A Multi-Objective Chaotic Particle Swarm Optimization Algorithm Based on Improved Inertial Weights

Zhi-yuan PAN, Da-min ZHANG\*, Dong LIU, Jun YANG and Juan-min CHEN

College of Data and Information engineering, Guizhou University, Guiyang, Guizhou, 550025, PR China

*Corresponding author

Keywords: Multi-Objective particle swarm optimization algorithm, Logistic map, Elite archiving mechanism, Mutation probability.

Abstract. In order to enhance the convergence and distribution of multi-objective particle swarm algorithm, an improved multi-objective particle swarm optimization algorithm was proposed. Linear decreasing inertia weight was used to update. The method can improve the deficiency that the algorithm falls into local optimal easily. The improved Logistic mapping was used to increase ergodicity of the particles. The method can expand the search scope. At the same time, the elite archiving mechanism and the mutation probability were introduced to increase the disturbance. The method can improve the local optimal. Compared with the real Pareto front, NSGA-II and MOEAD algorithm, the simulation shows that the algorithm proposed in the paper is effective.

Introduction

There are many optimization problems in real life, and meanwhile, the optimization problems need to optimize multiple objectives at the same time. In general, the multiple objective functions in the same problem are contradictory and interaction\[^1\]. So the final results are a series of compromise solutions being called Pareto optimal solutions sets or non-dominated solution sets.

Because particle swarm optimization (PSO) has the features of less parameter, good optimization performance, and fast convergence speed. It has been widely concerned and been successfully applied in many fields such as function optimization, data mining and wireless sensor network in recent years.

Focusing on the research status at home and abroad, the main research results are summarized as improving the updating strategy of external archive, the selection strategy of global guide, combining the particle swarm algorithm with other algorithms and so on. In literature [2], the external particles were used to guide the flight of the population particles, and the optimal solutions in the group search process were stored in a file. The algorithm is easy to fall into the local optimum being used to optimize multi-peak function. In literature [3], \(\varepsilon\)-domination was introduced to determine the global extremum of particles. The method can improve the diversity and the uniformity of particles distribution. In literature [4], the preference \(\varepsilon\)-Pareto domination was introduced to short search time and enhance convergence. In literature [5], a dynamic domain strategies and new particle memory strategies were adopted in the algorithm. In literature [6], a multi-objective interactive PSO algorithm was proposed. To maximize the non-dominated solutions with the adaptive grid mechanism, adaptive mutation operation and user decision functions were used in the algorithm. In literature [7], Pareto entropy was proposed to assess population diversity and current evolutionary states. The evolution strategy can be designed to balance the convergence and diversity with the Pareto entropy. In literature [8], a multi-sub-population co-evolution mechanism was proposed. An external archive and elite learning strategies were introduced to improve distribution and convergence of the algorithm. In literature [9], Chaotic sequence was used to initialize population. The population can be more evenly distributed in the decision space with the method.

According to the above researches, a chaotic particle swarm multi-objective algorithm based on linear decreasing inertia weight was proposed in this paper. Chaos mapping was used to optimize...
the optimal location, Elite archiving strategy was used to archive non-dominated solutions, the
crowd distance between particles was calculated and in descending order, the mutation probability
was introduced to perturb the particles to increase convergence and diversity of the algorithm.

The paper is organized as follows: The theoretical basis is described in Section 2. The improved
algorithm is presented in Section 3. Simulation results and analysis are provided in Section 4.
Conclusions are drawn in Section 5.

Theoretical Basis

Multi-Objective Optimization

Generally, a multi-objective optimization problem with $d$ decision variables and $k$ target variables
can be expressed as follows\[^{[10]}\]:

$$
\begin{align*}
\min & \quad y = (f_1(x), f_2(x), \ldots, f_k(x))^T \\
\text{s.t.} \quad & g_i(x) \leq 0, \ i = 1, 2, \ldots, q \\
& h_j(x) = 0, \ j = 1, 2, \ldots, p
\end{align*}
$$

(1)

Where $x = (x_1, \ldots, x_d) \in X \subset \mathbb{R}^d$ is the $d$-dimensional decision vector, $X$ is the $d$-dimensional decision space. $y = (y_1, \ldots, y_k) \in Y \subset \mathbb{R}^k$ is the $k$-dimensional target vector, $Y$ is the $k$-dimensional target space. Objective
function $F(x)$ is defined as mapping from the $k$ decision space to the target space.

Basic Particle Swarm Optimization Algorithm

The basic particle swarm optimization algorithm (PSO) is an optimization algorithm based on
swarm intelligence theory, which was proposed by Kennedy and Eberhart in 1995\[^{[9]}\]. The PSO
simulates the bird’s behavior of randomly searching for food\[^{[11]}\]. In PSO algorithm, every particle
represents a potential solution and it has a fitness value determined by objective functions.
Meanwhile every particle has a velocity to determine the direction and distance of “flying”. The
particles can follow current optimal particle to find the optimal solution by iteration. The update
formula is as follows:

$$
v_{id} = \omega \cdot v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id})
$$

(2)

$$
x_{id} = x_{id} + v_{id}
$$

(3)

Where $v_{id}$ and $x_{id}$ represent current velocity and position of the $i$ particle. $p_{id}$ is personal best
particle. $p_{gd}$ is global best particle. $\omega$ is inertia weight of particle. $c_1$ and $c_2$ are learning factors.
$r_1$ and $r_2$ are random numbers distributed in $[0, 1]$.

Generally, at the beginning of iteration, the larger inertia weight is more beneficial for global
search, and at the later stages of iteration, smaller inertia weights can enhance local searching
ability. A method of linear decreasing inertia weight is used in this paper, and defined as follows:

$$
\omega = \omega_{min} + (\omega_{max} - \omega_{min}) \frac{MaxIt - it}{MaxIt}
$$

(4)

In the above formula, $it$ is current the population iteration. $MaxIt$ is the total number of
population iterations. $\omega_{min}$ is the minimum inertia weight. $\omega_{max}$ is the maximum inertia weight.

A Multi-Objective Chaotic Particle Swarm Optimization Algorithm Based on Improved
Inertial Weights

Chaos Optimization

The chaos optimization achieves global optimum through the ergodicity of chaotic mapping. The
method can prevent the algorithm from falling into local optimum. An improved chaotic Logistics mapping function is used in this paper, and it is defined as:

\[ x_{n+1} = 1 - 2x_n^2 \]  \hspace{1cm} (5)

Where \( x_n \) is in the range of \([0,1]\). In this paper, the optimal solutions in the archive are mapped into the definition domain of formula (5). The chaotic sequences are generated by iteration according to the mapping formula, and then are returned to the original solution space by inverse mapping. The fitness value of the new solutions can be calculated and the best solutions are released into the archive.

**Elite Archiving Mechanism**

In traditional single-objective evolutionary algorithms, the individual with large fitness tends to be elite individual which are directly copied to the next generation without crossover and mutation and so on. According to the elite archival strategy, the superiority or inferiority particle is determined based on the corresponding dominated degree of target vector of particle. If the new particle dominates the particle in the archive, the dominated particle can be replaced by the new particle. If the new particle is dominated by particle in the archive, the new particle can't be put into the archive. If the new particle and the particle in the archive are not dominated each other, the new particle can be put into the archive.

**Mutation Strategy**

In the early stages of search, it is not easy to obtain a local optimum. Oppositely, as the iteration progress, it is very difficult for particle to jump out of the local optimum based on its knowledge and social behavior. Above this, a mutation strategy is introduced. The particle position can be disturbed in a small range according to a certain probability. The particle mutation probability being called \( P_m \) is determined according to the number of iteration. The mutation probability and disturbance formulas are as follows:

\[ P_m = (1 - (it - 1) / (MaxIt - 1))^2 \]  \hspace{1cm} (6)

\[ x_i^d = x_i^d + (x_{\text{max}}^d - x_{\text{min}}^d) \times \sigma \]  \hspace{1cm} (7)

Where \( x_{\text{max}}^d \) and \( x_{\text{min}}^d \) are the upper and lower bounds of the particle in the d-dimension. \( \sigma \) is the uniformly distributed random number between \([0,1]\). In formula (6), the mutation probability of particles increases with the number of iteration increases. At the later iteration, the probability is gradually reduced, which can effectively avoid the error judgment for local optimum and mutation.

**Crowding Distance**

Crowding distance is used to estimate the density of other solutions around a solution. For every objective function, the non-inferior solutions of the elite solution set are sorted in descending order according to the fitness value. Let the distance between the first and last particles is infinite, then calculate the crowding distance. Being inspired by Euclidean distance, an improved crowding distance is defined as follows:

\[ \text{crowd}_i = \sum_{m=1}^{M} \frac{(f_m(i-1) - f_m(i+1))^2}{f_m^{\text{max}} - f_m^{\text{min}}} \]  \hspace{1cm} (8)

Where, \( f_m(i-1) \) and \( f_m(i+1) \) represent the fitness value of two adjacent particles in m-dimensions respectively. \( f_m^{\text{max}} \) and \( f_m^{\text{min}} \) represent the maximum and minimum fitness values of particle in m-dimensions. The smaller the crowded distance, the more density and less diversity the solutions. Oppositely, the larger the crowded distance is, the less density and more diversity the solutions are.
The Algorithm Flow

The steps of the multi-objective particle swarm algorithm process are as follows:

Step 1: Initialize the population size \( N \) and the number of iteration, set the parameters of the algorithm and the number of chaotic optimization.

Step 2: Randomly initialize the velocity and position of particles, and at the same time, remove two particles into the archive at least.

Step 3: Calculate the fitness value of particle, put the non-inferior solution which satisfies the Pareto optimal into the archive and update the non-inferior solution.

Step 4: Calculate the crowding distance of particles and sort in descending order. Update the individual extremum and decide whether to replace the update according to Pareto dominance relationship.

Step 5: Optimize optimal position \( p_{gd} \) with chaos. Update the position of new generation according to (2) and (3) and disturb the particles in the elite archive according to (6).

Step 6: If the maximum number of iteration is reached, all non-inferior solutions are obtained. Otherwise, go to the step 3.

Simulation Comparison

In order to verify the performance of the algorithm, six standard test functions SCHAFFER, ZDT1-ZDT6\(^{(14)} \) are selected except ZDT5 which is an Boolean function and needs to be re-encoded in binary. The functions are tested with NSGA-II algorithm, MOEAD algorithm and CMOPSO algorithm. The results are compared with the real Pareto front.

Performance Index

In order to evaluate the convergence and distribution of the solutions, the convergence index \( \gamma \) and the distribution index \( SP \) are used in this paper\(^{(15)} \). Let \( Z \) is the non-dominated solution obtained by the algorithm, \( p_r \) is the real Pareto optimal solution set.

1. The convergence index \( \gamma \) which is used to evaluate the approximation between \( Z \) and \( p_r \). It is as follow:

\[
\gamma = \left( \sum_{i=1}^{1} d_i \right) / (|Z|)
\]

Where \( d_i \) represents the minimum Euclidean distance between the \( i \)th particle and \( p_r \) in the target space. The smaller \( \gamma \) is, the closer the non-dominated solution is to the real Pareto front, and the better convergence of the algorithm is.

2. The distribution index \( SP \) which is used to evaluate the distribution of the solution obtained by the algorithm. It is as follow:

\[
SP = \frac{1}{|Z| - 1} \sum_{i=1}^{1} \left( \bar{d} - d_i \right)
\]

\[
d_i = \min_{j \neq (i, k)} \left\{ \sum_{n=1}^{n} |f_n(x_j) - f_n(x_i)| \right\}
\]

Where \( \bar{d} \) is the average of \( d_i \). \( k \) is the number of objective function. \( n \) is the number of the non-inferior solution. The smaller \( SP \) is, the more uniform the Pareto front is, and the better distribution of the algorithm is. If \( SP = 0 \), it indicates that the front corresponding to the non-dominated solutions set obtained by the algorithm is completely uniform.
Simulation Results and Analysis

Set the population size of all algorithms to be 50, the maximum number of iteration to be 100, and the archive size to be 100. Every algorithm runs 10 times independently. The test functions used are shown in Table 1.

<table>
<thead>
<tr>
<th>Code</th>
<th>Objective functions to be minimized</th>
<th>Variables</th>
<th>Objectives</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHAFER</td>
<td>$f_1 = x^2$, $f_2 = (x-2)^2$</td>
<td>1</td>
<td>2</td>
<td>$x \in [-10^3, 10^3]$</td>
</tr>
</tbody>
</table>
| ZDT1   | $f_1 = x_1$  
$f_2 = g(x)[1 - \sqrt{x_1} / g(x)]$  
$g(x) = 1 + 9 \left( \sum_{i=2}^{n} x_i \right)^{(n-1)}$  | 30        | 2          | $x \in [0,1]$       |
| ZDT2   | $f_1 = x_1$  
$f_2 = g(x)[1 - (x_1 / g(x))^2]$  
$g(x) = 1 + 9 \left( \sum_{i=2}^{n} x_i \right)^{(n-1)}$  | 30        | 2          | $x \in [0,1]$       |
| ZDT3   | $f_1 = x_1$  
$f_2 = g(x)[1 - \sqrt{x_1} / g(x) - x_1 \sin(10 \pi x_1) / g(x)]$  
$g(x) = 1 + 9 \left( \sum_{i=2}^{n} x_i \right)^{(n-1)}$  | 30        | 2          | $x \in [0,1]$       |
| ZDT4   | $f_1 = x_1$  
$f_2 = g(x)[1 - \sqrt{x_1} / g(x)]$  
$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n} \left( x_i^2 - 10 \cos(4 \pi x_i) \right)$  | 10        | 2          | $x \in [0,1]$       | $x_i \in [-5,5]$ $i = 2, ..., n$ |
| ZDT6   | $f_1 = \exp(-4x_1) \sin^3(6\pi x_1)$  
$f_2 = g(x)[1 - (x_1 / g(x))^2]$  
$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n} \left( x_i^2 - 10 \cos(4 \pi x_i) \right)$  | 10        | 2          | $x \in [0,1]$       |

The front of non-inferior solution of the standard test functions are shown in Figure 1 to Figure 6 and are compared with the real Pareto front. The running time, convergence index $\gamma$ and distribution index $SP$ are shown in the Table 2 to Table 4.
Table 2. The running time [second].

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CMOPSO</th>
<th>MOEAD</th>
<th>NSGA-II</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHAFFER</td>
<td>3.3867</td>
<td>36.8049</td>
<td>26.2543</td>
</tr>
<tr>
<td>ZDT1</td>
<td>2.3132</td>
<td>21.1093</td>
<td>23.7799</td>
</tr>
<tr>
<td>ZDT2</td>
<td>2.7718</td>
<td>19.7723</td>
<td>37.7379</td>
</tr>
<tr>
<td>ZDT3</td>
<td>1.0341</td>
<td>24.3322</td>
<td>70.0710</td>
</tr>
<tr>
<td>ZDT4</td>
<td>2.3726</td>
<td>19.2741</td>
<td>36.6331</td>
</tr>
<tr>
<td>ZDT6</td>
<td>0.4319</td>
<td>17.8783</td>
<td>33.2695</td>
</tr>
</tbody>
</table>

Table 3. The convergence index $\gamma$.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CMOPSO</th>
<th>MOEAD</th>
<th>NSGA-II</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHAFFER</td>
<td>0.0018</td>
<td>1.9708</td>
<td>0.0029</td>
</tr>
<tr>
<td>ZDT1</td>
<td>0.1233</td>
<td>1.8550</td>
<td>0.3441</td>
</tr>
<tr>
<td>ZDT2</td>
<td>0.2277</td>
<td>1.9871</td>
<td>0.6134</td>
</tr>
<tr>
<td>ZDT3</td>
<td>0.0105</td>
<td>7.2397</td>
<td>0.2602</td>
</tr>
<tr>
<td>ZDT4</td>
<td>0.0028</td>
<td>5.4741</td>
<td>1.7583</td>
</tr>
<tr>
<td>ZDT6</td>
<td>0.1425</td>
<td>0.0457</td>
<td>1.4758</td>
</tr>
</tbody>
</table>

Table 4. The distribution index $SP$.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CMOPSO</th>
<th>MOEAD</th>
<th>NSGA-II</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHAFFER</td>
<td>0.0281</td>
<td>0</td>
<td>1.1446</td>
</tr>
<tr>
<td>ZDT1</td>
<td>0.0062</td>
<td>37.7808</td>
<td>0.4442</td>
</tr>
<tr>
<td>ZDT2</td>
<td>0.0090</td>
<td>0</td>
<td>0.4678</td>
</tr>
<tr>
<td>ZDT3</td>
<td>0.0160</td>
<td>49.7469</td>
<td>0.5973</td>
</tr>
<tr>
<td>ZDT4</td>
<td>0.0067</td>
<td>19.5889</td>
<td>0.2965</td>
</tr>
<tr>
<td>ZDT6</td>
<td>0.2358</td>
<td>0</td>
<td>0.5007</td>
</tr>
</tbody>
</table>

From Table 2, the CMOPSO converges faster than the other two algorithms. Especially, the convergence speed of the CMOPSO is 70 times that of NAGS-II for ZDT3.

For SCHAFFER, it can be seen from the tables that the performance of CMOPSO is better than the others. It can be seen from the Figure 1 that the distribution of the non-inferior solution is also better than the others.

For ZDT1, the non-inferior solution obtained by CMOPSO completely cover the real Pareto front while the other algorithms have non-coinciding parts. That is to say, CMOPSO is better distribution than other algorithms.

For ZDT2, all of the algorithms coincide with the real Pareto front, but as can be seen from the Figure 2, CMOPSO obtains the most non-inferior solution and is more uniform, covers more points. The conclusion can be confirmed from the data of distribution and convergence. That is to say, the algorithm is superior to other algorithms.

For ZDT3 and ZDT4, the algorithm covers the real Pareto front. The gap between MOEAD and the real Pareto front is more, which verifies the convergence of CMOPSO.

In ZDT6, all algorithms have solutions that are not covered with the Pareto front, but overall, the algorithm has more overlapping numbers, more coverage, and better performance than others.
For MOEAD, the distribution index $SP$ is 0, which is better than others in ZDT2, ZDT6 and SCHAFFER. But it can be seen from the figures that the gap between MOEAD and the real Pareto front is larger. That is to say, CMOPSO is superior to MOEAD.

Based on the above analysis, the convergence and distribution of the algorithm proposed in the paper are better than that of the other two algorithms. The experiments confirm the effectiveness of CMOPSO algorithm.

**Conclusion**

In order to improve the convergence and distribution of multi-objective particle swarm algorithm, a chaotic particle swarm optimization algorithm based on linear inertia weight was proposed. The speed was updated with the linear decreasing inertia weight method; the searched non-inferior solution was stored in elite files and optimized with the chaotic maps. At the same time, the mutation probability is added to carry out perturbation, expand the search range, and improve the algorithm's insufficiency in local optimum. Simulation results show that the convergence and distribution of the algorithm proposed in the paper are improved.

**Acknowledgement**

This research was financially supported by the Science Foundation of Guizhou Province (No. [2017]1047). The authors also gratefully acknowledge the useful suggestions of the reviewers and editors.

**References**


[4] Li Jing, Huang Tianmin, Chen Shangyun, Multi-objective particle swarm optimization based on angle preference for $\varepsilon$-Paerto domination, Journal of Xihua University (Natural Science Edition), 2018(02) 70-74[2018-04-01].


[14] Han Zhitao, Jing Yuanwei, Duan Xiaodong, Zhang Siyi, Relative degree and linearization of general nonlinear system, Control and Decision, 2006, 21(9) 1065-1067.