2010 3rd International Conference on Computer and Electrical Engineering (ICCEE 2010)
IPCSIT vol. 53 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V53.No.1.43

Particle Swarm Optimization with Particle Similarity Based Mutation

Xin Jin[+] and Yong-quan Liang

College of Information Science and Engineering, Shandong University of Science and Technology
Qingdao, China

Abstract. Particle swarm optimization (PSO) has received increasing interest from the optimization community due to its simplicity in implementation and its inexpensive computational overhead. However, PSO has premature convergence, especially in complex multimodal functions. To overcome the premature convergence problem, a PSO variant with particle similarity based mutation is proposed. The mutation is operated on the positions of the particles and the similarity of the particles is used to determine the particles to mutate. Experiment results show that the proposed method has apparently improved performance in complex multimodal functions and solves the premature convergence problem to some extent.

Keywords: Particle swarm optimization; mutation; particle similarity; swarm intelligence

1. Introduction

Particle swarm optimization (PSO) is a relatively new kind of global search method. This derivative-free method is particularly suited to continuous variable problems and has received increasing attention in the optimization community. PSO is originally developed by Kennedy and Eberhart [1] and inspired by the paradigm of birds flocking. PSO consists of a swarm of particles and each particle represents a potential solution for a problem. Every particle flies through the multi-dimensional search space with a velocity, which is constantly updated by the particle's previous best position and by the global best position found by the whole swarm so far. PSO has been used in many applications as it can be easily implemented and is computationally inexpensive. Many researchers have done a lot of work on the improvements, theoretical analyses and applications of PSO [2-3]. However, even though PSO is a good and fast search algorithm, it has premature convergence, especially in complex multi-peak-search problems.

Simple as it seems, the PSO presents formidable challenge for the people who want to understand it through theoretical analyses. A fully comprehensive mathematical model of particle swarm optimization is still not available by now. One reason for this is that the forces are stochastic, which means the use of standard mathematical tools used in the analysis of dynamical systems is impossible. Another reason is that the PSO is made up of a large number of interacting particles. Although each particle's behavior is simple, understanding the dynamics of the whole is nontrivial. Because of these difficulties, most researchers [4-9] have often been forced to make simplified assumptions in order to obtain models that could be studied mathematically.

Many adjustments and improvements have been made to the basic PSO algorithm over the past decade to deal with the premature convergence problem and to get a high search speed. As the parameters used in PSO are believed to have great influence on the performance of the algorithm, many researches [10-12] had been done to adaptively adjust the parameters for different problems. PSO is hybridized with other methods [12-17] to enhance its ability on a larger number of applications. Unlike the conventional PSO which have identical particles, some researches [18-19] proposed heterogeneous PSO in which the particles have different

[+] Corresponding author.
E-mail address: sdjinxin@gmail.com.

behaviors. The information dissemination of several neighborhood topologies is analyzed theoretically and a PSO with varying topology is proposed in [20]. To avoid the premature convergence problem, the methods to keep the swarm diversity [21-22] were proposed. Several approaches to improve the performance of the PSO were described in [23]. Some of the previous methods improved the general performance of PSO and others improved performance on particular kinds of problems.

To avoid premature convergence of PSO, a new form of PSO with particle similarity based mutation is proposed in this paper. There have already existed some PSO variants with mutation, but the mutation used in those methods is operated on the velocities of the particles or on the global best particles only. In our method, the mutation is operated on the positions of the particles. In the biological world, the species will separate and migrate if they find the density of the species in one place is too high. This phenomenon is represented as mutation of the positions of the particles in this paper. With the purpose of preventing the swarm from losing its diversity rapidly, the mutation is based on the particle similarity. One of the two particles with the largest similarity in the current swarm is selected to mutate. The proposed approach is expected to have better performance and strong capability of escaping from local optima.

The rest of this paper is organized as follows: Section II describes the traditional PSO algorithm. In Section III, the specific methods and procedure of the PSO with particle similarity based mutation are described. Experiment and result analyses are presented in Section IV. Finally, Section V concludes this paper.

2. Particle swarm Optimization Algorithm

A standard particle swarm optimization maintains a swarm of particles which are flying with some velocities in the n-dimensional search space. The particles have no weight and no volume. The velocity of each particle is determined by the particle's cognitive knowledge, that is its personal best position, and social knowledge, the best position find by the whole swarm so far. The current position of each particle is represented by $X_i=(x_{i1},x_{i2,} \cdots, x_{in})$, where n is the dimension of the search space. The personal best position of each particle is $pbest_i=(pbest_{i1}, pbest_{i2}, \cdots, pbest_{in})$, and the current velocity of each particle is $V_i=(v_{i1},v_{i2,} \cdots, v_{in})$. The global best position found by the whole population so far is $gbes\ t= (gbest_1, gbest_2, \cdots, gbest_n)$. To construct the search course, for each particle we update its velocity V_i and position X_i through each variable dimension d using (1)(2) as follows:

$$v_{id}^{t+1} = v_{id}^t + c_1 \times r_1^t \times (pbest_{id}^t - x_{id}^t) + c_2 \times r_2^t \times (gbest_d^t - x_{id}^t) \tag{1}$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{2}$$

In (1) c_1 is a coefficient for individual confidence, c_2 is a coefficient for social confidence, and r_1^t and r_2^t are random real numbers which are uniformly distributed on [0, 1]. Clerc and Kennedy [5] proposed a modified form of PSO that has constriction factor. The velocity update formula is illustrated in (3)(4).

$$v_{id}^{t+1} = \chi \times \left(v_{id}^t + c_1 \times r_1^t \times (pbest_{id}^t - x_{id}^t) + c_2 \times r_2^t \times (gbest_d^t - x_{id}^t)\right) \tag{3}$$

$$\chi = 2 \Big/ \left(\left| 2 - (c_1 + c_2) - \sqrt{(c_1 + c_2)^2 - 4 \times (c_1 + c_2)} \right| \right) \tag{4}$$

According to Clerc's constriction method, c_1 and c_2 are set to 2.05, respectively, and the constriction factor, χ , is approximately 0.7298.

3. PSO with Particle Similarity based Mutation

One problem found in the standard PSO is that it could easily fall into local optima in many optimization problems. One reason for PSO to converge to local optima is that particles in PSO can quickly converge to the best position once the best position has no change. When all particles become similar, there is little hope to find a better position to replace the best position found so far. To overcome the premature convergence of the PSO, the mutation based on the similarity of different particles is used in the proposed algorithm.

3.1 Particle similarity based mutation

The purpose of the mutation in Genetic Algorithm is to generate a better individual that may not be otherwise obtained through normal evolutionary process. Since the mutation operation happens by chance, there is no guarantee that the new individual would be better than the one before mutation. Therefore, a mutation probability is set to control the happening of the mutation and this probability is usually very small. The success of the mutation operator depends on the selection of the mutation probability. If mutation probability is too large, no consistent search path will be followed. As a result, divergence of the search may occur. On the contrary, if the mutation probability is too small, no benefits of mutation operation will be observed. There have been several mutation methods in PSO, but they either operate on the velocities of the particles or on the global best particle only. A new form of mutation is proposed, it operates on the positions of the particles. If a dimension of a particle's position is selected to mutate, it will randomly choose a position in the search space, which is shown in (5), where, r is a random real numbers which is uniformly distributed on [0, 1], *xhigh*, *xlow* are the upper bound and the lower bound of the search space respectively. To avoid the mutation disturbing the normal search course of the PSO, at most one dimension of a particle will mutate in each iteration.

$$x_{id} = xlow + r \times (xhigh - xlow) \tag{5}$$

The biota in nature, such as ants, bees, will separate and migrate to new places if the species density is too high in one place. In the PSO, the premature convergence appears when the diversity of the swarm loses fast. To keep the diversity of the swarm and to learn form the biological phenomenon, particle similarity is proposed. If two particles' distance is short, their similarity is high. The distance between two particles is defined in (6), where x_{ik}, x_{jk} is the position of the particle, D is the dimension of the search space. When the mutation operation happens, it is one of the two particles that with the highest similarity is selected to mutate. In this way, the diversity of the swarm can be preserved better.

$$dis\tan ce_{ij} = \sum_{k=1}^{D} \left(x_{ik} - x_{jk} \right) \tag{6}$$

3.2 The proposed algorithm

Based on the mutation operation discussed above, the PSO algorithm can be modified accordingly. The PSO algorithm with particle similarity based mutation (PSO-M) works as shown in Algorithm 1. The mutation probability controls the happing of the mutation is set to be 0.08 in the algorithm. To compute the similarity of the particles, the swarm is separated into two sub-swarms, each contains the same number of particles. The particles in each sub-swarm are numbered, and the particles from the two sub-swarms that have the same number are chose to compute the similarity. We can find the two particles that their similarity is the highest, and among the two particles the one from the second sub-swarm is selected to mutate. In this way, the particles in the first sub-swarm will never mutate. This reduces the disturbance effect of the mutation operation to some extent. Moreover, for the particles to mutate, only one dimension of each particle is randomly selected to mutate.

```
Algorithm 1: PSO Algorithm with particle similarity based mutation (PSO-M)
1. Initialize a swarm of particles with random positions and velocities on D dimensions in the search space.
2. For each particle, evaluate its fitness of the desired optimization fitness function.
3. Compare particle's fitness evaluation with its personal best fitness. If current fitness value is better than its personal best fitness, then set the personal best fitness equal to the current value, and set $pbest_i$ equal to the current position $X_i$ in D-dimensional search space.
4. Find the particle in the swarm with the best fitness so far, and memorize its fitness value and position $gbest$.
5. Update the velocities and positions of the particles according to the (2)(3)(4).
6. Generate a random real number which is uniformly distributed on [0,1], if it is greater than the mutation probability, go to step 7. Else, compute the similarity of the particles with the same number in different sub-swarms, and choose one particle of the two particles with the highest similarity to mutate according to (5).
7. If a termination criterion is met (usually a sufficiently good fitness or a maximum number of iterations), exit loop. Else, go to step 2.
```

4. Experiments and Results

We have conducted intensive experiments to evaluate the performance of the proposed algorithms. The traditional PSO with constriction factor are used to compare with the proposed approach.

4.1 Test functions and experimental settings

Seven mathematical functions are chosen to test the proposed algorithms. The first 3 test functions listed in TABLE I are unimodal functions and the rest 4 functions are complex multimodal functions with many local optima. All these functions are to be minimized. The dimensions of the functions are all set to 30 and every function's search range, their actual minimum and the acceptable worst solutions are shown in TABLE I.

Table 1 Seven test functions for comparison

Test functions	D	Search range	f_{min}	Accept
$f_1(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2$	30	$[-100,100]^D$	0	0.01
$f_2(x)=\max\{\lvert x_i\rvert,1\le i\le D\}$	30	$[-100,100]^D$	0	0.01
$f_3(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	30	$[-10,10]^D$	0	100
$f_4(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{\lvert x_i\rvert})$	30	$[-500,500]^D$	-12596.5	-10000
$f_5(x)=\sum_{i=1}^{D}[x_i^2-10\cos(2\pi x_i)+10]$	30	$[-5.12, 5.12]^D$	0	50
$f_6(x)=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i)+20+e$	30	$[-32, 32]^D$	0	0.01
$f_7(x)=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]$ $+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ Where, $y_i=1+\frac{(x_i-1)}{4}$, $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<a\end{cases}$	30	$[-50 ,50]^D$	0	0.01

The PSO related parameters for the traditional PSO and the proposed PSO variant are the same. That is, the c_1 and c_2 are both set to 2.05, and the constriction factor, χ, is approximately 0.7298. Moreover, every particle's velocities are constricted within 20% of the search range of the corresponding dimension. In our proposed algorithm, the mutation probability is 0.08. To compare the performances of different methods, the traditional PSO algorithm and the proposed algorithm use the same population size of 40. Moreover, these algorithms use the same maximal number of 2×10^5 fitness evaluations (FEs) for each test function. On the purpose of avoiding stochastic error, we simulate 25 independent trials on each test function and use the average, median, best and worst results and the standard deviation over 25 runs for comparisons. The success rate is also considered, which means the percentage of the runs that get acceptable solutions.

4.2 Experiment results and analysis

The results of the traditional PSO algorithm and the proposed algorithm are shown in TABLE II and TABLE III, where success means the success rate and STD means the standard deviation. It can be seen that every run of the PSO algorithm with particle similarity based mutation can get an acceptable result. However, the traditional PSO can only get the acceptable results for the simple unimodal functions, such as f1, f2 and f3. For the multimodal functions with many local optima, the traditional PSO seldom get an acceptable result, such as f4, f5, and f6. For the simple functions f1 and f2, traditional PSO's results are a little better than the proposed PSO-M algorithm. On the more complex functions, the PSO-M algorithm gets better results. This is because that when the problem is simple, the traditional PSO can get the right results quickly, the mutation operator in the PSO-M algorithm does no good to the search course. But when the problem is more difficult, the traditional PSO has the premature convergence, and may be trapped in local optima. In this circumstance, the PSO-M algorithm can get rid of the premature convergence problem and find the global optima through mutation operation. So the proposed PSO-M algorithm can be used to solve the difficult problems that can not be solved by the traditional PSO algorithm.

Table 2 Experiment results for the traditional PSO algorithm on the 7 test functions.

Function	Success (%)	Best	Mean	Median	Worst	STD
f1	100	9.62E-13	6.14E-11	1.94E-11	7.36E-10	1.45E-10
f2	100	3.55E-08	6.29E-07	2.93E-07	3.15E-06	8.32E-07
f3	100	0.042	8.69	10.20	17.99	4.70
f4	0	-9212.03	-7964.61	-8045.84	-6446.81	734.28
f5	36	24.87	54.72	56.713	94.52	18.63
f6	32	7.99E-15	1.13	1.34	2.32	0.89
f7	68	1.57E-32	0.13	1.58E-32	1.56	0.34

Table 3 Experiment results for the PSO algorithm with particle similarity based mutation on the 7 test functions.

Function	Success (%)	Best	Mean	Median	Worst	STD
f1	100	8.55E-11	1.38E-09	6.90E-10	8.97E-09	1.91E-09
f2	100	6.17E-07	4.96E-06	3.41E-06	1.88E-05	4.82E-06
f3	100	0.06	7.89	11.12	16.59	5.89
f4	100	-11977.3	-11526.5	-11503.5	-10694.2	321.47
f5	100	0	0.28	1.78E-15	2.98	0.67
f6	100	1.51E-14	2.41E-14	2.22E-14	1.00E-13	1.73E-14
f7	100	1.58E-32	1.71E-32	1.64E-32	2.38E-32	1.85E-33

5. Conclusions

The PSO algorithm has the premature convergence problem and may be trapped in local optima, this is in part results form the fact that the diversity of the swarm loses quickly through the search process. To solve this problem, a new form of mutation, the mutation operated on the particles position and considered the similarity of the particles in the swarm is proposed. The experiment results demonstrate that the proposed method can get better results for the more complex problems. So the mutation operation proposed has the ability to help the PSO jump out of local optima and find the global optima and solves the premature convergence problem. As the mutation operation can not guarantee that it generates better new individual every time, although we have considered the particle similarity when selecting the particles to mutate, a mutation probability is provided to control the happening of mutation. We will study when the mutation is really needed and develop an adaptive method to take the mutation operation when it is needed in the future.

6. References

[1] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the 1995 IEEE International Conference on Neural Networks. Part 4 (of 6), Nov 27 - Dec 1 1995, IEEE, Perth, Aust, 1995, pp. 1942-1948.

[2] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization, Swarm Intelligence, 1 (2007) 33-57.

[3] X.-F. Xie, W.-J. Zhang, Z.-L. Yang, Overview of particle swarm optimization, Kongzhi yu Juece/Control and Decision, 18 (2003) 129-134.

[4] E. Ozcan, C.K. Mohan, Analysis of a simple particle swarm optimization system, in: Proceedings of the 1998 Artificial Networks in Engineering Conference, ANNIE, November 1, 1998 - November 4, 1998, ASME, St.Louis, MO, USA, 1998, pp. 253-258.

[5] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation, 6 (2002) 58-73.

[6] K. Yasuda, A. Ide, N. Iwasaki, Adaptive particle swarm optimization, in: System Security and Assurance, October 5, 2003 - October 8, 2003, Institute of Electrical and Electronics Engineers Inc., Washington, DC, United states, 2003, pp. 1554-1559.

[7] B. Brandstatter, U. Baumgartner, Particle swarm optimization - Mass-spring system analogon, in, Institute of Electrical and Electronics Engineers Inc., 2002, pp. 997-1000.

[8] J. Zeng, Z. Cui, Unified model of particle swarm optimization and its theoretical analysis, Jisuanji Yanjiu yu Fazhan/Computer Research and Development, 43 (2006) 96-100.

[9] W. Hu, Z.-S. Li, Simpler and more effective particle swarm optimization algorithm, Ruan Jian Xue Bao/Journal of Software, 18 (2007) 861-868.

[10] R. He, Y.-J. Wang, Q. Wang, J.-H. Zhou, C.-Y. Hu, Improved particle swarm optimization based on self-adaptive escape velocity, Ruan Jian Xue Bao/Journal of Software, 16 (2005) 2036-2044.

[11] C. Kurosu, T. Saito, K. Jin'no, Growing Particle Swarm Optimizers with a Population-Dependent Parameter, in: Neural Information Processing, 2009, pp. 234-241.

[12] C.-S. Zhang, J.-G. Sun, D.-T. Ouyang, Y.-G. Zhang, A self-adaptive hybrid particle swarm optimization algorithm for flow shop scheduling problem, Jisuanji Xuebao/Chinese Journal of Computers, 32 (2009) 2137-2146.

[13] S. Yu, Z. Wu, H. Wang, Z. Chen, A Hybrid Particle Swarm Optimization Algorithm Based on Space Transformation Search and a Modified Velocity Model, in: High Performance Computing and Applications, 2010, pp. 522-527.

[14] P. Kim, J. Lee, An integrated method of particle swarm optimization and differential evolution, Journal of Mechanical Science and Technology, 23 (2009) 426-434.

[15] P.-Y. Yin, F. Glover, M. Laguna, J.-X. Zhu, Cyber Swarm Algorithms - Improving particle swarm optimization using adaptive memory strategies, European Journal of Operational Research, 201 (2010) 377-389.

[16] M.-R. Chen, X. Li, X. Zhang, Y.-Z. Lu, A novel particle swarm optimizer hybridized with extremal optimization, Applied Soft Computing, 10 (2010) 367-373.

[17] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing, 10 (2010) 629-640.

[18] L. Cartwright, T. Hendtlass, A Heterogeneous Particle Swarm, in: Artificial Life: Borrowing from Biology, 2009, pp. 201-210.

[19] C. Hu, X. Wu, Y. Wang, F. Xie, Multi-swarm Particle Swarm Optimizer with Cauchy Mutation for Dynamic Optimization Problems, in: Advances in Computation and Intelligence, Springer Berlin / Heidelberg, 2009, pp. 443-453.

[20] Q.-J. Ni, Z.-Z. Zhang, Z.-Z. Wang, H.-C. Xing, Dynamic probabilistic particle swarm optimization based on varying multi-cluster structure, Ruan Jian Xue Bao/Journal of Software, 20 (2009) 339-349.

[21] J. Jie, J. Zeng, C. Han, Self-organized particle swarm optimization based on feedback control of diversity, Jisuanji Yanjiu yu Fazhan/Computer Research and Development, 45 (2008) 464-471.

[22] Z. Xinchao, A perturbed particle swarm algorithm for numerical optimization, Applied Soft Computing, 10 (2010) 119-124.

[23] T.-Y. Chen, T.-M. Chi, On the improvements of the particle swarm optimization algorithm, Advances in Engineering Software, 41 (2010) 229-239.