AIAA JOURNAL
Vol. 37, No. 2, February 1999

Using the Two-Branch Tournament Genetic Algorithm for Multiobjective Design

William A. Crossley,* Andrea M. Cook,† and David W. Fanjoy‡
Purdue University, West Lafayette, Indiana 47907-1282
and
Vipperla B. Venkayya§
U.S. Air Force Research Laboratory, Wright–Patterson Air Force Base, Ohio 45433-7542

The two-branch tournament genetic algorithm is presented as an approach to determine a set of Pareto-optimal solutions to multiobjective design problems. Because the genetic algorithm searches using a population of points rather than using a point-to-point search, it is possible to generate a large number of solutions to multiobjective problems in a single run of the algorithm. The two-branch tournament and its implementation in a genetic algorithm (GA) to provide these solutions are discussed. This approach differs from most traditional methods for GA-based multiobjective design; it does not require the nondominated ranking approach nor does it require additional fitness manipulations. A multiobjective mathematical benchmark problem and a 10-bar truss problem were solved to illustrate how this approach works for typical multiobjective problems. These problems also allowed comparison to published solutions. The two-branch GA was also applied to a problem combining discrete and continuous variables to illustrate an additional advantage of this approach for multiobjective design problems. Results of all three problems were compared to those of single-objective approaches providing a measure of how closely the Pareto-optimal set is estimated by the two-branch GA. Finally, conclusions were made about the benefits and potential for improvement of this approach.

Nomenclature

A = truss member cross-sectional area
c = penalty coefficient
E = Young's modulus
f = fitness function
g = constraint function
L = truss member length
n_{obj} = number of objectives
P^* = penalty scaling factor
w = vertical displacement
x = design variable
α = weighting coefficient
γ = weight density
ρ = mass density
σ = truss member axial stress
σ_y = yield stress
ϕ = objective function

Introduction

ONE of the implications of multidisciplinary design optimization is that a problem may require addressing not only a single objective with constraints based on different disciplinary analyses, but also several objectives based on separate disciplines. Finding solutions to a multiobjective problem can be difficult, as no single best design exists. This paper presents the results and provides discussion of an effort to determine a wide range of optimal solutions to a multiobjective problem using a genetic algorithm with the two-branch tournament operator.

Multiobjective Optimization

In multiobjective optimal design problems, the aim is to minimize (or maximize) a vector-valued function, whose components are each of the individual objectives. For example,

$$\text{minimize } \boldsymbol{\varphi}(\boldsymbol{x}) = \begin{Bmatrix} \phi_1(\boldsymbol{x}) \\ \phi_2(\boldsymbol{x}) \\ \vdots \\ \phi_{n_{obj}}(\boldsymbol{x}) \end{Bmatrix} \tag{1}$$

There is seldom one optimal solution to this problem; rather, a family of optimal solutions exists. This family is considered the Pareto-optimal set of solutions, after the work of the engineer/economist Vilfredo Pareto.[1] For each of these designs, no other designs may be found that are better on all objectives; in other words, these designs are nondominated. In a mathematical sense, a design with a vector of objectives $\boldsymbol{v}$ dominates a design with a vector of objectives $\boldsymbol{u}$, as follows:

$$\text{if} \quad \forall_i \quad v_i \le u_i \quad \text{and} \quad \exists_i \quad v_i < u_i \qquad i = 1, 2, \ldots, n \tag{2}$$

Many practical engineering problems are multiobjective problems, such as minimizing the weight and cost of a structure. Consider that a design that is lightweight with a high cost and a heavy design with a low cost may both be Pareto optimal. In many cases, finding the set of Pareto designs is desirable because the tradeoff among various objectives can be observed in this set.

Traditional optimization approaches search from point-to-point and result in one optimal design. One common approach is the weighted objectives approach. In this approach, the objective function of the problem is posed as

$$\hat{\phi}(\boldsymbol{x}) = \alpha_1\phi_1(\boldsymbol{x}) + \alpha_2\phi_2(\boldsymbol{x}) + \cdots + \alpha_{n_{obj}}\phi_{n_{obj}}(\boldsymbol{x}) \tag{3}$$

Presented as Paper 98-1914 at the AIAA/ASME/ASCE/AHS/ASC 39th Structures, Structural Dynamics, and Materials Conference, Long Beach, CA, April 20–23, 1998; received May 6, 1998; revision received Aug. 5, 1998; accepted for publication Oct. 27, 1998. Copyright © 1998 by the authors. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

*Assistant Professor, School of Aeronautics and Astronautics, 1282 Grissom Hall. Senior Member AIAA.

†Student, School of Aeronautics and Astronautics, 1282 Grissom Hall. Student Member AIAA.

‡Graduate Research Assistant, School of Aeronautics and Astronautics, 1282 Grissom Hall. Student Member AIAA.

§Principal Scientist, Structures Division, Building 45, 2130 8th Street, Suite 01. Fellow AIAA.

Choosing the various values of the weighting coefficients α is rather arbitrary, and for different combinations of α values, different solutions can be found. This and similar approaches can be adopted for multiobjective design, yet only one design is generated per run of the optimization algorithm. This requires several optimization runs to be conducted to find various Pareto-optimal designs.

Genetic Algorithm

As a computational representation of natural selection, the genetic algorithm (GA) has been adopted for search and optimization by making the analogy that in survival of the fittest an individual more fit to its environment is akin to a more optimal design.[2] This analogy includes representing designs as individuals in a population, performing selection (survival of the fittest) and crossover (mating) of a generation of these designs to create children, who in turn become the population for the next generation. Furthering this mimicry of natural selection, an individual design is represented by a chromosome, generally a binary string of 1s and 0s that represent the design parameter values for each individual.

The GA originated in the 1970s, and the application of various versions of the GA has rapidly expanded in the past decade. The GA has been used for global optimization, including numerous structural optimization studies ranging from truss design to composite structure design.

The GA has also been adopted for multiobjective design; see, e.g., Refs. 3–5. The common theme of most previous multiobjective GA work has been to describe the fitness function in a way so that nondominated individuals receive a better fitness value than dominated designs. Generally, this is conducted by either a weighted-sum approach, which combines all objectives into a single fitness function, or some form of ranking, which assigns better fitness values to designs based on their nondominance.

Among its many features, the GA uses a population of points to conduct its search. Because the GA is evaluating an entire population each generation, this provides the opportunity to generate a set of nondominated individuals in one run of the algorithm. The procedure described in this paper takes a different approach to the multiobjective problem by allowing each objective to be independently measured by a corresponding fitness function. In the cases described, two objectives are measured via two fitness functions, and the selection operator itself is used to perform the multiobjective design. This idea builds upon previous explorations.[6,7]

Two-Branch Tournament Selection

As noted earlier, most GA-based approaches for multiobjective design have involved assigning a single fitness value to each individual, whether via a weighted-sum approach or some form of ranking scheme. Using the selection operator to perform multiobjective design rather than using the fitness function is the significant difference of this approach. This approach also generates a representation of the Pareto-optimal set, rather than a single Pareto-optimal design, making efficient use of the GA's population-based search.

Fitness Funtions

The genetic algorithm uses a fitness function value for the selection operator; therefore this function must reflect the objective and any constraint violations. The fitness function is, therefore, constructed in the manner of a sequential unconstrained minimization technique objective with external penalty functions to handle constraints. This approach is common for single-objective optimization problems addressed via a GA.[2] Because the GA does not require derivatives, or even function continuity, several options are available for the form of the fitness function.[8] For this work, a constant coefficient with a linear external function strategy was used to impose the penalty. Combining an objective function ϕ with the constraint functions g_i in a penalty provides a fitness function suitable for use with the GA:

$$f = \phi + \sum_{i=1}^{n_{\text{con}}} c_i \max[0, g_i] \tag{4}$$

In a scaled form, the constraint functions are posed as in Eq. (5) to enforce a value greater than the allowed value, or as in Eq. (6) to enforce a value less than the allowed value. These functions are negative valued when the constraints are satisfied and are positive valued when violated:

$$g_i(\boldsymbol{x}) = 1 - \frac{\text{value}(\boldsymbol{x})}{\text{allowed}} \leq 0 \tag{5}$$

$$g_i(\boldsymbol{x}) = \frac{\text{value}(\boldsymbol{x})}{\text{allowed}} - 1 \leq 0 \tag{6}$$

The description of the fitness function in Eq. (4) still requires a somewhat arbitrary choice of the c_i coefficients. To simplify this, the same value of c is used for all constraints, requiring the choice of one value. The scaled form of the constraint functions makes this an acceptable simplification.

For multiobjective problems, difficulties may arise when the objectives have different orders of magnitude. If a design violates constraints, it is desired to apply the same order of penalty to all fitness values. To address this, a scaled penalty approach has been adopted. First, a value is chosen for the penalty coefficient c; the magnitude of this choice is made with respect to the first objective function. The total penalty is calculated along with the objective function, and the first fitness value is computed as in Eq. (3). For the remaining fitness values, a penalty scaling factor is computed from the penalty and first objective function values. This factor is then used to compute the remaining fitness values and permits all other objectives to be penalized by the same ratio:

$$P^* = f_1/\phi_1 \tag{7}$$

$$f_1 = P^*\phi_j \qquad j = 2, \ldots, n_{\text{obj}} \tag{8}$$

For example, if an infeasible design's first fitness function is 20% higher than its first objective function value, all other fitness functions will be 20% higher than their corresponding objective functions. This avoids potential problems when the magnitudes of the objective functions are quite different.

Selection Mechanics

The two-branch tournament selection approach is organized so that designs compete on one of two objectives. As the name implies, one branch of the tournament evaluates designs on the first objective; the second branch evaluates individuals on the second objective. To do this, the entire population of a current generation is placed in a pot. Two individuals are randomly selected without replacement from the pot and compete on the first fitness; the better performing individual is copied to the parent pool. This continues until the pot is empty, at which time half of the current generation has been copied to the pool. At this time, the pot is refilled once again with the entire population of the current generation, and the process is repeated with individuals competing on the second objective. Figure 1 presents a flowchart of this process.

After the second branch has been completed, the parent pool is full. The individuals in the parent pool are then randomly selected without replacement, two at a time, and mated in the crossover operation to form two children. These children eventually become the next generation of designs.

Selection Analysis

In the two-branch tournament just described, there are certain guarantees about the selection process, even though individuals are randomly chosen from the pot. In each branch of the tournament, individuals are selected without replacement, so that all individuals will have one opportunity to compete on each objective. The best performing design on the first objective will always win its competition during the first objective branch, and so it is guaranteed one copy in the parent pool. The same is true for the best individual in the second objective branch. Because all individuals compete twice, the most copies an individual can have in the parent pool is two. These features provide a slight selection pressure toward individuals better in each objective, but still maintain diversity that is needed to represent a spectrum of designs across the Pareto-optimal front.

Whereas the selection scheme described ensures that the best individual in each objective receives one copy in the parent pool,

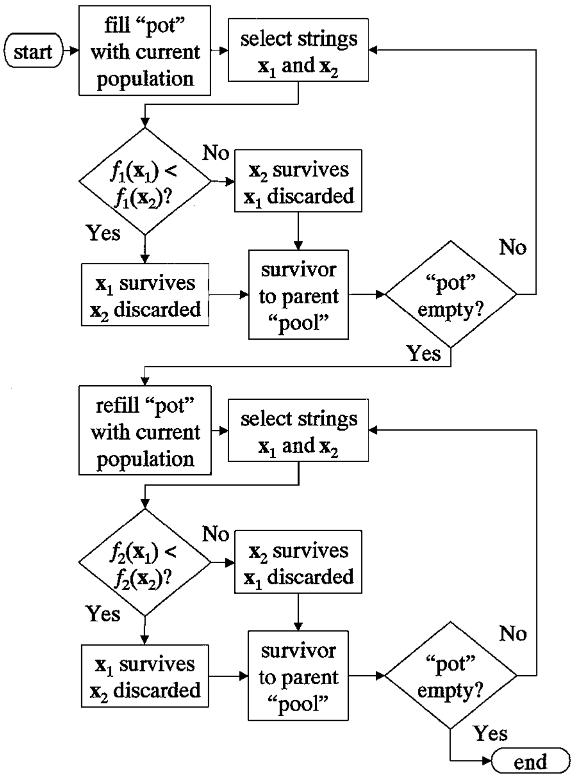

Fig. 1 Two-branch tournament flowchart.

there is no guarantee that any of the children will be better than either of the best-in-each-objective parents. In an effort to spread the generated designs along the Pareto front, an elitism strategy has been employed. After selection, crossover, and mutation have been completed, the best performing individual in each objective from the previous generation is copied into the current generation, randomly replacing two child designs.

Collection of Nondominated Designs

Many designs (thousands) are evaluated during the run of the GA, and the goal of multiobjective design is to find a range of nondominated designs that approximate the Pareto-optimal set. Therefore, any feasible, nondominated design that has been evaluated during the GA run is of interest, not just the nondominated individuals from the last generation. In this way, a large number of designs can be found that represent the Pareto-optimal set.

The collection of these individuals begins with the initial generation. The feasible nondominated individuals in the generation are stored in a list. In subsequent generations, the feasible nondominated individuals are identified. These are then compared to the stored list. If individuals currently in the list do not dominate a newly identified individual, this new individual is added to the list. Similarly, if the newly identified individual dominates those currently in the list, the dominated individuals are removed from the list. In this manner, all nondominated individuals ever encountered during the run of the GA are stored in this list.

Multiobjective Design Problems

Three multiobjective design problems were selected to demonstrate the capabilities of the two-branch GA. The first of these problems is a mathematical problem used in previous optimization benchmark studies. The second is a classical structural optimization problem, and the third is a variation of the structural optimization problem to demonstrate the GA's unique ability to combine discrete and continuous variables.

Mathematical Problem

Test problem 6 of Ref. 9 was selected to demonstrate the creation of a set of nondominated individuals over the course of a run. This problem was presented in mathematical form, although it represents the optimization of a gear reduction system. Originally, this was a single-objective design problem with 11 inequality constraints and bounds on its 7 design variables; it has been converted to a two-objective problem with 10 inequality constraints, keeping the side constraints.

To perform this conversion, one of the inequality constraints reported as active in the single-objective solution was made into an objective for minimization. The two objectives to be minimized are

$$\phi_1 = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.477\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \tag{9}$$

$$\phi_2 = \frac{\left[(745x_4/x_2x_3)^2 + 16.9 \times 10^6\right]^{0.5}}{0.1x_6^3} \tag{10}$$

In converted form, there are 10 inequality constraints for this problem. These were further converted into scaled form so that when the constraint is satisfied, the function is negative valued. They are presented in the following equations:

$$g_1 = \frac{27}{x_1x_2^2x_3} - 1 \tag{11}$$

$$g_2 = \frac{395.5}{x_1x_2^2x_3^2} - 1 \tag{12}$$

$$g_3 = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \tag{13}$$

$$g_4 = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \tag{14}$$

$$g_5 = \frac{\left[(745x_5/x_2x_3)^2 + 157.5 \times 10^6\right]^{0.5}/0.1x_7^3}{850} - 1 \tag{15}$$

$$g_6 = (x_2x_3/40) - 1 \tag{16}$$

$$g_7 = 1 - (x_1/5x_2) \tag{17}$$

$$g_8 = (x_1/12x_2) - 1 \tag{18}$$

$$g_9 = \frac{(1.5x_6 + 1.9)}{x_4} - 1 \tag{19}$$

$$g_{10} = \frac{(1.1x_7 + 1.9)}{x_5} - 1 \tag{20}$$

10-Bar Truss

A classical structural optimization problem was desired to present the efficacy of the two-branch tournament on a representative engineering design problem. For this, the 10-bar truss problem of Ref. 10 was selected. This provides an excellent example because of the history and documentation of various optimization solutions found for this problem. Traditionally, solutions to this problem were found in attempts to minimize weight subject to vertical displacement and stress constraints. Several nuances of this problem have provided significant tests for many optimization algorithms.

The 10-bar truss problem consists of a series of connected cylindrical aluminum elements subjected to two 100,000 lb (445,000 N) loads (as shown in Fig. 2). The aluminum has a Young's modulus of 10×10^6 lb/in.^{2} (68.9×10^9 Pa) and a weight density of 0.101 lb/in.^{3} (mass density 2.8×10^{-6} kg/mm^3).

To convert this problem to a multiobjective problem, the original displacement constraints were converted into a second objective, so that the truss is designed to minimize weight as one objective and to minimize vertical displacement of the four nonrestrained nodes as the second objective, subject to stress constraints. Design variables

Table 1 Material properties for the multimaterial 10-bar truss

Material	E, lb/in.^{2} (Pa)	$\gamma(\rho)$, lb/in.^{3} (kg/mm^3)	σ_y, lb/in.^{2} (Pa)
Aluminum	10.0×10^6 (68.9×10^9)	0.101 (2.79×10^{-6})	25.0×10^3 (172.3×10^6)
Nickel	28.0×10^6 (192.9×10^9)	0.284 (7.86×10^{-6})	20.0×10^3 (137.8×10^6)
Steel	30.0×10^6 (206.7×10^9)	0.318 (8.80×10^{-6})	13.3×10^3 (91.6×10^6)
Titanium	16.5×10^6 (113.7×10^9)	0.162 (4.48×10^{-6})	73.3×10^3 (505.0×10^6)

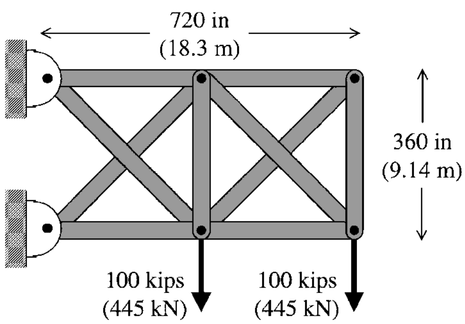

Fig. 2 Classical 10-bar truss problem.

are the cross sectional areas of the truss elements. In this form, the objectives can be expressed as

$$\phi_1 = \gamma \sum_{i=1}^{10} A_i L_i \tag{21}$$

$$\phi_2 = \max[w_1, w_2, w_3, w_4] \tag{22}$$

Constraints on the problem limit the stress in each of the 10 members to below $\sigma_y = 25 \times 10^3$ lb/in.^{2} (172×10^6 Pa). The constraint functions are posed in the following manner:

$$g_j = (|\sigma_j|/\sigma_y) - 1 \qquad j = 1, \ldots, 10 \tag{23}$$

The two fitness functions for the GA incorporate penalty functions and the penalty scaling factor as described earlier.

For this problem, analyses were performed by ANALYZE,[11] a finite-element analysis code. With very minimal modification of the code, consisting of a subroutine to interface the fitness evaluation, ANALYZE was coupled with the GA for the 10-bar truss problem. This allowed for rapid evaluation of the truss designs, providing the resulting values of weight, displacements, and stresses for use in the fitness functions.

Multimaterial 10-Bar Truss

The two preceding problems are well-posed, continuous problems. Whereas the two-branch GA allows for an estimation of the Pareto set of designs, single Pareto-optimal designs could still be found via more traditional optimization approaches. An additional feature of the GA is its ability to combine continuous and discrete variables in a single problem through the coding of the design variables into chromosomes. To illustrate this additional capability of the two-branch GA, the 10-bar truss problem was further modified to include the material of each truss element as design variables.

Four discrete material choices were allowed for each of the 10 truss elements. With the additional variables, material properties needed to be considered. Table 1 shows the properties associated with each of these material choices.

The objectives remain to minimize weight and displacement under the same loading condition as used earlier. These objectives now include the effect of material selection:

$$\phi_1 = \sum_{i=1}^{10} \gamma_i A_i L_i \tag{24}$$

$$\phi_2 = \max[w_1, w_2, w_3, w_4] \tag{25}$$

Stress constraints are also imposed on the design as before, with the exception that the allowable yield stress value varies with the material selected,

$$g_j = \frac{|\sigma_j|}{(\sigma_y)_j} - 1 \qquad j = 1, \ldots, 10 \tag{26}$$

Similarly, the objectives and constraints are converted into two fitness functions. The ANALYZE program was again used to provide weight, stress, and displacement information.

Multiobjective Design Runs and Results

The two-branch GA used for this effort utilizes uniform crossover and a Gray coding of the variables into a binary-alphabet chromosome. Population size and mutation rates were selected using the guidelines given in Ref. 12.

Mathematical Problem

The seven variables of test problem 6 (Ref. 9) were encoded to binary strings. Upper and lower bounds on each variable incorporated the original problem's side constraints. For each variable, a resolution between each value of approximately 0.01 was desired; this provided the final number of bits used for each. This coding is presented in Table 2.

Using the chromosome string length of 48 bits, the population size was selected as 180 and the mutation rate at 0.003. The GA was allowed to run for 300 generations. At the end of this run, the collected set of nondominated individuals can be plotted. Figure 3 shows the set of feasible, nondominated individuals generated as the approximate Pareto set for this run. This set contains 224 different collected individuals.

10-Bar Truss

For the 10-bar truss problem, each of the 10 design variables was encoded using 8 bits. The minimum value for each variable was 0.1 in.^{2} (64.5 mm^2), whereas the maximum was 40.0 in.^{2} (25,800 mm^2). All truss members were aluminum. With a chromosome length of 80 bits, a population size of 400 and a mutation rate of 0.003 were used. The two-branch GA ran for 300 generations. The first run consisted of an all-aluminum truss design. The collected set of 137 feasible nondominated designs is presented in Fig. 4.

Table 2 Variable bounds, bit length, and resolution for the mathematical test problem

Variable	Minimum value	Maximum value	Bits	Resolution
x_1	2.6	3.6	7	0.00787
x_2	0.7	0.8	4	0.00667
x_3	17.0	28.0	10	0.01075
x_4	7.3	8.3	7	0.00787
x_5	7.3	8.3	7	0.00787
x_6	2.9	3.9	7	0.00787
x_7	5.0	5.5	6	0.00794

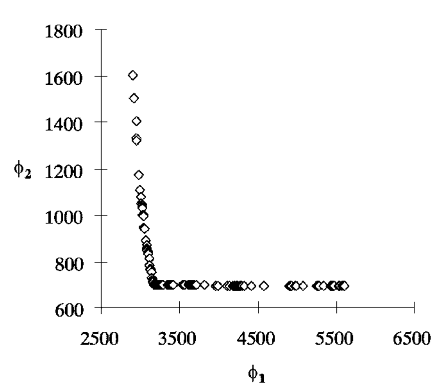

Fig. 3 Nondominated individuals generated for test problem 6 (Ref. 9).

Table 3 Gray coding and material for multimaterial GA

Binary Gray code	Material
00	Steel
01	Aluminum
11	Nickel
10	Titanium

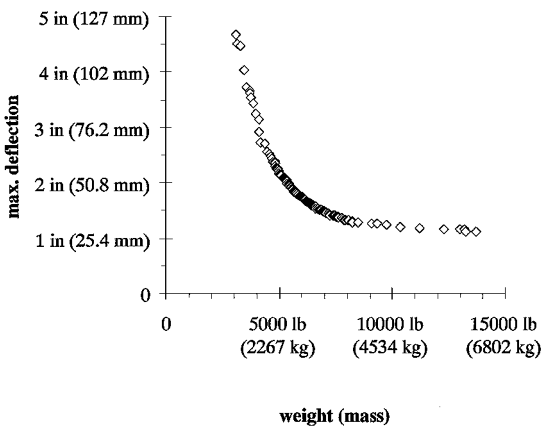

Fig. 4 Nondominated individuals for the 10-bar truss problem.

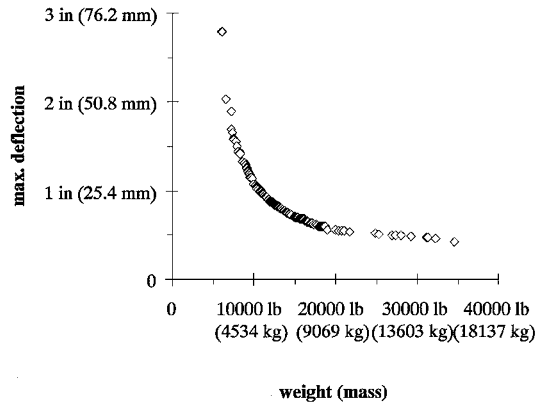

Fig. 5 Nondominated individuals for multimaterial 10-bar truss problem.

Multimaterial 10-Bar Truss

In this problem, 10 variables were added to represent the materials of the truss members. These variables were added as 2-bit strings, shown in Table 3. With a total of 20 variables, the string length was 100 bits. A population size of 400 and a mutation rate of 0.003 were used. Here, the GA ran for 300 generations. The feasible nondominated individuals are shown in Fig. 5. This set contains 140 designs.

Discussion

To see how well the two-branch GA performed, it was compared to results from other optimization methods. First, the Pareto-front approximation generated by the two-branch GA was compared to published results (where available), to single-objective GA results, and to results from a traditional optimization program. This provided a measure of how well the GA approximated the optimum designs in the Pareto set. In the case of the multimaterial 10-bar truss, several single-material Pareto fronts from the two-branch GA were compared to that of the multimaterial problem. This illustrates the unique designs that the GA can obtain.

Comparison to Single-Objective Results

Both the test problem 6 (Ref. 9) and the aluminum 10-bar truss have optimal results published for their respective single-objective versions. These were compared to the approximate Pareto front.

To further compare the multiobjective GA's results, a single-objective GA was run several times to obtain optimal designs along the Pareto front. To do this, one objective was kept as the single objective, and the second objective was treated as a constraint. By changing the value used in this constraint on the second objective, different designs along the Pareto front were found. For example, the 10-bar truss was minimized for weight with a constraint on maximum displacement. By tightening the constraint on displacement, another Pareto-optimal point was found.

Finally, a traditional optimization program, CONMIN,[13] was used to obtain single-objective optimum points to compare to the two-branch GA's Pareto front. CONMIN uses the method of feasible directions to optimize a design, given an initial design. The initial designs for these comparison runs were the points obtained by the single-objective GA already mentioned. Again, the one objective was kept as the single objective, whereas the second was converted to a constraint. Using CONMIN provided additional information about the final designs, including satisfaction of Kuhn–Tucker conditions, something that a GA cannot provide.

Results for the mathematical problem are compared in Fig. 6. The individuals representing the Pareto front as estimated by the two-branch tournament are shown as a connected line rather than individual points. The published result is taken from Ref. 9. Here, the two-branch GA results show excellent correlation to the other approaches.

Results from the all-aluminum 10-bar truss problem are similarly compared in Fig. 7. Here, the published result from Ref. 10 falls directly on the two-branch GA, single-objective GA, and CONMIN results. In this plot, correlation among the various approaches is good across most of the Pareto front; however, the single-objective approaches were able to find some designs lighter in weight than the two-branch GA.

The points from the single-objective GA and from CONMIN reemphasize that the two-branch approach provides an estimate of the Pareto front rather than the exact representation. The two-branch GA representation of the front still does not seem to stretch across

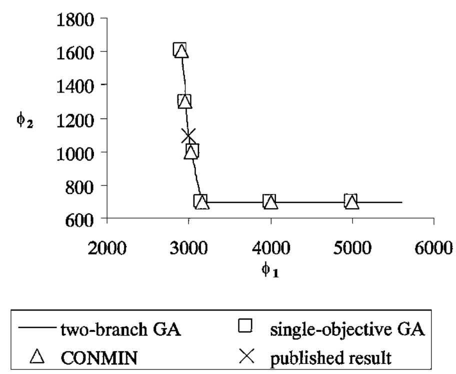

Fig. 6 Two-branch GA results for test problem 6 (Ref. 9) compared to other approaches.

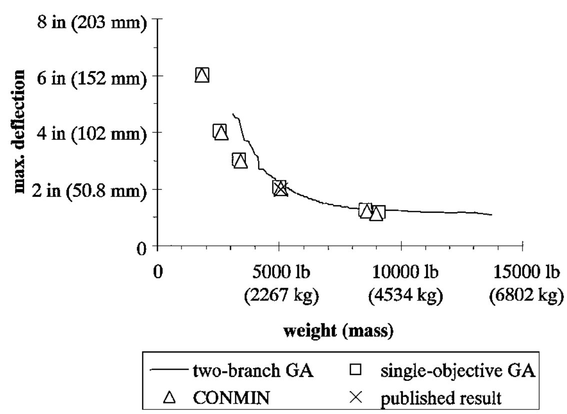

Fig. 7 Two-branch GA results for the all-aluminum 10-bar truss compared to other approaches.

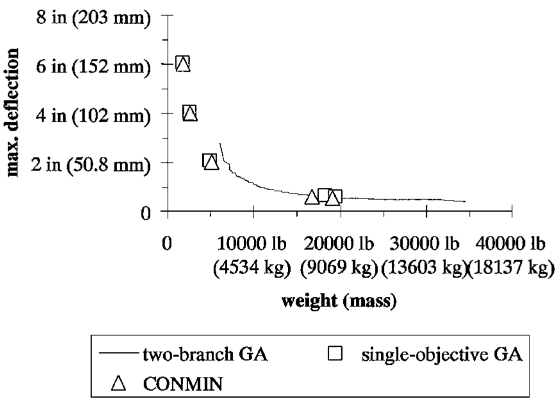

Fig. 8 Two-branch GA results for the multimaterial 10-bar truss compared to other approaches.

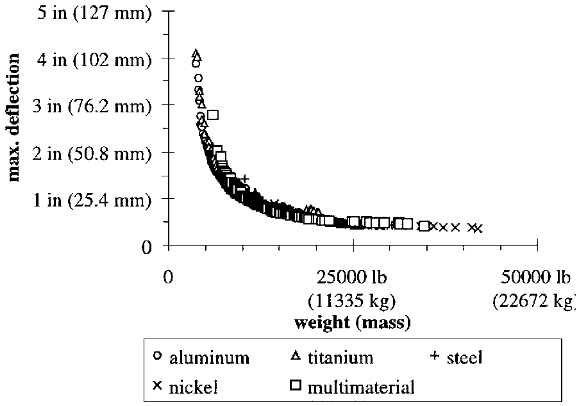

Fig. 9 Two-branch GA set of nondominated designs for multimaterial, all-aluminum, all-steel, all-nickel, and all-titanium 10-bar truss problems.

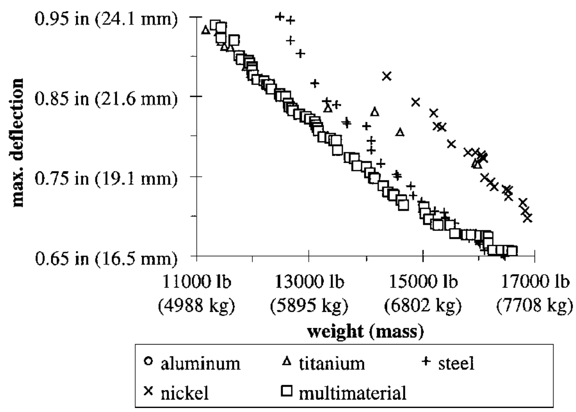

Fig. 10 GA-generated set of unique nondominated designs for multimaterial 10-bar truss.

the entire front. Using the elitism strategy helps, but does not fully capture the entire front.

A contributing factor here is the stopping criterion for the GA. In single-objective GAs, it is possible to judge that a population has converged when the best individual does not improve over several generations. This approach has been very useful when only one optimal design exists; with the multiobjective problem, a set of Pareto-optimal designs exists, and so this criterion is impossible to enforce. For all results in this paper, the GA was stopped after reaching a maximum number of generations. Potentially, longer run times may help stretch the estimated Pareto front. Techniques such as fitness sharing and niching have been employed in multiobjective GAs that use fitness ranking and stochastic selection operators.[5,14] Appropriately adapted for the tournament selection, these techniques may stretch out the nondominated designs and give a better approximation of the entire Pareto front.

Figure 8 displays the comparison of approaches for the multimaterial 10-bar truss problem. No published optimal designs were found, so that comparison is made only to the single-objective GA and CONMIN. Further, results were obtained using material selection as discrete variables in the single-objective GA, then the materials were fixed, and only the cross-sectional areas were used as design variables to obtain results from CONMIN.

The two-branch GA results still show reasonable correlation with some of the single-objective results, but the single-objective approaches were able to find some designs that the two-branch GA was unable to find. This problem, having a combination of discrete and continuous variables, also has an increase in complexity. In spite of this increased complexity, the two-branch GA was still able to locate a good portion of the Pareto front. Also, the two-branch GA was able to locate designs that could not be found via traditional approaches. This is discussed in the following section.

Multimaterial vs Single-Material Results

The GA's coding of design variables allowed the discrete material selection variables to be directly incorporated in the problem statement and solution approach. With this approach, the two-branch GA was able to find several unique designs that could not be found assuming all truss members were constructed of the same material.

To illustrate this, the two-branch, single-objective GA was run separately for an all-aluminum, an all-titanium, an all-steel, and an all-nickel 10-bar truss. The four resulting Pareto-front approximations were then superimposed with the multimaterial results as shown in Fig. 9.

Figure 9 shows several interesting consequences of using the GA to include material selection as a design variable. First, the approximate Pareto front for the multimaterial problem does not stretch as far to the extreme points of the design space as the single-material results. The all-aluminum results lie further to the left side of the front, and the all-nickel results lie further to the right. As the preceding section described, the two branch has some shortcomings covering the entire spread of the Pareto front.

Second, and perhaps more interestingly, the Pareto front of the multimaterial problem includes an area in which the multimaterial designs dominate any of the single-material designs. This unique area of the objective space is expanded in Fig. 10. The multimaterial designs dominate the single-material designs from about 12,000 lb (5450 kg mass) to about 15,700 lb (7129 kg mass).

Whereas the number of multimaterial designs that are truly Pareto optimal is relatively small for this problem, this illustrates an advantage of the two-branch GA. Using the material choice as a discrete variable allowed these designs to be discovered in one run of the GA. Because each of the 10 material variables had four discrete choices, there were 4^{10} (1,048,576) different possible combinations of materials. Without a priori knowledge of the design space, this suggests that discovering these superior multimaterial designs may have required over one million different runs if material selection were not used as variables.

Conclusions

As a result of this effort, several conclusions can be made about the two-branch tournament GA for multiobjective design.

First, it should be recognized that a GA might not be the most efficient approach to optimizing test problem 6 (Ref. 9) or the all-aluminum 10-bar truss as single-objective problems. These problems have continuous variables and could readily be solved for considerably less computational effort via a traditional calculus-based approach.

However, when using the two-branch tournament GA to generate an approximation to the Pareto front, the computational effort becomes comparable to that for traditional approaches. For example, 224 different designs were generated in one run of the two-branch GA for test problem 6 (Ref. 9). With a population size of 180 run over 300 generations, the two-branch GA required 54,000 function evaluations. If an approach was used that can only produce a single solution at a time, 224 runs of this single-solution method would be required. Assuming that numerical derivatives were used, a calculus-based approach may require approximately 50–80 function evaluations to find a solution, including evaluations needed for a line-search. This corresponds to between 11,200 and 17,920 total evaluations to find 224 different Pareto-optimal designs, assuming that local minima are not encountered in the design space.

Using discrete variables in the two-branch GA approach to solving the multimaterial 10-bar truss illustrates a condition where the GA-based approach becomes less computationally intensive. As mentioned in the discussion section, to find the multimaterial designs that dominate the single-material designs would require optimization runs over the complete enumeration of all discrete material combinations. Even to find single solutions to all of the possible combinations of materials would require over one million runs, making the multiobjective problem nearly intractable via a traditional approach. The two-branch GA was able to find these designs using a population size of 400 after 300 generations, so that only 120,000 function evaluations were needed.

The Pareto fronts approximated by the two-branch GA were fairly good in comparison to published results, single-objective GA results and CONMIN optimization results. This was especially true near the middle of the Pareto front; fortunately, for most multiobjective engineering problems this compromise region is generally of the most interest. The deficiency in the two-branch GA to find the extreme points in the design space may be overcome using techniques such as fitness sharing and niching if appropriately adapted for the tournament selection. These techniques may help to stretch out the nondominated designs and give a better approximation of the entire Pareto front.

Finally, this work has illustrated that the two-branch tournament selection operator in a GA can provide a practical and successful means for multiobjective design. The results presented demonstrate that it is possible to generate a good representation of the Pareto-optimal set in just one run of the two-branch GA. In this manner, potentially complex multiobjective optimization problems could be quickly reduced to simple tradeoff curves from which suitable designs could be chosen.

Acknowledgments

A. M. Cook was supported for her contribution to this work through a summer employment experience at U.S. Air Force Wright Laboratory. D. W. Fanjoy was supported for his contribution through the U.S. Air Force Office of Scientific Research Summer Graduate Research Program.

References

[1]Pareto, V., "General Notion of Economic Equilibrium," *Manuale di Economia Politica*, Societa Editrice Libraria, Milan, Italy, 1906 (translated to English by A. S. Schwier, *Manual of Political Economy*, Macmillan, New York, 1971), pp. 103–180.

[2]Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison–Wesley, Reading, MA, 1989, pp. 1–57.

[3]Schaffer, J. D., "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, edited by J. J. Grefenstette, Lawrence Erlbaum Publishers, Northvale, NJ, 1985, pp. 93–100.

[4]Horn, J., and Nafpliotis, N., "Multiobjective Optimization Using the Niched Pareto Genetic Algorithm," Illinois Genetic Algorithm Lab., Rept. 93005, Urbana, IL, July 1993.

[5]Fonseca, C. M., and Fleming, P. J., "An Overview of Evolutionary Algorithms in Multiobjective Optimization," *Evolutionary Computation*, Vol. 3, No. 1, 1995, pp. 1–16.

[6]Crossley, W. A., "Genetic Algorithm Approaches for Multiobjective Design of Rotor Systems," *Proceedings of the AIAA/NASA/ISSMO 6th Symposium on Multidisciplinary Analysis and Optimization* (Bellevue, WA), AIAA, Reston, VA, 1996, pp. 384–394.

[7]Norris, S., and Crossley, W. A., "Selecting Pareto-Optimal Gains for a Simple Control System Using a Genetic Algorithm," AIAA Paper 98-1010, Jan. 1998.

[8]Crossley, W. A., and Williams, E. A., "A Study of Adaptive Penalty Functions for Constrained Genetic Algorithm Based Optimization," AIAA Paper 97-0083, Jan. 1997.

[9]Venkayya, V. B., "Benchmark Studies Using Various Optimization Algorithms," NATO/Deutsche Forschungsgemeinschaft (DFG) Advanced Study Inst., Univ. of Essen, Berchtesgaden, Germany, Sept.–Oct. 1991.

[10]Venkayya, V. B., "Design of Optimum Structures," *Computers and Structures*, Vol. 1, Nos. 1/2, 1971, pp. 265–309.

[11]Venkayya, V. B., and Tischler, V. A., *"ANALYZE"—Analysis of Aerospace Structures with Membrane Elements*, U.S. Air Force Flight Dynamics Lab., TR AFFDL-TR-78-170, Wright–Patterson AFB, OH, Dec. 1978.

[12]Williams, E. A., and Crossley, W. A., "Empirically-Derived Population Size and Mutation Rate Guidelines for a Algorithm with Uniform Crossover," *Soft Computing in Engineering Design and Manufacturing*, edited by P. K. Chawdhry, R. Roy, and R. K. Pant, Springer–Verlag, Berlin, 1998, pp. 163–172.

[13]Vanderplaats, G. N., "CONMIN—A FORTRAN Program for Constrained Function Minimization: User's Manual," NASA TM-X-62,282, Aug. 1973, reprinted 1991.

[14]Cheng, F. Y., and Li, D., "Multiobjective Optimization Design with Pareto Genetic Algorithm," *Journal of Structural Engineering*, Vol. 123, No. 9, 1997, pp. 1252–1261.

A. D. Belegundu
Associate Editor