Non-dominated Sorting Differential Evolution (NSDE): An Extension of Differential Evolution for Multi-objective Optimization.

Conference Paper · January 2005
Source: DBLP

Dr B V Babu
Graphic Era University

Non-dominated Sorting Differential Evolution (NSDE): An Extension of Differential Evolution for Multi-objective Optimization

Rakesh Angira and B.V. Babu

Department of Chemical Engineering, Birla Institute of Technology & Science
Pilani (Rajasthan) -333 031, India
{angira, bvbabu}@bits-pilani.ac.in

Abstract. Most of the real world optimization problems are multi-objective in nature. Recently, Evolutionary algorithms are gaining popularity for solving Multi-Objective Optimization Problems (MOOPs) due to their inherent advantages over traditional methods. In this paper, Differential Evolution (an evolutionary algorithm that is significantly faster and robust for optimization problems over continuous domain) is extended for solving MOOPs and we call this extended algorithm as Non-dominated Sorting Differential Evolution (NSDE). The proposed algorithm is applied successfully to two different benchmark test problems. Also, the effect of various key parameters on the performance of NSDE is studied. A high value of crossover constant ($\cong$ 1) and a value of 0.5 for scaling factor are found suitable for both the problems.

1 Introduction

Many engineering applications involve multiple criteria, and recently, the exploration of Evolutionary Multi-Objective Optimization (EMOO) techniques has increased [1]. The ideal approach for a multi-objective problem is the one that optimizes all conflicting objectives simultaneously. Classical optimization methods can at best find one solution in a single run, on the other hand evolutionary algorithms can find multiple optimal solutions in a single run due to their population based approach. Additionally, evolutionary algorithms are less susceptible to problem dependent characteristics, such as the shape of the Pareto front (convex, concave, or even discontinuous), and the mathematical properties of the search space, whereas these issues are of concerns for mathematical programming techniques for mathematical tractability.

Schaffer [2] proposed the first practical approach to multi-criteria optimization using EAs, Vector Evaluated Genetic Algorithm (VEGA). After that there have been several other versions of evolutionary algorithms that attempt to generate multiple non-dominated solutions such as [3, 4]. The concept of Pareto-based fitness assignment was first proposed by [5], as a means of assigning equal probability of reproduction to all non-dominated individuals in the population. Fonseca and Fleming [6] have proposed a multi-objective genetic algorithm (MOGA). Srinivas and Deb [7] proposed NSGA, where a sorting and fitness assignment procedure based on Gold-

Bhanu Prasad (Editor): IICAI-05, pp. 1428-1443, 2005.

berg's version of Pareto ranking is implemented. Horn et al. [8] proposed Niched Pareto Genetic Algorithm (NPGA) using a tournament selection method based on Pareto dominance. Knowles and Corne [9] proposed a simple evolution strategy (ES), (1+1)-ES, known as the Pareto Archived Evolution Strategy (PAES) that keeps a record of limited non-dominated individuals. The more recent algorithms include the Strength Pareto Evolutionary Algorithm (SPEA) algorithm [10], NSGA-II [11], Pareto-frontier Differential Evolution [12], and Multi-Objective Differential Evolution [13, 14, 15, 16, 17].

Differential Evolution (DE), a recent optimization technique proposed by [18], is an exceptionally simple evolution strategy, which is significantly faster & robust at numerical optimization and is more likely to find a function's true global optimum. Simple GA uses a binary coding for representing problem parameters whereas DE uses real coding of floating point numbers. Among the DE's advantages are its simple structure, ease of use, speed and robustness.

Original DE [18] dealt with a single strategy. Later on ten different strategies have been suggested [18]. A set of control parameters that works out to be the best for a given problem may not work well when applied for a different problem. The best value of control parameters to be adopted for each problem is to be determined separately by trial & error. Similarly, the strategy that works out to be the best for a given problem may not be effective when applied to some other problem.

DE has been successfully applied in various fields. Previous studies [20, 21, 22, 23, 24, 25, 26] have shown that DE is an efficient, effective and robust evolutionary optimization method. The detailed DE algorithm and code are available in literature [18, 19]. Onwubolu & Babu [27] and Babu [28] included in their books many successful applications of DE for various engineering problems.

Previously, a few researchers [11, 13, 14, 15, 16] studied the extension of differential evolution to multi-objective optimization problem in continuous domain, but using different approach from that described in this paper. In this paper, NSDE, a simple extension of DE (where same mutation & crossover scheme is used as in DE, however the selection criterion is modified as it is being used for solving MOOPs) is proposed and tested on the two test problems. One test problem is Schaffer's function and the other is cantilever design problem.

2 Multi Objective Optimization Problems (MOOPs)

As the name suggests, a multi objective optimization problem deals with more than one objective function. In most practical decision-making problems multiple objectives or multiple criteria are evident. Multi-Objective optimization is sometimes referred to as vector optimization, because a vector of objectives, instead of a single objective, is optimized. General form of the multi-objective optimization problem (MOOP) is given as follows [29]:

$$\text{Minimize / Maximize} \qquad f_m(\mathbf{x}) \qquad m = 1, 2, \ldots, M.$$

$$\text{Subject to} \quad g_j(\mathbf{x}) \quad j = 1, 2, \ldots, J. \tag{1}$$

$$h_k(\mathbf{x}) \quad k = 1, 2, \ldots, K.$$

$$x_i^{(L)} \le x_i \le x_i^{(U)} \quad i = 1, 2, \ldots n.$$

A solution $\mathbf{x}$ is a vector of n decision variable: $\mathbf{x} = (x_1, x_2 \ldots x_n)^T$. The last set of constraints are called variable bounds, restriction each decision variable x_i to take a value with a lower limit $x_i^{(L)}$ and an upper $x_i^{(U)}$ bound. These bounds constitute a decision variable space D, or simply the decision space. Associated with the problem is J inequality and K equality constraints. The terms g_j ($\mathbf{x}$) and h_k ($\mathbf{x}$) are called constraint functions. The inequality constraints are treated as 'greater-than-equal-to' types although a 'less-than-equal-to' type inequality constraint must be converted into a 'greater-than-equal-to' type constraint by multiplying the constraint by -1. A solution $\mathbf{x}$ that does not satisfy all of the $(J + K)$ constraints and all of the $2n$ variable bounds stated above is called an infeasible solution. On the other hand, if any solution $\mathbf{x}$ satisfies all constraints and variable bounds, it is known as a feasible solution. Therefore, we realize that in the presence of constraints, the entire decision variable space D need not be feasible. The set of all feasible solutions is called the feasible region, of S.

A multi-objective optimization problem and its global optimal solution(s) can be defined in many ways. A solution is Pareto-optimal if it is dominated by no other feasible solution, which means that there exists no other solution that is superior at least in case of one objective function value and equal or superior with respect to the other objective functions values [29].

3 Non-dominated Sorting Differential Evolution (NSDE)

NSDE algorithm is a simple extension of DE for solving multi-objective optimization problems. The working of NSDE and DE is similar except the selection operation that is modified in order to solve the multi-objective optimization problems. The detail of the NSDE algorithm is as follows:

First of all set the set the key parameters, i.e., *CR* - crossover constant, *F* - scaling factor, *NP* - population size, *Max_gen* – maximum number of generations of NSDE algorithm. And then randomly initialize the population points within the bounds of decision variables. After initialization of population, randomly choose three mutually different vectors for mutation and crossover operation (as is done in DE algorithm) to generate trial vector. Evaluate the trial and target vector and perform a dominance check. If trial vector dominates the target vector, the trial vector is copied into the population for next generation otherwise target vector is copied into population for next generation. This process of mutation crossover, and dominance check is repeated for specified number of generations. Evaluate and then sort this final population to obtain the non-dominated solutions. Sorting can be done using any of the standard approaches reported in [29]. In the present study, naïve and slow approach. In this approach, each solution *i* is compared with every other solution in the population to check if it is dominated by any solution in the population. If no solution is found to dominate solution *i*, it is member of the non-dominated set otherwise it does not belong the non-dominated set. This is how any other solution in the population can be checked to see if it belongs to the non-dominated set.

The stopping criteria for the algorithm can be any one of the following conditions:

(a). There is no new solution added to the non-dominated front for a specified number of generations.

(b). Till the specified number of generations.

However, in this study, the second condition is used as termination criterion. The pseudo code of NSDE algorithm used in the present study is given below:

Set the values of NSDE parameters *D, NP, CR* and *Max_gen* (maximum generations).

. *Initialize* all the vectors of the population randomly within the bounds.

for i = 1 to NP
for j = 1 to D
$\mathbf{X}_{i,j}$ *=Lower bound+ random number *(upper bound - lower bound);*
End for
End for

Perform mutation, crossover, selection and evaluation of the objective function for trial and target vector for a specified number of generations.

While (gen < Max_gen)
{ for i = 1 to NP *`/** first for loop***/`*
{ *For each vector* $\mathbf{X}_i$ *(target vector), select three distinct vectors* $\mathbf{X}_a$, $\mathbf{X}_b$ *and* $\mathbf{X}_c$ *randomly from the current population other than the vector* $\mathbf{X}_i$
do
*{ r1 = random number * NP*
*r2 = random number * NP*
*r3 = random number * NP*
} While (r1=i) OR (r2=i) OR (r3=i) OR (r1=r2) OR (r2=r3) OR (r1=r3)
Perform mutation and crossover for each target vector $\mathbf{X}_i$ and create a trial vector, $\mathbf{X}_{t,i}$.
For binomial crossover:
{ p = random number
j_{rand} *= int (rand[0,1]* D)+1*
for n = 1 to D
{ if (p<CR or n = j_{rand} *)*
$\mathbf{X}_{t,i} = \mathbf{X}_{a,i} + F\ (\mathbf{X}_{b,i} - \mathbf{X}_{c,i})$
} else $\mathbf{X}_{t,i} = \mathbf{X}_{i,j}$
}

Perform selection for each target vector, $\mathbf{X}_i$ *by comparing its function value with that of the trial vector,* $\mathbf{X}_{t,i}$ *. If* $\mathbf{X}_{t,i}$ *dominates* $\mathbf{X}_i$ *then select* $\mathbf{X}_{t,i}$ *otherwise select* $\mathbf{X}_i$ *for the next generation population.*

If ($\mathbf{X}_{t,i}$ *dominates* $\mathbf{X}_i$)

Put $\mathbf{X}_{t,i}$ *into next generation population*

else Put $\mathbf{X}_i$ *into next generation population*

*} /** End of first for loop***/*

*} /** End of while loop***/*

Evaluate the objective functions for each vector.

for i = 1 to NP

$\mathbf{C}_{i,j} = funct_j(\,)$. *j = 1,.., no of objectives*

Remove all the dominated solutions using any one of the approaches proposed in [29]. *In the present study, the naïve and slow approach is used.*

Print the results (after the stopping criteria is met).

4 Test Problems

The algorithm is tested on the following two test problems [29]. The first problem is of one dimension while the other is of two dimensions.

Schaffer's function. This problem consists of two objectives to be minimized as shown in equation-2.

$$\begin{aligned} &\text{Minimize} \quad f(x) = x^2 \\ &\text{Minimize} \quad g(x) = (x-2)^2 \\ &\text{where} \quad -1000 < x < 1000 \end{aligned} \tag{2}$$

Cantilever Design Problem. A cantilever design problem with two decision variables is considered i.e. diameter (d) and length (l). The beam has to carry an end weight load P. Minimization of weight f_1 and minimization of end deflection f_2. The first objective will resort to an optimum solution having the smaller dimensions of d and l, so that the overall weight of the beam is minimum. Since the dimensions are small, the beam will not be adequately rigid and the end deflection of the beam will be large. On the other hand, if the beam is minimized for end deflection, the dimensions of the beam are expected to be large, thereby making the weight of the beam large.

$$\text{Minimize} \quad f_1 \text{ and } f_2$$

$$\text{Where} \quad f_1(d,l) = \frac{\rho d^2 l}{4}, \quad \text{and} \quad f_2(d,l) = \delta = \frac{64Pl^3}{3E\pi d^4} \tag{3}$$

Subject to $\sigma_{\max} \le S_y$ and $\delta \le \delta_{\max}$

$\delta_{\max}$ is calculated using $\sigma_{\max} = \dfrac{32Pl}{\pi d^3}$ and the following parameters are used ρ = 7800kg/m^3, P = 1kN, E = 207GPa, S_y = 300Mpa, $\delta_{\max}$ = 5mm.

5 Results and Discussion

5.1 Schaffer's function

Various experiments have been carried out in order to test the proposed algorithm by studying the effect of *Max_gen*, *CR*, and *F* for Schaffer's function.

Effect of *Max_gen*. The Pareto optimal front obtained using NSDE algorithm is shown in Fig. 1. The key parameters used are *NP* = 40, *CR* = 0.5, *F* = 0.2, *Max_gen* = 300, and seed = 10. It can be seen that maximum value of *f* is 4.0 and maximum value of *g* is 4.0. All other values of *f* and *g* will lie within these values for Pareto optimal solution. Fig. 2 shows effect of *Max_gen* on the number of solutions in final Pareto set (*NPS*) taking *NP* = 100. It is clear that with increase in *Max_gen*, *NPS* also increases and after *Max_gen* = 200, there is no significant increase in *NPS*. Therefore, an appropriate value of *Max_gen* seems to be about 200 giving *NPS* = 99% of initial population, i.e., *NP*.

When ten experiments with different seed values are carried out, the value of *NPS* was found to vary from 95 to 100% with an average of 98%. When an experiment is done with *NP* = 400, *Max_gen* = 300, and other parameters same as mentioned earlier, *NPS* is found to be 379 (Fig. 1) and at *Max_gen* = 200, *NPS* is found to be 375. This further establishes the fact that an appropriate value of *Max_gen* is 200 giving *NPS* about 94 to 100% of initial population, i.e., *NP*. However, an appropriate value of *Max_gen* may vary from problem to problem. Higher population size leads to a better Pareto front in terms of number of choices for optimal solutions (Fig. 1).

Effect of *CR* and *F*. The effect of *CR* and *F* is studied using the following settings: *NP* = 100, *Max_gen* = 300, *CR* = 0.5 if *F* is varied and *F* = 0.5 if *CR* is varied. Fig. 3 shows the Pareto front when *F* is fixed at 0.5 and *CR* is varied in steps of 0.1. It is observed that *CR* has no effect on *NPS* and Pareto front. In all the experiments with *CR* varying from 0.1 to 1.0, *NPS* is found to be 99 and exactly same Pareto front is obtained (Fig.3a). However for a different seed value, the distribution of solutions on the Pareto front is different (Fig. 3b) but *NPS* is nearly same (98).

Also, it is found that *F* does have little effect on *NPS* (95 to 99) but the distribution (spread) of solutions on Pareto front is different for different values of *F* (Fig. 4a, 4b) for same seed value. It is important to note that spread is different not only for different values of *F* but also for different seed values.

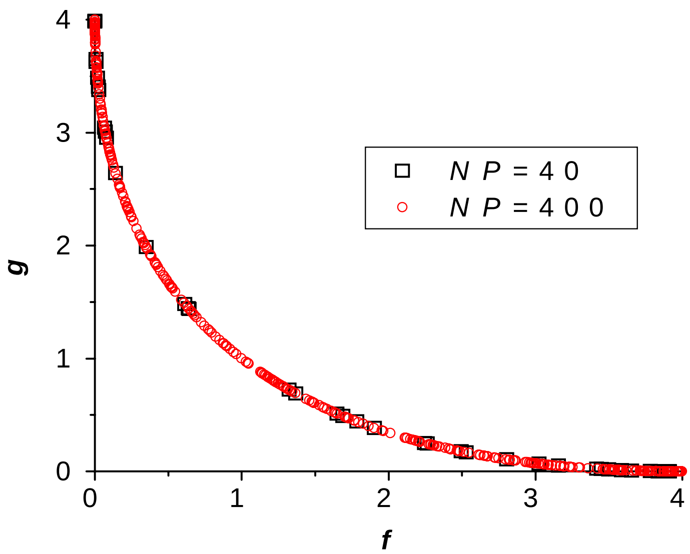


Fig. 1. Pareto Optimal Front for Schaffer's function using different *NP*

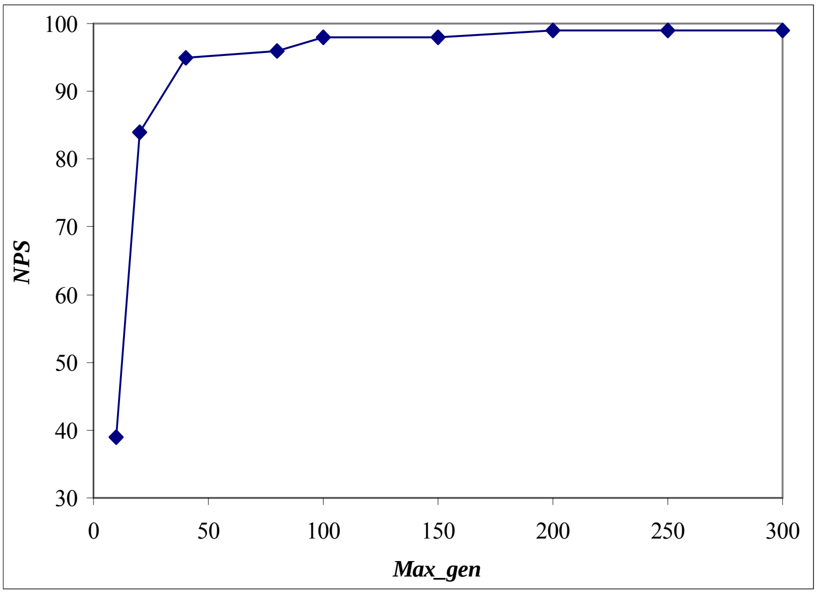
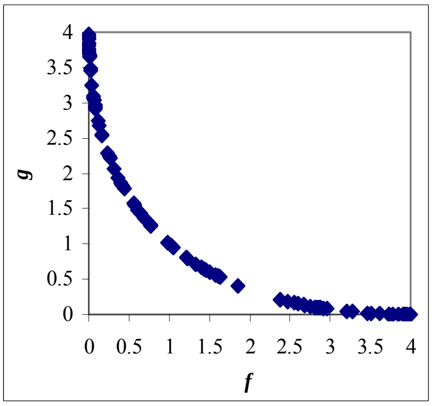
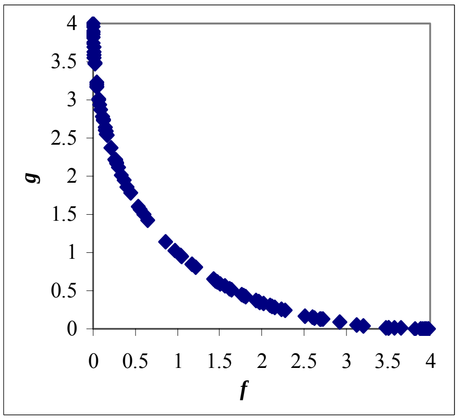


Fig. 2. Effect of *Max_gen* on *NPS*

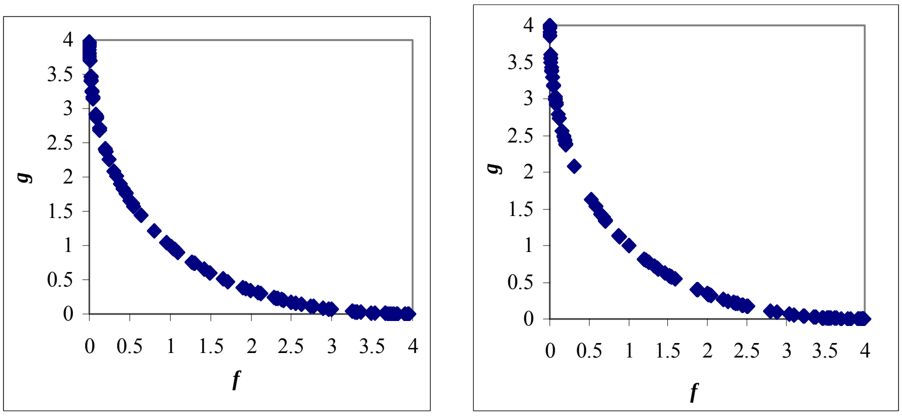


(a). (Seed = 10)



(b). (Seed = 23)

Fig. 3. Effect of *CR* and seed on Pareto optimal front for Schaffer's function



(a). (Seed = 10, F = 1.2) **(b).** (Seed = 10, F = 0.2)

Fig. 4. Effect of F on Pareto optimal front for Schaffer's function

5.2 Cantilever design problem

Effect of *Max_gen*. The Pareto optimal front is shown in Fig. 5 for the following setting of parameters: population size (*NP*) = 100, *CR* = 0.5, *F* = 0.5, seed = 11, and *Max_gen* =500. It can be seen that maximum value of weight (kg) is 3.04 and maximum value of Deflection (mm) is 2.06 as compared literature value of 3.06 & 2.04 respectively. All other values of weight and deflection will lie within these values of Pareto optimal solution. Fig. 6 shows effect of *Max_gen* on *NPS*. It is clear that with increase in *Max_gen*, *NPS* also increases and after *Max_gen* = 300, there is no significant increase in *NPS* but a good shape of Pareto optimal front is found after 500 generations. Therefore, an appropriate value of *Max_gen* seems to be about 500 giving *NPS* = 58% of initial population (*NP*).

When ten runs with different seed values are carried out, the value of *NPS* was found to vary from 50 to 58% with an average of 54%. When an execution is done with *NP* = 400 and keeping other parameters same as mentioned earlier, *NPS* is found to be 34.5% as against 54% for *NP* = 100. At *Max_gen* = 700 and 1000, *NPS* is found to be 39% and 45% respectively. This indicates that increase in *NP* does not lead to a corresponding increase in *NPS* if number of *Max_gen* is held constant. This is not similar to that found for Schaffer's function. Higher population size needs higher number of generations to get a better Pareto front in terms of both quantity (*NPS*) and quality (spread & shape i.e. global Pareto front).

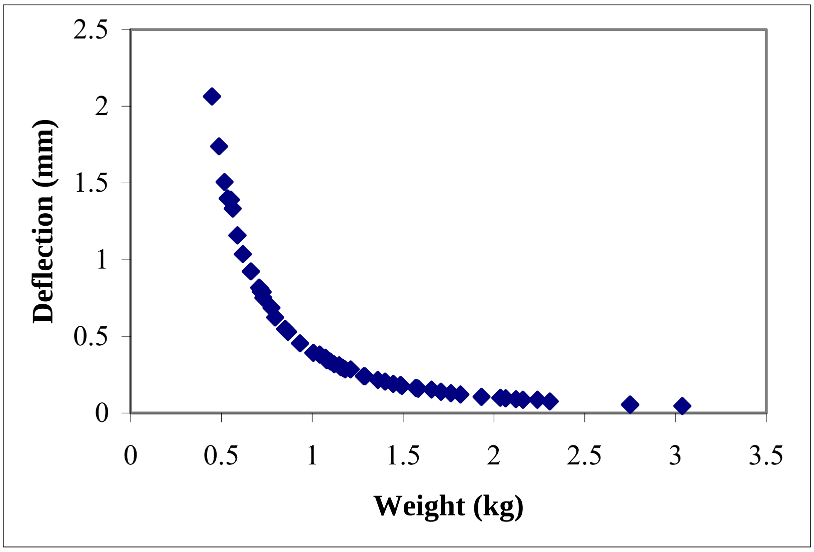


Fig. 5. Pareto Optimal Solutions for Cantilever Design Problem

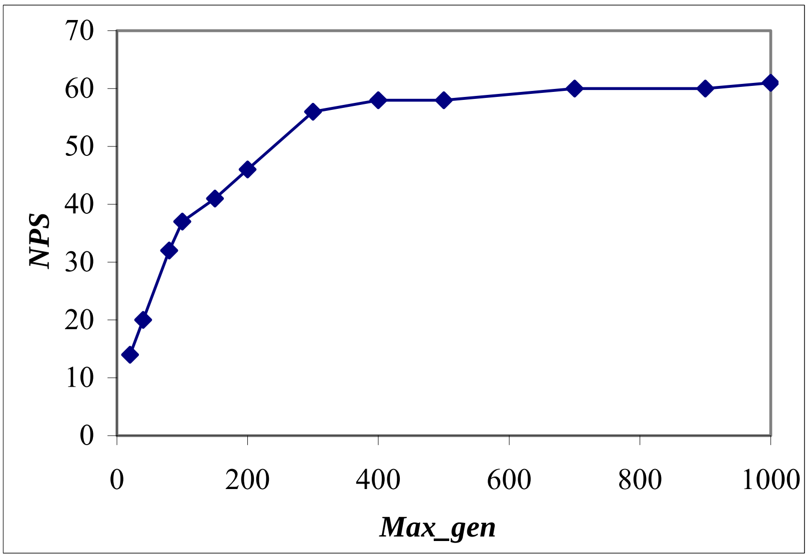
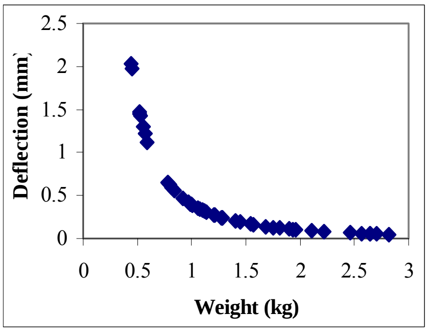
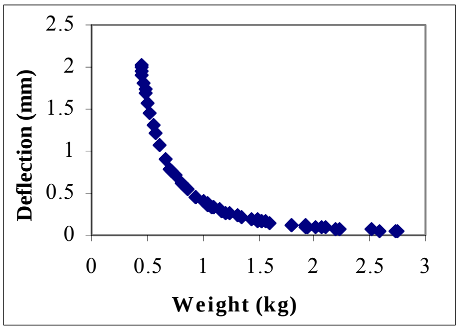


Fig. 6. Effect of *Max_gen* on *NPS*

Effect of *CR* & *F*. First the effect of *F* is studied using *CR* = 0.5, *NP* = 100, *Max_gen* = 500, seed = 10. Fig. 7 shows the Pareto optimal solution obtained for two different values of *F* (0.1 and 0.5). It is clear that the value of *F* not only affects the *NPS* but also the distribution (spread) of Pareto optimal solutions on Pareto front. It is found that values of *NPS* are 47 and 58 for *F* = 0.1 and 0.5 respectively. *NPS* is found to vary from 47 to 58% as shown in Fig. 8. As is evident from Fig. 8, *F* = 0.5 seems to be good giving highest *NPS* and good distribution of solutions on Pareto front (Fig. 7b).



(a). *F* = 0.1



(b). *F* = 0.5

Fig. 7. Effect of *F* on Pareto optimal front for cantilever design problem

The change in maximum and minimum values of objective functions with *F* is shown in Table-1. This gives information about how sensitive the extremes values of objective functions are as value of *F* changes. And the variation is significant in values of Weight (max) & Deflection (max) for *F* = 0.5 and 0.8.

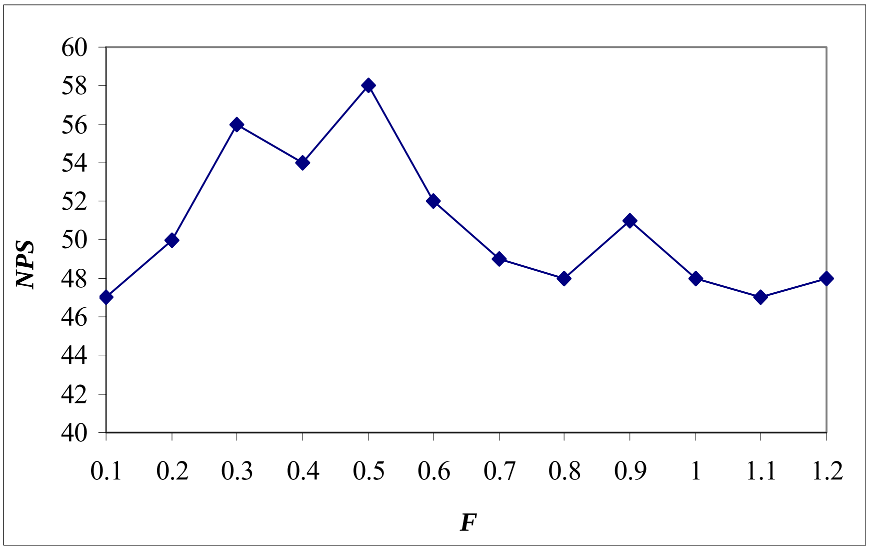
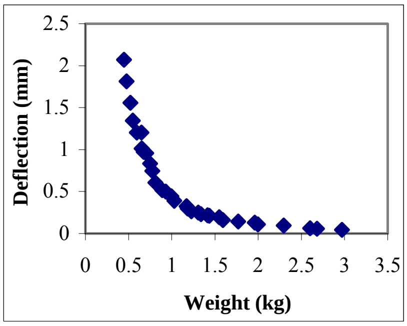


Fig. 8. Effect of ***F*** on ***NPS***

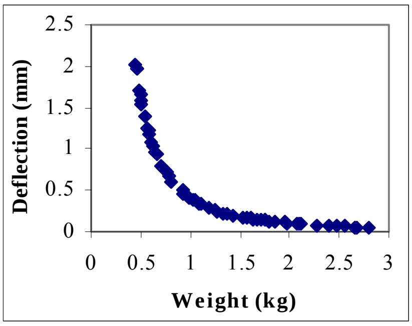
Table 1. Variation of objective function values (maximum & minimum) with *F*

F	Weight (max)	Deflection (max)	Weight (min)	Deflection (min)
0.1	2.8218	2.0285	0.4412	0.0500
0.5	2.7471	2.0169	0.4426	0.0526
0.8	2.9966	1.8470	0.4668	0.0499
1.2	2.9657	1.9456	0.4672	0.0490

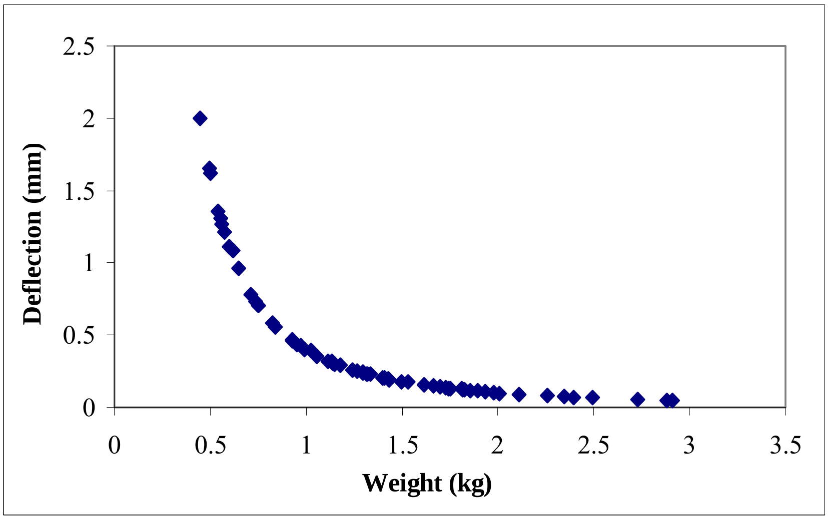
To study the effect of *CR*, the parameter setting is as follows: $F = 0.5$, $NP = 100$, *Max_gen* = 500, seed = 10. The *CR* is changed in steps of 0.1 from 0.1 to 1.0. Fig. 9a, 9b, and 9c show the Pareto optimal solutions for $CR = 0.1$, 0.5, and 1.0 respectively. It is evident from Fig. 9 that higher value of *CR* ($\cong 1.0$) results in good shape, i.e., global Pareto front. As shown in Fig. 9a and 9b, some of the solutions are not lying on the Pareto front for lower value of *CR*, i.e., 0.1 & 0.5. This indicates that for a given *Max_gen*, a higher value of $CR \cong 1.0$ works better. Fig. 10 shows the variation of *NPS* with *CR*. It is clear that *NPS* first increases from 33 to 53 till $CR = 0.3$, then decreases slightly at $CR = 0.4$ and again increases to 59 at $CR = 0.6$ and then decreases slightly to 57 at $CR = 0.7$. And then it increases to a value of 62 at $CR = 1.0$.



(a). *CR* = 0.1



(b). *CR* = 0.5



(c). *CR* = 1.0

Fig. 9. Effect of *CR* on Pareto optimal front for cantilever design problem

The change in maximum and minimum values of objective functions with *F* is shown in Table-2. This is clear from Table-2 that extreme values of the objective functions change with *CR*. However the variation is small as compared to effect of *F* (Table-1).

Table 2. Variation of objective function values (maximum & minimum) with *CR*

CR	Weight (max)	Deflection (max)	Weight (min)	Deflection (min)
0.1	2.9703	2.0723	0.4465	0.0447
0.5	2.7960	2.0095	0.4430	0.0531
1.0	2.9103	1.9982	0.4451	0.0469

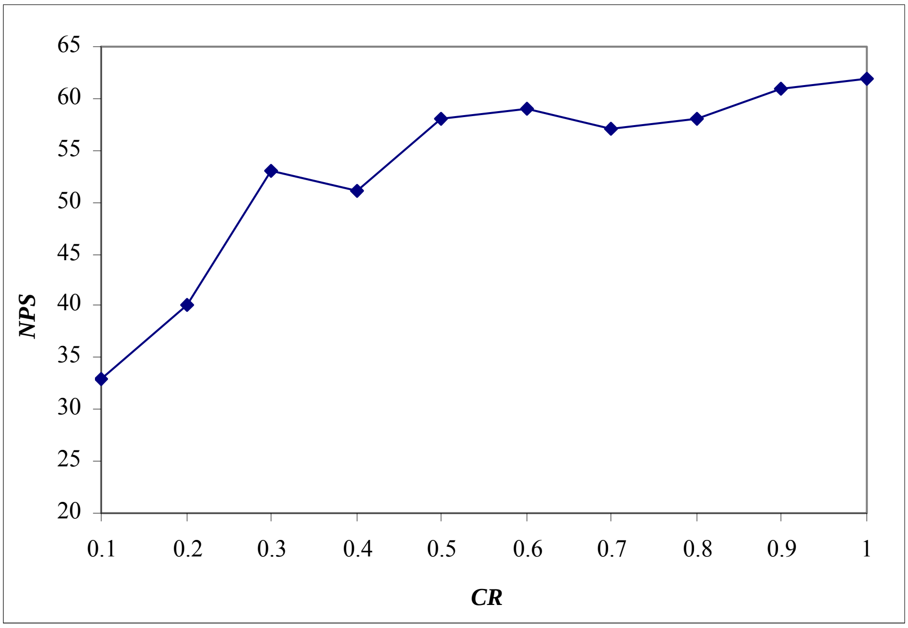


Fig. 10. Effect of *CR* on *NPS*

6 Conclusions

This paper deals with the extension of DE (i.e. NSDE) for solving MOOPs. Two problems, one standard test problem and cantilever design problem are solved using proposed NSDE algorithm. The results indicate that NSDE is able to locate the global Pareto front for two test problem studied. The effects of various parameters of NSDE i.e. *Max_gen*, *CR*, and *F* are discussed & analyzed. It is found that for both the test problems a increase in *Max_gen* increases the *NPS* up to a certain value after that there is no significance increase in *NPS*. This certain value is problem dependent and is found to be different for the two problems studied. For Schaffer's function it is about 200, while for cantilever design problem it is about 500. The effect of *CR* is significant in case of cantilever design problem while it does not affect the Schaffer's function. A high value of *CR* ($\cong 1$) is found suitable for both the problems. It is important to note that *F* not only affects the *NPS* but also the distribution of solutions in Pareto front for both the problems. A value of *F* ($= 0.5$) is found suitable for both the problems.

Future study will demonstrate the application of NSDE to more test functions, engineering problems and the comparison of performance with other techniques. Also, the performance of NSDE algorithm will be evaluated for constraint MOOPs.

References

1. Coello, C. A.: A Comprehensive Survey of Evolutionary- Based Multi objective Optimization Techniques. International Journal of Knowledge and Information Systems 1(1999) 269-308
2. Schaffer, J. D.: Multiple objective optimization with vector evaluated genetic algorithm. Proceeding of the First International Conference on Genetic Algorithms (1985) 93-100
3. Kursawe, F.: A variant of evolution strategies for vector optimization. In: Schwefel H. P. and Manner, R. (eds.): Parallel problem solving from nature, First workshop proceedings. Lecture notes in computer science, Vol. 496. Springer-Verlag, Berlin Heidelberg New York (1991) 193-197
4. Hajela, P., and Lin, C. Y.: Genetic search strategies in multicriterion optimal design. Structural Optimization 4 (1992) 99-107
5. Goldberg, D. E.: Genetic algorithms in search, optimization and machine learning. Reading, MA: Addison-Wesley (1989)
6. Fonseca C. M. and Fleming, P. J.: Genetic algorithms for multi objective optimization: formulation, discussion, and generalization. Genetic Algorithms: Proceeding of the Fifth International Conference (1993) 416-423
7. Srinivas, N. and Deb, K.: Multi-objective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation, 2 (1995) 221-248
8. Horn, J., Nafpliotis, N., and Goldberg, D.E.: A niched Pareto genetic algorithm for multi objective optimization. Proceedings of the First IEEE Conference on Evolutionary Computation (1994) 82-87
9. Knowles, J., and Corne, D.: Approximating the nondominated front using the Pareto archived evolution strategy. Evolutionary computation 8 (2000) 142-172
10. Zitzler, E., and Thiele, L.: Multi-objective evolutionary algorithms: A comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation 3 (1999) 257-271
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multi objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6 (2002) 182-197
12 Abbass, H.A., Sarkar, R., and Newton, C.: PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems. Proceedings of IEEE congress on Evolutionary Computation (2001) 971-978
13. Xue, F., Sanderson, A. C., Graves, R. J.: Pareto-based multi-objective differential evolution. Proceedings of the Congress on Evolutionary Computation .2 (2003) Canberra, Australia, IEEE Press, 862–869
14. Babu, B. V., Pallavi, G. C., and Mubeen, J. H.S.: Multiobjective Differential Evolution (MODE) for Optimization of Adiabatic Styrene Reactor. Chemical Engineering Science 60 (2005) 4824-4839
15. Babu, B. V., J. H. Syed Mubeen, and Pallavi G. C.: Multi-objective Optimization Using Differential Evolution. TechGenesis-The Journal of Information Technology 2 (2005) 4 - 12
16. Babu, B. V. and Jehan, M. M. L.: Differential Evolution for Multi-Objective Optimization. Proceedings of Congress on Evolutionary Computation (CEC-2003), Canberra, Australia, December 8-12, 2696-2703
17. Babu, B. V. and Anbarasu, B.: Multi-Objective Differential Evolution (MODE): An Evolutionary Algorithm for Multi-Objective Optimization Problems (MOOPs). To be presented at The Third International Conference on Computational Intelligence, Robotics, and Autonomous Systems (CIRAS-2005), December 13-16, Singapore (2005)
18. Price, K. & Storn, R.: Differential Evolution - A simple evolution strategy for fast optimization. Dr. Dobb's Journal 22 (1997) 18 - 24 and 78

19. Price, K. and Storn, R.: Homepage of DE, May, 2005. The URL of which is: http://www.ICSI.Berkeley.edu/~storn/code.html
20. Wang, F. S. and Chiou, J. P.: Optimal control and optimal time location problems of differential- algebraic systems by differential evolution. Industrial & Engineering Chemistry Research 36 (1997) 5348-5357
21. Chiou, J. P., and Wang, F. S.: Hybrid method of evolutionary algorithms for static and dynamic optimization problems with application to a fed-batch fermentation process. Computers & Chemical Engineering 23 (1999) 1277-1291
22. Babu, B. V. and Sastry, K. K. N.: Estimation of heat transfer parameters in a trickle-bed reactor using differential evolution and orthogonal collocation. Computers & Chemical Engineering 23 (1999) 327-339
23. Lu, J. C. and Wang, F. S.: Optimization of Low Pressure Chemical Vapor Deposition Reactors Using Hybrid Differential Evolution. Canadian Journal of Chemical Engineering 79 (2001) 246-254
24. Babu, B. V. and Angira, R.: A Differential Evolution Approach for Global Optimization of MINLP Problems. Proceedings of 4th Asia-Pacific Conference on Simulated Evolution And Learning (2002a), Singapore, November 18-22, Vol. 2, 866-870
25. Babu, B. V. and Angira, R.: Optimization of Non-Linear Chemical Processes Using Evolutionary Algorithm. Proceedings of International Symposium & 55th Annual Session of IIChE (2002b), Hyderabad, December 19-22.
26. Angira, R. and Babu, B. V.: Evolutionary Computation for Global Optimization of Non-Linear Chemical Engineering Processes. Proceedings of International Symposium on Process Systems Engineering and Control (ISPSEC '03)- For Productivity Enhancement through Design and Optimization (2003), IIT-Bombay, Mumbai, January 3-4, Paper No. FMA2, 87-91
27. Onwubolu, G. C. and Babu, B. V.: New Optimization Techniques in Engineering. Springer-Verlag, Heidelberg, Germany (2004)
28. Babu, B. V.: Process Plant Simulation. Oxford University Press, India (2004)
29. Deb, K.: Multi-objective optimization using evolutionary algorithms. John Wiley & Sons, New York (2001)