A modified immune clone algorithm for energy-saving scheduling in permutation flow shop

Yang Yang, Wang Yong Qi
Advanced Vocational Technical College, Shanghai University of Engineering, Shanghai 210620, China
Electronic and Electrical Engineering College, Shanghai University of Engineering, Shanghai 210620, China
Email: yydaisy@163.com

Abstract. For the permutation flow shop scheduling problem, an energy-saving scheduling model has been presented considering the energy consumption and idle time of deferment machines. The mathematical programming model was established to minimize the total energy consumption for idle time, and a modified immune clone algorithm was developed. Four mutation operators have been introduced to ensure its exploration ability. Meanwhile, a LCS-EDA neighbor search method was presented based on estimation of distribution algorithm, which aims to enhance the exploitation ability of MICA. Finally, the experiments were carried out based according to two standard testing collections. The results have validated the feasibility and effectiveness of the proposed energy-saving model and modified metaheuristic.

1. Introduction
The permutation of the flow shop scheduling problem is the simplified model as a class of production line scheduling problem, in the manufacturing industry it has a wide range of application background[1], its research has important theoretical significance and engineering value. Many domestic and foreign scholars have done an in-depth study of PFSP scheduling problems, and according to the different performance indicators these studies are divided into three categories[2]: a) minimize the maximum completion time of the PFSP problem; b) minimize the total completion of the PFSP problem; c) minimize the duration of the PFSP problem. PFSP-related problem solving methods include accurate algorithm, constructive algorithm and meta-heuristic algorithm[3]. In recent years, the simulation algorithm has been developed rapidly and successfully applied to PFSP-related scheduling problem. The results are satisfactory. Typical algorithms include genetic algorithm[4], simulated annealing algorithm[5] and artificial ant colony algorithm[6].

Immune clone algorithm (ICA) is a new metaheuristic algorithm proposed by immune system related concepts and theories. However, the ICA algorithm still has the disadvantages of slow convergence, easy to fall into local optimum. This paper constructs an improved immune clone algorithm and applies it to ESS-PFS problem. MICA algorithm constructs the LCS-EDA neighborhood transformation operator to strengthen the performance of its deep search based on the distribution estimation. Finally, the simulation experiment is carried out by using the standard test set, which proves the validity of the ESS-PFS model and the MICA algorithm.
2. Mathematical modeling of ESS-PFS

The ESS-PFS model can be described as: n jobs are processed in the same order on the serial line consisting of m machines, and the processing time is known. All jobs can be processed at the initial moment, one job can only be processed on one machine at a time, and one machine can only process one job. The energy function of each machine is related to its idle time. Based on the above description, the ESS-PFS problem is modeled as follows:

\[
\text{Min } E(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{m} \theta_i(\beta_{i,j})
\]

\[
C_{i,j} = \sum_{k=1}^{n} \tau_{i,j}, \quad i \in \{1,2,\ldots,m\}
\]

\[
C_{k,j} = \sum_{i=1}^{m} \tau_{i,j}, \quad k \in \{1,2,\ldots,n\}
\]

\[
C_{i,j} = \max(C_{i-1,j}, C_{i,k}) + \tau_{i,j}, \quad i \in \{2,\ldots,m\}, \quad k \in \{2,\ldots,n\}
\]

\[
\theta_{i,j} = 0, \quad j_{i,j} = 0
\]

\[
\theta_{i,j} = \max\{C_{i,j} - \tau_{i,j} - C_{i,k} = 0\}, \quad i \in \{2,\ldots,m\}, \quad k \in \{2,\ldots,n\}
\]

\[
j_{a,b} = \{1,2,\ldots,n\}, \quad a \neq b, \quad \forall a,b \in \{1,2,\ldots,n\}
\]

Here, the objective function is to minimize the total idle energy consumption of the machines. The formulas (1) and (3) define the completion time of the different machines. The equations (4) and (5) define the idle time of the machine, the formulas (6) and (7) indicate that the machining process is not interrupted and the machine can’t be preempted.

3. Improved immune clone algorithm

In order to overcome the shortcomings of ICA algorithm, the MIC A algorithm is improved in the following aspects: (1) Inversion is introduced to improve the quality of the initial solution; (2) Four kinds of operators are introduced in the antibody mutation stage to ensure the global exploration ability of the algorithm; (3) Based on the information of the excellent antibodies in the memory, the solution space of the problem is modeled by the distribution estimation algorithm, and the LCS-EDA using the longest common subsequence between the excellent antibodies is constructed to achieve the purpose of strengthening the depth search algorithm. The specific steps of MICA algorithm are as follows:

Step1. Initialize the parameters of the MICA algorithm.

Step2. Construct the initial population and calculate the antibody affinity values, and according to the antibody affinity ranking divide the current population into excellent memory and general population.

Step3. For each antibody, calculate the clone size based on affinity and perform antibody clone operation.

Step4. Perform the mutation operation for each antibody after the clone operation.

Step5. Update the probabilistic model of the solution space using excellent memory bank. A new antibody is generated by LCS-EDA neighborhood transformation method for each excellent antibody in the memory.

Step6. Calculate the affinity of new antibodies generated in Step4 and Step5, update the excellent memory and general population according to the affinity, and partially extinct and reinitialize those with lower affinity in the current population.

Step7. If the termination condition is satisfied, stop MICA algorithm and output the current optimal antibody. Otherwise, go to Step3.
3.1. Antibody expression and initialization

Aiming at the energy saving scheduling model of the permutation of flow shop constructed in this paper, the antibody in MICA algorithm uses the permutation of labels of n jobs to express the solution of the problem. As shown in Fig. 1, for n = 6, the codings of the two scheduling schemes are \( \pi_1 = \{2, 1, 6, 3, 5, 4\} \) and \( \pi_2 = \{3, 1, 4, 5, 2, 6\} \).

![Figure 1 Antibody representation](image)

In the decoding process, n and m are given parameters, and calculates \( C_{i,j} \) taking time \( O(m \cdot n) \), on this basis obtains \( \beta_{i,j} \) taking time \( O(m \cdot n) \), so the time complexity of calculating \( E(\pi) \) is \( O(m \cdot n) \).

At the initial stage, the size of population \( S \) is given. MICA algorithm first uses the random method to generate \( S \) antibodies and the same amount of new antibodies by the reverse operation. Finally, select \( S \) antibodies with smaller objective function values to form the initial population \( \pi_0 = \{\pi_1, \pi_2, \cdots, \pi_S\} \). As shown in Fig. 2, the codings of the antibody \( \pi \) and its reversed antibody \( \pi^o \) are given.

![Figure 2 Inversion operation](image)

3.2. Affinity calculation

MICA algorithm uses affinity to evaluate each scheduling scheme in current population. Given \( g \)-generation antibody population \( \pi_1^g = \{\pi_1, \pi_2, \cdots, \pi_S\} \), the affinity \( F(\pi_s) \) in current population is obtained based on the objective function values \( E(\pi_s) \) of the antibodies \( \pi_s \), and calculated as follows:

\[
F(\pi_s) = \frac{\max \{E(\pi_l) | l = 1, 2, \cdots, S\} - E(\pi_s)}{\sum_{l=1}^{S} E(\pi_l)} \tag{8}
\]

S antibodies are ordered descending by affinity values, select the first S antibodies \( (S^F < S) \) to build an excellent memory \( P^g_s \), the remaining antibodies constitute a general population \( P^g_r \), that is, \( P_g = P^g_s \cup P^g_r \) and \( P^g_s \cap P^g_r = \emptyset \).

3.3. Antibody clone and mutation

For \( \forall s \in \{1, 2, \cdots S\} \), the clone size \( q_s \) is calculated from the affinity value \( F(\pi_s) \) of antibody \( \pi_s \) and its affinity order. The formula is as follows:

\[
q_s = \text{ceil}(F(\pi_s) \cdot S \cdot S^F \cdot \frac{\gamma}{\pi}) \tag{9}
\]

Here, the parameter \( \gamma \) is the clone scale factor, the parameter \( \pi \) is the position label of affinity in the antibody \( \pi_s \). In addition, in order to improve the speed of the algorithm, to avoid relying solely on improving the size of the antibody to achieve the optimal solution to the problem, MICA algorithm specifies the number of antibody clone threshold as \( S/2 \), i.e., \( q_s \leq S/2 \).

The mutations are performed for each antibody obtained at the clone stage, given antibody \( \pi \), MICA algorithm uses four kinds of operators, including RS, RSS, RAS and RSRS. These four operators can effectively guarantee the global optimization ability of the algorithm and avoid searching the blind spot. The specific operation is as follows:

1. RS, randomly select the two gene positions in \( \pi \), and exchange the job entries at the positions
2. RSS, randomly select two sub-sequences in \( \pi \), and exchange jobs at the positions
(3) RAS, randomly select a sub-sequence in $\pi$ and reverse the job item at the positions
(4) RSRS, randomly select two sub-sequences in $\pi$, exchange and reverse the job entries at the positions

![Antibody mutation](image)

**Figure 3 Antibody mutation**

### 3.4 Probabilistic Model of Memory Space

For the ESS-PFS model with $n$ jobs, the probability model of the solution space is represented by the $n \times n$-order probability matrix $D=[d_{u,v}]$, here, the element $d_{u,v}$ represents the probability value at which the job term $u$ is at the position $v$.

In the initialization phase, $\forall u,v \in \{1, 2, \ldots, n\}$, the value $d_{u,v}$ is set to $1/n$; in the iterative process, the Hebb rule is used to update the probability matrix, given the learning factor $\lambda$ ($\lambda \in (0,1)$), the update formula is as follows:

$$d_{u,v}^e = (1-\lambda) \cdot d_{u,v}^{e-1} + \lambda \cdot \frac{1}{S^e} \sum_{j=1}^{S^e} I_{\beta}(u,v)$$

(10)

Here, the symbol $I_{\beta}(u,v)$ is an indicative function and is defined as follows:

$$I_{\beta}(u,v)=
\begin{cases}
1 & \text{the antibody $\pi_0$ in the memory $P_0^e$ at the pos $V$} \\
0 & \text{others}
\end{cases}$$

(11)

For each iteration process, the time complexity of updating the probability matrix $D$ is $O(n(n+S^e))$

### 3.5 Solving the longest common subsequence

The longest common subsequence of two antibodies was obtained by dynamic programming. Given the scheduling scheme $\pi_1 = \{j_i\}$, the beginning $a$ elements of the scheduling scheme constitute the sub-sequence $\pi_1 = \{j_1, j_2, \cdots, j_a\}$. Similarly, the beginning $b$ elements of the scheduling scheme $\pi_2 = \{j'_1, j'_2, \cdots, j'_b\}$ constitute the sub-sequence $\pi_2 = \{j'_1, j'_2, \cdots, j'_b\}$. Let $L_{a,b}$ be the longest common subsequence of $\pi_1$ and $\pi_2$, and its length is defined as $L_{a,b}$, then:

$$L_{a,b} =
\begin{cases}
0 & \text{if } a = 0 \text{ or } b = 0 \\
L_{a-1,b-1} + 1 & \text{if } a > 0, b > 0, \text{and } j_a = j'_b \\
\max\{L_{a-1,b}, L_{a,b-1}\} & \text{if } a > 0, b > 0, \text{and } j_a \neq j'_b
\end{cases}$$

(12)

Furthermore, a longest common subsequence of antibodies $\pi_1$ and $\pi_2$ is obtained using the $L_{a,b}$ value, and the recursion procedure is as follows:
\[ Z_{a,b} = \begin{cases} 
Z_{a-1,b-1} \circ j_a & \text{if } j_a = f'_b \\
Z_{a,b-1} & \text{if } j_a \neq f'_b \text{, and } L_{a,b-1} > L_{a-1,b} \\
Z_{a-1,b} & \text{if } j_a \neq f'_b \text{, and } L_{a,b-1} \leq L_{a-1,b} 
\end{cases} \]  

(13)

Here, the \( \circ \) operation of the permutation of \( Z_{a,b} \) and element \( j_a \) places \( j_a \) at the end of \( Z_{a,b} \) to constitute a new permutation. For example, \( Z_{a,b} = \{2,1\} \) and \( j_a = 4 \), then \( Z_{a-1,b-1} \circ j_a = \{2,1,4\} \). In order to effectively illustrate the DP algorithm to calculate \( L_{a,b} \), taking two antibodies \( \pi_1 (\{5,6,1,4,2,3\}) \) and \( \pi_2 (\{1,6,2,4,5,3\}) \), the calculation process is shown in Fig.4.

In addition, the DP algorithm obtains LCS of two antibodies in \( O(n) \) time. Thus, the time complexity required to solve the LCS for each iteration process of the MICA algorithm is \( O(n \cdot S^E) \).

3.6. LCS-EDA neighborhood transformation

LCS-EDA neighborhood transformation is performed on the excellent antibody in memory bank \( P^E_g \). For \( \forall \pi_i = \{j_k\}_n \in P^E_g \), randomly select another antibody in the memory \( \pi_j \in P^E_g \). Let \( U \) be the set of all the elements in the longest common subsequence of the antibodies \( \pi_i \) and \( \pi_j \), for given inheritance coefficients \( \sigma \in (0,1) \), the process of updating \( \pi_i \) is as follows.

Step 1. Let \( k \leftarrow 1, \phi \leftarrow \pi_i, \psi \leftarrow \{1,2,\ldots,n\} \), go to step 2.

Step 2. If the job label at the kth position of \( \pi_i \) is \( j_k \in U \) and \( \text{rand}(\cdot) \leq \sigma \) is satisfied, then keep job \( j_k \) at position \( k \) in \( \pi_i \), let \( \phi \leftarrow \phi \setminus \{j_k\}, \psi \leftarrow \psi \setminus \{k\} \), go to step 3.

Step 3. Let \( k \leftarrow k + 1 \), if \( k \leq n \), go to step 2; otherwise, go to step 4.

Step 4. If \( |\phi| \neq 0 \), go to step 2; otherwise, go to step 5.

Step 5. Randomly select an element \( j_\psi \) in \( \phi \) and select the gene position \( v' \) with the probability value \( d_{j_\psi,v} / \sum_{v \in \psi} d_{j_\psi,v} \) to insert the job \( j_\psi \), let \( \phi \leftarrow \phi \setminus \{j_\psi\}, \psi \leftarrow \psi \setminus \{v'\} \). Go to step 4.

Step 6. Finish LCS-EDA transformation to generate new antibody \( \pi'_i \).

For \( \pi_i (\{5,6,1,4,2,3\}) \) and \( U (\{1,2,3\}) \), perform LCS-EDA neighborhood transformation to the antibody, and the resulting new antibody is \( \pi'_i (\{4,2,1,6,5,3\}) \).

Updating a single antibody using LCS-EDA method to performs the neighborhood transformation takes time \( O(n^2) \). Therefore, the LCS-EDA method takes total time \( O(n^2 \cdot S^E) \) in each iteration process of the MICA algorithm.

3.7. The antibody’s extinction
Eliminate $d\%$ antibodies with lower affinity in the current population and use the initialization operation to supplement the corresponding amount of new antibodies to maintain the diversity of the population.

3.8. Algorithm termination condition

The termination condition of the MICA algorithm in this paper consists of two parts: (1) the number of iterations of the algorithm reaches the preset number of iterations $G$; (2) The number of steps in the optimal solution of the algorithm reaches the pre-set stagnation step threshold $G'$. 

4. Simulation test and case study

4.1. Example illustrates

The test case consists of two parts: eight small-scale examples from the Car standard test set and nine examples from the Ta standard test set. In the former example, the number of jobs is $n \leq 14$, the number of machines is $m \leq 9$; in the latter case, $n \in \{20, 50, 100\}$, $m \in \{5, 10, 20\}$. In addition, assuming that the energy consumption function $\theta(\beta)$ of the machine $i$ ($\forall i$) is proportional to the idle time of the machine, the proportional coefficient obeys the uniform distribution, and the scale coefficients in the two test sets are in the range of $(2, 3)$ and $(20, 30)$, respectively, in units of $10^3$J/s. The algorithm is tested on a computer with clock rate 2.40GHz Intel Core i5 CPU, 4G memory, and Matlab@2016a.

4.2. MICA algorithm parameter setting

The MICA algorithm constructed in this paper has three important parameters that need to be set up reasonably: memory scale parameters $\sigma (S^e = \text{ceil}(S \cdot \sigma))$, probability model learning coefficient $\lambda$ and gene inheritance coefficient $\varpi$. The other parameters are configured: the size of the population is 40, the clone scale factor $\gamma$ is 3, the antibody mortality rate $d\%$ is 0.05, the total number of iterations $G$ is 800, and the stagnation step $G'$ is 200.

The test example selects the Ta051 example in the Ta data set, and the number of machines in the example is $m = 20$, and the number of jobs is $n = 50$. Orthogonal test program scale is $L_n(4^k)$, each test program simulate 30 times and take the average, the test results are shown in Table 1.

| Test number | $\sigma$ | $\lambda$ | $\varpi$ | Average value |
|-------------|----------|-----------|----------|---------------|
| 1           | 1        | 1         | 1        | 72.97         |
| 2           | 1        | 2         | 2        | 71.17         |
| 3           | 1        | 3         | 3        | 73.42         |
| 4           | 1        | 4         | 4        | 72.97         |
| 5           | 2        | 1         | 2        | 66.07         |
| 6           | 2        | 2         | 1        | 65.37         |
| 7           | 2        | 3         | 4        | 77.67         |
| 8           | 2        | 4         | 3        | 65.97         |
| 9           | 3        | 1         | 3        | 64.02         |
| 10          | 3        | 2         | 4        | 71.48         |
| 11          | 3        | 3         | 1        | 76.01         |
| 12          | 3        | 4         | 2        | 73.01         |
| 13          | 4        | 1         | 4        | 66.25         |
| 14          | 4        | 2         | 3        | 68.77         |
Table 2 shows the simulation results, and determines the degree of influence of parameters in MICA algorithm optimization performance. It can be seen that the probabilistic model learning coefficient $\lambda$ and the genetic inheritance coefficient $\varpi$ have important influence on the MICA algorithm. It is helpful to effectively use the LCS-EDA method proposed in this paper to enhance the depth search capability of the algorithm. In addition, the size of memory $\sigma$ is used to distinguish between current population of good and bad antibodies, too small $\sigma$ value will make the MICA algorithm inability to take full advantage of the current population of excellent antibodies, too large $\sigma$ value will make excellent antibody sampling not accurate and influences the algorithm search depth. In summary, the parameters are set as follows: $\sigma = 0.4$, $\lambda = 0.2$, $\varpi = 0.4$.

| Level | $\sigma$ | $\lambda$ | $\varpi$ |
|-------|---------|-----------|---------|
| 1     | 72.63   | 67.33     | 72.94   |
| 2     | 68.77   | 69.20     | 70.08   |
| 3     | 71.13   | 74.29     | 68.04   |
| 4     | 70.63   | 72.34     | 72.09   |
| Very poor | 3.86 | 6.96   | 4.90 |
| Sorting | 3       | 1         | 2       |
| Best level | 2     | 1         | 3       |

Table 2 Range analysis

4.3. MICA algorithm performance verification

In order to verify the optimization performance of the MICA algorithm constructed in this paper, the algorithm optimization result is compared with the CPLEX accurate solution. Select 8 small-scale examples in the Car data set and 30 simulations for each study, Table 3 shows the minimum deviation and the mean deviation between the optimization results of the MICA algorithm and the optimal solution obtained by the DP algorithm, and records the run time of both algorithms. From Table 3 we can see that the minimum deviation $\epsilon_{\text{min}}$ of each study is zero, and the mean deviation $\epsilon_{\text{mean}}$ is zero or relatively small. It is shown that the MICA algorithm can obtain the optimal solution of these examples in most cases. In addition, MICA algorithm is less time-consuming. Take Car04 as an example, it takes a short period of time to effectively solve the small scale example, the running time is 6.37s, while CPLEX running time is 706.92s.

| test group | scale $m \times n$ | CPLEX Exact solution time | MICA $\epsilon_{\text{min}}$ $\epsilon_{\text{mean}}$ time |
|------------|---------------------|--------------------------|-----------------------|
| Car01      | 5×11                | 0.36 142.87              | 0 0.003 5.87          |
| Car02      | 4×13                | 0.49 395.67              | 0 0.011 6.44          |
| Car03      | 5×12                | 0.07 171.71              | 0 0.008 6.23          |
| Car04      | 4×14                | 0.25 706.92              | 0 0.016 6.37          |
| Car05      | 6×10                | 2.23 41.82               | 0 0 5.78              |
Car06 9×8 5.22 29.68 0 0 5.61
Car07 7×7 1.08 16.67 0 0 5.42
Car08 8×8 5.86 25.94 0 0 5.44

5. Conclusion
On the basis of fully considering the energy consumption of different machines and their idle time, this paper constructs the energy-saving scheduling model of permutation flow shop with the aim of minimizing the total energy consumption of the machine. In order to effectively solve the ESS-PFS model, the MICA algorithm is proposed to enhance the optimization ability of the algorithm by introducing four kinds of mutation operators and LCS-EDA neighborhood transformation method. The results show that the MICA algorithm has some advantages in solving the energy-saving scheduling model.

References
[1] Nagano M S, Moccellin J V. Reducing mean flow time in permutation flow shop [J]. Journal of the Operational Research Society, 2008, volume 59(59): 939-945.
[2] Nagarajan V, Sviridenko M. Tight Bounds for Permutation Flow Shop Scheduling [J]. Mathematics of Operations Research, 2009, 34(2): 417-427.
[3] Sheibani K. A fuzzy greedy heuristic for permutation flow-shop scheduling [J]. Journal of the Operational Research Society, 2010, 61(5): 813-818.
[4] Vallada E, Ruiz R. Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem [J]. Omega, 2010, 38(1–2): 57-67.
[5] S. Parthasarathy, Chandrasekharan Rajendran. A simulated annealing heuristic for scheduling to minimize mean weighted tardiness in a flowshop with sequence-dependent setup times of jobs-a case study [J]. Production Planning & Control, 1997, 8(5): 475-483.
[6] Tasgetiren M F, Pan Q K, Suganthan P N, et al. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops [J]. Information Sciences, 2011, 181(16): 3459-3475.