Particle Swarm Optimization with Adaptive Polynomial Mutation

Tapas Si*, N.D Jana* and Jaya Sil†
* Department of Information Technology
National Institute of Technology,Durgapur, West Bengal, India
Email: {c2.tapas,nanda.jana}@gmail.com
† Department of Computer Science and Technology
Bengal Engineering and Science University, Shibpur, West Bengal, India
Email:js@cs.becs.ac.in

***Abstract*—Particle Swarm Optimization (PSO) has shown its good search ability in many optimization problem.But PSO easily gets trapped into local optima while dealing with complex problems.In this work, we proposed an improved PSO, namely PSO-APM,in which adaptive polynomial mutation strategy is employed on global best particle with the hope that it will help the particles jump out local optima.In this work, we carried out our experiments on 8 well-known benchmark problems.Finally the results are compared with classical PSO and PSO with power mutation(PMPSO).**

I. INTRODUCTION

Particle swarm optimization [1] is a population based global search technique having a stochastic nature. It has shown its good search ability in many optimization problems with faster convergence speed.However, due to lack of diversity in population,PSO easily trapped into local optima while dealing with complex problems.Different mutation strategies like Cauchy Mutation [4], Gaussian Mutation [3],Power Mutation [8], Adaptive Mutation [7] are introduced into PSO for solving local optima problem. Changhe Li et al. [5] introduced a fast particle swarm optimization with cauchy mutation and natural selection strategy in the year 2007. Andrew Stacey et al. [6] used mutation in PSO with probability 1/d, where d is the dimension of particles.JIAO Wei et al. [12] proposed elite particle swarm optimization with mutation.Xiaoling Wu and Min Zhong [8] introduced power mutation into PSO (PMPSO).Coello [18] presented a hybrid PSO algorithm that incorporates a non-uniform mutation operator similar to the one used in evolutionary algorithms.Pant [17] used an adaptive mutation operator in PSO.Higashi [3] proposed a PSO algorithm with Gaussian mutation(PSO-GM).Jun Tang and Xiaojuan Zhao [7] proposed a new adaptive mutation by dynamically adjusting the mutation size in terms of current search space.A.J. Nebro et al. [14] applied a polynomial mutation to the 15 percentage of the particles.

In this work, our objective is to use adaptive polynomial mutation on global best solution in PSO to solve local optima problem and to analysis the performance and effectiveness of adaptive polynomial mutation. A comparative study is also made with PSO and PMPSO.

II. PARTICLE SWARM OPTIMIZATION (PSO)

PSO is a kind of algorithm to search for the best solution by simulating the movement and flocking of birds. PSO algorithms use a population of individual called particles. Each particle has its own position and velocity to move around the search space. Particles have memory and each particle keep track of previous best position and corresponding fitness. The previous best value is called as pbest. Thus pbest is related only to a particular particle. It also has another value called gbest, which is the best value of all the particles pbest in the swarm. The basic concept of PSO technique lies in accelerating each particle towards its pbest and the locations at each time step.

$$v_{ij}(t+1) = \omega * v_{ij}(t) + c_1 r_1 (x_{pbest} - x_{ij}(t)) + c_2 r_2 (x_{gbest} - x_{ij}(t)) \quad (1)$$

$$x_{ij}(t+1) = v_{ij}(t) + x_{ij}(t) \quad (2)$$

where c_1 and c_2 are two acceleration coefficients.r_1 and r_2 are uniformly distributed random number in [0,1]. The search region of each of the test function is specified by $(X_{min}, X_{max})^D$ where D is the dimension of the search space and X_{min} and X_{max} determines the extension in each dimension.Bounds are placed on both the displacement and velocity values that a particle can have in each dimension. The velocity is constrained within the range $(V_{min}, V_{max})^D$. Similarly when updating the velocity of a particle,if a component is greater than V_{max}, it is set back to $V_{max} * r$ where r is a uniformly distributed random number in [0,1], and if it is less than V_{min}, it is set back to $V_{min} * r$. Generally, $V_{min} = -V_{max}$ and V_{max} is set to 10%-50% of the search space range X_{max}.

III. PSO BASED ON MUTATION

The particle swarm optimization algorithm converges rapidly during the initial stages of a search, but often slows considerably and can get trapped in local optima. When particles converge to the global best particle, personal best x_{pbest} and global best x_{gbest} are equal and last two terms of the Eq.(1) become zero and the resulting equation become

$$v_{ij}(t+1) = \omega * v_{ij}(t) \quad (3)$$

978-1-4673-0126-8/11/$26.00

when t tends to infinity ,$v_{ij}(t+1) \simeq 0$ as $0 < \omega < 1$. In this circumstance,particles can not move further in the search space. Mutation strategies are used into PSO with the hope of preventing PSO from getting trapped into a local optimum through long jumps made by the mutation. Andrew Stacey et al. [6] used mutation with basic PSO (BPSO) and constriction PSO (CPSO) to solve unconstrained optimization problems. When a solution is chosen to be mutated or not with probability 1/d, where d is the number of components in the vector. On average one component is mutated. To mutate a component, a number randomly generated from a Cauchy distribution is added to the component, where p.d.f for the distribution is given by

$$f(x) = \frac{a}{\pi} \frac{1}{x^2 + a^2} \tag{4}$$

with a=0.2. Other choices of mutation operator and values for a were tried but appeared to be less effective. The Cauchy distribution is similar to the normal distribution but has more of its probability in the trails. This increases the probability that large values will be generated.

L. Kang et al. [13] proposed a fast particle swarm optimization (FPSO) algorithm by combining PSO with Cauchy mutation and an evolutionary selection strategy.

Coello [18] presented a hybrid PSO algorithm that incorporates a non-uniform mutation operator similar to the one used in evolutionary algorithms. Assume $X = (x_1, x_2, ..., x_n)$ is a solution. The mutation operator is computed as:

$$x_j = \begin{cases} x_j + \triangle(t, UB - x_j), & r = 0 \\ x_j - \triangle(t, x_j - LB), & r = 1 \end{cases} \tag{5}$$

where r is a random digit and t is the iteration index, LB and UB are the lower and upper bounds of variable respectively. The function $\triangle(t, y)$ returns a value in the range [0,1]. The function $\triangle(t, y)$ is:

$$\triangle(t, y) = y * (1 - r^{(1-1/T)^b}) \tag{6}$$

where r is a random number in the range [0, 1], T is the total number of generations and b is a system parameter determining the degree of dependency on iteration number.

Pant [17] used an adaptive mutation operator in PSO. The particles are mutated at the end of each iteration according to the following rule:

$$x_j(t+1) = x_j(t) + \sigma_j^{`} * Betarand_j() \tag{7}$$

where $\sigma_j^{`} = \sigma_j * exp(\tau N_j(0,1) + \tau^{`} N_j(0,1))$, $N_j(0,1)$ denotes a normally distributed random number with mean zero and standard deviation one. $N_j(0,1)$ indicates a different random number for each value j. τ and $\tau^{`}$ are set as $1/\sqrt{2n}$ and $1/\sqrt{2/\sqrt{n}}$ respectively. $Betarand_j()$ is a random number generated by beta distribution with parameters less than 1.

Higashi [3] proposed a PSO algorithm with Gaussian mutation, called PSO-GM, in which Gaussian distribution is used to update the velocity and position. The technique is described as:

$$x_j(t+1) = x_j(t) * (1 + G(\sigma)) \tag{8}$$

where σ is set to be 0.1 times the length of the search space in one dimension, G ( ) is a random number based on Gaussian distribution. The particles are selected at a predefined probability and their positions are determined at the probability under Gaussian distribution.

A.J. Nebro et al. [14] proposed Speed-constrained Multi-objective PSO (SMPSO) allows producing new effective particle positions in those cases in which the velocity becomes too high. Other features of SMPSO include the use of polynomial mutation as a turbulence factor and an external archive to store the non-dominated solutions found during the search. In SMPSO, a polynomial mutation is applied to the 15 percentage of the particles.

JIAO Wei et al. [12] proposed elite particle swarm optimization with mutation where bad particles are replaced by elite particles and global best individual is mutated to generate a new particle. If the new particles fitness value is better than the current best fitness value, this new particle will substitute for forgoing particle, taken as the new global best position. The mutation operation is showed as the following equation

$$P_g^{`} = P_g(1 + 0.5\eta) \tag{9}$$

where η is a random number normally distributed in the range of [0, 1] .

Chen [19] proposed a PSO algorithm with adaptive mutation to avoid premature convergence. A weighted mutation is applied on the global best particle with an adaptive probability. The probability was dynamically adjusted according to the changes of population diversity.

$$x_j(t+1) = x_j(t) + 0.5 * randn() * gbest_j(t) \tag{10}$$

where $gbest_j$ is the jth vector of the global best particle, and $randn()$ is a Gaussian distributed random number with zero mean and variance 1.

Xiaoling Wu and Min Zhong [8] introduced power mutation into PSO (PMPSO). This proposed mutation operator based on power distribution. Its distribution function is given by

$$f(x) = px^{p-1}, 0 \leq x \leq 1 \tag{11}$$

And the density function is given by

$$F(x) = x^p, 0 \leq x \leq 1 \tag{12}$$

where p is the index of the distribution. The power mutation is defined as follows:

$$x_j = \begin{cases} x_j - s(x_j - x^l), & t < r \\ x_j + s(x^u - x_j), & t \geq r \end{cases} \tag{13}$$

$$x^l = \min\{x_j\}, x^u = \max\{x_j\}, j = 1, 2, ..., D \tag{14}$$

where $t = (x - x^l)/(x^u - x)$ and $[x^l, x^u]$ is the boundary of the decision variables in the current search space, r is a random number between 0 and 1, and s is computed according to equation (11).

Jun Tang and Xiaojuan Zhao [7] proposed a new adaptive mutation by dynamically adjusting the mutation size in terms of current search space.

$$gbest_j(t+1) = gbest_j(t) + [b_j(t) - a_j(t)] * rand() \tag{15}$$

$$a_j(t) = \min(x_{ij}(t)), b_j(t) = \max(x_{ij}(t)) \quad (16)$$

where $gbest_j$ is the jth vector of the global best particle,and are the minimum and maximum values of the jth dimension in current search space respectively, $rand()$ is a random number within [0,1] and t indicates the generations.

From the above study, we found that mutation operator is used on particles in three ways. Mutation is applied on individual particles [6], [14], [17], [18] , particles along with their velocity [3] and global best particle [7], [8], [12], [19]. In our work, we have concentrated on mutating global best particles. Mutation size is an important factor while dealing with local optima problem. A long jump is very useful when global best is far away from the global optima. When global best trapped into a local optima which is near the global optima, a long jump may go to the unfeasible solution space or drive it out towards another better local optima which is far away from the global optima.In that case, mutation become non-effective. That's why mutation size should be control while dealing mutation on global best particle to get it out from the local optima. Therefore, a controlled mutation-size should be employed and we employed adaptive polynomial mutation on global best in which mutation size is decreased with increased iteration.

IV. PSO WITH ADAPTIVE POLYNOMIAL MUTATION(PSO-APM)

Polynomial Mutation is used in real coded genetic algorithm(GA) [9], [10], [16] Polynomial mutation is based on polynomial probability distribution.

$$x_j(t+1) = x_j(t) + (x_j^u - x_j^l) * \delta_j \quad (17)$$

where x_j^u is the upper bound and x_j^l is the lower bound of x_j.The parameter δ_i is calculated from the polynomial probability distribution

$$P(\delta) = 0.5(\eta_m + 1)(1 - \delta^{|\eta_m|}) \quad (18)$$

η_m is the polynomial distribution index.

$$\delta_i = \begin{cases} (2r_i)^{1/(\eta_m+1)} - 1, & r < 0.5 \\ 1 - [2(1-r_i)]^{1/(\eta_m+1)}, & r \geq 0.5 \end{cases} \quad (19)$$

The property of η_m is such that by varying its value, the perturbance can be varied in the mutated solution. If the value of η_m is large, a small perturbance in the value of η_m a variable is achieved. To achieve gradually decreasing perturbance in the mutated solutions, the value of η_m is gradually increased. The following rule is to achieve the above adaptation which is known as adaptive polynomial mutation:

$$\eta_m = 80 + t \quad (20)$$

where t is the current iteration.

In this work, we used adaptive polynomial mutation on global best solution in PSO using the following equation:

$$mgbest_j(t) = gbest_j(t) + (x_j^u - x_j^l) * \delta_j \quad (21)$$

where x_j^u is the upper bound and x_j^l is the lower bound of x_j in $gbest_j$. If mutated global best $mgbest_j$ is better than $gbest_j$, then $gbest_j$ is replaced by $mgbest_j$.

V. PARAMETTER SETTINGS

A. Benchmark Problems

There are 8 different global optimization problems, including 4 uni-modal functions($f_1 - f_4$) and 4 multi-madal functions($f_5 - f_8$), are chosen in our experimental studies.These functions were used in an early study by X. Yao et al. [21]. All functions are used in this work to be minimized.The description of these benchmark functions and their global optima are given in Table I.

B. PC configuration

- System:Fedora 13(i386)
- CPU: P IV 2GHz (Core 2 Duo)
- RAM: 3 GB
- Software: Matlab 2010b

C. Parameters of PSO

- Population Size(N)=20.
- Number of Generations=$\frac{FES}{Populationsize}$ (where FES is the number of function evalution permitted)
- FES=1×10^5
- Initial distribution index for polynomial mutation $\eta_m = 100$
- $c_1 = c_2 = 1.49445$
- $\omega = 0.72984$

VI. SIMULATION RESULTS AND DISCUSSION

A. Function Values Achieved

The obtained results are presented in Tables II and III. "Mean Best" is the mean of best solutions and standard deviation of the best solution of 30 runs for each test problem are reported in Table II and III. The convergence graph of PSO,PSO-APM and PMPSO for function f_8 are given in Fig. 1,2 and 3 respectively.

B. Mutation Rate

Mutation rate is the ratio of successful mutation(when muted global best solution is better than global best solution) and total number of mutation in a single run. Mean Mutation Rate of each method for 30 runs are given in Table III.

From Table II, we observed that the PSO-APM performed better than PMPSO except the multi-modal function f_5 and PSO-APM also produces higher mutation rate for all functions in this work.

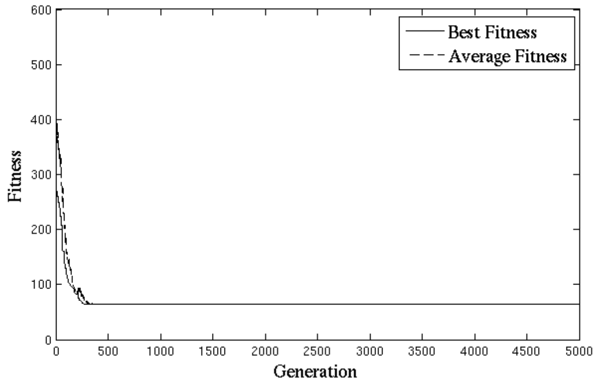

Fig. 1. Convergence graph of PSO for f_8

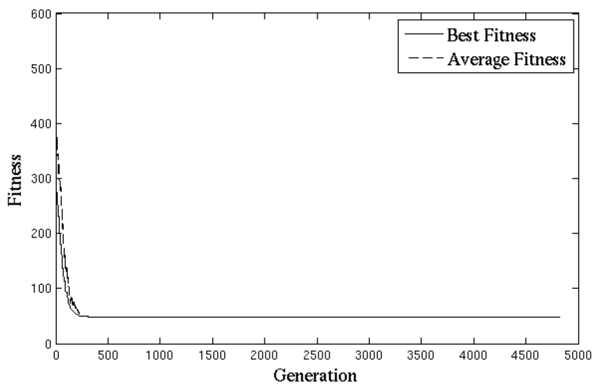

Fig. 2. Convergence graph of PSO-APM for f_8

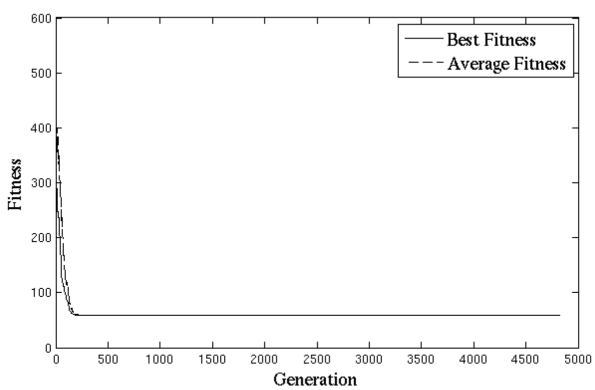

Fig. 3. Convergence graph of PMPSO for f_8

VII. CONCLUSION

In this work, we proposed adaptive polynomial mutation on global best solution in particle swarm optimization and applied on well-known benchmark functions.We also conducted our experiments with basic PSO and PMPSO.In the present investigation, we study mutation on global best particle to jump it out from local optima. PSO with adaptive polynomial mutation on global best solution produces the better results than PMPSO and PSO. Mutation Rate is also higher than that of PMPSO.But PSO-APM produces poor performance for multi-modal function f_5 and f_8. Our future works will be directed towards solving local minima problem in complex multi-modal function optimization.

REFERENCES

[1] Kennedy and R. Eberhart, *Particle Swarm Optimization*, IEEE International Conference on Neural Networks, Perth,Australia, 1995.

[2] Swagatam Das, Ajith Abraham and Amit Konar,*Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives*, Springer-Verlog, Berlin Heidelberg, Studies in Computational Intelligence(SCI) ,116, 1-38 (2008).

[3] N. Higashi and H. Iba, *Particle Swarm Optimization with Gaussian Mutation*, IEEE Swarm Intelligence Symposium, Indianapolis, 2003, pp. 72-79

[4] H. Wang, Y. Liu, C. H. Li, and S. Y. Zeng, *A hybrid particle swarm algorithm with Cauchy mutation*, In Proc. of IEEE Swarm Intelligence Symposium, 2007, pp. 356-360.

[5] C.Li,Y. Liu, A. Zhao,L. Kang and H. Wang , *A Fast Particle Swarm Algorithm with Cauchy Mutation and Natural Selection Strategy*, Springer-Verlag,LNCS 4683, 2007, pp. 334-343.

[6] A. Stacey, M. Jancic and I. Grundy,*article swarm optimization with mutation*, in Proc. Congr. Evol. Comput, 2003, pp. 14251430, doi: 10.1109/CEC.2003.1299838.

[7] Jun Tang and Xiaojuan Zhao,*Particle Swarm Optimization with Adaptive Mutation*, WASE International Conference on Information Engineering,2009

[8] Xiaoling Wu and Min Zhong, *Particle Swarm Optimization Based on Power Mutation*, ISECS International Colloquium on Computing, Communication, Control, and Management,2009

[9] Amit saha, Rituparna Datta and Kalyanmoy Deb, *Hybrid Gradient Projection based Genetic Algorithms for Constrained Optimization*, IEEE Congress on Evolutionary Computation - CEC , pp. 1-8, 2010,10.1109/CEC.2010.5586303

[10] M.M. Raghuswanshi and O.G. Kakde, *Survey on multiobjective evolutionary and real code genetic algorithms*, Complexity International, Volume 11,2005

[11] Yuelin Gao and Zihui Ren, *Adaptive Particle Swarm Optimization Algorithm With Genetic Mutation Operation*, Third International Conference on Natural Computation(ICNC 2007).

[12] JIAO Wei,LIU Guangbin and LIU Dong, *Elite Particle Swarm Optimization with Mutation*, Asia Simulation Conference - 7th International Conference on System Simulation and Scientific Computing, 2008. ICSC 2008.

[13] L. Kang, Y. Liu and S. Zeng et. al., *A Fast Particle Swarm Optimization Algorithm with Cauchy Mutation and natural Selection Strategy*, ISICA 2007, LNCS 4683, pp. 334-343,2007

[14] A.J. Nebro, *SMPSO: A New PSO-based Metaheuristic for Multi-objective Optimization*, ieee symposium on Computational intelligence in miulti-criteria decision-making, 2009

[15] M. Reyes Sierra and C. A. Coello Coello, *Improving PSO-Based Multiobjective Optimization Using Crowding, Mutation and Dominance*, In Evolutionary Multi-Criterion Optimization (EMO 2005), LNCS 3410, pages 505519, 2005.

[16] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley and Sons,2001.

[17] M. Pant, R. Thangaraj and A. Abraham, *Particle Swarm Optimization using Adaptive Mutation*, in Proc. 19th International Conference on Database and Expert Systems Application, 2008, pp. 519-523

[18] S. C. Esquivel and C. A. Coello Coello, *On the use of particle swarm optimization with multimodal functions*, in Proc. Congr. Evol.Comput., 2003, pp. 1130-1136,

[19] J. Chen, Z. Ren and X. Fan, *Particle swarm optimization with adaptive mutation and its application research in tuning of PID parameters*, in Proc. 1st International Symposium on Systems and Control in Aerospace and Astronautics, 2006, pp. 990994

[20] Y. Shi and R.C Eberhart, *A modified particle swarm optimizer*, in proc. IEEE World Congr. Comput. Intell.,1998,pp. 69-73

[21] X. Yao, Y. Liu and G. Lin, *Evolutionary programming made faster* , IEEE Transactions on Evolutionary Computation.,vol. 3,1999,pp. 82-102

TABLE I
THE 8 BENCHMARK FUNCTIONS USED IN OUR EXPERIMENTS,WHERE D IS THE DIMENSION OF THE FUNCTIONS, f_{min} IS THE MINIMUM VALUES OF THE FUNCTIONS, AND $S \subseteq R^D$ IN THE SEARCH SPACE

Test Function	D	S	f_{min}
$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	[-100,100]	0
$f_2(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	[-100,100]	0
$f_3(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{n-1}} x_i^2$	30	[-100,100]	0
$f_4(x) = \sum_{i=1}^{D} [100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2]$	30	[-100,100]	0
$f_5(x) = \sum_{i=1}^{D} -x_i * sin(\sqrt{\mid x_i \mid})$	30	[-500,500]	-12569.5
$f_6(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600,600]	0
$f_7(x) = -20 * exp(-0.2 * \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D} \sum_{i=1}^{D} cos(2\pi x_i)) + 20 + e$	30	[-32,32]	0
$f_8 = \sum_{i=1}^{D} [x_i - 10\cos(2\pi x_i) + 10]$	30	[-5.12,5.12]	0

TABLE II
FUNCTION VALUES ACHIEVED BY PSO,PSO-APM,PMPSO

Problem	**PSO**		**PSO-APM**		**PMPSO**	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
f_1	1.74e-42	9.52e-42	**1.32e-49**	**1.24e-50**	5.45e-48	2.20e-47
f_2	1.45e-006	3.82e-006	**1.33e-006**	**5.50e-049**	7.17e-007	1.57e-006
f_3	7.727236e-34	1.41e-034	**1.77e-042**	**6.237515e-042**	3.39e-042	1.86e-041
f_4	2.93e+001	3.01e+001	**2.50e+001**	**2.36e+001**	3.76e+001	3.03e+001
f_5	-7.25e+003	7.90e+002	-7.37e+003	1.00e+003	**-7.41e+003**	**6.85e+002**
f_6	8.90e-002	1.36e-001	**0.0666**	**0.0425**	7.23e-002	1.70e-001
f_7	2.91	1.21	**2.67**	**1.22**	3.52	1.44
f_8	62.5	18.1	**58.77**	**12.53**	59.3	13

TABLE III
AVERAGE MUTATION RATE FOR 30 RUNS

Problem	PSO-APM	PMPSO
f_1	1.81e-003	1.17e-004
f_2	4.22e-003	2.32e-006
f_3	1.67e-003	1.67e-004
f_4	3.28e-003	2.78e-005
f_5	1.74e-003	8.33e-005
f_6	1.89e-003	6.11e-005
f_7	1.93e-003	5.56e-005
f_8	1.48e-003	3.89e-005