Received July 3, 2019, accepted July 18, 2019, date of publication July 22, 2019, date of current version August 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2930281

Energy-Conscious Scheduling Problem in a Flexible Job Shop Using a Discrete Water Wave Optimization Algorithm

YI LU[1], JIACHENG LU[1], AND TIANHUA JIANG[2]
[1]Henan Institute of Technology, Xinxiang 453003, China
[2]School of Transportation, Ludong University, Yantai 264025, China

Corresponding author: Jiacheng Lu (ljc608@163.com)

This work was supported in part by the Special Research and Promotion Program of Henan Province under Grant 182102210257, in part by the Project of Henan Province Higher Educational Key Research Program under Grant 16A120011, in part by the High-level Talent Research Foundation of Henan Institute of Technology under Grant KQ1863, and in part by the Shandong Provincial Natural Science Foundation of China under Grant ZR2016GP02.

ABSTRACT As more and more attention is paid to green manufacturing, production scheduling has been proved to be an efficient method for the reduction of environmental pollution. It is well-known that the flexible job shop scheduling problem (FJSP) is a very complex combinatorial optimization problem with strong theoretical and background for application. However, the problem has been extensively investigated and historically concerned with some indicators related to time, e.g., flow time, makespan, and workload. In this study, an energy-conscious FJSP is investigated with the consideration of the energy consumption. First, a mathematical model of the energy-conscious FJSP is built with the objective of optimizing the sum of the energy consumption cost and the completion-time cost. Due to the fact that the basic water wave optimization (WWO) was developed for various continuous problems, a discrete water wave optimization (DWWO) algorithm is proposed to solve the model. In our DWWO algorithm, a three-string encoding approach is first adopted to represent each individual wave. To make the algorithm adapt for the considered scheduling problem, three discrete evolutionary operations are redesigned according to the characteristics of the problem, i.e., propagation, refraction, and breaking. Finally, extensive experimental simulations are conducted to test the proposed DWWO algorithm. The comparison results demonstrate that the proposed DWWO algorithm is efficient for the energy-conscious FJSP.

INDEX TERMS Flexible job shop, energy-conscious scheduling, energy consumption. discrete water wave optimization algorithm.

I. INTRODUCTION

Flexible job shop scheduling problem (FJSP) is an extended version of the job shop scheduling problem (JSP), which presents a closer approximation to the real-life production than the JSP. It has attracted much attention of researchers in the manufacturing field [1]–[15]. However, these previous researches mainly focused on the FJSP with some traditional time-related objectives, such as makespan, flow time, workload, and so on. Nowadays, under the current environmental pressure, some environmental metrics are taken into account in the scheduling problems, such as energy consumption, CO2 emission, carbon footprint, etc. The energy-conscious scheduling problem has become a new research hotspot in the manufacturing filed [16]. Some previous studies can be summarized according to the research workshops as below. (1) Single-machine scheduling problem. Shrouf *et al.* [17] investigated the problem to minimize energy consumption costs by considering variable energy prices during a day. Yildirim and Mouzon [18] established a mathematical model of the problem and developed a genetic algorithm to minimize the energy consumption and the total completion time. Che *et al.* [19] proposed a greedy insertion algorithm to solve the established model of a single-machine system under time-of-use (TOU) policy. Mouzon and Yildirim [20] proposed a greedy randomized adaptive algorithm to minimize the total energy consumption and the total tardiness.

The associate editor coordinating the review of this manuscript and approving it for publication was Jihwan P. Choi.

(2) Flow shop scheduling problem. Lu *et al.* [21] established a mathematical model of a permutation flow shop scheduling problem and presented a hybrid backtracking search algorithm to optimize the makespan and the energy consumption. Tang *et al.* [22] investigated a dynamic flexible flow shop scheduling problem aiming to reduce the energy consumption and the makespan. A particle swarm optimization algorithm was proposed to obtain the Pareto optimum. Liu *et al.* [23] considered a fuzzy flow shop scheduling problem and proposed a hybrid genetic algorithm to minimize the energy consumption and the tardiness. Yan *et al.* [24] developed a multi-level optimization method for an energy-efficient flexible flow shop scheduling. Mansouri *et al.* [25] proposed a multi-objective model of a two-machine flow shop and proposed a heuristic method to find the Pareto front of the makespan and the total energy consumption. Zhang *et al.* [26] established a model to optimize the electricity cost and the carbon footprint under time-of-use tariffs. Ding *et al.* [27] considered a permutation flow shop scheduling problem to optimize the total carbon emission and the makespan. Meng *et al.* [28] established five mixed integer linear programming models for an energy-conscious hybrid flow shop scheduling problem with the energy-saving strategy of turning off and on. For the model, an improved genetic algorithm was proposed.

(3) Job shop scheduling problem. Zhang and Chiong [29] developed a genetic algorithm to solve a multi-objective energy-efficient job shop scheduling problem. Liu *et al.* [30] adopted a non-dominant sorting genetic algorithm to minimize the total electricity consumption and the total weighted tardiness. May *et al.* [31] developed a genetic algorithm to improve the productive and the environmental performances. Salido *et al.* [32] presented a genetic algorithm to solve an energy-efficient job shop scheduling problem with machine speed scaling. Tang and Dai [33] built a mixed integer programming mathematical model and proposed a genetic-simulated annealing algorithm to find the optimal solution of the energy-efficient JSP. Escamilla *et al.* [34] presented a genetic algorithm to solve an energy-efficient JSP, where machines can work at different speeds.

With regards to the literature about the energy-conscious production scheduling problems, most of existing studies focus on some simple manufacturing systems. Flexible job shop is a very important workshop type and plays a great role in the manufacturing filed. However, to the best of the authors' knowledge, the research on the energy-conscious FJSP has only just started [35]. Jiang *et al.* [36] constructed a model of a multi-objective FJSP and proposed a non-dominated sorting genetic algorithm to minimize the makespan, processing cost, energy consumption and cost-weighted processing quality. Mokhtari and Hasani [37] investigated an energy-efficient FJSP to optimize total completion time, total availability of the system, and total energy cost. Piroozfard *et al.* [38] established a model with the consideration of the energy consumption and preventive maintenance of machines. An NSGA-II algorithm was presented to obtain the minimization of total production energy costs and total maintenance energy costs. Meng *et al.* [39] investigated the flexible job shop scheduling problem to minimize the total energy consumption and proposed six mixed integer linear programming models with turning off/on strategy. Meng *et al.* [40] studied the dual-resource constrained flexible job shop scheduling problem with minimizing energy consumption and proposed a variable neighborhood search algorithm for the problem. However, facing to the environmental pressure, considerable work has yet to be carried out, and some realistic constraints should be considered in the energy-conscious FJSP.

For the above reviewed literature, the processing time of each job on machines are fixed and certain. However, in some actual manufacturing systems, such as CNC machines, machines can work at different speeds when dealing with different jobs. When working at a higher speed, the processing time decreases but the energy consumption increases, and when working at a lower speed, the processing time increases while the energy consumption decreases [32]. This mechanism provides an opportunity to control the energy consumption by adjusting the machine speed. Therefore, the machine speed should be taken as a decision-making variable. The energy-conscious FJSP is consisted of three sub-problems: machine assignment, operation permutation and speed selection. The addition of speed selection greatly increases the complexity of the problem. There are rather few literature about the energy-conscious FJSP with adjustable machine speeds. Lei *et al.* [41] proposed a shuffled frog-leaping algorithm (SFLA) to minimize the workload balance and the energy consumption. Wu and Sun [42] formulated a mathematical model of FJSP with the objective of saving energy consumption. A non-dominated sorted genetic algorithm was developed to determine when to turn-on/off machines and which speed level to choose. In view of the complexity of the considered problem, it is very difficult to get the optimal solution through exact methods even for a small-scale instance. In recent years, the application of intelligence algorithms on workshop scheduling problems has become a research hotspot. Therefore, more effective and efficient intelligence algorithms are highly desirable for solving the energy-conscious FJSP.

Water wave optimization (WWO) is a new population-based intelligence algorithm, which originates from shallow water wave models to deal with continuous optimization problems [43]. Three wave motions (propagation, breaking and refraction) are included in the algorithm to implement the iterative searching process. The cooperation between the operators makes the WWO maintain a good balance between exploration and exploitation. Meanwhile, compared with other meta-heuristic algorithms, WWO performs well with a small-scale population, which make it has low computational effort. Currently, WWO has been successfully employed to deal with various optimization problems [44]–[49]. Nevertheless, due to the fact that WWO has been proposed not so long time, the applications of the algorithm are still scare. To the best of our knowledge, the application of WWO for solving the energy-conscious FJSP has not yet been reported. Therefore, the reasons above motivate us to design a discrete WWO (DWWO) to solve the energy-conscious FJSP. The main contribution can be summarized as follows: (1) This study attempts to study the energy-conscious FJSP with adjustable machine speeds, which is seldom considered in the existing literature. (2) A new mathematical model of the energy-conscious FJSP is established with the criterion to minimize the sum of the energy consumption cost and the completion-time cost. (3) As mentioned before, WWO was originally proposed for global optimization in continuous search spaces. However, the considered problem in this study is a discrete problem. Therefore, according to the characteristics of the problem, the three evolutionary operators are redesigned to make the algorithm directly search in a discrete scheduling domain. For the propagation operator, three mutation operators are adopted to obtain a new wave. The seeking memory pool mechanism of cat swarm optimization (CSO) is used to avoid generating poor solutions; For the refraction operator, the crossover operation is conducted between the poor solution and the current best solution to absorb some desirable information; For the breaking operator, a local search strategy is designed to enhance the exploitation capability.

The rest of the paper is organized as follows. Section II introduces the description of the problem and the model. Section III addresses the basic water wave optimization algorithm. Section IV describes the implementation of the proposed DWWO algorithm. Section V shows the experimental results and Section VI gives conclusions and future works.

II. MATHEMATICAL MODEL OF THE ENERGY-CONSCIOUS FJSP

The energy-conscious FJSP can be described as follows: there is a set of n jobs to be processed on m machines. Each job consists of a sequence of several operations. The processing time depends on the assigned machine of each operation and the selected speed level of the machine. Here, a finite and discrete speed set $\boldsymbol{v} = \{v_1, v_2, \cdots, v_d\}$ is considered for each machine. The selected speed affects the production rate and the energy consumption. Here, O_{ij} means operation i of job j. If O_{ij} is assigned to machine k, a basic processing time is represented by q_{ijk}. If O_{ij} is processed on machine k at speed v_d, the actual processing time p_{ijkd} equals to q_{ijk}/v_d, and the energy consumption cost per unit time is measured by E_{kd}. Furthermore, if $v_{d'} > v_d$, $E_{kd'} \times p_{ijkd'} > E_{kd} \times p_{ijkd}$ is assumed to be met. When waiting for process, machine runs with an energy consumption cost per unit time measured by SE_k in the stand-by mode. Some constraints are involved as below.

(1) All machines and jobs are ready at the beginning time.

(2) Each machine must process at most one operation at the same time.

(3) Each job must not be reassigned when it is being processed.

(4) Preemption is not permitted for each operation.

(5) Each operation cannot be processed until the immediate predecessor is completed.

(6) Setup and breakdown of machines are not considered.

(7) The speed of a machine must not be changed when processing a job.

(8) Any machine cannot be stopped until all assigned jobs are finished.

The optimization objective is to minimize the sum of total energy consumption cost and the completion-time cost. For the mathematical model, some symbols and variables are listed as below.

n : The number of jobs;

m : The number of machines;

J_i : The number of operations in job i;

γ : The completion-time cost per unit time;

$C_{\max}$: The final completion time;

ST_{ij} : The start time of O_{ij};

CT_{ij} : The completion time of O_{ij};

ξ : A big positive constant number;

CS : The total cost in the workshop;

CT_k : The completion time of machine k;

W_k : The workload of machine k;

y_{ijkd} : 0-1 variable, if O_{ij} is processed on machine k at speed v_d, $y_{ijkd} = 1$; otherwise, $y_{ijkd} = 0$;

$z_{iji'j'k}$: 0-1 variable, if O_{ij} is processed on machine k prior to $O_{i'j'}$, $z_{iji'j'k} = 1$; otherwise, $z_{iji'j'k} = 0$.

$$\min\ CS = \sum_{i=1}^{n}\sum_{j=1}^{J_i}\sum_{k=1}^{m}\sum_{d=1}^{D} E_{kd}\, y_{ijkd}\, p_{ijkd} + \sum_{k=1}^{m} SE_k(CT_k - W_k) + \gamma C_{\max} \tag{1}$$

$$\text{s.t. } CT_{ij} - ST_{ij} = \sum_{k=1}^{m}\sum_{d=1}^{D} y_{ijkd}\, p_{ijkd},\ i=1,2,\cdots,n;\ j=1,2,\cdots,J_i \tag{2}$$

$$ST_{i(j+1)} - CT_{ij} \geq 0,\ \ i=1,2,\cdots,n; j=1,2,\cdots,J_i-1 \tag{3}$$

$$ST_{i'j'} + \xi(1 - z_{iji'j'k}) \geq CT_{ij},\quad i,i'=1,2,\cdots,n;\ j,j'=1,2,\cdots,J_i;\ k=1,2,\cdots,m \tag{4}$$

$$ST_{ij} + \xi z_{iji'j'k} \geq CT_{i'j'},\quad i,i'=1,2,\cdots,n;\ j,j'=1,2,\cdots,J_i;\ k=1,2,\cdots,m \tag{5}$$

$$\sum_{k=1}^{m}\sum_{d=1}^{D} y_{ijkd} = 1,\ \ i=1,2,\cdots,n;\ \ j=1,2,\cdots,J_i \tag{6}$$

$$y_{ijkd} \in \{0,1\},\quad i=1,2,\cdots,n; j=1,2,\cdots,J_i; k=1,2,\cdots,m;\quad d=1,2,\cdots,D \tag{7}$$

$$z_{iji'j'k} \in \{0,1\},\quad i,i'=1,2,\cdots,n;\ j,j'=1,2,\cdots,J_i;\ k=1,2,\cdots,m \tag{8}$$

Equation (1) is the optimization function with three items, each of which represents a different cost. The first item is the total energy consumption cost for processing operations, the second is the total energy consumption cost when machines run in the stand-by mode, and the third is the cost related to the completion-time; Constraint (2) indicates that preemption is not allowed during the processing period of each operation; Constraint (3) shows the precedence relationship of operations. That is to say, each operation cannot be processed until the immediate predecessor is completed; Constraints (4) and (5) guarantee that each machine must process no more than one operation simultaneously; Constraint (6) means that any operation cannot be assigned to another machine when it has been started, and the speed of a machine cannot be changed when it is processing a job. Constraints (7) and (8) presents the decision variables.

III. INTRODUCTION TO THE BASIC WATER WAVE OPTIMIZATION

Water wave optimization (WWO) algorithm is inspired from shallow wave models and proposed by Zheng [43] for solving various continuous optimization problems. In the WWO, each individual is taken as a wave with a height h and a wavelength λ. The searching space corresponds to the seabed area. For a maximization optimization problem, the fitness of each solution is measured inversely by its seabed depth [43]. For each wave $\boldsymbol{x} = \{x(1), x(2), \cdots, x(l)\}$, h is initially set to be a constant $h_{\max}$ and λ is set to be 0.5. During the evolutionary process, three operations (propagation, refraction and breaking) are involved in the algorithm and performed to waves.

In the propagation phase, for each individual, a new wave $\boldsymbol{x'}$ is created based on the original one η at each generation, which can be shown in Equation (9).

$$x'(a) = x(a) + rand(-1, 1)\lambda L(a) \tag{9}$$

$rand(-1, 1)$ defines a random number in the range [−1,1] with a uniform distribution, and $L(a)$ represents the length of ath dimension of the search space. If $\boldsymbol{x}'$ is better than $\boldsymbol{x}$, $\boldsymbol{x}$ is replaced by $\boldsymbol{x}'$, and the height h of $\boldsymbol{x}'$ is reset to $h_{\max}$. Otherwise, $\boldsymbol{x}$ is unchanged, but its height h is decreased by one. After each generation, the wavelength of each wave $\boldsymbol{x}$ is updated by Equation (10).

$$\lambda = \lambda\alpha^{-(f - f_{\min}+\varepsilon)/(f_{\max} - f_{\min}+\varepsilon)} \tag{10}$$

where f is the fitness function, $f_{\max}$ and $f_{\min}$ represents the maximum and minimum values of f, α is the wavelength reduction coefficient, and ε is a factor used to avoid division-by-zero.

After the propagation operation, if a wave has not been improved and its height h equals to 0, a refraction operation in Equation (11) is employed to make the wave move to a new position by learning from the current best wave x^*.

$$x'(a) = N(\frac{x^*(a) + x(a)}{2}, \frac{|x^*(a) - x(a)|}{2}) \tag{11}$$

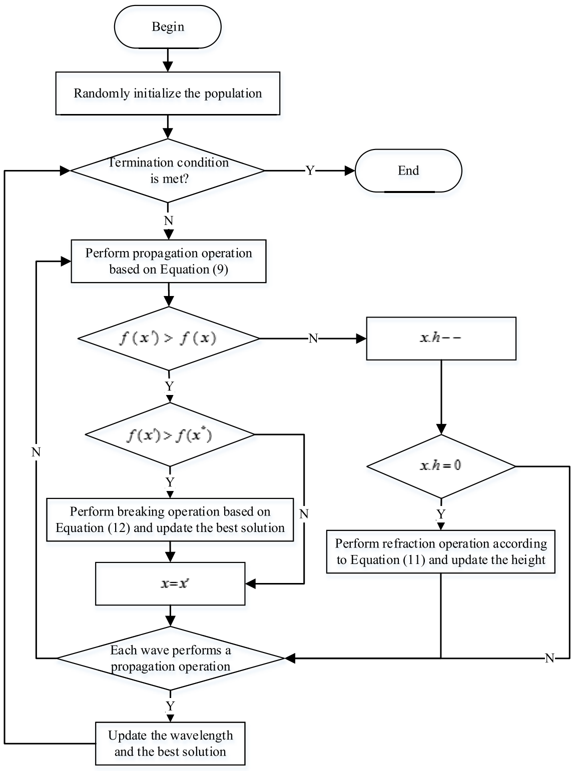
Begin
Randomly initialize the population
Termination condition is met?
Y
End
N
Perform propagation operation based on Equation (9)
f(x') > f(x)
N
x.h− −
Y
f(x') > f(x*)
x.h = 0
N
Y
Perform breaking operation based on Equation (12) and update the best solution
N
Y
Perform refraction operation according to Equation (11) and update the height
x=x'
Each wave performs a propagation operation
N
Y
Update the wavelength and the best solution

FIGURE 1. **The framework of the original WWO algorithm.**

where N means a Gaussian random number with specified mean and standard deviation. After the refraction operation, $\boldsymbol{x}$ is taken place by $\boldsymbol{x}'$, and its height h is reset to $h_{\max}$, and the wavelength λ is calculated by $\lambda' = \lambda\frac{f(x)}{f(x')}$.

In the WWO, the breaking operator is used to enhance local search around a promising solution. If a wave is better than the current best solution x^*, kk dimensions of the waves will be randomly selected first. Then the breaking operation in Equation (12) will be performed to each selected dimension to generate kk new waves.

$$x'(a) = x(a) + N(0, 1)\beta L(a) \tag{12}$$

where β is named as the breaking coefficient, kk is a random number selected from $[1, kk_{\max}]$, $kk_{\max}$ is a predefined parameter. If none of the new waves are better than $\boldsymbol{x}^*$, $\boldsymbol{x}^*$ is unchanged; otherwise, $\boldsymbol{x}^*$ is replaced by the best one among the new waves generated by Equation (12). The steps of the WWO algorithm can be shown by Figure 1.

IV. IMPLEMENTATION OF THE PROPOSED DWWO

A. ENCODING AND DECODING APPROACH

As stated above, WWO was originally proposed for solving continuous optimization problems. Therefore, the first task is to design an appropriate encoding approach to represent the scheduling solution. Here, a three-string encoding method is adopted for the energy-conscious FJSP. The first string defines the machine assignment, the second shows the speed level selection, and the third is the operation permutation.

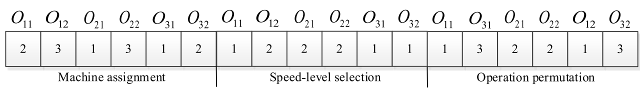
O_{11} O_{12} O_{21} O_{22} O_{31} O_{32} O_{11} O_{12} O_{21} O_{22} O_{31} O_{32} O_{11} O_{31} O_{21} O_{22} O_{12} O_{32}
2 3 1 3 1 2 1 2 2 2 1 1 1 3 2 2 1 3
Machine assignment
Speed-level selection
Operation permutation

FIGURE 2. Encoding scheme for a 3 × 3 × 2 energy-conscious FJSP.

For a 3 × 3 × 2 problem, three jobs are assumed to be processed on three machines, and two speed levels can be selected for each machine. Each job consists of two operations. The encoding scheme can be illustrated in Figure 2. In the first string, each element is the code of the selected machine for each operation. In the second string, each element is the selected speed level for processing each operation. In the third string, elements with the same values represent different operations in the same job.

In this paper, the population is randomly initialized according to the encoding method. In order to obtain a feasible scheduling scheme, the decoding procedure can be described as follows:

Step 1: Scan the operation permutation in the first string from left to right, read the machine information in the first string, and find the speed level of the assigned machine in the second string.

Step 2: The first operation in the third string is first arranged, and then the second one is done, and so on; each operation is allocated in the best available time on its assigned machine. The procedure is repeated until a scheduling scheme is obtained.

B. PROPAGATION OPERATOR

In the WWO algorithm, the propagation operator is used to generate a new wave. During the evolutionary process, each wave has to propagate to implement the individual updating. Observed from Figure 2, Equation (9) cannot be directly conducted to scheduling solutions. Thus, a modified propagation operator is presented according to the characteristics of the problem.

In the proposed DWWO, a perturbation procedure is proposed based on three mutation operations and performed to obtain a new solution. The wavelength λ is taken as the mutation probability, which can be calculated according to Equation (13), where f is the fitness function represented by W/CS (W is a constant). Seen from Equation (13), the wavelength of each wave varies according to the fitness. A poor solution has a larger λ, and thus has a larger probability of being mutated; on the contrary, a good solution has a smaller λ, and thus has a smaller probability to be mutated.

$$\lambda = \frac{f_{\max} - f_k}{f_{\max} - f_{\min}} \tag{13}$$

Three mutation operators are adopted as follows:

Machine Assignment: Randomly select an operation with more than one alternative machine from the first part of the scheduling solution, and then a different machine is randomly selected from the alternative machine set of the selected operation to replace the original machine.

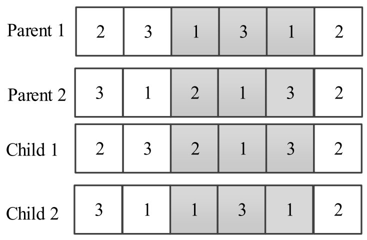

FIGURE 3. TPX crossover for the machine assignment part.

Speed-level Selection: Randomly selected an operation with more than one alternative speed level from the second part, and then a different speed level is randomly selected to replace the original speed.

Operation Permutation: Randomly select two operations belonging to different jobs in the third part, and then the positions of the selected operations are exchanged.

In every iteration, for each individual, a random number is generated from [0,1] following the uniform distribution. If the random number is smaller than the wavelength λ, a new solution is created by randomly performing one mutation operator or one combination of them to the current solution. It should be indicated that a random perturbation may result to a poor solution. To overcome this drawback, the seeking memory pool (SMP) mechanism of the cat swarm optimization (CSO) proposed by Chu [50] is adopted for each current wave, which can be described as follows:

Step 1: Generate η copies of the current wave $\boldsymbol{x}$.

Step 2: Perform the mutation operators to each copy at random.

Step 3: Evaluate the fitness values of all copies, and find the copy $\boldsymbol{x}'$ with the best fitness.

Step 4: If the best copy $\boldsymbol{x}'$ is better than the original wave $\boldsymbol{x}$, update the wave $\boldsymbol{x}$ by $\boldsymbol{x}'$; otherwise, maintain the value of the original wave.

C. REFRACTION OPERATOR

The refraction operator is mainly used to make poor solution obtain some desirable information from the current best solution. If $\boldsymbol{x}$ is not improved after several propagation operations and its wave height λ equals to zero, a refraction operation should be conducted to $\boldsymbol{x}$. In this study, a crossover operator is employed to implement the refraction operation, which is carried out between the current wave $\boldsymbol{x}$ and the best wave $\boldsymbol{x}^*$. Here, a two-point crossover (TPX) is used both for the machine assignment part and the speed-level selection part, and the precedence preserving order-based crossover (POX) is adopted for the operation permutation part.

The detailed steps of the TPX are illustrated in Figures 3-4 and described as follows:

Step 1: Randomly select two positions from machine assignment part or speed-level selection part.

Step 2: Exchange the elements between the selected positions.

Step 3: End the procedure.

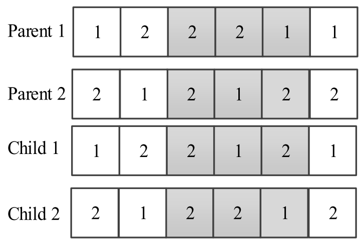

FIGURE 4. TPX crossover for the speed-level selection part.

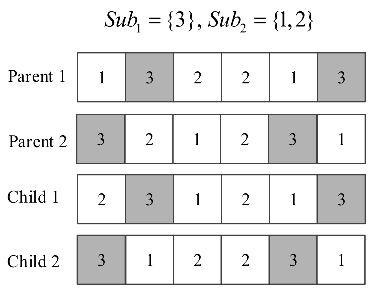

FIGURE 5. POX crossover operation.

The detailed steps of the POX crossover are illustrated in Figure 5 and described as follows:

Step 1: Create two job sets Job_1 and Job_2.

Step 2: Randomly select some jobs to Job_1, others are selected to Job_2.

Step 3: Copy the jobs in Job_1 from Parent 1 to Child 1 and from Parent 2 to Child 2.

Step 4: Copy the jobs in Job_2 from Parent 2 to Child 1 and from Parent 1 to Child 2.

Step 5: End the procedure.

D. BREAKING OPERATOR

The breaking operator aims to enhance the exploitation ability around the optimal solution. When a new wave generated from the propagation phase is better than the current best solution, the breaking operator is performed to the new wave and generate a better solution. The local search strategy (LS) has been widely used to find more promising solutions. Here, it is employed to be the breaking operator. In the proposed LS, the mutation operators in Section B are taken as the neighborhood structures. The procedure of the LS can be shown as below.

Step 1: Obtain the initial solution $\boldsymbol{x}$, and set $\rho_{\max} \leftarrow 3$, $\zeta \leftarrow 1$ and the maximum iteration $\zeta_{\max}$.

Step 2: Set $\rho \leftarrow 1$.

Step 3: Perform the below procedure until $\rho > \rho_{\max}$.

```
    if $\rho = 1$ then $\boldsymbol{x}' \in$ OperationPermutation($\boldsymbol{x}$)
    elseif $\rho = 2$ then $\boldsymbol{x}' \in$ MachineAssignment($\boldsymbol{x}$)
    else $\boldsymbol{x}' \in$ SpeedSelection($\boldsymbol{x}$)
    endif
    if $f(\boldsymbol{x}') > f(\boldsymbol{x})$ then $\boldsymbol{x} \leftarrow \boldsymbol{x}'$, $\rho \leftarrow \rho$
    else $\rho \leftarrow \rho + 1$
    endif
```

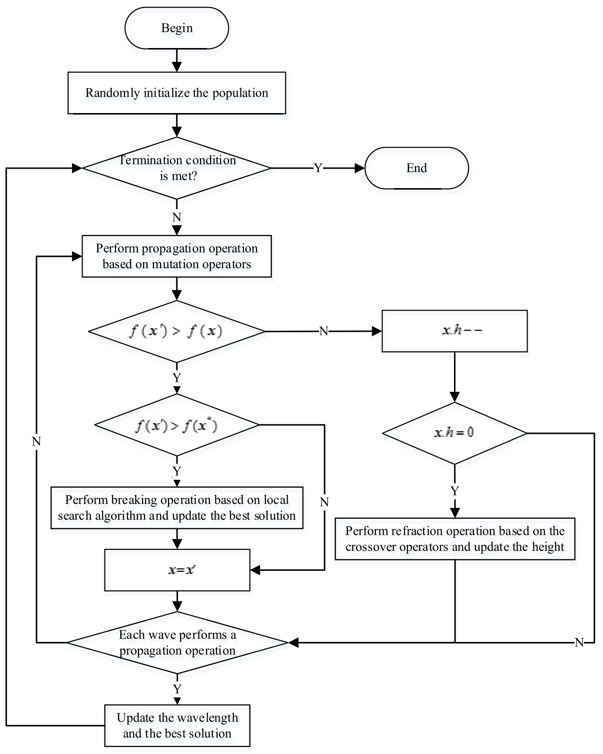
Begin
Randomly initialize the population
Termination condition is met?
Y
End
N
Perform propagation operation based on mutation operators
$f(x') > f(x)$
N
$x.h--$
Y
$f(x') > f(x^*)$
$x.h = 0$
N
Y
Perform breaking operation based on local search algorithm and update the best solution
N
Y
Perform refraction operation based on the crossover operators and update the height
$x = x'$
Each wave performs a propagation operation
N
Y
Update the wavelength and the best solution

FIGURE 6. The procedure of the proposed DWWO.

Step 4: Set $\zeta \leftarrow \zeta + 1$, if $\zeta > \zeta_{\max}$, go to Step 5, otherwise, go to Step 2.

Step 5: End the procedure.

E. PROCEDURE OF THE PROPOSED DWWO

Based on the above description, the whole framework of DWWO algorithm is illustrated in Figure 6.

V. COMPUTATIONAL EXPERIMENTS

The DWWO algorithm is implemented by using FORTRAN language and run on a VMware Workstation Pro 14 with 2GB main memory under WinXP. Here, 56 instances (RM01-RM56) are designed to evaluate the performance of the proposed DWWO. For each instance, ten independent replications are conducted for different algorithms.

A. EXPERIMENTAL INSTANCES

The RM01-RM56 instances are constructed according to the number of machines ($m \in \{10, 15, 20, 25, 30, 35, 40\}$) and the number of jobs ($n \in \{20, 30, 50, 70, 80, 100, 120, 150\}$). In addition, some main parameters are randomly generated following a discrete uniform distribution. The number of operations of each job is generated in the range [1, 5], the number of alternative machines for each operation is generated in the range [1, m], the basic processing time of each operation is drawn from the range [1, 20]. The speed of each machine can be selected from $\boldsymbol{v} = \{v_1, v_2, v_3, v_4, v_5\} = \{1.0, 1.2, 1.5, 2.0, 2.5\}$. Moreover, E_{kd} can be measured

TABLE 1. Parameter levels.

Factor	Level			
	1	2	3	4
PS	30	50	80	100
$h_{\max}$	5	10	15	20
$\zeta_{\max}$	10	20	30	40
η	5	10	15	20

TABLE 2. Orthogonal array and Avg values.

Combinatio n number	Factor				Avg
	PS	$h_{\max}$	$\zeta_{\max}$	η	
1	1	1	1	1	3153.1
2	1	2	2	2	3227.8
3	1	3	3	3	3126.4
4	1	4	4	4	3095.0
5	2	1	2	3	3062.4
6	2	2	1	4	3124.0
7	2	3	4	1	3471.8
8	2	4	3	2	3252.1
9	3	1	3	4	3035.5
10	3	2	4	3	3138.1
11	3	3	1	2	3309.2
12	3	4	2	1	3551.4
13	4	1	4	2	3035.6
14	4	2	3	1	3336.5
15	4	3	2	4	3115.4
16	4	4	1	3	3189.7

by $\xi_k \times v_d^2, d = 1, 2, 3, 4, 5$, where ξ_k is randomly chosen in [2, 4] with a discrete uniform distribution. SE_k can be calculated by $\xi_k/4$. And γ is set to be 15.0.

B. PARAMETERS SETTING

In the DWWO, four parameters need to be tuned, i.e., the population size (PS), the maximum height ($h_{\max}$), the maximum iteration ($\zeta_{\max}$) of local search strategy in the breaking operator, the number of copies (η) generated for each solution in the propagation operator. Here, the Taguchi method of Design of Experiments (DOE) is used to determine these parameters. For each parameter, four levels are considered in Table 1. For each parameter setting, the DWWO algorithm is conducted with the maximum iteration $5 \times m \times n$. The orthogonal array $L_{16}(4^4)$ is listed in Table 2. In the table, experiments are conducted based on an instance with 15 machines and 50 operations (RM11). The average value of objective function in the ten runs (Avg) is adopted to verify the computational results.

According to Table 2, the response value and the significance rank are shown in Table 3. Then, the factor level trend of each parameter is illustrated in Figure 7. It can be observed from Table 3 that η is the most significant parameter. The second significant parameter is $h_{\max}$. Besides, Table 3 shows that PS is more significant than $\zeta_{\max}$ for the performance of the

TABLE 3. Response value and significance rank.

Level	PS	$h_{\max}$	$\zeta_{\max}$	η
1	3150.575	3071.65	3194.0	3378.2
2	3227.575	3206.6	3239.25	3206.175
3	3258.55	3255.7	3187.625	3129.15
4	3169.3	3272.05	3185.125	3092.475
Delta	107.975	200.4	54.125	285.725
Rank	3	2	4	1

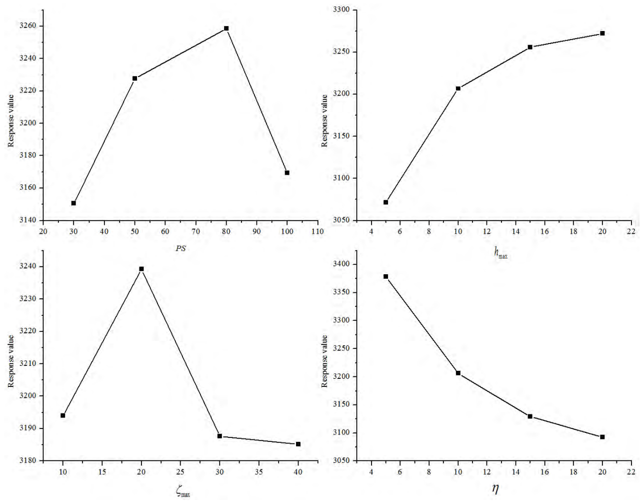

FIGURE 7. Factor level trend of parameters in DWWO.

DWWO. According to Figure 7, we set $PS = 30$, $h_{\max} = 5$, $\zeta_{\max} = 40$, and $\eta = 20$ for DWWO in the following simulation test.

C. RESULTS AND COMPARISON

To test the performance of the DWWO algorithm, we compare it with the modified genetic algorithm (MGA) [32], the improved whale optimization algorithm (IWOA) [51], and the grey wolf optimization algorithm with double-searching mode (DMGWO) [52]. In the compared algorithms, the initial solutions are also generated at random. The machine assignment is added to the MGA and IWOA algorithms. The machine assignment and speed level selection are added to the DMGWO algorithm. In the MGA and IWOA, the evolutionary operations performed on the machine assignment part are the same with those for the speed level selection part in the original algorithms. In the DMGWO, the TPX crossover operator is added to perform on machine assignment part and speed level selection part to implement the individual updating. In addition, the mutation operators in this paper are taken as the neighborhood structures of the local search strategy in the DMGWO.

To facilitate the comparison, the population size and the maximum iteration of the compared algorithms are same with those of the DWWO. In addition, the mutation rate and the crossover rate of MGA are 0.2 and 0.8, respectively; the

TABLE 4. **Comparison between different algorithms.**

Instance	$m \times n$	DWWO					MGA				
		Best	*Avg*	*ARPD*	*SD*	*Time(s)*	*Best*	*Avg*	*ARPD*	*SD*	*Time(s)*
RM01	10×20	1866.7	**1900.1**	**3.6**	**22.3**	11.5	1846.6	1965.4	7.2	91.0	2.5
RM02	10×30	2889.1	2930.1	6.1	**29.4**	28.3	**2762.9**	2925.9	5.9	128.6	6.6
RM03	10×50	4414.4	4510.1	4.7	**47.1**	96.7	4503.0	4872.8	13.1	202.1	25.0
RM04	10×70	5855.5	5908.6	3.6	**33.7**	235.5	7127.6	7328.7	28.5	145.1	66.4
RM05	10×80	6848.2	6946.4	3.7	65.5	324.2	7641.2	8444.4	26.0	365.4	98.6
RM06	10×100	8871.6	8953.1	4.6	**62.5**	595.7	10536.7	10990.3	28.4	256.7	188.9
RM07	10×120	11335.4	11464.8	6.3	**64.5**	965.1	13750.8	14031.3	30.1	196.1	317.9
RM08	10×150	13751.4	13915.0	5.2	**97.5**	1926.1	17486.8	17982.4	35.9	234.0	657.9
RM09	15×20	1366.5	**1391.9**	**6.3**	**15.1**	17.3	**1309.0**	1583.0	20.9	149.6	3.7
RM10	15×30	**2050.8**	**2089.7**	**1.9**	**19.8**	45.5	2201.0	2511.3	22.5	147.0	9.9
RM11	15×50	**2980.4**	**3018.2**	**1.3**	**22.4**	158.1	4099.2	4502.5	51.1	221.3	39.7
RM12	15×70	4525.8	4598.0	4.1	53.7	361.5	6168.1	6701.4	51.7	278.6	105.4
RM13	15×80	4981.0	5052.5	3.4	47.7	513.6	7949.1	8217.6	68.2	158.5	155.1
RM14	15×100	6691.4	6785.9	4.7	**68.9**	924.7	10975.8	11176.3	72.5	156.9	305.5
RM15	15×120	8123.1	8221.5	5.4	67.4	1511.7	12999.2	13370.3	71.5	280.7	512.4
RM16	15×150	9881.9	9968.6	5.5	69.6	2871.5	16957.5	17444.4	84.7	231.8	976.9
RM17	20×20	**1273.4**	**1287.0**	**1.1**	**12.0**	21.7	1429.1	1500.9	17.9	55.5	5.0
RM18	20×30	**1546.7**	**1564.0**	**1.1**	**11.4**	62.4	1780.3	2359.4	52.5	217.7	13.6
RM19	20×50	**2401.3**	**2432.5**	**1.3**	**17.7**	209.6	4138.8	4351.6	81.2	104.0	53.0
RM20	20×70	3588.9	**3637.8**	**1.4**	**41.2**	507.5	6656.9	6909.4	92.6	146.5	138.3
RM21	20×80	4011.2	4029.4	1.9	**18.9**	704.1	7498.2	7886.5	99.4	213.3	210.1
RM22	20×100	5538.1	5644.3	4.1	65.5	1287.8	10209.6	10460.0	92.8	189.3	398.9
RM23	20×120	6667.7	6765.3	3.9	47.4	2045.1	12665.4	13163.2	102.2	322.3	704.5
RM24	20×150	8084.3	8173.7	5.2	**54.2**	3891.3	16077.2	16713.9	115.1	280.1	1315.8
RM25	25×20	985.8	**995.7**	**1.1**	**7.0**	20.0	1098.0	1263.1	28.2	84.7	6.1
RM26	25×30	**1290.1**	**1300.7**	**0.8**	**7.7**	67.9	2016.6	2285.2	77.1	113.2	17.9
RM27	25×50	**2122.9**	**2145.0**	**1**	**20.6**	237.5	3855.1	4147.4	95.4	176.1	69.4
RM28	25×70	**3142.3**	**3191.7**	**1.6**	32.9	623.1	6568.0	6745.5	114.7	141.0	177.0
RM29	25×80	**3563.3**	**3611.6**	**1.1**	**19.3**	897.1	7297.7	7787.8	118.1	269.3	260.6
RM30	25×100	**4620.6**	**4653.7**	**0.7**	**24.4**	1618.4	9967.8	10446.6	126.1	264.5	489.7
RM31	25×120	**5451.4**	5575.8	2.3	86.4	2492.2	12358.1	12872.7	136.1	257.9	811.3
RM32	25×150	7052.6	7183.9	2	87.8	4640.2	16166.1	16575.0	135.3	356.0	1523.4
RM33	30×20	**859.4**	**873.3**	**1.6**	**9.3**	28.0	1072.6	1177.0	37.0	83.3	8.1
RM34	30×30	**1146.1**	**1163.7**	**1.5**	**14.6**	68.4	1942.7	2117.8	84.8	101.4	21.4
RM35	30×50	**1923.3**	**1945.4**	**1.1**	**13.7**	271.5	4058.4	4283.5	122.7	97.2	782.7
RM36	30×70	**2773.3**	**2810.1**	**1.3**	**23.7**	659.2	5950.5	6401.0	130.8	265.6	208.8
RM37	30×80	**3236.0**	**3292.5**	**1.7**	**31.6**	979.4	7554.2	7755.6	139.7	121.8	319.7
RM38	30×100	**4122.9**	**4157.1**	**0.8**	**30.0**	1856.7	9894.5	10274.1	149.2	210.4	589.7
RM39	30×120	5083.1	**5116.6**	**0.9**	**28.5**	3210.9	12471.7	12759.7	151.6	144.4	1002.9
RM40	30×150	**6240.6**	**6354.9**	**1.8**	85.9	5710.8	15986.1	16126.3	158.4	159.7	1899.0
RM41	35×20	842.7	855.5	3.5	**6.9**	23.9	1147.5	1269.2	53.6	76.4	11.0
RM42	35×30	**1090.9**	**1114.0**	**2.1**	**10.9**	59.2	1840.6	2010.8	84.3	86.6	26.9
RM43	35×50	**1839.6**	1868.2	1.6	**16.1**	264.9	3915.9	4134.3	124.7	150.1	104.4

TABLE 4. ***(Continued.)*** **Comparison between different algorithms.**

RM44	35×70	**2592**	**2633.2**	**1.6**	25.1	649.9	6221.7	6451.2	148.9	136.3	269.3
RM45	35×80	3028.7	**3052.0**	**2.2**	**19.3**	1003.9	7212.2	7690.2	157.5	272.9	413.3
RM46	35×100	**3871.5**	**3913.0**	**1.1**	**27.4**	1837.6	9646.3	9912.5	156.0	156.8	762.3
RM47	35×120	4630.2	4672.0	1.3	**37.5**	2933.7	11759.4	12185.1	164.3	323.1	1240.4
RM48	35×150	5783.1	5806.8	2.2	**21.7**	5812.9	15323.3	15788.4	177.9	254.2	2414.9
RM49	40×20	**730.5**	**741.0**	**1.4**	**8.6**	25.4	944.2	1071.8	46.7	70.3	11.2
RM50	40×30	1055.2	**1077.1**	**2.4**	**13.4**	68.3	1882.5	1987.4	89.0	52.6	31.2
RM51	40×50	1791.8	1808.1	1.7	**10.2**	290.5	3838.8	4137.7	132.8	121.7	126.1
RM52	40×70	**2597.3**	**2622.9**	**1.0**	**19.4**	796.9	6201.1	6560.4	152.6	183.4	322.9
RM53	40×80	2833.3	2879.5	2.0	30.7	1278.5	7357.5	7572.2	168.1	131.3	463.3
RM54	40×100	**3630.2**	**3682.3**	**1.4**	**32.5**	2284.9	9169.8	9933.4	173.6	346.3	898.0
RM55	40×120	4399.9	4430.3	1.1	**20.5**	3436.6	12128.4	12230.5	179.1	84.2	1455.8
RM56	40×150	**5381.8**	**5433.0**	**1.0**	33.7	6299.9	15287.2	15602.5	189.9	205.1	2759.6
Mean	-	4093.9	4145.5	2.5	**34.2**	1245.8	7516.2	7838.4	94.2	183.4	460.9

Instance	$m \times n$	IWOA					DMGWO				
		Best	*Avg*	*ARPD*	*SD*	*Time(s)*	*Best*	*Avg*	*ARPD*	*SD*	*Time(s)*
RM01	10×20	2823.0	3093.7	68.7	239.3	6.8	**1833.8**	1907.1	4.0	44.4	6.6
RM02	10×30	4600.8	4938.0	78.7	268.9	17.9	2776.5	**2848.2**	**3.1**	36.5	17.3
RM03	10×50	7702.3	8039.7	86.7	317.9	63.7	**4306.9**	**4406.0**	**2.3**	50.0	62.1
RM04	10×70	11503.1	11818.1	107.3	314.6	158.1	**5701.1**	**5783.4**	**1.4**	45.6	155.9
RM05	10×80	13289.1	13981.9	108.7	356.1	227.2	**6699.8**	**6806.7**	**1.6**	**60.7**	223.0
RM06	10×100	17472.2	18793.6	119.5	632.2	413.5	**8561.4**	**8689.0**	**1.5**	95.4	420.5
RM07	10×120	23984.9	24947.9	131.2	590.0	691.3	**10788.6**	**11090.8**	**2.8**	173.8	695.7
RM08	10×150	29502.5	30951.8	134.0	841.0	1292.1	**13229**	**13444.5**	**1.6**	209.0	1253.3
RM09	15×20	2350.2	2472.0	88.8	118.7	9.7	1396.6	1433.4	9.5	23.7	14.3
RM10	15×30	3610.8	3891.3	89.7	242.5	25.3	2062.2	2103.8	2.6	34.3	26.5
RM11	15×50	6027.3	6972.0	133.9	417.2	90.8	3010.9	3069.7	3.0	60.3	92.0
RM12	15×70	10251.1	10674.4	141.6	292.3	234.3	**4418.0**	**4523.0**	**2.4**	**45.1**	329.8
RM13	15×80	12607.2	13366.2	173.6	471.2	330.1	**4885.4**	**4932.9**	**1.0**	**45.1**	329.8
RM14	15×100	16732.1	17674.2	172.7	485.0	645.1	**6480.8**	**6606.1**	**1.9**	108.9	658.6
RM15	15×120	21293.7	22657.5	190.6	786.5	1034.4	**7797.4**	**7896.1**	**1.3**	**62.6**	1055.2
RM16	15×150	28686.5	29796.9	215.5	542.5	1953.2	**9445.0**	**9563.7**	**1.3**	**59.9**	1972.3
RM17	20×20	2172.5	2398.1	88.3	164.5	13.5	1281.6	1306.3	2.6	20.3	15.4
RM18	20×30	2982.6	3215.4	107.9	202.9	36.7	1554.4	1604.2	3.7	24.5	41.8
RM19	20×50	5440.0	5830.5	142.8	257.7	130.1	2441.0	2461.6	2.5	19.6	138.9
RM20	20×70	10135.0	10771.5	200.2	322.4	322.8	**3588.2**	3670.1	2.3	56.5	357.8
RM21	20×80	11432.4	12364.2	212.6	433.3	455.4	**3954.8**	**4015.1**	**1.5**	57.4	494.4
RM22	20×100	16356.7	17169.7	216.5	448.4	849.0	**5424.2**	**5498.2**	**1.4**	**35.2**	928.3
RM23	20×120	21002.2	21666.2	232.8	522.5	1400.7	**6510.4**	**6581.9**	**1.1**	**36.1**	1567.0
RM24	20×150	26683.3	28257.6	263.7	962.8	2609.3	**7769.2**	**7860.9**	**1.2**	59.4	2844.0
RM25	25×20	1632.4	1760.8	78.7	97.3	15.8	**985.2**	1019.3	3.5	19.7	20.3
RM26	25×30	2771.3	2958.4	129.3	157.4	43.1	1292.0	1334.3	3.4	26.0	55.1
RM27	25×50	5344.2	5794.2	172.9	267.1	149.0	2125.6	2205.8	3.9	36.4	182.7
RM28	25×70	8870.2	9663.1	207.5	497.2	372.7	3221.0	3264.2	3.9	**28.9**	446.4
RM29	25×80	11053.9	11907.0	233.5	488.2	540.4	3570.8	3661.8	2.5	43.8	638.6

TABLE 4. *(Continued.)* **Comparison between different algorithms.**

RM30	25×100	15238.8	16312.3	253.0	627.8	1027.2	4646.2	4694.2	1.6	49.6	1158.3
RM31	25×120	19923.1	20836.5	282.2	787.3	1632.4	5460.0	**5540.4**	**1.6**	**43.0**	1903.4
RM32	25×150	25018.0	26908.4	282.0	925.3	3092.5	**7044.0**	**7098.4**	**0.8**	**42.4**	3460.9
RM33	30×20	1343.7	1512.4	76.0	150.2	21.1	865.6	886.4	3.1	18.5	28.8
RM34	30×30	2223.5	2404.0	109.8	129.1	50.8	1152.0	1170.6	2.1	14.9	69.7
RM35	30×50	4971.3	5555.1	188.8	414.6	183.9	1934.0	1974.3	2.7	34.0	256.0
RM36	30×70	8543.3	9129.3	229.2	450.4	447.3	2803.0	2841.7	2.5	26.9	552.4
RM37	30×80	11231.7	11775.9	263.9	532.0	673.7	3286.7	3341.9	3.3	49.7	782.8
RM38	30×100	15176.5	16159.9	292.0	496.2	1290.7	4135.8	4185.1	1.5	32.0	1476.8
RM39	30×120	18667.2	19528.2	285.1	681.9	2090.8	**5071.0**	5125.8	1.1	34.6	2580.7
RM40	30×150	25837.3	26352.6	322.3	472.1	3811.3	6346.0	6387.9	2.4	**44.0**	4401.1
RM41	35×20	1347.8	1481.0	79.3	116.4	26.1	**826.2**	**845.6**	**2.3**	13.0	32.7
RM42	35×30	2115.7	2306.7	111.4	118.9	62.6	1097.3	1124.8	3.1	18.3	77.5
RM43	35×50	4680.1	5106.2	177.6	326.7	227.3	**1839.6**	**1856.4**	**0.9**	16.2	267.8
RM44	35×70	8137.2	8764.6	238.1	523.5	579.4	2629.5	2649.4	2.2	**15.4**	622.5
RM45	35×80	10765.1	11166.5	274.0	343.7	864.9	**2986.0**	3053.3	2.3	31.2	960.3
RM46	35×100	15056.1	15519.6	300.9	407.5	1761.6	3883.6	3925.8	1.4	35.2	1722.6
RM47	35×120	18638.1	19073.2	313.6	231.7	2633.3	**4611.0**	**4655.7**	**1.0**	42.1	2851.5
RM48	35×150	23685.0	25182.4	343.2	927.4	4807.8	**5682.0**	**5777.7**	**1.7**	51.5	5193.4
RM49	40×20	1191.6	1236.1	69.2	31.4	29.4	731.4	749.2	2.6	13.3	41.2
RM50	40×30	1946.9	2247.5	113.7	157.6	91.5	**1051.6**	1078.8	2.6	17.1	104.5
RM51	40×50	4526.5	5170.2	190.9	364.1	281.7	**1777.6**	**1797.9**	**1.1**	13.6	378.0
RM52	40×70	8035.2	9012.8	247.0	581.9	693.0	2598.6	2637.3	1.5	30.4	879.4
RM53	40×80	9485.6	10867.7	284.8	778.1	1001.7	**2824.0**	**2873.2**	**1.7**	**25.3**	1239.5
RM54	40×100	14769.3	15117.5	316.4	325.2	1835.5	3657.0	3716.1	2.4	34.4	2200.0
RM55	40×120	18353.3	18685.3	326.3	266.7	3033.7	**4382.8**	**4414.4**	**0.7**	24.2	3504.6
RM56	40×150	23978.4	24739.0	359.7	622.5	5942.4	5412.0	5446.7	1.2	**28.2**	6154.8
Mean	-	11664.3	12321.0	190.3	420.1	934.4	**4033.0**	**4097.1**	**2.3**	43.2	1034.7

TABLE 5. **ANOVA for ARPD of the compared algorithms.**

Source	DF	Sum of Squares	Mean Square	F	p-value
Factor	3	1.35469E6	451563.875 16	180.09989	0
Error	220	551605.305 89	2507.29684		
Total	223	1.9063E6			

maximum iteration of local search in DMGWO is equal to 5. The comparison results are listed in Table 4. In the table, '*ARPD*' represents the average relative percent deviation, i.e., $ARPD = \sum_{l=1}^{L} \frac{100\times(Alg_l - Min)}{Min}/L$, '*L*' is the number of runs, '*Min*' is the minimum result among all the conducted experiments, Alg_l is the obtained value in the lth run by an algorithm. '*SD*' is the standard deviation of computational results obtained in the ten runs. '*Time*' represents the average run time (in seconds). '*Mean*' represents the average values of the data in each column. The boldface represents the better solutions obtained by the compared algorithms.

Seen from Table 4, it can be summarized that: (1) For the '*Best*' value, DMGWO obtains 29 boldface values. Our DWWO yields 26 boldface values, which is the second best algorithm. According to the last row, the proposed DWWO algorithm obtains an average value of 4093.9, which is better than those of MGA and IWOA. (2) For the '*Avg*' value, DWWO gets 31 boldface values out of 56 instances, which performs better than other three algorithms. The second best algorithm, namely DMGWO, can obtain 26 boldface values. According to the last row, the proposed DWWO algorithm can obtain an average value of 4145.5, which is better than those of MGA and IWOA. (3) For the '*ARPD*' value, DWWO obtains 31 boldface values, which is better than other algorithms. The second best algorithm, namely DMGWO, can obtain 26 boldface values. According to the

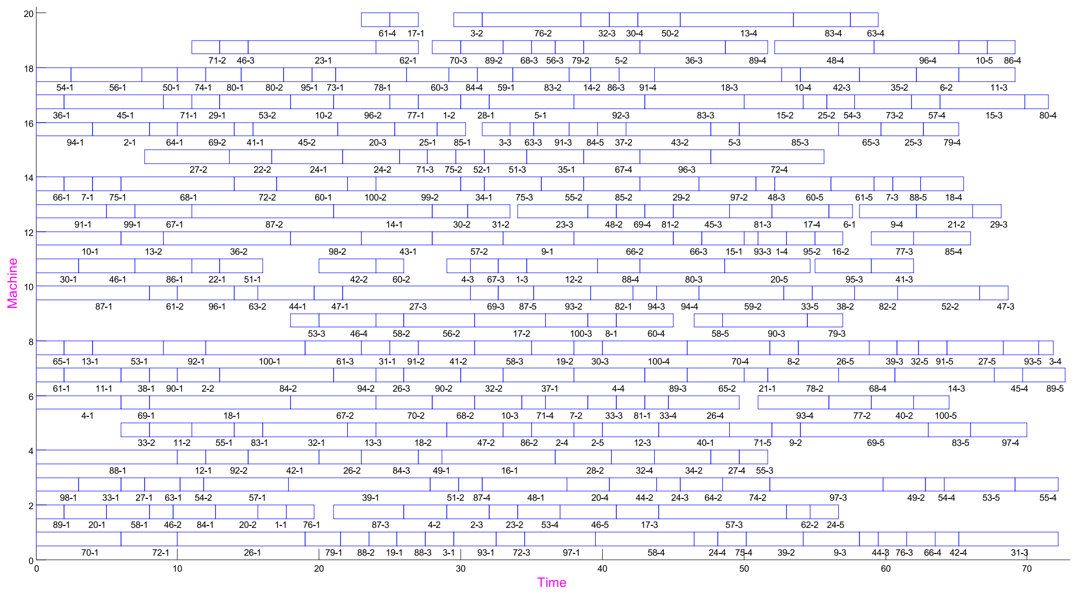

FIGURE 8. Gantt chart for instance RM22.

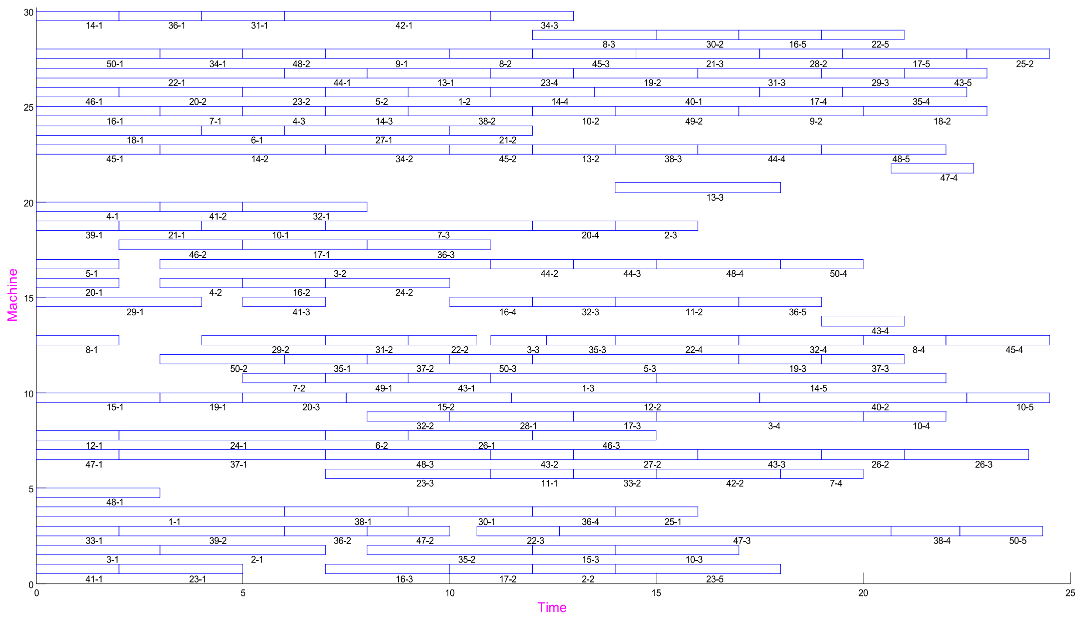

FIGURE 9. Gantt chart for instance RM43.

last row, the proposed DWWO algorithm yields an average value of 2.5, which is better than those of MGA and IWOA. (4) For the '*SD*' value, DWWO yields 42 boldface values. The second best algorithm, namely DMGWO, can obtain 14 boldface values. According to the last row, the proposed DWWO algorithm obtains an average value of 34.2, which is better than those of other three algorithms. It demonstrates the strong robustness of our proposed algorithm. (5) For the '*Time*' value, DWWO spends more time to run. However, our algorithm effectively improves the quality of the solution when compared with other algorithms.

Figures 8∼10 show the Gantt charts of the instances RM22, RM43 and RM53 obtained by our DWWO algorithm.

The analysis of variance (ANOVA) is conducted in Table 5 according to the data in Table 4. The algorithms are taken as levels, and ARPD is regarded as the response variable. In the table, the results of statistical analysis show that there are significant differences among the algorithms due to the fact that p-value is equal to zero.

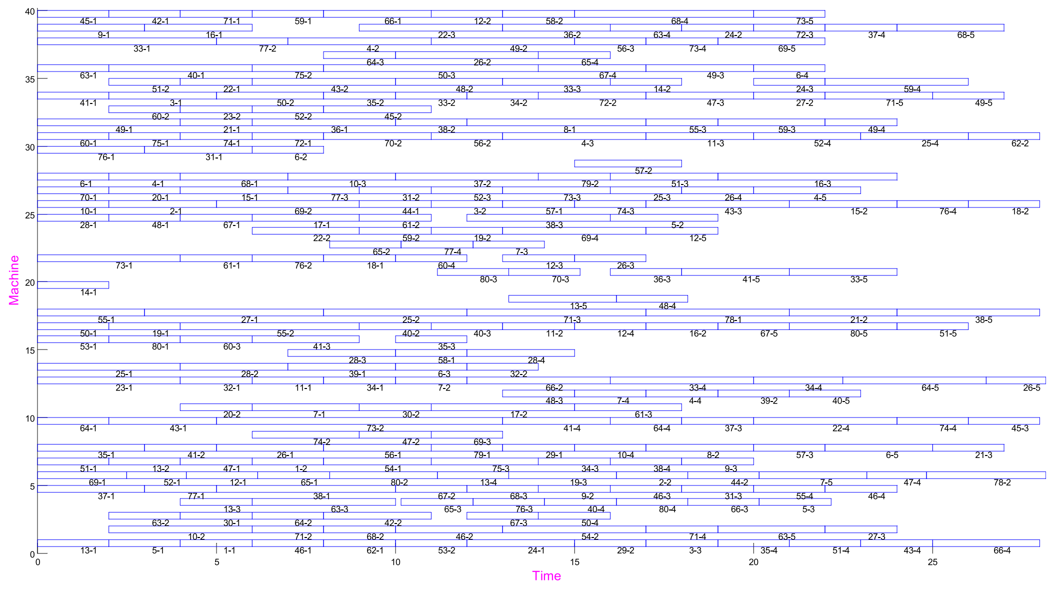

FIGURE 10. Gantt chart for instance RM53.

VI. CONCLUSION

In this paper, a mathematical model of the energy-conscious flexible job shop scheduling problem, which considers the machine speed selection and the energy consumption reduction simultaneously, was built. With the promotion of sustainable development, the energy-conscious FJSP is more suitable for practical production compared with the traditional FJSP.

A new discrete water wave optimization (DWWO) algorithm was proposed to solve the energy-conscious FJSP. According to the characteristics of the problem, three operators (propagation, breaking and refraction) were redesigned to make the algorithm directly work in a discrete domain. In the experimental section, 56 instances (RM01-RM56) were designed to evaluate the performance of the proposed DWWO algorithm. Based on the comparison results, the proposed algorithm has advantages in solving the energy-conscious FJSP.

In the future, we will study the energy-conscious FJSP in greater depth. Some constraints will be taken into consideration as follows: (1) In real-life production, workers can only operate a fraction of the machines with different skill levels. Therefore, the dual-resource constrained energy-conscious FJSP with worker flexibility will be studied in the future work; (2) Time-varying electricity price will be considered to estimate the total energy-consumption costs, such as time-of-use pricing policy; (3) Renewable energy can reduce non-renewable energy consumption in the production process. The energy-conscious FJSP with renewable energy will be a new research direction.

REFERENCES

[1] T.-H. Jiang and C. Zhang, "Adaptive discrete cat swarm optimisation algorithm for the flexible job shop problem," *Int. J. Bio-Inspired Comput.*, vol. 13, no. 3, pp. 199–208, Apr. 2019.

[2] G. Moslehi and M. Mahnam, "A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search," *Int. J. Prod. Econ.*, vol. 129, no. 1, pp. 14–22, Jan. 2011.

[3] K.-Z. Gao, P. N. Suganthan, Q.-K. Pan, T. J. Chua, T. X. Cai, and C.-S. Chong, "Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling," *Inf. Sci.*, vol. 289, pp. 76–90, Dec. 2014.

[4] H.-C. Chang, Y.-P. Chen, T.-K. Liu, and J.-H. Chou "Solving the flexible job shop scheduling problem with makespan optimization by using a hybrid Taguchi-genetic algorithm," *IEEE Access*, vol. 3, pp. 1740–1754, 2015.

[5] K. Z. Gao, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time," *Int. J. Prod. Res.*, vol. 53, no. 19, pp. 5896–5911, 2015.

[6] Y. Yuan and H. Xu, "Multiobjective flexible job shop scheduling using memetic algorithms," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 336–353, Jan. 2015.

[7] K. Z. Gao, P. N. Suganthan, T. J. Chua, C. S. Chong, T. X. Cai, and Q. K. Pan, "A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 7652–7663, Nov. 2015.

[8] K.-Z. Gao, P. N. Suganthan, Q. K. Pan, T. J. Chua, T. X. Cai, and C. S. Chong, "Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives," *J. Intell. Manuf.*, vol. 27, no. 2, pp. 363–374, 2016.

[9] K. Z. Gao, P. N. Suganthan, Q. K. Pan, M. F. Tasgetiren, and A. Sadollah, "Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion," *Knowl.-Based Syst.*, vol. 109, pp. 1–16, Oct. 2016.

[10] K. Z. Gao, P. N. Suganthan, Q. K. Pan, T. J. Chua, C. S. Chong, and T. X. Cai, "An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time," *Expert Syst. Appl.*, vol. 65, pp. 52–67, Dec. 2016.

[11] X. Wu and S. Wu, "An elitist quantum-inspired evolutionary algorithm for the flexible job-shop scheduling problem," *J. Intell. Manuf.*, vol. 28, no. 6, pp. 1441–1457, Aug. 2017.

[12] T. Jiang and C. Zhang, "Application of grey wolf optimization for solving combinatorial problems: Job shop and flexible job shop scheduling cases," *IEEE Access*, vol. 6, pp. 26231–26240, 2018.

[13] L. Shen, S. Dauzère-Pérès, and J. S. Neufeld, "Solving the flexible job shop scheduling problem with sequence-dependent setup times," *Eur. J. Oper. Res.*, vol. 265, no. 2, pp. 503–516, Mar. 2018.

[14] K. Gao, F. Yang, M. Zhou, Q. Pan, and P. N. Suganthan, "Flexible job-shop rescheduling for new job insertion by using discrete Jaya algorithm," *IEEE Trans. Cybern.*, vol. 49, no. 5, pp. 1944–1955, May 2019.

[15] K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan, "A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 4, pp. 904–916, Jul. 2019.

[16] T. H. Jiang, C. Zhang, and Q.-M. Sun, "Green job shop scheduling problem with discrete whale optimization algorithm," *IEEE Access*, vol. 7, pp. 43153–43166, 2019.

[17] F. Shrouf, J. Ordieres-Meré, and A. García-Sánchez, "Optimizing the production scheduling of a single machine to minimize total energy consumption costs," *J. Cleaner Prod.*, vol. 67, pp. 197–207, Mar. 2014.

[18] M. B. Yildirim and G. Mouzon, "Single-machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm," *IEEE Trans. Eng. Manag.*, vol. 59, no. 4, pp. 585–597, Nov. 2012.

[19] A. Che, Y. Zeng, and K. Lyu, "An efficient greedy insertion heuristic for energy-conscious single machine scheduling problem under time-of-use electricity tariffs," *J. Cleaner Prod.*, vol. 129, pp. 565–577, Aug. 2016.

[20] G. Mouzon and M. B. Yildirim, "A framework to minimise total energy consumption and total tardiness on a single machine," *Int. J. Sustain. Eng.*, vol. 1, no. 2, pp. 105–116, 2008.

[21] C. Lu, L. Gao, X. Li, Q. Pan, and Q. Wang, "Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm," *J. Cleaner Prod.*, vol. 144, pp. 228–238, Feb. 2017.

[22] D. Tang, M. Dai, M. A. Salido, and A. Giret, "Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization," *Comput. Ind.*, vol. 81, pp. 82–95, Sep. 2016.

[23] G.-S. Liu, Y. Zhou, and H.-D. Yang, "Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time," *J. Cleaner Prod.*, vol. 147, pp. 470–484, Mar. 2017.

[24] J. Yan, L. Li, F. Zhao, F. Zhang, and Q. Zhao, "A multi-level optimization approach for energy-efficient flexible flow shop scheduling," *J. Cleaner Prod.*, vol. 137, pp. 1543–1552, Nov. 2016.

[25] S. A. Mansouri, E. Aktas, and U. Besikci, "Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption," *Eur. J. Oper. Res.*, vol. 248, no. 3, pp. 772–788, 2016.

[26] H. Zhang, F. Zhao, K. Fang, and J. W. Sutherland, "Energy-conscious flow shop scheduling under time-of-use electricity tariffs," *CIRP Ann.*, vol. 63, no. 1, pp. 37–40, Apr. 2014.

[27] J.-Y. Ding, S. Song, and C. Wu, "Carbon-efficient scheduling of flow shops by multi-objective optimization," *Eur. J. Oper. Res.*, vol. 248, no. 3, pp. 758–771, 2016.

[28] L. Meng, C. Zhang, X. Shao, Y. Ren, and C. Ren, "Mathematical modelling and optimisation of energy-conscious hybrid flow shop scheduling problem with unrelated parallel machines," *Int. J. Prod. Res.*, vol. 57, no. 4, pp. 1119–1145, Jul. 2019.

[29] R. Zhang and R. Chiong, "Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption," *J. Cleaner Prod.*, vol. 112, pp. 3361–3375, Jan. 2016.

[30] Y. Liu, H. B. Dong, N. Lohse, S. Petrovic, and N. Gindy, "An investigation into minimising total energy consumption and total weighted tardiness in job shops," *J. Cleaner Prod.*, vol. 65, pp. 87–96, Feb. 2014.

[31] G. May, B. Stahl, M. Taisch, and V. Prabhu, "Multi-objective genetic algorithm for energy-efficient job shop scheduling," *Int. J. Prod. Res.*, vol. 53, no. 23, pp. 7071–7089, Jan. 2015.

[32] M. A. Salido, J. Escamilla, A. Giret, and F. Barber, "A genetic algorithm for energy-efficiency in job-shop scheduling," *Int. J. Adv. Manuf. Technol.*, vol. 85, nos. 5–8, pp. 1303–1314, 2016.

[33] D. Tang and M. Dai, "Energy-efficient approach to minimizing the energy consumption in an extended job-shop scheduling problem," *Chin. J. Mech. Eng.*, vol. 28, no. 5, pp. 1048–1055, Sep. 2015.

[34] J. Escamilla, M. A. Salido, A. Giret, and F. Barber, "A metaheuristic technique for energy-efficiency in job-shop scheduling," *Knowl. Eng. Rev.*, vol. 31, pp. 475–485, Nov. 2016.

[35] T. Jiang and G. Deng, "Optimizing the low-carbon flexible job shop scheduling problem considering energy consumption," *IEEE Access*, vol. 6, pp. 46346–46355, 2018.

[36] Z. Q. Jiang, L. Zuo, and E. Mingcheng, "Study on multi-objective flexible job-shop scheduling problem considering energy consumption," *J. Ind. Eng. Manage.*, vol. 7, no. 3, pp. 589–604, May 2014.

[37] H. Mokhtari and A. Hasani, "An energy-efficient multi-objective optimization for flexible job-shop scheduling problem," *Comput. Chem. Eng.*, vol. 104, pp. 339–352, Sep. 2017.

[38] H. Piroozfard, K. Y. Wong, and W. P. Wong, "Minimizing total carbon footprint and total late work criterion in flexible job shop scheduling by using an improved multi-objective genetic algorithm," *Resour., Conservation Recycling*, vol. 128, pp. 267–283, Jan. 2018.

[39] L. Meng, C. Zhang, X. Shao, and Y. Ren, "MILP models for energy-aware flexible job shop scheduling problem," *J. Cleaner Prod.*, vol. 210, pp. 710–723, Feb. 2019.

[40] L. Meng, C. Zhang, B. Zhang, and Y. Ren, "Mathematical modeling and optimization of energy-conscious flexible job shop scheduling problem with worker flexibility," *IEEE Access*, vol. 7, pp. 68043–68059, 2019.

[41] D. Lei, Y. Zheng, and X. Guo, "A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3126–3140, 2017.

[42] X. Wu and Y. Sun, "A green scheduling algorithm for flexible job shop with energy-saving measures," *J. Cleaner Prod.*, vol. 172, nos. 3249–3264, Jan. 2018.

[43] Y.-J. Zheng, "Water wave optimization: A new nature-inspired metaheuristic," *Comput. Oper. Res.*, vol. 55, no. 1, pp. 1–11, Mar. 2015.

[44] F. Zhao, H. Liu, Y. Zhang, W. Ma, and C. Zhang, "A discrete Water Wave Optimization algorithm for no-wait flow shop scheduling problem," *Expert Syst. Appl.*, vol. 91, pp. 347–363, Jan. 2018.

[45] X.-B. Wu, J. Liao, and Z.-C. Wang, "Water wave optimization for the traveling salesman problem," in *Proc. Int. Conf. Intell. Comput.* Cham, Switzerland: Springer, 2015, pp. 137–146.

[46] X. Wu, Y. Zhou, and Y. Lu, "Elite opposition-based water wave optimization algorithm for global optimization," *Math. Problems Eng.*, vol. 2017, Jan. 2017, Art. no. 3498363.

[47] J. Zhang, Y. Zhou, and Q. Luo, "An improved sine cosine water wave optimization algorithm for global optimization," *J. Intell. Fuzzy Syst.*, vol. 34, no. 4, pp. 2129–2141, Apr. 2018.

[48] Y. Zhou, J. Zhang, X. Yang, and Y. Ling, "Optimal reactive power dispatch using water wave optimization algorithm," *Oper. Res.*, 2018. doi: 10.1007/s12351-018-0420-3.

[49] Z. Shao, D. Pi, and W. Shao, "A novel discrete water wave optimization algorithm for blocking flow-shop scheduling problem with sequence-dependent setup times," *Swarm Evol. Comput.*, vol. 40, pp. 53–75, Jun. 2018.

[50] S.-C. Chu and P.-W. Tsai, "Computational intelligence based on the behavior of cats," *Int. J. Innov. Comput., Inf. Control*, vol. 3, no. 1, pp. 163–173, 2007.

[51] T. Jiang, C. Zhang, H. Q. Zhu, J. Gu, and G. Deng, "Energy-efficient scheduling for a job shop using an improved whale optimization algorithm," *Mathematics*, vol. 6, no. 11, pp. 220–239, Oct. 2018.

[52] T. Jiang, C. Zhang, H. Zhu, and G. Deng, "Energy-efficient scheduling for a job shop using grey wolf optimization algorithm with double-searching mode," *Math. Problems Eng.*, vol. 2018, Oct. 2018, Art. no. 8574892.

YI LU was born in Xinxiang, China, in 1987. She received the B.A. degree in ecological planning from Hainan University, Haikou, China, in 2009, and the M.T.A. degree in tourism management from Guangxi Normal University, Guilin, China, in 2018.

From 2010 to 2016, she was a Teaching Assistant with the Henan Institute of Technology, Xinxiang, where she has been a Lecturer, since 2016. She has authored two books and more than 10 articles. Her research interests include ecotourism, production scheduling, and intelligence algorithm.

JIACHENG LU was born in Jiaozuo, China, in 1962. He received the B.S. degree in semiconductor device from the Hefei University of Technology, Hefei, China, in 1983.

From November 1992 to October 2000, he was an Engineer with the Henan Institute of Technology, Xinxiang, China, where he has been a Senior Engineer, since 2000. He has authored four books and more than 20 articles. His research interests include industrial automation and intelligence algorithm.

TIANHUA JIANG was born in Weihai, China, in 1983. He received the B.S. degree in automation from Jinan University, Jinan, China, in 2007, the M.S. degree in control science and engineering from the Sichuan University of Science and Engineering, Zigong, China, in 2010, and the Ph.D. degree from the MOE Key Laboratory of Measurement and Control of Complex Systems of Engineering and the School of Automation, Southeast University, China, in 2015.

Since 2015, he has been a Lecturer with the School of Transportation, Ludong University, Yantai, China. His research interests include production scheduling and intelligent algorithm. His recent publications have appeared in some peer-reviewed journals, such as the *International Journal of Bio-Inspired Computation*, IEEE Access, *Journal of Intelligent and Fuzzy Systems*, *International Journal of Industrial Engineering: Theory, Applications and Practice*, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *Mathematics*, *Mathematical Problems in Engineering*, and so on.