axioms

MDPI

Article

A New Interior Search Algorithm for Energy-Saving Flexible Job Shop Scheduling with Overlapping Operations and Transportation Times

Lu Liu [1,2], Tianhua Jiang [1,2,3,*], Huiqi Zhu [1,2] and Chunlin Shang [1,2]

[1] School of Transportation, Ludong University, Yantai 264025, China; tliulut@126.com (L.L.); zhuhuiqi0505@126.com (H.Z.); shangchunlin007@126.com (C.S.)
[2] Shandong Marine Aerospace Equipment Technological Innovation Center, Yantai 264025, China
[3] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China
* Correspondence: jth1127@163.com

Abstract: Energy-saving scheduling has been pointed out as an interesting research issue in the manufacturing field, by which energy consumption can be effectively reduced through production scheduling from the operational management perspective. In recent years, energy-saving scheduling problems in flexible job shops (ESFJSPs) have attracted considerable attention from scholars. However, the majority of existing work on ESFJSPs assumed that the processing of any two consecutive operations in a job cannot be overlapped. In order to be close to real production, the processing overlapping of consecutive operations is allowed in this paper, while the job transportation tasks are also involved between different machines. To formulate the problem, a mathematical model is set up to minimize total energy consumption. Due to the NP-hard nature, a new interior search algorithm (NISA) is elaborately proposed following the feature of the problem. A number of experiments are conducted to verify the effectiveness of the NISA algorithm. The experimental results demonstrate that the NISA provides promising results for the considered problem. In addition, the computational results indicate that the increasing transportation time and sub-lot number will increase the transportation energy consumption, which is largely responsible for the increase in total energy consumption.

Keywords: flexible job shop; overlapping operation; job transportation; total energy consumption; interior search algorithm

MSC: 97P10

check for updates

Citation: Liu, L.; Jiang, T.; Zhu, H.; Shang, C. A New Interior Search Algorithm for Energy-Saving Flexible Job Shop Scheduling with Overlapping Operations and Transportation Times. *Axioms* **2022**, *11*, 306. https://doi.org/10.3390/axioms11070306

Academic Editor: Nodari Vakhania

Received: 30 May 2022
Accepted: 19 June 2022
Published: 24 June 2022

1. Introduction

Suffering from growing energy costs and a worsening ecological environment, it is quite necessary to adopt some measures to achieve energy saving and consumption reduction. Reviewing this situation, some managers try to purchase more energy-efficient equipment, others attempt to redesign the products. However, these previous attempts inevitably impose substantial capital investment, which is impossible for some small and medium companies to afford such extra financial burden. In view of this fact, many researchers have turned their attention to reasonable operational management for the reduction of energy consumption. Energy-saving scheduling has been viewed as one of the most effective manners by researchers all around the world. A number of research achievements have been yielded for various manufacturing workshops, such as single machine [1–5], parallel machines [6–11], flow shop [12–18], and job shop [19–25].

In recent years, considering the high theoretical complexity and strong application background, the energy-saving flexible job shop scheduling problem (ESFJSP) has become a new research hotspot in the manufacturing field. Wu et al. [26] investigated the ESFJSP with the consideration of a deterioration effect. A multi-objective mathematical model was established to minimize total energy consumption and makespan. Then, pigeon-inspired optimization and simulated annealing algorithm are hybridized for the problem. Caldeira et al. [27] addressed an ESFJSP with new job arrivals and turning on/off of the mechanism. A mathematical model was built to optimize the makespan, energy consumption, and instability simultaneously. An improved backtracking search algorithm was proposed to obtain the trade-off among the objectives. Dai et al. [28] formulated an ESFJSP with transportation constraints to optimize energy consumption and makespan. Then, an improved genetic algorithm was presented to solve the problem. Yin et al. [29] proposed a mathematical model with the consideration of flexible machining speed. A multi-objective genetic algorithm was developed to optimize productivity, energy efficiency, and noise reduction simultaneously. Liu et al. [30] addressed an ESFJSP with crane transportation. A mixed-integer programming model was built to get the trade-off between energy consumption and makespan. Then, an integrated algorithm, consisting of a genetic algorithm, glowworm swarm optimization, and green transport heuristic strategy, was presented for the proposed model. Jiang et al. [31] investigated an ESFJSP considering the sequence-dependent setup times and proposed an improved African buffalo optimization to get the minimum total energy consumption. Li and Lei [32] reported an ESFJSP with transportation and sequence-dependent setup times and proposed an imperialist competitive algorithm with feedback to minimize the makespan, total tardiness and total energy consumption simultaneously. Zhang et al. [33] studied an ESFJSP aiming to minimize makespan and total energy consumption. A multi-objective model was formulated with the consideration of sequence-dependent setup and transportation times. Then, an effective novel heuristic method was proposed to solve the problem. Gong et al. [34] proposed a multi-objective mathematical model of a double flexible job shop scheduling problem considering the indicators of processing time, green production and human factor. Then, a hybrid genetic algorithm was presented for the model. Peng et al. [35] addressed a multi-objective ESFJSP with job transportation and learning effect. A hybrid discrete multi-objective imperial competition algorithm was developed to solve the problem. Zhu et al. [36] considered an ESFJSP considering worker learning effect and proposed a memetic algorithm to minimize the makespan, total carbon emission and total cost of workers. Wei et al. [37] addressed an energy-efficient FJSP with the consideration of variable machine speed. Then some hybrid energy-efficient scheduling measures are developed to minimize the makespan and total energy consumption. Jiang et al. [38] established a mathematical model of an ESFJSP considering job transportation and deterioration. The animal migration optimization algorithm was modified to minimize the total energy consumption. Jiang et al. [39] considered a dual-resource constrained ESFJSP and proposed a discrete animal migration optimization algorithm to get the minimum total energy consumption.

Regarding the above literature, the ESFJSP problems have attracted considerable attention from scholars. Some research endeavors have been undertaken to narrow the gap between the scheduling problem and production application. Various practical factors have been investigated in the previous work, e.g., job deterioration effect [26,38], new job arrivals and turning on/off strategy [27], job transportation constraints [28,30,32,33,35,38], flexible machining speed [29,37], machine setup times [31–33] and double-resource constraint [34–36,39]. However, the above previous ESFJSP problems usually assume that the processing of the successful operation of a job cannot be started until its predecessor is finished completely. That is, the processing of any two consecutive operations of the same job is not permitted to be overlapped. In some real-life industries, e.g., metal or automobile industries, a job (lot) can always be divided into several similar parts (sublots), each of which can be dealt with individually. For such a job, it does not need to be transferred to the next step until all sublots are completed. In this situation, consecutive operations of a job can be overlapped for processing. The overlapping in operations are illustrated in Figure 1. This strategy is often reported as lot streaming in the existing literature [40]. One of the most important advantages of this strategy is the improvement of production efficiency. Demir and İşleyen [41] formulated the FJSP with the consideration of overlapping operations. A genetic algorithm was proposed for solving the problem. Meng et al. [42] designed a hybrid artificial bee colony for the FJSP with overlapping operations to minimize the total flow time. However, they only focus on improving the production efficiency and neglect the energy consumption generated in the manufacturing process. As far as we know, up to now, an ESFJSP with overlapping operations is seldom studied in the published literature. Furthermore, according to the above review, job transportation has been frequently considered in the ESFJSP [28,30,32,33,35,38]. The main reason for this situation is that there exist some strong interactions between production and transportation tasks in practical manufacturing. On the one hand, the machine selection of two consecutive operations in a job determines the transportation time. On the other hand, the transportation tasks could affect the idle times of machines in terms of different operation sequences. Furthermore, the energy consumption that occurs in the transportation process is nonnegligible. Based on the above review, in this study, the overlapping of operations and transportation times between machines are simultaneously considered in the ESFJSP problem.

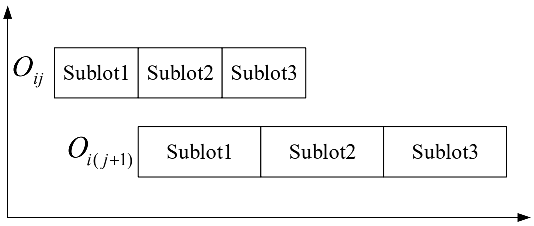


Figure 1. The overlapping in operations.

Observed from the reviewed literature, the general solution process is to first establish a mathematical model with the expected objectives and the related constraints. Afterward, an effective algorithm is designed to solve the problem. For the ESFJSPs models, some were built as standard mathematical program models, e.g., mixed-integer linear programming [27,28,33], mixed-integer programming [29,30]; others were not converted to standard forms or the authors did not state the model type clearly [26,31,32,34–39]. In general, the solution methods for workshop scheduling problems fall into two main categories: exact and approximate methods [43]. However, the ESFJSP is essentially an extended version of the traditional FJSP, consisting of many jobs, machines and objectives, which is inefficient to solve exactly and more suitable to be solved by approximate methods [43]. In recent years, many meta-heuristics have been developed to deal with the ESFJSPs, e.g., pigeon-inspired optimization [26], genetic algorithm [28–30,34], glowworm swarm optimization [30], African buffalo optimization [31], imperialist competitive algorithm [32,35], memetic algorithm [36], animal migration optimization [38,39]. Nevertheless, none of them can be guaranteed to outperform other algorithms in all types of ESFJSPs. This fact is in line with the famous No Free Lunch Theorem [44], which inspires scholars to continuously present new algorithms or modify existing ones.

Interior search algorithms (ISA) are a novel meta-heuristic algorithm that originated from aesthetic behaviors of interior design and decoration [45]. Since it was proposed, ISA has attracted increasing interest in dealing with various complex optimization problems [46–50]. In the ISA algorithm, individuals are split into two groups with the exception of the fittest one, namely the composition group and the mirror group. These two groups are in charge of global exploration and local exploitation, respectively. A specific controlling parameter α is employed to select the group for each individual. This search mechanism gives a fine opportunity for the implementation of the cooperation between exploration and exploitation, which motivates us to select the ISA for the considered problem. Our main work is summarized as follows: (1) A mathematical model is built for the ESFJSP with the consideration of overlapping operations and transportation times simultaneously. (2) To solve the model, a NISA algorithm is elaborately designed based on the characteristics of the problem. The design work mainly includes encoding/decoding, population initialization, discrete composition optimization, discrete mirror search, tuning of parameter α and random walk. (3) Extensive experiments are performed to validate the competitive performance of the NISA algorithm and analyze the effect of transportation time and sublot number.

The remainder of this paper is organized as follows. Section 2 reports the mathematical model of the ESFJSP with overlapping operations and transportation times. Section 3 implements the NISA algorithm. Section 4 assesses the performance of the NISA algorithm. Section 5 reports the conclusion and next work.

2. Problem Description and Mathematical Model

2.1. Problem Description

In the considered problem, n jobs are expected to be processed on m machines. For each job, J_i operations are sequentially processed in a certain order. For processing an operation, any machine can be selected from the operation's compatible machine set. The capacity of the selected machine determines the processing time of each operation on the machine. In this work, each job is split into s_i sublots with equal size. Once the processing of each sublot is finished, it will be transferred to another machine. It assumes that there are enough transporters equipped in the workshop. Meanwhile, the transporter can convey one sublot at a time, and the transportation times are known in advance. In the considered ESFJSP, the optimization objective is to get the minimum of the total energy consumption (TTEC), which contains processing energy consumption (PEC), idle energy consumption (IEC), transportation energy consumption (TEC) and common energy consumption (CEC). PEC is generated by machines when processing operations, IEC occurs when a machine is waiting for processing, TEC is consumed by transporters and CEC is consumed by accessory equipment.

Some assumptions are necessary as follows: jobs are released and machines are available at time 0; the setup times of machines are contained in the processing times; machine breakdowns are not considered; each machine cannot process two or more sublots at a given time; the number of sublots in each job is known in advance and fixed; for each operation, all sublots must be performed on the same machine; for each operation, no preemption is permitted; each machine cannot be shut down unless all jobs assigned to it are finished.

2.2. Mathematical Model

i: Index of jobs, $i = 1, 2, 3, \cdots, n$;
k: Index of machines, $k = 1, 2, 3, \cdots, m$;
J_i: Number of operations contained in job i, $j = 1, 2, 3, \cdots, J_i$;
s_i: Total number of sublots of job i, $l = 1, 2, 3, \cdots, s_i$;
O_{ij}: The jth operation of job i;
O_{ijl}: The lth sublot of O_{ij};
p_{ijlk}: Processing time of O_{ijl} on machine k;
$TTEC$: Total energy consumption;
PE_{ijlk}: The PEC coefficient of O_{ijl} on machine k;
SE_k: The IEC coefficient of machine k in idle state;
CE: The CEC coefficient;
TE: The TEC coefficient;
C_k: Completion time of machine k;
S_k: Start time of machine k;
WL_k: Workload of machine k, the total processing times of jobs assigned to machine k;
$C_{\max}$: Makespan;
$TT_{i(j-1)lk,ijlw}$: Transportation time between machine k and w for $O_{i(j-1)l}$ and O_{ijl};
ST_{ijl}: Starting time of O_{ijl};
CT_{ijl}: Completion time of O_{ijl};

Γ: A large positive number;
x_{ijk}: 0–1 variable, if O_{ij} is assigned to machine k, x_{ijk} = 1; otherwise, x_{ijk} = 0;
z_{ijlk}: 0–1 variable, if O_{ijl} is assigned to machine k, z_{ijlk} = 1; otherwise, z_{ijlk} = 0;
$y_{iji'j'k}$: 0–1 variable, if O_{ij} precedes $O_{i'j'}$ on machine k, $y_{iji'j'k}$ = 1; otherwise, $y_{iji'j'k}$ = 0.

Jiang et al. [38] established the mathematical model of the energy-saving flexible job shop scheduling problem with the transportation time and deterioration effect. However, the overlapping in operations is not considered in their model. Here, we refer to their works to define the energy consumption and deal with the transportation constraints. The mathematical model of the ESFJSP with overlapping operations and transportation times is built as Equations (1)–(15).

$$\min TTEC = \sum_{i=1}^{n}\sum_{j=1}^{J_i}\sum_{l=1}^{s_i}\sum_{k=1}^{m} PE_{ijlk}p_{ijlk}z_{ijlk} + \sum_{k=1}^{m} SE_k(C_k - S_k - WL_k) + \sum_{i=1}^{n}\sum_{j=2}^{J_i}\sum_{l=1}^{s_i}\sum_{w=1}^{m}\sum_{k=1}^{m} TE \cdot TT_{i(j-1)lw,ijlk}z_{i(j-1)lw}z_{ijlk} + CE \times C_{\max} \quad (1)$$

$$\text{s.t.} \quad CT_{ijl} - ST_{ijl} = \sum_{k=1}^{m} z_{ijlk}p_{ijlk}, \ \ i = 1,2,\cdots,n; \ j = 1,2,\cdots,J_i; \ l = 1,2,\cdots,s_i \quad (2)$$

$$\sum_{k=1}^{m} x_{ijk} = 1, \quad i = 1,2,\cdots n; \ \ j = 1,2,\cdots,J_i \quad (3)$$

$$\sum_{l=1}^{s_i} z_{ijlk} = s_i \times x_{ijk}, \quad i = 1,2,\cdots n; \ \ j = 1,2,\cdots,J_i; \ \ k = 1,2,\cdots,m \quad (4)$$

$$ST_{ijl} - CT_{ij(l-1)} \geq 0, \quad i = 1,2,\cdots,n; \ \ j = 1,2,\cdots,J_i; \ \ l = 2,\cdots,s_i \quad (5)$$

$$ST_{ijl} \geq CT_{i(j-1)l} + \sum_{w=1}^{m}\sum_{k=1}^{m} TT_{i(j-1)lk,ijlw}z_{i(j-1)lk}z_{ijlw}, \quad i = 1,2,\cdots,n; \ \ j = 2,\cdots,J_i; \ \ l = 1,2,\cdots,s_i \quad (6)$$

$$ST_{i'j'1} + \Gamma(1 - y_{iji'j'k}) \geq CT_{ijs_i}, \quad i,i' = 1,2,\cdots,n; j = 1,2,\cdots,J_i; j' = 1,2,\cdots,J_{i'}; k = 1,2,\cdots,m \quad (7)$$

$$ST_{ij1} + \Gamma y_{iji'j'k} \geq CT_{i'j's_{i'}}, \quad i,i' = 1,2,\cdots,n; j = 1,2,\cdots,J_i; j' = 1,2,\cdots,J_{i'}; k = 1,2,\cdots,m \quad (8)$$

$$WL_k = \sum_{i=1}^{n}\sum_{j=1}^{J_i}\sum_{l=1}^{s_i} p_{ijlk}z_{ijlk}, \ \ k = 1,2,\cdots,m \quad (9)$$

$$C_k = \max\left\{CT_{ijl}z_{ijlk}\right\}, \ \ i = 1,2,\cdots,n; j = 1,2,\cdots,J_i; l = 1,2,\cdots,s_i; \ k = 1,2,\cdots,m \quad (10)$$

$$S_k = \min\left\{ST_{ijl}z_{ijlk}\right\}, \ \ i = 1,2,\cdots,n; j = 1,2,\cdots,J_i; l = 1,2,\cdots,s_i; k = 1,2,\cdots,m \quad (11)$$

$$ST_{ijl} \geq 0, \ \ i = 1,2,\cdots,n; j = 1,2,\cdots,J_i; l = 1,2,\cdots,s_i \quad (12)$$

$$x_{ijk} \in \{0,1\}, \quad i = 1,2,\cdots,n; j = 1,2,\cdots,J_i; \ k = 1,2,\cdots,m \quad (13)$$

$$z_{ijlk} \in \{0,1\}, \quad i = 1,2,\cdots,n; j = 1,2,\cdots,J_i; l = 1,2,\cdots,s_i; \ k = 1,2,\cdots,m \quad (14)$$

$$y_{iji'j'k} \in \{0,1\}, \quad i,i' = 1,2,\cdots,n; j = 1,2,\cdots,J_i; j' = 1,2,\cdots,J_{i'}; \ k = 1,2,\cdots,m \quad (15)$$

Equation (1) calculates the total energy consumption; Constraint (2) defines that no interruption is allowed during the processing of each sublot; Constraint (3) denotes that any operation must be assigned to only one machine; Constraint (4) guarantees that the number of sublots O_{ijl} processing on machine k equals the total sublot number of O_{ij}. Constraint (5) gives the precedence relationships between the sublots of each operation, i.e., the lth sublot of O_{ij} must be started after the $(l-1)$th sublot is completed; Constraint (6) defines the precedence relationships between the operations belonging to the same job, i.e., the lth sublot of O_{ij} must be started after the lth sublot of $O_{i(j-1)}$ is completed and transported to the next machine. Constraints (7) and (8) are disjunctive constraints where only one constraint can be met. That is, O_{ij} and $O_{i'j'}$ assigned to machine k cannot be processed simultaneously; Constraint (9) denotes the machine workload; Constraints (10) and (11) define the machine's completion time and start time; Constraint (12) means that the start time of each operation is not smaller than zero; Constraints (13)–(15) state 0–1 variables.

3. Overview of the Basic ISA Algorithm

The interior search algorithm (ISA) mimics the behavior of an interior designer and decorator. There are mainly two search operators that are contained in the algorithm, i.e., composition optimization and mirror search. In every iteration, individuals are split into two groups, namely the composition group and the mirror group. In the composition group, the position of each individual is changed randomly in the feasible space. In the mirror group, for each individual, a mirror is first located near the global best solution, and then a new position will be generated depending on the information of the mirror. The steps of the basic ISA algorithm are presented below.

Step 1. Randomly generate the initial positions of individuals within the restriction of lower and upper bounds.

Step 2. Evaluate each individual and find out the current global best solution $\boldsymbol{X}_{gb}^t$.

Step 3. For each individual, a randomly generated number $r_1 \in [0,1]$ will be compared with a controlling parameter α. If $r_1 \leq \alpha$, the individual goes to the mirror group; otherwise, it is allocated to the composition group.

Step 4. For the global best individual $\boldsymbol{X}_{gb}^t$, it is changed using a random walk as local search. This process can be formulated by Equation (16). t is the current iteration number. $\boldsymbol{r}_n$ is a random number vector with normal distribution, and λ is a scale factor.

$$\boldsymbol{X}_{gb}^t = \boldsymbol{X}_{gb}^{t-1} + \boldsymbol{r}_n \times \lambda \tag{16}$$

Step 5. For the composition group, the individual position is randomly changed in a limited search space, which is represented by Equation (17). $\boldsymbol{X}_i^t$ is the ith individual in the tth iteration. $\boldsymbol{LB}^t$ and $\boldsymbol{UB}^t$ are the lower and upper bounds of the composition group elements in tth iteration, respectively, and they are the vector of minimum and maximum values of each dimension of all individuals in $(t-1)$th iteration. r_2 is a random number between 0 and 1.

$$\boldsymbol{X}_i^t = \boldsymbol{LB}^t + (\boldsymbol{UB}^t - \boldsymbol{LB}^t) \times r_2 \tag{17}$$

Step 6. For the mirror group, a mirror is randomly located between each individual and the global best individual following Equation (18). $\boldsymbol{X}_{m,i}^t$ is the mirror position of ith individual in the tth iteration. r_3 is a random number between 0 and 1. The position of each individual is updated following the mirror's position, which can be represented by Equation (19).

$$\boldsymbol{X}_{m,i}^t = r_3 \boldsymbol{X}_i^{t-1} + (1 - r_3)\boldsymbol{X}_{gb}^t \tag{18}$$

$$\boldsymbol{X}_i^t = 2\boldsymbol{X}_{m,i}^t - \boldsymbol{X}_i^{t-1} \tag{19}$$

Step 7. Evaluate each new individual. If it is superior to the original one, accept it; otherwise, keep the original position unchanged.

Step 8. Determine whether the stop condition is satisfied. If yes, go to Step 9; otherwise, go to Step 2.

Step 9. Output the results.

4. Implementation of the NISA

4.1. Encoding and Decoding Approach

Similar to other meta-heuristics, one of the key issues is to design an encoding approach to implement the conversion between the solution space and the search space. In this paper, the problem is contained by machine assignment (MA) and operation permutation (OP). To represent the scheduling solutions, an encoding scheme with two vectors is employed to indicate the information of MA and OP. In the MA vector, a machine is chosen from the compatible machine set of each operation. In the OP vector, operations are sequenced to represent the precedence relationships on machines.

To illustrate the encoding approach, an instance with three jobs and three machines is given in Figure 2. Each job contains three operations. In the MA vector, each integer number corresponds to the index of the machine for an operation. In the OP vector, each integer number corresponds to the job code. The appearance times of a job code represent the number of operations contained in the job. Figure 1 gives the scheduling solution as follows: $(O_{11}, M_2) \rightarrow (O_{12}, M_1) \rightarrow (O_{21}, M_3) \rightarrow (O_{13}, M_3) \rightarrow (O_{22}, M_1) \rightarrow (O_{31}, M_3) \rightarrow (O_{32}, M_2) \rightarrow (O_{23}, M_1) \rightarrow (O_{33}, M_2)$.

O_{11}	O_{12}	O_{13}	O_{21}	O_{22}	O_{23}	O_{31}	O_{32}	O_{33}
2	1	3	3	1	1	3	2	2

Machine Assignment

O_{11}	O_{12}	O_{21}	O_{13}	O_{22}	O_{31}	O_{32}	O_{23}	O_{33}
1	1	2	1	2	3	3	2	3

Operation Permutation

Figure 2. The encoding scheme.

Following the above scheduling scheme, the start times and completion times of all sublots in each operation can be determined in the decoding process. For each sublot O_{ijl}, it cannot be processed until some necessary conditions must be satisfied: (1) The assigned machine k of O_{ijl} must be available, and the available time is represented by mt_k; (2) If $j = 1$, O_{i1l} can be immediately started once the assigned machine k is available, i.e., $ST_{i1l} = mt_k$; (3) If $j > 1$, O_{ijl} must be started after $O_{i(j-1)l}$ is finished and then transported from machine k to w, i.e., $ST_{ijl} = \max(mt_k, C_{i(j-1)l} + TT_{i(j-1)lk,ijlw})$. The completion time of O_{ijl} can be measured by $CT_{ijl} = ST_{ijl} + \sum_{k=1}^{m} z_{ijlk} p_{ijlk}$.

4.2. Population Initialization

For a meta-heuristic algorithm, generating the initial population is vital for the convergence speed and solution quality. Based on the above encoding structure, the initial solutions will be created separately for the two vectors.

To obtain a machine assignment scheme, three heuristic rules [51] are randomly adopted to choose a machine from each operation's compatible machine set, i.e., Global Selection (GS), Local Selection (LS) and Random Rule (RR).

For a given machine assignment, three dispatching rules [52] are randomly applied to sequence operations on machines, i.e., Most Work Remaining (MWR), Most Number of Operations Remaining (MOR) and Random Rule.

4.3. Discrete Composition Optimization

As can be seen from Equation (17), each individual updates its position randomly within a limited search space, which is derived from the individuals in the composition group. In the ISA algorithm, the composition optimization operator plays the role of global search. However, as observed from Equation (17), it cannot be applied to solving the discrete scheduling problem in this paper. Thus, the original composition optimization operator should be amended to adapt to the considered problem. It is well-known that the crossover operation is used to explore the search space and finding the global optimal solution. In view of this consideration, we develop a crossover-based component optimization operator to acquire new individuals. In order to implement it, an individual is randomly selected from the composition group at first. Then, crossover operations are carried out between the current individual and the selected one.

Based on the characteristics of the problem, two types of crossover operators [53] are employed for the two vectors of a scheduling solution. One type is used for the OP vector, i.e., precedence preserving order-based crossover (POX) and job-based crossover (JBX), while the other is employed for the MA vector, i.e., two-point crossover (TPX) and multi-point crossover (MPX). When performing the component optimization operator, one crossover operator is randomly selected from each of the two types to act on the two vectors. It is notable that two offspring individuals will be generated by the crossover operation. After evaluating their fitness, the better offspring will be judged on whether to join the composition group or not. If the offspring is superior to the current individual, it will be accepted to replace the current individual. Otherwise, the current individual will remain unchanged.

4.4. Discrete Mirror Search

For each individual in the mirror group, a mirror is first randomly placed near the global best individual, and then the current individual is updated by absorbing the information from the mirror. However, according to Equations (18) and (19), the original mirror search operator is also unsuitable for the considered problem. Therefore, some modifications need to be conducted following the characteristics of the problem. Here, we present a neighborhood-crossover-based mirror search operator, which can be implemented below.

For each individual, two types of neighborhood structures are first randomly performed on the global best individual to generate a mirror. After the mirror generation, a crossover operation is performed between the current individual and the mirror to obtain a new individual. Herein, we employ $\lceil \lambda \rceil$ to represent the execution times of neighborhood operation. If λ is large, the mirror may drop into the remote area of the global best; otherwise the mirror locates near the global best. Therefore, λ determines the degree of exploitation of the mirror search operator. In this paper, the value of λ is dynamically adjusted along with the iteration process. In the early iteration of the algorithm, individuals are updated by learning from the mirrors that are far away from the global best individual, and in the later iteration, individuals are updated by learning from the mirrors close to the global best individual. This adjustment process of λ can be formulated by Equation (20), where t is the current iteration number; $t_{\max}$ represents the maximum iteration number; $\lambda_{\min}$ and $\lambda_{\max}$ represent the minimum and maximum values of λ, which are set to be 1 and 5, respectively.

$$\lambda = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min}) \times (t_{\max} - t)/t_{\max} \tag{20}$$

When performing the crossover operations, the POX and JBX are randomly selected for the OP vector, and the TPX and MPX are randomly selected for the MA vector. In addition, the neighborhood structures mentioned above are described below.

(1) Type 1 for machine assignment

TMA1: Randomly choose a position in the MA vector and randomly choose a different machine from the compatible machine set of the selected operation to take the place of the original machine.

TMA2: Randomly choose a position in the MA vector and choose the machine with the shortest processing time from the compatible machine set of the selected operation to replace the original machine.

TMA3: Randomly choose a position in the MA vector and choose the machine with the smallest PEC coefficient from the compatible machine set of the selected operation to replace the original machine.

(2) Type 2 for operation permutation

TOP1: Randomly select two positions with different values in the OP vector and swap their values.

TOP2: Randomly select two positions in the OP vector and insert the second position in front of the first one.

TOP3: Randomly select two positions in the OP vector and invert the values between the two positions.

4.5. Tuning of Parameter α

In the basic ISA, individuals are divided into two groups controlled by parameter α, which determines the degree of emphasis on exploration and exploitation during the iteration search process. That is, if parameter α has a small value, more individuals join the composition group, and the algorithm has a stronger exploration capacity. Otherwise, the algorithm emphasizes the exploitation capacity. To acquire a balance between exploration and exploitation, Gandomi and Roke [54] proposed a linear adjustment approach of α in Equation (21), which indicates that the search focuses on exploration by using composition optimization at the early stage, and then it is gradually switched to mirror search to emphasize exploitation at the latter stage.

$$\alpha = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})t/t_{\max} \tag{21}$$

4.6. Random Walk

In the ISA algorithm, a random walk acts as a local search to boost the local search capacity of the algorithm around the global best individual. To this end, a local search algorithm is constructed on the basis of the neighborhood structures in Section 4.4. The steps of the local search algorithm are stated below.

Step 1. Set the current global best solution as the initial solution.
Step 2. Set $\zeta \leftarrow 1$.
Step 3. Perform two neighborhood structures on the MA and the OP vectors, respectively. For the two neighborhood structures, one is randomly selected from TMA1-TMA3, the other from TOP1-TOP3.
Step 4. Conduct the comparison between the new individual and the original one. If the new individual outperforms the original one, update the current best solution.
Step 5. Set $\zeta \leftarrow \zeta + 1$, if $\zeta > \zeta_{\max}$, go to Step 6; otherwise, go to Step 2.
Step 6. Terminate the algorithm.

4.7. Steps of the NISA

Step 1. Initialize the parameters, i.e., the population size PS, the maximum iteration of NISA $t_{\max}$ and the maximum iteration of local search algorithm $\zeta_{\max}$.
Step 2. Create the initial population by using the approach in Section 4.2.
Step 3. Find out the current global best solution $\boldsymbol{X}_{gb}^{t}$.
Step 4. Calculate the value of parameter α, and divide the population into the composition group and the mirror group.
Step 5. Perform the local search algorithm on $\boldsymbol{X}_{gb}^{t}$.
Step 6. Perform the crossover-based composition optimization operator on the individuals in the composition group.
Step 7. Perform the neighborhood-crossover mirror search operator on the individuals in the mirror group.
Step 8. Evaluate each new individual. If it is superior to the original one, accept it; otherwise, keep the original solution unchanged.
Step 9. Determine whether the stop condition is met. If yes, go to Step 10; otherwise, go to Step 3.
Step 10. Terminate the NISA algorithm.

5. Numerical Experiments

Extensive experiments are conducted in this section to test the performance of the NISA. All algorithms are coded in Fortran language and run on VMware Workstation with 2GB RAM under Windows XP.

5.1. Test Instance

Two sets of test instances are examined in this section. The first set refers to 15 small-scale instances modified from benchmark instances of the traditional FJSP, and the second set is 24 large-scale instances randomly generated with a certain number of jobs and machines. That is, there are 39 instances of different scales considered in this section. For each instance, all compared algorithms are independently run 10 times to obtain the comparison results.

Small-scale instances: Fifteen benchmark instances (Kacem01-Kacem05, MK01-MK10) were proposed by Kacem et al. [55] and Brandimarte [56]. In those benchmark instances, the information on job split, energy consumption and transportation times are not involved. Therefore, we modified the original instances by setting some additional information in a certain range with a discrete uniform distribution. i.e., $s_i \in [1, 3]$, $PE_{ijkl} \in [10, 15]$, $SE_k \in [6, 10]$, $CE \in [12, 18]$, $TE \in [5, 10]$ and $TT_{i(j-1)lk,ijlw} \in [5, 15]$.

Large-scale instances: Twenty-four instances (RM01-RM24) are generated with the number of jobs $n \in \{50, 60, 70, 80, 90, 100\}$ and machines $m \in \{10, 15, 20, 25\}$. In addition, other data are set with a discrete uniform distribution, i.e., $s_i \in [1, 5]$, $J_i \in [1, 3]$, $nop \in [2, m]$, $p_{ijk} \in [10, 20]$, $PE_{ijkl} \in [10, 15]$, $SE_k \in [6, 10]$, $CE \in [12, 18]$, $TE \in [5, 10]$ and $TT_{i(j-1)lk,ijlw} \in [5, 10]$. nop represents the size of compatible machine set for each operation.

5.2. Parameter Tuning

There are three parameters to be tuned, i.e., population size PS, maximum iteration of NISA $t_{\max}$ and maximum iteration of local search algorithm $\zeta_{\max}$. Here, the design of the experiment is carried out to get the best combination of these parameters based on the instance RM12. Tables 1 and 2 show the factor levels and the orthogonal array $L_{16}(5^3)$. In Table 2, Avg denotes the average value gained from the ten runs of NISA. Table 3 gives the response value and the significance rank, which reflects that PS and $t_{\max}$ are much more significant than $\zeta_{\max}$. The trend of the factor level is illustrated in Figure 3. According to the computational results, the three parameters are fixed as follows: PS = 300, $t_{\max}$ = 1500, $\zeta_{\max}$ = 40.

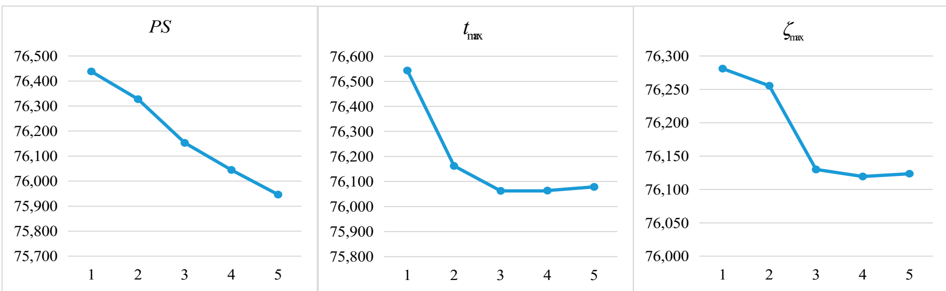


Figure 3. Factor level trend of parameters.

Table 1. Parameter levels.

Factor	Level				
	1	2	3	4	5
PS	100	150	200	250	300
$t_{\max}$	500	1000	1500	2000	2500
$\zeta_{\max}$	10	20	30	40	50

Table 2. Orthogonal array and *Avg* values.

Number	*PS*	$t_{\max}$	$\zeta_{\max}$	Avg
1	1	1	1	77,119.7
2	1	2	2	76,567.3
3	1	3	3	76,178.2
4	1	4	4	76,062.4
5	1	5	5	76,173.3
6	2	1	2	76,774.1
7	2	2	3	76,156.1
8	2	3	4	76,227.6
9	2	4	5	76,190.4
10	2	5	1	76,236.3
11	3	1	3	76,414.9
12	3	2	4	76,078.1
13	3	3	5	76,096.7
14	3	4	1	76,169.8
15	3	5	2	76,084.4
16	4	1	4	76,271.5
17	4	2	5	76,012.7
18	4	3	1	75,977.1
19	4	4	2	76,070.8
20	4	5	3	75,946.5
21	5	1	5	76,141.6
22	5	2	1	75,890.4
23	5	3	2	75,882.3
24	5	4	3	75,914.1
25	5	5	4	75,927.3

Table 3. Response value and significance rank.

Level	*PS*	$t_{\max}$	$\zeta_{\max}$
1	76,438.4	76,543.3	76,281.1
2	76,327.7	76,162.0	76,255.5
3	76,152.9	76,062.7	76,130.0
4	76,044.7	76,063.6	76,119.6
5	75,946.2	76,078.3	76,123.7
Delta	492.2	480.6	161.5
Rank	1	2	3

5.3. Comparison Results of Different Algorithms

5.3.1. Effectiveness of the Population Initialization Approach

To guarantee the quality of the initial population, a population initialization approach is adopted in Section 4.2. Here, the effectiveness of the approach is first validated through a comparison between NISA and ISARR. For the ISARR, it is an abbreviated algorithm of the proposed NISA, where the machine assignment and the operation permutation of each initial scheduling solution are both generated at random. Table 4 reports the comparison data of the two algorithms. '*Best*' is the best value collected by each algorithm. '*Avg*' is the average result of each algorithm in ten runs. '*Time*' is the average running time (in seconds). The data in bold denote the best value collected by all compared algorithms.

Table 4. Effectiveness analysis of the population initialization approach.

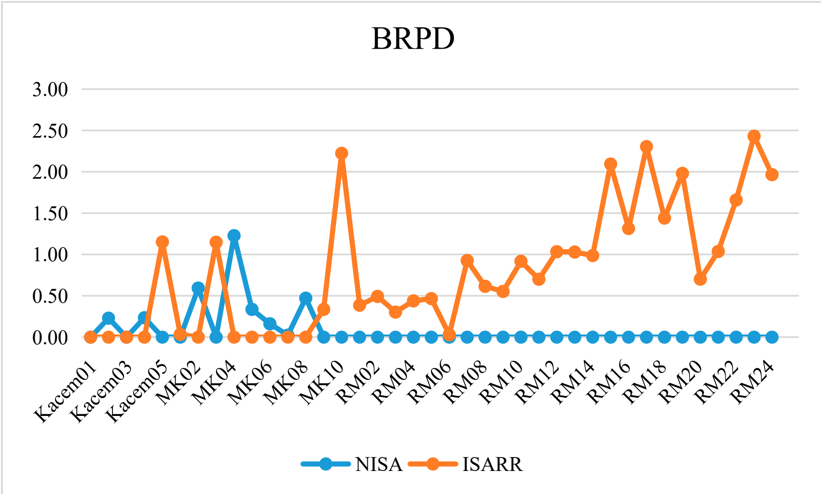
Instance	$m \times n$	NISA			ISARR		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*
Kacem01	5 × 4	**1762**	**1762.0**	36.5	**1762**	**1762.0**	37.2
Kacem02	8 × 8	3494	**3495.2**	77.2	**3486**	3500.8	78.2
Kacem03	7 × 10	**3117**	**3125.4**	82.9	**3117**	3126.9	83.4
Kacem04	10 × 10	3842	3883.0	85.6	**3833**	**3860.9**	87.8
Kacem05	10 × 15	**7029**	**7186.9**	158.8	7110	7233.1	162.9
MK01	6 × 10	**8971**	9050.5	149.2	8974	**9005.3**	149.5
MK02	6 × 10	10,163	10,256.2	154.8	**10,103**	**10,185.9**	156.6
MK03	8 × 15	**32,034**	**32,387.1**	411.8	32,402	32,653.1	424.2
MK04	8 × 15	15,496	15,710.9	247.9	**15,308**	**15,462.4**	260.9
MK05	4 × 15	27,182	27,272.5	311.5	**27,091**	**27,148.9**	296.9
MK06	15 × 10	36,672	37,152.9	417.2	**36,613**	**37,058.3**	430.9
MK07	5 × 20	20,965	21,019.4	276.8	**20,960**	**20,998.2**	288.1
MK08	10 × 20	88,152	89,319.2	659.8	**87,738**	**88,714.4**	662.8
MK09	10 × 20	**62,277**	**63,514.5**	680.2	62,486	63,606.0	705.8
MK10	15 × 20	**72,341**	**73,085.5**	708.3	73,950	75,542.6	717.3
RM01	10 × 50	**37,600**	**37,751.8**	410.3	37,745	37,926.7	411.1
RM02	10 × 60	**40,503**	**40,597.0**	495.9	40,703	40,882.7	501.6
RM03	10 × 70	**46,416**	**46,592.9**	624.0	46,557	46,799.9	625.4
RM04	10 × 80	**64,828**	**65,022.4**	716.4	65,113	65,623.4	719.9
RM05	10 × 90	**59,776**	**59,854.9**	884.1	60,055	60,313.3	871.4
RM06	10 × 100	**66,035**	**66,192.8**	1004.3	66,055	66,457.6	990.1
RM07	15 × 50	**37,031**	**37,240.9**	430.3	37,374	37,610.7	426.1
RM08	15 × 60	**37,546**	**37,769.8**	520.6	37,777	37,963.1	527.6
RM09	15 × 70	**48,581**	**48,751.7**	638.5	48,850	49,203.7	654.9
RM10	15 × 80	**57,888**	**58,022.8**	766.3	58,419	58,727.1	771.6
RM11	15 × 90	**57,058**	**57,238.5**	902.8	57,458	58,100.2	905.4
RM12	15 × 100	**75,616**	**75,872.3**	1028.8	76,398	76,816.3	1024.3
RM13	20 × 50	**36,252**	**36,489.7**	463.2	36,625	36,947.2	479.4
RM14	20 × 60	**42,683**	**42,876.9**	585.6	43,104	43,580.2	601.8
RM15	20 × 70	**44,171**	**44,464.3**	694.1	45,096	45,589.0	706.7
RM16	20 × 80	**59,297**	**59,508.9**	822.6	60,076	60,535.1	827.9
RM17	20 × 90	**61,150**	**61,321.0**	966.5	62,559	62,801.9	1016.3
RM18	20 × 100	**63,135**	**63,446.7**	1089.4	64,045	64,477.8	1061.9
RM19	25 × 50	**32,512**	**32,659.4**	491.2	33,156	33,360.9	473.9
RM20	25 × 60	**39,477**	**39,660.7**	625.5	39,754	40,167.6	595.6
RM21	25 × 70	**41,535**	**41,872.1**	799.9	41,966	42,400.4	700.5
RM22	25 × 80	**46,759**	**46,998.3**	929.6	47,535	48,328.4	826.2
RM23	25 × 90	**63,567**	**63,880.3**	1096.9	65,112	65,709.1	1024.3
RM24	25 × 100	**65,134**	**65,345.0**	1163.7	66,414	67,460.8	1119.6
Mean	-	**41,488.4**	**41,706.5**	579.7	42,888.5	42,247.2	574.5

It can be obviously observed that: (1) In comparisons of *Best* values, NISA performs better than ISARR in 31 out of 39 instances. In the small-scale instances, NISA can obtain 7 bold values out of 15 instances, which is less than those (10 bold values) obtained by ISARR. However, in the large-scale instances (RM01-RM24), NISA performs better than ISARR in all instances. (2) In comparisons of *Avg* values, NISA also performs better than ISARR in 31 out of 39 instances. In the small-scale instances, NISA yields 7 bold values, and ISARR obtains 9 bold values. However, in the large-scale instances (RM01-RM24), NISA is also superior to ISARR in all instances. (3) In comparison to *Time*, the difference between the two algorithms is very small. (3) The *Mean* value also reflects the superior performance of the NISA algorithm. In addition, to illustrate the comparison results more clearly, the curves of BRPD and ARPD are shown in Figure 4. BRPD and ARPD are two kinds of relative percentage deviation (RPD), which can be measured by Equations (22) and (23), respectively.

$$BRPD = 100 \times (A_i^* - A^*)/A^* \tag{22}$$

$$ARPD = 100 \times (\overline{A}_i - A^*)/A^* \quad (23)$$

where A_i^* is the *Best* value obtained by algorithm i; $\overline{A}_i$ is the *Avg* value acquired by algorithm i; and A^* is the best solution among all the compared algorithms. Following Equations (22) and (23), the values of BRPD and ARPD are, respectively, determined by *Best* and *Avg*. According to the results in Table 4 and Figure 4, it can be concluded that the population initialization approach is applicable for the considered problem.



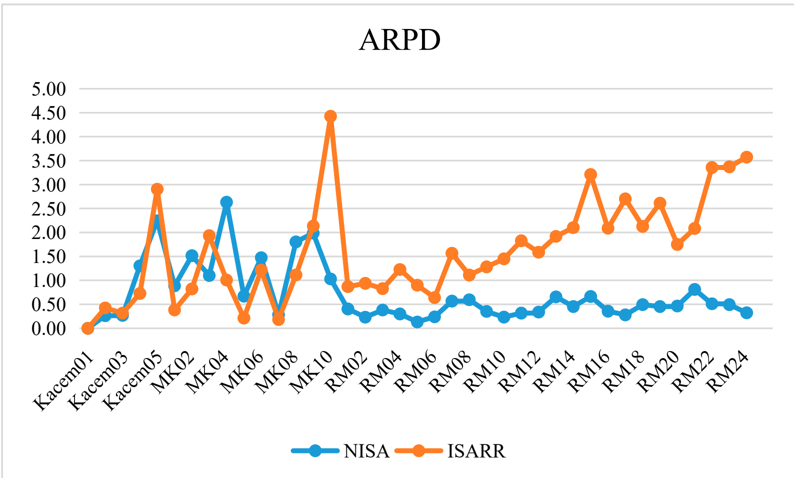


Figure 4. The curves of BRPD and ARPD in the comparison between NISA and ISARR.

5.3.2. Effectiveness of the Dynamic Adjustment on λ

In Section 4.4, a mirror is first acquired by dynamically changing the execution time of neighborhood operation in the iteration process. To validate the effectiveness of the dynamic adjustment strategy, five algorithms with different values of λ, namely ISA1-ISA5, are compared with the proposed NISA. The comparison results are reported in Table 5, where the data in bold represent the best values among all compared algorithms. It can be easily observed that: (1) For the *Best* value, NISA received 14 bold values out of 39 instances. In the small-scale instances, NISA yields only 4 bold values, but it can obtain 10 bold values in large-scale instances. The second-best algorithm, namely ISA1, can only achieve 12 boldface values. In the small-scale instances, ISA1 yields 7 bold values, but it can obtain 5 bold values in large-scale instances. (2) For the *Avg* value, NISA yields 16 boldface values out of 39 instances. In the small-scale instances, NISA obtains 5 bold values; meanwhile, it can get 11 bold values in large-scale instances. The second-best algorithm, namely ISA1, can only receive 10 boldface values. In the small-scale instances, ISA1 obtains only 3 bold values; meanwhile, it gets only 7 bold values in large-scale instances. (3) For the *Time* value, the differences between NISA and other compared algorithms are not particularly obvious. (4) The *Mean* value also demonstrates the superior performance of the NISA algorithm. In addition, the curves of BRPD and ARPD are shown in Figure 5. According to the results in Table 5 and Figure 5, the dynamic adjustment strategy on λ is effective for the considered problem.

Table 5. Effectiveness analysis of the dynamic adjustment on λ.

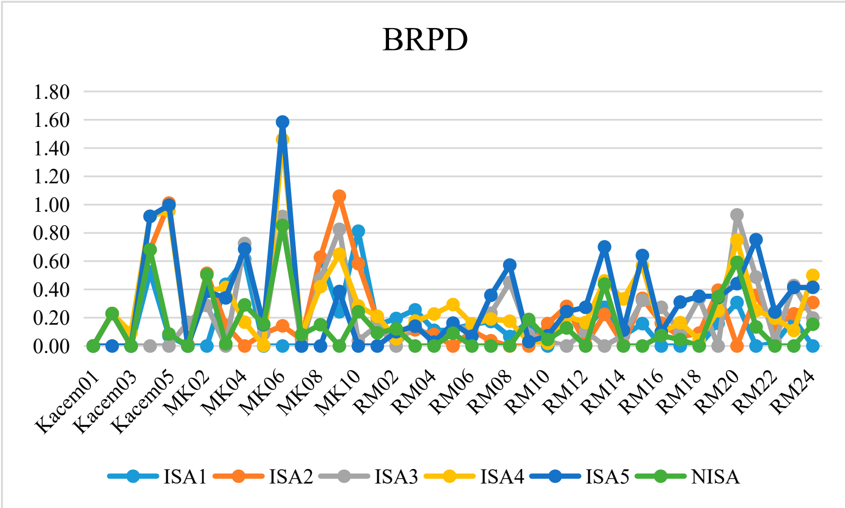
Instance	$m \times n$	NISA			ISA1 (λ)			ISA2 (λ)		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*	*Best*	*Avg*	*Time*
Kacem01	5×4	**1762**	**1762.0**	36.5	**1762**	1763.6	36.2	**1762**	1765.2	36.4
Kacem02	8×8	3494	3495.7	77.2	3494	3516.9	74.9	3494	3500.8	75.1
Kacem03	7×10	**3117**	**3125.4**	82.9	**3117**	3132.4	80.2	**3117**	3133.1	80.0
Kacem04	10×10	3842	3883.0	85.6	3836	3881.2	84.3	3842	3884.7	84.4
Kacem05	10×15	7029	**7113.9**	158.8	7028	7128.4	156.7	7094	7151.3	157.7
MK01	6×10	**8971**	9025.8	149.2	**8971**	9030.0	144.6	8972	9021.4	145.2
MK02	6×10	10,163	10,239.4	154.8	**10,112**	**10,219.7**	152.3	10,164	10,239.9	152.5
MK03	8×15	32,034	**32,387.1**	411.8	32,169	32,403.7	400.0	32,076	32,462.6	403.8
MK04	8×15	15,496	15,640.8	247.9	15,546	15,796.9	245.6	**15,451**	**15,618.9**	244.8
MK05	4×15	27,182	27,272.5	311.5	**27,140**	27,255.1	278.2	27,164	27,251.0	279.6
MK06	15×10	36,672	37,003.7	417.2	**36,362**	**36,743.1**	407.4	36,414	36,780.1	409.2
MK07	5×20	20,965	21,019.4	276.8	**20,948**	**20,993.0**	272.3	20,958	21,026.2	273.1
MK08	10×20	88,152	89,219.2	659.8	88,570	89,204.6	640.9	88,572	89,173.4	635.6
MK09	10×20	**62,277**	63,514.5	680.2	62,428	63,594.6	664.2	62,937	63,545.5	685.2
MK10	15×20	72,341	**73,005.0**	708.3	72,752	73,265.1	697.3	72,586	73,386.8	709.6
RM01	10×50	37,600	37,736.8	410.3	37,621	37,712.8	419.3	37,635	37,755.9	422.1
RM02	10×60	40,503	**40,556.4**	495.9	40,533	40,598.3	558.3	40,477	40,575.1	552.7
RM03	10×70	**46,416**	**46,538.8**	624.0	46,535	46,684.5	632.1	46,469	46,648.2	643.8
RM04	10×80	**64,828**	**64,957.7**	716.4	64,900	65,078.8	783.7	64,881	65,035.1	764.0
RM05	10×90	59,776	**59,854.9**	884.1	59,794	60,028.1	949.6	**59,722**	59,982.6	912.8
RM06	10×100	**66,035**	**66,192.8**	1004.3	66,136	66,300.1	1032.8	66,106	66,220.2	1030.8
RM07	15×50	**37,031**	**37,188.1**	430.3	37,095	37,268.4	434.8	37,045	37,209.8	440.7
RM08	15×60	37,546	37,683.6	520.6	37,570	**37,675.9**	549.1	**37,545**	37,692.4	562.3
RM09	15×70	48,581	48,704.5	638.5	48,575	48,789.0	661.3	**48,490**	48,700.0	672.6
RM10	15×80	57,888	**58,002.9**	766.3	**57,861**	58,079.7	864.4	57,953	58,044.2	797.9
RM11	15×90	57,058	**57,215.1**	902.8	57,118	57,371.6	944.1	57,145	57,272.1	946.1
RM12	15×100	**75,616**	**75,826.8**	1028.8	75,705	75,984.0	1077.6	75,624	75,829.2	1084.9
RM13	20×50	36,252	36,429.6	463.2	36,193	36,339.1	455.8	36,175	**36,330.1**	465.0
RM14	20×60	**42,683**	42,876.9	585.6	42,714	42,886.9	614.5	42,687	42,866.5	585.5
RM15	20×70	**44,171**	**44,464.3**	694.1	44,241	44,465.7	693.2	44,320	44,502.4	704.1
RM16	20×80	59,297	**59,431.9**	822.6	**59,253**	59,577.9	829.7	59,349	59,541.3	834.0
RM17	20×90	61,150	61,321.0	966.5	**61,122**	61,298.8	1075.7	61,161	**61,227.9**	1027.3
RM18	20×100	**63,135**	63,446.7	1089.4	63,137	**63,372.9**	1103.8	63,193	63,469.4	1198.4
RM19	25×50	32,512	32,659.4	491.2	32,451	32,603.6	529.8	32,527	32,613.1	525.2
RM20	25×60	39,477	39,660.7	625.5	39,366	**39,531.5**	653.3	**39,245**	39,576.1	656.8
RM21	25×70	41,535	41,872.1	799.9	**41,479**	**41,673.1**	778.1	41,628	41,779.6	785.5
RM22	25×80	**46,759**	46,998.3	929.6	46,768	**46,900.2**	915.5	46,823	46,945.9	937.6
RM23	25×90	**63,567**	63,880.3	1096.9	63,688	**63,869.5**	1074.9	63,712	63,930.9	1095.6
RM24	25×100	65,134	65,345.0	1163.7	**65,034**	**65,267.4**	1237.3	65,234	65,344.6	1239.3
Mean	-	**41,488.4**	**41,706.5**	**579.7**	41,516.0	41,725.3	595.0	41,532.0	41,720.1	596.3

Table 5. *Cont.*

Instance	$m \times n$	ISA3 (λ = 3)			ISA4 (λ = 4)			ISA5 (λ = 5)		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*	*Best*	*Avg*	*Time*
Kacem01	5 × 4	**1762**	1765.2	36.4	**1762**	1768.4	36.8	**1762**	**1762.0**	37.3
Kacem02	8 × 8	3494	3498.0	76.1	3494	**3495.2**	76.0	**3486**	3497.0	77.1
Kacem03	7 × 10	**3117**	3129.0	81.1	3120	3136.6	81.3	**3117**	3143.3	81.5
Kacem04	10 × 10	**3816**	**3870.3**	84.7	3851	3884.5	85.4	3851	3883.9	85.4
Kacem05	10 × 15	**7023**	7160.9	157.8	7090	7156.5	158.3	7093	7188.7	159.6
MK01	6 × 10	8986	9035.3	147.8	**8971**	9034.4	147.2	**8971**	**9021.0**	146.8
MK02	6 × 10	10,141	10,233.2	153.2	10,151	10,228.2	152.8	10,151	10,245.0	153.1
MK03	8 × 15	**32,029**	32,428.4	403.5	32,162	32,498.7	405.9	32,138	32,431.9	404.6
MK04	8 × 15	15,563	15,687.8	244.6	15,477	15,639.2	244.7	15,557	15,689.3	245.8
MK05	4 × 15	27,164	**27,245.8**	280.3	27,141	27,260.0	281.3	27,181	27,268.6	280.1
MK06	15 × 10	36,695	37,168.5	409.7	36,893	37,272.9	411.5	36,938	37,330.4	413.8
MK07	5 × 20	20,951	21,006.8	273.8	20,965	21,004.3	274.5	20,948	21,007.8	273.7
MK08	10 × 20	88,437	89,212.9	631.0	88,388	**88,919.3**	628.7	**88,019**	89,016.8	632.1
MK09	10 × 20	62,791	63,645.0	682.3	62,681	63,572.2	684.5	62,517	**63,475.3**	684.4
MK10	15 × 20	72,194	73,090.7	699.6	72,371	73,111.0	701.1	**72,166**	73,120.5	721.4
RM01	10 × 50	37,620	37,708.5	436.7	37,644	37,714.6	428.8	**37,565**	**37,702.7**	425.5
RM02	10 × 60	**40,454**	40,589.6	540.1	40,475	40,569.9	538.9	40,495	40,563.2	549.5
RM03	10 × 70	46,491	46,610.7	647.8	46,498	46,651.1	647.2	46,481	46,676.1	651.5
RM04	10 × 80	64,835	65,008.8	775.3	64,975	65,109.8	766.2	64,846	65,054.0	774.9
RM05	10 × 90	59,793	59,997.8	901.8	59,897	60,033.6	903.9	59,819	60,033.3	910.3
RM06	10 × 100	66,123	66,281.2	1031.4	66,138	66,270.5	1033.2	66,081	66,280.0	1025.1
RM07	15 × 50	37,117	37,263.7	445.3	37,102	37,267.3	450.3	37,164	37,303.9	458.3
RM08	15 × 60	37,714	37,822.2	562.7	37,611	37,759.2	572.8	37,760	37,860.1	566.6
RM09	15 × 70	48,529	**48,686.8**	672.0	48,504	48,737.4	679.3	48,505	48,733.9	680.1
RM10	15 × 80	57,880	58,025.8	799.2	57,872	58,008.6	827.3	57,903	58,080.9	797.7
RM11	15 × 90	**56,985**	57,247.1	947.4	57,105	57,241.9	957.4	57,124	57,252.8	948.5
RM12	15 × 100	75,696	75,971.2	1085.3	75,742	75,939.0	1092.4	75,823	75,987.5	1081.4
RM13	20 × 50	**36,094**	36,440.0	463.8	36,260	36,426.8	465.9	36,347	36,523.7	466.7
RM14	20 × 60	42,716	**42,860.7**	600.7	42,825	42,970.1	589.2	42,731	42,884.4	601.2
RM15	20 × 70	44,314	44,577.2	711.2	44,422	44,572.5	707.7	44,454	44,550.8	711.6
RM16	20 × 80	59,415	59,665.5	845.6	59,311	59,495.6	852.6	59,312	59,623.3	882.6
RM17	20 × 90	61,181	61,414.7	996.0	61,223	61,418.2	991.7	61,312	61,543.3	994.5
RM18	20 × 100	63,348	63,512.9	1180.2	63,162	63,506.3	1122.4	63,357	63,675.4	1176.6
RM19	25 × 50	32,399	32,579.8	517.6	32,480	32,613.6	465.1	32,510	32,647.2	442.7
RM20	25 × 60	39,609	39,667.4	653.3	39,539	39,649.8	650.2	39,418	39,630.4	620.4
RM21	25 × 70	41,681	41,838.0	782.9	41,583	41,926.5	777.4	41,791	41,993.2	694.7
RM22	25 × 80	46,774	46,969.0	925.9	46,851	47,033.6	917.6	46,871	47,041.6	804.2
RM23	25 × 90	63,839	63,976.6	1121.8	63,637	64,105.1	1099.3	63,830	64,129.6	954.4
RM24	25 × 100	65,163	65,451.7	1243.1	65,359	65,510.2	1226.8	65,304	65,548.5	1077.2
Mean	-	41,536.7	41,752.4	596.1	41,557.2	41,756.7	593.2	41,556.4	41,779.5	581.9

5.3.3. Effectiveness of the Dynamic Adjustment on α

To cooperate the abilities of exploration and exploitation, a linear adjustment approach of α is employed to divide individuals into two groups. In this subsection, we validate the effectiveness of the dynamic adjustment approach based on the comparison between NISA and ISAF. In the ISAF, α is set as the recommended value in [44], i.e., α = 0.2. It can be clearly observed from Table 6 that: (1) In comparison to the *Best* values, NISA performs better than ISAF in 34 out of 39 instances. (2) In comparison to the *Avg* values, NISA is superior to ISAF in 33 out of 39 instances. (3) In comparison to the *Time* value, the running time of NISA is less than that of ISAF in 19 out of 39 instances. (4) The *Mean* value also demonstrates the superior performance of the NISA algorithm. In addition, the curves of BRPD and ARPD are shown in Figure 6. According to the results in Table 6 and Figure 6, it can be concluded that the dynamic adjustment strategy on α is also effective for the considered problem.



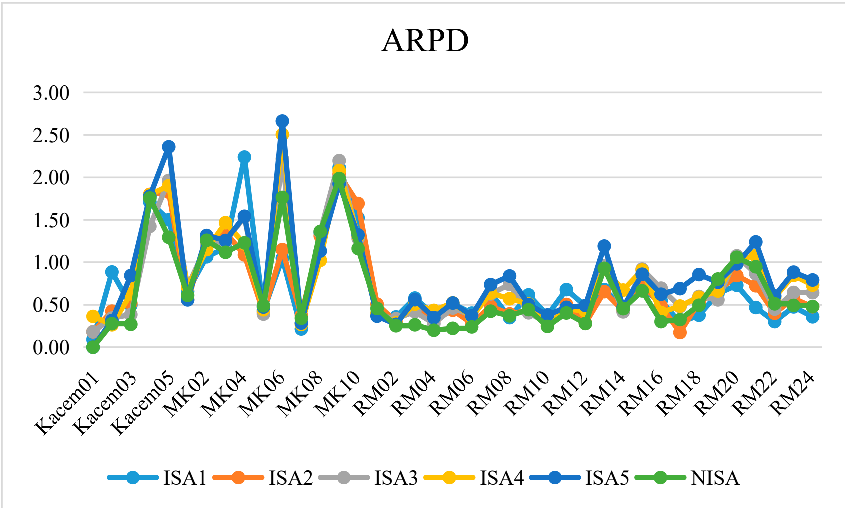


Figure 5. The curves of BRPD and ARPD in the comparison between NISA and ISA1-5.

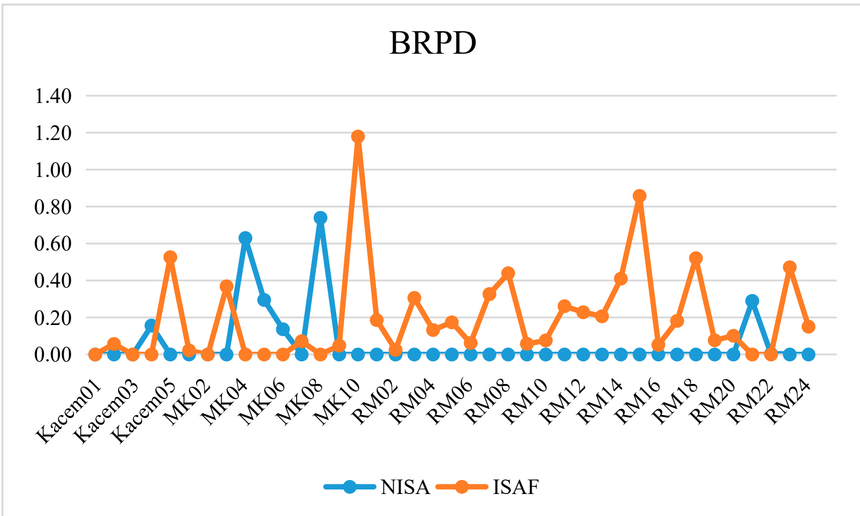


Figure 6. *Cont.*

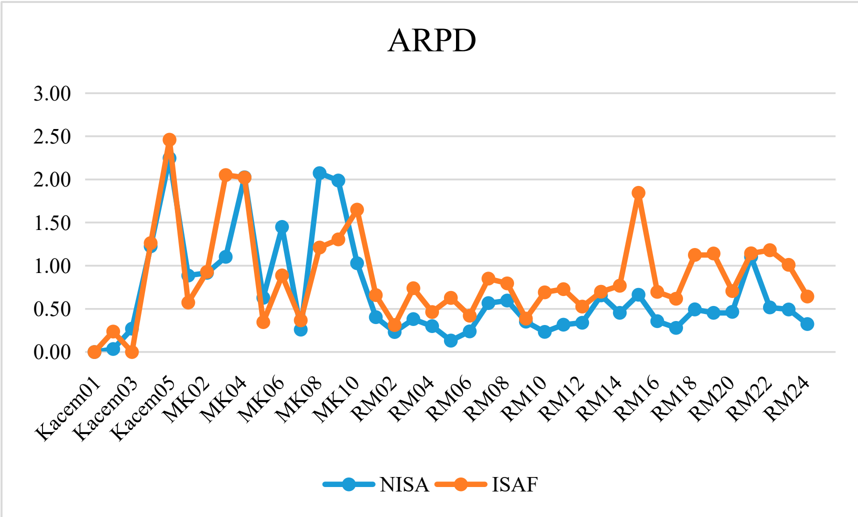


Figure 6. The curves of BRPD and ARPD in the comparison between NISA and ISAF.

Table 6. Effectiveness analysis of the dynamic adjustment on α.

Instance	$m \times n$	**NISA**			**ISAF**		
		Best	***Avg***	***Time***	***Best***	***Avg***	***Time***
Kacem01	5 × 4	**1762**	**1762.0**	36.5	**1762**	**1762.0**	36.1
Kacem02	8 × 8	**3494**	**3495.2**	77.2	3496	3502.3	75.8
Kacem03	7 × 10	**3117**	**3125.4**	82.9	3135	3142.8	79.8
Kacem04	10 × 10	3842	**3883.0**	85.6	**3836**	3884.4	85.3
Kacem05	10 × 15	**7029**	**7186.9**	158.8	7066	7201.9	159.6
MK01	6 × 10	**8971**	9050.5	149.2	8973	9022.4	145.3
MK02	6 × 10	**10,163**	**10,256.2**	154.8	**10,163**	10,257.4	152.7
MK03	8 × 15	**32,034**	**32,387.1**	411.8	32,152	32,691.1	406.1
MK04	8 × 15	15,496	15,710.9	247.9	**15,399**	**15,710.3**	245.8
MK05	4 × 15	27,182	27,272.5	311.5	**27,102**	**27,195.7**	306.7
MK06	15 × 10	36,672	37,152.9	417.2	**36,622**	**36,946.9**	414.5
MK07	5 × 20	**20,965**	**21,019.4**	276.8	20,980	21,042.0	271.5
MK08	10 × 20	88,152	89,319.2	659.8	**87,505**	**88,564.9**	652.7
MK09	10 × 20	**62,277**	63,514.5	680.2	62,307	**63,089.3**	670.9
MK10	15 × 20	**72,341**	**73,085.5**	708.3	73,194	73,534.5	699.6
RM01	10 × 50	**37,600**	**37,751.8**	410.3	37,670	37,847.8	451.1
RM02	10 × 60	**40,503**	**40,597.0**	495.9	40,513	40,629.4	566.3
RM03	10 × 70	**46,416**	**46,592.9**	624.0	46,558	46,759.2	636.6
RM04	10 × 80	**64,828**	**65,022.4**	716.4	64,914	65,128.4	769.7
RM05	10 × 90	**59,776**	**59,854.9**	884.1	59,880	60,150.9	974.2
RM06	10 × 100	**66,035**	**66,192.8**	1004.3	66,076	66,312.8	1232.9
RM07	15 × 50	**37,031**	**37,240.9**	430.3	37,152	37,345.9	436.8
RM08	15 × 60	**37,546**	**37,769.8**	520.6	37,711	37,844.8	545.6
RM09	15 × 70	**48,581**	**48,751.7**	638.5	48,609	48,768.8	658.9
RM10	15 × 80	**57,888**	**58,022.8**	766.3	57,932	58,288.3	781.8
RM11	15 × 90	**57,058**	**57,238.5**	902.8	57,207	57,473.6	964.4
RM12	15 × 100	**75,616**	**75,872.3**	1028.8	75,789	76,013.7	1096.6
RM13	20 × 50	**36,252**	**36,489.7**	463.2	36,327	36,505.0	491.0
RM14	20 × 60	**42,683**	**42,876.9**	585.6	42,858	43,010.9	615.8
RM15	20 × 70	**44,171**	**44,464.3**	694.1	44,550	44,985.8	734.6
RM16	20 × 80	**59,297**	**59,508.9**	822.6	59,328	59,709.7	890.7
RM17	20 × 90	**61,150**	**61,321.0**	966.5	61,261	61,527.8	1045.3
RM18	20 × 100	**63,135**	**63,446.7**	1089.4	63,464	63,845.1	1192.8
RM19	25 × 50	**32,512**	**32,659.4**	491.2	32,537	32,883.0	504.2
RM20	25 × 60	**39,477**	**39,660.7**	625.5	39,517	39,755.6	615.6
RM21	25 × 70	41,535	**41,872.1**	799.9	**41,415**	41,888.7	739.3
RM22	25 × 80	46,759	**46,998.3**	929.6	**46,757**	47,308.7	875.6
RM23	25 × 90	**63,567**	**63,880.3**	1096.9	63,867	64,208.9	1072.3
RM24	25 × 100	**65,134**	**65,345.0**	1163.7	65,232	6,5553.1	1146.6
Mean	-	**41,488.4**	**41,706.5**	**579.7**	41,559.4	41,828.0	601.1

5.3.4. Comparison with Existing Algorithms

To further demonstrate the advantage of the proposed NISA algorithm, we compared it with three published algorithms, i.e., genetic algorithm (GA) [41], modified animal migration optimization (MAMO) [38] and hybrid particle swarm optimization and genetic algorithm (PSO-GA) [57]. The GA was proposed for the FJSP with overlapping operations, but job transportation times and energy consumption are neglected. The MAMO was proposed for the energy-saving FJSP considering transportation time and deterioration effect simultaneously, but the overlapping in operations is not considered. The PSO-GA was presented for energy-saving FJSP with assembly operations, but job transportation times and overlapping in operations are not involved. The three compared algorithms are easily implemented for the considered problem. For the GA, to enhance the search capacity, the proposed population initialization approach and local search algorithm are shared with the NISA. For the PSO-GA, the population initialization approach and crossover operator are also the same as the NISA. The neighborhood structures are randomly selected as the mutation operator. The parameters of the two algorithms are set as follows: In the GA, the population size PS is 300, the maximum iteration $t_{\max}$ is 1500, the maximum iteration of local search algorithm $\zeta_{\max}$ is 40, the crossover rate is 0.8 and the mutation rate is 0.2. In the MAMO, the population size PS is 300, the maximum iteration $t_{\max}$ is 1500 and the crossover rate is 0.6. In the PSO-GA, the population size PS is 300, the maximum iteration $t_{\max}$ is 1500, the crossover rate is 0.8 and the mutation rate is 0.2.

According to the comparison results in Table 7, the following observations can be obtained: (1) In comparison to the *Best* values, NISA outperforms the other three algorithms in 38 out of 39 instances. (2) In comparison to the *Avg* values, NISA performs best in 38 out of 39 instances. (3) In comparison to the *Time* value, GA performs best among the four algorithms. (4) The last row suggests that the proposed algorithm can obtain better computational results, but it consumes more time than GA and MAMO. The curves of BRPD and ARPD are shown in Figure 7. According to the results in Table 7 and Figure 7, it can be concluded that the NISA algorithm is effective in solving the considered problem.

Table 7. Comparison results of NISA and existing algorithms.

Instance	$m \times n$	NISA			GA		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*
Kacem01	5 × 4	**1762**	**1762.0**	36.5	**1762**	1776.0	3.8
Kacem02	8 × 8	3494	**3495.2**	77.2	3518	3540.9	9.3
Kacem03	7 × 10	**3117**	**3125.4**	82.9	3147	3198.5	10.1
Kacem04	10 × 10	**3842**	**3883.0**	85.6	3903	3932.1	11.1
Kacem05	10 × 15	**7029**	**7186.9**	158.8	7223	7398.4	22.1
MK01	6 × 10	**8971**	**9050.5**	149.2	9050	9155.4	18.7
MK02	6 × 10	**10,163**	**10,256.2**	154.8	10,263	10,387.8	19.9
MK03	8 × 15	**32,034**	**32,387.1**	411.8	32,632	33,205.2	53.6
MK04	8 × 15	**15,496**	**15,710.9**	247.9	15,755	15,963.0	32.9
MK05	4 × 15	**27,182**	**27,272.5**	311.5	27,308	27,414.3	35.7
MK06	15 × 10	**36,672**	**37,152.9**	417.2	37,194	37,716.1	57.2
MK07	5 × 20	**20,965**	**21,019.4**	276.8	21,072	21,272.0	36.9
MK08	10 × 20	**88,152**	**89,319.2**	659.8	88,859	89,563.2	94.4
MK09	10 × 20	**62,277**	63,514.5	680.2	62,708	**63,234.3**	95.8
MK10	15 × 20	**72,341**	**73,085.5**	708.3	72,859	73,967.9	108.2
RM01	10 × 50	**37,600**	**37,736.8**	410.3	37,673	38,106.7	78.5
RM02	10 × 60	**40,503**	**40,556.4**	495.9	40,614	41,005.9	102.8
RM03	10 × 70	**46,416**	**46,538.8**	624.0	46,985	47,254.8	129.5
RM04	10 × 80	**64,828**	**64,957.7**	716.4	65,653	66,146.7	159.3
RM05	10 × 90	**59,776**	**59,854.9**	884.1	60,434	60,666.0	190.8
RM06	10 × 100	**66,035**	**66,192.8**	1004.3	66,728	67,099.3	219.3
RM07	15 × 50	**37,031**	**37,188.1**	430.3	37,578	37,790.0	86.4
RM08	15 × 60	**37,546**	**37,683.6**	520.6	37,983	38,445.8	112.5
RM09	15 × 70	**48,581**	**48,704.5**	638.5	49,029	49,447.0	140.1
RM10	15 × 80	**57,888**	**58,002.9**	766.3	58,955	59,345.8	170.1
RM11	15 × 90	**57,058**	**57,215.1**	902.8	57,988	58,378.4	205.1
RM12	15 × 100	**75,616**	**75,826.8**	1028.8	77,023	77,441.6	237.3

Table 7. *Cont.*

Instance	$m \times n$	NISA			GA		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*
RM13	20 × 50	**36,252**	**36,429.6**	463.2	36,940	37,196.8	91.4
RM14	20 × 60	**42,683**	**42,876.9**	585.6	43,390	43,672.5	121.1
RM15	20 × 70	**44,171**	**44,464.3**	694.1	45,181	45,573.7	149.0
RM16	20 × 80	**59,297**	**59,431.9**	822.6	60,067	60,611.2	178.5
RM17	20 × 90	**61,150**	**61,321.0**	966.5	62,349	62,768.9	216.3
RM18	20 × 100	**63,135**	**63,446.7**	1089.4	64,265	64,743.9	251.1
RM19	25 × 50	**32,512**	**32,659.4**	491.2	32,890	33,317.2	97.0
RM20	25 × 60	**39,477**	**39,660.7**	625.5	40,164	40,445.1	134.9
RM21	25 × 70	**41,535**	**41,872.1**	799.9	42,327	42,633.0	155.1
RM22	25 × 80	**46,759**	**46,998.3**	929.6	47,722	48,117.1	187.6
RM23	25 × 90	**63,567**	**63,880.3**	1096.9	6,4887	65,330.0	224.5
RM24	25 × 100	**65,134**	**65,345.0**	1163.7	66,596	66,917.2	266.1
Mean	-	**41,488.4**	**41,706.5**	579.7	42,068.6	42,414.9	**115.7**

Instance	$m \times n$	MAMO			PSO-GA		
		Best	*Avg*	*Time*	*Best*	*Avg*	*Time*
Kacem01	5 × 4	**1762**	1780.8	17.9	**1762**	1811.0	57.9
Kacem02	8 × 8	3500	3601.2	41.7	**3350**	3532.2	122.4
Kacem03	7 × 10	3218	3313.8	44.6	3155	3201.9	131.8
Kacem04	10 × 10	3963	4048.5	47.8	3885	3984.5	138.9
Kacem05	10 × 15	7365	7550.7	99.3	7267	7413.7	252.3
MK01	6 × 10	9047	9145.7	78.9	9044	9162.7	236.6
MK02	6 × 10	10,290	10,500.8	82.6	10,254	10,484.6	249.0
MK03	8 × 15	33,121	33,610.7	245.8	32,671	33,889.2	660.0
MK04	8 × 15	15,693	15,926.0	148.6	15,827	16,166.5	387.4
MK05	4 × 15	27,309	27,437.8	163.4	27,260	27,494.6	442.9
MK06	15 × 10	37,652	38,402.6	263.4	37,206	37,865.1	656.2
MK07	5 × 20	21,228	21,412.9	170.3	21,344	21,512.9	434.5
MK08	10 × 20	90,105	90,949.1	428.1	89,788	90,887.7	1003.2
MK09	10 × 20	63,985	64,657.1	457.8	63,536	64,780.1	1063.7
MK10	15 × 20	73,476	75,002.8	502.5	74,530	75,979.6	1096.0
RM01	10 × 50	37,852	38,049.5	375.0	38,035	38,285.9	611.2
RM02	10 × 60	40,823	41,303.7	474.1	41,064	41,313.0	807.5
RM03	10 × 70	46,894	47,228.1	631.3	47,260	47,713.3	904.6
RM04	10 × 80	65,530	66,025.1	720.3	65,903	66,683.6	1081.3
RM05	10 × 90	60,341	60,614.0	878.9	60,704	61,144.0	1265.8
RM06	10 × 100	66,443	66,836.0	1076.1	67,538	68,230.5	1419.5
RM07	15 × 50	37,593	38,237.9	387.6	37,955	38,250.0	650.0
RM08	15 × 60	38,092	38,363.5	519.2	38,629	39,101.6	806.6
RM09	15 × 70	49,418	49,821.4	630.9	49,759	50,151.9	984.4
RM10	15 × 80	58,925	59,419	777.9	59,849	60,194.9	1204.7
RM11	15 × 90	58,131	58,594.9	955.3	59,675	60,094.2	1324.5
RM12	15 × 100	77,121	77,765.6	1101.1	78,758	79,438.1	1462.0
RM13	20 × 50	36,964	37,375.6	428.8	37,415	37,773.0	688.1
RM14	20 × 60	43,547	44,136.5	556.8	44,248	44,616.4	863.3
RM15	20 × 70	45,552	45,896.6	697.2	46,502	46,796.9	1062.8
RM16	20 × 80	60,368	60,868.1	883.8	61,575	62,043.1	1238.9
RM17	20 × 90	62,458	62,977.2	1015.5	64,168	64,685.2	1462.6
RM18	20 × 100	64,406	64,908.4	1232.4	65,930	66,685.2	1664.5
RM19	25 × 50	33,364	33,759.7	495.6	33,634	34,102.0	748.7
RM20	25 × 60	40,191	40,478.9	653.2	40,852	41,142.5	972.2
RM21	25 × 70	42,498	42,762.6	770.5	43,070	43,559.5	1133.5
RM22	25 × 80	48,047	48,269.7	977.6	49,251	49,864.2	1360.3
RM23	25 × 90	65,025	65,454.0	1136.4	66,768	68,007.8	1563.9
RM24	25 × 100	66,248	67,187.4	1310.1	68,721	69,347.5	1687.9
Mean	-	42,244.7	42,658.3	550.9	42,772.9	43,266.4	869.3

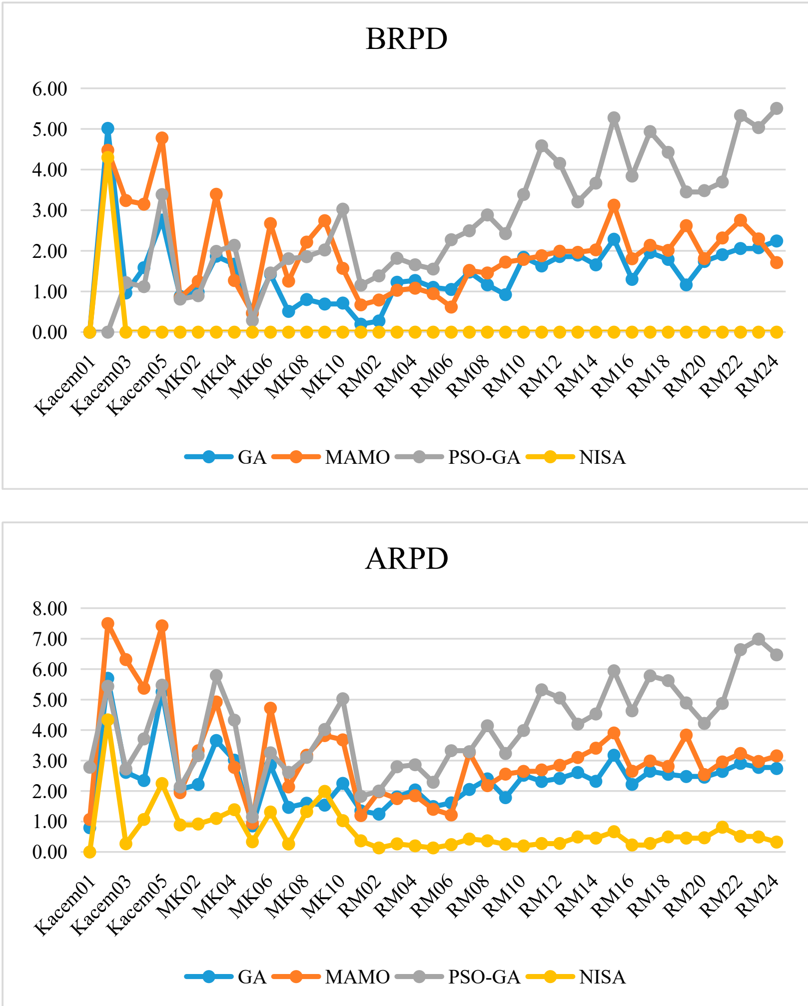


Figure 7. The curves of BRPD and ARPD in the comparison between NISA and the published algorithms.

5.3.5. Analysis of the Effect of Transportation Times

In this subsection, two scenarios with different levels of transportation times are used to investigate the effect of transportation times. For two scenarios, transportation times are randomly generated with two discrete uniform distributions U [1,5] and U [5,10], respectively. Table 8 reports that TTEC increases along with the increase in transportation times. In addition, four instances (RM01, RM06, RM09, RM16) are taken as examples, and the histograms of energy consumption are shown in Figure 8. It can be easily seen that the increase in TTEC is mainly attributed to the increase in TEC. It can also be inferred that the increase in transport times does not have a great impact on the other three kinds of energy consumption.

5.3.6. Analysis of the Effect of Sublot Number

In this subsection, two scenarios with different levels of sublot number are employed to analyze the effect of the sublot number. In the two scenarios, sublot numbers are randomly generated with two discrete uniform distributions U [1,5] and U [5,10], respectively. Table 9 indicates that TTEC increases along with the increase in the sublot numbers. In addition, for four instances (RM01, RM06, RM09, RM16), the histograms of energy consumption are shown in Figure 9. It can be easily observed that the increase in TEC is largely responsible for the increase in TTEC. It can also be inferred that the increase in sublot number has a relatively small effect on the other three types of energy consumption.

Table 8. Comparison results for two scenarios with different transportation times.

Instance	$m \times n$	Scenario 1		Scenario 2	
		Best	*Avg*	*Best*	*Avg*
RM01	10 × 50	27,695	27,953.2	37,600	37,736.8
RM02	10 × 60	32,382	32,457.8	40,503	40,556.4
RM03	10 × 70	36,686	36,773.1	46,416	46,538.8
RM04	10 × 80	47,185	47,347.3	64,828	64,957.7
RM05	10 × 90	47,286	47,391.0	59,776	59,854.9
RM06	10 × 100	52,287	52,508.5	66,035	66,192.8
RM07	15 × 50	26,438	26,520.4	37,031	37,188.1
RM08	15 × 60	29,410	29,509.5	37,546	37,683.6
RM09	15 × 70	35,964	36,112.9	48,581	48,704.5
RM10	15 × 80	41,916	42,100.6	57,888	58,002.9
RM11	15 × 90	44,756	44,921.0	57,058	57,215.1
RM12	15 × 100	53,575	53,771.0	75,616	75,826.8
RM13	20 × 50	25,539	25,694.2	36,252	36,429.6
RM14	20 × 60	30,629	30,837.8	42,683	42,876.9
RM15	20 × 70	33,501	33,598.2	44,171	44,464.3
RM16	20 × 80	41,555	41,686.2	59,297	59,431.9
RM17	20 × 90	44,758	44,927.0	61,150	61,321.0
RM18	20 × 100	47,687	47,937.8	63,135	63,446.7
RM19	25 × 50	23,912	23,988.0	32,512	32,659.4
RM20	25 × 60	28,630	28,801.0	39,477	39,660.7
RM21	25 × 70	31,910	32,321.8	41,535	41,872.1
RM22	25 × 80	35,984	39,191.4	46,759	46,998.3
RM23	25 × 90	45,566	45,812.0	63,567	63,880.3
RM24	25 × 100	47,788	47,928.7	65,134	65,345.0
Mean	-	38,043.3	38,337.1	51,022.9	51,201.9

Figure 8. The histograms in the two scenarios of transportation time.

Table 9. Comparison results for two scenarios with different sublot number.

Instance	$m \times n$	Scenario 1		Scenario 2	
		Best	*Avg*	*Best*	*Avg*
RM01	10 × 50	37,600	37,736.8	55,665	55,958.1
RM02	10 × 60	40,503	40,556.4	56,095	56,311.2
RM03	10 × 70	46,416	46,538.8	65,348	65,560.6
RM04	10 × 80	64,828	64,957.7	98,949	99,153.4
RM05	10 × 90	59,776	59,854.9	83,832	84,085.0
RM06	10 × 100	66,035	66,192.8	92,725	93,068.6
RM07	15 × 50	37,031	37,188.1	57,140	57,261.3
RM08	15 × 60	37,546	37,683.6	53,053	53,217.7
RM09	15 × 70	48,581	48,704.5	72,076	72,430.1
RM10	15 × 80	57,888	58,002.9	87,837	88,217.8
RM11	15 × 90	57,058	57,215.1	80,785	80,973.4
RM12	15 × 100	75,616	75,826.8	117,685	117,919.0
RM13	20 × 50	36,252	36,429.6	55,647	55,941.3
RM14	20 × 60	42,683	42,876.9	64,505	64,857.6
RM15	20 × 70	44,171	44,464.3	64,637	65,035.9
RM16	20 × 80	59,297	59,431.9	92,765	93,353.1
RM17	20 × 90	61,150	61,321.0	92,102	92,250.8
RM18	20 × 100	63,135	63,446.7	92,755	93,318.0
RM19	25 × 50	32,512	32,659.4	48,313	48,535.1
RM20	25 × 60	39,477	39,660.7	58,820	59,222.6
RM21	25 × 70	41,535	41,872.1	58,833	59,187.7
RM22	25 × 80	46,759	46,998.3	67,400	67,575.7
RM23	25 × 90	63,567	63,880.3	97,668	98,202.7
RM24	25 × 100	65,134	65,,345.0	99,061	99,281.6
Mean	-	51,022.9	51,201.9	75,570.7	75,871.6

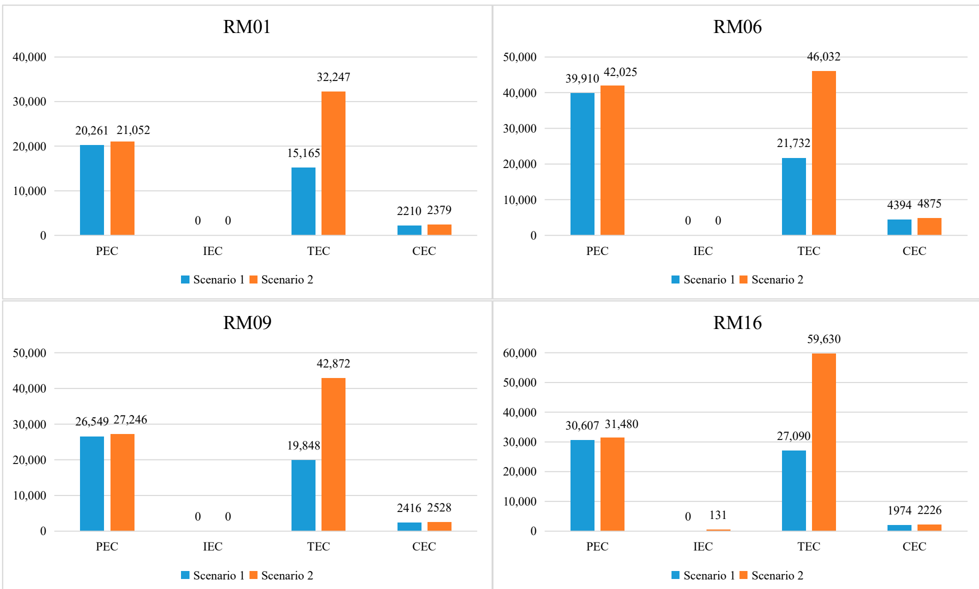


Figure 9. The histograms in the two scenarios of sublot number.

6. Conclusions and Future Work

In this paper, an ESFJSP is considered with overlapping operations and transportation times simultaneously. First, a mathematical model is constructed with the objective of minimizing the total energy consumption. Secondly, a new interior search algorithm (NISA) is presented according to the characteristics of the problem. To implement the algorithm, the design work mainly includes encoding/decoding, population initialization, discrete composition optimization, discrete mirror search, tuning of parameter α and random walk. Thirdly, extensive experiments are conducted to test the NISA's performance. The comparison results demonstrate that NISA is very competitive in solving the ESFJSP with overlapping operations and transportation times. In addition, the computational results indicate that the increase in transportation time and sublot number will incur an increase in transportation energy consumption, which is largely responsible for the increase in TTEC.

The model of the considered problem is abstracted and assumed in this work. In the next work, more practical constraints need to be integrated to be close to the real production, such as the dynamic/uncertain manufacturing environment, limited manufacturing resources (transporter, worker, etc.), job deterioration effect, time-of-use electricity strategy and so on. Moreover, we will extract some more efficient search rules from the problem, by which the computational efficiency of the algorithm will be further improved.

Author Contributions: Conceptualization, L.L. and T.J.; methodology, L.L. and T.J.; software, H.Z.; writing—original draft preparation, L.L. and H.Z.; writing—review and editing, T.J. amd C.S.; funding acquisition, L.L., T.J. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Fundamental Research Funds for the Central Universities, JLU; the Natural Science Foundation of Shandong Province (ZR2021MG008, ZR2020QG005); the Youth Entrepreneurship and Technology of Colleges and Universities in Shandong Province (2019KJN002); and the Yantai Science and Technology Planning Project (2021xdhz072); and the Major Innovation Projects in Shandong Province (2020CXGC010702, 2021CXGC010702).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are included within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mouzon, G.; Yildirim, M.B. A framework to minimise total energy consumption and total tardiness on a single machine. *Int. J. Sustain. Eng.* **2008**, *1*, 105–116. [CrossRef]
2. Shrouf, F.; Ordieres-Meré, J.; García-Sánchez, A.; Ortega-Mier, M. Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *J. Clean. Prod.* **2014**, *67*, 197–207. [CrossRef]
3. Yildirim, M.B.; Mouzon, G. Single-machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm. *IEEE Trans. Eng. Manag.* **2011**, *59*, 585–597. [CrossRef]
4. Chen, L.; Wang, J.; Xu, X. An energy-efficient single machine scheduling problem with machine reliability constraints. *Comput. Ind. Eng.* **2019**, *137*, 106072. [CrossRef]
5. Che, A.; Wu, X.; Peng, J.; Yan, P. Energy-efficient bi-objective single-machine scheduling with power-down mechanism. *Comput. Oper. Res.* **2017**, *85*, 172–183. [CrossRef]
6. Wang, S.; Wang, X.; Yu, J.; Ma, S.; Liu, M. Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan. *J. Clean. Prod.* **2018**, *193*, 424–440. [CrossRef]
7. Zhang, L.; Deng, Q.; Gong, G.; Han, W. A new unrelated parallel machine scheduling problem with tool changes to minimise the total energy consumption. *Int. J. Prod. Res.* **2020**, *58*, 6826–6845. [CrossRef]
8. Liang, P.; Yang, H.; Liu, G.; Guo, J. An ant optimization model for unrelated parallel machine scheduling with energy consumption and total tardiness. *Math. Probl. Eng.* **2015**, *2015*, 90703. [CrossRef]
9. Nanthapodej, R.; Liu, C.H.; Nitisiri, K.; Pattanapairoj, S. Variable neighborhood strategy adaptive search to solve parallel-machine scheduling to minimize energy consumption while considering job priority and control makespan. *Appl. Sci.* **2021**, *11*, 5311. [CrossRef]
10. Jia, Z.; Zhang, Y.; Leung, J.Y.T.; Li, K. Bi-criteria ant colony optimization algorithm for minimizing makespan and energy consumption on parallel batch machines. *Appl. Soft Comput.* **2017**, *55*, 226–237. [CrossRef]
11. Li, Z.; Yang, H.; Zhang, S.; Liu, G. Unrelated parallel machine scheduling problem with energy and tardiness cost. *Int. J. Adv. Manuf. Technol.* **2016**, *84*, 213–226. [CrossRef]
12. Yan, J.; Li, L.; Zhao, F.; Zhang, F.; Zhao, Q. A multi-level optimization approach for energy-efficient flexible flow shop scheduling. *J. Clean. Prod.* **2016**, *137*, 1543–1552. [CrossRef]
13. Liu, G.S.; Zhou, Y.; Yang, H.D. Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time. *J. Clean. Prod.* **2017**, *147*, 470–484. [CrossRef]

14. Gong, G.; Chiong, R.; Deng, Q.; Han, W.; Zhang, L.; Lin, W.; Li, K. Energy-efficient flexible flow shop scheduling with worker flexibility. *Expert Syst. Appl.* **2020**, *141*, 112902. [CrossRef]
15. Liu, Z.; Yan, J.; Cheng, Q.; Yang, C.; Sun, S.; Xue, D. The mixed production mode considering continuous and intermittent processing for an energy-efficient hybrid flow shop scheduling. *J. Clean. Prod.* **2020**, *246*, 119071. [CrossRef]
16. Chen, J.; Wang, L.; Peng, Z. A collaborative optimization algorithm for energy-efficient multi-objective distributed no-idle flow-shop scheduling. *Swarm Evol. Comput.* **2019**, *50*, 100557. [CrossRef]
17. Han, Y.; Li, J.; Sang, H.; Liu, Y.; Gao, K.; Pan, Q. Discrete evolutionary multi-objective optimization for energy-efficient blocking flow shop scheduling with setup time. *Appl. Soft Comput.* **2020**, *93*, 106343. [CrossRef]
18. Wu, X.; Che, A. Energy-efficient no-wait permutation flow shop scheduling by adaptive multi-objective variable neighborhood search. *Omega* **2020**, *94*, 102117. [CrossRef]
19. Salido, M.A.; Escamilla, J.; Giret, A.; Barber, F. A genetic algorithm for energy-efficiency in job-shop scheduling. *Int. J. Adv. Manuf. Technol.* **2016**, *85*, 1303–1314. [CrossRef]
20. Masmoudi, O.; Delorme, X.; Gianessi, P. Job-shop scheduling problem with energy consideration. *Int. J. Prod. Econ.* **2019**, *216*, 12–22. [CrossRef]
21. Yin, L.; Li, X.; Gao, L.; Lu, C.; Zhang, Z. Energy-efficient job shop scheduling problem with variable spindle speed using a novel multi-objective algorithm. *Adv. Mech. Eng.* **2017**, *9*, 1–21. [CrossRef]
22. He, L.; Chiong, R.; Li, W.; Dhakal, S.; Cao, Y.; Zhang, Y. Multiobjective optimization of energy-efficient job-shop scheduling with dynamic reference point-based fuzzy relative entropy. *IEEE Trans. Ind. Inform.* **2021**, *18*, 600–610. [CrossRef]
23. Giglio, D.; Paolucci, M.; Roshani, A. Integrated lot sizing and energy-efficient job shop scheduling problem in manufacturing/remanufacturing systems. *J. Clean. Prod.* **2017**, *148*, 624–641. [CrossRef]
24. Abedi, M.; Chiong, R.; Noman, N.; Zhang, R. A multi-population, multi-objective memetic algorithm for energy-efficient job-shop scheduling with deteriorating machines. *Expert Syst. Appl.* **2020**, *157*, 113348. [CrossRef]
25. Afsar, S.; Palacios, J.J.; Puente, J.; Vela, C.R.; González-Rodríguez, I. Multi-objective enhanced memetic algorithm for green job shop scheduling with uncertain times. *Swarm Evol. Comput.* **2022**, *68*, 101016. [CrossRef]
26. Wu, X.; Shen, X.; Li, C. The flexible job-shop scheduling problem considering deterioration effect and energy consumption simultaneously. *Comput. Ind. Eng.* **2019**, *135*, 1004–1024. [CrossRef]
27. Caldeira, R.H.; Gnanavelbabu, A.; Vaidyanathan, T. An effective backtracking search algorithm for multi-objective flexible job shop scheduling considering new job arrivals and energy consumption. *Comput. Ind. Eng.* **2020**, *149*, 106863. [CrossRef]
28. Dai, M.; Tang, D.; Giret, A.; Salido, M.A. Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robot. Comput. Integr. Manuf.* **2019**, *59*, 143–157. [CrossRef]
29. Yin, L.; Li, X.; Gao, L.; Lu, C.; Zhang, Z. A novel mathematical model and multi-objective method for the low-carbon flexible job shop scheduling problem. *Sustain. Comput. Inform. Syst.* **2017**, *13*, 15–30. [CrossRef]
30. Liu, Z.; Guo, S.; Wang, L. Integrated green scheduling optimization of flexible job shop and crane transportation considering comprehensive energy consumption. *J. Clean. Prod.* **2019**, *211*, 765–786. [CrossRef]
31. Jiang, T.; Zhu, H.; Deng, G. Improved African buffalo optimization algorithm for the green flexible job shop scheduling problem considering energy consumption. *J. Intell. Fuzzy Syst.* **2020**, *38*, 4573–4589. [CrossRef]
32. Li, M.; Lei, D. An imperialist competitive algorithm with feedback for energy-efficient flexible job shop scheduling with transportation and sequence-dependent setup times. *Eng. Appl. Artif. Intell.* **2021**, *103*, 104307. [CrossRef]
33. Zhang, H.; Xu, G.; Pan, R.; Ge, H. A novel heuristic method for the energy-efficient flexible job-shop scheduling problem with sequence-dependent set-up and transportation time. *Eng. Optim.* **2021**. (*in process*) [CrossRef]
34. Gong, G.; Deng, Q.; Gong, X.; Liu, W.; Ren, Q. A new double flexible job-shop scheduling problem integrating processing time, green production, and human factor indicators. *J. Clean. Prod.* **2018**, *174*, 560–576. [CrossRef]
35. Peng, Z.; Zhang, H.; Tang, H.; Feng, Y.; Yin, W. Research on flexible job-shop scheduling problem in green sustainable manufacturing based on learning effect. *J. Intell. Manuf.* 2021. (in process). [CrossRef]
36. Zhu, H.; Deng, Q.; Zhang, L.; Hu, X.; Lin, W. Low carbon flexible job shop scheduling problem considering worker learning using a memetic algorithm. *Optim. Eng.* **2020**, *21*, 1691–1716. [CrossRef]
37. Wei, Z.; Liao, W.; Zhang, L. Hybrid energy-efficient scheduling measures for flexible job-shop problem with variable machining speeds. *Expert Syst. Appl.* **2022**, *197*, 116785. [CrossRef]
38. Jiang, T.; Zhu, H.; Liu, L.; Gong, Q. Energy-conscious flexible job shop scheduling problem considering transportation time and deterioration effect simultaneously. *Sustain. Comput. Inform. Syst.* **2022**, *35*, 100680. [CrossRef]
39. Jiang, T.; Zhu, H.; Gu, J.; Liu, L.; Song, H. A discrete animal migration algorithm for dual-resource constrained energy-saving flexible job shop scheduling problem. *J. Intell. Fuzzy Syst.* **2022**, *42*, 3431–3444. [CrossRef]
40. Lei, D.; Guo, X. Scheduling job shop with lot streaming and transportation through a modified artificial bee colony. *Int. J. Prod. Res.* **2013**, *51*, 4930–4941. [CrossRef]
41. Demir, Y.; İşleyen, S.K. An effective genetic algorithm for flexible job-shop scheduling with overlapping in operations. *Int. J. Prod. Res.* **2014**, *52*, 3905–3921. [CrossRef]
42. Meng, T.; Pan, Q.K.; Sang, H.Y. A hybrid artificial bee colony algorithm for a flexible job shop scheduling problem with overlapping in operations. *Int. J. Prod. Res.* **2018**, *56*, 5278–5292. [CrossRef]
43. Li, M.; Wang, G.G. A review of green shop scheduling problem. *Inf. Sci.* **2022**, *589*, 478–496. [CrossRef]

44. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [CrossRef]
45. Gandomi, A.H. Interior search algorithm (ISA): A novel approach for global optimization. *ISA Trans.* **2014**, *53*, 1168–1183. [CrossRef]
46. Fathy, A.; Rezk, H. Robust electrical parameter extraction methodology based on Interior Search Optimization Algorithm applied to supercapacitor. *ISA Trans.* **2020**, *105*, 86–97. [CrossRef] [PubMed]
47. Han, B.; Huang, C.; Tang, S.; Xuan, Y.; Zhang, Z.; Huan, Z. Random orthocenter strategy in interior search algorithm and its engineering application. *Soft Comput.* **2020**, *24*, 5933–5948. [CrossRef]
48. Torabi, N.; Tavakkoli-Moghaddam, R.; Najafi, E.; Lotfi, F.H. Multi-objective interior search algorithm for optimization: A new multi-objective meta-heuristic algorithm. *J. Intell. Fuzzy Syst.* **2018**, *35*, 3307–3319. [CrossRef]
49. Torabi, N.; Tavakkoli-Moghaddam, R.; Najafi, E. A Two-Stage Green Supply Chain Network with a Carbon Emission Price by a Multi-objective Interior Search Algorithm. *Int. J. Eng.* **2019**, *32*, 828–834.
50. Yıldız, B.S. Natural frequency optimization of vehicle components using the interior search algorithm. *Mater. Test.* **2017**, *59*, 456–458. [CrossRef]
51. Zhang, G.; Gao, L.; Shi, Y. An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Syst. Appl.* **2011**, *38*, 3563–3573. [CrossRef]
52. Pezzella, F.; Morganti, G.; Ciaschetti, G. A genetic algorithm for the flexible job-shop scheduling problem. *Comput. Oper. Res.* **2008**, *35*, 3202–3212. [CrossRef]
53. Zhu, H.; Jiang, T.; Wang, Y. Discrete African Buffalo Optimization Algorithm for the Low-carbon Flexible Job Shop Scheduling Problem. *J. Adv. Manuf. Syst.* **2020**, *19*, 837–854. [CrossRef]
54. Gandomi, A.H.; Roke, D.A. Engineering optimization using interior search algorithm. In Proceedings of the 2014 IEEE Symposium on Swarm Intelligence, Orlando, FL, USA, 9–12 December 2014; pp. 1–7.
55. Kacem, I.; Hammadi, S.; Borne, P. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2002**, *32*, 1–13. [CrossRef]
56. Brandimarte, P. Routing and scheduling in a flexible job shop by tabu search. *Ann. Oper. Res.* **1993**, *41*, 157–183. [CrossRef]
57. Ren, W.; Wen, J.; Yan, Y.; Hu, Y.; Guan, Y.; Li, J. Multi-objective optimisation for energy-aware flexible job-shop scheduling problem with assembly operations. *Int. J. Prod. Res.* **2021**, *59*, 7216–7231. [CrossRef]