Asian Journal of Applied Sciences (ISSN: 2321 – 0893)
Volume 05 – Issue 05, October 2017

Based Cost Differential Evolution Algorithm for Multi-objective Optimization Problems

Chiha Ibtissem[1], Liouane Hend[2]

Electrical Engineering Department, ENIM Monastir
Tunisia

[1]*Email: chiha.ibtissem [AT] yahoo.fr*
[2]*Email: noureddine.liouane [AT] enim.rnu.tn*

ABSTRACT— ***In this paper, we present A new variant of the Differential Evolution algorithm. Our proposed algorithm called Based cost Differential Evolution (Bc-DE), generated the mutant vector by adding a weighted difference of two cost function multiplied by a vector selected randomly to the best individual vector. The performance of the Bc-DE algorithm is broadly evaluated on several bound-constrained nonlinear and non-differentiable continuous numerical optimization problems and compares with the conventional DE and several others DE variants. It is demonstrated that the new method converges faster and with more certainty than many other acclaimed global optimization methods.***

Keywords— ***Differential Evolution, global numerical optimization,multi-objective optimisation***

1. INTRODUCTION

Inspired from genetic algorithms, Differential Evolution algorithm (DE) is a powerful set of global search techniques, evolves a population of potential solutions (individuals) in order to increase the convergence to optimal solutions and employs heuristics to allow escaping from local minima in order to produce good results on a wide class of problems in reasonable time [1,2]. DE successfully applied for solving numerical benchmark and complex engineering optimization problems [3-6]. Indeed, these problems usually appear with objective functions being non-convex, non-linear, multi-dimensional or including many hard constraints.

DE proposed by Storn and Price [7], it has successfully been applied to many real and artificial optimization problems such as nonlinear system identification, aerodynamic optimization [8], automatic image pixel clustering [5], and many others problems. A DE based neural network-training algorithm was first introduced in [9].In addition, DE is an algorithm population based which use crossover operators, mutation and selection that similar genetic algorithm. In establishing best solutions, the major difference is that the genetic algorithms exploit enormously the crossover operator for the exploitation phase while DE algorithm exploits the mutation concept via the differential operator.

The DE algorithm works as follows: A population of solutions (vectors) is dynamically controlled by addition, subtraction, and component swapping, until the population converges to the optimum solution. Many researchers have recommended many empirical rule for choosing trial vector generation strategies and their associated control parameters. Storn and Price [10] proposed that a practical value for should be between $5D$ and $10D$, and an excellent initial choice of F was 0.5. The successful range of values was recommended between 0.4 and 1. The pre-specified crossover rate CR has a role important in the speed of convergence .The first reasonable attempt of choosing CR value can be 0.1. However, if the problem is near unimodal or fast convergence is desired, the value of 0.9 for CR may also be an excellent initial choice. The values of F or NP can be increased, if the population converges too early. Price suggested in [11] to apply the trial vector generation strategy DE/current-to-rand/1 and parameter setting $NP = 20D$, $CR = 0.5$, $F = 0.8$ .For DE converges prematurely, the value of NP and F must be increased or the value of CR should be reduced. If we increase the value of F or NP or CR is chosen randomly in the interval $\left[0\ 1\right]$, the population stagnated. The DE/rand/1/bin strategy along with a small CR value is also applied. Gämperle *et al.* [12] proposed different parameter settings for DE on Rastrigin, Sphere and Rosenbrock functions. Based on their experimental results, they interpreted that the convergence speed and the searching capability are very sensitive to the choice of control parameters NP, F and CR. They suggested that the population size NP be between $3D$ and $8D$, the crossover rate CR be between $\left[0.3, 0.9\right]$ and the scaling factor F equal 0.6.

Moreover, There are several conflicting interpretations in the DE literature, which were established with regard to the rules for manually selecting the strategy and control settings to solve scientific and engineering problems. Indeed, researchers and engineers are not well justified there interpretations, therefore their validity is possibly restricted to the problems, strategies, and parameter values considered in the studies.

Many researchers have proposed several techniques to avoid manual choosing of the strategy and control settings such as fuzzy conventional DE [14] , adaptive differential evolution (FADE) [15] , adaptation for DE (ADE) [16], self-adapted DE for multi-objective optimization problems [17], differential evolution with self adapting populations (DESAP) [18], self-adapted DE (SDE)[17], (jDE) [19], Self-adaptative DE (SaDE) [20] …etc. The fuzzy adaptive differential evolution (FADE) algorithm was introduced by Liu and Lampinen using fuzzy logic controllers whose inputs integrate the relative function values and individuals of successive generations to adapt the parameters for the mutation and crossover operations [14]. So, there experimental results applied on test functions showed that the FADE algorithm more efficient than the conventional DE on superior dimensional problems. Zaharie and Petcu considered an adaptive Pareto DE algorithm for multi-objective optimization and analysed its parallel implementation [21]. In [17], Abbass implemented DE for multi-objective optimization problems by self-adapting the crossover rate. In recent times, the self-adaptive neighbourhood search DE algorithm was adopted into a novel cooperative co-evolution gateway [26]. In addition, researchers enhanced the performance of DE; they implemented local search [27] or opposition-based learning [28]. Likewise, several researchers concentrated on the adaptation of the control parameters F and CR .

In this work, we developed a novel approach namely Based Cost Differential Evolution Bc-DE for solving optimization problems taking in consideration the Based cost Differential Evolution (DE) Algorithm and the multi-objective optimisation problem.

The main concept of Bc-DE is to generate trial parameter vectors by adding the best individual vector a weighted difference of the two relative performances objectives functions of two vectors and multiplied by a randomly selected vector.

This article is organized as follows: The multi-objective optimization is developed in section 2. In section3, we present a set of test functions used to solve multi-objective optimization problems in various difficult situations. The proposed algorithm is developed in section 4. In section5, the effectiveness of our approach is tested on a set of test functions and a comparative study with the Multi-objective Evolutionary Algorithms based on Pareto elitist existing namely NSGA-II and SPEA2 is presented. Finally, some conclusions are given in section 6.

2. THE DIFFERENTIAL EVOLUTION CONCEPT

Differential Evolution DE is a simple stochastic optimization algorithm and evolves a population of potential solutions (individuals) in order to increase the convergence to optimal solutions. DE is a global search algorithm employ heuristics to allow escaping from local minima. The main concept of DE is generating trial parameter vectors by adding a weighted difference of two parameters to a third one. This process is the appropriate genetic mutation operator of evolutionary algorithms. Output vectors are combined with another vector (called the target vector). This operation is called crossover. The result of crossover is a new vector, called the trial vector. The trial vector is accepted as an individual in the new generation if its fitness function value is less than the target vector fitness function value. It evolves generation by generation until the termination conditions would be met [42]. The main difference between genetic algorithm and DE algorithm is the mutation and recombination phase. In the Genetic Algorithms and Evolutionary Strategies the perturbation of the population occurs in accordance with a random quantity. Alternatively, the Differential Evolution algorithm uses the weighted differences between solution vectors to perturb the population.

Several variants of DE algorithm and the convention used for the description is DE/α/β/γ[43]. Where:

-α: represents the vector to be perturbed (Rand: random vector, Best: Best vector),

-β: represents the number of difference vectors considered for perturbation,

-γ: represents the type of recombination operator (exp: exponential; bin: binomial).

Table I summarizes the most commonly used schemes for differential evolutionary algorithms.

TABLE 1 .DIFFERENTIAL EVOLUTION (DE) STRATEGIES

Variant Name	DE recombination operator
DE/Rand/1/Bin	$v_i^t = x_{r3}^t + F(x_{r1}^t - x_{r2}^t)$
DE/Best/1/Bin	$v_i^t = x_{Best}^t + F(x_{r1}^t - x_{r2}^t)$
DE/Rand/2/Bin	$v_i^t = x_{r5}^t + F(x_{r1}^t - x_{r2}^t + x_{r3}^t - x_{r4}^t)$
DE/Best/2/Bin	$v_i^t = x_{Best}^t + F(x_{r1}^t - x_{r2}^t + x_{r3}^t - x_{r4}^t)$
DE/RandToBest/1/Bin	$v_i^t = x_{r3}^t + F(x_{Best}^t - x_{r3}^t + x_{r1}^t - x_{r2}^t)$

$r_1,\ r_2, r_3, r_4$ and r_5 are mutually distinct integers and also different from the running index, i, randomly selected with uniform distribution from the set $\{1, 2, 3, \ldots, i-1, i+1, \ldots NP\}$ and NP represents the population size.

In the conventional DE algorithm, three control parameters NP, CR and F are fixed during the optimization process. In effect, there is a darkness of how to find reasonable differential operator reflecting rational values for given differential values between the sources of the mutation vectors. Another important design decision of the DE algorithm is the choice criterion of the selection strategies which is efficient and fast.

In our case we propose a new variant of Differential Evolution Algorithm named Based Cost Differential Evolution Algorithm.

There are four processes in the Based cost Differential Evolution algorithme (Bc-DE) which are:

- **initialization**

All parameter vectors in a population are randomly initialized and evaluated using the fitness function. The initial NP - D dimensional vectors are chosen randomly and should cover the entire parameter space x_i^0 ; where $i = 1, 2 \ldots NP$ and NP is size of population vectors, D is the number of decision variables.

- **Mutation operator**

DE generates new vectors by adding the difference between two vectors to a third one. The mutant vector is generated according to:

$$V_{new,G+1} = X_{Best,G} + F.(P_{Best,G} - P_{r,G}).X_{r,G}$$

Where: $$P_{Best,G} = \frac{f(X_{Best,G})}{f(X_{Best,G}) + f(X_{r,G})} \qquad P_{r,G} = \frac{f(X_{r,G})}{f(X_{Best,G}) + f(X_{r,G})}$$

$f(X_{Best,G})$: objective function of $X_{Best,G}$

$f(X_{r,G})$: objective function of $X_{r,G}$

Where v_i^t is a mutant vector; $X_{r,G}$ vector different to $X_{Best,G}$, F is a real and constant factor which controls the amplification of the perturbation vector $X_{r,G}$ (practically; $0 < F < 2$).

- **Crossover operator**

Considering the each best vector x_{Best}^t in the current population, a trial vector u_i^t is generated by crossing the best vector with the corresponding mutant vector v_i^t under a pre-specified crossover rate $CR \in [0\ 1]$. The mutated vector's elements are then mixed with the elements of the predetermined Best vector to yield the so-called trial vector. Moreover, choosing a set of crossover points, like crossover operator in genetic algorithms or evolution strategies, the crossover is introduced to increase the diversity of the perturbed parameter vectors.

The trial vector is formulated by:

$$u_i^t = \begin{cases} v_i^t & \text{if (Jrand(j)} \le \text{CR) or j=Rnbr(i)} \\ x_{Best}^t & \text{Otherwise} \end{cases} \qquad (2)$$
$$j = 1, 2, \ldots, D$$

Where Jrand(j) is the jth evaluation of a uniform random number generator with outcome $[0,1]$, CR is the crossover constant $[0,1]$, Rnbr(i) is randomly chosen index from1 to D , where D represents the number of dimension.

- **DE selection**

If the trial vector u_i^t has equal or better fitness function value than that of its corresponding best vector x_{Best}^t, it replaces the Best vector. Otherwise, the Best vector remains in the next iteration.

- **The Differential Evolution algorithm**

3. PROBLEMS TESTS

To validate a multi-objective optimization algorithm, it should be applied on a set of test functions. These functions are chosen to implement efficient methods studied in various difficult situations. These testing functions must be chosen such that:

1. They represent a particular threat to the convergence or for diversity;
2. The arrangement and shape of the Pareto surface are known and the values of variables corresponding decisions are easy to find.

We will use the test functions of Deb [11]. Each function has M objectives with $M \geq 2$ and x is the vector of decision variables.

$$x = (x_1, x_2, \dots, x_M). \qquad (4)$$

We partition the vector of decision variables into M groups.

Whether f_M a constructed function of a positive function g and a function h to M variables

. $$f_M(x) = g(x_M)h(f_1(x_1), \dots, f_{M-1}(x_{M-1}), g(x_M)) \qquad (5)$$

Where $x_m \in R^m$ for $m = 1, \dots, M - 1$.

Therefore, the optimization problem is defined by:

$$minimize\ f_m(x_m)_{m=1,\dots,M-1}, f_M(x) \qquad (6)$$

When the function g reaches its minimum, the optimal surface corresponds to these solutions is given as

$$f_M = g * h(f_1, \dots, f_{M-1}, g). \qquad (7)$$

From the function h that depends on M variables, we can define the shape of the surface of Pareto, this means that when the function h is multimodal then the optimal surface will be discontinuous when the function h is non-convex, and then the optimal surface is non-convex.

Hence, the function h has a very important role on the diversity of solutions. Thus the function g is used to construct the search space and is used to prevent convergence. If the function f_m for $m = 1, \dots, M - 1$, is non-linear, a variable density solutions will be found.

In what follows, we present five bi-objective tests that we use to validate our proposed approach. Our choice was fixed on these tests because they served as a common basis for the comparison of existing Evolutionary Algorithms for Multi-objective and the evaluation of new techniques. They are based on a generic form:

$$\begin{aligned} &min\ f_1 = f(x_1, \dots, x_M) \\ &min\ f_2 = g(x_{M+1}, \dots, x_N)h(f, g) \end{aligned} \qquad (8)$$

They allow for all kinds of difficulty:

- Convex Surface
- non-convex surface
- discontinuous surface
- non-uniform distribution

4. MO-Bc-DE ALGORITHM

Basically, Differential Evolution algorithm has been applied successfully to a wide range of single optimization problems [33]. Hence, several researchers have tried to extend it to handle MOPs. In single-objective optimization, the decision is easy and the candidate replaces the old vector solution only when the candidate is better than the new trial vector. In the case of multi-objectives programmation, the decision is not so straightforward due to the variety of optimal solutions, and the selection procedure has to be modified in order to search a cmpromises between several objective values. Our goal for the application of MO-Bc-DE will meet the following objectives:

1) Find solutions as close as possible to Pareto-optimal solutions, converging as possible to the front Pareto.
2) Accelerating and improving the population towards the Pareto Front.
3) Make the algorithm to find better solutions.

The computational procedure of MO-Bc-DE algorithm can be summarized as follows:

STEP 1

- Initialization of parameters: F amplification factor, the number of the population, the probability of crossover CR, the maximum number of G_{max} generation.
- Initialization of the population. The initial population may be generated using a uniform distribution of random numbers.

$$Pop_G = (X_{1,G}, \dots, X_{NP,G}) \qquad G = 0 \dots G_{max}$$

$$X_{l,G} = \left(x_{1l,G}, \ldots, x_{Dl,G} \right)^T \qquad l = 1 \ldots NP.$$

- Initialization of all Pareto to an empty set.

STEP 2

- Calculate separately the values of the objective functions

STEP 3

Classify populations using Pareto ranking criteria. The first non-dominant front is usually assigned a rank equal to 1. Similarly, the second non-dominant edge has a rank equal to 2 and so on. Solutions with a lower rank are the best candidates to be selected for the next generation.

STEP 4.Select a random number $r \in (1,..,NP)$ and select $X_{r,G}$

• Mutation: Generate mutant vector

$$V_{l,G} = \left(v_{1l,G}, \ldots, v_{Dl,G} \right) \quad \text{for each vector } X_{l,G}$$

$$V_{new,G+1} = X_{Best,G} + F.(P_{Best,G} - P_{r,G}).X_{r,G}$$

Where: $P_{Best,G} = \dfrac{f(X_{Best,G})}{f(X_{Best,G}) + f(X_{r,G})}$ $\qquad P_{r,G} = \dfrac{f(X_{r,G})}{f(X_{Best,G}) + f(X_{r,G})}$

$f(X_{Best,G})$: objective function of $X_{Best,G}$

$f(X_{r,G})$: objective function of $X_{r,G}$

STEP5. Crossover: Generate the test vector $T_{l,G+1} = (t_{1l,G+1}, t_{2l,G+1}, \ldots, t_{Dl,G+1})$ for each vector $X_{l,G}$

$$t_{kl,G+1} = \begin{cases} v_{kl,G+1} & \text{if } rand_k[0,1) \le CR \\ x_{kl,G} & \text{if } rand_k[0,1) > CR \end{cases} ; k \in \{1, \ldots, D\}$$

STEP 6. The selection operation allows individuals to choose to keep the new generation *(G+1).*
The function *f* must be minimized; the vector with the lowest value of *f* earns a place in the population of the next generation.

Replace the vector of the old population by the best vector of the total population. The solutions in the lower edges are initially selected row to replace the initial population.

STEP7.

- Update the archive with Pareto non-dominated.
- Reduce the size of the archive if necessary.
- Evaluate the solutions obtained for the different objectives.

STEP8.

- View the optimal parameter values
- These steps are repeated until the stopping criteria are met (If the maximum number of generations is reached ($G = G_{max}$).

5. EXPERIMENTAL RESULTS

In this section, we validate our contribution by the application of certain test functions used in the literature and were used for the evaluation of new techniques. To prove the efficiency of the Mo-Bc-DE algorithm , we performed a comparative study with the Multi-objective Evolutionary Algorithms based on Pareto elitist existing namely NSGA-II [10] and SPEA2 [7].

Our goal for the application of Mo-Bc-DE will meet the following objectives:

1) Find solutions as close as possible to Pareto-optimal solutions, converging as possible to the front Pareto.
2) Accelerating and improving the population towards the Pareto Front.
3) Make the algorithm to find better solutions.

We will use in what follows, the test functions of Deb [11]. Each function has M objectives, with M ≥ 2 and is *x* the vector of the decision variables

ZDT1

The first of this set of tests is the simplest corresponding Pareto front is continuous, convex and uniform distribution of solutions along the front.

$$ZDT1 = \begin{cases} f_1(x) = x_1 \\ g(x_2) = 1 + \frac{9}{N-1}\sum_{i=2}^{N} x_i \\ h(x) = 1 - \sqrt{f_1 / g} \end{cases} \qquad (9)$$

Where $x_i \in [0,1]$ and for all $i = 1,\ldots,N$, $N = 30$.

ZDT2

The difficulty of this problem is present in the non-convex Pareto front

$$ZDT2: \begin{cases} f_1(x) = x_1 \\ g(x_2) = 1 + \frac{9}{N-1}\sum_{i=2}^{N} x_i \\ h(x) = 1 - (f_1 / g)^2 \end{cases} \qquad (10)$$

ZDT3

The difficulty of this problem is that Pareto front is discontinuous.

$$ZDT3: \begin{cases} f_1(x) = x_1 \\ g(x_2) = 1 + \frac{9}{N-1}\sum_{i=2}^{N} x_1 \\ h(x) = 1 - \sqrt{f_1 / g} - (f_1 / g)\sin(10\pi f_1) \end{cases} \qquad (11)$$

ZDT4

This test models the convergence difficulty to the global Pareto front in the presence of several local fronts due to the fact that the function *g* is multimodal.

$$ZDT4: \begin{cases} f_1(x) = x_1 \\ g(x_2) = 1 + 10(N-1) + \sum_{i=2}^{N}(x_i^2 - 10\cos(4\pi x_i)) \\ h(x) = 1 - \sqrt{f_1 / g} \end{cases} \qquad (12)$$

ZDT6

The particularity of this problem is that the best solutions are not uniformly distributed along the Pareto Front. This effect is due to non-linearity of the function f_1

$$ZDT6: \begin{cases} f_1(x) = 1 - \exp(-4x_1)\sin^6(4\pi x_1) \\ g(x_2) = 1 + 9(\sum_{i=2}^{N} x_i / (\mathrm{N} - 1))^{1/4} \\ h(x) = 1 - (f_1 / g)^2 \end{cases} \qquad (13)$$

In our algorithm Mo-Bc-DE, we use the following parameter values :

- Population size = 100.
- Number of generations = 1000 for ZDT1and ZDT4, 500 for ZDT2, ZTD5 and ZDT6.
- The maximum number of iterations = 20.
- Rate crossover = 0.9.
- Mutation rate = 0.1.

Configuration settings NSGA -II and SPEA2 is:

- Population size = 100.
- Chromosome: Binary encoding , 50 bits for each

variable

- Operation Crossing: uniform crossover
- Rate crossover = 0.9.
- Mutation rate = 2/L, where L is the length of chromosome.

The following tables summarize the results of the simulation respecting the two following performance indicators:

- *Generational distance* (*GD*)[12] is defined as follows:

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \qquad (14)$$

Where n is the number of non-dominated vectors found by the algorithm and d_i is the Euclidean distance between each of them and the nearest edge of the true Pareto member.

- *Spacing* (*SP*): working on the surface of compromise [13]

1 - The total length of the surface is measured compromise
2 - Calculate the ideal spacing between two points
3 - The sum of the differences between the ideal spacing and spacing between two points is calculated

$$SP = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d} - d_i)^2} \qquad (15)$$
$$d_i = \mathbf{min}_{i,j\neq i}\left(\sum_{k=1}^{K}\left| f_i^k - f_j^k \right|\right)$$

Where n is the number of vectors in the front Pareto found by the algorithm being evaluated, K is the number of objectives, $\bar{d}$ is the average of all d_i.

A value of $SP = 0$ indicates that all non-dominated solutions founded are equidistant

From these two tables, we see that the values of measurement Generational Distance (*GD*) and the values of spacing (*SP*) calculated for the different test functions using our new contribution Mo-Bc-DE are much lower relative to those calculated by NSGA-II and SPAE2.

Table.2 Measurement of the generational distance (*GD*)

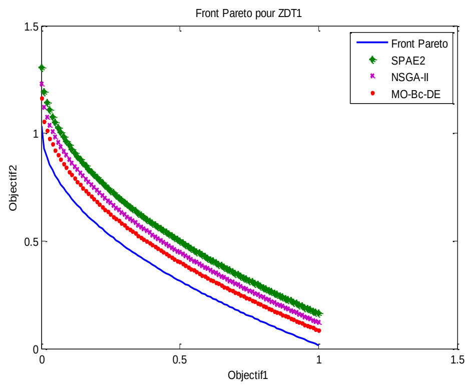
Algorithm	**ZDT1**	**ZDT2**	**ZDT3**	**ZDT4**	**ZDT6**
SPAE2	$1.95.10^{-3}$	$2.23.10^{-3}$	$2.39.10^{-3}$	0.779	0.0108
NSGA-II	$1.95.10^{-6}$	$7.44.10^{-4}$	$2.27.10^{-3}$	0.766	$7.5.10^{-3}$
Mo-Bc-DE	$0.93.10^{-6}$	$6.95.10^{-5}$	$1.06.10^{-3}$	0.53	$4.2.10^{-3}$

Table.2 Measurement of Spacing (*SP*)

Algorithm	**ZDT1**	**ZDT2**	**ZDT3**	**ZDT4**	**ZDT6**
SPAE2	0.571	0.586	0.658	0.638	0.838
NSGA-II	0.433	0.308	0.4844	0.481	0.618
Mo-Bc-DE	0.295	0.158	0.456	0.506	0.701

The Pareto fronts found by Mo-Bc-DE and NSGA-II SPAE2 for different test functions are shown in Figures1-5. Figure1 clearly shows that the algorithms Mo-Bc-DE and NSGA-II have a better convergence to the Pareto optimal front as SPAE2 for ZDT1. From Figure 4.15 it is clear that the solutions of Mo-Bc-DE have better distribution that NSGA-II and SPEA2 through the Pareto optimal front for problem ZDT2 which has a front non-convex Pareto optimal. The latter is due to the non-linearity of the function f_1.

Finally, the algorithm Mo-Bc-DE maintains a good distribution of solutions on the Pareto frontier for different problems and is able to converge better in all problems except ZDT4 where NSGA-II found a better convergence. Figure1 presents Pareto fronts founded by Mo-Bc-DE NSGA-II and SPAE2 for ZDT1.



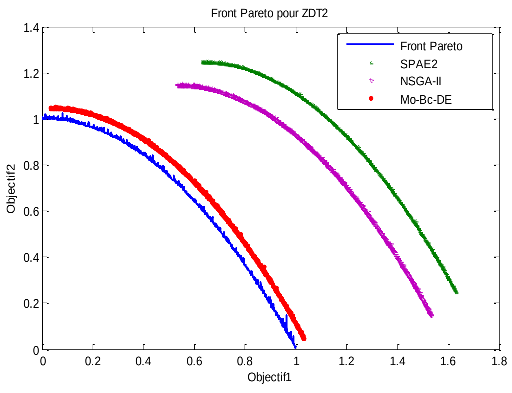
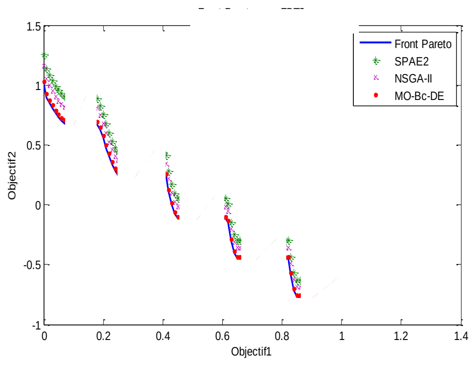


Figure1 Pareto fronts
found by Mo-Bc-DE, NSGA-II and SPAE2 for ZDT1

Figure 2 Pareto fronts
found by Mo-Bc-DE, NSGA-II and SPAE2 for ZDT2



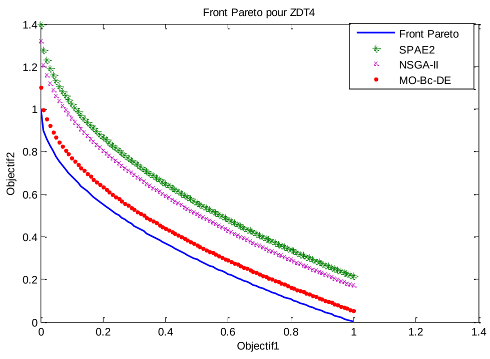


Figure 3 Pareto fronts
found by Mo-Bc-DE, NSGA-II and SPAE2 for ZDT3

Figure 4 Pareto fronts found
by Mo-Bc-DE, NSGA-II and SPAE2 for ZDT4

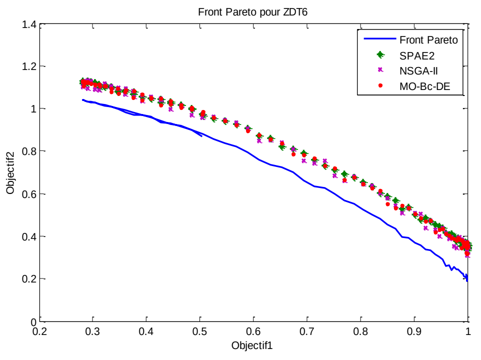


Figure5 Pareto fronts found by Mo-Bc-DE, NSGA-II and SPAE2 for ZDT6

Experimental results based on well-known test functions show that our new approach Mo-Bc-DE is easy to implement and can produce better than the solutions founded by the two well-known NSGA-II and SPEA2 methods solutions. Our proposed approach generates a Pareto front that is closer to true Pareto front, this approach also produces non-dominated solutions are uniformly distributed. For ZDT6, which is a very difficult problem he gave solutions not uniformly distributed along the Pareto front.

6. CONCLUSION

In this paper, we present a novel algorithm namely Multi-objective Based Cost Differential Evolution Mo-Bc-DE for solving a multi-objective optimization problem taking in consideration the Differential Evolution (DE) Algorithm. The main concept of our algorithm is to generate a new trial parameter vectors by adding the best individual vector a weighted difference of the two relatives performances objectives functions of two vectors and multiplied by a randomly selected vector. We validated our contribution by the application of certain test functions used in the literature and were used for the evaluation of new techniques. To prove the efficiency of the Mo-Bc-DE algorithm, we performed a comparative study with the Multi-objective Evolutionary Algorithms based on Pareto elitist existing namely NSGA-II and SPEA2. Experimental results show that our new approach Mo-Bc-DE is able to find solutions as close as possible to Pareto-optimal solutions, converging as possible to the front Pareto.

.

7. REFERENCES

[1] R. Storn and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," Technical Report, 1995, TR-95-012, ICSI,

[2] T. Rogalsky, R. W. Derksen and S. Kocabiyik, Differential evolution in aerodynamic optimization, Proc. of 46th Annual Conference of Canadian Aeronautics and Space Institute, 1999, pp. 29-36.

[3] S. Das and A. Konar, Automatic image pixel clustering with an improved differential evolution Applied Soft Computing 9 (2009), no. 1, 226-236

[4] J. Ilonen, J. K. Kamarainen and J. Lampinen, Differential evolution training algorithm for feed-forward neural networks, Neural Processing Letters 17 (2003), no. 1, 93-105..

[5] H. A. Abbass, and R. Sarker,: The Pareto Differential Evolution Algorithm, InternationalJournal on Artificial Intelligence Tools, Vol. 11, No. 4, (2002), 531-552

[6] B.V. Babu, and P. G.Chakole, Mubeen, J.H.S.: Multi-objective Differential Evolution (MODE) for Optimization of Adiabatic Styrene Reactor, Chemical Engineering Science, Vol. 60 (No. 17)(2005), 4822-4837

[7] E. Zitzler and L. Thiele "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach", IEEE Transactions on Evolutionary Computation, 3(4):257–271, November 1999.

[8] A. Farhang-Mehr and S. Azarm. "Diversity Assessment of Pareto Optimal Solution Sets: An Entropy Approach". In Congress on Evolutionary Computation (CEC'2002), vol.1, pp. 723–728, May 2002.

[9] J. Knowles and D. Corne "Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors" . IEEE Transactions on Evolutionary Computation, 7(2):100–116, April 2003.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II". IEEE Transactions on Evolutionary Computation, 6(2):182–197, April 2002.

[11] K. Deb, S. Agrawal, A. Pratap, and C. Meyarivan. "A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization" Nsga-ii". Proceedings of the Parallel Problem Solving from Nature VI Conference, pp. 849–858, 2000.

[12] A. Veldhuizen and G. Lamont, "On measuring multiobjective evolutionary algorithm performance", 2000 Congress on Evolutionary Computation, pp. 204–211, 2000.

[13] J. Schott. "Fault Tolerant Design Using single and Multicriteria Genetic Algorithm Optimization". Master thesis, Department of Aeronautics and Astronautics, Institute of Technology, Cambridge, Massachusetts, 1995.