Optimal Design of a Welded Beam via Genetic Algorithms

Kalyanmoy Deb*
University of Alabama, Tuscaloosa, Alabama 35486

Introduction

THE welded beam structure[1] is a practical design problem that is often used as a bench-mark problem in testing different optimization techniques. This problem is one of a family of structural optimization problems, which consists of a nonlinear objective function and five nonlinear constraints. There exist a number of optimization techniques[2] that are successfully used in solving such problems. Some of these methods, like geometric programming,[3] require an extensive problem formulation prior to the optimization procedure. Other methods, such as gradient search techniques, require derivative information that may not exist for others. This paper considers the application of a genetic algorithm (GA) in obtaining optimal design parameters for a welded beam structure. GAs are systematic search procedures—both global and efficient—based on the mechanics of natural genetics. GAs search through large spaces quickly even though they only require payoff information. Furthermore, because of the processing leverage associated with GAs, the method has a much more global perspective than many common methods in engineering optimization techniques. GAs have been applied to a variety of optimization problems—engineering, social sciences, physical sciences, computer sciences, biology, and others.[4] In the welded beam problem described here, GAs are compared with other optimization techniques and found to have surprising speed of convergence to near-optimal solutions. Simulation results suggest that GAs can be used to solve other problems of this class with similar efficiency.

Welded Beam

The welded beam assembly is shown in Fig. 1. The system consists of the beam A and the weld required to secure the beam to the member B. The objective is to find a feasible set of dimensions h, l, t, and b (denoted by $\bar{x} = [x_1, x_2, x_3, x_4]$) to carry a certain load (F) and still have a minimum total fabricating cost. The problem illustrated here is identical to the welded beam problem optimized via traditional techniques.[1,3] The objective function, $f(x)$, is the total fabricating cost that mainly comprises of the set-up cost, welding labor cost, and material cost:

$$\min \quad f(\bar{x}) = (1 + c_1)x_1^2 x_2 + c_2 x_3 x_4 (L + x_2) \tag{1}$$

where c_1 and c_2 are the cost of unit volume of weld material and bar stock, respectively. The associated functional constraints are $\tau_d - \tau(\bar{x}) \geq 0$; $\sigma_d - \sigma(\bar{x}) \geq 0$; $x_4 - x_1 \geq 0$; $P_c(\bar{x}) - F \geq 0$; $0.25 - \delta(\bar{x}) \geq 0$, and physical bounds on the variables are $0.125 \leq x_1 \leq 5$; $0.1 \leq x_2 \leq 10$; $0.1 \leq x_3 \leq 10$; $0.1 \leq x_4 \leq 5$, where τ_d = design shear stress of weld, $\tau(\bar{x})$ = weld shear stress, σ_d = design normal stress for bar material, $\sigma(\bar{x})$ = maximum bar bending stress, $P_c(\bar{x})$ = bar buckling load, and $\delta(\bar{x})$ = bar end deflection. Thus, the complete design problem consists of the minimization of the cost function [Eq. (1)] subject to the above constraints and physical bounds.

Workings of Genetic Algorithms

GAs are search procedures based on the mechanics of natural genetics and natural selection. Darwin's survival of the fittest principle is combined with simulated genetic operators abstracted from nature to form a generic search procedure. Detailed discussion on mechanisms of GAs can be found in the existing literature (Goldberg,[4] Holland[5]). It will suffice here to mention that GAs differ from traditional methods of search and optimization in a number of ways: 1) GAs work with a coding of the design variables as opposed to the design variables themselves—continuity of parameter space is not a requirement; 2) GAs work with a population of points as opposed to a single point—parallel processing of points reduces the chance of getting 'stuck' into a false optima; and 3) GAs require only the objective function values—minimal requirements broaden GA's application. In most GAs, finite-length binary-coded strings of ones and zeros are used to describe the parameters for each solution. In a multiparameter optimization problem, individual parameter codings are usually concatenated into a complete string. To decode a string, bit strings of specified length are extracted successively from the original string and individual substrings are then decoded and mapped into the desired interval in the corresponding solution space.

Three main operators responsible for the workings of GAs are reproduction, crossover, and mutation. The reproduction operator allows highly productive strings to live and reproduce, where the productivity of an individual is defined as a string's non-negative objective function value. There are many ways to achieve effective reproduction. Here, a simple proportionate scheme that selects individual strings based on their objective function values is used. The second operator, crossover, used with a specified probability p_c, exchanges genetic information by first cutting two strings at random and then joining the first part of one string with the second part of the other string. The third operator, mutation, is the occasional alteration of a string position with specified mutation probability p_m. These operators produce two new strings that become the members of the new population. This process continues until the population is filled with new individuals. The mechanics of reproduction and crossover operators are simple, involving string copies and partial string exchanges. However, their combined action is responsible for much of a genetic algorithm's power. A more rigorous understanding of their operators may be obtained by examining the processing of similarities among the strings.[4] Simply stated, a schema (schemata, plural), as defined by Holland,[5] is a similarity

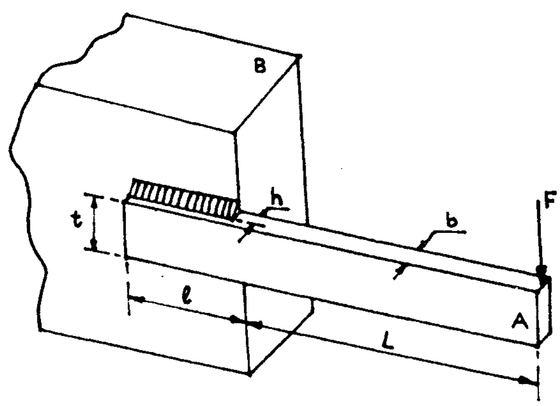

Fig. 1 The welded beam structure.

Received Feb. 1, 1990; presented at the AIAA/ASME/ASCE/AHS/ASC 31st Structures, Structural Dynamics, and Materials Conference, Long Beach, CA, April 2–4, 1990; revision received Sept. 10, 1990; accepted for publication Sept. 10, 1990.

*Graduate Student, Department of Engineering Mechanics; currently Visiting Research Assistant Professor, Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, IL. Student Member AIAA.

template describing a subset of strings with similarities at certain string positions. If we define * as a "don't care" symbol that represents either a 1 or a 0, then the schema $H = *1*0*$ represents eight strings having a 1 at the second position and a 0 at the fourth position. Two terms used in the GAs context to describe a schema H more specifically are its order, $o(H)$, and its defining length, $\delta(H)$. The order is defined as the number of defined positions (with ones and zeros) in H, and the defining length is defined as the distance between the outermost defined positions in H. In the earlier example, $o(H) = 2$ and $\delta(H) = 4 - 2 = 2$. With these parameters and operators discussed earlier, a lower bound on the expected number of any schema H in a population at generation $t+1$ may be computed from its known expected number $m(H, t)$ at generation t, as (Goldberg[4])

$$m(H,t+1) \geq m(H,t)\frac{f(H)}{f_{\text{avg}}}\left[1 - p_c\frac{\delta(H)}{l-1} - p_m o(H)\right] \quad (2)$$

where l is the string length, f_{avg} the average fitness of the population, and $f(H)$ the fitness of the schema H, defined as

$$f(H) = \frac{\sum_{s_i \in H} f(s_i)}{m(H,t)}$$

This is the fundamental theorem of genetic algorithms. Equation (2) shows that low-order, highly fit, and short defining length schemata receive exponentially increasing numbers of copies in succeeding generations. The schemata with these properties are called building blocks. These schemata build higher order schemata as the generations proceed, leading to optimal or near-optimal solutions. Furthermore, this processing goes on in parallel on many schemata [$\mathcal{O}(n^3)$, n is the population size] without special memory or processing.[4] This leverage of parallel processing of schemata (called implicit parallelism) gives GAs its rapid search capability.

In a constrained function optimization using GAs, penalty function methods are often used. In this Note, a bracket operator penalty term that assigns a squared-like penalty to the infeasible points and no penalty to boundary or feasible points is used.

Simulation Results and Discussion

The GA parameters used in the simulation runs are as follows: population size = 100; string length = 40; substring length for each parameter = 10; probability of crossover = 0.9; and probability of mutation = 0.01. High crossover probability, low mutation probability, and moderate population size considerations are consistent with De Jong's suggestions.[6]

Figure 2 shows the best-of-generation cost (total cost of fabrication) for three independent runs with different initial populations. The plot shows that the cost decreases with successive generations. A comparison of near-optimal solutions generated by genetic algorithms with the best result of a number of traditional optimization methods, illustrated in Ragsdell and Phillips,[3] is also shown in this figure. It is interesting to note that near-optimal solutions are obtained after only about 15 generations with approximately $0.9 \times 100 \times 15$ (or 1350) new function evaluations because at each generation only 0.9×100 or 90 points are new. It has been observed that constraints corresponding to the best solution were never violated in the simulation runs and they asymptotically satisfy the equality conditions with generation number. In other words, GAs find an optimal solution that optimally satisfies all constraints.

The welded beam problem has been attempted using different optimization techniques discussed in Siddall.[2] The best result obtained by some of these methods—APPROX, DAVID, GP, SIMPLEX, and RANDOM—as illustrated in Ragsdell and Phillips,[3] is reproduced in Table 1. Following these values, the best results obtained by three different genetic algorithms are shown. We see that all three GA runs are comparable to that of the other methods. Figure 3 compares the best design parameter values obtained via three genetic algorithms with the best results from the earlier methods. The percentage difference of three independent runs with GAs from the optimal is also shown. In all three cases, near-optimal results are obtained even though a genetic algorithm explores only a fraction of the huge search space (1350 function evaluations in a search space of $2^{40} = 1.099 \times 10^{12}$). In APPROX and DAVID methods, first-order derivative corresponding to each variable is required, limiting their application to a variety of problems. On the other hand, SIMPLEX and RANDOM methods do not require any derivative information and perform no better than GA simulations. Geometric programming method (GP) requires the formation of dual variables prior to the optimization procedure and that

Table 1 Comparison of genetic algorithms with other methods

	Methods	x_1	x_2	x_3	x_4	Cost
	APPROX	0.2444	6.2189	8.2915	0.2444	2.38
	DAVID	0.2434	6.2552	8.2915	0.2444	2.38
	GP	0.2455	6.1960	8.2730	0.2455	2.39
	SIMPLEX	0.2792	5.6256	7.7512	0.2796	2.53
	RANDOM	0.4575	4.7313	5.0853	0.6600	4.12
Three independent GA runs	GA1	0.2489	6.1730	8.1789	0.2533	2.43
	GA2	0.2918	5.2141	7.8446	0.2918	2.59
	GA3	0.2679	5.8123	7.8358	0.2724	2.49

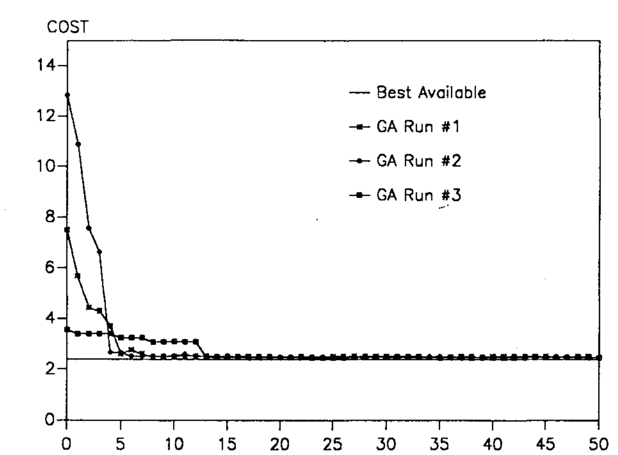

Fig. 2 Best-of-generation fabrication cost vs generation number.

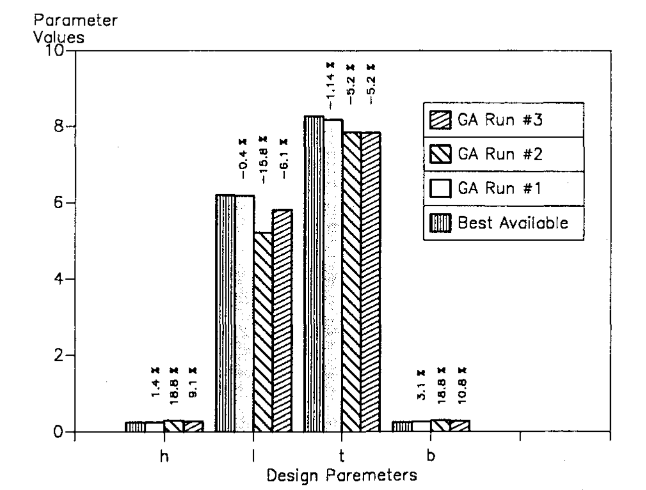

Fig. 3 Design parameter comparison—genetic algorithms vs the best of traditional methods.

may be cumbersome in certain problems. Even though no other problem information except the objective function values is used, GAs quickly find a solution that is comparable with other methods.

Conclusion

In this paper, a genetic algorithm has been used to optimize a welded beam structure consisting of a highly nonlinear objective function with five nonlinear constraints. A simple GA with reproduction, crossover, and mutation operators is able to converge fast to near-optimal design parameters after examining only a small fraction (one in over 800 million) of the search space. The primary advantages of GAs are that no gradient or other auxiliary problem information is required in the search process. Moreover, the implicit parallel processing of the problem information makes GAs less likely to converge to a sub optimal solution. Because of their simplicity in operation and minimal requirements, GAs are finding increasing popularity across a broad cross section of engineering problems. The successful operation in this particular application, which represents a large class of similar structural design problems, will broaden GAs applicability to a wider spectrum of disciplines.

Acknowledgments

I wish to thank David E. Goldberg for introducing me to genetic algorithms and for many insightful suggestions and stimulating discussions about genetic algorithms and their applications. I am also grateful for support provided by National Science Foundation under Grant CTS-8451610 and Graduate Research Council Fellowship from the University of Alabama, Tuscaloosa.

References

[1]Rekliatis, G. V., Ravindran, A., and Ragsdell, K. M., *Engineering Optimization Methods and Applications,* Wiley, New York, 1984.

[2]Siddall, J. N., *Analytical Decision-Making in Engineering Design,* Prentice-Hall, Englewood Cliffs, NJ, 1972.

[3]Ragsdell, K. M., and Phillips, D. T., "Optimal Design of a Class of Welded Structures Using Geometric Programming," *ASME Journal of Engineering for Industries,* Series B, Vol. 98, No. 3, 1976, pp. 1021–1025.

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

[5]Holland, J. H., *Adaptation in Natural and Artificial Systems,* Univ. of Michigan Press, Ann Arbor, MI, 1975.

[6]De Jong, K. A., "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," *Dissertation Abstracts International,* Vol. 36, No. 10, 5140B, University of Michigan Press, Ann Arbor, MI (University Microfilms 76-9381), 1975.