Universe Scientific Publishing

ORIGINAL ARTICLE

A Multi-objective Particle Swarm Optimization Algorithm Based on Reverse Learning

Guanglei Wen*, Huimin Ge, Hongpeng Li, Jiahui Meng, Qingyuan Zhao, Yu Zhang

Shijiazhuang Tiedao University, Shijiazhuang 50043, China. E-mail: 1919294055@qq.com

Abstract: In order to solve the contradiction between population diversity and convergence in particle swarm optimization algorithm, in this paper, a particle swarm optimization algorithm with reverse learning is proposed. On this basis, the values of learning factor and constraint factor parameters are modified, and the linear decreasing weight strategy was adopted. By modifying the learning factor and the constraint factor, the algorithm improves the particle optimization ability. It balances the global search and local search of the particle, and the convergence speed is improved by using the inertia weight. When it is detected that the algorithm falls into the local optimal region, the position information of these poor particles is used to guide some particles to reverse learning at a faster flight speed, and the particles are quickly pulled out of the local optimal region. The reverse learning process can not only improve the diversity of particle population, but also ensure the global detection ability of the algorithm. Experimental results show that, compared with the basic MOPSO algorithm, this algorithm has fast convergence speed and high solution accuracy in function optimization.

Keywords: Particle Swarm Algorithm; MOPSO; PSO; Test Ctions

1. Introduction

Particle swarm optimization was proposed by Kennedy, *et al* in 1997. It was first applied to single-objective optimization problems, which showed good performance. However, most of the optimization problems in real life are multi-objective optimization problems. Particle swarm optimization algorithm has the advantages of fast search speed, high efficiency and simple algorithm, so it is widely applied to the optimization of multi-objective problems. Therefore, a large number of multi-objective particle swarm optimization algorithms are proposed. In order to improve the diversity of the population and the convergence of the algorithm, Liu Ming, *et al*[1] proposed a multi-objective particle swarm optimization algorithm based on the regular competitive learning mechanism, which combined the multi-objective particle swarm optimization algorithm with the competitive learning mechanism to maintain the diversity of the population, effectively improving the convergence of the algorithm; Chen, *et al*[2] proposed a multi-objective decomposition particle swarm optimization algorithm (D-CLMOPSO) based on comprehensive learning strategy, which was used to solve multi-objective problems to avoid premature convergence. The archiving mechanism is used to store the non-dominant solution and polynomial variation in the optimization process to avoid the algorithm falling into the local optimal, but it may not converge to the complete Pareto front when dealing with complex multimodal problems; Li, *et al*[3]

doi: 10.18686/esta.v7i2.138

proposed a multi-objective particle swarm optimization algorithm (SSlMOPSO) using the quadratic reinforcement learning strategy. By using the speed-free multi-objective particle swarm framework and integrating the decomposition strategy into the multi-objective particle swarm algorithm, the two-time reinforcement learning of particles can be realized to enhance the algorithm's ability to jump out of local optimal. It also improves the diversity of the population.

Thus, a large number of outstanding scholars on the improved multi-objective particle swarm optimization algorithm get the excellent achievements, but the multi-objective particle swarm optimization (PSO) algorithm is easy to fall into local optimum and precocious defect still haven't been solved well, thus improving population diversity and convergence algorithm is still an area worthy of study. In view of the prematurity of particle swarm optimization algorithm, reverse learning strategy was introduced into MOPSO algorithm in this paper to improve the searching ability of the algorithm.

2. Basic theory

2.1 Basic particle swarm optimization

The standard PSO algorithm is a random search algorithm derived from the bird foraging model, that is, if there is only one piece of food in a certain area, the optimal strategy is to find the area around the bird closest to the food. In this problem, there is a bird of space position corresponding to the problem of the solution, and the bird is called the "particles". Each "particle" has its own velocity and position (decided to birds flying direction and distance), all the particles "have" in each iteration a adaptive value is determined by the function of optimization, then all particles with the optimal particle in all within the scope of the search the solution space. During each iteration, the particle updates itself by tracking two extremes. An extremum is an individual extremum, the optimal solution is found by the particle itself. The other extreme is the optimal solution of the whole particle population, which is called the global extreme. The particle then updates its position and velocity through individual extremum and global extremum. The updating formula of position and speed is as follows:

$$v_{id}^{k+1} = v_{id}^{k} + c_1 rand_1^k\left(pbest_{id}^k - x_{id}^k\right) + c_2 rand_2^k\left(gbest_{id}^k - x_{id}^k\right) \quad (1)$$

$$x_i d^{(k+1)} = x_i d^k + v_i d^k \quad (2)$$

Type: v_{id}^k is the d-dimensional velocity of particle I in the KTH iteration, $v_{id}^k \in [-v_{max}, v_{max}]$; c1, c2 is the acceleration factor (or learning factor), $rand_1^k$ 、$rand_2^k$ is the random number between$^{[0,1]}$ in the KTH iteration, x_{id}^k is the d-dimensional position of particle I in the KTH iteration, $pbest_{id}$ is the position of the individual extremum of particle I in the DTH dimension, $gbest_d$ is the global optimal value of the whole particle population.

C1 and c2 represent learning factors and the learning ability of particles to their own historical and population optimal locations. When c1 = 0 and c2 ≠ 0, the algorithm has strong global convergence ability, but it is easy to fall into local optimal. When c1 ≠ 0, c2 = 0, the global convergence rate of the algorithm is slow. So by constantly changing their values, we can find the most suitable result.

Inertia weight is used to describe the influence of the particle velocity on the current generation, value is larger, the global optimization ability is strong, the local optimization ability is weak; on the contrary, local optimization ability will be gradually strengthened. In order to achieve a balance between search speed and precision, the algorithm generally has a higher global search capability in the early stage and a stronger local search capability in the later stage. Therefore, dynamic can get better optimization results than fixed values.

The most common method is to change the weight of inertia by linear decline. The formula is as follows:

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min}) \times \frac{it}{MaxIt} \quad (3)$$

Where, ω_{max}、ω_{min} are the maximum and minimum values of inertia weight respectively, MaxIt is the maximum number of iterations, and it is the current number of iterations.

Because of the change of inertia weight of linear decrease only considers the change of iteration number, this method cannot deal with complex and nonlinear problems well. Therefore, factors such as particle distribution can be considered on this basis.

2.2 MOPSO

MOPSO is a PSO-based multi-objective

optimization algorithm, which generally includes optimization of two or more functional objectives with constraints between objective functions. MOPSO can be described mathematically:

$$\begin{aligned} &\min f(x) = (f_1(x), f_2(x), \cdots, f_m(x)) \\ &g_j(x) < 0, j = 1,2,\cdots,A \\ &h_k(x) = 0, k = 1,2,\cdots,B \\ &x_d^{min} < x_d < x_d^{max}, d = 1,2,\cdots,D \\ &\text{s.t. } 0 \le x_i \le 1, i = 1,2,\cdots,D \end{aligned}$$

Where, M is the number of objective functions, $f_i(x)$ is the ith objective function, and $g_j(x)$、$h_k(x)$ are the constraints of the kthand jth inequalities. A and B are the sum of$g_j(x)$、$h_k(x)$constraints, respectively. x_d^{min}、x_d^{max} are the upper and lower limits of the d-dimensional positions of particles respectively.

3. Inverse learning

3.1 Definitions

The concept of reverse learning was first proposed by Tizhoosh[4] in 2005. By finding the reverse solution of a feasible solution of a problem, and comparing the original feasible solution with its reverse solution, a better solution is selected as the optimal learning strategy for the next generation of individuals. In reverse learning, the definition of reverse point and reverse learning optimization[5] is as follows:

Reverse point: Assuming $x = x_1, x_2, \cdots, x_D$ is any point in the d-dimensional space, and$x_1, x_2, \cdots, x_D \in R, x_i \in [a_i, b_i]$, then the global reverse point corresponding to x is defined as $ox = (ox_1, ox_2, \cdots, ox_D)$, where

$$ox_i = a_i + b_i - x_i \quad (4)$$

Back learning optimization[5]: Suppose $x = (x_1, x_2, \cdots, x_D)$ is any point in d-dimensional space, and its global reverse point is defined as

$$ox = (ox_1, ox_2, \cdots, ox_D) \quad (5)$$

For the problem of minimization, if $f(ox) < f(x)$, then $x = ox$, which is called backward learning optimization.

3.2 Reverse solution

Back learning can expand the search range of the population. Under a certain probability, solving the generated particles, generating reverse solutions and comparing them to better find the optimal solution are crucial. Generating the reverse solution can be explained as follows:

Let the current particle be $X_{i,j}$, and the corresponding inverse solution

$$X_{i,j}^* = k(a_j + b_j) - X_{i,j} \quad (6)$$

Where $a_j = \min(X_{i,j}), b_j = \max(X_{i,j}), i = 1,2,...,NP, j = 1,2,3,...,D, X_{i,j} \in [a_j, b_j]$, NP is the number of generated particles, D is the number of dimensions, and $k \in [0,1]$ is the generalization coefficient, which is used to generate different inverse solutions.

Algorithm 1: The Framework of Multi-Objective Particle Swarm Algorithm

Input: nPop, Population size; MaxIt, maximum iterations; c1、c2, Learning factor

Output: The best solution for the task.

```
Initialize the population
Calculate fitness function
Update the best individual
Identify the leader
Archive and create grid
for it=1:MaxIt
    for i=1:nPop
        Get the leader
        If rand>gl
            Use formula (1) (2) to update the particle position and velocity
            Calculate fitness value
        End
        If rand<gl
            Get population speed and position
            Calculate a_j、b_j
            Use formula (6) to obtain a new position
            Position update, calculate fitness value
            Calculate the fitness value and optimal selection before and after the reverse solution respectively
        End
        Apply mutations, solve updates
    End
    Screening and archiving based on dominance
    Update grid
End
```

4. Experimental simulation and result analysis

4.1 Test function

In Matlab2019 environment, the four test functions of ZDT1, ZDT2, ZDT3, ZDT6 in **Table 1** are simulated and verified. The next step is to compare the Pareto obtained in Matlab2019 with the improved optimization algorithm in this paper with the Pareto obtained in Matlab2019 with the basic MOPSO algorithm.

name	
ZDT1	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{f_1(x)/g(x)}\right]$ $g(x) = 1 + \frac{9(\sum_{i=2}^{n} x_i)}{(n-1)}, x = (x_1,\cdots,x_2)^T \in [0,1]^n$
ZDT2	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \left(\frac{f_1(x)}{g(x)}\right)^2\right]$ $g(x) = 1 + \frac{9(\sum_{i=2}^{n} x_i)}{(n-1)}, x = (x_1,\cdots,x_2)^T \in [0,1]^n$
ZDT3	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - g\left(\frac{f_1(x)}{g(x)}\right) - g\left(\frac{f_1(x)}{g(x)}\right)\sin(10\pi x_1)\right]$ $g(x) = 1 + \frac{9(\sum_{i=2}^{n} x_i)}{(n-1)}, x = (x_1,\cdots,x_2)^T \in [0,1]^n$

ZDT6	$\min f_1(x_1) = 1 - \exp(-4x_1 \sin^6(6\pi x_1))$ $\min f_2(x) = g(1 - (f_1/g)^2)$ $g(x) = 1 + 9\left(\left(\sum_{i=2}^{m} x_i/(m-1)\right)^{0.25}\right)$ $s.t.0 \le x_i \le 1, i = 1,2,\cdots,10$

4.2 Simulation analysis

Take the population size nPop = 10, the maximum number of iterations MaxIt = 50, c1 = 1.41, c2 = 2, gl = 0.01 for simulation experiments. The simulation results are as follows:

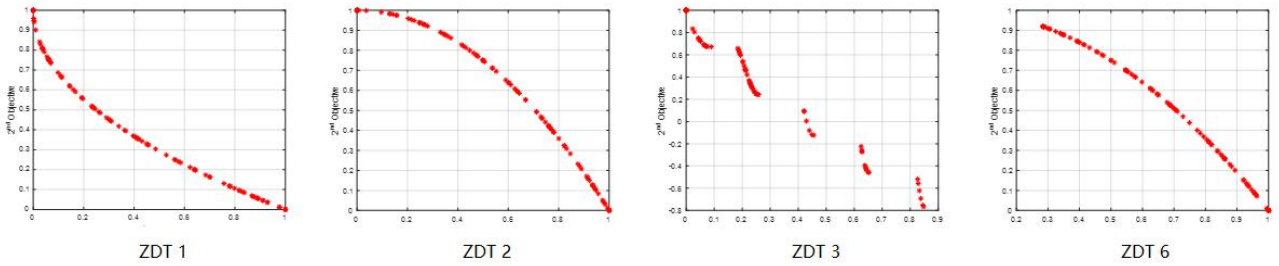

Figure 1. Multi-objective particle swarm optimization operation curve.

The ideal pareto front curve obtained by ZDT1 simulation is concave; the ideal pareto front curve obtained by ZDT2 and ZDT6 is convex; the ideal pareto front curve obtained by ZDT3 is discontinuous. The simulation results of ZDT1, ZDT2, ZDT3, and ZDT6 can all obtain a pareto front with good effect, and the stability of the curves obtained by ZDT3 and ZDT6 has been significantly improved.

5. Conclusion

In order to alleviate the contradiction between the convergence speed of the MOPSO algorithm and the "premature" problem, and to further improve its optimization performance, this paper proposes a particle swarm optimization algorithm with reverse learning. Value is modified and a linear decreasing weight strategy is used. The algorithm adjusted the inertial weights, learning factors, and constraint factors to balance the search ability of particles, and it improves the convergence speed; when the algorithm appeared "precocious", it started the reverse learning process. In this process, some particles used their worst individual history. The combined force of the solution and multiple worst solutions of the initial population escapes the local optimum to improve the diversity of the population; after the backward learning is over, the algorithm enters the normal iterative optimization process again. Compared with the basic MOPSO algorithm on the test function, the simulation results show that the algorithm proposed in this paper has stronger global and local optimization capabilities, faster convergence speed, and better stability of the obtained non-dominated solution. However, the improvement of the performance of the algorithm proposed in this paper is not very large. What should do next is continuing to study the MOPSO algorithm to improve the performance of the algorithm.

References

1. Liu M, Dong M, Jing C. Multi-objective particle swarm optimization algorithm based on periodic competitive learning (in Chinese). Journal of Computer Applications 2019; 39(2): 330-335. doi: 10.11772/j.issn.1001-9081.2018061201.
2. Chen Y, Xu Y. Multi-objective decomposition particle swarm optimization algorithm based on comprehensive learning strategy. Microelectronics Computer 2018; 35(10): 75-79.
3. Li H, Zhang P, Liu Z, *et al*. Multi-objective particle swarm optimization algorithm with quadratic reinforcement learning strategy (in Chinese). Journal of Chinese Computer Systems 2018; 39(11): 2413-2418.
4. Tizhoosh HR. Opposition-based learning: A new scheme for machine intelligence. The IEEE International Conference of Intelligent for Modeling. Control and Automation 2005; 695-701. doi: 10.1109/CIMCA.2005.1631345.
5. Zhai J, Qin Y. Opposition-based learning in global harmony search algorithm. Control and Decision 2019; 34(7). doi: 10.13195/j.kzyjc.2017.1743.