International Journal of Grid Distribution Computing
Vol. 8, No.4, (2015), pp. 123-134
http://dx.doi.org/10.14257/ijgdc.2015.8.4.12

Particle Swarm Optimization with Chaotic Maps and Gaussian Mutation for Function Optimization

Dongping Tian[1, 2]

[1] *Institute of Computer Software, Baoji University of Arts and Sciences, Baoji, Shaanxi, 721007, China*
[2] *Institute of Computational Information Science, Baoji University of Arts and Sciences, Baoji, Shaanxi, 721007, China*
{tdp211}@163.com

Abstract

Particle swarm optimization (PSO) is a population-based stochastic optimization that has been widely applied to a variety of problems. However, it is easily trapped into the local optima and appears premature convergence during the search process. To address these problems, we propose a new particle swarm optimization by introducing chaotic maps (tent map and logistic map) and Gaussian mutation into the PSO algorithm. On the one hand, the chaotic map is employed to initialize uniform distributed particles so as to improve the quality of the initial population, which is a simple yet very efficient method to improve the quality of initial population. On the other hand, the Gaussian mutation mechanism based on the maximal focus distance is adopted to help the algorithm escape from the local optima and make the particles proceed with searching in other regions of the solution space until the global optimal or the closer-to-optimal solutions can be found. Experimental results on two benchmark functions demonstrate the effectiveness and efficiency of the PSO algorithm proposed in this paper.

Keywords: *PSO, Tent map, Logistic map, Uniformity, Maximal focus distance, Gaussian mutation, stability*

1. Introduction

Particle swarm optimization (PSO) is a swarm-based stochastic optimization algorithm originating from artificial life and evolutionary computation, which is first introduced by Kennedy and Eberhart [1] and used for the optimization of continuous nonlinear functions. PSO is similar to other population-based evolutionary algorithms in that it is initialized with a population of random solutions, such as ant colony optimization (ACO) [2] and genetic algorithm (GA) [3]. It is unlike most of other population-based evolutionary algorithms, how- ever, in that PSO is motivated by the simulation of social behavior instead of survival of the fittest, and each candidate solution is associated with a velocity. Due to the convenience of realization and promising optimization ability, PSO has been paid much attention to by researchers since its advent and has been widely applied in various optimization problems. However, there are a couple of problems remain to be solved for PSO due to its poor exploration. First, both standard PSO and the various improved versions of PSO, such as HPSO [4], AEPSO [5] and other PSO variants, behave the characteristics of low stability. The reason, just as demonstrated in the literature [5], lies in the non-uniform distribution of the initial particles. Second, similar to other evolutionary algorithms, PSO also tends to get stuck in local optima, especially in solving complex multimodal problems.

Early work by Lu et al [6] puts forward a new adaptive mutation particle swarm optimizer based on the variance of the population's fitness to prevent the premature convergence. He et al. [5] propose an improved particle swarm optimization based on

ISSN: 2005-4262 IJGDC

self-adaptive escape velocity, in which they point out the reason for the low stability of PSO algorithm, i.e., lack of uniform distributed initial particles, but no strategies for solving the problem are provided. Followed by Li et al. [7] integrate PSO with the Cauchy mutation as well as an evolutionary selection strategy together. The core idea is to introduce the Cauchy mutation into PSO algorithm to prevent it from trapping into a local optimum via long jumps made by the Cauchy mutation. Subsequently they investigate several mutation operators (Cauchy, Gaussian and Levy mutation) based on the global best particles, an adaptive mutation operator is then formulated by integrating the three mutation operators described above together at different stages for the best optimization performance [8]. In the meanwhile, Pant et al [9] present two variants of PSO with adaptive mutation, in which both the personal best position and the global best swarm position are mutated by Beta distribution respectively. In addition, the opposition-based learning algorithm is introduced into PSO as so to accelerate the learning and searching process [10], in which the mutation threshold is automatically adapted in terms of the evolutionary information of the global best particles. In order to overcome the pre- mature convergence and low stability of particle swarm optimization algorithm, Li et al. [11] come up with a PSO based on adaptive periodic mutation (APM-PSO), in which the average fitness distance comparing with the sort ascending fitness distance and the inertia weight with periodic mutation are employed to update the velocity of the particles. Recently, a new adaptive mutation operator is formulated based on the fitness variance and space position aggregation degree, and it is implemented at the best position for each particle in each iteration [12]. Besides, an improved particle swarm optimization based on double mutation is constructed [13], which adopts two strategies (to dynamically adjust inertia weight and to introduce mutations) to improve the PSO algorithm. It is important to note that the mutation operator has been extensively applied in PSO algorithm to increase the search diversity and avoid the stagnation in local optima. However, it has significant impact on the performance of PSO by introducing mutation operators, and the parameters used to perform mutation are generally very hard to determine. For this purpose, Wang et al. [14] propose a unified tabu and mutation framework with parameter adaptations in the context of the particle swarm optimizer. In more recent work [15], Xu et al. come up with a new improved particle swarm optimization based on the analysis of the reason why PSO is prone to trap into local minima, in which the mutation strategy is introduced and determined by the variance of the individual concentration and convergence function. In this manner, the global and local search capability can be expected to coordinate and locate the global optimum quickly.

As is well known, chaos [16] is a bounded unstable dynamic behavior that exhibits sensitive dependence on initial conditions and includes infinite unstable periodic motions in nonlinear systems. In recent years, chaos has become much popularized in various hybrid soft computing due to its robustness and specific performance [17-23]. As the representative work of hybrid PSO and chaos, Liu et al.[17] propose a chaotic particle swarm optimization with adaptive inertia weight factor and chaos to form a chaotic PSO algorithm, which can reasonably combine the population-based evolutionary searching ability of PSO and chaotic searching behavior. Followed by Coelho [18] presents a novel quantum-behaved PSO using chaotic mutation operator, the application of chaotic sequences based on chaotic Zaslavskii map instead of random sequences is a powerful strategy to diversify the population and improve PSO performance in preventing premature convergence. In addition, Alatas et al. [19] present chaos embedded particle swarm optimization algorithms by using different chaotic systems to substitute random numbers for different parameters of PSO. Recently, Tian et al. [20] exploit chaotic maps to improve the initial population of PSO algorithm. Yang et al.[21] put forward an accelerated chaotic particle swarm optimization for data clustering by randomly generating initial particles and substituting the random parameters r_1 and r_2 of PSO with the sequences generated by the logistic map. More recently, a chaotic particle swarm optimization, which combines chaotic optimization algorithm with PSO and T-S fuzzy modeling, is proposed by Jiang et al. [22] to perform constrained predictive control. In [23], Liu et al. employ chaotic opposition-based population initialization instead of a pure random initialization as well as a stochastic search technique to improve the performance of PSO algorithm.

As briefly reviewed above, most of these approaches can achieve encouraging performance and motivate us to explore better PSO methods with the help of their excellent experiences and knowledge. Hence, in this paper we present a novel particle swarm optimization frame- work by introducing chaotic maps and Gaussian mutation strategy. On the one hand, the chaotic map (tent map and logistic map) is utilized to initialize uniform distributed particles so as to improve the quality of the initial population, which is a simple yet very efficient method to improve the quality of initial population. On the other hand, the Gaussian mutation mechanism based on the maximal focus distance is adopted to help the algorithm jump out of the local optima and make the particles proceed with searching until the global optimal or the closer-to-optimal solutions can be found. Experimental results on two benchmark functions demonstrate the effectiveness and efficiency of the PSO proposed in this paper. The rest of this paper is organized as follows. Section 2 outlines the standard PSO briefly. In section 3, two sets of chaotic maps, i.e., tent map and logistic map, are first introduced, and then details how to generate uniform distributed initial particles by the chaotic maps together with their initial performance comparison respectively. Section 4 elaborates the proposed particle swarm optimization based on chaotic maps and Gaussian mutation. Experimental results on two well -known benchmark functions are reported and analyzed in section 5. Finally, we end this paper with some important conclusions and future work in section 6.

2. Standard PSO

PSO is inspired by natural concepts such as bird flocking and fish schooling. In PSO system, each candidate solution is called a particle, each particle moves in the search space with a velocity that is dynamically adjusted according to the corresponding particle's experience and the particle's companions' experience. Mathematically, the particles are manipulated according to the following equations:

$$v_{id}(t+1) = \omega \times v_{id}(t) + c_1 \times r_1 \times [p_{id}(t) - x_{id}(t)] + c_2 \times r_2 \times [p_{gd}(t) - x_{id}(t)] \tag{1}$$

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

$$\omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{iter_{\max}} \times iter_{cur} \tag{3}$$

where c_1 and c_2 are positive constants, called acceleration coefficients. r_1 and r_2 are two random functions in the range [0,1]. ω is the inertia weight defined by Eq.(3), where ω_{max} is the initial weight, ω_{min} is the final weight, $iter_{max}$ denotes the maximum iteration number and $iter_{cur}$ is the current iteration number. It has characteristics that are reminiscent of the temperature parameter in the simulated annealing (SA).A large inertia weight facilitates a global exp- loration while a small inertia weight facilitates a local exploitation. The *i*-th particle is represented as $X_i=(x_{i1},x_{i2},\ldots,x_{iD})$. The best previous position (the position giving the best fitness value) of the *i*-th particle is recorded *pbest* and represented as $P_i=(p_{i1},p_{i2},\ldots,p_{iD})$. The index of the best particle among all the particles in the population is denoted as *gbest* and represented by $P_g=(p_{g1},p_{g2},\ldots,p_{gD})$. The rate of the position change (velocity) for particle *i* is represented as $V_i=(v_{i1},v_{i2},\ldots,v_{iD})$, and the value of each dimension of every velocity vector v_i is clamped to the range $[-v_{max},v_{max}]$ to reduce the likelihood of the particle leaving the search space. *D* represents the dimension of the search space.

3. Chaotic Maps

Chaos can be described as a bounded nonlinear system with deterministic dynamic behavior that has ergodic and stochastic properties. Mathematically, chaos is random and unpredictable, yet it also possesses an element of regularity. In this paper, two kinds of chaos, tent and logistic maps are utilized to generate uniform distributed initial particles to enhance the quality of the initial population as well as to reinitialize the same number of particles when the population inclines to stagnation. Details of them will be described in the following subsections.

3.1. Tent Map

Tent map [24] is studied in the mathematics of dynamical systems because it has several interesting properties such as chaotic orbits, simple shape and so on. More importantly, tent map shows the outstanding advantages and has higher iterative speed than logistic map, because the probability density function (pdf.) of the chaotic sequence for tent map is a uniform function whereas the pdf. of chaotic sequence for logistic map is a Chebyshev-type function [25]. The expression of tent map is defined by Eq.(4), where x denotes the chaotic variable. Its variant through Bernoulli shift transformation is shown by Eq. (5).

$$g(x) = \begin{cases} 2x, 0 \le x \le 1/2 \\ 2(1-x), 1/2 < x \le 1 \end{cases} \tag{4}$$

$$g^{'}(x) = \begin{cases} 2x, 0 \le x \le 1/2 \\ 2x-1, 1/2 < x \le 1 \end{cases} \tag{5}$$

Note that Eq. (5) can be compressed into one representation, that is, $x_{n+1}=g(x_n)=(2x_n)$ *mod* 1. It has been proved that tent map and logistic map are topologically conjugating [26] and the iterative speed of tent map is faster than that of logistic map. The bifurcation diagram of it is illustrated in Figure1 (a). However, tent map also shows some limitations. The reason lies that due to the computer word length is limited, the binary digits of the fractional part of floating-point numbers will tend to be all-zero after a certain number of unsigned left shifting, viz., and plunge into some fixed points. Such as the 4-period (0.2, 0.4, 0.8, 0.6) as well as some unstable periodic points 0.25, 0.5 and 0.75, which make it get stuck at the fixed point 0 after some steps of iteration. As previously mentioned, we illustrate the pseudo-code of tent map used to generate initial particles as follows, which can rapidly generate uniform distributed data sequence and effectively avoid plunging into the small periodic cycles.

Algorithm 1: Pseudo-code of tent map for initialization

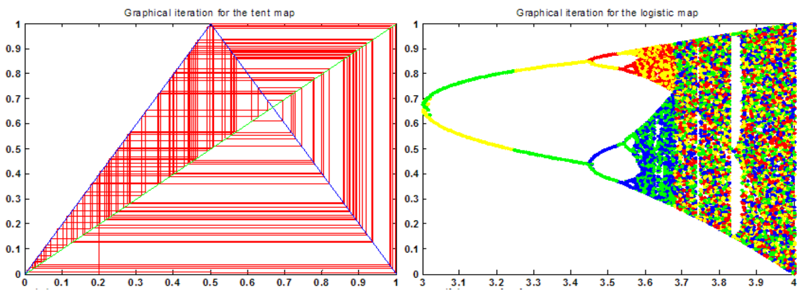
```
Begin
  Randomly initialize chaotic variables
  while(number of maximal iterations is not met)
    if chaotic variable plunges into the fixed points or the small periodic cycles
      Implement a very small positive random perturbation
      Remap them by Eq.(5)
    else
      Update the variables by Eq.(5) directly
    end
      next generation until stopping criterion
      Map the chaotic variables into the optimization problem space
  End
```

3.2. Logistic Map

As one of the simplest chaotic maps, logistic map [27] has been paid much attention to by researchers over the last few decades. It can be described as follows:

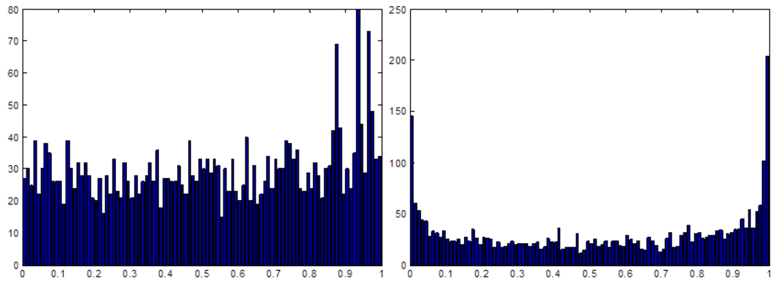
$$x_{n+1} = f(\mu, x_n) = \mu x_n (1 - x_n), n = 0, 1, 2, \cdots \qquad (6)$$

where x_n represents the *n-th* chaotic number. $x_n \in (0, 1)$ under the conditions that the initial $x_0 \in (0, 1)$ except for some periodic fixed points (0, 0.25, 0.5, 0.75, 1), μ usually is a constant predetermined. When μ increases from zero, the dynamic system generated by Eq. (6) will change from one fixed point to two, and until 2^n. It should be noted that μ has a limit value μ_t=3.569945672. In general, the range [μ_t, 4] is considered as the chaotic region of the whole system. Its bifurcation diagram is illustrated in Figure1 (b) the basic idea of chaotic initialization is that; to begin with, generate the same number of chaotic variables corresponding to those of the optimization problem. Then run a preset number of chaotic iterations. Followed by, remap these chaotic variables into the original problem optimization space as the initial variables. Here, equation (6) is chosen as the chaotic signal generator, where μ is called the bifurcation parameter and is set to 4. Since the procedure of using logistic map to generate uniform distributed variables is very similar to that of tent map, so we do not reiterate it any more here.

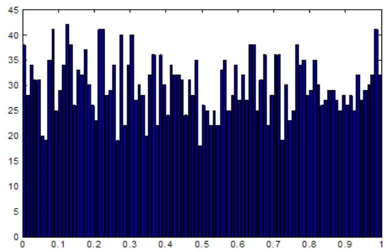


(a) Tent map **(b) Logistic map**

Figure 1. Bifurcation Diagrams of Tent Map and Logistic Map



(a) Tent map (Max.80-Min.15) **(b) Logistic map (Max.204-Min.11)**



(c) Random map (Max.42-Min.18)

Figure 2. Histograms of 3000 Observations for Tent, Logistic and Random Maps

Figure 2 displays the histogram comparison of tent map, logistic map and random map for 3000 iterations in the range [0, 1], respectively. By comparison, it is easy to see that tent map is better than logistic map, which is fully consistent with the conclusions obtained in the literature [26]. In addition, the histogram trend of logistic map is obviously superior to that of random map.

4. PSO with Chaotic Maps and Gaussian Mutation

The PSO proposed in this paper is based on the framework of standard particle swarm optimization. On the one hand, the chaotic map (tent map and logistic map) is employed to generate uniform distributed particles to improve the quality of the initial population. On the other hand, Gaussian mutation mechanism based on the maximal focus distance is utilized to help the algorithm break away from the local optima when stagnation happens, which makes the PSO proceed with searching in other regions of the solution space. Inspired by [28], to check whether the PSO proposed in this paper plunges into the local optima or not, the maximal focus distance (MaxDist) is introduced as well as a threshold of the MaxDist pre- determined as follows:

$$MaxDist = \max_{i=1\cdots m}\left(\sqrt{\sum_{d=1}^{D}(p_{ld}-x_{id})^2}\right) \tag{7}$$

where m is the number of neighborhood particles, p_{ld} is the previous best position, and x_{id} represents the sub-vector of the d-th dimension of the i-th particle in the search space. The general procedure of PSO algorithm proposed in this paper is given as below.

Algorithm 2: Pseudo-code of the proposed PSO algorithm

1. **Begin**
2. Randomly initialize the velocity of the particles and employ chaotic map to initialize the position of the particles, let f_i denote the fitness of each particle, *t-MFD* denote the preset threshold of *MFD*, f_{ave} presents the average fitness of the whole swarm.
3. **while** (number of maximal iterations is not met)
4. **for** *n*=1 to number of particle
5. Find *pbest*
6. Find *gbest*
7. Calculate *MFD* by Eq.(7)
8. **if** $MFD \leq t\text{-}MFD$
9. Calculate f_i and f_{ave}
10. **case 1**: execute Gaussian mutation for the position of the particles that satisfy with $f_i \leq f_{ave}$
11. **case 2**: otherwise, reinitialize the same number of particles using the chaotic map
12. **end**
13. **next *n***
14. Update the velocity and position of the other particles by Eqs.(1) and (2)
15. **next generation until stopping criterion**
16. **End**

5. Experimental Results and Analysis

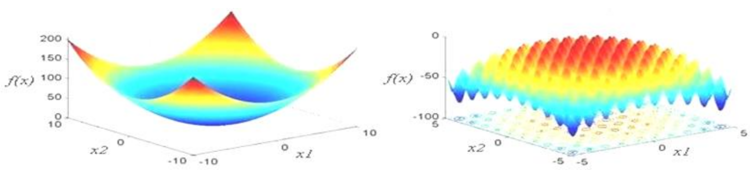
5.1. Experimental Setting

To validate the effectiveness of the PSO algorithm proposed in this paper, two well-known benchmark functions are leveraged to evaluate its performance. The expressions of them are defined as follows:

(1) Sphere function: $f_1(x) = \sum_{i=1}^{d} x_i^2$, with $-100 \leq x_i \leq 100$

(2) Rastrigin function: $f_2(x)=\sum_{i=1}^{d}[x_i^2-10\cos(2\pi x_i)+10]$, with $-10 \le x_i \le 10$

It is worth noting that Sphere function is a unimodal function with only one peak value while Rastrigin function is a multimodal function with a considerable number of local optima in the region of interest. To make a fair comparison, the parameters are set at: the fixed inertia weight is set to *w*=1, while the linearly decreasing inertia weight varies from *wmax*=0.9 at the beginning of the search to *wmin*=0.4 at the end. The acceleration coefficients c_1 and c_2 are set to 2. All the populations consist of 40 individuals. The dimension of the test functions *d* is set to 10. The stopping criterion is set as reaching the maximal iteration of 800. The threshold of MaxDist is predetermined to be 0.28 by trial and error. The distribution situation of the global and local optimal solutions of Sphere and Rastrigin functions with 2-dimensional decision variables is shown in Figure3. For the sake of comparison, different combinations for PSO with an initial population of random map, tent map or logistic map solutions and a constant inertia weight or a linearly decreasing inertia weight are employed, they are abbreviated as Rand- CwPSO, Rand-LDwPSO, Tent-CwPSO, Tent-LDwPSO, Logistic-CwPSO and Logistic- LDwPSO, respectively.



(a) Sphere function **(b) Rastrigin function**

Figure 3. Graphical Shows of the Test Functions with 2-Dimensional Decision Variables

5.2. Experimental Results

For the given functions, thirty independent runs of each of the different PSO algorithms are executed, and each run loops 800 iterations. Tables 1-2 show the best solution (BS) and the standard deviations (SD) of the experimental results for the benchmark functions with dimension *n*=10. Specifically, the optimization variables in each dimension that correspond to the best solution for each PSO algorithm are summarized in these two tables, which can effectively help to investigate the variables distribution in the whole allowable solution space.

Table 1. Optimization Results of Sphere Function

Methods	BS	Variables Corresponding to the Best Solutions					SD
Rand-CwPSO	0.14914	-0.003077	-0.0044609	0.00075782	-0.00026028	-0.0039318	2.3207e-001
		-0.0012973	0.0018788	-0.0034154	0.003874	5.5459e-005	
Rand-LDwPSO	7.7358e-005	0.059487	-0.10917	-0.010107	-0.090094	0.007449	6.1999e-001
		0.30223	0.078552	0.091461	-0.13445	0.038096	
Tent-CwPSO	2.9538e-038	-9.0326e-025	-5.353e-023	6.8835e-020	-2.0286e-021	2.111e-028	1.1772e-002
		-4.5176e-026	7.5317e-024	5.4316e-021	-1.0377e-020	1.4566e-026	
Tent-LDwPSO	1.4582e-168	9.4801e-086	3.8133e-085	-1.9857e-085	-9.855e-086	1.2263e-087	1.6916e-002
		-2.8295e-087	2.717e-086	8.5076e-086	1.1039e-086	-3.4184e-086	
Logistic-CwPSO	2.3428e-032	-4.5454e-023	1.7444e-020	-2.0963e-019	-4.0801e-021	-3.0782e-020	1.5832e-001
		6.1439e-022	-9.6475e-018	4.3978e-021	-3.2897e-019	-3.5345e-022	
Logistic-LDwPSO	4.3971e-168	7.115e-087	1.2673e-087	5.681e-090	-2.8067e-088	5.7709e-087	9.7222 e-001
		-2.2975e-087	2.5523e-086	6.9404e-085	3.627e-090	-2.3973e-087	

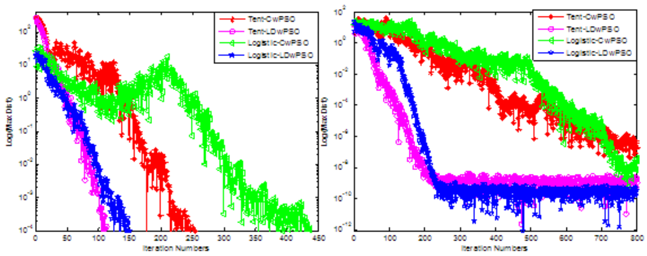
Table 2. Optimization Results of Rastrigin Function

Methods	BS	Variables Corresponding to the Best Solutions					SD
Rand-CwPSO	1.2429e+001	-0.98943	1.0011	0.0017759	-1.0017	-1.9622	3.6324e-001
		-0.99991	0.0011034	-2.0081	-0.034885	0.0040926	
Rand-LDwPSO	1.9088e+001	-0.99615	-1.0084	-0.98446	0.99568	-2.9735	1.1945e-001
		1.9967	-0.011551	-0.98169	-0.0028941	-0.98292	
Tent-CwPSO	0	-3.4843e-013	1.5023e-011	8.2058e-013	-1.4813e-011	-5.2657e-012	2.4754e-002
		-4.6612e-010	-1.7924e-011	7.9807e-012	-6.1206e-013	1.863e-010	
Tent-LDwPSO	0	8.4731e-013	3.3049e-011	5.5892e-011	-1.3799e-010	4.1791e-011	4.3012e-002
		-1.1463e-010	3.4119e-012	-3.8024e-011	9.2722e-011	-1.8092e-011	
Logistic-CwPSO	0	2.4606e-010	6.0267e-011	-1.2684e-009	2.3004e-011	2.0445e-010	6.3006e-002
		4.1455e-011	-8.3035e-012	6.2432e-011	-1.8678e-011	-5.3117e-011	
Logistic-LDwPSO	0	1.6409e-010	-1.0814e-012	-2.592e-012	4.6591e-012	6.3078e-012	2.8791e-001
		-4.3424e-010	-1.3159e-011	-6.1496e-011	1.3633e-012	-1.0278e-011	

From these tables, it is easy to see that the logistic-based PSO proposed in this paper is evidently superior to the rand-based PSO, i.e., the standard PSO algorithm. This implies that the distribution of initial particles can be improved by using logistic map initialization. Mean- while, the premature convergence of population can be effectively prevented by adopting the strategy of Gaussian mutation based on the maximal focus distance among particles. By this way, the performance of PSO can be improved to a certain extent. In addition, it should be noted that the tent-based PSO proposed in this paper, especially the Tent-LDwPSO, outperforms all the others. As can be seen from Tab.1 and Tab.2, the standard deviations of Tent-PSOs are smaller than those of other algorithms, which mean that the PSO with an initial population of tent map solutions is relatively stable. In other words, this further illustrates the importance of the uniform distributed initial particles to the convergence performance of the PSO algorithm and the linearly decreasing inertia weight to be balancing the global and local search.

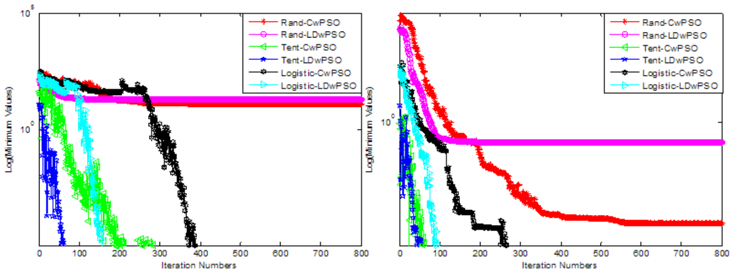
Figure 4 graphs the evolution curves of MaxDist of each algorithm for the two tested functions. To show the evolutionary processes clearly, here, the y-axes of Figure4 denote the fitness logarithm values. Especially in Figure4 (a), the former part is scaled up to a certain degree so as to show the tendency more clearly. In actual fact, each algorithm corresponding to each curve shown in Figure4 (a) is run for 800 iterations. As can be seen from Figure4, the points where the curve of Tent-LDwPSO decreases rapidly implies the particles tend to trap into the local optima. Then Gaussian mutation is employed in time to help PSO escape from the local optima and make the particles proceed with searching in other regions of the solution space until the global optimum is found. In comparison, the curves of logistic-based PSO decrease slowly as search proceeds.

Alternatively, Figure 5 illustrates the evolution curves of the best convergence solutions for the tested functions. Similarly, we can see that Tent-LDwPSO performs much better than the other five PSO methods. At the same time, it consistently keeps fast speed of evolution and finally converges to the global optimum effectively.



(a) Sphere function **(b) Rastrigin function**

Figure 4. The Evolution Curves of the Maxdist for Test Functions



(a) Sphere function **(b) Rastrigin function**

Figure 5. The Evolution Curves of the Best Solutions for Test Functions

6. Conclusion and Expectation

In this paper, we have proposed a new particle swarm optimization algorithm by introducing two sets of chaotic maps (tent and logistic map) and Gaussian mutation mechanism based on the maximal focus distance, which can generate uniform distributed initial particles to improve the low stability of PSO algorithm and help it to proceed with searching in other regions of the solution space effectively. Simulation results on two benchmark functions demonstrate that the PSO proposed in this paper has not only the powerful ability to search the global optimum, but also effectively prevent the premature convergence in time, especial- ly the Tent-PSO. As future work, we plan to introduce this approach into the other real-world research fields, such as integrated circuit design, multimedia semantic under- standing and engineering optimization scheduling, etc. Lastly, and arguably most importantly, the qualitative relationship between the initial particles' distribution and the convergence of PSO algorithm, from the viewpoint of mathematics, will be elaborated and proved comprehensively.

Acknowledgements

The work is supported by the National Natural Science Foundation of China (No.61035003, No.61072085, No.60933004, No.60903141), the National Program on Key Basic Research Project (973 Program) (No.2013CB329502), the National High-tech R&D Program of China (863 Program) (No.2012AA011003), the National Science and Technology Support Program of China (2012BA107B02), the Natural Science Basic Research Plan in Shaanxi Province of China (No.2014JQ2-6036) and the Science and Technology R&D Program of Baoji City (No. 2013R2-2).

References

[1] J. Kennedy and R. Eberhart, “Particle swarm optimization”, In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, **(1995)**, pp. 1942-1948.
[2] M. Dorigo and L. Gambardella, “Ant colony system, a cooperative learning approach to the traveling salesman problem”, IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, **(1997)**, pp. 53-66.
[3] J. Holland, “Adaptation in natural and artificial systems”, University of Michigan Press, **(1975)**.
[4] A. Ratnaweera, S. Halgamuge and H. Watson, “Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients”, IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, **(2004)**, pp. 240-255.
[5] R. He, Y. Wang, Q. Wang, J. Zhou and C. Hu, “An improved particle swarm optimization based on self-adaptive escape velocity”, Journal of Software, vol. 16, no. 12, **(2005)**, pp. 2036-2044.
[6] Z. Lu and Z. Hou, “Particle swarm optimization with adaptive mutation”, Acta Electronica Sinica, vol. 32, no. 3, **(2004)**, pp. 416-420.
[7] C. Li, Y. Liu, A. Zhou, L. Kang and H. Wang, “A fast particle swarm optimization algorithm with Cauchy mutation and natural selection strategy”, In: Proceedings of the 2nd International Sympo- sium, **(2007)**, pp.334-343.
[8] C. Li, S. Yang, I. Korejo, “An adaptive mutation operator for particle swarm optimization”, In: Proceedings of the 2008 UK Workshop on Computational Intelligence, **(2008)**, pp. 165-170.
[9] M. Pant, R. Thangaraj and A. Abraham, “Particle swarm optimization using adaptive mutation”, In: Proceedings of the 19th International Conference on Database and Expert Systems Applications (DEXA’08), **(2008)**, pp.519-523.
[10] Z. Chen and Y. Li, “An improved particle swarm optimization algorithm with opposition mutation”, In: Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA’08), **(2008)**, pp.5344-5347.
[11] X. Li, J. Zhuang, S. Wang and Y. Zhang, “A particle swarm optimization algorithm based on adaptive periodic mutation”, In: Proceedings of the 4th International Conference on Natural Computation (ICNC’08), **(2008)**, pp.150-155.
[12] Y. Gao and Y. Duan, “A new particle swarm optimization algorithm with adaptive mutation operator”, In: Proceedings of the 2nd International Conference on Information and Computing Science (ICICS’09), **(2009)**, pp.58-61.
[13] C. Wang, P. Sui and W. Liu, “Improved particle swarm optimization algorithm based on double mutation”, ICIC Express Letters, vol. 3, no. 4, **(2009)**, pp. 1417-1422.
[14] Y. Wang, Q. Xiang and Z. Zhao, “Particle swarm optimizer with adaptive tabu and mutation: a unified framework for efficient mutation operators”, ACM Transactions on Autonomous and Adaptive Systems, vol. 5, no. 1, **(2010)**, pp. 1-27.
[15] D. Xu, Y. Li, X. Tang, *et al*, “Adaptive particle swarm optimization with mutation”, In: Procee- dings of the 30th Chinese Control Conference (CCC’11), **(2011)**, pp.2044-2049.
[16] H. Schuster and W. Just, “Deterministic Chaos: An Introduction”, 4th Revised and Enlarged Edition, WILEY-VCH Verlag GmbH & Co. KGaA **(2006)**.
[17] B. Liu, L. Wang, Y. Jin, F. Tang and D. Huang, “Improved particle swarm optimization combined with chaos, Chaos, Solitons and Fractals”, vol. 25, no. 5, **(2005)**, pp. 1261-1271.
[18] L. S. Coelho, “A quantum particle swarm optimizer with chaotic mutation operator, Chaos, Solitons and Fractals”, vol. 37, no. 5, **(2008)**, pp. 1409-1418.
[19] B. Alatas, E. Akin and A. Ozer, “Chaos embedded particle swarm optimization algorithms, Chaos, Solitons and Fractals”, vol. 40, no. 4, **(2009)**, pp. 1715-1734.
[20] D. Tian and T. Zhao, “Particle swarm optimization based on tent map and logistic map”, Journal of Shaanxi University of Science and Technology, vol. 28, no. 2, **(2010)**, pp. 17-23.
[21] L. Chuang, C. Hsiao and C. Yang, “Chaotic particle swarm optimization for data clustering”, Expert Systems with Applications, vol. 38, no. 12, **(2011)**, pp. 14555-14563.
[22] H. Jiang, C. Kwong, Z. Chen and Y. Ysim, “Chaos particle swarm optimization and T-S fuzzy modeling approaches to constrained predictive control”, Expert Systems with Applications, vol. 39, no. 1, **(2012)**, pp. 194-201.
[23] W. Gao, S. Liu and L. Huang, “Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique”, Communications in Nonlinear Science and Nume- rical Simulation, vol. 17, no. 11, **(2012)**, pp. 4316-4327.
[24] H. Peitgen, H. Jurgens and D. Saupe, “Chaos and fractals: new frontiers of science”, New York, NY: Springer-Verlag, **(1992)**.
[25] W. Steeb, “The nonlinear workbook: chaos, fractals, cellular automata, genetic algorithms, gene expression programming, support vector machine, wavelets, hidden Markov models, fuzzy logic with C++, Java and symbolic C++ programs”, World Scientific Publisher, **(2005)**.
[26] L. Shan, H. Qiang, J. Li and Z. Wang, “Chaotic optimization algorithm based on tent map”, Control and Decision, vol. 20, no. 2, **(2005)**, pp. 179-182.
[27] R. May, “Simple mathematical models with very complicated dynamics”, Nature, vol. 261, **(1976)**, pp. 459-467.

[28] N. Li, F. Liu and D. Sun, “A study on the particle swarm optimization with mutation operator constrained layout optimization”, Chinese Journal of Computers, vol. 27, no. 7, **(2004)**, pp. 897-903.

