Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 5914360, 13 pages
https://doi.org/10.1155/2018/5914360

Research Article

A New Imperialist Competitive Algorithm for Multiobjective Low Carbon Parallel Machines Scheduling

Zixiao Pan, Deming Lei, and Qingyong Zhang

School of Automation, Wuhan University of Technology, Wuhan 430070, China

Correspondence should be addressed to Qingyong Zhang; qyzhang@whut.edu.cn

Received 22 October 2017; Revised 6 March 2018; Accepted 11 March 2018; Published 16 April 2018

Academic Editor: Erik Cuevas

This paper considers low carbon parallel machines scheduling problem (PMSP), in which total tardiness is regarded as key objective and total energy consumption is a non-key one. A lexicographical method is used to compare solutions and a novel imperialist competitive algorithm (ICA) is presented, in which a new strategy for initial empires is adopted. Some new improvements are also added in ICA to obtain high quality solutions, which are adaptive assimilation, adaptive revolution, imperialist innovation, and alliance and the novel way of imperialist competition. Extensive experiments are conducted to test the search performance of ICA by comparing it with methods from literature. Computational results show the promising advantages of ICA on low carbon PMSP.

1. Introduction

Parallel machine scheduling problem (PMSP) is the typical problem in the actual manufacturing processing [1]. Allocating the manufacturing resources and scheduling the production tasks must be designed to achieve the specified performance. In the traditional PMSP, objectives such as total tardiness have extensively been investigated; however, energy consumption and carbon emission are seldom considered [2, 3]. If there is no consideration on energy related objective, then the corresponding research results are difficult to meet the increasingly stringent requirements of energy conservation and environmental protection on manufacturing activities in China. Thus, it is necessary to focus on low carbon PMSP to improve the traditional scheduling performance and reduce carbon emissions and energy consumption.

In recent years, few papers discussed low carbon PMSP. Several heuristic algorithms has been proposed by Li et al. [4] to minimize makespan or the total completion time under the prerequisite that the total of energy cost and disposal cost is not more than a given threshold. Che et al. [5] gave a mixed integer programming model and two-stage heuristic method on unrelated parallel machine scheduling problem with different power cost. Wang et al. [6] proposed a two-stage algorithm to determine the cutting speed and minimize the maximum completion time of jobs under the given power load peak demand. Liang et al. [3] presented an ant colony optimization (ACO) algorithm to optimize the weighted sum of energy consumption and total tardiness; Li et al. [2] described the model of the problem and proposed 10 kinds of heuristic algorithms.

Meanwhile, many works have been done on multiobjective PMSP by using various metaheuristics. Lin et al. [7] proposed tabu-enhanced iterated Pareto greedy algorithm to minimize total tardiness, makespan, and total weighted completion time. Ying [8] gave a multiobjective multipoint simulated annealing for PMSP with total tardiness, total weighted completion time, and maximum completion time. Li et al. [9] provided a hybrid nondominated sorting genetic algorithm-II (NSGA-II [10]) with fuzzy logic controller and an exact method. Pakzad-Moghaddam [11] presented a Lévy flight embedded particle swarm optimization for PMSP with learning and adapting. Afzalirad and Rezaeian [12] reported a multiobjective ant colony optimization to minimize mean weighted flow time and mean weighted tardiness. Zarandi and Kayvanfar [13] applied two multiobjective metaheuristics for PMSP with total cost of tardiness and maximum completion time.

As mentioned above, low carbon PMSP is seldom studied. The major existing methods for low carbon PMSP are heuristic algorithms. Although metaheuristics have been extensively used to solve multiobjective PMSP without energy related objective, they are hardly applied to optimize low carbon scheduling on parallel machines. To the best of our knowledge, only ACO is found to solve PMSP with energy consumption and other metaheuristics such as imperialist competitive algorithm (ICA) are not used. In addition, the literature on multiobjective PMSP hardly considers the relative importance of objectives. In fact, objectives should be treated differently in many actual manufacturing cases. Take make-to-order production as example, the delay delivery of orders often leads to losses in profits and market reputation; when manufacturers face the high pressure of on-time delivery, it is natural for them to regard on-time delivery as key objective and energy consumption as non-key one, so it is necessary to focus on low carbon PMSP with the consideration on the relative importance of objectives.

ICA is a metaheuristic that mimics sociopolitical imperialist competition for global optimal solution [14]. ICA starts with an initial population, which is a group of countries. The countries with best cost are regarded as imperialists and the rest of countries are called colonies. ICA consists of assimilation of colonies, revolution of colonies, and imperialistic competition among empires and so on. ICA has been successfully applied to solve various optimization problems such as the traveling salesman problem [15], skeletal structure optimization [16], facility layout [17, 18], dynamic economic dispatch scheduling [19], assembly line balancing [20–22], artificial neural network optimization [23], supply chain network design [24], and scheduling [25–34]. However, as stated above, ICA is not used to solve the problem of low carbon parallel machine scheduling. ICA is an effective global optimization method and has strong local search ability with flexible structure [35]. These features make ICA have some advantages in solving low carbon PMSP; for example, it is not necessary to improve local search ability like GA. These things on ICA motivate us to apply ICA to solve low carbon PMSP.

In this study, PMSP with total tardiness (key objective) and total energy consumption (non-key objective) is considered and a novel ICA is proposed for minimizing total tardiness with the consideration on total energy consumption. The lexicographical method is adopted to compare solutions. In ICA, some new strategies are applied to generate high quality solutions, which include the novel method for initial empires, adaptive assimilation, adaptive revolution, imperialist innovation, and alliance and the new imperialist competition. A number of experiments are conducted using many instances. Computational results show the effectiveness and advantages of the new ICA on the low carbon PMSP.

The remainder of the paper is organized as follows. Problem under study and introduction to ICA are described in Sections 2 and 3. ICA for the problem is reported in Section 4. Numerical test experiments on ICA are shown in Section 5 and the conclusions are summarized in the final section and some topics of the future research are provided.

2. Problem Description

Low carbon PMSP is described as follows. There are n independent jobs processed on m unrelated parallel machines. Each job i is available at time zero, has a processing time p_{ij} on machine M_j, $j = 1, 2, \ldots, m$, and a due date d_i. E_j^* indicates energy consumption of each machine M_j per unit time. At any time, each machine can process at most one job and each job can be processed on at most one machine.

Low carbon PMSP is composed of machine assignment subproblem and scheduling subproblem. The former is to decide an appropriate machine for each job and the latter is to obtain the processing sequence of jobs on each machine.

The goal of the problem is to minimize the following two objectives simultaneously:

$$\begin{aligned} \min \quad & f_1 = \sum_{i=1}^{n} \max\{C_i - d_i, 0\} \\ \min \quad & f_2 = \sum_{j=1}^{m} E_j = \sum_{j=1}^{m} \sum_{i=1}^{n} E_j^* p_{ij} x_{ij}, \end{aligned} \tag{1}$$

where f_1 is total tardiness and f_2 is total energy consumption, C_i represents the completion time of job i, E_j is the total energy consumption of machine M_j, and x_{ij} is the 0-1 variable. If job i is processed on machine M_j, x_{ij} is 1; otherwise x_{ij} is equal to 0.

In the literature on multiobjective PMSP, all objectives are often set to have the same importance. In fact, the importance of objectives is not different in many real-life manufacturing situations. For example, in make-to-order production environment, on-time delivery is vital for manufacturers. The delayed delivery will cause loss in revenue, customers, and reputation of company and so on, so it is essential to regard total tardiness as key objective and total energy consumption as non-key one to implement as many on-time deliveries for customers as possible.

The lexicographical method is often utilized to deal with multiple objectives with different importance, which gives key objectives higher priority than non-key ones. When the lexicographical approach is applied, if $f_1(x) < f_1(y)$ or $f_1(x) = f_1(y)$ and $f_2(x) < f_2(y)$, then $x \to y$, which denotes that x substitutes for y. The above conditions mean three cases: (1) $f_1(x) < f_1(y)$, $f_2(x) \leq f_2(y)$; (2) $f_1(x) < f_1(y)$, $f_2(x) > f_2(y)$; (3) $f_1(x) = f_1(y)$, $f_2(x) < f_2(y)$; that is, once $f_1(x) > f_1(y)$, y cannot be replaced with x even if $f_2(x) < f_2(y)$. In this study, the lexicographical method is used to compare solutions.

3. Introduction to ICA

ICA is a population-based metaheuristic. Each individual of population represents a country and some best countries are selected as imperialists in the initialization. After initial

empire is built by using an imperialist and colonies, new solutions are generated by the assimilation and revolution of colonies, the exchange of imperialist and colony if possible, and imperialist competition.

The procedure of ICA [14] is shown as follows.

(1) Initialization: generate an initial population P.

(2) Construct initial empire: compute the cost c_i for each individual; sort c_i in descending order for all solutions; select N_{im} best solutions from P as imperialists; and assign N_{col} remaining countries to the imperialists.

(3) For each empire, execute assimilation of colonies, perform revolution of some colonies, and exchange position of colony and imperialist if possible.

(4) Achieve imperialist competition.

(5) Eliminate the empire without any countries.

(6) If the termination criterion is not met, go to step (3); otherwise, stop the search.

The cost of a country is calculated according to objective function. The better a solution is, the less its cost is. N_{im} best solutions with the lowest cost are chosen as imperialists. The rest of countries are set to be colonies. There are N_{col} colonies; $N_{\text{col}} = N - N_{\text{im}}$. The initial empires are formed by assigning colonies to imperialists according to the power of imperialist

$$P_k = \left| \frac{Y_k}{\sum_{v=1}^{N_{\text{im}}} Y_v} \right|, \tag{2}$$

where P_k is the power of imperialist k and $Y_v = \max_k\{c_k\} - c_v$ indicates the normalized cost, where c_k is the cost of imperialist k.

The number of initial colonies possessed by imperialist k is calculated as round$\{P_k \times N_{\text{col}}\}$, where round is a function that gives the nearest integer of a fractional number. H_k is the set of colonies of imperialist k.

In the assimilation process, a colony in each empire moves ε along with e direction toward its imperialist. The moving distance ε is a random number gotten by random distribution in interval $[0, s \times e]$, where $s \in (1, 2)$ and e is distance among colony and imperialist. Setting $s > 1$ causes the colony to move toward the imperialist direction. However, imperialist cannot absorb their colonies in direct movement, resulting in a deviation from the direct line. The deviation is represented by θ, which follows uniform distribution in $[-\varphi, \varphi]$, where φ is an arbitrary parameter.

Revolution is about a change in position of some colonies because of an unexpected changes in their characteristics. For example, by changing the language or religion of a colony, its characteristics will be changed and, accordingly, its position will be changed [14]. The revolution in ICA is similar to mutation in GA, which increases exploration and prevents the early convergence to local optima.

After assimilation and revolution are done in an empire, the cost of each colony is compared with that of its imperialist and a colony is swapped with the imperialist if the colony has less cost than the imperialist.

Imperialist competition is an important step based on the total power of an empire. Let TC_k be the total cost of empire k; we first calculate TC_k for each empire k by

$$\text{TC}_k = c_k + \zeta \times \text{mean}\left\{\text{Cost}\left(\text{colonies of empire } k\right)\right\}, \tag{3}$$

where ζ is a positive number between 0 and 1 and close to 0.

We then compute the normalized total cost of empire k and the power of empire k by

$$\begin{aligned} \text{NTC}_k &= \max_h \{\text{TC}_h\} - \text{TC}_k, \\ \text{EP}_k &= \left| \frac{\text{NTC}_k}{\sum_{v=1}^{N_{\text{im}}} \text{NTC}_v} \right|. \end{aligned} \tag{4}$$

After a vector $[\text{EP}_1 - s_1, \text{EP}_2 - s_2, \ldots, \text{EP}_{N_{\text{im}}} - s_{N_{\text{im}}}]$ is defined, the weakest colony from the weakest empire is assigned to the empire with the largest index, where s_i denotes a random number following uniform distribution in $[0, 1]$.

4. A Novel ICA for Low Carbon Parallel Machines Scheduling Problem

In general, imperialist is just updated with the evolved colony and seldom changed in other way; moreover, the search on imperialist and the information exchange between imperialists are seldom considered in the existing ICA. Imperialist is good solution in population and the local and global search on these good solutions are easy to produce high quality solution and improve search efficiency, so it is necessary to introduce the innovation and alliance of imperialist to simulate local search and global search of imperialist. Two objectives of the problem have different importance; some new strategies are applied to form initial empires and do imperialist competition to adapt to the above characteristic. A novel ICA is proposed based on the above ideas. In the following, we describe the detailed steps of ICA.

4.1. Coding and Decoding. For PMSP, each job is required to allocate a machine and processing sequence is needed to decide for all jobs on each machine. In this study, two-dimensional coding method is applied to describe a solution of low carbon PMSP. Each solution is represented as two strings.

$$\begin{aligned} S &= \begin{Bmatrix} S_1 \\ S_2 \end{Bmatrix}, \\ S_1 &= \left[\pi_1, \pi_2, \ldots, \pi_n\right], \; S_2 = \left[B_1, B_2, \ldots, B_n\right], \end{aligned} \tag{5}$$

where $S_1 = [\pi_1, \pi_2, \ldots, \pi_n]$ is used to decide processing sequence of jobs, $\pi_i \in \{1, 2, \ldots, n\}$, the second string is for machine assignment, and B_i is the machine allocated to job π_i.

Figure 1 shows a solution of low carbon PMSP with 10 jobs and 5 machines. It can be seen that jobs 2, 6, and 7 are processed on machine M_5 and their processing sequence is 2-7-6.

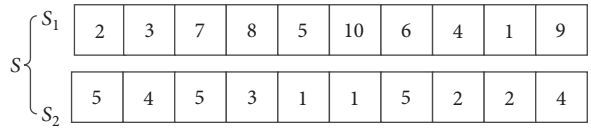


Figure 1: An example of coding strings.

4.2. Initial Empires. After N initial solutions are produced, the cost of all countries is compared and imperialists are chosen, and then each imperialist is assigned colonies. There are two objectives with different importance; we construct initial empires using different ways.

Initial empires are constructed as follows.

(1) For each country $i = 1, 2, \ldots, N$, compute its main cost $c_{1i} = \varepsilon + f_1^i$ and the secondary cost $c_{2i} = f_2^i$, where f_j^i indicates the objective f_j of solution i, $\varepsilon > 0$ is close to zero, and the usage of ε is to make $(\varepsilon + f_1^i) > 0$; then calculate $C_{1i} = 1/c_{1i}$ and $C_{2i} = 1/c_{2i}$.

(2) Sort all countries in the ascending order of the main cost; if more than one country has the same main cost, then sort these countries in the ascending order of the secondary cost. After all countries are sorted, the first N_{im} countries are chosen as imperialists and other countries are colonies; obviously, $N_{\text{col}} = N - N_{\text{im}}$.

(3) Compute the power using the main cost for each imperialist k, $P_k = C_{1k} / \sum_{l=1}^{N_{\text{im}}} C_{1l}$ and allocate colonies to each imperialist according to the procedure in Section 3.

Normalized cost is not computed and the reciprocal of cost is used. The reciprocal of cost has the feature that the bigger the reciprocal of cost is, the better the country is; moreover, the usage of the reciprocal of cost can avoid the zero power of at least one imperialist. For those imperialists with the same main cost, they have the identical power, so they should be allocated the same or similar number of colonies.

4.3. Assimilation. In this section, an adaptive assimilation is used and an adaptive assimilation factor δ is introduced:

$$\delta = e^{-((g-0.5G)/G)^2}, \tag{6}$$

where g and G denote the current generation and the maximum generation, respectively.

The assimilation is shown below for colony $i \in H_k$.

(1) Randomly produce an integer $\alpha \in \{1, 2, \ldots, n\}$; define transfer step $\beta = \text{round}\{\delta \times m/2\}$.

(2) If $(\alpha + \beta) \leq n$, then for the string S_1 of imperialist k, start with the job on position α; suppose that the job is located on the position l of string S_1 of colony i; delete job and machine on the position l of two strings of colony i; repeat the above steps until the job on position $\alpha + \beta$ is considered; if $(\alpha + \beta) > n$, then start with job on position $\alpha - \beta$; the above steps on imperialist k and colony i are executed until the job on position α is considered.

(3) A new solution z is obtained by inserting the segment of imperialist k between α and $\alpha + \beta$ or $\alpha - \beta$ and α into the position of the first deleted job on colony i.

(4) Produce a random number string with length of n, $\text{RS} = \{s_1, s_2, \ldots, s_n\}$.

(5) Start with the first element of RS; if $s_l > \delta$, then B_l of new solution z is replaced with that of imperialist k; repeat the above step until $l = n$.

(6) Lexicographical method is used to compare new solution z and colony i and decide if z can substitute for colony i.

Figure 2 gives an example of the above assimilation. Figure 2(a) is about the first assimilation and Figure 2(b) is the second assimilation.

After assimilation is done in an empire, each colony is compared with its imperialist according to the lexicographical method; if a colony is better than its imperialist, then the colony becomes the new imperialist and the former imperialist turns into a colony.

4.4. Revolution. Generally, assimilation makes colony approximate to imperialist and the similarity between colony and its imperialist increases; as a result, the diversity of population decreases and it is harmful to the search efficiency of ICA. To keep high diversity, new strategies are applied to implement revolution of colonies by using an adaptive revolution probability and probability selection matrix.

Probability selection matrix of imperialist k is defined by

$$Q_k = \begin{bmatrix} q_{k11} & \cdots & q_{k1n} \\ \vdots & \ddots & \vdots \\ q_{km1} & \cdots & q_{kmn} \end{bmatrix}, \tag{7}$$

where q_{kij} indicates the selection probability of job i processed on machine j in imperialist k.

Initially, $q_{kij} = 1/m$; that is, each machine has the same probability of being chosen to process job i. q_{kij} is updated by

$$q_{kij}^{\text{new}} = \theta \times q_{kij} + \lambda \times \rho_{kij}, \tag{8}$$

where $\theta, \lambda \in (0, 1)$ are updating factor, $\theta + \lambda = 1$; if machine j is allocated to job i in imperialist k, then $\rho_{kij} = 1$; otherwise 0.

After all q_{kij}^{new} are obtained, a new probability selection matrix is produced.

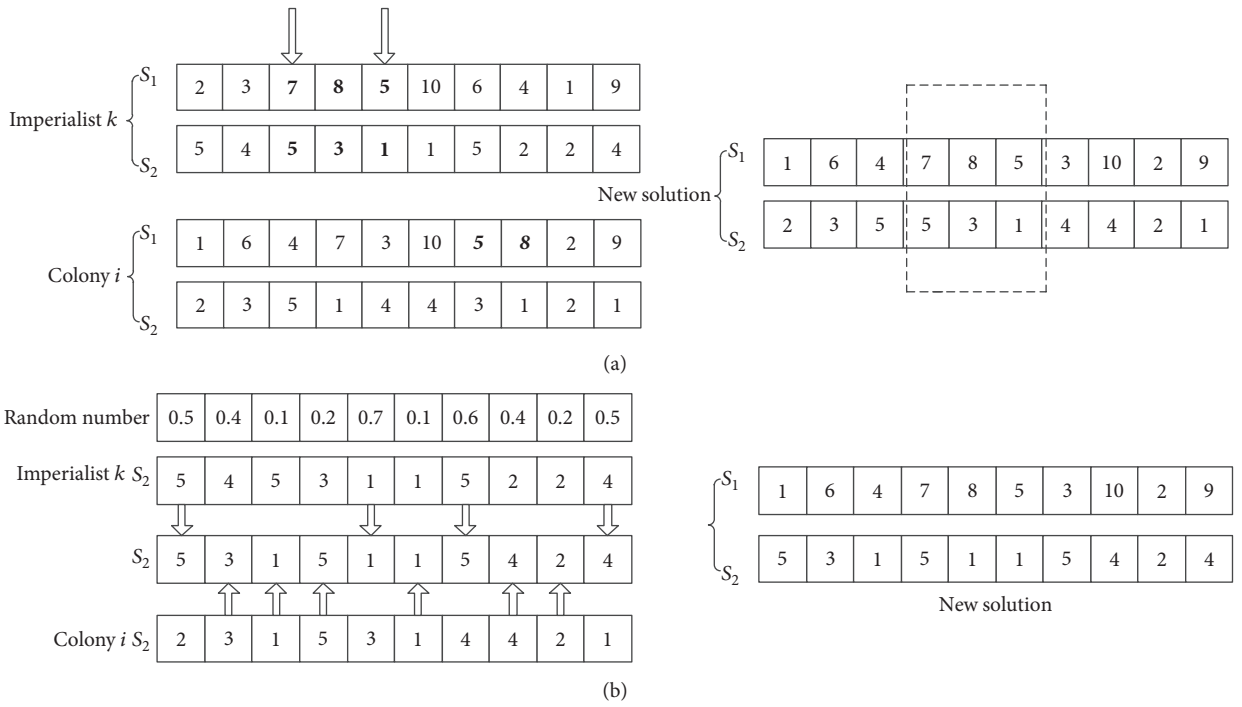


Figure 2: An illustrative example of assimilation.

Revolution probability is fixed in the previous ICA [14–34]; in this study, an adaptive revolution probability is given by

$$\mathrm{pr}_{ki} = \min\left\{\left(\frac{\delta\left(C_{1k} - C_{1i}\right)}{C_{1i}} + \mathrm{rand} \times (1 - \delta)\right), 1\right\}, \quad (9)$$

where pr_{ki} is a revolution probability of colony i of imperialist k; rand is a random number following uniform distribution on interval $[0, 1]$.

The revolution for the colony i of imperialist k is as follows.

(1) Update probability selection matrix of imperialist k.

(2) If pr_{ki} is bigger than a given threshold ϑ, then inverse operator is done on the first string S_1 of colony i; new second string S_2 is generated by probability selection matrix; the newly generated solution directly substitutes for colony i,

where $0 < \vartheta < 1$.

In step (2), the second string S_2 is obtained in the following way: for each job i, let machine j correspond to interval $(\sum_{l=1}^{j-1} q_{kil}, \sum_{l=1}^{j} q_{kil}]$; start with job $i = 1$; if a random number rand $\in (\sum_{l=1}^{j-1} q_{kil}, \sum_{l=1}^{j} q_{kil}]$ is obtained, then machine j is allocated to job i; repeat the above step until each job is assigned a machine. Inverse operator is used to reallocate jobs between two random positions in a reverse order.

In the above procedure, no comparison between the new solution and colony before colony is replaced; as a result, the diversity of population improves.

4.5. Imperialist Innovation and Alliance. Imperialist is just updated by assimilation and revolution and the updating rate of imperialist is limited. Imperialist is often good solution in population and the search on good solution is easier to produce high quality solution, so the innovation and alliance of imperialists are introduced to improve the updating rate of imperialist.

Alliance is done between two imperialists and is described below.

(1) Sort all imperialists according to their main cost; let $k = 1$.

(2) Repeat the following steps until no pair of imperialist is chosen:

For imperialists $N_{\mathrm{im}} - k, k$, suppose that imperialist $N_{\mathrm{im}} - k$ is better than imperialist k; execute the operator between them like the first assimilation; a new solution z is obtained; solution z is compared with imperialist k in terms of the lexicographical method, and imperialist k is replaced with z if the condition is met, $k = k + 1$.

Innovation is done for each imperialist and described as follows.

(1) Randomly generate two integers $a < b$, $a, b \in \{1, 2, \ldots, n\}$.

(2) In two strings of imperialist k, reverse jobs and machines between a and b.

(3) Start with B_i $(i = 1)$ of S_2 of imperialist k; produce a random integer number rand $\in [1, 10]$ and if rand $> \mu$,

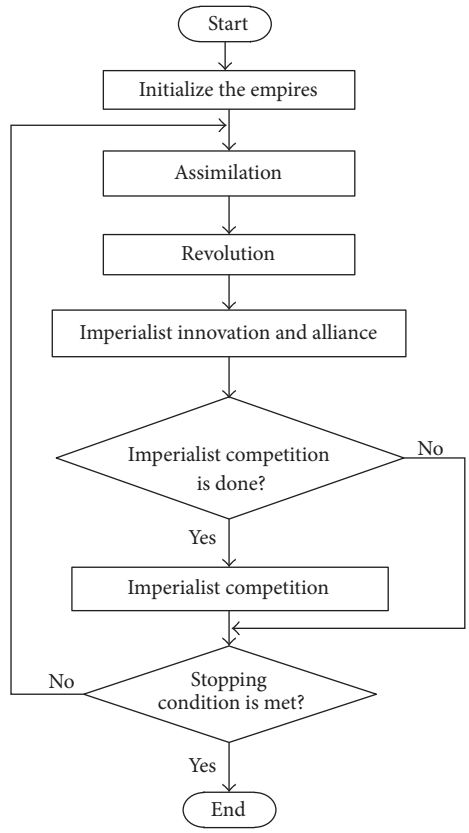


Figure 3: The flowchart of new ICA.

then assign a new machine for B_i. A new solution z is obtained.

(4) The lexicographical method is used to decide if imperialist k can be replaced with z,

where μ is an integer.

We decide $\mu = 5$ based on a number of experiments. When rand is chosen from $[1, 10]$, the condition rand $> \mu$ means that B_i is changed with the possibility of 50%.

4.6. Imperialist Competition. Imperialist competition is the reallocating process of colonies. However, if imperialist competition is done in high frequency, N_{im} empires can be easily combined into one empire and prematurity may occur, so, in this study, imperialist competition is not done in each generation and is executed every g_{ic} generations. In this study, $g_{ic} = 50$ based on a number of experiments.

There are main cost and second cost for a country, so we calculate TC_{1k} and TC_{2k}, which is defined by

$$\begin{aligned} \text{TC}_{1k} &= (1-\eta)\, C_{1k} + \frac{\eta \sum_{i \in H_k} C_{1i}}{|H_k|}, \\ \text{TC}_{2k} &= (1-\eta)\, C_{2k} + \frac{\eta \sum_{i \in H_k} C_{2i}}{|H_k|}, \end{aligned} \tag{10}$$

where TC_{lk} is the weighted sum of C_{lk} of imperialist k and the average C_{li} of all colonies of imperialist k, $l = 1, 2$. η is a real number; we set $\eta = 0.1$ by a number of test experiments.

When the condition of imperialist competition is met, imperialist competition is executed as follows.

(1) Calculate TC_{1k}, TC_{2k} and NTC_{1k}, NTC_{2k} of each empire $k = 1, 2, \ldots, N_{\text{im}}$.

(2) Compute power $\text{TP}_{1k} = \text{NTC}_{1k} / \sum_{l=1}^{N_{\text{im}}} \text{NTC}_{1l}$ and $\text{TP}_{2k} = \text{NTC}_{2k} / \sum_{l=1}^{N_{\text{im}}} \text{NTC}_{2l}$.

(3) Produce a vector $R = [r_1, r_2, \ldots, r_{N_{\text{im}}}]$, where r_i, $i = 1, 2, \ldots, n$ follows uniform distribution on $[0, 1]$.

(4) Define vectors $D_1 = [\text{TP}_{11} - r_1, \text{TP}_{12} - r_2, \ldots, \text{TP}_{1N_{\text{im}}} - r_{\text{im}}]$ and $D_2 = [\text{TP}_{21} - r_1, \text{TP}_{22} - r_2, \ldots, \text{TP}_{2N_{\text{im}}} - r_{\text{im}}]$.

(5) Decide the maximum element in vector D_1; if only one element, for instance, the kth element in D_1, reaches the maximum value, then allocate the weakest colony from the weakest empire into empire k; if more than one element is equal to the maximum value, for example, the k_1th element and k_2th element, then compare the corresponding elements in D_2; if the k_1th element is bigger than k_2th element, then allocate the weakest colony from the weakest empire into empire k_1,

where $\text{NTC}_{1k} = \text{TC}_{1k} - \min_l\{\text{TC}_{1l}\}$, $\text{NTC}_{2k} = \text{TC}_{2k} - \min_l\{\text{TC}_{2l}\}$.

4.7. Algorithm Description. Figure 3 describes the flowchart of our ICA. The termination condition is the maximum

number of objective function evaluations. Compared with the basic ICA, our ICA has the following new features. (1) Imperialist innovation and alliance are introduced to make full use of the advantages of imperialists as good solutions. (2) Adaptive mechanism is used in assimilation and revolution. (3) Revolution is implemented in a new way; there is no comparison between the obtained new solution and colony and the new solution directly substitutes for colony.

5. Computational Experiments

To test the performance of ICA for the low carbon PMSP, extensive experiments are conducted on a set of problems. All experiments are implemented by using Matlab2015b and run on 4.0 G RAM 2.20 GHz CPU PC.

5.1. Test Instances and Comparative Algorithms. 31 instances are randomly produced, in which p_{ij} is from interval [1, 100], energy consumption of machine per unit time E_j^* is selected from [1, 50], and due date d_i is set to $\rho \max_{j=1,2,\ldots,m} p_{ij}$, $\rho = 0.3$.

As stated above, only ACO [3] is applied to solve low carbon PMSP, so we first choose ACO [3] as comparative algorithm. In ACO [3], total tardiness and energy consumption are combined into one objective by the weighted method. We revised ACO as follows: lexicographical method is used to compare solutions and pheromone is updated according to key objective. Jin et al. [36] reported a multiobjective memetic algorithm (MOMA) for integrated process planning and scheduling. MOMA can be directly used to solve the considered PMSP, so we apply MOMA as another comparative algorithm. Meanwhile, in order to prove the effectiveness of the new ICA, we construct ICA1 as a comparative algorithm without adaptive assimilation factor, the adaptive revolution probability, or imperialist innovation and alliance. On the other hand, some metaheuristics such as the famous NSGA-II cannot be directly used to solve our problem because no relative importance of objectives is adopted in these algorithms.

5.2. Results and Discussions. New ICA has four parameters: N (population size), N_{im} (number of imperialist), ϑ (threshold of revolution rate), and G (maximum generation). We obtained the following settings for parameters: $N = 100$, $N_{\text{im}} = 10$, and $\vartheta = 0.98$ and results on G are shown in Table 1.

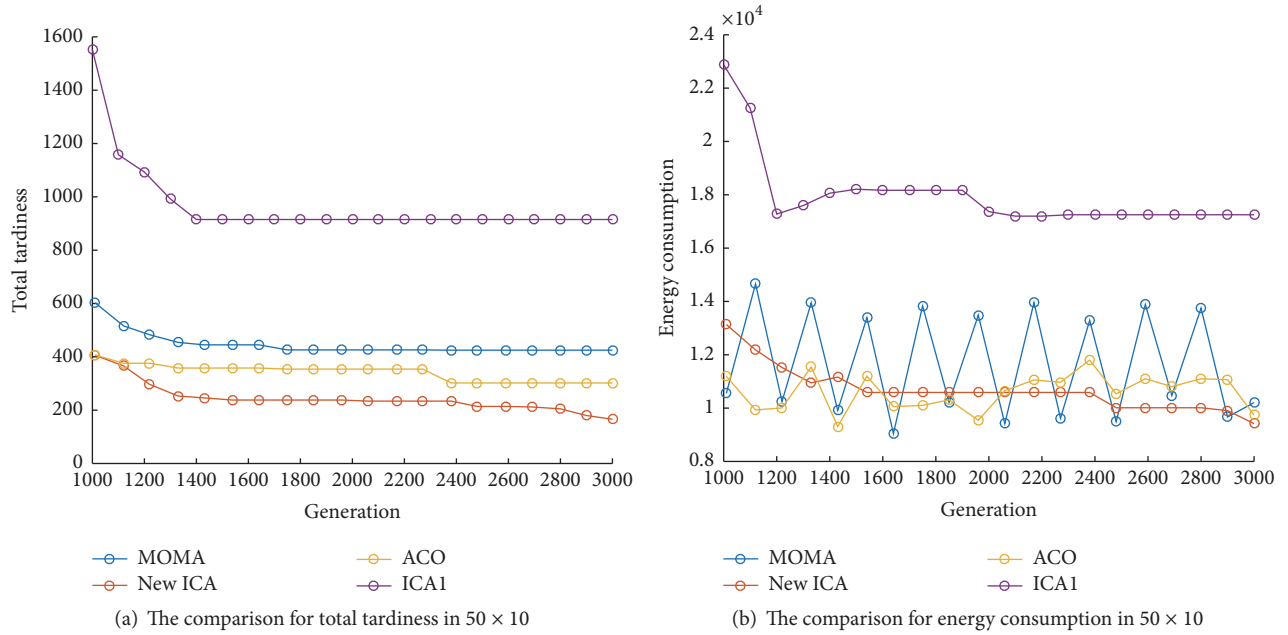
We directly adopt all settings of parameters of ACO proposed by Liang et al. [3] except stopping condition. The parameter settings of MOMA are chosen from Jin et al. [36] except that the termination criterion and the parameter settings of ICA1 are the same as the new ICA. Four algorithms have the same stopping condition: when the maximum generation is reached, the search is stopped. Each algorithm randomly runs 20 times for each instance. The computational results are shown in Tables 2, 3, and 4. In each run an algorithm outputs a final solution: BEST indicates the best final solution founded, WOR denotes the worst final solution in 20 runs, and AVG is the average value of 20 final solutions. Table 5 reports the average running time of four algorithms. Figure 4 shows the comparison of four algorithms for convergence.

Table 1: Maximum generation of all instances.

$m \times n$	G
10 × 5	1000
20 × 5	1000
20 × 8	1000
30 × 5	2000
30 × 8	2000
30 × 10	2000
50 × 5	3000
50 × 8	3000
50 × 10	3000
50 × 12	3000
80 × 8	4000
80 × 10	4000
80 × 12	4000
80 × 15	4000
100 × 10	5000
100 × 12	5000
100 × 15	5000
120 × 10	6000
120 × 12	6000
120 × 15	6000
150 × 10	7000
150 × 15	7000
150 × 20	7000
180 × 10	8000
180 × 15	8000
180 × 20	8000
200 × 10	9000
200 × 15	9000
200 × 20	9000
220 × 15	10000
220 × 20	10000

As shown in Table 2, when the best solutions of four algorithms are compared with the lexicographical method, it can be found that BEST of ICA is better than that of MOMA, ACO, and ICA1 on 30 of 31 instances; moreover, with respect to best solution, two objectives of ICA are less than or equal to those of other three algorithms on 21 instances. Obviously, new ICA converges better than MOMA, ACO, and ICA1. It also can be seen that WOR of ICA is also better than that of other algorithms on 29 instances. The worst final solution of ICA has smaller objectives than or identical objective value with that of MOMA, ACO, and ICA1 on 18 instances, so we can conclude that new ICA performs better than its comparative algorithms in stability. The results in Table 4 also reveal that new ICA is also superior to other three algorithms in average performance; in a word, new ICA can provide better results than MOMA, ACO, and ICA1 using less computational times.

It can be found from Tables 2 and 3 that when new ICA obtains smaller total tardiness than MOMA, ACO, and ICA1, total energy consumption generated by ICA is often



(a) The comparison for total tardiness in 50 × 10

(b) The comparison for energy consumption in 50 × 10

Figure 4: The comparison of four algorithms for convergence.

Table 2: Comparisons among MOMA, ACO, ICA1, and new ICA on BEST.

$m \times n$	MOMA		ACO		ICA1		New ICA	
	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
10 × 5	65	4284.8	56	4316.5	56	4316.5	56	4316.5
20 × 5	459	6357.5	299	4979.1	285	5095.2	**283**	**4979.1**
20 × 8	64	9017.8	34	8732.8	94	10595.9	**32**	8790.1
30 × 5	1003	13959.3	817	13228.5	701	12838.2	**690**	14352.3
30 × 8	240	9133.2	199	9363.0	252	11925.5	**155**	9363.0
30 × 10	173	6969.1	131	7356.4	234	8543.3	**97**	7063.8
50 × 5	3673	14550.2	3043	17291.1	2848	19014.1	**2104**	16113.8
50 × 8	826	15724.8	617	14706.7	1430	21788.4	**355**	**14119.3**
50 × 10	355	10224.1	276	9760.1	779	17857.8	**156**	9836.1
50 × 12	197	15869.4	190	15325.0	582	23658.3	**61**	**14486.2**
80 × 8	3160	20287.4	3257	20065.4	4413	32267.7	**1907**	**19370.0**
80 × 10	1197	16631.4	1303	10504.0	2937	25036.6	**621**	10718.7
80 × 12	864	22537.9	896	22677.1	2187	39664.3	**375**	**20038.8**
80 × 15	1073	19746.7	343	18324.4	1919	51067.8	**115**	**17289.6**
100 × 10	2622	21530.8	3372	24818.4	4756	34906.4	**1918**	21792.5
100 × 12	1917	29604.6	1697	26113.0	4290	56967.8	**826**	**24302.9**
100 × 15	658	18938.2	911	20650.6	4131	52051.7	**345**	**18906.4**
120 × 10	4405	40114.0	4450	35485.3	9308	76955.5	**2344**	**32028.0**
120 × 12	2595	27478.8	3033	28249.0	5842	56348.2	**1325**	**26127.6**
120 × 15	1028	22847.1	1104	21434.9	6272	76945.4	**391**	**19089.7**
150 × 10	8522	51214.5	9486	54150.3	18380	102872.3	**4785**	**44933.8**
150 × 15	2554	32634.0	2775	33334.5	10905	99357.3	**1270**	**28796.0**
150 × 20	435	21052.4	723	21349.2	7844	97740.7	**186**	**20600.5**
180 × 10	8773	45778.1	11599	48224.9	28093	124923.7	**5854**	**42417.5**
180 × 15	3450	34563.6	4590	34576.2	18005	115186.6	**2120**	**30726.2**
180 × 20	1159	25442.1	1613	24713.5	11510	110158.3	**515**	24968.7
200 × 10	12403	67204.2	17142	65871.5	36559	169238.0	**8303**	**57896.8**
200 × 15	4921	31526.7	6674	32713.7	22069	102733.5	**2984**	**29347.2**
200 × 20	2378	35482.6	2934	33490.5	14639	125260.9	**1106**	**30533.7**
220 × 15	5451	57772.2	7312	52861.4	30813	187739.9	**3079**	**45814.6**
220 × 20	2399	28342.2	3017	28682.6	20876	122877.3	**1036**	**25557.8**

Table 3: Comparisons among MOMA, ACO, ICA1, and new ICA on WOR.

$m \times n$	MOMA		ACO		ICA1		New ICA	
	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
10 × 5	65	4284.8	58	3978.7	104	5567.2	59	3562.7
20 × 5	509	6621.4	345	5026.6	580	5470.6	**301**	5090.9
20 × 8	87	8790.1	50	9282.9	307	13706.5	59	8882.9
30 × 5	1149	15091.0	906	13940.9	1063	17773.4	**704**	**12740.3**
30 × 8	283	9367.8	261	9097.7	572	15152.1	**195**	9154.0
30 × 10	146	7059.2	156	7640.0	712	13399.8	**128**	8048.4
50 × 5	3864	14856.1	3365	18781.4	4790	26693.8	**2164**	16395.0
50 × 8	911	17356.0	725	13732.6	2754	33195.8	**400**	13861.2
50 × 10	403	10306.2	356	9656.7	2019	27188.6	**193**	9783.6
50 × 12	359	17569.1	229	15118.9	1659	36914.3	**116**	16647.7
80 × 8	3403	20247.4	3492	19547.1	7946	38695.1	**2064**	**19526.6**
80 × 10	1310	14868.1	1389	12914.2	6092	33508.8	**768**	**12133.7**
80 × 12	1062	24214.1	990	23511.2	4480	53144.3	**474**	**23154.6**
80 × 15	1293	31052.0	395	19309.4	3968	68440.8	**197**	**19277.9**
100 × 10	2830	23633.6	3511	23270.9	9211	58114.9	**2063**	**21944.6**
100 × 12	2348	35228.8	1857	25752.1	8102	77000.0	**938**	**24496.9**
100 × 15	827	20484.6	944	18988.6	6582	68940.3	**453**	20752.6
120 × 10	4754	44360.3	4807	37446.8	15357	107062.8	**2531**	**33887.4**
120 × 12	3956	31656.2	3372	29914.8	13505	80182.4	**1505**	**27595.6**
120 × 15	1617	29421.0	1207	22810.1	10015	99105.3	**535**	**22093.5**
150 × 10	9185	55098.9	10230	53571.8	30227	132490.7	**5212**	**48209.9**
150 × 15	2822	33247.7	3066	34228.2	20336	138752.4	**1455**	33806.7
150 × 20	714	22619.7	811	21343.0	13539	129451.0	**285**	22320.0
180 × 10	9683	56680.2	12490	48304.2	46450	179610.1	**6226**	**43798.2**
180 × 15	4468	41928.6	5017	37636.6	25597	148557.0	**2497**	**34139.7**
180 × 20	1569	30121.8	1842	26161.1	19943	153944.5	**685**	26339.8
200 × 10	13439	79430.0	17822	67572.5	54463	199616.5	**8687**	**58893.9**
200 × 15	6084	35743.2	7162	34106.5	34111	113485.7	**3262**	**30318.8**
200 × 20	3125	43934.0	3118	34478.9	25433	168075.6	**1301**	**32160.0**
220 × 15	6490	62264.3	7729	52519.3	46779	238774.3	**3299**	**47618.0**
220 × 20	2829	31359.5	3382	29615.7	32229	165592.3	**1248**	**28203.8**

less than the corresponding results of MOMA, ACO, and ICA1. This feature shows that total energy consumption really diminishes with the optimization of total tardiness. The lexicographical method is effective.

A nonparametric significance test known as Wilcoxon signed-rank test is conducted [37, 38]. In this test, a null hypothesis is assumed: there is no difference between the median of the solutions achieved by two compared algorithms. The significance level is $\alpha = 0.05$. p value is produced by Wilcoxon signed-rank test for the pairwise comparison of two groups. If p value is smaller than or equal to significance level α, the null hypothesis is rejected. Experimental results of Wilcoxon signed-rank test for new ICA and other algorithms are shown in Tables 6, 7, and 8.

As shown in Tables 6, 7, and 8, p value of Wilcoxon signed-rank test is 0.001. The results show that new ICA performs significantly better than MOMA, ACO, and ICA1 in statistical meaning.

The good performance of new ICA mainly results from its inclusion of imperialist innovation and alliance and the adaptive implementation of assimilation and revolution. These strategies intensify global search ability and improve local search ability; moreover, the search of imperialist is reinforced; as a result, the good balance between exploration

Table 4: Comparisons among MOMA, ACO, ICA1, and new ICA on AVG.

$m \times n$	MOMA		ACO		ICA1		New ICA	
	f_1	f_2	f_1	f_2	f_1	f_2	f_1	f_2
10 × 5	65.0	4284.8	56.4	4299.6	68.1	4144.9	57.0	4090.4
20 × 5	486.0	5676.6	331.5	5006.6	390.8	5288.5	**292.1**	**4976.3**
20 × 8	71.2	8852.8	40.9	8772.5	166.3	113187	42.8	8815.5
30 × 5	1038.5	14091.6	864.7	13590.1	865.2	14598.2	**693.1**	**13229.3**
30 × 8	261.3	9382.7	234.4	9702.1	373.3	12442.4	**169.2**	9817.7
30 × 10	160.5	7492.1	144.2	7429.6	466.1	12116.8	**113.2**	7580.5
50 × 5	3783.3	16170.0	3212.2	16624.5	3745.1	21015.5	**2122.5**	**15361.4**
50 × 8	867.0	16640.3	681.7	14456.6	1964.6	27652.4	**379.2**	**14215.9**
50 × 10	378.0	10193.2	327.0	9917.7	1237.8	22180.8	**174.3**	9939.3
50 × 12	225.6	15917.4	210.9	15568.1	1000.85	28624.0	**91.3**	15580.1
80 × 8	3281.2	20688.2	3353.5	20922.4	5821.7	34995.0	**1960**	**19405.7**
80 × 10	1262.8	13977.3	1341.8	12847.2	4251.4	31254.6	**655.5**	**11552.4**
80 × 12	951.7	22710.6	941.7	22686.0	3291.4	48613.3	**416.1**	**20696.4**
80 × 15	1160.0	30869.8	380.9	18989.3	2987.9	62847.9	**163.6**	**18456.3**
100 × 10	2675.1	23006.5	3442.7	23513.0	7412.3	50818.8	**1972.7**	**22049.0**
100 × 12	2005.0	29498.0	1799.4	26628.0	6143.1	67251.9	**869.1**	**24351.5**
100 × 15	750.9	19579.8	950.5	19980.4	5321.6	60929.4	**405.7**	**19255.9**
120 × 10	4479.1	39922.6	4707.4	35942.0	12306.8	91298.3	**2416.6**	**32750.1**
120 × 12	2890.0	29353.2	3270.4	29007.4	9670.1	71468.9	**1406.2**	**26648.1**
120 × 15	1340.0	25829.9	1142.6	21910.3	7699.5	85988.5	**434.7**	**19997.7**
150 × 10	8872.0	52160.2	9851.8	51504.6	24125.9	118001.4	**4895.2**	**45484.0**
150 × 15	2688.6	34085.4	2956.4	32941.1	14207.1	115536.4	**1339.4**	**30486.1**
150 × 20	576.6	22785.5	770.1	21811.6	10505.2	114924.6	**231.8**	**20557.3**
180 × 10	9223.9	51447.5	12186.2	48382.6	33668.1	135846.2	**6006.9**	**43092.3**
180 × 15	3855.4	38299.9	4844.6	36326.4	21764.8	127503.5	**2274.0**	**32632.4**
180 × 20	1323.8	28139.1	1757.2	26291.4	15884.7	132815.9	**579.9**	**24413.5**
200 × 10	12727.7	70269.8	17488	67196.6	42108.6	171637.6	**8687.0**	**59111.7**
200 × 15	5581.1	35524.7	7020.9	33513.2	28278.5	115073.2	**3086.6**	**29394.6**
200 × 20	2759.7	37720.7	3019.1	34398.4	20894.1	150294.9	**1199.6**	**31177.5**
220 × 15	5962.6	58193.9	7539.0	53527.8	37930.3	215962.5	**3213.7**	**47173.8**
220 × 20	2562.9	28535.4	3244.6	29012.2	26132.5	144387.1	**1125.2**	**25759.7**

and exploitation is kept; thus, we can conclude that new ICA is a very competitive method for low carbon PMSP.

6. Conclusions

PMSP has been extensively considered in the past decade; however, these results are mainly about the optimization of time related indices such as total tardiness. Energy related objective is seldom coped with in parallel machines environments. In this study, low carbon PMSP is studied, in which total tardiness is regarded as key objective and total energy consumption is non-key one. A lexicographical method is used to compare solutions and a novel ICA is presented by using new strategy for forming initial empires, an adaptive assimilation factor and an adaptive revolution, imperialist innovation, and alliance and the way to imperialist competition. Extensive experiments are conducted and ICA is compared with ACO and MOMA. The results validate that new ICA has promising advantages on low carbon PMSP.

We will continue to focus on low carbon PMSP in the near future; for example, we pay attention to PMSP with energy consumption constraint and find a good way to deal with the energy constraint. The applications of ICA to other production scheduling problems such as distributed scheduling are also our future main topics.

Table 5: Running times of four algorithms.

$m \times n$	Average running time (s)			
	MOMA	ACO	ICA1	New ICA
10 × 5	5.19	1.25	4.63	3.80
20 × 5	6.14	5.31	4.22	4.42
20 × 8	6.05	5.00	4.97	**4.46**
30 × 5	15.11	18.13	10.48	**10.33**
30 × 8	13.69	17.50	11.63	**10.35**
30 × 10	13.69	18.13	11.13	**10.51**
50 × 5	25.73	67.50	21.33	**20.78**
50 × 8	25.64	67.59	20.24	**20.00**
50 × 10	25.73	66.56	21.22	**19.98**
50 × 12	25.88	66.46	20.31	**20.04**
80 × 8	44.55	211.25	34.65	36.07
80 × 10	45.05	217.50	33.07	36.70
80 × 12	44.72	212.50	37.87	**36.08**
80 × 15	44.42	216.22	30.89	35.95
100 × 10	66.75	418.70	45.36	54.23
100 × 12	64.84	410.94	51.27	52.14
100 × 15	64.56	409.38	51.11	52.17
120 × 10	88.80	682.50	72.84	75.63
120 × 12	87.44	701.22	60.21	73.65
120 × 15	87.08	697.55	74.18	**73.39**
150 × 10	122.31	1251.30	107.39	**101.70**
150 × 15	124.17	1246.91	85.04	104.15
150 × 20	122.83	1231.56	97.04	115.22
180 × 10	165.88	2235.00	107.81	145.10
180 × 15	164.14	2060.00	107.14	141.40
180 × 20	167.72	2030.07	109.95	138.26
200 × 10	202.55	2798.43	138.81	182.48
200 × 15	202.05	2854.68	129.42	177.57
200 × 20	200.72	2843.42	156.61	177.95
220 × 15	241.42	3743.75	189.67	202.34
220 × 20	241.44	3781.25	155.95	201.76

Table 6: Experimental results of Wilcoxon signed-rank test for new ICA and ACO.

New ICA-ACO	Number	Mean rank	Sum of ranks
Negative ranks	30	15.50	465.00
Positive ranks	0	0.00	0.00
Ties	1		
Total	31		
	p_value = 0.001		

Table 7: Experimental results of Wilcoxon signed-rank test for new ICA and MOMA.

New ICA-MOMA	Number	Mean rank	Sum of ranks
Negative ranks	31	16.00	496.00
Positive ranks	0	0.00	0.00
Ties	0		
Total	31		
	p_value = 0.001		

Table 8: Experimental results of Wilcoxon signed-rank test for new ICA and ICA1.

New ICA-ICA1	Number	Mean rank	Sum of ranks
Negative ranks	30	15.50	465.00
Positive ranks	0	0.00	0.00
Ties	1		
Total	31		
	p_value = 0.001		

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

References

[1] T. C. E. Cheng and C. C. S. Sin, "A state-of-the-art review of parallel-machine scheduling research," *European Journal of Operational Research*, vol. 47, no. 3, pp. 271–292, 1990.

[2] Z. Li, H. Yang, S. Zhang, and G. Liu, "Unrelated parallel machine scheduling problem with energy and tardiness cost," *The International Journal of Advanced Manufacturing Technology*, vol. 84, no. 1-4, pp. 213–226, 2016.

[3] P. Liang, H.-D. Yang, G.-S. Liu, and J.-H. Guo, "An ant optimization model for unrelated parallel machine scheduling with energy consumption and total tardiness," *Mathematical Problems in Engineering*, vol. 2015, Article ID 907034, 8 pages, 2015.

[4] K. Li, X. Zhang, J. Y.-T. Leung, and S.-L. Yang, "Parallel machine scheduling problems in green manufacturing industry," *Journal of Manufacturing Systems*, vol. 38, pp. 98–106, 2016.

[5] 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," *Journal of Cleaner Production*, vol. 129, pp. 565–577, 2016.

[6] Y.-C. Wang, M.-J. Wang, and S.-C. Lin, "Selection of cutting conditions for power constrained parallel machine scheduling," *Robotics and Computer-Integrated Manufacturing*, vol. 43, pp. 105–110, 2017.

[7] S.-W. Lin, K.-C. Ying, W.-J. Wu, and Y.-I. Chiang, "Multi-objective unrelated parallel machine scheduling: A Tabu-enhanced iterated Pareto greedy algorithm," *International Journal of Production Research*, vol. 54, no. 4, pp. 1110–1121, 2016.

[8] K. C. Ying, "A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems," *International Journal of Production Research*, vol. 53, pp. 1065–1076, 2015.

[9] X. Li, F. Yalaoui, L. Amodeo, and H. Chehade, "Metaheuristics and exact methods to solve a multiobjective parallel machines scheduling problem," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1179–1194, 2012.

[10] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Parallel Problem Solving from Nature PPSN VI*, vol. 1917 of *Lecture Notes in Computer Science*, pp. 849–858, Springer, Berlin, Germany, 2000.

[11] S. H. Pakzad-Moghaddam, "A Lévy flight embedded particle swarm optimization for multi-objective parallel-machine scheduling with learning and adapting considerations," *Computers & Industrial Engineering*, vol. 91, pp. 109–128, 2016.

[12] M. Afzalirad and J. Rezaeian, "A realistic variant of bi-objective unrelated parallel machine scheduling problem: NSGA-II and MOACO approaches," *Applied Soft Computing*, vol. 50, pp. 109–123, 2017.

[13] M. H. F. Zarandi and V. Kayvanfar, "A bi-objective identical parallel machine scheduling problem with controllable processing times: a just-in-time approach," *The International Journal of Advanced Manufacturing Technology*, vol. 77, no. 1-4, pp. 545–563, 2015.

[14] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 4661–4667, September 2007.

[15] Z. Ardalan, S. Karimi, O. Poursabzi, and B. Naderi, "A novel imperialist competitive algorithm for generalized traveling salesman problems," *Applied Soft Computing*, vol. 26, pp. 546–555, 2015.

[16] A. Kaveh and S. Talatahari, "Optimum design of skeletal structures using imperialist competitive algorithm," *Computers & Structures*, vol. 88, no. 21-22, pp. 1220–1229, 2010.

[17] S. Hosseini, A. A. Khaled, and S. Vadlamani, "Hybrid imperialist competitive algorithm, variable neighborhood search, and simulated annealing for dynamic facility layout problem," *Neural Computing and Applications*, vol. 25, no. 7-8, pp. 1871–1885, 2014.

[18] K. Lian, C. Zhang, L. Gao, and X. Shao, "Single row facility layout problem using an imperialist competitive algorithm," in *Proceedings of the in Proceedings of the 41st International Conference on Computers and Industrial Engineering*, pp. 578–586, 2011.

[19] B. Mohammadi-ivatloo, A. Rabiee, A. Soroudi, and M. Ehsan, "Imperialist competitive algorithm for solving non-convex dynamic economic power dispatch," *Energy*, vol. 44, no. 1, pp. 228–240, 2012.

[20] M. Bagher, M. Zandieh, and H. Farsijani, "Balancing of stochastic U-type assembly lines: An imperialist competitive algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 54, no. 1-4, pp. 271–285, 2011.

[21] Z. Wei, J. Yan, Y. Li, C. Xia, and J. Zheng, "Imperialist competitive algorithm for assembly sequence planning," *The International Journal of Advanced Manufacturing Technology*, vol. 67, no. 9-12, pp. 2207–2216, 2013.

[22] B. Wang, Z. Guan, D. Li, C. Zhang, and L. Chen, "Two-sided assembly line balancing with operator number and task constraints: a hybrid imperialist competitive algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 74, no. 5-8, pp. 791–805, 2014.

[23] M. A. Ahmadi, M. Ebadi, A. Shokrollahi, and S. M. J. Majidi, "Evolving artificial neural network and imperialist competitive

algorithm for prediction oil flow rate of the reservoir," *Applied Soft Computing*, vol. 13, no. 2, pp. 1085–1098, 2013.

[24] K. Devika, A. Jafarian, and V. Nourbakhsh, "Designing a sustainable closed-loop supply chain network based on triple bottom line approach: a comparison of metaheuristics hybridization techniques," *European Journal of Operational Research*, vol. 235, no. 3, pp. 594–615, 2014.

[25] J. Behnamian and M. Zandieh, "A discrete colonial competitive algorithm for hybrid flowshop scheduling to minimize earliness and quadratic tardiness penalties," *Expert Systems with Applications*, vol. 38, no. 12, pp. 14490–14498, 2011.

[26] S. M. Goldansaz, F. Jolai, and A. H. Zahedi Anaraki, "A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 37, no. 23, pp. 9603–9616, 2013.

[27] D. Lei, M. Li, and L. Wang, "A two-phase meta-heuristic for multiobjective flexible job shop scheduling problem with total energy consumption threshold," *IEEE Transactions on Cybernetics*, pp. 1–13, 2018.

[28] B. Naderi and M. Yazdani, "A model and imperialist competitive algorithm for hybrid flow shops with sublots and setup times," *Journal of Manufacturing Systems*, vol. 33, no. 4, pp. 647–653, 2014.

[29] A. Mortazavi, A. A. Khamseh, and B. Naderi, "A novel chaotic imperialist competitive algorithm for production and air transportation scheduling problems," *Neural Computing and Applications*, vol. 26, no. 7, pp. 1709–1723, 2015.

[30] S. Forouharfard and M. Zandieh, "An imperialist competitive algorithm to schedule of receiving and shipping trucks in cross-docking systems," *The International Journal of Advanced Manufacturing Technology*, vol. 51, no. 9-12, pp. 1179–1193, 2010.

[31] P. Zhang, Y. Lv, and J. Zhang, "An improved imperialist competitive algorithm based photolithography machines scheduling," *International Journal of Production Research*, pp. 1–13, 2017.

[32] N. Karimi, M. Zandieh, and A. A. Najafi, "Group scheduling in flexible flow shops: A hybridised approach of imperialist competitive algorithm and electromagnetic-like mechanism," *International Journal of Production Research*, vol. 49, no. 16, pp. 4965–4977, 2011.

[33] S. Karimi, Z. Ardalan, B. Naderi, and M. Mohammadi, "Scheduling flexible job-shops with transportation times: mathematical models and a hybrid imperialist competitive algorithm," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 41, pp. 667–682, 2017.

[34] H. Seidgar, M. Zandieh, H. Fazlollahtabar, and I. Mahdavi, "Simulated imperialist competitive algorithm in two-stage assembly flow shop with machine breakdowns and preventive maintenance," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 230, no. 5, pp. 934–953, 2016.

[35] S. Hosseini and A. Al Khaled, "A survey on the Imperialist Competitive Algorithm metaheuristic: Implementation in engineering domain and directions for future research," *Applied Soft Computing*, vol. 24, pp. 1078–1094, 2014.

[36] L. Jin, C. Zhang, X. Shao, X. Yang, and G. Tian, "A multi-objective memetic algorithm for integrated process planning and scheduling," *The International Journal of Advanced Manufacturing Technology*, vol. 85, no. 5-8, pp. 1513–1528, 2016.

[37] U. Güvenç and F. Katırcıoğlu, "Escape velocity: a new operator for gravitational search algorithm," *Neural Computing and Applications*, pp. 1–16, 2017.

[38] A. Zhang, G. Sun, Z. Wang, and Y. Yao, "A hybrid genetic algorithm and gravitational search algorithm for global optimization," *Neural Network World*, vol. 25, no. 1, pp. 53–73, 2015.