Received March 2, 2019, accepted March 24, 2019, date of publication March 29, 2019, date of current version April 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2908200

Green Job Shop Scheduling Problem With Discrete Whale Optimization Algorithm

TIANHUA JIANG[1], CHAO ZHANG[2], AND QI-MING SUN[3]
[1]School of Transportation, Ludong University, Yantai 264025, China
[2]Department of Computer Science and Technology, Henan Institute of Technology, Xinxiang 453003, China
[3]College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China

Corresponding author: Tianhua Jiang (jth1127@163.com)

This work was supported in part by the Shandong Provincial Natural Science Foundation of China under Grant ZR2016GP02 and Grant ZR2017PEE025, in part by the Project of Shandong Province Higher Educational Science and Technology Program under Grant J17KA199, in part by the Special Research and Promotion Program of Henan Province under Grant 182102210257, and in part by the Project of Henan Province Higher Educational Key Research Program under Grant 16A120011.

ABSTRACT In the last few decades, production scheduling problems have been studied for optimizing production efficiency involving the time-related indicators, such as completion time, earliness/tardiness time, or flow time. Currently, with the consideration of sustainable development, the green scheduling problem has been paid more and more attention. Here, a green job shop scheduling problem is considered to minimize the sum of energy-consumption cost and completion-time cost in the workshop. In this paper, a mathematical model is first established with the consideration of multi-speed machines. A discrete whale optimization algorithm (DWOA) is then proposed for solving the model. In the proposed algorithm, a two-string encoding is adopted to represent the two sub-problems: job permutation and speed selection. Then, a heuristic method is used to initialize the population to enhance the quality of initial solutions. By considering the discrete characteristics of the problem, the individual updating operators are redesigned to ensure the algorithm work directly in a discrete scheduling domain. In addition, a variable neighborhood search strategy is embedded to further improve the search ability. The extensive experiments have been performed to test the DWOA. The computational data reveal the promising advantages of the DWOA on the considered problem.

INDEX TERMS Job shop scheduling, multi-speed machine, energy consumption, discrete whale optimization algorithm.

I. INTRODUCTION

In recent years, there has been a growing concern over the sustainable development of the world economy. According to the statistical data, the world's demand for energy has doubled over the last 40 years and will be double once again by 2030 [1]. As the intensive energy consumer, manufacturing industry is responsible for 33% of the global energy consumption. Many manufacturing enterprises are forced to adopt effective energy-saving measures under the current environmental pressure. Production scheduling has been proved to be an effective approach to control the amount of energy consumption, which is easy to be applied to the existing production systems with a modest investment. However, in the last few decades, the traditional production scheduling has only emphasized the production efficiency [2], [3]. Therefore, it is very necessary to introduce the environmental metrics into the green production scheduling problem both for economic and environmental reasons. Some previous studies can be summarized as follows:

(1) Single-machine scheduling problem. Mouzon *et al.* [4] proposed some dispatching rules to control the machine so that the energy consumption can be effectively reduced. In addition, a multi-objective mathematical model was built with the objective of minimizing the energy consumption and total completion time. Subsequently, their work was extended to a single-machine scheduling problem [5], where a greedy randomized multi-objective adaptive search algorithm was developed to optimize total energy consumption and total tardiness. Yildirim and Mouzon [6] proposed a genetic algorithm to solve a single-machine scheduling to optimize the energy consumption and completion time. Shrouf *et al.* [7]

The associate editor coordinating the review of this manuscript and approving it for publication was Najah Abuali.

presented a genetic algorithm to solve a single-machine scheduling with variable energy prices.

(2) Parallel machine scheduling problem. Ding *et al.* [8] considered an unrelated parallel machine scheduling problem under the time of use scheme with the objective of minimizing the total electricity cost. Che *et al.* [9] established an improved continuous-time mixed-integer linear programming model of the unrelated parallel machine scheduling problem under time-of-use scheme. To solve the problem, a two-stage heuristic algorithm was proposed to optimize the total electricity cost. Zheng and Wang [10] investigated the resource constrained unrelated parallel machine green scheduling problem aiming at minimizing the makespan and the total carbon emission. A collaborative multiobjective fruit fly optimization algorithm was proposed to deal with the problem.

(3) Flow shop scheduling problem. Fang *et al.* [11] established a mathematical model of the flow shop scheduling problem considering the peak power load, energy consumption, carbon footprint and cycle time. Liu *et al.* [12] presented a branch-and-bound algorithm to optimize the wasted energy consumption in a permutation flow shop. Dai *et al.* [13] built an energy-efficient scheduling model in a flexible flow shop and proposed a genetic-simulated annealing algorithm to make a trade-off between makespan and energy consumption. Ding *et al.* [14] addressed a carbon-efficient scheduling problem in a permutation flow shop with the criterion to minimize the total carbon emissions and makespan. Mansouri *et al.* [15] developed multi-objective genetic algorithms to make a trade-off between energy consumption and makespan in a two-machine flow shop. Wang *et al.* [16] investigated a blocking flow shop scheduling problem to optimize makespan and energy consumption. In order to solve the problem, a multi-objective parallel variable neighborhood search algorithm was proposed according to the characteristics of the problem.

According to the above reviewed literature, previous works about the green scheduling problem concentrate on the sample manufacturing systems, i.e., single-machine, parallel-machine and flow shop. In real life, many problems can be treated as a job-shop scheduling problem. The green job shop scheduling problem is more close to the actual production. However, the green job shop scheduling problem is not fully studied [17], [18]. Therefore, in this work, the manufacturing system of a job shop type is selected to study the green scheduling problem.

In the previous researches, Dai *et al.* [13] studied the on/off control framework in a flexible flow shop to optimize the energy consumption. For some actual manufacturing systems, it is unable to turn off machines completely during the idle periods, due that a considerable amount of additional energy will be consumed when restarting machines. An alternative to the on/off control framework is a new method on the basis of machine speed scaling. If machine works at different speeds, the processing time and the energy consumption are also different. In this case, the scheduling problem possesses important significance both in theories and practical applications. In comparison with the standard JSP, the complexity of the considered problem lies on the addition of energy-related consideration and speed selection for machines. It is clear that the problem is hard to be solved by exact methods, and intelligence algorithms may obtain acceptable solutions in a reasonable time. However, the application of intelligence algorithms on the green JSP with multi-speed machine appears to be limited. Salido *et al.* [19] designed a multi-objective genetic algorithm to get the trade-off between the makespan and the energy consumption. Zhang and Chiong [20] developed a multi-objective genetic algorithm to minimize the total energy consumption and total weighted tardiness.

With the continuous exploration of the real world, more and more intelligence algorithms are developed for solving various optimization problems. Whale Optimization Algorithm (WOA) is a new nature-inspired algorithm by considering the hunting behavior of humpback whales [21]. It has been proved that WOA can yield competitive results when compared with some famous algorithms, such as PSO and GA. The main advantage of the WOA algorithm is that it can maintain a good relationship between exploration and exploitation during the iteration process. At present, WOA has been applied to different optimization problems in various fields [22]–[26]. In this study, we try to develop an effective algorithm for solving the green job shop scheduling problem with multi-speed machine. As WOA is originally developed for the continuous problems, the evolutionary process is conducted on the basis of the continuous updating of individual positions. Therefore, it cannot be directly used to solve the discrete production scheduling problem. Here, according to the characteristics of the considered problem, we propose a discrete whale optimization algorithm (DWOA), where some modified discrete updating approaches are developed to make the algorithm suitable for the discrete scheduling problem. Comprehensive experiments demonstrate that our proposed algorithm is effective for the problem at hand.

The rest of this paper is organized as follows: Section II introduces the description of the problem. Section III addresses the original whale optimization algorithm. Section IV describes the implementation of our proposed DWOA algorithm. Section V shows the experimental results of DWOA and Section VI provides conclusions and future works.

II. PROBLEM DESCRIPTION

The green job shop scheduling problem with multi-speed machine can be described as follows:

(1) There exists n jobs and m machines in the job shop, where each job consists of m operations to be processed. Here, each machine can work at adjustable speed levels represented by $\boldsymbol{v} = \{v_1, v_2, \ldots, v_d\}$.

(2) It is assumed that there is a basic processing time q_{ik}, when job i is processed on machine k. If the speed v_d is selected for job i on machine k, the processing time can be

defined as $P_{ikd} = q_{ik}/v_d$, and the energy consumption cost per unit time can be measured by E_{kD}. If $v_{d'} > v_d$, then $E_{kd'} \times p_{ikd'} > E_{kd} \times p_{ikd}$ is met. That is to say, if a machine works at a higher speed, the processing time will decrease but the energy consumption cost will increase.

(3) The considered problem is consisted of two sub-problems: operation permutation and speed selection. In the problem, some additional assumptions are involved as follows: (i) No jobs can be simultaneously processed on more than one machine; (ii) A machine can only process one operation at a time; (iii) No preemption is permitted once a job is started; (iv) Machine setup and breakdown are not considered here, (v) The speed of each machine is not allowed to be adjusted during the processing of an operation, (vi) Each machine will not be stopped until all jobs assigned to it are finished. During the waiting times for jobs, the machine will be on a stand-by mode with energy consumption cost per unit time SE_k.

To formulate the problem, the mathematical model of the problem under study is established in this section. The optimization objective is aiming to obtain an optimal scheduling scheme to minimize the total sum of energy-consumption cost and completion-time cost. For the model, some symbols and variables are shown as follows:

F : the optimization function;
O'_{ik} : the operation of job i processing on machine k;
C_{ik} : the completion time of O'_{ik};
S_{ik} : the start time of O'_{ik};
C_k : the final completion time of machine k;
W_k : the total workload of machine k;
C_{max} : the makespan of the workshop;
η : the completion time related cost per unit time;
P_{ikd} : the processing time of job i on machine k with speed d;
D_{ik} : the number of adjustable speed levels of machine k;
Q : a big positive constant;
P_{ik} : the predecessor operation of O'_{ik} in job i;
$C_{P_{ik}}$: the completion time of P_{ik};
x_{ikd}: 0-1 variable, if job i is processed on machine k with speed d, $x_{ikd} = 1$, otherwise, $x_{ikd} = 0$;
z_{ilk}: 0-1 variable, if job i is processed on machine k prior to job l, $z_{ilk} = 1$; otherwise, $z_{ilk} = 0$.

$$\min F = \sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{d=1}^{D_k} E_{kd} p_{ikd} x_{ikd} + \sum_{k=1}^{m} SE_k\,(C_k - W_k) + \eta C_{\max} \tag{1}$$

$$s.t \sum_{d=1}^{D_k} x_{ikd} = 1, i = 1, 2, \ldots, n; k = 1, 2, \cdots, m \tag{2}$$

$$C_{ik} - \sum_{d=1}^{D_k} p_{ikd} x_{ikd} \geq C_{P_{ik}}, \quad i = 1, 2, \ldots, n; k = 1, 2, \ldots, m \tag{3}$$

$$C_{lk} - C_{ik} + Q(1 - z_{ilk}) \geq \sum_{d=1}^{D_k} p_{lkd} x_{lkd}, \quad i, l = 1, 2, \ldots, n;\ k = 1, 2, \ldots, m \tag{4}$$

$$W_k = \sum_{i=1}^{n}\sum_{d=1}^{D_k} p_{ikd} x_{ikd}, \quad k = 1, 2, \cdots, m \tag{5}$$

$$C_k = \max\{C_{ik}\}, \quad i = 1, 2, \ldots, n \tag{6}$$

$$C_{ik} \geq 0, i = 1, 2, \ldots, n; k = 1, 2, \ldots, m \tag{7}$$

$$x_{ikd} = \begin{cases} 0, & \text{if job } i \text{ is processed on machine } k \\ & \text{with speed } d \\ 1, & \text{otherwise } i = 1, 2, \ldots, n; \\ & k = 1, 2, \ldots, m; d = 1, 2, \ldots, D_k \end{cases} \tag{8}$$

$$z_{ilk} = \begin{cases} 0, & \text{if job } i \text{ is processed on machine } k \\ & \text{prior to job } l \\ 1, & \text{otherwise } i, l = 1, 2, \ldots, n; \\ & k = 1, 2, \ldots, m \end{cases} \tag{9}$$

Equation (1) represents the optimization objective; constraint (2) ensures that the speed of a machine cannot be changed during the processing of an operation; constraint (3) shows the processing precedence constraints between operations of a job; constraint (4) means that each machine can only process one operation at a time; constraint (5) shows the workload of each machine; constraint (6) gives the final completion time of each machine; constraint (7) addresses the nonnegative feature of completion time; constraints (8) and (9) show the logic constraints.

III. THE ORIGINAL WOA

Whale optimization algorithm (WOA) is inspired from the bubble-net hunting behavior of humpback whales in nature. The whales hunt the prey near the water surface by swimming around them and creating bubbles in a circle shape. There are two searching phases in the algorithm for exploitation and exploration. In the first phase, based on the best solution found so far, bubble-net attacking, including encircling the prey and the spiral shape path, is used to represent the exploitation of the algorithm. In the second phase, to implement the exploration, the position of each whale is updated according to a randomly selected search agent.

A. EXPLOITATION PHASE

1) ENCIRCLING THE PREY

In the hunting process, whales first observe the position of the prey and encircle them. It is assumed that the current best whale is close to the optimal solution. The other whales update their positions according to the best whale. The behavior can be defined by the following equations:

$$\boldsymbol{D} = \left|\boldsymbol{C} \cdot \boldsymbol{X}_p(t) - \boldsymbol{X}(t)\right| \tag{10}$$

$$\boldsymbol{X}(t+1) = \boldsymbol{X}_p(t) - \boldsymbol{A} \cdot \boldsymbol{D} \tag{11}$$

$$\boldsymbol{A} = 2\boldsymbol{a}\boldsymbol{r} - \boldsymbol{a} \tag{12}$$

$$\boldsymbol{C} = 2\boldsymbol{r} \tag{13}$$

t represents the current iteration, $\boldsymbol{X}_p$ denotes the position vector of the current best solution, and $\boldsymbol{X}$ indicates the position vector of an individual whale. $\boldsymbol{A}$ and $\boldsymbol{C}$ represent the coefficient vectors. || is the absolute value, and $\cdot$ is an element-by-element multiplication. $\boldsymbol{r}$ is a random vector in the range [0,1]. The elements in $\boldsymbol{a}$ are linearly decreased from 2 to 0 over the course of iterations.

2) SPIRAL UPDATING MECHANISM

Under this mechanism, the distance from the whale and the prey is first obtained. A spiral path is then created to mimic the movement of humpback whales with a helix shape, which can be formulated as follows:

$$\boldsymbol{X}(t+1) = \boldsymbol{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \boldsymbol{X}_p(t) \tag{14}$$

$$\boldsymbol{D}' = \left|\boldsymbol{X}_p(t) - \boldsymbol{X}(t)\right| \tag{15}$$

where b represents a constant value for determining the shape of the logarithmic spiral, l means a random number inside [−1,1].

In the exploitation phase, there is a probability of 50% to select the shrinking encircling or the spiral movement path in Equation (16), where p' is a random number inside [0,1].

$$\boldsymbol{X}(t+1) = \begin{cases} \boldsymbol{X}_p(t) - \boldsymbol{A} \cdot \boldsymbol{D} & \text{if } p' < 0.5 \\ \boldsymbol{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \boldsymbol{X}_p(t) & \text{if } p' \geq 0.5 \end{cases} \tag{16}$$

B. EXPLORATION PHASE

To formulate the random search for the prey, $\boldsymbol{A}$ is used with random values greater than 1 or less than −1. When $|\boldsymbol{A}| \leq 1$, the exploitation is implemented by updating the positions towards the current best search agent in Section 3.1, when $|\boldsymbol{A}| > 1$, the exploration is used to search the global optimum by randomly selecting a position vector $\boldsymbol{X}_{\text{rand}}$ from the current position. The model can be defined as follows:

$$\boldsymbol{D} = |\boldsymbol{C} \cdot \boldsymbol{X}_{\text{rand}}(t) - \boldsymbol{X}(t)| \tag{17}$$

$$\boldsymbol{X}(t+1) = \boldsymbol{X}_{\text{rand}}(t) - \boldsymbol{A} \cdot \boldsymbol{D} \tag{18}$$

C. PSEUDO CODE OF WOA

The steps of the original WOA can be presented as below.

Step 1. Initialize the whale population.

Step 2. Evaluate the fitness values of whales and find out the best search agent.

Step 3. Perform the procedure below until the termination condition is met.

```
for each search agent
  if $p' < 0.5$ then
  if $|A| \leq 1$ then $X(t+1) = X_p(t) - A \cdot D$
  elseif $|A| > 1$ then
    $X(t+1) = X_{rand}(t) - A \cdot D$
    endif
 elseif $p' \geq 0.5$ then
    $X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X_p(t)$
   endif
endfor
Evaluate the fitness of $X(t+1)$ and update $X_p$
```

Step 4. Terminate the algorithm and output $\boldsymbol{X}_p$.

IV. THE PROPOSED DWOA

A. ENCODING AND DECODING APPROACH

The green job shop scheduling problem is consisted of two subproblems, such as speed selection and operation permutation. To determine a feasible schedule, it needs to select speeds for jobs on machines and arrange the operation permutation on each machine. Here, the scheduling scheme can be represented by a two-string solution, whose sizes are both equal to mn. Taking a 4×2 (4 jobs, 2 machines) JSP with multi-speed machine for example, three different speed levels are considered for each job on machines. The scheduling solution can be illustrated by Figure 1. In the first string, each element represents the selected speed level for each operation, which is stored in a predefined order. In the second string, each element equals the job code, where the elements with the same values correspond to different operations of the same job. In addition, O_{11} represents the first operation of job 1, O_{12} represents the second operation of job 1, and so on.

For each solution, the following procedure is adopted as the decoding method:

1) Read the operation permutation from left to right, and decide the machine speed level for each operation;
2) The first operation in the operation permutation string is scheduled firstly, then the second operation is processed and so on; each operation is processed in the earliest available time on the corresponding machine.
3) Repeat the above procedure to obtain a schedule scheme.

B. INITIAL SCHEDULING GENERATION

The initial solutions are crucial for the performance of a swarm-based intelligence algorithm. For the speed selection, the initial speeds are randomly generated for operations on each machine. For the operation permutation, we employ five common dispatching rules: Most Work Remaining (MWR), Shortest Processing Time (SPT), Most Operation Remaining (MOR), Longest Processing Time (LPT) and Random Rule (RR).

The initial solutions can be obtained as follows:

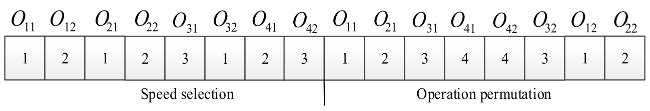
Step 1: Generate the speed selection scheme at random.

Step 2: For each speed selection scheme, a predefined number of operation permutations are randomly generated according to the five dispatching rules. The best combination of the two components is selected to be an initial scheduling solution.

Step 3: Repeat Steps 1 and 2 until all the initial scheduling solutions are obtained.

C. MODIFIED DISCRETE INDIVIDUAL UPDATING METHOD

In the original WOA, whales update their positions in a continuous domain by Equations (16) and (18). However, observed from Figure 1, the considered problem possesses discrete characteristics, which results that the standard WOA cannot directly used here. Therefore, we attempt to develop some discrete individual updating methods in order to make the algorithm directly work in a discrete search space.



O_{11} O_{12} O_{21} O_{22} O_{31} O_{32} O_{41} O_{42} O_{11} O_{21} O_{31} O_{41} O_{42} O_{32} O_{12} O_{22}
1 2 1 2 3 1 2 3 1 2 3 4 4 3 1 2
Speed selection
Operation permutation

FIGURE 1. Two-string encoding method.

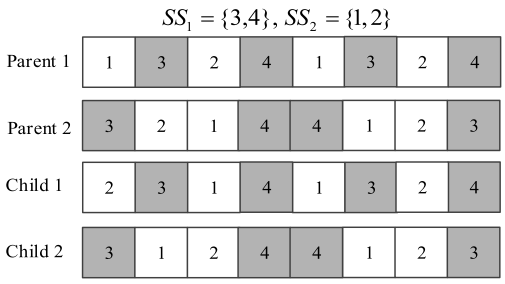


FIGURE 2. POX crossover operation.

1) MODIFIED EXPLOITATION PHASE

In the exploitation phase, individuals are updated according to the information of the current best solution. Here, the discrete individual updating method is developed based on the crossover operator of genetic algorithm, which can be shown by Equation (19). f_1 and f_2 define two different discrete crossover operations, by which partial information of the best solution can be used to update the candidate individual. However, as we know, two children will be generated by the crossover between the current individual and the best solution. The better one will be taken as the new solution by the proposed updating method.

$$\boldsymbol{X}(t+1) = \begin{cases} f_1\left(\boldsymbol{X}(t), \boldsymbol{X}_p(t)\right), & \text{if } p' < 0.5 \\ f_2\left(\boldsymbol{X}(t), \boldsymbol{X}_p(t)\right), & \text{if } p' \geq 0.5 \end{cases} \tag{19}$$

f_1: The precedence preserving order-based crossover (POX) is used for the operation permutation, and the two-point crossover (TPX) is adopted for the speed selection.

The detailed steps of the POX can be illustrated by Figure 3 and described as follows:

Step 1: Construct two job sets SS_1 and SS_2.

Step 2: Randomly choose jobs into SS_1 , others are selected to SS_2.

Step 3: Copy the jobs in SS_1 from Parent 1 to Child 1 and from Parent 2 to Child 2.

Step 4: Copy the jobs in SS_2 from Parent 2 to Child 1 and from Parent 1 to Child 2.

Step 5: Terminate the procedure.

The detailed steps of the TPX can be illustrated by Figure3 and described as follows:

Step 1: Randomly select two positions in the speed selection.

Step 2: Exchange the values between the two selected positions in parent individuals.

Step 3: Terminate the procedure.

f_2 : The job-based crossover (JBX) is used for the operation permutation, and the multi-point crossover (MPX) is adopted for the speed selection.

The detailed steps of the JBX can be illustrated by Figure4 and described as follows:

Step 1: Construct two job sets SS_1 and SS_2.

Step 2: Randomly choose jobs into SS_1, others are selected to SS_2.

Step 3: Copy the jobs in SS_1 from Parent 1 to Child 1, and copy the jobs in SS_2 from Parent 2 to Child 2.

Step 4: Copy the jobs in SS_2 from Parent 2 to Child 1 and copy the jobs in SS_1 from Parent 1 to Child 2.

Step 5: Terminate the procedure.

The detailed steps of the MPX can be illustrated by Figure5 and described as follows:

Step 1: Randomly generate a 0-1 set *BL*.

Step 2: Copy the speed level in the same place with '1' in set *BL* from Parent 1 to Child 1 and from Parent 2 to Child 2.

Step 3: Exchange the rest speed levels in Parent 1 and Parent 2 to obtain Child 1 and Child 2.

Step 4: Terminate the procedure.

2) MODIFIED EXPLORATION PHASE

In the exploration phase, it aims to search the global optimum, which is realized by randomly selecting a search agent to enhance the randomicity and improve the global search ability. Here, a modified discrete individual method in the exploration phase is proposed in Equation (20). $\bar{\boldsymbol{X}}_{\text{rand}}$ is a scheduling solution randomly selected from the current population; f_3 represents a discrete crossover operation between the current individual and the selected individual. Here, f_3 adopts the same crossover operations with f_2.

$$\boldsymbol{X}(t+1) = f_3\left(\boldsymbol{X}(t), \boldsymbol{X}_{\text{rand}}(t)\right) \tag{20}$$

3) BALANCE BETWEEN EXPLOITATION AND EXPLORATION

For a meta-heuristic algorithm, the balance between exploration and exploitation is very important for the searching ability. To implement it, a selection probability $pb \in [0, 1]$ is proposed in Equation (21), where $pb_{\max}$ and $pb_{\min}$ are the maximum and minimum values of pb. For each individual, a random number $rand \in [0, 1]$ is generated in the current generation. If $rand \leq pb$, the individual will be updated in the exploitation phase, otherwise, it will be updated in the exploration phase.

$$pb = pb_{\max} - (pb_{\max} - pb_{\min}) \times (t/t_{\max})^2 \tag{21}$$

D. VARIABLE NEIGHBBORHOOD SEARCH

To further improve the solution quality, a variable neighborhood search (VNS) strategy is embedded into the algorithm, which starts from the current best solution in each iteration and stops after the predefined number of iterations. In the VNS, we employ two types of neighborhood structures as follows:

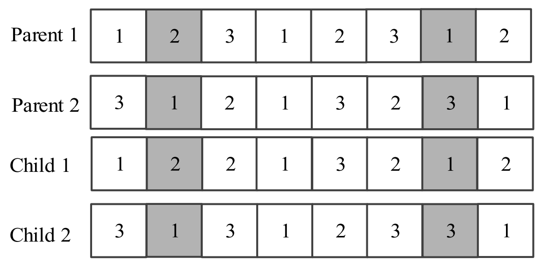

Parent 1 | 1 2 3 1 2 3 1 2
Parent 2 | 3 1 2 1 3 2 3 1
Child 1 | 1 2 2 1 3 2 1 2
Child 2 | 3 1 3 1 2 3 3 1

FIGURE 3. TPX crossover operation.

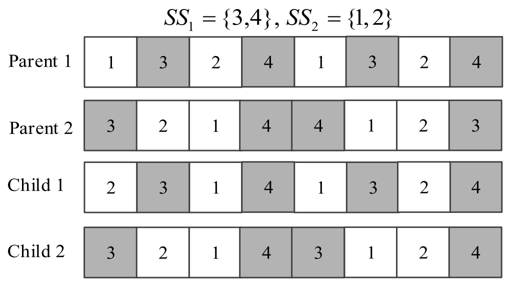

$SS_1 = \{3,4\}, SS_2 = \{1,2\}$
Parent 1 | 1 3 2 4 1 3 2 4
Parent 2 | 3 2 1 4 4 1 2 3
Child 1 | 2 3 1 4 1 3 2 4
Child 2 | 3 2 1 4 3 1 2 4

FIGURE 4. JBX crossover operation.

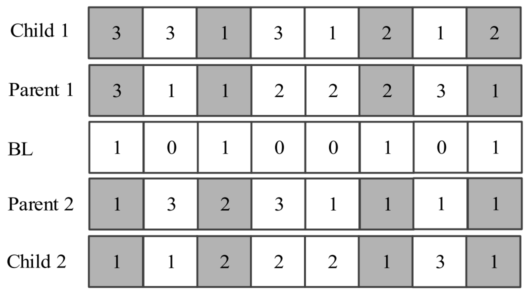

Child 1 | 3 3 1 3 1 2 1 2
Parent 1 | 3 1 1 2 2 2 3 1
BL | 1 0 1 0 0 1 0 1
Parent 2 | 1 3 2 3 1 1 1 1
Child 2 | 1 1 2 2 2 1 3 1

FIGURE 5. MPX crossover operation.

1) OPERATION PERMUTATION NEIGHBORHOOD

Swap (SP): Randomly select two operations belonging to different jobs in the second string of a scheduling solution, and then exchange the positions of the two selected operations.

Insert (IT): Randomly select two operations ps_1 and ps_2 belonging to different jobs in the second string of a scheduling solution, and then insert ps_2 before ps_1.

Inverse (IS): Randomly select two elements belonging to different jobs in the second string of a scheduling solution, and invert all items between the two selected operations.

2) SPEED SELECTION NEIGHBORHOOD

Random selection (RS): Randomly select an operation with more than one alternative speed level in the first string of a scheduling solution. Then a different speed level is randomly selected to replace the original one.

Slow down (SD): Randomly select an operation with more than one alternative speed level in the first string of a scheduling solution. Then the lowest speed level is selected to replace the original one.

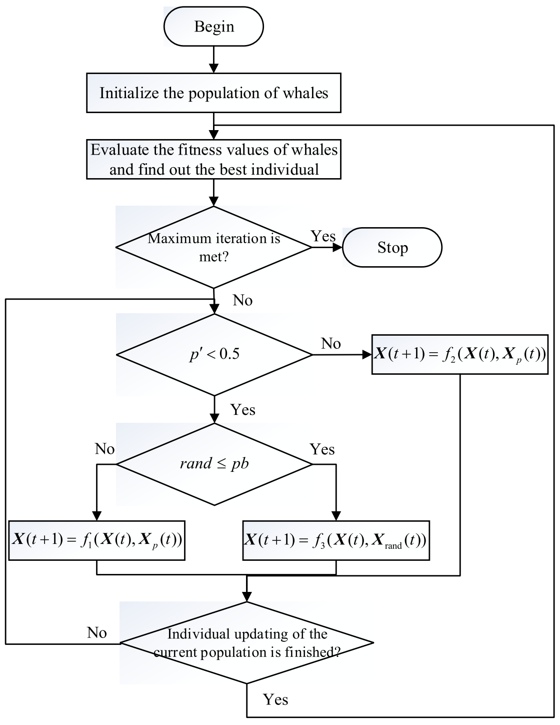

Begin
Initialize the population of whales
Evaluate the fitness values of whales and find out the best individual
Maximum iteration is met?
Yes
Stop
No
$p' < 0.5$
No
$X(t+1) = f_2(X(t), X_p(t))$
Yes
No
$rand \le pb$
Yes
$X(t+1) = f_1(X(t), X_p(t))$
$X(t+1) = f_3(X(t), X_{rand}(t))$
No
Individual updating of the current population is finished?
Yes

FIGURE 6. Flowchart of the DWOA.

Speed up (SU): Randomly select an operation with more than one alternative speed level in the first string of a scheduling solution. Then the highest speed level is selected to replace the original one.

Based on the above neighborhood structures, the detailed steps of the VNS are shown as below.

Step 1. Acquire the initial solution X, set $\rho \leftarrow 1$, $\lambda_{\max} \leftarrow 9$ and the maximum iteration $\rho_{\max}$.

Step 2. Set $\lambda \leftarrow 1$.

Step 3. Perform the procedure below until $\lambda > \lambda_{\max}$.

```
if $\lambda = 1$ then $\mathbf{X}' \in$ SP+RS($X$)
elseif $\lambda = 2$ then $\mathbf{X}' \in$ SP+SD($X$)
elseif $\lambda = 3$ then $\mathbf{X}' \in$ SP+SU($X$)
elseif $\lambda = 4$ then $\mathbf{X}' \in$ IT+RS($X$)
elseif $\lambda = 5$ then $\mathbf{X}' \in$ IT+SU($X$)
elseif $\lambda = 6$ then $\mathbf{X}' \in$ IT+SU($X$)
elseif $\lambda = 7$ then $\mathbf{X}' \in$ IS+RS($X$)
elseif $\lambda = 8$ then $\mathbf{X}' \in$ IS+SD($X$)
elseif $\lambda = 9$ then $\mathbf{X}' \in$ IS+SU($X$)
endif
$\mathbf{X}'' \in$ Local search($\mathbf{X}'$)
if $F(\mathbf{X}'') < F(X)$ then $X \leftarrow \mathbf{X}''$ and $\lambda \leftarrow 1$
else $\lambda \leftarrow \lambda + 1$
endif
```

Step 4. Set $\rho \leftarrow \rho + 1$. If $\rho > \rho_{\max}$ holds, go to Step 5; otherwise, go to Step 2.

Step 5. Terminate the VNS and output X.

The procedure of the local search in VNS is given as below.

Step 1. Get the initial solution $\mathbf{X}'$ and set the termination condition $h_{\max}$.

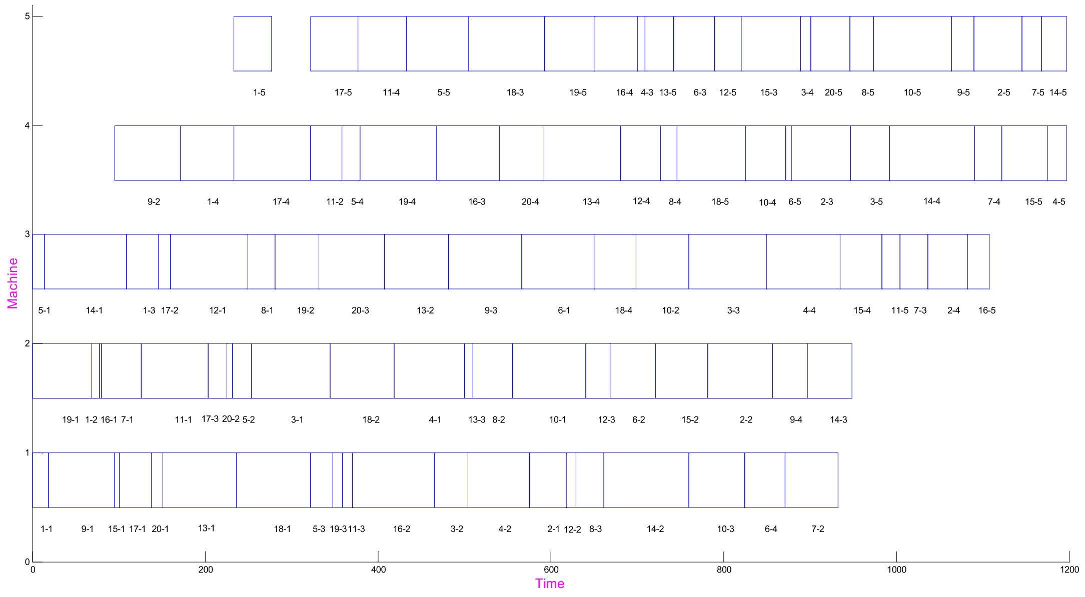


FIGURE 7. Gantt chart for instance FT20.

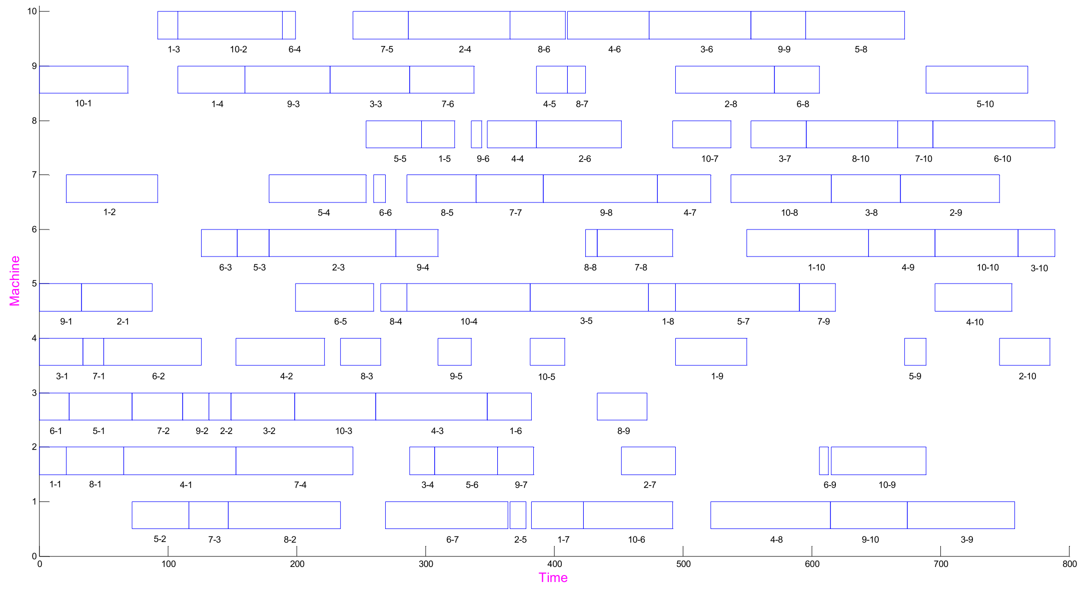


FIGURE 8. Gantt chart for instance LA16.

```
Step 2. Perform the procedure below until $h > h_{\max}$
    $\bar{X}$ := Randomly perform one of the nine combinations of neighborhoods to $\mathbf{X}'$
    if $F(\bar{X}) < F(\boldsymbol{X}')$ then
        $X' = \bar{X}$
    endif
    $h \leftarrow h+1$
Step 3. Set $\mathbf{X}'' \leftarrow \bar{X}$ and output the local solution $\mathbf{X}''$.
```

E. STEPS OF DWOA

The steps of DWOA are described below and illustrated in Figure 6.

Step 1. Initialize a predefined number of whales with the population initialization method in Section 4.2.

Step 2. Evaluate the fitness values of whales and find out the individual with the best fitness.

```
Step 3. Perform the individual updating procedure below.
    for each whale individual
        if $p' < 0.5$ then
            if $rand > pb$ then
                $\boldsymbol{X}(t+1) = f_1(\boldsymbol{X}(t), \boldsymbol{X}_p(t))$
            elseif $rand \leq pb$ then
                $\boldsymbol{X}(t+1) = f_3(\boldsymbol{X}(t), \boldsymbol{X}_{\text{rand}}(t))$
            endif
        elseif $p' \geq 0.5$ then
            $\boldsymbol{X}(t+1) = f_2(\boldsymbol{X}(t), \boldsymbol{X}_p(t))$
        endif
    endfor
```

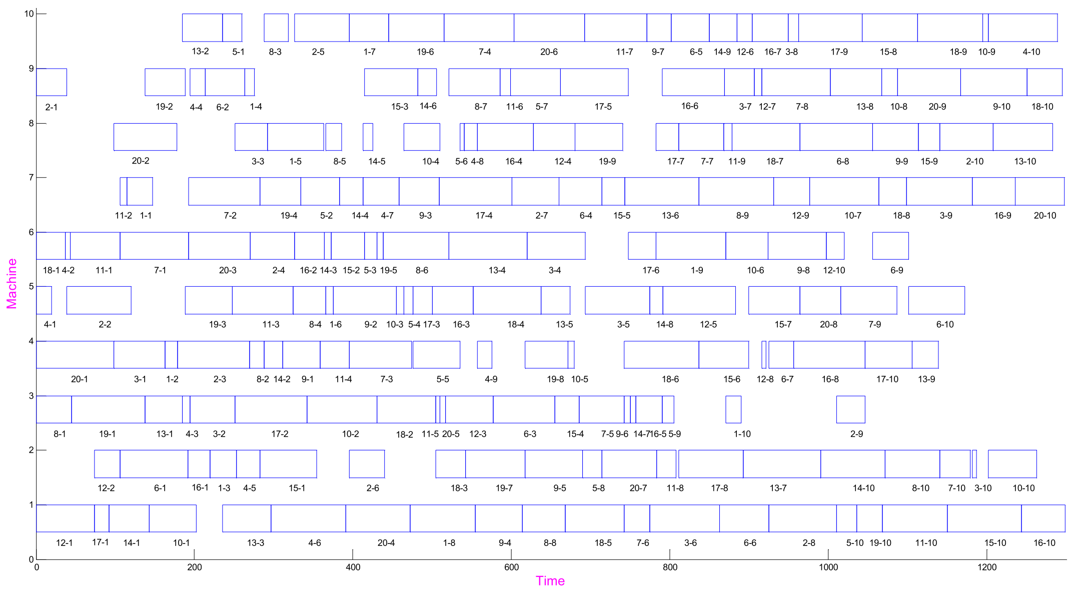


FIGURE 9. Gantt chart for instance LA30.

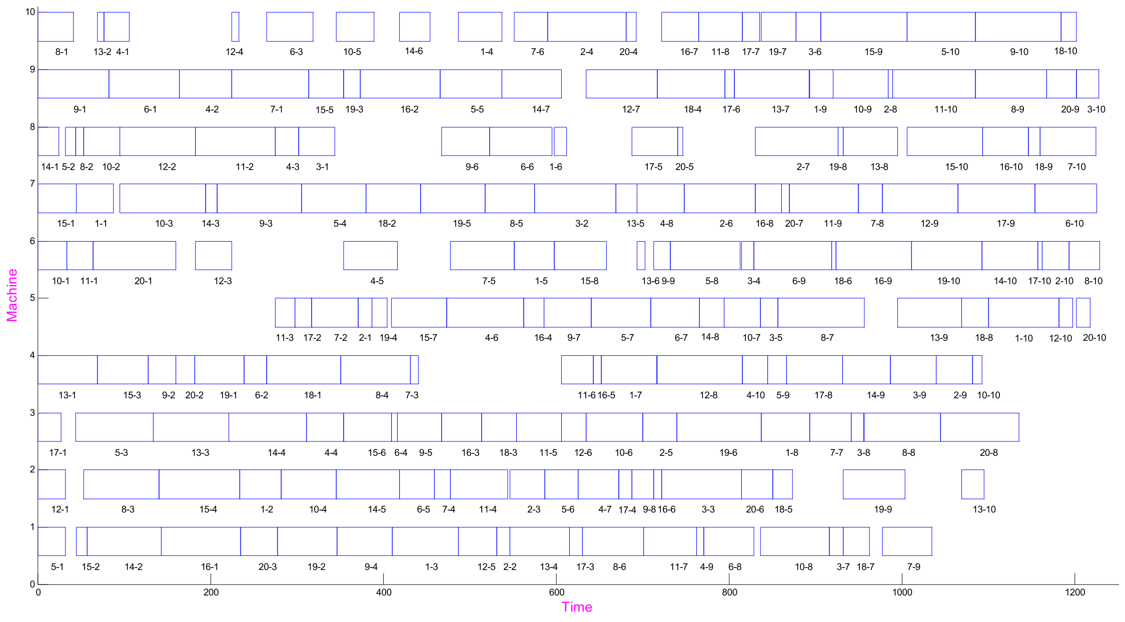


FIGURE 10. Gantt chart for instance RM10.

Step 4. Check whether the maximum iteration is met. If yes, go to Step 5; otherwise, go to Step 2.

Step 5. Terminate the algorithm and output X_p.

V. COMPUTATIONAL RESULTS

To evaluate the performance of the proposed DWOA, we coded the algorithm in FORTRAN and run it on VMware Workstation with 2GB main memory under WinXP.

A. EXPERIMENTAL SETTINGS

In this section, simulation instances are designed based on 43 benchmark instances of the standard JSP (FT06, FT10, FT20 designed by Fisher and Thompson [27] and LA01∼LA40 designed by Lawrence [28]) and 18 random instances (RM01∼RM18). In the random instances, processing times of operations are randomly generated following a discrete uniform distribution in [5,100], and the processing routing of each job is also generated at random. The original processing times of the standard JSP are taken as the basic processing times. The speed of each machine is chosen from $\mathbf{v} = \{v_1, v_2, v_3, v_4, v_5\} = \{1.0,1.3,1.5,1.8,2.0\}$. In addition, E_{kd} is measured by $\xi_k \times v_d^2, d = 1, 2, 3, 4, 5$, where ξ_k is randomly generated following a discrete uniform distribution in [2], [4]. SE_k is calculated by $\xi_k/4$. η is set to be 10.0.

TABLE 1. Comparison between different algorithms for experiment 1.

Instance	(m,n)	MGA				MTLBO			
		Best	*Avg*	*ARPD*	*Time*	*Best*	*Avg*	*ARPD*	*Time*
FT06	(6,6)	1326.0	1371.2	9.01	4.6	1399.9	1418.6	12.78	64.4
FT10	(10,10)	33699.4	34302.1	22.33	18.1	34251.3	34573.3	23.30	291.9
FT20	(5,20)	36664.5	37606.0	23.09	20.3	36034.9	36309.4	18.85	592.7
LA01	(5,10)	18645.3	19094.5	14.43	7.6	19588.4	19842.5	18.92	138.6
LA02	(5,10)	17847.4	18661.4	16.59	7.5	18592.7	18867.1	17.87	138.2
LA03	(5,10)	16563.8	17058.6	16.70	7.2	16901.8	17118.4	17.11	133.4
LA04	(5,10)	16280.5	17346.9	15.85	7.2	17114.7	17579.4	17.41	139.9
LA05	(5,10)	14746.7	15296.6	13.35	7.2	15570.5	15771.0	16.86	139.1
LA06	(5,15)	25388.0	26249.4	14.88	12.8	27087.6	28425.2	24.40	346.5
LA07	(5,15)	24816.0	25809.5	18.49	12.6	25711.2	26173.5	20.16	347.5
LA08	(5,15)	25090.4	25713.6	16.57	12.7	26319.0	26628.9	20.72	335.4
LA09	(5,15)	27694.2	28192.6	13.09	12.7	29609.4	29772.7	19.42	347.0
LA10	(5,15)	25805.2	26351.9	11.89	12.8	27450.2	27797.6	18.03	339.1
LA11	(5,20)	35361.3	35936.3	17.50	19.7	36477.4	36844.2	20.47	637.0
LA12	(5,20)	30185.5	31117.7	15.24	19.3	31779.7	32200.9	19.25	656.2
LA13	(5,20)	34008.5	34466.7	15.76	19.4	35403.0	35720.6	19.97	668.3
LA14	(5,20)	35176.3	36147.2	14.64	19.4	36807.3	37225.0	18.05	661.3
LA15	(5,20)	36923.5	37567.4	19.33	19.8	37405.5	38068.0	20.92	631.2
LA16	(10,10)	33441.5	34807.1	22.03	17.8	35129.2	35435.1	24.23	331.9
LA17	(10,10)	29746.1	30196.7	19.85	17.8	30507.5	30963.1	22.89	309.4
LA18	(10,10)	33043.6	33808.5	21.95	17.9	34093.3	34474.4	24.36	307.6
LA19	(10,10)	33091.0	33697.1	21.22	17.6	34006	34226.6	23.13	336.4
LA20	(10,10)	35510.4	36006.2	21.04	17.9	36205.3	36790.1	23.67	329.1
LA21	(10,15)	48847.0	49665.9	26.15	32.7	50335.5	50819.1	29.08	762.3
LA22	(10,15)	45259.4	46048.9	29.70	32.2	46124.4	46455.2	30.84	768.7
LA23	(10,15)	47539.2	48930.7	28.20	32.5	50103.0	50565.4	32.48	764.8
LA24	(10,15)	45639.7	46985.0	27.93	32.3	48148.6	48325.4	31.58	721.5
LA25	(10,15)	46750.4	47321.3	28.26	32.6	46224.0	47615.9	29.06	733.3
LA26	(10,20)	62648.5	64371.1	30.78	52.7	64740.8	65212.9	32.49	1418.9
LA27	(10,20)	63661.5	65736.4	29.42	52.1	66724.4	67318.8	32.54	1413.5
LA28	(10,20)	64480.8	65386.0	29.85	52.8	66158.3	66520.9	32.1	1557.2
LA29	(10,20)	60359.2	61843.6	30.12	48.2	61941.1	62482.7	31.47	1474.1
LA30	(10,20)	64380.6	65606.2	28.42	48.6	65706.7	66626.8	30.42	1527.8
LA31	(10,30)	91121.1	92507.1	32.00	92.8	92440.4	92946.4	32.63	3920.8
LA32	(10,30)	96631.3	99711.9	32.53	94.4	100383.9	100736.1	33.89	3768.6
LA33	(10,30)	88775.9	90107.7	31.39	95.4	91388.4	91645.5	33.63	3646.6
LA34	(10,30)	91031.9	92079.1	30.69	96.7	92742.3	93384.4	32.55	3724.6
LA35	(10,30)	92345.1	94558.9	31.97	95.6	95033.7	95106.3	32.73	3871.4
LA36	(15,15)	76319.4	77337.2	30.36	59.2	77780.2	78063.6	31.58	1177.2
LA37	(15,15)	80534.0	83323.9	29.83	58.5	82272.3	82912.2	29.19	1189.9
LA38	(15,15)	73094.9	74220.0	27.55	58.4	73373.1	74202.0	27.52	1213.7
LA39	(15,15)	74264.9	76027.6	29.83	60.1	76042.3	76313.1	30.32	1217.2
LA40	(15,15)	76155.5	77382.3	32.85	57.5	75216.5	75820.2	30.16	1265.3
Mean	-	46765.0	47812.9	23.09	35.2	48054.1	48495.3	25.33	1031.6

TABLE 1. ***(Continued.)*** **Comparison between different algorithms for experiment 1.**

Instance	(m,n)	MDGWO				DWOA			
		Best	*Avg*	*ARPD*	*Time*	*Best*	*Avg*	*ARPD*	*Time*
FT06	(6,6)	1258.2	1267.3	0.75	74.7	**1257.9**	**1265.9**	**0.64**	91.2
FT10	(10,10)	28109.6	**28401.1**	**1.29**	221.4	**28039.9**	28487.8	1.60	273.9
FT20	(5,20)	30611.9	**30960.5**	**1.34**	244.7	**30550.7**	31005.3	1.49	292.0
LA01	(5,10)	**16685.9**	16855.7	1.02	106.4	16726.4	**16850.3**	**0.99**	131.3
LA02	(5,10)	**16006.3**	**16245.1**	**1.49**	109.1	16119.6	16320.5	1.96	129.6
LA03	(5,10)	**14617.6**	14905.6	1.97	107.2	14628.4	**14826.0**	**1.43**	128.1
LA04	(5,10)	14986.5	15159.3	1.24	105.9	**14973.0**	**15122.0**	**1.00**	131.2
LA05	(5,10)	**13495.6**	13558.6	0.47	100.1	13509.2	**13554.0**	**0.43**	130.4
LA06	(5,15)	22856.8	22953.6	0.46	166.3	**22849.4**	**22932.4**	**0.36**	215.7
LA07	(5,15)	21783.8	21894.4	0.52	166.8	**21782.2**	**21857.7**	**0.35**	215.0
LA08	(5,15)	22104.6	22227.5	0.76	166.4	**22058.9**	**22210.7**	**0.67**	216.9
LA09	(5,15)	24935.1	24958.5	0.11	165.0	**24930.3**	**24947.3**	**0.07**	218.5
LA10	(5,15)	**23551.1**	**23570.5**	**0.08**	160.3	23604.7	23622.2	0.30	216.1
LA11	(5,20)	**30582.9**	**30649.1**	**0.22**	254.1	30587.5	30667.1	0.28	298.7
LA12	(5,20)	27002.6	27050.4	0.18	249.5	**27001.9**	**27035.3**	**0.12**	298.7
LA13	(5,20)	29822.2	29869.2	0.32	245.2	**29773.9**	**29794.6**	**0.07**	301.7
LA14	(5,20)	**31532.3**	31608.3	0.24	239.2	31568.6	**31607.0**	**0.23**	305.4
LA15	(5,20)	31488.9	31677.5	0.62	246.3	**31483.2**	**31624.9**	**0.45**	301.4
LA16	(10,10)	28697.5	**29067.5**	**1.91**	251.7	**28523.2**	29193.2	2.35	295.2
LA17	(10,10)	25451.9	25588.2	1.56	235.2	**25195.7**	**25511.2**	**1.25**	294.6
LA18	(10,10)	**27722.3**	**28079.6**	**1.29**	239.0	27806.3	28350.5	2.27	280.3
LA19	(10,10)	**27798.0**	**28095.4**	**1.07**	233.6	27865.9	28315.2	1.86	288.7
LA20	(10,10)	29792.9	30362.7	2.07	228.0	**29748.1**	**30093.8**	**1.16**	292.4
LA21	(10,15)	39393.0	39985.9	1.56	379.4	**39370.3**	**39802.7**	**1.10**	474.3
LA22	(10,15)	**35504.5**	**36281.1**	**2.19**	374.0	35890.7	36734.8	3.47	462.4
LA23	(10,15)	38804.3	39041.4	2.29	389.1	**38166.9**	**38930.1**	**2.00**	468.8
LA24	(10,15)	36899.2	37699.6	2.65	389.6	**36726.6**	**37557.3**	**2.26**	459.2
LA25	(10,15)	37175.4	37539.8	1.75	386.6	**36893.4**	**37483.6**	**1.60**	448.5
LA26	(10,20)	**49222.0**	50263.0	2.11	563.2	49442.4	**49959.1**	**1.50**	672.3
LA27	(10,20)	51333.0	51987.9	2.36	554.8	**50791.2**	**51429.1**	**1.26**	676.1
LA28	(10,20)	50417.3	50910.3	1.10	595.8	**50356.1**	**50885.2**	**1.05**	653.7
LA29	(10,20)	47731.8	**48366.1**	**1.77**	577.1	**47527.1**	48392.8	1.82	665.7
LA30	(10,20)	51792.1	52297.6	2.37	566.5	**51086.2**	**51974.4**	**1.74**	668.4
LA31	(10,30)	70437.8	71119.4	1.48	936.6	**70080.2**	**70605.1**	**0.75**	1196.5
LA32	(10,30)	75660.5	76425.2	1.58	1037.4	**75236.1**	**75781.6**	**0.73**	1136.2
LA33	(10,30)	69075.4	69432.0	1.24	928.3	**68581.8**	**68739.7**	**0.23**	1104.3
LA34	(10,30)	70851.2	71540.1	1.54	939.6	**70454.4**	**70897.2**	**0.63**	1133.8
LA35	(10,30)	**71651.4**	72334.2	0.95	970.0	71864.0	**72195.7**	**0.76**	1127.3
LA36	(15,15)	59608.4	**60067.6**	**1.25**	656.4	**59326.0**	60406.8	1.82	780.2
LA37	(15,15)	64502.5	65201.6	1.60	583.3	**64176.8**	**64881.8**	**1.10**	759.4
LA38	(15,15)	**58187.6**	**58967.8**	**1.34**	627.2	58374.6	59076.5	1.53	786.4
LA39	(15,15)	58959.1	60123.7	2.67	624.1	**58557.4**	**59728.4**	**2.00**	774.7
LA40	(15,15)	**58249.3**	**59251.4**	**1.72**	643.8	58967.6	59680.5	2.46	766.7
Mean	-	38054.7	38461.4	1.31	396.3	37964.1	38379.9	1.19	478.2

B. EFFECTIVENESS OF DWOA

To verify the superiority of our proposed DWOA algorithm, we compared it with three algorithms, such as modified genetic algorithm (MGA), modified teaching–learning based optimization (MTLBO) algorithm and modified discrete grey wolf optimization (MDGWO) algorithm. It is well known that genetic algorithm (GA) has been often modified according to the characteristics of the JSP problem. Here, GA is first

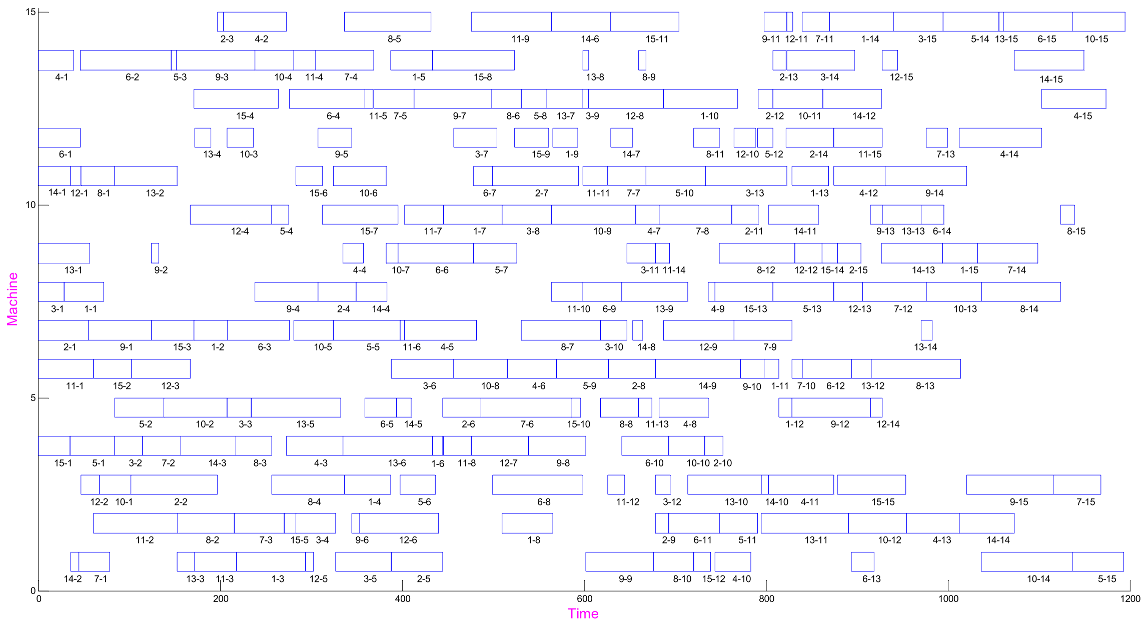


FIGURE 11. Gantt chart for instance RM15.

compared with our algorithm. In the MGA, the proposed population initialization method and the crossover operator f_1 are used. The swap mutation and one-point mutation are respectively adopted to the operation permutation and speed selection. In addition, due the fact that few literature work on the problem considered in this study, two algorithms for the standard JSP, TLBO [29] and DGWO [30], are modified with the addition of speed selection to implement the comparison with our algorithm. After plenty of preliminary experiments, the parameters for the proposed DWOA algorithm are set as follows: the population size is 200, the maximum of iteration is 1500, the maximum iteration of variable neighborhood search and local search are 10 and 20, respectively. To facilitate the comparison, the population size and the maximum of iterations of other algorithms are also 200 and 1500. In addition, the crossover rate and the mutation rate of the MGA are set to be 0.8 and 0.1, respectively; the maximum iteration of variable neighborhood search and local search of the MDGWO are set to be 10 and 20, respectively. For each instance, ten independent runs are conducted by different algorithms. The boldface means the best values obtained by all the algorithms.

1) EXPERIMENT 1

The experiments are first conducted on the modified 43 benchmark instances (FT06, FT10, FT20 and LA01∼LA40). In Table 1, the problem names are listed in the first column, and the computational results are reported in the following columns. ‘*Best*’ means the best value in the ten runs of each algorithm. ‘*Avg*’ represents the average results of the ten runs. ‘*ARPD*’ is the average relative percent difference, i.e., $ARPD = 100 \times (Avg - Min)/Min$, where ‘*Min*’ is the minimum value obtained by compared algorithms among the all conducted experiments for each instance. In the last row, ‘Mean’ defines the average results. It can be observed from Table 1 as follows:

TABLE 2. ANOVA for ARPD of the algorithms in Experiment 1.

Source	DF	Sum of Squares	Mean Square	F	p-value
Factor	3	22764.56521	7588.1884	341.47257	0.00
Error	168	3733.28859	22.22196		
Total	171	26497.85379			

For the ‘*Best*’ value, DWOA can obtain 29 boldface values out of 43 instances. The secondly best algorithm, MDGWO, can obtain 14 boldface values. In addition, DWOA can obtain the better mean value than those of other algorithms.

For the ‘*Avg*’ value, DWOA can obtain 30 boldface values out of 43 instances. The secondly best algorithm, MDGWO, can obtain 13 boldface values. In addition, DWOA can obtain the better mean value than those of other algorithms.

For the ‘*ARPD*’ value, DWOA can obtain 30 boldface values out of 43 instances. The secondly best algorithm is MDGWO, which can obtain 13 boldface values. In addition, DWOA can obtain the better mean value than those of other algorithms.

For the ‘*Time*’ value, GA spends the shortest time among the compared algorithms. The computational time of the proposed DWOA is shorter than MTLBO.

Figures 7∼9 show the Gantt charts obtained by our DWOA algorithm for three instances (FT20, LA16 and LA30).

TABLE 3. Comparison between different algorithms for experiment 2.

Instance	(m,n)	MGA				MTLBO			
		Best	*Avg*	*ARPD*	*Time*	*Best*	*Avg*	*ARPD*	*Time*
RM01	(5,5)	9875.1	10009.7	5.25	3.3	10298.7	10443.4	9.81	38.7
RM02	(5,10)	15854.2	16411.8	14.20	7.4	16546.8	16943.0	17.89	145.5
RM03	(5,15)	24769.6	25284.1	14.90	13.1	25884.9	26138.8	18.79	331.6
RM04	(5,20)	35017.0	35535.4	18.49	20.6	35761.5	36084.0	20.32	633.2
RM05	(5,25)	43762.4	44395.7	16.88	30.1	45282.0	45747.0	20.43	1103.4
RM06	(5,30)	53719.0	54780.6	17.71	40.9	55072.5	55856.4	20.02	1952.2
RM07	(10,5)	16481.4	16789.6	9.46	7.5	17107.3	17315.9	12.89	84.3
RM08	(10,10)	30636.1	33093.7	20.18	18.5	34067.4	34454.6	25.12	369.4
RM09	(10,15)	48775.8	49926.0	26.06	33.5	50352.4	50915.5	28.55	864.2
RM10	(10,20)	61528.7	63160.5	28.23	52.0	63764.4	64319.3	30.59	1605.9
RM11	(10,25)	76612.1	78388.7	30.41	73.8	79507.6	79824.6	32.80	2558.2
RM12	(10,30)	94461.0	96125.0	30.17	97.8	97148.6	97733.5	32.35	3696.3
RM13	(15,5)	27998.4	28462.2	8.93	13.0	28535.8	28921.4	10.68	142.7
RM14	(15,10)	53415.4	54503.6	21.06	34.0	55011.7	55534.5	23.35	522.6
RM15	(15,15)	75060.7	75998.4	28.47	60.0	76636.3	76935.9	30.05	1149.0
RM16	(15,20)	98751.9	99510.9	29.23	91.9	101926.9	102323.8	32.88	2236.2
RM17	(15,25)	120879.3	122216.0	35.40	129.1	122380.7	122841.1	36.10	3941.5
RM18	(15,30)	143447.8	144878.3	34.59	173.2	146886.3	147789.9	37.30	6101.3
Mean	-	57280.3	58303.9	21.65	50.0	59009.5	59451.3	24.44	1526.5

Instance	(m,n)	MDGWO				DWOA			
		Best	*Avg*	*ARPD*	*Time*	*Best*	*Avg*	*ARPD*	*Time*
RM01	(5,5)	**9510.1**	**9571.1**	**0.64**	52.9	**9510.1**	9576.3	0.70	63.4
RM02	(5,10)	**14371.3**	14489.9	0.83	109.4	14423.9	**14482.4**	**0.77**	134.5
RM03	(5,15)	**22004.5**	**22018.3**	**0.06**	173.8	22005.1	22090.2	0.39	212.3
RM04	(5,20)	30094.8	30243.7	0.85	262.2	**29989.0**	**30111.7**	**0.41**	316.3
RM05	(5,25)	**37985.2**	38010.1	0.07	350.9	37991.2	**38006.0**	**0.05**	413.8
RM06	(5,30)	46540.4	46641.1	0.22	439.5	**46538.8**	**46586.1**	**0.10**	539.4
RM07	(10,5)	**15339.1**	15553.0	**1.39**	107.9	15484.3	**15552.5**	**1.39**	134.9
RM08	(10,10)	27628.2	**28046.7**	**1.85**	226.0	**27536.7**	28068.9	1.93	284.3
RM09	(10,15)	**39606.4**	**40338.9**	**1.85**	377.0	39690.5	40401.0	2.01	480.0
RM10	(10,20)	49555.7	50166.0	1.85	557.8	**49254.6**	**49987.9**	**1.49**	647.6
RM11	(10,25)	60570.4	61292.9	1.97	741.7	**60107.3**	**60770.1**	**1.10**	911.5
RM12	(10,30)	74273.9	74687.5	1.14	928.9	**73845.2**	**74393.3**	**0.74**	1226.1
RM13	(15,5)	**26129.6**	**26327.0**	**0.76**	194.4	26162.4	26411.3	1.08	229.4
RM14	(15,10)	**45022.8**	**45438.4**	**0.92**	382.2	45311.0	45542.9	1.16	510.7
RM15	(15,15)	59310.1	60096.0	1.59	630.4	**59156.8**	**59912.4**	**1.28**	820.6
RM16	(15,20)	77044.8	77964.1	1.25	939.9	**77002.6**	**77791.6**	**1.02**	1155.5
RM17	(15,25)	91159.9	92217.0	2.17	1220.1	**90260.9**	**91293.7**	**1.14**	1157.4
RM18	(15,30)	109124.6	110010.5	2.20	1594.9	**107641.8**	**108586.7**	**0.88**	1816.8
Mean	-	46404.0	46839.6	1.20	516.1	**46217.3**	**46642.5**	**0.98**	614.1

To check the significant differences from algorithms in Table 1, an analysis of variance (ANOVA) test is performed in Table 2, where the four compared algorithms are taken as factors. The results demonstrate that there is a statistically significant difference between the compared algorithms as p-value is equal to zero.

TABLE 4. ANOVA for ARPD of the algorithms in experiment 2.

Source	DF	Sum of Squares	Mean Square	F	p-value
Factor	3	8745.13869	2915.04623	74.12318	0.00
Error	68	2674.23974	39.32705		
Total	71	11419.37843			

2) EXPERIMENT 2

The experiments are conducted on the instances designed according to 18 random instances (RM01∼RM18). It can be observed from Table 3 as follows:

For the '*Best*' value, DWOA can obtain 11 boldface values out of 18 instances. The secondly best algorithm, MDGWO, can only obtain 8 boldface values. In addition, DWOA can obtain the better mean value than those of other algorithms.

For the '*Avg*' value, DWOA can obtain 12 boldface values out of 18 instances. The secondly best algorithm, MDGWO, can obtain 6 boldface values. In addition, DWOA can obtain the better mean value than those of other algorithms.

For the '*ARPD*' value, DWOA can obtain 12 boldface values out of 18 instances. The secondly best algorithm, MDGWO, can obtain 6 boldface values. In addition, DWOA can obtain the better value than those of other algorithms.

Figures 10 and 11 show the Gantt charts obtained by our DWOA algorithm for two instances (RM10 and RM15).

To check the significant differences from algorithms in Table 3, an analysis of variance (ANOVA) test is performed in Table 4, where the four compared algorithms are taken as factors. The results demonstrate that there is a statistically significant difference between the compared algorithms as p-value is equal to zero.

VI. CONCLUSIONS

In this study, a discrete whale optimization algorithm (DWOA) is proposed to solve a green job shop scheduling problem with multi-speed machine to minimize the sum of energy-consumption cost and completion-time cost in the workshop.

A two-string encoding method is first developed to represent the two subproblems of the problem, and a heuristic-based initialization approach is used to generate the initial population. According to the characteristics of the problem, some modified individual updating methods are developed to make the algorithm work directly in a discrete scheduling domain. In addition, a variable neighborhood search (VNS) strategy is added to further enhance the search ability.

The proposed DWOA algorithm is tested on problems with different scales, which are designed based on the 43 modified benchmarks instances and 18 random instances. The experimental results show the superiority of the proposed algorithm. In the future work, the green production scheduling will be further studied by considering some practical constraints, e.g., flexible processing routing and time-of-use electricity policy, etc. Furthermore, in the real-life production, there always exists some dynamic factors, such as machine breakdown and the arrival of new jobs, etc. It is clear that the problem with these dynamic factors has not only a theoretical but also a practical significance.

REFERENCES

[1] 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.

[2] Y. Yin, T. C. E. Cheng, D.-J. Wang, and C.-C. Wu, "Two-agent flowshop scheduling to maximize the weighted number of just-in-time jobs," *J. Scheduling*, vol. 20, no. 4, pp. 313–335, Aug. 2017.

[3] 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.

[4] G. Mouzon, M. B. Yildirim, and J. Twomey, "Operational methods for minimization of energy consumption of manufacturing equipment," *Int. J. Prod. Res.*, vol. 45, pp. 4247–4271, Sep. 2007.

[5] 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.

[6] 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.

[7] 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.

[8] J.-Y. Ding, S. Song, R. Zhang, R. Chiong, and C. Wu, "Parallel machine scheduling under time-of-use electricity prices: New models and optimization approaches," *IEEE. Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 1138–1154, Apr. 2016.

[9] A. Che, S. Zhang, and X. Wu, "Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs," *J. Cleaner Prod.*, vol. 156, pp. 688–697, Jul. 2017.

[10] X.-L. Zheng and L. Wang, "A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem," *IEEE. Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 5, pp. 790–800, May 2018.

[11] K. Fang, N. Uhan, F. Zhao, and J. W. Sutherland, "A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction," *J. Manuf. Syst.*, vol. 30, no. 4, pp. 234–240, 2011.

[12] G.-S. Liu, B.-X. Zhang, H.-D. Yang, X. Chen, and G. Q. Huang, "A branch-and-bound algorithm for minimizing the energy consumption in the PFS problem," *Math. Problem Eng.*, vol. 2013, Jan. 2013, Art. no. 546810.

[13] M. Dai, D. B. Tang, A. Giret, M. A. Salido, and W. D. Li, "Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm," *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 5, pp. 418–429, 2013.

[14] 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.

[15] 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.

[16] F. Wang, G. Deng, T. Jiang, and S. Zhang, "Multi-objective parallel variable neighborhood search for energy consumption scheduling in blocking flow shops," *IEEE. Access*, vol. 6, pp. 68686–68700, Nov. 2018.

[17] 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.

[18] 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, pp. 1–12, Oct. 2018.

[19] 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.

[20] 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.
[21] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[22] D. Oliva, M. A. El Aziz, and A. E. Hassanien, "Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm," *Appl. Energy*, vol. 200, pp. 141–154, Aug. 2017
[23] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neuro Comput.*, vol. 260, pp. 302–312, Oct. 2017.
[24] J. Wang, P. Du, T. Niu, and W. Yang, "A novel hybrid system based on a new proposed algorithm—Multi-objective whale optimization algorithm for wind speed forecasting," *Appl. Energy*, vol. 208, pp. 344–360, Dec. 2017.
[25] M. Abdel-Basset, D. El-Shahat, and A. K. Sangaiah, "A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 3, pp. 495–514, Mar. 2019.
[26] M. Abdel-Basset, G. Manogaran, D. El-Shahat, and S. Mirjalili, "A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem," *Future Gener. Comput. Syst.*, vol. 85, pp. 129–145, Aug. 2018.
[27] H. Fisher and G. L. Thompson, "Probabilistic learning combinations of local job-shop scheduling rules," *Ind. Scheduling*, vol. 3, pp. 225–251, Mar. 1963.
[28] S. Lawrence, "Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques," Graduate school of Industrial Administration (GSIA), Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. 12, 1984.
[29] A. Baykasoğlu, A. Hamzadayi, and S. Y. Köse, "Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial problems: Flow shop and job shop scheduling cases," *Inf. Sci.*, vol. 276, pp. 204–218, Aug. 2014.
[30] 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, May 2018.

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, 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 recent publications have appeared in some peer-reviewed journals, such as the *Journal of Intelligent and Fuzzy Systems*, the *International Journal of Industrial Engineering: Theory, Applications and Practice*, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, the IEEE Access, *Mathematics*, and *Mathematical Problem in Engineering*. His research interests include production scheduling and intelligence algorithm.

CHAO ZHANG was born in Xinxiang, China, in 1983. He received the B.S. degree in automation from the Zhongyuan University of Technology, Zhengzhou, China, in 2007, and the M.S. degree in control science and engineering from the University of Science and Technology Beijing, Beijing, China, in 2010. He is currently pursuing the Ph.D. degree with the MOE Key Laboratory of Measurement and Control of Complex Systems of Engineering, School of Automation, Southeast University, China.

Since 2014, he has been a Lecturer with the Department of Computer Science and Technology, Henan Institute of Technology, Xinxiang, China. His research interests include adaptive control and intelligence algorithm.

QI-MING SUN was born in Datong, China, in 1987. He received the B.S. degree from the Tianjin University of Technology, Tianjin, China, in 2010, and the Ph.D. degree in control theory and control engineering from Southeast University, Nanjing, China, in 2018.

Since 2019, he has been a Lecturer with the College of Information Science and Technology, Nanjing Forestry University. His research interests include adaptive control and intelligence algorithm.