Particle Swarm Optimization Using Sobol Mutation

Millie Pant[1], Radha Thangaraj[1], Ved Pal Singh[1] and Ajith Abraham[2]
[1]*Department. of Paper Technology, IIT Roorkee, India*
[2]*Center of Excellence for Quantifiable Quality of Service,*
Norwegian University of Science and Technology, Norway
millifpt@iitr.ernet.in, t.radha@ieee.org, singhfpt@iitr.ernet.in, ajith.abraham@ieee.org

Abstract

In this paper, we present a new mutation operator called the Systematic Mutation (SM) operator for enhancing the performance of Basic Particle Swarm Optimization (BPSO) algorithm. The SM operator unlike most of its contemporary mutation operators do not use the random probability distribution for perturbing the swarm population, but uses a quasi random Sobol sequence to find new solution vectors in the search domain. The comparison of SM-PSO is made with BPSO and some other variants of PSO. The empirical results show that SM operator significantly improves the performance of PSO.

1. Introduction

Particle Swarm Optimization (PSO) is relatively a newer addition to a class of population based search technique for solving numerical optimization problems. Metaphorically, PSO imitates the collective and cooperative behavior of species moving in groups. Some classic examples being a swarm of birds, school of fish, cooperative behavior of ants and bees etc.

In original PSO, developed by Kennedy and Eberhart in 1995 [1], each particle adjusts its position in the search space from time to time according to the flying experience of its own and of its neighbors (or colleagues). The particles or members of the swarm fly through a multidimensional search space looking for a potential solution. Researchers have shown that although PSO finds solutions much faster than most of the contemporary search techniques like Evolutionary and Genetic Algorithms, it usually do not improve the quality of solutions as the number of iterations increase and thus becomes a victim of premature convergence resulting in a suboptimal solution. This draw back of PSO is due to the lack of diversity, which forces the swarm particles to converge to the global optimum found so far (after a certain number of iterations), which may not even be a local optimum. Thus without an effective diversity enhancing mechanism the optimization algorithm/ technique is not able to efficiently explore the search space.

One of the methods for maintaining the diversity of the population is inclusion of the concept of mutation (a phenomenon borrowed from Evolutionary Algorithms). Most of the modern mutation operators defined in literature make use of random probability distribution (for example Gaussian mutation [2], Cauchy mutation [3] etc). In the present work, instead of using the random probability distribution, we have defined a SM operator which uses quasi random (Sobol) sequence mainly because quasi random sequences cover the search domain more evenly in comparison to the random probability distributions, thereby increasing the chances of finding a better solution. The SM operator defined in this paper is applied to two versions of BPSO called SM-PSO1 and SM-PSO2. In SM-PSO1, mutation is applied to the global best (gbest) particle, where as in SM-PSO2, the worst particle of the swarm is mutated.

The remaining organization of the paper is as follows: section 2 gives a brief review of Quasi Random Sequences (QRS) and Sobol Sequence. Section 3 describes the Basic Particle Swarm Optimization. In Section 4, we give the proposed algorithms, Section 5, gives the experimental settings and numerical results of some selected unconstrained benchmark problems. The paper finally concludes with Section 6.

2. Quasi Random Sequences (QRS)

QRS or low discrepancy sequences are less random than pseudorandom number sequences, but are more useful for computational methods, which depend on the generation of random numbers. Some of these tasks involve approximation of integrals in higher dimensions, simulation and global optimization. Some well known QRS are: Vander Corput, Sobol, Faure and Halton. These sequences have been applied to initialize the swarm and the numerical results show a marked improvement over the traditional BPSO, which uses uniformly, distributed random numbers [4], [5].

QRS are said to be better than pseudo random sequences, because of their ability to cover the search space more evenly in comparison to pseudo random sequences (see Figures 1 and 2).

2.1. Discrepancy of a Sequence

Mathematically, discrepancy of a sequence is the measure of its uniformity. It is computed by comparing the actual number of sample points in a given volume of a multi-dimensional space with the number of sample points that should be there assuming a uniform distribution defined.

For a given set of points $x^1, x^2, ..., x^N \in I^S$ and a subset $G \subset I^S$, define a counting function $S_N(G)$ as the number of points $x^i \in G$. For each $x = (x_1, x_2, ...x_S) \in I^S$, let G_x be the rectangular S dimensional region, such that $G_x = [0,x_1)$ x $[0,x_2)$ x...x$[0,x_S)$, with volume $x_1x_2…x_N$. Then the discrepancy of points is given by $D^*_N(x^1, x^2, x^3 ....x^N) = \text{Sup} \left| S_N(G_x) - N x_1 x_2 ... x_S \right|, x \in I^S$.

2.2. Discrepancy of a Sequence

The construction of the Sobol sequence [6] uses linear recurrence relations over the finite field, F2, where F2 = {0, 1}. Let the binary expansion of the non-negative integer n be given by $n = n_1 2^0 + n_2 2^1 + ... + n_w 2^{w-1}$. Then the nth element of the jth dimension of the Sobol Sequence, $X_n^{(j)}$, can be generated by:

$$X_n^{(j)} = n_1 v_1^{(j)} \oplus n_2 v_2^{(j)} \oplus ...... \oplus n_w v_w^{(j)}$$

where $v_i^{(j)}$ is a binary fraction called the ith direction number in the jth dimension.

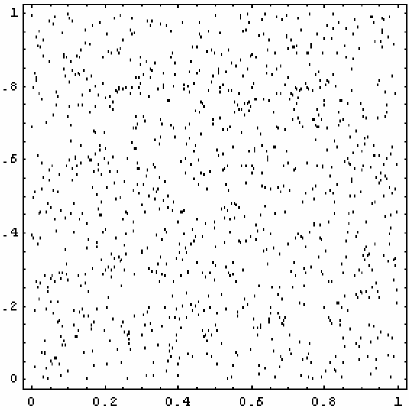


Figure 1 Sample points of pseudo random sequence

These direction numbers are generated by the following q-term recurrence relation:

$$v_i^{(j)} = a_1 v_{i-1}^{(j)} \oplus a_2 v_{i-2}^{(j)} \oplus ... \oplus a_q v_{i-q+1}^{(j)} \oplus v_{i-q}^{(j)} \oplus (v_{i-q}^{(j)} / 2^q)$$

We have i > q, and the bit a_i, comes from the coefficients of a degree-q primitive polynomial over F2.

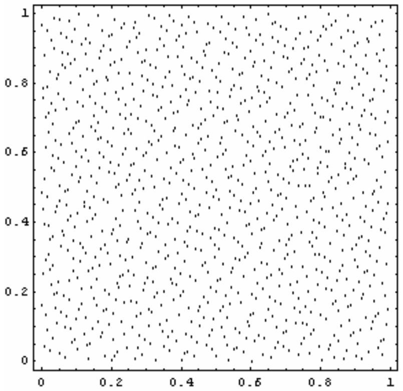


Figure 2 Sample points of quasi random sequence

3. Basic Particle Swarm Optimization

For a D-dimensional search space the position of the ith particle is represented as $X_i = (x_{i1}, x_{i2}, …, x_{iD})$. Each particle maintains a memory of its previous best position $P_i = (p_{i1}, p_{i2}… p_{iD})$. The best one among all the particles in the population is represented as $P_g = (p_{g1}, p_{g2}… p_{gD})$. The velocity of each particle is represented as $V_i = (v_{i1}, v_{i2}, … v_{iD})$. In each iteration, the P vector of the particle with best fitness in the local neighborhood, designated g, and the P vector of the current particle are combined to adjust the velocity along each dimension and a new position of the particle is determined using that velocity. The two basic equations which govern the working of PSO are that of velocity vector and position vector given by:

$$v_{id} = wv_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \qquad (1)$$

$$x_{id} = x_{id} + v_{id} \qquad (2)$$

The first part of equation (1) represents the inertia of the previous velocity, the second part is the cognition part and it tells us about the personal thinking of the particle, the third part represents the cooperation among particles and is therefore named as the social component. Acceleration constants c_1, c_2 and inertia weight w are the predefined by the user and r_1, r_2 are the uniformly generated random numbers in the range of [0,1].

4. Proposed Algorithm

The proposed algorithm is an extension to the Basic Particle Swarm Optimization, by including the component of mutation in it. The mutation operator defined in the present work uses quasi random Sobol sequence and is called a systematic mutation (SM) operator. We have proposed two versions using SM, called SM-PSO1 and SM-PSO2. The two versions differ from each other in the sense that in SM-PSO1, the best particle of the swarm is mutated, whereas in SM-PSO2, the worst particle of the swarm is mutated.

The SM operator is defined as

$SM = R_1 + (R_2 / \ln R_1)$,
Where R1 and R2 are random numbers in a Sobol sequence.

The idea behind applying the mutation to the worst particle is to push the swarm from the back. The quasi random numbers used in the SM operator allows the worst particle to move forward systemically.

4.1. Pseudo Code of SMPSO1 Algorithm

The Pseudo code of SM-PSO1 is described as follows:

```
Initialize the population
Do
    w linearly decreases from 0.9 to 0.4
    For i=1 to population size M
        For d=1 to dimension D
            $v_{id}=w*v_{id}+c_1*r_1*(P_{id}-x_{id})+c_2*r_2*(P_{gd}-x_{id})$
            $x_{id}=x_{id}+v_{id}$
        End for
        If ( f(Xi) < f(Pi) )      Pi = Xi.
            If ( f(Pi) < f(Pg) )   Pg = Pi
            End if
        End if
        For d=1 to dimension D
            $R_1$ = sobolrand();  $R_2$ = sobolrand();
            $temp_d = R_1 + (R_2 / \ln R_1)$
        End for
        If ( f(temp) < f(Pg) )  Pg = temp
        End if
    End for
Until stopping criteria is reached
```

The function sobolrand() returns a random number distributed by sobol sequence. The computational steps of SMPSO2 are same as that of SMPSO1, except for the fact that the worst particle in the swarm is mutated instead of the best particle.

5. Benchmark Problems and Results

In the present study we have taken 3 benchmark problems (Table 1), which are considered to be starting point for checking the credibility of any optimization algorithm.

All the test problems are highly multimodal and scalable in nature. Each function is tested with a swarm size of 20, 40 and 80 for dimension 10, 20, 30. The maximum number of generations is set as 1000, 1500 and 2000 corresponding to the dimensions 10, 20 and 30 respectively. A total of 30 runs for each experimental setting are conducted and the average fitness of the best solutions throughout the run is recorded.

Table 1 Numerical benchmark problems

Function	Dim (n)	Range
Rastringin $f_1(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	10 20 30	[2.56,5.12]
Griewank $f_2(x) = \frac{1}{4000} \sum_{i=0}^{n-1} x_i^2 - \prod_{i=0}^{n-1} \cos(\frac{x_i}{\sqrt{i+1}}) + 1$	10 20 30	[300,600]
Rosenbrock $f_3(x) = \sum_{i=0}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	10 20 30	[15,30]

Table 2 Results of Rastringin function (Mean best)

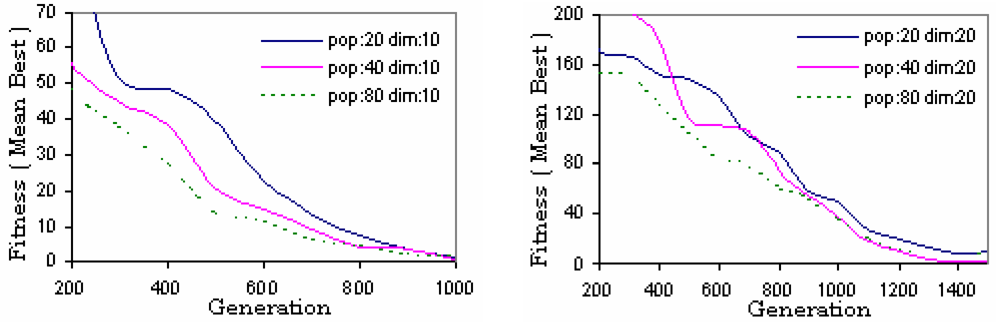
Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [8]	QPSO [8]	Mutation gbest [7]	Mutation gbest [8]
20	10	1000	0.881465	**0.641812**	5.5382	5.2543	5.2216	4.3976
	20	1500	5.014802	**4.52709**	23.1544	16.2673	16.1562	14.1678
	30	2000	13.152097	**12.669938**	47.4168	31.4576	26.2507	25.6415
40	10	1000	1.241561	**0.85634**	3.5778	3.5685	3.3361	3.2046
	20	1500	5.91223	**5.472557**	16.4337	11.1351	10.9072	9.5793
	30	2000	**13.005205**	14.523385	37.2896	22.9594	19.6360	20.5479
80	10	1000	1.182363	**0.813593**	2.5646	2.1245	2.0185	1.7166
	20	1500	5.501107	**4.97266**	13.3826	10.2759	7.7928	7.2041
	30	2000	**10.210538**	15.028891	28.6293	16.7768	14.9055	15.0393

Table 3 Results of Griewank function (Mean best)

Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [8]	QPSO [8]	Mutation gbest [7]	Mutation gbest [8]
20	10	1000	**0.006896**	0.007877	0.09217	0.08331	0.0627	0.0780
	20	1500	0.009177	**0.008486**	0.03002	0.02033	0.0209	0.0235
	30	2000	0.025227	**0.014541**	0.01811	0.01119	0.0110	0.0099
40	10	1000	0.009677	**0.009515**	0.08496	0.06912	0.0539	0.0641
	20	1500	0.017195	**0.012269**	0.02719	0.01666	0.0238	0.0191
	30	2000	0.030103	**0.011066**	0.01267	0.01161	0.0119	0.0098
80	10	1000	0.00886	**0.006402**	0.07484	0.03508	0.0419	0.0460
	20	1500	**0.010828**	0.01296	0.02854	0.01460	0.0136	0.0186
	30	2000	0.024265_	**0.004692**	0.01258	0.01136	0.0120	0.0069

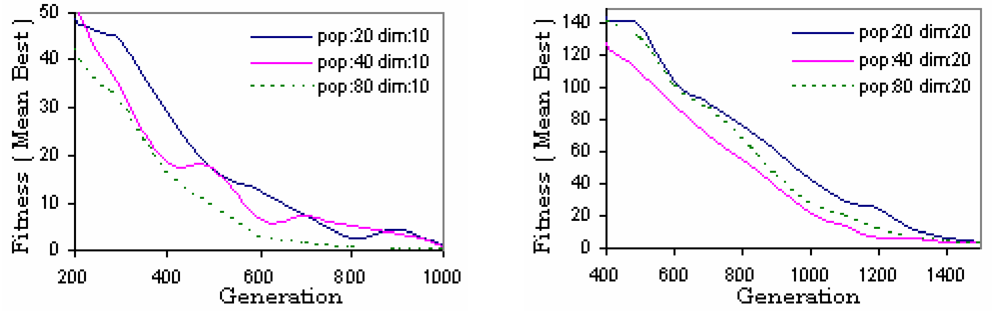
Table 4 Results of Rosenbrock function (Mean best)

Pop	Dim	Gne	SMPSO1 (gbest)	SMPSO2 (gworst)	BPSO [8]	QPSO [8]	Mutation gbest [7]	Mutation gbest [8]
20	10	1000	6.416553	**6.410466**	94.1276	59.4764	27.4620	21.2081
	20	1500	17.311169	**17.287586**	204.336	110.664	49.1176	61.9268
	30	2000	30.566478	**28.259791**	313.734	147.609	97.5952	86.1195
40	10	1000	6.4147	**6.401132**	71.0239	10.4238	7.8741	8.1828
	20	1500	**17.23444**	17.250421	179.291	46.5957	28.4435	40.0749
	30	2000	**28.114756**	28.640997	289.593	59.0291	62.3854	65.2891
80	10	1000	6.416151	**6.345346**	37.3747	8.63638	6.7098	7.3686
	20	1500	17.440593	**17.190714**	83.6931	35.8947	31.0929	30.1607
	30	2000	**28.324733**	30.153352	202.672	51.5479	43.7622	38.3036



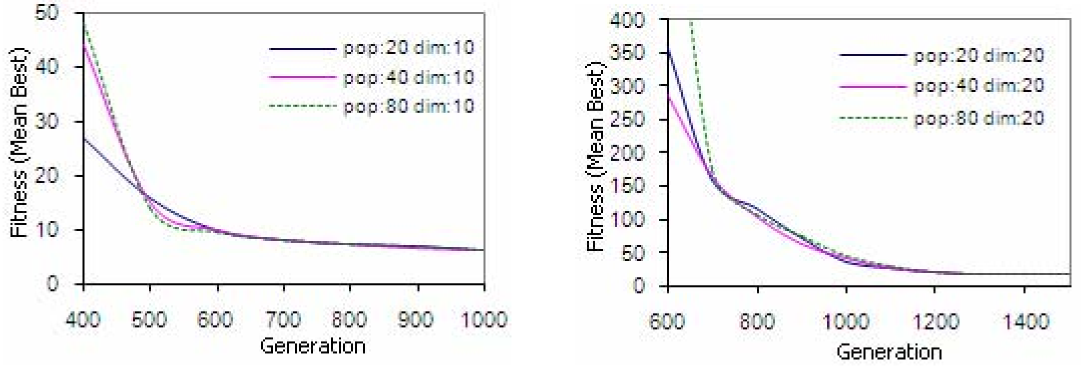
(a) (b)

Figure 3 Performance for Rastringin function of SMPSO1
(a) Dimension 10 (b) Dimension 20



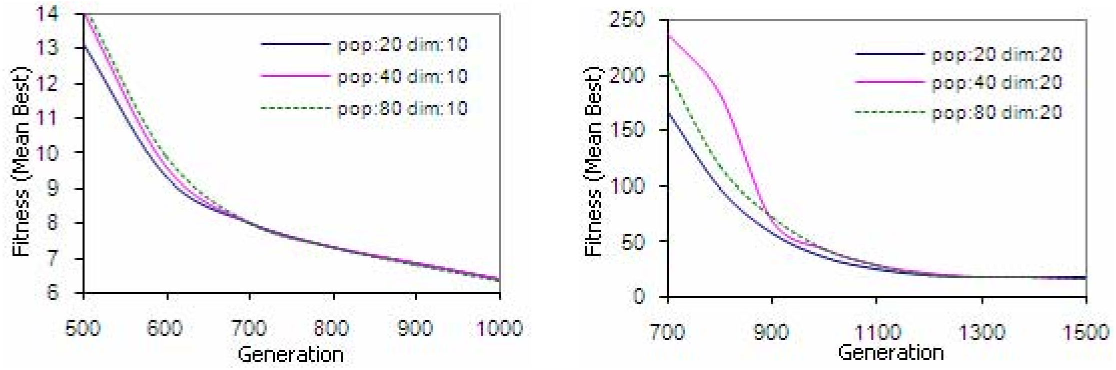
(a) (b)

Figure 4 Performance for Rastringin function of SMPSO2
(a) Dimension 10 (b) Dimension 20



(a) (b)

Figure 5 Performance for Rosenbrock function of SMPSO1
(a) Dimension 10 (b) Dimension 20



(a) (b)

Figure 6 Performance for Rosenbrock function of SMPSO2
(a) Dimension 10 (b) Dimension 20

The mean best fitness value for the functions $f_1 - f_3$ are given in Tables 2 – 4, respectively, in which Pop represents the swarm population, Dim represents the dimension and Gne represents the maximum number of permissible generations. Figures 3 and 4 show the mean best fitness curves for the Rastringin function corresponding to the algorithms SM-PSO1 and SM-PSO2 respectively. Figures 5 and 6 show the mean best fitness curves for the Rosenbrock function corresponding to the algorithms SM-PSO1 and SM-PSO2 respectively.

Comparison of the proposed algorithms is done with one of the latest variant of BPSO, called Quantum PSO (QPSO). According to the researchers, QPSO performs better than the BPSO for well-known benchmark problems (for more details on QPSO please see [7], [8]). The numerical results show that in all the test cases the proposed algorithms perform much better than the other algorithms. If we compare the performance of SMPSO1 and SMPSO2 with each other then from the numerical results we can see that SMPSO2 in which the worst particle of the swarm is mutated gave better results than SMPSO1 in 20 test cases out of the total 27 cases tried.

6. Conclusion

In this paper, we proposed a new mutation operator called SM mutation operator for improving the performance of a BPSO. Two versions of algorithms called SMPSO1 and SMPSO 2 are defined. The empirical studies show that the proposed versions are better than the BPSO and QPSO quite significantly. However, we would like to add that we have tested the algorithms on a narrow platform of benchmark problems. Thus making any concrete judgment about the proposed algorithms is not justified. In future, we will test the proposed algorithms on a wider platform consisting of more complex, multidimensional problems having larger number of variables. In addition, it will be interesting to do some theoretical research on the better performance of PSO when the worst particle is mutated.

References

[1] J. Kennedy and R. C. Eberhart, “Particle Swarm Optimization”, IEEE Int. Conf. on Neural Networks

(Perth, Australia), IEEE Service Center, Piscataway, NJ, 1995, pp. 1942-1948.

[2] H. Higashi and H. Iba, “Particle Swarm Optimization with Gaussian Mutation”, In Proc. of the IEEE swarm Intelligence Symposium, 2003, pp. 72 – 79.

[3] A. Stacey, M. Jancic and I. Grundy, “Particle Swarm Optimization with Mutation”, in Proc. of the IEEE Congress on Evolutionary Computation, 2003, pp. 1425 – 1430.

[4] Mille Pant, Radha Thangaraj and Ajith Abraham, “Improved Particle Swarm Optimization with Low-Discrepancy Sequences”, to appear in Proc. of IEEE Congress on Evolutionary Computation, 2008.

[5] Nguyen X. H., Nguyen Q. Uy., R. I. Mckay and P. M. Tuan, “Initializing PSO with Randomized Low-Discrepancy Sequences: The Comparative Results”, In Proc. of IEEE Congress on Evolutionary Algorithms, 2007, pp. 1985 – 1992.

[6] H. M. Chi, P. Beerli, D. W. Evans, and M. Mascagni, “On the Scrambled Sobol Sequence”, In Proc. of Workshop on Parellel Monte Carlo Algorithms for Diverse Applications in a Distributed Setting, LNCS 3516, Springer Verlog, 1999, pp. 775 – 782.

[7] Liu J, Xu W, Sun J. Quantum-Behaved Particle Swarm Optimization with Mutation Operator. In Proc. of the 17th IEEE Int. Conf. on Tools with Artificial Intelligence, Hong Kong (China), 2005.

[8] Liu J, Sun J, Xu W, Quantum-Behaved Particle Swarm Optimization with Adaptive Mutation Operator. ICNC 2006, Part I, Springer-Verlag: 959 – 967, 2006.