Clonal Particle Swarm Optimization and Its Applications

Y. Tan, *Senior Member, IEEE*, Z. M. Xiao

***Abstract*— Particle swarm optimization (PSO) is a stochastic global optimization algorithm inspired by social behavior of bird flocking in search for food, which is a simple but powerful, and widely used as a problem-solving technique to a variety of complex problems in science and engineering. A novel particle swarm optimization algorithm based on immunity-clonal strategies, called as clonal particle swarm optimization (CPSO), is proposed at first in this paper. By cloning the best individual of ten succeeding generations, CPSO has better optimization solving capability and faster convergence performance than the conventional standard particle swarm optimization (SPSO) based on a number of simulations. A detailed description and explanation of the CPSO algorithm are given in the paper. Several experiments on six benchmark test functions are conducted to demonstrate that the proposed CPSO algorithm is able to speedup the evolution process and improve the performance of global optimizer greatly, while avoiding the premature convergence on the multidimensional variable space.**

I. INTRODUCTION

The particle swarm optimization (PSO) developed by Eberhart and Kennedy in 1995 is a stochastic global optimization technique inspired by social behavior of bird flocking or fish schooling [1]. It simulates the behaviors of bird flocking involving the scenario of a group of birds randomly looking for food in an area. All the birds don't know where the food is located, but they just know how far from the food location. So an effective strategy for the birds to find food is to follow the bird which is nearest to the food. PSO is motivated from this scenario and developed to solve complex optimization problems.

In the original form of PSO, each particle in a swarm population adjusts its position in the search space based on the best position it has found so far, and the position of the known best-fit particle in the entire population. The essence of PSO is to use these particles with best known positions to guide the swarm population to converge to a single optimum in the search space. Unlike other population-based evolutionary algorithms, i.e., genetic algorithms, PSO does not need genetic operators such as crossover and mutation. Thus it has advantages of easy implementation, fewer parameters to be adjusted, strong capability to escape from local optima as well as rapid convergence. In addition, because the PSO comprises a very simple concept and paradigms can be implemented more easily with it, it has been demonstrated in certain instances that PSO outperforms other population based evolutionary computing algorithms in many practical engineering domains.

Y. Tan and Z.M. Xiao are with State Key Laboratory of Machine Perception, Peking University, and with Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, P.R. China (phone: 86-10-6276-7611; fax: 86-10-6276-7611; email: ytan@pku.edu.cn).

In recent years, PSO has been used increasingly as an effective technique for solving complex and difficult optimization problems. PSO has been successfully applied to function optimization, artificial neural network training, fuzzy system control, blind source separation as well as machine learning, just to name a few. Furthermore, the PSO has also been found to be robust and fast in solving nonlinear, non-differentiable and multi-modal problems [5]. So, it is very important and necessary to exploit some new mechanisms and principles from other domains or fields to improve and promote the performance of the conventional standard PSO. In this paper, the clonal mechanism found in the natural immune system of creatures are introduced into the PSO, resulting in building a novel clonal PSO (CPSO, for short).

The remainder of this paper is organized as follows. Section 2 describes the conventional standard PSO algorithm and its related modified versions. Section 3 introduces the proposed CPSO by introducing clonal mechanism in NIS into standard PSO and its implementation. In Section 4, extensive experimental results are presented to illustrate the effectiveness and efficiency of the proposed CPSO in comparing to SPSO. Finally, concluding remarks are drawn in Section 5.

II. RELATED WORKS

A. Conventional Standard PSO

In conventional PSO algorithm, each single solution to an optimization problem is considered as a particle in the search space. The exploration of a problem space was done in PSO by a population of particles called a swarm. All particles in the swarm have fitness values which are evaluated by the fitness function related to the optimization problem to be solved. So, the PSO algorithm is originally initialized with a swarm of particles placed on the search space randomly and is used to search for optimal solution by evolving generation by generation. In each iteration, the position and the velocity of each particle are updated according to its own previous best position ($P_{iBd}(t)$) and the best position of all particles ($P_{gBd}(t)$) in the swarm so far. The updating formula for each particle's velocity and position in conventional standard PSO is written as

$$V_{id}(t+1) = wV_{id}(t) + c_1r_1(P_{iBd}(t) - X_{id}(t)) + c_2r_2(P_{gBd}(t) - X_{id}(t)) \tag{1}$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \tag{2}$$

where $i = 1, 2, \cdots, n$, n is the number of particles in the swarm, $d = 1, 2, \cdots, D$, and D is the dimension of solution space.

1-4244-1340-0/07$25.00 ©2007 IEEE

In Eqs. (1) and (2), the learning factors c_1 and c_2 are nonnegative constants, r_1 and r_2 are random numbers uniformly distributed in the interval [0,1], $V_{id} \in [-V_{max}, V_{max}]$, where V_{max} is a designated maximum velocity which is a constant preset by users according to the objective optimization function. If the velocity on one dimension exceeds the maximum, it will be set to V_{max}. This parameter controls the convergence rate of the PSO and can prevent the method from growing too fast. The parameter w is the inertia weight used to balance the global and local search abilities, which is a constant in the interval [0, 1]. Since a large inertia weight is more appropriate for global search, and a small inertia weight facilitates local search. A linearly decreasing inertia weight over the course of search was proposed and analyzed in details by Shi and Eberhart [3] and gave a good performance.

The termination criterion for iterations is determined according to whether the presetting maximum generation or a designated value of the fitness is reached.

For the purpose of convenience, we call the PSO expressed in Eqs. (1) and (2) as standard PSO (abbreviated as SPSO) in the remainder of this paper.

B. *Variants of the PSO*

Since its invent, PSO has attracted an extensive attentions and interests of researchers from different scientific domains. Many researchers have worked on improving its performance in various ways, thereby deriving many interesting variants of PSO.

One of the variants introduces a parameter called inertia weight into the original PSO algorithms [3]. A clever technique for creating a discrete binary version of the PSO introduced by Kennedy and Eberhart [2] in 1997 uses the concept of velocity as a probability that a bit takes on one or zero. By analyzing the convergence behavior of the PSO, a variant of the PSO with a constriction factor was introduced by Clerc and Kennedy [4], which guarantees the convergence and at the same time improves the convergence speed sharply. Parsopoulos and Vrahatis proposed a unified particle swarm optimizer (UPSO) which combined both the global version and local version together [8]. A cooperative particle swarm optimizer was also proposed in [9]. Furthermore, El-Abd and Kamel proposed a Hierarchal Cooperative Particle Swarm Optimizer [11]. In [10], Peram et al. proposed the fitness-distance-ratio based particle swarm optimization (FDR-PSO),by defining the ”neighborhood” of a particle as the n closest particles of all particles in the population. Very recently, a comprehensive learning particle swarm optimizer (CLPSO) was proposed to improve the performance of the original PSO on multi-modal problems greatly by a novel learning strategy [12]. Although there are numerous variants of the PSO, they need much time to finish evaluations of fitness function, and give similar results in the early parts of convergence. Hence, we here choose a variant of PSO with the inertia weight as a foundation of our standard PSO and use it as our basic and standard algorithm for comparisons.

III. Clonal Particle Swarm Optimization

A. *Clonal Expansion Process in Natural Immune System*

Artificial immune systems (AIS) is a novel computational intelligence paradigm inspired by the natural immune system (NIS). Like artificial neural networks and genetic algorithm, AIS are highly abstract models of their biological counterparts applied to solve a number of complex problems in different domains. Some work processes of NIS are used as metaphors to develop novel computing models in computational intelligence, such as negative selection, clonal selection, to name a few, to solve many complex problems in science and engineering [6], [7], [13].

Originally, according the theory of clonal selection, when the B and T lymphocytes in NIS recognize an antigen as nonself, NIS will start to proliferate by cloning upon recognition of such antigen. When a B cell is activated by binding an antigen, many clones are produced in response, via a process called clonal expansion. The resulting cells can undergo somatic hypermutation, creating offspring B cells with mutated receptors. The higher the affinity of a B cell to the available antigens, the more likely it will clone. This is called as a Darwinian process of variation and selection, i.e., affinity maturation [6], [7].

The essence of the conventional PSO is to use these particles with best known positions to guide the swarm or the population to converge to a single optimum in the search space. However, how to choose the best-fit particle to guide each particle in the swarm is a critical issue. This becomes even more acute when the problem to be solved has multiple optima since the entire swarm or population could potentially be misled to local optima. In order to deal with this case, a clonal expansion in NIS is probably a good way to guide or direct the SPSO escaping from local optima whilst searching for the global optima efficiently. Therefore, here we want to introduce the clonal expansion process in NIS into the SPSO to strength the interaction between particles in a swarm and improve the convergent performances of the SPSO greatly.

B. *Clonal Particle Swarm Optimization (CPSO) Algorithm*

According to the clonal expansion process in natural immune system discussed above, we propose a clonal operator for the SPSO. The clonal operator is to clone one particle as N same particles in the solution space according to its fitness function at first, then generate N new particles via clonal mutation and selection processes which are related to the concentration mechanisms used for antigens and antibodies in NIS. Here we call the SPSO with such clonal operator as clonal particle swarm optimization (abbreviated as CPSO) algorithm. For simplification of our presentation, we will use the abbreviated CPSO algorithm directly later on.

As indicated in [12], CLPSO’s learning strategy abandons the global best information, the past best information of other particles is used to update the particles’ velocity instead. In such a way, the CLPSO can significantly improve the performance of the original PSO on multi-modal problems.

Here in order to present our CPSO clearly and efficiently, we adopt the similar definitions used in AIS paradigms. Antigen, antibody, and the affinity between antigen and antibody are corresponding to objective optimization function, solution candidate, and the fitness value of the solution on the objective optimization function, respectively. The clonal operator is used to copy one point as N same points according to its fitness function, and then generate N new particles by undergoing mutation and selection processes. In general, the state transition process of a swarm of particles in the CPSO can be schematically expressed as follows.

$$P(t) \longrightarrow^{clone} C(t) \longrightarrow^{mutation} M(t) \longrightarrow^{sel} P(T+1) \quad (3)$$

Where the arrow represents the transition process between two states while symbols over the arrows show the operations needed for the transition processes.

Note that the population of particles $P(t)$ at time t can be transited as $C(t)$ via clone process, then next generation population $P(t+1)$ can be generated by using mutation and selection processes for the cloned population $C(t)$.

Briefly, the CPSO algorithm can be summarized as follows.

Step 1: initialization. Assume $a = 1$, $c_1 = 2$, $c_2 = 2$, and w be from 0.9 to 0.4 linearly.

Step 2: the state evolution of particles is iteratively updated according to Eqs. (1) and (2).

Step 3: memory the global best-fit particle of each generation, P_{gB}, as a mother particle of the clonal operator in Step 4.

Step 4: after M generations, clone the memorized M global best particles, $P_{gB}^{(i)}, i = 1, \cdots, M$.

Step 5: Mutation Process: all of the cloned particles are mutated to some extents to differentiate with original or mother particle. Here, the mutation process is implemented by using some random disturbances such as Gaussian noise. Assume P_{gB_k} be the k-th entry of the vector P_{gB} and μ is an Gaussian random variable with zero mean and unity variance, then one can have the following random mutation process,

$$P_{gB_k} = (1-\mu)P_{gB_k} \quad (4)$$

Step 6: Selection Process: we store the current P_{gB} in memory, but the other particles are selected according to a strategy of the diversity keeping of the concentration mechanism so that in next generation of particles, a certain concentration of particles will be maintained for each fitness layer. Here the concentration of i-th particle are defined as follows.

$$D(x_i) = (\sum_{j=1}^{N+M} |f(x_i) - f(x_j)|)^{-1}, i = 1, 2, \cdots, N+M. \quad (5)$$

where x_i and $f(x_i)$ in Eq.(5) denote the i-th particle and its fitness value, respectively.

According to above Eq.(5), one can derive a selection probability in terms of the concentration of particles as

$$p(x_i) = \frac{\frac{1}{D(x_i)}}{\sum_{j=1}^{N+M} \frac{1}{D(x_j)}}, i = 1, 2, \cdots, N+M, \quad (6)$$

It can be seen from Eqs.(5) and (6) that the more the particles are similar to the antibody i, the less the probability the particle i can be chosen, and vice verse. In such a way, the particle with low fitness value also has an opportunity to evolve. Therefore, this kind of probability selection mechanism in terms of the concentration of particles is able to guarantee the diversity of the antibodies theoretically and endows the method with the ability of escaping from local minima.

Step 7: Termination. The algorithm can be terminated by some common stop criteria such as a given maximum number of generations or a presetting accuracy of the solution. In our experiments in the paper, we adopt the former stop criterion, i.e. a maximum number of generations.

Through keeping current global optima, the proposed improved algorithm can guarantee to maintain the good performance of original standard PSO. In the meantime, the essence of the clonal operator is to generate a new particle swarm near the promising candidate solution according to the value of the fitness function such that the search space are enlarged greatly and the diversity of clones is increased to avoid trapping in local minima. On the other hand, the speed of convergence and performance could be raised rapidly.

IV. Experiments

A. Benchmark Functions for Simulation

In order to test and verify the performance of our proposed CPSO, and make a comparison with SPSO, eleven benchmark functions listed in Table I are used for our following simulations.

TABLE I
LIST OF ELEVEN BENCHMARK TEST FUNCTIONS AND THEIR PARAMETERS FOR OUR FOLLOWING SIMULATIONS

Functions	Expression	Dim D	Search Space	V_{max}
Shaffer f6	F_1	$10, 30$	$(-100, 100)^D$	100
Sphere	F_2	10	$(-100, 100)^D$	100
Rosenbrock	F_3	10	$(-100, 100)^D$	100
Griewank	F_4	10	$(-600, 600)^D$	600
Rastrigrin	F_5	10	$(-10, 10)^D$	10
Ackley 2	F_6	10	$(-100, 100)^D$	100
Ellipse	F_7	30	$(-100, 100)^D$	100
Cigar	F_8	30	$(-100, 100)^D$	100
Tablet	F_9	30	$(-100, 100)^D$	100
Sumcan	F_{10}	30	$(-100, 100)^D$	100
Schwefel	F_{11}	30	$(-32, 32)^D$	32

Due to limited space for the above table, we list the formula of some benchmark functions as follows.

$$F_1 = \frac{sin^2(\sqrt{x_1^2 - x_2^2} - 0.5)}{(1.0 + 0.001(x_1^2 - x_2^2))^2 + x_2^2} + 0.5, \quad (7)$$

$$F_2 = \sum_{i=1}^{D} x_i^2, \tag{8}$$

$$F_3 = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i)^2 + (x_i - 1)^2), \tag{9}$$

$$F_4 = 1 + \sum_{i=1}^{D} \frac{x_i^2}{4000} + \Pi_{i=1}^{D} cos(x_i/\sqrt{i}), \tag{10}$$

$$F_5 = \sum_{i=1}^{D} (x_i^2 - 10cos(2\pi x_i) + 10), \tag{11}$$

$$F_6 = 20 + e - 20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \tag{12}$$

$$-exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi x_i^2)), \tag{13}$$

$$F_7 = \sum_{i=1}^{D} 10^{4\frac{i-1}{D-1}} x_i^2, \tag{14}$$

$$F_8 = x_1^2 + \sum_{i=2}^{D} 10^4 x_i^2, \tag{15}$$

$$F_9 = 10^4 x_1^2 + \sum_{i=2}^{D} x_i^2, \tag{16}$$

$$F_{10} = \frac{1}{10^{-5} + \sum_{i=1}^{D} |\sum_{j=1}^{i} x_j|}, \tag{17}$$

$$F_{11} = \sum_{i=1}^{D} ((x_1 - x_i^2)^2 + (x_i - 1)^2), \tag{18}$$

For more complex and compound benchmark test functions, interested readers refer to [12], [14], etc.

B. Generations of Clones versus Performance

For the number of generations of clones, denoted by symbol iter_no, being 1, 2, 4, and 10, respectively, the performances of the CPSO on Sphere function are illustrated in Figure 1.

It can be seen from Figure 1 that 'iter_no' has little effect on the performance of the proposed CPSO on Sphere function. So in the following experiments, we let 'iter_no' be 10 for simplification and convenience.

In addition, for the six test functions chosen in our experiments, we fixed the number of particles in a swarm to be 40 for convenient comparisons later on.

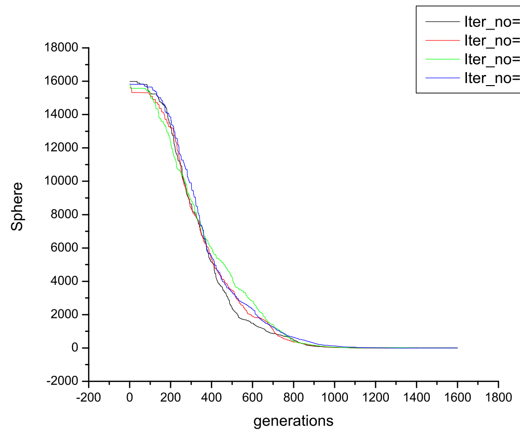


Fig. 1. Convergent Performances of the CPSO on Sphere Function with Different Generations of Clones.

C. Performance Comparison Between CPSO and SPSO

The performance comparisons between the proposed CPSO algorithm and original SPSO algorithm are shown in Figures 2- 4.

The convergent curves of the performances on eleven typical benchmark test functions shown in Figures 2- 7 are drawn from the averaged values of independent 20 runs. In such a way, these curves can give the stable performances of the CPSO and SPSO algorithms completely and reliably. Beside the global optimum of Shaffer f6 function is 1, the global optimum of the other five benchmark test functions are 0. As we can seen from these figures that our proposed CPSO algorithm has much more speed of convergence and more accurate solution than that of the SPSO algorithm on all eleven benchmark test functions.

Furthermore, in order to verify the validation and efficiency of our CPSO, we have conducted a little more test experiments again. We use eleven benchmark test functions to test our CPSO and SPSO by using a swarm of 40. In Table II, we give the statistical means and standard deviations of our obtained solutions of the eleven benchmark test functions, some of which are listed in Table I, by using the CPSO and SPSO, over 50 independent runs, respectively. It has been seen from the averaged solutions in Table II that our proposed CPSO outperforms SPSO dramatically.

It turns out from the comparisons of performances between CPSO and SPSO that the CPSO not only has a faster convergence speed than the SPSO, but also has more accurate optimal solution than the SPSO on all of eleven benchmark functions used in the experiments. Therefore, It is concluded that the proposed CPSO speeds up the convergence tremendously, while keeping a good search capability of global solution with much more accuracy. All of the simulation results in our experiments show that the introduction of clonal mechanism in natural immune system

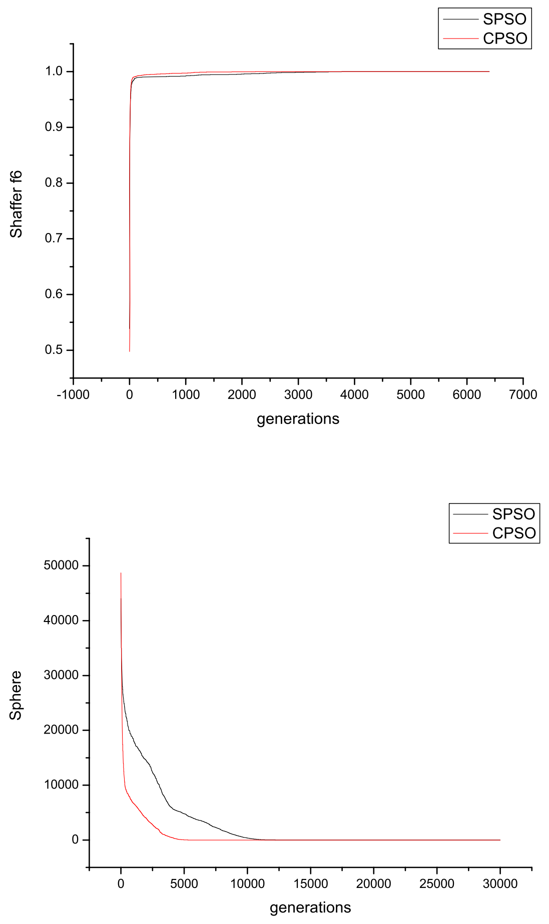


Fig. 2. Performances of the CPSO and SPSO on Shaffer F6 and Sphere Benchmark Functions on 20 Independent Runs with 40 Particles in a Swarm.

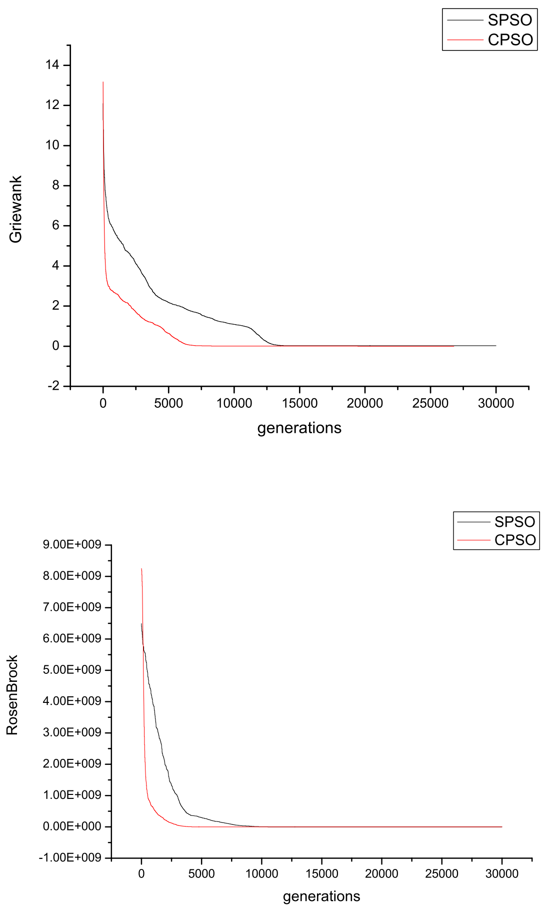


Fig. 3. The Averaged Performances of the CPSO and SPSO on Griewank and Rosenbrock Benchmark Functions on 20 Independent Runs with 40 Particles in a Swarm.

to the PSO achieves a complete success and give out a promising performance in all of our conducted experiments.

V. Conclusions

A clonal particle swarm optimization (CPSO) is proposed and implemented in this paper according to immunity-clonal strategies. By cloning the best individual of every ten succeeding generations, the CPSO has better optimization solving capability and convergence performance than the conventional SPSO. Compared to the SPSO algorithm, the experimental results on eleven benchmark test functions have demonstrated that the proposed CPSO algorithm is able to speedup the evolution process and improve the performance of global optimizer greatly, while avoiding the premature convergence on the multidimensional variable space. Its advantages will make the CPSO find more and more applications in a variety of practical scientific and engineering domains and fields in near future.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under grant no. 60673020. This work was also supported in part by the Research Fund for the Doctoral Program of Higher Education (RFDP).

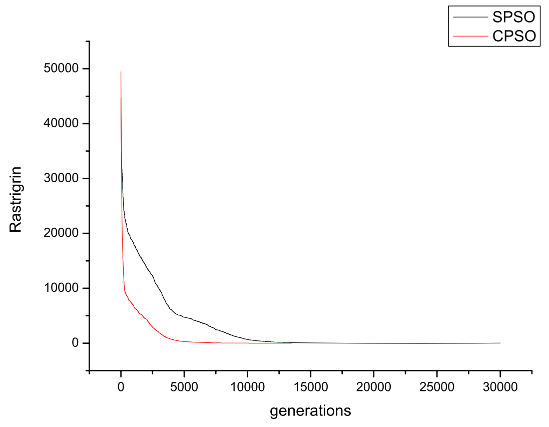
References

[1] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *Proc. Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, IEEE Service Center, Vol. 4. Piscataway, NJ, 1995, pp. 1942C1948.

[2] J. Kennedy, "The Particle Swarm: Social Adaption of Knowledge," *Proc. Proceedings of the IEEE International Conference on Evolutionary Computation*, April 1997.

[3] Y. H. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," *Proc. IEEE World Congress on Computational Intelligence*, Alaska, ALTEC, vol. 1, 1998, pp. 69-73.

[4] M. Clerc and J. Kennedy, "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 1, 2002, pp. 58-73.



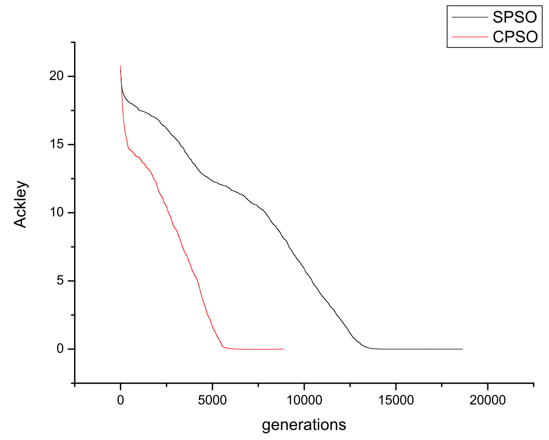
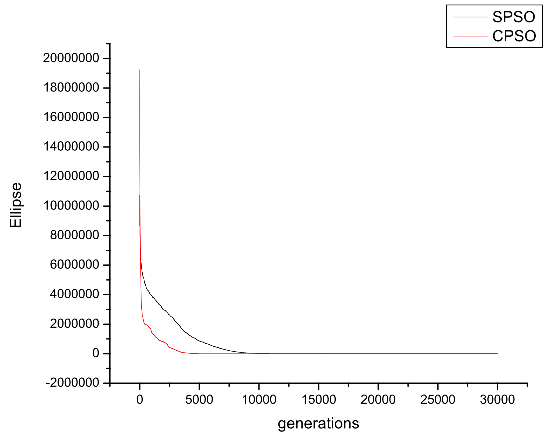


Fig. 4. Performances of the CPSO and SPSO on Rastrigin and Ackley2 Benchmark Functions via 20 Independent Runs with 40 Particles in a Swarm.



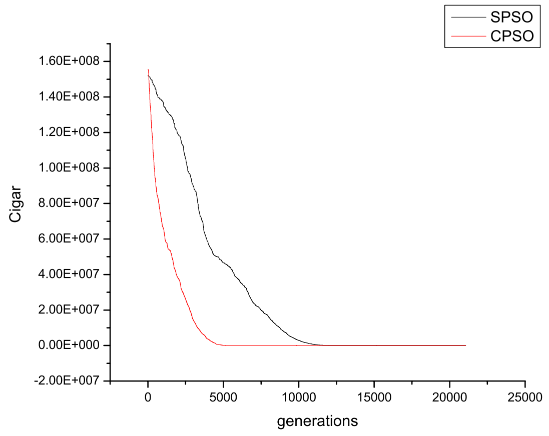


Fig. 5. Performances of the CPSO and SPSO on Ellipse and Cigar Benchmark Functions via 50 Independent Runs with 40 Particles in a Swarm.

[5] H. W. Ge, Y. C. Liang, Y. Zhou, X. C. Guo, "A Particle Swarm Optimization-based Algorithm for Job-shop Scheduling Problem," *International Journal of Computational Methods*, vol. 2, no. 3, 2005, pp. 419-430.

[6] D. Dasgupta and N. Attoh-Okine, "Immunity-based systems: a survey," *Proc. Proceedings of IEEE International Conference on Systems, Man, and Cybernetics,* Orlando, Florida, Oct. 1997.

[7] L. N. de Castro and J. I. Timmis, "Artificial Immune System as a Novel Soft Computing Paradigm," *soft computing journal,* vol. 7, no.8, 2003, pp. 526-544.

[8] K. E. Parsopoulos and M. N. Vrahatis, "UPSO-A united particle swarm optimization scheme," *Lecture Series on Computational Sciences,* 2004, pp. 868-873.

[9] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. on Evolutionary Computation*, vol. 8, 2004, pp. 225-239.

[10] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," *Proc. Proceedings of Swarm Intelligence Symposium,* 2003, pp. 174-181.

[11] M. El-Abd and M. S. Kamel, "A Hierarchal Cooperative Particle Swarm Optimizer," *Proc. Proceedings of Swarm Intelligence Symposium,* 2006, pp. 43-47.

[12] J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. on Evolutionary Computation,* vol. 10, 2006, pp. 281-296.

[13] L. N. de Castro, "Learning and Optimization Using the Clonal Selection Principle" *IEEE Transactions on Evolutionary Computation,* Vol.6, 2002, pp. 239-251.

[14] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," *Proc. Proceedings of IEEE Congress on Evolutionary Computation,* 2002, pp. 1671-1676.

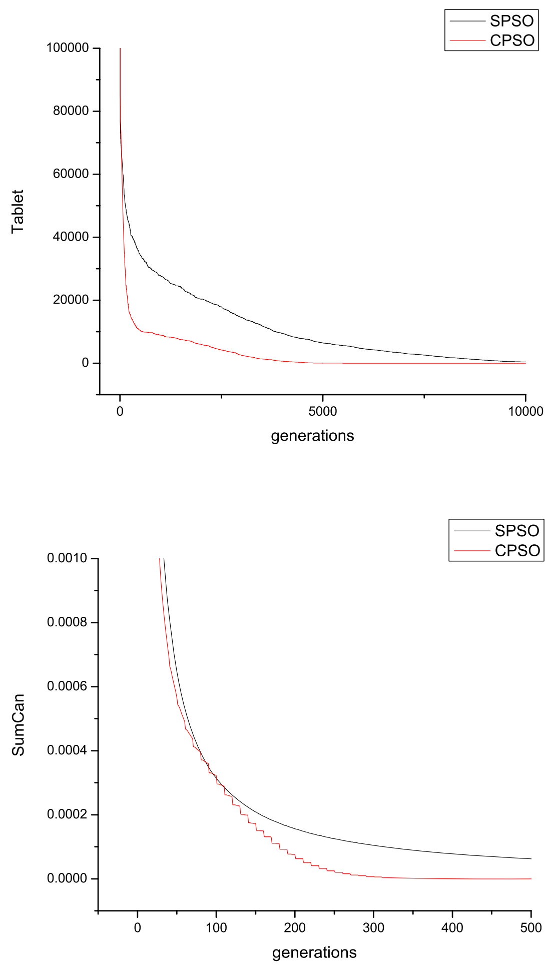


Fig. 6. Performances of the CPSO and SPSO on Tablet and Sumcan Benchmark Functions via 50 Independent Runs with 40 Particles in a Swarm.

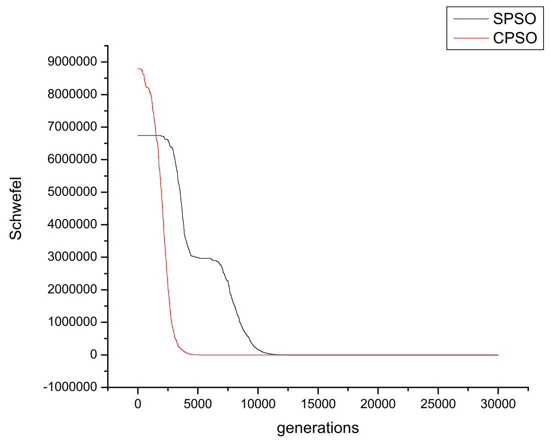


Fig. 7. Performances of the CPSO and SPSO on Schwefel Benchmark Test Function via 50 Independent Runs with 40 Particles in a Swarm.

TABLE II

STATISTICAL MEANS AND STANDARD DEVIATIONS OF THE SOLUTIONS OF ELEVEN BENCHMARK TEST FUNCTIONS, LISTED IN TABLE I, GIVEN BY THE CPSO AND THE SPSO OVER 50 INDEPENDENT RUNS.

Functions	**Gens**	**CPSO's Mean $\pm$ StD**	**SPSO's Mean $\pm$ StD**
Shaffer f6	1000	0.997432 $\pm$ 0.004230	0.992357 $\pm$ 0.003018
Sphere	5000	29.859739 $\pm$ 53.196925	4804.200195 $\pm$ 1254.233280
Rosenbrock	10000	66.587219 $\pm$ 204.290749	5692076.000000 $\pm$ 4087432.037220
Griewank	10000	0.003693 $\pm$ 0.011792	1.088648 $\pm$ 0.042218
Rastrigrin	10000	6.769299 $\pm$ 7.701368	676.154907 $\pm$ 197.969455
Ackley 2	6000	0.029793 $\pm$ 0.067038	11.717416 $\pm$ 1.226893
Ellipse	10000	0.000000 $\pm$ 0.000000	12670.500977 $\pm$ 6473.115751
Cigar	10000	0.000000 $\pm$ 0.000000	2875340.750000 $\pm$ 1245908.471350
Tablet	10000	0.000000 $\pm$ 0.000000	381.082153 $\pm$ 211.210144
SumCan	400	0.000000 $\pm$ 0.000001	0.000078 $\pm$ 0.000002
Schwefel	10000	1.701862 $\pm$ 1.049991	164871.140625 $\pm$ 106403.972545