doi:10.21311/001.39.9.38

Resource Matching Model of Cloud Manufacturing Platform Based on Granularity Optimization of the SFLA

Hexin Lv, Zhen Xu
Institute of Information Technology, Zhejiang Shuren University, Hangzhou 310015, China

Abstract
For the convergence defects of the shuffleg frog leaping algorithm (SFLA) and needs of the cloud manufacturing platform resources matching, this paper proposes a resource matching model of cloud manufacturing platform based on the granularity optimization of the SFLA. Specifically, Cauchy mutation operator is introduced into the basic SFLA to improve the local update strategy. By improving the random disturbance to expand the search range of the frog, diversity of population is increased. Then, the model uses the adaptive adaptive fitness to optimize the inertia weight of the algorithm, introduces an accelerated contraction factor and adopts the particle swarm optimization algorithm to optimize the SFLA algorithm. Finally, a resource matching model of the cloud manufacturing platform is set up. After the rough selection of data, the improved SFLA is used to subdivide it and match for the model. Simulation results show that performance on the convergence of the original SFLA algorithm is greatly improved by the SFLA optimization strategy, and in application of building a resource matching model for the cloud manufacturing platform, the improved algorithm has good optimization ability.

Key words: Cloud manufacturing, Resource matching, SLFA, Granularity optimization, Search domain, Accelerating shrinking factor, Adaptive adaptive fitness

1. INTRODUCTION
Cloud manufacturing is to provide the whole life cycle service for manufacturing process (Li and Zheng, 2015), which has the features of convenience, high security and low cost. Since it is realized based on technologies of the fusion of cloud computing, cloud security, high performance computing, and internet and so on, it can obtain various virtual and convenient resources for planned management, operation and use. The core idea of cloud manufacturing is to integrate all kinds of distributed manufacturing resources into the manufacturing platform and manage these resources. By accessing the cloud platform, users can use the manufacturing resources in the platform. It can provide the users with efficient and low consumption products and services. By adopting the cloud manufacturing technology, an enterprise can share the resources among enterprises, industries and regions, and use efficiently to improve the enterprise competitiveness in the global market (Liu and Zhang, 2015).

As the manufacturing resources distribution has the regional characteristic and diversity, the different forms of resources are lack of unified and perfect management. The personized needs with a wide range of manufacturing resource services intelligent matching can make users find the required resources fast and accurately from the cloud manufacturing platform, which becomes one of the key technologies to implement the cloud manufacturing platform (Sheng, Zhang and Lu, 2015).

Many researchers have studied in the field of resource service matching in both domestic and abroad. At abroad, Massimo (Paolucci , Kawamura and Payne, 2015) and others proposed I/O matching algorithm based on the description logic OWL-S. And Zhou (Zhou, Liang and Chia, 2015) and others implement the service matching in two stages based on the functional and non-functional requirements of service. At the first stage, the basic functional requirement (the input and output of the service, the premise) is analyzed to ensure the return of basic satisfied requirements. At the second stage, after analyzing the requirement of non-functional Qos, the authors used the description language OWL-S to describe Qos, implemented the service matching for the first stage, and got the best service. Katia (Katia Sycam and Klusch, 2012) and others defined a distance by using the semantic description language, considered the implementation of the approximate service matching and also balance between service quality and efficiency, and proposed a multi-service matching algorithm. Mark Klein (Mark Klein and Bemstern, 2014) and others implemented the matching for query request and web service description, proposed an algorithm of high precision service matching based on process ontology. In domestic, Shi Zhongzhi (Shi and Jiang, 2014) and others proposed the main service matching method based on description logic, gave the description of the effective reasoning function, combined with the multi-agent system services, and carried out the research on the theory of describing logic and reasoning mechanism in detail to realize the
layered service matching, and finally proposed five kinds of matching algorithms, which can overcome the deficiency of the semantic distance service matching. Yin Sheng (Yin, Yin and Liu, 2009) and others developed the resource integration service mechanism by extending the UDDI networked collaborative products, and proposed the three-layer search algorithm by extending the QWSDL to improve lack of flexibility of web service matching algorithm. Also in this paper, similarity functions is introduced to describe quantitatively the similarity of the relaxed match web services, which can improve the matching efficiency of the service with high precision and recall rate. Zhao Yan (Zhao and Mo, 2009) and others established a hierarchical satisfaction model and a matching degree model to alleviate the mismatch degree between manufacturing resources and manufacturing process in the cross-enterprise process diffusion industry, and used parameter to describe each level. Zheng Libin (Zheng, Gu and Dai, 2009) and others achieved the manufacturing resource discovery matching by extending the OWL-S, describing the dynamic characteristics of the resource service to solve the lack of dynamic description problem of the existing models. Sun Weihong (Sun and Feng, 2010) and others considered that the manufacturing task must have the matched resources under the background of networked manufacturing, which can solve the problems encountered in the management of complex parts for the leader enterprise. It can also realize the unified data platform and improve the collaboration between enterprises. Gao Yicong (Gao, Feng and Tan, 2012) and others considered the interaction between cloud manufacturing resources and needs, introduced the fuzzy integral to get the similarity between service resources, and then proposed a service matching model of cloud manufacturing based on the fuzzy integral.

For the defects of the convergence in SFLA and the needs of the cloud manufacturing platform resources matching, this paper proposes a resource matching model of cloud manufacturing platform based on the granularity optimization of the SFLA. At first, we improve the search field and inertia weight of the original algorithm, respectively. Then, the particle swarm optimization algorithm is introduced to optimize the SFLA. Simulation results show that the improved strategy proposed in this paper is effective.

2. THE PERFORMANCE ANALYSIS OF SFLA

2.1. Shufflefrog Leaping Algorithm (SFLA)

SFLA is a way of global collaborative search in bionic optimization algorithm, which based on the frog foraging behavior. There often have many stones in the living environment of the frogs, and the frogs jump back and forth between different stones when they search for food. Each frog jumps on different stones, and they communicate each other to convey information.

For the \( M \)-dimensional optimization problem, let \( S \) be the frog population size. Denote \( U(i) = \{ U_{ij}, U_{k2}, \ldots, U_{ij}, \ldots, U_{ij} \} \) as a solution for frog \( i \) (\( 1 \leq i \leq S, 1 \leq j \leq M \)). The adaptive adaptive fitness value of each frog in the population can be obtained, and be sorted in a decending order. Assume that the frog population has \( m \) groups, and each group has \( n \) frogs. Thus, \( S = m \times n \). Based on the above sorting, we classify the groups as follows: there is one frog in group one, two frogs in group two and so on, until \( m \) frogs in group \( m \), which results in that \( m + 1 \) ones are put in group one, \( m + 2 \) ones are put in group two and so on, until all the frogs are classified. Let \( P_b, P_w \) be frogs with the best and the worst adaptive fitness value, respectively. Denote \( P_a \) as the frog with the best adaptive fitness value in the whole population. If the iteration number does not meet the preset number, the local search needs to be done within all the groups, which makes \( P_a \) learn from \( P_b \) to achieve the optimization. After the local search of each group, a new population of all the frogs is formed. Then, the global search is completed.

SFLA can be basically divided into three processes: global search, local search and mixed operation. The process of global search is as follows:

1. Algorithm parameters. Let \( m \) be the number of groups, \( J_{\text{max}} \) be the maximum number of iterations of local search, and \( Q_{\text{max}} \) be the maximum number of iterations of global search. There are \( n \) numbers of frogs in each group, and \( S = m \times n \) numbers of frogs in total. The solutions in the optimization space have \( M \) dimensions in total. Let \( H \) and \( L \) represent the upper and lower limit of the search space, respectively.

2. Population initialization. In the optimization of \( M \) dimensions, \( S \) number of frogs are selected. Written as \( U(1), U(2), \ldots, U(S) \). Denote the \( i \)-th frog as \( U(i) = \{ U_{ij}, U_{k2}, \ldots, U_{ij}, \ldots, U_{ij} \} \) (\( 1 \leq i \leq S, 1 \leq j \leq M \) and \( U_{ij} \in [L, H] \)).

3. Adaptive fitness function. The adaptive fitness of solution \( U(i) \) can be obtained through adaptive fitness function \( f(i) \).
(4) Classification of frog groups. According to the adaptive fitness function, we can get the adaptive fitness value of frogs, sort $S$ number of values in descending order, and get the array $P = \{U(i), f(i), i = 1, 2, \ldots, S\}$. $U(1)$ represents the one with the best adaptive fitness of the whole population. Denote the global extreme value as $P_1 = U(l) = \{U_{11}, U_{12}, \ldots, U_{1m}\}$. Classification of frog groups: classify the groups according to formula (1), where $i = 1, 2, \ldots, n, k = 1, 2, \ldots, m$. Distribute the sorted frogs into $m$ numbers of groups $Z_1, Z_2, \ldots, Z_m$, and there have $n$ numbers of frogs in each group. For example, if $m = 3$, $n = 2$, then $S = 6$. We can conclude that $U(1)$ belongs to group $Z_1$, $U(2)$ belongs to group $Z_2$, $U(3)$ belongs to group $Z_3$, $U(4)$ belongs to group $Z_4$, $U(5)$ belongs to group $Z_2$, and $U(6)$ belongs to group $Z_3$.

The optimization will be implemented inside each group. Each frog is affected by the other frogs within the group, and it will learn from the individual with the best adaptive fitness and approach during the process of evolution.

The local search process of SFLA is as follows:

(1) The initialization $L_n = 0$ represents the group count, and the value range of $L_n$ is from 0 to $m$. The initialization $J_n = 0$ represents iteration value of the local search inside the group, and the value range of $J_n$ is from 0 to $J_{\text{max}}$. $P_n = \{U_{k1}, U_{k2}, \ldots, U_{km}\}$ represents the best and worst individuals within the group, respectively.

(2) $L_n = L_n + 1$

(3) $J_n = J_n + 1$

(4) If $J_n \leq J_{\text{max}}$, and the premise of the following two rules are satisfied, by using formula(2) and formula (3), the location of current group $P_n$ can be optimized, which can improve the adaptive fitness value of $P_n$. The consensus of rules and its description are as follows:

Rule 1. Let $D$ be the moving step length of the frog $P_w$ with the worst adaptive fitness, $D = \{D_1, D_2, \ldots, D_i, \ldots, D_M\}, i \in [1, M]$, and $D_i \in [-D_{\text{max}}, D_{\text{max}}]$. If $D_i < -D_{\text{max}}$, set $D_i$ as $-D_{\text{max}}$. If $D_i > D_{\text{max}}$, set $D_i$ as $D_{\text{max}}$.

Rule 2. $\hat{P}_w = \{\hat{U}_{w1}, \hat{U}_{w2}, \ldots, \hat{U}_{wj}, \ldots, \hat{U}_{wm}\}$ represents the updated frog individuals, $j \in [1, M], \hat{U}_{wj} \in [L, H]$. If $\hat{U}_{wj} < L$, set $D_j$ as $L$. If $\hat{U}_{wj} > H$, set $D_j$ as $H$.

The moving step length of frog is calculated as follows:

$$D = \text{rand}() \times (P_n - P_w)$$

The new position of $P_w$ can be calculated as follows:

$$\hat{P}_w = P_w + D$$

The above $\text{rand}()$ is a random number between 0 and 1, which indicates the degree of approximation from $P_n$ to $P_w$.

(5) If the adaptive value optimization of $P_w$ can be obtained according to the steps above, we can get a better solution and the frog will replace the original frog, and it represents as $P_w = \hat{P}_w$. Otherwise, $P_w$ is used to replace $P_n$ and re-execute the above optimization steps. If the adaptive value optimization of $P_w$ cannot be obtained according to the steps above, a randomly generated solution in the group is used to replace $P_w$.

(6) If $J_n < J_{\text{max}}$, execute (3).

(7) If $L_n < m$, return to (2). Otherwise, the frogs between the groups will form a new population, and the global search comes to an end.

2.2. The Performance Analysis of SFLA

The mixed operation process of SFLA is as follows:

(1) After the completion of the local search of $m$ numbers of groups, the frogs move and jump between groups, and they will be remerged into a new population.

299
(2) Denote the global iteration numbers as \( I = I + 1 \). If \( I < Q_{\text{max}} \), perform process (3) which is the process of implementing algorithm parameter, population initialization, adaptive fitness function and frog group classification.

(3) If \( I < Q_{\text{max}} \), we can get the global optimal solution. When the number of iterations reaches maximum \( Q_{\text{max}} \), we can consider it as the termination signal of global.

Select minima function as the objective function:

\[
 f(x) = \frac{1}{N} \sum_{j=1}^{N} (x_j^3 - 16x_j^2 + 5x_j)
\]

(4)

where \(|x_j| \leq 5\) and \(\min f(x^*) = -78.3323\). The mode of this function is complex, and its extreme value points will increase exponentially with the increase of the dimension, and it can be applied to verify the performance of the algorithm.

In the experiment, the specific algorithm parameters are set as follows: the population of the frogs is 200. It is divided into 20 groups, and there are 10 frogs in each group. The maximum number of iterations is 500, and the number of local iteration is 10. Denote the step length as \( D_{\text{max}} = X_{\text{max}} / 5 \) (\( X_{\text{max}} \) represents the maximum value of search scope). The final test result is the average of 30 times independent performances. And the 30 dimensional minima function is optimized by four different control strategies, and its optimization curve is shown in figure 1 below. The horizontal coordinate represents the evolution algebra, and the vertical coordinate represents the average optimal value of the Minima function.

![Figure 1. 30 dimensional minima function optimization curve](image)

Both the results of group performance and average optimization can be affected by three update operators in some way. Therefore, the operators must be generated randomly in the group to ensure the diversity of the groups. If the process of the learning operator towards the global extreme value is deleted, the efficiency and accuracy of the algorithm can be improved.

3. GRANULARITY CLASSIFICATION OPTIMIZATION OF THE SFLA

3.1. The Search Domain Optimization Strategy Based on Disturbed Cauchy Distribution

In the basic SFLA, only \( X_i \) in the group is updated in each iteration. Formula (2) is used to update the frog individuals every time. If the frog ignores some search domain, the algorithm is easy to fall into local optimum. The search domain optimization strategy based on disturbed Cauchy distribution is proposed to make full of the information among groups, which can solve the mentioned problem.

Cauchy distribution is widely used in mathematic theory and engineering application, etc. Cauchy density function is defined as:

\[
 f(x) = \frac{\beta}{\pi \left[\beta^2 + (x - \alpha)^2\right]}, \beta > 0
\]

(5)

And it is usually denoted as \( C(\alpha, \beta) \). Cauchy probability distribution function is as:

\[
 F(x) = 0.5 + \frac{\arctan((x - \alpha) / \beta)}{\pi}
\]

(6)

When \( \alpha = 0, \beta = 1 \), it is called as the standard Cauchy distribution, and denoted as \( C(0,1) \).
The shape of two wings of Cauchy distribution are relatively flat and wide. The pattern is a peak shape curve above the horizontal axis (shown as in figure 1).

![Cauchy Distribution](image)

**Figure 2.** The standard Cauchy distribution

The Cauchy mutation operator is introduced in the basic algorithm of SFLA to optimize the local updating strategy. The worst individual $X_n$ learns from the best one $X_s$ to be updated based on formula (7). The strategy of the update is:

$$D(t + 1) = \text{rand} \times (X_s - X_n) \times C(0,1)$$

The Cauchy mutation can reduce the possibility of the frog jumping into local optimal domain by adding random disturbance. Yet, when the algorithm enters the final period of searching process and approaches the global optimal extreme value, its mutation step length is long enough to pass the better search area. This paper proposed an improved strategy. And the local update strategy is:

$$D(t + 1) = \begin{cases} \text{rand} \times (X_s - X_n) \times C(0,1), R \leq Q \\ \text{rand} \times (X_s - X_n), R > Q \end{cases}$$

where $R$ is a random number in $[0,1]$, $Q$ is the probability of disturbances. The improved SFLA makes the frog have suitable step length, and the scope of frog search is greatly expanded to ensure the diversity of population, which results in improving the speed of the algorithm.

### 3.2. The Inertia Weight Optimization Based on Adaptive Fitness

Experimental results show that the larger inertia weight can improve the speed of the algorithm, while the smaller $\omega$ can improve the convergence speed. The local adaptive fitness and the global adaptive fitness are affected by the values of $\omega$. Thus, $\omega$ optimization strategy is proposed based on adaptive fitness.

Set $\omega \in [\omega_{\min}, \omega_{\max}]$, where $\omega_{\min}$ represents the minimum of the inertia weight, and $\omega_{\max}$ represents the maximum of the inertia weight. $f_{\text{glob}}(t)$ represents the global optimal adaptive fitness of all frogs in the $t$-th iteration, and $f_i(t)$ represents the local optimal adaptive fitness of the $i$-th frog in the $t$-th iteration. The inertia weight of the $i$-th frog can be obtained based on formula (10) below:

$$\omega_i(t) = \omega_{\min} - (\omega_{\max} - \omega_{\min}) \times e^{3/t}$$

From formula (10), we know that $\omega_i(t) \in [\omega_{\min}, \omega_{\max}]$. The general trend of $\omega$ is in negative correlation to the number of iterations $t$ which can help to improve the convergence speed of the algorithm, but it can also lead to getting the local optimal solution through the convergence. Therefore, the adaptive fitness value $f_i(t)$ is introduced in formula (11), which can help to achieve the optimization of $\omega$ through the current position of the frog. It is shown in formula (11) and (12) below:

$$f_i(t) = \lambda \times \frac{f_{\text{glob}}(t)}{f_i(t)}$$

where $\lambda$ is a positive constant. $t = it_j \times tit_k$, where $it_j$ represents the number of iterations in the sub-group, $it$ represents the total number of iterations in the sub-group, $tit_k$ represents the number of current mixed iterations, and $tit$ represents the total number of mixed iterations. Also integer $j \in [1, \text{it}]$ and integer $k \in [1, \text{tit}]$.

Put $\lambda = 1$. If $f_i(t)$ approaches 1, the results of formula (11) and formula (12) are the same. The optimal solution can be obtained by reducing $\omega$ to shorten the step length and narrowing the range. On the other hand,
if \( f_i(t) \) is far from 1, it means that the current position of the frog has been far from the optimal solution. Then, \( \omega \) should be enlarged to increase the step length through formula (12), and thus, the speed of approaching the optimal solution can be improved. The constant \( \lambda \) given in formula (12) is used to determine the speed of change of \( \omega \). In fact, \( \omega \) should slow down with the increase of \( \lambda \). Therefore, selection of appropriate \( \lambda \) can help to improve the convergence speed of the algorithm. Experiments will show that it is appropriate to set \( \lambda = 5 \) in this paper.

3.3 The Granularity Optimization Based on Accelerated Shrinkage Factor

The parameter \( c \) is introduced in this paper as an acceleration search. The optimization of SFLA is realized based on the Particle Swarm Optimization (PSO) to speed up the algorithm. However, the selection of \( c \) should be carried out specifically in each experiment, and also the convergency of algorithm cannot be ensured. Thus, the shrinkage factor \( x \) is introduced to ensure the convergency of the PSO algorithm. The worst individual \( P_w \) should be made to approach the best individual \( P_b \) in the sub-group with the fastest speed to get the global optimal individual \( P_g \). Besides, shrinkage factor \( x \) and acceleration factors \( c_1 \) and \( c_2 \) in PSO are introduced to the basic SFLA to make the update strategy perfect. The worst individual \( X_w \) can approach the best individual \( X_b \) in the subgroup to get the global optimal individual \( P_g \) based on formula (13). Thus, The approaching speed from the worst individual \( X_w \) to the best individual \( P_b \) and the search efficiency of the algorithm can be greatly improved, and the convergency of the algorithm can also be ensured. The selection of shrinkage factor \( x \), acceleration factor \( c_1 \) and \( c_2 \) are determined by formula (15). Usually \( c_1 = c_2 = 2.05 \), then \( x = 0.729 \).

\[
\text{Dis}(t+1) = x \cdot \text{rand}() \cdot c_1 \cdot (P_b - P_w) \quad \text{(13)}
\]

\[
\text{Dis}(t+1) = x \cdot \text{rand}() \cdot c_2 \cdot (P_g - P_w) \quad \text{(14)}
\]

\[
x = \frac{2}{2 - \phi - \sqrt{\phi^2 - 4\phi}}, \quad \phi = c_1 + c_2 > 4 \quad \text{(15)}
\]

If the worst individual cannot learn from the best individual \( P_b \) in the subgroup or the global optimal individual \( P_g \) in the whole group, \( P_w \) is replaced by a random generated solution within the search range. Although the number of individuals in the group is increased, \( P_w \) has not been studied, which results in slowing down the convergence speed of the algorithm. If the worst individual cannot learn from the best individual \( P_b \) in the subgroup or the global optimal individual \( P_g \) in the whole group, a small radius search is carried out based on formula (16) to make full use of the information of \( P_w \), and \( \text{new} P_w \) can be obtained with better adaptive fitness than \( P_w \). Thus, we can use \( \text{new} P_w \) to replace \( P_w \). Conversely, the search based on formula (16) should be implemented repeatedly until a randomly generated \( \text{new} P_w \) satisfies the condition. If the condition still cannot be satisfied after several times of repetition, a randomly generated solution within the search range is used to replace \( P_w \).

\[
\text{new} P_w = P_w + (2\text{rand}() - 1) \cdot \text{Step} \quad \text{(16)}
\]

where \( \text{Step} \) represents the random moving step length of individual, and the value range of \( \text{rand}() \) is \([0, 1]\). The value range of \( \text{Step} \) should be adjusted dynamically based on formula (17) and (18).

\[
\text{Step} = \text{Step} \cdot a + \text{Step}_{\min} \quad \text{(17)}
\]

\[
a = \exp(-30 \cdot (t / T_{\max})^2) \quad \text{(18)}
\]

Under normal circumstances, the initial value of \( \text{Step} \) is \( X_{\max} / 16 \) (\( X_{\max} \) is the maximum value within the search range). \( \text{Step}_{\min} = 0.002 \), \( t \) represents the current iteration number, \( T_{\max} \) represents the maximum iteration number. And the value range of \( s \) in this paper is \([1, 30]\).

4. THE RESOURCE MATCHING MODEL OF CLOUD MANUFACTURING PLATFORM BASED ON THE IMPROVED SFLA

Because of the variety of cloud manufacturing service resources and tens of thousands of suppliers under different levels of different manufacturing resources, the rough section of the resources is particularly important,
which can improve the efficiency of the late match and reduce the time of match. For easy understanding and simplified description, the mathematical description of cloud manufacturing platform resources is as follows:
\[ CSR = (AI, FI, SI, EI) \]  
(19)

where \( AI \) represents the subsidiary information of cloud manufacturing service resources, \( FI \) represents function information, \( SI \) represents state information, and \( EI \) represents assessment information.

The significance of rough selection is to exclude the hardware unqualified resources from the reference range, and these resources include the unqualified one certain index or multi-indexes of each resource. In the process of rough selection, we mainly consider the first three indexes: \( AI, FI \) and \( SI \), which is shown in the formula below.
\[ CSR' = (RT_{i_1},...,RT_{i_p},RT_{n_k},...,RT_{n_u}) \]  
(20)

where \( i \) represents the category of the manufacturing resources, \( s \) represents the level of the manufacturing resources and \( p \) represents the serial number of the manufacturing resources in the category or level. \( RT_{i_u} \) represents the resource class multi-corresponding to the small task, its listing order in \( CSR' \) is based on the processing sequence of the small task.

Then sub-division is carried out based on the improved SFLA. Since \( CSR' \) includes \( p \) numbers of \( RT_{i_u} \), and \( RT_{i_u} \) also includes \( n_u \) numbers of \( RT_{i} \), a two-dimensional matrix is used for individual coding in this paper, and the size of the matrix is \( p \times n_u \), where \( n_u \) refers to the maximum value in \( RT_{i_u} \). Each gene position represents a small task, and the listing order of the gene represents the processing sequence of the small task, which cannot be changed. The gene number indicates the number of processing resources that can provide processing services.

Every gene represents one \( RU_{i_u,k} \) in \( CSR' \), and \( k \) represents the number of genes. \( p \) and \( n_u \) are two fixed parameters, which are determined by the steps of dividing the task into small tasks, and they represents the number of small tasks (the number of genes in each individual) and the maximum value of \( n_u \), respectively. As the maximum value of \( n_u \) is a constant value, and cannot be reflected in individual coding after the beginning of search. Finally, the obtained individual coding is put into the proposed SLFA in the paper to carry out the cloud manufacturing platform resource matching.

5. SIMULATIONS
5.1. Simulations of the Improved Convergence of the Algorithm

To confirm the improved convergence of SFLA, three typical test functions with different characteristics are selected, and they are Rosenbrock function, Rastrigin function, and Griewank function, respectively. The formulas are shown in (22) and (23), and graphs of the functions are shown in figure 3.

\[ f_1(x) = \sum_{i=1}^{n} (100(x_{i+1}^2 - x_i)^2 + (1 - x_i)^2), -30 < x_i < 30 \]  
(21)
\[ f_2(x) = \sum_{i=1}^{n} x_i^2 - 10 \cos(2 \pi x_i) + 10, -5.12 < x_i < 5.12 \]  
(22)
\[ f_3(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos \frac{x_i}{\sqrt{i}} + 1, -600 < x_i < 600 \]  
(23)

![Figure 3. Three diagrams of test functions](image-url)
In the simulation, the parameter setting of the improved SFLA is as follows: the population size is 200, the group size is 20, the number of frogs in the group is 10, evolutionary times is 500, and the latitude of test functions is 30. The 30-dimentional test function is run 30 times independently through the standard SFLA and the improved SFLA. The obtained average optimal value is shown in the following table:

<table>
<thead>
<tr>
<th>Test Function</th>
<th>The theoretical optimal value</th>
<th>Algorithm</th>
<th>The average optimal value</th>
<th>Standard deviation</th>
<th>Running time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rosenbrock</td>
<td>0</td>
<td>SFLA</td>
<td>110.43</td>
<td>49.23</td>
<td>2.4</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Improved-SFLA</td>
<td>28.74</td>
<td>12.93</td>
<td>1.3</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>0</td>
<td>SFLA</td>
<td>9.34</td>
<td>3.94</td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Improved-SFLA</td>
<td>0.62</td>
<td>1.22</td>
<td>1.1</td>
</tr>
<tr>
<td>Griewank</td>
<td>0</td>
<td>SFLA</td>
<td>0.59</td>
<td>0.21</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Improved-SFLA</td>
<td>0.01</td>
<td>0.01</td>
<td>0.8</td>
</tr>
</tbody>
</table>

By running the three test functions through the standard SFLA and the improved SFLA under the same iteration numbers, we can find from table 1 that the obtained results from improved SFLA have been greatly improved than those of the standard SFLA. And the standard deviations of the improved SFLA are obviously less than those of the standard SFLA, which shows the improved SFLA has better stability.

Figure 4 shows the evolution curve of mean extreme value of the function, where the horizontal coordinate shows the natural logarithm of the mean extreme value, and the vertical coordinate shows the evolution algebra.

Figure 4. The evolution curve of mean extreme value
We can find from figure 4 that the convergence accuracy and the convergence speed of late period are greatly improved through the improved SFLA under the same iteration numbers.

Figure 5 is the population diversity curve of the function, where the horizontal coordinate shows the population diversity of the function.

![Figure 5](image)

**Figure 5.** The population diversity curve of the function

We can find from the results of figure 5 that the SFLA loses the population diversity soon during the running process of the algorithm. While the improved SFLA has better ability to keep the population diversity compared with the standard SFLA, and its ability of searching the optimal solution can be enhanced with the increase of evolution algebras.

### 5.2. The Simulations of the Cloud Manufacturing Resource Matching Model

We randomly select one scheme from the cloud manufacturing resources matching related design schemes, and implement the optimal resource matching through the resource matching model of cloud manufacturing platform. According to the design, the task has two decomposition ways of small tasks, which means that CSR has two sets, as showed in the following:

\[
CSR_1^\prime = \{RT_{4,ER}RT_{1,SE}RT_{6,SE}RT_{6,PC}RT_{2,PC}RT_{6,SE}RT_{2,PC}RT_{6,SE}RT_{1,PC}RT_{1,PC}\} \tag{24}
\]

\[
CSR_2^\prime = \{RT_{2,ER}RT_{3,SE}RT_{1,SE}RT_{6,SE}RT_{2,PC}RT_{6,SE}RT_{2,PC}RT_{1,PC}\} \tag{25}
\]

where 4 in \(RT_{4,ER}\) represents the resource belongs to the fourth category, which is knowledge resources, and \(ER\) represents the resource belongs to the knowledge resources of enterprises.

The result of rough selection is shown as follows:

\[
CSR_1 = \{RU_{12}RU_{21}RU_{34}RU_{45}RU_{53}RU_{64}RU_{75}RU_{86}RU_{96}RU_{106}\} \tag{26}
\]

\[
CSR_2 = \{RU_{13}RU_{26}RU_{25}RU_{42}RU_{43}RU_{65}RU_{77}RU_{41}\} \tag{27}
\]

The result of rough selection is subdivided through SFLA and Improved SFLA, and independently running is implemented 40 times. By comparing the average value and the optimal value of each search of \(CSR_1^\prime\) and \(CSR_2^\prime\), we can get the graphs as shown in figure 6 and figure 7.
Figure 6. The optimization result of $CSR_1'$

Figure 7. The optimization result of $CSR_2'$

From figure 6 and figure 7 above, we can draw the conclusion that the optimization ability of the improved SFLA is obviously higher than that of standard SFLA.

6. CONCLUSIONS

In recent years, with the development of the Cloud Computing theory, the concept of cloud manufacturing has gradually been brought to our world. Cloud manufacturing is the landing and extending of Cloud Computing in the manufacturing industry, and shows the concept of “distributed resources being integrated for one task” and “integrated resources being distributed for services”. It realizes the high degree of sharing and utilization of the manufacturing resources by integrating and centralized managing the distributed resources with many to many service mode to improve the enterprise production efficiency, which can provide users with higher satisfaction an more environmentally friendly products and services. In order to solve the convergence defects in SFLA and the resource matching problem in cloud manufacturing platform, the model of cloud manufacturing platform resource matching is proposed in this paper based on the SFLA granularity optimization. Simulations show that the convergence of the improved SFLA has been greatly improved, and also the improved SFLA has better optimization ability in the application of cloud manufacturing platform resource matching model.

ACKNOWLEDGEMENT

This work was supported by the Zhejiang Provincial Natural Science Foundation (Grant No. Z15F020005), the National Natural Science Foundation of China (Grant No.51675490), the Department of Science and Technology of Zhejiang Province (Grant No. 2015C31091), and the Department of Science and Technology of Zhejiang Province (Grant No. 2016C31116), the Department of Science and Technology of Zhejiang Province (Grant No. 2017C31050), the Department of Science and Technology of Zhejiang Province (Grant No. 2017C31063), and the Department of Science and Technology of Zhejiang Province (Grant No. 2017C33018).

REFERENCES