Expert Systems With Applications 281 (2025) 127496

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

An improved genetic algorithm for dual-resource constrained flexible job shop scheduling problem with tool-switching dependent setup time

Guodong Teng

School of Information Science and Technology, Hangzhou Normal University, NO.2318, Yuhangtang Rd, Yuhang District, Hangzhou, 311121, Zhejiang, China

ARTICLE INFO

Keywords:
Flexible job shop scheduling
Multi-objective optimization
Tool-switching optimization
Resource selection rule
Neighborhood search rule
NSGA-II

ABSTRACT

This paper studies the dual-resource constrained flexible job shop scheduling problem with tool-switching dependent setup time (DRCFJSP-TDST), considering the tools of two adjacent workpieces on the same machine to calculate the setup time. A mixed-integer linear programming (MILP) model is constructed to simultaneously minimize the makespan, maximum workload of machines, and total setup time. An improved genetic algorithm (IGA) is designed, featuring a three-vector chromosome encoding method and four heuristic resource selection rules (RND, SPT, CML, and HCL). Additionally, a neighborhood search operator with four specialized rules (N1, N2, N3, and N4) is introduced to enhance the local search ability and improve convergence of the IGA. To evaluate the effectiveness of IGA, sixty instances of the DRCFJSP-TDST are constructed, four key parameters of the IGA are calibrated using the Taguchi method, and three experiments are conducted. The first experiment demonstrates that the HCL rule among the four heuristic resource selection rules generated the highest-quality population. The second experiment verifies the effectiveness of the neighborhood search operator by comparing IGA with four neighborhood search rules and without the neighborhood operator, showing that the N3 rule is the most effective. The third experimental results validate that the proposed IGA algorithm significantly enhances scheduling performance across sixty instances benchmarks, with improvements ranging from 6.24% to 18.19% compared to state-of-the-art reference algorithms. These findings contribute to the field by demonstrating that integrating the resource selection rule, neighborhood search rule, and an external archive set into genetic algorithm yields an effective solution to complex DRCFJSP problems and its variants.

1. Introduction

Scheduling has direct impacts on the production efficiency and costs of a manufacturing system, thus it has attracted a great deal of research attentions since 1956 (Zhang, Ding, Zou, Qin, & Fu, 2019). As a complex scheduling problem where each job requires a given series of operations, or routing, to be processed on different machines, the job shop scheduling problem (JSP) has been studied for decades (Xiong, Shi, Ren, & Hu, 2022). At the beginning of the 1990s, an important extension of the JSP started to be investigated, allowing each operation to be performed on any machine within a specified subset of the available machines. This problem is called the flexible job shop scheduling problem (FJSP) (Dauzère-Pérès, Ding, Shen, & Tamssaouet, 2024), and there is a well-established methodological system for the study of typical FJSP (Gao et al., 2019; Li et al., 2022).

However, in practical production scenarios, auxiliary resources should not be overlooked in FJSP research, as they often serve as critical constraints to production scheduling. Machines and auxiliary resources represent potential capacity constraints on the FJSP, which is referred to as a dual-resource constrained FJSP (DRCFJSP) (Dhiflaoui, Nouri, & Driss, 2018). Common auxiliary resources in DRCFJSP include manpower, cutters, fixtures, pallets and so on (Liu et al., 2024). The machining shop is a prime example of machines and auxiliary resources constrained flexible job workshop (Wei, Tang, Li, Lei, & Wang, 2024). In this context, the identical machining centers are treated as the primary resources, while the tools are considered as auxiliary resources to complete all the required operations for a specific part (Liao, Zhang, Chen, & Song, 2024).

Due to multiple varieties, small batches, multiple types of parallelization, long process durations, and complex production processes, frequent switches between these processes imply complex changes among different types of tools (Liao, Zhang et al. 2024). Conventional studies of auxiliary resources, however, often assume that setup times are uniform constants or do not consider them at all. The uniform model generalizes an important issue that switching tools might require different times as it is affected by the type of tools to be replaced and the type of tool to be installed (Adjiashvili, Bosio et al. 2015). For example, switching an end milling tool with a turning tool requires

E-mail address: teng@hnzu.edu.cn.

https://doi.org/10.1016/j.eswa.2025.127496
Received 23 December 2024; Received in revised form 16 March 2025; Accepted 28 March 2025
Available online 9 April 2025

a longer switching time as the two tools have different tool holders. Meanwhile, switching an end milling tool with another end milling tool with a different size may require a shorter time than the previous process (Tri Windras Mara, Sutoyo et al. 2023). Motivated by the exploration of the DRCFJSP problem involving both machines and tools, this paper originally considers tool-switching dependent setup time (TDST) constraints and incorporates tool-switching optimization to enhance scheduling efficiency.

Commercial solvers such as CPLEX are highly effective for very small-scale DRCFJSP problems (Luo, Deng, Gong, Guo, & Liu, 2022; Luo, Deng, Xie, & Gong, 2023) but face significant challenges when applied to real-world, industrial-scale issues. The exponential growth in the solution space leads to unacceptable computational latency and memory constraints. To overcome these limitations, researchers (Amjad et al., 2018; Fan, Zhang, Liu, Shen, & Gao, 2022; Li, Qian, Hu, Chang, & Yang, 2019) have increasingly focused on metaheuristic approaches like genetic algorithms (GA). These algorithms strike a balance between computational efficiency and solution quality. Studies (Liu et al., 2024; Shi & Xiong, 2024) indicate that advanced GA variants can achieve near-optimal solutions within practical timeframes. This capability is essential in dynamic production environments where rapid rescheduling and high-quality solutions are required. Therefore, employing improved GA to tackle complex DRCFJSP problems represents a practical and effective strategy.

More precisely, this study focuses on the dual-resource constrained flexible job shop scheduling problem with tool-switching dependent setup time (DRCFJSP-TDST), which considers the tools of two adjacent workpieces on the same machine to determine the setup time. The contributions of this paper are generalized as follows.

(1) A novel mixed integer linear programming (MILP) model is established to simultaneously minimize the makespan, maximum workload of machines, and total setup time for DRCFJSP-TDST problem, and has been verified on a small-scall instance and validated by mainstream solver CPLEX.
(2) An improved genetic algorithm (IGA) is designed to solve the proposed MILP model with well-designed initialization, genetic operators, and problem-focused selection rule and neighborhood search rule.
(3) Extensive experiments are conducted to evaluate the performance of the proposed IGA by comparing it with its variants and other well-known multi-objective algorithms. The results demonstrate that the proposed IGA significantly improves scheduling efficiency across sixty benchmark instances.

The rest of this paper is organized as follows. The related works are reviewed in Section 2, and the MILP of DRCFJSP-TDST is built in Section 3. The proposed IGA is descripted in Section 4. Comparison experiments and discussions are performed in Section 5. Finally, the conclusion and future studies are listed in Section 6.

2. Literature review

2.1. FJSP model and algorithm

As an extension of typical JSP, FJSP was first proposed in 1990 (Brucker & Schlie, 1990) and had been proven to be an NP-hard problem (Fattahi, Saidi Mehrabad, & Jolai, 2007). This complexity has driven extensive research into mathematical modeling and optimization techniques. MILP has emerged as a predominant modeling framework due to its mathematical rigor and ability to provide theoretical benchmarks. As summarized in Table 1, nearly all existing studies have developed MILP formulations explicitly tailored to distinct problem parameters, variables, objectives, and constraints. However, existing models inadequately addressed the integrated optimization of dual-resource and TDST constraints – a gap this study aims to fill through a novel MILP formulation. Increasingly, the design of objectives in MILP models needs to align with emerging sustainability requirements, which drives a paradigm shift from conventional single-objective frameworks toward systematic multi-dimensional optimization.

Conventional scheduling research has primarily focused on single economic objective at the expense of environmental and social sustainability considerations, such as makespan minimization (Liu et al., 2024; Mahmoodjanloo, Tavakkoli-Moghaddam, Baboli, & Bozorgi-Amiri, 2020; Shen, Dauzère-Pérès, & Neufeld, 2018) or tardiness reduction (Fan et al., 2022; Liao et al., 2024). Recent studies on sustainable scheduling have progressively adopted the Triple Bottom Line (TBL) framework (Budak, 2020), which integrates economic efficiency (e.g., minimizing makespan Barak, Javanmard, & Moghdani, 2024), social responsibility (e.g., balancing machine or worker workloads Luo et al., 2023), and environmental sustainability (e.g., reducing energy consumption Zhou, Du, Liu, & Shen, 2024), while its application in production scheduling represents a new paradigm that has generated significant research opportunities (Han et al., 2020). Unlike traditional multi-objective optimization, the TBL framework requires explicit alignment of objectives with sustainability dimensions. For instance, Fathollahi-Fard, Woodward, and Akhrif (2021) introduced the mapping of makespan, the number of work days lost, and energy consumption to economic, social, and environmental objectives, respectively, in a sustainable distributed permutation flow shop scheduling problem (DPFSP). Similarly, building upon the TBL framework, Fallahi, Bani, and Varmazyar (2025) proposed an integrated scheduling model for unrelated batch processors that simultaneously optimizes total cost, social benefit, and energy consumption. Following this trend, the present study incorporates the scheduling entities (e.g., operations, machines, and tools) of the DRCFJSP-TDST problem within the TBL framework, emphasizing economic objectives (joint optimization of makespan and total setup time) while advancing social objective (maximum workload minimization).

Over the past 30 years, numerous researchers have applied a variety of methods and techniques to solve JFSP, which include (1) exact methods, e.g., mixed integer programming (Park & Ham, 2022), branch and bound (Soto et al., 2020), or constraint programming (Kasapidis, Dauzère-Pérès, Paraskevopoulos, Repoussis, & Tarantilis, 2023), (2) heuristics (Gao, Suganthan, Tasgetiren, Pan, & Sun, 2015; Ortíz, Betancourt, Negrete, De Felice, & Petrillo, 2018), (3) metaheuristics, such as simulated annealing (SA) (Cruz-Chávez, Martínez-Rangel, & Cruz-Rosales, 2017), Tabu search (Hajibabaei & Behnamian, 2021), variable neighborhood search (Kemmoé-Tchomté, Lamy, & Tchernev, 2017), the particle swarm optimization (Ding & Gu, 2020), ant colony optimization (Rossi, 2014), and GA, (4) learning-based methods, e.g., neural networks and reinforcement learning (Lei et al., 2022; Li, Liao et al., 2023; Song, Chen, Li, & Cao, 2022; Yuan et al., 2024).

Given the need for both time efficiency and solution quality, GA and its variants have emerged as one of the most widely adopted and effective optimization approaches for FJSPs. As evidenced by the comprehensive literature survey in Table 1, a significant body of research has successfully applied GA-based frameworks, including NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002) and memetic algorithms (MA), to address various FJSP complexities. Recent advancements (Shi & Xiong, 2024) in multi-objective GA optimization further underscore the effectiveness of hybrid approaches, which integrate NSGA-II's Pareto-driven framework with enhanced genetic operators and problem-specific adaptations. These developments highlight the continued relevance and superiority of GA-based methods in tackling the intricate challenges of FJSPs, particularly in balancing exploration and exploitation while handling multiple objectives.

Additionally, although CPLEX, as an exact method, is constrained by problem complexity and impractical for production applications, it remains widely utilized in FJSP research. Its primary role is to validate model correctness and benchmark algorithmic performance. As shown in Table 1, a significant number of algorithms adopted this strategy, first validating the MILP model with CPLEX and then employing metaheuristic algorithms to solve the problem.

2.2. FJSP with dual-resource constrained

It is common that multiple primary or auxiliary resources are needed in real-world production environments to execute a predetermined manufacturing plan (Liao et al., 2024). These resources can include combinations such as machines and workers, machines and fixtures, computer numerical control (CNC) machining centers and tools, reconfigurable manufacturing systems (RMS) and auxiliary modules (AM), etc. Subsequently, relevant research results will be presented in sequence according to these four categories.

Luo et al. (2022) investigated a distributed FJSP considering machine and worker arrangement. They employed an improved memetic algorithm (IMA) with an adaptive neighborhood search method to minimize the makespan, the maximum workload of machines, and the maximum workload of workers. To minimize the same objectives for FJSP with worker cooperation flexibility, Luo et al. (2023) proposed a Pareto-based two-stage evolutionary algorithm (PTSA). The algorithm includes a well-tailored initialization and NSGA-II for global exploration, and a competitive objective-based local search operator for improved local search ability. Wei et al. (2024) formulated a resource constrained FJSP, which aims to cope with uncertain events by simultaneously adjusting the machine, worker and process parameters of the original schedule. A multi-objective optimization model is constructed to minimize the makespan, worker cost, machine energy consumption and deviation index. Furthermore, an IMA was developed for solving the proposed problem.

Thörnblad, Strömberg, Patriksson, and Almgren (2015) presented a time-indexed formulation of the FJSP on the machines and fixtures. The objective was to minimize a weighted sum of the completion times and tardiness for the jobs, and they proposed a fast iterative heuristic approach to finding a suitable value. Lee, Shin, and Lee (2020) addressed an operations scheduling problem for an advanced flexible manufacturing system with multi-fixturing pallets for pallet input sequencing, pallet routing and pallet process sequencing. A fast heuristic approach was proposed that used a combination of priority rules for the objectives of minimizing makespan and total tardiness. Liu et al. (2024) provided an optimal strategy for the multi-resource constrained FJSP with fixture-pallet combinatorial optimization to minimize the maximal completion time of production tasks. Then, a novel GA integrated with feasibility correction strategy and self-learning variable neighborhood search (VNS) was proposed to address the complicated scheduling problem.

Tian, Gao, Zhang, Chen, and Wang (2023) studied the impact of cutting-tool degradation in the manufacturing process and formulated an FJSP model that integrates tool life prediction and machining power prediction. A multi-objective optimization model was developed to minimize the makespan, total machine tool load, the frequency of machine tool start-ups and shutdowns, the number of cutting tool changes, and the total energy consumption (TEC) of the workshop. The NSGA-II algorithm was employed to solve the model. Liao et al. (2024) tackled an FJSP on identical CNC machining centers, considering the simultaneous availability of machines and tools, and optimizing the total weighted tardiness, and then proposed an efficient artificial bee colony algorithm (ABC) to solve the scheduling problem.

To enhance production flexibility, advanced reconfigurable machine tools (RMT) are equipped with auxiliary modules (AM). Fan et al. (2022) investigated an FJSP with machine reconfigurations (MR) aimed at minimizing the total weighted tardiness (TWT). For large-scale problems, they proposed an improved GA that includes specific encoding, decoding, and genetic operators.

The above literature reveals that various heuristic and metaheuristics algorithms are employed to solve the DRCFJSP with different single or multiple optimization objectives, but the setup time of primary and auxiliary resources is typically ignored during the operation sequencing and resources assignment. The setup time, nevertheless, has a certain impact on production scheduling in real industries.

2.3. FJSP with setup time constrained

Setup time is the time required to prepare the necessary resource (people, machines) to perform a task (operation, job) (Allahverdi, 2015). FJSP with sequence-dependent setup times (SDST) indicates that the magnitude of setup times is determined by the current job as well as the direct previous job processed on the same machines (Wu, Peng, Xiao, & Wu, 2021). According to the constraint of different dependency configurations on setup time, SDST may include operation-dependent or sequence-dependent setup, machine-reconfiguration setup, and anticipatory setup, etc. Next, relevant research findings will be introduced sequentially based on these four classifications.

Shen et al. (2018) addressed an FJSP with operation-dependent setup times, focusing on minimizing the makespan. They developed a Tabu search algorithm that integrates specific neighborhood functions and a diversification structure to explore solutions efficiently. Li et al. (2019) proposed an elitist non-dominated sorting hybrid algorithm to tackle the multi-objective FJSP, where setup times depend on a machine's current and next operations. Their approach aims to minimize both the makespan and total setup costs. Barak et al. (2024) introduced a modified multi-objective invasive weed optimization (MOIWO) algorithm, further enhanced by a fuzzy sorting mechanism for competitive exclusion. This method optimizes the assignment of operations to multi-skilled machines and operators, targeting reductions in operator idleness costs and expenses associated with operation-dependent setup times. Chen et al. (2025) extended the FJSP by integrating single crane transportation and sequence-dependent setup times. A multipopulation evolutionary multi-task optimization (EMTO) framework combined with genetic programming (GP) is developed to minimize makespan and total tardiness. Zhao, Li, Zhu, Xu, et al. (2025) investigated the multi-objective distributed heterogeneous FJSP with sequence-dependent setup times, developing a multi-objective fitness landscape-based estimation of distribution algorithm (MFLEDA) to address the dual objectives of minimizing makespan and TEC.

Mahmoodjanloo et al. (2020) explored a variant of the FJSP that includes reconfigurable machine tools, with the objective of minimizing the makespan. They introduced configuration-dependent setup times, which are influenced by the current and next configurations of a machine, necessitating changes in auxiliary modules. A self-adaptive differential evolution (DE) algorithm was employed to address this challenge. Wu et al. (2021) investigated the FJSP considering fixture loading and unloading times, proposing a multi-objective mathematical model to minimize both the makespan and total setup time. To tackle this problem, they enhanced the NSGA-II algorithm with a strategy for reducing setup times. Fan et al. (2022) examined reconfigurable machine configuration setup times involving auxiliary modules in the FJSP, aiming to minimize total weighted tardiness (TWT). They designed an improved GA and utilized a disjunctive graph along with a modified k-insertion technique to identify and analyze bottlenecks in the scheduling process.

By integrating the selection of auxiliary resources with the FJSP and subdividing SDST into fixed preset assembly and disassembly times or fixed startup and shutdown times on machines, the approach is termed anticipatory. Jiang et al. (2022) addressed the energy-efficient FJSP for manufacturing aerospace complex components, aiming to optimize TEC, makespan, and processing cost. They utilized an improved ABC algorithm, accounting for setup energy required for tool exchanges and settings, which depend on anticipatory tool changing and setting times. Zhou et al. (2024) introduced a novel machine-fixture-pallet resource-constrained FJSP that incorporates loading and unloading times within a pallet automation system. The objective was to minimize the makespan using an improved adaptive large neighborhood search algorithm (IALNS) algorithm enhanced with SA for local search. This approach considers each operation's clamping and placement by available fixture-pallets, taking into account anticipatory loading and unloading times. Luo, Li, Gong, and Gao (2025) modeled

the multimodal multi-objective flexible job shop scheduling problem with variable-speed machines (MMFJSP-S) under fixed setup times exclusively during machine startup and shutdown phases, proposing an affinity propagation hierarchical memetic algorithm (APHMA) to simultaneously minimize makespan and TEC.

From the above, it is evident that setup time have garnered significant attention in the field of production scheduling and are extensively studied in resource-constrained flexible job shop environments. However, to the best of our knowledge, there are no studies that address the DRCFJSP with TDST. Therefore, it could be both necessary and valuable to explore TDST factors in the context of DRCFJSP.

2.4. Key insights and research directions

The comprehensive analysis of FJSP literatures reveals several critical findings that shape the direction of this research.

First, while MILP modeling has proven invaluable in FJSP research, its effectiveness heavily depends on careful customization to specific problem contexts. Existing models often fail to capture the intricate interdependencies between dual-resource and setup time constraints, particularly in tool-intensive manufacturing environments.

Second, the evolution of scheduling objectives reflects a paradigm shift toward sustainable manufacturing. Minimizing the maximum workload directly addresses social equity by preventing worker burnout through fair task allocation. While setup time reduction may indirectly lower energy consumption, its primary impact lies in improving equipment utilization, thereby jointly characterizing the economic performance of production systems.

Third, GA and its variants has established itself as a dominant methodology for FJSP solutions, particularly when enhanced with problem-specific adaptations. Yet, its application to problems combining dual-resource constraints with tool-switching dependent setup times remains underexplored, presenting opportunities for algorithmic innovation.

Fourth, the classification of setup time constraints into operation-dependent, machine-reconfiguration, and anticipatory types provides a useful framework. However, the integration of these categories with dual-resource constraints, particularly in tool-switching scenarios, remains a significant gap in the literature.

To address these challenges, this study constructs a novel multi-objective MILP model tailored to the specific problem and employs an improved GA algorithm to solve it. The MILP model explicitly captures the interdependencies between dual-resource constraints and tool-switching dependent setup times, while the improved algorithm incorporates problem-specific encoding, decoding, resource selection and neighborhood search rules. This integrated approach aims to provide high-quality solutions that balance economic and social objectives in complex manufacturing scenarios. Notably, direct environmental metrics (e.g., energy consumption, carbon emissions) are not explicitly included in this study. This limitation is further discussed in Section 6 to guide future research.

3. Model construction

3.1. Problem description

The DRCFJSP-TDST addressed in this paper can be described as follows. In a flexible machine workshop, given a set of n independent jobs $J = \{J_1, J_2, \dots, J_n\}$, where ith job J_i contains N_i operations $\{O_{i,1}, O_{i,2}, \dots, O_{i,N_i}\}$ to be processed in accordance with the given sequence. The jth operation $O_{i,j}$ of J_i can be processed by a machine with a tool selected from an available machine set $M_{i,j}$ and alternative tool set $A_{i,j}$, respectively.

For each operation, it must be processed once under the premise of fulfilling operation sequence, machine, and tool constraints. Operation sequence constraint requires that the next operation of a job can be processed only if its previous one is handled. Machine or tool constraint specifies that a machine or tool can process at most one operation at a time.

After determining the sequence of operations and the allocation of machines and tools for processing, the subsequent step is to evaluate the tool-switching setup time between consecutive operations on the same machine, e.g., the machine with same tool, the machine with different tools of same tool type, and the machine with different tool types. When an operation is completed, the tool used for processing this operation can be disassembled from the machine, and becomes available for other pending operations, unless the next operation on that machine also requires this same tool, which remains assembled on the machine.

The following assumptions are considered in the DRCFJSP-TDST:

(1) There is no preemption for all the jobs.
(2) All jobs can be processed at time 0, and all the machines and tools are available from time 0.
(3) Each job's operations must follow a specified sequence.
(4) Operations from different jobs are independent and can be processed in any order.
(5) A job can only have one operation being processed at a time.
(6) There are no interrupt and malfunction during processing.
(7) A machine can only process one operation at a time.
(8) A tool can only be attached to one machine at a time.
(9) The transfer time are not considered.
(10) The disassembly time of tools are not considered.

The task of the DRCFJSP-TDST is to make the decisions of operation sequence, machine selection and tool assignment, and the objective is to simultaneously minimize the makespan, maximum workload of machines, and total setup time.

3.2. MILP formulation

Indices

i, i': Index of jobs, $i, i' \in \{1, 2, \dots, n\}$, $i \neq i'$
j, j': Index of operations, $j, j' \in \{1, 2, \dots, N_i\}$, $j \neq j'$
m: Index of machines, $m \in \{1, 2 \dots, k\}$
c, c': Index of tool types, $c, c' \in \{1, 2, \dots, v\}$, $c \neq c'$
a, a': Index of tools, $a, a' \in \{1, 2, \dots, s\}$
p: Index of positions on machine m, $p \in \{1, 2, \dots, g\}$
q: Index of positions on tool a, $q \in \{1, 2, \dots, h\}$

Parameters

n: Number of jobs
N_i: Number of operations of job i
k: Number of machines
v: Number of tool types
s: Number of tools, $s = \sum_{c=1}^{v} N_c$
g: Number of positions on machine m
h: Number of positions on tool a
I: Set for jobs, where $I = \{1, 2, \dots, n\}$
O_i: Set for operations of job i, where $O_i = \{1, 2, \dots, N_i\}$
M: Set for machines, where $M = \{1, 2, \dots, k\}$
A: Set for tools, where $A = \{1, 2, \dots, s\}$
$M_{i,j}$: Available machine set of operation $O_{i,j}$
$A_{i,j}$: Available tool set of operation $O_{i,j}$
P_α: Set for position of machine m, where $P_\alpha = \{1, 2, \dots, g\}$
P_β: Set for position of machine m, where $P_\beta = \{1, 2, \dots, g-1\}$
Q_α: Set for position of tool a, where $Q_\alpha = \{1, 2, \dots, h\}$
Q_β: Set for position of tool a, where $Q_\beta = \{1, 2, \dots, h-1\}$
N_{imax}: Maximum number of operations of each job, $N_{imax} = max\{N_i\}$
N_c: Number of tools belonging to tool type c

Table 1
Literature review of FJSP with dual-resource and setup constraints.

Authors	Auxiliary resource	Setup dependent	Objectives	Methodologies
Thörnblad et al. (2015)	fixture	–	makespan, tardiness	heuristic
Shen et al. (2018)	–	operation	makespan	Tabu
Li et al. (2019)	–	operation	makespan, total setup	GA
Mahmoodjanloo et al. (2020)	tool	configuration	makespan	CPLEX, DE
Lee et al. (2020)	fixture	–	makespan, tardiness	heuristic
Wu et al. (2021)	fixture	configuration	makespan, total setup	GA
Fan et al. (2022)	module	configuration	tardiness	CPLEX, GA
Jiang et al. (2022)	tool	anticipatory	makespan, TEC, processing cost	ABC
Luo et al. (2022)	worker	–	makespan, machines workload, workers workload	CPLEX, MA
Tian et al. (2023)	tool	–	makespan, TEC, tool load and change	GA
Luo et al. (2023)	worker	–	makespan, machines workload, workers workload	CPLEX, GA
Wei et al. (2024)	worker	–	makespan, TEC, worker cost	MA
Liu et al. (2024)	fixture	–	makespan	CPLEX, GA
Liao et al. (2024)	tool	–	tardiness	CPLEX, ABC
Barak et al. (2024)	worker	operation	makespan, total setup	MOIWO
Zhou et al. (2024)	fixture	anticipatory	makespan, TEC, processing cost	SA
Chen et al. (2025)	–	sequence	makespan, tardiness	EMTO-GP
Zhao et al. (2025)	–	sequence	makespan, TEC	EDA
Luo et al. (2025)	–	anticipatory	makespan, TEC	CPLEX, MA
This study	tool	tool-switching	makespan, machines workload, total setup	CPLEX, GA

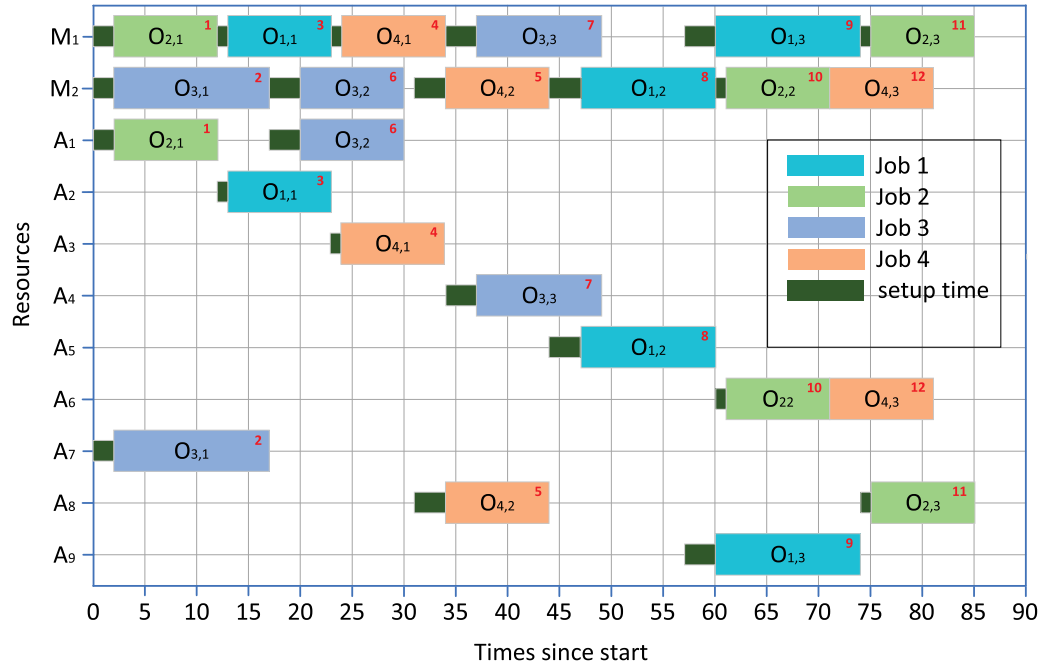

Fig. 1. A special solution for the example problem of the DRCFJSP-TDST.

N_{cmax}: Maximum number of tools belonging to each tool type, $N_{cmax} = max\{N_c\}$

$T_{i,j,m}$: Processing time of operation $O_{i,j}$ on machine m

TP_a: Tool type of tool a

$ST_{m,a',a}$: Tool-switching dependent setup time when the tools is switched from a' to a on machine m. $ST_{m,a',a} > 0$, if $a' \neq a$; $ST_{m,a',a} = 0$, if $a' = a$; $ST_{m,a',a} < ST_{m,a'',a}$, if $TP_{a'} = TP_a$ and $TP_{a''} \neq TP_a$

$ST_{m,a}$: The initial setup time of machine m and tool a

L: A large positive number

Variables

$x_{i,j,m,p}$: Equals to 1 if operation $O_{i,j}$ is processed on position p of machine m and otherwise equals to 0

$y_{i,j,a,q}$: Equals to 1 if operation $O_{i,j}$ is processed on position q of tool a and otherwise equals to 0

$z_{i,j,m,a}$: Equals to 1 if operation $O_{i,j}$ is processed on machine m with tool a and otherwise equals to 0

Table 2
Example of the DRCFJSP-TDST.

Job	Operation	Machine	Tool type	Tool set	Processing time
J_1	$O_{1,1}$	M_1	C_1	$\{A_1, A_2, A_3\}$	10
		M_2	C_1	$\{A_1, A_2, A_3\}$	12
	$O_{1,2}$	M_1	C_2	$\{A_4, A_5, A_6\}$	10
		M_2	C_2	$\{A_4, A_5, A_6\}$	13
	$O_{1,3}$	M_1	C_3	$\{A_7, A_8, A_9\}$	14
		M_2	C_3	$\{A_7, A_8, A_9\}$	10
J_2	$O_{2,1}$	M_1	C_1	$\{A_1, A_2, A_3\}$	10
		M_2	C_1	$\{A_1, A_2, A_3\}$	12
	$O_{2,2}$	M_1	C_2	$\{A_4, A_5, A_6\}$	12
		M_2	C_2	$\{A_4, A_5, A_6\}$	10
	$O_{2,3}$	M_1	C_3	$\{A_7, A_8, A_9\}$	10
		M_2	C_3	$\{A_7, A_8, A_9\}$	15
J_3	$O_{3,1}$	M_1	C_3	$\{A_7, A_8, A_9\}$	10
		M_2	C_3	$\{A_7, A_8, A_9\}$	15
	$O_{3,2}$	M_1	C_1	$\{A_1, A_2, A_3\}$	12
		M_2	C_1	$\{A_1, A_2, A_3\}$	10
	$O_{3,3}$	M_1	C_2	$\{A_4, A_5, A_6\}$	12
		M_2	C_2	$\{A_4, A_5, A_6\}$	10
J_4	$O_{4,1}$	M_1	C_1	$\{A_1, A_2, A_3\}$	10
		M_2	C_1	$\{A_1, A_2, A_3\}$	15
	$O_{4,2}$	M_1	C_3	$\{A_7, A_8, A_9\}$	12
		M_2	C_3	$\{A_7, A_8, A_9\}$	10
	$O_{4,3}$	M_1	C_2	$\{A_4, A_5, A_6\}$	12
		M_2	C_2	$\{A_4, A_5, A_6\}$	10

$st_{m,p,a',a}$: Equals to 1 if at the beginning of operation p of machine m, the machine's tool is switched from a' to a and otherwise equals to 0, where $a' \neq a$. Especially, $st_{m,p,a,a} = 0$

$so_{i,j}$: Starting time of operation $O_{i,j}$

$sm_{m,p}$: Starting time of position p on machine m

$sa_{a,q}$: Starting time of position q on tool a

Objective

The objective (1) is to simultaneously minimize the makespan f_1, maximum workload of machines f_2, and total setup time f_3.

$$min\{f_1, f_2, f_3\} \tag{1}$$

$$f_1 = max\left\{so_{i,N_i} + \sum_{m\in M}\sum_{p\in P_\alpha} x_{i,N_i,m,p}T_{i,N_i,m}, \forall i\right\} \tag{2}$$

$$f_2 = max\left\{\sum_{i\in I}\sum_{j\in O_i}\sum_{p\in P_\alpha} x_{i,j,m,p}T_{i,j,m}, \forall m\right\} \tag{3}$$

$$f_3 = \sum_{m\in M}\sum_{p\in P_\alpha}\sum_{a'\in A}\sum_{a\in A} st_{m,p,a',a}ST_{m,a',a} \tag{4}$$

Subject to:

$$so_{i,j} + \sum_{m\in M_{i,j}}\sum_{p\in P_\alpha} x_{i,j,m,p}T_{i,j,m} \le so_{i,j+1}, \forall i, j \in \{1, 2, \ldots, N_i - 1\} \tag{5}$$

$$\sum_{m\in M_{i,j}}\sum_{p\in P_\alpha} x_{i,j,m,p} = 1, \forall i, j \tag{6}$$

$$\sum_{a\in A_{i,j}}\sum_{q\in Q_\alpha} y_{i,j,a,q} = 1, \forall i, j \tag{7}$$

$$\sum_{m\in M_{i,j}}\sum_{a\in A_{i,j}} z_{i,j,m,a} = 1, \forall i, j \tag{8}$$

$$\sum_{a'\in A}\sum_{a\in A} st_{m,p,a',a} = 1, \forall m, p \tag{9}$$

$$\sum_{i'\in I}\sum_{j'\in O_i} x_{i',j',m,p+1} \le \sum_{i\in I}\sum_{j\in O_i} x_{i,j,m,p} \le 1, \forall m, p \in P_\beta \tag{10}$$

$$\sum_{i'\in I}\sum_{j'\in O_i} y_{i',j',a,q+1} \le \sum_{i\in I}\sum_{j\in O_i} y_{i,j,a,q} \le 1, \forall a, q \in Q_\beta \tag{11}$$

$$\sum_{p\in P_\alpha} x_{i,j,m,p} = \sum_{a\in A_{i,j}} z_{i,j,m,a}, \forall i, j, m \in M_{i,j} \tag{12}$$

$$\sum_{q\in Q_\alpha} y_{i,j,a,q} = \sum_{m\in M_{i,j}} z_{i,j,m,a}, \forall i, j, a \in A_{i,j} \tag{13}$$

$$st_{m,p+1,a',a} \ge x_{i,j,m,p+1} + x_{i',j',m,p} + z_{i,j,m,a} + z_{i',j',m,a'} - 3, \quad \forall i, j, i', j', m, p \in P_\beta, a', a \tag{14}$$

$$so_{i,j} - L(1 - x_{i,j,m,p}) \le sm_{m,p} \le so_{i,j} + L(1 - x_{i,j,m,p}), \forall i, j, m, p \tag{15}$$

$$so_{i,j} - L(1 - y_{i,j,a,q}) \le sa_{a,q} \le so_{i,j} + L(1 - y_{i,j,a,q}), \forall i, j, a, q \tag{16}$$

$$sm_{m,p+1} \ge sm_{m,p} + x_{i,j,m,p}T_{i,j,m} + \sum_{a'\in A}\sum_{a\in A} st_{m,p+1,a',a}ST_{m,a',a} - L(1 - x_{i,j,m,p}), \forall i, j, m, p \in P_\beta \tag{17}$$

$$sa_{a,q+1} \ge sa_{a,q} + y_{i,j,a,q}T_{i,j,m} - L(1 - y_{i,j,a,q}), \forall i, j, a, q \in Q_\beta \tag{18}$$

$$so_{i,j} \ge 0; sm_{m,p} \ge 0; sa_{a,q} \ge 0, \forall i, j, m, a, p, q \tag{19}$$

In the model, three objectives $f1$, $f2$, and $f3$ are calculated by formulas (2), (3), and (4), respectively. Inequality (5) ensures the constraint of the operation sequence. Equality (6) specifies that each operation can be processed only once at a position on one machine. Equality (7) indicates that each operation can be processed only once at a position on one tool. Equality (8) shows that each operation must be processed once with one machine and one tool. Equality (9) ensures that for each operation processed on machine, there is precisely one tool switching event, which may or may not involve an actual tool change. Inequality (10), (11) guarantee the constraints of the processing sequence on machines and tools respectively. Equality (12), (13) determine the constraints of processed times on machines and tools, respectively. Equality (14) records a tool switch for the next operation on a machine. Inequality (15) lists the constraint of start time between machines and operations. Inequality (16) gives the constraint of start time between tools and operations. Inequality (17), (18) require that there are no overlaps on machines and tools during processing, and guarantee that the start time of each machine position be greater than the start time of the previous position plus a possible setup time and the processing time of an operation. Inequality (19) defines that the variables are nonnegative.

3.3. Objective conflict analysis

(1) Trade-off between f_1 and f_2

The minimization of f_1 inherently conflicts with the reduction of f_2. Aggressive scheduling strategies aimed at shortening f_1 often lead to the concentration of operations on fewer machines, as indicated by Formulas (2), where minimizing so_{i,N_i} tends to drive assignments $x_{i,j,m,p}$ toward a limited set of machines. This clustering effect increases the cumulative processing times $T_{i,j,m}$ on these machines, thereby elevating f_2 as shown in Formulas (3). Conversely, strategies to balance machine loads and reduce f_2 may introduce idle periods, which can extend f_1. This inherent trade-off highlights the necessity of flexible resource allocation policies to achieve a balance between temporal efficiency and machine utilization.

(2) Trade-off between f_1 and f_3

Reducing f_1 often involves relocating operations to different machines. However, due to the presence of TDST, such relocations may increase setup times between operations, thereby raising f_3. On the other hand, if the same tool is used before or after the relocated operation, setup times may be minimized, potentially reducing f_3. For example, a schedule that achieves $f_3 \approx 0$ (no setups) would require dedicating machines and tools to specific operations, which could create sequential bottlenecks and extend f_1. This illustrates the importance of carefully considering operational dependencies and context when optimizing process allocation to effectively balance f_1 and f_3. A similar trade-off relationship exists between f_2 and f_3.

Table 3
Different operation sequence with different objectives.

Case	Operation sequence	f_1	f_2	f_3
1	$O_{2,1}$, $O_{1,1}$, $O_{3,1}$, $O_{2,2}$, $O_{1,2}$, $O_{3,2}$, $O_{4,1}$, $O_{4,2}$, $O_{4,3}$, $O_{2,3}$, $O_{3,3}$, $O_{1,3}$	88	67	28
2	$O_{3,1}$, $O_{4,1}$, $O_{2,1}$, $O_{3,2}$, $O_{4,2}$, $O_{1,1}$, $O_{4,3}$, $O_{2,2}$, $O_{3,3}$, $O_{1,2}$, $O_{1,3}$, $O_{2,3}$	85	75	25
3	$O_{3,1}$, $O_{2,1}$, $O_{1,1}$, $O_{3,2}$, $O_{4,1}$, $O_{4,2}$, $O_{1,2}$, $O_{2,2}$, $O_{1,3}$, $O_{2,3}$, $O_{4,3}$, $O_{3,3}$	84	62	29
4	$O_{1,1}$, $O_{2,1}$, $O_{2,2}$, $O_{4,1}$, $O_{1,2}$, $O_{3,1}$, $O_{3,2}$, $O_{4,2}$, $O_{3,3}$, $O_{1,3}$, $O_{4,3}$, $O_{2,3}$	88	63	26

Table 4
Computational results of CPLEX for optimizing f_1.

Ins.	n	N_{imax}	k	N_{cmax}	v	u_T	u_{ST}	f_1	f_2	f_3	Time (s)
N01	2	2	2	2	2	50	5	110	105	10	0.22
N02	3	2	2	2	2	50	5	152	141	23	2.19
N03	3	3	2	3	3	50	5	227	212	35	269.08
N04	3	4	3	3	3	50	5	–	–	–	>14 400
N05	4	3	3	3	3	50	5	–	–	–	>14 400

3.4. Verification of a small-scale example

For clarity, an example of the DRCFJSP-TDST is given in Table 2 and Fig. 1 to facilitate understanding. As shown in Table 2, this example includes a total of four jobs $\{J_1, J_2, J_3, J_4\}$, two machines $\{M_1, M_2\}$, three tool types $\{C_1, C_2, C_3\}$, and nine tools $\{A_1, A_2, \ldots, A_9\}$. Each job respectively include three operations, and each operation has two available machines and three available tools belonging to one tool type. The triplet of job, operation, and machine, also known as parameter $T_{i,j,m}$, determines the processing time of the operation. Taking the job J_1 as an example, it includes three sequential operations $O_{1,1}$, $O_{1,2}$, and $O_{1,3}$. For operation $O_{1,1}$, it can be processed on either of the available machines M_1 or M_2. The only available tool set $\{A_1, A_2, A_3\}$, all of which belong to tool type C_1. If the machine M_1 is selected, the processing time $T_{1,1,1}$ of operation $O_{1,1}$ is 10, regardless of which tool from type C_1 is chosen. In addition, to simplify the configuration in this example, the initial setup time is uniformly set to 2. The tool-switching dependent setup time is set to 1 when switching two tools between the same tool type on the same machine, and it is set to 3 when switching two tools between different tool types on the same machine.

After the parameters of the example are given, all the operations are sorted by a certain scheduling strategy, such as a random strategy. In this example, the sequence of operations is arranged as $O_{2,1}$, $O_{3,1}$, $O_{1,1}$, $O_{4,1}$, $O_{4,2}$, $O_{3,2}$, $O_{3,3}$, $O_{1,2}$, $O_{1,3}$, $O_{2,2}$, $O_{2,3}$, $O_{4,3}$. The scheduling strategy then assigns each operation to a specific machine and tool, determining the setup time, start time, and end time for each operation on the assigned resources. As shown in Fig. 1, the final processing result can be represented by a Gantt chart. The processing order is shown in the upper right corner of each operation, and the rectangular bar in front of the operation represents the setup time. For a specific solution of this example, the objective value can be clearly identified. The makespan $f_1 = 85$ which is same as the end time of the last operation $O_{2,3}$, the maximum the workload of machines $f_2 = 68$ which is same as workload of machine M_2, and total setup time $f_3 = 23$.

In this scheduling outcome, the first operation of job J_2, denoted as $O_{2,1}$, is processed on machine M_1 using tool A_1. The initial setup time is 2, resulting in a start time of 2 and an end time of 12. The third sequential operation $O_{1,1}$ of job J_1 is assigned to machine M_1 with tool A_2. The setup time before operation $O_{1,1}$ is 1 because the preceding operation $O_{2,1}$ on the same machine uses a tool A_1 of the same type as A_2. Consequently, the start time for $O_{1,1}$ is 13, and the end time is 23. The setup time before the eighth sequential operation $O_{1,2}$ is 3 because the preceding process $O_{4,2}$ on machine M_2 uses a different tool type, compared to the tool A_5, Then, $O_{1,2}$ starts at time 47 and ends at time 60. The setup time before the twelfth sequential operation $O_{4,3}$ is 0 because the preceding process $O_{2,2}$ is also assigned to the same machine M_2 and uses the same tool A_6. That is to say, there is no need to switch tools between these two operations $O_{2,2}$ and $O_{4,3}$. Thus, the start time and the end time of $O_{4,3}$ is 71 and 81, respectively. Clearly from Fig. 1, an operation can start only when the selected machine and tool become available.

To illustrate the impact of different scheduling strategies on the three objectives, four experimental scenarios are designed by varying the operation sequence while maintaining consistent machine-tool assignments. As shown in Table 3, transitioning from Case 1 to Case 2 reduces f_1 but increases f_2, while moving from Case 1 to Case 3 decreases both f_1 and f_2. Similarly, changes in f_1 and f_3 can be observed across Cases 1, 2, and 3, while variations in f_2 and f_3 are evident in Cases 1, 2, and 4. These results demonstrate that optimizing one objective often leads to unpredictable trade-offs with others, highlighting the inherent conflicts in multi-objective optimization.

3.5. Validation of MILP model

To validate the MILP model for the DRCFJSP-TDST problem, IBM ILOG CPLEX, Version 22.1.1.0, is utilized, interfaced via its Java API with default configurations. For this purpose, five very small instances, labeled N01–N05, are constructed. The methodology for constructing these instances and their download links are detailed in Section 5.1.

As shown in Table 4, the successful execution of instances N01–N03 confirms the correctness of the model, demonstrating its capability to produce results under minimal scale scenarios. However, instances N04 and N05 do not complete within four hours which is equivalent to half a workday for a planner, indicating that solving larger instances using CPLEX is computationally infeasible within a reasonable timeframe. This highlights the limitations of the CPLEX approach when applied to more complex problem sizes.

4. The proposed IGA

4.1. Framework of IGA

To obtain high-quality non-dominated solutions for the proposed multi-objective DRCFJSP-TDST, an improved genetic algorithm (IGA) is designed. This IGA integrates the strengths of NSGA-II and a problem-focused neighborhood search operator. The core steps are outlined below.

Step 1: initialization. Set $t = 0$, initialize core parameters, including population size (N), crossover probability (P_c), mutation probability (P_m), and neighborhood search probability (P_n). Then, encode chromosome and generate an initial population P_t with N individuals. Next, randomize operation sequence, select resources, and decode to update population P_t. Finally, initialize result set $R = \Phi$.

Step 2: crossover. Random select a pair of individuals from current population P_t, use the crossover operator to create new offspring. After repeating this process $N \cdot P_c$ times, continue by selecting resources, decoding, and applying the elite strategy to finally obtain a new population C_t. Next, save Pareto front (PF) of C_t and R to R.

Step 3: mutation. Randomly select $N \cdot P_m$ individuals from the current population C_t, apply the mutation operator to generate new individuals. Then, select resources, decode and adopt elite strategy to obtain a new population M_t. Next, save PF of M_t and R to R.

Step 4: combination. Combine the population P_t with M_t to form a new population Q_t. Then, carry out the non-dominated sorting and crowding distance computing for Q_t to obtain the population Q'_t.

Step 5: neighborhood. Determine critical path and critical block in the population Q'_t. Then, utilize the neighborhood rule for $N \cdot P_n$ individuals of the population Q'_t. Next, perform elite strategy to obtain P_{t+1}. Finally, save PF of P_{t+1} and R to R.

Step 6: stop criteria. if the stopping criteria is met, output R; else set $t = t + 1$, turn to **Step 2**.

Based on the above description, the framework of the proposed IGA is illustrated in Fig. 2. The detailed steps of the algorithm are elaborated in the following section.

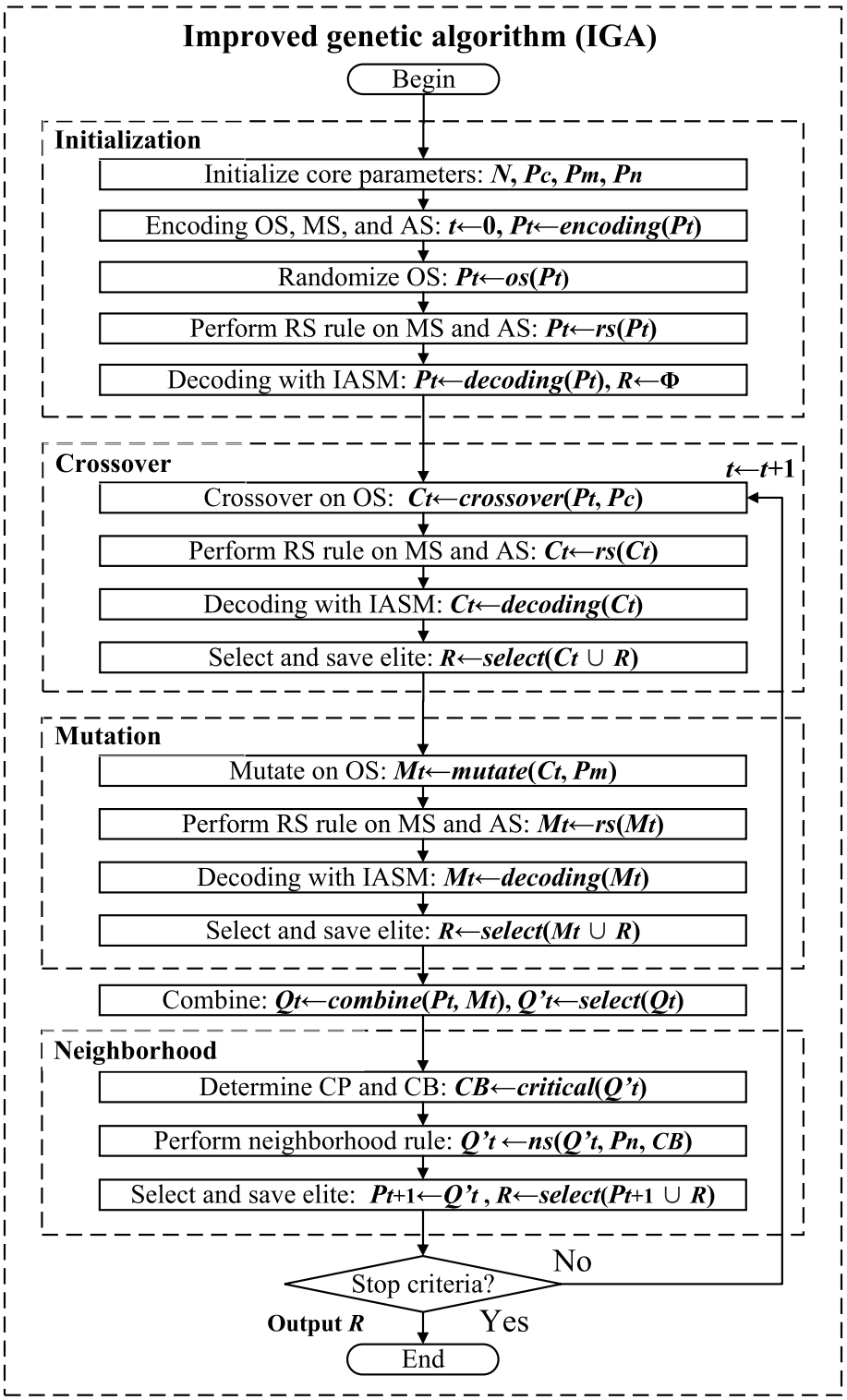

Fig. 2. Flowchart of the proposed IGA.

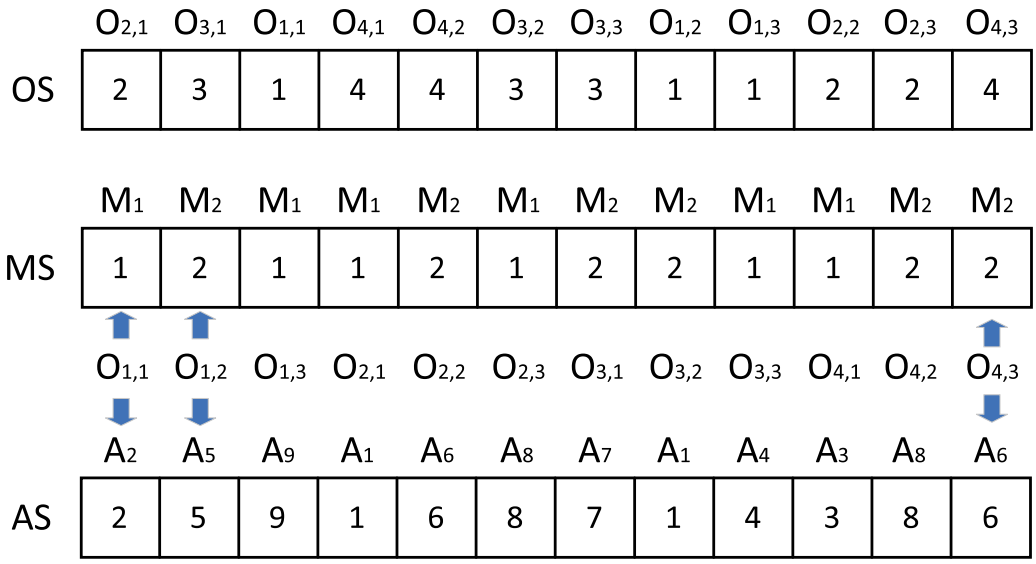

Fig. 3. A three-vector chromosome encoding.

4.2. Initialization

In the initialization phase, the first step is to initialize the core parameters, N, P_c, P_m, and P_n. Then, a three-vector chromosome encoding method, comprising the operation sequence (OS), machine selection (MS), tool selection (AS), is designed to represent a scheduling solution. The OS vector consists of serial numbers of jobs, which indicates the operations to be scheduled by the numbers of appearance. The MS and AS vectors include information for both machine and tool selection. To ensure clarity, Fig. 3 provides an example using the parameters listed in Table 2. In this case, the OS vector represents four jobs and twelve operations, and a number i appearing in the jth time maps to the operation $O_{i,j}$. The MS and AS vectors are arranged by operation number, i.e., the operation $O_{1,1}$ is assigned to machine M_1 with tool A_2, the operation $O_{1,2}$ is assigned to machine M_2 with tool A_5, and the operation $O_{4,3}$ is assigned to machine M_2 with tool A_6.

For the initial population, the OS vector is determined by a random approach, while the MS and AS assignments for each operation are produced by different heuristic resource selection (RS) rules.

Rule 1: the random rule (RND). The machine and tool assignments for each operation are generated randomly. Taking $O_{1,1}$ as an example from Table 2, the operation can randomly chooses one of the six possible combinations of machines $\{M_1, M_2\}$ and tools $\{A_1, A_2, A_3\}$.

Rule 2: the shortest processing time rule (SPT). The machine with the shortest processing time is chosen for each operation, and then the tool is selected randomly. For example, as shown in Table 2, operation $O_{1,1}$ takes 10 to process on machine M_1 and 12 on machine M_2. Therefore, M_1 is selected, and any one of the tools in $\{A_1, A_2, A_3\}$ can be chosen randomly.

Rule 3: the minimum cumulative machine load rule (CML). The machines with minimum cumulative workload are selected for all operations. Tool selected for each machine are generated randomly. As shown in Fig. 1, before determining the machine and tool for $O_{1,1}$, operations $O_{2,1}$ and $O_{3,1}$ have already selected M_1 with A_1 and M_2 with A_7, respectively. The current workloads for M_1 and M_2 are 10 and 15, respectively. Therefore, to ensure the minimum cumulative load of all machines, the operation $O_{1,1}$ selects M_1 due to its lower cumulative load, and then randomly chooses one tool from $\{A_1, A_2, A_3\}$.

Rule 4: the hybrid cumulative load rule (HCL). The rule for selecting machine is the same as CML, and when selecting tool, an equal probability strategy is used to choose between two tool determination rules: (1) choose any tool from the available tool set, and (2) choose the tool with minimum the cumulative tool load. For example, if the minimizing the cumulative tool load rule is used, the operation $O_{1,1}$ will choose a tool between A_2 and A_3, because A_1 already has an operation load $O_{2,1}$.

For chromosome decoding, the active scheduling method is popular, and it has been proved to be effective in the FJSP community (Luo et al., 2020). Its core idea is that the operations are processed as early as possible by using the earlier idle time of machines (Luo et al., 2022). Inspired by this, this paper proposes an improved active scheduling method (IASM) to decode all solutions by considering TDST. The IASM algorithm will start searching for available time intervals in the future from the earliest idle time of the machine, requiring both the machine and the tool to meet the constraints of the end time of the previous operation, setup time, and processing time. To provide a clearer description of the IASM decoding method, three different decoding cases are presented below.

Case 1: head, e.g., $O_{2,1}$ and $O_{3,1}$ in Fig. 1. When decoding these two operations, both machines and tools are idle. Considering the initial setup time and processing time, these two operations are scheduled at the beginning of their respective machine and tool. As shown in Fig. 1, for operation $O_{2,1}$ which assigned to machine M_1 with tool A_1, the setup time ranges from 0 to 2, and the processing time ranges from 2 to 12.

Case 2: tail, e.g., $O_{1,1}$, $O_{4,1}$, $O_{4,2}$, $O_{3,3}$, $O_{1,2}$, $O_{1,3}$, $O_{2,2}$, $O_{2,3}$, and $O_{4,3}$ in Fig. 1. When decoding these operations, the idle time at the end of the machine and the tool can meet the requirements of the scheduling at the same time. Considering the end time of previous operation on the same job, as well as the end time and tool type of the previous operation on the same machine, these operations will be arranged at the end of their respective machines and tools in sequence. As shown in Fig. 1, the start time of $O_{1,3}$ which assigned to machine M_1 with tool A_9, selects the larger value 60 between the end time 60 of $O_{1,2}$ and the sum 52 of the end time 49 of $O_{3,3}$ and the setup time 3 of $O_{1,3}$, where $O_{1,2}$ is the previous operation of $O_{1,3}$ in the same job J_1, and $O_{3,3}$ is the previously assigned same machine M_1 with different tool type of $O_{1,3}$.

Case 3: middle, e.g., $O_{3,2}$ in Fig. 1. This operation will be assigned to machine M_2 with tool A_1. Before arranging this operation, there is an idle time between two operations $O_{3,1}$ and $O_{4,2}$, and the length of idle time 17 is greater than the sum of the setup time of $O_{3,2}$, processing time of the operation $O_{3,2}$, and the setup time of $O_{4,2}$. Under other constraints like Case 2, the operation $O_{3,2}$ is inserted between operations $O_{3,1}$ and $O_{4,2}$.

Therefore, this paper designs an initialization phase based on above strategy, and the main steps are as follows.

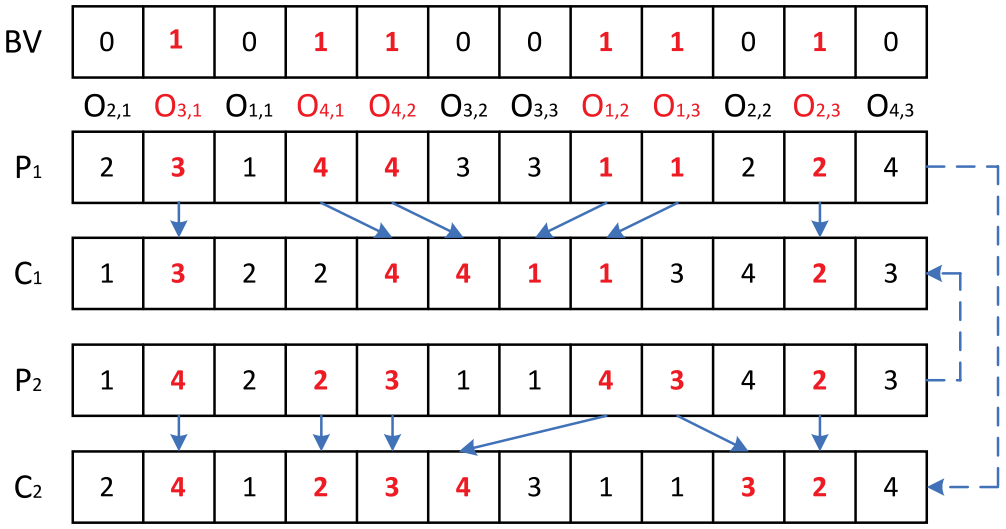

Fig. 4. MOX crossover on OS vector.

Step 1: initialize core parameters. N, P_c, P_m, and P_n.
Step 2: encoding OS, MS, and AS. $t \leftarrow 0$, $P_t \leftarrow encoding(P_t)$.
Step 3: randomize OS. $P_t \leftarrow os(P_t)$.
Step 4: perform RS rule on MS and AS. $P_t \leftarrow rs(P_t)$.
Step 5: decoding with IASM. $P_t \leftarrow decoding(P_t)$, $R \leftarrow \Phi$.

4.3. Crossover

The crossover operator is typically used to generate the offspring solution from the parent solution (Yuan, Li, Zhang, & Pei, 2021). In each parent, there are three vectors: OS, MS, and AS. For the OS vector, crossover is employed to generate the next generation. However, for the MS and AS vectors, updates are still performed using the heuristic RS rule. Given that two parents, P_1 and P_2, have been selected, if a randomly generated number from the interval (0, 1) is less than the crossover probability P_c, crossover operator will be performed on these two parents. The offspring C_1 and C_2 is generated by the Multiple Operation Crossover (MOX) (Luo et al., 2022) method. Taking Fig. 4 as an example, the offspring can be acquired by the following steps.

Step 1: generate a binary vector (BV) of a size equal to the total number of operations randomly, and then determine the crossover points. In this case, the crossover positions marked with 1 are located at indices 2, 4, 5, 8, 9 and 11.

Step 2: initialize offspring C_1 and C_2 by copying P_1 to C_2, and P_2 to C_1, respectively.

Step 3: identify all information at the crossover points in P_1, which are $O_{3,1}$, $O_{4,1}$, $O_{4,2}$, $O_{1,2}$, $O_{1,3}$, and $O_{2,3}$ according to BV. Similarly, identify all information at the crossover points in P_2, which are $O_{4,1}$, $O_{2,2}$, $O_{3,1}$, $O_{4,2}$, $O_{3,2}$, and $O_{2,3}$.

Step 4: locate the operations $O_{3,1}$, $O_{4,1}$, $O_{4,2}$, $O_{1,2}$, $O_{1,3}$, and $O_{2,3}$ in C_1, corresponding to the positions 2, 5, 6, 7, 8, and 11 in C_1. Similarly, locat the operations $O_{4,1}$, $O_{2,2}$, $O_{3,1}$, $O_{4,2}$, $O_{3,2}$, and $O_{2,3}$ in C_2, corresponding to the positions 2, 4, 5, 6, 10, and 11 in C_2.

Step 5: copy the operations $O_{3,1}$, $O_{4,1}$, $O_{4,2}$, $O_{1,2}$, $O_{1,3}$, and $O_{2,3}$ in P_1 to the positions 2, 5, 6, 7, 8, and 11 of C_1 from left to right. Similarly, copy the operations $O_{4,1}$, $O_{2,2}$, $O_{3,1}$, $O_{4,2}$, $O_{3,2}$, and $O_{2,3}$ in P_2 to the positions 2, 4, 5, 6, 10, and 11 of C_2 from left to right.

When generating the next generation population through crossover, the conventional NSGA-II based method employs an elite selection and retention strategy. This involves performing non-dominated sorting and applying the crowding distance operator to the offspring populations to prevent the loss of PF solutions during the evolutionary process, thereby significantly accelerating convergence. However, this approach can lead to a considerable deviation of the solution set from the true PF. To address these issues, an external archive set, namely the result set R in Section 4.1, as proposed by Wang, Ge, Miao, Jiang, and Shen (2023), is utilized to store the PF after each population variation. The external archive set is updated with the change in the PF and does not participate in population evolution. Finally, the external archive set is outputted at the end.

Thus, the detailed steps of crossover are described as follows.
Step 1: crossover on OS. $C_t \leftarrow crossover(P_t, P_c)$.
Step 2: perform RS rule on MS and AS. $C_t \leftarrow rs(C_t)$.
Step 3: decoding with IASM. $C_t \leftarrow decoding(C_t)$.
Step 4: select and save elite. $R \leftarrow select(C_t \cup R)$.

4.4. Mutation

The mutation operator can increase the diversity of the population, reduce the probability of premature convergence of the algorithm, and enhance the local search ability of the algorithm (Sun et al., 2023). Similar to crossover, only OS vector performs the mutation operator, whereas the MS and AS vectors still rely on heuristic RS rule. Under the condition of the mutation probability P_m, the mutation operator is executed on the OS vector to produce one offspring C_1 from one parent P_1 at a time. The mutation on the OS segment is performed by reversing the adjacent genes of a random position. In Fig. 8, the random gene is 3 located in the position 9 of P_1, and the child C_1 is generated by reversing the genes 1, 3 and 4.

After completing the mutation operator on the OS vector, it is essential to adjust the machine and auxiliary resources to update the MS and AS vectors accordingly. Following this adjustment, the next steps involve decoding, selecting, and saving the elite solutions. Therefore, the detailed steps of the mutation operator are outlined as follows.

Step 1: mutate on OS. $M_t \leftarrow mutate(C_t, P_m)$.
Step 2: perform RS rule on MS and AS. $M_t \leftarrow rs(M_t)$.
Step 3: decoding with IASM. $M_t \leftarrow decoding(M_t)$.
Step 4: select and save elite. $R \leftarrow select(M_t \cup R)$.

4.5. Neighborhood

Neighborhood search is necessary and effective to improve the local search ability of the GA, and its application could be found widely in multiple shop scheduling problems (Li, Chen et al., 2023; Shi & Xiong,

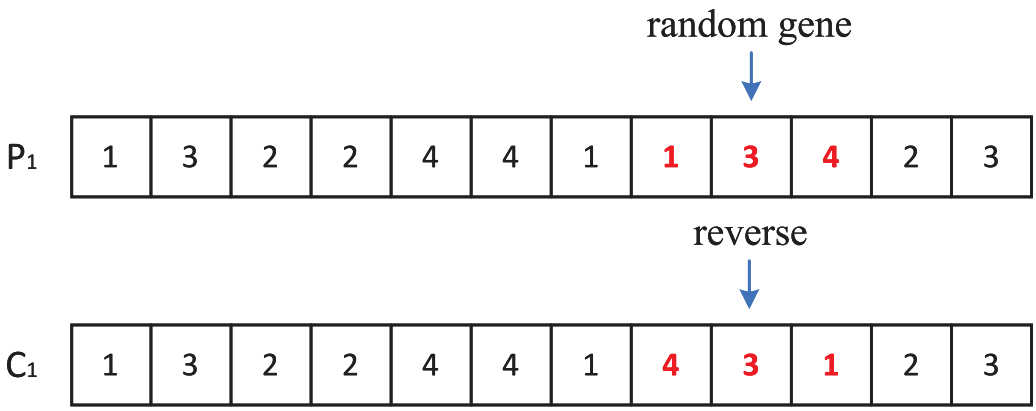

Fig. 5. Mutation on OS vector.

2024; Zhang, Zheng, & Ahmad, 2023). According to Fan, Shen, Gao, Zhang, and Zhang (2021), strategies based on the critical path (CP) and critical block (CB) are particularly effective for solving FJSP. The critical path is identified by determining the longest sequence of operations from the start time of 0 to the maximum ending time, ensuring there are no gaps between the operations. Similarly, the critical block focuses on contiguous operations on the same machine within the critical path, which are often bottlenecks in scheduling. By targeting these critical structures, neighborhood search techniques can significantly enhance solution quality. For instance, reducing the critical path or optimizing operations within critical blocks helps concentrate on the most impactful operations, directly improving the overall completion time. This approach not only facilitates more efficient resource allocation but also potentially reduces the makespan. Thus, leveraging CP and CB concepts in neighborhood search provides a focused and effective mechanism for refining solutions in FJSP.

As shown in Fig. 5, the critical path can be determined as ($O_{2,3}$, $O_{1,3}$, $O_{1,2}$, $O_{4,2}$, $O_{4,1}$, $O_{1,1}$, $O_{2,1}$) from the end to start. Each operation on the critical path is considered a critical operation, and continuous critical operations on the same machine are defined as a critical block. In this case, the critical blocks are ($O_{2,1}$, $O_{1,1}$, $O_{4,1}$), ($O_{4,2}$, $O_{1,2}$), and ($O_{1,3}$, $O_{2,3}$). This paper introduces four neighborhood search rules, which based on the critical path or critical block, and they are abbreviated as N1 (Luo et al., 2022), N2, N3, and N4.

N1: insert the operations within the critical block. Suppose a critical block ($O_{2,1}$, $O_{1,1}$, $O_{4,1}$) is selected from all critical blocks. The first operation $O_{2,1}$ is sequentially inserted from right to left. The insertion process stops if a new, improved solution is generated during this stage. If no better solution is found, the last operation $O_{4,1}$ is then sequentially inserted from left to right until either a better individual is generated or the maximum insertion position is reached.

N2: adjust auxiliary resource on the adjacent operations in critical path. Suppose two adjacent operations in critical path have same machine and different tools of same type. Then, change the tool from the next operation to the tool from the previous operation. Taking Fig. 6 as an example, two adjacent operations $O_{1,3}$ and $O_{2,3}$ in the critical path, set tool A_9 to $O_{2,3}$. This rule not only reduces the setup time between adjacent operations, but also shortens the makespan of critical path.

N3: combine the N1 and N2 rules according to sequential conditions. First, adopt the N1 rule. if none of better solution is produced for the critical path of one individual after the N1 rule, then adopt the N2 rule to this critical path. Next, the N3 rule is terminated if a new, better solution is generated during the traversal process of operations in the critical path using the N2 rule, or if the traversal process reaches its end.

N4: select the N1 and N2 rules according to equal probability strategy. For each population, either N1 or N2 is applied with equal likelihood, ensuring that their usage ratio remains balanced. Regardless of which rule is selected, if a better result is obtained during the process, this result is saved, and the process is immediately terminated.

In fact, different neighborhood search rules are used to adjust operations and resources within a local scope, so the results after each local adjustment need to be decoded to determine whether they produce better results. The final result will be selected based on the elite strategy. Thus, the detailed steps of neighborhood search operator are described as follows.

Step 1: determine the CP and CB. $CB \leftarrow critical(Q'_t)$.

Step 2: perform neighborhood rule. $Q'_t \leftarrow ns(Q'_t, P_n, CB)$.

Step 3: select and save elite. $P_{t+1} \leftarrow Q'_t$, $R \leftarrow select(P_{t+1} \cup R)$.

4.6. Computational complexity

The proposed IGA encompasses four primary stages: initialization, crossover, mutation, and neighborhood search. Among these, the latter three are iterative processes that significantly contribute to the computational complexity of the algorithm. Let N_d denotes the objective function, which dictates the computational complexity associated with decoding. Let l represents the number of iterations executed, and N_{cp} be the length of the critical path. The complexity involved in resource selection is considerably lower compared to decoding and can thus be disregarded in this analysis. The complexities for each of the main operations are as follows: (1) crossover $O(P_c N N_d + N^2)$ (2) mutation $O(P_m N N_d + N^2)$ (3) neighborhood $O(P_n N N_{cp} N_d + N^2)$, where $O(N^2)$ represents the computational complexity of select and save elite step. Given these individual contributions, the overall computational complexity of the proposed IGA algorithm can be expressed as $O(max\{lP_c N N_d + lN^2, lP_m N N_d + lN^2, lP_n N N_{cp} N_d + lN^2\})$.

5. Comparison experiments and discussions

A series of experiments are conducted to evaluate the performance of the IGA algorithm, including comparisons with four resource selection rules, four neighborhood search rules, and benchmarking against algorithms such as IMA (Luo et al., 2022), PTEA (Luo et al., 2023), and NSGA-II. The proposed IGA algorithm is encoded in Java 1.8 and run on a PC with Apple M1 Pro CPU and 32G RAM. For a fair comparison, the CPU time criterion $50 \times n \times k \times N_{cmax}$ ms are adopted for each algorithm. To avoid randomness, the final non-dominated solution set is composed of all solutions obtained after running each comparison algorithm 30 times. This aggregated set ensures that the results are robust and not influenced by random variations.

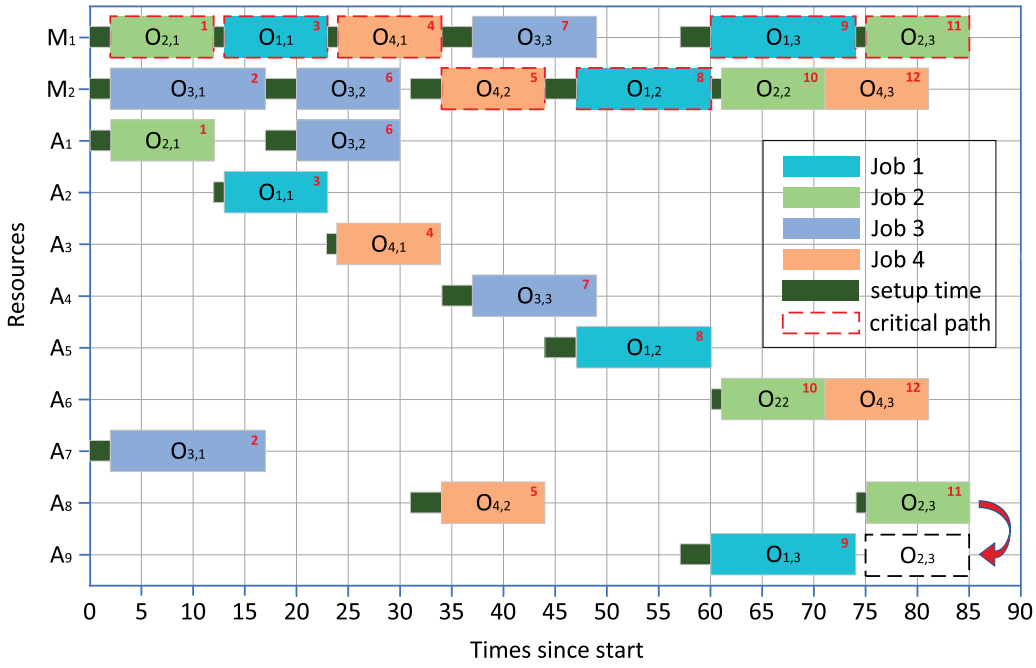

Fig. 6. Critical path and neighborhood rule.

5.1. Benchmarks construction

No standard benchmark instances are available for DRCFJSP-TDST in the literature. Therefore, to test the effectiveness of the algorithm, sixty instances D01–D60 are listed in Table 5 which extended from DRCFJSP examples DW16–DW35 and F19–F58 proposed by Luo et al. (2022, 2023), respectively. The extended content includes available tool types and tool set for each operation, as well as the setup times required for switching tools on the same machine. They are named as $n\times N_{imax}\times k\times N_{cmax}\times v\times u_T\times u_{ST}$, where u_T is the expectation of $T_{i,j,m}$, and u_{ST} is the expectation of $ST_{m,a',a}$. The instances D01–D25 are classified as small-size examples, and the remaining examples are recorded as large-scale instances. Processing time $T_{i,j,m}$ is generated according to a Gaussian distribution whose expectations is u_T and standard deviation variable is $0.15\times u_T$, $N(u_T,(0.15\times u_T)^2)$. Setup time $ST_{m,a',a}$, if $TP_{a'}=TP_a$ and $a'\neq a$ is generated by Gaussian distribution $N(u_{ST},(0.15\times u_{ST})^2)$. Setup time $ST_{m,a'',a}$, if $TP_{a''}\neq TP_a$ is generated by Gaussian distribution $N(2\times u_{ST},(0.15\times 2\times u_{ST})^2)$. The initial setup time generated by Gaussian distribution $N(1.5\times u_{ST},(0.15\times 1.5\times u_{ST})^2)$. The constants in the setup time expression mentioned above, i.e., 0.15, refer to setup time value in Barak et al. (2024), Fan et al. (2022) and Tri Windras Mara, Sutoyo, Norcahyo, and Pratama Rifai (2023). In each instance, the operation can select any machine, and the machine is randomly associated with a tool type. Any tool belonging to this specific tool type is considered an available tool. Additionally, the instances N01–N05 used in Section 3.5 are also generated using the same method. Detailed data used in this study can be downloaded from https://github.com/walkcap/IGA.

5.2. Evaluation index

The selection of Hypervolume (HV), Generational Distance (GD), and Inverted Generational Distance (IGD) as evaluation metrics is grounded in their complementary capabilities to quantify both convergence and diversity of Pareto-optimal solutions, which aligns with the tri-objective nature of the DRCFJSP-TDST problem. While HV provides a global quality measure of the Pareto front, GD offers insights into the proximity of the solutions to the ideal Pareto front, focusing on convergence. In contrast, IGD not only evaluates the proximity but also penalizes gaps in the coverage of the Pareto front, ensuring both convergence and diversity. This triad addresses limitations of single-metric evaluations, as demonstrated in recent multi-objective FJSP studies (Wei et al., 2024).

(1) HV

The HV is a comprehensive indicator that represents the convergence and distribution of the solution set. The larger the value of HV, the better quality of the solution set. It can be formulated as follows.

$$HV(PF,P)=\cup_{x\in PF}volume(x,P) \tag{20}$$

where $volume(x,P)$ is the hypervolume of the xth solution in obtained Pareto front PF and the reference point set P.

(2) GD

GD is the mean distance from obtained PF to the optimal Pareto front PF^*, indicating the convergence performance of the PF. The calculation of the GD metric is shown in formula (21).

$$GD(PF,PF^*)=\frac{\sqrt{\sum_{i=1}^{|PF|}distance^2(i,PF^*)}}{|PF|} \tag{21}$$

where $distance^2(i,PF^*)$ is the shortest Euclidean distance between the ith solution of PF and PF^*. $|PF|$ is the number of solutions in the PF. A smaller GD indicates better algorithm convergence performance.

(3) IGD

The IGD represents the average distance from PF^* to the obtained PF. A smaller IGD indicates that the obtained PF is closer to PF^*, which reflects better algorithm performance. It can be formulated as follows.

$$IGD(PF,PF^*)=\frac{\sqrt{\sum_{i=1}^{|PF^*|}distance^2(i,PF)}}{|PF^*|} \tag{22}$$

where $distance^2(i,PF)$ is the shortest Euclidean distance between the ith solution of PF^* and PF. $|PF^*|$ is the number of solutions in the PF^*.

It should be noted that the PF^* of the solved instances may be unknown. In such cases, PF^* is approximated by the Pareto front of all solutions obtained by all algorithms involved in the evaluation. Furthermore, to ensure the accuracy of the result, the objective values are normalized using Eq. (23).

$$\widetilde{f}(x)=\frac{f_i(x)-f_i^{min}}{f_i^{max}-f_i^{min}},i=1,2,3 \tag{23}$$

Table 5
Benchmarks scales.

Ins.	n	N_{imax}	k	N_{cmax}	v	u_T	u_{ST}
D01	10	5	6	5	5	50	5
D02	10	5	6	5	5	50	5
D03	10	5	6	5	5	50	5
D04	10	5	6	5	5	25	5
D05	10	5	6	5	5	25	5
D06	15	5	6	5	5	50	5
D07	15	5	6	5	5	50	5
D08	15	5	6	5	5	50	5
D09	15	5	6	5	5	25	5
D10	15	5	6	5	5	25	5
D11	20	5	6	5	5	50	5
D12	20	5	6	5	5	50	5
D13	20	5	6	5	5	50	5
D14	20	5	6	5	5	25	5
D15	20	5	6	5	5	25	5
D16	10	10	6	5	10	50	5
D17	10	10	6	5	10	50	5
D18	10	10	6	5	10	50	5
D19	10	10	6	5	10	25	5
D20	10	10	6	5	10	25	5
D21	10	10	6	10	10	50	5
D22	10	10	6	10	10	50	5
D23	10	10	6	10	10	50	5
D24	10	10	6	10	10	25	5
D25	10	10	6	10	10	25	5
D26	15	10	10	9	10	50	5
D27	15	10	10	9	10	50	5
D28	15	10	10	9	10	50	5
D29	15	10	10	9	10	25	5
D30	15	10	10	9	10	25	5
D31	20	10	10	9	10	50	5
D32	20	10	10	9	10	50	5
D33	20	10	10	9	10	50	5
D34	20	10	10	9	10	25	5
D35	20	10	10	9	10	25	5
D36	30	10	10	9	10	50	5
D37	30	10	10	9	10	50	5
D38	30	10	10	9	10	50	5
D39	30	10	10	9	10	25	5
D40	30	10	10	9	10	25	5
D41	15	15	10	9	15	50	5
D42	15	15	10	9	15	50	5
D43	15	15	10	9	15	50	5
D44	15	15	10	9	15	25	5
D45	15	15	10	9	15	25	5
D46	15	10	8	20	10	50	5
D47	15	10	8	20	10	50	5
D48	15	10	8	20	10	50	5
D49	15	10	8	20	10	25	5
D50	15	10	8	20	10	25	5
D51	20	10	8	20	10	50	5
D52	20	10	8	20	10	50	5
D53	20	10	8	20	10	50	5
D54	20	10	8	20	10	25	5
D55	20	10	8	20	10	25	5
D56	30	10	8	20	10	50	5
D57	30	10	8	20	10	50	5
D58	30	10	8	20	10	50	5
D59	30	10	8	20	10	25	5
D60	30	10	8	20	10	25	5

where f_i^{max} and f_i^{min} are the maximum and minimum values of the ith objective $f_i(x)$ obtained by all algorithms involved in the comparison. Normalization facilitates the comparison of objective values on different scales by converting them into a unified measurement standard, thereby enhancing the effectiveness of the optimization process.

5.3. Parameters setting

The proposed IGA contains four key parameters, involving population size N, crossover rate P_c, mutation rate P_m, neighborhood search rate P_n. Taguchi analysis (Luo et al., 2023) is performed to acquire the best combination of these parameters, where five small-scale instances and five large-size instances in D01–D60 are selected randomly for experiments. Specifically, each factor includes four levels: $N \in \{100, 150, 200, 250\}$, $P_c \in \{0.6, 0.7, 0.8, 0.9\}$, $P_m \in \{0.1, 0.2, 0.3, 0.4\}$, $P_n \in \{0.1, 0.2, 0.3, 0.4\}$. A total of L16(4 ^ 4) combinations are formulated in Minitab 22.1 version. Under the CPU time criterion, the average IGD values from these calculations are chosen to evaluate the factor levels.

Table 6
The IGD results of parameters.

ID	N	P_c	P_m	P_n	IGD
1	1	1	1	1	0.0292148
2	1	2	2	2	0.0306835
3	1	3	3	3	0.0296086
4	1	4	4	4	0.0303213
5	2	1	2	3	0.0324703
6	2	2	1	4	0.0344656
7	2	3	4	1	0.0295726
8	2	4	3	2	0.0284472
9	3	1	3	4	0.0357191
10	3	2	4	3	0.0330443
11	3	3	1	2	0.0327268
12	3	4	2	1	0.0314866
13	4	1	4	2	0.0345899
14	4	2	3	1	0.0304517
15	4	3	2	4	0.0355087
16	4	4	1	3	0.0345891

According to the Taguchi analysis, the IGD results and the main effect of core parameters in this experiment are listed in Table 6 and Fig. 7. Each row in Table 6 represents a combination of algorithm parameters and the corresponding IGD results. In Fig. 7, the x-coordinate represents different levels of key parameters and y-coordinate denotes the average value of IGD. From these results, the best combination of $N = 100$, $P_c = 0.9$, $P_m = 0.3$, and $P_n = 0.1$ can be derived.

5.4. Comparison between the proposed IGA and its variants

5.4.1. Comparison of resource selection rules

In Section 4.2, four heuristic resource selection rules, RND, SPT, CML, and HCL are proposed to select machine and tool. The rules for resource selection are used in the initialization, crossover, mutation, and neighborhood stages, which play an important role in the efficiency of the algorithm. Therefore, it is necessary to evaluate the effectiveness of the algorithm under different rules. Here, IGA sequentially uses a single rule RND, SPT, CML, and HCL, named IGA1, IGA2, IGA3, and IGA4, respectively. To validate their performance, sixty instances D01–D60 are computed for each algorithm under the same CPU time criterion and run times. To be fair, except for resource selection rule, the parameter combination in initialization, crossover, mutation, and neighborhood search operator are identical in these four algorithms. The corresponding results based on HV, GD, and IGD metrics are listed in Table 7, where the best performance in each instance is shown in bold.

As shown in Table 7, for the 25 small-scale instances (D01–D25), the number of maximum HV values for the four resource selection variants are [0, 1, 3, 21], respectively. This indicates that IGA3 achieves the best diversity, convergence, and spread performance. The number of minimum GD and IGD values are [0, 2, 7, 16] and [0, 0, 5, 20], respectively, indicating that the solutions generated by IGA4 are closest to the Pareto optimal set. For the 35 large-scale instances (D26–D60), the situation is similar, with the maximum HV, minimum GD, and minimum IGD values being [0, 0, 14, 21], [0, 3, 12, 20], and [0, 0, 13, 22], respectively. To facilitate a clearer comparison of numerical values, Fig. 9 presents the total number of dominant values in the form of a histogram. Based on these results, it can be concluded that the IGA with the HCL resource selection rule is the most effective among the four resource selection rules.

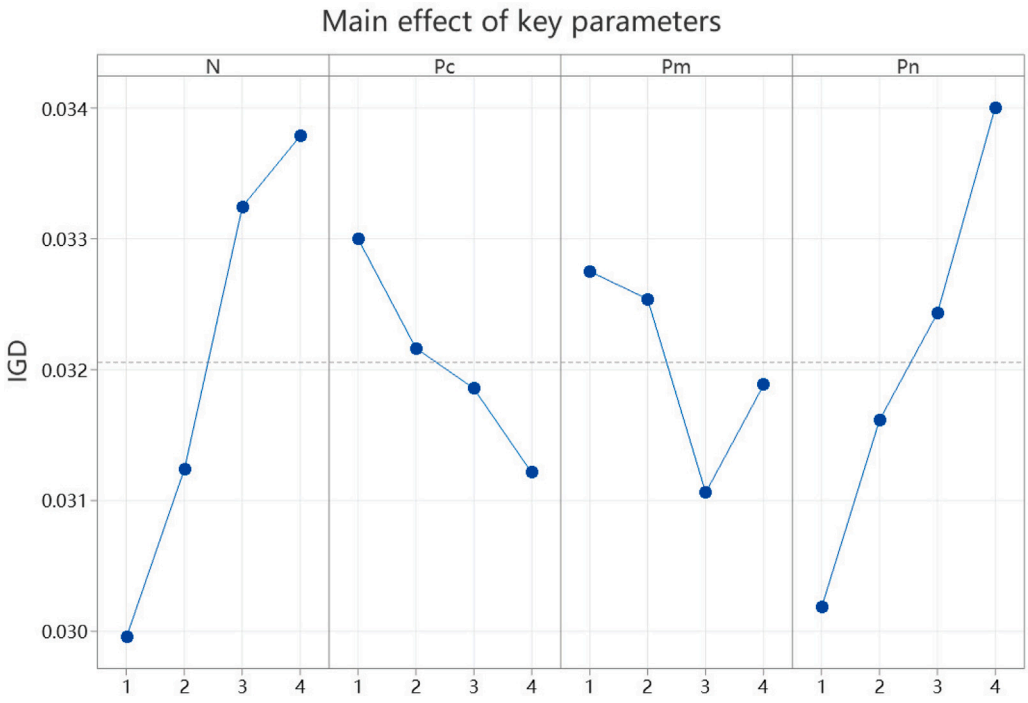

Fig. 7. Main effect of key parameters.

For the evaluation experiment results of these instances, if IGA3 and IGA4 are compared separately with IGA1 and IGA2, both IGA3 and IGA4 are significantly better than IGA1 and IGA2, because IGA3 and IGA4 consider machine load balancing in resource selection, which is directly related to the objective f_2, the maximum the workload of machines. Although both IGA3 and IGA4 use machine load balancing and random selection tools, IGA4 also utilizes tool load balancing to further improve the efficiency of the algorithm.

5.4.2. Comparison of neighborhood search rules

To verify the effectiveness of the neighborhood search operator, the IGA is compared with five variants, including IGA with N1 (IGA_N1), IGA with N2 (IGA_N2), IGA with N3 (IGA_N3), IGA with N4 (IGA_N4), and IGA without neighborhood operator (IGA_NA). In the comparative experiments, the parameter combination, the encoding strategy, resource selection rule, decoding method, evolutionary operators, elite selection and retain solution method remain the same in all tested algorithms, except for the neighborhood search rule. A total of sixty examples D01–D60 are scheduled by these five algorithms under the same CPU time criterion and run times, and the comparison results based on HV, GD, and IGD indicators are collated in Tables 8 and 9.

In Table 8, it can be seen that IGA_N3 produces the best HV in most instances. To be specific, in sixty examples, IGA_N1, IGA_N2, IGA_N3, IGA_N4, and IGA_NA win 16, 5, 34, 5, and 0 times, respectively. This result means IGA_N3 has a higher convergence rate. As shown in Table 9, IGA_N3 also obtains the best results of GD and IGD in most cases. For GD, the IGA_N1, IGA_N2, IGA_N3, IGA_N4, and IGA_NA produce the best results 11, 15, 30, 13, and 0 times, respectively. But no optimal result is generated by IGA_NA among all instances. The result on GD demonstrates that IGA_N3 with the neighborhood search operator obtains the best convergence and spread performance. As for IGD values, IGA_N3 wins 26 times in 60 examples and exhibits the best results among the five algorithms. The IGD results imply that the non-dominated solutions of IGA with N3 rule are closer to the overall Pareto optimal solutions. The above results means that the neighborhood methods are effective to generate the non-dominated solutions because the IGA_N1, IGA_N2, IGA_N3, and IGA_N4 are better than IGA_NA. Meanwhile, the IGA_N3 with the neighborhood search operator is better than the IGA_N1 and IGA_N2 with a single rule since the IGA_N3 wins more times in the experiments. To facilitate a clearer comparison of numerical values, Fig. 10 presents the total number of dominant values in the form of a histogram.

Both N3 and N4 rules integrate the advantages of N1 and N2 rules. It was hypothesized that these combined strategies would outperform any single strategy. The experimental results confirm this hypothesis, showing that N3 and N4 are indeed superior to N1 and N2, respectively. Moreover, the N3 rule leverages the complementary strengths of N2 to enhance N1, while the N4 rule alternates between N1 and N2. Experimental results demonstrate that N3 significantly outperforms N4. According to the analysis above, it can be conclude that the neighborhood search operator with N3 rule is effective.

5.4.3. Time cost of neighborhood search

To improve the effectiveness of the algorithm, IGA incorporates additional time overhead. In this experiment, the number of iterations is fixed at 100, with Step 2 through 6 constituting a single iteration cycle. By applying different neighbor rules or none, the execution times are measured across instances D01 to D60. The results are presented in Fig. 8.

As shown in Fig. 8, the x-axis represents the index of instances from D01 to D60, labeled sequentially from 1 to 60, while the y-axis indicates the execution time in milliseconds. Overall, execution time increases with sample size, particularly for D36–D40 and D56–D60, where the job count reaches 30, resulting in notably longer execution times compared to other cases. The observed execution times follow the order, IGA_NA < IGA_N2 < IGA_N4 < IGA_N1 < IGA_N3. This outcome is expected because IGA_NA, which does not utilize any neighborhood operator, incurs the least overhead. IGA_N2 consumes relatively little time, closely matching that of IGA_NA, with additional computational overhead incurred only when needed to reduce setup times within critical blocks. In contrast, IGA_N1 requires forward and backward insertions into critical blocks, leading to significantly higher execution times. IGA_N4, which selects the N1 and N2 rules according to equal probability strategy, has an execution time that falls between those of IGA_N1 and IGA_N2. Finally, IGA_N3, designed as a compensatory combination, inevitably surpasses both N1 and N2 in terms of time

Table 7
Comparison results of resource selection rules on HV, GD, and IGD.

Ins	HV				GD				IGD			
	IGA1	IGA2	IGA3	IGA4	IGA1	IGA2	IGA3	IGA4	IGA1	IGA2	IGA3	IGA4
D01	0.0000	0.0787	0.6066	**0.6444**	0.2288	0.0322	0.0154	**0.0101**	0.1681	0.0853	**0.0114**	0.0161
D02	0.1787	0.0000	0.6235	**0.6506**	0.0653	0.0255	**0.0078**	0.0080	0.0613	0.1286	0.0279	**0.0265**
D03	0.1911	0.0000	0.7059	**0.7290**	0.0561	0.0393	0.0115	**0.0074**	0.0729	0.1528	0.0329	**0.0299**
D04	0.3741	0.1546	0.7646	**0.8533**	0.0732	0.1950	0.0096	**0.0070**	0.0613	0.1043	0.0259	**0.0234**
D05	0.1001	0.0000	0.7379	**0.7412**	0.0775	0.0232	0.0132	**0.0085**	0.0970	0.1349	0.0357	**0.0310**
D06	0.0176	0.0000	0.6401	**0.6674**	0.1222	0.0140	0.0125	**0.0112**	0.1052	0.1168	**0.0218**	0.0220
D07	0.0254	0.0000	0.5434	**0.5971**	0.1099	0.0284	0.0071	**0.0041**	0.0876	0.1048	0.0251	**0.0234**
D08	0.1973	0.0000	0.6986	**0.7467**	0.0667	0.0518	0.0124	**0.0080**	0.0772	0.1484	0.0401	**0.0351**
D09	0.0195	0.0000	0.5757	**0.5841**	0.1309	0.0314	0.0137	**0.0140**	0.1082	0.1211	0.0367	**0.0362**
D10	0.1482	0.0000	0.5816	**0.6076**	0.0801	**0.0000**	0.0053	0.0130	0.0619	0.1261	0.0352	**0.0340**
D11	0.0008	0.0000	0.5574	**0.5878**	0.1283	0.0181	**0.0081**	0.0129	0.1385	0.1343	0.0367	**0.0289**
D12	0.0000	**0.5850**	0.3802	0.4544	0.4414	**0.0000**	0.0395	0.0430	0.3696	0.0645	0.0466	**0.0406**
D13	0.0000	0.0000	0.5434	**0.6032**	0.2049	0.1017	0.0125	**0.0102**	0.1384	0.0963	0.0156	**0.0138**
D14	0.1419	0.0000	**0.6332**	0.5851	0.0815	0.0900	**0.0120**	0.0132	0.1048	0.2125	0.0531	**0.0451**
D15	0.1898	0.0000	0.6624	**0.7262**	0.0804	0.0149	0.0069	**0.0046**	0.0684	0.1339	0.0421	**0.0413**
D16	0.3764	0.6616	0.8732	**0.8753**	0.0748	0.0352	0.0045	**0.0029**	0.0527	0.0273	0.0158	**0.0157**
D17	0.0000	0.0000	0.4602	**0.4731**	0.4811	1.4696	**0.0125**	0.0159	0.2198	0.6423	0.0107	**0.0094**
D18	0.2718	0.4095	0.8027	**0.8474**	0.0829	0.0720	0.0071	**0.0053**	0.0727	0.0566	0.0272	**0.0246**
D19	0.0000	0.0000	0.5420	**0.5679**	0.4104	0.1716	**0.0126**	0.0166	0.2925	0.1165	0.0149	**0.0129**
D20	0.0000	0.0000	**0.4129**	0.3829	0.3337	0.0289	**0.0140**	0.0193	0.2923	0.0989	0.0262	**0.0211**
D21	0.0000	0.0000	0.5199	**0.5514**	0.5235	0.8879	0.0136	**0.0128**	0.3434	0.4739	**0.0126**	0.0135
D22	0.0000	0.0000	0.5820	**0.6170**	0.3627	0.9323	0.0121	**0.0076**	0.2385	0.3538	**0.0085**	0.0094
D23	0.0000	0.0000	0.6790	**0.7273**	0.1230	0.0597	0.0082	**0.0066**	0.1428	0.1452	**0.0353**	0.0399
D24	0.0965	0.0000	**0.8542**	0.7981	0.0909	0.0622	**0.0128**	0.0159	0.1382	0.2027	0.0786	**0.0777**
D25	0.1465	0.0000	0.8280	**0.8585**	0.1178	0.0625	0.0091	**0.0059**	0.1029	0.1742	0.0328	**0.0314**
D26	0.0232	0.0000	**0.8178**	0.7970	0.0806	0.1016	**0.0070**	0.0084	0.1327	0.1610	**0.0293**	0.0301
D27	0.0000	0.0000	0.6641	**0.7137**	0.2035	0.0547	0.0123	**0.0077**	0.2192	0.1090	**0.0259**	0.0298
D28	0.0000	0.0000	0.5138	**0.6030**	0.1692	0.0533	0.0176	**0.0094**	0.2019	0.1237	**0.0360**	0.0412
D29	0.0000	0.0000	**0.7405**	0.7086	0.1199	0.1086	**0.0077**	0.0084	0.1717	0.1447	0.0192	**0.0179**
D30	0.0000	0.0000	**0.4875**	0.4636	0.3987	0.0284	0.0110	**0.0094**	0.3307	0.0957	0.0211	**0.0200**
D31	0.0000	0.0000	0.5215	**0.5804**	0.1942	0.0445	0.0135	**0.0067**	0.2380	0.1353	0.0384	**0.0338**
D32	0.0000	0.0000	**0.5027**	0.4961	0.5016	0.0193	0.0127	**0.0127**	0.3620	0.0972	0.0272	**0.0268**
D33	0.0000	0.0000	0.5309	**0.5642**	0.1597	0.0149	0.0107	**0.0100**	0.2094	0.1404	0.0486	**0.0442**
D34	0.0000	0.0000	0.6467	**0.6946**	0.1632	0.0993	**0.0122**	0.0131	0.2280	0.1683	0.0402	**0.0395**
D35	0.0174	0.0000	0.6633	**0.6657**	0.0793	0.0187	0.0064	**0.0049**	0.1259	0.1411	0.0429	**0.0414**
D36	0.0000	0.0000	**0.5786**	0.5660	0.6474	0.0890	**0.0110**	0.0170	0.5070	0.0928	**0.0132**	0.0156
D37	0.0000	0.0000	0.6204	**0.6377**	0.1445	0.0430	0.0080	**0.0078**	0.1966	0.1367	0.0293	**0.0268**
D38	0.0000	0.0000	0.5577	**0.5718**	0.2281	0.0364	0.0150	**0.0118**	0.2872	0.1427	0.0495	**0.0398**
D39	0.0000	0.0000	**0.6904**	0.6657	0.1528	0.1069	**0.0103**	0.0182	0.2141	0.1751	0.0478	**0.0456**
D40	0.0000	0.0000	0.7415	**0.7988**	0.1359	0.1141	0.0098	**0.0065**	0.1974	0.1543	0.0207	**0.0177**
D41	0.0000	0.0000	0.5653	**0.6179**	0.7187	1.2709	**0.0191**	0.0195	0.6717	0.6139	0.0191	**0.0142**
D42	0.0000	0.0000	0.4915	**0.5780**	0.8431	2.3658	0.0122	**0.0090**	0.5341	0.7881	0.0131	**0.0106**
D43	0.1052	0.0407	0.8260	**0.9130**	0.0861	0.1754	**0.0067**	0.0089	0.1363	0.1496	**0.0306**	0.0315
D44	0.0000	0.0000	0.6021	**0.6384**	0.2355	0.0815	0.0161	**0.0091**	0.2242	0.1230	0.0402	**0.0351**
D45	0.2236	0.7214	**0.8101**	0.7897	0.0813	0.0221	0.0060	**0.0052**	0.0931	0.0356	**0.0256**	0.0267
D46	0.0000	0.0000	**0.6309**	0.6309	0.2343	0.0287	0.0098	**0.0088**	0.2220	0.1405	**0.0230**	0.0269
D47	0.1498	0.0000	0.5961	**0.6443**	0.0920	0.0231	**0.0060**	0.0065	0.0817	0.1677	0.0487	**0.0485**
D48	0.0000	0.0000	0.7256	**0.7499**	0.1169	0.0832	0.0131	**0.0088**	0.1566	0.1442	0.0247	**0.0238**
D49	0.0000	0.0000	**0.7659**	0.7229	0.1623	0.0951	**0.0123**	0.0135	0.2531	0.1456	**0.0211**	0.0228
D50	0.0000	0.0000	0.6461	**0.6719**	0.2121	**0.0157**	0.0172	0.0271	0.2388	0.1421	**0.0608**	0.0663
D51	0.0000	0.0000	**0.6058**	0.5123	0.1999	0.0399	**0.0123**	0.0157	0.2346	0.1478	0.0658	**0.0592**
D52	0.0000	0.0000	0.5612	**0.6037**	0.5020	0.0494	0.0230	**0.0084**	0.3971	0.1145	**0.0241**	0.0290
D53	0.0080	0.0000	0.6788	**0.7025**	0.0939	0.1276	0.0075	**0.0050**	0.1455	0.1832	**0.0233**	0.0241
D54	0.0000	0.0000	0.5280	**0.5663**	0.3881	**0.0000**	0.0115	0.0257	0.3367	0.1479	0.0505	**0.0405**
D55	0.0603	0.0000	**0.6636**	0.6023	0.1055	**0.0070**	0.0074	0.0094	0.1411	0.2031	**0.0788**	0.0823
D56	0.0000	0.0422	0.4906	**0.5487**	0.8812	0.0857	**0.0126**	0.0161	0.6444	0.0977	0.0174	**0.0134**
D57	0.0000	0.0000	**0.5116**	0.4910	0.6010	0.0370	0.0097	**0.0081**	0.4496	0.1078	0.0318	**0.0293**
D58	0.0000	0.0000	**0.5381**	0.5289	0.7179	0.0398	0.0190	**0.0177**	0.5674	0.1069	0.0244	**0.0176**
D59	0.0000	0.0000	0.5439	**0.5907**	0.1152	0.0168	0.0106	**0.0059**	0.1917	0.1662	0.0721	**0.0757**
D60	0.0006	0.0000	**0.6696**	0.5930	0.1312	0.0181	**0.0052**	0.0054	0.1474	0.1376	**0.0534**	0.0548

consumption. For smaller instances D01–D25, the average execution time of IGA_N3 is approximately 2.1 times that of IGA_NA, while for larger instances D26–D60, this ratio increases to about 2.88.

Despite the notable increase in execution time for IGA_N3 compared to IGA_NA, the results from Section 5.4.2 show that within the same timeframe, IGA_N3 outperforms IGA_NA across all evaluation metrics, making the additional overhead acceptable given its superior performance.

5.5. Comparison to other algorithms

To further evaluate the performance of the proposed IGA algorithm, a comparison is made with three well-known multi-objective algorithms: IMA, PTEA, and NSGA-II. The selection of these algorithms is based on criteria that consider their ability to solve similar problems, the similarity of their algorithm frameworks, and their optimization capabilities. The rationale behind choosing these algorithms includes the following details.

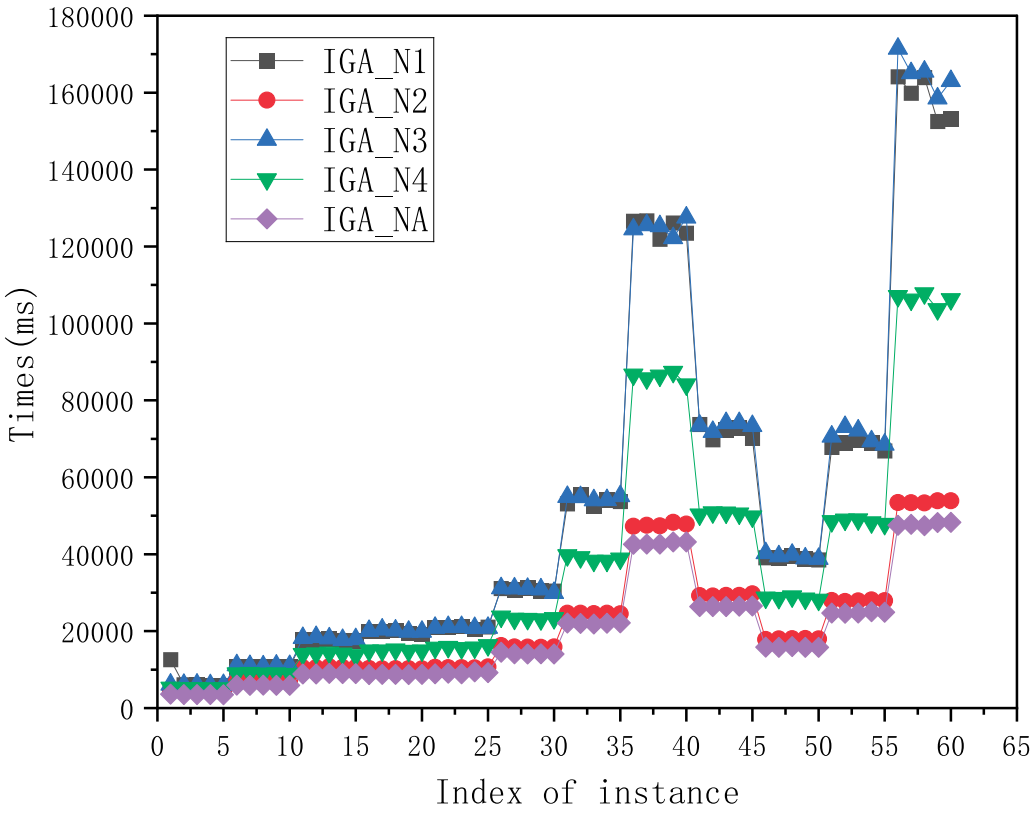

Fig. 8. Time costs using different neighborhood rules.

(1) Specialization and generality. IMA and PTEA are specialized algorithms designed to address DRCFJSP, while NSGA-II is a general classical algorithm applicable to a wide range of FJSP.
(2) Similar algorithm framework. Similar to the IGA algorithm, these three algorithms are all based on the non-dominated sorting genetic algorithm (NSGA) framework.
(3) Enhanced local search. IMA and PTEA incorporate a neighbor search operator to improve local search capabilities, enhancing their performance in finding optimal solutions.

In the comparative experiment, sixty instances D01–D60 are computed for each algorithm under the same CPU time criterion and run times. To ensure fairness, identical parameter settings (e.g., N, P_c, P_m, and P_n), crossover, and mutation operators are set for the compared algorithms.

The results based on HV, GD, and IGD metrics are listed in Table 10. It is clear that the proposed IGA exhibits superior performance in HV metrics, achieving the best results 51 times, compared to 6 times for IMA, 3 times for PTEA, and 0 times for NSGA-II. It is also clear that IGA achieves very promising results in terms of GD metrics, achieving the best results 52 times, compared to 8 times for IMA, and 0 times for both PTEA and NSGA-II. Finally, IGA outperforms the other three algorithms on most instances in terms of the IGD metric, obtaining the best results 44 times, compared to 13 times for IMA, 3 times for PTEA, and 0 times for NSGA-II. To facilitate a clearer comparison of numerical values, Fig. 11 displays the total number of dominant values using a histogram. Based on the HV, GD, and IGD results, it is concluded that the proposed IGA outperforms the other comparison algorithms in most benchmarks, thereby demonstrating its effectiveness in solving the DRCFJSP-TDST.

Additionally, four boxplots are drawn to provide a clearer visualization of the results. These include the overall and individual objectives. The target values are normalized using Eq. (23) and are displayed in Figs. 12, 13, 14, and 15. Fig. 12 shows the overall performance of all tested algorithms. The median of IGA is located at a lower y-coordinate position compared to the others, and it has the shortest range from the best to the worst results. The experimental results demonstrate that the proposed IGA algorithm achieves a median performance value of 1.06065, outperforming the reference algorithms IMA (1.13128), PTEA (1.22844), and NSGA-II (1.29675) in minimizing the combined objectives. Specifically, IGA shows a 6.24% improvement over IMA, a 13.65% enhancement compared to PTEA, and a significant 18.19% performance gain relative to NSGA-II. For the performance of the objectives f_1, f_2, and f_3, Figs. 13, 14, and 15 exhibit that IGA consistently produces a lower median value compared to the other algorithms. Also, the Q1 and Q3 values of the IGA boxplots for f_1, f_2, and f_3 are lower than those of the other three algorithms, respectively. This indicates that IGA has a significant advantage in generating optimal Pareto solutions. Evidently, the trends in the boxplots are consistent with the experimental results based on the HV, GD, and IGD metrics. This confirms that the proposed IGA can obtain the best solutions in most cases.

While the boxplots provide a strong visual indication of IGA's superiority, establishing the statistical significance of these observed differences necessitates formal hypothesis testing. To rigorously validate the distributional characteristics of the overall objectives across algorithms, normality assessment and non-parametric statistical analysis are systematically implemented. The Ryan–Joiner normality test indicates non-normal distribution of overall objectives across all algorithms ($P < 0.01$), prompting the use of the Kruskal–Wallis non-parametric test under formal hypotheses $H0$: all medians are equal versus $H1$: at least one median differs. As shown in Table 11, results ($H = 1852.90$, $P < 0.001$, consistent with/without tie adjustment) strongly rejected the null hypothesis (H_0) of equal medians. Specifically, IGA demonstrates the optimal median (1.06065) with the lowest mean rank (3950.0), and Z-value (−32.97), while NSGA-II shows the highest median (1.29675), mean rank (7324.8), and Z-value (33.14). IMA (median = 1.13128, $Z = -14.06$) and PTEA (median = 1.22845, $Z = 11.94$) occupy intermediate positions. The extreme Z-value divergence (range: −32.97 to 33.14) reveals hierarchical performance

Table 8
Comparison results of neighbor search rules on HV.

Ins	HV				
	IGA_N1	IGA_N2	IGA_N3	IGA_N4	IGA_NA
D01	0.5441	0.5748	**0.6052**	0.5676	0.4880
D02	0.5523	0.5342	**0.5881**	0.5650	0.4747
D03	0.5187	0.5430	**0.5903**	0.5103	0.4442
D04	0.5452	0.5991	**0.6183**	0.5636	0.5143
D05	0.2831	0.3312	0.3347	**0.3476**	0.2475
D06	0.5608	0.5715	0.5884	**0.6043**	0.5298
D07	**0.6222**	0.5983	0.5946	0.5551	0.4964
D08	0.4697	0.4807	**0.4985**	0.4362	0.4420
D09	0.4883	0.4859	0.4655	**0.5227**	0.3699
D10	**0.6682**	0.5785	0.5795	0.5362	0.4463
D11	0.5462	0.5524	**0.6297**	0.6061	0.5930
D12	0.5342	0.5665	**0.6203**	0.5297	0.4907
D13	0.6185	**0.6337**	0.5974	0.5486	0.4719
D14	**0.5797**	0.4969	0.5645	0.5520	0.4616
D15	0.3944	0.4501	**0.4819**	0.4538	0.3873
D16	0.4531	0.4576	**0.4748**	0.4395	0.3913
D17	**0.5559**	0.4965	0.5311	0.5082	0.4495
D18	0.4616	0.5344	**0.5726**	0.4800	0.4507
D19	0.5374	0.5316	**0.5476**	0.5169	0.4497
D20	0.6011	0.5615	**0.6288**	0.5907	0.4911
D21	0.5666	0.5545	**0.6528**	0.5636	0.5638
D22	**0.4185**	0.3065	0.3716	0.3368	0.3218
D23	0.6348	0.6671	**0.6978**	0.6273	0.6164
D24	0.5401	0.5347	**0.5530**	0.5526	0.4612
D25	**0.5982**	0.5609	0.5252	0.5191	0.4172
D26	0.6354	0.6400	0.6647	**0.6790**	0.5787
D27	0.4823	0.4305	**0.5448**	0.4455	0.3761
D28	0.5848	**0.5913**	0.5726	0.5604	0.5079
D29	**0.5208**	0.4795	0.5189	0.4989	0.4386
D30	**0.4841**	0.4549	0.4810	0.4052	0.3545
D31	0.5769	0.5304	**0.6007**	0.5306	0.5091
D32	0.3934	0.3689	0.4131	**0.4268**	0.4139
D33	0.5887	0.6070	**0.6667**	0.6483	0.5216
D34	0.4779	0.4732	**0.5310**	0.5011	0.4491
D35	0.5107	0.4426	**0.5109**	0.4706	0.3748
D36	0.6450	0.6302	**0.6597**	0.6127	0.6062
D37	0.5351	0.5269	**0.5798**	0.5493	0.4719
D38	0.4681	0.4209	**0.4898**	0.4609	0.4476
D39	0.5711	0.5769	**0.6205**	0.6195	0.5468
D40	0.6009	0.6042	**0.6414**	0.5708	0.5438
D41	**0.4160**	0.4151	0.4134	0.4145	0.4021
D42	**0.6037**	0.5707	0.6022	0.5529	0.4992
D43	0.5661	0.5623	**0.5849**	0.5512	0.4987
D44	0.6723	**0.6730**	0.6489	0.6666	0.6175
D45	0.5255	0.5436	**0.6530**	0.5411	0.4163
D46	0.5383	0.5657	0.5960	**0.6128**	0.4388
D47	0.5523	**0.5731**	0.5600	0.5071	0.4281
D48	**0.5749**	0.5574	0.5659	0.5101	0.4322
D49	0.5124	0.5176	**0.5605**	0.5448	0.4890
D50	0.5009	0.5451	**0.5869**	0.5116	0.4683
D51	**0.5559**	0.4966	0.5403	0.5435	0.4765
D52	**0.6770**	0.5246	0.5943	0.5728	0.4834
D53	0.5784	**0.5968**	0.5516	0.4681	0.4142
D54	0.4695	0.4262	**0.5138**	0.4837	0.3566
D55	**0.5449**	0.5367	0.5335	0.2632	0.2894
D56	0.3381	0.4549	**0.5091**	0.4700	0.4090
D57	0.5981	**0.6574**	0.6339	0.5633	0.4929
D58	0.3807	0.3264	0.3920	**0.4447**	0.2461
D59	**0.4861**	0.4262	0.4170	0.4621	0.3242
D60	0.5172	**0.5720**	0.5305	0.5053	0.4540

differences, statistically confirming IGA's superiority in multi-objective optimization.

Figs. 16, 17, and 18 display the mean plots with 95% CI regarding HV, GD, and IGD metrics, from which significant distinctions among the four algorithms can be observed. Fig. 16 shows that the mean HV of IGA is 0.6392, significantly higher than IMA's 0.5909, PTEA's 0.5415, and NSGA-II's 0.5019. Fig. 17 shows that the mean GD of IGA is 0.0120, significantly lower than IMA's 0.0180, PTEA's 0.0318, and NSGA-II's 0.0254. Finally, Fig. 18 shows that the mean IGD of IGA is 0.01411, significantly lower than IMA's 0.0175, PTEA's 0.0210, and NSGA-II's 0.0219. It proves that IGA is obviously superior to other comparison algorithms.

Observational analysis of Pareto front distributions across four random instances including D01 (Fig. 19), D18 (Fig. 20), D34 (Fig. 21), and D55 (Fig. 22) using 3D representations with corresponding 2D projections for D01 (Figs. 23, 24, and 25) demonstrates two distinct advantages of IGA over benchmark algorithms namely enhanced solution uniformity across the Pareto front compared to IMA, PTEA and NSGA-II as well as characteristic solution clustering near the coordinate origin which suggests potential dominance through effective multi-objective balance.

The NSGA-II algorithm performs poorly due to its lack of neighborhood rules and local optimization strategies tailored to the problem and objectives. In contrast, the IGA algorithm, which incorporates problem-focused and objective-oriented strategies, outperforms both the IMA and PTEA algorithms. This analysis confirms the IGA's superior performance in solution distribution, convergence precision, and the number of non-dominated solutions.

5.6. Further discussion

The proposed IGA demonstrates superior performance over IMA, PTEA, and NSGA-II in solving the DRCFJSP-TDST problem, as evidenced by comparative experiments. The enhanced efficacy of IGA can be attributed to three key innovations.

Firstly, while IMA and PTEA utilize traditional resource selection rules such as RND, SPT, CML, and Comprehensive Balance, this study identifies that adopting the HCL rule significantly improves algorithmic efficiency. The HCL rule integrates the strengths of CML and Comprehensive Balance, effectively balancing machine load and tool utilization, which is particularly advantageous for addressing the auxiliary resource constraints and tool-switching dependent setup time inherent to the DCFJSP-TDST problem.

Secondly, IMA and PTEA employ the N1 rule for neighborhood search. To better align with the problem characteristics, this study introduces the N2 rule. Experimental results reveal that standalone use of N1 or N2 yields limited improvements. Consequently, hybrid strategies N3 and N4 are proposed. N3 combines N1 and N2 in a complementary manner, while N4 alternates between the two rules. Comparative evaluations confirm that N3 outperforms N1, N2, and N4, achieving enhanced convergence and solution diversity through its synergistic approach.

Finally, unlike IMA, PTEA, and NSGA-II, IGA incorporates an external archive set during the computation of genetic operators. This mechanism preserves non-dominated solutions across generations, effectively mitigating the loss of Pareto-optimal solutions and ensuring robust multi-objective optimization outcomes.

The superiority of IGA stems from (1) the HCL rule's ability to harmonize resource selection criteria, (2) the hybrid neighborhood search strategy (N3) that optimizes exploitation, and (3) the integration of an external archive set to maintain solution diversity and quality. These innovations collectively address the unique challenges of the DCFJSP-TDST problem, positioning IGA as a more effective solution compared to existing algorithms.

6. Conclusions and future studies

This paper studied DRCFJSP-TDST, considering the tools of two adjacent workpieces on the same machine to calculate the setup time. A MILP model was constructed to simultaneously minimize the makespan, maximum workload of machines, and total setup time. The IGA was designed, featuring a three-vector chromosome encoding method and four heuristic resource selection rules (RND, SPT, CML, and HCL). Additionally, a neighborhood search operator with four specialized rules (N1, N2, N3, and N4) was introduced to enhance the local search ability and improve convergence of the IGA.

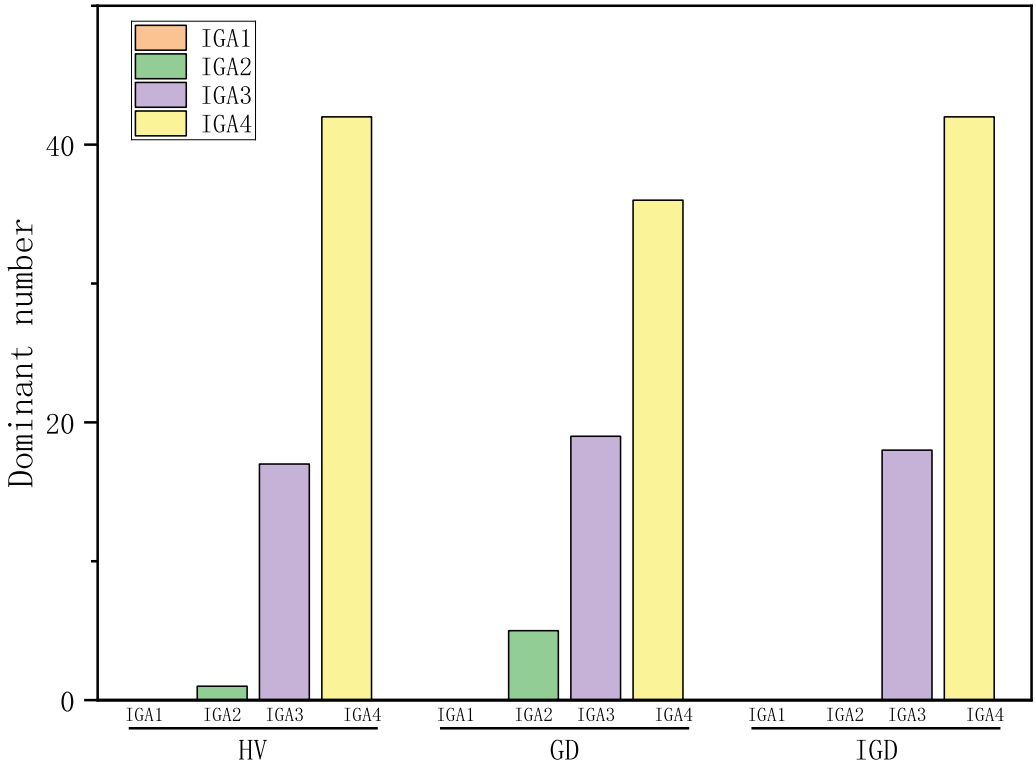

Fig. 9. Histogram of comparison results of resource selection rules.

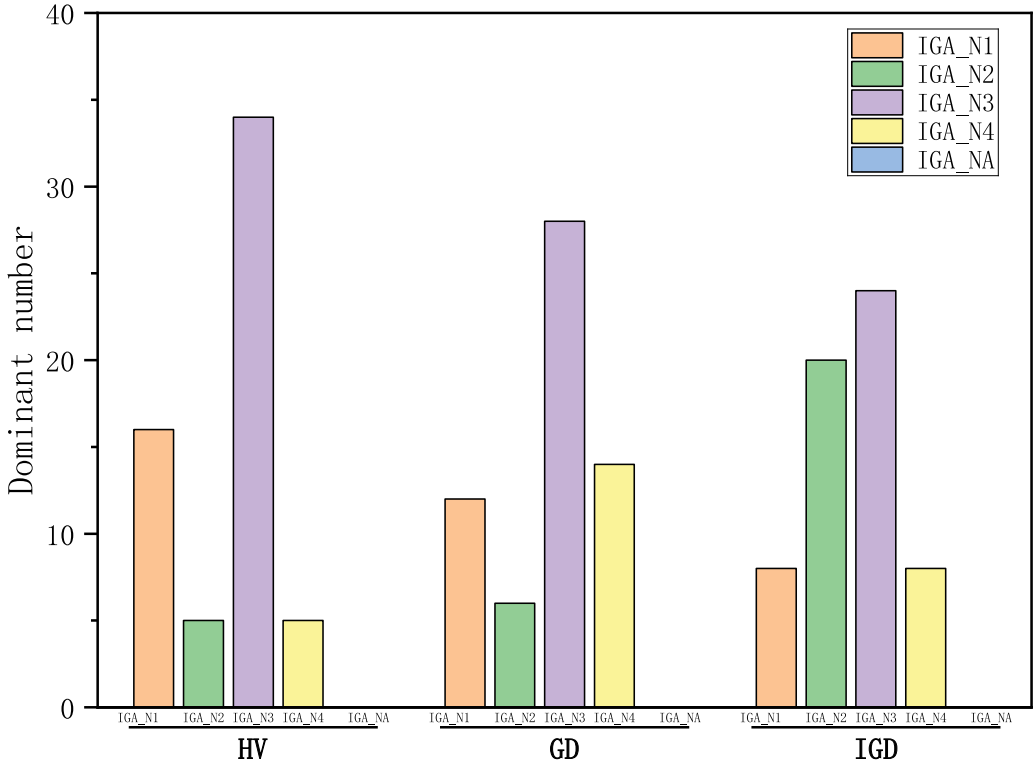

Fig. 10. Histogram of comparison results of neighbor search rules.

Table 9
Comparison results of neighbor search rules on GD and IGD.

Ins	GD					IGD				
	IGA_N1	IGA_N2	IGA_N3	IGA_N4	IGA_NA	IGA_N1	IGA_N2	IGA_N3	IGA_N4	IGA_NA
D01	0.0205	0.0204	**0.0153**	0.0184	0.0244	0.0150	**0.0130**	0.0131	0.0133	0.0191
D02	0.0187	0.0112	**0.0077**	0.0145	0.0181	0.0124	0.0095	**0.0095**	0.0114	0.0135
D03	0.0182	0.0251	**0.0103**	0.0181	0.0242	0.0170	0.0149	**0.0129**	0.0168	0.0185
D04	0.0304	0.0191	0.0236	**0.0182**	0.0316	0.0246	0.0208	0.0197	**0.0193**	0.0255
D05	0.0362	**0.0311**	0.0321	0.0326	0.0552	0.0260	**0.0191**	0.0195	0.0269	0.0281
D06	0.0246	**0.0176**	0.0193	0.0222	0.0239	0.0180	**0.0153**	0.0180	0.0200	0.0186
D07	0.0281	0.0310	0.0257	**0.0243**	0.0284	0.0273	0.0232	0.0244	**0.0218**	0.0308
D08	0.0206	0.0141	**0.0138**	0.0193	0.0366	0.0184	**0.0122**	0.0142	0.0169	0.0200
D09	0.0345	0.0289	**0.0221**	0.0323	0.0537	0.0255	**0.0243**	0.0294	0.0272	0.0348
D10	**0.0185**	0.0283	0.0413	0.0377	0.0490	**0.0192**	0.0202	0.0216	0.0250	0.0278
D11	0.0194	0.0179	**0.0114**	0.0185	0.0253	0.0194	**0.0158**	0.0240	0.0216	0.0182
D12	0.0218	0.0198	**0.0145**	0.0194	0.0254	0.0186	**0.0163**	0.0222	0.0174	0.0240
D13	**0.0181**	0.0287	0.0213	0.0267	0.0258	**0.0186**	0.0193	0.0192	0.0186	0.0246
D14	0.0237	0.0274	0.0207	**0.0166**	0.0264	0.0244	0.0195	**0.0190**	0.0229	0.0218
D15	0.0424	0.0226	**0.0185**	0.0290	0.0376	0.0267	0.0194	**0.0194**	0.0206	0.0293
D16	0.0274	0.0238	**0.0161**	0.0192	0.0320	0.0205	0.0151	**0.0149**	0.0199	0.0312
D17	**0.0259**	0.0285	0.0295	0.0287	0.0287	0.0305	0.0226	**0.0201**	0.0207	0.0279
D18	0.0170	0.0277	**0.0092**	0.0164	0.0211	0.0139	0.0143	0.0136	**0.0129**	0.0132
D19	0.0287	0.0372	0.0280	**0.0245**	0.0408	0.0324	**0.0275**	0.0287	0.0344	0.0288
D20	0.0211	0.0250	**0.0200**	0.0221	0.0325	0.0289	0.0268	**0.0134**	0.0201	0.0259
D21	0.0157	0.0200	**0.0113**	0.0245	0.0401	**0.0151**	0.0194	0.0177	0.0198	0.0211
D22	**0.0202**	0.0315	0.0469	0.0444	0.0655	0.0217	0.0261	**0.0210**	0.0245	0.0315
D23	0.0296	0.0249	**0.0145**	0.0212	0.0330	0.0254	**0.0167**	0.0278	0.0297	0.0301
D24	**0.0206**	0.0268	0.0213	0.0260	0.0361	0.0212	**0.0196**	0.0197	0.0221	0.0243
D25	**0.0231**	0.0299	0.0305	0.0283	0.0333	0.0316	**0.0184**	0.0340	0.0274	0.0355
D26	0.0138	0.0135	**0.0099**	0.0115	0.0194	0.0118	0.0104	**0.0098**	0.0131	0.0142
D27	0.0366	0.0244	**0.0229**	0.0232	0.0332	0.0239	0.0236	0.0257	**0.0228**	0.0279
D28	0.0274	0.0244	0.0214	**0.0173**	0.0318	0.0252	0.0174	0.0173	**0.0162**	0.0267
D29	0.0181	0.0155	**0.0132**	0.0149	0.0202	0.0154	0.0136	**0.0118**	0.0157	0.0146
D30	0.0194	0.0250	0.0284	**0.0189**	0.0308	0.0197	0.0179	**0.0165**	0.0187	0.0227
D31	0.0221	0.0272	0.0223	**0.0185**	0.0340	**0.0153**	0.0183	0.0200	0.0163	0.0187
D32	0.0305	0.0243	**0.0215**	0.0376	0.0522	0.0190	0.0220	**0.0189**	0.0189	0.0251
D33	0.0190	0.0188	**0.0135**	0.0250	0.0253	0.0206	0.0223	**0.0142**	0.0230	0.0265
D34	0.0214	0.0158	**0.0099**	0.0164	0.0260	0.0178	0.0144	**0.0139**	0.0164	0.0186
D35	0.0278	0.0279	0.0285	**0.0271**	0.0331	**0.0298**	0.0301	0.0299	0.0319	0.0361
D36	0.0136	0.0153	**0.0116**	0.0202	0.0191	0.0168	0.0185	**0.0168**	0.0177	0.0177
D37	0.0203	0.0201	**0.0171**	0.0234	0.0274	0.0269	0.0265	0.0273	**0.0173**	0.0284
D38	0.0256	0.0368	**0.0185**	0.0360	0.0524	0.0157	0.0158	0.0171	**0.0157**	0.0172
D39	0.0236	0.0211	0.0212	**0.0183**	0.0287	0.0252	**0.0189**	0.0228	0.0250	0.0251
D40	0.0177	0.0247	**0.0138**	0.0243	0.0232	0.0212	**0.0154**	0.0175	0.0212	0.0186
D41	0.0181	0.0235	0.0169	**0.0120**	0.0270	0.0173	0.0177	**0.0122**	0.0142	0.0169
D42	**0.0152**	0.0281	0.0162	0.0166	0.0268	0.0139	0.0170	**0.0136**	0.0147	0.0180
D43	0.0164	0.0159	**0.0150**	0.0181	0.0241	0.0142	0.0130	**0.0107**	0.0145	0.0180
D44	**0.0141**	0.0142	0.0146	0.0174	0.0193	0.0160	**0.0126**	0.0134	0.0145	0.0200
D45	0.0201	0.0205	**0.0134**	0.0343	0.0288	0.0201	**0.0188**	0.0198	0.0192	0.0239
D46	0.0297	0.0214	0.0180	**0.0130**	0.0324	0.0189	0.0184	**0.0137**	0.0206	0.0252
D47	0.0148	**0.0099**	0.0129	0.0185	0.0311	0.0122	0.0141	**0.0119**	0.0142	0.0191
D48	0.0165	0.0197	**0.0124**	0.0218	0.0262	**0.0125**	0.0209	0.0130	0.0176	0.0230
D49	0.0224	0.0156	**0.0152**	0.0174	0.0280	0.0221	**0.0158**	0.0181	0.0243	0.0257
D50	0.0172	**0.0149**	0.0183	0.0186	0.0257	0.0160	**0.0140**	0.0184	0.0178	0.0234
D51	0.0194	0.0184	0.0192	**0.0160**	0.0279	0.0179	0.0180	0.0177	**0.0165**	0.0324
D52	**0.0150**	0.0281	0.0196	0.0257	0.0331	0.0128	0.0145	**0.0127**	0.0192	0.0184
D53	**0.0250**	0.0273	0.0280	0.0445	0.0337	0.0205	**0.0151**	0.0194	0.0270	0.0321
D54	**0.0212**	0.0352	0.0300	0.0323	0.0524	**0.0218**	0.0263	0.0279	0.0252	0.0336
D55	**0.0167**	0.0195	0.0176	0.0809	0.0648	0.0191	**0.0135**	0.0171	0.0389	0.0354
D56	0.0560	0.0511	**0.0118**	0.0250	0.0407	0.0324	0.0272	**0.0176**	0.0213	0.0297
D57	0.0257	**0.0243**	0.0276	0.0292	0.0372	0.0248	0.0325	**0.0238**	0.0247	0.0325
D58	0.0357	0.0505	0.0351	**0.0262**	0.0478	0.0376	**0.0346**	0.0408	0.0393	0.0481
D59	0.0363	0.0480	0.0268	**0.0259**	0.0421	**0.0213**	0.0435	0.0260	0.0375	0.0371
D60	0.0232	**0.0153**	0.0325	0.0281	0.0250	0.0262	0.0268	**0.0239**	0.0272	0.0275

To evaluate the effectiveness of IGA, sixty instances of the DRCFJSP-TDST were constructed, four key parameters of the IGA were calibrated using the Taguchi method, and three experiments were conducted. The first experiment demonstrated that the HCL rule among the four heuristic resource selection rules generated the highest-quality population. The second experiment verified the effectiveness of the neighborhood search operator by comparing IGA with four neighborhood search rules and without the neighborhood operator, showing that the N3 rule was the most effective. Finally, comprehensive experiments revealed that IGA outperformed three well-known multi-objective algorithms in most cases, highlighting its superior performance in solving the DRCFJSP-TDST.

The findings of this study offer significant research and practical value for addressing FJSP with auxiliary resources and setup time constraints. Future research could expand the scope to scenarios involving multiple auxiliary resources or multi-type auxiliary resources, such as integrating fixtures, operators, or multi-category tools, to better reflect real-world manufacturing complexities. While the hybrid rules (e.g., HCL for resource selection and N3 for neighborhood search) adopted in this work demonstrate strong performance, their predefined nature raises opportunities for further exploration, such as developing

Table 10
HV, GD, and IGD results of all compared algorithms.

Ins	HV				GD				IGD			
	IGA	IMA	PTEA	NSGA-II	IGA	IMA	PTEA	NSGA-II	IGA	IMA	PTEA	NSGA-II
D01	**0.7173**	0.5898	0.5792	0.5325	**0.0025**	0.0194	0.0224	0.0215	**0.0116**	0.0125	0.0140	0.0155
D02	**0.7274**	0.6649	0.6248	0.6050	**0.0062**	0.0124	0.0197	0.0190	0.0140	**0.0120**	0.0178	0.0153
D03	**0.7021**	0.6376	0.5722	0.5533	**0.0066**	0.0117	0.0200	0.0156	**0.0105**	0.0172	0.0151	0.0134
D04	**0.6241**	0.5323	0.4667	0.4339	**0.0064**	0.0175	0.0257	0.0276	**0.0103**	0.0165	0.0204	0.0221
D05	**0.6558**	0.5703	0.5316	0.4886	**0.0067**	0.0182	0.0264	0.0332	**0.0065**	0.0209	0.0240	0.0220
D06	**0.7486**	0.6668	0.6041	0.5966	**0.0077**	0.0127	0.0201	0.0156	**0.0108**	0.0116	0.0144	0.0133
D07	**0.6532**	0.5517	0.5486	0.5089	**0.0065**	0.0313	0.0248	0.0276	**0.0074**	0.0141	0.0177	0.0173
D08	**0.7109**	0.6723	0.6065	0.6077	**0.0043**	0.0101	0.0144	0.0143	0.0105	**0.0096**	0.0121	0.0124
D09	**0.5799**	0.4728	0.4040	0.3132	**0.0229**	0.0403	0.0483	0.0436	**0.0149**	0.0289	0.0347	0.0394
D10	**0.5258**	0.4480	0.3435	0.3657	**0.0110**	0.0301	0.0418	0.0384	**0.0092**	0.0183	0.0241	0.0273
D11	**0.6936**	0.6597	0.6121	0.6135	**0.0066**	0.0196	0.0203	0.0186	**0.0101**	0.0115	0.0130	0.0146
D12	**0.7373**	0.6870	0.6328	0.6055	**0.0091**	0.0139	0.0184	0.0187	**0.0136**	0.0141	0.0164	0.0166
D13	**0.6963**	0.6311	0.5856	0.5310	**0.0063**	0.0184	0.0169	0.0280	**0.0088**	0.0098	0.0149	0.0141
D14	**0.6144**	0.5690	0.5339	0.4847	**0.0074**	0.0155	0.0253	0.0213	**0.0167**	0.0183	0.0246	0.0228
D15	**0.7246**	0.6062	0.5379	0.5218	**0.0022**	0.0166	0.0235	0.0299	**0.0091**	0.0126	0.0177	0.0170
D16	**0.6454**	0.6277	0.5722	0.5449	**0.0101**	0.0123	0.0257	0.0149	**0.0116**	0.0117	0.0154	0.0163
D17	**0.6553**	0.6410	0.5852	0.5451	**0.0045**	0.0112	0.0192	0.0195	**0.0111**	0.0135	0.0157	0.0172
D18	**0.7058**	0.6140	0.5475	0.5163	**0.0084**	0.0230	0.0348	0.0243	**0.0241**	0.0249	0.0307	0.0365
D19	**0.7050**	0.6823	0.6608	0.6258	**0.0080**	0.0120	0.0179	0.0248	**0.0107**	0.0175	0.0260	0.0183
D20	**0.5017**	0.4247	0.4147	0.2943	**0.0225**	0.0285	0.0493	0.0352	**0.0128**	0.0214	0.0312	0.0326
D21	0.5058	**0.5191**	0.3865	0.4197	0.0307	**0.0113**	0.0658	0.0478	0.0144	**0.0136**	0.0231	0.0252
D22	**0.6398**	0.5858	0.5234	0.4928	**0.0072**	0.0148	0.0270	0.0245	0.0166	**0.0155**	0.0196	0.0260
D23	**0.6682**	0.5688	0.5514	0.4763	**0.0172**	0.0262	0.0369	0.0326	0.0261	**0.0176**	0.0269	0.0317
D24	**0.6770**	0.6467	0.6181	0.5685	**0.0146**	0.0198	0.0180	0.0212	0.0219	**0.0196**	0.0227	0.0245
D25	0.6802	0.6579	**0.7072**	0.5807	**0.0113**	0.0163	0.0199	0.0219	**0.0183**	0.0197	0.0208	0.0235
D26	**0.6925**	0.6667	0.6002	0.5643	**0.0059**	0.0099	0.0172	0.0155	**0.0111**	0.0120	0.0127	0.0153
D27	**0.7168**	0.6741	0.6141	0.5814	**0.0046**	0.0136	0.0264	0.0161	**0.0072**	0.0109	0.0125	0.0125
D28	**0.6651**	0.6137	0.4987	0.4936	**0.0092**	0.0180	0.0245	0.0191	0.0156	**0.0150**	0.0198	0.0203
D29	**0.5916**	0.5670	0.5881	0.5407	0.0226	**0.0177**	0.0187	0.0280	0.0218	**0.0212**	0.0158	0.0253
D30	0.6843	**0.6949**	0.6058	0.5687	**0.0068**	0.0087	0.0185	0.0153	**0.0090**	0.0142	0.0154	0.0145
D31	**0.6575**	0.5770	0.5595	0.5326	**0.0092**	0.0109	0.0241	0.0230	**0.0097**	0.0109	0.0174	0.0162
D32	**0.5983**	0.5673	0.5103	0.5064	**0.0103**	0.0143	0.0272	0.0253	0.0134	**0.0130**	0.0179	0.0176
D33	**0.6054**	0.5950	0.4812	0.5207	**0.0077**	0.0124	0.0244	0.0217	0.0136	**0.0131**	0.0151	0.0174
D34	**0.6184**	0.5456	0.5180	0.4830	**0.0111**	0.0245	0.0333	0.0305	**0.0084**	0.0151	0.0175	0.0193
D35	**0.6269**	0.6136	0.5442	0.5804	**0.0075**	0.0128	0.0160	0.0158	0.0192	0.0177	**0.0143**	0.0191
D36	**0.5688**	0.5493	0.4868	0.4763	**0.0078**	0.0137	0.0286	0.0203	0.0188	**0.0134**	0.0172	0.0170
D37	**0.7010**	0.5904	0.5544	0.5532	**0.0150**	0.0195	0.0262	0.0217	**0.0127**	0.0193	0.0222	0.0198
D38	**0.7016**	0.6699	0.5620	0.5850	**0.0062**	0.0126	0.0194	0.0221	**0.0118**	0.0157	0.0183	0.0160
D39	**0.6275**	0.5923	0.5424	0.4815	**0.0114**	0.0157	0.0310	0.0211	**0.0140**	0.0199	0.0229	0.0186
D40	**0.6939**	0.5772	0.4950	0.5258	**0.0136**	0.0212	0.0321	0.0333	**0.0122**	0.0184	0.0250	0.0217
D41	**0.6635**	0.6515	0.5939	0.5707	0.0093	**0.0052**	0.0349	0.0223	**0.0070**	0.0108	0.0122	0.0121
D42	**0.6291**	0.5865	0.5863	0.5418	**0.0096**	0.0111	0.0216	0.0182	**0.0098**	0.0103	0.0113	0.0114
D43	**0.6064**	0.5871	0.5338	0.5135	0.0127	**0.0108**	0.0254	0.0212	**0.0115**	0.0153	0.0200	0.0178
D44	**0.7575**	0.6707	0.6193	0.6187	**0.0061**	0.0173	0.0238	0.0227	**0.0169**	0.0189	0.0261	0.0247
D45	**0.6555**	0.5680	0.5719	0.5358	0.0310	**0.0218**	0.0362	0.0360	**0.0123**	0.0201	0.0273	0.0200
D46	**0.5947**	0.5388	0.4642	0.4059	**0.0118**	0.0215	0.0334	0.0204	**0.0121**	0.0212	0.0157	0.0229
D47	0.6181	**0.6455**	0.5745	0.4285	0.0258	**0.0217**	0.1003	0.0326	**0.0167**	0.0178	0.0283	0.0350
D48	**0.6177**	0.5738	0.5310	0.4893	**0.0072**	0.0111	0.0182	0.0198	0.0149	0.0170	**0.0141**	0.0216
D49	0.5968	**0.6193**	0.5304	0.5020	**0.0136**	0.0174	0.0463	0.0265	**0.0174**	0.0219	0.0256	0.0265
D50	0.5580	0.5628	**0.5671**	0.4351	**0.0296**	0.0424	0.0790	0.0382	**0.0267**	0.0509	0.0640	0.0509
D51	0.4815	**0.5013**	0.4900	0.3527	**0.0156**	0.0180	0.0311	0.0219	**0.0128**	0.0157	0.0195	0.0237
D52	**0.5223**	0.4958	0.4572	0.3991	0.0181	**0.0141**	0.0717	0.0232	**0.0120**	0.0144	0.0260	0.0235
D53	0.5346	0.5006	**0.5389**	0.3889	**0.0189**	0.0213	0.0393	0.0259	**0.0160**	0.0168	0.0218	0.0261
D54	**0.6181**	0.5676	0.5277	0.4289	**0.0151**	0.0339	0.0526	0.0418	0.0206	0.0241	**0.0203**	0.0277
D55	**0.6766**	0.5677	0.5497	0.5072	**0.0159**	0.0176	0.0242	0.0274	**0.0192**	0.0201	0.0236	0.0271
D56	0.5855	**0.6307**	0.5360	0.4633	0.0212	**0.0127**	0.0630	0.0330	**0.0162**	0.0256	0.0220	0.0273
D57	**0.4986**	0.4739	0.4343	0.3223	**0.0209**	0.0300	0.0377	0.0302	**0.0219**	0.0351	0.0239	0.0354
D58	**0.5997**	0.5404	0.4908	0.4642	**0.0154**	0.0247	0.0538	0.0344	**0.0141**	0.0235	0.0241	0.0238
D59	**0.6348**	0.5419	0.4335	0.4064	**0.0159**	0.0233	0.0277	0.0364	0.0197	**0.0184**	0.0293	0.0285
D60	**0.6572**	0.6065	0.5466	0.5225	**0.0129**	0.0142	0.0383	0.0207	0.0187	**0.0170**	0.0266	0.0197

Table 11
Statistical comparisons of overall objectives.

Algorithm	Sample number	Normality test P-value	Median	Mean rank	Z-value	Test statistics		
						DF	H-value	P-value
IGA	2773	<0.01	1.06065	3950.0	−32.97			
IMA	2803	<0.01	1.13128	4997.1	−14.06			
NSGA-II	3494	<0.01	1.29675	7324.8	33.14	3	1852.90	0.000
PTEA	2462	<0.01	1.22845	6476.8	11.94			
Overall	11,532			5766.5				

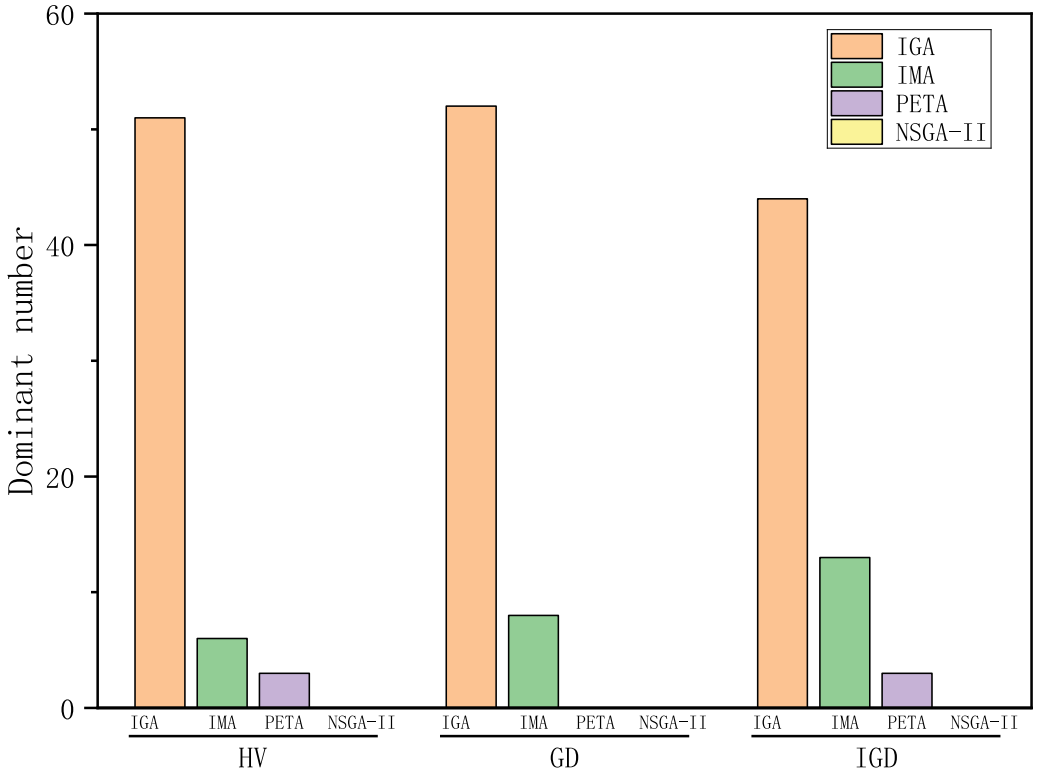

Fig. 11. Histogram of four different algorithms comparison result.

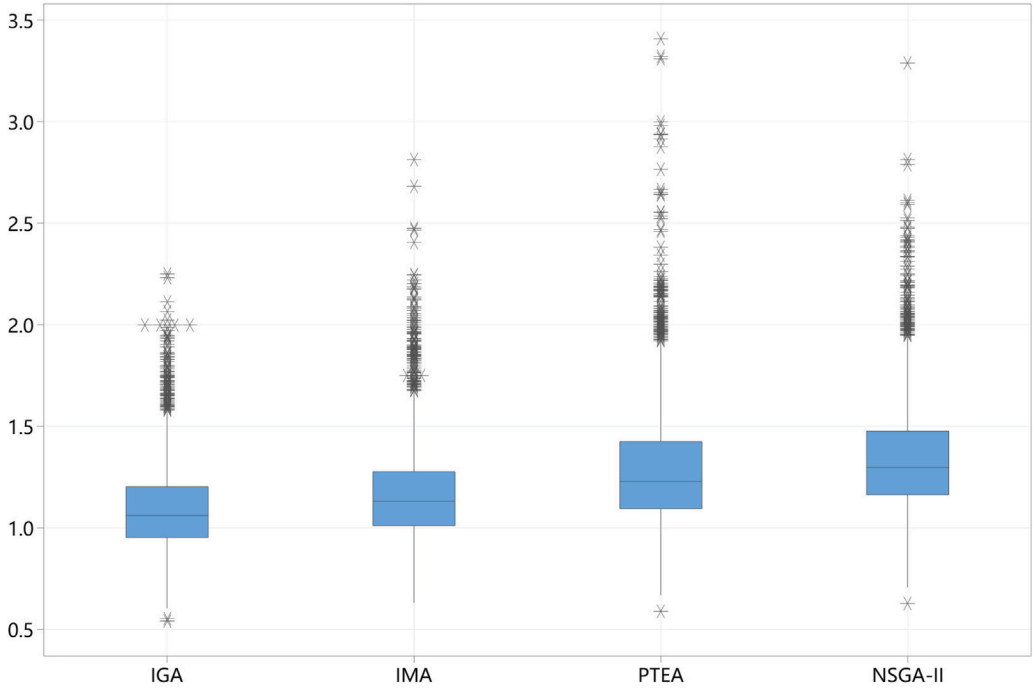

Fig. 12. Boxplot of the overall objective.

self-adaptive mechanisms to dynamically adjust these rules during iterative optimization processes. Additionally, extending the proposed algorithm to ultra-large-scale FJSP instances with hundreds of jobs and machines would test its scalability and computational efficiency, potentially leveraging parallel computing or decomposition-based strategies to manage complexity.

While the preceding discussion primarily focuses on potential algorithmic enhancements, further opportunities exist to refine objectives within the TBL framework. This study primarily considers the economic-social dimension, and future research could further explore the integration of social-environmental or economic-social-environmental dimensions. First, the workload fairness framework could benefit from incorporating skill heterogeneity across operational

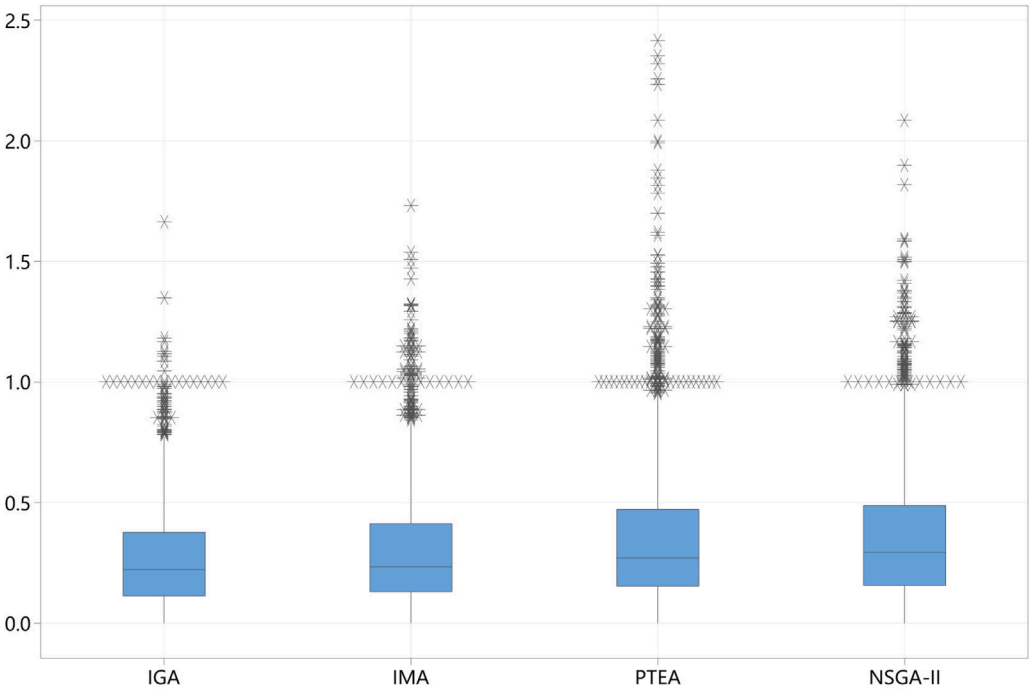

Fig. 13. Boxplot of the objective f_1.

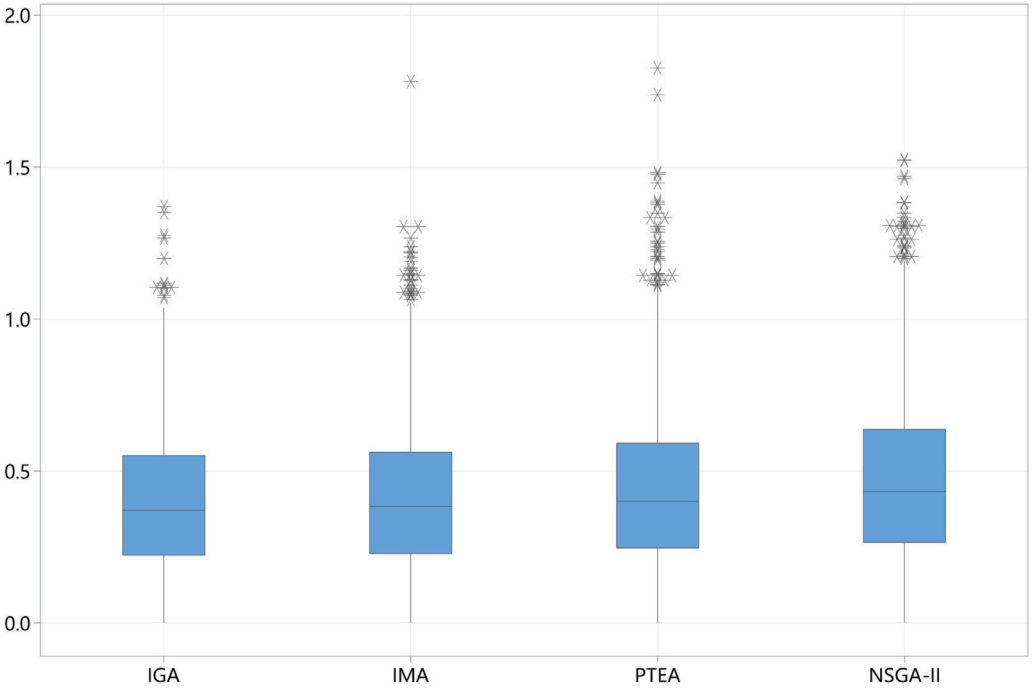

Fig. 14. Boxplot of the objective f_2.

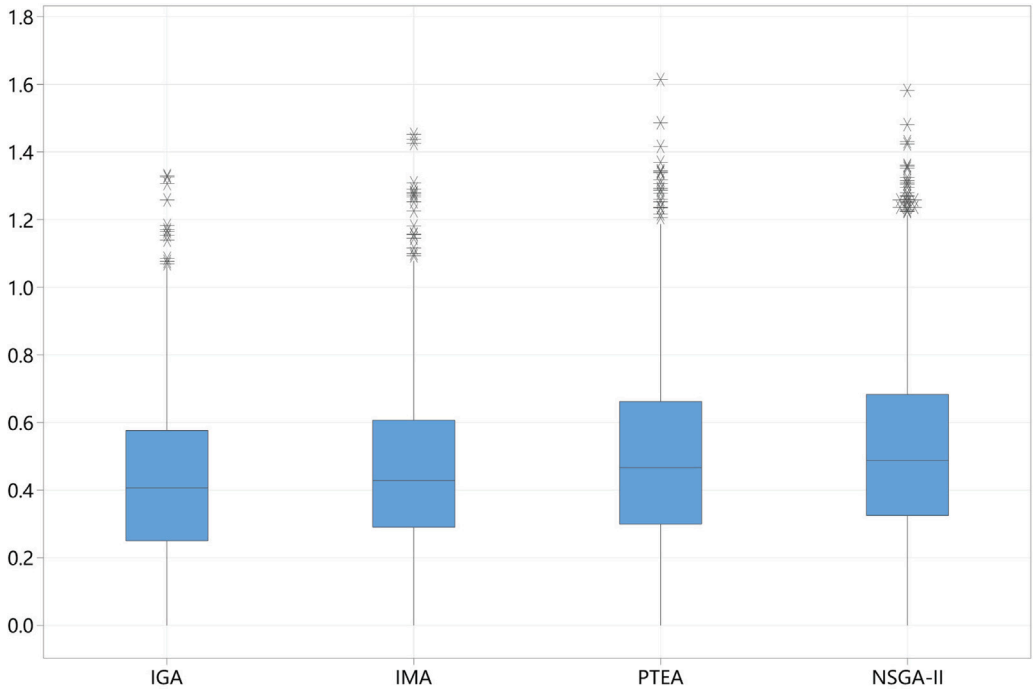

Fig. 15. Boxplot of the objective f_3.

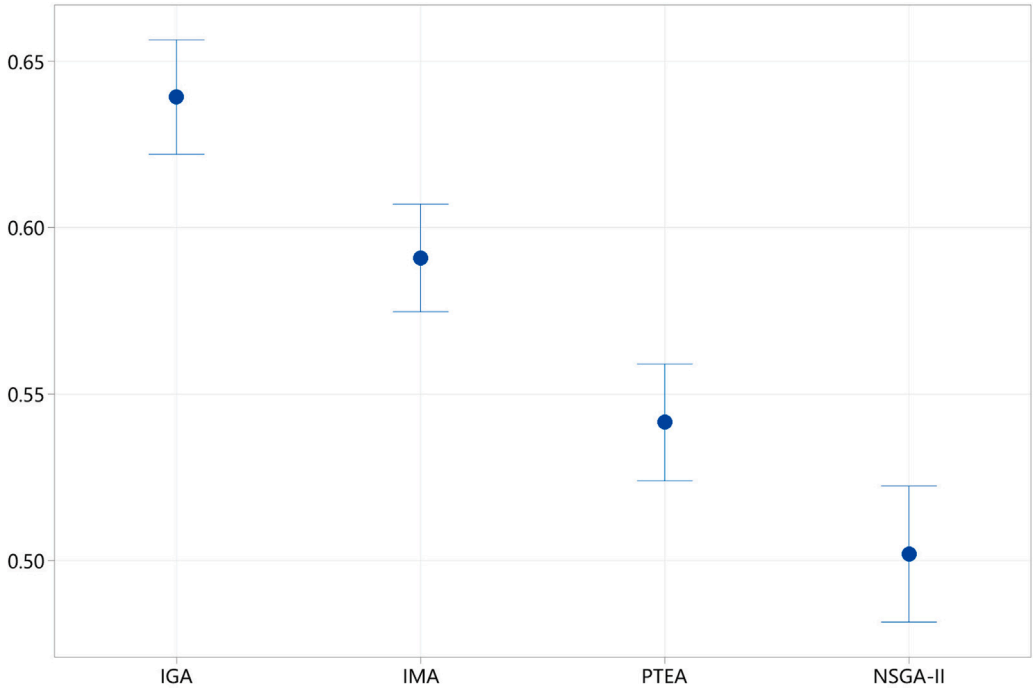

Fig. 16. Mean plot with 95% CI of four algorithms on HV metric.

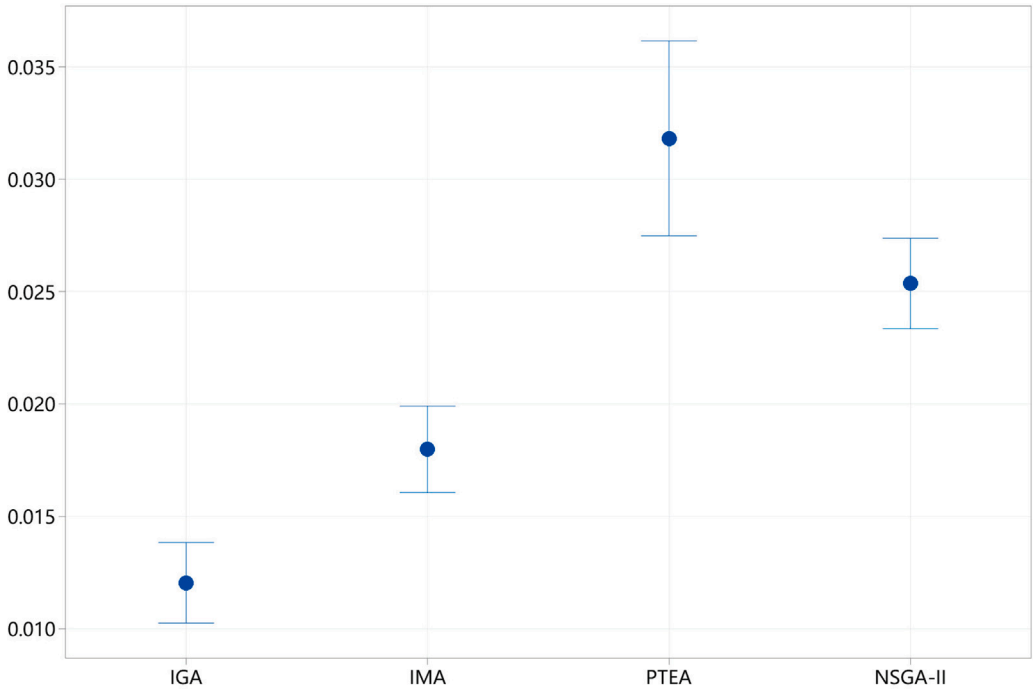

Fig. 17. Mean plot with 95% CI of four algorithms on GD metric.

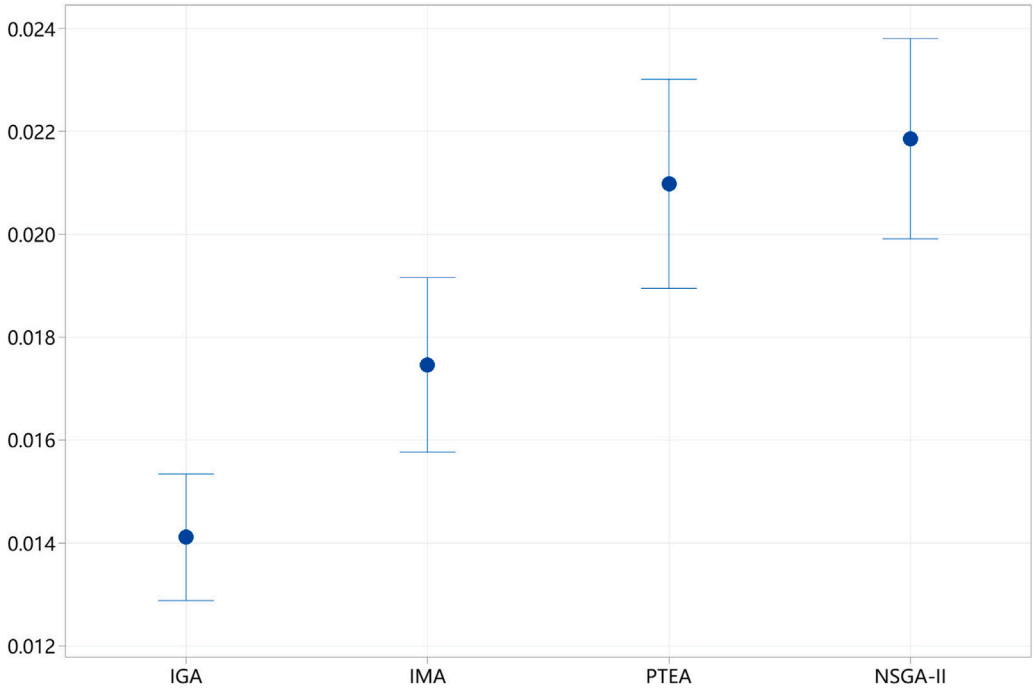

Fig. 18. Mean plot with 95% CI of four algorithms on IGD metric.

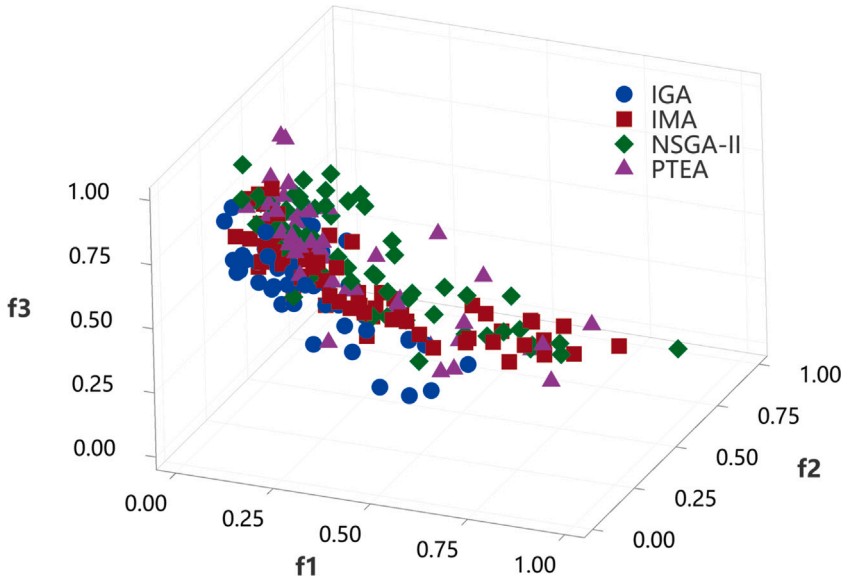

Fig. 19. Pareto front plots of instance D01.

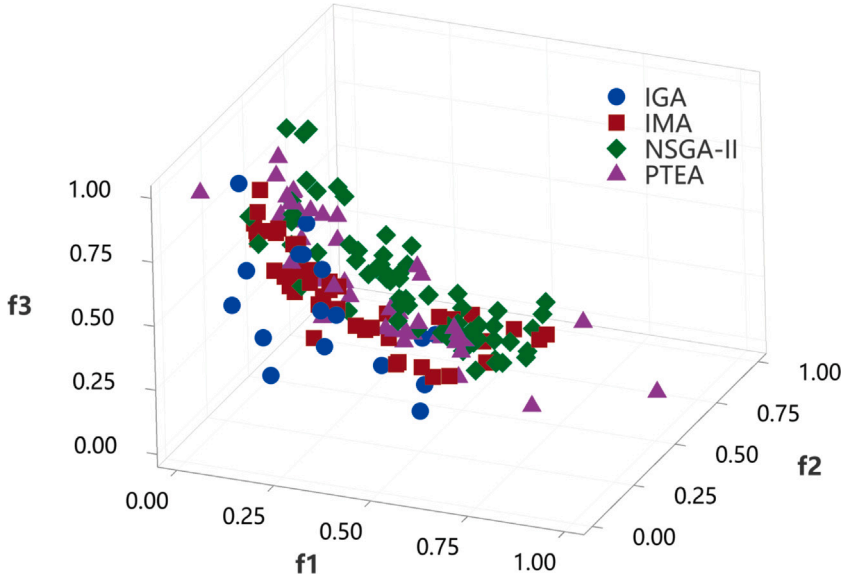

Fig. 20. Pareto front plots of instance D18.

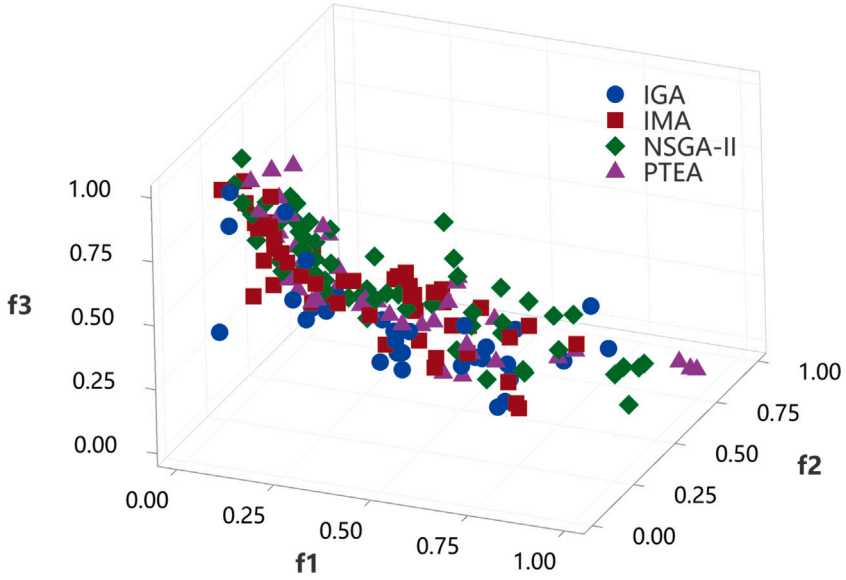

Fig. 21. Pareto front plots of instance D34.

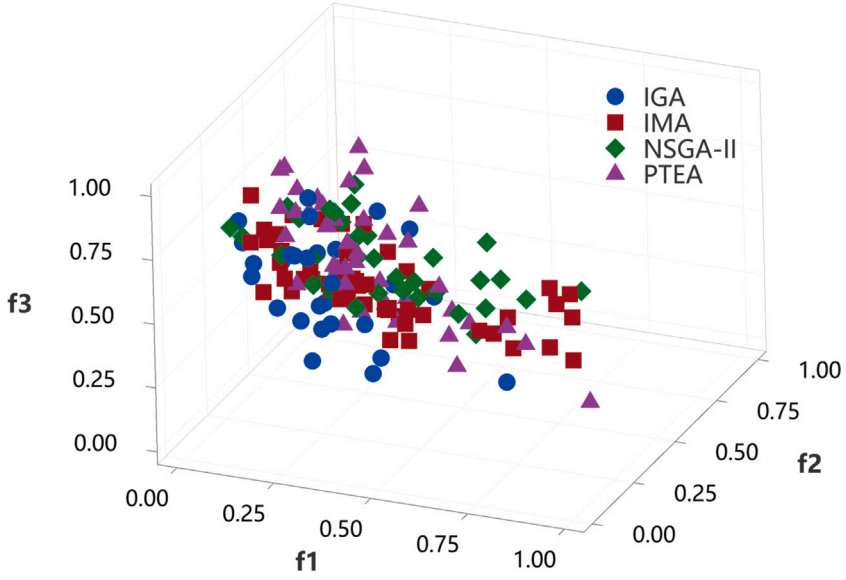

Fig. 22. Pareto front plots of instance D55.

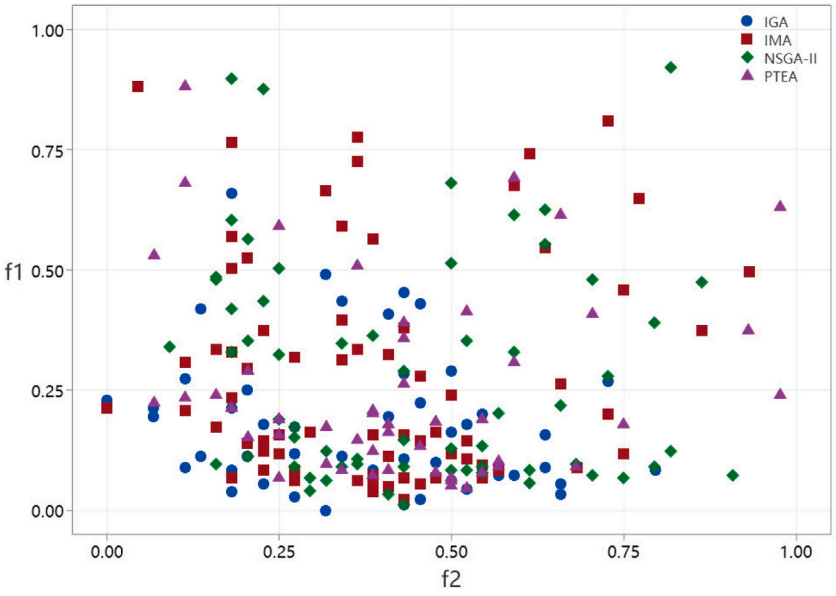

Fig. 23. 2D Pareto front plots of instance D01 about $f1$ and $f2$.

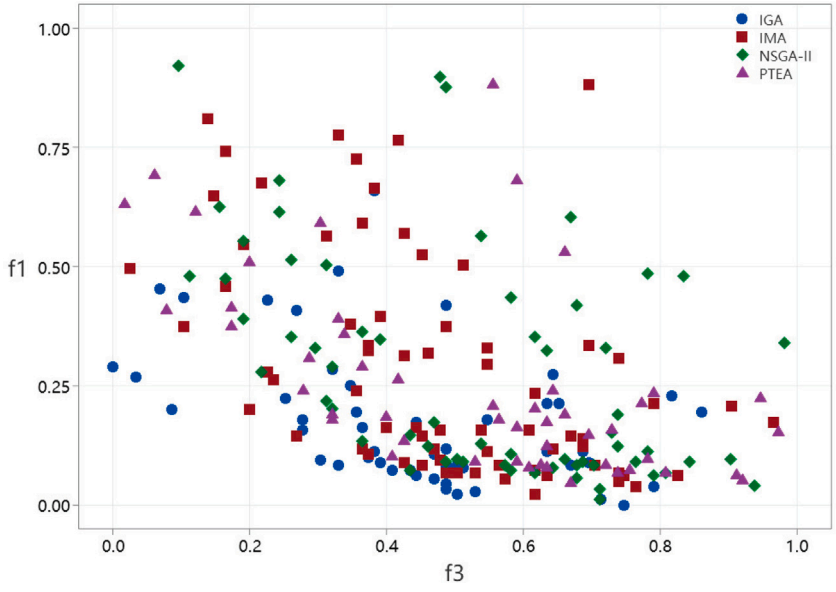

Fig. 24. 2D Pareto front plots of instance D01 about $f1$ and $f3$.

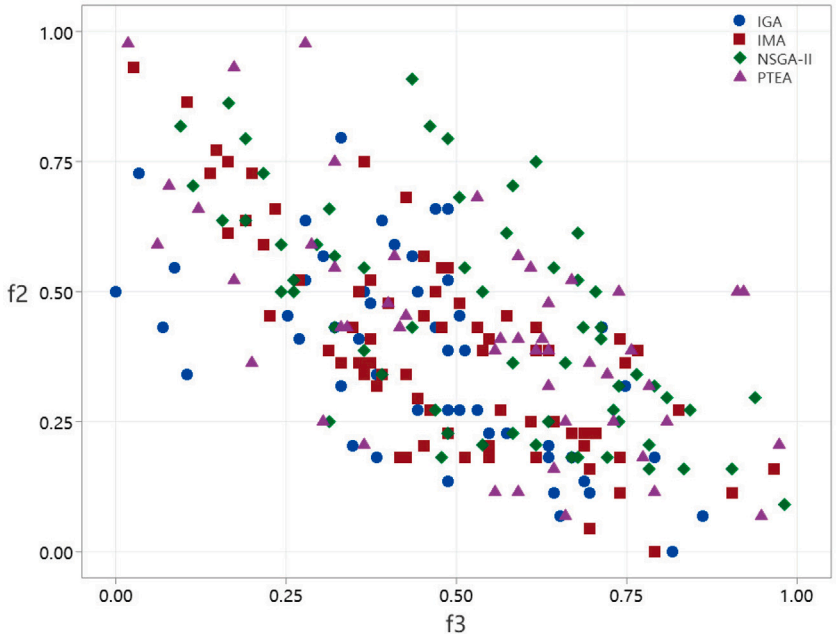

Fig. 25. 2D Pareto front plots of instance D01 about $f2$ and $f3$.

roles, particularly through integrating job difficulty coefficients to enhance equity metrics. Second, the TBL framework's robustness could be further enhanced through explicit incorporation of environmental metrics, although setup time reduction may demonstrate indirectly energy conservation benefits as discussed in Section 2.4. These directions aim to enhance the adaptability and applicability of scheduling algorithms in dynamic industrial environments while addressing emerging challenges in advanced manufacturing systems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, *246*(2), 345–378. http://dx.doi.org/10.1016/j.ejor.2015.04.004.

Amjad, M. K., Butt, S. I., Kousar, R., Ahmad, R., Agha, M. H., Faping, Z., et al. (2018). Recent research trends in genetic algorithm based flexible job shop scheduling problems. *Mathematical Problems in Engineering*, *2018*(1), Article 9270802.

Barak, S., Javanmard, S., & Moghdani, R. (2024). Dual resource constrained flexible job shop scheduling with sequence-dependent setup time. *Expert Systems*, *n/a*(n/a), Article e13669. http://dx.doi.org/10.1111/exsy.13669.

Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multipurpose machines. *Computing*.

Budak, A. (2020). Sustainable reverse logistics optimization with triple bottom line approach: An integration of disassembly line balancing. *Journal of Cleaner Production*, *270*, Article 122475.

Chen, X., Li, J., Wang, Z., Chen, Q., Gao, K., & Pan, Q. (2025). Optimizing dynamic flexible job shop scheduling using an evolutionary multi-task optimization framework and genetic programming. *IEEE Transactions on Evolutionary Computation*.

Cruz-Chávez, M. A., Martínez-Rangel, M. G., & Cruz-Rosales, M. H. (2017). Accelerated simulated annealing algorithm applied to the flexible job shop scheduling problem. *International Transactions in Operational Research*, *24*(5), 1119–1137.

Dauzère-Pérès, S., Ding, J., Shen, L., & Tamssaouet, K. (2024). The flexible job shop scheduling problem: A review. *European Journal of Operational Research*, *314*(2), 409–432. http://dx.doi.org/10.1016/j.ejor.2023.05.017.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. http://dx.doi.org/10.1109/4235.996017.

Dhiflaoui, M., Nouri, H. E., & Driss, O. B. (2018). Dual-resource constraints in classical and flexible job shop problems: A state-of-the-art review. *Procedia Computer Science*, *126*, 1507–1515. http://dx.doi.org/10.1016/j.procs.2018.08.123.

Ding, H., & Gu, X. (2020). Improved particle swarm optimization algorithm based novel encoding and decoding schemes for flexible job shop scheduling problem. *Computers & Operations Research*, *121*, Article 104951.

Fallahi, A., Bani, E. A., & Varmazyar, M. (2025). Towards sustainable scheduling of unrelated parallel batch processors: A multiobjective approach with triple bottom line, classical and data-driven robust optimization. *Computers & Operations Research*, *173*, Article 106863.

Fan, J., Shen, W., Gao, L., Zhang, C., & Zhang, Z. (2021). A hybrid Jaya algorithm for solving flexible job shop scheduling problem considering multiple critical paths. *Journal of Manufacturing Systems*, *60*, 298–311.

Fan, J., Zhang, C., Liu, Q., Shen, W., & Gao, L. (2022). An improved genetic algorithm for flexible job shop scheduling problem considering reconfigurable machine tools with limited auxiliary modules. *Journal of Manufacturing Systems*, *62*, 650–667. http://dx.doi.org/10.1016/j.jmsy.2022.01.014.

Fathollahi-Fard, A. M., Woodward, L., & Akhrif, O. (2021). Sustainable distributed permutation flow-shop scheduling model based on a triple bottom line concept. *Journal of Industrial Information Integration*, *24*, Article 100233.

Fattahi, P., Saidi Mehrabad, M., & Jolai, F. (2007). Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. *Journal of Intelligent Manufacturing*, *18*, 331–342.

Gao, K., Cao, Z., Zhang, L., Chen, Z., Han, Y., & Pan, Q. (2019). A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. *IEEE/CAA Journal of Automatica Sinica*, *6*(4), 904–916. http://dx.doi.org/10.1109/JAS.2019.1911540.

Gao, K. Z., Suganthan, P. N., Tasgetiren, M. F., Pan, Q. K., & Sun, Q. Q. (2015). Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion. *Computers & Industrial Engineering*, *90*, 107–117.

Hajibabaei, M., & Behnamian, J. (2021). Flexible job-shop scheduling problem with unrelated parallel machines and resources-dependent processing times: a tabu search algorithm. *International Journal of Management Science and Engineering Management*, *16*(4), 242–253.

Han, Y., Li, J., Sang, H., Liu, Y., Gao, K., & Pan, Q. (2020). Discrete evolutionary multi-objective optimization for energy-efficient blocking flow shop scheduling with setup time. *Applied Soft Computing*, *93*, Article 106343.

Jiang, X., Tian, Z., Liu, W., Suo, Y., Chen, K., Xu, X., et al. (2022). Energy-efficient scheduling of flexible job shops with complex processes: A case study for the aerospace industry complex components in China. *Journal of Industrial Information Integration*, *27*, Article 100293. http://dx.doi.org/10.1016/j.jii.2021.100293.

Kasapidis, G. A., Dauzère-Pérès, S., Paraskevopoulos, D. C., Repoussis, P. P., & Tarantilis, C. D. (2023). On the multiresource flexible job-shop scheduling problem with arbitrary precedence graphs. *Production and Operations Management*, *32*(7), 2322–2330.

Kemmoé-Tchomté, S., Lamy, D., & Tchernev, N. (2017). An effective multi-start multi-level evolutionary local search for the flexible job-shop problem. *Engineering Applications of Artificial Intelligence*, *62*, 80–95.

Lee, D.-K., Shin, J.-H., & Lee, D.-H. (2020). Operations scheduling for an advanced flexible manufacturing system with multi-fixturing pallets. *Computers & Industrial Engineering*, *144*, Article 106496. http://dx.doi.org/10.1016/j.cie.2020.106496.

Lei, K., Guo, P., Zhao, W., Wang, Y., Qian, L., Meng, X., et al. (2022). A multi-action deep reinforcement learning framework for flexible Job-shop scheduling problem. *Expert Systems with Applications*, *205*, Article 117796.

Li, Y., Chen, X., An, Y., Zhao, Z., Cao, H., & Jiang, J. (2023). Integrating machine layout, transporter allocation and worker assignment into job-shop scheduling solved by an improved non-dominated sorting genetic algorithm. *Computers & Industrial Engineering*, *179*, Article 109169.

Li, X., Guo, X., Tang, H., Wu, R., Wang, L., Pang, S., et al. (2022). Survey of integrated flexible job shop scheduling problems. *Computers & Industrial Engineering*, *174*, Article 108786. http://dx.doi.org/10.1016/j.cie.2022.108786.

Li, Y., Liao, C., Wang, L., Xiao, Y., Cao, Y., & Guo, S. (2023). A reinforcement learning-artificial bee colony algorithm for flexible job-shop scheduling problem with lot streaming. *Applied Soft Computing*, *146*, Article 110658.

Li, Z. C., Qian, B., Hu, R., Chang, L. L., & Yang, J. B. (2019). An elitist nondominated sorting hybrid algorithm for multi-objective flexible job-shop scheduling problem with sequence-dependent setups. *Knowledge-Based Systems*, *173*, 83–112. http://dx.doi.org/10.1016/j.knosys.2019.02.027.

Liao, X., Zhang, R., Chen, Y., & Song, S. (2024). A new artificial bee colony algorithm for the flexible job shop scheduling problem with extra resource constraints in numeric control centers. *Expert Systems with Applications*, *249*, Article 123556. http://dx.doi.org/10.1016/j.eswa.2024.123556.

Liu, M., Lv, J., Du, S., Deng, Y., Shen, X., & Zhou, Y. (2024). Multi-resource constrained flexible job shop scheduling problem with fixture-pallet combinatorial optimisation. *Computers & Industrial Engineering*, *188*, Article 109903. http://dx.doi.org/10.1016/j.cie.2024.109903.

Luo, Q., Deng, Q., Gong, G., Guo, X., & Liu, X. (2022). A distributed flexible job shop scheduling problem considering worker arrangement using an improved memetic algorithm. *Expert Systems with Applications*, *207*, Article 117984. http://dx.doi.org/10.1016/j.eswa.2022.117984.

Luo, Q., Deng, Q., Gong, G., Zhang, L., Han, W., & Li, K. (2020). An efficient memetic algorithm for distributed flexible job shop scheduling problem with transfers. *Expert Systems with Applications*, *160*, Article 113721.

Luo, Q., Deng, Q., Xie, G., & Gong, G. (2023). A Pareto-based two-stage evolutionary algorithm for flexible job shop scheduling problem with worker cooperation flexibility. *Robotics and Computer-Integrated Manufacturing*, *82*, Article 102534. http://dx.doi.org/10.1016/j.rcim.2023.102534.

Luo, C., Li, X., Gong, W., & Gao, L. (2025). Affinity propagation hierarchical memetic algorithm for multimodal multi-objective flexible job shop scheduling with variable speed. *IEEE Transactions on Evolutionary Computation*.

Mahmoodjanloo, M., Tavakkoli-Moghaddam, R., Baboli, A., & Bozorgi-Amiri, A. (2020). Flexible job shop scheduling problem with reconfigurable machine tools: An improved differential evolution algorithm. *Applied Soft Computing*, *94*, Article 106416. http://dx.doi.org/10.1016/j.asoc.2020.106416.

Ortíz, M. A., Betancourt, L. E., Negrete, K. P., De Felice, F., & Petrillo, A. (2018). Dispatching algorithm for production programming of flexible job-shop systems in the smart factory industry. *Annals of Operations Research*, *264*, 409–433.

Park, M.-J., & Ham, A. (2022). Energy-aware flexible job shop scheduling under time-of-use pricing. *International Journal of Production Economics*, *248*, Article 108507.

Rossi, A. (2014). Flexible job shop scheduling with sequence-dependent setup and transportation times by ant colony with reinforced pheromone relationships. *International Journal of Production Economics*, *153*, 253–267.

Shen, L., Dauzère-Pérès, S., & Neufeld, J. S. (2018). Solving the flexible job shop scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, *265*(2), 503–516. http://dx.doi.org/10.1016/j.ejor.2017.08.021.

Shi, S., & Xiong, H. (2024). Solving the multi-objective job shop scheduling problems with overtime consideration by an enhanced NSGA-II. *Computers & Industrial Engineering*, *190*, Article 110001.

Song, W., Chen, X., Li, Q., & Cao, Z. (2022). Flexible job-shop scheduling via graph neural network and deep reinforcement learning. *IEEE Transactions on Industrial Informatics*, *19*(2), 1600–1610.

Soto, C., Dorronsoro, B., Fraire, H., Cruz-Reyes, L., Gomez-Santillan, C., & Rangel, N. (2020). Solving the multi-objective flexible job shop scheduling problem with a novel parallel branch and bound algorithm. *Swarm and Evolutionary Computation*, *53*, Article 100632.

Sun, K., Zheng, D., Song, H., Cheng, Z., Lang, X., Yuan, W., et al. (2023). Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system. *Expert Systems with Applications*, *215*, Article 119359. http://dx.doi.org/10.1016/j.eswa.2022.119359.

Thörnblad, K., Strömberg, A.-B., Patriksson, M., & Almgren, T. (2015). Scheduling optimisation of a real flexible job shop including fixture availability and preventive maintenance. *European Journal of Industrial Engineering*, *9*(1), 126–145. http://dx.doi.org/10.1504/EJIE.2015.067451.

Tian, Y., Gao, Z., Zhang, L., Chen, Y., & Wang, T. (2023). A multi-objective optimization method for flexible job shop scheduling considering cutting-tool degradation with energy-saving measures. *Mathematics*, *11*(2), 324.

Tri Windras Mara, S., Sutoyo, E., Norcahyo, R., & Pratama Rifai, A. (2023). The job sequencing and tool switching problem with sequence-dependent set-up time. *Journal of King Saud University (Engineering Sciences)*, *35*(1), 53–61. http://dx.doi.org/10.1016/j.jksues.2021.02.015.

Wang, Y., Ge, J., Miao, S., Jiang, T., & Shen, X. (2023). Application of hybrid artificial bee colony algorithm based on load balancing in aerospace composite material manufacturing. *Expert Systems with Applications*, *215*, Article 119375. http://dx.doi.org/10.1016/j.eswa.2022.119375.

Wei, S., Tang, H., Li, X., Lei, D., & Wang, X. V. (2024). An improved memetic algorithm for multi-objective resource-constrained flexible job shop inverse scheduling problem: An application for machining workshop. *Journal of Manufacturing Systems*, *74*, 264–290. http://dx.doi.org/10.1016/j.jmsy.2024.03.005.

Wu, X., Peng, J., Xiao, X., & Wu, S. (2021). An effective approach for the dual-resource flexible job shop scheduling problem considering loading and unloading. *Journal of Intelligent Manufacturing*, *32*(3), 707–728. http://dx.doi.org/10.1007/s10845-020-01697-5.

Xiong, H., Shi, S., Ren, D., & Hu, J. (2022). A survey of job shop scheduling problem: The types and models. *Computers & Operations Research*, *142*, Article 105731. http://dx.doi.org/10.1016/j.cor.2022.105731.

Yuan, M., Li, Y., Zhang, L., & Pei, F. (2021). Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm. *Robotics and Computer-Integrated Manufacturing*, *71*, Article 102141.

Yuan, E., Wang, L., Cheng, S., Song, S., Fan, W., & Li, Y. (2024). Solving flexible job shop scheduling problems via deep reinforcement learning. *Expert Systems with Applications*, *245*, Article 123019.

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under industry 4.0. *Journal of Intelligent Manufacturing*, *30*(4), 1809–1830. http://dx.doi.org/10.1007/s10845-017-1350-2.

Zhang, W., Zheng, Y., & Ahmad, R. (2023). The integrated process planning and scheduling of flexible job-shop-type remanufacturing systems using improved artificial bee colony algorithm. *Journal of Intelligent Manufacturing*, *34*(7), 2963–2988.

Zhao, F., Li, M., Zhu, N., Xu, T., et al. (2025). Multi-objective fitness landscape-based estimation of distribution algorithm for distributed heterogeneous flexible job shop scheduling problem. *Applied Soft Computing*, *171*, Article 112780.

Zhou, Y., Du, S., Liu, M., & Shen, X. (2024). Machine-fixture-pallet resources constrained flexible job shop scheduling considering loading and unloading times under pallet automation system. *Journal of Manufacturing Systems*, *73*, 143–158. http://dx.doi.org/10.1016/j.jmsy.2024.01.010.