Entropy-Guided Maximin-Fitness MOPSO with Simulated Annealing for Multi-Objective Dual-Resource Flexible Job Shop Scheduling

1st Jiajie Fan
School of Computer Science
South China Normal University
Guangzhou, China
2024023480@m.scnu.edu.cn

2nd Kailai Zhuang
School of Software
Tiangong University
Tianjin, China
2430111298@tiangong.edu.cn

3rd Rundong Gao
College of Computer Science and Technology
Shenyang University of Chemical Technology
Shenyang, China
3573119488@qq.com

***Abstract*—In modern manufacturing systems, dual-resource scheduling has emerged as a critical challenge, where each operation must be simultaneously assigned to both an eligible machine and a qualified worker. This joint allocation requirement introduces tight resource coupling, which significantly expands the combinatorial search space while severely constraining the feasible solution region and resulting in complex multi-dimensional trade-offs between completion time and operational costs. This paper investigates a Dual-Resource Constrained Flexible Job Shop Scheduling Problem (DRC-FJSP) that minimizes the makespan and a comprehensive production cost integrating raw-material holding, work-in-process holding, direct processing costs, and earliness/tardiness penalties. To effectively approximate Pareto-optimal solutions, we propose EM-MFF-MOPSO, a discrete multi-objective framework. The algorithm features an entropy-guided mechanism for stage-based parameter adaptation, a Maximin Fitness Function (MFF) strategy for leader selection, and a Simulated Annealing (SA) local search to refine solutions. Experiments on reproducible benchmark instances of increasing scales demonstrate that EM-MFF-MOPSO consistently produces more competitive and better-distributed Pareto fronts than representative baselines.**

***Index Terms*—Multi-objective Optimization, Particle Swarm Optimization, Dual-Resource Constraints, Flexible Job Shop Scheduling, Entropy Mechanism**

I. Introduction

The Flexible Job Shop Scheduling Problem (FJSP) represents a fundamental challenge in production planning, generalizing the classical job shop problem by allowing each operation to be processed on any machine from a qualified set. Constructing a feasible schedule thus necessitates solving the coupled sub-problems of operation sequencing and machine assignment, resulting in an NP-hard combinatorial search space that defies traditional exact optimization methods [1]. With the growing emphasis on sustainability and cost-aware production, FJSP research has also moved beyond single makespan minimization toward multi-objective formulations, such as energy-efficient scheduling [2], [3], low-carbon scheduling with human-related factors [4], and cost-aware settings under time-varying energy prices [5]. Meanwhile, the evolutionary computation community continues to propose new hybrid paradigms for FJSP, including restart-enhanced genetic algorithms [6] and learning-assisted evolutionary scheduling strategies [7].

However, many real manufacturing systems are not constrained by machines alone. In machining, assembly, and inspection-intensive workshops, each operation often requires the simultaneous allocation of a machine and a qualified worker (operator), forming a machine–worker pair that executes the task. This dual-resource coupling leads to the Dual-Resource Constrained Flexible Job Shop Scheduling Problem (DRC-FJSP), where feasibility depends not only on machine capacities but also on worker availability and qualification constraints. Compared with classic FJSP, DRC-FJSP yields a substantially tighter feasible region and stronger resource interactions, because a local change in either the machine assignment or the worker assignment may trigger cascading conflicts in the schedule. Moreover, when the production environment requires both time efficiency and economic performance, it becomes necessary to model multi-dimensional trade-offs between completion time and costs (e.g., inventory/holding, processing expenses, and due-date related penalties).

Existing DRC-FJSP studies have demonstrated the necessity of jointly optimizing operation sequencing and machine–worker allocation. Zheng and Wang proposed a knowledge-guided fruit fly optimization algorithm with a dedicated encoding to simultaneously handle job sequence, machine assignment, and worker assignment under a makespan objective [8]. Andrade-Pineda *et al.* formulated a dual-resource flexible job shop model where worker expertise affects eligibility, and optimized makespan together with due-date-related performance measures [9]. Fan and Wang incorporated preparation times and worker proficiency into a genetic-algorithm-based DRC-FJSP framework [10]. Nevertheless, compared with the extensive literature on multi-objective FJSP, the multi-objective DRC-FJSP remains relatively under-explored, especially when the cost objective integrates multiple practical components (e.g., inventory holding, direct processing costs, and earliness/tardiness penalties). On the algorithmic side, Particle Swarm Optimization (PSO) [11] is attractive due to

its simple structure and fast convergence, and it has been extended to handle Pareto dominance in multi-objective settings (MOPSO) [12]. For discrete FJSP variants, researchers have proposed various adaptation strategies to handle combinatorial constraints, such as discrete PSO combined with simulated annealing (SA) [13], two-level PSO separating machine mapping and sequencing [14], and improved PSO with novel encoding/decoding and communication mechanisms [15]. These advances motivate investigating a discrete multi-objective PSO framework tailored to the dual-resource coupling in DRC-FJSP.

Despite the above progress, solving a dual-resource constrained multi-objective FJSP at realistic scales remains difficult. First, each operation involves coupled discrete decisions—its position in the global operation sequence, the selected machine, and the selected worker—so the search space grows multiplicatively, while feasibility becomes fragile due to the simultaneous non-overlap requirements on both machines and workers. Second, when objectives go beyond makespan and include time-dependent cost terms (e.g., holding costs and earliness/tardiness penalties), solution evaluation requires accurate decoding from a discrete representation into a full schedule timeline, which increases the computational burden and makes the optimization landscape more rugged. Third, multi-objective optimization demands not only convergence toward the Pareto front but also a well-distributed approximation set; this is particularly challenging in DRC settings where feasible solutions can be sparse and clustered. Finally, algorithmic robustness across different instance scales calls for adaptive balance between exploration and exploitation—static parameters may perform well on one scale but cause premature convergence or diversity collapse on another.

To address these challenges, this paper makes the following contributions:

- **Problem and benchmarks:** We formulate a multi-objective DRC-FJSP and provide a three-scale reproducible benchmark protocol.
- **EM module:** An entropy-guided multi-stage mechanism adaptively tunes (w_t, c_1^t, c_2^t) and local-search intensity.
- **MFF module:** A maximin fitness function assisted leader sampling strategy strengthens convergence pressure toward the Pareto front while maintaining archive diversity.
- **SA module:** A simulated annealing local search refines discrete schedules to strengthen exploitation.

The remainder of this paper is organized as follows. Section II formulates multi-objective DRC-FJSP. Section III presents EM-MFF-MOPSO. Section IV reports experimental results. Section V concludes the paper.

II. Problem Formulation

We consider a production system with N jobs, M machines, and G workers. Let $\mathcal{J} = \{1, \ldots, N\}$, $\mathcal{M} = \{1, \ldots, M\}$, and $\mathcal{G} = \{1, \ldots, G\}$ denote the corresponding index sets. Each job $i \in \mathcal{J}$ consists of n_i ordered operations $\{O_{i1}, \ldots, O_{in_i}\}$ subject to strict precedence constraints.

TABLE I
Notation for the multi-objective DRC-FJSP

Symbol	Description
$\mathcal{J}, \mathcal{M}, \mathcal{G}$	Sets of jobs, machines, and workers
N, M, G	Numbers of jobs, machines, and workers
n_i	Number of operations of job i
O_{ik}	k-th operation of job i
$\mathcal{M}_{ik}$	Eligible machine set for O_{ik}
$\mathcal{G}_m$	Eligible worker set for machine m
m_{ik}, g_{ik}	Selected machine and worker for O_{ik}
p_{ikm}	Processing time of O_{ik} on machine m
d_i	Due date of job i
ϵ_i	Integer due-date slack (randomly sampled) in benchmark generation
α_i	Raw-material value coefficient of job i
β_m, γ_g	Unit cost coefficients of machine m and worker g
η	Inventory holding cost rate per unit time
ϕ_i, ψ_i	Unit penalty costs for earliness and tardiness of job i
S_{ik}, C_{ik}	Start and completion times of O_{ik}
C_i	Completion time of job i ($C_i = C_{in_i}$)
c_{ik}	Direct processing cost of O_{ik} under (m_{ik}, g_{ik})

For each operation O_{ik}, an eligible machine set $\mathcal{M}_{ik} \subseteq \mathcal{M}$ is given. For each machine $m \in \mathcal{M}$, an eligible worker set $\mathcal{G}_m \subseteq \mathcal{G}$ is given. Processing O_{ik} requires selecting one machine $m_{ik} \in \mathcal{M}_{ik}$ and one compatible worker $g_{ik} \in \mathcal{G}_{m_{ik}}$ simultaneously. The processing time depends on the machine choice, while the worker affects feasibility and labor cost.

A. Notation

Let $\mathcal{O} = \{(i,k) \mid i \in \mathcal{J},\, k = 1, \ldots, n_i\}$ be the set of all operations. Main symbols are summarized in Table I.

B. Objectives

We consider a multi-objective minimization problem of minimizing $(F_1(\mathbf{X}),\, F_2(\mathbf{X}))$. The two objectives are defined as follows:

$$F_1 \;=\; C_{\max} \;=\; \max_{i \in \mathcal{J}} C_i, \tag{1}$$

$$F_2 \;=\; F_{21} + F_{22} + F_{23}, \tag{2}$$

$$c_{ik} \;=\; (\beta_{m_{ik}} + \gamma_{g_{ik}})\, p_{ikm_{ik}}, \tag{3}$$

$$F_{21} \;=\; \sum_{i \in \mathcal{J}} \eta\, \alpha_i\, S_{i1}, \tag{4}$$

$$F_{22} \;=\; \sum_{(i,k) \in \mathcal{O}} \eta\, (C_i - S_{ik})\, c_{ik}, \tag{5}$$

$$F_{23} \;=\; \sum_{(i,k) \in \mathcal{O}} c_{ik} + \sum_{i \in \mathcal{J}} \big(\phi_i E_i + \psi_i T_i\big), \tag{6}$$

$$E_i \;=\; \max(0, d_i - C_i), \qquad T_i \;=\; \max(0, C_i - d_i). \tag{7}$$

Equation (1) defines the makespan, where C_i is the completion time of the last operation of job i. The production-cost objective in Eqs. (2)–(6) consists of three interpretable components: (i) raw-material holding cost (Eq. (4)), which charges the inventory tied up from time 0 until the first start time S_{i1}; (ii) WIP holding cost (Eq. (5)), which charges the time span from an operation start S_{ik} to the job completion C_i weighted by the operation cost proxy c_{ik} in Eq. (3); (iii)

direct processing cost plus due-date penalties (Eq. (6)), where earliness/tardiness are defined in Eq. (7).

C. Constraints and Feasibility

The coupled assignment must satisfy:

$$m_{ik} \in \mathcal{M}_{ik}, \qquad g_{ik} \in \mathcal{G}_{m_{ik}}, \quad \forall (i,k) \in \mathcal{O}. \tag{8}$$

Job precedence constraints require:

$$S_{ik} \geq C_{i,k-1}, \quad \forall i \in \mathcal{J},\ k = 2, \ldots, n_i, \tag{9}$$

$$C_{ik} = S_{ik} + p_{ikm_{ik}}, \quad \forall (i,k) \in \mathcal{O}. \tag{10}$$

Machines and workers are unary, non-preemptive resources: operations assigned to the same machine or the same worker cannot overlap in time. Instead of introducing disjunctive binary variables, we enforce unary feasibility through a serial schedule generation scheme (SSGS) decoder (Algorithm 2), which schedules each operation at its earliest feasible start time considering precedence and current machine/worker availability.

III. Proposed EM-MFF-MOPSO

A. Algorithm Overview

Since DRC-FJSP is a combinatorial optimization problem with discrete decision variables, the traditional PSO velocity-position update paradigm relying on vector addition in continuous space is inapplicable. To address this, EM-MFF-MOPSO reformulates the fundamental PSO behaviors into a probabilistic discrete operation framework.

Let a particle $\mathbf{X}_i$ be represented by the triple $(\mathbf{OS}_i, \mathbf{MA}_i, \mathbf{WA}_i)$. The evolution of $\mathbf{X}_i$ is governed by three probabilistic components that mimic the mechanics of standard PSO:

1) **Inertial Component (w):** In continuous PSO, inertia maintains the particle's previous trajectory. In our discrete framework, this is modeled as a perturbation probability. With probability w_t, the particle executes a mutation operator (e.g., swap mutation) to maintain exploration momentum and prevent stagnation.
2) **Cognitive Component (c_1):** This represents learning from the particle's own history. With probability c_1^t, the particle undergoes crossover with its personal best solution ($pBest_i$), pulling the trajectory toward known successful regions.
3) **Social Component (c_2):** This represents learning from the swarm. With probability c_2^t, the particle undergoes crossover with a global leader ($\mathbf{G}$) selected from the external archive $\mathcal{A}$, guiding the search toward the Pareto front.

Consequently, the velocity update equation is replaced by a cascade of stochastic operators. Furthermore, to overcome the limitations of static parameters in varying search landscapes, we introduce an entropy-guided stage adaptation mechanism that dynamically adjusts the probabilities (w_t, c_1^t, c_2^t).

The overall procedure is outlined in Algorithm 1. The framework iterates through four key phases: (1) **State Assessment**, where the population distribution entropy is calculated; (2) **Adaptive Parameter Setting**, where search preferences are tuned based on the entropy stage; (3) **Discrete Evolutionary Search**, where particles evolve via the probabilistic operators; and (4) **Local Refinement**, where a Simulated Annealing (SA) based local search is applied to a subset of particles to enhance exploitation. A visual workflow of these interactions is presented in Fig. 1.

Algorithm 1 EM-MFF-MOPSO Framework

Require: Max generations $T_{\max}$, Population size N_{pop}, Archive size N_A, Grid B, Entropy thresholds θ_1, θ_2
Ensure: External Archive $\mathcal{A}$

Initialization: Generate initial population $\mathcal{P}$ using mixed heuristics (e.g., MNOR, LNOR)
Evaluate objectives for all $\mathbf{X}_i \in \mathcal{P}$ using SSGS;
Initialize $pBest_i \leftarrow \mathbf{X}_i$ and Archive $\mathcal{A}$ via non-dominated sorting;
for $t = 1$ to $T_{\max}$ **do**
  // Phase 1: Entropy-based State Assessment
  Normalize objectives in $\mathcal{A}$; Calculate normalized entropy H_{norm}^t using $B \times B$ grid;
  Determine search stage $s_t \in \{1, 2, 3\}$ based on thresholds θ_1, θ_2;
  // Phase 2: Parameter Adaptation
  Update parameters $(w_t, c_1^t, c_2^t, p_{\text{LS}}^t)$ according to stage s_t;
  // Phase 3: Discrete Evolutionary Search
  for $i = 1$ to N_{pop} **do**
    Select global leader $\mathbf{G}$ from $\mathcal{A}$ using MFF strategy;
    $\mathbf{X}_i \leftarrow \textsc{ApplyInertia}(\mathbf{X}_i, w_t)$; *// Mutation*
    // Crossover with $pBest_i$
    $\mathbf{X}_i \leftarrow \textsc{ApplyCognition}(\mathbf{X}_i, pBest_i, c_1^t)$;
    // Crossover with leader $\mathbf{G}$
    $\mathbf{X}_i \leftarrow \textsc{ApplySocial}(\mathbf{X}_i, \mathbf{G}, c_2^t)$;
    $\mathbf{X}_i \leftarrow \textsc{Repair}(\mathbf{X}_i)$; *// Ensure feasibility*
    Evaluate $\mathbf{X}_i$; Update $pBest_i$ and $\mathcal{A}$;
  end for
  // Phase 4: Local Refinement
  Sample $N_{LS} = \lceil p_{\text{LS}}^t \cdot N_{\text{pop}} \rceil$ particles;
  Apply SA-based Local Search to sampled particles;
  Update $\mathcal{A}$ with refined solutions;
end for
return $\mathcal{A}$

B. Solution Encoding and SSGS Decoding

To handle the dual-resource constraints, each particle represents a candidate schedule through a triple-vector encoding scheme. A particle $\mathbf{X}_i$ at iteration t consists of three position-wise aligned vectors:

$$\mathbf{X}_i^t = \left(\mathbf{OS}_i^t,\ \mathbf{MA}_i^t,\ \mathbf{WA}_i^t\right), \quad i = 1, \ldots, N_{\text{pop}}, \tag{11}$$

where the common length is $N_{\text{op}} = \sum_{i \in \mathcal{J}} n_i$.

Operation sequence (OS). The vector $\mathbf{OS}$ is a job-based sequence with repetitions. A gene $\mathbf{OS}[j] = i$ indicates that the next unscheduled operation of job i is dispatched at

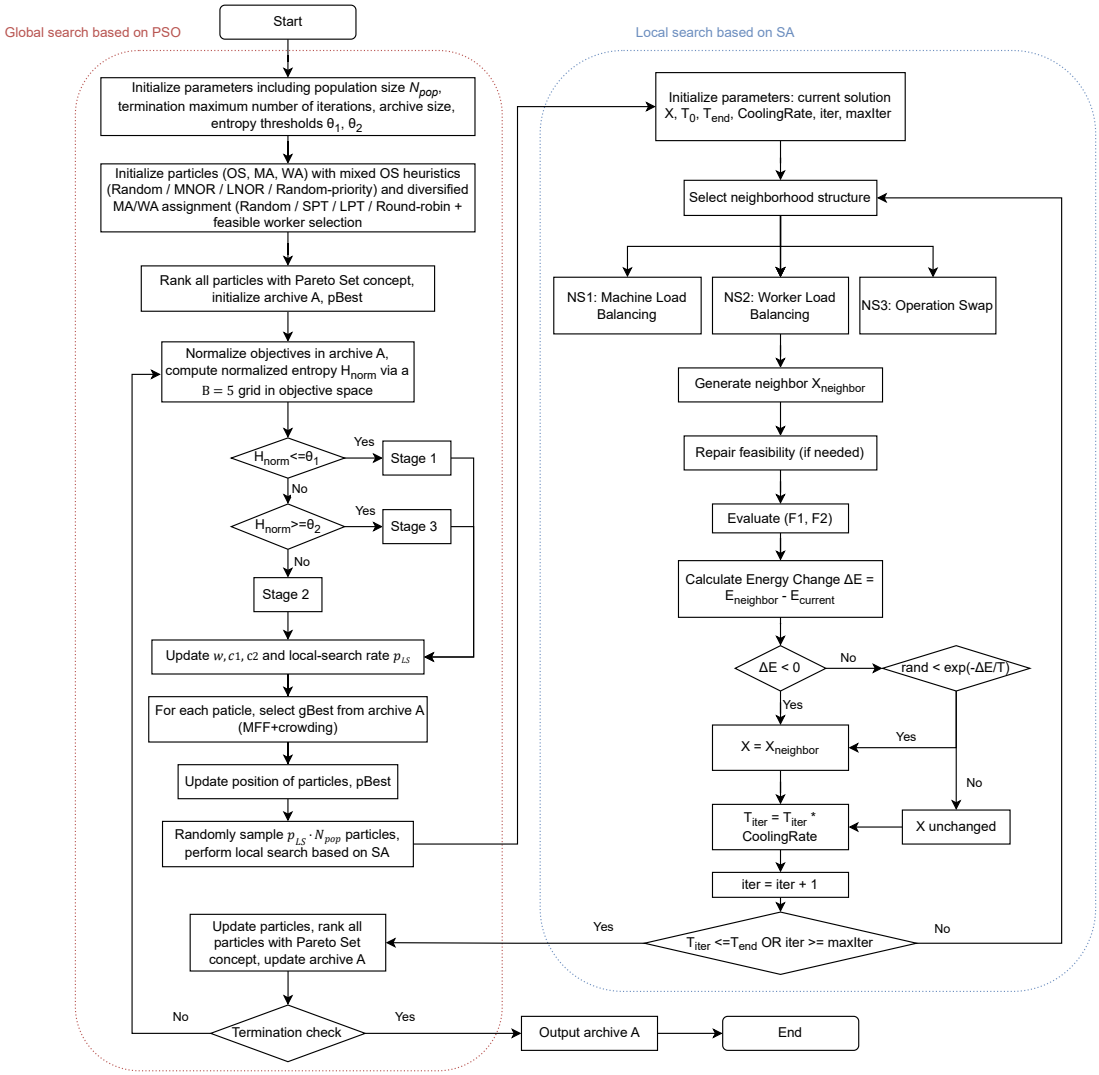

Fig. 1. Workflow of EM-MFF-MOPSO

position j. The corresponding operation index is obtained by an occurrence counter: $k \leftarrow cnt[i] + 1$ and $cnt[i] \leftarrow k$, hence the k-th occurrence of job i corresponds to operation O_{ik}.

Assignments (MA/WA). $\mathbf{MA}[j]$ and $\mathbf{WA}[j]$ specify the selected machine m and worker g for the operation represented at position j. Thus, each position j uniquely determines a concrete tuple (i, k, m, g).

SSGS decoding. As shown in Algorithm 2, decoding scans **OS** from left to right and schedules each derived operation O_{ik} at the earliest feasible start time:

$$S_{ik} = \max\left(T_{\text{job}}[i],\, T_{\text{mach}}[m],\, T_{\text{work}}[g]\right), \tag{12}$$

followed by $C_{ik} = S_{ik} + p_{ikm}$ and availability updates for job i, machine m, and worker g. This construction guarantees feasibility of precedence and dual-resource non-overlap, and yields the schedule used to compute (F_1, F_2).

C. Population Initialization

To improve early-stage coverage, we adopt a mixed initialization strategy. For OS, four rules are mixed: random generation, maximum number of remaining operations (MNOR), least number of remaining operations (LNOR), and a random-priority rule. For MA/WA, we alternate among (i) random assignment, (ii) shortest/longest processing time selection to diversify assignments, and (iii) round-robin selection over eligible machines; the worker is then sampled from the eligible set of the chosen machine.

D. Entropy-Guided Stage Adaptation

1) Objective normalization and entropy: At each iteration, archive objectives are min–max normalized to $[0, 1]^2$:

$$\tilde{F}_k(\mathbf{X}) = \frac{F_k(\mathbf{X}) - F_k^{\min}}{F_k^{\max} - F_k^{\min}}, \quad k = 1, 2, \tag{13}$$

Algorithm 2 SSGS Decoder for Dual-Resource Feasibility

Require: Sequence $\mathbf{OS}$, Assignments $\mathbf{MA}, \mathbf{WA}$
Ensure: Schedule $\mathcal{S}$, Objectives F_1, F_2
Initialize available times: $T_{\text{mach}}[m] \leftarrow 0$, $T_{\text{work}}[g] \leftarrow 0$, $T_{\text{job}}[i] \leftarrow 0$
Initialize operation counter: $cnt[i] \leftarrow 0, \forall i \in \mathcal{J}$
for $j = 1$ to length($\mathbf{OS}$) **do**
  Get job index: $i \leftarrow \mathbf{OS}[j]$
  Update operation index: $k \leftarrow cnt[i] + 1$; $cnt[i] \leftarrow k$
  Get resources: $m \leftarrow \mathbf{MA}[j]$, $g \leftarrow \mathbf{WA}[j]$
  $p_{ikm} \leftarrow$ ProcessTime(i, k, m)
  Determine Start Time:
  $S_{ik} \leftarrow \max\left(T_{\text{job}}[i], T_{\text{mach}}[m], T_{\text{work}}[g]\right)$
  $C_{ik} \leftarrow S_{ik} + p_{ikm}$
  Update Availability:
  $T_{\text{job}}[i] \leftarrow C_{ik}$
  $T_{\text{mach}}[m] \leftarrow C_{ik}$
  $T_{\text{work}}[g] \leftarrow C_{ik}$
  Record S_{ik}, C_{ik} to Schedule $\mathcal{S}$
end for
Calculate F_1, F_2 using Eqs. (1)–(6)
return $\mathcal{S}, F_1, F_2$

where $F_k^{\min} = \min_{\mathbf{X}\in\mathcal{A}} F_k(\mathbf{X})$ and $F_k^{\max} = \max_{\mathbf{X}\in\mathcal{A}} F_k(\mathbf{X})$. We partition $[0,1]^2$ into $B \times B$ grids and compute Shannon entropy over the archive distribution. Let n_b be the number of archive solutions in cell b, $A = |\mathcal{A}|$, and $p_b = n_b/A$. Then

$$H^t = -\sum_{b:p_b>0} p_b \ln p_b, \quad H^t_{\text{norm}} = \frac{H^t}{\ln(B^2)}. \tag{14}$$

2) Stage identification:

$$s_t = \begin{cases} 1, & H^t_{\text{norm}} \le \theta_1, \\ 2, & \theta_1 < H^t_{\text{norm}} < \theta_2, \\ 3, & H^t_{\text{norm}} \ge \theta_2, \end{cases} \tag{15}$$

where $\theta_1 < \theta_2$. Low entropy indicates a clustered archive (diversity risk), while high entropy indicates a well-spread archive.

3) Adaptive parameter control: We adapt operator-triggering probabilities (w_t, c_1^t, c_2^t) and SA local-search probability p_{LS}^t:

$$w_t = \begin{cases} w_{\max}, & s_t = 1, \\ w_{\min} + (w_{\max} - w_{\min})(1 - H^t_{\text{norm}}), & s_t = 2, \\ w_{\min}, & s_t = 3, \end{cases} \tag{16}$$

$$c_1^t = \begin{cases} c_{1,\min}, & s_t = 1, \\ c_{1,\min} + (c_{1,\max} - c_{1,\min}) H^t_{\text{norm}}, & s_t = 2, \\ c_{1,\max}, & s_t = 3, \end{cases}$$
$$c_2^t = \begin{cases} c_{2,\max}, & s_t = 1, \\ c_{2,\max} - (c_{2,\max} - c_{2,\min}) H^t_{\text{norm}}, & s_t = 2, \\ c_{2,\min}, & s_t = 3. \end{cases} \tag{17}$$

$$p_{\text{LS}}^t = \begin{cases} p_{\text{LS,max}}, & s_t = 1, \\ p_{\text{LS,min}} + (p_{\text{LS,max}} - p_{\text{LS,min}})(1 - H^t_{\text{norm}}), & s_t = 2, \\ p_{\text{LS,min}}, & s_t = 3. \end{cases} \tag{18}$$

E. Discrete Particle Update via Multi-Operator Fusion

Given particle $\mathbf{X}_i^t$, personal best $pBest_i$, and a leader $\mathbf{G}$ sampled from the archive, we generate an offspring by sequentially applying three operator categories, where each operator is activated with its associated probability, followed by feasibility repair on assignment genes:

- **Inertia-like mutation (probability w_t):** swap two positions in $\mathbf{OS}$ and swap the aligned genes in $\mathbf{MA}$ and $\mathbf{WA}$; then reassign one machine gene within the eligible set and sample a compatible worker for it.
- **Cognitive-like recombination (probability c_1^t):** recombine with $pBest_i$ using the precedence-preserving order crossover (POX) for $\mathbf{OS}$ and the rand-point preservation crossover (RPX) for $\mathbf{MA}$ and $\mathbf{WA}$, where RPX preserves a randomly selected set of assignment positions from one parent and fills the remaining positions from the other parent.
- **Social-like recombination (probability c_2^t):** recombine with leader $\mathbf{G}$ using the same operators (POX for $\mathbf{OS}$ and RPX for $\mathbf{MA/WA}$).

Subsequently, a repair mechanism ensures feasibility by replacing any invalid machine or worker assignment with a valid candidate sampled from the corresponding eligible set.

After decoding and evaluation, $pBest_i$ is updated by Pareto dominance; if the current solution and $pBest_i$ are mutually non-dominated, $pBest_i$ is replaced with probability 0.5 to maintain search mobility.

F. Archive Maintenance and MFF-Based Leader Sampling

1) Archive maintenance: The archive $\mathcal{A}$ stores non-dominated solutions. Let N_A denote the archive capacity and $A = |\mathcal{A}|$ the current archive size. After merging new candidates, dominated and duplicate solutions are removed. If $|\mathcal{A}| > N_A$, truncation based on crowding distance is applied.

2) Crowding distance: Crowding distance is computed on normalized objectives. Boundary solutions receive large distance. For an interior solution $\mathbf{X}_i$ (after sorting by each objective),

$$D_{\text{dis}}(\mathbf{X}_i) = \sum_{k=1}^{2} \left| \tilde{F}_k(\mathbf{X}_{i+1}) - \tilde{F}_k(\mathbf{X}_{i-1}) \right|. \tag{19}$$

3) Maximin fitness function (MFF): Following [16], the MFF score is

$$F_{\text{mf}}(\mathbf{X}_i) = \max_{j\neq i} \left(\min_{k\in\{1,2\}} \left[\tilde{F}_k(\mathbf{X}_i) - \tilde{F}_k(\mathbf{X}_j) \right] \right). \tag{20}$$

For minimization problems, F_{mf} is typically non-positive within a non-dominated archive. We use $-F_{\text{mf}}$ as a sampling score so that higher values correspond to stronger competitiveness.

4) Leader sampling: Let $\tilde{D}_{\mathrm{dis}}(\cdot)$ and $\tilde{S}(\cdot) = \mathrm{norm}(-F_{\mathrm{mf}}(\cdot))$ be linearly normalized to $[0,1]$ within the archive. The leader sampling probability is

$$P(\ell) \propto \lambda\, \tilde{D}_{\mathrm{dis}}(\mathbf{X}_\ell) + (1-\lambda)\, \tilde{S}(\mathbf{X}_\ell), \quad \mathbf{X}_\ell \in \mathcal{A}, \tag{21}$$

and a leader is drawn by roulette-wheel sampling.

G. SA-Enhanced Local Search

We apply simulated annealing (SA) [17] to a subset of particles per generation. Since SA requires a scalar acceptance criterion, we adopt an additive energy consistent with the implementation:

$$E(\mathbf{X}) = \tilde{F}_1(\mathbf{X}) + \tilde{F}_2(\mathbf{X}). \tag{22}$$

Given a neighbor $\mathbf{X}'$, SA accepts it with probability

$$P_{\mathrm{acc}} = \begin{cases} 1, & \Delta E < 0, \\ \exp(-\Delta E/T), & \Delta E \geq 0, \end{cases} \quad \Delta E = E(\mathbf{X}') - E(\mathbf{X}). \tag{23}$$

Neighborhoods. We use three neighborhoods: (i) machine reassignment with consistent worker reassignment, (ii) worker reassignment under a fixed machine, and (iii) operation-sequence swap with aligned MA/WA swap. Each move is followed by feasibility repair.

Batch application. Each generation, we sample

$$N_{\mathrm{LS}} = \max\left(1,\ \mathrm{round}\left(p_{\mathrm{LS}}^{t} \cdot N_{\mathrm{pop}}\right)\right) \tag{24}$$

particles and apply SA to each. The returned solution replaces the current particle if it dominates it; if mutually non-dominated, it is accepted with a small probability (0.3 in our implementation). If the new solution dominates the current one, $pBest$ is updated accordingly; otherwise $pBest$ is kept unchanged.

H. Computational Complexity

Recall that $N_{\mathrm{op}} = \sum_{i \in \mathcal{J}} n_i$ denotes the total number of operations and $A = |\mathcal{A}|$ denotes the archive size. SSGS decoding for one particle costs $O(N_{\mathrm{op}})$; thus objective evaluation is $O(N_{\mathrm{pop}} N_{\mathrm{op}})$ per iteration. Entropy computation is $O(A)$ for fixed grids. Computing MFF is $O(A^2)$, and crowding distance computation is $O(A \log A)$. The SA module adds $O(N_{\mathrm{LS}} L_{\mathrm{SA}} N_{\mathrm{op}})$ per iteration, where L_{SA} is the SA inner-iteration budget. Overall time complexity is:

$$O\Big(T_{\max} \cdot \big(N_{\mathrm{pop}} N_{\mathrm{op}} + (A^2 + A \log A) + N_{\mathrm{LS}} L_{\mathrm{SA}} N_{\mathrm{op}}\big)\Big). \tag{25}$$

IV. Experimental Study

A. Benchmark Instances

We evaluate three instance scales (`small`, `medium`, `large`). In our implementation, the sizes are:

- `small`: $N = 3$, $M = 3$, $G = 2$, with $(n_i)_{i=1}^{3} = (3, 2, 3)$;
- `medium`: $N = 6$, $M = 6$, $G = 4$, with $(n_i)_{i=1}^{6} = (3, 4, 3, 4, 3, 4)$;
- `large`: $N = 10$, $M = 10$, $G = 6$, where each n_i is independently sampled from $\{3, 4, 5, 6\}$.

For `small`, all processing times, eligible machine sets, eligible worker sets, cost coefficients, and due dates are explicitly specified (a deterministic instance). For `medium` and `large`, instance data are generated as follows.

For each operation, the eligible machine set size is uniformly sampled as 2–4 (`medium`) or 2–5 (`large`) without replacement. Processing times on eligible machines are sampled as integers in $[2, 10]$ (`medium`) or $[3, 15]$ (`large`).

For each machine m, the eligible worker set size is uniformly sampled from $\{2, \ldots, G\}$ and workers are selected uniformly without replacement.

For `medium`, $\alpha_i \sim U(5, 15)$, $\beta_m \sim U(3, 8)$, and $\gamma_g \sim U(5, 13)$. For `large`, $\alpha_i \sim U(5, 20)$, $\beta_m \sim U(3, 11)$, and $\gamma_g \sim U(5, 17)$. In both cases, $\eta = 0.01$ and $(\phi_i, \psi_i) = (2, 5)$ are fixed for all jobs.

Due dates are estimated from a lower bound of total processing time:

$$d_i = 1.5 \sum_{k=1}^{n_i} \min_{m \in \mathcal{M}_{ik}} p_{ikm} + \epsilon_i, \tag{26}$$

where ϵ_i is an integer slack sampled in $[5, 15]$ for `medium` and $[10, 30]$ for `large`.

B. Compared Algorithms

We compare EM-MFF-MOPSO with representative baselines, including canonical MOPSO [12], NSGA-II [18], and MOEA/D [19], all configured with standard settings commonly used in the literature.

To ensure a strictly fair comparison, all algorithms share the same discrete solution representation, SSGS decoder, and objective evaluation functions. Furthermore, they are executed with the same population size, ensuring an equivalent number of objective evaluations across all comparisons.

C. Parameter Settings

All algorithms use $N_{\mathrm{pop}} = 200$, $T_{\max} = 300$, and archive size $N_A = 50$. EM-MFF-MOPSO uses $B = 5$, $(\theta_1, \theta_2) = (0.25, 0.75)$, and $\lambda = 0.6$. SA uses $(T_0, T_{\mathrm{end}}, \rho) = (5, 0.01, 0.85)$ and a fixed inner-iteration budget. Each algorithm is executed for 10 independent runs; we report mean $\pm$ standard deviation.

D. Performance Metrics

Let $\mathcal{P}$ be the obtained non-dominated set and $\mathcal{P}^\star$ a reference Pareto set. Both objectives are minimized. We report: (i) MID, computed as the mean Euclidean distance of points in $\mathcal{P}$ to the origin in the objective space, $\mathrm{MID} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \|(F_1(\mathbf{x}), F_2(\mathbf{x}))\|_2$; (ii) MS (Maximum Spread), $\mathrm{MS} = \sqrt{(\max F_1 - \min F_1)^2 + (\max F_2 - \min F_2)^2}$; and standard multi-objective metrics IGD [20] and HV [21]. For HV, objectives are first normalized to $[0, 1]$ using the ideal and nadir values estimated from $\mathcal{P}^\star$, and the reference point is set to $\mathbf{r} = (1, 1)$ in the normalized objective space.

Since the true Pareto front is unknown, we construct $\mathcal{P}^\star$ by aggregating all non-dominated solutions across all algorithms and runs, then removing dominated points and duplicates.

TABLE II
PERFORMANCE COMPARISON ON THREE INSTANCE SCALES (MEAN ± STD OVER 10 RUNS).

Scale	Algorithm	MID↓	MS↑	IGD↓	HV↑
small	EM-MFF-MOPSO	**268.56±5.99**	**103.85±22.67**	**6.1110±1.8846**	**0.1186±0.0087**
	MOPSO	291.18±3.24	44.70±6.72	26.7641±2.7019	0.0862±0.0039
	NSGA-II	287.39±3.28	49.54±7.41	24.1664±0.9559	0.0907±0.0020
	MOEA/D	287.80±4.40	56.68±13.62	24.6971±1.0391	0.0851±0.0034
medium	EM-MFF-MOPSO	1170.08±53.28	**237.33±169.73**	**54.7092±26.4239**	**0.0667±0.0212**
	MOPSO	1373.61±57.69	58.94±59.23	287.1007±58.8053	0.0008±0.0025
	NSGA-II	1152.32±53.71	82.91±60.04	71.2818±20.5130	0.0565±0.0126
	MOEA/D	**1133.90±27.11**	111.11±79.91	75.7301±17.8714	0.0352±0.0109
large	EM-MFF-MOPSO	**5949.88±199.76**	449.14±539.80	**299.6522±72.9364**	**0.0678±0.0089**
	MOPSO	8122.67±466.93	1454.80±943.55	1481.1534±311.2263	0.0011±0.0030
	NSGA-II	6455.57±319.69	542.83±292.73	369.3368±118.6797	0.0552±0.0123
	MOEA/D	7215.18±324.16	**1957.32±1229.12**	680.7578±139.9099	0.0000±0.0000

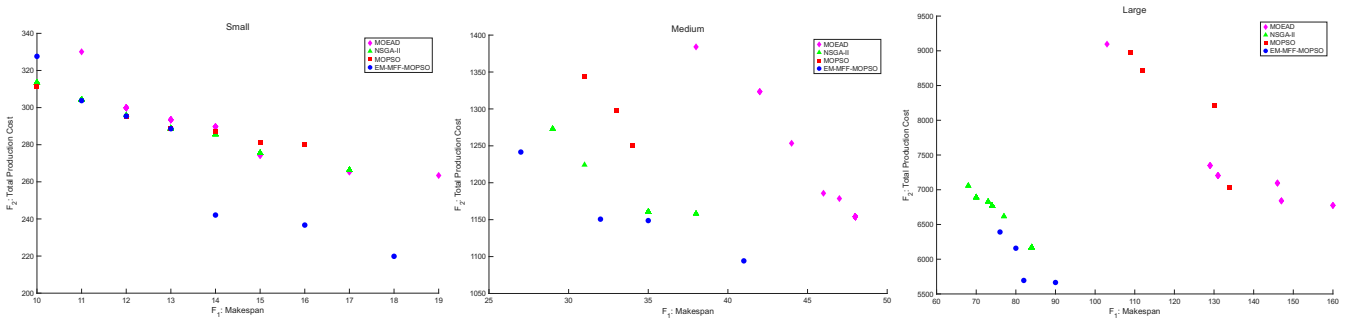

Fig. 2. Pareto front comparisons on three instance scales.

E. Comparative Results and Discussion

Table II summarizes the quantitative results, and Fig. 2 visualizes the obtained Pareto fronts.

Small scale. EM-MFF-MOPSO achieves the best overall performance on `small`, simultaneously improving convergence (lower MID and IGD) and diversity (higher MS and HV). The Pareto front in Fig. 2 is also more uniformly distributed with a clearer trade-off coverage.

Medium scale. On `medium`, EM-MFF-MOPSO attains the best diversity-related metrics (MS and HV) and the best IGD, while MOEA/D yields a slightly smaller MID. This indicates that EM-MFF-MOPSO provides a better approximation set to the reference front with improved coverage, whereas a single scalar indicator (MID) may not fully reflect set quality in rugged feasible landscapes.

Large scale. On `large`, EM-MFF-MOPSO obtains the best MID, IGD, and HV. Although MOEA/D reports the largest MS, its HV remains very small, suggesting that the large spread is caused by a few extreme points rather than a well-covered front. Canonical MOPSO exhibits near-zero HV on `medium` and `large`, implying a severe diversity collapse, which is consistent with the leader-guided swarm dynamics being prone to clustering under strong dual-resource coupling.

Overall, these results demonstrate that the proposed framework effectively overcomes the scalability challenges inherent in DRC-FJSP. By dynamically adapting search behaviors and enforcing diversity pressure, EM-MFF-MOPSO successfully avoids the local optima and diversity collapse that handicap traditional algorithms in highly constrained dual-resource environments.

F. Ablation Study

To isolate the contribution of each proposed component, we evaluate three variants: (i) **noEM**, which disables the entropy-guided stage identification and keeps operator probabilities fixed; (ii) **noMFF**, which removes the MFF term in leader sampling and relies on a standard density-based leader selection; and (iii) **noSA**, which disables the SA-based local search module. All variants keep the same encoding/decoder, archive size, and evaluation budget as the full method.

Fig. 3 shows that removing any component degrades the final Pareto front, and the degradation becomes more pronounced as the instance scale increases. In particular, **noEM** relies on static parameters and thus lacks cross-scale adaptability: settings that work well on one scale may cause premature convergence or diversity collapse on another. **noMFF** exhibits insufficient convergence pressure toward the Pareto front, as the leader selection becomes less competitive without the maximin-based guidance. **noSA** results in consistently poorer convergence, indicating that local refinement is critical for improving solution quality in the rugged feasible landscape induced by dual-resource coupling.

V. CONCLUSION AND FUTURE WORK

In this study, we addressed the multi-objective DRC-FJSP by integrating a comprehensive production cost model that

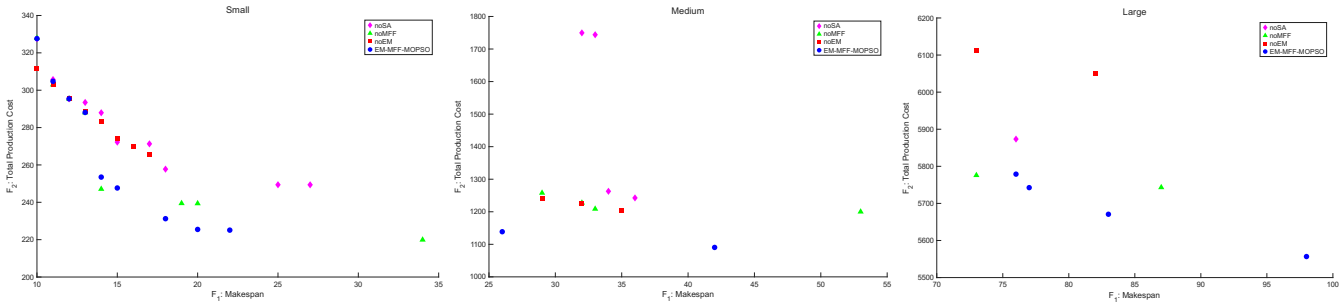

Fig. 3. Pareto front comparisons on three variants.

accounts for inventory holding, processing expenses, and delivery penalties. To navigate the tightly coupled resource constraints, we proposed EM-MFF-MOPSO. The results suggest that entropy-guided stage control effectively mitigates archive concentration, while the MFF-assisted leader sampling improves guidance robustness; the SA refinement further strengthens exploitation on rugged feasible basins, leading to better IGD/HV performance.

Future work may incorporate richer shop-floor constraints (e.g., sequence-dependent setups, transportation times, machine breakdowns, and worker shift calendars), additional objectives (e.g., energy and emissions), scalable parallel implementations, and preference-based decision support for selecting final schedules.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 62573201, U23A20303, 62573326, 62533016, 61825203, 62332007, and U22B2028), Zhejiang Provincial Natural Science Foundation of China (Grant No. LZ25F030007), Outstanding Youth Project of Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023B1515020064), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2024A1515140109).

Artificial intelligence tools were used to polish the language and improve the readability of this paper.

References

[1] I. A. Chaudhry and A. A. Khan, "A research survey: review of flexible job shop scheduling techniques," *International Transactions in Operational Research*, vol. 23, no. 3, pp. 551–591, 2016.

[2] H. Mokhtari and A. Hasani, "An energy-efficient multi-objective optimization for flexible job-shop scheduling problem," *Computers & Chemical Engineering*, vol. 104, pp. 339–352, 2017.

[3] M. Li and D. Lei, "An imperialist competitive algorithm with feedback for energy-efficient flexible job shop scheduling with transportation and sequence-dependent setup times," *Engineering Applications of Artificial Intelligence*, vol. 103, p. 104307, 2021.

[4] H. Zhu, Q. Deng, L. Zhang, X. Hu, and W. Lin, "Low carbon flexible job shop scheduling problem considering worker learning using a memetic algorithm," *Optimization and Engineering*, vol. 21, no. 4, pp. 1691–1716, 2020.

[5] S. C. Burmeister, D. Guericke, and G. Schryen, "A memetic nsga-ii for the multi-objective flexible job shop scheduling problem with real-time energy tariffs," *Flexible Services and Manufacturing Journal*, vol. 36, no. 4, pp. 1530–1570, 2024.

[6] D. Hutter, M. Hellwig, and T. Steinberger, "An interior-point genetic algorithm with restarts for flexible job shop scheduling problems," in *2024 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2024, pp. 01–09.

[7] J. Yang, H. Wang, and J. Xiong, "Evolutionary strategies with dual graph reinforcement learning for flexible job shop scheduling problem," in *2025 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2025, pp. 1–8.

[8] X.-l. Zheng and L. Wang, "A knowledge-guided fruit fly optimization algorithm for dual resource constrained flexible job-shop scheduling problem," *International Journal of Production Research*, vol. 54, no. 18, pp. 5554–5566, 2016.

[9] J. L. Andrade-Pineda, D. Canca, P. L. Gonzalez-R, and M. Calle, "Scheduling a dual-resource flexible job shop with makespan and due date-related criteria," *Annals of operations research*, vol. 291, no. 1, pp. 5–35, 2020.

[10] D. Fan and C. Wang, "A genetic algorithm for the dual resource constrained flexible job shop scheduling problem considering preparation times," in *2024 12th International Conference on Traffic and Logistic Engineering (ICTLE)*. IEEE, 2024, pp. 128–132.

[11] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[12] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 256–279, 2004.

[13] H. Tang, R. Chen, Y. Li, Z. Peng, S. Guo, and Y. Du, "Flexible job-shop scheduling with tolerated time interval and limited starting time interval based on hybrid discrete pso-sa: An application from a casting workshop," *Applied Soft Computing*, vol. 78, pp. 176–194, 2019.

[14] R. Zarrouk, I. E. Bennour, and A. Jemai, "A two-level particle swarm optimization algorithm for the flexible job shop scheduling problem," *Swarm Intelligence*, vol. 13, no. 2, pp. 145–168, 2019.

[15] H. Ding and X. Gu, "Improved particle swarm optimization algorithm based novel encoding and decoding schemes for flexible job shop scheduling problem," *Computers & Operations Research*, vol. 121, p. 104951, 2020.

[16] R. Balling, "The maximin fitness function; multi-objective city and regional planning," in *International Conference on Evolutionary multi-criterion optimization*. Springer, 2003, pp. 1–15.

[17] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[20] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *Mexican international conference on artificial intelligence*. Springer, 2004, pp. 688–697.

[21] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: Methods and applications*. Shaker Ithaca, 1999, vol. 63.