Elite Particle Swarm Optimization with Mutation

JIAO Wei, LIU Guangbin, and LIU Dong

***Abstract*—An improved algorithm for Particle Swarm Optimization (PSO) named Elite Particle Swarm Optimization with Mutation (EPSOM) is proposed in this paper. Elite particles and bad particles are distinguished from the swarm after some initial iteration steps. Bad particles are replaced with the same number of elite particles, and a new swarm is generated. To avoid losing diversity of the swarm and to decrease the risk of trapping in local optimum, mutation operation is introduced in evolution process. The results of several simulations for different benchmark functions illustrate that EPSOM algorithm has the ability of local exploitation and global exploration. EPSOM algorithm outperforms the Linearly Decreasing Weight Particle Swarm Optimization (LDW-PSO) and Random Mutation Particle Swarm Optimization (RM-PSO) in respects of calculation accuracy and convergence.**

I. Introduction

Particle Swarm Optimization (PSO) is an evolutionary computation technique inspired by social behavior observable in nature, such as flocks of birds, proposed by R. Eberhart and J. Kennedy in 1995 [1], [2]. The behavior of birds' seeking food is mimicked by PSO to solve the optimal problems. A flock of birds is randomly initialized in the solution space, every bird is called "particle". The fundament for the development of PSO is hypothesis that a potential solution to an optimization problem is treated as a bird without quality and volume, flying through a *D*-dimensional space, adjusting its position in search space according to its own experience and that of its neighbors.

Compared with other evolutionary algorithms, such as genetic algorithm (GA), the advantages of PSO are that PSO is easy in implementation and there are few parameters to adjust. Moreover, PSO has a special characteristic of memory, which redounds to tracing current searching state dynamically. So it has a better stability and global convergence, and is fit for solving complex optimal problems. PSO has already been applied successfully in many application areas, including electric power system [3], [4], robot [5], automatic system [6], [7], image processing techniques [8], [9] and other areas. PSO also has obvious disadvantages even though. Firstly, each particle in PSO often continues to move roughly in the same direction (towards the global best position) until there is a change in the global best position. This leads to all particles are attracted to local optima whose fitness maybe low. Secondly, the search space usually is very large to avoid being trapped in local optima, so that PSO wastes a considerable amount of computational effort by visiting states of poor fitness values. We present here a new approach to offset these limitations by introducing Elite Particle Swarm Optimization with Mutation (EPSOM).

The rest of the paper is organized as follows. Section II describes the Standard Particle Swarm Optimization (SPSO) along with its variants. Section III expatiates on the EPSOM algorithm and gives the main calculation flow of the algorithm. The benchmark functions are used for testing the algorithm's performance, and the experimental settings are analyzed in Section IV. Finally, the conclusion is presented in Section V.

Manuscript received June 20, 2008.

JIAO Wei is with the 303 room of Xi'an research institute of hi-tech, Shaanxi, 710025 P. R. China. (tel: +86-29-84744111; fax: +86-29-84744640 e-mail: jiaowei1981@yahoo.com.cn).

LIU Guangbin is with Xi'an research institute of hi-tech, Shaanxi, 710025 P. R. China.

LIU Dong is with the 303 room of Xi'an research institute of hi-tech, Shaanxi, 710025 P. R. China. (e-mail: ld_yzy@yahoo.com.cn).

II. Review of Standard PSO

Assume that the search space is *D*-dimensional and a particle swarm consists of *m* particles. The *i*th particle is represented as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$ in *D*-dimensional space, where $x_{id} \in [l_d, u_d]$, $d \in [1, D]$, l_d, u_d are the lower and upper bounds of the *d*th dimension. The velocity of *i*th particle is represented as $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$, which is clamped to a maximum value $V_{\max}$, specified by users. The particles' position and velocity updating rule is given by:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 rand_1[p_{id}(t) - x_{id}(t)] + c_2 rand_2[p_{gd}(t) - x_{id}(t)] \quad (1)$$

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

where $i = 1, 2, \cdots, m$, $d = 1, 2, \cdots, D$. c_1 and c_2 are non-negative constants, generally assigned to 2.0. $rand_1$ and $rand_2$ are two independent random numbers uniformly distributed in the range of [0, 1]. $P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$ is the previous best position of the *i*th particle. $P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$ is the best previous position among all the particles in iteration *t* (memorized in a common repository). ω, named the inertia weight, often decreases linearly from 0.9 to 0.4 during a run. In general, ω is set according to the following equation:

$$\omega(t) = \omega_{final} + \frac{Iter_{\max} - Iter}{Iter_{\max}} \times \left(\omega_{initial} - \omega_{final}\right) \quad (3)$$

where $\omega_{initial}$ and ω_{final} are the initial weight and final weight, respectively, $Iter_{\max}$ is the maximum number of allowable iterations, and $Iter$ is the current iteration number. This

978-1-4244-1787-2/08/$25.00 ©

method, presented by Y. Shi and R. Eberhart [10], is called Standard Particle Swarm Optimization (SPSO) or Linearly Decreasing Weight Particle Swarm Optimization (LDW-PSO).

III. ELITE PSO WITH MUTATION

In order to avoid wasting a considerable amount of computational effort by visiting states of poor fitness values and take full advantage of outstanding particles in the whole swarm, the "elite strategy" is adopted in the paper. With the intention of finding outstanding particles, each particle will be aligned according to its fitness value after some initial iteration steps. The particles, whose fitness values in the sequence are better, named "elite particles". On the contrary, the particles, whose fitness values in the sequence are worse, named "bad particles". Elite strategy is a method that bad particles will be substitute by elite particles after some steps of initial iteration, so a new particle swarm is generated. An optimum is searched continuously in the searching space according to the new particle swarm. Keep in mind that the number of "elite particles" and "bad particles" must be correspondent.

Suppose that the optimum searching is executed directly according to aforementioned method, the diversity of particle swarm will be decreased sharply during the earlier stage of evolutional iteration. Consequently, it is easy to be trapped in local optimum's basin during the earlier phase. Inspired by GA, mutation operator is introduced here to increase the diversity of particle swarm and decrease the risk of plunging into local optimum, i.e., the global best individual may be mutated to generate a new particle. If the new particle's fitness value is better than the current best fitness value, this new particle will substitute for foregoing particle, taken as the new global best position. Because the new global best individual attracts all members of the swarm, it is possible to lead the swarm away from a current location. This improved algorithm is named Elite Particle Swarm with Mutation (EPSOM) here. The mutation operation is showed as the following equation:

$$P_g' = P_g(1 + 0.5\eta) \tag{4}$$

where η is a random number normally distributed in the range of [0, 1]. The mutation probability is suggested as a random number in the range of [0.1, 0.3].

EPSOM algorithm can be summarized in the following steps:

Step1: Initialize the state of each particle. Set P_i as the best current position of each particle, set P_g as the best current of whole swarm.

Step2: If the stopping condition is satisfied, go to *Step8*. Otherwise, go to *Step3*.

Step3: Each particle is executed K iterative steps as follows:

Step3.1: Update the velocity and position of particle according to equation (1) ~ equation (3).

Step3.2: If $V_i > V_{\max}$ or $V_i < V_{\min}$, set $V_i = V_{\max}$ or $V_i = V_{\min}$; if $X_i > X_{\max}$ or $X_i < X_{\min}$, set $X_i = X_{\max} \times rand$ or $X_i = X_{\min} \times rand$, where *rand* stands for a random number uniformly distributed in the range of [0, 1].

Step3.3: If necessary, update and store the individual best position and individual best fitness of each particle, update and store the global best position and global best fitness of whole swarm.

Step4: Align the particles according to their corresponding fitness values. Let the best n elite particles replace of the worst n particles.

Step5: Execute the updating of the new swarm as the *Step3.1~Step3.3*.

Step6: Generate a random number $r \in [0,1]$, if $r < p_m$, execute mutation operation according to equation (4). Otherwise, go to *Step7*.

Step7: If stopping condition is not satisfied go to *Step5*. Otherwise, go to *Step8*.

Step8: Give the global best position P_g and global best fitness of whole swarm.

IV. SIMULATIONS

A. Benchmark Functions

A set of well-known benchmarks, which are commonly used in literatures, were used to evaluate the performance both in terms of solution quality and convergence rate of the proposed algorithms. The benchmark problems used are a set of three non-linear functions, as minimization problems, which present different difficulties to the algorithms to be evaluated. These benchmark functions are as follows:

The first test function is the Rosenbrock function:

$$f_1(x) = \sum_{i=1}^{n} 100(x_{i+1}^2 - x_i) + (1 - x_i)^2 \quad -100 \le x_i \le 100 \tag{5}$$

The second test function is the Rastrigrin function:

$$f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10) \quad -10 \le x_i \le 10 \tag{6}$$

The third test function is the Griewank function:

$$f_3(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1 \quad -600 \le x_i \le 600 \tag{7}$$

Rosenbrock is a unimodal function, which has a global minimum of 0 at the point (1…1), it is usually used to test the local exploitation ability of algorithms. Rastrigrin and Griewank are multimodal functions that have many local minima. Rastrigrin is commonly used to test the global exploration ability and avoid trapping in local optimum. Rastrigrin and Griewank both have a global minimum of 0 when the vector is (0…0).

B. Experimental Settings and Analysis

EPSOM algorithm introduces two presetting parameters: one is the percentage of elite particles with respect to the whole swarm signed as *Ep* here; the other is initial iteration steps signed as *K*. In addition to, mutation probability p_m is also a presetting parameter. *Ep* and *K* influence the performance of EPSOM directly. If *Ep* is too large, the

diversity of particle swarm is badly destroyed and leads to trapping in local optima. Otherwise, if *Ep* is too small, the elite particles' active influence cannot be fully exerted. If the value of *K* is too small, elite particles cannot be effectively distinguished from bad particles. On the contrary, if *K* is too large, elite particles found from the swarm are wondrously possible particles whose fitness values are local optima. So it is necessary to study relationships among *Ep*, *K* and success rate.

The population size of swarm is set to 30, the number of the maximum permissive iteration is set to 2000, initial iteration steps *K* is respectively 5,10,15,20,40,60,80 and 100. *Ep* is respectively 1/2, 1/3, 1/5 and 1/10.Take 30-dimisional Rosenbrock function as an example, the relationship between *Ep* and success rate (acceptable solution is 100) with different *K* is plotted in Fig.1 after 100 trial runs respectively. As shown in Fig.1 *Ep* = 1/2 the success rates are higher than other cases on the whole. Set *Ep* = 1/2, other settings are same as foresaid, 100 runs of the algorithm were performed respectively. The relationship between *K* and success rate with different test functions is shown in Fig.2. The acceptable solutions of Rosenbrock, Rastrigrin and Griewank are 100,70,0.01 respectively. From Fig.2 we can see that the higher success rates can be obtained in all test functions when *K* is set during 5 to 15. Note that, the two trials above are not taken mutation operator into account in order to study relationships between *Ep*, *K* and success rate separately.

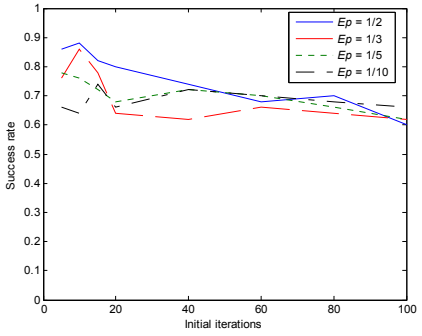


Fig. 1. Relationships between *Ep* and success rates with different *K*

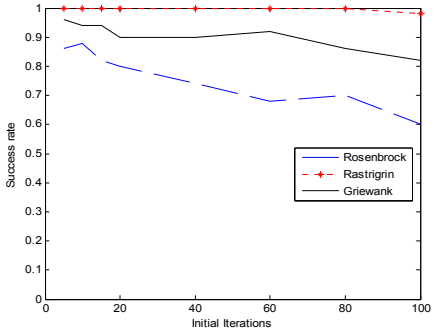


Fig. 2. Relationships between *K* and success rates with different test functions

C. *Experimental Results and Discussions*

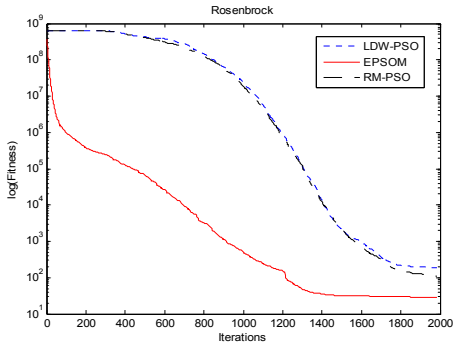
According to aforementioned analysis, set $E_p = 1/2$ and $K = 10$ in EPSOM algorithm. Mutation operation is executed as equation (4). The population size is only 20, inertial weight ω decreases linearly from 0.9 to 0.4 during a run, and the largest permissive iteration steps is set to 2000, V_{max} is set equal to X_{max} and V_{min} is set equal to X_{min}. EPSOM was compared with LDW-PSO and Random Mutation PSO (RM-PSO) at the same initialization condition, where mutation probability is 0.05 in RM-PSO. The search space and initialization range for each test function were listed in Table I. Table II lists the mean fitness values of the best solutions achieved by three algorithms on Rosenbrock, Rastrigrin and Griewank functions with each experimental setting. The results of each algorithm were averaged over 30 trial runs respectively. Fig.3 shows the evolution of logarithmic average fitness of 30-dimensional test functions for LDW-PSO, RM-PSO and EPSOM.

TABLE I
SEARCH SPACE AND INITIALIZATION RANGE

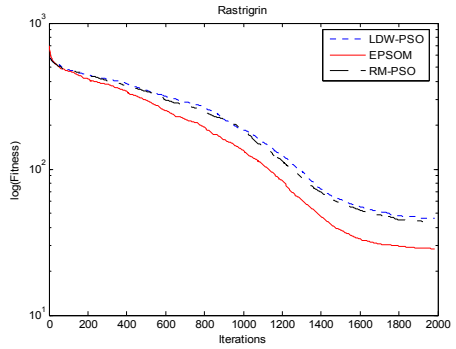
Function	Search space	Initialization ranges
Rosenbrock	$-100 \le x_i \le 100$	$15 \le x_i \le 30$
Rastrigrin	$-10 \le x_i \le 10$	$2.56 \le x_i \le 5.12$
Griewank	$-600 \le x_i \le 600$	$300 \le x_i \le 600$

TABLE II
THE MEAN BEST FITNESS VALUES FOR THE BENCHMARK FUNCTIONS

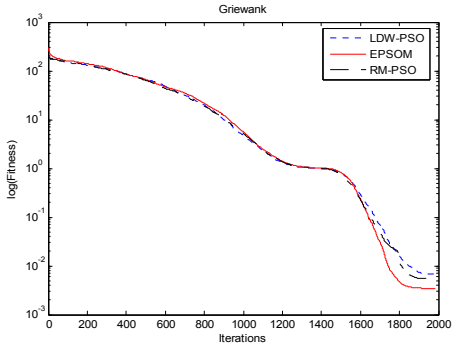
Function	Dim	LDW-PSO	RM-PSO	EPSOM
Rosenbrock	10	14.0282	12.3792	8.9516
	20	73.9728	38.4311	18.9174
	30	183.9424	112.2603	28.1556
Rastrigrin	10	4.0462	4.0793	3.6150
	20	21.7190	22.0556	16.0358
	30	46.1274	43.6590	28.5360
Griewank	10	0.0847	0.0840	0.0788
	20	0.0196	0.0201	0.0169
	30	0.0067	0.0054	0.0035



(a)



(b)



(c)

Fig. 3. Performance on 30-dimensional functions: (a) Rosenbrock, (b) Rastrigrin and (c) Griewank.

The experimental results indicate that EPSOM algorithm exhibits good performance. It outperforms LDW-PSO and RM-PSO on all mentioned benchmark problems, especially for Rosenbrock and Rastrigrin function, i.e., EPSOM algorithm has not only good local exploitation ability but also favorable global exploration ability, meanwhile, the evolutionary rate of convergence is improved.

V. Conclusion

EPSOM algorithm improves the individual quality of the swarm and accelerates the convergence. Meanwhile, mutation operation which is employed in this new algorithm is to guarantee the diversity of the swarm and to decrease the risk of plunging into local optimum. E_p and K, which are presetting parameters, influence the algorithm's performance directly. The relationship between E_p, K and success rate is analysed, and the optimal values of the two parameters are given in the paper. The experimental results indicate that EPSOM algorithm is superior to LDW-PSO and RM-PSO for the three benchmark test functions.

References

[1] J. Kennedy and R. Eberhart. "Particle swarm optimization" *Proceeding of IEEE International Conference on Neural Networks (ICNN'95)*, vol. 4, Perth, Western Australia, IEEE, 1995, pp. 1942–1947.

[2] R. Eberhart. "A new optimizer using particle swarm theory", *Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39-43.

[3] Wen Zhang and Yutian Liu. "Adaptive Particle Swarm Optimization for Reactive Power and Voltage Control in Power Systems", *Lecture Notes in Computer Science, LNCS*, 2005, pp. 449-452.

[4] Jiang Chuanwen and Etorre Bompard. "A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization", *Mathematics and Computers in Simulation*, vol. 68, 2005, pp. 57-65.

[5] Y angmin Li and Xin Chen. "Mobile Robot Navigation Using Particle Swarm Optimization and Adaptive NN", *Lecture Notes in Computer Science*, *LNCS*, 2005, pp. 628-631.

[6] Chwen-Tzeng Su and Jui-Tsung Wong. "Designing MIMO controller by neuro-traveling particle swarm optimizer approach", *Expert System with Applications*, vol. 32, 2007, pp. 848-855.

[7] V. Mukherjee and S. P. Ghoshal. "Intelligent particle swarm optimized fuzzy PID controller for AVR system", *Electric Power Systems Research*, vol. 77, 2007, pp. 1689-1698.

[8] Yifeng Niu and Lincheng Shen. "An Adaptive Multi-objective Particle Swarm Optimization for Color Image Fusion", *Lecture Notes in Computer Science*, *LNCS*, 2006, pp. 473-480.

[9] Wensheng Yi, Min Yao and Zhiwei Jiang. "Fuzzy Particle Swarm Optimization Clustering and Its Application to Image Clustering", *Lecture Notes in Computer Science*, *LNCS*, 2006, pp. 459-467.

[10] Y. Shi and R. Eberhart. "A modified particle swarm optimizer", *Proceedings of the IEEE Conference on Evolutionary Computation*, Singapore, 1998, pp. 69–73.