Invasive Weed Optimization Algorithm for Solving Permutation Flow-Shop Scheduling Problem

Article *in* Journal of Computational and Theoretical Nanoscience · March 2013

DOI: 10.1166/jctn.2013.2759

Huan Chen
Xidian University

Yong-Quan Zhou
Guangxi University for Nationalities

Journal of Computational and Theoretical Nanoscience
Vol. 10, 1–6, 2013

Invasive Weed Optimization Algorithm for Solving Permutation Flow-Shop Scheduling Problem

Huan Chen[1], Yongquan Zhou[1,2,*], Sucai He[1], Xinxin Ouyang[1], and Peigang Guo[1]

[1] *College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China*
[2] *Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, China*

An Invasive Weed Optimization (IWO) scheduling algorithm for solving Permutation Flow-shop Scheduling Problem (PFSP) is proposed. The Most Position Value (MPV) method is used to coding the weed individuals so that fitness values can be calculated. Then, the global exploration capacity of IWO is used to select the best fitness value and its corresponding processing sequence of job by evaluating the fitness of individuals. The results of 8 PFSP benchmarks compared with other algorithms show that PFSP can be effectively solved by IWO with adaptability and robustness.

Keywords: Permutation Flow-Shop Scheduling Problem, Invasive Weed Optimization, Most Position Value Method, Global Exploration Capacity.

1. INTRODUCTION

The Flow-shop Scheduling Problem is a simplified model of the actual production scheduling process, about 25% of manufacturing systems, assembly lines and information services and facilities can be simplified for the FSP.[1] Therefore, its study has important theoretical significance and engineering value. In particular, if all jobs processed on each machine in the same order, such FSP is called the Permutation FSP (PFSP). At present PFSP with more than 3 machines has been proved as NP-hard questions.[2]

In 2006, a novel stochastic optimization model, invasive weed optimization (IWO) algorithm,[3] was proposed by Mehrabian and Lucas, which is inspired from a common phenomenon in agriculture: colonization of invasive weeds. The algorithm has a simple structure, less parameters, strong robustness, easy to understand and easy to program features. At present, IWO as a new optimization method has been successfully applied to the antenna array design,[4] the piezoelectric actuator placement,[5] and constrained engineering design[6] and other issues.

In this paper, the IWO algorithm is used as a solution for solving PFSP. Firstly, weed individuals in IWO are encoded to construct mapping between weed individuals and scheduling solution. Then, the global exploration capacity of IWO is used to select the best fitness value and its corresponding processing sequence of jobs by evaluating the fitness of individuals. The simulation results show that the IWO can effectively solve the PFSP, showing a strong adaptive and robustness.

2. DESCRIPTION OF PFSP

Firstly, the general characterizations of PFSP are as follows: there are n jobs and m machines, all jobs are processed in the same sequence to minimize a given objective function, every job is processed in one machine only once and each machine can only process one job at a time, machines may be idle within the schedule period and at last, all jobs are processed in an identical processing order on all machines. Secondly, mathematical model of PFSP is given in following:

$$
\begin{aligned}
&C(j_1, 1) = t_{j_1,1} \\
&C(j_i, 1) = C(j_{i-1}, 1) + t_{j_i,1}, \quad i = 2, 3, \ldots, n \\
&C(j_1, k) = C(j_1, k-1) + t_{j_1,k}, \quad k = 2, 3, \ldots, m \\
&C(j_i, k) = \max\{C(j_{i-1}, k), C(j_i, k-1)\} + t_{j_i,k} \\
&\qquad i = 2, 3, \ldots, n, \; k = 2, 3, \ldots, m
\end{aligned} \tag{1}
$$

$$\pi^* = \arg\{C_{\max}(\pi) = C(j_n, m)\} \to \min, \quad \forall \pi \in \Pi$$

where, n is the number of all jobs and m is the number of all machines, times of job i processed on machine j is denoted as $t_{i,j}$ $(1 \le i \le n, 1 \le j \le m)$, assuming preparation time for each job is zero or is included in the processing time $t_{i,j}$, $\pi = (j_1, j_2, \ldots, j_n)$ is a scheduling solution, all scheduling solutions are denoted as Π, $C(j_i, k)$ represent the completion time of job j_i processed on machine k.

*Author to whom correspondence should be addressed.

RESEARCH ARTICLE

3. INVASIVE WEED OPTIMIZATION ALGORITHM FOR SOLVING PFSP

3.1. Invasive Weed Optimization Algorithm

In the basic IWO, weeds represent the feasible solutions of problems and population is the set of all weeds. A finite number of weeds are being dispread over the search area. Every weed produces new weeds depending on its fitness. The generated weeds are randomly distributed over the search space by normally distributed random numbers with a mean equal to zero. This process continues until maximum number of weeds is reached. Only the weeds with better fitness can survive and produce seed, others are being eliminated. The process continues until maximum iterations are reached or hopefully the weed with best fitness is closest to optimal solution.

The process is addressed in details as follows:

Step 1: *Initialize a population*. A population of initial solutions is being dispread over the D dimensional search space with random positions.

Step 2: *Reproduction*. The higher the weed's fitness is, the more seeds it produces. The formula of weeds producing seeds is

$$weed_n = \frac{f - f_{\min}}{f_{\max} - f_{\min}}(s_{\max} - s_{\min}) + s_{\min} \tag{2}$$

where, f is the current weed's fitness. $f_{\max}$ and $f_{\min}$ respectively represent the maximum and the least fitness of the current population. $s_{\max}$ and $s_{\min}$ respectively represent the maximum and the least value of a weed.

Step 3: *Spatial dispersal*. The generated seeds are randomly distributed over the D dimensional search space by normally distributed random numbers with a mean equal to zero, but with a varying variance. This ensures that seeds will be randomly distributed so that they abide near to the parent plant. However, standard deviation (σ) of the random function will be reduced from a previously defined initial value (σ_{init}) to a final value (σ_{final}) in every generation. In simulations, a nonlinear alteration has shown satisfactory performance, given as follows

$$\sigma_{\text{cur}} = \frac{(iter_{\max} - iter)^n}{(iter_{\max})^n}(\sigma_{\text{init}} - \sigma_{\text{final}}) + \sigma_{\text{final}} \tag{3}$$

where, $iter_{\max}$ is the maximum number of iterations, σ_{cur} is the standard deviation at the present time step and n is the nonlinear modulation index. Generally, n is set to 3.

Step 4: *Competitive exclusion*. After passing some iteration, the number of weeds in a colony will reach its maximum (P_MAX) by fast reproduction. At this time, each weed is allowed to produce seeds. The produced seeds are then allowed to spread over the search area. When all seeds have found their position in the search area, they are ranked together with their parents (as a colony of weeds). Next, weeds with lower fitness are eliminated to reach the maximum allowable population in a colony. In this way, weeds and seeds are ranked together and the ones with better fitness survive and are allowed to replicate. The population control mechanism also is applied to their offspring to the end of a given run, realizing competitive exclusion.

3.2. Individual Coding Scheme

In this paper, the most position value method (MPV) will be used to transform weed individuals from encoded by float numbers into jobs sequence. Then weed individuals' fitness can be calculated by their corresponding sequence. Suppose that $x = (x_1, x_2, \ldots, x_n)$ is a weed individual, $\pi = (j_1, j_2, \ldots, j_n)$ represent the transformed corresponding jobs sequence, where j_1 the column number of the maximum float in is x, j_2 is the column number of the second largest float in x, and so on. There may existing such situation that many position have the same float value, in this case, the left position is chosen priority. For example, x_1, x_2 have the same float values, so the column number of x_1 will be firstly saved in jobs sequence π, and then is x_2. Suppose $x = [0.06, 2.99, 1.86, 3.73, 1.86, 0.67]$, the following mapping is given by Table I.

3.3. Fitness Calculating

The following method will be selected to calculate individual's fitness. Suppose the number of all jobs is n, m is the total number of machines, $x = (x_1, x_2, \ldots x_n)$ represent individual and its corresponding jobs sequence is $\pi = (j_1, j_2, \ldots, j_n)$, matrix T that include the times of every job processed on each machine are given as following

$$T = \begin{bmatrix} t_{j_1,1}, & t_{j_1,2}, & \ldots, & t_{j_1,m} \\ t_{j_2,1}, & t_{j_2,2}, & \ldots, & t_{j_2,m} \\ \vdots & \vdots & \vdots & \vdots \\ t_{j_n,1}, & t_{j_n,2}, & \ldots, & t_{j_n,m} \end{bmatrix} \tag{4}$$

where $t_{j_i,k}$ denotes the time of job j_i processed on machine k, $1 \le i \le n, 1 \le k \le m$. Fitness of the individual will be represent by the completed time of the last job processed on machine m. the following formula will be obtain

$$t_{j_i,k} = \begin{cases} t_{j_1,1} & \text{if } i = 1,\ k = 1 \\ t_{j_1,k-1} + t_{j_1,k} & \text{if } i = 1,\ k \ne 1 \\ t_{j_{i-1},1} + t_{j_i,1} & \text{if } i \ne 1,\ k = 1 \\ \max(t_{j_{i-1},k}, t_{j_i,k-1}) + t_{j_i,k} & \text{if } i \ne 1,\ k \ne 1 \end{cases} \tag{5}$$

at last, $t_{j_n,m}$ is the fitness value of the individual.

Table I. Locations of individual and its corresponding jobs sequence.

Locations	1	2	3	4	5	6
x	0.06	2.99	1.86	3.73	1.86	0.67
π	4	2	3	5	6	1

RESEARCH ARTICLE

Table II. Parameters settings.

Parameter meaning	Variable	Value
Lower bound of individual	$X\_\min$	−200
Upper bound of individual	$X\_\max$	200
The initial number of population	G_SIZE	10
The maximum number of population	P_MAX	15
The maximum iteration number	$iter\_\max$	500
The initial variance	$step\_\text{ini}$	100
The final variance	$step\_\text{final}$	0.001
The maximum number of seed generated	$seed\_\max$	15
The minimum number of seed generated	$seed\_\min$	1
The nonlinear modulation index	n	4

Table III. Results obtained by IWO and NEH, PSOMA, HGA (1).

			NEH	PSOMA		
Problems	n, m	t^*	RE	BRE	ARE	WRE
Car 1	11.5	7038	0	0	0	0
Car 2	13.4	7166	2.91	0	0	0
Car 3	12.5	7312	1.19	0	0	0
Car 4	14.4	8003	0	0	0	0
Car 5	10.6	7720	1.49	0	0.018	0.375
Car 6	8.9	8505	3.151	0	0.114	0.764

Table III. Results obtained by IWO and NEH, PSOMA, HGA (2).

			HGA			IWO		
Problems	n, m	t^*	BRE	ARE	WRE	BRE	ARE	WRE
Car 1	11.5	7038	0	0	0	0	0	0
Car 2	13.4	7166	0	0.88	2.93	0	0.004	0.029
Car 3	12.5	7312	0	1.05	1.19	0	0.004	0.011
Car 4	14.4	8003	0	0	0	0	0	0
Car 5	10.6	7720	0	0.17	1.45	0	0.003	0.013
Car 6	8.9	8505	0	0.50	0.76	0	0.003	0.007

3.4. Pseudo Code of Solving PFSP by IWO Algorithm

```
BEGIN
Initialize(pop), pop_size = N;
iter = 1;
While iter < iter_max
      job_sequence_weed = sort(pop);
   For i = 1:N
      fitness(i) = −F(job_sequence_weed);
   End For
      BestFitness = max(fitness);
      WorstFitness = min(fitness);
stepLength = (iter_max-iter)^n * (stepLength_ini-
      stepLength_final)/(iter_max)^n * stepLength_final;
   For i = 1:N
   num = (seed_max-seed_min) * (fitness(i)-
      WorstFitness)/(BestFitness-WorstFitness)
      + seed_min;
   seed = normrnd(0, stepLength, num);
      End For
   job_sequence_seed = sort(seed);
   If num(weed, seed) > P_SIZE
   pop = selectBetter(weed, seed, P_SIZE);
      N = P_SIZE;
      Else
   pop = join(weed, seed); N = num(weed, seed);
End If
End While
Output: select the best job_sequence(job_sequence_
   weed or job_sequence_seed), −F(job_sequence);
END
```

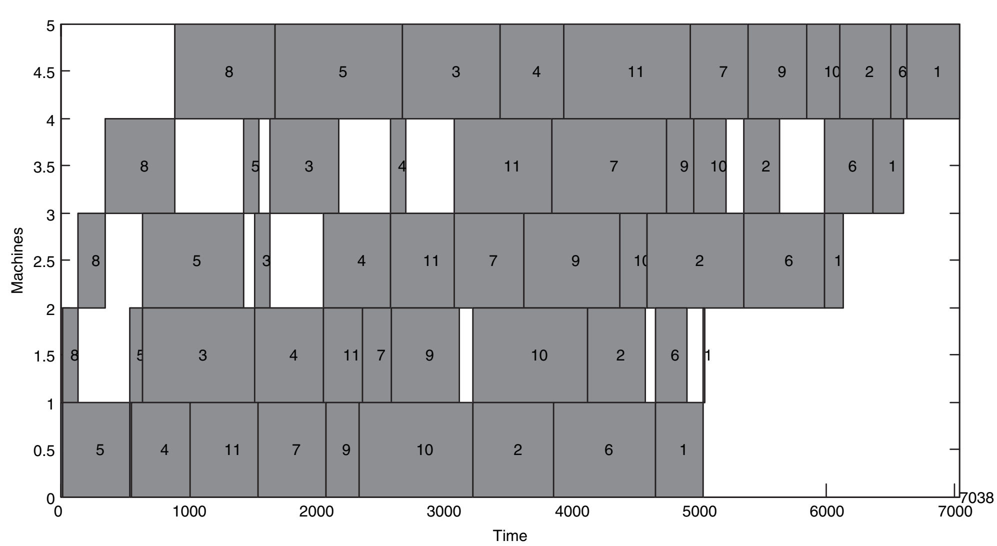

Fig. 1. Gantt chart of Car 1.

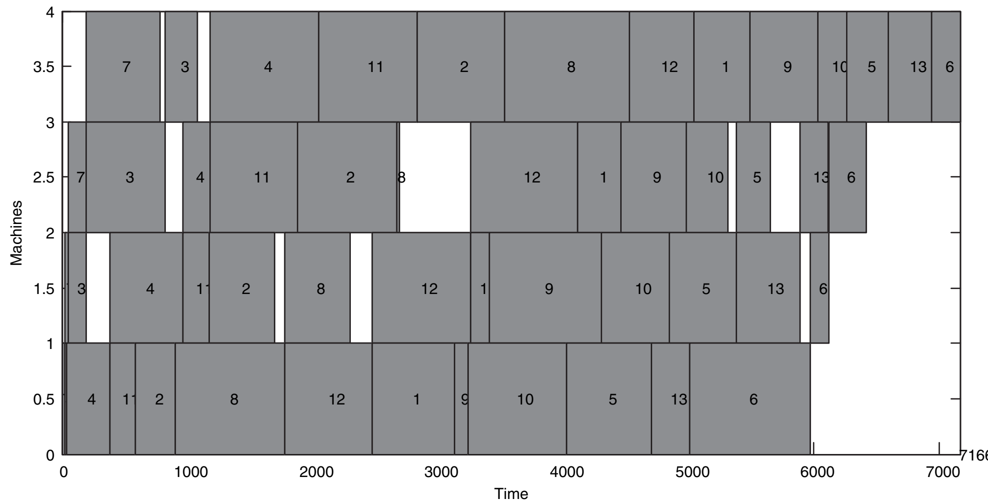

Fig. 2. Gantt chart of Car 2.

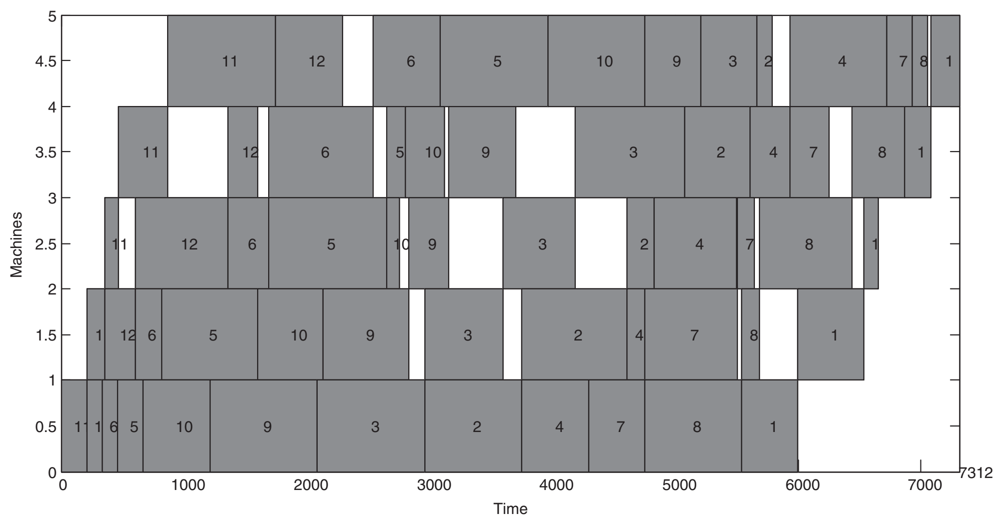

Fig. 3. Gantt chart of Car 3.

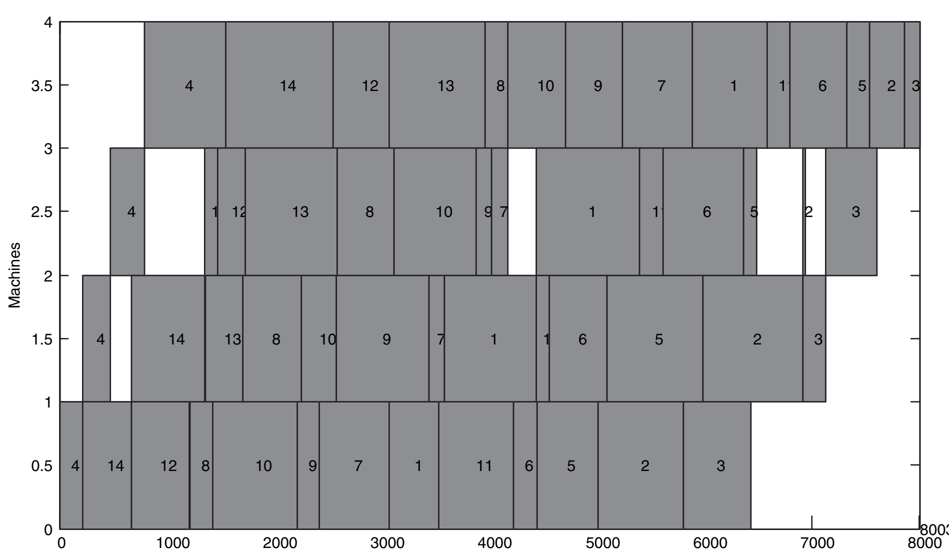

Fig. 4. Gantt chart of Car 4.

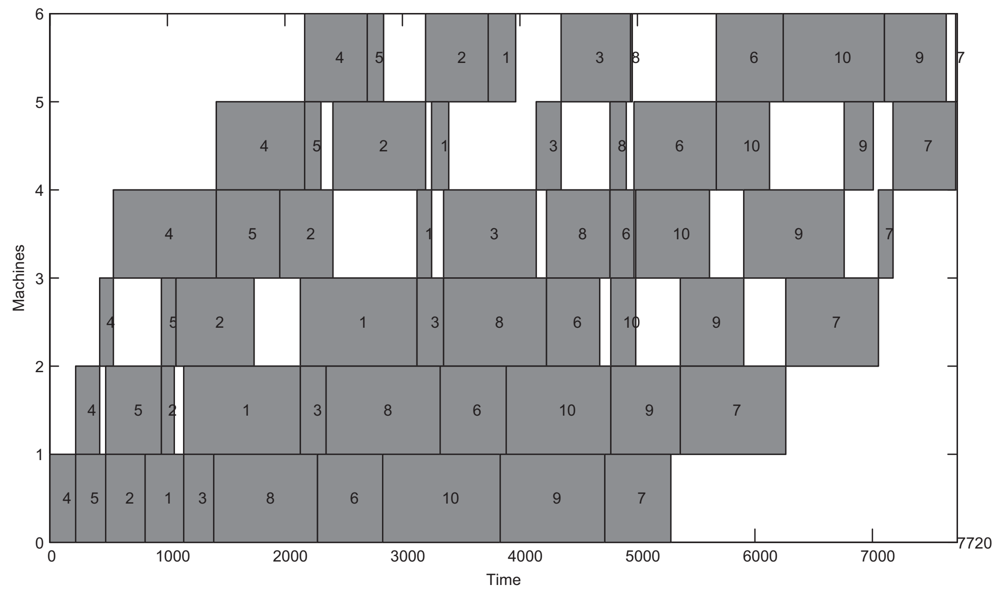

Fig. 5. Gantt chart of Car 5.

4. SIMULATION EXPERIMENTS AND RESULTS ANALYSIS

4.1. Test Platform and Parameters Settings

The experimental program testing platform as: Processor: CPU Intel Core i3-370, Frequency: 2.40 GHz, Memory: 4 GB, Operating system: Windows 7, Run software: Matlab 7.6. Parameters settings are given by Table II as following.

4.2. Testing PFSP and Comparison

In this section, 6 PFSP benchmarks designed by Carlier[21] will be tested and the results obtained by IWO will be compared with literature.[7] Every PFSP benchmark runs 20 times and results are available in Table III. t^* represents the theory Makespan, RE, $(t_{j_n,m} - t^*)/t^* \times 100\%$, represent the relative percentage error of results obtained to t^*, and BRE is the best relative percentage error, ARE is mean

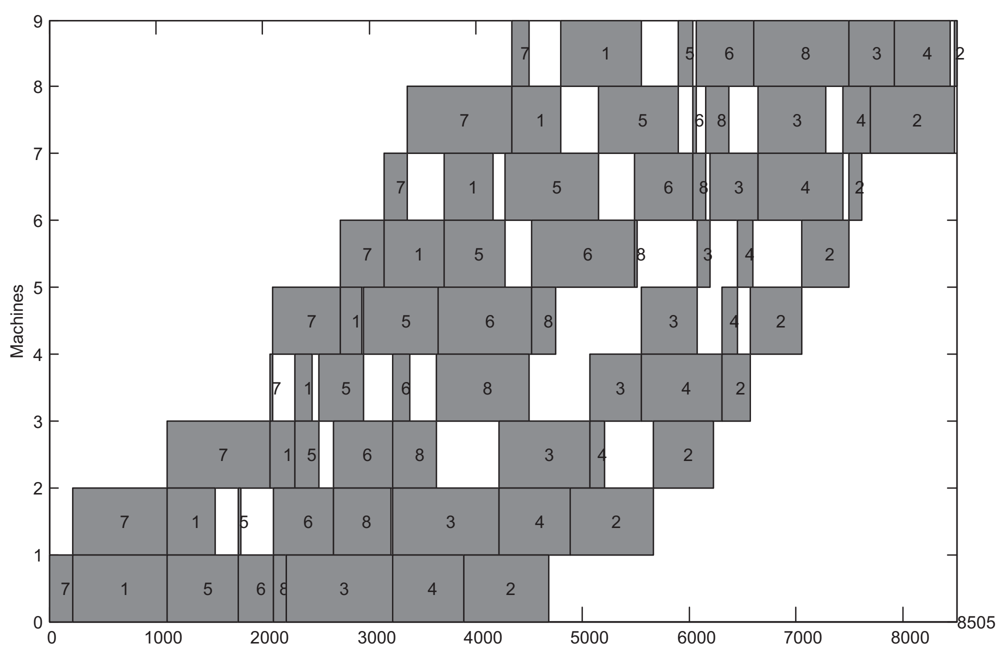

Fig. 6. Gantt chart of Car 6.

RESEARCH ARTICLE

relative percentage error, WRE is the worst relative percentage error.

From Table III(1)–(2), we can see that theory Makespan of 6 PFSP benchmarks all can be obtained by IWO. The ARE and WRE about Car 2 and Car 3 obtained by IWO are worse than that by PSOMA, but are better than NEH and HGA. Results of BRE, ARE and WRE obtained by IWO have achieved the first place compared with NEH, PSOMA and HGA in the remaining six benchmarks' testing.

Gantt charts of the 6 PFSP benchmarks are given from Figures 1 to 6. The best scheduling solution can be got by Gantt chart. For example, for Car 1, the best scheduling sequence is 8, 5, 3, 4, 11, 7, 9, 10, 2, 6, 1, the best Makespan is 7038.

5. CONCLUSIONS

Aiming to the PFSP, MPV method is used to encode individuals of IWO which is originally used to solve continuous space problem so that scheduling sequence can be obtained. Then, the global exploration capacity of IWO is used to select the best fitness value and its corresponding processing solution of jobs by evaluating the fitness of individuals. The simulation results show that the best scheduling sequence can be obtained effectively by IWO which showing a strong adaptive and robustness.

Acknowledgments: This work was supported in part by NSFC under Grant No. 61165015. Key Project of Guangxi Science Foundation (2012GXNSFDA053028) and the funded by open research fund program of key lab of intelligent perception and image understanding of ministry of education of china under Grant (IPIU01201100).

References

1. M. Pinedo, Scheduling Theory, Algorithms and Systems, Englewood Cliffs, Prentice-Hall, NJ **(2002)**.
2. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco **(1979)**.
3. A. R. Mehrabian and C. A. Lucas, *Ecological Informatics* 1, 355 **(2006)**.
4. A. R. Mallahzadeh and H. Oraizi, *Progress in Electromagnetic Research* 70, 137 **(2008)**.
5. A. R. Mehrabian and A. Y. Koma, *Journal of the Franklin Institute* 348, 12 **(2011)**.
6. S. B. Su, J. W. Wang, and L. Zhang, *Journal of University of Science and Technology of China* 39, 885 **(2009)**.
7. L. Wang and B. Liu, Particle Swarm Optimization and Scheduling Algorithms, Tsinghua University Press, Beijing **(2008)**.

Received: 26 April 2012. Accepted: 23 May 2012.