Particle Swarm Optimization with Exploratory Move

Nanda Dulal Jana[1] and Jaya Sil[2]

[1] Department of Information Technology, National Institute of Technology, Durgapur, WB, India
nanda.jana@gmail.com

[2] Department of Computer Science & Technology, Bengal Engineering & Science University, Shibpur, WB, India
js@cs.becs.ac.in

Abstract. Particle Swarm Optimization (PSO) algorithm is a swarm based algorithm deliver good performance in many optimization problems. However, PSO has tendency of trapping into local optima. In the paper, an improved PSO algorithm has been proposed by employing Exploratory Move on global best particle of the swarm called as PSO with exploratory move (ExPSO) algorithm. In the proposed approach in order to preventing PSO algorithm from trapping into local optima, particles are jumped to an unknown position made by the exploratory move. The performance of the ExPSO algorithm has been investigated on a set of eight standard benchmark functions and results are compared with the simple PSO, constriction factor PSO (CFPSO) and inertia weight PSO (IWPSO). The numerical results show that the ExPSO algorithm performs better, robust and statistically significant on most of the test cases.

Keywords: Particle Swarm Optimization, Exploratory Move, Exploration and Exploitation, Local Optima.

1 Introduction

PSO is a population based stochastic optimization algorithm inspired by the social behaviour of bird flocking, firstly introduced by Kennedy and Eberhart [1,2]. In the early stage of development of PSO algorithm, it was used for continuous optimization problems. Now PSO has received more and more attentions by many researchers due to its promising optimization capacity in various fields [3,4]. However, PSO is often trapped into local optima due to premature convergence while the convergence rate decreases in the latter period of evolution. Therefore, accelerating convergence rate and avoiding local optima become the two most important and appealing challenges in PSO research.

To overcome the above drawbacks of PSO, existing methods attempted to improve the performance by introducing variable parameters [5] or modifying the updating equations [6] or adopting the operators of the optimization algorithm

[7,8]. In [9], a constriction factor algorithm was introduced to ensure the convergence of PSO. The lack of population diversity in PSO algorithms is the reason of premature convergence [10]. Therefore, in addition of search/move operator to PSO, enhancement of global search capacity is most important to improve its performance. In the paper, an improved PSO is proposed by introducing exploratory move on global best particle of the swarm. In the exploratory move, the current point is perturbed one at a time along each variable in positive and negative direction and the best point is recorded. Therefore, the proposed method creates set of search directions iteratively in such a way so that the search directions completely cover the search space. As a result, global search capability increases for its long jump ability. The proposed method shows the fast convergence speed and greatly overcome the tendency of trapping into local optima.

The rest of the paper is organized as follows: section 2 describes the basic concepts of the PSO. In section 3, exploratory move and the proposed ExPSO algorithm has been described while in section 4 experimental analyses, results and discussion are presented. Section 5 concludes the paper with a possible direction in future works.

2 Particle Swarm Optimization Algorithm

In PSO algorithm, each member of the swarm is called a 'particle' and each particle flies around in the D-dimension search space with a velocity. Each particle in the PSO has a position and a velocity, its evaluation is achieved using the objective function/fitness function (f) of the optimization problem, whose variables are the particle position dimensions. The particle updating method tries to move particles to better positions by accelerating them towards personal best position of a particle and global best position of particles.

In general, a particle moves in a D-dimensional search space and a swarm contains N such particles. The position vector and velocity vector of the i^{th} particle is represented by $X_i = (x_{i1}, x_{i2}, .., x_{iD})$ and $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$ respectively. Each particle maintains a memory of its previous best position which is represented by $X_{pbest} = (x_{pbest1}, x_{pbest2}, ..., x_{pbestD})$ and best of all the particles in the swarm by $X_{gbest} = (x_{gbest1}, x_{gbest2}, .., x_{gbestD})$. The basic PSO algorithm can be described using equations (1)and (2).

$$V_i(t+1) = \omega * V_i(t) + c_1 r_1 (X_{pbesti}(t) - X_i(t)) + c_2 r_2 (X_{gbesti}(t) - X_i(t)) \quad (1)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

Where c_1 and c_2 are positive constants and called the acceleration coefficients, r_1 and r_2 are two uniformly distributed random number in the interval [0, 1]. V_i is the velocity of i^{th} individual of dimension D and X_i is the current position of i^{th} individual on dimension D. X_{pbesti} is the best position of the i^{th} particle and X_{gbesti} represents the best position found so far by all particles in the swarm at time t. In Eq. (1), ω is the inertia weight which provides the necessary diversity to the swarm by changing the momentum of particles.

Algorithm 1. Exploratory Move

```
Input: Initialize current solution ($X^c$) with dimension D. $X_i^c$ is the $i^{th}$ dimension perturbed by$\Delta_i$
Output: New best point
begin
    for $i = 1$ to $D$ do
        $y_i^+ = x_i^c + \Delta_i$
        // $y_i^+$ is the $i^{th}$ component of $Y^+ = (x_1^c, x_2^c, ...., x_i^c + \Delta_i, x_{i+1}^c, ...x_D^c)$
        $y_i^- = x_i^c - \Delta_i$
        // $y_i^-$ is the $i^{th}$ component of $Y^- = (x_1^c, x_2^c, ...., x_i^c - \Delta_i, x_{i+1}^c, ...x_D^c)$
        evaluate $f = f(X^c), f^+ = f^+(Y^+) and f^- = f^-(Y^-)$
        find $f_{min}, min(f, f^+, f^-)$. Set X corresponds to $f_{min}$
        if $i = D$ then
            Break
    if $X^c \neq X$ then
        Success
    else
        failure
```

3 Proposed Methodology

3.1 Exploratory Move

An exploratory move is performed systematically in the vicinity of the current point to find the best point around the current point. In this move, the current point is perturbed in positive and negative directions along each variable one at a time and the best point is recorded. The current point is changed to the best point at the end of each variable perturbation. If the point found at the end of all variable perturbations is different than the original point, the exploratory move becomes success otherwise not. The exploratory move procedure shown in Algorithm 1. In the proposed method, a set of search directions are iteratively generated to cover the search space completely. The process starts from any point in the search space and the search can be reached to any other point in the search space by travelling along the search directions. In an N-dimensional problem, its requires at least N linearly independent search directions. Among many possible combinations of N search directions, some combinations may help to reach the destination faster with less number of iterations.

3.2 Proposed Mechanism

PSO has been shown rapid convergence in the first part of the search and then slow down or no improvement has been observed in the fitness function. This behaviour has been attributed as the loss of diversity in the population. It is possible to lead the swarm away from a current location by improving a single individual if the improved individual becomes the new global best. The global best individual attracts all members of the swarm. In the paper, we consider *gbest* position as the current position and exploratory move is applied on this current position along each component,selected randomly. Suppose

Algorithm 2. Algorithm for Proposed Mechanism

```
Input: Initialize the particles position and velocity
Output: Global best
begin
    Calculate fitness. Updating pbest and gbest
    while (stopping condition is not reached) do
        for $i =$ to $N$ do
            Updating velocity and position using Eq. (1) and (2)
            calculate fitness. updating the pbest and the gbest
        call "Exploratory Move" on global best and find the new global best.
        if the new global best is better than the gobal best, then replace the global best.
```

$X_{gbest} = (x_{gbest1}, x_{gbest2}, ..., x_{gbestD})$ and select a random integer k between 1 to D. Exploratory move is therefore, performed on k^{th} component, X_{gbestk} of X_{gbest} to compute $X_{gbest}(+)$ and $X_{gbest}(-)$ along the positive and negative direction in the search space by random increment and decrement. This is represented by
$X_{gbest}(+) = (x_{gbest1}, x_{gbest2}, ..., x_{gbestk} + r_1, ..., x_{gbestD})$
$X_{gbest}(-) = (x_{gbest1}, x_{gbest2}, ..., x_{gbestk} - r_2, ..., x_{gbestD})$

Evaluating $f(X_{gbest}(+))$ and $f(x_{gbest}(-))$,the best position is selected based on the fitness values of the points X_{gbest}, $X_{gbest}(+)$and $X_{gbest}(-)$. This point corresponds to the current position of the next iteration of the exploratory move. Exploratory move terminates when fitness value of the best position reaches to the desired error. Otherwise, it is performed on all components of the dimensions. The proposed algorithm is shown in Algorithm 2. The proposed approach escapes local optima and new positions become the new global best position.

4 Experimental Studies

4.1 Benchmark Functions

Eight benchmark functions are considered for experiment, which are widely adopted in global optimization algorithms [11,12]. The test functions f_1 and f_2 are unimodal, having only one global minimum 0. The benchmark functions from f_3 to f_7 are multimodal functions having the global minimum at the origin or very near to the origin. We have also taken a noisy function f_8, where a uniformly distributed random noise is added to the function. The description of these benchmark functions and their global optima are given in Table 1.

4.2 Parameter Settings

For the purpose of performance evaluation, we compare the purposed algorithm with other PSO algorithms, simple PSO, constriction factors PSO (CFPSO) and inertia weights PSO (IWPSO) over 50 independent runs. Experiment is carried out for eight benchmark problems having 30 dimensions and population size is 50. The parameters of the proposed algorithm are $c_1 = c_2 = 1.49618$ and decreasing inertia weight (ω) in each iteration starting from 0.9 to 0.4. The acceleration

Table 1. The Benchmark Functions

Function	Mathematical Representation	Range	Option
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	0
Schwefel	$f_2(x) = \sum_{i=1}^{D} \sum_{j=1}^{j} x_i^2$	$[-100, 100]$	0
Griewank	$f_3(x) = \frac{1}{40000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]$	0
Rastrigin	$f_4(x) = \sum_{i=1}^{D} (x_i^2 - 10cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	0
Rosenbrock	$f_5(x) = \sum_{i=1}^{D-1} [(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2]$	$[-100, 100]$	0
Ackley	$f_6(x) = 20 + e - 20e^{-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{D} x_i^2}}$ $-e^{\frac{1}{n}\sum_{i=1}^{D} cos(2\pi x_i)}$	$[-32, 32]$	0
Weierstrass	$f_7(x) = \sum_{i=1}^{D} (\sum_{k=0}^{k_{max}} [a^k cos(2\pi b^k (x_i + 0.5))])$ $-n \sum_{k=0}^{k_{max}} [a^k cos(2\pi b^k 0.5)]$ with $a = 0.5$, $b = 3$ and $k_{max} = 20$	$[-100, 100]$	0
Dejong's Noisy Function	$f_8(x) = \sum_{i=0}^{D-1} (i + 1) x_i^4 + rand[0, 1]$	$[-1.28, 1.28]$	0

coefficients ($c_1 and c_2$) for the simple PSO, CFPSO and IWPSO are set to 2. The inertia weight (ω) for simple PSO is set to 0.732 and for IWPSO is in decreasing order and in each iterations set from 0.9 to 0.4. Maximum velocity, $V_{max} = X_{max}$ where $[X_{min}, X_{max}]$ is the search space range. The same initial population is used for all PSO algorithms. In this work, the termination criteria are considered as maximum number of generations i.e. 4000 and $E = |f(X) - f(X^*)| \leq e$ ($f(X)$ is the current best and $f(X^*)$is the global optimum) is the best-error of a run of the algorithm and e is the threshold error. In our experiment error $e = 0.001$. Algorithms are implemented using MATLAB 7.6.0 (R2008a) applied on Intel (R) Core (TM) i7-2670QM CPU @ 2.20 GHz with 8 GB RAM on windows 7 Home Premium platform.

4.3 Results and Discussion

Table 2 presents the mean, standard deviation, average number of generations and success rate (frequency of hitting the optimum) of the benchmark functions respectively using the four PSO algorithms over 50 independent runs respectively. The best results are marked in boldface. Convergence characteristics of each functions are comaperd with the PSO algorithms are shown in Figure 1. From the results it has been observed that the proposed method outperforms over other algorithms with 100% Success Rate(SR) for the function f_1, f_2, f_4, f_6 and f_7. In case of the functions f_3 and f_5, proposed ExPSO algorithm produced better solution but no improvement in SR as well as convergence speed. The ExPSO achieved better results than other algorithms with zero SR for function f_8. The quality of solution obtained by the proposed method with minimum number of average generations than simple PSO, CFPSO and IWPSO except the function f_8 as shown in Table2.

Table 3 shows results of unpaired t-tests between the best algorithm and the second best in each case (standard error difference of the two means, 95% confidence interval of this difference, the t- value and the two - tailed P value). For all cases in Table 3 sample size is 50 and degree of freedom is 98. It is interesting

Table 2. Result of 8 functions

Function no.	Algorithm	Evaluation Metrics			
		Mean	Std. Dev.	Avg no. Generations	SR
f_1	ExPOS	**8.740e-004**	**1.343e-004**	**165.36**	**100.00**
	PSO	$2.391e+001$	$1.766e+001$	4000.00	0.00
	CFPSO	$2.431e+003$	$2.314e+002$	4000.00	0.00
	IWPSO	$5.200e+003$	$6.141e+003$	3438.36	54.00
f_2	ExPOS	**9.864e-004**	**1.436e-005**	**1944.22**	**100.00**
	PSO	$1.618e+004$	$5.285e+003$	4000.00	0.00
	CFPSO	$4.612e+003$	$7.257e+002$	4000.00	0.00
	IWPSO	$5.804e+004$	$2.953e+004$	4000.00	0.00
f_3	ExPOS	**1.259e-001**	**8.128e-002**	**3870.20**	**4.00**
	PSO	$3.012e+000$	$1.270e+001$	4000.00	0.00
	CFPSO	$2.294e+001$	$2.155e+000$	4000.00	0.00
	IWPSO	$4.340e+001$	$5.548e+001$	3758.00	24.00
f_4	ExPOS	**9.553e-004**	**4.573e-005**	**767.08**	**100.00**
	PSO	$1.020e+002$	$2.627e+001$	4000.00	0.00
	CFPSO	$1.882e+002$	$1.227e+001$	4000.00	0.00
	IWPSO	$1.697e+002$	$4.690e+001$	4000.00	0.00
f_5	ExPOS	**1.280e+001**	**2.740e+001**	**3468.58**	**30.00**
	PSO	$6.024e+005$	$5.174e+005$	4000.00	0.00
	CFPSO	$4.508e+007$	$1.010e+007$	4000.00	0.00
	IWPSO	$7.600e+005$	$4.314e+005$	4000.00	0.00
f_6	ExPOS	**9.571e-004**	**4.182e-005**	**834.34**	**100.00**
	PSO	$3.216e+000$	$1.918e+000$	4000.00	0.00
	CFPSO	$1.041e+001$	$3.605e-001$	4000.00	0.00
	IWPSO	$1.996e+001$	$3.048e-003$	4000.00	0.00
f_7	ExPOS	**0.000e+000**	**0.000e+000**	**3.96**	**100.00**
	PSO	$1.819e-005$	$1.286e-004$	24.02	**100.00**
	CFPSO	$1.053e+001$	$1.124e+001$	3714.36	14.00
	IWPSO	$1.383e-013$	$2.766e-014$	8.54	**100.00**
f_8	ExPOS	**2.609e-003**	**1.052e-003**	4000.00	0.00
	PSO	$4.004e+003$	$9.258e+003$	4000.00	0.00
	CFPSO	$6.813e+002$	$1.441e+002$	4000.00	0.00
	IWPSO	$5.600e+003$	$1.280e+004$	4000.00	0.00

Table 3. Results of Unpaired t-test on the data of Table 2

Function	Std. Error	T	95% Conf. Interval	Two-Tailed P	Significance
f_1	2.498	9.5732	$(-28.865, -18.953)$	< 0.0001	Extremely Significant
f_2	102.629	44.9383	$(-4815.664, -4408.3341)$	< 0.0001	Extremely Significant
f_3	1.796	1.6074	$(6.451, 0.677)$	0.1112	Not Significant
f_4	3.715	27.4550	$(-109.377, -94.627)$	< 0.0001	Extremely Significant
f_5	73171.410	8.2325	$(-747593.512, -457180.888)$	< 0.0001	Extremely Significant
f_6	0.271	11.8529	$(-3.753, -2.677)$	< 0.0001	Extremely Significant
f_7	0.000	1.0002	$(-0.0001, -0.0000)$	0.3197	Not Significant
f_8	20.379	33.4316	$(-721.739, -640.856)$	< 0.0001	Extremely Significant

to see from Table 2 and Table 3 that in most of the cases the proposed method meets or beats the nearest competitor in a satistically meaningfull way. These results show that the proposed method leads to significant improvements in most cases.

5 Conclusions

The proposed algorithm shows better performance both in early period of generations and later period of generations. Thus it achieves effective trade-off

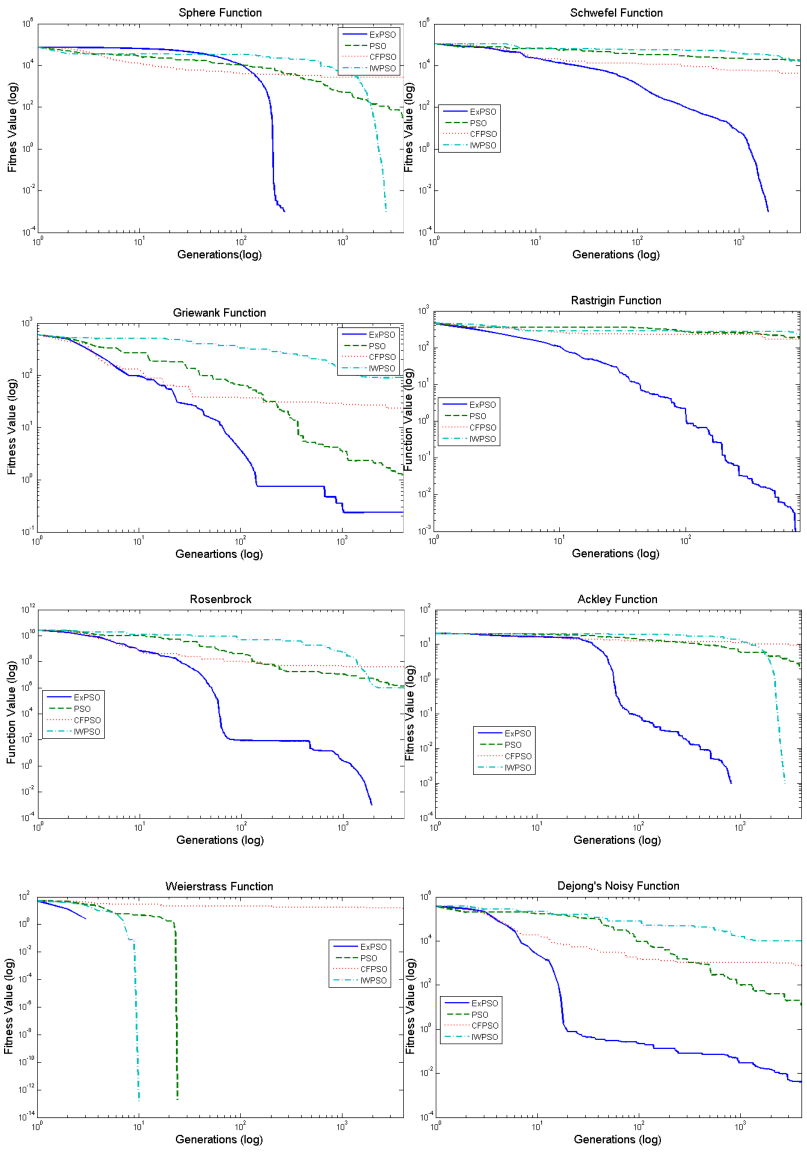

Fig. 1. Convergence characteristics of 8 benchmark functions for 30 dimensions

between exploration and exploitation. The experimental results show that for the optimization problems described by the benchmark functions, our algorithm can obtain better performances than simple PSO, constriction factor PSO and inertia weight PSO. In our future study, a method that can adaptively tune the parameters (inertia weight and constriction factor) of PSO will be investigated using exploratory move.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE International Conference on Neural Networks (ICNN 1995), pp. 1942–1948. IEEE Press, Australia (1995)
2. Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: 6th International Symposium on Micro Machine and Human Science (ISMMHS 1995), Nagoya, Japan, pp. 39–43 (1995)
3. Ho, S.Y., Lin, H.S., Liauh, W.H., Ho, S.J.: OPSO: Orthogonal Particle Swarm Optimization and its application to task assignment problems. Journal of IEEE Trans. Syst., Man, Cybern. A, Syst., Humans 38(2), 288–298 (2008)
4. Liu, B., Wang, L., Jin, Y.H.: An effective PSO-based memetic algorithm for flow shop scheduling. Journal of IEEE Trans. Syst., Man, Cybern. B, Cybern. 37(1), 18–27 (2007)
5. Jiao, B., Lian, Z.G., Gu, X.S.: A dynamic inertia weight particle swarm optimization algorithm. Journal of Chaos, Solitons and Fractal 37(1), 698–705 (2008)
6. Ling, S.H., Iu, H.H.C., Chan, K.Y.: Hybrid particle swarm optimization with wavelet mutation and its industrial applications. Journal of IEEE Trans. Syst. Man Cybernetics 38(3), 743–763 (2008)
7. Omran, M.G.H., Engelbrecht, A.P., Salman, A.: Differential evolution based particle swarm optimization. In: IEEE Conf. Swarm Intelligence Symposium (SIS), pp. 112–119 (2007)
8. Chen, M.R., Li, X., Zhang, X., Lu, Y.Z.: A novel particle swarm optimizer hybridized with extremal optimization. Journal of Appl. Soft Comput. 10(2), 367–373 (2010)
9. Clerc, M.: The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: The Congress on Evolutionary Computation (CEC 1999), pp. 1951–1957. IEEE Service Center, Piscataway (1999)
10. Ratnaweera, A., Halgamuge, S.K., Watson, H.C.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. Journal of IEEE Transactions on Evolutionary Computation 8(3), 240–255 (2004)
11. Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. Journal of IEEE Trans. Evol. Comput. 10(3), 281–295 (2006)
12. Zhan, Z., Zhang, J., Li, Y., Chung, H.S.: Adaptive Particle Swarm Optimization. Journal of IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 39(6), 1362–1381 (2009)