Constraint-Handling techniques for optimization using Differential Evolution

Er. Anuj Kumar Parashar, Dr. BDK Patro, Dr. C Patvardhan

Abstract— Differential Evolution which is used in global optimization over continuous spaces. In general, the task is to optimize certain properties of a system by pertinently choosing the system parameters. Differential evolutions have been widely used to solve difficult constrained optimization problem. Differential evolution gives good results with unconstrained problem. In this project Differential evolution is used for constrained problem with constraints handling techniques. Penalty function is a most widely used constraints handling technique which is used in this project basically there are two kind of penalty function one is static penalty and other one is dynamic. Static penalty function is used here. Differential evolution with static penalty function is used here which gives great quality of solution. And in this project we compared the solution of Differential Evolution with Static penalty function and the evolutionary strategies with feasibility rules and stochastic ranking.

Index terms - Differential Evolution, Constraint Handling, Penalty Function, Static penalty function, Evolutionary Algorithm, Feasibility Rules, Stochastic Ranking

----------*----------

I. INTRODUCTION

We need the following basic components to implement an EA in order to solve a problem
[1]:
1. A representation of the potential solutions to the problem.
2. A way to create an initial population of potential solutions (this is normally done randomly, but deterministic approaches can also be used).
3. An evaluation function that plays the role of the environment, rating solutions in terms of their ''fitness''.
4. A selection procedure that chooses the parents that will reproduce.
5. Evolutionary operators that alter the composition of children (normally, crossover and mutation).
6. Values for various parameters that the evolutionary algorithm uses (population size, probabilities of applying evolutionary operators, etc.).

Testing of probabilistic population based algorithms like Evolutionary Algorithm (EA) is complicated as each run of the EA can produce different results due to the stochastic nature of the search process. Most of the papers, which propose a new algorithm, often have a comparative analysis section where previously known algorithms are compared on the quality of the solutions [2], [3]. EAs essentially performs unconstrained search i.e. there are no in built mechanisms in a typical EA to handle constraints seamlessly. Therefore, it is necessary to find ways of handling constraint. It is an important design decision as it affects the performance of the entire algorithm [4]. Constraints handling techniques Evolutionary Algorithms have been quite successful in a wide range of applications [5, 6, 7, 8, 9, 10, 11, 12]. However, an aspect normally disregarded when using them for optimization (a rather common trend) is that these algorithms are unconstrained optimization procedures, and therefore is necessary to find ways of incorporating the constraints (normally existing in any real-world application) into the fitness function.

The most common way of incorporating constraints into an Evolutionary Algorithms have been penalty functions (we will be referring only to exterior penalty functions in this paper) [13, 14]. However, due to the well-known difficulties associated with them [14], researchers in evolutionary computing have proposed different ways to automate the definition of good penalty factors, which remains as the main drawback of using penalty functions. Additionally, several researchers have developed a considerable amount of alternative approaches to handle constraints, mainly to deal with specific features of some complex optimization problems in which it is difficult to estimate good penalty factors or to even generate a single feasible solution. Penalty functions are the oldest approach used to incorporate constraints into unconstrained optimization algorithms (including Evolutionary Algorithms).

Penalty Function

The most common approach in the Evolutionary Algorithms community to handle constraints (particularly, inequality constraints) is to use penalties. Penalty functions were originally proposed by Courant in the 1940s [15] and later

expanded by Carroll [16] and Fiacco and McCormick [17]. The idea of this method is to transform a constrained-optimization problem into an unconstrained one by adding (or subtracting) a certain value to from the objective function based on the amount of constraint violation present in a certain solution.

$$\Phi(x)=f(x) \sim [\sum r_i * G_i + \sum c_j * L_j],$$

Where $\Phi(x)$ is the new (expanded) objective function to be optimized, G_i and L_j are functions of the constraints $g_i(x)$ and $h_j(x)$, respectively, and r_i and c_j are positive constants normally called ''penalty factors''. The most common form of G_i and L_j is

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

Ideally, the penalty should be kept as low as possible, just above the limit below which infeasible solutions are optimal (this is called, the minimum penalty rule [18, 19, 20]).

Static Penalty Function

Under this category, we consider approaches in which the penalty factors do not depend on the current generation number in any way, and therefore, remain constant during the entire evolutionary process. Homaifar et al. [21] proposed an approach in which the user defines several levels of violation, and a penalty coefficient is chosen for each in such a way that the penalty coefficient increases as we reach higher levels of violation. This approach starts with a random population of individuals (feasible or infeasible).
An individual is evaluated using [1]

$$\text{Fitness}(x) = f(x) + \sum(R_{ki} * \max[0,g_i(x)]^2),$$

Where Rk,i are the penalty coefficients used, m is the total number of constraints (Homaifar et al. [22] transformed equality constraints into inequality constraints), f(x) is the npenalized objective function and k =1, 2, . . . , l, where l is the number of levels of violation defined by the user. The idea of this approach is to balance individual constraints separately by defining a different set of factors for each of them through the application of a set of deterministic rules.

II. Differential Evolution

Differential Evolution (DE) is a population-based and directed search method [23], [24]. Like many other evolutionary algorithms, it starts with an initial population vector, which is randomly generated when no a priori knowledge about the solution space is available. Let us assume that $X_{i,G}(i = 1, 2, ...,N_P)$ are candidate solution vectors in the generation G (N_P: population size). Successive populations are generated by adding the weighted difference of two randomly selected vectors to a third randomly selected vector.

Mutation

For each vector $X_{i,G}$ in generation G a mutant vector $V_{i,G}$ is defined by

$$V_{i,G} = X_{a,G} + F(X_{b,G} - X_{c,G}),$$

where $i = \{1, 2, ...,N_P\}$ and a, b, and c are mutually different random integer indices selected from $\{1, 2, ...,N_P\}$. Further, i, a, b, and c are different so that $N_P >= 4$ is required. $F \in [0, 2]$ is a real constant which determines the amplification of the added differential variation of $(X_{b,G} - X_{c,G})$. Larger values for F result higher diversity in the generated population and lower values cause faster convergence.

Crossover

DE utilizes the crossover operation to increase the diversity of the population. It defines the following trial vector:
$U_{i,G} = (U_{1i},G,U_{2i},G, ...,U_{Di},G)$,
where D is the problem dimension and

$$U_{ji,G} = \begin{cases} V_{ji,G} \text{ if randj}(0, 1) <= Cr, \\ X_{ji,G} \text{ otherwise.} \end{cases}$$

$C_r \in (0, 1)$ is the predefined crossover rate constant, and randj(0, 1) is the jth valuation of uniform random number generator. Most popular values for C_r are in the range of (0.4, 1) [23].

Selection The approach that must decide which vector ($U_{i,G}$ or
$X_{i,G}$) should be a member of next (new) generation, G + 1. For a maximization problem, the vector with the higher fitness value is chosen. There are other variants based on different mutation strategies [24].

III. COMPARATIVE STUDY

A comparative study between Feasibility Rules, Stochastic Ranking constraint handling techniques using evolutionary strategies [25] and penalty function using differential evolution has been performed to determine their relative strengths. In order to perform fair holistic performance evaluation algorithm. A standard suit of 13 benchmark functions defined in [26], [27] were used for comparing the two techniques. Thirty independent runs have been performed for each problem in each experiment on both the constraint-handling techniques.

The test results of the algorithm for both the constraint handling techniques are summarized in Table I and Table II

The Results of the Best, Mean and Worst values of Objective function obtained by penalty function technique compared with optimal value is given in the Table I and the comparative study is given in Table II

IV. RESULTS

The Results of the Best, Mean and Worst values of Objective function obtained by penalty function technique compared with optimal value is given in the Table I

	DIFFERENTIAL EVOLUTION				
	Penalty				
#	BEST	WORST	MEAN	MEDIAN	OPTIMUM
G01	-15	-15	-15	-14.9999927	-15
GO2	-0.46112	-0.40569	-0.43753	-0.4388441	-0.80362
G03	0.05618	0.05618	0.05618	0.05618023	-1.0005
G04	-30665.5	-30665.5	-30665.5	-30665.5387	-30665.5
G05	5126.767	5126.767	5126.767	5126.766506	5126.497
G06	-6961.81	-6961.81	-6961.81	-6961.81388	-6961.81
G07	24.35136	24.41344	24.37486	24.37306	24.30621
G08	-0.09583	-0.09583	-0.09583	-0.09582504	-0.09583
G09	680.6301	680.6301	680.6301	680.6300574	680.6301
G10	7049.312	7049.467	7049.368	7049.364188	7049.248
G11	0.75	0.75	0.75	0.75	0.7499
G12	-1	-1	-1	-1	-1
G13	0.05395	0.05395	0.05395	0.053949848	0.053942

Table I

The comparison of DE with Penalty Function Technique, ES with Feasibility Rules and ES with Stochastic Ranking techniques are made and Results are good with all the functions Comparisons are given in Table II

#		Best			Mean			Worst		
	DE PF	ES FR	ES SR	DE PF	ES FR	ES SR	DE PF	ES FR	ES SR	OPTIMUM
G01	-15	-15	-15	-15	-15	-15	-15	-15	-15	-15
GO2	-0.46112	-0.80362	-0.80362	-0.43753	-0.76981	-0.76643	-0.80362	-0.40569	-0.80362	-0.80362
G03	0.05618	-0.02827	-1	0.05618	-0.00424	-0.99999	0.05618	0	-0.99998	-1.0005
G04	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5	-30665.5
G05	5126.767	5127.748	5126.498	5126.767	5284.984	5126.498	5126.767	5775.853	5126.498	5126.497
G06	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81	-6961.81
G07	24.35136	24.30639	24.30626	24.37486	24.33047	24.30858	24.41344	24.44928	24.33557	24.30621
G08	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583	-0.09583
G09	680.6301	680.6301	680.6301	680.6301	680.6301	680.6301	680.6301	680.6308	680.6301	680.6301
G10	7049.312	7049.248	7049.248	7049.368	7055.399	7050.646	7049.467	7163.841	7058.849	7049.248
G11	0.75	0.9921	0.75	0.75	0.9992	0.75	0.75	1	0.75	0.7499
G12	-1	-1	-1	-1	-1	-1	-1	-0.99244	-1	-1
G13	0.05395	0.864691	0.05395	0.05395	0.983599	0.16942	0.05395	1	0.438851	0.053942

Table II

Where DE PF is value obtained using Differential Evolution with Penalty Function technique

ES FR is value obtained using Evolutionary Strategies with Feasibility Rules

ES SR is value obtained using Evolutionary Strategies with Stochastic Ranking

V. CONCLUSION

Constraints handling is an important design issue in Evolutionary algorithms. The overall performance of the algorithm is dependent on both the constraint handling technique and Evolutionary mechanism. However, till date no fair comparison between any two constraint handling techniques on same Evolutionary Algorithm has ever been made. The authors generally propose a new constraint handling technique and implement it on an algorithm of their choice. They compare the proposed technique with an existing one implemented on an entirely different Evolutionary algorithms. Further, such comparisons are often made to highlight the improvement in objective functions

values and number of function evaluations consumed in specific situations.

VI. REFERENCES

1. Z. Michalewicz, Genetic Algorithms+Data Structures¼Evolution Programs, third ed., Springer, Berlin, 1996.

2. C. A. Coello Coello, "Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of state of the artc Computer. Methods Appl. Mech. Eng., vol. 191, no. 11/12, pp. 1245–1287, Jan. 2002.

3. J. J. Liang and P. N. Suganthan, "Comparison of Results on the 2006 CEC Benchmark Function Set," available at: http://www3.ntu.edu.sg/home/EPNSugan/indexfiles/CEC-06/Comparison-of-Results.pdf, 2006.

4. A. Mani and C. Patvardhan, "A Novel Hybrid Constraint Handling Technique for Evolutionary Optimization," Proc. IEEE CEC 2009, doi.ieeecomputersociety.org/10.1109/ CEC.2009.4983265.

5. T. B€ack (Ed.), Proceedings of the Seventh International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, July 1997.

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

7. M. Gen, R. Cheng, Genetic Algorithms & Engineering Design, Wiley, New York, 1997.

8. D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

9. Z. Michalewicz, Genetic Algorithms+Data Structures¼Evolution Programs, second ed., Springer, Berlin, 1992.

10. M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA, 1996.

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

12 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, vol. 1447, Springer, San Diego, CA, March 1998.

13 D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

14 J.T. Richardson, M.R. Palmer, G. Liepins, M. Hilliard, Some guidelines for genetic algorithms with penalty functions, in: J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, George Mason University, Morgan Kaufmann, Reading, MA, 1989, pp. 191–197.

15. R. Courant, Variational methods for the solution of problems of equilibrium and vibrations, Bull. Am. Math. Soc. 49 (1943) 1–23.

16. C.W. Carroll, The created response surface technique for optimizing nonlinear restrained systems, Operations Research 9 (1961) 169–184.

17. A.V. Fiacco, G.P. McCormick, Extensions of SUMT for nonlinear programming: Equality constraints and extrapolation, Manage. Sci. 12 (11) (1968) 816–828.

18.L. Davis, Genetic Algorithms and Simulated Annealing, Pitman, London, 1987.

19. R.G. Le Riche, R.T. Haftka, Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm, AIAA J. 31 (5) (1993) 951–970.

20. A.E. Smith, D.M. Tate, Genetic optimization using a penalty function, in: S. Forrest (Ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, University of Illinois at Urbana-Champaign, Morgan Kaufmann, San Mateo, CA, July 1993, pp. 499–503.

21.A. Homaifar, S.H.Y. Lai, X. Qi, Constrained optimization via genetic algorithms, Simulation 62 (4) (1994) 242–254.

22.R. Storn and K. Price, Differential Evolution- A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization 11, pp. 341-359, 1997

23.G.C. Onwubolu and B.V. Babu, New Optimization Techniques in Engineering, Berlin ; New York : Springer, 2004.

24. S. Das, A. Konar, U.K. Chakraborty, Two Improved Differential Evolution Schemes for Faster Global Search, Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 991-998, Washington, USA, 2005.

26. Sulabh, Ashish, and C Patvardhan Is Stochastic Ranking really better than Feasibility Rules for Constraint Handling in Evolutionary Algorithms? World Congress on Nature & Biologically Inspired Computing (NaBIC 2009)

27. T. P. Runarsson and X. Yao, "Stochastic Ranking for Constrained Evolutionary Optimization," IEEE Transactions On Evolutionary Computation,vol. 4, no.3, pp. 284-294, 2000.

28. J. J. Liang, Thomas Philip Runarsson, Efren Mezura-Montes,Maurice Clerc, P. N. Suganthan, Carlos A. Coello Coello, K. Deb, "Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-

Parameter Optimization," Technical Report, Nanyang Technological University, Singapore, Dec 2005. Available at : http://www.ntu.edu.sg/home/EPNSugan/.