An Adaptive Mutation Operator for Particle Swarm Optimization

Changhe Li
Dept of Computer Science,
University of Leicester
University Road
Leicester LE1 7RH, UK
cl160@mcs.le.ac.uk

Shengxiang Yang
Dept of Computer Science,
University of Leicester
University Road
Leicester LE1 7RH, UK
s.yang@mcs.le.ac.uk

Imtiaz Korejo
Dept of Computer Science,
University of Leicester
University Road
Leicester LE1 7RH, UK
iak5@mcs.le.ac.uk

Abstract

Particle swarm optimization (PSO) is an efficient tool for optimization and search problems. However, it is easy to be trapped into local optima due to its information sharing mechanism. Many research works have shown that mutation operators can help PSO prevent premature convergence. In this paper, several mutation operators that are based on the global best particle are investigated and compared for PSO. An adaptive mutation operator is designed. Experimental results show that these mutation operators can greatly enhance the performance of PSO. The adaptive mutation operator shows great advantages over non-adaptive mutation operators on a set of benchmark test problems.

1 Introduction

Particle swarm optimization (PSO) was first introduced by Kennedy and Eberhart in 1995 [2, 5]. There are two main models of the PSO algorithm, called *gbest* (global best) and *lbest* (local best), which are different in the way of defining particle neighborhood. Kennedy and Poli [6, 10] pointed out that the gbest model has a fast convergence speed with a higher chance getting stuck in local optima. On the contrary, the lbest model is less vulnerable to the attraction of local optima but has a slower convergence speed than *gbest*. In this paper, all the PSO algorithms studied are based on the gbest model, aiming to reduce the probability of being trapped into local optima by introducing mutation opertor and keep the advantage of fast convergence of the gbest model.

There are several major versions of the PSO algorithms. The following version modified by Shi and Eberhart [12] is used in this paper. Each particle is represented by a position and a velocity, which are updated as follows:

$$\vec{V}_i' = \omega\vec{V}_i + \eta_1 r_1(\vec{P}_i - \vec{X}_i) + \eta_2 r_2(\vec{P}_g - \vec{X}_i) \quad (1)$$

$$\vec{X}_i' = \vec{X}_i + \vec{V}_i' \quad (2)$$

where $\vec{X}_i'$ and $\vec{X}$ represent the current and previous positions of particle i, $\vec{V}_i$ and $\vec{V}_i'$ are the previous and current velocity of particle i, $\vec{P}_i$ and $\vec{P}_g$ are the best-so-far position of particle i and the best position found in the whole swarm so far respectively. $\omega \in (0, 1]$ is an inertia weight which determines how much the previous velocity is preserved, η_1 and η_2 are acceleration constants, and r_1 and r_2 are random numbers generated from the interval [0.0, 1.0].

Ratnaweera et al. [11] stated that the lack of population diversity in PSO algorithms is understood to be a factor in their convergence on local optima. Therefore, the addition of a mutation operator to PSO should enhance its global search capacity and thus improve its performance. There are mainly two type of mutation operators: one type is based on particle position [7, 8, 4, 13, 3] and the other type is based on particle velocity [11, 9, 14, 15]. The former method is by far the most common technique found in the literature. However, the work of the latter one is very few. There has been few attempt to different mutation operators, and so far no comparison of effectiveness of different mutation operators on particle velocity.

In this paper, several mutation operators that are based on the global best particle are investigated for PSO. An adaptive mutation operator is also designed for PSO. An experimental study is carried out to compare the performance of the investigated mutation operators for PSO on a set of benchmark test problems.

2 An Adaptive Mutation Operator for PSO

Different mutation operators can be used to help PSO jump out of local optima. However, a mu-

tation operator may be more effective than other ones on a certain type of problems and may be worse on another type of problems. In fact, it is the same even for a specific problem at different stage of the optimization process. That is, the best mutation results can not be achieved by a single mutation operator, instead several mutation operators may have to be applied at different stages for the best performance. This paper designs a mutation operator that can adaptively select the most suitable mutation operator for different problems. Before presenting the adaptive mutation operator, three mutation operators designed for the global best particles are described as follows.

2.1 Three Mutation Operators

A. Cauchy mutation operator

$$\vec{V}_g' = \vec{V}_g exp(\delta) \tag{3}$$

$$\vec{X}_g' = \vec{X}_g + \vec{V}_g' \delta_g \tag{4}$$

where $\vec{X}_g$ and $\vec{V}_g$ represent the position and velocity of the global best particle. δ and δ_g denote Cauchy random numbers with the scale parameter of 1.

B. Gaussian mutation operator

$$\vec{V}_g' = \vec{V}_g exp(N) \tag{5}$$

$$\vec{X}_g' = \vec{X}_g + \vec{V}_g' N_g \tag{6}$$

where $\vec{X}_g$ and $\vec{V}_g$ represent the position and velocity of global best particle. N and N_g are Gaussian distribution numbers with the mean 0 and the variance 1.

C. Levy mutation operator

$$\vec{V}_g' = \vec{V}_g exp(L(\alpha)) \tag{7}$$

$$\vec{X}_g' = \vec{X}_g + \vec{V}_g' L_g(\alpha), \tag{8}$$

where $L(\alpha)$ and $L_g(\alpha)$ are random numbers generated from the Levy distribution with a parameter α. In this paper, α is set to 1.3.

2.2 The Adaptive Mutation Operator

The proposed adaptive mutation operator uses the three mutation operators described above. All mutation operators have an equal initial selection ratio with 1/3. Each mutation operator is applied according to its selection ratio and its offspring fitness is evaluated. The mutation operators that result in higher fitness values of offspring have their selection ratios increased. The mutation operators that result in lower fitness values of offspring have their selection ratios decreased. Gradually, the most suitable mutation operator will be chosen automatically and control all the mutation behavior in the whole swarm. Without lose of generality, we discuss the minimization optimization problems in this paper.

First, some definitions are given below: The progress value $prog_i(t)$ of operator i at generation t is defined as follows:

$$prog_i(t) = \sum_{j=1}^{M_i} f(p_j^i(t)) - \min\left(f(p_j^i(t)), f(c_j^i(t))\right), \tag{9}$$

where $p_j^i(t)$ and $c_j^i(t)$ denote a parent and its child produced by mutation operator i at generation t and M_i is the number of particles that select mutation operator i to mutate.

The reward value $reward_i(t)$ of operator i at generation t is defined as follows:

$$reward_i(t) = exp(\frac{prog_i(t)}{\sum_{j=1}^{N} prog_j(t)}\alpha + \frac{s_i}{M_i}(1-\alpha)) + c_i p_i(t) - 1 \tag{10}$$

where s_i is the number of particles whose children have a better fitness than themselves after being mutated by mutation operator i, $p_i(t)$ is the selection ratio of mutation operator i at generation t, α is a random weight between $(0, 1)$, N is the number of mutation operators, and c_i is a penalty factor for mutation operator i, which is defined as follows:

$$c_i = \begin{cases} 0.9, & \text{if } s_i = 0 \text{ and } p_i(t) = \max_{j=1}^{N}(p_j(t)) \\ 1, & \text{otherwise} \end{cases} \tag{11}$$

if the previous best operator has no contribution at current generation, then the selection ratio of the current best operator will decrease.

With the above definitions, the selection ratio of mutation operator i is updated according to the following equation:

$$p_i(t+1) = \frac{reward_i(t)}{\sum_{j=1}^{N} reward_j(t)}(1 - N*\gamma) + \gamma, \tag{12}$$

where γ is the minimum selection ratio for each mutation operator, which is set 0.01 for all the experiments in this paper. This selection ratio update equation considers four factors: the progress value, the ratio of successful mutations, previous selection ratio, and the minimum selection ratio. Another important parameter for the adaptive mutation operator is the frequency of

updating the selection ratios of mutation operators. That is, the selection ratio of each mutation operator can be updated at a fixed frequency, e.g., every U_f generations, instead of every generation.

The framework of the PSO algorithm with one of the three mutation operators described above is given as follows:

Step 1: Generate the initial particles by randomly generating the position and velocity for each particle.

Step 2: Evaluate the fitness of each particle.

Step 3: For each particle i, if its fitness is smaller than the fitness of its previous best position ($\vec{P}_i$), update $\vec{P}_i$.

Step 4: For each particle, if its fitness is smaller than the fitness of the best position ($\vec{p}_g$) of all particles, update $\vec{P}_g$.

Step 5: Update each particle according to Eqs. (1) and (2).

Step 6: Mutate $\vec{P}_g$ according to one of the three mutation operators for T times (T is the local search size for the global best particle).

Step 7: Compare the best one $\vec{P}_g^*$ of the mutants with $\vec{P}_g$ and select the better one as the new global best particle.

Step 8: Stop if the stop criterion is satisfied; otherwise, go to Step 3.

The PSO with the adaptive mutation operator differs from the above PSO algorithms in Step 6. At Step 6, we select one of the three mutation operators according to their selection ratios to mutate $\vec{P}_g$ for T times and then update the selection ratio for each mutation operator according to Eq. (18)at a fixed frequency, e.g., every U_f generations.

3 Experimental Study

Seven benchmark functions ($f_1 - f_7$) are used in this paper. Functions f_1 is unimodal functions while $f_2 - f_7$ have many local optima. Table 1 gives the details of these functions. The proposed algorithm are compared with another algorithm called FEP[16, 17] on all test problems.

Algorithm parameters are set as follows: the acceleration constants $\eta_1 = \eta_2 = 1.496180$ and the inertia weight $\omega = 0.729844$ as suggested by den Bergh [1]. In order to have the same number of function evaluations, the population size is 50 and $T = 10$ for PSO for mutation operators. In PSO and FEP, the population size is 60, the tournament size is 6 for selection and initial standard deviation is 3.0 in FEP algorithm [16]. For the adaptive mutation operator, the initial selection ratio is 1/3 and the minimum selection ratio γ is 0.01 for each mutation operator, and the update frequency U_f is set to 5. For all algorithms, we run 50 times independently till generation 2000 for each test problem.

3.1 Performance Comparison

The average results of six algorithms for the test problems are summarized in Table 2. Table 3 also shows the statistical comparison of the adaptive mutation operator over other five algorithms, using the two-tailed T-test with 98 degrees of freedom at a 0.05 level of significance. In Table 3, the performance difference is significant if the absolute value of the T-test result is greater than 1.984.

From Table 2 and Table 3, it can be seen that the algorithms with mutation operator perform better than PSO for most problems. All the four algorithms with mutation operators present much better results than PSO on f_1, f_2, f_6 and f_7. Especially for multimodal functions, algorithms with mutation perform obviously better than PSO. FEP performs better than PSO on f_6 and f_7, but worse than PSO on the other problems. The mean values obtained by Levy and Adaptive are much better than FEP on all test problems.

As expected, adaption mutation shows its advantages and presents at least the second best result among all the mutation algorithms on all test problems. To our surprise, the adaptive mutation achieves the best results among all the algorithms on function f_2 and f_4. In this point of view, adaptive mutation can probably be regarded as the one with the average best performance for PSO on all test problems. Fig. 1 shows the evolutionary process of the average global best particle of PSO, Adaptive, and FEP. Due to page limitation, the results of function f_3, f_5 and f_6 are not presented. Adaptive mutation gives the fastest convergence rate on all test functions.

Fig. 2 presents the results of selection ratio of Cauchy, Gaussian, and Levy at generation 2000 for 50 runs in the adaptive mutation.Due to page limitation, the results of function f_3, f_5 and f_6 are not presented. We can see that the frequencies of three mutation operators being selected are quite different for each test problem. The selection ratio of Levy for function f_1, f_2 and f_4

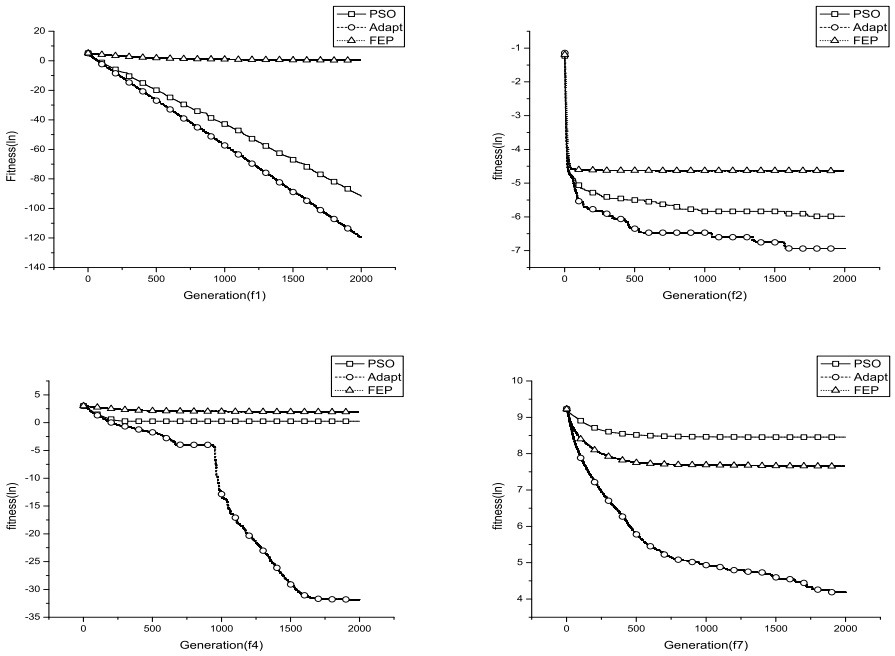

Figure 1: Evolution process of the average best particle of PSO, Adaptive mutation and FEP.

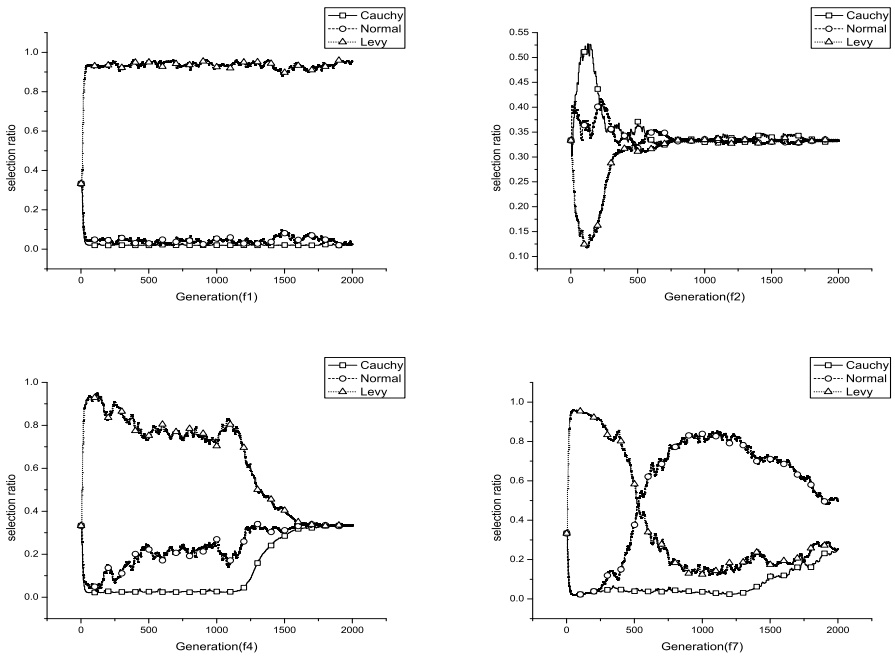

Figure 2: The average selection ratio of Cauchy,Gaussian and Levy in the adaptive mutation.

Table 1: Details of test functions, where n is the dimension of the function, $f_{min}$is the minimum value of the function, and $S \in R_n$

Test function	n	S	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$(-5.12, 5.12)$	0
$f_2(x) = \frac{(sin^2\sqrt{x_1^2+x_2^2})-0.5}{(1.0+0.001(x_1^2+x_2^2))^2} + 0.5$	30	(-100,100)	0
$f_3(x) = \frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i-100}{\sqrt{i}}) + 1$	30	(-300,300)	0
$f_4(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) -$ $\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	30	(-30,30)	0
$f_5(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$	30	(-5.12,5.12)	0
$f_6(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	(-500,500)	-12569.5
$f_7(x) = 418.9829 \cdot n + \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	(-500, 500)	0

Table 2: Comparison of PSO, Cauchy, Gaussian, Levy, Adaptive and FEP. The results are mean best function values found at generation 2000 over 50 runs

Test function	PSO	Cauchy	Gaussian	Levy	Adaptive	FEP
f_1	1.24e-40	1.17e-41	2.45e-43	1.40e-53	6.51e-53	1.424
f_2	0.00252	0.001748	0.00136	0.00116	0.00097	0.00968
f_3	0.01865	0.01062	0.01877	0.01454	0.01298	1.2003
f_4	1.28691	1.586e-14	1.19315	1.486e-14	1.45e-014	6.84345
f_5	42.0469	33.1918	46.564	27.1624	28.0578	69.283
f_6	-7804.39	-12523.7	-8249.72	-12569.5	-12533.9	-10522.4
f_7	4700.71	101.543	4239.15	2.36915	65.9316	2099.74

Table 3: The T-test results between adaptive mutation and the other five algorithms

Test function	PSO	Cauchy	Gaussian	Levy	FEP
f_1	-1.75169	-2.73526	-1.84775	2.33678	-5.44373
f_2	-2.10759	-1.14885	-0.61043	-0.31655	-8.68015
f_3	-1.68669	0.862596	-1.65334	-0.526	-8.06837
f_4	-10.4887	-1.28531	-10.6313	-0.29368	-16.0647
f_5	-7.65198	-2.62769	-8.36267	0.49368	-14.1893
f_6	-74.0754	-0.70909	-55.7929	4.20043	-28.7369
f_7	-62.7898	-1.42823	-52.0554	4.09011	-28.9074

keeps at high level before population converges, it dominates the mutation behaviour during this period. However, the selection ratio of Gaussian surpasses Levy after 500 generation for function f_7. The selection ratio of Cauchy almost keeps at a very low level during the optimization run for all test problems.

The experimental results validate our expectation that one mutation operator may perform better than other ones on a certain type of problems but worse on another type of problems. Even for a particular problem, different mutation operators are needed at different evolving stage to achieve a good result.

4 Conclusions

This paper investigates three mutation operators that are based on the particle velocity for PSO. A technique of how to design an adaptive mutation operator is presented in this paper and an adaptive mutation operator is proposed for PSO. By introducing mutation, PSO greatly improves its global search capability without losing its fast convergence property. Though different mutation operators give a different performance on different test problems, the adaptive mutation operator shows a balanced performance on all test problems and it gives the best performance on some problems. That is, the adaptive mutation is more robust than any other mutation operators investigated in this paper. Hence,

the integration of adaptive mutation operators is a promising work for improving the performance of PSO.

Acknowledgement

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/E060722/1.

References

[1] F. van den Bergh. An Analysis of Particle Swarm Optimizers. *PhD thesis*, Department of Computer Science, University of Pretoria, South Africa, 2002.

[2] R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. *Proc. of the 6th Int. Symp. on Micro Machine and Human Science*, pp. 39-43, 1995.

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

[4] N. Higashi and H. Iba. Particle swarm optimization with Gaussian mutation, *Proc. of the 2003 IEEE Swarm Intelligence Symphosium*, pp. 72-79, 2003.

[5] J. Kennedy and R. C. Eberhart. Particle Swarm Optimization. *Proc. of the 1995 IEEE Int. Conf. on Neural Networks*. pp. 1942-1948, 1995.

[6] J. Kennedy and R. C. Eberhart. Swarm Intelligence. *Morgan Kaufmann Publishers*, 2001.

[7] R. A. Krohling. Gaussian particle swarm with jumps. *Proc. of the 2005 IEEE Congr. on Evol. Comput.*, pp. 1226-1231, 2005.

[8] R. A. Krohling and L. dos Santos Coelho. PSO-E: Particle swarm with exponential distribution. *Proc. of the 2006 IEEE Congr. on Evol. Comput.*, pp. 1428-1433, 2006.

[9] C. Li, Y. Liu, L. Kang, and A. Zhou. A Fast Particle Swarm Optimization Algorithm with Cauchy Mutation and Natural Selection Strategy. *ISICA2007, LNCS4683*, pp. 334-343, 2007.

[10] R. Poli, J. Kennedy, and T. Blackwell. Particle Swarm Optimization: An overview. *Swarm Intelligence* , **1**(1): 33-58, 2007.

[11] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Trans. on Evol. Comput.*, **8**(3): 240-255, 2004.

[12] Y. Shi and R. C. Eberhart. A Modified Particle Swarm Optimizer. *Proc. of the IEEE Int. Conf. on Evol. Comput.*, pp. 69-73, 1998.

[13] A. Stacey, M. Jancic, and I. Grundy. Particle swarm optimization with mutation, *Proc. of the 2003 IEEE Congr. on Evol. Comput.*, pp. 1425-1430,2003.

[14] H. Wang, Y. Liu, C. Li, and S. Zeng, A Hybrid Particle Swarm Algorithm with Cauchy Mutation, *Proc.of the 2007 IEEE Swarm Intelligence Symposium*, 2007.

[15] H. Wang, Y. Liu, S. Zeng, and C. Li. Opposition-based Particle Swarm Algorithm with Cauchy Mutation. *Proc. of the 2007 IEEE Congr. on Evol. Comput.*, 2007.

[16] X. Yao and Y. Liu. Fast evolutionary programming, *Proc. of the Fifth Annual Conference on Evolutionary Programming (EP'96)*, pp. 451-460, 1996.

[17] X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster. *IEEE Trans. on Evol. Comput.*, **3**(1): 82-102, 1999.