

## Two modified differential evolution algorithms and their applications to engineering design problems

Musrrat Ali\*, Millie Pant, V. P. Singh

Department of Paper Technology, Indian Institute of Technology Roorkee, Saharanpur Campus, Saharanpur 247001, India

(Received March 23 2009, Accepted September 25 2009)

**Abstract.** Differential Evolution (DE) is a stochastic, population based search technique, which can be classified as an Evolutionary Algorithm (EA) using the concepts of selection crossover and reproduction to guide the search. It has emerged as a powerful tool for solving optimization problems in the past few years. However, the convergence rate of DE still does not meet all the requirements, and attempts to speed up differential evolution are considered necessary. In order to improve the performance of DE, We propose two versions of (DE) called Differential Evolution with Parent Centric Crossover (DEPCX) and Differential Evolution with probabilistic Parent Centric Crossover (Pro. DEPCX). The proposed algorithms are validated on a test bed of seven real life, nonlinear engineering design problems and numerical results are compared with original differential evolution (DE). Empirical analysis of the results indicates that the proposed schemes enhance the performance of basic DE in terms of convergence rate without compromising with the quality of solution.

**Keywords:** stochastic optimization, differential evolution, mutation, crossover

### 1 Introduction

Optimization problems encountered in the various fields of engineering often turn out to be nonlinear in nature which may further be classified broadly as unimodal and multimodal problems. Unimodal problems have only one absolute optimum value known as the global optimal value while multimodal problems have several local and global optima. Ideally, the user is interested in determining the global optimal solution because it gives the true objective function value. Most of the traditional methods available in literature are gradient based search techniques which require auxiliary properties like differentiability and continuity of the objective function and also they do not guarantee to obtain the global optimum.

In the past few years scientist and researchers have laid emphasis on nontraditional optimization techniques which can obtain the global optimum solution and are also independent of the nature of the problems. Out these the most popular techniques are population based search heuristics like Evolutionary Algorithms (EA).

These techniques work with a population of solutions rather than a single solution, which makes them different from optimization methods, such as hill-climbing<sup>[19]</sup> and simulated annealing<sup>[16]</sup>. One of the reasons of the success of EAs is their population based strategy which prevents them from getting trapped in a local optimal solution and consequently increases their probability of finding a global optimal solution. Thus, EAs can be viewed as global optimization algorithms. Some frequently used EAs include Evolutionary Programming (EP)<sup>[13]</sup>, Evolution Strategies (ES)<sup>[6]</sup> and Genetic Algorithms (GA)<sup>[15]</sup>.

DE, proposed by Storn and Price<sup>[26]</sup>, is comparatively a newer addition to the class of EAs. DE is similar to GAs in the sense that it uses same evolutionary operators like mutation and crossover for guiding the

\* Corresponding author. Tel.: +91-0132-2714356, +91-9720102630. fax: +91-0132-2714011.  
E-mail address: musrrat.iitr@gmail.com.

particles towards the optimum solution. Nevertheless, it's the application of these operators that makes DE different from GA. The main difference between GAs and DE is that; in GAs, mutation is the result of small perturbations to the genes of an individual while in DE mutation is the result of arithmetic combinations of individuals. Also in DE, mutation plays a prominent role whereas, in GA, crossover is the major operator. At the beginning of the evolution process, the mutation operator of DE favors exploration. As evolution progresses, the mutation operator favors exploitation. Hence, DE automatically adapts the mutation increments (i.e. search step) to the best value based on the stage of the evolutionary process. Mutation in DE is therefore not based on a predefined probability density function.

DE is easy to implement, requires little parameter tuning and exhibits fast convergence. It has been successfully applied to solve a wide range of optimization problems such as clustering<sup>[23]</sup>, unsupervised image classification<sup>[20]</sup>, digital filter design<sup>[26]</sup>, optimization of non-linear functions<sup>[5]</sup>, global optimization of non-linear chemical engineering processes<sup>[3]</sup>, and multi-objective optimization<sup>[2]</sup> etc.

Although, DE has given good results in most of the real life and test problems, researchers have observed that its convergence rate do not meet the desired expectations in some of the cases. In order to overcome this drawback of DE, several modifications have been suggested in literature [1, 8, 10, 21, 25, 27, 28]. To further continue the research in the direction to improve the convergence rate of DE without disturbing the basic structure of DE and without compromising with the quality of solutions the authors propose two modified versions of basic DE called Differential Evolution with Parent Centric Crossover (DEPCX) and probabilistic DEPCX. Both the schemes are inspired by the parent centric crossover operator (PCX) used in GA and are embedded in the DE/1/rand/bin version of basic DE, which is perhaps the most commonly used version of DE.

Here we would like to mention we have presented a preliminary version of this work as a conference paper<sup>[22]</sup> in which the DEPCX algorithm is tested on a set of few benchmark problems. However, in this paper along with DEPCX we also present another algorithm called Pro. DEPCX and test both the algorithms on a set of 7 real life engineering design problems. The only structural difference between the two algorithms is the manner in which mutation operator is applied. DEPCX uses parent centric approach to generate new solution vectors while Pro. DEPCX, stochastically uses the parent centric mutation operator along with the basic DE mutation operation. These modifications provide a measure to tune the balance between the convergence rate and the robustness of the algorithm and accelerate the convergence velocity of DE algorithm locally without compromising with the quality of solution.

Remaining of the paper is organized as follows. In section 2, we give a brief description of DE. In section 3, the PCX operator is described. In section 4, we explain the proposed DEPCX algorithm. Real life problems are given in section 5. Section 6 deals with experimental settings and parameter selection. Section 7 makes comparison of algorithms. Finally the conclusions based on the present study are drawn in section 8.

## 2 Differential Evolution (DE)

In this section, we briefly describe the classical DE algorithm as given by Storn and Price. DE attempts to replace each point in population set  $S$  with a new better point. Therefore, in each generation,  $NP$  (population size) competitions are held to determine the members of  $S$  for the next generation. The  $i$ th ( $i = 1, 2, \dots, NP$ ) competition is held to replace  $X_{i,G}$  in  $S$ . Considering  $X_{i,G}$  as the target point, a trial point  $U_{i,G+1}$  is found from two points (parents), the point  $X_{i,G}$ , i.e., the target point and the mutated point  $V_{i,G+1}$  are determined by the mutation operation. In its mutation phase, DE randomly selects three distinct points from the population. The  $i$ th perturbed individual,  $V_{i,G+1}$ , is therefore generated based on the three chosen individuals as follows:

$$V_{i,G+1} = X_{r_3,G} + F * (X_{r_1,G} - X_{r_2,G}). \quad (1)$$

Where,  $i = 1, \dots, NP, r_1, r_2, r_3 \in \{1, \dots, NP\}$  are randomly selected such that  $r_1 \neq r_2 \neq r_3 \neq i$ . Scaling factor  $F$  ( $F \in [0, 1.2]$ ) is a control parameter of the DE algorithm. The perturbed individual,  $V_{i,G+1} = (V_{1,i,G+1}, \dots, V_{D,i,G+1})$ , The perturbed individual,  $X_{i,G} = (x_{1,i,G}, \dots, x_{D,i,G})$ , are then subject to the crossover operation, that finally generates the population of candidates, or "trial" vectors,  $U_{i,G+1} = (u_{1,i,G}, \dots, u_{D,i,G+1})$ , as follows:

$$u_{j,i,G+1} = \begin{cases} v_{j,i,G+1} & \text{if } \text{rand}_j \leq C_r \vee j = k \\ x_{j,i,G} & \text{otherwise} \end{cases} \quad (2)$$

Where  $j, k \in \{1, \dots, D\}$ ,  $k$  is a random parameter index, chosen once for each  $i$ ,  $C_r$  is crossover factor  $\in [0, 1]$ .

The population for the next generation is selected from the individuals in current population and its corresponding trial vector according to the following rule:

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & \text{if } f(U_{i,G+1}) \leq f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \quad (3)$$

Thus, each individual of the temporary (trial) population is compared with its counterpart in the current population. The one with the lower objective function value survives the tournament selection and go to the next generation. As a result, all the individuals of the next generation are as good as or better than their counterparts in the current generation. A notable point in DE's selection scheme is that a trial vector is not compared against all the individuals in the current generation, but only against one individual, its counterpart, in the current generation.

### 3 Parent centric crossover

The concept of parent centric approach is not new to the field of evolutionary algorithms and has been used in Genetic Algorithms. Several versions of parent centric crossover operators are available in the literature of GAs. A detailed study of these versions may be found in [14]. However, to the best of our knowledge, it has not been applied to field of differential evolution. The main idea in this approach is to produce new candidate solution vectors (offspring) in the vicinity of the original solution vectors (parents). In the present study we considered the Parent Centric Crossover (PCX) operator given by Deb<sup>[11, 12]</sup>. The PCX operator given by Deb is a multiparent operator consisting of more than one parent. Its working may be described as follows:

In the beginning  $\mu = n + 1$  parents are selected, where  $n$  denotes the number of decision variables. From the selected parents, the mean vector  $\vec{g}$  is computed and one parent  $\vec{X}_{p,G}$  is selected at random to generate offspring. Direction vector  $d_p = \vec{X}_{p,G} - \vec{g}$  is calculated with respect to the selected parent. From the remaining  $(\mu - 1)$  parents perpendicular distances  $D_i$  to the line  $\vec{d}_p$  are computed and their average  $\bar{D}$  is found. Finally the offspring is created as follows:

$$V_{i,G+1} = \vec{X}_{p,G} + w_\zeta d_p + \sum_{i=1, i \neq p}^{\mu} w_\eta \bar{D} \vec{e}_t. \quad (4)$$

Where  $\vec{e}_i$  are the  $(\mu - 1)$  orthonormal bases that span the sub space perpendicular to  $d_p$ . The parameters  $w_\zeta$  and  $w_\eta$  are zero mean normally distributed variables with variances  $\sigma_\zeta^2$  and  $\sigma_\eta^2$  respectively.

The advantage of parent centric approach is that it increases the probability of the offspring to be generated in the vicinity of the parent. It is self adaptive type approach where a new solution vector keeps moving towards the optimum systematically. If the parents are located close to each other, the offspring are located densely around the parents whereas if the parents are located far from each other, the offspring are sparsely located. Since the parents are selected at random and offspring are generated around them, it also maintains the diversity of the population. The use of parent centric approach in GA has reportedly given good performance in comparison to other stochastic algorithms.

### 4 Proposed algorithms

The main purpose of this research is to observe the effect of parent centric approach in DE, without disturbing the basic structure of DE. As discussed in section 1 and section 2, mutation is the main phase of DE therefore we applied the parent centric approach to generate the mutant vector in DE. This is in contrast to GA where parent centric approach is used as a crossover operator. Based on the use of parent centric operator we propose two algorithms which are described in subsections 4.1 and 4.2 respectively.

#### 4.1 Differential evolution with parent centric crossover (DEPCX)

In the proposed algorithm, instead of selecting  $(n+1)$  parents, we took only three parent vectors,  $\vec{a} \neq \vec{b} \neq \vec{c}$  such that  $\vec{a} = \vec{X}_{\text{best},G}$  and  $\vec{b}$  and  $\vec{c}$  are randomly selected from the rest of the population.  $\vec{X}_{\text{best},G}$  represent the point with best fitness function value. The reason for selecting the point with best fitness function value is to create an elite type model so that the probability of getting an offspring near the best parent increases. Another striking difference between PCX operator of Deb and proposed algorithms is that Deb suggested the generation of  $\lambda (> 1)$  offspring around any of the chosen parents. This suggestion is made to preserve the diversity of the population and to increase the chances of getting better candidates for the next generation. On the other hand, it is quite natural to think that such a scheme will take more time in comparison to the scheme in which only one offspring is generated. In our proposed versions we have considered the generation of a single offspring in the vicinity of the best parent vector. In the proposed algorithms the mutant vector is generated as follows:

$$V_{i,G+1} = \vec{X}_{\text{best},G} + w_{\zeta} d_p + \sum_{i=1, i \neq p}^{\mu} w_{\eta} \bar{D} \vec{e}_t. \quad (5)$$

The only difference between Eq. (4) and Eq. (5) is that  $\vec{X}_{p,G}$  is replaced by  $\vec{X}_{\text{best},G}$ . In the next phase which is reproduction, a new candidate for the next generation is produced according to Eq. (2). As discussed earlier in section 3, the newly generated offspring will have the genes of both the parents and since one of the parents is the one having the best fitness function value found so far, the probability of the offspring getting good genes also increases. The pseudo code of DEPCX algorithm is given here:

- (1) Randomly initialize the population  $P_G$ .
- (2) REPEAT Until search converged.
  - (3) REPEAT for each individual  $i \in P_G$ .
    - (4) Mutation operation by Eq. (5).
    - (5) Crossover operation.
    - (6) Evaluation of the function.
    - (7) Selection.
  - (8) ENDREPEAT.
- (9) ENDREPEAT.

#### 4.2 Differential evolution with probabilistic parent centric crossover (Pro. DEPCX)

Differential Evolution with probabilistic Parent Centric Crossover (Pro. DEPCX) also use the parent centric operator (PCX) but in different manner. The main difference between DEPCX and Pro. DEPCX is that in DEPCX, Eq. (5) simply replaces Eq. (1) whereas in Pro. DEPCX Eq. (1) is applied stochastically along with Eq. (5). A predefined parameter called parent centric probability operator denoted as PD is taken. This operator determines probabilistically whether to apply the classical DE mutant vector or parent centric mutant vector. Pseudo codes of the proposed version are given as follows:

- (1) Randomly initialize the population  $P_G$ .
- (2) REPEAT Until search converged.
  - (3) REPEAT for each individual  $i \in P_G$ .
    - (4) Mutation operation.
      - (4.1) perform mutation using Eq. (5) with probability  $P_D$ .
      - (4.2) perform mutation using Eq. (5) with probability  $1 - P_D$ .
    - (5) Crossover operation.
    - (6) Evaluation of the function.
    - (7) Selection.
  - (8) ENDREPEAT.
- (9) ENDREPEAT.

## 5 Real life problems

The performance of proposed algorithms is evaluated on a test bed of seven unconstrained real life problems that are common in various fields of engineering designs. These are: (F1) Gas Transmission compressor design (F2) Optimal capacity of gas production facility (F3) Design of a gear train (F4) optimal thermohydraulic performance of an artificially roughened air heater (F5) Frequency modulation sound parameter identification (F6) The spread spectrum radar poly-phase code design problem (F7) The Lennard-Jones atomic cluster problem. The dimensions of these problems vary from 2 to 20. The sixth problem F6 is tested for two sets of dimensions; 10 and 20 which are named as F6 (a) and F6 (b) respectively. All the problems considered in the present study are highly nonlinear in nature. Mathematical models of the problems are given below:

(1) F1: Gas transmission compressor design<sup>[7]</sup>.

$$\text{Min } f(x) = 8.61 \times 10^5 \times x_1^{1/2} x_2 x_3^{-2/3} (x_2 - 1)^{-1/2} + 3.69 \times 10^4 \times x_3 + 7.72 \times 10^8 \times x_1^{-1} x_2^{0.219} - 765.43 \times 10^6 \times x_1^{-1}.$$

$$\text{Bounds: } 10 \leq x_1 \leq 55, 1.1 \leq x_2 \leq 2, 10 \leq x_3 \leq 40.$$

(2) F2: Optimal capacity of gas production facilities<sup>[7]</sup>.

$$\text{Min } f(x) = 61.8 + 5.72x_1 + 0.2623 \left[ (40 - x_1) \ln \left( \frac{x_2}{200} \right) \right]^{-0.85} + 0.087(40 - x_1) \ln \left( \frac{x_2}{200} \right) + 700.23x_2^{-0.75}.$$

$$\text{Subject to: } x_1 \geq 17.5, x_2 \geq 200.$$

$$\text{Bounds: } 17.5 \leq x_1 \leq 40, 300 \leq x_2 \leq 600.$$

(3) F3: Design of a gear train<sup>[4]</sup>.

This is a common problem in the field of mechanical engineering. The gear train is to be designed such that the gear ratio is as close as possible to 1/6.931. For each gear the number of teeth must be between 12 and 60. Since the number of teeth is to be an integer, the variables must be integers. The mathematical model of gear train design is given by:

$$\text{Min } f = \left\{ \frac{1}{6.931} - \frac{T_d T_b}{T_a T_f} \right\}^2 = \left\{ \frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4} \right\}^2.$$

$$\text{Subject to: } 12 \leq x_i \leq 60, i = 1, 2, 3, 4.$$

$$[x_1, x_2, x_3, x_4] = [T_d, T_b, T_a, T_f], x_i's \text{ should be integers.}$$

Where  $T_a, T_b, T_d$  and  $T_f$  are the number of teeth on gears A, B, D and F respectively. The design of a compound gear train is shown in Fig. 1.

(4) F4: Optimal thermohydraulic performance of an artificially roughened air heater<sup>[24]</sup>.

$$\text{Max } L = 2.51 \ln e^+ + 5.5 - 0.1R_M - G_H.$$

$$\text{Where } R_M = 0.95x_2^{0.53}; GH = 4.5(e^+)^{0.28}(0.7)^{0.57}; e^+ = x_1x_3(\bar{f}/2)^{1/2}; \bar{f} = (f_s + f_r)/2;$$

$$f_s = 0.079x_3^{-0.25}; f_r = 2(0.95x_3^{0.53} + 2.5 \ln(1/2x_1)^2 - 3.75)^{-2}.$$

$$\text{Bounds: } 0.02 \leq x_i \leq 0.8, 10 \leq x_i \leq 40, 3000 \leq x_i \leq 20000.$$

(5) F5: Frequency modulation sound parameter identification<sup>[18]</sup>.

The problem is to satisfy six parameters  $a_1, w_1, a_2, w_2, a_3, w_3$  of the frequency modulation sound model is given below:

$$y(t) = a_1 \times \sin(w_1 \times t \times \theta + a_2 \times \sin(w_2 \times t \times \theta + a_3 \times \sin(w_3 \times t \times \theta))).$$

With  $\theta = (2.\pi/100)$ . The fitness function is defined as the sum of square error between the evolved data and the model data as follows:  $f(a_1, w_1, a_2, w_2, a_3, w_3) = \sum_{t=0}^{100} (y(t) - y_0(t))^2$ . Where the model data are given by the following equation:

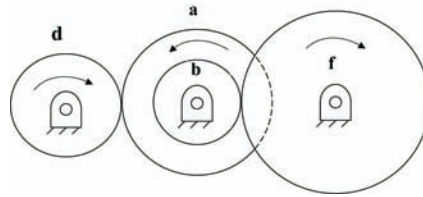


Fig. 1. Compound gear train

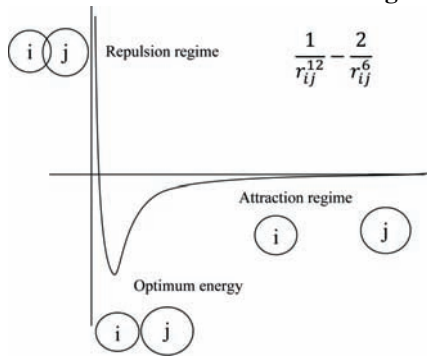


Fig. 2. LJ potential of a single pair of atoms

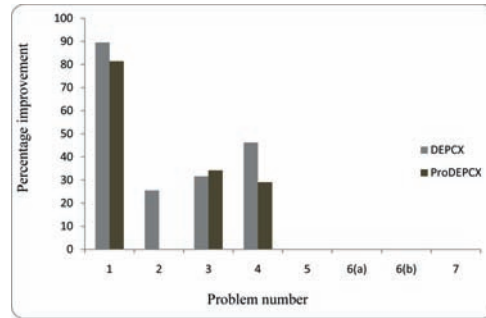


Fig. 3. Percentage improvement of DEPCX and ProDEPCX with respect to basic DE

$$y_0(t) = 1.0 \times \sin(5.0 \times t \times \theta + 1.5 \times \sin(4.8 \times t \times \theta + 2.0 \times \sin(4.9 \times t \times \theta)))$$

Each parameter is in the range  $(-6.4, 6.35)$ . This is highly complex multimodal problem with minimum value zero.

(6) F6: The spread spectrum radar poly-phase code design problem<sup>[17]</sup>.

A famous problem of optimal design arises in the field of spread spectrum radar poly-phase codes. Such a problem is very well suited for the application of global optimization algorithms like DE. The problem can be formally stated as:

$$\text{Min } f(X) = \max\{f_1(X), \dots, f_{2m}(X)\}.$$

Where  $X = \{(x_1, \dots, x_n) \in R^n | 0 \leq x_j \leq 2\pi, j = 1, \dots, n\}$  and  $m = 2n - 1$ , with:

$$f_{2i-1}(x) = \sum_{j=i}^n \cos\left(\sum_{k=|2i-j-1|+1}^j x_k\right) \quad i = 1, 2, \dots, n;$$

$$f_{2i}(x) = 0.5 + \sum_{j=i+1}^n \cos\left(\sum_{k=|2i-j-1|+1}^j x_k\right) \quad i = 1, 2, \dots, n - 1;$$

$$f_{m+i}(X) = -f_i(X), i = 1, 2, \dots, m.$$

Here the objective is to minimize the module of the biggest among the samples of the so-called autocorrelation function which is related to the complex envelope of the compressed radar pulse at the optimal receiver output, while the variables represent symmetrized phase differences. This problem belongs to the class of continuous min-max global optimization problems. They are characterized by the fact that the objective function is piecewise smooth.

(7) F7: The lennard-jones atomic cluster problem<sup>[9]</sup>.

The Lennard-Jones (LJ) model of inert gas cluster has been investigated intensively as a challenging testing ground for global optimization algorithms. The LJ problem can be formulated as follows: Let  $N$

atoms be given in three dimensional space. Let  $x^i = (x_1^i, x_2^i, x_3^i)^T$  represent the coordinates of atom  $i$ . let  $\mathbf{X} = ((x^1)^T, \dots, (x^N)^T)^T$  the LJ potential energy function  $v(r_{ij})$  of a pair of atoms  $(i, j)$  is given by  $v(r_{ij}) = \frac{1}{r_{ij}^{12}} - \frac{1}{r_{ij}^6}$ ,  $1 \leq i, j \leq N$  where  $r_{ij} = \|x^i - x^j\|$ .

The LJ potential function for a single pair of neutral atoms is a simple unimodal function. This is illustrated by Fig. 2. It is easy to find the overall minimum of this function that is assumed at 1 with energy  $-1$ . In a complex system, many atoms interact and we need to sum up the LJ potential functions for each pair of atoms in a cluster. The result is a complex energy landscape with many local minima. It is given by:

$$\text{Min } V(X) = \sum_{i < j} v(\|x^i - x^j\|) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N \left( \frac{1}{\|x^i - x^j\|^{12}} - \frac{1}{\|x^i - x^j\|^6} \right).$$

As it is known the LJ problem is a highly nonlinear, non convex function with numerous local minimum that takes the LJ problem a challenging standard test problem for the global optimization algorithms. For illustration, we fix one atom's position and let the others move around the fixed one. A cluster of more than 20 atoms has many of local minima along its LJ PES.

## 6 Experimental settings and parameter selection

Population size for all the algorithms is taken as  $NP (= 3D)$ , where  $D$  is the dimension of the problem. Crossover rate (Cr) and scaling factor ( $F$ ) are fixed at 0.01 and 0.5 respectively; Maximum generations for all the algorithms are fixed at 8000; PCX has two additional parameters  $\sigma_\zeta$  and  $\sigma_\eta$ , which are taken as 0.1 each. For Pro. DEPCX, the probability of applying parent centric mutation is taken as  $P_D = 0.01$ . All the algorithms were run 30 times for each of the problems to determine the performance measure in terms of mean fitness, standard deviation, number of function evaluations and generations. In every case, a run was terminated when the function values of all points in population  $S$  were identical to an accuracy of four decimal places, i.e.,  $|f_{\max} - f_{\min}| \leq \varepsilon = 10^{-4}$  or when the maximum generation was reached. All the algorithms are executed on a Pentium IV PC using Dev C++.

## 7 Numerical results and comparisons of algorithms

The performances of proposed DEPCX and Pro. DEPCX are demonstrated on a set of seven nonlinear engineering design problems and the numerical results are compared with the classical DE. Standard performance measures like average fitness function value, standard deviation (Std), average number of function evaluations and number of generations are taken to compare the performance of the algorithms. Average fitness function value implies the mean of the best fitness values obtained by the algorithm till the stopping criteria is satisfied, Standard Deviation (Std) signifies the consistency of an algorithm. Smaller Std indicates that the algorithm is more consistent. Number of function evaluations (NFE) tells about the speed of an algorithm. Smaller NFE shows that the algorithm converges more quickly. Numerical results based on these performance measures are given in Tab. 1 and Tab. 2. From Tab. 1, which gives the average fitness function value, standard deviation and average numbers of functions evaluations (NFE), we see that in terms of average fitness function value and standard deviation all the algorithms gave more or less similar results although in some cases the proposed algorithms gave a marginally better performance than basic DE. However if we compare the NFE the superior performance of the proposed algorithms become more evident. For the first problem  $F1$ , the basic DE took 8438 NFE, whereas DEPCX took only 882 NFE (an improvement of about 90%) and Pro. DEPCX took 1566 NFE (an improvement of more than 80%). Likewise for  $F3$  and  $F4$  also DEPCX and Pro. DEPCX showed an improvement of about 32% and 35% in terms of NFE in comparison to basic DE. For  $F2$ , basic DE took 306 NFE while DEPCX took 228 NFE. However, basic DE gave a better performance than Pro. DEPCX for  $F2$ . For the remaining problems, all the algorithms took same number of NFE to meet the desired stopping criteria. If we talk about the number of generations then we can see from Tab. 2 that the proposed algorithms converged in a lesser numbers of generations in comparison to the basic DE for problems  $F1$ ,  $F2$ ,  $F3$  and  $F4$ . For the remaining problems all the algorithms took same number of generations to satisfy the given stopping criteria.

**Table 1.** Mean of fitness function value, standard deviation (Std) and average of number of functions evaluations for all algorithms

Fun	Dim.	DE		DEPCX		Pro. DEPCX	
		Fitness	NFE	Fitness	NFE	Fitness	NFE
$f_1$	3	<b>2.96438e+006</b> (0.264829)	8438	2.97765e+06 <b>(0.0)</b>	<b>882</b>	2.96447e+006 (4.65661e-010)	1566
$f_2$	2	169.844 (0.000021)	306	169.846 (2.84217e-14)	<b>228</b>	<b>169.844</b> <b>(2.84217e-014)</b>	318
$f_3$	4	1.76382e-008 (3.51577e-08)	463	6.5688e-009 (1.65436e-24)	<b>312</b>	2.68164e-008 (3.30872e-024)	300
$f_4$	3	4.21422 (5.08471e-07)	838	4.20779 (0.0021397)	<b>451</b>	<b>4.21421</b> <b>(8.88178e-016)</b>	594
$f_5$	6	3.01253 (0.367899)	144000	5.35737 (0.2346895)	144000	<b>2.08167</b> (0.39679)	144000
$f_6(a)$	10	0.626379 (0.0821391)	240000	<b>0.568925</b> <b>(0.053931)</b>	240000	0.67054 (0.0537723)	240000
$f_6(b)$	20	1.07813 (0.0812955)	480000	1.14054 <b>(0.024262)</b>	480000	1.08687 (0.109388)	480000
$f_7$	10	-26.0421 <b>(0.446389)</b>	240000	-26.1463 (0.87264)	240000	<b>-26.348</b> (0.548505)	240000

**Table 2.** Average number of Generations taken by the algorithms to satisfy the desired stopping criteria

Fun	Dim.	DE	DEPCX	Pro. DEPCX
		Mean Gen.	Mean Gen.	Mean Gen.
$f_1$	3	937	<b>98</b>	174
$f_2$	2	51	<b>38</b>	53
$f_3$	4	38	26	<b>25</b>
$f_4$	3	93	<b>50</b>	66
$f_5$	6	8000	8000	8000
$f_6(a)$	10	8000	8000	8000
$f_6(b)$	20	8000	8000	8000
$f_7$	10	8000	8000	8000

## 8 Conclusion

The main aim of the present research is to observe the effect of parent centric approach, which is a well known phenomenon in GAs, in DE. The use of this approach has reportedly given better performance in comparison to the other versions of GA and some other stochastic techniques.

In this article we presented two modified versions namely DEPCX and Pro. DEPCX. In these algorithms parent centric approach is applied in the mutation phase which is in contrast with GA, where parent centric approach is used as a crossover operator. In terms of the quality of solution, all algorithms performed at par with each other by giving more or less similar average fitness function value. However in terms of average number of function evaluations and number of generations, the proposed algorithms fared much better than the basic DE. Empirical analysis of the algorithms shows that the proposed schemes help in improving the performance of basic DE algorithm. The positive feature of parent centric approach i.e. getting a solution

nearby the parent is that it helps in exploration of the neighborhood of the best point thereby increasing the probability of getting a better candidate for the next generation.

## References

- [1] H. Abbass. The self-adaptive pareto differential evolution algorithm. **in:** *Proceedings of the 2002 Congress on Evolutionary Computation*, 2002, 831–836.
- [2] H. Abbass. A memetic pareto evolutionary approach to artificial neural networks. *Lecture Notes in Artificial Intelligence*, 2002a, **2256**: 1–12.
- [3] R. Angira, B. Babu. Evolutionary computation for global optimization of non-linear chemical engineering processes. **in:** *Proceedings of International Symposium on Process Systems Engineering and Control*, Mumbai, 2003, 87–91.
- [4] B. Babu. *New optimization techniques in engineering*. Springer-Verlag, Berlin Heidelberg, 2004.
- [5] B. Babu, R. Angira. Optimization of non-linear functions using evolutionary computation. **in:** *Proceedings of the 12th ISME International Conference on Mechanical Engineering*, India, 2001, 153–157.
- [6] T. Back, F. Hoffmeister, H. Schwefel. A survey of evolution strategies. **in:** *Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications*, 1991, 2–9.
- [7] C. Beightler, D. Phillips. *Applied geometric programming*. John Wiley and Sons, New York, 1976.
- [8] J. Brest, S. Greiner, et al. Self-adapting Control parameters in differential evolution: a comparative study on numerical benchmark problems. *IEEE Transactions on Evolutionary Computation*, 2006, **10**(6): 646 – 657.
- [9] U. chakraborty, S. Das, A. konar. Differential evolution with local neighborhood. IEEE congress on evolutionary computation, Canada, 2006, 2042–2049.
- [10] S. Das, A. Konar, U. Chakraborty. Two improved differential evolution schemes for faster global search. ACM-SIGEVO Proceedings of GECCO, Washington D. C., 2005, 991–998.
- [11] K. Deb. A population-based algorithm-generator for real-parameter optimization. *Soft Computation*, 2005, **9**: 236–253.
- [12] K. Deb, A. An, D. Joshi. A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary Computation*, 2002, **10**(4): 371–395.
- [13] L. Fogel. Evolutionary programming in perspective: The top-down view. **in:** J. Zurada, R. Marks, C. Robinson, (Eds.), *Computational Intelligence: Imitating Life*, IEEE Press, Piscataway, NJ, USA, 1994.
- [14] C. Garcia-Martinez, M. Lozano, et al. Global and local real-coded genetic algorithms based on parent-centric crossover operators. *European Journal of Operations Research*, 2008, **185**: 1088–1113.
- [15] D. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley, 1989.
- [16] V. Laarhoven, P. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, 1987.
- [17] N. Maldenovic, J. Petrovic, et al. Solving spread spectrum radar poly phase code design problem by tabu search and variable neighborhood search. *European journal of operational research*, 2003, **151**: 389–399.
- [18] C. Martinez, M. Lozano, et al. Global and local real coded genetic algorithms based on parent centric crossover operators. *European journal of operational research*, 2008, **185**: 1088–1113.
- [19] Z. Michalewicz, D. Fogel. *How to Solve It: Modern Heuristics*. 2000.
- [20] M. Omran, A. Engelbrecht, A. Salman. Differential evolution methods for unsupervised image classification. **in:** *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 2, 2005a, 966–973.
- [21] M. Omran, A. Salman, A. Engelbrecht. Self-adaptive differential evolution, computational intelligence and security. *Proceedings Lecture Notes In Artificial Intelligence*, 2005, **3801**: 192–199.
- [22] M. Pant, M. Ali, V. Singh. *Differential evolution with parent centric crossover*. Second UKSIM European symposium on computer modeling and simulation, Liverpool UK, 2008.
- [23] S. Paterlini, T. Krink. High performance clustering with differential evolution. **in:** *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 2, 2004, 2004–2011.
- [24] B. Prasad, J. Saini. Optimal thermo hydraulic performance of artificially roughened solar air heater. *Journal Solar Energy*, 1991, **47**: 91–96.
- [25] S. Rahnamayan, H. Tizhoosh, M. Salama. Opposition-Based Differential Evolution. *IEEE Transactions on Evolutionary Computation*, 2008, **12**(1): 64 – 79.
- [26] R. Storn. *Differential evolution design for an IIR-filter with requirements for magnitude and group delay*. Technical Report TR-95-026, International Computer Science Institute, Berkeley, CA, 1995.
- [27] J. Teo. Exploring dynamic self-adaptive populations in differential evolution. **in:** *Soft Computing—A Fusion of Foundations*, DOI: 10.1007/s00500-005-0537, 2006.
- [28] Z. Yang, J. He, X. Yao. Making a Difference to Differential Evolution, in *Advances in Metaheuristics for Hard Optimization*. Springer, 2007, 415–432.