IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 1, No. 1, March 2012, pp. 1~10
ISSN: 2252-8938

Dynamic Particle Swarm Optimization for Multimodal Function

H. Omranpour*, M. Ebadzadeh*, S. Shiry*, S. Barzegar**
* Departement of Computer Engineering, Amirkabir University of Technology
** Departement of Computer Engineering, Islamic Azad University

Article Info

Article history:

Received Jan 4th, 2012
Revised Aug Feb 10th, 2012
Accepted Mar 16th, 2012

Keyword:

optimization
swarm intelligence
particle swarm optimization
localextremum
search space
the initial distribution

ABSTRACT

In this paper, a technical approach to particle swarm optimization method is presented. The main idea of the paper is based on local extremum escape. A new definition has been called the worst position. With this definition, convergence and trapping in extremumlocal be prevented and more space will be searched. In many cases of optimization problems, we do not know the range that answer is that.In the results of examine on the benchmark functions have been observed that when initialization is not in the range of the answer, the other known methods are trapped in local extremum. cThe method presented is capable of running through it and the results have been achieved with higher accuracy. (9 pt).

Corresponding Author:

Hesam Omranpour,
Departement Computer Engineering,
Amirkabir University of Technology,
424 Hafez Ave, Tehran, Iran, 15875-4413.
Email: H_Omranpour@aut.ac.ir

1. INTRODUCTION

The PSO algorithm is one of the modern evolutionary algorithms. Kennedy and Eberhart first proposed this algorithm. PSO was developed through simulation of a simplified social system, and has been found to be robust in solving continuous non-linear optimization problems [1–3].

The PSO algorithm can produce high-quality solutions within shorter calculation time and more stable convergence characteristics than other stochastic methods [1–3].

Several other researchers have proposed alterations to the particle swarm algorithm to allow it to operate on binary spaces. Agrafiotis and Cedeño [4] used the locations of the particles as probabilities to select features in a pattern-matching task. Each feature was assigned a slice of a roulette wheel based on its floating-point value, which was then discretized to {0, 1}, indicating whether the feature was selected or not. Mohan and Al-Kazemi [5] suggested several ways that the particle swarm could be implemented on binary spaces. One version, which he calls the “regulated discrete particle swarm,” performed very well on a suite of test problems. In Pamparä et al. [6], instead of directly encoding bit strings in the particles, each particle stored the small number of coefficients of a trigonometric model (angle modulation), which was then run to generate bit strings.

Extending PSO to more complex combinatorial search spaces is also of great interest. The difficulty there is that notions of velocity and direction have no natural extensions for TSP tours, permutations, schedules, etc. Nonetheless, progress has recently been made [Clerc 7, 8;Moraglio et al. 9]but it is too early to say if PSO can be competitive in these spaces.

Dynamic problems are challenging for PSO, a self-adapting multi-swarm has been derived [Blackwell 10]The multi-swarm with exclusion has been favorably compared, on the moving peaks problem, to the hierarchical swarm, PSO re-initialization and a state-of-the-art dynamic-optimization evolutionary algorithm known as self-organizing scouts.

To deal with discrete events, an algorithm based on discrete-particle-swarm-optimization was developed in [11]. This approach solves the overlapping coalition formation problem in multiple virtual organizations.

Recently, in [12] presented a PSO method demonstrating a significant performance improvement over the SPSO, QIPSO, UPSO, FIPS, DMSPSO, and CLPSO algorithms. Because the proposed method utilizes fuzzy set theory for the adaptation of parameters, it is referred to as the adaptive fuzzy PSO (AFPSO).

The rest of the paper is as follows. In Section 2, the PSO algorithm is presented. The proposed our algorithm is presented in Sections 3. In Section 4, we present the experimental results. The paper concludes in Section 5.

2. THE PSO ALGORITHM

In the PSO (PSO) algorithm, each particle searches for an optimal solution to the objective function in the search space. Each particle dynamically updates its position based on its previous position and new information regarding velocity. Its best location found in the search space so far is called pbest and the best location found for all the particles in the population is called gbest.

PSO emulates the swarm behavior and the individuals represent points in the D-dimensional search space. A particle represents a potential solution. The velocity Vid and position Xid of the dth dimension of the ith particle are updated as follows (1), (2):

Vik+1 = wVik + c1 * random1 * (pbesti – Xik) + c2 * random2 * (gbest -Xik) (1)

Xik+1 =Xik + Vik+1 (2)

WhereXi = (Xi1, Xi2, …,XiD) is the position of $i$the particle; Vi = (Vi1, Vi2, …, ViD) represents velocity of particle i. pbestiis the best previousposition yielding the best fitness value for the ith particle;; and gbestis the best position discovered by the whole population. c1 and c2 are the acceleration constants reflecting the weighting of stochastic acceleration terms that pull each particle toward pbestand gbest positions, respectively. random1and random2are two random numbers in the range (0, 1). A particle's velocity on each dimension is clamped to a maximum magnitude Vmax[13].

The inertia weight w is given by

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times k \quad (3)$$

Where Wmax is the initial weight, Wmin is the final weight, itermax is the maximum iteration number, k is the current iteration number. This formula usually has used in fuzzy methods.In some algorithms in the amount of w is equal to 0.7.This parameter is accountable for balancing between local and global search, consequently, needing less or more iterations for the algorithm to converge. A small value of inertia weight implies in a local search; a high one leads to a global search, yet with a high computational cost.However, linear decreasing inertia function may also be used if it is interested in reduce the influence of past velocities during the optimization process.

3. THE PROPOSED ALGORITHM

In this section, we propose aDynamic PSO algorithm. We create a new element and it is called pworst.In our method,all the particles in the population search their backspace too, as you can see in follow figure, might be there is the best minimum in out of vision of the particles.

In this example, according to the previous algorithms, because of one of the particles is in the local minimum, the other particleswill converge to it. In result all the particles will trap in local minimum (Figure 2).

As can be seen in the Figure 3,with using the formula (4) and given thatthe rand is between zero and one, the space of Ais not searched, in result, the space of B is searchedhope get to the national extremum.So in our algorithm (Dynamic PSO), the particles could search its backspace, and will find the best national extermum.

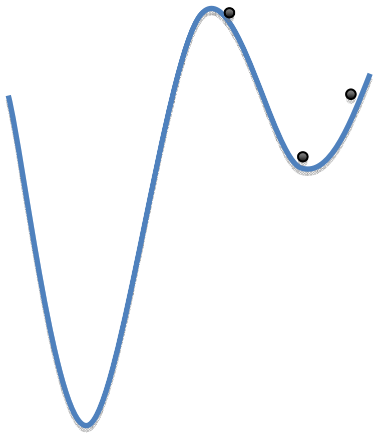


Figure1. Example of PSO’s problem

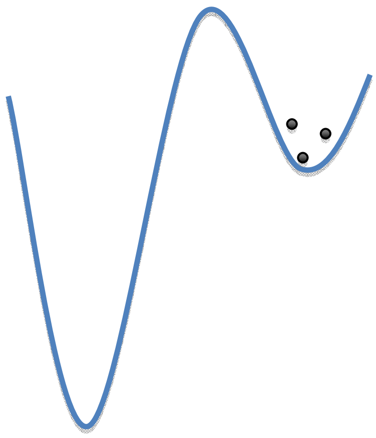


Figure2. Local Minimum

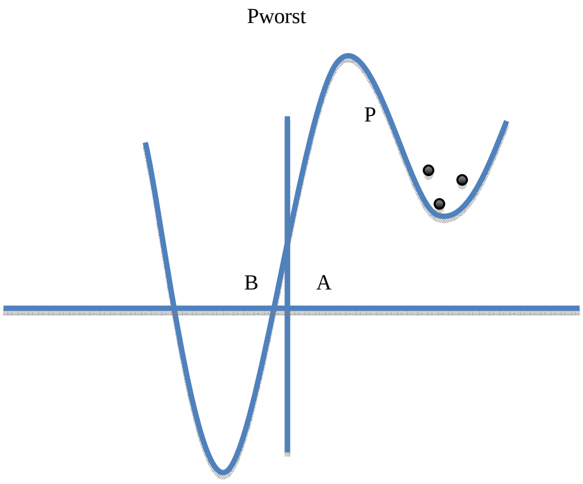


Figure 3.Pworst

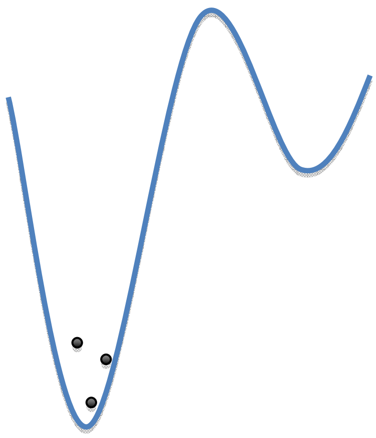


Figure4. Best Minimum.

The velocity Vid and position Xid of the dth dimension of the ith particle are updated as follows 4 and 5.

Vik+1 = wVik + c1 * random1 * (pbesti – Xik) + c2 * { (pworsti – Xik)+ random2 * (pworsti – Xik) } (4)

Xik+1 =Xik + Vik+1 (5)

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times k \quad (6)$$

As can be seen in the flowchart (Figure 5), the proposed algorithm is as follows:

The first, the particles are initialized, amount of Pbest and Pworstin the first stage are equaled initial place,then for each particle is defined as a variable that will count the number of times that fis not better (such CLPSO), if this variable does not reach the max, in the moving of particles, the C1i value iszero and C2ivalue is equal to two.If Mreaches the max, this means that number of the maxmove, the particle has not moved to a better place, in result, the C1i value change to zero and C2i value change to two.This action is done only once to avoid the trap of local minimum.

Each particle also has ind-Pworseand ind-Pbest indexesthat represent the particle iin place Piwill useof the particles ind-Pworseand ind-Pbest to update their movements.This action is for more variety in the population and search environment. Until M value has not been reached the max, theseindexes arefixed for the particle i move to better place of the particle correctly. But the moment that Misequaledthe max,theseindexes take a new value randomly and M be equaled to zero and untilM has not reached the max again, these indexesremain constant.

At the moment that M b equaled the max, after one move (leading to the trigger and escape from the local extremum), the C1i value change totwo and C2i value change to zero to maintain convergence of the algorithm.

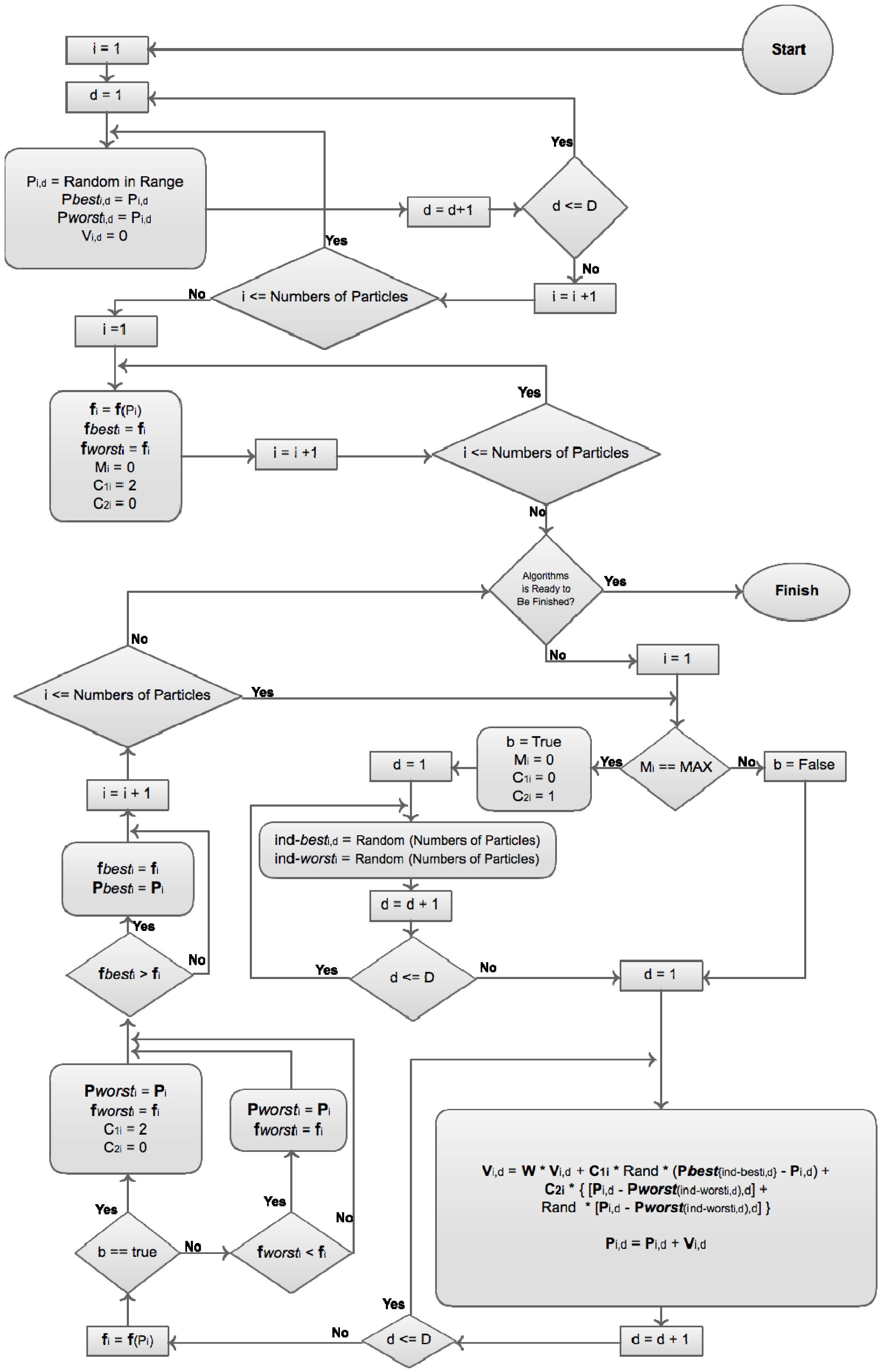


Figure5. Flowchart of Dynamic PSO

4. The experimental results

In this section, the proposed DPSO algorithm is compared with theAFPSO [12],CLPSO [13],andCPSO[14]algorithms. The main aim of this paper was to improve the performance of the PSO when dealing with multimodal problems; therefore, we tested the proposed algorithms with various multimodal functions. The forth-multimodal benchmark functions with respective dimensions 10 and 30 are for comparison. Functions 1-4 were selected from [13].

These functions are multimodal problems to be minimized. All the test functions are shown as follows:

1) Rastrigin Function

$$f_6(x) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$$

2) Rosenbrock Funcion

$$f_2(x) = \sum_{i=1}^{D-1} \left(100\left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right)$$

3) SchewfelFuncion

$$f_8(x) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin\left(|x_i|^{\frac{1}{2}}\right)$$

4) AckleyFuncion

$$f_3(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$$

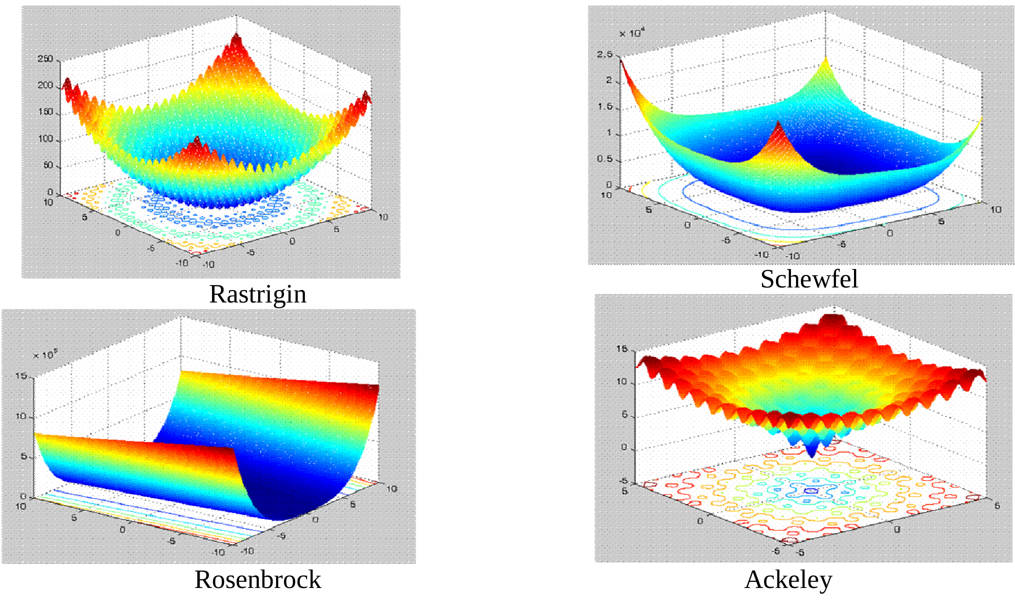
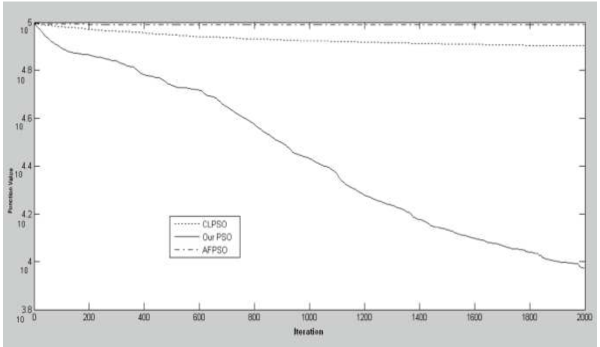
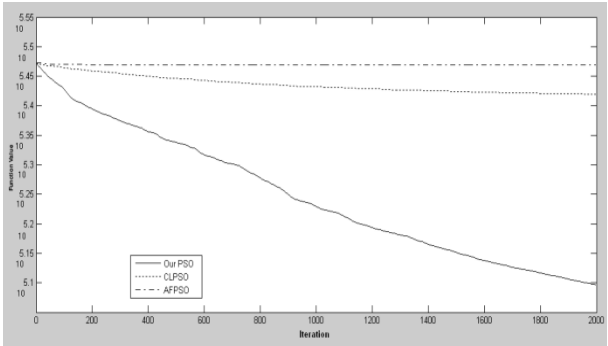


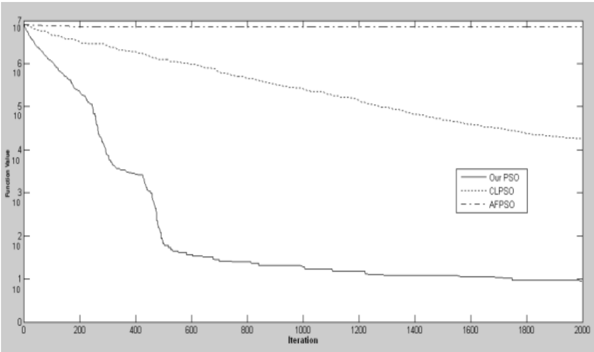
Figure6. Benchmark Functions



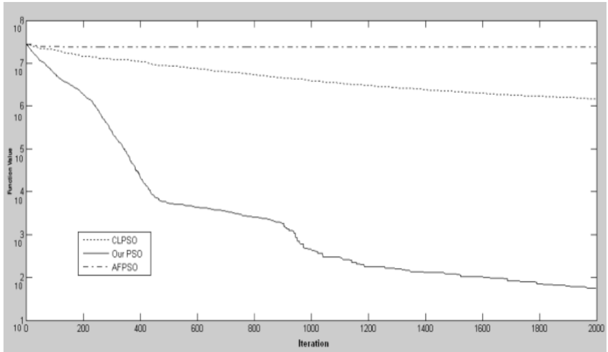
Rastrigin Function – 10D



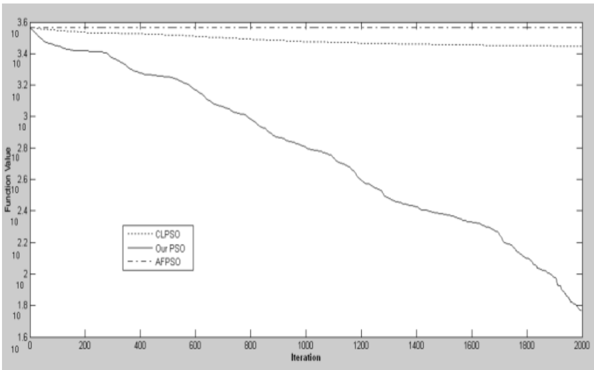
Rastrigin Function – 30D



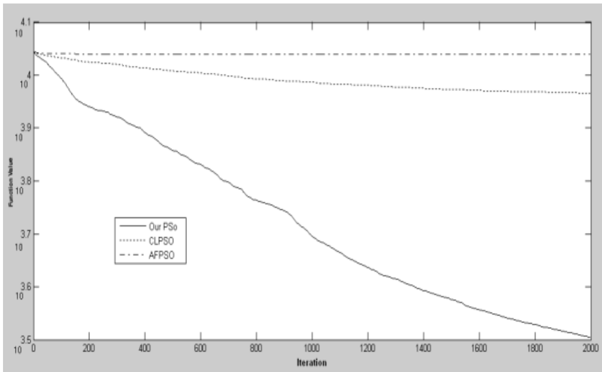
Rosenbrock Function – 10D



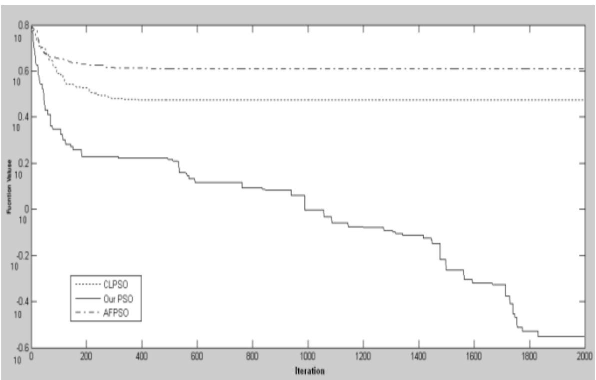
Rosenbrock Function – 30D



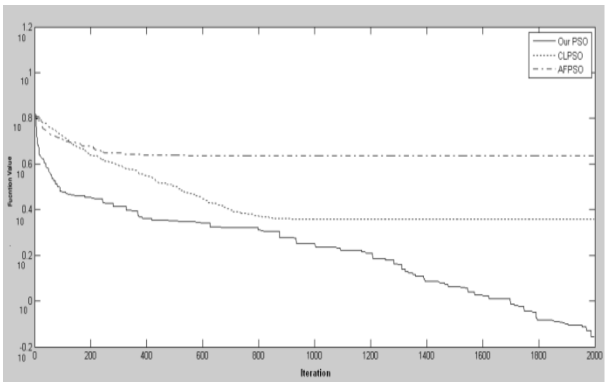
Schewfel Funcion – 10D



Schewfel Funcion – 30D



Ackley Funcion – 10D



Ackley Funcion – 30D

Figure7. Convergence performance of the 3 different PSOs on the 4 test functions (10D and 30 D)

Table 1. GLOBAL OPTIMUM, RESULTS for 10D and 30Dproblems, SEARCH RANGES AND INITIALIZATION RANGES OF THE RastriginFUNCTION

Gbestval	Range	x*	F(x*)	Iteration	Particle numbers	Dimension	PSOs
7.9721e+004	-100 , -99	0	0	2000	10	10	CLPSO
1.2158e+004	-100, -99	0	0	2000	10	10	DPSO
9.8010e+004	-100, -99	0	0	2000	10	10	CPSO
9.8022e+004	-100, -99	0	0	2000	10	10	AFPSO
2.6201e+005	-100, -99	0	0	2000	30	30	CLPSO
1.2119e+005	-100, -99	0	0	2000	30	30	DPSO
2.94030e+5	-100, -99	0	0	2000	30	30	CPSO
2.9408e+005	-100, -99	0	0	2000	30	30	AFPSO

Table 2. GLOBAL OPTIMUM, RESULTS for 10D and 30D problems, SEARCH RANGES AND INITIALIZATION RANGES OF THE Rosenbrock FUNCTION

Gbestval	Range	x*	F(x	Iteration	Particle numbers	Dimension	PSOs
1.4394e+006	-10, -9	0	0	2000	10	10	CLPSO
56.1162	-10, -9	0	0	2000	10	10	DPSO
2.3492900 e+007	-10, -9	0	0	2000	10	10	CPSO
2.3600e+007	-10, -9	0	0	2000	10	10	AFPSO
8.4873e+003	-10, -9	0	0	2000	30	30	CLPSO
10.3352	-10, -9	0	0	2000	30	30	DPSO
7.290900e+6	-10, -9	0	0	2000	30	30	CPSO
7.3225e+006	-10, -9	0	0	2000	30	30	AFPSO

Table 3. GLOBAL OPTIMUM, RESULTS for 10D and 30D problems, SEARCH RANGES AND INITIALIZATION RANGES OF THE SchewfelFUNCTION

Gbestval	Range	x*	F(x*)	Iteration	Particle numbers	Dimensio	PSOs
2.8281e+003	360,361	420.96	0	2000	10	10	CLPSO
132.5651	360,361	420.96	0	2000	10	10	DPSO
3.6488e+003	360,361	420.96	0	2000	10	10	CPSO
3.6492e+003	360,361	420.96	0	2000	10	10	AFPSO
9.1486e+003	360,361	420.96	0	2000	30	30	CLPSO
2.8546e+003	360,361	420.96	0	2000	30	30	DPSO
1.0946e+004	360,361	420.96	0	2000	30	30	CPSO
1.0949e+004	360,361	420.96	0	2000	30	30	AFPSO

Table 4. GLOBAL OPTIMUM, RESULTS for 10D and 30D problems, SEARCH RANGES AND INITIALIZATION RANGES OF THE Ackley FUNCTION

Gbestval	Range	x*	F(x*)	Iteration	Particle numbers	Dimension	PSOs
2.4083	-2,-1	0	0	2000	10	10	CLPSO
0.7720	-2,-1	0	0	2000	10	10	DPSO
3.6254	-2,-1	0	0	2000	10	10	CPSO
4.4784	-2,-1	0	0	2000	10	10	AFPSO
2.3168	-2,-1	0	0	2000	30	30	CLPSO
0.2515	-2,-1	0	0	2000	30	30	DPSO
3.6254	-2,-1	0	0	2000	30	30	CPSO
3.6344	-2,-1	0	0	2000	30	30	AFPSO

5. CONCLUSION

Particle swarm optimization is one of the intelligent methods for solving optimization problems. This paper presents a method based on optimization of particles that have too much power over the local extremum. Use the search space for particles moving in the worst behind them that has not been observed,this algorithm makes the rest of the population not involved.

The using of the dynamic steps in the vector of the worst particle to search the space behind them that has not been observed, in this algorithm makes the population not involved in the static mode.As was observed in a variety of functions, the proposed algorithm as well as the local extremum passes.

Future Work, change constant coefficients to the dynamic mode in the related equations to calculate particle velocity according to the current situation of populationand set the appropriate amount in each step.

REFERENCES

[1] J. Kennedy,R. Eberhart,“Particle swarm optimization,” IEEE Int Conf Neural Networks, pp.1942-1948,1995.

[2] T. Niknam, M. Nayeripour,J. Olamaei, A. Arefi,“An efficient hybrid evolutionary optimization algorithm for daily Volt/Var control at distribution system including DGs,”. Int Rev Electr Eng, 3(3), pp.1–11,2008

[3] J. Olamaei,T. Niknam, G. Gharehpetian,“Application of particle swarm optimization for distribution feeder reconfiguration considering distributed generators,” Appl Math Comput J, 201(1–2) pp.575–86, 2008.

[4] D.K. Agrafiotis, W.Cedeño,“Feature selection for structure-activity correlation using binary par- ticle swarms,” Journal of Medicinal Chemistry, 45(5), pp.1098–1107,2002.

[5] C.K. Mohan, B. Al-Kazemi, “Discrete particle swarm optimization,” In Proceedings of the workshop on particle swarm optimization, Indianapolis, IN, Purdue School of Engineering and Technology, IUPUI,2001.

[6] G.Pamparä, N. Franken, A.P. Engelbrecht, “Combining particle swarm optimization with angle modulation to solve binary problems,” In Proceedings of the IEEE congress on evolutionary computation (CEC) Piscataway: IEEE,2005, pp.225–239.

[7] M. lerc, “Discrete particle swarm optimization, illustrated by the traveling salesman problem,” In B. V. Babu & G. C. Onwubolu (Eds.), New optimization techniques in engineering,Berlin: Springer,2004, pp.219–239.

[8] M. Clerc, Particle swarm optimization, London: ISTE,2006b.

[9] A. Moraglio,C. Di Chio, R. Poli, “Geometric particle swarm optimization,” In M. Ebner et al. (Eds.), Lecture notes in computer science: Vol. 4445. Proceedings of the European conference on genetic programming (EuroGP),Berlin: Springer, 2007, pp. 125–136.

[10] T.M. Blackwell, “Particle swarm optimization in dynamic environments,” In S. Yand, Y. Ong, &Y. Jin (Eds.), Evolutionary computation in dynamic environments Springer, Berlin,2007, pp.29–49. DOI 10.1007/978-3-540-49774-5-2.

[11] G. Zhang, J. Jiang, Z. Su, M. Qi, H. Fang, “Searching for overlapping coalitions in multiple virtual organizations,” Information Sciences 180 (17), 2010,pp.3140–3156.

[12] Y.T. Juang et al., “Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions,” Inform. Sci, 2010, DOI 10.1016/j.ins.2010.11.025.

[13] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, “Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions,” IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 10, NO. 3, JUNE 2006.

[14] F. Bergh and A.P. Engelbrecht, “A cooperative approach to particle swarm optimization,” IEEE Trans. Evol. Comput., vol. 8, pp.225–239, Jun. 2004.