Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Solving the flexible job-shop scheduling problem based on the hierarchical collaborative evolution multi-objective artificial raindrop algorithm

Zhibo Zhai
Hebei University of Engineering
Shaoxuan Wang
3285844128@qq.com
Hebei University of Engineering https://orcid.org/0009-0004-5242-6481
Boyang Shi
Hebei University of Engineering
Liang Ma
Hebei University of Engineering
Haolong Wang
Hebei University of Engineering
Jiangchao Zhou
Hebei University of Engineering

Research Article

Keywords: flexible job shop scheduling, multi-objective optimization, artificial raindrop algorithm, hierarchical co-evolution, Pareto front

Posted Date: August 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10603642/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Solving the flexible job-shop scheduling problem based on the hierarchical collaborative evolution multi-objective artificial raindrop algorithm

Zhai Zhibo[1†], Wang Shaoxuan[2†], Shi Boyang[3], Ma Liang[4], Wang Haolong[5], Zhou Jiangchao[*]

[1]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.
[2]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.
[3]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.
[4]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.
[5]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.
[6*]Hebei University of Engineering, School of Mechanical and Equipment Engineering, 19 Taichi Road, Handan, 056038, Hebei, China.

Contributing authors: zhaizhibo@hebeu.com; 3285844128@qq.com; 1242006231@qq.com; 3037755145@qq.com; wanghaolong@525gmail.com; 2711582006@qq.com;
†These authors contributed equally to this work.

Abstract

In the multi-objective flexible job shop scheduling problem, makespan, total energy consumption, and total cost are mutually conflicting and therefore difficult to optimize simultaneously. To address these limitations, this study proposes a hierarchical co-evolutionary multi-objective artificial raindrop algorithm (HCMOARA). Latin hypercube sampling is first used to initialize the population and improve the uniformity of the initial solution distribution. The population is then partitioned into exploration, exploitation, and basic-update subpopulations according to nondominated rank and crowding distance. DE-best-2 global

search, SBX–RHC local search, and an adaptive flow-factor update strategy are applied to the three subpopulations, respectively, to enhance both global exploration and local exploitation. In addition, an external Pareto archive is employed to retain high-quality nondominated solutions. HCMOARA is evaluated on the Brandimarte benchmark instances MK01–MK10 and compared with several representative algorithms. The results demonstrate that HCMOARA achieves strong overall performance in terms of Pareto-set quality, makespan optimization, inverted generational distance (IGD), and hypervolume (HV). Ablation experiments further confirm the effectiveness of the proposed improvement strategies, indicating that HCMOARA is an effective approach for multi-objective flexible job shop scheduling.

Keywords: flexible job shop scheduling; multi-objective optimization; artificial raindrop algorithm; hierarchical co-evolution; Pareto front

1 Introduction

With the development of green and intelligent manufacturing, manufacturing systems are expected to achieve high production efficiency while simultaneously accounting for energy consumption, production cost, and environmental impact. As a critical component of manufacturing-system operations management, job shop scheduling directly affects makespan, equipment utilization, energy consumption, and processing cost. Therefore, the development of efficient scheduling optimization methods for green manufacturing is of substantial practical importance[1–5].

The flexible job shop scheduling problem (FJSP) is an important extension of the conventional job shop scheduling problem (JSP). Unlike the JSP, in which each operation must be processed on a predetermined machine, the FJSP allows each operation to be assigned to one of several eligible machines. Consequently, both operation sequencing and machine assignment must be determined. Although this flexibility improves the adaptability of scheduling plans, it also substantially enlarges the solution space and increases problem complexity[6–8]. Early studies on the FJSP primarily focused on minimizing makespan[9, 10]. In practical production, however, enterprises must often consider multiple conflicting objectives, including production efficiency, energy consumption, and processing cost. The multi-objective flexible job shop scheduling problem (MOFJSP) has therefore become an important research topic in intelligent manufacturing and green scheduling.

Existing methods for solving the MOFJSP mainly include classical multi-objective evolutionary algorithms and swarm-intelligence optimization algorithms. Classical multi-objective evolutionary algorithms provide fundamental frameworks such as nondominated sorting, diversity preservation, and decomposition-based optimization[11], whereas swarm-intelligence algorithms simulate natural behaviors or collective biological mechanisms and offer flexible search capabilities for complex scheduling problems[12, 13]. In recent years, nondominated swarm-intelligence algorithms such as particle swarm optimization (NSPSO), dung beetle optimization (NSOOA), osprey optimization (NSDBO), and crayfish optimization (NSCOA) have increasingly been

applied to multi-objective optimization, providing new solution approaches for the MOFJSP. Nevertheless, in complex multi-objective scheduling environments, these methods may converge slowly, become trapped in local optima, or produce unevenly distributed Pareto solution sets, making it difficult to maintain both convergence and diversity consistently.

The Artificial Raindrop Algorithm (ARA) is a swarm-intelligence optimization method inspired by natural rainfall. It performs the search by simulating raindrop falling, flowing, aggregation, and evaporation. Owing to its relatively simple structure and flexible search mechanism, ARA has potential for a range of optimization problems[14, 15]. Although the existing Multi-objective Artificial Raindrop Algorithm (MOARA) introduces a nondominated sorting mechanism to address multi-objective problems[16], it still suffers from an unevenly distributed initial population, fixed flow factors, a uniform individual evolution strategy, and an inadequate balance between global exploration and local exploitation. These limitations reduce its performance on the MOFJSP.

To address these issues, this study proposes a Hierarchical Co-evolutionary Multi-objective Artificial Raindrop Algorithm (HCMOARA) for a multi-objective FJSP that considers makespan, total energy consumption, and total cost. Latin hypercube sampling is first used to initialize the population and improve the uniformity of the initial solutions. Fast nondominated sorting and crowding distance are then used to partition the population into exploration, exploitation, and basic-update subpopulations, for which differentiated search strategies are designed. Specifically, the exploration subpopulation employs DE-best-2 to enhance global search, the exploitation subpopulation uses SBX-RHC to improve local refinement, and the basic-update subpopulation adopts adaptive flow factors to improve the balance of the search process. An external Pareto archive is also maintained to preserve high-quality nondominated solutions and strengthen the algorithm's multi-objective optimization capability.

2 Multi-objective Flexible Job Shop Scheduling Model

The flexible job shop scheduling problem (FJSP) is an important extension of the classical job shop scheduling problem (JSP). Unlike the conventional JSP, in which the processing machine for each operation is fixed, the FJSP allows each operation to be assigned flexibly to one machine selected from a set of eligible machines. This flexibility better reflects practical production but also produces a larger solution space and greater computational difficulty. To address the conflicts among completion time, energy consumption, and production cost in practical manufacturing, this study formulates a tri-objective FJSP model as the basis for evaluating the proposed scheduling algorithm.

2.1 Problem Description and Notation

Assume that the workshop contains n jobs to be processed, denoted by the set $J = \{J_1, J_2, \ldots, J_n\}$, and m machines, denoted by the set $M = \{M_1, M_2, \ldots, M_m\}$. Each job J_j consists of h_j operations, denoted by $O_{j1}, O_{j2}, \ldots, O_{jh_j}$.Unlike the conventional

job shop scheduling problem, the flexible job shop scheduling problem allows operation O_{jh} to be processed on any machine selected from its eligible-machine set $\Omega_{jh} \subseteq M$. Processing times, energy consumption, and processing costs vary across machines. The following assumptions are made to formulate the mathematical model:

1. All jobs and machines are available for processing at time zero.
2. Each job consists of multiple operations, which must be processed sequentially according to the prescribed technological route.
3. Each operation must be processed on exactly one machine selected from its eligible-machine set.
4. At any given time, each machine can process only one operation.
5. Once an operation starts, it cannot be interrupted.
6. No precedence constraints exist between different jobs.
7. A machine consumes processing energy while operating and idle energy while waiting.
8. Dynamic factors such as job transportation time, machine breakdowns, newly inserted jobs, and stochastic processing-time fluctuations are not considered.

The notation used in the mathematical model is summarized in Table 1.

Table 1 Notation used in the FJSP model

Symbol	Description
n	Total number of jobs.
m	Total number of machines.
i	Machine index, $i = 1, 2, \ldots, m$.
j, k	Job indices, $j, k = 1, 2, \ldots, n$.
h, l	Operation indices, $h = 1, 2, \ldots, h_j$.
J	Set of jobs, $J = \{J_1, J_2, \ldots, J_n\}$.
M	Set of machines, $M = \{M_1, M_2, \ldots, M_m\}$.
Ω_{jh}	Eligible-machine set for the h-th operation of job j.
T	Total number of operations of all jobs.
O_{jh}	The h-th operation of job j.
p_{ijh}	Processing time of operation O_{jh} on machine i.
S_{ijh}	Start time of operation O_{jh} on machine i.
C_{jh}	Completion time of operation O_{jh}.
C_j	Completion time of job j.
$C_{\max}$	Makespan.
P_i^w	Processing power of machine i.
P_i^s	Idle power of machine i.
c_i	Processing cost per unit time of machine i.
L	A sufficiently large positive number.
x_{ijh}	Binary machine-assignment variable; $x_{ijh} = 1$ if operation O_{jh} is assigned to machine i, and $x_{ijh} = 0$ otherwise.

Continued on the next page

Table 1 continued from the previous page

Symbol	Description
y_{jhkl}	Binary sequencing variable; $y_{jhkl} = 1$ if operation O_{jh} precedes operation O_{kl}, and $y_{jhkl} = 0$ otherwise.

2.2 Mathematical Model

Production efficiency, energy conservation, and economic cost are considered simultaneously. The three optimization objectives are the minimization of makespan, total energy consumption, and total cost. The objective functions are formulated as follows.

(1) Minimization of makespan

$$f_1 = \min\left(\max_{1\le j\le n} C_j\right). \tag{1}$$

Here, C_j denotes the completion time of job j.

(2) Minimization of total energy consumption

$$f_2 = \min\sum_{i=1}^{m}\left[P_i^w\sum_{j=1}^{n}\sum_{h=1}^{h_j}x_{ijh}p_{ijh} + P_i^s\left(C_{\max} - \sum_{j=1}^{n}\sum_{h=1}^{h_j}x_{ijh}p_{ijh}\right)\right]. \tag{2}$$

The first term in Eq. (2) represents the processing energy consumption, whereas the second term represents the idle energy consumption of the machines.

(3) Minimization of total cost

$$f_3 = \min\left(\sum_{j=1}^{n}\sum_{h=1}^{h_j}\sum_{i\in\Omega_{jh}}x_{ijh}p_{ijh}c_i\right). \tag{3}$$

The mathematical model is subject to the following constraints:

$$S_{ijh} + p_{ijh} \le S_{kl} + L\left(1 - y_{jhkl}\right), \tag{4}$$

$$\sum_{i\in\Omega_{jh}} x_{ijh} = 1, \tag{5}$$

$$S_{ijh} \ge C_{j(h-1)}, \tag{6}$$

$$C_{jh} = S_{ijh} + \sum_{i\in\Omega_{jh}} x_{ijh} * p_{ijh}, \tag{7}$$

$$S_{jh} \ge 0, \qquad \forall i = 1,2,\ldots,m, \quad j = 1,2,\ldots,n, \quad h = 1,2,\ldots,h_j. \tag{8}$$

Equation (4) represents the machine exclusivity constraint, which prevents two operations assigned to the same machine from being processed simultaneously. Equation (5) ensures that each operation is assigned to exactly one machine from its

eligible-machine set. Equation (6) represents the technological precedence constraint, indicating that a subsequent operation of the same job can start only after the previous operation has been completed. Equation (7) defines the completion time of each operation. Equation (8) ensures that the start time of every operation is nonnegative.

3 Improved Multi-objective Artificial Raindrop Algorithm

The original MOARA extends the artificial raindrop algorithm to multi-objective optimization by introducing nondominated sorting and an external archive. However, when applied to the MOFJSP, it still exhibits an uneven initial population distribution, a uniform individual-update strategy, and an insufficient balance between global exploration and local exploitation. To overcome these limitations, a hierarchical coevolutionary multi-objective artificial raindrop algorithm, HCMOARA, is proposed. Its core idea is to stratify the population according to nondominated rank and crowding distance and to assign differentiated evolutionary strategies to individuals at different levels. Specifically, the first group of individuals expands the search range, the second group intensifies local exploitation, and the remaining individuals retain the basic flow-based update mechanism of MOARA. Cooperative evolution among the subpopulations enables a better balance between convergence and diversity.

3.1 Encoding and Decoding

A two-segment continuous encoding scheme is used to represent a scheduling solution. The first segment encodes operation-sequencing priorities, and the second segment encodes machine-selection information. The operation sequence is obtained by sorting the continuous priority values in ascending order, whereas the machine-selection variables are mapped to integer indices within the eligible-machine sets. This representation links a continuous optimization algorithm to a discrete FJSP schedule and reduces the generation of infeasible solutions.

For machine-selection decoding, each continuous value is directly mapped to an integer index in the eligible-machine set, as follows:

$$\text{jobIndex} = \text{round}\left(\frac{1}{2\,\text{jobNum}}\,(x + \text{jobNum})\,(\text{candidateMachine} - 1) + 1\right). \tag{9}$$

The two encoding segments describe the operation-sequencing and machine-selection decisions, respectively, and jointly generate a complete schedule during decoding. Compared with direct discrete encoding, this scheme is better suited to continuous swarm-intelligence algorithms, reduces the occurrence of infeasible solutions, and improves the applicability of the algorithm to the FJSP.

3.2 Latin Hypercube Sampling Initialization

To improve the spatial coverage of the initial population, Latin hypercube sampling (LHS) is used for initialization. Compared with conventional random initialization,

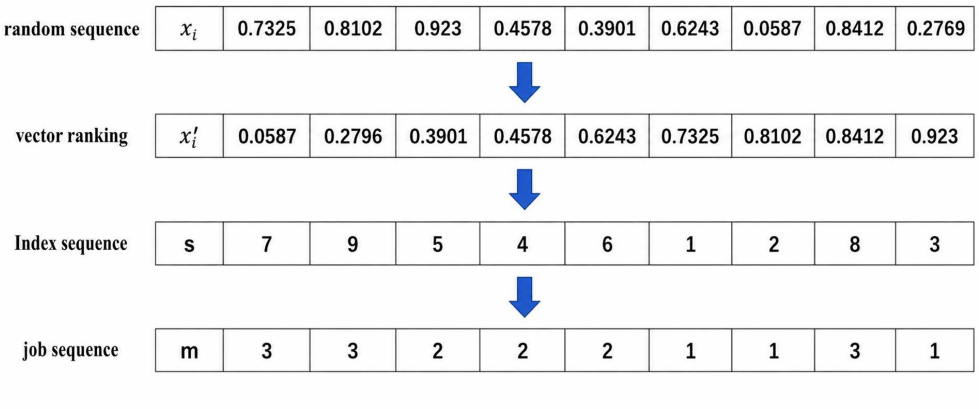


Fig. 1 Schematic illustration of the operation-sequencing decoding process

LHS stratifies the samples in each decision-variable dimension, thereby reducing clustering and uncovered regions and providing a more uniformly distributed set of initial solutions[17]. The mapping to the original variable range is given by:

$$x_{ij} = lb_j + (ub_j - lb_j)\,\frac{\pi_j(i) - 1 + u_{ij}}{N}. \tag{10}$$

where x_{ij} is the component of sample i in dimension j; lb_j and ub_j are the lower and upper bounds of dimension j, respectively; and u_{ij} is a random number generated in the $[0, 1]$ interval according to the LHS scheme.

3.3 Population Stratification Strategy

After population initialization, the fitness of each particle is evaluated. Fast nondominated sorting is then performed, and the first two fronts, Rank 1 and Rank 2, are identified as the best and second-best nondominated levels, respectively. The remaining individuals are assigned to lower levels. Crowding distance is calculated within each level, and the population is divided into three subpopulations according to rank and crowding distance: the exploration subpopulation, the exploitation subpopulation, and the basic-update subpopulation. The partitioning rule is:

$$\begin{cases} Pop_1 = Pop\left(Rank = 1 \;\&\; CD > \text{median}(CD)\right), \\ Pop_2 = Pop\left(Rank = 1 \;\&\; CD < \text{median}(CD)\right) \cup Pop(Rank = 2) \\ Pop_3 = Pop(Rank > 2). \end{cases} \tag{11}$$

The crowding distance CD is calculated using Eqs. (12) and (13):

$$CD_{im} = \frac{f_m\left(x_{i+1}\right) - f_m\left(x_{i-1}\right)}{f_m\left(x_{\max}\right) - f_m\left(x_{\min}\right)}. \tag{12}$$

$$CD_i = \sum_{m=1}^{M} CD_i^{(m)}. \tag{13}$$

CD_{im} denotes the crowding-distance contribution of individual i for objective m; f_m denotes the m-th objective. After normalization, the total crowding distance is obtained by summing the contributions over all objectives.

The exploration subpopulation consists of Rank 1 particles with relatively large crowding distances. A global competition strategy is used, and parents are selected from the entire population for crossover and mutation. The exploitation subpopulation comprises Rank 1 particles with relatively small crowding distances and all Rank 2 particles to balance exploration and exploitation. If Rank 1 contains only one particle, the exploitation subpopulation consists solely of all Rank 2 particles. During exploitation, parents are selected from within the subpopulation to perform variation and refine promising regions. The basic-update subpopulation contains particles with Rank > 2 and retains the original MOARA evolution mechanism.

3.4 Design of Hierarchical Evolutionary Operators

3.4.1 DE-best-2 Global Search Strategy (Pop_1)

To enhance the global exploration capability of the exploration subpopulation Pop_1, a crowding-distance-guided DE-best-2 strategy is introduced[18]. At each iteration, the nondominated solution with the largest crowding distance in the external Pareto archive is selected as the leader:

$$x_{\text{leader}}^t = \arg\max_{x \in RP} CD(x). \tag{14}$$

where RP denotes the external Pareto archive and $CD(x)$ is the crowding distance of solution x. Selecting a solution from a relatively sparse region of the Pareto front helps preserve population diversity and expands the search coverage. Based on the selected leader, the DE-best-2 mutation operator is formulated as

$$v_i^t = x_{\text{leader}}^t + F\left(x_{r_1}^t - x_{r_2}^t\right) + F\left(x_{r_3}^t - x_{r_4}^t\right). \tag{15}$$

where v_i^t is the mutant vector generated for the i-th individual in Pop_1; $x_{r_1}^t$, $x_{r_2}^t$, $x_{r_3}^t$, and $x_{r_4}^t$ are four mutually distinct individuals randomly selected from the current population; and F is the differential scaling factor, which is set to 0.5 in this study. The two difference vectors enhance population diversity, while the archive leader guides the search toward sparsely distributed nondominated regions.

3.4.2 SBX–RHC Local Hybrid Search Strategy (Pop_2)

To strengthen local exploitation, Pop_2 is equipped with a hybrid local search strategy combining Simulated Binary Crossover (SBX) and Random Hill Climbing (RHC)[19].

For the exploitation subpopulation, parent individuals $X_1(x_1^1, x_2^1, \ldots, x_n^1)$ and $X_2(x_1^2, x_2^2, \ldots, x_n^2)$ are selected from within the subpopulation, and offspring are generated as follows:

$$\begin{cases} c_i^1 = 0.5\left[(1+\beta)x_i^1 + (1-\beta)x_i^2\right], \\ c_i^2 = 0.5\left[(1-\beta)x_i^1 + (1+\beta)x_i^2\right]. \end{cases} \tag{16}$$

where c_i is the generated offspring and β is the spread factor, defined as follows:

$$\beta = \begin{cases} (\text{rand} \times 2)^{\frac{1}{1+\eta}}, & \text{rand} \leq 0.5, \\ \left(\frac{1}{2-\text{rand}\times 2}\right)^{\frac{1}{1+\eta}}, & \text{otherwise.} \end{cases} \tag{17}$$

where r is a random number in $[0, 1]$; η is the distribution index controlling the deviation of offspring from their parents. In this study, $\eta = 30$, which causes offspring to be generated primarily near their parents and thus strengthens local search.

After SBX is applied, Random Hill Climbing is used to perturb each offspring within a small neighborhood and further improve local exploitation. A random Gaussian perturbation is introduced, and the solution is iteratively updated according to dominance. The multi-objective greedy acceptance rule is defined as follows: let x be the current solution and x' be the candidate solution. If $x' \prec x$, then $x = x'$ is accepted.

$$\begin{cases} x' = x + \delta, \qquad \delta \sim N\left(0, \sigma^2\right), \\ x_{t+1} = \begin{cases} x', & x' \prec x_t, \\ x_t, & \text{else.} \end{cases} \end{cases} \tag{18}$$

where δ denotes the Gaussian perturbation and σ denotes the perturbation step size. Each offspring generated by SBX undergoes two random hill-climbing moves. This enables the exploitation subpopulation to conduct a more refined search near promising regions and improves the convergence quality of the Pareto solution set.

3.4.3 Multi-objective Artificial Raindrop Algorithm with Adaptive Flow Factors (Pop_3)

In the original MOARA, the flow factor c_1, c_2 is fixed, so the weights assigned to individual and social experience remain unchanged throughout the iterations, limiting dynamic adjustment of the search behavior. The adaptive flow factors are defined as follows:

$$\begin{cases} c_1(t) = c_{1,\text{start}} + (c_{1,\text{end}} - c_{1,\text{start}}) \dfrac{t}{T} \\ c_2(t) = c_{2,\text{start}} + (c_{2,\text{end}} - c_{2,\text{start}}) \dfrac{t}{T} \end{cases} \tag{19}$$

where t is the current iteration number and T is the maximum number of iterations.

After all three subpopulations have been updated using their respective strategies, the parent population and newly generated offspring are merged. Duplicate solutions are removed, and inferior particles are eliminated according to rank and crowding distance until the population size returns to N. The nondominated solutions are simultaneously added to the raindrop pool RP.

3.5 Algorithm Description and Flowchart

The pseudocode of HCMOARA is presented in Table **??** to describe the overall solution procedure clearly.

Table 2 Description of HCMOARA

Input:
Population size N; maximum number of iterations T; decision-space dimension D; lower and upper bounds LB and UB; scaling factor F; SBX distribution index η; maximum size of the external Pareto archive $RP_{\max}$.

Output:
External Pareto archive RP and the corresponding objective-function values.

Initialize population Pop using Latin hypercube sampling;
Decode each individual in Pop and calculate makespan, total energy consumption, and total cost;
Perform fast nondominated sorting on the initial population and calculate crowding distance;
Initialize the external Pareto archive RP;
Set the current iteration counter $t = 1$;
while $t \leq T$ do
Perform fast nondominated sorting on the current population Pop and obtain the nondominated rank $Rank$ of each individual;
Calculate the crowding distance CD of each individual;
Divide the population into three subpopulations according to $Rank$ and CD: Exploration subpopulation Pop_1; Exploitation subpopulation Pop_2; Basic-update subpopulation Pop_3;
Apply the DE-best-2 global search strategy to the exploration subpopulation Pop_1 to generate offspring Off_1;
Apply the SBX–RHC local hybrid search strategy to the exploitation subpopulation Pop_2 to generate offspring Off_2;
Apply the adaptive MOARA update strategy to the basic-update subpopulation Pop_3 to generate offspring Off_3;
Merge the parent and offspring populations;
Perform fast nondominated sorting and crowding-distance calculation on Pop_{all};
Conduct environmental selection according to nondominated rank and crowding distance, retaining N individuals as the next population Pop;
Add the individuals in the first nondominated front to the external Pareto archive RP;
Remove duplicate solutions from RP;
If the size of RP exceeds $RP_{\max}$, remove individuals from overcrowded regions according to crowding distance;
Update the iteration counter: $t = t + 1$;
end while
Output the external Pareto archive RP.

The overall workflow of HCMOARA is shown in Figure 2. The algorithm first sets the parameters, initializes the population, and evaluates fitness. The population is then stratified using nondominated sorting and crowding distance. Differentiated evolutionary strategies are applied to the three subpopulations to generate offspring.

The parent and offspring populations are merged, and environmental selection is performed to form the next generation while the external Pareto archive is updated simultaneously. When the maximum number of iterations is reached, the final set of Pareto-nondominated solutions is output.

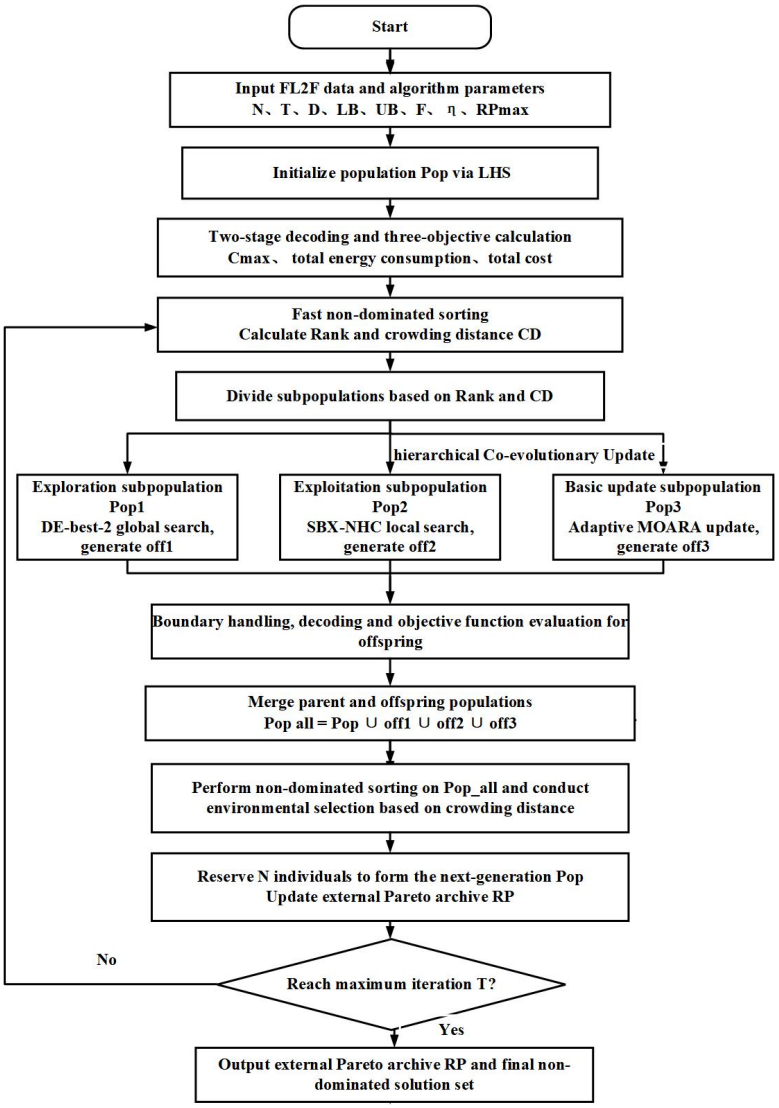


Fig. 2 Flowchart of HCMOARA

4 Simulation Experiments and Results

4.1 Experimental Setup

Ten flexible job shop scheduling instances, MK01–MK10, from the Brandimarte benchmark set are used in the simulation experiments. These instances contain different numbers of jobs, machines, and operations and therefore represent the solution difficulty of FJSPs at different scales.

The multi-objective FJSP model minimizes makespan, total energy consumption, and total cost. Because these objectives conflict with one another, Pareto-nondominated solution sets are used to characterize the trade-offs among alternative schedules.

All experiments were conducted in MATLAB R2023a under Windows 10 on a computer equipped with an AMD Ryzen 5 9600X CPU and an NVIDIA GeForce RTX 5060. To ensure a fair comparison, all algorithms used the same population size, maximum number of iterations, and termination criterion. The population size and maximum number of iterations were both set to 200. To reduce the influence of stochastic variation, the mean and standard deviation of IGD and HV were calculated based on 30 independent runs for each algorithm on each benchmark instance. The tri-objective optimization results, Pareto solution sets, convergence curves, IGD values, HV values, and representative scheduling Gantt charts were recorded.

To evaluate the effectiveness of HCMOARA, the original MOARA, NSPSO, NSOOA, NSDBO, and NSCOA were selected as comparison algorithms. Algorithm parameters were set according to the original publications or commonly used settings, and the principal experimental parameters were kept consistent to minimize the effect of parameter differences.

4.2 Comparative Results and Discussion

4.2.1 Comparison of Tri-objective Optimization Results

Table 3 presents the optimal solution results of each algorithm on 10 MK cases when the focus is on minimizing the maximum completion time (Makespan).

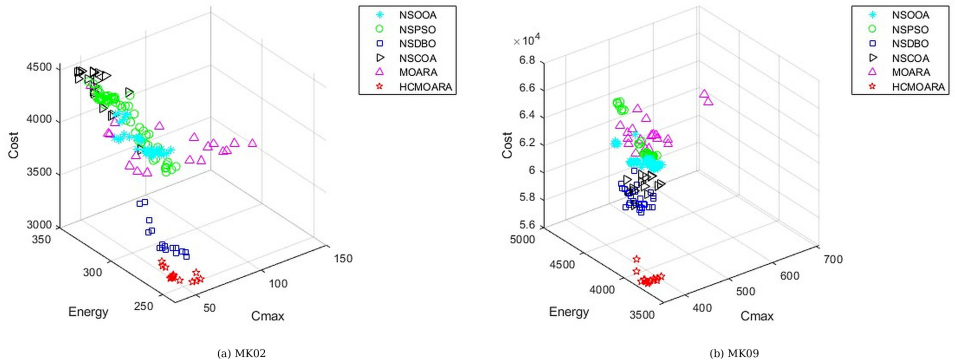
Table 3 Solution Results of Each Algorithm in the MK Test Cases

Instance	Objective	NSPSO	NSOOA	NSDBO	NSCOA	MOARA	HCMOARA
	T	44	43	42	44	44	**42**
MK01	E	336.94	342.56	319.48	338.94	335.93	**325.56**
	C	4525	4607	4255	4444	4382	**4328**
	T	42	37	39	43	38	**33**
MK02	E	306.34	306.86	305.1	338.9	344.68	**256.75**
	C	3993	4167	4150	4703	4732	**3379**
	T	221	214	214	223	243	**204**
MK03	E	2211.94	2210.71	1732.7	2088.15	2085.41	**1494.71**
	C	34794	35640	28821	33117	33849	**24950**
	T	76	74	71	73	76	72
MK04	E	612.83	580.4	567.38	542.16	590.45	559.57
	C	10104	10414	10088	9703	10446	9105
	T	185	178	181	182	187	**177**
MK05	E	1401.27	1389.64	1379.39	1407.34	1416.7	**1387.78**
	C	19418	18963	18928	19254	19287	**18837**
	T	118	104	102	121	119	**82**
MK06	E	1076.94	1038.89	866.02	1049.33	1059.73	**703.5**
	C	16330	15010	12251	15383	14417	**9646**
	T	193	184	181	191	185	**162**
MK07	E	1572.04	1437.19	1322.54	1532.39	1413.16	**1263.47**
	C	21937	20268	18861	21277	20168	**17598**
	T	530	523	527	526	530	**523**
MK08	E	4260.04	4046.02	4163.03	4213.74	4319.7	**4094.09**
	C	64765	62224	62390	63363	63740	**62182**
	T	396	362	385	386	415	**341**
MK09	E	4222.55	4323.7	4322.82	4065.98	4494.65	**3956.65**
	C	63933	65476	63655	60507	66097	**59382**
	T	335	322	327	335	348	**302**
MK10	E	4068.75	4001.27	4012.1	4008.48	4159.27	**3598.68**
	C	58364	57491	57547	57707	56606	**49442**

As shown in Table 3, HCMOARA obtained the uniquely best makespan on seven instances, namely MK02, MK03, MK05, MK06, MK07, MK09, and MK10, and tied for the best result on MK01 and MK08. In particular, HCMOARA maintained a relatively small makespan on medium- and large-scale instances such as MK06, MK09, and MK10, indicating that hierarchical co-evolution and differentiated search strategies effectively strengthen both global exploration and local exploitation. HCMOARA also achieved relatively low total energy consumption and total cost on most instances, demonstrating that it can reduce the scheduling cycle while accounting for energy efficiency and economic performance.

4.2.2 Analysis of Pareto Solution Sets

The outcome of a multi-objective flexible job shop scheduling problem is generally not a single optimal solution but a set of Pareto-nondominated solutions. To visually assess the multi-objective search capability of HCMOARA, representative Brandimarte instances were selected, and the Pareto solution sets obtained by the different algorithms were compared in three-dimensional objective space, as shown in Figure 3.



(a) MK02 (b) MK09

Fig. 3 Three-dimensional visualization of the solution sets obtained by the algorithms

For the small-scale MK02 instance and large-scale MK09 instance, the HCMOARA solution sets are generally located closer to the region of lower values for all three objectives, indicating favorable convergence. The solutions are also relatively dispersed and cover different compromise schedules, suggesting that HCMOARA can preserve solution diversity. The Pareto-set distributions therefore demonstrate that HCMOARA can effectively address the conflicts among makespan, total energy consumption, and total cost in the multi-objective FJSP.

4.2.3 Analysis of Iterative Convergence Curves

To further analyze the search performance of different algorithms during the iterative process, this paper selects MK02 and MK09 as representative examples, and compares the average iterative convergence curves of each algorithm on the maximum completion time objective. The results are shown in Figure 4. Among them, MK02 is a smaller-scale example, and MK09 is a larger-scale example, which can to some extent reflect the convergence performance of different algorithms under different problem scales.

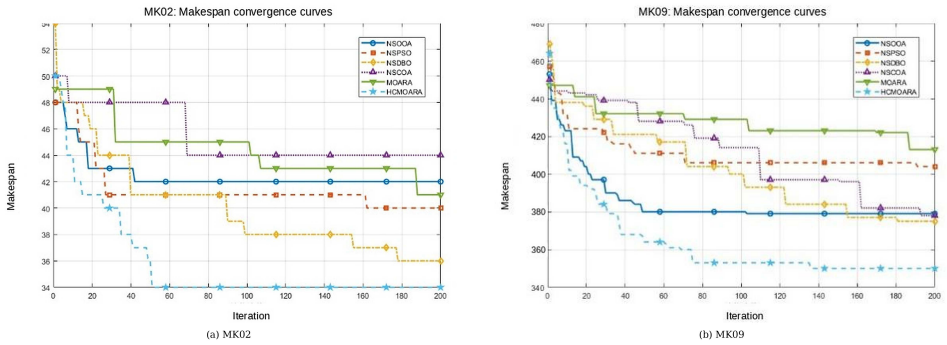


Fig. 4 Makespan convergence curves of the algorithms

Figure 4 shows that HCMOARA exhibits favorable iterative convergence on both MK02 and MK09. By contrast, the comparison algorithms improve only slightly during the later iterations, and some show search stagnation, resulting in poorer final convergence than HCMOARA. These results indicate that Latin hypercube sampling, DE-best-2 global search, SBX–RHC local search, and the adaptive flow factors effectively improve convergence speed and optimization capability, further demonstrating the effectiveness of HCMOARA for the multi-objective flexible job shop scheduling problem.

4.2.4 Analysis of IGD and HV

To quantitatively evaluate the quality of the Pareto solution sets obtained by the different algorithms, inverted generational distance (IGD) and hypervolume (HV) were calculated for the MK01–MK10 instances[20, 21]. IGD measures the distance between an algorithm's nondominated solution set and the reference Pareto front; a smaller IGD indicates that the obtained set is closer to the reference front. HV measures the region of objective space dominated by the nondominated solution set; a larger HV indicates better overall quality of the Pareto solution set.

The three objectives considered in this study – makespan, total energy consumption, and total cost have markedly different numerical scales. If IGD and HV are calculated directly from the original objective values, objectives with larger magnitudes may exert a disproportionate influence on the evaluation. Therefore, min–max normalization is applied to the three objectives before IGD and HV are calculated:

$$f_i^{norm} = \frac{f_i - f_i^{\min}}{f_i^{\max} - f_i^{\min}}. \tag{20}$$

In the formula, f_i represents the original value of the i-th objective, and $f_i^{\min}$ and $f_i^{\max}$ represent the minimum and maximum values of the non-dominated solutions obtained by all algorithms for the i-th objective in the same test case, respectively. f_i^{norm} represents the normalized objective value. In this paper, the same maximum

and minimum values are used for normalization within the same MK case to ensure the consistency of evaluation criteria among different algorithms.

IGD and HV are calculated as follows:

$$IGD(P,Q) = \frac{1}{|P|}\sum_{v\in P} d(v,Q). \tag{21}$$

$$HV(Q,r) = \text{Volume}\left(\bigcup_{q\in Q}[q_1,r_1]\times[q_2,r_2]\times\cdots\times[q_m,r_m]\right). \tag{22}$$

In the formula, P represents the point set uniformly distributed on the actual Pareto surface, $|p|$ represents the number of point sets distributed on the actual Pareto surface, $d(v,Q)$ represents the minimum Euclidean distance from an individual in P to the population Q, and $r = r_1, r_2, r_3$ represents the reference point.

Table 4 The average (standard deviation) results of IGD for each algorithm in the MK test cases

Instance	NSPSO	NSOOA	NSDBO	NSCOA	MOARA	HCMOARA
	0.8252	0.2528	0.0147	0.1415	0.1914	0.0361
MK01	(0.0935)	(0.0697)	(0.0071)	(0.0777)	(0.0439)	(0.0061)
	1.0293	0.4615	0.3907	0.6739	0.5613	**0.0000**
MK02	(0.1203)	(0.0701)	(0.0625)	(0.1166)	(0.0486)	**(0.0000)**
	1.1498	0.6148	0.2654	0.6585	0.6923	**0.0000**
MK03	(0.0461)	(0.0787)	(0.0642)	(0.0822)	(0.0693)	**(0.0000)**
	0.7379	0.2371	0.0338	0.1797	0.2297	0.0697
MK04	(0.0623)	(0.0393)	(0.0207)	(0.0702)	(0.0282)	(0.0181)
	0.7454	0.0184	0.0310	0.2093	0.2029	0.0675
MK05	(0.0771)	(0.0116)	(0.0134)	(0.0349)	(0.0519)	(0.0067)
	1.2481	0.7218	0.5122	0.9431	0.7902	**0.0000**
MK06	(0.0696)	(0.0437)	(0.0695)	(0.0498)	(0.0650)	**(0.0000)**
	1.2141	0.6662	0.1182	0.6358	0.3861	**0.0003**
MK07	(0.0620)	(0.0778)	(0.0290)	(0.0845)	(0.0733)	**(0.0010)**
	0.8890	0.0615	0.1081	0.2795	0.2874	**0.0410**
MK08	(0.1095)	(0.0754)	(0.0370)	(0.0673)	(0.0688)	**(0.0164)**
	0.9723	0.5089	0.4438	0.6000	0.7095	**0.0000**
MK09	(0.0654)	(0.0397)	(0.0346)	(0.0777)	(0.0722)	**(0.0000)**
	0.6903	0.2423	0.1661	0.3218	0.4439	**0.0762**
MK10	(0.1176)	(0.1307)	(0.0991)	(0.1549)	(0.1309)	**(0.0856)**

Table 4 shows that HCMOARA achieved the best IGD on seven instances. Its IGD was 0 on MK02, MK03, MK06, and MK09, indicating that, under the reference Pareto front constructed in this study, the solution sets obtained by HCMOARA covered the corresponding reference points. HCMOARA also produced small IGD values on larger instances such as MK07, MK08, and MK10, demonstrating that the proposed hierarchical co-evolutionary strategy retains strong search capability in complex search spaces.

Table 5 The average (standard deviation) results of HV for each algorithm in the MK test cases

Instance	NSPSO	NSOOA	NSDBO	NSCOA	MOARA	HCMOARA
	0.1058	0.6683	1.1359	0.8620	0.7492	**1.1478**
MK01	(0.0478)	(0.1280)	(0.0307)	(0.1800)	(0.0835)	**(0.0334)**
	0.1047	0.5761	0.7199	0.3506	0.5167	**1.3090**
MK02	(0.0454)	(0.0736)	(0.0583)	(0.0741)	(0.0443)	**(0.0104)**
	0.0500	0.3989	0.8351	0.3396	0.3306	**1.2958**
MK03	(0.0185)	(0.0707)	(0.0947)	(0.0715)	(0.0280)	**(0.0271)**
	0.1649	0.7039	1.0747	0.8240	0.6983	**1.1207**
MK04	(0.0384)	(0.0911)	(0.0453)	(0.1321)	(0.0437)	**(0.0274)**
	0.1272	1.1024	1.0840	0.7516	0.7172	1.0679
MK05	(0.0379)	(0.0440)	(0.0522)	(0.0738)	(0.0831)	(0.0457)
	0.0454	0.3353	0.5200	0.1775	0.2795	**1.3202**
MK06	(0.0175)	(0.0255)	(0.0775)	(0.0302)	(0.0516)	**(0.0039)**
	0.0191	0.2921	1.0052	0.3061	0.5613	**1.2267**
MK07	(0.0094)	(0.0611)	(0.0514)	(0.0721)	(0.0928)	**(0.0139)**
	0.1052	1.1042	1.0200	0.7282	0.6906	**1.1882**
MK08	(0.0566)	(0.1202)	(0.0499)	(0.0736)	(0.0830)	**(0.0646)**
	0.1354	0.5678	0.6234	0.4309	0.3599	**1.3188**
MK09	(0.0396)	(0.0497)	(0.0519)	(0.0728)	(0.0653)	**(0.0047)**
	0.1703	0.7351	0.9235	0.6047	0.4657	**1.1589**
MK10	(0.0414)	(0.0783)	(0.1241)	(0.1254)	(0.0914)	**(0.1237)**

As shown in Table 5, HCMOARA exhibits an even more pronounced advantage in terms of HV. It achieved the best HV on nine of the ten benchmark instances and was only slightly inferior to NSOOA on MK05. Because a larger HV indicates broader coverage of objective space by the nondominated solution set, these results demonstrate that the Pareto sets obtained by HCMOARA have favorable distribution and coverage.

4.3 Ablation Study

To quantify the contribution of each improvement strategy in HCMOARA, an ablation study was conducted by removing the strategies individually. MOARA denotes the original multi-objective artificial raindrop algorithm without any improvement. HCMOARA-noLHS replaces Latin hypercube sampling with random initialization. HCMOARA-noAF removes the adaptive flow factors and uses fixed flow factors. HCMOARA-noDE removes the DE-best-2 global search strategy from the exploration subpopulation. HCMOARA-noSBX removes the SBX-RHC local search strategy from the exploitation subpopulation. Full-HCMOARA denotes the complete algorithm containing all improvement strategies. All values reported in Table 6 are averages.

Table 6 Comparison of ablation results on representative instances

Algorithm	Instance	LHS	Hierarchy	AF	DE	SBX	$C_{\max}$	Energy	Cost	IGD	HV
	Mk02						44	288.89	3747	0.7034	0.2493
MOARA	MK04	×	×	×	×	×	78.9	558.65	7997.7	0.3811	0.3662
	MK09						420.2	4234.27	62077.3	0.8727	0.2096
	Mk02						34.7	239.22	3111.2	0.0523	1.1917
noLHS	MK04	×	✓	✓	✓	✓	74.2	492.54	7404.1	0.0947	0.9972
	MK09						354.5	3830.7	56946	0.0755	1.1369
	Mk02						34.7	241.77	3142.5	0.0786	1.1435
noAF	MK04	✓	✓	×	✓	✓	74.8	494.06	7421.4	0.0918	0.9928
	MK09						358.7	3843.53	57092.6	0.0856	1.1086
	Mk02						34.9	238.9	3104.5	0.052	1.2031
noDE	MK04	✓	✓	✓	×	✓	74.3	494.36	7413.5	0.0863	0.9986
	MK09						354	3813.28	56685.8	0.0548	1.205
	Mk02						33.7	238.77	3119.2	0.0532	1.2065
noSBX	MK04	✓	✓	✓	✓	×	75.1	493.95	7344.4	0.0793	0.9934
	MK09						352.9	3839.98	56903.4	0.0806	1.1315
	Mk02						33.5	236.3	3058.5	0.0256	1.2748
Full	MK04	✓	✓	✓	✓	✓	73.5	492.37	7278.3	0.0699	1.0063
	MK09						351.2	3807.77	56269.8	0.0446	1.2442

Table 6 shows that Full-HCMOARA achieved strong overall performance on the three representative instances MK02, MK04, and MK09. Removing LHS, AF, DE, or SBX-RHC degraded at least some performance indicators. HCMOARA-noLHS produced poorer IGD and HV values than Full-HCMOARA, indicating that Latin hypercube sampling improves the initial population distribution and subsequent search quality. HCMOARA-noAF was inferior to the complete algorithm, suggesting that adaptive flow factors regulate the search process and improve the balance between exploration and exploitation. HCMOARA-noDE yielded poorer IGD and HV values than Full-HCMOARA, demonstrating that DE-best-2 strengthens global exploration.

4.4 Analysis of Representative Scheduling Gantt Charts

To visualize the schedules obtained by HCMOARA, Figures 5 and 6 present the Gantt charts for MK02 and MK09, respectively, under the minimum-makespan objective. The vertical axis represents machines and the horizontal axis represents processing time. Rectangles of the same color belong to the same job, and the label "J1-1" denotes the first operation of Job 1.

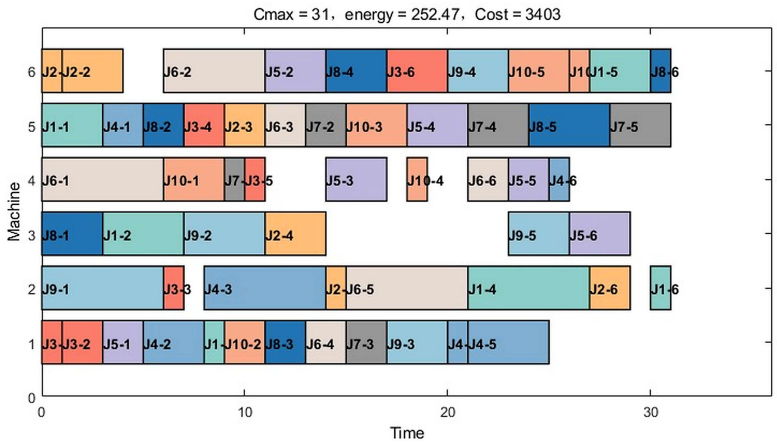


Fig. 5 Gantt chart obtained by HCMOARA for MK02

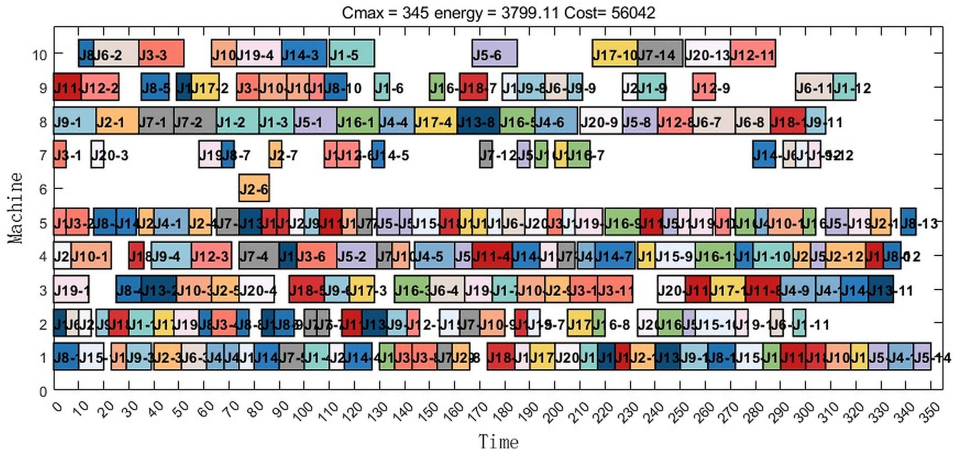


Fig. 6 Gantt chart obtained by HCMOARA for MK09

The Gantt charts show that Machine 3 in MK02 and Machine 6 in MK09 have relatively low utilization and substantial idle time, possibly because only a limited number of operations are eligible for processing on these machines. Overall, machine loads are reasonably balanced and successive operations are arranged compactly, demonstrating the effectiveness of HCMOARA in machine assignment and operation sequencing.

5 Conclusions

This study addressed the conflicting objectives of makespan, total energy consumption, and total cost in the multi-objective flexible job shop scheduling problem by

proposing HCMOARA, a hierarchical co-evolutionary multi-objective artificial raindrop algorithm. Latin hypercube sampling improves the uniformity of the initial population. Fast nondominated sorting and crowding distance are then used to divide the population into exploration, exploitation, and basic-update subpopulations, which employ DE-best-2 global search, SBX–RHC local hybrid search, and adaptive MOARA updates, respectively, to balance global exploration and local exploitation. Simulation experiments on the Brandimarte MK01–MK10 benchmark instances, together with comparisons against NSPSO, NSOOA, NSDBO, NSCOA, and the original MOARA, show that HCMOARA obtains superior schedules on most instances and provides favorable convergence and distribution in terms of IGD and HV. Overall, the proposed algorithm is effective for multi-objective green flexible job shop scheduling and provides a feasible approach to multi-objective scheduling optimization in complex manufacturing systems. The present study is limited to a static scheduling environment; future work may consider dynamic factors such as machine breakdowns, job arrivals, and processing-time disturbances to improve applicability in practical production settings.

References

[1] Li J, Li H, He P, Xu L, He K, Liu S. Flexible job shop scheduling optimization for green manufacturing based on improved multi-objective wolf pack algorithm. *Applied Sciences*. 2023;13(14):8535. https://doi.org/10.3390/app13148535.

[2] Shi J, Liu W, Yang J. An enhanced multi-objective evolutionary algorithm with reinforcement learning for energy-efficient scheduling in the flexible job shop. *Processes*. 2024;12(9):1976. https://doi.org/10.3390/pr12091976.

[3] Wang Z, He M, Wu J, Chen H, Cao Y. An improved MOEA/D for low-carbon many-objective flexible job shop scheduling problem. *Computers & Industrial Engineering*. 2024;188:109926. https://doi.org/10.1016/j.cie.2024.109926

[4] Gong G, Tang J, Huang D, Luo Q, Zhu K, Peng N. Energy-efficient flexible job shop scheduling problem considering discrete operation sequence flexibility. *Swarm and Evolutionary Computation*. 2024;84:101421. https://doi.org/10.1016/j.swevo.2023.101421.

[5] Piroozfard H, Wong KY, Wong WP. Minimizing total carbon footprint and total late work criterion in flexible job shop scheduling by using an improved multi-objective genetic algorithm. *Resources, Conservation and Recycling*. 2018;128:267–283. https://doi.org/10.1016/j.resconrec.2016.12.001.

[6] Chaudhry IA, Khan AA. A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research*. 2016;23(3):551–591. https://doi.org/10.1111/itor.12199.

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

[8] Brandimarte P. Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*. 1993;41:157–183. https://doi.org/10.1007/BF02023073.

[9] Johnson SM. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*. 1954;1(1):61–68. https://doi.org/10.1002/nav.3800010110.

[10] Adams J, Balas E, Zawack D. The shifting bottleneck procedure for job shop scheduling. *Management Science*. 1988;34(3):391–401. https://doi.org/10.1287/mnsc.34.3.391.

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

[12] Coello Coello CA, Toscano Pulido G, Salazar Lechuga M. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*. 2004;8(3):256–279. https://doi.org/10.1109/TEVC.2004.826067.

[13] Zhang G, Shao X, Li P, Gao L. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Computers & Industrial Engineering*. 2009;56(4):1309–1318. https://doi.org/10.1016/j.cie.2008.07.021.

[14] Li W, Wang L, Jiang Q, Hei X, Wang B. Cloud particles evolution algorithm. *Mathematical Problems in Engineering*. 2015;2015:434831. https://doi.org/10.1155/2015/434831.

[15] Li W, Wang L, Yao Q, Jiang Q, Yu L, Wang B, Hei X. Cloud particles differential evolution algorithm: A novel optimization method for global numerical optimization. *Mathematical Problems in Engineering*. 2015;2015:497398. https://doi.org/10.1155/2015/497398.

[16] Jiang Q, Wang L, Hei X, Yu G, Lin Y, Lu X. MOEA/D-ARA+SBX: A new multi-objective evolutionary algorithm based on decomposition with artificial raindrop algorithm and simulated binary crossover. *Knowledge-Based Systems*. 2016;107:197–218. https://doi.org/10.1016/j.knosys.2016.06.007.

[17] McKay MD, Beckman RJ, Conover WJ. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*. 1979;21(2):239–245. https://doi.org/10.1080/00401706.1979.10489755.

[18] Storn R, Price K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*. 1997;11(4):341–359. https://doi.org/10.1023/A:1008202821328.

[19] Deb K, Agrawal RB. Simulated binary crossover for continuous search space. *Complex Systems*. 1995;9(2):115–148.

[20] Li M, Yao X. Quality evaluation of solution sets in multiobjective optimisation: A survey. *ACM Computing Surveys*. 2019;52(2):26. https://doi.org/10.1145/3300148.

[21] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*. 1999;3(4):257–271. https://doi.org/10.1109/4235.797969.

Statements and Declarations

The authors declare that no funds, grants, or other financial support were received during the preparation of this manuscript. The authors have no relevant financial or non-financial competing interests to disclose. All authors contributed to the conception and design of the research. Material preparation, data collection, and analysis were performed by Zhai Zhibo , Wang Shaoxuan , Shi Boyang and Ma Liang. The first draft of the manuscript was written by Zhou Jiangchao and Wang Haolong, and all authors commented on previous versions of the manuscript. All authors have read and approved the final manuscript.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- usermanual.pdf