AN APPROACH TO MULTIOBJECTIVE OPTIMIZATION USING GENETIC ALGORITHMS[1]

CARLOS A. COELLO COELLO AND ALAN D. CHRISTIANSEN[2]
Department of Computer Science, Tulane University, New Orleans, LA 70118, USA

ABSTRACT:
In this paper, we present a technique that uses a combination of the genetic algorithm (GA) and the global criterion method to find the optimum, in the min-max sense, of a multiobjective optimization design problem with multiple constraints. The objectives may be conflicting and noncommensurable, and the technique can deal with minimization and maximization problems (or a mixture of both). The optimum, in the min-max sense, gives a solution that treats all the objectives on terms of equal importance, and presents the advantage of being very efficient and easy to implement. Furthermore, when the min-max approach is combined with the weighting method, we can generate the set of Pareto (nondominant) solutions for both convex and nonconvex problems. Taking advantage of the floating point representation used for the GA, we could solve design problems that involve a mix of continuous, discrete and integer design variables. This technique was tested with multiobjective engineering design problems found in the literature, and our results were compared with traditional mathematical programming techniques that use other search strategies.

In this paper, we present a technique that uses a combination of the genetic algorithm (GA) and the global criterion method to find the optimum, in the min-max sense, of a multiobjective optimization design problem with multiple constraints. The objectives may be conflicting and noncommensurable, and the technique can deal with minimization and maximization problems (or a mixture of both). The optimum, in the min-max sense, gives a solution that treats all the objectives on terms of equal importance, and presents the advantage of being very efficient and easy to implement. Furthermore, when the min-max approach is combined with the weighting method, we can generate the set of Pareto (nondominant) solutions for both convex and nonconvex problems. Taking advantage of the floating point representation used for the GA, we could solve

[1] This work was supported in part by EPSCoR grant: NSF/LEQSF (1992-93)-ADP-04.

[2] Coello, Carlos A. & Christiansen, Alan D. "An Approach to Multiobjective Optimization Using Genetic Algorithms", In Dagli, C. H., Akay, M., Chen, C. L. P., Fernández, B. R., and Ghosh, J. (editors), *Intelligent Engineering Systems Through Artificial Neural Networks*, Volume 5, Fuzzy Logic and Evolutionary Programming, pp. 411-416, ASME Press, St. Louis Missouri, USA, November, 1995.

design problems that involve a mix of continuous, discrete and integer design variables. This technique was tested with multiobjective engineering design problems found in the literature, and our results were compared with traditional mathematical programming techniques that use other search strategies.

INTRODUCTION

Engineering optimization is currently a very active area of research. However, it has been until recently that more attention has been focused on multiobjective problems, even when it is well known that real-world applications normally have several (possibly conflicting) objectives to be optimized at the same time. To deal with these problems, a lot of mathematical programming techniques have been developed (Cohon and Marks, 1975), (Hwang and Masud, 1979), (Hwang et al., 1980), (Stadler, 1984).

On the other hand, the potential of genetic algorithms (GAs) for multiobjective optimization was recognized since its early days (Rosenberg, 1967). Since then, several attempts have been made to combine the objective functions in different ways (Fonseca and Fleming, 1994) from simple weighted sum approaches (Jakob et al., 1992) to target vector optimization (Wienke et al., 1992). Our approach is very simple and easy to implement, and allows the generation of the Pareto set of solutions, providing a single final optimum solution in the min-max sense.

STATEMENT OF THE PROBLEM

The multiobjective optimization problem can be defined as follows:

Find the vector $\bar{x}^* = \left[x_1^*, x_2^*, \ldots, x_n^*\right]^T$ which will satisfy the *m* inequality constraints:

$$g_i(\bar{x}) \geq 0 \qquad \mathrm{i} = 1,2,\ldots,\mathrm{m} \tag{1}$$

the *p* equality constraints

$$h_i(\bar{x}) = 0 \qquad \mathrm{i} = 1,2,\ldots,\mathrm{p} \tag{2}$$

and optimize the vector function

$$\bar{f}(\bar{x}) = \left[f_1(\bar{x}), f_2(\bar{x}), \ldots, f_k(\bar{x})\right]^T \tag{3}$$

where $\bar{x} = \left[x_1, x_2, \ldots, x_n\right]^T$ is the vector of decision variables. In other words, we wish to determine from among the set of all numbers which satisfy equations (1) and (2), the particular set $x_1^*, x_2^*, \ldots, x_k^*$ which yields the optimum values of all the objective functions.

OUR APPROACH

In our implementation, we used the equation

$$L_p(x) = \left[\sum_{i=1}^{k} w_i^p \left| \frac{f_i(x) - f_i^0}{f_{i\max} - f_i^0} \right|^p \right]^{1/p} \quad (4)$$

to generate the set of non-dominated solutions, with different weights (w_i) for the given objectives, and p=1. In this equation, $f_{i\max}$ is the worst possible value for criterion i; $f_i(\bar{x})$ is the result of implementing decision $\bar{x}$ with respect to the ith criterion and f_i^0 is the ideal solution for objective i. Additionally, we compute the min-max optimum considering equal weights for all the objectives. It should be mentioned that even though is necessary that the user provides the different weight combinations that he wants to try, it is not necessary to perform any sort of scaling of the fitness function since we are measuring relative deviations from the ideal vector, and the noncommesurable nature of the objectives is not relevant for our technique. Our system allows the automatic encoding and decoding of design variables, and the use of a mix of continuous, discrete and integer variables. The operation of our system can be described as follows:

1. The user should provide the objective functions and equality and inequality constraints imposed by the problem.
2. Different sets of weights should be input to the system such that their respective sum is always one.
3. The design variables should be defined, together with their desired precision, type and range. If the variables are discrete, then the set of possible values should be provided. If they are continuous, then the precision required will be requested.
4. The system will automatically encode the design variables, choosing the most appropriate representation scheme for the GA. It will normally use floating point representation for continuous variables and binary for integer and discrete variables.
5. The user may modify the preset parameters of the GA (population size=400, tournament selection, two-point crossover, max. number of generations=50).
6. Generate *n* processes, each corresponding to a different objective function.
7. Generate *m* processes, each corresponding to a different weight combination provided by the user. Since there is no relationship between any of these processes, we may generate the Pareto set by running separate processes with different weight combinations at the same time. Within each process, we run the GA 81 times (we loop the crossover and mutation rates from 0.1 to 0.9 in a nested manner). The final result will be the best of all these runs. The fitness function consisted of the maximum relative deviation between the given solution and the ideal vector.
8. Output the Pareto set of solutions.
9. Output the optimum solution in the min-max sense.

It should be noticed that we focused this work on the generation of the Pareto-optimal set of solutions assuming that the set of weights is given. However, it is possible to generate such weight combinations using the GA itself (Hajela and Lin, 1992). This is, nevertheless, an additional problem that has to take into account other considerations (i.e., the designer can impose additional constraints based on his/her own experience, and we have to decide how many weight combinations will be used), and that will not be addressed in this work.

EXAMPLES

Our first example is the design of an I-beam. The details of this example can be found in (Osyczka, 1985). This problem has two (conflicting) objective functions: minimize the cross section of the beam while minimizing its deflection, and four design variables. We generated the four Pareto-optimal solutions presented by Hajela and Lin (1992), and we included the ideal solution vector and the optimum value in the min-max sense. To evaluate our results, we used the maximum deviation from the optimum, which is defined by

$$L_p(f) = \sum_{i=1}^{n} w_i \left| \frac{f_i^0 - f_i(x)}{\rho_i} \right| \tag{5}$$

where n is the number of objective functions, and $\rho_i = f_i^0$, or $f_i(x)$, depending on which gives the maximum value for $L_p(f)$. Our results are shown on Table 1. As you can see, our approach gives slightly better results than the Branch-and-Bound algorithm used by Hajela and Lin (1992). The optimum solution in the min-max sense is obtained when $w_1 = w_2 = 0.5$, and it turns out to be slightly better using our approach. Notice that in this problem, all design variables were considered continuous.

Our second example is the design of a machine tool spindle taken from Schenauer et al. (1990). This problem has two conflicting objective functions (minimize the volume of the spindle while minimizing its static displacement) and four design variables that define the geometry of the spindle. Unfortunately, we could not find enough results in the literature to compare with our approach for this problem. Nevertheless, we compared two different weight combinations with the results produced by CAMOS (Schenauer et al., 1990). It should be pointed out that CAMOS generates solutions that sometimes violate one of the constraints imposed in this problem, whereas our approach generates only completely valid solutions. As it can be seen from the results, even in the presence of such constraint violation, our technique produced a better solution in one of the two cases of comparison. The remaining results were generated only by our approach, and include the ideal solution vector, and a subset of the Pareto-optimal set of solutions. Notice that two of the decision variables are discrete.

TABLE 1: PARETO-OPTIMAL AND OPTIMUM SOLUTIONS FOR THE I-BEAM OF THE FIRST EXAMPLE									
Method	w_1	w_2	f_1	f_2	x_1	x_2	x_3	x_4	$L_p(f)$
Hajela	1	0	127.443	0.005934	61.78	40.81	0.9	0.9	0.03
GA	1	0	127.413	0.061616	60.3822	41.4937	0.9	0.9	0.0
Hajela	0	1	850.000	0.005903	80.00	50.00	5.0	5.0	0.0
GA	0	1	850.000	0.005903	80.00	50.00	5.0	5.0	0.0
Hajela	0.45	0.55	307.53	0.0127	79.99	49.99	0.9	2.39	1.269441
GA	0.45	0.55	333.0058	0.011658	80.00	50.00	0.9	2.6579	1.262331
Hajela	0.55	0.45	276.55	0.0143	80.00	50.00	0.9	2.083	1.283902
GA	0.55	0.45	281.3624	0.014051	80.00	50.00	0.9	2.132	1.285694
Hajela	0.65	0.35	247.88	0.0163	79.99	50.00	0.9	1.791	1.231026
GA	0.65	0.35	237.4179	0.017119	80.00	50.00	0.9	1.6845	1.226213
Hajela	0.80	0.20	206.14	0.0205	80.00	39.79	0.9	1.725	0.988876
GA	0.80	0.20	177.1722	0.024727	80.00	50.00	0.9	1.071	0.950208
Hajela	0.50	0.50	291.43	0.01351	79.99	49.99	0.9	2.235	1.28798
GA	0.50	0.50	305.9026	0.012794	80.00	50.00	0.9	2.3819	1.284127

CONCLUSIONS AND FUTURE WORK

Our results show that our system provides good solutions when compared to other weigthed min-max approaches. Besides the two engineering problems indicated in this paper, we have tested it with other more complicated optimization problems, such as the counterweight balancing of a robot arm, and the design of plane and space trusses. This approach is very simple and easy to implement, and it avoids the need for scaling the fitness function. Furthermore, it is very efficient and it can generate both the Pareto set of solutions and a single final optimum solution in the min-max sense. We have seen how our implementation can handle a mixture of integer, continuous and discrete design variables, and how it is able to deal with maximization and minimization problems or any mixture of them. We also provided with a way of measuring the quality of a certain solution to a multiobjective optimization problem when using the weighted min-max method. Genetic search has proven to be very efficient and reliable even in the presence of complex constraints and objectives. A lot of work remains to be done in terms of expanding our system and improving the user interface. More GA-based techniques and mathematical programming techniques will be incorporated in the near future. Finally, we are also interested on experimenting with different genetic operators and selection strategies that have been previously used by the scientific community interested on multiobjective optimization using GAs.

TABLE 2: PARETO-OPTIMAL AND OPTIMUM SOLUTIONS FOR THE MACHINE TOOL SPINDLE OF THE SECOND EXAMPLE									
Method	w_1	w_2	f_1	f_2	x_1	x_2	x_3	x_4	$L_p(f)$
Schenauer	0.30	0.70	694101.0	0.0230779	66.454	183.365	95.00	85.00	0.411114
GA	0.30	0.70	682106.223	0.0215272	71.649	187.826	95.00	90.00	0.338192
Schenauer	0.70	0.30	531059.8	0.0301825	63.894	183.286	85.00	80.00	0.328228
GA	0.70	0.30	536509.715	0.030518	63.999	187.826	85.00	80.00	0.342329
GA	1	0	474658.099	0.0371834	59.999	187.813	80.00	75.00	0.0
GA	0	1	1645985.12	0.0166127	25.00	190.466	95.00	90.00	0.0
GA	0.50	0.50	671530.495	0.0216886	71.999	187.826	95.00	90.00	0.360155
GA	0.20	0.80	862859.245	0.0194481	65.378	187.826	95.00	90.00	0.300111
GA	0.80	0.20	474767.012	0.0371792	59.995	187.818	80.00	75.00	0.247783

REFERENCES

Cohon, J. L., and Marks, D. C. (1975). A review and evaluation of multiobjective programming techniques. *Water Resources Research*, 11(2), Apr, 208-220.

Fonseca, C. M., and Fleming, P. J. (1994). An overview of evolutionary algorithms in multiobjective optimization. Technical report, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, U.K.

Hajela, P., and Lin, C. Y. (1992). Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4, 99-107.

Hwang, C. L., and Masud, S. M. (1979). Multiple objective decision-making methods and applications. In *Lecture Notes in Economics and Mathematical Systems*, volume 164, Springer-Verlag, New York.

Hwang, C. L. Paidy, S. R., and Yoon, K. (1980). Mathematical programming with multiple objectives: a tutorial. *Computing and Operational Research*, 7, 5-31.

Jakob, W., Gorges-Schleuter, M., and Blume, C. (1992). Application of genetic algorithms to task planning and learning. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature, 2nd Workshop, Proceedings*, Lecture Notes in Computer Science, North-Holland Publishing Company, Amsterdam, 291-300.

Osyczka, A. (1985). Multicriteria optimization for engineering design. In John S. Gero, editor, *Design Optimization*, Academic Press, 193-227.

Rosenberg, R. S. (1967). Simulation of genetic populations with biochemical properties. PhD Thesis, University of Michigan, Ann Harbor, Michigan.

Schenauer, H. A., Osyczka, A., and Schäfer. (1990). Interactive multicriteria optimization in design process. In H. Schenauer, J. Koski, and A. Osyczka, editors, *Multicriteria Design Optimization. Procedures and Applications*, chapter 3, Springer-Verlag, Berlin, 71-114.

Stadler, W. (1984). A survey of multicriteria optimization or the vector maximum problem, part I : 1776-1960. *Journal of Optimization Theory and Applications*, 29(1), Sep, 1-52.

Wienke, P. B., Lucasius, C. and Kateman, G. (1992). Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta*, 265(2), 211-225.