ON IMPROVING MULTIOBJECTIVE GENETIC ALGORITHMS FOR DESIGN OPTIMIZATION

S. Narayanan and S. Azarm
Department of Mechanical Engineering
University of Maryland
College Park, Maryland 20742
USA

1. Abstract

The paper presents some new improvements to Multi-Objective Genetic Algorithms (MOGAs) as reported in the literature. MOGA modifies certain operators within the GA itself to produce a multiobjective optimization technique. The improvements are made to overcome some of the shortcomings in niche formation, stopping criteria and interaction with a design decision-maker. The technique involves filtering, mating restrictions, and the use of objective constraints. A step-by-step procedure for an improved MOGA has been developed and demonstrated via two multiobjective engineering design examples: (i) two-bar truss design, and (ii) vibrating platform design.

2. Keywords

Multiple objectives, genetic algorithms, design optimization.

3. Introduction

A general Multi-Objective Optimization Problem (MOOP) with m objective functions is expressed as,

$$\begin{aligned} &\text{Minimize} \quad \mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}),\ldots,f_i(\mathbf{x}),\ \ldots,f_m(\mathbf{x})\} \\ &\text{subject to:} \quad \mathbf{x} \in D \\ &\qquad\qquad D = \{\mathbf{x}:\ g_j(x) \le 0,\ j=1,\ldots,J,\ h_k(\mathbf{x}) = 0,\ k=1,\ldots,K\} \end{aligned} \tag{1}$$

Due to the conflicting design objectives in a MOOP, the term "minimize" generally refers to a solution (hereafter called a Pareto solution) around which there is no way of improving any objective without worsening at least one other objective (see, for example, [1]). In the paper, to distinguish between a Pareto point (or set) in transition (i.e., Pareto for the current population) and a MOOP Pareto point (or final solution set), the former is referred to as a non-inferior point or set while the latter is referred to as a Pareto point or set.

Genetic Algorithms (GAs) have been modified to handle multiple objectives [2] so that the problem need not be converted into a series of single objective optimization problems. These modified GAs have come to be known as Multi-Objective Genetic Algorithms (MOGAs). MOGAs rely on a fitness assignment strategy different from that of single objective GAs. In MOGAs, a number of schemes have been proposed for forming a fitness function (see, for example, Fonseca and Fleming [2] and Jones et al.[3], wherein further references can be found).

In this paper, some new improvements to the MOGA developed by Fonseca and Fleming [2] have been proposed. The main shortcomings of the current MOGA techniques are described in Section 4. This is followed with an overview of some proposed improvements to MOGA in Section 5. Two engineering design examples that demonstrate the application of the improved MOGA is presented in Section 6. The present paper is a short version of the work reported by Narayanan and Azarm [4] wherein additional details are given.

4. Some MOGA Problems

MOGA obtains the Pareto set in an all-at-once manner and thus it must be able to sample as large a Pareto set as possible and produce solution points which are uniformly spread across the set. Also, the smoothness of the Pareto set may depend on how early the MOGA converges. That in turn depends on the stopping criteria used in the algorithm. For single objective GAs, one of many different stopping criteria can be used depending on the problem. But stopping criteria such as "no improvement in an objective function value for N generations" cannot be used in a MOGA. This calls for a technique to detect when the Pareto set has been obtained. Three main problems are identified, as discussed next.

Problem 1: The first problem is how to guide the population out of closed-form niches and encourage it to sample as large a Pareto set as possible. In order to achieve this, two requirements need to be met. Firstly, the method needs to identify "groups" or "clusters" of individuals in the non-inferior set. There are usually several niches that form early on during the design evolution. As MOGA proceeds, these niches tend to spread out, but the process can be hastened

considerably by building in some sort of rule at the parent selection stage. Secondly, the method needs to prevent these niches from deepening and growing. This would involve a method to prevent parents from the same region or niche from mating to produce offspring.

Problem 2: The second problem is how to impose stopping criteria which reliably detect when the Pareto set has been obtained and whether or not the Pareto set is "uniformly spread". A uniformly spread Pareto set should not have groups of solutions clustered together in some places and gaps in other places. These gaps and clusters are the result of niche forming tendencies of populations.

Problem 3: The third problem is how to give the designer the flexibility to choose a particular region of the Pareto set in order to recursively zoom into the region. An advantage of an interactive method like this is that the population size needed at a given stage in the process could be very small. For example, if the designer chooses an a posteriori MOOP method (e.g., see the review in [4]), a detailed Pareto set might have to be generated. This would involve using a large population because it has been noted that the number of points on the Pareto set is almost equal the population size itself, since the population tends to become non-inferior and inferior solutions are weeded out early. Hence, an interactive method would involve a small population that converges much faster than a large population. The disadvantage would be that it would be more difficult to spread out the small population across the entire set.

5. Improvements to MOGAs

A four pronged approach is developed to resolve the above mentioned problems (see [4], for details). The modifications described next make improvements over existing practices (see, for example, [2]) in MOGAs. The thrust is in making MOGA interactive with the designer and incorporating decision-making strategies in the fitness assignment stage. The approach also involves filtering, mating restrictions and the idea of objective constraints. The stopping criteria introduced helps in detecting when the Pareto set has been obtained.

Stopping criteria modification: Developing a stopping criteria for a MOGA is more difficult than those for a single objective GA. To detect improvement in the non-inferior set between one generation and the next, one has to maintain two sets of non-inferior solutions. A three-step scheme based on a L_2 norm (the Euclidean norm) has been implemented here to simultaneously detect improvement and spread uniformity of the non-inferior set.

The scheme is as follows. First, for each individual in the non-inferior set, the distance from the ideal point (or a good point as identified by the designer) is computed. Hence, for each generation, one set of L_2 norms is obtained. Second, the mean and standard deviation of the L_2 norms are calculated. Third, as the non-inferior set improves across generations, the points on the set get closer and closer to the ideal point. Therefore, the distance of each point on the set from the ideal point decreases as the convergence gradually is achieved. This decrease in L_2 norms can be measured by their mean, and the spread can be measured by the increase in standard deviation. If the improvement in the mean is less than some small number (tolerance), the MOGA can be assumed to have converged and is stopped.

Filtering and mating restrictions improvement: At each generation, once the non-inferior set has been obtained, a filtering routine can go through the population and delete some individuals from each niche. The number of individuals to delete would depend on how crowded the niche is, i.e., what the density is. This process is similar to "infighting" observed within natural communities, although the reason there is competition for scarce resources. In the MOGA, the reason is to help the designer focus clearly on the problem and understand it better. The deleted individuals are then replaced by those randomly generated.

The observation that parents in a niche tend to proliferate extensively leads one to the conclusion that less mating among close "cousins" might lead to a more uniform sampling of the Pareto set. Hence, a limit using a distance metric in the objective function space could be used in preventing close parents from mating. One way to implement it would be to compute the distance between the two parents selected for reproduction. If this distance is less than some limiting distance (tolerance), the parents shall not be allowed to mate, otherwise, they are allowed to mate. The limiting distance would depend on how dense the niches are and what the spread of the population is.

Objective constraints improvement: Once the entire spread of the non-inferior set is shown to the designer, (s)he should have the flexibility to zoom into a desired region by imposing objective constraints. This essentially involves (i) pausing the GA, (ii) imposing the necessary constraints, and (iii) resuming the GA. This process can be carried out every time the GA converges or as often as the designer wishes. The MOGA can be easily adapted to prevent any individual violating the objective constraints from reproducing.

Constraint evaluation improvement: Constraints are usually evaluated for all individuals in the population. However, if they are evaluated only for the non-inferior individuals instead of the entire population, then it is likely that substantial saving in computational effort is obtained. Moreover, if a constraint is violated by a non-inferior individual, it is undesirable to completely preventing it from reproducing into the next generation. This is because such an

individual may contain some genes that might give rise to future Pareto solutions. For this reason, only non-inferior individuals are evaluated for constraint violation and if any constraint is violated, a mild penalty is imposed in the form of reduced fitness. This reduced fitness is achieved by assigning the rank of N to the individual violating the constraint (N is the population size).

6. Examples

6.1. A Two-Bar Truss Example

This example was originally formulated in the literature [5] as a single objective two-bar truss problem.The problem has been modified here to a two-objective problem:

$$\text{Minimize } f_{volume} = x_1(16+y^2)^{1/2} + x_2(1+y^2)^{1/2}$$

$$\text{Minimize } f_{stress,AC} = \frac{20(16+y^2)^{1/2}}{yx_1}$$

Subject to: (2)

$$f_{volume} \leq 0.1$$

$$f_{stress\ AC} \leq 100000$$

$$f_{stress\ BC} \leq 100000$$

$$1 \leq y \leq 3$$

$$x_1, x_2 > 0$$

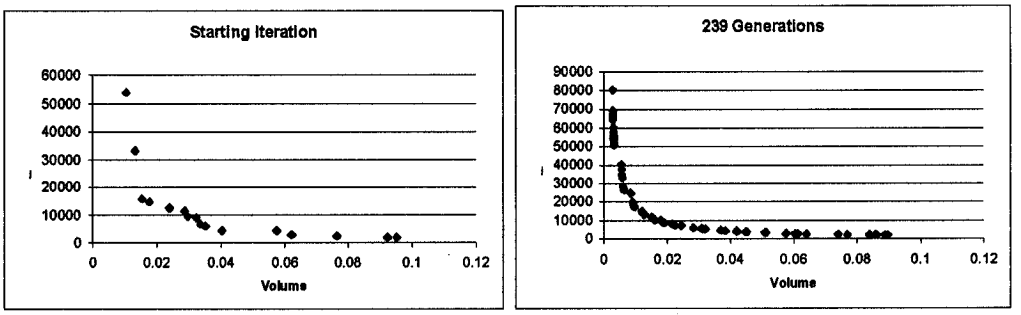

Figure 1. Two-bar truss solution results with the improved MOGA in the objective space

Fig. 1 shows the non-inferior set after a number of iterations. The total number of function evaluations used to obtain the Pareto set is 9,523. This number is significantly less than that obtained with MOGA without improvements (i.e., 27,296). The number however may change significantly depending upon the GA parameters.

6.2. A Vibrating Platform Example

The second example is a modification of a vibrating platform problem that was originally formulated as a single objective being the maximization of fundamental frequency, with an estimated cost being one of the constraints [6]. It has been modified here to include cost as the second objective, and also by making the problem combinatorial (both material and geometry of the platform is synthesized). The problem is to design a platform with a motor mounted on it. The machine setup is simplified as a pin-pin supported beam carrying a weight. A vibratory disturbance is imparted from the motor onto the beam, which is of length L, width b, and symmetrical about its mid-plane. The beam has a sandwich structure made up of three-layer materials 1, 2, and 3. Variables d_1 and d_2, respectively, locate the contact of materials 1 and 2, and 2 and 3. Variable d_3 locates the top of the beam. The combinatorial variables M_i refers to the material type for layer i (i=1,2,3) that each of the three layers 1, 2, and 3 of the beam can be made of. The mass density (ρ), Young's modulus of elasticity (E), and cost per unit volume (c) for each material type is also given in Table 1.

Material type M_i	ρ (Kg/m^3)	E(N/m^2)	c (\$/volume)
1	2,770	$70x10^9$	1,500
2	100	$1.6x10^9$	500
3	7,780	$200x10^9$	800

Table 1. Vibrating platform example – material properties

The objective functions are the fundamental frequency, f_1, and the cost of the setup, f_2. The complete formulation is as follows:

$$\begin{aligned}
&\text{Maximize } f_1(d_1,d_2,d_3,b,L,M_i) = (\pi/2L^2)(EI/\mu)^{1/2} \\
&\quad EI = (2b/3)[E_{M_1} d_1^{\,3} + E_{M_2}(d_2^{\,3} - d_1^{\,3}) + E_{M_3}(d_3^{\,3} - d_2^{\,3})] \\
&\quad \mu = 2b[\rho_{M_1} d_1 + \rho_{M_2}(d_2 - d_1) + \rho_{M_3}(d_3 - d_2)] \\
&\text{Minimize } f_2(d_1,d_2,d_3,b,M_i) = 2b[c_{M_1} d_1 + c_{M_2}(d_2 - d_1) + c_{M_3}(d_3 - d_2)] \\
&\quad \text{Subject to:} \qquad (3) \\
&\quad g_1 = \mu L - 2800 \le 0 \\
&\quad g_2 = d_2 - d_1 - 0.01 \le 0 \\
&\quad g_3 = d_3 - d_2 - 0.01 \le 0 \\
&\quad g_4 = 0.05 \le d_1 \le 0.5 \\
&\quad g_5 = 0.2 \le d_2 \le 0.5 \\
&\quad g_6 = 0.2 \le d_3 \le 0.6 \\
&\quad g_7 = 0.35 \le b \le 0.5 \\
&\quad g_8 = 3 \le L \le 6
\end{aligned}$$

The aim is to design the sandwich beam, i.e., find L, b, d_1, d_2, d_3, M_1, M_2, M_3, in order to minimize the vibration of the beam that results from the motor disturbance, as well as to minimize cost. This example was solved with the improved MOGA and the results are given in Fig. 2.The convergence was again somewhat faster with the improvements (127 generations as compared to 150 generations without) and the number of function evaluations required is 2115. In addition, as a result of greater sharing between non-inferior solutions and mating restrictions, the final number of the solutions detected, which is 9, is also greater than the MOGA without the improvements, which was 7.

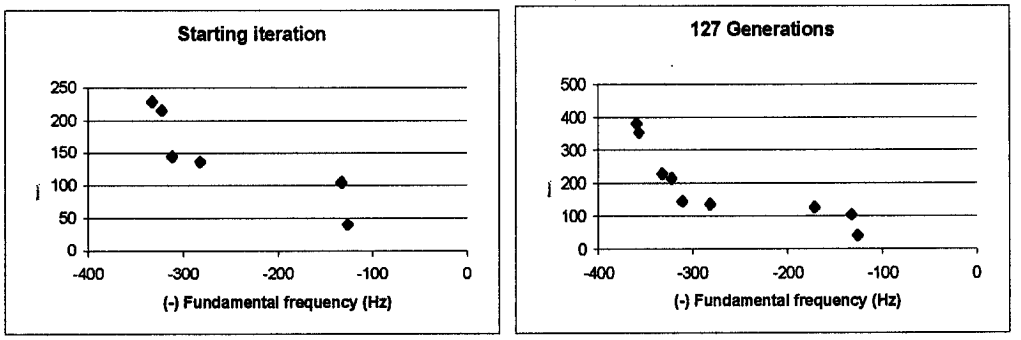

Figure 2. Evolution of non-inferior set (MOGA with improvements) for the vibrating platform

7. Conclusion

Four new improvements are made in MOGA. They include (i) introduction of a new stopping criterion that is based on statistical rather than deterministic metrics, (ii) a metric to ensure uniformity and spread of Pareto solutions, (iii) incorporating an interactive decision-making technique in the fitness assignment stage, and (iv) reducing computation time by eliminating some constraint evaluations. The MOGA presented in this paper can detect the Pareto set reliably and quickly. It can also obtain the Pareto set irrespective of whether or not it's in a non-convex region. This is especially useful in engineering design problems where non-convex Pareto sets are encountered. MOGA with the improvements also 'spreads' out in search of the Pareto set, thereby detecting a larger range of the Pareto optimal points when compared with MOGA without the improvements.

It can be clearly seen that for the vibrating platform problem, there are only a few Pareto solutions. This is not surprising, since in a discrete problem there are a finite number of Pareto points and niche formation deters the detection of all Pareto solutions. On the other hand, for continuous variable problems such as the two-bar truss, the Pareto set is essentially a continuous curve, i.e., it is always possible to find a Pareto solution which lies in the neighborhood of any two Pareto points. Therefore, in continuous problems, two close Pareto points can be used as parents to generate another Pareto point in the neighborhood. That is precisely how the MOGA proceeds, it can thus keep proliferating and gradually detect a large part of the Pareto set.

As shown in the paper , the stopping criterion introduced can greatly reduce the number of function evaluations. An important part of this criterion is to assess when the Pareto set has been obtained. The L_2-based technique described does detect, in a heuristic sense, the Pareto set. There is however scope for further improvement in assessing the quality of the Pareto set in a more reliable and quantitative terms. This aspect should be further investigated as part of the future research in this area.

8. Acknowledgements

The work presented in this paper was supported in part by NSF grant DMI-9700059, Maryland Industrial Partnerships,and by an ONR contract N000149810842.

9. References

[1] Eschenauer, H., Koski, J., and Osyczka, A.(Editors), 1990, Multicriteria Design Optimization, Springer-Verlag, New York.

[2] Fonseca, C.M. and Fleming, P.J., 1998, "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms-Part I: A unified formulation," IEEE Trans. on Systems, Man, Cyber., 28(1), 26-37.

[3] Jones, D. F., Tamiz, M., and Mirrazavi, S. K. 1998, "Investigation into the Incorporation and Use of Multiple Objectives in Genetic Algorithms," *Proceedings of MOPGP '98*, Quebec City, Quebec, Canada.

[4] Narayanan, S., and Azarm, S., 1999, "On Improving Multiobjective Genetic Algorithms for Design Optimization," Structural Optimization (accepted).

[5] Kirsch, U. 1981, *Optimal structural design*. New York: McGraw-Hill.

[6] Messac A. 1996, "Physical Programming: Effective Optimization for Computational Design," *AIAA Journal*, 34(1), 149-158.