The Nucleus ***53****, No. 2 (2016) 155-161*

www.thenucleuspak.org.pk

The Nucleus

ISSN 0029-5698 (Print)

ISSN 2306-6539 (Online)

A Penalty Function Based Differential Evolution Algorithm for Constrained Optimization

H. Wazir, M. A. Jan, W.K. Mashwani* and T.T. Shah

Department of Mathematics, Kohat University of Science & Technology, KPK, Pakistan

hamzawazir@hotmail.com; majan@kust.edu.pk; mashwanigr8@gmail.com; tayyabashah37@yahoo.com

ARTICLE INFO

Article history :

Received : 10 April, 2016

Revised : 03 June, 2016

Accepted : 14 June, 2016

Keywords :

Differential evolution,
JADE,
Constraints handling,
Penalty function,
Static penalty function

ABSTRACT

Differential evolution (DE) and its various dialects are basically designed for solving unconstrained optimization problems and have been widely used .Adaptive differential evolution with optional external archive (JADE)is one of the efficient and updated versions of DE. This paper enhances the capability of JADE to solve constrained optimization problems (COPs). The enhancement is based on introducing a static penalty function in the selection scheme of JADE to handle constraints. The performance of the modified algorithm, abbreviated as CJADE-S is tested on a well-known test suit of COPs, CEC2006. The experimental results show the better performance of CJADE-S on most of the test problems of CEC2006.

1. Introduction

Evolution, being a two-step process of random variation and selection can be modelled mathematically as $x[t+1] = s(v(x[t]))$, where $x[t]$ and $x[t+1]$ are the populations at times t and *t+1,* respectively, obtained after execution of selection (s) and variation (v) operators. The process of evolution can also be modelled algorithmically and simulated on a computer [1]. The modelled algorithm is known as evolutionary algorithm (EA).EAs are inspired by the natural evolution of species. They randomly generate an initial population, which undergoes mutation and crossover to generate offspring. A selection scheme then plays the role of selecting good individuals among the parents and offspring on the basis of their fitness value. This process continues from one generation to another till the desired optimal value is achieved or a pre-specified stopping criteria is met.

EAs have been widely used for solving several types of unconstrained optimization problems (e.g., see) [2, 3]. However, they need some modifications to handle constraints and thus to solve COPs. From last so many decades, the resulting constraint handling EAs have got much attention [4]. By combining different constraint handling methods and EAs, researchers have designed a good number of constrained optimization evolutionary algorithms [5, 6]. They have been considerably successful in a wide range of applications, e.g., see [7-10], [11-14]. This paper implants the static penalty function proposed byHomaifar et al. [4] in the selection scheme of JADE to solve COPs, and thus proposes CJADE-S.

The remainder of this paper is organized as follows. Section 2 describes a short review of DE and JADE. Section 3 discusses some constraint handling techniques. It also details that how our chosen static penalty function is incorporated in the selection scheme of JADE. Sections 4 and 5 show the experimental results and comparison with some other algorithms, respectively. Finally, Section 6 concludes this paper.

2. Classic and Adaptive Differential Evolution

This section discusses the classical DE algorithm and one of its efficient and updated version, JADE.

2.1 Differential Evolution

Differential Evolution (DE) is one of the well-known EAs. It was first proposed by Rainer Storn and Kenneth Price in 1996, and is used for solving single-objective unconstrained optimization problems [15, 16]. DE also works on the same mechanism as other EAs. Its mutation, crossover and selection schemes are detailed as follows.

Mutation: An operator which maintains genetic diversity of one generation population to the next generation is called mutation. In each generation g of DE, a mutant vector, $\boldsymbol{v}_{i,g}$ for each individual of the current population, $\{\boldsymbol{x}_{i,g} | i = 1,2,\dots,NP\}$is designed by using, for example, one of the following strategies [17] :

1. *"DE/rand/1":*

$$\boldsymbol{v_{i,g}} = \boldsymbol{x}_{r0,g} + F_i * (\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}) \quad (1)$$

* Corresponding author

2. "DE/current-to-best/1"

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{i,g} + F_i * \left(\boldsymbol{x}_{best,g} - \boldsymbol{x}_{i,g}\right) + F_i * \left(\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}\right), \quad (2)$$

where $\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}$is a difference variation vector corresponding parent $\boldsymbol{x}_{i,g}$, $\boldsymbol{x}_{best,g}$ is the best individual of the current generations, coefficient $F_i$of variation frequently chosen from the interval$(0,1+)$. In classical DE, $F_i = F,$ a fixed value is operated, throughout. For more mutation strategies see ref. [17].

Crossover: A process of generating child solution from one or more parent solutions is called crossover. After mutation, this operation constitutes a final test/offset vector

$$\boldsymbol{u}_{i,g} = (u_{1,i,g}, u_{2,i,g}, \ldots, u_{n,i,g}) \text{ [17]:}$$

$$\boldsymbol{u}_{j,i,g} = \begin{cases} v_{j,i,g}, if\ rand(a,b) \le CR_i\ or\ j = j_{rand}; \\ x_{j,i,g}, \qquad\qquad\qquad otherwise, \end{cases} \quad (3)$$

Whereas $rand(a,b)$is an unvarying arbitrary number in the interval $[a, b]$, and each i and j generated independently. Crossover probability $\mathrm{CR}_i \epsilon [0,1]$ is roughly equivalent to inherit the mutant vector, vector components from an average score. In classical DE, parameter $\mathrm{CR}_i = CR$ is fixed.

Selection: An operation or process of choosing between parent vector $\boldsymbol{x}_{i,g}$and trial vector $\boldsymbol{u}_{i,g}$ according totheir fitness value is called selection [17].

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_{i,g}, \quad if\ f\left(\boldsymbol{u}_{i,g}\right) < f\left(\boldsymbol{x}_{i,g}\right); \\ \boldsymbol{x}_{i,g}, \qquad\qquad otherwise. \end{cases} \quad (4)$$

DE runs different mutation strategies. The strategy DE/current-to-best/1/bin, due to its greedy nature by preferring best solution, may lead it to premature convergence. However, if the information of the best soultion(s) is/are properly used, then it may lead algorithm towards the true optimum. Also, DE needs the right set of parameters for different problems at different stages of evolution [18-20]. The three main parameters of DE are population size NP, the scaling factor F and the crossover rate Cr, which affect its performance. For best-tuning of these parameters, the time taken ``trial and error" mechanism is used. In order to overcome this problem, researchers have introduced adaptive and self-adaptive mechanisms [21, 22].

These mechanisms vigorously update parameters, without user's aforesaid knowledge of the relationship between the parameters and parameter's settings [17]. JADE is one of the adaptive versions of DE. It is briefly described below.

2.2 JADE

As stated above, JADE [17] is an enhanced and adaptive variant of DE. In JADE, the authors developed a new greedy scheme, i.e., "*DE/current-to-pbest/1*", which is the generalization of Eq. 2. Any of the top $100p$ %, $p \in (0, 1]$ solutions can be randomly chosen in "*DE/current-to-pbest"* to play the role for the single best solution in Eq. 2 [17]. It uses the best solution information and information of the other good solutions too. Although the proposed mutation mechanism is of greedy nature, it diversifies the population. Thus problem like premature convergence can be removed. JADE also adjusts the parameters F and CF adaptively, the details of which can be found in [17]. Further, it uses the same operations of crossover and selection is same as given in Eq. (4) and (5), respectively. Since JADE is designed for unconstrained optimization problems. For solving COPs, it requires some additional technique to handle constraints.

3. Constraints Handling Techniques

Constrained optimization problems (COPs) arise in many real-world applications, and are thus gaining a growing interest. By knowing the difficulty of EAs to handle constraints, researchers have hybridized different constraints handling strategies with EAs, and thus solved COPs. The most simple and common one of them is to use a penalty function method.

3.1 Penalty Function Methods

Penalty function methods are commonly used for handling constraints in COPs [4]. They transform a constrained problem into an unconstrained problem by defining a new evaluation function, where a penalty term is added to the original cost function [23], [24].

Penalty function method was initially derived by Cournt in 1940s [25] and later raised by Carol [26] and Fiacco and McCormick [27]. Mathematically, it is given by following equation:

$$\omega(\boldsymbol{x}) = f(\boldsymbol{x}) \pm \left[\sum r_i * G_i + \sum c_j * L_j\right] \quad (5)$$

wherein $\omega(\boldsymbol{x})$ is the new optimized objective function.G_i and L_j are functions of constraints g_i and h_j, respectively.r_i and c_j are positive constants, often referred to as "penalty factor". General forms of G_i and L_j are:

$$G_i = max[0, g_1(\boldsymbol{x})]^{\beta}\ ;\ L_j = |h_j(\boldsymbol{x})|^{\gamma} \quad (6)$$

There are different types of penalty functions. In this paper, we use a static penalty function with JADE, which is described below.

Table 1: Experimental Results Generated by Algorithm CJADE-S

Problems	Known Optimum	Best Achived	Worst	Median	Mean	**Std**
G01	-15.000000	-15.000001	-15.000000	-15.000000	-15.000000	9.285e-008
G02	-0.803619	-0.802539	-0.800449	-0.801265	-0.801247	4.955e-004
G03	-1.000000	-1.000704	-0.954599	-0.996941	-0.992415	1.271e-002
G04	-30665.539000	-30666.835783	-30671.779950	-30670.121586	-30669.946710	1.262e+000
G05	5126.498000	5126.496176	5269.055484	5126.496183	5145.088350	3.961e+001
G06	-6961.814000	-6983.814860	-6983.814860	-6983.814860	-6983.814860	4.502e-010
G07	24.306000	24.306166	24.341437	24.306167	24.309498	8.746e-003
G08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	1.416e-017
G09	680.630000	680.630046	680.630046	680.630046	680.630046	4.417e-011
G10	7049.331000	7033.978422	6947.378149	6981.506522	6983.423317	2.343e+001
G11	0.750000	0.749874	0.749874	0.745158	0.744601	4.502e-003
G12	-1	-1	-1	-1	-1	0
G13	0.053950	0.060123	0.793421	0.461116	0.441500	1.808e-001

3.1.1 Modification of JADE with Static Penalty Function(CJADE-S)

In a static penalty function, the penalty factor does not depends in any way on the current generation. It remains unchanged throughout the evolution. Homaifar et al. [4] proposed this technique, in its various levels of violation $(1,2,\dots,n)$ of constraints are defined by the handler. It is given as follows [4]:

$$F(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum R_{ij}\, max\left[0, g_j(\boldsymbol{x})\right]^2, \qquad (7)$$

where R_{ij} is a penalty parameter corresponding to i^{th} constraint violationandj^{th} constraint,and

$$g_j(\boldsymbol{x}) = \begin{cases} 0, & if\ g_j(\boldsymbol{x}) \le 0, 1 \le i \le p; \\ \left|g_j(\boldsymbol{x})\right|, & otherwise. \end{cases} \qquad (8)$$

Here,$g_j(\boldsymbol{x})$ are the inequality constraints.

This static penalty function converts equality constraints, $h_j(\boldsymbol{x}) = 0$ intoinequality constraints, $\left|h_j(\boldsymbol{x})\right| - \epsilon \le 0$, where ϵ is a small positive number.

In this paper, we used Eq. (7) in the selection scheme of JADE to penalize infeasible solutions. As a result, it gives us a new algorithm, denoted by CJADE-S for solving COPs.

4 Experimental Study

We evaluated the performance of CJADE-S on CEC2006 functions suit with constraints of different types [28]. Population size fixed at 100. For each test function, we run CJADE-S 25 time's idependently. Their best, worst, mean, median and standard deviation(std) data is obtained after 500000 function evaluations (FES). These statistics are shown in Table 1. We also plot the convergence graphs of each function againstgenerations, see Figures 1-13. All the functions graphs show the convergence towards the optimum in a smooth way, except the function G13 due to its exponential nature.

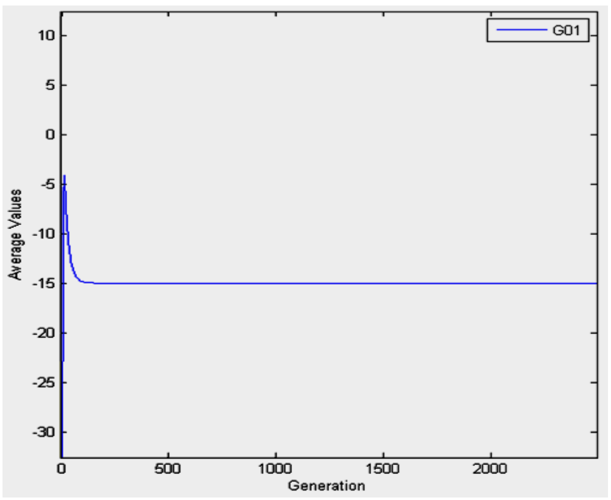

Fig. 1 Convergence graph of G01

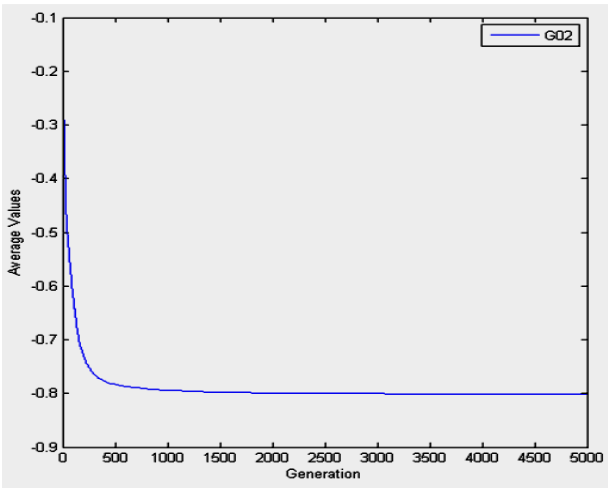

Fig. 2: Convergence graph of G02

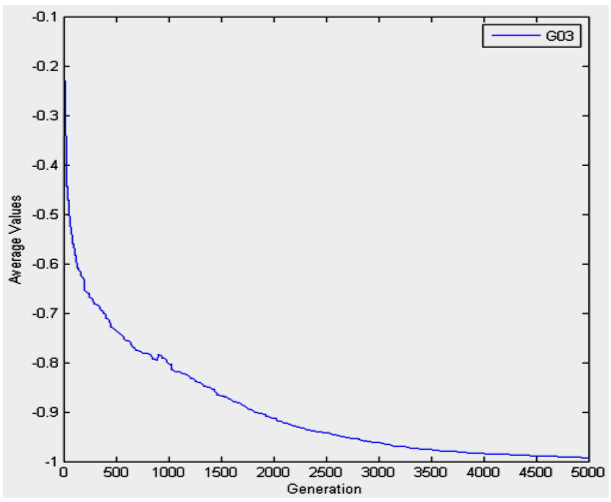

Fig. 3: Convergence graph of G03

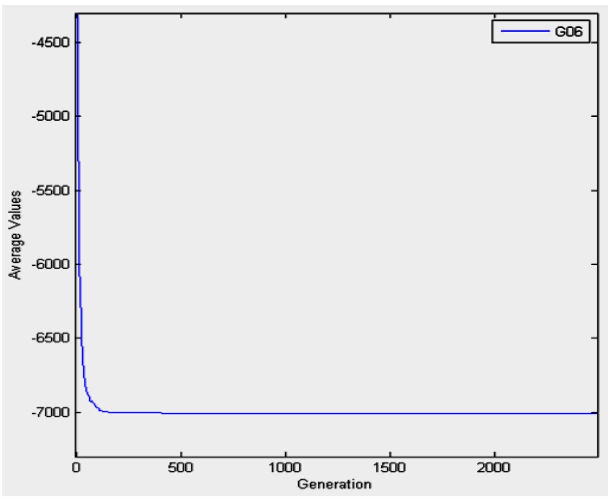

Fig. 6: Convergence graph of G06

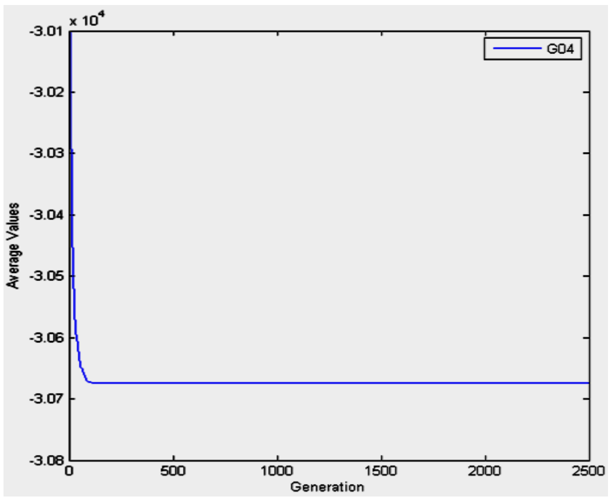

Fig. 4: Convergence graph of G04

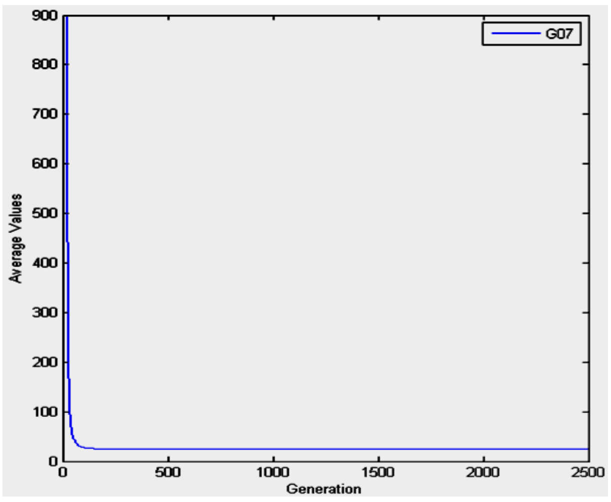

Fig. 7: Convergence graph of G07

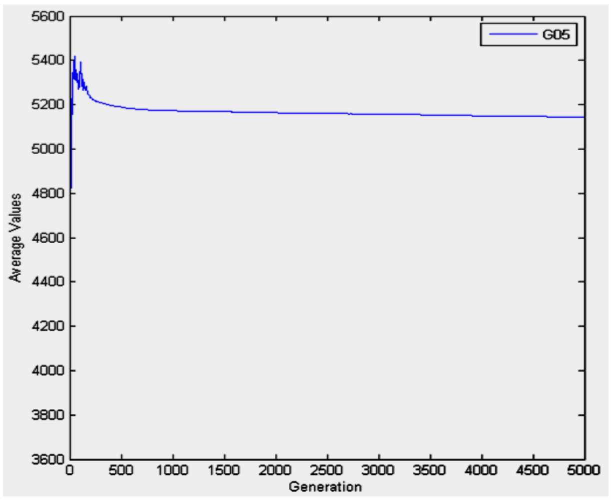

Fig. 5: Convergence graph of G05

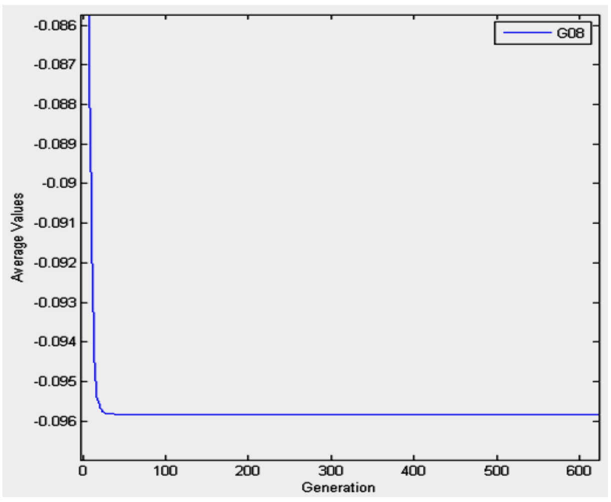

Fig. 8: Convergence graph of G08

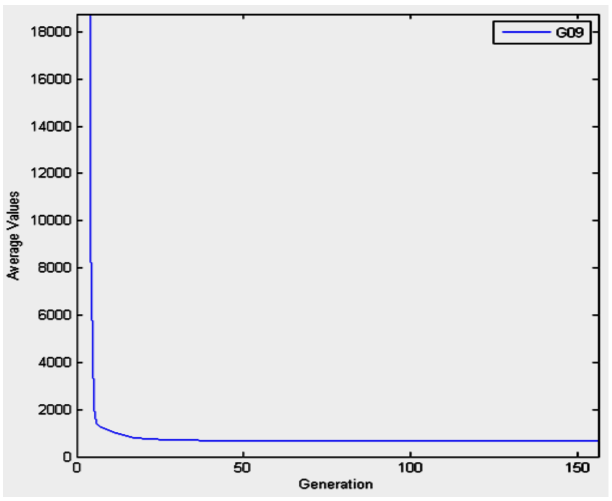

Fig. 9: Convergence graph of G09

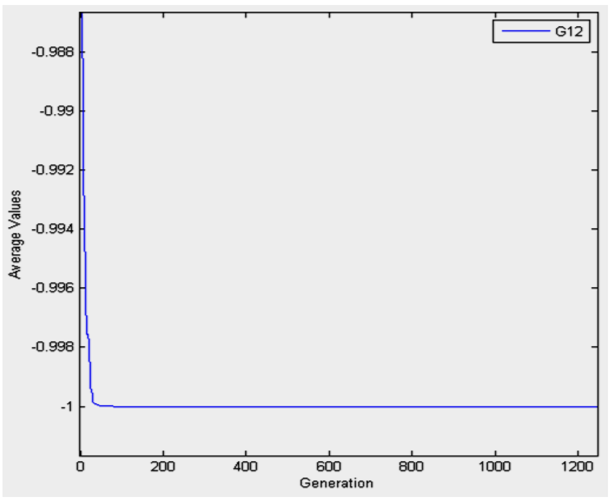

Fig. 12: Convergence graph of G12

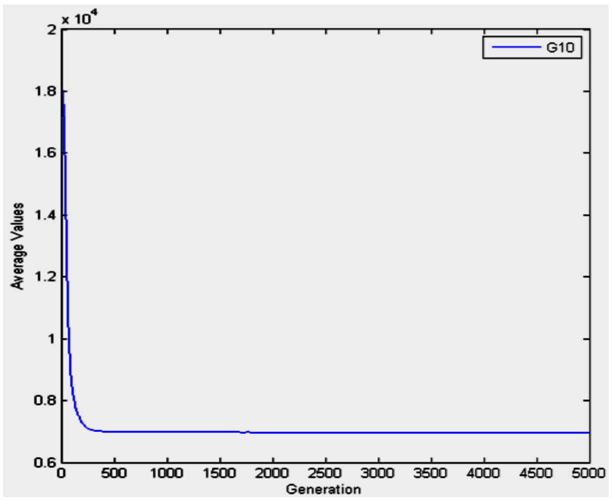

Fig. 10: Convergence graph of G10

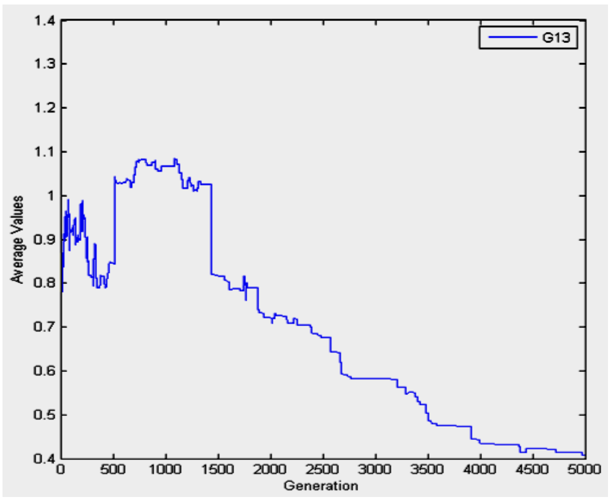

Fig. 13: Convergence graph of G13

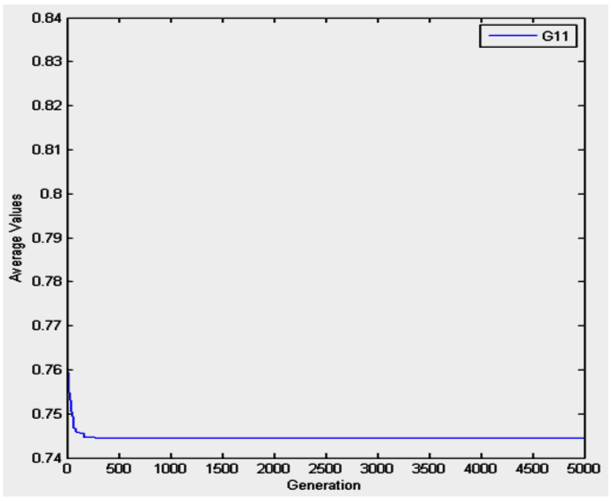

Fig. 11: Convergence graph of G11

5. Comparison

CJADE-S algorithm has been observed and showed good competitive results. The achieved values were very encouraging, as most problems have 100 percent result, some have results around the optimum. The results are compared with other algorithms too, shown in Table 2 and Table 3.

We set $MAX\ FES = 500000$ and $total\ runs = 25$ for each function and calculated

The results by using MATLAB® 7.12(R2011a). Table 2 shows the comparison between CJADE-S and DE with penalty [23]. The best, mean and median values are compared. Table 3 shows the comparison between CJADE-S and Simple Multi-member Evolution Strategy (SMES) [29]. The best, mean and median results are compared.

Table 2: Comparison of CJADE-Sand DE with penalty

	Optimum Best		Mean Values		Median Values	
Problems	CJADE-S	DE with penalty	CJADE-S	DE with penalty	CJADE-S	DE with penalty
G01	-15.000001	-15	-15.000000	-15	-15.000000	-14.9999927
G02	-0.802539	-0.46112	-0.801247	-0.43753	-0.801265	-0.4388441
G03	-1.000704	0.05618	-0.992415	0.05618	-0.996941	0.05618023
G04	-30666.835783	-30665.5	-30669.946710	-30665.5	-30670.121586	-30665.5387
G05	5126.496176	5126.767	5145.088350	5126.767	5126.496183	5126.766506
G06	-6983.814860	-6961.81	-6983.814860	-6961.81	-6983.814860	-6961.81388
G07	24.306166	24.35136	24.309498	24.37486	24.306167	24.37306
G08	-0.095825	-0.09583	-0.095825	-0.09583	-0.095825	-0.09582504
G09	680.630046	680.301	680.630046	680.6301	680.630046	680.6300574
G10	7033.978422	7049.312	6983.423317	7049.368	6981.506522	7049.364188
G11	0.749874	0.75	0.744601	0.75	0.745158	0.75
G12	-1	-1	-1	-1	-1	-1
G13	0.060123	0.05395	0.441500	0.05395	0.461116	0.053942

Both algorithms CJADE-S and DE (penalty) used the same static penalty. The results of optimum best value achived by functions G02, G03, G05, G07 and G09 from CJADE-S are better, and funtions G04, G06, G10, G11 and G13 shows better result from DE(penalty). The functions G01, G08 and G12 are equal. In compariosion of mean value, the functions G02, G03, G07 and G09 from CJADE-S are better, and funtions G04, G05, G06, G10, G11 and G13 shows better result from DE(penalty), The funtions G01, G08 and G012 are equal. The comparsion betwenn median values of algorithms, the functions G01, G02, G03, G05, G07 and G09 shown better results by CJADE-S algorithm and functions G04, G06, G10, G11 and G13 are better by DE(penalty). The function G08 and G12 are equal in results with both algorithms.

Table 3: Comparison of CJADE-S and Simple Multimember Evolution Strategy (SMES)

Problems	Optimum Best		Mean Values		Median Values	
	JADE-S	SMES	JADE-S	SMES	JADE-S	SMES
G01	-15.000001	-15.000	-15.000000	-15.000	-15.000000	-15.000
G02	-0.802539	0.803601	-0.801247	0.785238	-0.801265	0.792549
G03	-1.000704	1.000	-0.992415	1.000	-0.996941	1.000
G04	-30666.835783	-30665.539	-30669.946710	-30665.539	-30670.121586	-30665.539
G05	5126.496176	5126.599	5145.088350	5174.492	5126.496183	5160.198
G06	-6983.814860	-6961.814	-6983.814860	-6961.284	-6983.814860	-6961.814
G07	24.306166	24.327	24.309498	24.475	24.306167	24.426
G08	-0.095825	0.095825	-0.095825	0.095825	-0.095825	0.095825
G09	680.630046	680.632	680.630046	680.643	680.630046	680.642
G10	7033.978422	7051.903	6983.423317	7253.047	6981.506522	7253.603
G11	0.749874	0.75	0.744601	0.75	0.745158	0.75
G12	-1	1.000	-1	1	-1	1.000
G13	0.060123	0.053986	0.441500	0.166385	0.461116	0.061873

The above mentioned results are the outputs of CJADE-S and SMES algorithms. The results of optimum best value achived by functions G05, G07, G09 and G10 from CJADE-S are better, and funtions G02, G03,G04, G06, G11 and G13 shows better result from SMES algorithm. The functions G01, G08 and G12 are equal. In compariosion of mean value, the functions G02, G05, G07, G09 and G10 from CJADE-S are better in result and funtions G03, G04, G06, G11 and G13 shows better result from SMES. The funtions G01, G08 and G012 are equal. The comparsion betwenn median values of algorithms, the functionsG02, G05, G07, G09 and G10 shown better results at CJADE-S algorithm and functions G03, G04, G06, G11 and G13 are better at SMES. The functions G01, G08 and G12 are equal in median value results.

6. Conclusion

In this research work, we modify JADE algorithm to examine its capability for solving COPs. In proposed modified JADE, we introduce static penalty function in the selection scheme of JADE algorithm for handling optimization problems with constraints functions.

The performance of algorithm CJADE-S, is tested on known COPs, CEC2006. The experimental results show the better performance of CJADE-S on most of the problems.

In the future, the performance of CJADE-S algorithm will be compared with several techniques available in the research history, to check its effectiveness. We will study some more static penalty functions in this approach to enhance the reliability of the search technique for more COPs.

References:

[1] Goldberg, D.E "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, Reading, Massachusetts, 1998.

[2] D.V. Arnold, "Noisy Optimization With Evolution Strategies", Norwell, MA. Kluwer. 2002.

[3] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, "Evolutionary Algorithms for Solving Multi-Objective Problems", Norwell, MA:Kluwer, 2002.

[4] A. Homaifar, S.H.Y. Lai and X. Qi, "Constrained optimization via genetic algorithms", Simulation, vol. 62, pp. 242-254, 1994.

[5] Z. Michalewicz, K. Deb, M. Schmidt, and T. Stidsen, "Test-case generator for constrained parameter optimization techniques", IEEE Trans. Evol. Comput., vol. 4, no. 3, pp. 197-215. 2000.

[6] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithm for con-strained parameter optimization problems", Evol. Comput., vol. 4, no.1, pp. 1-32, 1996.

[7] T. Beack (Ed.), "Proceedings of the Seventh International Conference on Genetic Algorithms", Morgan Kaufmann, San Mateo, CA, 1997.

[8] D.B. Fogel, "Evolutionary Computation. Toward a New Philosophy of Machine Intelligence", The Institute of Electrical and Electronic Engineers, New York, 1995.

[9] M. Gen, R. Cheng, "Genetic Algorithms Engineering Design", Wiley, New York, 1997.

[10] D.E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, Reading, MA, 1989.

[11] Z. Michalewicz, "Genetic Algorithms + Data Structures Evolution Programs", 2nd Ed. Springer, Berlin, 1992.

[12] M. Mitchell, "An Introduction to Genetic Algorithms", MIT Press, Cambridge, MA, 1996.

[13] I. Parmee (Ed.), "The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation", Springer, Plymouth, UK, 1998.

[14] V.W. Porto, N. Saravanan, D. Waagen, A.E. Eiben (Eds.), "Evolutionary Programming" VII", Proceedings of the Seventh Annual Conference on Evolutionary Programming, Lecture Notes in Computer Science,1447, Springer, San Diego, CA, 1998.

[15] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces", International Computer Science Institute, Berkeley, Tech. Rep. TR-95-012, 1995.

[16] R. Storn and K. V. Price, "Differential evolution-A simple and efficient heuristic for global optimization over continuous Spaces", J. Global optim., vol. 11, pp. 341-359, 1997.

[17] J. Zhang and A. C.Sanderson, "JADE: Adaptive Differential Evolution with Optional External Archive", IEEE Transactions on Evolutionary Computation, vol 13, no 5, pp. 945-958, 2009 .

[18] R. Gamperle, S. D. Muller, and P. Koumoutsakos, "A Parameter study for differential evolution, "Proceedings of the International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation (WSEAS '02), Interlaken, Switzerland, pp. 11–15, February 2002.

[19] E. Mezura-Montes, J. Velzquez-Reyes, and C. A. Coello Coello, "A Comparative study of Differential Evolution variants for Global Optimization", Proceedings of the 8th Annual Conference on Genetic EvolutionaryComputation. C Seattle, Washington., USA, pp. 485-492, 2006..

[20] K. V. Price, R. M. Storn and J. A. Lampinen, "Differential Evolution: A Practical Approach to Global Optimization". 1st Ed. New York: Springer-Verlag, 2005

[21] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm", Proc. IEEE Congr. Evol. Comput., 1. Honolulu, HI, pp. 831-836, 2002

[22] Z. Yang, K. Tang, and X. Yao, "Self-adaptive Differential Evolution with Neighbourhood Search", Proc. of the 2008 IEEE Congress on Evolutionary Computing , Hong Kong, China, pp. 1110-1116, 2008.

[23] Er. Anuj Kumar Parashar, Dr. BDK Patro, Dr. C Patvardhan, "Constraint-Handling techniques for optimization using Differential Evolution", International Journal of Scientific & Engineering Research, vol. 4, no. 7, ISSN 2229-5518, 2013.

[24] Ozgu Yeniay, "Penalty Function Methods for con-strained optimization with genetic algorithms", Mathematical and Computational Applications, vol. 10, no. 1, pp. 45-56, 2005.

[25] R. Courant, "Variational methods for the solution of problems of equilibrium and vibrations", Bull. Am. Math. Soc., vol. 49, pp. 1-23, 1943.

[26] C.W. Carroll, "The created response surface technique for optimizing nonlinear restrained systems", Operations Research, vol. 9, pp. 169-184,1961.

[27] A.V. Fiacco, G.P. McCormick, "Extensions of SUMT for nonlinear programming: Equality constraints and extrapolation", Manage. Sci., vol. 12, no. 11, pp. 816-828, 1968.

[28] J. J. Liang, T. P. Runarssaon and P.N. Suganthan, Problems definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization", J. Appl. Mechanics, Technical Report, vol. 41, no. 8, 2006.

[29] E. M. Montes and C. A. C. Coello, "A Simple Multimembered Evolution Strategy to Solve Constrained Optimization Problems", IEEE Trans. Evol. Comput, vol. 9, no. 1, pp. 1-15, 2005.