A Study of Mechanisms to Handle Constraints in Evolutionary Algorithms

Efrén Mezura-Montes and Carlos A. Coello Coello[⋆]

Evolutionary Computation Group at CINVESTAV-IPN (EVOCINV)
Electrical Eng. Department, Computer Science Dept.
Av. IPN No. 2508 Col. San Pedro Zacatenco, México D.F. 07300, MÉXICO
emezura@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx

Abstract. In this work, we study different mechanisms to incorporate constraints into an evolutionary algorithm used for global optimization. The aim of the work is twofold. First, we propose a competitive constraint-handling approach which does not require a penalty function (nor penalty factors), and which is able to produce very competitive results while performing less fitness function evaluations than other algorithms representative of the state-of-the-art in the area. Second, we measure the rate at which our approach reaches either the feasible region of the search space or even the global optimum solution. Finally, we propose additional test functions and perform an empirical study that aims to find some of the features that make a constrained optimization problem difficult to solve by an evolutionary algorithm.

1 Introduction

Evolutionary algorithms (EAs) have been successfully used to solve different types of optimization problems. However, in their original form, they lack an explicit mechanism to handle the constraints of a problem. This has motivated the development of a considerable number of approaches to incorporate constraints into the fitness function of an EA [1]. Particularly, in this paper we are interested in the general nonlinear programming problem in which we want to: Find $\boldsymbol{x}$ which optimizes $f(\boldsymbol{x})$ subject to: $g_i(\boldsymbol{x}) \leq 0, \;\; i = 1, \ldots, n \; h_j(\boldsymbol{x}) = 0, \;\; j = 1, \ldots, p$ where $\boldsymbol{x}$ is the vector of solutions $\boldsymbol{x} = [x_1, x_2, \ldots, x_r]^T$, n is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or nonlinear). The most common approach adopted to deal with constrained search spaces is the use of penalty functions [1]. When using a penalty function, the amount of constraint violation is used to punish or "penalize" an infeasible solution so that feasible solutions are favored by the selection process. Nonetheless, the main drawback of penalty functions is that they require a careful fine tuning of the penalty factors that accurately estimates the degree of penalization to be applied so that we can approach efficiently the feasible region [1].

In this work we analyze the constraint handling (CH) problem in EAs. Besides the typical parameters defined by the user of an EA, constraint handling techniques usually

[⋆] The first author acknowledges support from the Mexican Consejo Nacional de Ciencia y Tecnología (CONACyT) through a scholarship to pursue graduate studies at CINVESTAV-IPN's. The second author acknowledges support from CONACyT through project number 42435-Y.

add more of them (e.g., penalty factors). Moreover, CH techniques are usually created either to solve a specific problem (e.g., in engineering design, or combinatorial optimization) or to be tested on the well-known benchmark of 13 test functions found in [2]. The features that make difficult a constrained problem for an evolutionary algorithm remains open and has attracted little interest in the specialized literature [3]. Also, there is a noticeable lack of metrics that allow a more in-depth comparison among CH techniques used with evolutionary algorithms. Our research work was precisely motivated by the previous issues. The main objectives of this work were to propose a competitive constraint-handling technique based on the separation of the objective function and the constraints of the problem. Aspects such as efficiency and simplicity were major concerns. Also, we aimed not to use a penalty function. As part of our work, we also proposed to use a measure that may help to know in more detail the rate at which our approach reaches the feasible region (or the global optimum) of the search space. This metric may also be used to compare different approaches. Finally, we performed a empirical study in order to known what characteristics make a constrained problem difficult to solve by an EA. In this regard, we emphasize the fact that the EA studied (our own) was found to be highly competitive in the benchmark described in [2], and still had difficulties to solve some of the 11 new test problems that we proposed.

		Best Result		Mean Result		Worst Result	
Problem	**Optimal**	**SMES**	**SR**	**SMES**	**SR**	**SMES**	**SR**
g01	−15.00	−15.000	−15.000	−15.000	−15.000	−15.000	−15.000
g02	0.803619	**0.803601**	0.803515	**0.785238**	0.781975	**0.751322**	0.726288
g03	1.00	1.000	1.000	1.00	1.000	1.000	1.000
g04	−30665.539	−30665.539	−30665.539	−30665.539	−30665.539	−30665.539	−30665.539
g05	5126.498	5126.599	5126.497	5174.492	5128.881	5304.167	5142.472
g06	−6961.814	−6961.814	−6961.814	−**6961.284**	−6875.940	−**6952.482**	−6350.262
g07	24.306	24.327	24.307	24.475	24.374	24.843	24.642
g08	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
g09	680.63	680.632	680.630	**680.643**	680.656	**680.719**	680.763
g10	7049.25	**7051.903**	7054.316	**7253.047**	7559.192	**7638.366**	8835.655
g11	0.75	0.75	0.75	0.75	0.75	0.75	0.75
g12	1.000	1.000	1.000	1.000	1.000	1.000	1.000
g13	0.053950	0.053986	0.053957	0.166385	0.057006	0.468294	0.216915

Table 1. Comparison of our SMES with respect to Stochastic Ranking (SR) [2]. A result in **boldface** means a better solution of the SMES

2 Description of Our Approach

In order to avoid the use of a penalty function, we explored the capabilities of evolutionary multiobjective optimization concepts such as Pareto dominance, when used as a selection criterion. We also studied the use of Pareto ranking and population-based approaches to solve global optimization problems [4]. In this regard, we performed a comparison among 4 representative approaches based on multiobjective optimization concepts [4]. Our results indicated that the separation of the objective function and the constraints is a viable mechanism but the use of multiobjective concepts was not particularly effective. Then, we also realized that the most competitive constraint-handling approaches normally use an evolution strategy. This led us to hypothesize the following: (1) The self-adaptation mechanism of an ES helps to sample the search space well enough as to reach the feasible region reasonably fast and (2) the simple addition of selection criteria based on feasibility to an ES should be enough to guide the search in such a way that the global optimum can be approached efficiently. The main issue here was how to maintain diversity and to avoid the total elimination of infeasible individuals from the population. The first mechanism that we proposed in [5] was based on a variant of a $(\mu + 1)$-ES coupled with three simple selection criteria based on feasibility. This version provided good results but presented premature convergence in some test problems and had difficulties to reach the feasible region in some others. The three selection criteria adopted were the following: (1) Between 2 feasible solutions, the one with the highest fitness value wins (assuming a maximization problem/task). (2) If one solution is feasible and the other one is infeasible, the feasible solution wins. (3) If both solutions are infeasible, the one with the lowest sum of constraint violation is preferred. Based on the fact that the diversity maintenance mechanism was not too effective, in [6] we proposed a second version of our algorithm, but now using a $(1 + \lambda)$-ES and adding a diversity mechanism which consisted on allowing solutions with a good value of the objective function to remain as a new starting point in the next generation of the search, regardless of their feasibility. The results improved with respect to the first version. However, the robustness was not as good as other state-of-the-art algorithms and, for one test function, no feasible solutions were found. We realized that the main problem with the second version of our approach had to do with the exploratory limitations of the $(1 + \lambda)$-ES adopted. Also, we realized that keeping infeasible solutions was not necessarily useful. What we needed was to keep infeasible solutions that lied close to the boundary between the feasible and the infeasible region. This motivated the development of the third version of our approach (reported in this paper), in which we use a $(\mu + \lambda)$-ES (a Simple Multimembered Evolution Strategy SMES) combined with the same three selection criteria adopted before [7] , a simple combination of discrete and intermediate panmictic recombination and a reduction of the initial stepsize value of the mutation operator. Furthermore, we now add an improved diversity mechanism which, although simple, provides a significant improvement in terms of performance. The detailed features of the improved diversity mechanism are the following: At each generation, we allow the infeasible individual with the best value of the objective function and with the lowest amount of constraint violation to survive for the next generation. We call this solution the best infeasible solution. In fact, there are two best infeasible solution at each generation, one from the μ parents and one from the λ offspring. Either of them can be chosen with a 50% of probability. This process of allowing the best infeasible solution to survive for the next generation happens 3 times every 100 during the same generation. Therefore, the same best infeasible solution can be copied more than once into the next population. However, this is a desired behavior because a few copies of this solution will allow its recombination with several solutions in the population, specially with feasible ones. Recombining feasible solutions with infeasible solutions in promising areas (based on the good value of the objective function) and close to the boundary of the feasible region will allow the ES to reach optimum solutions located in the boundary of the feasible region of the search space (which are known to be the most difficult to reach). When the selection process occurs, the best individuals among the parents and offspring are selected based on the three selection criteria previously indicated. The selection process will pick feasible solutions with a better value of the objective function first, followed by infeasible solutions with a lower constraint violation.

To evaluate the performance of the proposed approach we used the 13 test functions described in [2]. The test functions chosen contain characteristics that are representative of what can be considered "difficult" global optimization problems for an evolutionary algorithm. We performed 30 independent runs per function. A summary of our results and a comparison with respect to stochastic ranking [2] is shown in Table 1. Note that we also compared results with respect to other approaches, but stochastic ranking was found to be the most competitive and therefore it was chosen for inclusion in this paper. In Table 1, we can see that our approach was able to find the global optimum in seven test functions and it found solutions very close to the global optimum in all the others. Our new approach, which uses a $(100 + 300)$-ES, performed $240,000$ fitness function evaluations (FFE), and Stochastic Ranking performed $350,000$. Finally, an analysis of variance (ANOVA) was performed to detect sensitivity of the three parameters of the evolution strategy used (μ, λ and number of generations). The results indicated that the approach was not sensitive to any of these three parameters.

3 Measuring Performance

After discussing the quality, robustness and competitiveness of the our algorithm we wanted to verify the rate at which our algorithm reached the feasible region, because in real-world problems it is important for an optimization algorithm to provide reasonably good results (where "good" may mean feasible solutions) with a relatively low number of objective function evaluations. Therefore, we performed an analysis of the rate at which our approach reached the feasible region. For this sake, we monitored the percentage of feasible solutions in the population at every 200 generations (the total number of generations of our approach was 800). The results are presented in Figure 1.

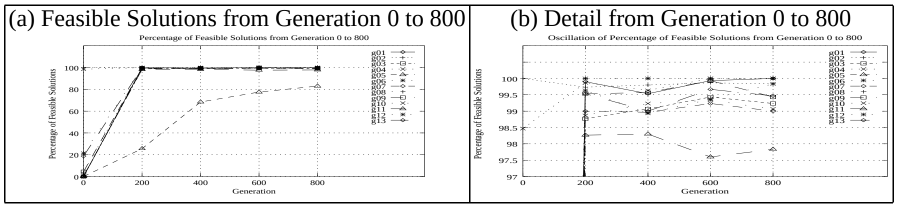

Fig. 1. Percentage of feasible solutions (a) every 200 generations (from 0 to 800), (b) a detailed oscillation of feasible and infeasible solutions (from 0 to 800)

It should be clear from Figure 1 that the feasible region is reached before generation 200 for all the 13 test functions. Our results also indicated that for problems where the global optimum was found, the algorithm required 200 generations on average to reach it. This suggests that our approach was able to find the global optimum relatively fast for different types of problems.

Problem	n	Type of function	ρ	Linear inequalities	Nonlinear inequalities	Linear equalities	Nonlinear equalities
g14	10	nonlinear	0.00%	0	0	3	0
g15	3	quadratic	0.00%	0	0	1	1
g16	5	nonlinear	0.0204%	4	34	0	0
g17	6	nonlinear	0.00%	0	0	0	4
g18	9	quadratic	0.00%	0	12	0	0
g19	15	nonlinear	33.4761%	0	5	0	0
g20	24	linear	0.00%	0	6	2	12
g21	7	linear	0.00%	0	1	0	5
g22	22	linear	0.00%	0	1	8	11
g23	9	linear	0.00%	0	2	3	1
g24	2	linear	79.6556%	0	2	0	0

Table 2. Dimensionality, type of objective function, values of ρ and type and number of constraints for 11 test problems proposed by the authors.

4 Identifying Sources of Difficulty

Most of the previous work on constraint-handling techniques relates to the benchmark proposed in [2]. However, we know (from the No Free Lunch Theorems for search [8]) that using such a limited set of functions does not guarantee, in any way, that an algorithm that performs well on them will necessarily be competitive in a different set of problems. This motivated us to identify new test functions to validate our algorithm. What we expected to find was functions in which our approach (which was found to be highly competitive in the traditional benchmark from [2]) did not exhibit a good performance. We expected to identify in such functions certain features that could be associated with sources of difficulty for a constraint-handling mechanism. Thus, we added 11 new problems taken from different sources, and which have features that we hypothesized that would decrease the performance of the algorithm (nonlinear equality constraints and dimensionality). The new functions are detailed in [9] and the summary of their features is presented in Table 2, where ρ is an estimate of the size of the feasible region with respect to the whole search space. We performed 30 independent runs using exactly the same parameters adopted with first set of test functions [7]. The overall results suggest that the two main factors that affect the performance of our EA are the dimensionality (this coincides with the conclusions from Schmidt & Michalewicz for the static penalty function approach [3]) and the increasing number of nonlinear equality constraints. The factors that do not seem to decrease the performance of our EA were a high number of inequality constraints (even nonlinear) and (somewhat surprisingly) the type of objective function. This small study is far from being conclusive, but it provides some insights regarding the factors that make difficult for an EA to reach the global optimum in constrained search spaces.

5 Conclusions and Future Work

We have proposed a novel approach to handle constraints, based on a multimembered evolution strategy and a separation of constraints from the objective function. The algorithm does not use a penalty function and also does not require the fine-tuning of any extra parameters. Despite its simplicity, the approach provided a highly competitive performance against an state-of-the-art technique at a lower computational cost (measured in terms of fitness function evaluations performed). In addition, we found

		Statistical Results of the SMES for the new 11 Problems				
Problem	**Optimal**	**Best**	**Mean**	**Median**	**Worst**	**St. Dev.**
g14	-47.656	-47.535	-47.368	-47.386	-47.053	1.33E-1
g15	961.715	$*961.698$	963.922	964.058	967.787	1.79E+0
g16	1.905	**1.905**	**1.905**	**1.905**	**1.905**	0
g17	8927.589	$*8890.183$	$*8954.136$	$*8948.686$	$*9163.677$	40.83E+0
g18	0.866	**0.866**	0.716	0.674	0.648	8.19E-2
g19	-32.386	-34.223	-37.208	-36.430	-41.251	2.10E+0
g20	0.0967	$*0.2114$	$*0.2511$	$*0.2524$	$*0.3044$	2.33E-2
g21	193.778349	$*347.980927$	$*678.392445$	$*711.847260$	$*985.782166$	158.49E+0
g22	12812.500	$*2340.617$	$*9438.255$	$*9968.156$	$*17671.535$	4360.88E+0
g23	NA	$* - 1470.152588$	$* - 363.508270$	$* - 333.251541$	$*177.252640$	316.16E+0
g24	-5.508	$\mathbf{-5.508}$	$\mathbf{-5.508}$	$\mathbf{-5.508}$	-5.507	1.0E-5

Table 3. Statistical results for the SMES with the 11 new test functions. "*" means infeasible.

that our algorithm tends to approach the feasible region at a high rate. Finally, in an additional study, we determined that the main sources of difficulty for our approach are related to high dimensionality and a high number of nonlinear equality constraints. Our future work consists on defining in a more formal way a metric for the rate at which the feasible region is reached. We also plan to analyze more carefully the new test functions in order to detect other features that may also affect the performance of our approach. We hope to use this information to design more effective mechanisms to incorporate constraints into the fitness function of an evolutionary algorithm.

References

1. Coello, C.A.C.: Theoretical and Numerical Constraint Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. Computer Methods in Applied Mechanics and Engineering **191** (2002) 1245–1287
2. Runarsson, T.P., Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization. IEEE Transactions on Evolutionary Computation **4** (2000) 284–294
3. Schmidt, M., Michalewicz, Z.: Test-Case Generator TCG-2 for Nonlinear Parameter Optimisation. In et al., M.S., ed.: Proceedings of PPSN VI, Heidelberg, Germany, Springer-Verlag (2000) 539–548 Lecture Notes in Computer Science Vol. 1917.
4. Hernández-Aguirre, A., Botello-Rionda, S., Coello, C.A.C., Lizárraga-Lizárraga, G., Mezura-Montes, E.: Handling Constraints Using Multiobjective Optimization Concepts. International Journal for Numerical Methods in Engineering **59** (2004) (accepted for publication).
5. Mezura-Montes, E., Coello, C.A.C.: A Simple Evolution Strategy to Solve Constrained Optimization Problems. In et al., E.C.P., ed.: Proceedings of GECCO'2003, Heidelberg, Germany, Springer Verlag (2003) 640–641 Lecture Notes in Computer Science Vol. 2723.
6. Mezura-Montes, E., Coello, C.A.C.: Adding a Diversity Mechanism to a Simple Evolution Strategy to Solve Constrained Optimization Problems. In: Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003). Volume 1., Piscataway, New Jersey, IEEE Service Center (2003) 6–13
7. Mezura-Montes, E., Coello, C.A.C.: An Improved Diversity Mechanism for Solving Constrained Optimization Problems using a Multimembered Evolution Strategy. In: Proceedings of GECCO'2004, Heidelberg, Germany, Springer Verlag (2004) (accepted for publication).
8. Wolpert, D.H., Macready, W.G.: No Free Lunch Theorems for Optimization. IEEE Transactions on Evolutionary Computation **1** (1997) 67–82
9. Mezura-Montes, E., Coello, C.A.C.: What Makes a Constrained Problem Difficult to Solve by an Evolutionary Algorithm. Technical Report EVOCINV-01-2004, CINVESTAV-IPN, México (2004) Available at http://www.cs.cinvestav.mx/˜constraint/.