: Ideal spare part delivery time window for order J_i .
- WT_i : The maximum acceptable waiting time tolerance for customer J_i . When the repair time exceeds this threshold, the out-of-warranty order is lost.
- $price_i$: Profit generated by order J_i .
- $rate_i$: Profit decay coefficient for order J_i .
- DC : Calculating transportation costs is usually determined by a combination of unit transportation costs and transportation time.
- B : A sufficiently large number.
- Decision variables:**
 - ST_{it} : Start processing time for order J_i on machine l .
 - CT_{it} : Completion time for order J_i on machine l .
 - RT_i : Actual delivery time for order J_i .
 - WAT_i : Worker arrival time for order J_i .
 - WST_i : Maintenance start time for order J_i .

WET_i : Maintenance completion time for order J_i .

$object1_i$: Satisfaction value for order J_i .

$object2_i$: Maintenance profit value for order J_i .

x_{ikq} : 1 if order J_i is scheduled at position q in factory k ; 0 otherwise.

z_{igp} : 1 if order J_i is the p -th task serviced by worker g ; 0 otherwise.

A_i : 1 if the outbound order is lost; 0 otherwise.

Objective function:

$$ob1 = \max \sum_{i=1}^v object1_i \quad (1)$$

$$ob2 = \max \sum_{i=v+1}^n object2_i \quad (2)$$

$$WST_i \leq D_i^+, object_i = 1, \forall i \in SV \quad (3)$$

$$WST_i > D_i^+, object1_i = \left(\exp \left(-\frac{WST_i - D_i^+}{WT_i - D_i^+} \right) \right), \forall i \in SV \quad (4)$$

$$WST_i \leq D_i^+, object2_i = price_i \cdot (1 - A_i) \quad (5)$$

$$WST_i > D_i^+, object2_i = Price_i \cdot \left(\exp \left((-rate_i) \cdot \max \left(0, \frac{WST_i - D_i^+}{WT_i - D_i^+} \right) \right) \right), \forall i \in SW \quad (6)$$

Constraints:

$$\sum_{k=1}^f \sum_{q=1}^n x_{ikq} \leq 1, \forall i \quad (7)$$

$$\sum_{k=1}^f \sum_{q=1}^n x_{ikq} + A_i = 1, \forall i; A_i = 0, \forall i \quad (8)$$

$$\sum_{i=1}^n x_{ikq} \leq 1, \forall k, q \quad (9)$$

$$\sum_{i=1}^n x_{ikq} \leq \sum_{i=1}^n x_{ik(q-1)}, \forall q \in \{2, \dots, n\}, k \quad (10)$$

$$ST_{il} \geq CT_{cl} + B(x_{ikq} + x_{ck(q-1)} - 2), \forall q \in \{2, \dots, n\}, i, c, k, l \quad (11)$$

$$ST_{i(l+1)} \geq CT_{il}, \forall l \in \{1, 2, \dots, m-1\}, i \quad (12)$$

$$CT_{il} = ST_{il} + \sum_{k=1}^f \sum_{q=1}^n x_{ikq} \cdot P_{il}, \forall i, l \quad (13)$$

$$RT_i = CT_{im} + \sum_{k=1}^f \sum_{q=1}^n x_{ikq} \cdot TF_{ki}, \forall i \quad (14)$$

$$\sum_{g=1}^w \sum_{p=1}^n z_{igp} \leq 1, \forall i \quad (15)$$

$$\sum_{g=1}^w \sum_{p=1}^n z_{igp} + A_i = 1, \forall i \quad (16)$$

$$\sum_{i=1}^n z_{igp} \leq 1, \forall g, p \quad (17)$$

$$\sum_{i=1}^n z_{igp} \geq \sum_{i=1}^n z_{ig(p-1)}, \forall p \in \{2, \dots, n\}, g \quad (18)$$

$$WAT_i \geq TB_{ji} \cdot z_{ig1}, \forall i, g \quad (19)$$

$$WST_i = \max\{RT_i, WAT_i, D_i^-\}, \forall i, g, p \quad (20)$$

$$WST_i \leq WT_i \cdot \sum_{g=1}^w \sum_{p=1}^n z_{igp}, \forall i, g, p \quad (21)$$

$$WST_i - WT_i \leq \sum_{g=1}^w \sum_{p=1}^n z_{igp}, \forall i \quad (22)$$

$$WET_i = WST_i + \sum_{g=1}^w \sum_{p=1}^n z_{igp} \cdot PT_{ti}, \forall i \quad (23)$$

$$ST_{il} \geq 0, WST_i \geq 0, WT_i \geq 0, \forall i, l, g \quad (24)$$

Equations (1) and (2) define the two objective functions, representing the satisfaction of in-warranty orders and the total profit from out-of-warranty orders, respectively. Equations (3) and (4) provide the detailed expansions of Equation (1), while Equations (5) and (6) elaborate on Equation (2). Constraint (7) ensures that each operation of an order is assigned to exactly one processing location within a factory. Constraint (8) states that if an out-of-warranty order is lost, no spare parts production is required for it. Constraint (9) restricts each processing location in a factory to scheduling at most one workpiece at a time. Constraint (10) specifies that no vacant location may exist before a scheduled order within the same factory's sequence. Constraint (11) enforces the processing sequence constraints on the same machine. Constraint (12) guarantees the continuity between preceding and succeeding processes for the same order. Constraint (13) defines the relationship between the processing start time and completion time. Constraint (14) calculates the actual arrival time of an order when the spare parts are supplied from the factory. Constraint (15) limits the maintenance of any order to at most one worker. Constraint (16) stipulates that if an in-warranty order is lost, no maintenance worker needs to be scheduled for it. Constraint (17) ensures that each worker can perform only one repair task at any given time at their assigned position. Constraint (18) guarantees that no vacant positions exist

before any occupied processing positions in a worker's schedule. Constraint (19) defines the time required for repair worker g to reach their first task location as the travel time from the local service center to demand point i . Constraints (20) and (21) calculate the actual repair start time for orders. Constraint (22) specifies that maintenance on an order can only begin if both the spare part has arrived and the worker has arrived within the order's maximum waiting time tolerance. Constraint (23) defines the relationship between the maintenance start time and completion time for each worker-order assignment. Constraint (24) sets the initial conditions for the system.

4. Proposed QLNSGA

This section details the proposed Q-learning-based NSGA-II algorithm (QLNSGA). First, the overall framework of QLNSGA is outlined in Section 4.1. The codec design is described in Section 4.2. Section 4.3 introduces the design of cross-over and mutation strategies. Section 4.4 introduces the design of eight knowledge-driven local search strategies, while Section 4.5 presents the design of a Q-learning mechanism for dynamically selecting key local search strategies.

4.1. Algorithm Framework

Currently, the NSGA-II genetic algorithm and its improved variants have been extensively applied to workshop scheduling-related problems [30,31], owing to NSGA-II's superior global search capability and efficient non-dominated sorting rules. This paper designs an algorithm based on the Q-learning mechanism for solving ISSPD within the structural framework of NSGA-II, termed QLNSGA.

The overall flowchart of the QLNSGA algorithm is illustrated in Figure 3. The process begins with the random generation of an initial population, which is then decoded using knowledge-driven rules (e.g., order loss reconstruction, first-in-first-process) to construct a high-quality initial solution set. Following this, the initial solutions undergo non-dominated sorting and crowding distance calculation to update the elite archive. Cross-over and mutation operations are applied to this elite population to expand the global search space. The resulting offspring population is then re-decoded and undergoes environmental selection. The high-quality individuals filtered from this step are again processed through non-dominated sorting and crowding distance calculation to update the population, after which local search operations are performed. We designed eight local search strategies tailored to the characteristics of the ISSPD problem and integrated them with a Q-learning mechanism. This integration enables the dynamic selection of the most suitable local search strategy, thereby enhancing the algorithm's neighborhood search capability. This approach combines the adaptive decision-making of Q-learning with the global search advantage of the genetic algorithm, forming a collaborative optimization framework. After the local search, the population is once more subjected to non-dominated sorting and crowding distance calculations to select a new elite population. This elite population is evaluated against predefined termination criteria. If the criteria are met, the results are output; otherwise, the process returns to the initialization phase to begin a new optimization cycle.

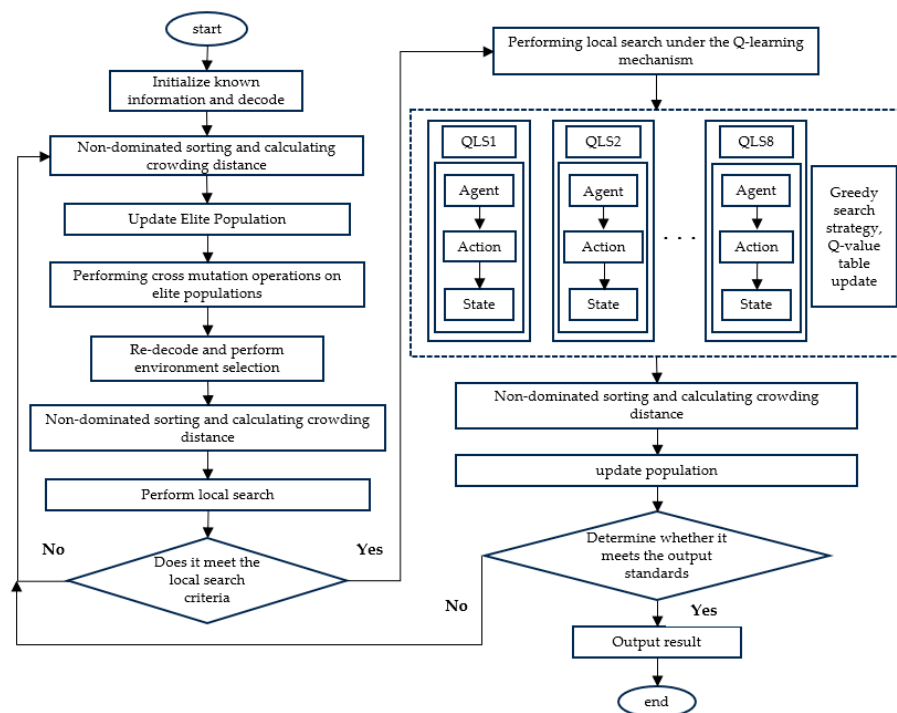


Figure 3. Organizational framework diagram of QLNSGA.

4.2. Encoding and Decoding

To address the ISSPD problem, this paper divides it into two phases: production and maintenance. The production phase assigns processing factories to spare parts and determines the processing sequence within each factory. The maintenance phase assigns workers to repair orders and determines the repair sequence for each worker. To represent a complete solution, this paper designs a four-dimensional encoded vector comprising: Factory Assignment Sequence (FS), Spare Part Processing Sequence (FOS), Worker Assignment Sequence (WOS), and Spare Part Maintenance Sequence (WS). The structure of this four-dimensional vector is shown in Figure 4, illustrating a coding example comprising three factories, eight spare parts, and four workers.

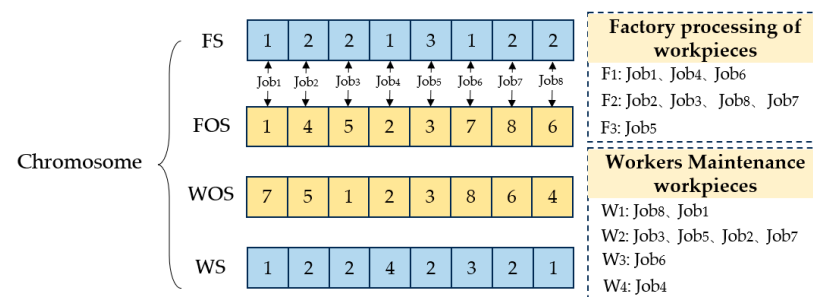


Figure 4. Chromosome coding.

Factory Assignment Sequence (FS): Indicates the processing factory to which each spare part is assigned. For example, the code “12213122” signifies that spare parts numbered 1, 4, and 6 are processed at Factory 1, while spare parts numbered 2, 3, 7, and 8 are processed at Factory 2, and so on.

Spare Part Processing Order (FOS): Indicates the processing sequence of each spare part on machines within its assigned factory. The code represents the processing order from left to right, with total length equal to the order quantity.

Worker Assignment Sequence (WS): Indicates the service workers assigned to each maintenance order, using a coding principle similar to FS.

Spare Part Maintenance Sequence (WOS): Defines the maintenance sequence of spare part orders for each assigned worker.

To illustrate the decoding mechanism more intuitively, Figure 5 shows an example of decoding the chromosome in Figure 4. In Factory F1, spare parts 1, 4, and 6 must be processed sequentially on machines M1, M2, and M3 according to the production order 1, 2, 3. Each machine requires a preparation period before operation. If the next machine's spare part is not yet processed, the completed spare part on the current machine must wait for its completion before proceeding to subsequent processing. Production efficiency varies among machines in different factories, resulting in differing total processing times across facilities.

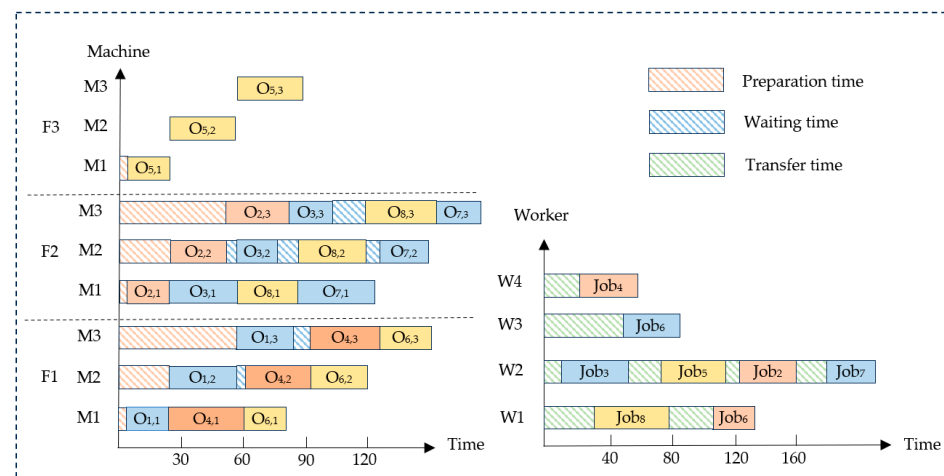


Figure 5. Decoding mechanism.

During decoding, considering the varying needs of customers, order loss may occur when servicing customers outside the warranty period. If an order is lost, the FOS, WOS, and WS layer values for the lost order will be reset to zero. Re-decoding will then proceed using strategies such as first-come-first-served and earliest processing completion time. Figure 6 illustrates an example scenario: if Orders 3 and 7 are lost, the FOS, WOS, and WS layer values corresponding to these orders become zero. Consequently, the processing sequence for spare parts at Factory 2 undergoes the production order adjustments depicted in Figure 6, and the worker maintenance sequence will also be modified accordingly.

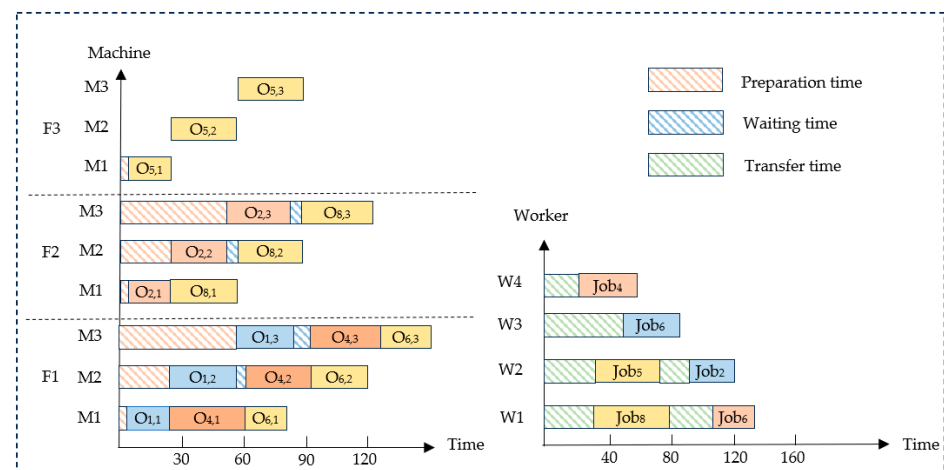
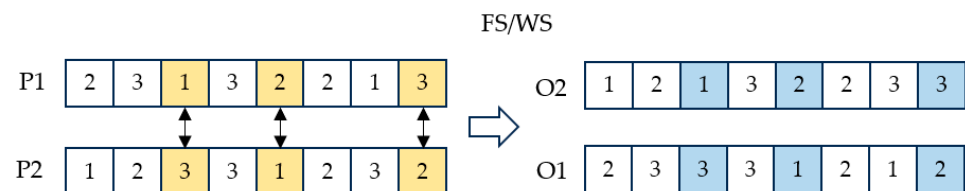


Figure 6. Re-decoding mechanism.

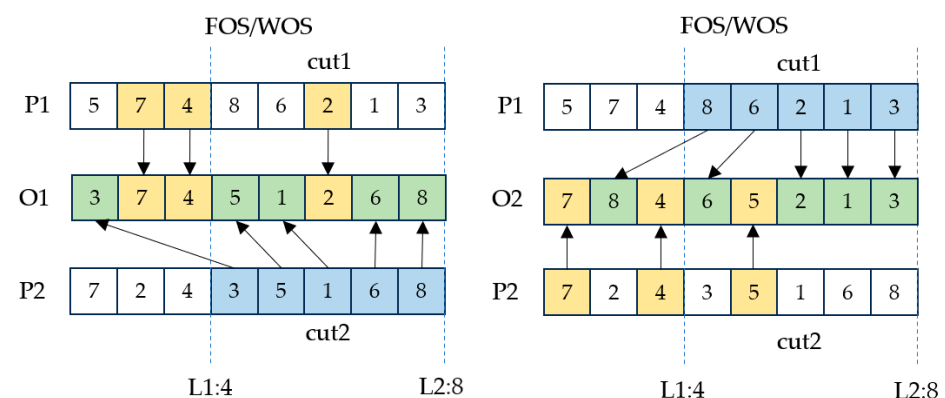
4.3. Cross-over and Mutation

To enhance the algorithm's global exploration capability and expand solution space diversity, this study designed a cross-over strategy tailored for four-layer chromosome structures. This strategy facilitates information exchange between parent solutions to generate offspring. As a core component of genetic algorithms (such as NSGA-II, NSGA-III, etc.) [32], cross-over typically involves recombination of two parent chromosomes through operations such as insertion, transposition, or inversion.

In this study, two crossing methods, two-point crossing and sequence-based crossing, were designed to address the different characteristics of the four layers of chromosomes. Specific diagrams will be used for illustration below. For the factory selection layer (FS) and worker selection layer (WS), two-point crossover is applied. Taking Figure 7 as an example, a random integer not exceeding the chromosome length (3 in this case) is first generated. Subsequently, corresponding crossover positions are randomly selected ({3, 5, 8} in this example). Parent chromosomes P1 and P2 exchange gene values at these positions, thereby generating offspring O1 and O2.

**Figure 7.** FS/WS cross operation.

For the factory operation sequence (FOS) and worker operation sequence (WOS) within the factory, a sequence-based crossover method is applied. Two gene positions are randomly generated without exceeding the gene length. Taking Figure 8 as an example, the two gene positions are designated as L1 and L2. The gene segments between L1 and L2 are identified and named cut1 and cut2, respectively. Genes identical to cut2/cut1 within P1/P2 are removed. Subsequently, cut2/cut1 is inserted into the vacated positions while preserving the original sequence order. Genes outside these segments remain unchanged. It should be noted that all crossover operations are executed with the same crossover probability p_c .

**Figure 8.** FOS/WOS cross operation.

To enhance the algorithm's global search capability and prevent convergence to local optima, this study introduces mutation operations following crossover operations. The mutation strategy fundamentally differs from the cross-over strategy in its mechanism, aiming to maintain population diversity through local perturbations. Two mutation strategies were designed to address the distinct characteristics of the four-layer chromosome structure:

For the spare parts factory selection layer (FS) and worker assignment sequence layer (WS), inverted mutation is applied. This involves randomly selecting a gene segment and reversing its order. For the spare parts production sequence layer (FOS) and worker-part maintenance sequence layer (WOS), two-point exchange mutation is employed. This technique randomly selects two gene positions and swaps their values. Detailed explanations are provided in Figure 9.

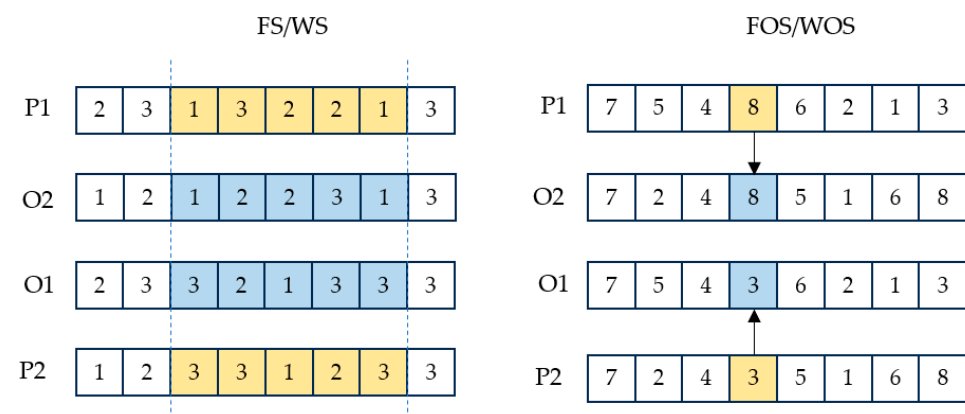


Figure 9. Mutation operations.

Due to potential loss of out-of-warranty orders (corresponding to gene values of 0) during maintenance, genes with values of 0 may appear in the FOS and WOS layers after cross-over and mutation. Failure to correct these values will generate invalid solutions, compromising the feasibility of subsequent iterations. To address this, this study introduces a decoding correction mechanism after cross-over and mutation operations to maintain the validity and stability of the encoding layer. The pseudocode for the specific correction logic is shown in Algorithm 1.

Algorithm 1 Maintenance strategy

Input: Population child = [child_fs; child_os; child_ps; child_ws], warranty period orders warranty, spare parts quantity n

1. **For** j = 1: warranty do % Correct orders of the specified type within the warranty period
 2. **If** child(2,j) = 0 and child(3,j) = 0 do
 3. Identify missing orders and rearrange their processing sequence.
 4. **End if**
 5. **End for**
 6. **For** j = warranty + 1:n do % Correct orders for out-of-warranty items
 7. **If** child(2,j) = 0 and child(3,j) ≠ 0 do
 8. If the loss is due to delays in the production stage, the lost orders will be rescheduled for production.
 9. **Else if** child(2,j) ≠ 0 and child(3,j) = 0 do
 10. If the loss is due to delays during the repair phase, the lost orders will be rescheduled for repair. No corrections will be made for other circumstances.
 11. **End if**
 12. **End for**
 13. **Output:** The new population child after correction
-

4.4. Design of Local Search Strategies

To enhance the algorithm's local search capability and avoid getting stuck in local optima [33], this paper designs eight problem-specific local search strategies (*LS1–LS8*). These strategies are developed by comprehensively considering the trade-off between customer satisfaction and supplier profit. Their core idea is to perturb the solution in a way that optimizes one objective function while keeping the other constant or similarly improved, thereby driving the population toward a higher-quality Pareto front. The specific designs are as follows:

LS1: Targets orders lost due to exceeding the maximum waiting time (affecting objective 2). These orders are sorted in descending order of their waiting time tolerance (WT_i). For the most urgent order, the strategy identifies whether the loss originated in the production or maintenance phase. It then swaps the sequence of this order with a non-lost order in the corresponding production or maintenance sequence. The population is updated based on the weighted improvement of both objective functions. A detailed description is provided in Algorithm 2.

LS2: Focuses on delayed in-warranty orders (impacting objective 1). These orders are sorted in descending order of WT_i . For a selected order, the strategy identifies the phase causing the delay. If it is the production phase, its processing sequence is swapped with a non-delayed order processed after it in the same factory. If it is the maintenance phase, its maintenance sequence is swapped with a non-delayed in-warranty order that follows it in the same worker's schedule.

LS3: Randomly selects an out-of-warranty order. If the order was not lost, it is reassigned to a different factory. If it was lost, an attempt is made to reassign it to the factory with the shortest delivery time to restore service.

LS4: Identifies the workpiece with the longest completion time and reassigns it to another factory to alleviate load imbalance among factories.

LS5: Randomly selects two workpieces from the factory with the longest total processing time. A swap of their sequences is permitted under two conditions: (1) if both are in-warranty or both are out-of-warranty; (2) if one is in-warranty and the other is out-of-warranty, provided the out-of-warranty order is not lost. Otherwise, a new selection is made.

LS6: Randomly selects a workpiece. If it is the first task in its assigned maintenance worker's schedule and was completed within its delivery deadline, it becomes eligible for a swap. A target pool is created based on warranty status matching, and an appropriate workpiece is randomly selected from this pool for exchange.

LS7: Identifies workpieces completed early and utilizes the freed-up resources (e.g., machine or worker time) to reschedule delayed orders, thereby reducing overall order tardiness.

LS8: Selects the most time-consuming order from the worker with the longest total maintenance time and reassigns it to the worker with the shortest total maintenance time to balance the workload among workers.

Algorithm 2 *LS1* local search strategy

Input: population, population size, customer satisfaction weight $\alpha = 0.5$, revenue weight $\beta = 1 - \alpha$

1. **For** $I = 1$: pop **do**
 2. Decode individuals to obtain chromosomes and objective values, then compute weighted fitness: $fit0 = \alpha \times obj0(1) + \beta \times obj0(2)$
 3. Filter out missing orders and sort them in descending order by urgency to obtain the array idx
 4. **If** idx is a non-empty set, do the following for each order in the set: Filter out the factory with the shortest shipping time and reassign the order
 5. **If** production completion time > delay caused by urgency, then execute critical block swapping strategy for
-

```

production sequence
6.   If the improvement is achieved, update the population and proceed to process the next order
7.   Else if filter out order idx1 based on delay level
8.       For ii = 1: idx1 do
9.           Based on the order type, choose to adjust the processing sequence or delete the order, re-encode,
           and update the population accordingly based on changes in target values
10.          If improvement is achieved, update the population
11.          Break
12.          End
13.      End
14. Else if Maintenance completion time > Delays caused by urgency do execute critical block swap strategy for
           maintenance sequence
15.      If improved, update the population and proceed to process the next order
16.      Else if Filter orders by delay level idx1
17.          For ii = 1: idx1 do
18.              Based on the order type, choose to adjust the processing sequence or delete the order, re-
              encode, and update the population accordingly based on changes in target values
19.              If improvement is achieved, update the population
20.              Break
21.              End
22.          End
23. End
24. End
25. Output: Population generated by LS1 rule

```

4.5. Local Search Selection Based on Q-Learning

To enhance the efficiency of local search algorithms and reduce computational resource waste, this paper designs a value function-based Q-learning mechanism for dynamically selecting local search strategy. This Q-learning mechanism is embedded within the QLNSGA-II algorithm framework, with its operational logic illustrated in Figure 10. Within this Q-learning mechanism, a 4×8 Q-table is constructed to record the state-action value function. The eight actions correspond to the eight local search strategies *LS1* to *LS8*, while the four states reflect the dynamic characteristics of the optimization objective. All initial values in the Q-table are set to zero and undergo iterative optimization through reward feedback and policy update mechanisms. The reward function evaluates changes in the objective function state following different actions, thereby guiding the algorithm toward selecting more effective search strategies. The detailed design of the reward mechanism and action selection strategy will be further elaborated below.

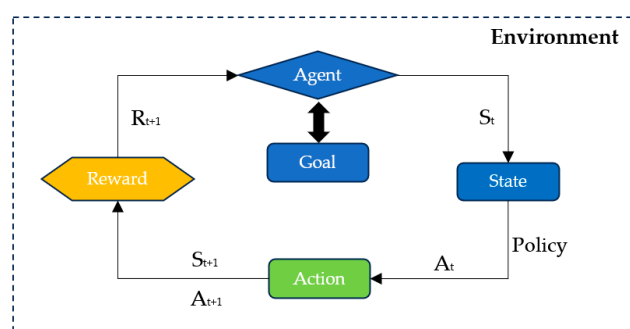


Figure 10. Logic of the Q-learning mechanism.

Action set: The actions in the Q-table correspond to the eight local search strategies LS1-LS8 described in Section 4.4.

State set: The state can be represented by the algorithm's convergence and distribution properties, denoted as metrics Z and X respectively. These are calculated using Equations (25) and (26). A smaller Z value indicates better convergence, while a larger X value indicates better distribution. According to Li et al. [34], when an action change occurs, the corresponding state variables can undergo four distinct combinations. For state S , the four state variable combinations are: (1) $S1: \Delta Z > 0, \Delta X > 0$; (2) $S2: \Delta Z > 0, \Delta X \leq 0$; (3) $S3: \Delta Z \leq 0, \Delta X > 0$; (4) $S4: \Delta Z \leq 0, \Delta X \leq 0$. Among these four states, state 3 represents the most ideal scenario.

$$Z(P, P^*) = \sqrt{\sum_{y \in P} \min_{x \in P^*} \text{dis}(x, y)^2} / |P| \quad (25)$$

Here, P denotes the Pareto front PF at time t within each generation, and P^* represents a set of uniform reference points.

$$X = \sum_{i=1}^{N-1} |d_i - \bar{d}| / (n - 1) \bar{d} \quad (26)$$

Here, d_i denotes the Euclidean distance between two adjacent individuals in PF at time t , $\bar{d}$ is the average value of d_i .

Reward set: After each execution of the actions, the Pareto front solution set is updated, altering the convergence and diversity of the population. The reward value R_t at time t is computed using formula (27). The Q-table is updated based on reward value R_t and formula (28), where α represents the learning rate, denoting the probability of performing Q-learning at each step, and γ is the discount factor, emphasizing recent rewards. Due to the accelerated convergence capability of the local search strategy, this reward scheme prioritizes algorithmic diversity.

$$R_t = \begin{cases} 10, \Delta Z > 0 \text{ and } \Delta X > 0 \\ 0, \Delta Z \leq 0 \text{ and } \Delta X \leq 0 \\ 5, \text{else} \end{cases} \quad (27)$$

$$Q_t(S_t - A_t) \leftarrow Q_t(S_t - A_t) + \alpha \cdot (R_t + \gamma \cdot \max Q_t(S_{t+1}, :) - Q_t(S_t - A_t)) \quad (28)$$

ε -greedy strategy: To prevent QLNSGA-II from getting stuck in local optima, an ε -greedy strategy is designed with the calculation formula $\varepsilon = 1 - e^{t/T_{max}}$, where T_{max} represents the termination time. As shown in Equation (29), the ε -greedy strategy guides the algorithm to conduct extensive searches in the early stages and gradually shifts toward more efficient method selection in the later stages.

$$A_t = \begin{cases} A'_t(S_t), V_{rand} < \varepsilon \\ \text{random}(A_t(S_t)), V_{rand} \geq \varepsilon \parallel Q(S_t, :) == 0 \end{cases} \quad (29)$$

Implementation Example: To better understand how the Q-learning mechanism guides the selection of local search strategies, an example is provided in Figure 11.

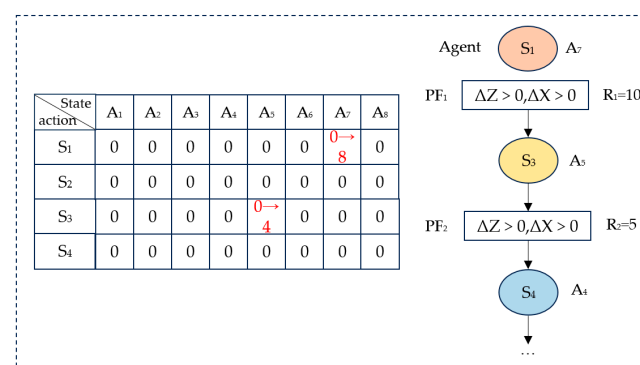


Figure 11. Example of a Q-learning mechanism.

Starting from initial state $S1$, since $Q(S1, :) = 0$ at the beginning, a random action is selected. Assuming the randomly chosen action is $A7$, with the input Pareto front being $PF1$, we can compute ΔZ and ΔX . Since $\Delta Z > 0$ and $\Delta X < 0$, $Q(S1, A7)$ yields a reward value $R1 = 10$, thus determining the next state as $S3$. Then, using formula (27), we calculate $Q(S1, A7) = Q(S1, A7) + 0.8 \times (10 + 0.8 \times \max(Q(S3, :))) - Q(S1, A7) = 8$. Proceeding to the second iteration: current state is $S3$. Random action $A5$ is selected. Calculating $PF2$ yields $\Delta Z < 0$ and $\Delta X < 0$, confirming $Q(S3, A5)$ has reward value $R2 = 5$. $Q(S3, A5) = 4$, leading to next state $S4$. This mechanism updates the Q-table iteratively until convergence criteria are met.

5. Experimental Results and Discussion

The primary objective of this section is to validate the effectiveness of the proposed algorithm. All experiments were conducted on a PC equipped with an Intel Core i9-9900K CPU running at 3.60 GHz and 32 GB RAM, utilizing MATLAB R2016b. This section will cover the following aspects: selection of test cases, parameter calibration, comparison of the improved strategy with the original algorithm, and comparison with other algorithms.

5.1. Construction of Instances and Setting of Performance Indicators

The benchmark dataset only includes partial features relevant to the problem under study. To comprehensively evaluate the performance of the proposed algorithm, we reference the benchmark instance generation method by Wang et al. [35]. Four key variables—the number of workpieces, factories, processing machines, and maintenance workers—are selected to construct diverse test scenarios. Specific parameter settings are as follows: $n \in \{50, 100, 150\}$, $F \in \{3, 4, 5\}$, $m \in \{6, 8, 10\}$, $w \in \{6, 8, 10\}$. Based on the scale $n \times F \times m \times w$, 81 instances of varying sizes were generated, named as $n \times F \times m \times w$. Each scale was further randomly generated into 5 sub-instances, resulting in a total of 405 test instances. Other parameters were randomly generated as shown in Appendix Table A1. The ideal delivery window and latest delivery time are calculated using Equations (30)–(33) below. To ensure fairness in algorithm comparisons, each test case was independently run 10 times. Following the approach in [36], the maximum CPU time per run was set to $T_{max} = n \times m \times F \times w \times 0.01$ s. This setting reasonably accounts for computational resource consumption based on problem scale, ensuring all algorithms are compared under identical time constraints.

$$finishtime = \max(\sum(P_{it}), TF_i) + \sum(TF_i)/5 \quad (30)$$

$$D_i^- = (0.8 + 0.5 \cdot rand) \cdot finishtime$$

$$D_i^+ = D_i^- + PT_{ti} \cdot (1 + 3 \cdot \frac{1+9 \cdot rand}{50}) \quad (31)$$

$$Price_i = price_i \cdot (1 + 0.5 \cdot rand) \quad (32)$$

$$WT_i = (1 + 0.5 \cdot rand) \cdot D_i^+ \quad (33)$$

Based on the approach of Jing et al. [33] and adapted to the specific context of this study, we established the ideal delivery window for each generated instance using Equations (30) and (31).

To comprehensively evaluate all algorithms, this study selected three metrics: Inter-generational Distance (IGD) [37], Hypervolume (HV) [38], and Solution Set Coverage (C)

[20]. These metrics respectively represent the algorithms' distributivity, convergence, and quantity aspects.

IGD: A performance metric measuring the distance between obtained solutions and known optimal solutions, reflecting algorithmic diversity and convergence. For each solution y in the optimal Pareto front of the actual solution set $\{A\}$ generated by the algorithm, find the closest solution x on the theoretically optimal Pareto front PF^* . Calculate the Euclidean distance $d(x, y)$ between them. Since the theoretical optimal frontier is difficult to obtain. Therefore, the aggregated Pareto front solutions from all algorithms are designated as PF^* . Before calculating IGD, the objective values of all solutions must be normalized, where i represents the number of objectives. A smaller IGD value indicates better diversity and convergence of the algorithm.

$$IGD(A, PF^*) = \frac{1}{|PF^*|} \sum_{x \in PF^*} \min_{y \in A} d(x, y) \quad (34)$$

$$f(x) = \frac{f_i(x) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}, i = 1, 2 \quad (35)$$

HV: Utilizes reference points and solution sets to measure the volume of the objective space. Each solution on the Pareto front is compared against the reference points to calculate its contribution to the hypervolume. Subsequently, the HV values for all frontier solutions are computed, with their magnitude reflecting the overall performance of the solution set. Similar to calculating IGD, objectives must be normalized during HV computation, where ref denotes different reference points. A higher HV value indicates superior overall performance of the solution set.

$$HV(ref, A) = \Lambda(\cup_{f \in A} [f_1, ref_1] \times \dots \times [f_r, ref_r]) \quad (36)$$

C: Assessing the dominance relationship between the solution sets of two algorithms reflects the advantages in the number of Pareto front solutions and algorithm convergence. Let A and B represent the Pareto front solutions of two algorithms, respectively. A larger $C(A, B)$ indicates higher quality of solution set A . However, two special cases warrant additional explanation: When $C(A, B) = 1$, it means all solutions in solution set B can be dominated by any solution in solution set A . When $C(A, B) = 0$, it indicates that no solution in set B can be dominated by any solution in set A .

$$C(A, B) = \frac{|\{x \in B | \exists y \in A: y \text{ dominates } x\}|}{|B|} \quad (37)$$

In summary, the smaller IGD and larger HV and C values indicate that the algorithm is more competitive.

5.2. Parameter Calibration

The configuration of algorithm parameters significantly impacts performance and solution quality. To ensure fairness and scientific rigor in parameter tuning, this study employs Taguchi experimental design methodology [39] for parameter calibration across all comparison algorithms. This approach robustly identifies the parameter combinations most influential to performance with minimal experimental iterations. For the proposed QLNSGA algorithm, five key parameters were calibrated: population size pop , cross-over probability pc , mutation probability pm , learning rate α , and discount factor γ . Each parameter was set at four levels: $pop \in \{60, 80, 100, 120\}$, $pc \in \{0.7, 0.8, 0.9, 1.0\}$, $pm \in \{0.1, 0.2, 0.3, 0.4\}$, $\alpha \in \{0.6, 0.7, 0.8, 0.9\}$, $\gamma \in \{0.6, 0.7, 0.8, 0.9\}$, yielding 16 combinations. For each parameter combination, nine randomly selected cases of varying scales were chosen from the 81 cases constructed in Section 5.1 to form the calibration case set for the Taguchi experiment. All experimental results were evaluated using the IGD metric, with the outcomes presented in Appendix Table A2.

Following the Taguchi experimental design method described in Section 5.1, the selected 9 test cases were independently run 10 times under each of the 16 parameter combinations of the DQNSGA. This yielded a total of $16 \times 9 \times 10 = 1440$ experimental results. Subsequently, evaluating these results produced 1440 IGD calculations. The mean IGD values for different test cases under the same parameter combination were calculated. Using Minitab 19 statistical software to perform an analysis of variance (ANOVA) on the experimental results yielded the main effects plot of parameters shown in Figure 12. The figure displays a factor trend plot with the mean IGD value as the response variable. Smaller IGD values indicate better algorithm performance. In Figure 12, we observe that for pop , selecting the third level ($pop = 100$) yields the minimum IGD value. Therefore, $pop = 100$ is chosen as the standard parameter for QLNSGA. Similarly, when parameters are set to $pc = 0.7$, $pm = 0.3$, $\alpha = 0.8$, $\gamma = 0.8$, the QLNSGA exhibits optimal performance. Consequently, this parameter combination can be applied to subsequent experimental designs involving QLNSGA.

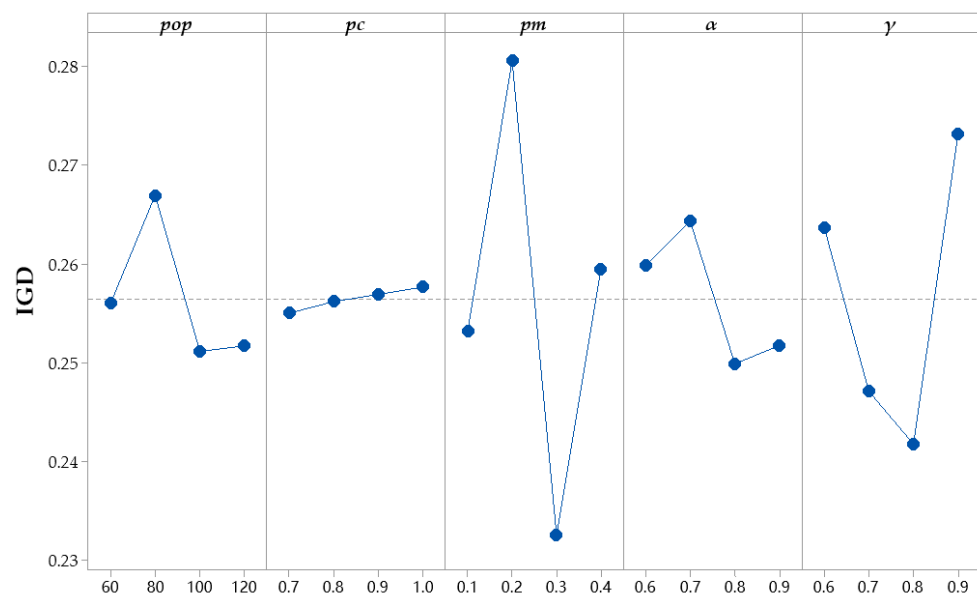


Figure 12. Main effects plot for QLNSGA parameter calibration.

5.3. Effectiveness of Improvement Strategies Within Their Own Algorithms

This paper proposes a novel hybrid algorithm named QLNSGA, which enhances the classic multi-objective optimization algorithm NSGA-II by incorporating eight problem-specific local search operators and an adaptive selection mechanism based on Q-learning. To systematically validate the effectiveness of the proposed strategies, two controlled variant algorithms are constructed: first, the variant without Q-learning guidance, NQGA, which removes the Q-learning decision module from QLNSGA and instead randomly selects a local search operator in each generation, aiming to isolate and evaluate the independent contribution of Q-learning in dynamically guiding the search direction and improving convergence efficiency; second, the variant without problem-specific local search, NLGA, which retains only the basic genetic operators (selection, crossover, mutation) and removes all eight problem-driven local search operators, aiming to verify the baseline effectiveness of the designed chromosome encoding and genetic framework. Through extensive computational experiments on numerous test instances, QLNSGA is compared with NQGA and NLGA respectively, in order to demonstrate the independent contributions of the Q-learning mechanism and the local search operators. All algorithms are

calibrated using the same Taguchi experimental design method, with the calibration results presented in Table 1.

Table 1. Statistical results of QLNSGA with variant algorithms IGD and HV on mean values of metrics.

Algorithm	NQGA					NLGA				
Parameters	<i>pop</i>	<i>pc</i>	<i>pm</i>	α	γ	<i>pop</i>	<i>pc</i>	<i>pm</i>	α	γ
Optimal level	1	2	1	4	2	1	2	1	3	2
Optimum value	60	0.8	0.1	0.9	0.7	60	0.8	0.1	0.8	0.7

For the NQGA and NLGA algorithms with calibrated parameters, experiments were conducted on 405 instances across 81 scales under the same termination criterion $T_{max} = n \times m \times F \times w \times 0.01$ as the QLNSGA algorithm. The mean values of computational results were evaluated using the performance metrics IGD, HV, and C, and mark the highest quality in bold.

The comparison among QLNSGA, NQGA, and NLGA validates the effectiveness of the Q-learning mechanism and the local search strategy, demonstrating the comprehensive superiority of the proposed complete algorithm. As shown in Appendix Table A3, QLNSGA outperforms NQGA on both IGD and HV metrics in the majority of test instances. Specifically, QLNSGA achieves better IGD values in 84% of the cases and superior HV values in 82.7% of the cases, while also dominating in C-metric comparisons in over 80% of cases (Appendix Table A4), indicating its higher solution quality and stability.

Furthermore, NQGA surpasses NLGA in 95% of cases on IGD and in 97.5% on HV, highlighting the significant improvement in diversity and convergence brought by the local search strategy. QLNSGA consistently outperforms NLGA across all instances in IGD, HV, and C-metric evaluations (Appendix Tables A3 and A4), confirming that the integrated strategy combining Q-learning and tailored local search is more effective for solving the ISSPD problem than algorithms without Q-learning or those using only random local search selection.

Figure 13 compares the 95% confidence intervals for the mean IGD, HV, and C-metric values of the three algorithms. For IGD, QLNSGA achieves the mean closest to zero with the lowest overall confidence interval, indicating superior convergence and distribution of its solution set. For HV, QLNSGA attains the mean closest to 0.7 with the highest overall confidence interval, reflecting better diversity and convergence. For the C-metric, QLNSGA reaches the mean closest to 1, showing stronger coverage dominance over solution sets produced by the other algorithms. Although NQGA underperforms on some metrics, its confidence interval for IGD remains wider than that of QLNSGA, and the upper bounds of its HV and C-metric intervals are approximately 0.1 and 0.3 lower, respectively, revealing a clear overall performance gap.

In summary, the experimental results consistently demonstrate that QLNSGA significantly outperforms both NQGA and NLGA across all evaluation metrics, exhibiting comprehensive performance advantages.

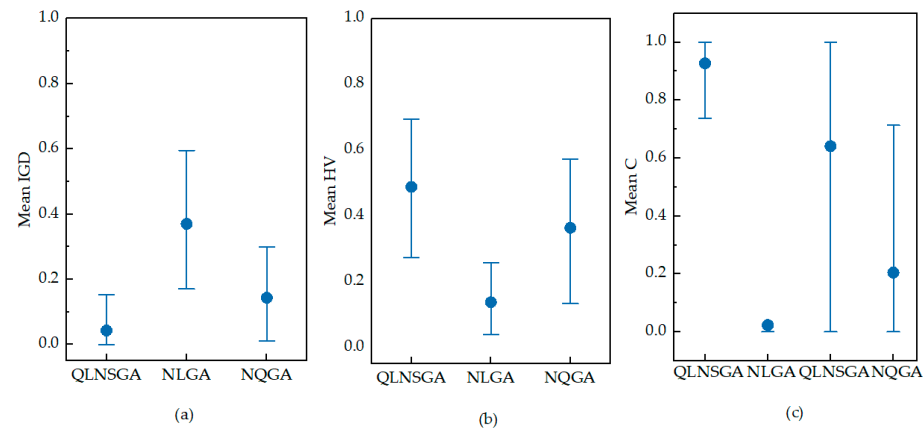


Figure 13. QLNSGA and variant algorithm 95% confidence interval plots for mean IGD, HV, and C metrics. (a) Confidence interval plot for the mean IGD. (b) Confidence interval plot for the mean HV. (c) Confidence interval plot for the mean C.

To more clearly illustrate the performance gap, Figure 14 presents a scatter plot of the mean values for the IGD and HV metrics across all test cases for the three algorithms. The figure demonstrates that QLNSGA exhibits significantly greater dispersion and convergence than NLGA. Although the difference between QLNSGA and NQGA is not pronounced in some test cases, QLNSGA overall outperforms NQGA.

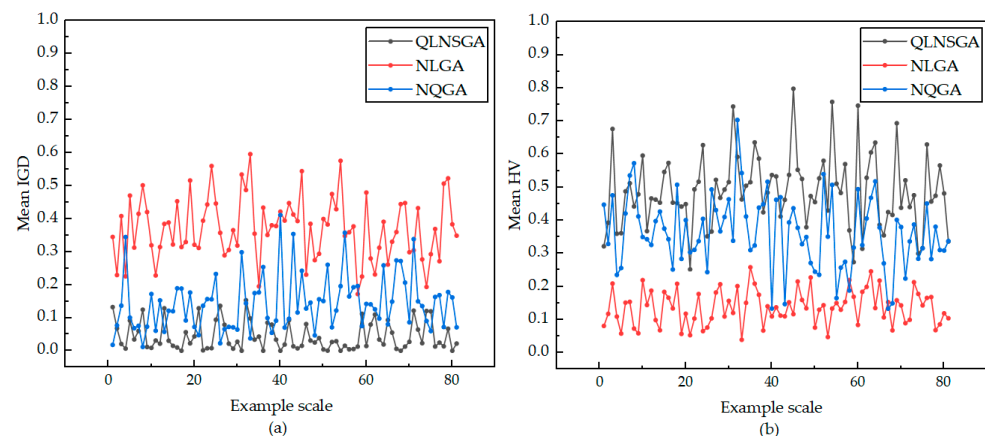


Figure 14. Scatter plots of mean IGD and HV metrics for QLNSGA and variant algorithms across different problem sizes. (a) Scatter plot of mean IGD. (b) Scatter plot of mean HV.

Based on the above results, we can conclude that QLNSGA significantly outperforms NQGA and NLGA in overall performance, demonstrating superior multi-objective optimization capabilities. Incorporating the Q-learning mechanism into the local search strategy designed based on problem characteristics effectively enhances the performance of the QLNSGA algorithm.

Furthermore, regarding the QLNSGA itself, by analyzing and summarizing its runtime across test instances of varying scales, it can be observed from Figure 15 that the runtime increases approximately linearly with the growth of problem size. Even for the largest-scale test instances, the average runtime on a standard PC remains under 800 s, which is acceptable for practical applications.

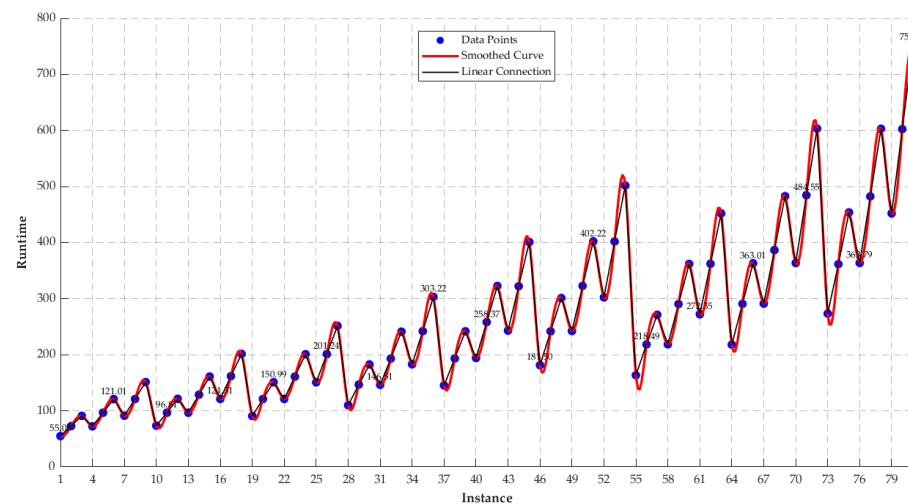


Figure 15. The running time of QLNLSGA under different scale examples.

5.4. Comparison with Other Algorithms

To further evaluate the superior performance of the QLNLSGA algorithm, this study selected four representative multi-objective optimization algorithms for comparative analysis: QL-MoPSO [40], MOEA/D [41], NNIA [42], and RIPG [43]. QL-MoPSO: This algorithm also integrates Q-learning with a metaheuristic (MOPSO), making it conceptually the closest to QLNLSGA and serving as a direct competitor to validate the advantages of our Q-learning module and overall design. MOEA/D: As a classic decomposition-based algorithm, it employs aggregation functions to decompose a multi-objective problem into multiple single-objective subproblems for collaborative optimization, aiming to compare the convergence efficiency under different optimization paradigms. NNIA: This algorithm utilizes a non-dominated neighbor-based selection mechanism to maintain population diversity, serving as a benchmark for evaluating the distribution and uniformity of the solution sets obtained by QLNLSGA. RIPG: Recognized for its robustness in handling complex problems, it is used to test the stability and effectiveness of QLNLSGA in addressing the specific complex scheduling problem studied in this research.

To ensure the fairness and validity of the comparative experiments, the Taguchi experimental design method from Section 5.1 was similarly employed to calibrate parameters for all comparison algorithms, as shown in Table 2. For QL-MoPSO: $pop = 60$, $pm = 0.1$, $\alpha = 0.7$, $\gamma = 0.7$; for MOEA/D: $pop = 100$, $pm = 0.4$, $TT = 6$; for NNIA, $pc = 0.8$, $pm = 0.2$, $Na = 50$, $Nc = 75$; for RIPG, $d = 1$, $Max = 0$. Similarly, all comparison algorithms were terminated at the same experimental runtime. The IGD, HV, and C evaluation metrics for the computational results are presented in Appendix Tables A5–A7.

Table 2. Calibration of comparison algorithm parameters.

Algorithm	QL-MoPSO				MOEA/D				NNIA			RIPG	
Parameters	pop	pm	α	γ	pop	pm	TT	pc	pm	Na	Nc	d	Max
Optimal level	1	1	2	2	3	4	2	2	2	4	3	1	1
Optimum value	60	0.1	0.7	0.7	100	0.4	6	0.8	0.2	50	75	1	0

As shown in Appendix Table A5, QLNLSGA outperforms all other comparison algorithms in the IGD metric, with the exception of RIPG, which it surpasses in 79% of the test cases. This indicates that the solution sets generated by QLNLSGA exhibit superior convergence and distribution compared to the other algorithms. The results for the HV metric in Appendix Table A6 show that QLNLSGA significantly outperforms the other algorithms across all problem scales. Its mean HV value exceeds 0.6 in nearly every instance, and

even its worst-case HV value is substantially higher than the best-case HV values of the other four algorithms. This demonstrates the superior comprehensive performance of the QLNSGA solution sets. Regarding the C-metric in Appendix Table A7, the results indicate that QLNSGA's solution sets maintain an absolute advantage over those of the other algorithms, exhibiting excellent coverage. In summary, QLNSGA outperforms the other algorithms in terms of diversity, convergence, and overall performance.

Similarly, to present the experimental results more comprehensively as in Section 5.3, Figure 16 displays the 95% confidence intervals for the mean IGD, HV, and C-metric values of all compared algorithms. The figure shows that QLNSGA significantly outperforms the other algorithms across all three evaluation metrics. In terms of the confidence interval width, QLNSGA demonstrates strong stability and a superior distribution of its mean values. Although the confidence intervals for the IGD metric are similar in width across the algorithms, the mean IGD value for QLNSGA is approximately 0.2, whereas the mean values of the other algorithms are significantly higher. This indicates QLNSGA's superior convergence and distribution. Similarly, for the HV metric, although QLNSGA's confidence interval is not the narrowest, its values are consistently higher than those of the other algorithms. Under comparable conditions, the maximum HV value achieved by any other algorithm is only around 0.05. Therefore, it can be concluded that the QLNSGA algorithm produces higher-quality solution sets.

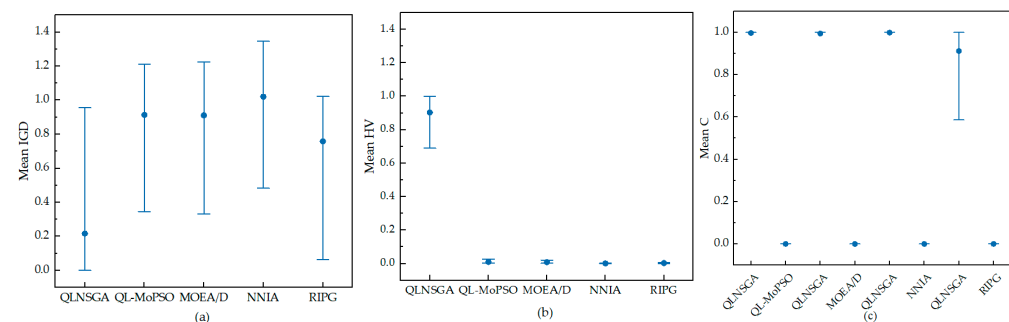


Figure 16. Plot of 95% confidence intervals for the mean values of all algorithms across IGD, HV, and C metrics. (a) Confidence interval plot for the mean IGD. (b) Confidence interval plot for the mean HV. (c) Confidence interval plot for the mean C.

Figure 17 presents scatter plots of the mean IGD and HV values for all algorithms across all problem scales. The results indicate that regarding the mean IGD, QLNSGA is only outperformed by QL-MoPSO, MOEA/D, and RIPG in a few isolated cases, demonstrating its overall stronger convergence compared to the other algorithms. As for the mean HV, although QLNSGA exhibits slightly greater variability in its values, its mean HV remains significantly higher than that of all other algorithms across all problem scales.

The higher variability of QLNSGA in the Hypervolume (HV) metric stems from two main factors. First, its dynamic balance between diversity and convergence adapts differently across problem instances of varying scales, leading to dispersed results. Second, while the Q-learning-based strategy selection enhances exploration in complex problems, the inherent randomness in this process can cause significant divergence in search paths across runs, occasionally trapping the search in local optima. This increases outcome variability, reflected in the HV metric. Despite this variability in individual runs, QLNSGA achieves stable and superior average performance over multiple executions and delivers a better Pareto front distribution.

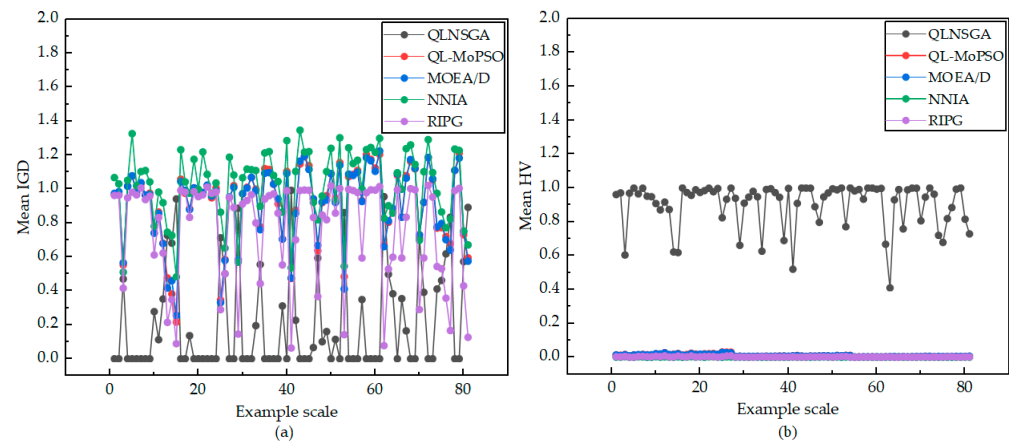


Figure 17. Scatter plots of mean IGD and HV metrics for comparison algorithms across different problem sizes. (a) Scatter plot of mean IGD. (b) Scatter plot of mean HV.

To demonstrate the performance differences between the QLNSGA algorithm and other algorithms in solving the ISSPD problem more clearly, Figure 17 presents the Pareto front of all algorithms for a randomly selected test case ($50 \times 4 \times 10 \times 10$). This allows for a rapid assessment of the quality and diversity of the solution sets obtained by different algorithms. In Figure 18, it can be observed that the solution set obtained by the QLNSGA algorithm is located closer to the coordinate axes (since both objective functions in this study are to be maximized, the inverse of the objective values is used when plotting the Pareto front). This indicates that the solutions found by the QLNSGA algorithm significantly dominate those of the other algorithms, demonstrating its superior search capability. Therefore, it can be concluded that the QLNSGA algorithm exhibits superior convergence and distribution characteristics when solving the ISSPD problem.

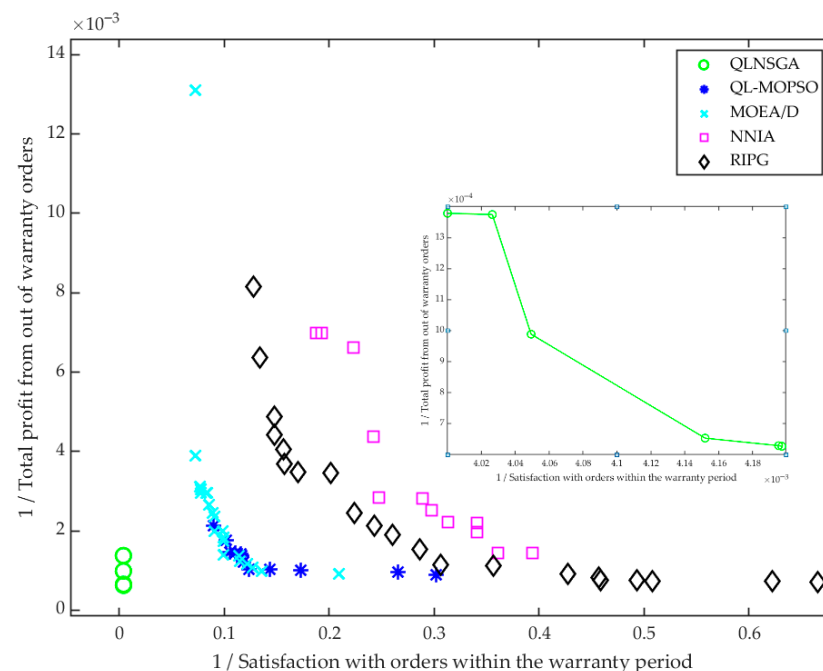


Figure 18. Pareto front diagram for all algorithms.

6. Conclusions

This study investigates the integrated resource scheduling problem for manufacturers considering differentiated in-warranty and out-of-warranty demands across both

production and after-sales maintenance. The problem encompasses scheduling spare parts production in a distributed permutation flowshop, allocating these parts to multiple field demand points, and assigning maintenance personnel for after-sales repairs. By optimizing collaborative decisions across different service models, the proposed method not only achieves operational improvements in resource utilization efficiency, waste reduction, and customer satisfaction enhancement, directly meeting corporate goals of cost reduction and efficiency improvement, but also creates multidimensional value from a system-wide perspective of sustainable manufacturing.

To effectively address this challenge, a multi-objective mathematical model is established with the goals of maximizing in-warranty customer satisfaction and out-of-warranty repair revenue. A Q-learning-based genetic algorithm (QLNSGA) is proposed for this problem. Within QLNSGA, a four-layer integer encoding scheme is designed to represent complete solutions, complemented by a knowledge-driven decoding method to enhance solution quality. Two cross-over operators and two mutation operators are further designed to improve population diversity. Additionally, a Q-learning-based problem-specific local search operator is incorporated to boost search efficiency and accelerate convergence. This local search operator comprises eight problem-characteristic-driven methods. Extensive experiments with diverse-scale test cases validate the effectiveness of QLNSGA in solving the ISSPD problem. Results demonstrate its comprehensive superiority: QLNSGA outperforms its variants (NQGA and NLGA) in 84% of test cases and surpasses RIPG in 79% of cases. Regarding the Hypervolume (HV) metric, it achieves a mean value exceeding 0.6, with its worst-case performance still higher than the best-case results of other compared algorithms. Furthermore, its solution set exhibits absolute dominance in the coverage (C-metric) comparisons. Overall, QLNSGA demonstrates significant and consistent superiority over other state-of-the-art algorithms, presenting a distinct advantage in the quality and distribution of its Pareto front solutions.

Examined from a broader sustainable manufacturing perspective, the outcomes of this study generate positive impacts across three key dimensions. Economically, the collaborative optimization enhances resource utilization and operational efficiency, directly strengthening corporate profitability resilience and cost-control capabilities. Environmentally, by reducing equipment idle waiting time, optimizing logistics, and improving the first-time fix rate, it effectively lowers implicit energy consumption and material waste, contributing to carbon emission reduction and green manufacturing. Socially, prioritizing in-warranty customer satisfaction as a core optimization objective emphasizes the fulfillment of service commitments, helping to build trust, promote consumer fairness, and extend product lifespan, reflecting active social responsibility. These connections indicate that the proposed model and algorithm not only provide an efficient operational solution but also offer quantifiable decision support for advancing sustainable development in manufacturing.

However, this study has certain limitations. For instance, we only considered the coordinated scheduling of two resources—spare parts production and service workers—under static workshop conditions, with precisely known demand times and a simplified assumption of homogeneous workers. For future work, we will explore incorporating additional real-world factors to enhance the model's practical value. This includes extending the research to dynamic workshop scenarios with uncertain disruptions and transportation planning. On the algorithmic front, we will investigate the application of deep reinforcement learning methods, such as Deep Q-Networks, to improve scheduling decisions. Furthermore, a key direction will be to relax the homogeneous worker assumption by modeling workforce heterogeneity—including differentiated skills, learning effects, and certification levels—to examine their impact on system performance and to uncover new managerial trade-offs in multi-resource collaborative scheduling.

Author Contributions: Conceptualization, Y.M., Q.D., and J.Z. (Juan Zhou); methodology, Y.M.; software, Y.M.; formal analysis, Y.M., J.Z. (Juan Zhou), and J.Z. (Jingxing Zhang); validation, J.Z. (Juan Zhou); writing—original draft preparation, Y.M.; supervision, Q.D.; funding acquisition, Q.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 52475517.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare that no conflicts of interest exists in the submission of this manuscript, and this manuscript is approved by all authors for publication.

Appendix A

Table A1. Construction of instances.

Parameters	Range	Generation Method
N	{50, 100, 150}	Randomly select
F	{3, 4, 5}	Randomly select
m	{6, 8, 10}	Randomly select
w	{6, 8, 10}	Randomly select
T_i	{5, 10}	Randomly select
P_{it}	[10, 40]	Randomly generated (numbers with two decimal places and evenly distributed)
PT_t	[20, 60]	Randomly generated (numbers with two decimal places and evenly distributed)
TF_{ki}	[60, 180]	Randomly generated (numbers with two decimal places and evenly distributed)
TB_{ji}	[10, 30]	Randomly generated (numbers with two decimal places and evenly distributed)
TBB_{ic}	[20, 60]	Randomly generated (numbers with two decimal places and evenly distributed)
$price_i$	[50, 80]	Randomly generated (numbers with two decimal places and evenly distributed)
D_i^-	-	Refer to formulas (29) and (30)
D_i^+	-	Refer to formulas (30)
WT_i	-	Refer to formulas (31)
$Price_i$	-	Refer to formulas (32)
DC	3	Constant
B	1000	Constant

Table A2. Calibration of QLNSGA algorithm parameters.

Combo	IGD													
	pop	pc	pm	α	γ	1	2	3	4	5	6	7	8	9
1	60	0.7	0.1	0.6	0.6	0.2806	0.5714	0.1763	0.1574	0.3194	0.1381	0.1447	0.3369	0.2328
2	60	0.8	0.2	0.7	0.7	0.3635	0.5790	0.1898	0.2113	0.2229	0.2150	0.2018	0.3149	0.2071
3	60	0.9	0.3	0.8	0.8	0.3072	0.5402	0.1130	0.2253	0.1541	0.0170	0.1406	0.1453	0.2598
4	60	1.0	0.4	0.9	0.9	0.2477	0.5770	0.1777	0.2251	0.2405	0.2057	0.1816	0.3277	0.2673
5	80	0.7	0.2	0.8	0.9	0.3976	0.5686	0.2339	0.2874	0.2512	0.1937	0.2341	0.3116	0.2201
6	80	0.8	0.1	0.9	0.8	0.2745	0.5767	0.1834	0.1747	0.2766	0.2363	0.0995	0.3407	0.0344
7	80	0.9	0.4	0.6	0.7	0.3117	0.6055	0.1490	0.1706	0.2684	0.1669	0.1514	0.3270	0.2297
8	80	1.0	0.3	0.7	0.6	0.3738	0.5765	0.1315	0.2491	0.2434	0.1909	0.1389	0.0494	0.3813
9	100	0.7	0.3	0.9	0.7	0.2946	0.2747	0.2409	0.2370	0.3027	0.1208	0.1109	0.1377	0.1867
10	100	0.8	0.4	0.8	0.6	0.3188	0.5286	0.1870	0.1849	0.2627	0.2333	0.2064	0.2804	0.0899
11	100	0.9	0.1	0.7	0.9	0.3224	0.5728	0.2393	0.1454	0.3121	0.2099	0.1371	0.2600	0.2578

12	100	1.0	0.2	0.6	0.8	0.2605	0.5931	0.0699	0.2080	0.3069	0.1959	0.1287	0.3583	0.2652
13	120	0.7	0.4	0.7	0.8	0.3888	0.5379	0.1851	0.1827	0.1459	0.0660	0.2177	0.3489	0.1457
14	120	0.8	0.3	0.6	0.9	0.2065	0.5600	0.1760	0.0951	0.1927	0.1590	0.1872	0.3746	0.2882
15	120	0.9	0.2	0.9	0.6	0.3399	0.5585	0.1294	0.2670	0.2452	0.1649	0.2218	0.4259	0.1571
16	120	1.0	0.1	0.8	0.7	0.3251	0.2914	0.2749	0.0516	0.1866	0.0985	0.1996	0.2927	0.3841

Table A3. Statistical results of QLNSGA and variant algorithms on the mean values of the C metrics.

Ins.	IGD			HV		
	QLNSGA	NQGA	NLGA	QLNSGA	NQGA	NLGA
50 × 3 × 6 × 6	0.1321	0.0177	0.3450	0.3214	0.4472	0.0809
50 × 3 × 8 × 6	0.0678	0.0767	0.2292	0.3928	0.3284	0.1174
50 × 3 × 10 × 6	0.0203	0.1367	0.4081	0.6762	0.4753	0.2087
50 × 3 × 6 × 8	0.0066	0.3447	0.2254	0.3595	0.2348	0.1089
50 × 3 × 8 × 8	0.0915	0.1003	0.4702	0.3610	0.2557	0.0575
50 × 3 × 10 × 8	0.0342	0.0682	0.3125	0.4874	0.4207	0.1512
50 × 3 × 6 × 10	0.0597	0.0753	0.4149	0.5119	0.5354	0.1534
50 × 3 × 8 × 10	0.1245	0.0114	0.5016	0.4418	0.5725	0.0727
50 × 3 × 10 × 10	0.0112	0.0729	0.4202	0.4786	0.4117	0.0585
50 × 4 × 6 × 6	0.0085	0.1720	0.3196	0.5956	0.3494	0.2192
50 × 4 × 8 × 6	0.0309	0.0605	0.2281	0.3670	0.3429	0.1442
50 × 4 × 10 × 6	0.0214	0.1528	0.3143	0.4662	0.3255	0.1874
50 × 4 × 6 × 8	0.1286	0.0571	0.3844	0.4625	0.3974	0.0986
50 × 4 × 8 × 8	0.0303	0.1210	0.3886	0.4533	0.4263	0.0678
50 × 4 × 10 × 8	0.0147	0.1196	0.3220	0.5460	0.3750	0.1836
50 × 4 × 6 × 10	0.0098	0.1890	0.4529	0.5733	0.3429	0.1659
50 × 4 × 8 × 10	0.0000	0.1882	0.3143	0.4532	0.2513	0.1337
50 × 4 × 10 × 10	0.0560	0.0917	0.3295	0.4521	0.5073	0.2083
50 × 5 × 6 × 6	0.0221	0.1765	0.5159	0.4415	0.2835	0.0566
50 × 5 × 8 × 6	0.0433	0.0722	0.3210	0.4490	0.4005	0.1177
50 × 5 × 10 × 6	0.1284	0.0471	0.3116	0.2515	0.3034	0.0528
50 × 5 × 6 × 8	0.0011	0.1362	0.3942	0.4935	0.3108	0.1031
50 × 5 × 8 × 8	0.0073	0.1565	0.4432	0.5165	0.3368	0.1773
50 × 5 × 10 × 8	0.0077	0.1560	0.5600	0.6272	0.4044	0.0650
50 × 5 × 6 × 10	0.0939	0.2326	0.4462	0.3513	0.2433	0.0760
50 × 5 × 8 × 10	0.1359	0.0225	0.3574	0.3659	0.4931	0.1036
50 × 5 × 10 × 10	0.0788	0.0654	0.2889	0.5222	0.4309	0.1817
100 × 3 × 6 × 6	0.0212	0.0722	0.3052	0.4678	0.3665	0.2067
100 × 3 × 8 × 6	0.0062	0.0711	0.3654	0.4929	0.4098	0.1092
100 × 3 × 10 × 6	0.0273	0.0640	0.3186	0.5154	0.4634	0.1565
100 × 3 × 6 × 8	0.0000	0.2986	0.5342	0.7439	0.3380	0.1199
100 × 3 × 8 × 8	0.1528	0.1437	0.4872	0.5914	0.7030	0.2012
100 × 3 × 10 × 8	0.0981	0.0373	0.5959	0.4632	0.5428	0.0387
100 × 3 × 6 × 10	0.0334	0.1745	0.3548	0.5045	0.4113	0.1506
100 × 3 × 8 × 10	0.0426	0.1767	0.1950	0.5150	0.3098	0.2580
100 × 3 × 10 × 10	0.0000	0.2537	0.4338	0.6355	0.3238	0.2085
100 × 4 × 6 × 6	0.0849	0.0991	0.3509	0.5864	0.4383	0.1748
100 × 4 × 8 × 6	0.0792	0.0542	0.3801	0.4240	0.4487	0.0666
100 × 4 × 10 × 6	0.0334	0.0912	0.3778	0.4836	0.5165	0.1402
100 × 4 × 6 × 8	0.0000	0.4113	0.4216	0.5368	0.1334	0.1096
100 × 4 × 8 × 8	0.0188	0.0701	0.3945	0.5327	0.4620	0.1368
100 × 4 × 10 × 8	0.0925	0.0967	0.4474	0.4114	0.4695	0.1121
100 × 4 × 6 × 10	0.0132	0.3536	0.4129	0.4619	0.1466	0.1099

100 × 4 × 8 × 10	0.0071	0.1157	0.3931	0.5372	0.3937	0.1522
100 × 4 × 10 × 10	0.0149	0.2424	0.5443	0.7977	0.4364	0.1163
100 × 5 × 6 × 6	0.0808	0.1281	0.2304	0.5524	0.3774	0.2149
100 × 5 × 8 × 6	0.0308	0.1457	0.3846	0.5250	0.3278	0.1593
100 × 5 × 10 × 6	0.0249	0.0466	0.2745	0.3790	0.3494	0.1338
100 × 5 × 6 × 8	0.0382	0.1559	0.2937	0.4732	0.2709	0.2272
100 × 5 × 8 × 8	0.0037	0.1505	0.3991	0.4552	0.2445	0.0756
100 × 5 × 10 × 8	0.0004	0.2605	0.3826	0.5268	0.2351	0.1297
100 × 5 × 6 × 10	0.0264	0.0708	0.4749	0.5799	0.5395	0.1433
100 × 5 × 8 × 10	0.0286	0.1217	0.4291	0.4295	0.3505	0.0472
100 × 5 × 10 × 10	0.0000	0.1959	0.5757	0.7578	0.5070	0.1332
150 × 3 × 6 × 6	0.0151	0.3575	0.3479	0.5105	0.1648	0.1498
150 × 3 × 8 × 6	0.0041	0.1644	0.3592	0.4827	0.2569	0.1298
150 × 3 × 10 × 6	0.0050	0.1924	0.3768	0.5699	0.2744	0.1532
150 × 3 × 6 × 8	0.0124	0.1960	0.1715	0.3696	0.1876	0.2194
150 × 3 × 8 × 8	0.1116	0.0748	0.2248	0.2737	0.3177	0.1690
150 × 3 × 10 × 8	0.0142	0.1419	0.4795	0.7464	0.4941	0.0838
150 × 3 × 6 × 10	0.0791	0.1403	0.2796	0.3139	0.3249	0.1839
150 × 3 × 8 × 10	0.1094	0.1257	0.2311	0.5284	0.4056	0.1987
150 × 3 × 10 × 10	0.0337	0.0973	0.3122	0.6051	0.4678	0.2456
150 × 4 × 6 × 6	0.0189	0.2588	0.3910	0.6348	0.5175	0.1349
150 × 4 × 8 × 6	0.0929	0.0800	0.2610	0.3860	0.3780	0.2176
150 × 4 × 10 × 6	0.0550	0.1486	0.3304	0.3543	0.2697	0.1067
150 × 4 × 6 × 8	0.0054	0.2738	0.3598	0.4249	0.1328	0.1527
150 × 4 × 8 × 8	0.0000	0.2719	0.4444	0.4162	0.1489	0.0673
150 × 4 × 10 × 8	0.0123	0.2063	0.4477	0.6935	0.4015	0.1583
150 × 4 × 6 × 10	0.0270	0.0860	0.2986	0.4379	0.3795	0.1430
150 × 4 × 8 × 10	0.1216	0.3382	0.3045	0.5207	0.2240	0.0891
150 × 4 × 10 × 10	0.0641	0.1496	0.4323	0.4400	0.3362	0.0992
150 × 5 × 6 × 6	0.0229	0.1355	0.2768	0.4757	0.3871	0.2130
150 × 5 × 8 × 6	0.1206	0.0902	0.1934	0.2994	0.2835	0.1776
150 × 5 × 10 × 6	0.1195	0.0596	0.2926	0.3156	0.3155	0.1428
150 × 5 × 6 × 8	0.0126	0.1631	0.3691	0.6292	0.4506	0.1654
150 × 5 × 8 × 8	0.0244	0.1683	0.2712	0.4565	0.2828	0.1682
150 × 5 × 10 × 8	0.0134	0.0720	0.5066	0.4740	0.3805	0.0681
150 × 5 × 6 × 10	0.0666	0.1779	0.5224	0.5657	0.3101	0.0854
150 × 5 × 8 × 10	0.0000	0.1616	0.3833	0.4811	0.3087	0.1187
150 × 5 × 10 × 10	0.0216	0.0708	0.3487	0.3359	0.3371	0.1039

Table A4. Statistical results on the mean of the C metrics for QLNSGA and variant algorithms.

Ins.	C (QLNSGA,NQGA)	C (NQGA,QLNSGA)	C (QLNSGA,NLGA)	C (NLGA,QLNSGA)
50 × 3 × 6 × 6	0.0000	0.8889	0.9375	0.1111
50 × 3 × 8 × 6	0.6000	0.3750	0.8571	0.0000
50 × 3 × 10 × 6	0.8125	0.2000	1.0000	0.0000
50 × 3 × 6 × 8	1.0000	0.0000	0.8750	0.0000
50 × 3 × 8 × 8	0.3333	0.1429	0.8333	0.0000
50 × 3 × 10 × 8	0.7273	0.1000	0.9524	0.0000
50 × 3 × 6 × 10	0.3333	0.6667	1.0000	0.0000
50 × 3 × 8 × 10	0.0625	0.7143	0.8333	0.0000
50 × 3 × 10 × 10	0.4000	0.3125	0.9091	0.0000
50 × 4 × 6 × 6	0.6667	0.0625	0.9000	0.0000
50 × 4 × 8 × 6	0.5833	0.3529	0.8182	0.0588
50 × 4 × 10 × 6	0.9091	0.0909	0.8182	0.0000

50 × 4 × 6 × 8	0.4000	0.0000	1.0000	0.0000
50 × 4 × 8 × 8	0.7143	0.3000	0.9091	0.0000
50 × 4 × 10 × 8	0.8000	0.0909	1.0000	0.0000
50 × 4 × 6 × 10	0.8750	0.0000	0.8750	0.0000
50 × 4 × 8 × 10	1.0000	0.0000	1.0000	0.0000
50 × 4 × 10 × 10	0.2000	0.3077	0.9231	0.0000
50 × 5 × 6 × 6	0.6667	0.0909	1.0000	0.0000
50 × 5 × 8 × 6	0.6000	0.1250	0.9474	0.0000
50 × 5 × 10 × 6	0.1818	0.5455	0.8125	0.0000
50 × 5 × 6 × 8	1.0000	0.0000	0.9375	0.0000
50 × 5 × 8 × 8	0.8000	0.0833	0.9286	0.0000
50 × 5 × 10 × 8	0.9091	0.0000	1.0000	0.0000
50 × 5 × 6 × 10	0.8000	0.0000	0.8333	0.0000
50 × 5 × 8 × 10	0.0000	0.6667	0.8889	0.0000
50 × 5 × 10 × 10	0.5556	0.1000	0.8889	0.0000
100 × 3 × 6 × 6	0.5556	0.1538	0.8750	0.0000
100 × 3 × 8 × 6	0.6250	0.0625	0.9545	0.0000
100 × 3 × 10 × 6	0.5455	0.4118	1.0000	0.0000
100 × 3 × 6 × 8	1.0000	0.0000	1.0000	0.0000
100 × 3 × 8 × 8	0.0000	0.6364	1.0000	0.0000
100 × 3 × 10 × 8	0.4444	0.5714	1.0000	0.0000
100 × 3 × 6 × 10	0.6000	0.5000	1.0000	0.0000
100 × 3 × 8 × 10	0.8333	0.1250	0.8182	0.1250
100 × 3 × 10 × 10	1.0000	0.0000	1.0000	0.0000
100 × 4 × 6 × 6	0.5000	0.0000	1.0000	0.0000
100 × 4 × 8 × 6	0.4000	0.6250	0.8750	0.1250
100 × 4 × 10 × 6	0.2000	0.5556	1.0000	0.0000
100 × 4 × 6 × 8	1.0000	0.0000	1.0000	0.0000
100 × 4 × 8 × 8	0.5000	0.2667	1.0000	0.0000
100 × 4 × 10 × 8	0.0000	0.6000	0.9333	0.2000
100 × 4 × 6 × 10	1.0000	0.0000	0.9286	0.1111
100 × 4 × 8 × 10	0.8000	0.1765	1.0000	0.0000
100 × 4 × 10 × 10	0.8750	0.0000	1.0000	0.0000
100 × 5 × 6 × 6	0.7778	0.2000	0.8667	0.2000
100 × 5 × 8 × 6	0.7778	0.0714	0.9375	0.0000
100 × 5 × 10 × 6	0.5000	0.4706	0.9333	0.1765
100 × 5 × 6 × 8	0.6250	0.0000	0.7778	0.2000
100 × 5 × 8 × 8	0.5000	0.1875	0.9333	0.0000
100 × 5 × 10 × 8	0.8333	0.0588	1.0000	0.0000
100 × 5 × 6 × 10	0.5556	0.3846	1.0000	0.0000
100 × 5 × 8 × 10	0.8333	0.1429	1.0000	0.0000
100 × 5 × 10 × 10	1.0000	0.0000	1.0000	0.0000
150 × 3 × 6 × 6	1.0000	0.0000	0.8667	0.1000
150 × 3 × 8 × 6	0.9286	0.0000	1.0000	0.0000
150 × 3 × 10 × 6	0.8750	0.0000	1.0000	0.0000
150 × 3 × 6 × 8	0.8750	0.0000	0.8889	0.0556
150 × 3 × 8 × 8	0.4444	0.5000	0.8235	0.2500
150 × 3 × 10 × 8	0.7500	0.2000	1.0000	0.0000
150 × 3 × 6 × 10	0.2000	0.5455	0.8889	0.0000
150 × 3 × 8 × 10	0.5000	0.2500	0.7368	0.0000
150 × 3 × 10 × 10	0.6250	0.0000	0.8125	0.0000
150 × 4 × 6 × 6	0.5000	0.2222	1.0000	0.0000
150 × 4 × 8 × 6	0.6364	0.1667	0.8000	0.0000
150 × 4 × 10 × 6	0.6667	0.1250	0.8667	0.1250
150 × 4 × 6 × 8	1.0000	0.0000	0.8750	0.0000
150 × 4 × 8 × 8	1.0000	0.0000	1.0000	0.0000
150 × 4 × 10 × 8	1.0000	0.0000	0.9000	0.0000
150 × 4 × 6 × 10	0.3333	0.5909	1.0000	0.0000
150 × 4 × 8 × 10	1.0000	0.0000	0.8182	0.0000

150 × 4 × 10 × 10	0.5714	0.0000	1.0000	0.0000
150 × 5 × 6 × 6	0.7500	0.1429	0.9000	0.0000
150 × 5 × 8 × 6	0.4545	0.2857	0.8571	0.0000
150 × 5 × 10 × 6	0.2500	0.5000	0.8889	0.0000
150 × 5 × 6 × 8	0.8333	0.0833	0.9375	0.0000
150 × 5 × 8 × 8	0.8889	0.0909	0.8571	0.0000
150 × 5 × 10 × 8	0.5714	0.2000	1.0000	0.0000
150 × 5 × 6 × 10	0.6667	0.0000	1.0000	0.0000
150 × 5 × 8 × 10	1.0000	0.0000	1.0000	0.0000
150 × 5 × 10 × 10	0.4000	0.2667	0.8750	0.0667

Table A5. Statistical results on the mean of the IGD metrics of the comparison algorithm.

Ins.	IGD				
	QLNSGA	QL-MoPSO	MOEA/D	NNIA	RIPG
50 × 3 × 6 × 6	0.0000	0.9687	0.9749	1.0676	0.9628
50 × 3 × 8 × 6	0.0000	0.9666	0.9845	1.0308	0.9634
50 × 3 × 10 × 6	0.4706	0.5529	0.5663	0.5111	0.4159
50 × 3 × 6 × 8	0.0000	1.0182	1.0165	1.0524	0.9490
50 × 3 × 8 × 8	0.0000	1.0763	1.0783	1.3266	0.9814
50 × 3 × 10 × 8	0.0000	0.9881	0.9784	1.0196	0.9656
50 × 3 × 6 × 10	0.0000	1.0307	1.0378	1.1032	1.0071
50 × 3 × 8 × 10	0.0000	0.9638	0.9651	1.1094	0.9380
50 × 3 × 10 × 10	0.0000	0.9760	0.9692	1.0426	0.9586
50 × 4 × 6 × 6	0.2777	0.7429	0.7402	0.7821	0.6118
50 × 4 × 8 × 6	0.1126	0.8651	0.8564	0.9824	0.8324
50 × 4 × 10 × 6	0.3525	0.6814	0.6790	0.9201	0.6231
50 × 4 × 6 × 8	0.7267	0.4768	0.4170	0.7457	0.2144
50 × 4 × 8 × 8	0.6812	0.3817	0.4604	0.7268	0.3485
50 × 4 × 10 × 8	0.9417	0.2159	0.2554	0.4826	0.0891
50 × 4 × 6 × 10	0.0000	1.0587	1.0448	1.2319	0.9920
50 × 4 × 8 × 10	0.0000	0.9748	0.9923	1.0424	0.9817
50 × 4 × 10 × 10	0.1362	0.8847	0.8796	0.9762	0.8334
50 × 5 × 6 × 6	0.0000	1.0071	1.0074	1.1754	0.9852
50 × 5 × 8 × 6	0.0000	0.9676	0.9582	0.9986	0.9561
50 × 5 × 10 × 6	0.0000	0.9797	0.9797	1.2188	0.9665
50 × 5 × 6 × 8	0.0000	1.0150	1.0232	1.0870	1.0116
50 × 5 × 8 × 8	0.0000	0.9484	0.9586	0.9692	0.9742
50 × 5 × 10 × 8	0.0000	1.0073	0.9903	1.0359	0.9830
50 × 5 × 6 × 10	0.7132	0.3465	0.3315	0.8641	0.2896
50 × 5 × 8 × 10	0.5011	0.5853	0.5800	0.6523	0.5011
50 × 5 × 10 × 10	0.0000	0.9492	0.9537	1.1879	0.9510
100 × 3 × 6 × 6	0.0000	1.0207	1.0106	1.0829	0.8910
100 × 3 × 8 × 6	0.8868	0.5845	0.5850	0.5695	0.1453
100 × 3 × 10 × 6	0.0000	0.9719	0.9733	1.0668	0.9122
100 × 3 × 6 × 8	0.0000	1.0106	1.0061	1.1182	0.9327
100 × 3 × 8 × 8	0.0000	1.0679	1.0688	1.1140	0.9668
100 × 3 × 10 × 8	0.1955	0.9995	0.9914	1.1095	0.8008
100 × 3 × 6 × 10	0.5557	0.7781	0.7613	0.8994	0.4434
100 × 3 × 8 × 10	0.0000	1.1211	1.0931	1.2132	0.9415
100 × 3 × 10 × 10	0.0000	1.1158	1.0975	1.2208	0.9612
100 × 4 × 6 × 6	0.0000	1.0293	1.0293	1.0804	0.9720
100 × 4 × 8 × 6	0.0000	0.9135	0.9432	1.0446	0.8583
100 × 4 × 10 × 6	0.3112	0.7059	0.7048	0.8642	0.5527

100 × 4 × 6 × 8	0.0000	1.1044	1.0922	1.2855	0.9905
100 × 4 × 8 × 8	0.9928	0.4732	0.4761	0.5368	0.0633
100 × 4 × 10 × 8	0.2271	0.8753	0.8575	1.1043	0.7009
100 × 4 × 6 × 10	0.0000	1.1474	1.1651	1.3471	0.9916
100 × 4 × 8 × 10	0.0000	1.1906	1.1939	1.2176	0.9944
100 × 4 × 10 × 10	0.0000	1.1383	1.1158	1.2212	0.9927
100 × 5 × 6 × 6	0.0663	0.9419	0.9423	0.9212	0.8329
100 × 5 × 8 × 6	0.5942	0.6372	0.6672	0.8192	0.3671
100 × 5 × 10 × 6	0.1008	0.9199	0.9244	0.9595	0.8472
100 × 5 × 6 × 8	0.1608	0.9618	0.9355	1.1029	0.8195
100 × 5 × 8 × 8	0.0000	1.0911	1.0911	1.2399	1.0188
100 × 5 × 10 × 8	0.1142	0.9472	0.9213	0.9342	0.8579
100 × 5 × 6 × 10	0.0000	1.1555	1.1413	1.3027	1.0033
100 × 5 × 8 × 10	0.8617	0.4848	0.4115	0.5457	0.1424
100 × 5 × 10 × 10	0.0000	1.0732	1.0883	1.2436	0.9987
150 × 3 × 6 × 6	0.0000	1.0871	1.0805	1.1504	0.9919
150 × 3 × 8 × 6	0.0000	1.1101	1.1003	1.1702	0.9793
150 × 3 × 10 × 6	0.3493	0.9322	0.9269	1.0001	0.5940
150 × 3 × 6 × 8	0.0000	1.2038	1.1850	1.2325	0.9783
150 × 3 × 8 × 8	0.0000	1.1760	1.1671	1.2453	0.9955
150 × 3 × 10 × 8	0.0000	1.1239	1.1046	1.2170	0.9938
150 × 3 × 6 × 10	0.0000	1.2045	1.2244	1.2997	1.0155
150 × 3 × 8 × 10	0.9554	0.6696	0.6611	0.8290	0.0776
150 × 3 × 10 × 10	0.4983	0.8044	0.8123	0.9021	0.5290
150 × 4 × 6 × 6	0.3828	0.8675	0.8859	0.8590	0.5994
150 × 4 × 8 × 6	0.0000	1.0814	1.0899	1.0961	0.9964
150 × 4 × 10 × 6	0.3550	0.8349	0.8347	0.9999	0.5943
150 × 4 × 6 × 8	0.1645	1.0792	1.0642	1.2381	0.8344
150 × 4 × 8 × 8	0.0000	1.1624	1.1744	1.2601	1.0029
150 × 4 × 10 × 8	0.0000	1.1197	1.1222	1.1451	0.9956
150 × 4 × 6 × 10	0.7119	0.7311	0.7304	0.6984	0.2898
150 × 4 × 8 × 10	0.3915	0.9218	0.9218	1.1023	0.5938
150 × 4 × 10 × 10	0.0000	1.1857	1.1871	1.2915	1.0216
150 × 5 × 6 × 6	0.0000	1.0542	1.0606	1.0975	0.9528
150 × 5 × 8 × 6	0.4113	0.7695	0.7770	0.9747	0.5438
150 × 5 × 10 × 6	0.4615	0.7727	0.7975	0.8649	0.5309
150 × 5 × 6 × 8	0.6185	0.7209	0.7019	0.7727	0.3572
150 × 5 × 8 × 8	0.8342	0.6769	0.6414	0.8240	0.1664
150 × 5 × 10 × 8	0.0000	1.1100	1.1105	1.2359	0.9908
150 × 5 × 6 × 10	0.0000	1.2119	1.1830	1.2278	1.0044
150 × 5 × 8 × 10	0.5728	0.7308	0.7492	0.7527	0.4299
150 × 5 × 10 × 10	0.8923	0.5951	0.5754	0.6710	0.1264

Table A6. Statistical results on the mean of the HV metrics of the comparison algorithm.

Ins.	HV				
	QLNSGA	QL-MoPSO	MOEA/D	NNIA	RIPG
50 × 3 × 6 × 6	0.9613	0.0145	0.0131	0.0013	0.0040
50 × 3 × 8 × 6	0.9713	0.0119	0.0105	0.0012	0.0033
50 × 3 × 10 × 6	0.6042	0.0167	0.0149	0.0035	0.0065
50 × 3 × 6 × 8	0.9695	0.0081	0.0075	0.0004	0.0014
50 × 3 × 8 × 8	0.9992	0.0142	0.0130	0.0001	0.0014
50 × 3 × 10 × 8	0.9655	0.0151	0.0151	0.0027	0.0058

50 × 3 × 6 × 10	0.9989	0.0170	0.0157	0.0025	0.0065
50 × 3 × 8 × 10	0.9515	0.0149	0.0130	0.0002	0.0042
50 × 3 × 10 × 10	0.9487	0.0145	0.0146	0.0007	0.0034
50 × 4 × 6 × 6	0.9079	0.0208	0.0209	0.0035	0.0073
50 × 4 × 8 × 6	0.8706	0.0197	0.0191	0.0012	0.0048
50 × 4 × 10 × 6	0.9167	0.0273	0.0261	0.0027	0.0090
50 × 4 × 6 × 8	0.8729	0.0158	0.0158	0.0010	0.0029
50 × 4 × 8 × 8	0.6220	0.0186	0.0161	0.0008	0.0031
50 × 4 × 10 × 8	0.6189	0.0224	0.0206	0.0027	0.0078
50 × 4 × 6 × 10	0.9999	0.0136	0.0121	0.0007	0.0040
50 × 4 × 8 × 10	0.9735	0.0167	0.0125	0.0019	0.0034
50 × 4 × 10 × 10	0.9567	0.0242	0.0215	0.0042	0.0086
50 × 5 × 6 × 6	0.9893	0.0192	0.0159	0.0015	0.0030
50 × 5 × 8 × 6	0.9734	0.0185	0.0174	0.0020	0.0039
50 × 5 × 10 × 6	0.9847	0.0206	0.0182	0.0003	0.0051
50 × 5 × 6 × 8	0.9988	0.0211	0.0180	0.0027	0.0054
50 × 5 × 8 × 8	0.9782	0.0227	0.0190	0.0026	0.0049
50 × 5 × 10 × 8	0.9972	0.0175	0.0168	0.0011	0.0025
50 × 5 × 6 × 10	0.8242	0.0309	0.0288	0.0007	0.0042
50 × 5 × 8 × 10	0.9334	0.0305	0.0273	0.0031	0.0088
50 × 5 × 10 × 10	0.9994	0.0305	0.0262	0.0022	0.0098
100 × 3 × 6 × 6	0.9393	0.0047	0.0044	0.0006	0.0023
100 × 3 × 8 × 6	0.6616	0.0064	0.0058	0.0008	0.0026
100 × 3 × 10 × 6	0.9106	0.0048	0.0045	0.0003	0.0024
100 × 3 × 6 × 8	0.9464	0.0054	0.0049	0.0001	0.0017
100 × 3 × 8 × 8	0.9804	0.0052	0.0047	0.0008	0.0022
100 × 3 × 10 × 8	0.9479	0.0058	0.0053	0.0004	0.0025
100 × 3 × 6 × 10	0.6265	0.0049	0.0049	0.0006	0.0035
100 × 3 × 8 × 10	0.9911	0.0062	0.0058	0.0006	0.0031
100 × 3 × 10 × 10	0.9958	0.0053	0.0050	0.0002	0.0016
100 × 4 × 6 × 6	0.9727	0.0064	0.0058	0.0007	0.0030
100 × 4 × 8 × 6	0.9449	0.0072	0.0057	0.0007	0.0017
100 × 4 × 10 × 6	0.6896	0.0074	0.0069	0.0007	0.0030
100 × 4 × 6 × 8	0.9975	0.0059	0.0055	0.0001	0.0014
100 × 4 × 8 × 8	0.5207	0.0087	0.0078	0.0020	0.0042
100 × 4 × 10 × 8	0.9093	0.0107	0.0094	0.0005	0.0016
100 × 4 × 6 × 10	1.0000	0.0072	0.0061	0.0003	0.0024
100 × 4 × 8 × 10	0.9997	0.0054	0.0048	0.0008	0.0024
100 × 4 × 10 × 10	0.9990	0.0071	0.0067	0.0006	0.0034
100 × 5 × 6 × 6	0.8895	0.0071	0.0063	0.0012	0.0028
100 × 5 × 8 × 6	0.7971	0.0083	0.0072	0.0010	0.0032
100 × 5 × 10 × 6	0.9471	0.0095	0.0084	0.0020	0.0036
100 × 5 × 6 × 8	0.9707	0.0083	0.0079	0.0009	0.0030
100 × 5 × 8 × 8	0.9956	0.0065	0.0061	0.0004	0.0021
100 × 5 × 10 × 8	0.9892	0.0106	0.0104	0.0017	0.0043
100 × 5 × 6 × 10	1.0000	0.0083	0.0084	0.0008	0.0044
100 × 5 × 8 × 10	0.7714	0.0112	0.0106	0.0018	0.0049
100 × 5 × 10 × 10	0.9999	0.0112	0.0091	0.0010	0.0022
150 × 3 × 6 × 6	0.9824	0.0018	0.0017	0.0000	0.0003
150 × 3 × 8 × 6	0.9916	0.0027	0.0025	0.0002	0.0016
150 × 3 × 10 × 6	0.9340	0.0030	0.0027	0.0004	0.0023
150 × 3 × 6 × 8	0.9998	0.0030	0.0029	0.0007	0.0019
150 × 3 × 8 × 8	0.9992	0.0031	0.0030	0.0004	0.0024

150 × 3 × 10 × 8	0.9928	0.0029	0.0029	0.0002	0.0018
150 × 3 × 6 × 10	0.9977	0.0024	0.0021	0.0002	0.0012
150 × 3 × 8 × 10	0.6678	0.0034	0.0034	0.0003	0.0028
150 × 3 × 10 × 10	0.4099	0.0037	0.0032	0.0003	0.0019
150 × 4 × 6 × 6	0.9303	0.0040	0.0039	0.0006	0.0018
150 × 4 × 8 × 6	0.9906	0.0033	0.0030	0.0005	0.0006
150 × 4 × 10 × 6	0.7585	0.0042	0.0040	0.0003	0.0027
150 × 4 × 6 × 8	0.9889	0.0030	0.0031	0.0001	0.0014
150 × 4 × 8 × 8	0.9996	0.0038	0.0032	0.0005	0.0009
150 × 4 × 10 × 8	0.9990	0.0045	0.0045	0.0008	0.0027
150 × 4 × 6 × 10	0.8063	0.0042	0.0040	0.0011	0.0021
150 × 4 × 8 × 10	0.9465	0.0059	0.0051	0.0005	0.0027
150 × 4 × 10 × 10	1.0000	0.0054	0.0050	0.0005	0.0016
150 × 5 × 6 × 6	0.9638	0.0041	0.0037	0.0006	0.0021
150 × 5 × 8 × 6	0.7202	0.0046	0.0045	0.0002	0.0023
150 × 5 × 10 × 6	0.6789	0.0047	0.0038	0.0006	0.0021
150 × 5 × 6 × 8	0.8192	0.0047	0.0047	0.0008	0.0029
150 × 5 × 8 × 8	0.8847	0.0049	0.0048	0.0008	0.0035
150 × 5 × 10 × 8	0.9930	0.0053	0.0048	0.0005	0.0024
150 × 5 × 6 × 10	1.0000	0.0043	0.0046	0.0008	0.0020
150 × 5 × 8 × 10	0.8159	0.0052	0.0046	0.0009	0.0028
150 × 5 × 10 × 10	0.7289	0.0056	0.0055	0.0009	0.0027

Table A7. Statistical results on the mean of the C metrics of the comparison algorithm.

Ins.	C							
	C (A,B)	C (B,A)	C (A,C)	C (C,A)	C (A,D)	C (D,A)	C (A,E)	C (E,A)
50 × 3 × 6 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 8 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 10 × 6	1.0000	0.0000	1.0000	0.0000	0.9375	0.0000	0.7917	0.0000
50 × 3 × 6 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 8 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 10 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 6 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 8 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 3 × 10 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 4 × 6 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.9375	0.0000
50 × 4 × 8 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.9565	0.0000
50 × 4 × 10 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.8667	0.0000
50 × 4 × 6 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.6500	0.0000
50 × 4 × 8 × 8	0.8462	0.0000	0.8889	0.0000	1.0000	0.0000	0.8000	0.0000
50 × 4 × 10 × 8	0.9231	0.0000	0.8000	0.0000	1.0000	0.0000	0.6429	0.0000
50 × 4 × 6 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 4 × 8 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 4 × 10 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.9565	0.0000
50 × 5 × 6 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 8 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 10 × 6	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 6 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 8 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 10 × 8	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
50 × 5 × 6 × 10	1.0000	0.0000	0.8750	0.0000	1.0000	0.0000	0.7143	0.0000
50 × 5 × 8 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.9545	0.0000

[illegible]

150 × 5 × 8 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.8400	0.0000
150 × 5 × 10 × 10	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	0.7667	0.0000

A represents QLNSGA, B represents QL-MoPSO, C represents MOEA/D, D represents NNIA, and E represents RIPG.

References

1. Zhou, J.Z.; Wen, X. Research on influencing factors and multiple driving paths of intelligent transformation in China's manufacturing industry. *J. Comput. Methods Sci. Eng.* **2021**, *21*, 1561–1573. <https://doi.org/10.3233/jcm-215086>.
2. Nambisan, S.; Wright, M.; Feldman, M. The digital transformation of innovation and entrepreneurship: Progress, challenges and key themes. *Res. Policy* **2019**, *48*, 103773. <https://doi.org/10.1016/j.respol.2019.03.018>.
3. Afsahi, M.; Kashan, A.H.; Ostadi, B. Optimizing decisions on under- and out-of-warranty products in a finite planning horizon. *Sci. Iran.* **2020**, *27*, 494–515. <https://doi.org/10.24200/sci.2018.50785.1879>.
4. Huang, Y.; Deng, Q.; Liu, X.; Huo, Z.; Liu, S.; Zhang, J. Integrated scheduling of multi-site repair tasks considering customer preferences under a novel two-dimensional combination warranty strategy. *Comput. Ind. Eng.* **2024**, *192*, 110249. <https://doi.org/10.1016/j.cie.2024.110249>.
5. Yang, S.L.; Wang, J.Y.; Li, W.D.; Xu, Z.G. Integrated optimisation of dynamic scheduling and reconfiguration for distributed reconfigurable flowshops via iterated greedy algorithm. *Int. J. Syst. Sci.-Oper. Logist.* **2025**, *12*, 2467782. <https://doi.org/10.1080/23302674.2025.2467782>.
6. Zou, W.Q.; Pan, Q.K.; Tasgetiren, M.F. An Effective Discrete Artificial Bee Colony Algorithm for Scheduling an Automatic-Guided-Vehicle in a Linear Manufacturing Workshop. *IEEE Access* **2020**, *8*, 35063–35076. <https://doi.org/10.1109/access.2020.2973336>.
7. Yuan, M.H.; Li, Y.D.; Zhang, L.Z.; Pei, F.Q. Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm. *Robot. Comput.-Integr. Manuf.* **2021**, *71*, 102141. <https://doi.org/10.1016/j.rcim.2021.102141>.
8. Jia, Y.H.; Yan, Q.; Wang, H.F. Q-learning driven multi-population memetic algorithm for distributed three-stage assembly hybrid flow shop scheduling with flexible preventive maintenance. *Expert Syst. Appl.* **2023**, *232*, 120837. <https://doi.org/10.1016/j.eswa.2023.120837>.
9. Li, P.Z.; Xue, Q.; Zhang, Z.T.; Chen, J.; Zhou, D.Q. Multi-objective energy-efficient hybrid flow shop scheduling using Q-learning and GVNS driven NSGA-II. *Comput. Oper. Res.* **2023**, *159*, 106360. <https://doi.org/10.1016/j.cor.2023.106360>.
10. Chen, Y.R.; Du, J.H.; Mumtaz, J.; Zhong, J.Y.; Rauf, M. An efficient Q-learning integrated multi-objective hyper-heuristic approach for hybrid flow shop scheduling problems with lot streaming. *Expert Syst. Appl.* **2025**, *262*, 125616. <https://doi.org/10.1016/j.eswa.2024.125616>.
11. Yang, H.B.; Wang, Z.Y.; Gao, Y.; Zhou, W. Bi-objective multi-mode resource-constrained multi-project scheduling using combined NSGA II and Q-learning algorithm. *Appl. Soft Comput.* **2024**, *152*, 111201. <https://doi.org/10.1016/j.asoc.2023.111201>.
12. Yine, K.C.; Lin, S.W.; Cheng, C.Y.; He, C.D. Iterated reference greedy algorithm for solving distributed no-idle permutation flowshop scheduling problems. *Comput. Ind. Eng.* **2017**, *110*, 413–423. <https://doi.org/10.1016/j.cie.2017.06.025>.
13. Sayah, A.; Aqil, S.; Lahby, M. Improved bio-inspired algorithms for scheduling distributed no-waiting flow shop with setup times. *Evol. Intell.* **2025**, *18*, 68. <https://doi.org/10.1007/s12065-025-01057-3>.
14. Ali, A.; Gajpal, Y.; Elmekawy, T.Y. Distributed permutation flowshop scheduling problem with total completion time objective. *Opsearch* **2021**, *58*, 425–447. <https://doi.org/10.1007/s12597-020-00484-3>.
15. Sleptchenko, A.; Al Hanbali, A.; Zijm, H. Joint planning of service engineers and spare parts. *Eur. J. Oper. Res.* **2018**, *271*, 97–108. <https://doi.org/10.1016/j.ejor.2018.05.014>.
16. Rahimi-Ghahroodi, S.; Al Hanbali, A.; Vliegen, I.M.H.; Cohen, M.A. Joint optimization of spare parts inventory and service engineers staffing with full backlogging. *Int. J. Prod. Econ.* **2019**, *212*, 39–50. <https://doi.org/10.1016/j.ijpe.2019.02.007>.
17. Bajestani, M.A.; Banjevic, D.; Beck, J.C. Integrated maintenance planning and production scheduling with Markovian deteriorating machine conditions. *Int. J. Prod. Res.* **2014**, *52*, 7377–7400. <https://doi.org/10.1080/00207543.2014.931609>.
18. Liao, W.Z.; Chen, M.C.; Yang, X.X. Joint optimization of preventive maintenance and production scheduling for parallel machines system. *J. Intell. Fuzzy Syst.* **2017**, *32*, 913–923. <https://doi.org/10.3233/jifs-161385>.
19. Zhang, K.G.; Xia, T.B.; Si, G.J.; Gebrael, N.; Wang, D.; Pan, E.R.; Xi, L.F. A Collaborative Scheduling Algorithm for Real-Time Production and Opportunistic Maintenance Under Cloud Manufacturing Paradigm. *IEEE Trans. Reliab.* **2024**, *73*, 170–185. <https://doi.org/10.1109/tr.2023.3311625>.

20. Miao, B.X.; Deng, Q.W.; Zhang, L.K.; Huo, Z.W.; Liu, X.Y. Collaborative scheduling of spare parts production and service workers driven by distributed maintenance demand. *J. Manuf. Syst.* **2022**, *64*, 261–274. <https://doi.org/10.1016/j.jmsy.2022.06.012>.
21. Vanlerberghe, J.; Maertens, T.; Walraevens, J.; De Vuyst, S.; Bruneel, H. On the optimization of two-class work-conserving parameterized scheduling policies. *4OR-Q. J. Oper. Res.* **2016**, *14*, 281–308. <https://doi.org/10.1007/s10288-016-0312-4>.
22. Yeh, C.W.; Fang, C.C. Optimal decision for warranty with consideration of marketing and production capacity. *Int. J. Prod. Res.* **2015**, *53*, 5456–5471. <https://doi.org/10.1080/00207543.2015.1010743>.
23. Jung, G.M.; Park, D.H. Optimal maintenance policies during the post-warranty period. *Reliab. Eng. Syst. Saf.* **2003**, *82*, 173–185. [https://doi.org/10.1016/s0951-8320\(03\)00144-3](https://doi.org/10.1016/s0951-8320(03)00144-3).
24. Zhang, Z.K.; Tang, Q.H.; Zhang, L.P.; Li, Z.X.; Cheng, L.X. A Q-learning-based multi-population algorithm for multi-objective distributed heterogeneous assembly no-idle flowshop scheduling with batch delivery. *Expert Syst. Appl.* **2025**, *263*, 125690. <https://doi.org/10.1016/j.eswa.2024.125690>.
25. Gao, Q.L.; Hu, H.L.; Liu, J.H.; Zhuang, C.B.; Ding, H.Y.; Li, H.T. Distributed assembly shop scheduling problem for complex products considering multiskilled worker assignment and transportation time. *Int. J. Prod. Res.* **2024**, 1–23. <https://doi.org/10.1080/00207543.2024.2383785>.
26. Li, L.; Chai, Z.Y. Energy-saving distributed flexible job-shop scheduling with fuzzy processing time in IIoT: A novel evolutionary multitasking algorithm. *J. Ind. Inf. Integr.* **2025**, *45*, 100829. <https://doi.org/10.1016/j.jii.2025.100829>.
27. Zuo, G.Z.; Jia, Z.Y.; Wu, Z.Y.; Shi, J.W.; 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. <https://doi.org/10.1016/j.eswa.2025.127882>.
28. Sugarindra, M.; Tontowi, A.E.; Herianto, Multitask Scheduling on Cloud Additive Manufacturing Using NSGA-II. *J. Ind. Eng. Manag.-Jiem* **2024**, *17*, 809–827. <https://doi.org/10.3926/jiem.7083>.
29. Liu, C.F.; Cheng, Q.; Cheng, C.Y.; Chu, H.Y.; Liu, Z.F.; Zhang, W. Enhancing energy efficiency in distributed hybrid forging flow-shops: A novel scheduling approach to capitalise on residual forging heat. *Int. J. Prod. Res.* **2025**, *63*, 2170–2189. <https://doi.org/10.1080/00207543.2024.2398580>.
30. Mei, Z.Y.; Lu, Y.J.; Lv, L.Y. Research on Multi-Objective Low-Carbon Flexible Job Shop Scheduling Based on Improved NSGA-II. *Machines* **2024**, *12*, 590. <https://doi.org/10.3390/machines12090590>.
31. Chen, C.; Ji, Z.C.; Wang, Y. NSGA-II applied to dynamic flexible job shop scheduling problems with machine breakdown. *Mod. Phys. Lett. B* **2018**, *32*, 1840111. <https://doi.org/10.1142/s0217984918401115>.
32. Zeng, L.; Shi, J.Y.; Li, Y.Y.; Wang, S.S.; Li, W.G. A Strengthened Dominance Relation NSGA-III Algorithm Based on Differential Evolution to Solve Job Shop Scheduling Problem. *CMC-Comput. Mater. Contin.* **2024**, *78*, 375–392. <https://doi.org/10.32604/cmc.2023.045803>.
33. Jing, X.-L.; Pan, Q.-K.; Gao, L.; Wang, Y.-L. An effective Iterated Greedy algorithm for the distributed permutation flowshop scheduling with due windows. *Appl. Soft Comput.* **2020**, *96*, 106629. <https://doi.org/10.1016/j.asoc.2020.106629>.
34. 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. Comput.* **2023**, *27*, 610–620. <https://doi.org/10.1109/tevc.2022.3175832>.
35. Wang, Z.; Deng, Q.; Zhang, L.; Li, H.; Li, F. Joint optimization of integrated mixed maintenance and distributed two-stage hybrid flow-shop production for multi-site maintenance requirements. *Expert Syst. Appl.* **2023**, *215*, 119422. <https://doi.org/10.1016/j.eswa.2022.119422>.
36. Zhao, F.Q.; Hu, X.T.; Wang, L.; Xu, T.P.; Zhu, N.N.; Zhu, N.N. A reinforcement learning-driven brain storm optimisation algorithm for multi-objective energy-efficient distributed assembly no-wait flow shop scheduling problem. *Int. J. Prod. Res.* **2023**, *61*, 2853–2871. <https://doi.org/10.1080/00207543.2022.2070786>.
37. Cui, G.Z.; Zhang, W.; Xu, W.W.; Bao, H.J. Efficient workflow scheduling using an improved multi-objective memetic algorithm in cloud-edge-end collaborative framework. *Sci. Rep.* **2025**, *15*, 29754. <https://doi.org/10.1038/s41598-025-08691-y>.
38. Li, J.; Li, S.F.; He, P.B.; Li, H.K. A multi objective collaborative reinforcement learning algorithm for flexible job shop scheduling. *Sci. Rep.* **2025**, *15*, 22838. <https://doi.org/10.1038/s41598-025-03681-6>.
39. Saidat, S.; Junoh, A.K.; Muhamad, W.; Yahya, Z. Modified job shop scheduling via Taguchi method and genetic algorithm. *Neural Comput. Appl.* **2022**, *34*, 1963–1980. <https://doi.org/10.1007/s00521-021-06504-7>.
40. Zhang, W.Q.; Geng, H.L.; Li, C.; Gen, M.S.; Zhang, G.H.; Deng, M.L. Q-learning-based multi-objective particle swarm optimization with local search within factories for energy-efficient distributed flow-shop scheduling problem. *J. Intell. Manuf.* **2025**, *36*, 185–208. <https://doi.org/10.1007/s10845-023-02227-9>.

41. Wang, G.C.; Li, X.Y.; Gao, L.; Li, P.G. Energy-efficient distributed heterogeneous welding flow shop scheduling problem using a modified MOEA/D. *Swarm Evol. Comput.* **2021**, *62*, 100858. <https://doi.org/10.1016/j.swevo.2021.100858>.
42. Zhu, K.K.; Gong, G.L.; Peng, N.T.; Zhang, L.Q.; Huang, D.; Luo, Q.; Li, X.Q. Dynamic distributed flexible job-shop scheduling problem considering operation inspection. *Expert Syst. Appl.* **2023**, *224*, 119840. <https://doi.org/10.1016/j.eswa.2023.119840>.
43. Zhang, Z.K.; Tang, Q.H.; Chica, M. Maintenance costs and makespan minimization for assembly permutation flow shop scheduling by considering preventive and corrective maintenance. *J. Manuf. Syst.* **2021**, *59*, 549–564. <https://doi.org/10.1016/j.jmsy.2021.03.020>.

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.