Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(12): 21467–21498.
DOI: 10.3934/mbe.2023950
Received: 19 September 2023
Revised: 19 November 2023
Accepted: 22 November 2023
Published: 05 December 2023

Research article

An improved memetic algorithm to solve the energy-efficient distributed flexible job shop scheduling problem with transportation and start-stop constraints

Yifan Gu, Hua Xu*, Jinfeng Yang and Rui Li

School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China

* **Correspondence:** Email: xuhua@jiangnan.edu.cn.

Abstract: In the current global cooperative production environment, modern industries are confronted with intricate production plans, demanding the adoption of contemporary production scheduling strategies. Within this context, distributed manufacturing has emerged as a prominent trend. Manufacturing enterprises, especially those engaged in activities like automotive mold production and welding, are facing a significant challenge in managing a significant amount of small-scale tasks characterized by short processing times. In this situation, it becomes imperative to consider the transportation time of jobs between machines. This paper simultaneously considers the transportation time of jobs between machines and the start-stop operation of the machines, which is the first time to our knowledge. An improved memetic algorithm (IMA) is proposed to solve the multi-objective distributed flexible job shop scheduling problem (MODFJSP) with the goal of minimizing maximum completion time and energy consumption. Then, a new multi-start simulated annealing algorithm is proposed and integrated into the IMA to improve the exploration ability and diversity of the algorithm. Furthermore, a new multiple-initialization rule is designed to enhance the quality of the initial population. Additionally, four improved variable neighborhood search strategies and two energy-saving strategies are designed to enhance the search ability and reduce energy consumption. To verify the effectiveness of the IMA, we conducted extensive testing and comprehensive evaluation on 20 instances. The results indicate that, when faced with the MODFJSP, the IMA can achieve better solutions in almost all instances, which is of great significance for the improvement of production scheduling in intelligent manufacturing.

Keywords: multi-objective optimization; memetic algorithm; transportation time; energy-saving strategy; start-stop constraint

1. Introduction

With the development of economic globalization, scheduling problems are pervasive in a growing number of application domains, which is a critical decision-making issue in the manufacturing system. The classical job shop scheduling problem (JSP) is widely studied because it can model various situations in the scheduling process [1, 2]. However, as the flexibility of processing in the shop continues to increase, each machine is becoming increasingly able to process multiple operations. Hence, this problem has been named the flexible JSP (FJSP) [3].

In the context of advancing global economic integration, contemporary factory operational scheduling introduces new imperatives for enhanced flexibility. The symbol of this transformation is the gradual shift from traditional single-factory manufacturing to distributed multi-factory manufacturing [4]. Flexible distributed factory scheduling will allocate jobs reasonably for factories and machines to reduce production costs. It is worth noting that the complexity of distributed scheduling not only lies in job sequencing and machine selection, but it also involves the factory allocation of jobs [5], which makes solving multi-objective distributed FJSPs (MODFJSP) a challenging task [6].

With the acceleration of industrialization processes, the increasing energy demand in industrial production has become an important issue of global concern in contemporary times [7]. Driven by the imperatives of environmental consciousness and the looming specter of energy scarcity, an expanding array of enterprises are now incorporating energy consumption considerations into their production strategies. It is important to emphasize that factory operational scheduling is emerging to play a pivotal role in this evolving landscape [8]. Therefore, for manufacturing enterprises, excellent scheduling also needs to balance energy consumption, taking into account energy conservation and sustainability.

Flexible systems are widely used in various industries, such as aerospace, cutting tools and molds, automotive, medical, optical- and engineering machinery. However, today's constantly changing customer demands and increasingly fierce market competition conflict with inefficient traditional machining methods. This situation is becoming increasingly severe in small- and medium-sized factories. Therefore, the manufacturing industry has gradually formed a market-oriented demand for "multi-variety, small batch" production, characterized by short processing times for many operations. In this production situation, the time spent transporting operations between machines becomes very important, as it has a significant impact on the entire production cycle and process, and this is precisely what many other studies have overlooked.

This paper presents an improved memetic algorithm (IMA), designed to address the MODFJSP. We propose a multi-start simulated annealing algorithm (MSSA) that selects the starting point by judging the similarity of chromosomes, and, for each selected point, we implement three perturbation strategies. Based on different acceptance probabilities, we choose whether to accept the new solution. This strategy effectively enhances the exploration ability of the IMA, enriches the diversity of the population and avoids the problem of the algorithm falling into local optima too early. Moreover, the proposed hybrid initialization rule, denoted as MIR, along with four efficient neighborhood structures, significantly enriches both the quality and diversity of the population. We conducted extensive testing and a comprehensive evaluation on Mk01-10 and DP01-10 instances, demonstrating the superior efficacy of the IMA across the majority of scenarios.

This paper builds upon the foundations of the multi-objective energy-efficient DFJSP (MOEDFJSP), integrating the transport duration between different machines within the same factory, the time needed for machine startup and shutdown and energy consumption, all simultaneously. This holistic approach aims to align more closely with practical demands. The contributions of this paper encompass the following facets:

1) A multi-objective distributed flexible job shop scheduling model with transportation time and start-stop constraints is established.

2) A new MSSA is proposed and integrated into the IMA to improve the exploration ability and diversity of the algorithm.

3) Four new hybrid initialization rules have been designed to improve the quality and diversity of the initial population.

4) Four efficient neighborhood structures have been designed to enhance the exploration ability for optimal solutions.

5) Two effective energy-saving strategies have been designed to significantly reduce energy consumption.

6) Comparative experiments of different combinations were performed on 20 instances to describe the improved performance of the IMA.

The remaining sections of this paper are organized as follows. Section 2 reviews the relevant research work on the MODFJSP in recent years. In Section 3, fundamental concepts underpinning the FJSP are elucidated. Section 4 delineates the developed MODFJSP model. Section 5 describes the overall framework and details of the IMA. Experimental results and analysis are given in Section 6. Finally, the conclusions and future research work are given in Section 7.

2. Literature review

The production scheduling problem stands as a pivotal concern within manufacturing systems to facilitate effective production planning and the enhancement of overall production efficiency. Among the array of scheduling challenges, the JSP looms large as one of the most prevalent and practical [9]. As factory machinery's demand for versatility surges, the FJSP emerges, characterized by each operation's potential to be accommodated by one or multiple machines. This extends the classical JSP into a realm more attuned to the discrete manufacturing environment [10].

In the domain of the FJSP, there are two key sub-problems: machine selection and process sequencing. Researchers have harnessed the power of meta-heuristic algorithms like the genetic algorithm (GA) [11, 12], simulated annealing (SA) [13], particle swarm optimization (PSO) [14] and ant colony optimization [15] to grapple with these intricacies. Additionally, other meta-heuristic algorithms, such as the artificial immune system-based algorithm [16], the artificial bee colony algorithm [17] and imperialist competitive algorithms [18], are also used. One particularly notable advancement of the GA is the memetic algorithm (MA), pioneered by Moscato [19] to mirror the evolution of human civilization. This innovation ingeniously fuses global and local search strategies. Across scholarly efforts, the MA has gained prominence in addressing FJSP-related conundrums. For instance, Phu-ang and Thammano [20] introduced an MA founded on bee marriages to sidestep local optima. Zhang et al. [21] leveraged an MA to resolve multi-objective FJSPs, incorporating worker flexibility. Li et al. [22] proposed a knowledge-driven MA for distributed green flexible scheduling. Lu et al. [23] proposed a knowledge-based multi-objective memetic optimization algorithm to solve the sustainable distributed permutation flow-shop scheduling problem with non-identical factories.

In the swiftly evolving landscape of the manufacturing industry, the bedrock of traditional centralized manufacturing confronts substantial challenges [24]. In response, the ascent of distributed manufacturing is reshaping conventions, ushering in swifter and leaner production paradigms. Notably, in practical scenarios, prominent manufacturers of large-scale engineering equipment frequently adopt multi-plant parallel production strategies, a configuration that can be abstracted into the framework of a distributed FJSP (DFJSP) [25]. In the last decade, the MODFJSP has garnered increasing amounts of attention among researchers [26]. Among notable contributions, Chang and Liu [27] introduced a hybrid GA tailored for the DFJSP, incorporating an innovative coding mechanism alongside effective crossover and mutation operators. Zhu et al. [28] proposed a hybrid genetic tabu search algorithm to solve a dynamic DFJSP with operation inspection. Marzouki et al. [29] proposed an innovative approach rooted in chemical reaction meta-heuristics to address DFJSPs, specifically focusing on the minimization of completion time.

Simultaneously, the discourse on global warming propels sustainable and energy-efficient manufacturing into the forefront [30]; within this context, the pursuit of energy-saving scheduling for DFJSPs garners significant attention. They presented a hyper-heuristic algorithm with Q-learning to address the energy efficiency of the distributed blocking flow-shop scheduling problem [31]. Lu et al. [32] designed a hybrid multi-objective optimization algorithm to solve the problem of the energy-efficient scheduling for a distributed flow shop with heterogeneous factories. Within the realm of MOEDFJSP research, a distinctive trend has emerged, involving the utilization of incentives derived from time-varying pricing schemes for electricity as a means to optimize energy costs [33]. This line of research also encompasses strategic decisions such as machine shutdown during idle periods [34, 35] and machine transition into standby mode [36]. Several of these studies extend their focus to include the interplay between machine switch-on times, power requirements and the machine's previous standby or switched-off state [37].

3. Multi-objective optimization model of the DFJSP

3.1. Problem description

The FJSP is characterized by n jobs being processed across m machines, with varying operation quantities per job. Each operation is potentially executed on any machine within a group of optional machines [38, 39]. In cases in which neighboring operations of the same job unfold on distinct machines, a transfer process arises, introducing additional transportation time [40].

It must be acknowledged that each machine calculates the time and energy consumption required to start before processing the initial job, and it subsequently shuts down after completing all of the jobs. The optimization objectives include minimizing the maximum completion time and energy consumption, considering the transportation time of jobs between machines and accounting for the state of machine startup and shutdown. The subsequent discussion in this paper is based on the following assumptions:

1) One job can only be processed on one machine simultaneously, precluding any interruption once commenced.

2) Multiple processes of a job can be processed by one machine.

3) Job preemption is not allowed, and each machine is only allowed to process one job at any time.

4) Upon job completion, swift transport to the subsequent machine takes place, accompanied by the corresponding transportation time.

5) The transportation time is only related to the machine number, and all transportation times are predefined.

6) The processing time of the job depends on the machine and is predefined.

7) The time and energy consumption required for machine startup and shutdown are predefined.

3.2. Notations

To facilitate clarity in subsequent discussions, we introduce symbol definitions in Table 1.

Table 1. Notations and indices for various parameters of the IMA.

Type	Symbol	Instruction
Indices		
	i, u	Index of jobs, $i = 1, 2, 3, ..., I$
	j, v	Index of operations, $j = 1, 2, 3, ..., J$
	m, m'	Index of machines, $m = 1, 2, 3, ..., M$
	f	Index of factories, $f = 1, 2, 3, ..., F$
Parameters		
	I	Total number of jobs
	J	Total number of operations
	M	Total number of machines
	M_f	Total number of machines in factory f
	F	Total number of factories
	O_j	Total number of operations for job j
	C_j	Completion time of job j
	$O_{i,j}(O_{u,v})$	jth (vth) process of job i (u)
	$S_{i,j,m,f}$	Start time of jth process of job i works on machine m in factory f
	$C_{i,j,m,f}$	Completion time of jth process of job i works on machine m in factory f
	C_{max}	Makespan
	TC_f	Total makespan of factory f
Variables		
	$X_{i,j,m,f}$	Binary variable that is equal to 1 if machine m of factory f is selected for $O_{i,j}$, and 0 otherwise
	$Y_{i,j,u,v,f}$	Binary variable that is equal to 1 if $O_{u,v}$ is not processed directly after $O_{i,j}$ on machine m in factory f, and 0 otherwise
	$Z_{m,f}$	Binary variable that is equal to 1 if machine m in factory f needs to start up and 0 otherwise
	$P_{i,j,m,f}$	Processing time of $O_{i,j}$ on machine m of factory f
	$PE_{i,j,m,f}$	Unit energy consumption of machine m of factory f to processes $O_{i,j}$
	$IE_{i,j,m,f}$	Unit energy consumption of idle waiting $O_{i,j}$ for m of factory f
	$TE_{i,j}$	Unit energy consumption of transportation $O_{i,j}$
	$T_{i,j,f}$	Time required for $O_{i,j}$ to transfer to $O_{i,j+1}$ in factory f
	$OE_{m,f}$	Unit energy consumption for the machine m to turn on or off in factory f
	$TS_{m,f}$	Time required for machine m to start up in factory f
	$TC_{m,f}$	Time required for machine m to shut down in factory f
	$MS_{m,f}$	Start time of machine m in factory f
	$ME_{m,f}$	Shutdown time of machine m in factory f
	AE_f	Unit auxiliary energy consumption in factory f
	PE	Energy consumption of machine processing
	IE	Energy consumption of machine idletime
	TE	Energy consumption of operation transportation
	OOE	Energy consumption of on/off machine
	AE	Energy consumption of auxiliary purposes
	ET	Energy consumption

3.3. Mathematical model

Regarding the mathematical model of the DFJSP, we have developed a mixed-integer linear programming (MILP) model to minimize both the makespan and energy consumption. The MILP model encompasses the objective function and various constraints. In addition, we introduced energy consumption modules for the production workshop. These modules account for different aspects of energy consumption, including the energy consumption module in the processing state (PE), the energy consumption module in the idle state (IE), the energy consumption module in the transportation state (TE), the energy consumption module in the on/off machine state (OOE) and the auxiliary energy consumption (AE) module for the entire production workshop. It is worth noting that, when two consecutive operations of a job are executed on different machines, we must also consider the energy consumption incurred during the movement of the job between these machines.

1) Energy consumption module for the PE. PE is the energy consumption generated by all machine processing jobs, and it is related to the time that each machine takes to process the job and the energy consumption required for processing per unit time. We take Figure 1 as an example, and, referring to Tables 2–4, we can calculate that $PE = 11 \times 4 = 44$ in this example.

$$PE = \sum_{i=1}^{I} \sum_{j=1}^{O_j} \sum_{f=1}^{F} \sum_{m=1}^{M_f} P_{i,j,m,f} PE_{i,j,m,f} X_{i,j,m,f} \tag{3.1}$$

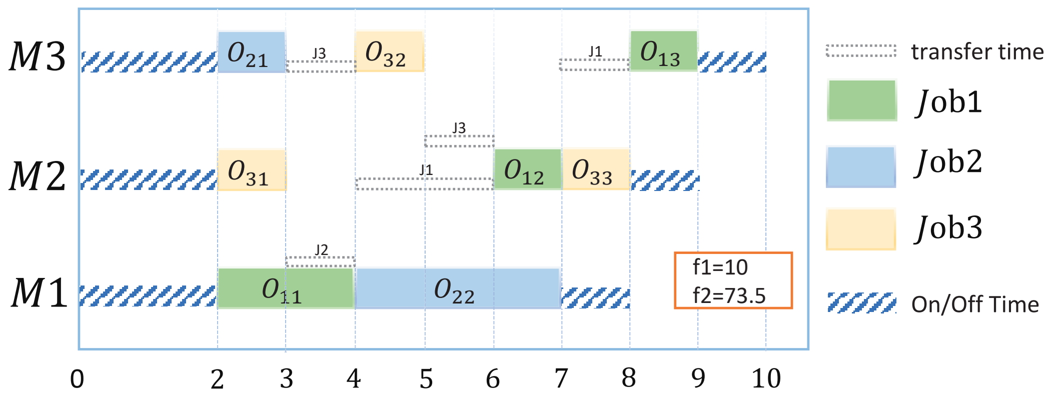


Figure 1. A feasible scheduling method.

Table 2. Processing times of different processes.

	O_{11}	O_{12}	O_{13}	O_{21}	O_{22}	O_{31}	O_{32}	O_{33}
M1	2	–	–	2	3	–	2	–
M2	3	1	2	2	–	1	–	1
M3	–	3	1	1	–	2	1	2

Table 3. Transportation time between machines.

	M1	M2	M3
M1	0	2	1
M2	2	0	1
M3	1	1	0

Table 4. Explanation of various energy consumption modules (unit time).

PE	IE	TE	OOE	AE
4	2	1	0.5	0.5

2) Energy consumption module for the IE. IE represents the energy consumption while waiting to process the next operation when a machine is in the idle running state. Idle time is the duration between two consecutive operations on the same machine, and it depends on machine selection, the operation sequence and the associated transportation time. We take Figure 1 as an example, and, referring to Tables 2–4, we can calculate that $IE = 7 \times 2 = 14$ in this example.

$$IE = \sum_{i=1}^{I} \sum_{j=2}^{O_j} \sum_{f=1}^{F} \sum_{m=1}^{M_f} (S_{i,j,m,f} - C_{i,j-1,m,f}) IE_{i,j,m,f} \tag{3.2}$$

3) Energy consumption module for the TE. TE is the energy consumption of the transportation resources for transporting a job whenever it needs to move from one machine tool to another. Transportation time depends on the distance between different machines. We take Figure 1 as an example, and, referring to Tables 2–4, we can calculate that $TE = 6 \times 1 = 6$ in this example.

$$TE = \sum_{i(u)=1}^{I} \sum_{j(v)=2}^{O_j} \sum_{f=1}^{F} TE_{i,j} T_{i,j-1,f} Y_{i,j,u,v,f} \tag{3.3}$$

4) Energy consumption module for the OOE. OOE is the energy consumption when the machines are turned on and off, and it is related to the number of machines being activated and the time required for the activation and deactivation process. We take Figure 1 as an example, and, referring to Tables 2–4, we can calculate that $OOE = 9 \times 0.5 = 4.5$ in this example.

$$OOE = \sum_{f=1}^{F} \sum_{m=1}^{M_f} (TS_{m,f} + TC_{m,f}) OE_{m,f} Z_{m,f} \tag{3.4}$$

5) AE module for the production workshop. AE is the auxiliary energy required to support the production environment, such as lighting, heating and air conditioning. We take Figure 1 as an example, and, referring to Tables 2–4, we can calculate that $AE = 10 \times 0.5 = 5$ in this example.

$$AE = \sum_{f=1}^{F} AE_f TC_f \tag{3.5}$$

In this paper, the two objectives of the scheduling optimization are to minimize two critical factors: the maximum completion time for all operations (commonly known as 'makespan') and the total energy consumption. Therefore, the MILP model of the DFJSP, which requires the minimization of the makespan and energy consumption, is formulated as follows.

1) Minimization of $C_{max}(f_1)$:

$$f_1 = max\{C_j | j = 1, 2, ..., J\} \tag{3.6}$$

2) Minimization of $ET(f_2)$:

$$\begin{aligned} f_2 =& PE + IE + TE + OOE + AE \\ =& \sum_{i=1}^{I} \sum_{j=1}^{O_j} \sum_{f=1}^{F} \sum_{m=1}^{M_f} P_{i,j,m,f} PE_{i,j,m,f} X_{i,j,m,f} \\ &+ \sum_{i=1}^{I} \sum_{j=2}^{O_j} \sum_{f=1}^{F} \sum_{m=1}^{M_f} (S_{i,j,m,f} - C_{i,j-1,m,f}) IE_{i,j,m,f} \\ &+ \sum_{i(u)=1}^{I} \sum_{j(v)=2}^{O_j} \sum_{f=1}^{F} TE_{i,j} T_{i,j-1,f} Y_{i,j,u,v,f} \\ &+ \sum_{f=1}^{F} \sum_{m=1}^{M_f} (TS_{m,f} + TC_{m,f}) OE_{m,f} Z_{m,f} \\ &+ \sum_{f=1}^{F} AE_f TC_f \end{aligned} \tag{3.7}$$

subject to the following:

$$\sum_{f=1}^{F} M_f = M \ , \ f = 1, 2, ...F \tag{3.8}$$

$$0 \leq T_{i,j,f} \leq 4 \ , \ i = 1, 2, ...I; j = 1, ...O_j; f = 1, 2, ...F \tag{3.9}$$

$$\sum_{f=1}^{F} \sum_{m=1}^{M_f} X_{i,j,m,f} = 1 \ , \ i = 1, 2, ...I; j = 1, 2, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \tag{3.10}$$

$$\sum_{i=1}^{I} \sum_{j=1}^{O_j} \sum_{f=1}^{F} Y_{i,j,u,v,f} \leq J - I \ , \ i = 1, 2, ...I; j = 1, ...O_j \tag{3.11}$$

$$\sum_{f=1}^{F}\sum_{m=1}^{M_f} Z_{m,f} \leq M \ , \ m = 1, 2, ...M_f; f = 1, 2, ...F \tag{3.12}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{O_j} O_{i,j} = J \ , \ i = 1, 2, ...I; j = 1, 2, ...O_j \tag{3.13}$$

$$ME_{m,f} + TC_{m,f} \leq C_{max} \ , \ m = 1, 2, ...M_f; f = 1, 2, ...F \tag{3.14}$$

$$C_{i,j,m,f} - C_{i,j-1,m,f} \geq P_{i,j,m,f} * X_{i,j,m,f} + T_{i,j,f} \ , \ i = 1, 2, ...I; j = 2, ...O_j \tag{3.15}$$

$$\begin{gathered} S_{i,j,m,f} \geq min(MS_{m,f}) + TS_{m,f} \ , \\ i = 1, 2, ...I; j = 1, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \end{gathered} \tag{3.16}$$

$$\begin{gathered} C_{i,j,m,f} \leq max(ME_{m,f}) + TC_{m,f} \ , \\ i = 1, 2, ...I; j = 1, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \end{gathered} \tag{3.17}$$

$$\begin{gathered} S_{i,j,m,f} \geq C_{i,j-1,m,f} + T_{i,j,f} \ , \\ i = 1, 2, ...I; j = 2, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \end{gathered} \tag{3.18}$$

$$\begin{gathered} 0 < S_{i,j,m,f} \leq C_{max} - PE_{i,j,m,f} - ME_{m,f} - TC_{m,f} \ , \\ i = 1, 2, ...I; j = 2, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \end{gathered} \tag{3.19}$$

$$\begin{gathered} MS_{m,f} + TS_{m,f} + PE_{i,j,m,f} < C_{i,j,m,f} \leq C_{max} - ME_{m,f} - TC_{m,f} \ , \\ i = 1, 2, ...I; j = 2, ...O_j; m = 1, 2, ...M_f; f = 1, 2, ...F \end{gathered} \tag{3.20}$$

3.4. Example problem

For ease of description, we provide a concise example of an FJSP involving three jobs and three machines. The processing times for distinct operations on each machine, corresponding to the processing of different jobs, are delineated in Table 2. Note that '–' indicates that this process cannot be executed on the corresponding machine. For example, the second process of Job 1, denoted as $O_{1,2}$, cannot be executed on the M1 machine; instead, it has a processing time of 1 on the M2 machine and 3 on the M3 machine. Additionally, Table 3 displays the transportation time between different machines. For example, the transportation time from M1 to M2 for a task is 2, and the transportation time from M1 to M3 requires a value of 1. Table 4 outlines the unit energy consumption for each energy-consuming element. Subsequently, Figure 1 illustrates a viable scheduling approach based on this context. With this scheduling, the completion time is 10 and the energy consumption is 73.5.

4. The proposed IMA for the MODFJSP

In the realm of multi-objective optimization problems, the NSGA-II algorithm possesses robust global search capabilities, but has limitations in terms of local search efficiency [41]. This aspect may be enhanced through the utilization of MAs. Furthermore, the domain of multi-objective flexible job shop scheduling, which includes considerations for job transport between machines and machine switching operations, features a notably intricate solution space compared to traditional single-objective flexible job shop scheduling. Based on these considerations, this paper presents an IMA to solve the MODFJSP. The algorithm flowchart is shown in Figure 2, and its core process is as follows.

Step 1: Formulate a population, Pt, with size Ps by using four distinct initialization strategies.

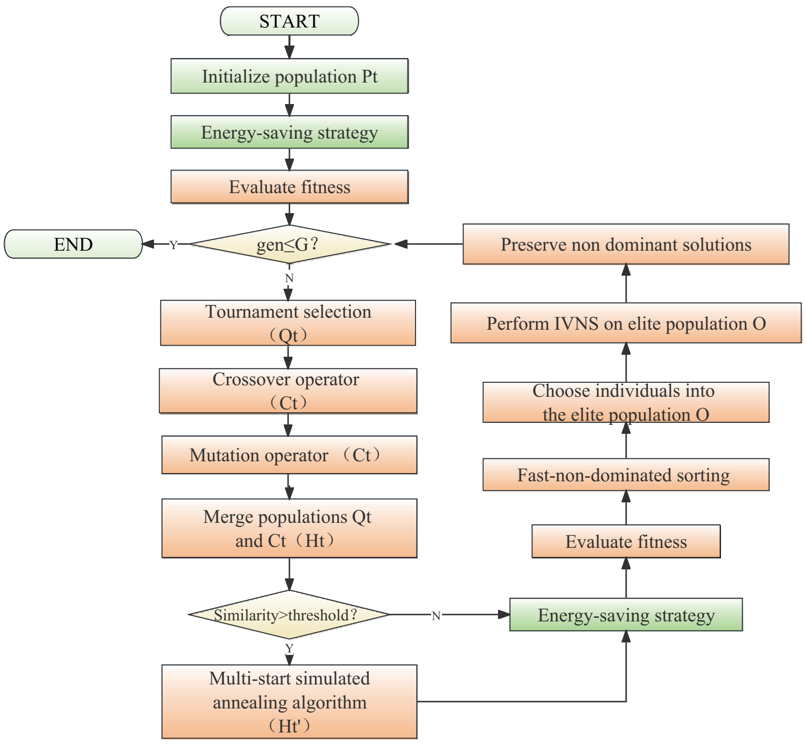

START
Initialize population Pt
Energy-saving strategy
Evaluate fitness
END
Y
gen≤G?
N
Preserve non dominant solutions
Tournament selection (Qt)
Perform IVNS on elite population O
Crossover operator (Ct)
Choose individuals into the elite population O
Mutation operator (Ct)
Fast-non-dominated sorting
Merge populations Qt and Ct (Ht)
Evaluate fitness
Similarity>threshold?
N
Energy-saving strategy
Y
Multi-start simulated annealing algorithm (Ht')

Figure 2. The main flowchart of the IMA.

Step 2: Execute the energy-saving strategy (ESS) on Pt, subsequently evaluating fitness.

Step 3: Employ tournament selection to choose individuals from Pt to populate the mating pool Qt.

Step 4: Traverse Qt, probabilistically selecting individuals and members of Pt for crossover and mutation (with probabilities of Pc and Pm) , and integrating them into the sub-population Ct.

Step 5: Combine Qt and Ct to form Ht, calculate the similarity and apply an MSSA to individuals exceeding the threshold to generate Ht'.

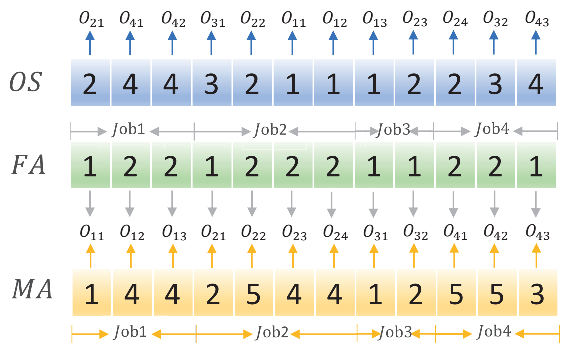
Step 6: Implement the ESS and compute the fitness for the new individuals within Ht' by using Eqs (3.6) and (3.7), followed by a fast non-dominated sorting operation.

Step 7: Within Ht', choose individuals into the elite archive O. Concurrently, apply four distinct forms of variable neighborhood search and update the elite archive O.

Step 8: Preserve the non-dominated solutions within O. If the stipulated termination condition is not met, proceed to Step 2 to continue the iterative process.

4.1. Chromosome coding

In this paper, a three-layer coding method containing operation sequencing (OS) , factory assignment (FA) and machine assignment vectors is designed. Figure 3 gives an example of this coding method. In this figure, the first row of 2, 4, 4, 3, 2, 1, 1, 1, 2, 2, 3, 4 represents the OS vector. Here, 1, 2, 3 and 4 represent jobs 1, 2, 3 and 4, respectively. The first 2 represents the first operation of job 2, and the second 2 represents the second operation of job 2. The operation, factory and machine sequence in this example are as follows: (O21, F1, M2), (O41, F2, M5), (O42, F2, M5), (O31, F1, M1), (O22, F2, M5), (O11, F1, M1), (O12, F2, M4), (O13, F2, M4), (O23, F2, M4), (O24, F2, M4), (O32, F1, M2) and (O43, F1, M3). Here, (O21, F1, M2) means that operation 1 of job 2 (O21) is processed by machine 2 of factory 1; similarly, (O41, F2, M5) means that operation 1 of job 4 (O41) is processed by machine 5 of factory 2, etc.



O_{21} O_{41} O_{42} O_{31} O_{22} O_{11} O_{12} O_{13} O_{23} O_{24} O_{32} O_{43}
OS
2 4 4 3 2 1 1 1 2 2 3 4
Job1 Job2 Job3 Job4
FA
1 2 2 1 2 2 2 1 1 2 2 1
O_{11} O_{12} O_{13} O_{21} O_{22} O_{23} O_{24} O_{31} O_{32} O_{41} O_{42} O_{43}
MA
1 4 4 2 5 4 4 1 2 5 5 3
Job1 Job2 Job3 Job4

Figure 3. An example of encoding method; MA: machine assignment.

4.2. Initialization

Initialization plays a crucial role in algorithmic optimization. Commonly employed methods include first-in-first-out, shortest processing time first (SPT) and longest processing time first (LPT). However, most of these initialization methods yield single results, and a single initialization rule may not adapt well to various instances, resulting in a lack of diversity in the initial population.

To tackle this problem, we introduce an innovative initialization approach called the multi-initialization rule (MIR). The MIR employs four distinct initialization rules, each emphasizing unique aspects. This strategy aims to generate high-quality initial populations while maintaining essential diversity within the population's composition.

The first rule within the MIR is improved SPT (ISPT). Its implementation process is as follows.

Step 1: Assuming that the order of each process is the same (for example, it is the first process), the process with the shortest processing time has the highest priority.

Step 2: When multiple processes have the same processing time, their order will be randomly disrupted.

For convenience of explanation, we will use the data in Table 2 as an example. Figure 4(a) shows a set of chromosomes obtained via ISPT. At this point, if ISPT regenerates into a set of identical chromosomes, it will randomly disrupt the sequence of processes with the same processing time. As shown in Figure 4(b), $O_{2,1}$ and $O_{3,1}$, $O_{1,2}$ and $O_{3,2}$, $O_{3,1}$ and $O_{3,3}$ will randomly disrupt the order. This avoids the disadvantage of only generating one chromosome per initialization rule and enhances the diversity of the population.

The second rule within the MIR is improved LPT (ILPT). Its implementation process is as follows.

Step 1: Assuming that the order of each process is the same (for example, it is the first process), the process with the longest processing time has the highest priority.

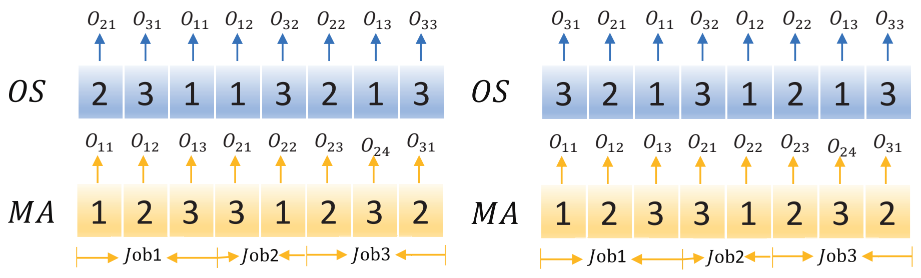
Step 2: When multiple processes have the same processing time, their order will be randomly disrupted.

Figure 5 also provides a visual example.

The third rule within the MIR is the shortest transportation time first (MTT). Under the constraint of 3.3, the process with the shortest transportation time takes priority.

The fourth rule, known as random initialization, involves generating each chromosome by using a random method.

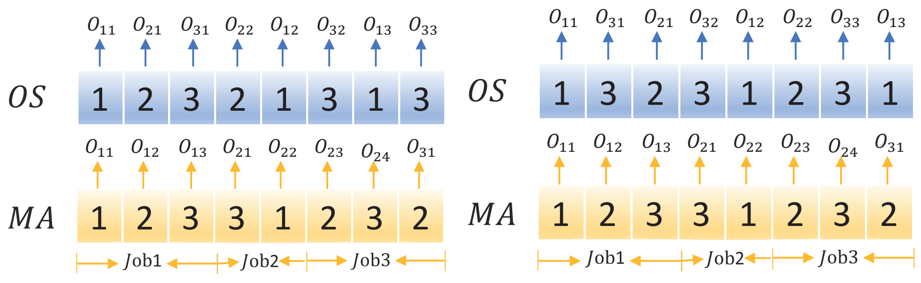
The initialization process assigns the following percentages to the aforementioned rules: 0.3, 0.3, 0.3 and 0.1.



OS
O_{21} O_{31} O_{11} O_{12} O_{32} O_{22} O_{13} O_{33}
2 3 1 1 3 2 1 3
MA
O_{11} O_{12} O_{13} O_{21} O_{22} O_{23} O_{24} O_{31}
1 2 3 3 1 2 3 2
Job1 Job2 Job3
OS
O_{31} O_{21} O_{11} O_{32} O_{12} O_{22} O_{13} O_{33}
3 2 1 3 1 2 1 3
MA
O_{11} O_{12} O_{13} O_{21} O_{22} O_{23} O_{24} O_{31}
1 2 3 3 1 2 3 2
Job1 Job2 Job3

(a) Step 1 of ISPT **(b)** Step 2 of ISPT

Figure 4. An initialization example of ISPT.



OS
O_{11} O_{21} O_{31} O_{22} O_{12} O_{32} O_{13} O_{33}
1 2 3 2 1 3 1 3
MA
O_{11} O_{12} O_{13} O_{21} O_{22} O_{23} O_{24} O_{31}
1 2 3 3 1 2 3 2
Job1 Job2 Job3
OS
O_{11} O_{31} O_{21} O_{32} O_{12} O_{22} O_{33} O_{13}
1 3 2 3 1 2 3 1
MA
O_{11} O_{12} O_{13} O_{21} O_{22} O_{23} O_{24} O_{31}
1 2 3 3 1 2 3 2
Job1 Job2 Job3

(a) Step 1 of ILPT **(b)** Step 2 of ILPT.

Figure 5. An initialization example of ILPT.

4.3. ESS

Energy efficiency has become a significant concern for manufacturing enterprises. Enhancing energy efficiency not only reduces energy consumption, it also lowers production costs. In the field of MODFJSPs, the pursuit of better energy efficiency relies on three key strategies: the modulation of machine working speeds, task scheduling informed by time-dependent energy pricing and the strategic engagement of energy-efficient machine modes, such as machine shutdown during idle periods.

It is worth noting that there has been a lack of prior research in the area of energy-saving scheduling that considers both transportation times and the on/off states of machines. This absence of an effective ESS in this important and widespread domain emphasizes the need for the development of more effective energy-saving approaches. In response, we have created the ESS, which consists of two distinct yet complementary energy-saving tactics:

The first optimization method (ESS1) is to maximize the delay in machine startup and job start time without affecting the completion time. This goal first involves determining the start and end times of the last process on each machine, and then conducting reverse calculations to find the latest feasible start time for the previous process. It is worth noting that, if a machine is not assigned a task, it will not start. We take Figure 1 as an example, where Figure 6 is the scheduling scheme generated after ESS1. It can be seen that, with the same scheduling order, the energy consumption decreased from 73.5 to 71.5 after using ESS1.

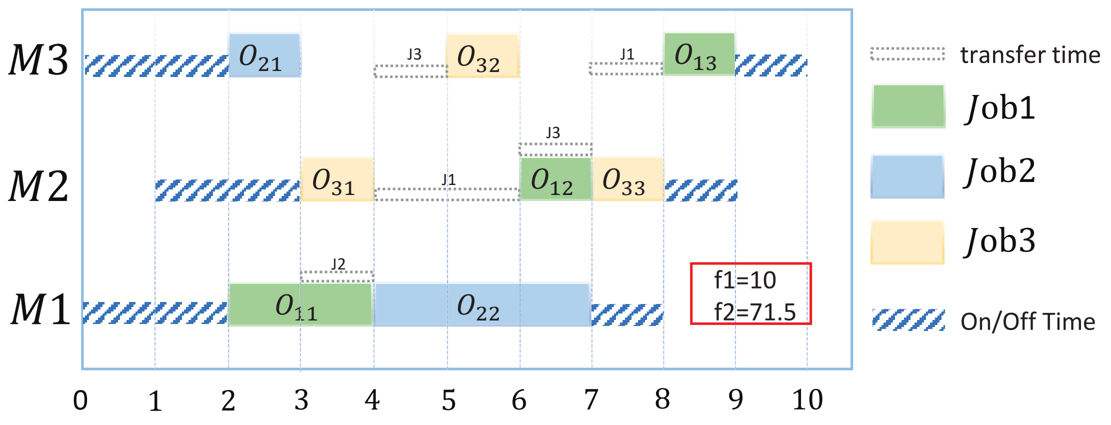


Figure 6. An example of ESS1.

The second optimization method is to temporarily shut down the machine when the idle time between two processes on the same machine is too long, and then restart it before the next process begins. This clever pause saves energy by reducing the accumulated waiting time of the machine. It should be noted that the switch operation of the machine cannot affect the start time of the next process. In addition, considering the potential impact of frequent machine switches on the lifespan of the machine, this strategy allows for, at most, one occurrence throughout the entire scheduling process. We take Figure 1 as an example, where Figure 7 is the scheduling scheme generated after ESS2. It can be seen that, with the same scheduling order, the energy consumption decreased from 73.5 to 69 after using ESS2.

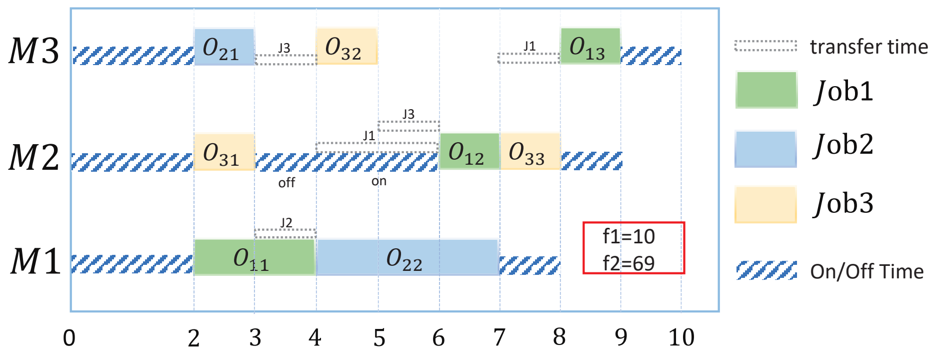


Figure 7. An example of ESS2.

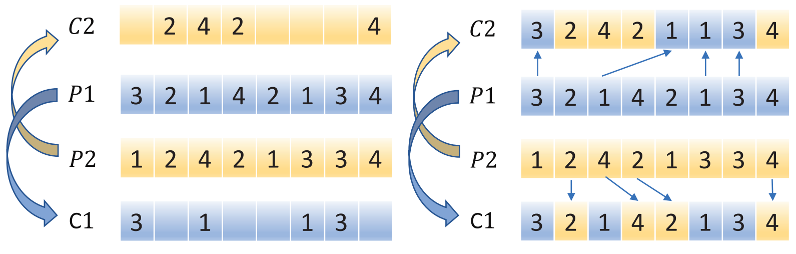
4.4. Crossover and mutation operator

4.4.1. Crossover operator

The purpose of crossover is to exchange information between parents and retain good information in the parental generation to generate new individuals. For two chromosomes, a direct "crossover" of information from factories, jobs and machines would produce an infeasible solution, because not all factories, jobs and machines can handle all operations. Here, we use the classic POX crossover:

Step 1: First, randomly take the fragment of the gene that needs to be crossed on the chromosome,and then the parent P1; the fragment taken is copied to C1; similarly, P2 is also the same operation.

Step 2: In P1, take out the job containing J2 and copy the remaining job to C2 in the original order; in P2, remove the job containing J1 and copy the remaining job to C1 in the original order. The specific process is shown in Figure 8.



(a) Step1 of POX crossover **(b)** Step2 of POX crossover

Figure 8. An example of POX crossover.

4.4.2. Mutation operator

In this work, we use commutative mutation operators to expand the solution space and maintain good solutions. The main process of exchanging mutation operators is described as follows (P1 and O1 are used to represent parents and children, respectively).

Step 1: Randomly select two different positions of chromosomes in P1 (if they are the same, then randomly select again until they are different).

Step 2: Swap the elements at the selected position to generate O1.

4.5. The proposed MSSA

Combining the MA and SA is a relatively new method, especially when the proposed MSSA is used after the population merging process. This method involves selecting individuals with excessive similarity for the MSSA to improve individuals from the MA.

Traditional MAs usually choose to delete duplicate individuals after the operation of population merging as a way to avoid the algorithm from falling into local optimum prematurely, but this operation often has the following problems. First, most of the results generated by the initialization rules in other studies are unique, and even if multiple initialization rules are combined, there are very many duplicate individuals in its initial population, which leads to a lack of diversity, and it is very easy to fall into local optima even if duplicate individuals are deleted. Second, some studies on repetitive individual judgments only consider their fitness values, which is not perfect. It is likely to delete individuals with the same fitness value but different OS, FA and machine assignment, leading to a decrease in diversity and a tendency to fall into local optima.

4.5.1. Chromosome similarity

We propose a method for calculating the similarity between two chromosomes, Eq (4.1) below is used to calculate the similarity between two chromosomes, where J represents the length of a chromosome, and S_j can be calculated by using Eq (4.2).

$$S = \frac{\sum_{j=1}^{J} S_j}{J} \tag{4.1}$$

$$S_j = \begin{cases} 1 & , \textit{genes of two chromosomes on position } j \textit{ are the same} \\ 0 & , \textit{otherwise} \end{cases} \tag{4.2}$$

Taking P1 ([3,2,1,4,2,1,3,4]) and P2 ([1,2,4,2,1,3,3,4]) as examples, after POX crossover, C1 and C2 are [3,2,1,4,2,1,3,4] and [3,2,4,2,1,1,3,4], respectively. At this point, the similarity values for P1 and C1 are the same, while the similarity values for P2 and C2 are 0.75.

4.5.2. Implementation process of MSSA

Then, we propose an MSSA. After population merging, we first select individuals with the same fitness values (maximum completion time and energy consumption), and then determine whether the the similarity value S is 1; if so, we use these individuals as the starting point of the MSSA. We use the following three perturbation strategies for the selected point:

1) Randomly exchange the values of two points on the chromosome, as shown in Figure 9;

2) Move the chromosome to the left by three distances, as shown in Figure 10;

3) Utilize the N5 neighborhood structure to change the processing order of two adjacent critical processes at the head or tail of the block, as shown in Figure 11.

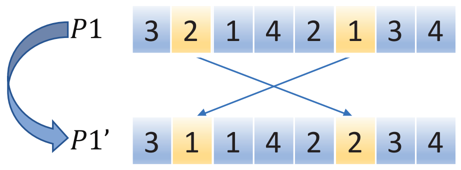


Figure 9. Perturbation strategy 1.

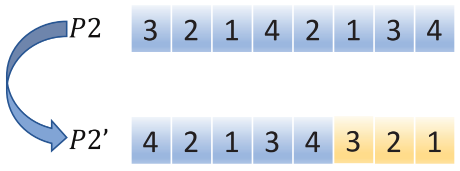


Figure 10. Perturbation strategy 2.

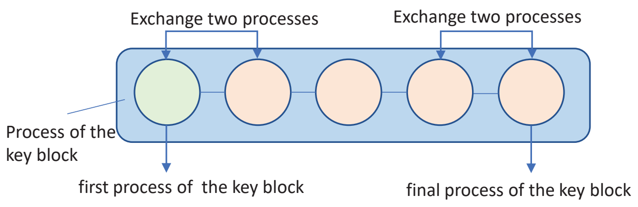


Figure 11. Perturbation strategy 3.

We search in three different directions with different focuses at the selected point to enhance diversity and avoid falling into local optima too early. Figure 12 shows the possible results of the three perturbation strategies.

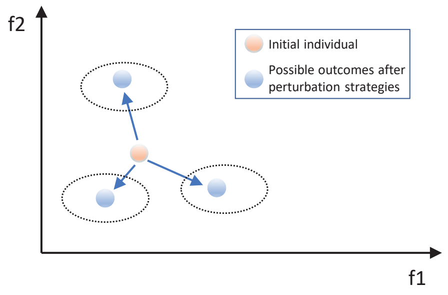


Figure 12. Schematic diagram of perturbation strategies.

However, SA is essentially a serial method, and running SA multiple times requires a long computational time, which does not meet our expectations. Therefore, for each selected individual, we execute the above three perturbation strategies once and calculate their respective fitness. Assuming the original solution is A, we select the best individual from the three new solutions as B. If the fitness values of both objectives of B ($f_1(B)$, $f_2(B)$) are better than those of A ($f_1(A)$, $f_2(A)$), then SA accepts B. Otherwise, SA accepts B based on a decreasing probability (p(B)):

$$\triangle E = [f_1(B) - f_1(A)] + [f_2(B) - f_2(A)] \tag{4.3}$$

$$p(B) = exp^{\triangle E/T} \tag{4.4}$$

where Δ is the difference between the fitness values of A and B, T is the reciprocal of the current number of iterations and *exp* is the exponential value. The core pseudo code of the MSSA is shown in Algorithm 1:

Algorithm 1: MSSA

```
   Input: P, OS, MA, FA, fitness
1  for $i$ = $1:P-1$ do
2      sim = Calculate similarity($P$,$Pi$);
3      if $sim > 0.95$ then
4          $P1$ = Perturbation strategy 1(P, OS, MA, FA, fitness);
5          $P2$ = Perturbation strategy 2(P, OS, MA, FA, fitness);
6          $P3$ = Perturbation strategy 3(P, OS, MA, FA, fitness);
7      end
8      $PN$ = best($P1$,$P2$,$P3$);
9      if $fitness(PN) > fitness(Pi)$ then
10         $Pi = PN$;
11     else
12         if $p(P) < exp^{\triangle E/T}$ then
13             $Pi = PN$;
14         else
15             delete($PN$);
16         end
17     end
18 end
```

4.6. The proposed IVNS strategy

In this section, we have designed four IVNS strategies, each capable of generating four neighborhood solutions. The first neighborhood structure, VNS1, randomly exchanges two different values on a chromosome. A visual representation of this process is illustrated in Figure 13. The second neighborhood structure, VNS2, involves the random selection of a chromosome value, followed by its repositioning to the subsequent process. A visual representation of this process is illustrated in Figure 14. The third neighborhood structure, VNS3, is designated as N6, a proven effective approach, exemplified in Figure 15. The fourth structure, an N6 variant depicted in

Figure 16, where, for the intermediate critical block, the intermediate process, or the head process, is randomly chosen to be inserted after the tail process.

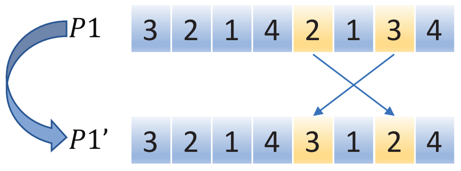
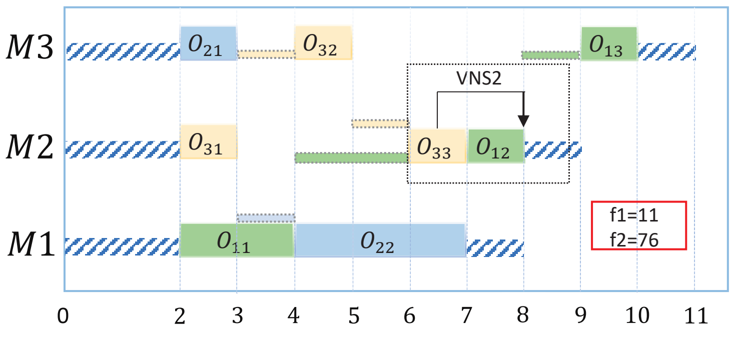
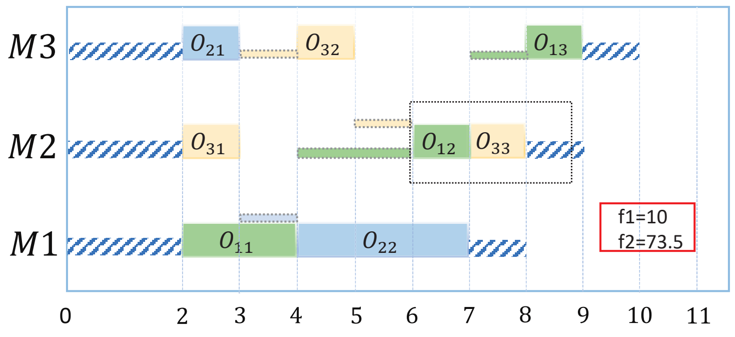


Figure 13. An example of VNS1.



(a) The original scheduling Gantt chart



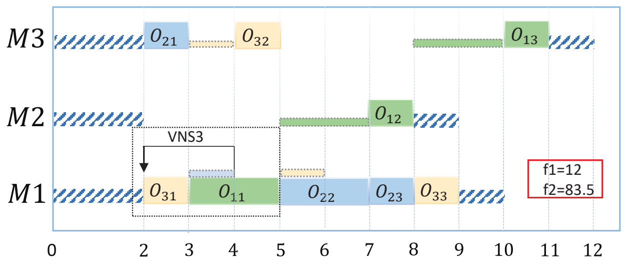
(b) The Gantt chart after VNS2 strategy.

Figure 14. An example of VNS2.

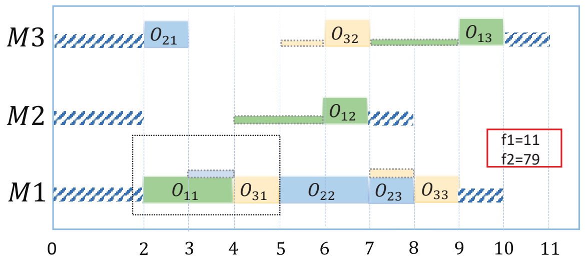
Taking the data in Table 4 as an example, we can see from Figure 14 that, after VNS2, the completion time has changed from 11 to 10, and the energy consumption has been reduced from 76 to 73.5.

Taking the data in Table 4 as an example, we can see from Figure 15 that, after VNS3, the completion time has changed from 12 to 11, and the energy consumption has been reduced from 83.5 to 79.

Taking the data in Table 4 as an example, we can see from Figure 16 that, after VNS4, the completion time has changed from 11 to 9, and the energy consumption has been reduced from 70 to 65.

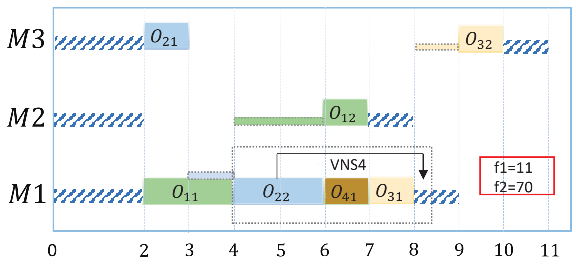


(a) The original scheduling Gantt chart.

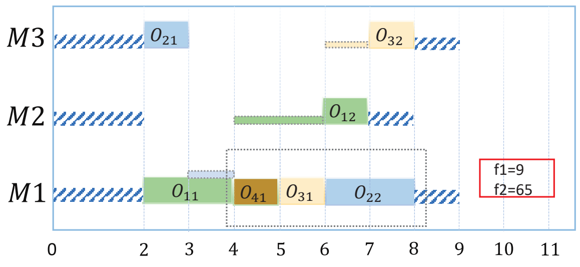


(b) The Gantt chart after VNS3 strategy.

Figure 15. An example of VNS3.



(a) The original scheduling Gantt chart.



(b) The Gantt chart after VNS4 strategy.

Figure 16. An example of VNS4.

5. Comprehensive experiments

This section describes the IMA in detail. All codes in this paper were programmed in Matlab2022a, running on Windows 11, AMD Ryzen 5 5600U, 2.30 GHz, 16 GB RAM.

5.1. Experimental instances and metrics

In order to measure the performance of the multi-objective optimization algorithm, this study adopts the hypervolume metric (HV) and the inverse generational distance (IGD) to represent the comprehensive performance, convergence and diversity of the IMA. The HV metric is used to measure the volume of the region in the objective space that is surrounded by the set of non-dominated solutions obtained via the algorithm and the reference points. The larger the value of HV, the better the comprehensive performance of the algorithm. The IGD metric measures the average distance from each reference point to the nearest solution; the smaller the IGD value, the better the algorithm's overall performance. HV and IGD are the most commonly used and intuitive metrics to reflect the overall performance of an algorithm [42].

5.2. Parameter settings

To validate the efficacy of the proposed algorithm, we have applied Mk01-10 and DP01-10 as instances. Additionally, all instances are constructed with two isomorphic plants equipped with identical sets of machines. The various energy consumption is shown in Table 4, and the transportation time between machines is shown in Table 5.

Table 5. Transportation times of jobs between different machines.

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
M1	0	4	2	4	2	4	3	4	2	2	1	1	3	4	1
M2	4	0	4	3	3	3	2	4	3	1	4	4	4	4	3
M3	2	4	0	4	3	4	1	3	4	4	3	3	2	3	4
M4	4	3	4	0	2	2	4	2	1	3	3	2	3	3	3
M5	2	3	3	2	0	1	3	3	2	1	4	2	2	3	2
M6	4	3	4	2	1	0	2	2	4	4	4	3	2	3	3
M7	3	2	1	4	3	2	0	3	1	4	2	3	2	3	3
M8	4	4	3	2	3	2	3	0	4	4	3	2	4	4	4
M9	2	3	4	1	2	4	1	4	0	1	2	3	4	4	2
M10	2	1	4	3	1	4	4	4	1	0	4	1	3	3	2
M11	1	4	3	3	4	4	2	3	2	4	0	4	4	4	4
M12	1	4	3	2	2	3	3	2	3	1	4	0	4	4	3
M13	3	4	2	3	2	2	2	4	4	3	4	4	0	3	4
M14	4	4	3	3	3	3	3	4	4	3	4	4	3	0	3
M15	1	3	4	3	2	3	3	4	2	2	4	3	4	3	0

Different parameter settings will affect the performance of the algorithm. Three parameters need adjustment: the population size (ps), the crossover rate (pc) and the mutation rate (pm). A Taguchi approach to the design of experiment was used to determine the parameters. The parameter level settings are as follows:

- $ps = 50, 100, 200$
- $pc = 0.6, 0.8, 1$
- $pm = 0.1, 0.2, 0.3$

An orthogonal array was used in the experiment. For fairness, we ran each parameter combination 10 times and recorded the average HV. Table 6 presents the HV calculation results for each combination. To quantify the statistical results, the average HV was calculated by using three tests within the same layer, and Figure 17 shows the factor-level trends of different parameters.

In addition, Figure 17 shows that ps has the highest values of HV at $level2$, and pc and pm have the highest HV at $level2$ and $level1$. The final parameter combination was determined as ps = 100, pc = 0.8 and pm = 0.1.

Table 6. Factor levels of the three key parameters.

Trial	Factor levels			HV
	ps	pc	pm	
1	1	1	1	0.67715
2	1	2	3	0.66834
3	1	3	2	0.67064
4	2	1	3	0.66768
5	2	2	2	0.70689
6	2	3	1	0.69632
7	3	1	2	0.66431
8	3	2	1	0.70967
9	3	3	3	0.67571

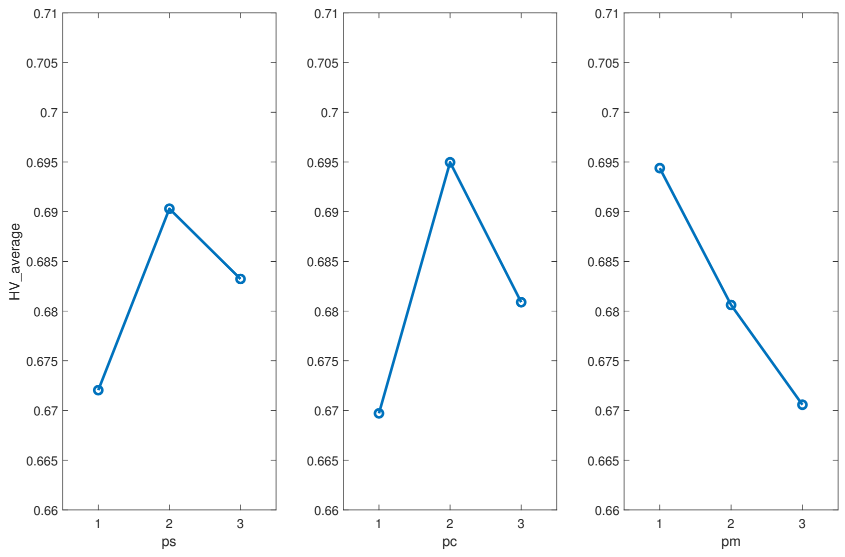


Figure 17. Parameter levels.

5.3. *Performance analysis*

Our proposed IMA comprises four primary components: 1) MIR, 2) ESS, 3) MSSA and 4) IVNS. These components are evaluated across a comprehensive set of 20 instances, including Mk01-10 and

DP01-10. To ensure impartiality, each algorithm was autonomously executed 20 times on each instance, spanning a maximum of 200 iterations.

In this paper, *HV* and *IGD* are used to verify the performance of the proposed IMA. *HV* evaluates the overall performance of the algorithm, while *IGD* assesses the convergence and uniformity of the distribution. A higher *HV* is desirable, whereas a lower *IGD* is preferred. The relevant equations for *HV* and *IGD* are as follows:

1) *HV* metric:

$$HV(P) = volume(\bigcup_{i=1}^{P} v_i) \tag{5.1}$$

where P is the obtained non-dominated solution set, and v_i is the hypercube formed between P and the reference point.

2) *IGD* metric:

$$IGD(P, P^*) = \frac{1}{|P^*|} \sum_{x \in P^*} \min_{y \in P} dis(x, y) \tag{5.2}$$

where P is the non-dominated solution set; P^* is the Pareto front solution set; $dis(x, y)$ is the Euclidean distance between x and y.

In addition, before calculating *IGD* and *HV*, we should use method to convert it and normalize it afterward, as shown below:

$$\widetilde{f_i}(x) = \frac{f_i(x) - f_i^{min}}{f_i^{max} - f_i^{min}}\ ,\ i = 1, 2 \tag{5.3}$$

where f_i^{max} and f_i^{min} are the maximum and minimum of the ith objective in all solutions.

5.3.1. Effect of the MIR

First, we compared the performance of the proposed IMA with its variant, IMA1, which excludes the MIR component. The results of these ablation experiments are presented in Table 7.

Table 7. Comparison of HV and IGD values in ablation experiments.

Instance	HV					IGD				
	IMA	IMA1	IMA2	IMA3	IMA4	IMA	IMA1	IMA2	IMA3	IMA4
Mk01	**0.697008**	0.686350	0.687158	0.678451	0.513899	**0.063059**	0.292920	0.269982	0.242755	0.225158
Mk02	0.650904	**0.685495**	0.652535	0.643115	0.539922	**0.064975**	0.313251	0.278411	0.234520	0.207354
Mk03	**0.672812**	0.623755	0.663073	0.643935	0.433202	**0.123285**	0.481062	0.414504	0.440899	0.474141
Mk04	0.720483	**0.775424**	0.726553	0.713493	0.491821	**0.172815**	0.426255	0.368692	0.348434	0.388389
Mk05	**0.612882**	0.607545	0.590346	0.593566	0.482244	**0.067436**	0.300838	0.237517	0.229288	0.242625
Mk06	**0.775788**	0.480569	0.513005	0.503795	0.445338	**0.020985**	0.439600	0.411603	0.420604	0.473418
Mk07	**0.595262**	0.588811	0.575125	0.570323	0.526268	**0.092555**	0.300407	0.238518	0.239293	0.248495
Mk08	**0.525119**	0.511179	0.469968	0.499948	0.301304	**0.198234**	0.476992	0.416335	0.426463	0.458929
Mk09	**0.547976**	0.440694	0.527078	0.525860	0.416216	**0.192776**	0.614409	0.549823	0.611572	0.617453
Mk10	**0.714032**	0.490423	0.675172	0.683258	0.567387	**0.020126**	0.674355	0.534824	0.640007	0.671195
DP01	**0.636001**	0.612112	0.595021	0.602130	0.232656	**0.144596**	0.555473	0.510615	0.515116	0.485194
DP02	**0.727061**	0.703321	0.712612	0.707359	0.517330	**0.130037**	0.440501	0.396732	0.312478	0.369240
DP03	**0.635148**	0.610721	0.593201	0.610985	0.444586	**0.063712**	0.402861	0.407035	0.367904	0.385437
DP04	0.656419	**0.673828**	0.643144	0.640646	0.385900	**0.153222**	0.503065	0.476446	0.433544	0.443414
DP05	**0.681177**	0.660793	0.655967	0.645409	0.493956	**0.102756**	0.347240	0.354396	0.271796	0.291469
DP06	0.720815	0.685001	0.724594	**0.728559**	0.555081	**0.118611**	0.407591	0.360225	0.317026	0.384421
DP07	0.483865	**0.570962**	0.478514	0.456368	0.494673	**0.311614**	0.545135	0.475118	0.453076	0.313972
DP08	**0.860201**	0.803648	0.813537	0.832922	0.846796	0.297039	0.431122	0.384083	0.366257	**0.249901**
DP09	**0.845015**	0.798354	0.797557	0.817717	0.822851	0.350830	0.459478	0.400401	0.392729	**0.223465**
DP10	**0.783946**	0.778010	0.766020	0.761243	0.773076	**0.250989**	0.507426	0.492891	0.435765	0.353606

The findings from Table 7 indicate that the IMA outperforms the MIR-absent version (IMA1) in terms of the *HV* metric across 80 percent of the instances. Furthermore, across all examined instances, the IMA consistently demonstrates superior *IGD* metrics compared to the MIR-absent version (IMA1). This substantiates that our devised MIR mechanism enhances the algorithm's convergence and diversity.

These results unequivocally affirm the affirmative influence of the proposed MIR as a means to enhance the algorithm's performance.

5.3.2. Effect of the ESS

Next, we compared the performance of the proposed IMA with its variant, IMA2, which excludes the ESS component. The outcomes of these ablation experiments are summarized in Table 7.

The results from Table 7 are indicative: across 19 instances, the IMA consistently outperforms the version devoid of an ESS (IMA2) in terms of the *HV* metric. Particularly notable is the significant metric discrepancy across a majority of instances, underscoring the effectiveness and reliability of our ESS. Furthermore, the IMA consistently exhibited superior *IGD* metrics compared to IMA2, signifying that our introduced ESS augments the algorithm's overall performance.

5.3.3. Effect of the MSSA

Subsequently, we compared the performance of the proposed IMA with its variant, IMA3, which excludes the MSSA component. The results of these ablation experiments are summarized in Table 7.

From the data presented in Table 7, it becomes evident that the IMA slightly lags behind the version without the MSSA (IMA3) solely on the Mk02, Mk04 and Mk06 instances. However, across the remaining instances, the IMA consistently showcases commendable performance. This is especially apparent in the significant reduction of the *IGD* metric in the IMA across all instances compared to IMA3. Such observations affirm the pivotal role of our proposed MSSA in enhancing quality and promoting diversity.

5.3.4. Effect of the IVNS

Finally, we compared the performance of the proposed IMA with its variant, IMA4, which excludes the IVNS component. The results of these ablation experiments are summarized in Table 7. The findings from Table 7 indicate that, across nearly all instances, the IMA outperforms its counterpart without the IVNS (IMA4) in terms of both *HV* and *IGD* metrics. This observation strongly suggests that our proposed IVNS significantly enhances the algorithm's potential to explore and uncover improved solutions.

Figure 18 shows the Pareto frontier results obtained by running each component on the Mk06 instance 10 times. It is evident that each component has a certain impact on algorithm performance, with the MSSA and IVNS having the most significant impact. This suggests that the MSSA and IVNS have the greatest improvement in algorithm performance on this instance.

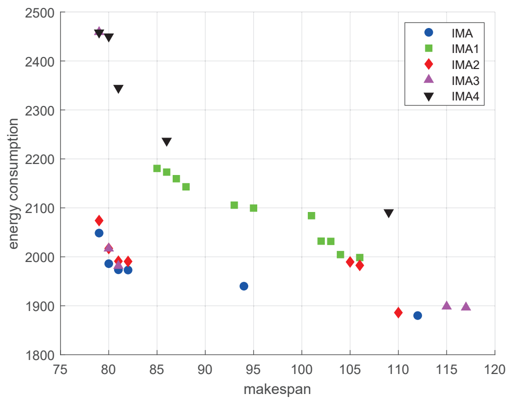


Figure 18. Pareto front results for ablation experiments on the Mk06 instance.

5.4. Comparison with other algorithms

To rigorously validate the efficacy of the proposed IMA in this research, we conducted a comprehensive comparison against a selection of classical and contemporary algorithms. The selected algorithms include the widely recognized multi-objective optimization algorithms, namely, NSGA-II and NSGA-III. These algorithms are established instances in the field due to their effectiveness and ease of replication. Additionally, we compared the proposed algorithms with two relatively recent algorithms, the HMMA [22] and SPAMA [25], both designed for MODFJSP optimization with the same objectives as this study. These algorithms are recognized for their excellence in multi-objective optimization.

To ensure a level playing field, all algorithms were independently executed on all test sets for 20 runs, using a consistent stopping condition of 200 maximum iterations. The comparative outcomes are presented in Table 8.

In Table 8, we have highlighted the largest HV metric in bold for each instance. It is evident that, for most instances, the HV metric of the IMA is the best and significantly ahead of the other four comparison algorithms, especially on the Mk03, Mk08 and DP04 instances.

This indicates that our proposed IMA is highly effective in solving the MODFJSP, and that our ESS also plays a very important role in reducing energy consumption and improving HV indicators. Although the HV metric on the DP07 and DP10 instances are lower than those for the SPAMA, the difference on DP10 is not significant. We analyzed that our IMA performed poorly on DP07 instance due to the unique nature of the DP07 instance, which resulted in our algorithm not fully leveraging its advantages.

Overall, the HV metric of our proposed IMA is optimal on most instances, indicating that the IMA exhibits superior algorithm performance.

Taking the Mk01 instance as an example, we recorded the average Pareto solution set obtained by each iteration of the five algorithms mentioned above. The horizontal axis in Figure 19 represents the number of iterations, and the vertical axis represents the maximum completion time.

We can see that the completion time of the IMA in the initial iteration stage is significantly lower than that of the other comparative algorithms, indicating that our proposed MIR strategy has played a very positive role in the population's initialization stage of the population.

Table 8. Statistical results for the HV metric for all algorithms in all instances.

Instance	HV				
	IMA	NSGA2	NSGA3	HMMA	SPAMA
Mk01	**0.697008**	0.464748	0.491488	0.522957	0.526256
Mk02	**0.650904**	0.491129	0.499831	0.542956	0.554878
Mk03	**0.672812**	0.302156	0.349842	0.349869	0.312427
Mk04	**0.720483**	0.468479	0.407806	0.486376	0.458702
Mk05	**0.612882**	0.394725	0.464664	0.489076	0.478961
Mk06	**0.775788**	0.350489	0.401084	0.319675	0.396394
Mk07	**0.595262**	0.460001	0.402988	0.488427	0.468144
Mk08	**0.525119**	0.198186	0.252408	0.300457	0.289718
Mk09	**0.547976**	0.136676	0.206897	0.176770	0.166425
Mk10	**0.714032**	0.167183	0.268791	0.194755	0.187758
DP01	**0.636001**	0.270433	0.214674	0.338950	0.326819
DP02	**0.721251**	0.425650	0.499945	0.556388	0.541867
DP03	**0.635148**	0.335208	0.386179	0.407674	0.395859
DP04	**0.656419**	0.358512	0.326319	0.436426	0.410603
DP05	**0.681177**	0.459597	0.409586	0.511175	0.498390
DP06	**0.720815**	0.466103	0.524809	0.548868	0.533089
DP07	0.483865	0.161575	0.132184	0.261754	**0.702817**
DP08	**0.860201**	0.276720	0.316925	0.364385	0.819594
DP09	**0.845015**	0.281028	0.342986	0.374806	0.817958
DP10	0.783946	0.217221	0.197483	0.280048	**0.838089**

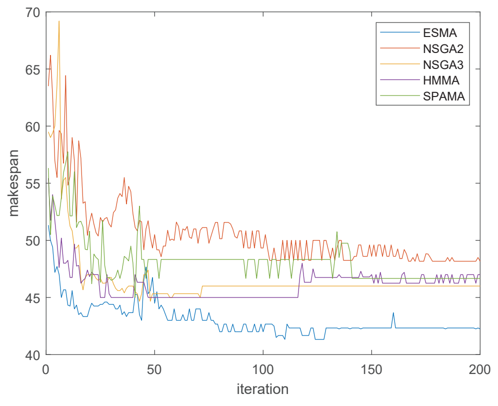


Figure 19. Convergence curve for makespan on Mk01.

Simultaneously, our proposed MSSA and IVNS strategies also, to some extent, prevent the population from falling into local optima too early; particularly, they maintain a certain degree of diversity in the middle and later stages of iteration.

The combination of the above multiple strategies has resulted in a more excellent Pareto solution set, demonstrating the effectiveness and excellent performance of the IMA.

In Table 9, we have highlighted the lowest *IGD* metric in bold for each instance. It is evident that, for most instances, the *IGD* metric of the IMA is the best and significantly ahead of the other four comparison algorithms, especially on the Mk01, Mk06 and DP03 instances.

This indicates that our proposed IMA is highly effective in solving the MODFJSP, and that our proposed MIR and IVNS strategies have played a crucial role in reducing *IGD* indicators. We infer that this is mainly due to increasing population diversity and improving population quality. Although the *IGD* indicators of the DP07 and DP09 instances are higher than those for the HMMA and SPAMA, the difference is small. Considering the normal volatility of the algorithm, these differences are acceptable.

Table 9. Statistical results for the IGD metric for all algorithms in all instances.

Instance	IGD				
	IMA	NSGA2	NSGA3	HMMA	SPAMA
Mk01	**0.063059**	0.230736	0.206119	0.192480	0.183226
Mk02	**0.064975**	0.209623	0.196690	0.150108	0.145771
Mk03	**0.123285**	0.384320	0.334850	0.342889	0.381302
Mk04	**0.172815**	0.233802	0.283277	0.216536	0.236514
Mk05	**0.067436**	0.245554	0.202496	0.191190	0.198707
Mk06	**0.020985**	0.438512	0.392742	0.407719	0.418003
Mk07	**0.092555**	0.180762	0.223791	0.166870	0.175088
Mk08	**0.198234**	0.391884	0.319164	0.297213	0.311293
Mk09	**0.192776**	0.532957	0.450175	0.481913	0.494134
Mk10	**0.020126**	0.659510	0.527398	0.634435	0.635972
DP01	**0.144596**	0.348890	0.396574	0.303244	0.316380
DP02	**0.130037**	0.285027	0.235042	0.186577	0.207190
DP03	**0.063712**	0.295624	0.279151	0.242072	0.263713
DP04	**0.153222**	0.269625	0.311625	0.209851	0.221640
DP05	**0.102756**	0.236842	0.254926	0.177740	0.190551
DP06	**0.118611**	0.251474	0.188846	0.171989	0.194196
DP07	0.311614	0.442047	0.506629	0.384155	**0.292972**
DP08	**0.297039**	0.391482	0.340065	0.298025	0.323837
DP09	0.350830	0.337049	0.275563	**0.250337**	0.363403
DP10	**0.250989**	0.423222	0.431548	0.347289	0.319615

Overall, the *IGD* metric of the proposed IMA is optimal on most instances, indicating that the IMA exhibits better diversity and convergence.

Taking the Mk01 instance as an example, we recorded the average Pareto solution set obtained by each iteration of the five algorithms mentioned. The x axis in Figure 20 represents the number of iterations and the y axis represents the energy consumption.

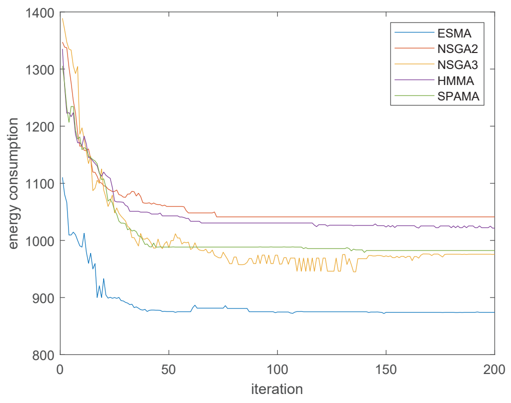


Figure 20. Convergence curve for energy consumption on Mk01.

It is evident that the energy consumption of the IMA in the initial iteration stage is significantly lower than that in the case of the other comparison algorithms, indicating that our proposed MIR and ESS have played a very positive role in the population initialization stage.

Simultaneously, our proposed MSSA and IVNS strategies also prevent the population from falling into local optima too early, thus providing more diversity during the iteration process.

From the Figure 20, it is evident that the energy consumption of the IMA is significantly lower than that for the other comparative algorithms at all stages of the iteration, indicating that our proposed ESS has played a significant role in effectively reducing energy consumption.

Figure 21 shows the Pareto front comparison of different algorithms. The solution set obtained via the IMA is closer to the coordinate origin, indicating that the IMA can yield a better solution set. Figures 22 and 23 provide Gantt charts for two factories on the Mk08 instance.

The combination of the above multiple strategies has resulted in a better Pareto solution set, proving the effectiveness and excellent performance of the IMA.

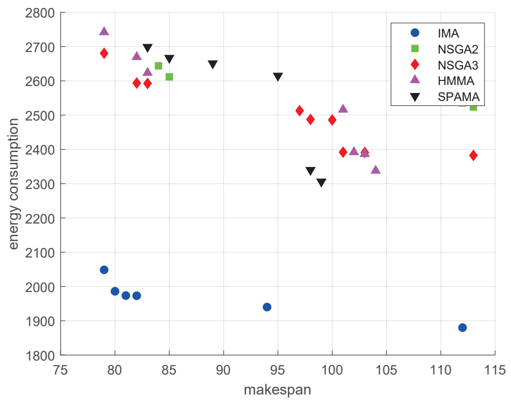


Figure 21. Pareto front results for different algorithms on the Mk06 instance.

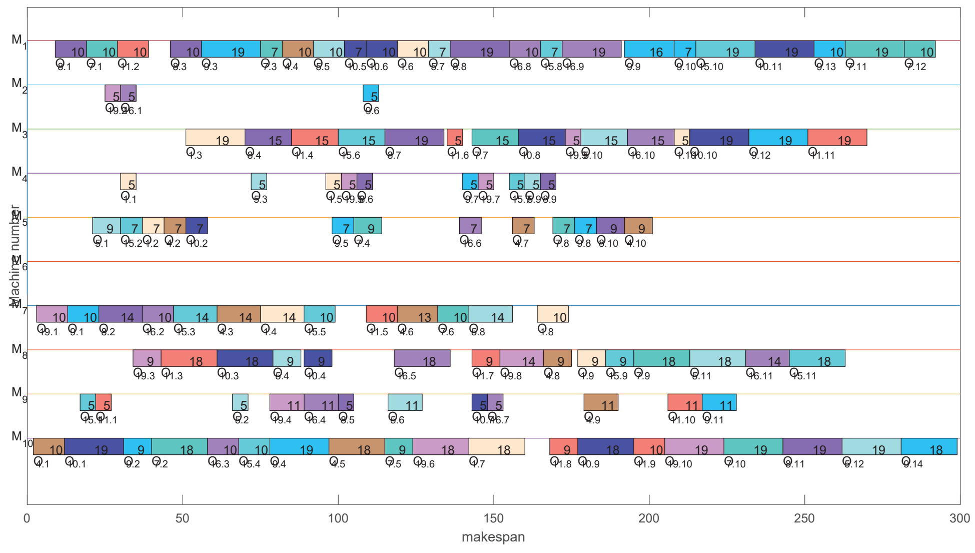


Figure 22. Gantt chart for Factory 1 based on Mk08 instance.

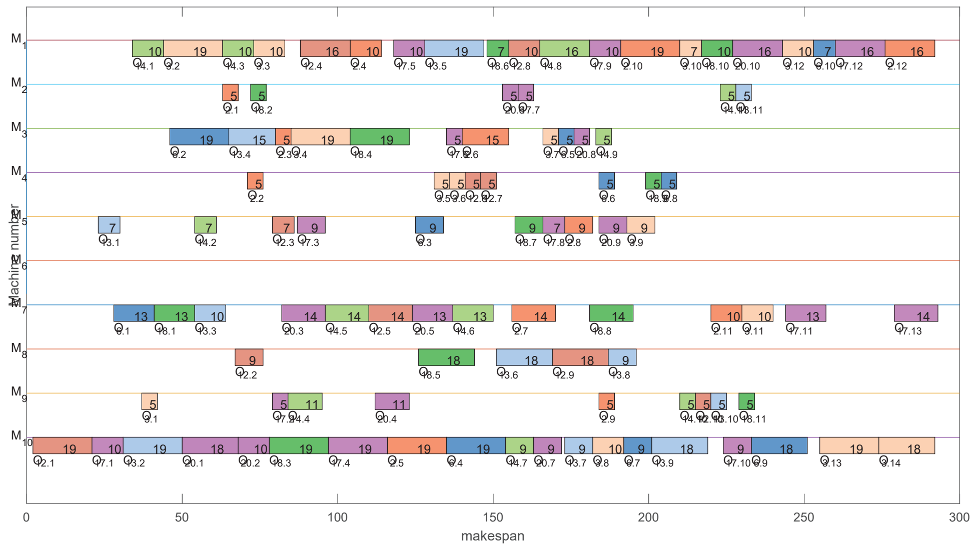


Figure 23. Gantt chart for Factory 2 based on Mk08 instance.

6. Conclusions and future studies

This paper introduces an IMA to tackle the challenges posed by the MODFJSP with transportation and start-stop constraints. The optimization objectives are defined as the maximum completion time and energy consumption. Considering the characteristics of the problem, we have formulated a multi-objective distributed flexible job shop scheduling model that incorporates transportation time and start-stop constraints. Subsequently, we have introduced an MIR and a novel multistart SA algorithm, integrating them into the IMA to enhance the algorithm's exploration ability and diversity. Moreover, we have designed four efficient neighborhood structures and two ESSs to improve search ability and decrease energy consumption. The effectiveness of the algorithm has been validated through evaluation and testing on 20 classic instances with diverse characteristics. Our aim with this research was to contribute to the improvement of the multi-objective distributed job shop scheduling system and offer guidance to workshop production managers.

Future work will focus on the application of the multi-objective FJSP in dynamic production scheduling environments, including scenarios such as machine failures and new order insertion. Additionally, we aspire for the algorithm to possess greater adaptability to confront more complex and ever-changing situations.

Use of AI tools declaration

The authors declare that they have not used artificial intelligence tools in the creation of this paper.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. J. Yang, H. Xu, J. Cheng, R. Li, Y. Gu, A decomposition-based memetic algorithm to solve the biobjective green flexible job shop scheduling problem with interval type-2 fuzzy processing time, *Comput. Ind. Eng.*, **183** (2023), 109513, https://doi.org/10.1016/j.cie.2023.109513
2. J. Xie, X. Li, L. Gao, L. Gui, A hybrid algorithm with a new neighborhood structure for job shop scheduling problems, *Comput. Ind. Eng.*, **169** (2022), 108205, https://doi.org/10.1016/j.cie.2022.108205
3. G. Zhang, J. Sun, X. Liu, G. Wang, Y. Yang, Solving flexible job shop scheduling problems with transportation time based on improved genetic algorithm, *Math. Biosci. Eng.*, **16** (2019),1334–1347. https://doi.org/10.3934/mbe.2019065
4. E. Jiang, L. Wang, Z. Peng, Solving energy-efficient distributed job shop scheduling via multi-objective evolutionary algorithm with decomposition, *Swarm Evol. Comput.*, **58** (2020), 100745. https://doi.org/10.1016/j.swevo.2020.100745
5. C. Lu, J. Zheng, L. Yin, R. Wang, An improved iterated greedy algorithm for the distributed hybrid flowshop scheduling problem, *Eng. Optim.*, (2023), 1–19. https://doi.org/10.1080/0305215X.2023.2198768
6. J. Yang, H. Xu, Hybrid memetic algorithm to solve multiobjective distributed fuzzy flexible job shop scheduling problem with transfer, *Processes*, **10** (2022), 1517. https://doi.org/10.3390/pr10081517
7. D. Lei, J. Cai, Multi-population meta-heuristics for production scheduling: A survey, *Swarm Evol. Comput.*, **58** (2020), 100739. https://doi.org/10.1016/j.swevo.2020.100739
8. Q. Liu, Q. Pan, L. Gao, X. Li, Multi-objective flexible job shop scheduling problem considering machine switching off-on operation, *Procedia Manuf.*, **39** (2019), 1167–1176. https://doi.org/10.1016/j.promfg.2020.01.353
9. S. Lin, K. Ying, Minimizing makespan and total flowtime in permutation flowshops by a bi-objective multi-start simulated-annealing algorithm, *Comput. Oper. Res.*, **40** (2013), 1625–1647. https://doi.org/10.1016/j.cor.2011.08.009
10. G. Zhang, J. Sun, X. Lu, H. Zhang, An improved memetic algorithm for the flexible job shop scheduling problem with transportation times, *Meas. Control*, **53** (2020), 1518–1528. https://doi.org/10.1177/0020294020948094
11. L. Asadzadeh, A local search genetic algorithm for the job shop scheduling problem with intelligent agents, *Comput. Ind. Eng.*, **85** (2015), 376–383. https://doi.org/10.1016/j.cie.2015.04.006
12. M. Kurdi, An effective new island model genetic algorithm for job shop scheduling problem, *Comput. Oper. Res.*, **67** (2016), 132–142. https://doi.org/10.1016/j.cor.2015.10.005
13. M. E. Aydin, T. C. Fogarty, A distributed evolutionary simulated annealing algorithm for combinatorial optimisation problems, *J. Heuristics*, **10** (2004), 269–292. https://doi.org/10.1023/B:HEUR.0000026896.44360.f9

14. J. Zhang, W. Wang, X. Xu, A hybrid discrete particle swarm optimization for dual-resource constrained job shop scheduling with resource flexibility, *J. Intell. Manuf.*, **28** (2017), 1961–1972. https://doi.org/10.1007/s10845-015-1082-0
15. I. Chaouch, O. B. Driss, K. Ghedira, A modified ant colony optimization algorithm for the distributed job shop scheduling problem, *Proc. Comput. Sci.*, **112** (2017), 296–305. https://doi.org/10.1016/j.procs.2017.08.267
16. S. Gopinath, C. Arumugam, T. Page, M. Chandrasekaran, Solving multi objective job shop scheduling problems using artificial immune system shifting bottleneck approach, *Appl. Mech. Mater.*, **766** (2015), 1209–1213. https://doi.org/10.4028/www.scientific.net/AMM.766-767.1209
17. L. Asadzadeh, A parallel artificial bee colony algorithm for the job shop scheduling problem with a dynamic migration strategy, *Comput. Ind. Eng.*, **102** (2016), 359–367. https://doi.org/10.1016/j.cie.2016.06.025
18. H. Piroozfard, K. Y. Wong, An imperialist competitive algorithm for the job shop scheduling problems, in *2014 IEEE International Conference on Industrial Engineering and Engineering Management*, (2014), 69–73. https://doi.org/10.1109/IEEM.2014.7058602
19. P. Moscato, *Memetic Algorithms: A Short Introduction*, McGraw-Hill, Maidenhead, 1999.
20. A. Phu-ang, A. Thammano, Memetic algorithm based on marriage in honey bees optimization for flexible job shop scheduling problem, *Memetic Comput.*, **9** (2017), 295–309. https://doi.org/10.1007/s12293-017-0230-9
21. C. Zhang, P. Li, Z. Guan, Y. Rao, A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem, *Comput. Oper. Res.*, **34** (2007), 3229–3242. https://doi.org/10.1016/j.cor.2005.12.002
22. R. Li, L. Wang, W. Gong, Knowledge-driven memetic algorithm for distributed green flexible job shop scheduling problem, *J. Huazhong Univ. Sci. Technol., Nat. Sci. Ed.*, **50** (2022), 55–60. https://doi.org/10.13245/j.hust.220606
23. C. Lu, L. Gao, W. Gong, C. Hu, X. Yan, X. Li, Sustainable scheduling of distributed permutation flow-shop with non-identical factory using a knowledge-based multi-objective memetic optimization algorithm, *Swarm Evol. Comput.*, **60** (2021), 100803. https://doi.org/10.1016/j.swevo.2020.100803
24. Y. Pan, K. Gao, Z. Li, N. Wu, Solving biobjective distributed flow-shop scheduling problems with lot-streaming using an improved Jaya algorithm, *IEEE Trans. Cybern.*, **53** (2023), 3818–3828. https://doi.org/10.1109/TCYB.2022.3164165
25. R. Li, W. Gong, L. Wang, C. Lu, X. Zhuang, Surprisingly popular-based adaptive memetic algorithm for energy-efficient distributed flexible job shop scheduling, *IEEE Trans. Cybern.*, **53** (2003), 8013–8023. https://doi.org/10.1109/TCYB.2023.3280175
26. Q. Luo, Q. Deng, G. Gong, L. Zhang, W. Han, K. Li, An efficient memetic algorithm for distributed flexible job shop scheduling problem with transfers, *Expert Syst. Appl.*, **160** (2020), 113721. https://doi.org/10.1016/j.eswa.2020.113721

27. H. Chang, T. Liu, Optimisation of distributed manufacturing flexible job shop scheduling by using hybrid genetic algorithms, *J. Intell. Manuf.*, **28** (2017), 1973–1986. https://doi.org/10.1007/s10845-015-1084-y

28. K. Zhu, G. Gong, N. Peng, L. Zhang, D. Huang, Q. Luo, et al., Dynamic distributed flexible job-shop scheduling problem considering operation inspection, *Expert Syst. Appl.*, **224** (2023), 119840. https://doi.org/10.1016/j.eswa.2023.119840

29. B. Marzouki, O. B. Driss, K. Ghédira, Solving distributed and flexible job shop scheduling problem using a chemical reaction optimization metaheuristic, *Procedia Comput. Sci.*, **126** (2018), 1424–1433. https://doi.org/10.1016/j.procs.2018.08.114

30. C. Lu, B. Zhang, L. Gao, J. Yi, J. Mou, A knowledge-based multiobjective memetic algorithm for green job shop scheduling with variable machining speeds, *IEEE Syst. J.*, **16** (2022), 844–855. https://doi.org/10.1109/JSYST.2021.3076481

31. F. Zhao, S. Di, L. Wang, A hyperheuristic with Q-learning for the multiobjective energy-efficient distributed blocking flow shop scheduling problem, *IEEE Trans. Cybern.*, **53** (2023), 3337–3350. https://doi.org/10.1109/TCYB.2022.3192112

32. C. Lu, L. Gao, J. Yi, X. Li, Energy-efficient scheduling of distributed flow shop with heterogeneous factories: A real-world case from automobile industry in China, *IEEE Trans. Ind. Inf.*, **17** (2021), 6687–6696. https://doi.org/10.1109/TII.2020.3043734

33. X. Gong, T. De Pessemier, L. Martens, W. Joseph, Energy- and labor-aware flexible job shop scheduling under dynamic electricity pricing: A many-objective optimization investigation, *J. Cleaner Prod.*, **209** (2019), 1078–1094. https://doi.org/10.1016/j.jclepro.2018.10.289

34. X. Wu, Y. Sun, A green scheduling algorithm for flexible job shop with energy-saving measures, *J. Cleaner Prod.*, **172** (2018), 3249–3264. https://doi.org/10.1016/j.jclepro.2017.10.342

35. N. Rakovitis, D. Li, N. Zhang, J. Li, X. Xiao, Novel approach to energy-efficient flexible job-shop scheduling problems, *Energy*, **238** (2022), 121773. https://doi.org/10.1016/j.energy.2021.121773

36. E. Xu, Y. Li, Y. Liu, J. Du, X. Gao, Energy saving scheduling strategy for job shop under TOU and tiered electricity price, *Alexandria Eng. J.*, **61** (2022), 459–467. https://doi.org/10.1016/j.aej.2021.06.008

37. Y. Li, W. Huang, R. Wu, K. Guo, An improved artificial bee colony algorithm for solving multi-objective low-carbon flexible job shop scheduling problem, *Appl. Soft Comput.*, **95** (2020), 106544. https://doi.org/10.1016/j.asoc.2020.106544

38. L. Wang, D. Zheng A modified genetic algorithm for job shop scheduling, *Int. J. Adv. Manuf. Technol.*, **20** (2002), 72–76. https://doi.org/10.1007/s001700200126

39. K. Ying, S. Lin, Minimizing total completion time in the no-wait jobshop scheduling problem using a backtracking metaheuristic, *Comput. Ind. Eng.*, **169** (2022), 108238. https://doi.org/10.1016/j.cie.2022.108238

40. G. Gong, R. Chiong, Q. Deng, Q. Luo, A memetic algorithm for multi-objective distributed production scheduling: minimizing the makespan and total energy consumption, *J. Intell. Manuf.*, **31** (2020), 1443–1466. https://doi.org/10.1007/s10845-019-01521-9

41. K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, Q. Pan, A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems, *IEEE/CAA J. Autom. Sin.*, **6** (2019), 904–916. https://doi.org/10.1109/JAS.2019.1911540

42. I. Chaouch, O. B. Driss, K. Ghedira, A novel dynamic assignment rule for the distributed job shop scheduling problem using a hybrid ant-based algorithm, *Appl. Intell.*, **49** (2019), 1903–1924. https://doi.org/10.1007/s10489-018-1343-7