Particle Swarm Optimization with Crossover Operator and its Engineering Applications

Millie Pant, *Member, IAENG,* Radha Thangaraj, *Member, IAENG*, and V. P. Singh

***Abstract*— This paper presents a variant of diversity guided Particle Swarm Optimization (PSO) algorithm named QIPSO for solving global optimization problems. In QIPSO the conventional framework of PSO is modified by including a crossover operator to maintain the level of diversity in the swarm population. Numerical results show that the induction of a crossover operator not only discourages premature convergence to the local optima but also explores promising regions of the search space effectively. Empirical results show the superior performance of QIPSO with conventional PSO and ARPSO. Efficiency of QIPSO is further validated by applying it to a set of five real life problems (RLPs) with constraints. Penalty method is used for dealing with constraints. Once again the simulation results show the compatibility of QIPSO for solving real life problems.**

***Index Terms*— Particle Swarm Optimization, Crossover operator, Quadratic Interpolation, Real life problems.**

I. Introduction

Population based search algorithms like Genetic Algorithms (GA), Evolutionary Programming (EP), Differential Evolution (DE), Particle Swarm Optimization (PSO) etc. are perhaps some of the most popular stochastic techniques for solving continuous global optimization problems. These techniques have shown their efficiency for solving complex test as well as real life problems. Of the above mentioned algorithms other than PSO all other algorithms are inspired by the phenomenon of Darwin's concept of '*survival of the fittest'*. Classical PSO on the other hand follows the policy of cooperation and social behavior displayed by various species like ants, birds, fish etc. Nevertheless all the above mentioned algorithms share some common features as pointed out by Angeline [1]:

- All are population based search techniques.
- None of the above mentioned algorithms require the auxiliary knowledge (like continuity, differentiability etc.) of the problem.
- In all the algorithms solutions belonging to the same population interact with each other during the search process.
- The quality of the solutions are improved using techniques inspired from real world phenomenon like human genetics in case of EA and cooperative behavior in case of PSO.

It is worth mentioning that although these algorithms have been successful in solving a wide variety of problems, their performance is criticized one certain aspects. For example the problem of the loss of diversity after subsequent iterations which lead to premature convergence leading to suboptimal solution [2]. Loss of diversity becomes more prominent for multimodal functions having several optima or noisy functions where the optimum keeps shifting from one position to other. Loss of diversity generally takes place when the balance between the two antagonists processes exploration (searching of the search space) and exploitation (convergence towards the optimum) is disturbed. In case of evolutionary algorithms the population diversity is generally lost during the process of evolution (crossover and mutation), whereas in case of PSO the diversity loss is generally attributed to the fast information flow between the swarm particles. Thus in absence of a good diversity enhancing mechanism the optimization algorithms are unable to explore the search space effectively.

One of the simplest methods to overcome the problem of diversity loss is to capitalize the strengths of EA and PSO together in an algorithm. A variety of methods combining the aspects of EA and PSO are available in literature. For detailed study, the reader is suggested [3] - [5], etc. Out of the EA operators, mutation is the most widely used EA tool applied in PSO [6] - [9] etc. However very few examples of selection and crossover operator are available in literature.

Keeping this in mind we proposed a variant of diversity guided PSO called QIPSO which make use of crossover operator to maintain the diversity of the population [10]. In [10], we defined a new nonlinear quadratic crossover operator which makes use of three members of the swarm to produce a new member. This operator is activated when the diversity of the swarm becomes less than a certain threshold (say d_{low}) and stops when the required diversity (say d_{high}) is achieved. We checked the QIPSO algorithm with thirteen test suit of benchmark problems (unconstrained), the experimental results shown that the new algorithm gave better results. Motivated by the preliminary good results shown by QIPSO and to further validate its efficiency, we used it to solve real life engineering design problems with constraints associated with them. Penalty function approach is used to deal with constraints.

The structure of the paper is as follows: in section II, we briefly explain the BPSO and ARPSO (another diversity guided PSO [11]), in section III; we describe the QIPSO algorithm and its performance on thirteen standard benchmark problems taken from literature. Section IV deals with the performance of QIPSO algorithm for constrained optimization problems. Finally the paper concludes with section V.

Manuscript received June 19, 2008.

Millie Pant is with the Department of Paper Technology, Indian Institute of Technology Roorkee, (Saharanpur Campus), Saharanpur – 247001, India. (e-mail: millifpt@iitr.ernet.in).

Radha Thangaraj is with the Department of Paper Technology, Indian Institute of Technology Roorkee, (Saharanpur Campus), Saharanpur – 247001, India. (e-mail: t.radha@ieee.org).

V.P.Singh is with the Department of Paper Technology, Indian Institute of Technology Roorkee, (Saharanpur Campus), Saharanpur – 247001, India. (e-mail: singhfpt@iitr.ernet.in).

II. ALGORITHMS USED FOR COMPARISON

In the present article we compared the performance of the proposed QIPSO with the conventional or basic PSO and ARPSO. ARPSO, described later in this section, uses the unique concept of repulsing the population points for enhancing the diversity of the population. Previous studies have shown ARPSO to be competent for solving numerical benchmark problems with sufficiently high dimension.

A. Basic Particle Swarm Optimization (BPSO)

Particle Swarm Optimization (PSO) is a relatively newer addition to a class of population based search technique for solving numerical optimization problems. A unique feature of PSO is that it maintains the memory of previous best positions. The particles or members of the swarm fly through a multidimensional search space looking for a potential solution. 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).

For a D-dimensional search space the position of the ith particle is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. Each particle maintains a memory of its previous best position $P_{besti} = (p_{i1}, p_{i2} \ldots p_{iD})$. The best one among all the particles in the population is represented as $P_{gbest} = (p_{g1}, p_{g2} \ldots p_{gD})$. The velocity of each particle is represented as $V_i = (v_{i1}, v_{i2}, \ldots v_{iD})$. 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:

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

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

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 [12]. Acceleration constants c_1, c_2 [13] and inertia weight w [14] are the predefined by the user and r_1, r_2 are the uniformly generated random numbers in the range of [0, 1].

B. ARPSO Algorithm

The ARPSO algorithm is diversity guided BPSO which uses an interesting concept of diverging the population points when the diversity becomes less than the desired threshold value. ARPSO consists of two phases and the swarm population keeps shuttling between the phases of attraction and repulsion according to the increase or decrease in diversity measure. In the attraction phase, the particles come towards each other following equation (1) as they do in BPSO. The movement of particles towards each other causes a gradual decrease in diversity of the population. When the diversity becomes lower then a certain specified value d_{low}, a repulsion phase obtained by inverting the velocity update formula is activated according to the following equation:

$$v_{id} = wv_{id} - c_1 r_1 (p_{id} - x_{id}) - c_2 r_2 (p_{gd} - x_{id}) \quad (3)$$

In this phase the particles are no longer attracted towards each other but move away or from each other. This generates a perturbation in the population and causes an increase in the diversity of the swarm population. The swarm particles stay in this phase until the diversity reaches a higher value d_{high}. As soon as the desired high diversity d_{high} is achieved, the swarm particles again come back to the attraction phase and the same process continues iteratively until the global optimum is obtained.

III. QIPSO ALGORITHM FOR UNCONSTRAINED OPTIMIZATION

A. Proposed QIPSO Algorithm

QIPSO is a simple and modified version of BPSO with an added crossover operator to enhance the performance of BPSO without disturbing the inherent features of BPSO. Like ARPSO, QIPSO uses diversity as a measure to guide the swarm, but instead of repulsing the population points, it makes use of crossover operator to explore the promising areas of the search domain.

The crossover operator applied in QIPSO is quadratic interpolation (QI) operator. QI is a gradient free direct search technique used for solving nonlinear optimization problems. The concept of using QI in optimization methods is not new. It has also been used as a method to generate new point in population based search techniques [15], [16]. Mathematically, the point generated by QI lies at the point of minima of the quadratic curve passing through three points. Initially, QI method used for solving nonlinear optimization problems considered three points to be equidistant from each other. However, in the present study in order to provide more randomness to explore the search space the particle having the best fitness value (global best particle, P_{gbest}) is always selected whereas the other two points are randomly chosen from the remaining population.

Mathematically, the working of QI operator may be defined as:

If $a = P_{gbest}$ represents the global best position of the swarm and let b, c represent randomly chosen swarm particles such that $a \neq b \neq c$, then the coordinates of the new point generated by using QI is given as $\tilde{x}^i = (\tilde{x}^1, \tilde{x}^2, \ldots \tilde{x}^n)$, where

$$\tilde{x}^i = \frac{1}{2} \frac{(b^{i^2} - c^{i^2}) * f(a) + (c^{i^2} - a^{i^2}) * f(b) + (a^{i^2} - b^{i^2}) * f(c)}{(b^i - c^i) * f(a) + (c^i - a^i) * f(b) + (a^i - b^i) * f(c)} \quad (4)$$

When the diversity is less than d_{low}, QI operator is activated and the particle having the best value (P_{gbest}) with the help of two other distinct random points starts exploring the domain in search of other potential candidates. This process continues till the desired diversity level is attained. Flow of the proposed QIPSO is given in Fig.1. A C++ style implementation of QI operator is as follows:

```
If (Diversity < d_low)
  do
    {
     // generate a new point using (4)
    } while (Diversity > d_high);
End if
```

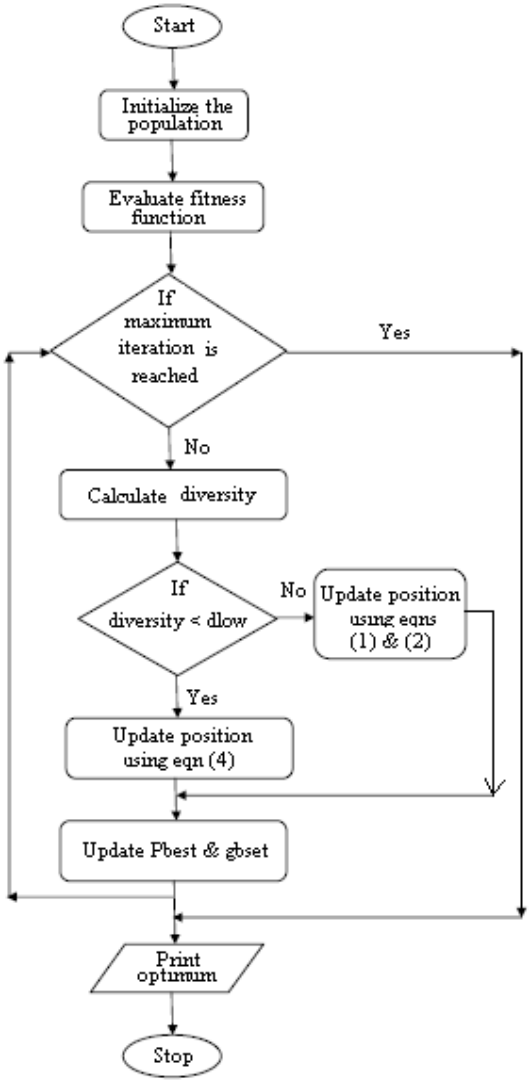

Fig. 1 Flowchart of QIPSO algorithm

B. Experimental Setting, Benchmark Problems and Result Analysis

1) Experimental Settings

All the parameters in PSO play an important role in deciding the success of an algorithm. In order to make a fair comparison of BPSO, ARPSO and QIPSO, we fixed the same seed for random number generation so that the initial swarm population is same for all the three algorithms. We performed a number of experiments with varying population sizes, inertia weight and acceleration constants. From the empirical results we observed that a swarm size of 2*n (where n is the number of variables) is sufficient for small and medium sized problems. A linearly decreasing inertia weight starting at 0.9 and finishing at 0.4 is taken and acceleration constants c_1 and c_2 are taken as 1.8. However depending on the complexity of the problem user may vary the parameters accordingly.

Diversity is an important aspect for checking the efficiency of the swarm. Ideally, the diversity of an algorithm should be high in the beginning decreasing slowly in the successive iterations. The diversity measure [17] of the swarm in the present study is taken as:

$$Diversity(S(t)) = \frac{1}{n_s}\sum_{i=1}^{n_s}\sqrt{\sum_{j=1}^{n_x}(x_{ij}(t)-\overline{x_j(t)})^2} \qquad (5)$$

where S is the swarm, $n_s = |S|$ is the swarm size, n_x is the dimensionality of the problem, x_{ij} is the j'th value of the i'th particle and $\overline{x_j(t)}$ is the average of the j-th dimension over all particles, i.e.

$$\overline{x_j(t)} = \frac{\sum_{i=1}^{n_s} x_{ij}(t)}{n_s}.$$

The diversity controlling parameters d_{low} and d_{high} are taken as $5.0*10^{-6}$ and 0.25 respectively. For all algorithms, the maximum number of iterations allowed was set to 10,000. A total of 30 runs for each experimental setting were conducted and the average fitness of the best solutions throughout the run was recorded.

2) Benchmark Problems

To test the credibility of the QIPSO algorithm we tested it on a test bed of 13 bench mark problems. The test bed comprises of a variety of problems ranging from a simple spherical function to highly multimodal functions and also a noisy function (with changing optima). These test problems provide a platform for testing the credibility of an algorithm and also represents the scenarios that may occur in real life cases. The first function f_1 is a simple unimodal function with objective function value as zero. Any decent optimization algorithm is able to solve it but when we increase the dimension of the problem the algorithms sometimes have difficulty in reaching the true objective function value. Functions f_7 to f_{13} are highly multimodal, where the number of peaks and valleys goes on increasing with increase in number of variables. All the functions are scalable and are tested for dimensions 30, 50 and 100.

In Table II, we give the results of problems with dimension 30, in Table III results of the same problems with increased dimension 50 are given and in Table IV results are recorded for dimension 100. Algorithms are compared in terms of mean best fitness value.

3) Results analysis for problems with dimension 30:

For dimension 30 (Table I), we have referred the results of BPSO and ARPSO presented in Ref. [18] for the purpose of comparison with QIPSO. From the numerical results it is evident that QIPSO gave a superior performance in comparison to the other two algorithms in 12 of out of 13 test cases which shows a success rate of more than 90%. For f_7 the performance of QIPSO is not as good as the other two algorithms but still is a satisfactory value. The best function value for f_7 is obtained from ARPSO. From Table III, which gives the percentage of improvement in the function value, the superior performance of QIPSO is evident.

4) Results analysis for problems with dimension50:

Generally the performance of an algorithm deteriorates on increasing the dimension of the problem. Therefore we increased the dimension of the test problems to 50. From Table II, it is clear that even for increased dimension the performance of QIPSO remains quite stable in terms of fitness function value. Surprisingly, for f_7 QIPSO gave a better value with increased dimension. For the remaining functions also QIPSO gave a superior performance in comparison to other algorithms. Moreover the stability of QIPSO can be seen from the objective function values which are near to the true function values. Similar observations can

be made from Table III which gives the percentage of improvement. Figure 2 gives the 3D plot of some selected benchmark problems.

Figures 3 - 7 show the mean best fitness curves for selected benchmark problems. Figures 8 – 9 show the comparison of fitness function for selected benchmark problems.

5) Results analysis for problems with dimension100:

For dimension 100 (Table IV) also, we have referred the results of BPSO and ARPSO presented in Ref. [18] for the purpose of comparison with QIPSO. Table V gives the percentage of improvement of QIPSO in comparison with BPSO and ARPSO. From Table IV, it is clear that even for increased dimension the performance of QIPSO remains quite stable in terms of fitness function value. Surprisingly, for function f_6 BPSO gave a better value than QIPSO and for function f_{10} ARPSO gave a better value. For the remaining functions QIPSO gave a superior performance in comparison to other algorithms.

The complete formulation of the test of problems is given below:

1. Sphere function

$\min_x \quad f_1(x) = \sum_{i=1}^{n} x_i^2$, $-5.12 \le x_i \le 5.12$, $x^* = (0,0,...,0)$, $f_1(x^*) = 0$.

2. Schwefel function 1

$\min_x \quad f_2(x) = \sum_{i=0}^{n-1} |x_i| + \prod_{i=0}^{n-1} |x_i|$, $-10 \le x_i \le 10$, $x^* = (0,0,...,0)$, $f_2(x^*) = 0$.

3. Function f3

$\min_x \quad f_3(x) = \sum_{i=0}^{n-1} (\sum_{j=0}^{i} x_i)^2$, $-100 \le x_i \le 100$, $x^* = (0,0,...,0)$, $f_3(x^*) = 0$.

4. Schwefel function 2

$\min_x \quad f_4(x) = \max |x_i|$, $0 \le i < n$, $-100 \le x_i \le 100$, $x^* = (0,0,...,0)$, $f_4(x^*) = 0$.

5. Rosenbrock function

$\min_x \quad f_5(x) = \sum_{i=0}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$, $-30 \le x_i \le 30$, $x^* = (1,1,...,1)$, $f_5(x^*) = 0$.

6. Step function

$\min_x \quad f_6(x) = \sum_{i=0}^{n-1} \lfloor x_i + 1/2 \rfloor^2$, $-100 \le x_i \le 100$, $x^* = (0,0,...,0)$, $f_6(x^*) = 0$.

7. Dejong's function with noise

$\min_x \quad f_7(x) = (\sum_{i=0}^{n-1} (i+1) x_i^4) + rand[0,1]$, $-1.28 \le x_i \le 1.28$, $x^* = (0,0,...,0)$, $f_7(x^*) = 0$.

8. Schwefel function 3

$\min_x \quad f_8(x) = -\sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$, $-500 \le x_i \le 500$, $x^* = (420.97,420.97,...,420.97)$, $f_8(x^*) = -418.9829 * n$.

9. Rastringin function

$\min_x \quad f_9(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$, $-5.12 \le x_i \le 5.12$, $x^* = (0,0,...,0)$, $f_9(x^*) = 0$.

10. Ackley's function

$\min_x \quad f_{10}(x) = 20 + e - 20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i))$, $-32 \le x_i \le 32$, $x^* = (0,0,...,0)$, $f_{10}(x^*) = 0$.

11. Griewank function

$\min_x \quad f_{11}(x) = \frac{1}{4000}\sum_{i=0}^{n-1} x_i^2 - \sum_{i=0}^{n-1} \cos(\frac{x_i}{\sqrt{i+1}}) + 1$, $-600 \le x_i \le 600$, $x^* = (0,0,...,0)$, $f_{11}(x^*) = 0$.

12. Generalized penalized function 1

$\min_x \quad f_{12}(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10\sin^2(y_{i+1}\pi)] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i,10,100,4)$, where $y_i = 1 + \frac{1}{4}(x_i + 1)$ $-50 \le x_i \le 50$, $x^* = (0,0,...,0)$, $f_{12}(x^*) = 0$.

13. Generalized penalized function 1

$\min_x \quad f_{13}(x) = (0.1)\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} ((x_i - 1)^2 (1 + \sin^2(3\pi x_{i+1}))) + (x_n - 1)(1 + \sin^2(2\pi x_n))\} + \sum_{i=0}^{n-1} u(x_i,5,100,4)$, $-50 \le x_i \le 50$, $x^* = (1,1,...,-4.76)$, $f_{13}(x^*) = -1.1428$.

In problem 12 1nd 13 the value of penalty function u is given by,

$u(x,a,b,c) = b(x-a)^c$, if $x > a$

$u(x,a,b,c) = b(-x-a)^c$, if $x < -a$

$u(x,a,b,c) = 0$, if $-a \le x \le a$.

IV. QIPSO Algorithm for Constrained Optimization

The QIPSO algorithm described in the previous section is used for solving constrained real life optimization problems in are given in the subsequent subsections. Penalty method [16], given in the following subsection, is used to handle the constraints.

A. Penalty method for solving constrained optimization problems

Many real-world optimization problems are solved subject to sets of constraints. The search space in Constrained Optimization Problems (COPs) consists of two kinds of solutions: feasible and infeasible. Feasible points satisfy all the constraints, while infeasible points violate at least one of them. Therefore the final solution of an optimization problem must satisfy all constraints.

In this paper, the two algorithms BPSO and QIPSO handle the constraints using the concept of penalty functions. In the penalty function approach, the constrained problem is transformed into an unconstrained optimization algorithm by penalizing the constraints and building a single objective function, which is minimized using an unconstrained optimization algorithm.

That is,

$$F(x) = f(x) + \lambda \ p(x) \tag{6}$$

Where $p(x_i, t) = \sum_{m=1}^{n_g + n_h} \lambda_m(t) p_m(x_i)$ (7)

$$p_m(x_i) = \max\{0, g_m(x_i)^{\alpha}\}$$

$$\text{if } m \in [1, \ldots\ldots, n_g] \text{ (inequality)} \tag{8}$$

$$p_m(x_i) = |h_m(x_i)|^{\alpha}$$

$$\text{if } m \in [n_g + 1, \ldots\ldots, n_g + n_h] \text{ (equality)} \tag{9}$$

with α a positive constant, representing the power of the penalty. The inequality constraints are considered as g(x) and h(x) represents the equality constraints. n_g and n_h denotes the number inequality and equality constraints respectively. λ is the constraint penalty coefficient

TABLE I COMPARISON RESULTS OF TEST PROBLEMS OF DIMENSION 30

Function	Mean Best Fitness		
	BPSO	ARPSO	QIPSO
f_1	0.000000e+00	6.808174e-13	**0.000000e+00**
f_2	0.000000e+00	2.089204e-02	**0.000000e+00**
f_3	0.000000e+00	0.000000e+00	**0.000000e+00**
f_4	2.107015e-16	1.418379e-05	**1.647667e-19**
f_5	4.026386e+00	3.550929e+02	**8.608759e-19**
f_6	4.000000e-02	1.898000e+01	**0.000000e+00**
f_7	1.908221e-03	**3.886668e-04**	1.51675e-01
f_8	-7.187408e+03	-8.598653e+03	**-1.054997e+04**
f_9	4.917079e+01	2.149141e+00	**0.000000e+00**
f_{10}	1.404689e+00	1.842273e-07	**6.292307e-10**
f_{11}	2.352893e-02	9.234456e-02	**0.000000e+00**
f_{12}	3.819961e-01	8.559789e-03	**5.505851e-13**
f_{13}	-5.968873e-01	-9.626354e-01	**-1.150438e+00**

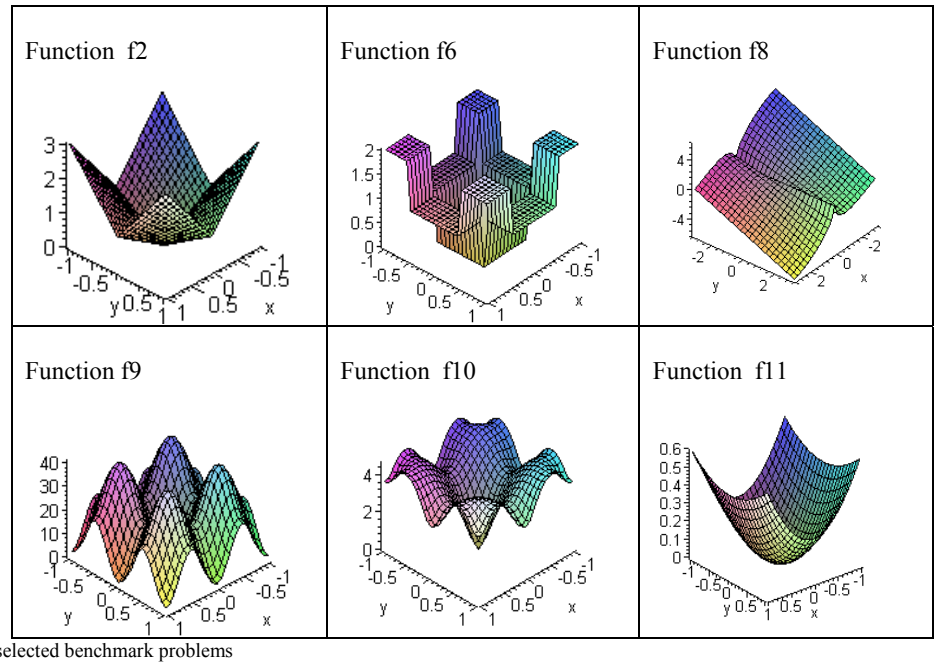

Fig. 2. 3D plots of selected benchmark problems

TABLE II COMPARISON RESULTS OF TEST PROBLEMS OF DIMENSION 50

Function	Mean Best Fitness		
	BPSO	ARPSO	QIPSO
f_1	4.867600e-08	2.621439e+01	**0.000000e+00**
f_2	9.782400e+00	1.07820e+01	**0.000000e+00**
f_3	7.462966e-01	8.566760e-03	**0.000000e+00**
f_4	2.633483e-07	8.837717e-07	**1.765932e-07**
f_5	3.322306e-14	3.986624e+00	**2.930120e-19**
f_6	1.320000e+02	4.000000e+00	**0.000000e+00**
f_7	9.166647e+00	4.871490e+00	**1.615060e-01**
f_8	-1.365149e+04	-1.542069e+04	**-1.702854e+04**
f_9	2.596818e+02	3.554837e+02	**1.069930e+02**
f_{10}	2.342900e+01	1.948324e+01	**1.646220e+00**
f_{11}	2.466000e-03	1.084202e-19	**0.000000e+00**
f_{12}	5.643325e+00	2.270458e-09	**3.303511e-13**
f_{13}	-1.144380e+00	**-1.150438e+00**	**-1.150438e+00**

TABLE III IMPROVEMENT (%) OF QIPSO IN COMPARISON WITH BPSO AND ARPSO

Function	Dimension 30		Function	Dimension 50	
	Improvement with BPSO (%)	Improvement with ARPSO (%)		Improvement with BPSO (%)	Improvement with ARPSO (%)
f_1	0.000000	100	f_1	100	100
f_2	0.000000	100	f_2	100	100
f_3	0.000000	0.000000	f_3	100	100
f_4	99.921801	100	f_4	32.943102	80.018233
f_5	100	100	f_5	99.999118	100
f_6	100	100	f_6	100	100
f_7	-	-	f_7	98.238113	96.684669
f_8	46.784070	22.693287	f_8	23.737529	10.426576
f_9	100	100	f_9	58.798422	69.902136
f_{10}	99.999999	99.658449	f_{10}	92.973579	91.55058
f_{11}	100	100	f_{11}	100	100
f_{12}	100	99.999999	f_{12}	100	99.98545
f_{13}	92.739567	19.509214	f_{13}	0.5293696	0.000000

TABLE IV COMPARISON RESULTS OF TEST PROBLEMS OF DIMENSION 100

Function	Mean Best Fitness		
	BPSO	ARPSO	QIPSO
f_1	0.0000000e+00	7.4869991e+02	**0.000000e+00**
f_2	1.8045813e+01	3.9637792e+01	**2.96156e-005**
f_3	3.6666668e+03	1.8174752e+01	**4.960550e-01**
f_4	5.3121806e+00	2.4367166e+00	**2.396920e+00**
f_5	2.0203629e+02	2.3609401e+02	**1.712801e+01**
f_6	**2.1000000e+00**	4.1183333e+02	1.200000e+01
f_7	2.7845728e-02	3.2324733e-03	**5.540010e-05**
f_8	-2.1579648e+04	-2.1209102e+04	**-3.931473e+04**
f_9	2.4359139e+02	4.8096522e+01	**3.360902e+01**
f_{10}	4.4934316e+00	**5.6281044e-02**	1.911440e+00
f_{11}	4.1715080e-01	8.5311042e-02	**2.466180e-03**
f_{12}	1.1774980e-01	9.219929e-02	**8.027411e-12**
f_{13}	-3.8604485e-01	3.3010679e+02	**-1.139458e+00**

TABLE V IMPROVEMENT (%) OF QIPSO IN COMPARISON WITH BPSO AND ARPSO: DIMENSION 100

Function	Improvement with BPSO (%)	Improvement with ARPSO (%)	Function	Improvement with BPSO (%)	Improvement with ARPSO (%)
f_1	0.000000	100	f_8	82.18429	85.36725
f_2	102.28467	99.99992	f_9	86.20270	30.12172
f_3	99.986471	97.27063	f_{10}	57.46144	----
f_4	54.87879	1.63318	f_{11}	99.40880	97.10915
f_5	91.52231	92.74525	f_{12}	100	100
f_6	----	97.08619	f_{13}	195.16272	100.34517
f_7	99.801045	98.28614			

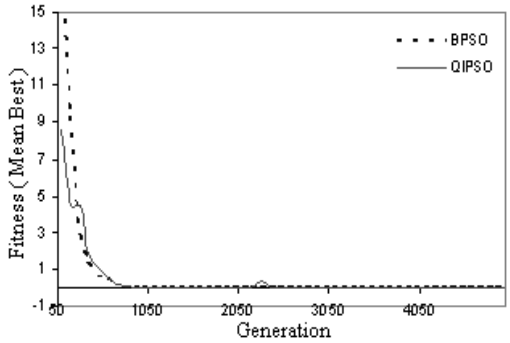

Fig. 3 Convergence graph for the function f_4

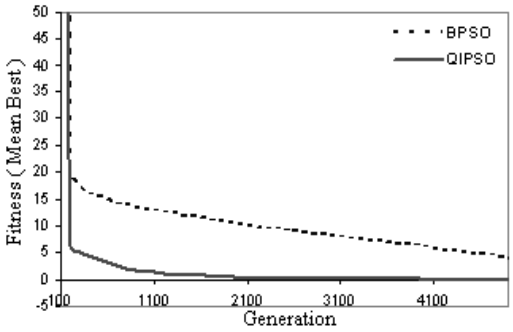

Fig. 4 Convergence graph for the function f_5

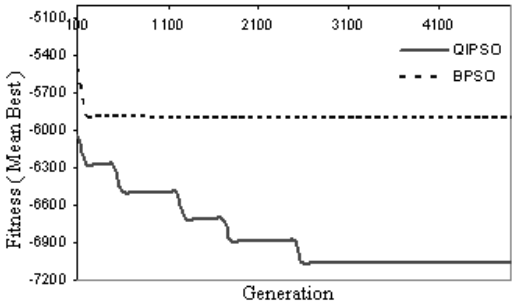

Fig. 5 Convergence graph for the function f_8

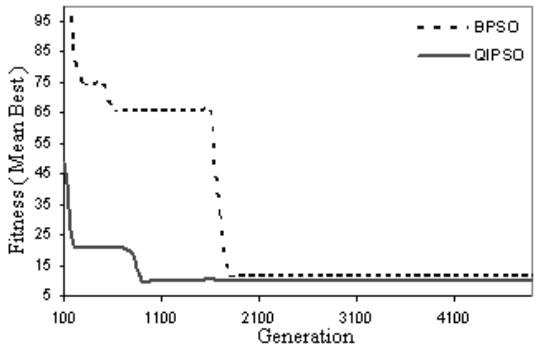

Fig. 6 Convergence graph for the function f_9

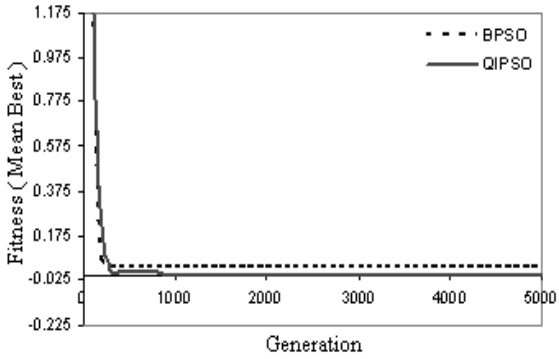

Fig. 7 Convergence graph for the function f_{11}

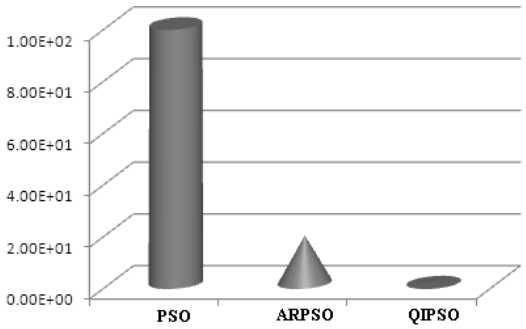

Fig. 8 Comparison of objective function values for the function f_3

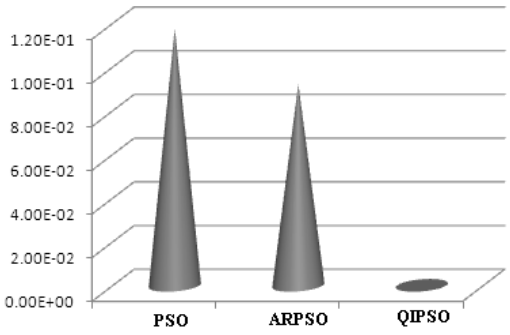

Fig. 9 Comparison of objective function values for the function f_{12}

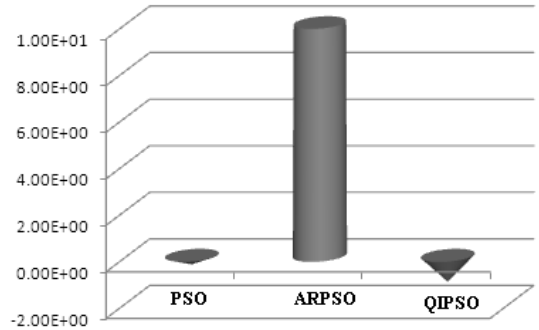

Fig. 10 Comparison of objective function values for the function f_{13}

B. Real life Problems

An optimization algorithm is said to be successful only if it is capable of solving real life problems, which may or may not be assisted with constrained, also along with the benchmark problems. Therefore in order to check the efficiency of the QIPSO algorithm, we tested them on five engineering design problems common in the field of electrical engineering. These are; Dynamic Power schedule problem [19], Static power scheduling problem [19], electric network optimization problem [20], Cost Minimization of Transformer Design [19] and Transistor Modeling [21].

A brief description of the real life problems along with mathematical models is given below.

1) Dynamic power scheduling problem

This problem is a representation of the problem of scheduling three generators to meet the demand for power over a period of time. The variable x_{3k+i} denotes the output from the i^{th} generator at time $t^{(k)}$. The constraints in the problem are upper and lower limits on the power available from each generator, bounds on the amount by which the output from a generator can change from time $t^{(k)}$ to $t^{(k+1)}$, and the condition that the at each time $t^{(k)}$ the power generated must at least satisfy the demand. The mathematical model of this power scheduling problem is,

Minimize

$$f(x)=\sum_{k=0}^{4}(2.3x_{3k+1}+0.0001x_{3k+1}^2+1.7x_{3k+2}+$$
$$0.0001x_{3k+2}^2+2.2x_{3k+3}+0.00015x_{3k+3}^2)$$

Subject to:

$$-7\le x_1-15\le 6,\ -7\le x_{3k+1}-x_{3k-2}\le 6 \quad k=1,.....,4$$
$$-7\le x_2-50\le 7,\ -7\le x_{3k+2}-x_{3k-1}\le 7 \quad k=1,.....,4$$
$$-7\le x_3-10\le 6,\ -7\le x_{3k+3}-x_{3k}\le 6 \quad k=1,.....,4$$
$$x_1+x_2+x_3\ge 60,\ x_4+x_5+x_6\ge 50,\ x_7+x_8+x_9\ge 70$$
$$x_{10}+x_{11}+x_{12}\ge 85,\ x_{13}+x_{14}+x_{15}\ge 100$$
$$0\le x_{3k+1}\le 90 \quad k=1,.....,4$$
$$0\le x_{3k+2}\le 120 \quad k=1,.....,4$$
$$0\le x_{3k+3}\le 60 \quad k=1,.....,4$$

2) Static power scheduling problem

In this problem the decision variables x_1 and x_2 are the real power outputs from two generators; x_3 and x_4 are the reactive power outputs; x_5, x_6 and x_7 are voltage magnitudes at three nodes of an electrical network and x_8 and x_9 are voltage phase angles at two of these nodes. The constraints other than the bounds are the real and reactive power balance equations, stating that the power flowing into a node must balance the power flowing out. The mathematical model is,

Minimize

$$f(x)=3000x_1+1000x_1^3+2000x2+666.667x_2^3$$

Subject to:

$$0.4-x_1+2Cx_5{}^2+x_5x_6[D\sin(-x_8)-C\cos(-x_8)]$$
$$+x_5x_7[D\sin(-x_9)-C\cos(-x_9)]=0$$
$$0.4-x_2+2Cx_6{}^2+x_5x_6[D\sin(x_8)-C\cos(x_8)]$$
$$+x_6x_7[D\sin(x_8-x_9)-C\cos(x_8-x_9)]=0$$
$$0.8+2Cx_7{}^2+x_5x_7[D\sin(x_9)-C\cos(x_9)]$$
$$+x_6x_7[D\sin(x_9-x_8)-C\cos(x_9-x_8)]=0$$
$$0.2-x_3+2Dx_5{}^2-x_5x_6[C\sin(-x_8)+D\cos(-x_8)]$$
$$-x_5x_7[C\sin(-x_9)+D\cos(-x_9)]=0$$
$$0.2-x_4+2Dx_6{}^2-x_5x_6[C\sin(x_8)+D\cos(x_8)]$$
$$-x_6x_7[C\sin(x_8-x_9)+D\cos(x_8-x_9)]=0$$
$$-0.337+2Dx_7{}^2-x_5x_7[C\sin(x_9)+D\cos(x_9)]$$
$$-x_6x_7[C\sin(x_9-x_8)+D\cos(x_9-x_8)]=0$$
$$x_i\ge 0 \quad i=1,2.$$
$$1.0909\ge x_i\ge 0.90909 \quad i=5,6,7.$$

Where $C=\sin(0.25)48.4/50.176$

$D=\cos(0.25)48.4/50.176$

3) Electric Network optimization

The mathematical model of the Electric Network Optimization problem is given by,

Minimize

$$f(x)=f_1(x_1)+f_2(x_2)$$

Constraints:

$$f_1(x_1)=\begin{cases}31x_1 & 0\le x_1\le 300\\ 30x_1 & 300\le x_1\le 400\end{cases}$$

$$f_2(x_2)=\begin{cases}28x_2 & 0\le x_2\le 100\\ 29x_2 & 100\le x_2\le 200\\ 30x_2 & 200\le x_2\le 1000\end{cases}$$

$$x_1=300-\frac{x_3x_4}{131.078}\cos(1.48477-x_6)$$
$$+\frac{0.90798x_3{}^2}{131.078}\cos(1.47588)$$

$$x_2=-\frac{x_3x_4}{131.078}\cos(1.48477+x_6)$$
$$+\frac{0.90798x_4{}^2}{131.078}\cos(1.47588)$$

$$x_5 = -\frac{x_3 x_4}{131.078}\sin(1.48477 + x_6) + \frac{0.90798 {x_4}^2}{131.078}\sin(1.47588)$$

$$200 - \frac{x_3 x_4}{131.078}\sin(1.48477 - x_6) + \frac{0.90798 {x_3}^2}{131.078}\sin(1.47588) = 0$$

$0 \le x_1 \le 400,\ 0 \le x_2 \le 1000,\ 340 \le x_3 \le 420$

$340 \le x_4 \le 420,\ -1000 \le x_5 \le 1000,\ 0 \le x_6 \le 0.5236.$

4) Cost Minimization of Transformer design

The objective function represents the worth of the transformer, including the operating cost, and the constraints refer to the rating of the transformer and the allowable transmission loss. The decision variables x_1, x_2, x_3 and x4 are physical dimensions of winding and core and the variables x_5, x_6 are magnetic flux density and current density respectively. The mathematical model of this problem is given by:

Minimize

$$f = 0.0204x_1x_4(x_1 + x_2 + x_3) + 0.0187x_2x_3(x_1 + 1.57x_2 + x_4) + 0.0607x_1x_4{x_5}^2(x_1 + x_2 + x_3) + 0.0437x_2x_3{x_6}^2(x_1 + 1.57x_2 + x_4)$$

Subject to:

$$x_1x_2x_3x_4x_5x_6 \ge 2.07 \times 10^3$$

$$1 - 0.00062x_1x_4{x_5}^2(x_1 + x_2 + x_3) - 0.00058x_2x_3{x_6}^2(x_1 + 1.57x_2 + x_4) \ge 0$$

$x_i \ge 0 \quad (i = 1,...,6).$

5) Transistor Modeling

The mathematical model of the transistor design is given by,

Minimize $f(x) = \gamma^2 + \sum_{k=1}^{4}({\alpha_k}^2 + {\beta_k}^2)$

Where

$$\alpha_k = (1 - x_1x_2)x_3\{\exp[x_5(g_{1k} - g_{3k}x_7 \times 10^{-3} - g_{5k}x_8 \times 10^{-3})] - 1\}g_{5k} + g_{4k}x_2$$

$$\beta_k = (1 - x_1x_2)x_4\{\exp[x_6(g_{1k} - g_{2k} - g_{3k}x_7 \times 10^{-3} + g_{4k}x_9 \times 10^{-3})] - 1\}g_{5k}x_1 + g_{4k}.$$

$$\gamma = x_1x_3 - x_2x_4$$

Subject to: $x_i \ge 0$

And the numerical constants g_{ik} are given by the matrix

0.485	0.752	0.869	0.982
0.369	1.254	0.703	1.455
5.2095	10.0677	22.9274	20.2153
23.3037	101.779	111.461	191.267
28.5132	111.8467	134.3884	211.4823

This objective function provides a least-sum-of-squares approach to the solution of a set of nine simultaneous nonlinear equations, which arise in the context of transistor modeling.

C. Simulation Results of Real life Problems

Parameter settings for all the real life problems are same as that of the test functions. Computational time for all the algorithms is more or less similar with marginal difference and is therefore not reported. The results of all the problems are given in Table VI.

The first RLP is a representation of the problem of scheduling three generators to meet the demand for power over a period of time. This problem has 15 decision variables, 35 inequality constraints and 30 boundary constraints. The simulation result shown that QIPSO gave a better performance than BPSO algorithm. The second RLP is a static power scheduling problem, it has 9 decision variables and 6 equality constraints. Here we would like to mention that while solving equality constraints, we have changed them to inequality with an accuracy of 10^{-4}. This is to say that a constraint $h_i = 0$, is changed into an in equation as $|h_i| <= 10^{-4}$. Again QIPSO algorithm gave a better result than BPSO algorithm. Like first and second RLP the QIPSO algorithm gave a better performance for all the remaining problems.

V. Conclusion

The present article presents a simple and modified version of the Basic PSO by inducing a crossover component in it. The crossover component is added to enhance the performance of the conventional PSO without disturbing the inherent features of BPSO. The QI crossover taken in this paper is a nonlinear operator which uses three particles of the swarm population to generate a new candidate solution. The presence of global best swarm particle makes it behave like a greedy search algorithm which explores its neighboring area in search of more potential candidates.

The integrity of QIPSO is validated by testing it on a set of small, medium and large sized unconstrained test problems with dimensions 30, 50 and 100. The results are compared with conventional PSO and ARPSO, which is a modified version of conventional PSO. The efficiency of QIPSO is further tested by applying it on five real life problems which are common in the field of electrical engineering. All the real problems have constraints associated with them which are dealt with the penalty function approach. Numerical results of real as well as test problems show the robustness and efficiency of QIPSO algorithm. Further investigations include theoretical analysis of QI operator and also the application of QIPSO for solving high dimension real life and test problems.

Acknowledgment

The second author would like to thank All India Council for Technical Education (AICTE), Government of India for giving financial support to her research work.

TABLE VI NUMERICAL RESULTS OF REAL LIFE PROBLEMS

Dynamic Power Scheduling problem				
Algorithm	Best	Average	Worst	Std
BPSO	666.6352	717.552687	802.85215	25.031686
QIPSO	**664.015**	703.223	742.139	17.5405
Static Power Scheduling				
Algorithm	Best	Average	Worst	Std
BPSO	5061.01219	5154.529748	5221.47662	42.762777
QIPSO	**5049.33**	5109.37	5136.04	37.7784
Electric Network Optimization				
Algorithm	Best	Average	Worst	Std
BPSO	8873.01753	9084.32555	9307.9852	141.867249
QIPSO	**8865.98**	9025.07	9234.88	106.193
Cost Minimization of Transformer Design				
BPSO	87.0734	91.8331	139.243	11.5574
QIPSO	**86.6171**	87.323	88.4159	0.598072
Transistor Modeling				
BPSO	0.069569	0.076523	0.09563	1.087695
QIPSO	**0.066326**	0.06978	0.07236	0.987302

REFERENCES

[1] Angeline P. J., "Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Difference," The 7th Annual Conference on Evolutionary Programming, San Diego,USA,(1998).

[2] H. Liu, A. Abraham and W. Zhang: A Fuzzy Adaptive Turbulent Particle Swarm Optimization. International Journal of Innovative Computing and Applications, Volume 1, Issue 1, 2007, pp. 39-47.

[3] J. Robinson, S. Sinton, and Y. Rahmat-Samii., "Particle Swarm, Genetic Algorithm, and Their Hybrids: Optimization of a Profiled Corrugated Horn Antenna", In Proc. of the IEEE Antennas and Propagation Society International Symposium and URSI national radio Science meeting, Vol. 1 (2002), pp. 314 – 317.

[4] Y. Shi and R. A. Krohling, "Co-Evolutionary Particle Swarm Optimization to Solve Min-Max Problems", In Proc. of the IEEE Congress on Evolutionary Computation, Vol.2 (2002), pp. 1682 – 1687.

[5] X. Shi, J. Hao, J. Zhou, and Y. Liang, "Hybrid Evolutionary Algorithms Based on PSO and GA", In Proc. of the IEEE Congrss on Evolutionary Computation, Vol. 4 (2003), pp. 2393 – 2399.

[6] X. Hu, R. C. Eberhart, and Y. Shi, "Swarm Intelligence for Permutation Optimization: A Case Study on n-Queens problem", In Proc. of IEEE Swarm Intelligence Symposium (2003), pp. 243 – 246.

[7] C-F. Juang, "A hybrid of genetic Algorithm and Particle Swarm Optimization for Recuurent Network Design", IEEE Trans. On Systems, Man, and Cybernetics – Part B: Cybernetics, 34(2), (2003) , pp. 997 – 1006.

[8] Kennedy, J. and Eberhart, R., "Particle Swarm Optimization", IEEE Int. Conf. on neural networks (Perth, Australia), IEEE service Center, Piscataway, NJ. (1995), pp. 1942-1948.

[9] B. R. Secrest and G. B. Lamont. Visualizing Particle Swarm Optimization – Gaussian Particle Swarm Optimization. In *Proceedings of the IEEE Swarm Intelligence Symposium*, (2003), pp. 198 – 204.

[10] Millie Pant, Radha Thangaraj, "A New Particle Swarm Optimization with Quadratic Crossover", Int. Conf. on Advanced Computing and Communications (ADCOM'07), India, IEEE Computer Society Press, pp. 81 – 86, 2007.

[11] J.Riget and J.S. Vesterstrom, "A diversity-guided particle swarm optimizer – the arPSO," Technical report, EVAlife, Dept. of Computer Science, University of Aarhus, Denmark,(2002).

[12] Kennedy, J., "The Particle Swarm: Social Adaptation of Knowledge," *IEEE International Conference on Evolutionary Computation* (Indianapolis, Indiana), *IEEE* Service Center, Piscataway, NJ,(1997), pg.303-308.

[13] R.C. Eberhart, Y. Shi, "Particle Swarm Optimization: developments, Applications and Resources," *IEEE International Conference on Evolutionary Computation*, (2001), pg. 81 -86.

[14] Shi, Y. H., Eberhart, R. C., "A Modified Particle Swarm Optimizer," *IEEE International Conference on Evolutionary Computation*, Anchorage, Alaska, (1998), pg. 69 – 73.

[15] C. Mohan and K. Shanker, "A Controlled Random Search Technique For Global Optimization using Quadratic Approximation", Asia-Pacific Journal of Operational Research, Vol. 11, pp. 93-101, 1994.

[16] M. M. Ali and A. Torn, "Population Set Based Global Optimization Algorithms: Some Modifications and Numerical Studies", www.ima.umn.edu/preprints, 2003.

[17] Engelbrecht, A. P., "*Fundamentals of Computational Swarm Intelligence*", John Wiley & Sons Ltd., (2005).

[18] J. Vesterstrom, R. Thomsen, "A Comparative study of Differential Evolution, Particle Swarm optimization, and Evolutionary Algorithms on Numerical Benchmark Problems," in *Proc. IEEE Congr. Evolutionary Computation*, Portland, OR, Jun. 20 – 23, (2004), pg. 1980 – 1987.

[19] M. C. B-Biggs, "A Numerical Comparison between Two Approaches to the Nonlinear Programming Problem", In: L. C. W. Dixon and G. P. Szego, eds., Towards Global Optimization 2, Amsterdam, Holland: North Holland Publishing Company, pp. 293 – 312, 1978.

[20] W. L. Price, "A Controlled Random Search Procedure for Global Optimization", In: L. C. W. Dixon and G. P. Szego, eds., Towards Global Optimization 2, Amsterdam, Holland: North Holland Publishing Company, pp. 71 – 84, 1978.

[21] D. M. Himmelblau, "Applied Non-Linear programming", New York: McGraw-Hill, 1972.

Millie Pant received the Bachelor's and Master's degrees from CCS University, Meerut in the years 1997 and 1999 respectively and Ph. D degree in 2003 from the Indian Institute of Technology Roorkee, India.

Dr. M. Pant is currently with Indian Institute of Technology Roorkee, India, where she is a Senior Lecturer in the Department of Paper Technology. Her special fields of interests include Numerical Optimization, Evolutionary Algorithms, Swarm Intelligence and Parallel Computing.

Radha Thangaraj received the Bachelor's, Master in Science and Master in Philosophy degrees in Mathematics from Manonmaniam Sundaranar University, Tirunelveli, India in 2001, 2003 and 2004 respectively.

Currently she is working towards the Ph. D degree at Department of Paper Technology, Indian Institute of Technology Roorkee, India. Her field of interest includes Evolutionary Computation and its Applications in Engineering.

V. P. Singh received the Bachelor's degree from Meerut University, India in the year 1970, Master's and Ph. D degrees in Applied mathematics from the University of Roorkee (now, Indian Institute of Technology Roorkee), India, in 1972, and 1978 respectively.

Prof. V. P. Singh is currently with Indian Institute of Technology Roorkee, India, where he is a Professor in the Department of Paper Technology. His special fields of interests include Applied and Industrial Mathematics, Mathematical Modeling of Pulp washing problems.