ELSEVIER

Comput. Methods Appl. Mech. Engrg. 186 (2000) 339–356

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Continuum structural topology design with genetic algorithms

Mark J. Jakiela [*], Colin Chapman, James Duda, Adenike Adewuya, Kazuhiro Saitou

Department of Mechanical Engineering, Washington University in St. Louis, Campus Box 1185, Jolley 305, One Brookings Drive, St. Louis, MO 63130-4899, USA

Received 30 March 1999

Abstract

The genetic algorithm (GA), an optimization technique based on the theory of natural selection, is applied to structural topology design problems. After reviewing the GA and previous research in structural topology optimization, we describe a binary material/void design representation that is encoded in GA chromosome data structures. This representation is intended to approximate a material continuum as opposed to discrete truss structures. Four examples, showing the broad utility of the approach and representation, are then presented. A fifth example suggests an alternate representation that allows continuously-variable material density. Concluding discussion suggests recommended uses of the technique and describes ongoing and possible future work. © 2000 Elsevier Science S.A. All rights reserved.

Keywords: Structural optimization; Topology; Genetic algorithm

1. Introduction

This article reviews and presents summary examples of the genetic algorithm (GA)-based approach to structural optimization developed by Jakiela and associated researchers [9–12,15]. Further details and example problems can be found in these references (from which much of this article was excerpted) and the related theses of Chapman [8] and Duda [14]. Using an evolutionary, survival-of-the-fittest optimization mechanism [16,28], the GA allows designs in a population to compete against one another to serve as parent designs. Parents then pair and mate, swapping portions of their 'genetic code' to create a generation of child designs of hopefully higher quality. After undergoing infrequent, random mutation, the child generation replaces the original generation, and the process then iterates until an optimal design is generated.

We emphasize that we perform a topological optimization. This is in contrast to sizing and shape optimization. Fig. 1 shows examples of each for the design of a beam cross-section. In sizing optimization, as shown in Fig. 1(a), the topology and a parameterized shape are held constant, while an optimal set of parameters is found. These are commonly rectilinear dimensions. Shape optimization maintains a constant topology but changes the shape, as shown in Fig. 1(b). Fig. 1(c) indicates that topology optimization is the next step; new boundaries may be created. In our approach, shape and sizing optimization are byproducts of topology optimization. Additionally, the approach described here should be contrasted with the (possibly topological) optimization of a truss structure, in which the dimensions of truss members are altered, or the members are deleted from the structure altogether. The binary material/void elements used here are not

[*] Corresponding author.

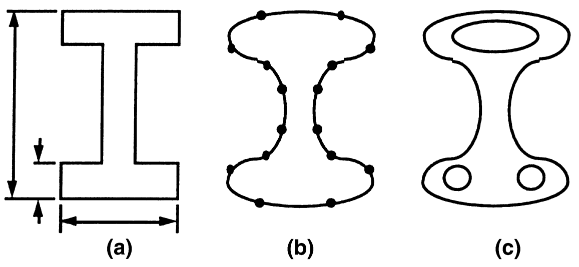

Fig. 1. Sizing, shape, and topology optimization [12].

intended to represent full size structural elements such as truss bars; the intent is rather that the topology and shape emerge from an aggregation of a large number of the elements.

The next two sections provide an introduction to GAs and a review of related work. This is followed by an explanation of how structural topologies are represented with GA chromosomes. Next, four examples, intended to show the broad utility of the approach, are presented. A fifth example suggests how real number chromosome data structures could represent material with continuously-variable density. The paper concludes with recommendations for use of GAs for structural optimization and directions for future work.

2. Genetic algorithms

GAs are an optimization strategy in which points in the design space are analogous to organisms involved in a process of natural selection. The term 'genetic' is used because, along with the expected design representation, GAs employ a coded representation of design attributes that is analogous to a chromosome [28]. This code is commonly a character string, with each character position being analogous to a gene, and each character assigned to a position being analogous to an allele. Organisms are generated and tested in generations, with offspring designs arising from parent designs. The creation of new designs for a new generation occurs with a process that is analogous to biological reproduction. Genetic crossover allows offspring designs to retain traits from parent designs, and infrequent mutations possibly yield radically improved designs, but almost always yield unsuitable configurations. The testing of new designs is done with a merit function, usually tailored to take the coded representation as input. In a given generation, designs with a higher merit are given a higher probability of creating offspring, and perhaps surviving themselves into the next generation.

Optimization occurs, therefore, through a process of natural selection. Designs in a given generation group in pairs (i.e., mate), with the better designs having a higher probability of pairing. These 'parent' designs produce offspring by genetic crossover. In 'single point' crossover, a point along the coded representations (the chromosomes) is chosen at random, and the segments of the code after the point are swapped (see Fig. 2). Infrequent, random mutations are then performed on individual alleles within the

• Parent Chromosomes:

```
11111111111111111111
00000000000000000000
```

•Perform Crossover:

```
1111111111111|1111111
0000000000000|0000000
```

• Resulting Child Chromosomes:

```
11111111111110000000
00000000000001111111
```

Fig. 2. Crossover of two genetic algorithm chromosomes [12].

chromosomes by changing the values. These operations yield two new codes which represent two new designs that possess traits from both parents. In this way a new generation is created. The process then iterates. After many generations, both the best design and the average quality should increase, because the merit function is more likely to allow better designs to produce offspring.

When applying GAs to engineering design domains, several points are worthy of note. In particular, many issues related to computing the merit function affect the performance of the overall search. Our examples will show cases in which design constraints are represented as penalty terms in the merit function. If these are not carefully normalized with the actual 'quality' terms of the function, the overall search can be erratic. Other problems arise if there is a large range of design qualities for a particular generation – the highest-performance designs will quickly dominate subsequent generations, filling the populations with a particular 'breed' of a possibly suboptimal design. A local sub-optima can be avoided if large differences among merit function values are attenuated during reproduction. This is often necessary in early generations.

Properly-sized penalty terms and suitable levels of fitness value attenuation are two of the many parameters which must be specified when conducting a GA-based search. In our GA-based optimization implementation, the following parameters must be specified.

Probability of crossover ($P_{\mathrm{CROSSOVER}}$): The probability that crossover will be performed between a pair of parent chromosomes. If crossover is not performed, the parents (unmodified) are treated as potential children for the next generation.

Probability of mutation (P_{MUTATION}): The probability that any given allele on any given chromosome will mutate.

Fitness scaling coefficient (C_{MULT}): A measure of the magnitude of fitness value attenuation. This should be selected so that a proper number of the fittest members are chosen to continue into the subsequent generation, without allowing them to dominate the generation. This typically represents the desired ratio of the maximum fitness in a population to the average fitness of the population.

Population size: Number of chromosomes in each GA generation. This must be chosen with care – too large a population is inefficient, while too small a population does not provide the amount of genetic material needed to explore much of the search space.

Crossover operator: The method used to mate, or combine, two parent chromosomes to create two child chromosomes which have attributes from both parents.

Selection scheme: The technique used to determine which chromosomes in a population will serve as parents for the next generation.

3. Related work

3.1. Introduction

Sizing, shape, and topology optimization based upon structural considerations have been active research areas for some time [27]. Our topological optimizations generate optimal distributions of material and void within a discretized design domain, as opposed to optimal discrete truss structures. Several approaches to this problem have been investigated and are reviewed here.

3.2. Homogenization-based

Bendsoe and Kikuchi [6] developed an approach based on material homogenization techniques [43]. A design domain is discretized into small, rectangular elements, where each element contains composite material of continuously-variable density and orientation. Each element's structural properties are a function of its material density and orientation, and are calculated using material homogenization techniques. An optimality criteria method [31] is used to determine how the material density and orientation in each element should change so that the structure's compliance is minimized, subject to a maximum volume constraint.

Several different microstructure models are used when determining the composite's structural properties. Bendsoe and Kikuchi [6] introduced a microstructure comprised of microscale rectangular holes, while Bendsoe et al. [5] studied a 'Rank-2' microstructure.

3.3. Simulated annealing

Anagnostou et al. [3] developed a simulated-annealing-based [30] approach. A design domain is discretized into small elements, where each element contains material or void. No intermediate densities are allowed. Simulated annealing is used to determine the optimal configuration of material and void within the domain such that the structure's weight is minimized subject to stress and manufacturability constraints.

3.4. Genetic algorithm

Sandgren et al. [39], Sandgren and Jensen [38] and Jensen [29] developed a GA-based approach. The design domain is discretized into small elements, where each element contains material or void. No intermediate densities are allowed. The GA determines the optimal configuration of material and void within the domain such that the structure's weight is minimized subject to displacement and possibly stress constraints. With regard to the optimization of truss structures, Goldberg and Samtani [18], Hajela [22,23], Rajeev and Krishnamoorthy [33], and Lin and Hajela [32] investigated cross-section sizing optimizations of discrete-member trusses. Jenkins [24,25], Richards and Sheppard [34] and Watabe and Okino [42] studied the shape optimization of structural members. GA-based topology optimization of *discrete* truss structures was investigated by Shankar and Hajela [41], Hajela et al. [21], and Grierson and Pak [20].

Our investigations extend the work of Jensen and associated researchers, specifically by addressing the following: (i) cantilevered plate topologies of high discretization, (ii) techniques for obtaining finely discretized topologies, (iii) techniques for obtaining families of highly fit designs, and (iv) a variety of different structural design fitness functions.

4. Our approach

4.1. Introduction

During our GA-based optimizations, the 'fitness' of a chromosome is determined by first converting the chromosome into a topology. Then, the topology's structural performance is evaluated. This is used to assign a fitness value to the chromosome which quantitatively describes the chromosome's objective function minimization (or maximization) and constraint satisfaction abilities.

4.2. Design representation

A two-dimensional design domain is discretized into small, square elements, where each element represents either material or void. The states of the individual elements define the distribution of material and void within the domain and therefore establish the topology. This binary, material-void design domain results in a discrete, typically non-convex search space [3] and allows for a precise, although discretized, topology boundary.

4.3. Converting chromosomes into topologies

Each chromosome contains one gene for every design domain element, and each gene controls the material-void state of a particular element. To convert a chromosome into a topology, the chromosome is mapped into the design domain, and elements controlled by genes with allele values of 1 become material while elements controlled by genes with allele values of 0 become void. Depending on the example problem, this mapping is done in one of two ways. Fig. 3 shows how a binary string chromosome is mapped to a

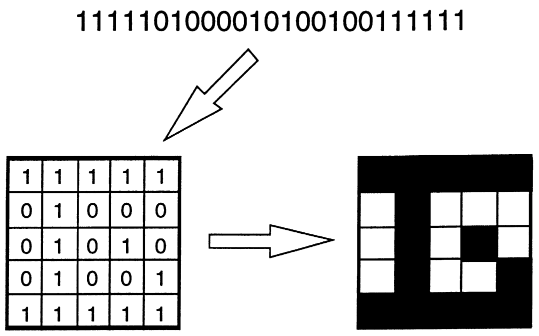

Fig. 3. Converting a binary string to a two-dimensional topology [9].

planar design domain. Alternately, a two-dimensional binary array chromosome can be mapped directly to a planar design domain.

4.4. Connectivity analysis

'Connectivity analysis' then sets to void all material elements in the topology which are not *connected* (whether directly or indirectly via other elements – see Fig. 4) to a specified *seed* element. Any two elements are considered *connected* if they share an edge; elements sharing only a corner are considered *disconnected* (Fig. 4). *Seed* elements are those elements required to contain material so that they may serve as a support boundary condition or point of load application. Connectivity analysis does not modify corresponding chromosome allele values (i.e. the genotype) when a disconnected material element is switched to void or a seed element is switched to material – only the state of the design domain element (i.e. the phenotype) is changed.

Connectivity analysis is performed principally because planar material elements connected only at a corner cannot withstand applied torques about the corner, and can therefore lead to a topology which cannot support various loads. Connectivity analysis is not equivalent to, nor do we perform, classical stability or buckling analysis.

4.5. Structural analysis

The topology's structural performance is then determined using finite element analysis. Using a finite element mesh containing four triangular finite elements for each design domain element (yielding a finite element node at each corner and in the center of every design domain element), finite elements corresponding to void (including those corresponding to disconnected material elements switched to void by connectivity analysis) are assigned a Young's Modulus 10^{-5} times that of finite elements corresponding to material. Note that this allows a constant mesh to be used, resulting in a considerable computational savings. Example 3 in [12] demonstrates the validity of using a constant mesh along with connectivity analysis.

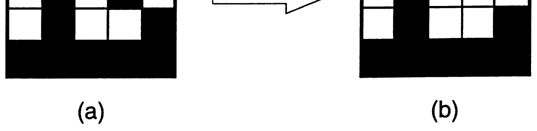

Fig. 4. Topology (a) before and (b) after connectivity analysis [9].

4.6. Fitness calculations

A fitness value must then be assigned to the chromosome corresponding to the topology. The fitness will often take into account the stiffness and volume of a structure. We take a structure's stiffness as the inverse of the deflection ($\delta_{\max}$) of a particular point of interest.

$$\text{Stiffness} \approx \frac{1}{|\delta_{\max}|}. \tag{1}$$

For planar structures (our interest) a topology's volume is assumed to be equal to the area of connected material in the topology.

4.7. GA parameters and runtime considerations

Unless otherwise specified, the GA routines utilized random initial populations, binary-coded chromosomes, single-point crossover, mutation, fitness scaling, and an elitist stochastic universal sampling selection strategy [4]. The probabilities of crossover ($P_{\text{CROSSOVER}} = 0.95$) and mutation ($P_{\text{MUTATION}} = 0.01$), and the population size (Number of chromosomes = 30) in each example were chosen according to values suggested by Grefenstette [19] and Schaffer et al. [40]. The fitness scaling coefficient ($C_{\text{MULT}} = 1.4$) used was suggested by Goldberg [16].

With the exception of where a statistical study is done (i.e. in Example 3), the results shown are from 'characteristic' runs, which are admittedly among the best from a fairly limited number.

5. Examples

5.1. Example 1 [11]: plane stress FEM model

This example describes the optimization of a cantilevered plate subject to a vertical load, applied at the FEM node on the right hand surface 2/5 of the distance from the bottom. The design domain is shown in Fig. 5. Three discretizations are used – 10 × 16, 15 × 24, and 20 × 32 grids of elements. Material 'seeds' were placed at the point of load application and points of support.

After a chromosome is mapped into the design domain and the material distribution is analyzed for connectedness, the resulting topology's fitness is determined. A topology's fitness is equal to its stiffness-to-weight ratio, where 'stiffness' (S) is defined by Eq. (1).

To determine $\delta_{\max}$, a finite element analysis is performed on the topology. $\delta_{\max}$ was set equal to the magnitude of the displacement (vector sum of x- and y-direction displacements) of the node where the point load was applied. The area of connected material is used as a qualitative measure of the topology's 'weight'. Hence, the topology's fitness is given by

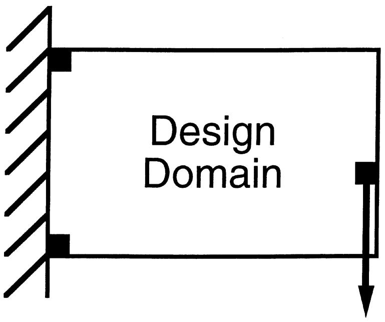

Fig. 5. Example 1 design domain [11].

$$\text{Fitness} = \frac{(1/\delta_{\max})}{\text{Area}}. \tag{2}$$

The 10 × 16 grid's optimization was run for 225 generations, while the 15 × 24 and 20 × 32 grid optimizations required 600 generations (all with population size 30). Fig. 6 shows the results. The topologies have well-defined, solid-material outer boundaries, while the interior regions generally have a 'composite-like' internal structure comprised of equally-distributed material and void. The 20 × 32 topological optimization 'hollowed out' several large interior holes, producing truss-like members. Chapman et al. [11] also described a hierarchical domain decomposition technique that allows an even finer 40 × 64 discretization.

5.2. Example 2 [9]: compliance minimization

We use the GA to minimize a cantilevered plate's compliance subject to a maximum volume constraint of 25% (Fig. 7). During finite element analyses, all nodes along the mesh's left-hand surface are constrained to have zero displacement, and a downward concentrated load is applied at the middle node along the mesh's right-hand surface. The two design domain elements surrounding the point of load application and the elements at the top and bottom of the domain's left-hand surface serve as seed elements during connectivity analysis. No symmetry constraints are imposed.

Fitness calculations begin when a topology's compliance is calculated, which is equal to the inner product of the topology's displacement ($\delta_{\max}$) at the point of load application and the applied load (F_{APPLIED}):

$$\text{Compliance} = \delta_{\max} \cdot F_{\text{APPLIED}}. \tag{3}$$

Compliance values decrease by several orders of magnitude during optimization, creating difficulty in selecting volume constraint violation penalty coefficients which work well for both early and later populations. Hence, the natural log of compliance is used and the topology's fitness (to be maximized) is

$$\text{Fitness} \approx \frac{1}{\ln(\text{Compliance})}. \tag{4}$$

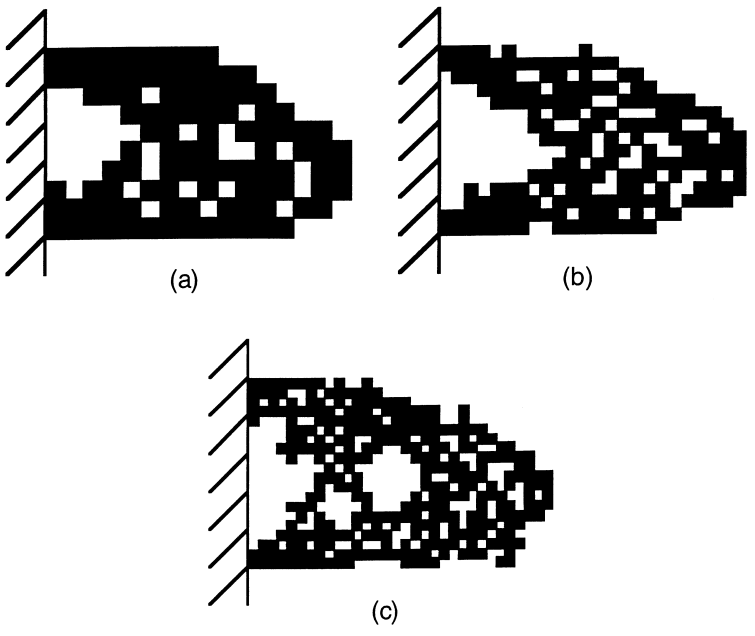

Fig. 6. Results of (a) 10 × 16, (b) 15 × 24, and (c) 20 × 32 optimizations [11].

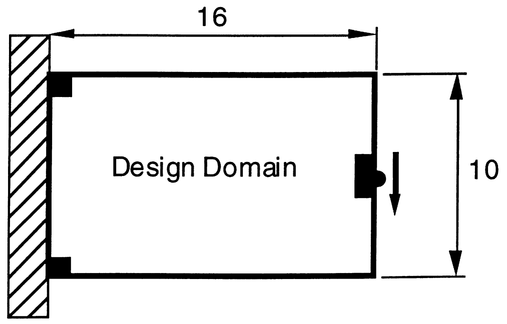

Fig. 7. Example 2 design domain [9].

The topology's volume (V) is then compared to the maximum volume constraint ($V_{\max}$). If there is no violation, the topology is assigned the fitness value of Eq. (4). If there is a violation, the topology's fitness is penalized 6% for every 10% volume constraint violation. The penalty is linearly attenuated according to the current generation number, with no penalty at generation 0 and full (6%) penalty at generation 175. Hence, the fitness of a structure violating the maximum volume constraint is

$$\text{Fitness} = \left\{1 - \frac{\text{generation}}{175} \cdot 0.6 \cdot \frac{V - V_{\max}}{V_{\max}}\right\} \frac{1}{\ln(\text{Compliance})} \tag{5}$$

prior to generation 175 and

$$\text{Fitness} = \left\{1 - 0.6 \cdot \frac{V - V_{\max}}{V_{\max}}\right\} \frac{1}{\ln(\text{Compliance})} \tag{6}$$

after generation 175. Search was performed using hierarchical subdivision. Please see [11] for the details.

The GA-based solution is shown in Fig. 8, while Fig. 9 depicts a homogenization-based solution (using a rectangular hole microstructure) to the same problem [36]. Care was taken that the homogenization-based solution used the same material sizes and properties along with the same loading conditions; a difference was that the finite element solver employed quadrilateral elements. Our GA-based solution contains 3% less material and exhibits 12% greater compliance than Rodrigues's homogenization-based solution.

In general, such a GA-based solution may require 10–100 times the number of function evaluations as would be required by a homogenization-based solution. In many cases, this is a prohibitive number, and techniques to parallelize the fitness function evaluations or the use of approximations must be considered. Primary advantages of a GA-based solution, on the other hand, are that it performs a global search (as demonstrated in Example 4 below) and readily allows a variety of fitness (objective) functions and constraints (as demonstrated in Example 3 below).

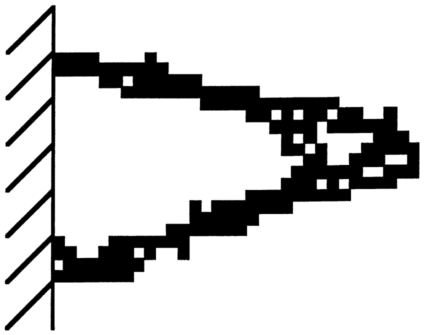

Fig. 8. Genetic algorithm-based solution for Example 2 [9].

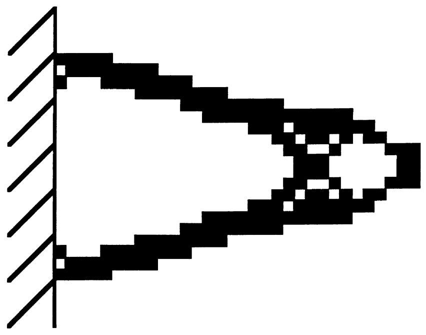
Fig. 9. Homogenization-based solution [36] in [9].

5.3. Example 3 [9]: topology simplification considerations

We recognize that designs such as those generated in Examples 1 and 2 will require interpretation and simplification for subsequent detail and sizing design. In addition to smoothing the edges created by the square design domain elements, the number of holes should be reduced to simplify the structure. There are similar concerns with homogenization-based designs, arising from the continuously-variable density (see [7,26,35]).

One approach to this problem is to agglomerate the holes by minimizing their number. This is done by taking the number of holes, the total hole area, total hole perimeter, and total structure perimeter (holes plus outer boundary) into account in the fitness functions used. Structures with fewer larger holes with only slightly diminished performance are preferred for interpretation and subsequent detail design. A case study of this process was performed with the design problem of Example 1 (using the 10×16 discretization).

To drive the GA search towards topologies exhibiting both high structural performance and varying numbers of internal holes (i.e., varying amounts of material and void element agglomeration), nine fitness functions were developed and used to optimize the plate's topology. Eight of the functions, in addition to maximizing stiffness-to-volume ratio, use a variety of techniques to reduce the number of internal holes. The ninth function (Function 1) is equivalent to the standard stiffness-to-weight ratio maximization function used previously.

Fitness Functions

- *Function* 1: Maximize stiffness-to-volume ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area}}.$$

- *Function* 2: Simultaneously maximize stiffness-to-volume ratio and volume-to-perimeter ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area}} + c\frac{\text{Area}}{\text{Perimeter}}.$$
$$c = 0.1, 0.2, 0.4, 0.6, 0.8, 1.0, 2.0.$$

- *Function* 3: Maximize stiffness-to-volume-to-perimeter ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area} \cdot \text{Perimeter}}.$$

- *Function* 4: Maximize stiffness-to-volume-to-HoleArea ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area} \cdot \text{HoleArea}}.$$

- *Function* 5: Maximize stiffness-to-volume-to-HolePerimeter ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area} \cdot \text{HolePerimeter}}.$$

- *Function* 6: Maximize stiffness-to-volume-to-NumberOfHoles ratio.

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area} \cdot (\text{NumberOfHoles} + 1)}.$$

- *Function* 7: Simultaneously maximize stiffness-to-volume ratio and HoleArea-to-HolePerimeter ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area}} + c\frac{\text{Area}}{\text{HolePerimeter}}.$$
$$c = 0.2, 0.4, 0.6, 0.8, 1.0, 2.0.$$

- *Function* 8: Simultaneously maximize stiffness-to-volume ratio and minimize HolePerimeter-to-HoleArea ratio

$$\text{Fitness} = \frac{1}{\text{Displacement} \cdot \text{Area}} - c\frac{\text{HolePerimeter}}{\text{HoleArea}}.$$
$$c = 0.05, 0.10, 0.15, 0.20, 0.25, 0.30.$$

- *Function* 9: Maximize stiffness-to-volume ratio, subject to a 'maximum number of internal holes' constraint. The topology's stiffness-to-volume ratio is penalized by a ($a = 6$, 10, 14, 18, 22, 26) percent for every internal hole in addition to the b ($b = 0$, 1, 2, 3, 4) allowable holes. The penalty is attenuated according to the current generation number.

$$\text{Fitness} = \left\{1.0 - \left[\frac{a}{100} \cdot \frac{\text{generationNumber}}{175} \cdot (\text{NumberOfHoles} - b)\right]\right\} \frac{1}{\text{Displacement} \cdot \text{Area}}.$$
$$a = 6, 10, 14, 18, 22, 26, \quad b = 0, 1, 2, 3, 4.$$

Including all parameter values (i.e., Functions 2, 7, and 8) and parameter combinations (i.e., Function 9), 54 unique fitness functions were represented in the function suite. So that statistically-significant comparisons could be made between the functions, 10 experiment replications were conducted with each function variant (resulting in a total of 540 optimization runs, each with different random initial populations). Each optimization used a population of 30 160-gene chromosomes and was run for 225 generations. At the conclusion of each optimization, the optimal topology was recorded, as was its stiffness-to-volume ratio and number of internal holes. Note that because the above fitness functions take stiffness to be inversely proportional to maximum displacement, an assumption valid only for single loading, this example's conclusions are not necessarily valid for general loading cases.

Results of the experiment, shown in Fig. 10(a–f) (which depict the topologies found with highest stiffness-to-volume ratio for each specified number of internal holes), demonstrate that the GA can generate topologies combining high stiffness-to-volume ratio with varying numbers of internal holes.

While best stiffness-to-volume ratio performance is obtained using Functions 1, 2, 3, 7, and 9 (statistically equivalent effectiveness with respect to stiffness-to-volume ratio, see [9]), topologies with the lowest number of internal holes are obtained using Functions 3, 4, 5, 6, and 9 (statistically equivalent effectiveness with respect to reducing number of internal holes, see [9]), and topologies with the highest number of internal holes are obtained using Function 1. Hence, Functions 3 and 9 should be used to generate topologies combining highest stiffness-to-volume ratio with highest material and void element agglomeration, while Function 1 should be used to generate topologies combining the highest stiffness-to-volume ratio with the highest number of small, uniformly distributed internal holes. Fig. 11 shows the mean performance of the 9 fitness function groups.

5.4. Example 4 [14,15]: design space subdivision with speciation

This example shows how a speciating GA is used to distribute subsets of the evolving population of solutions over the design space. This distribution of solutions is analogous to different species exploiting different niches in an ecosystem. Additionally, we use statistical cluster analysis techniques to quantify the

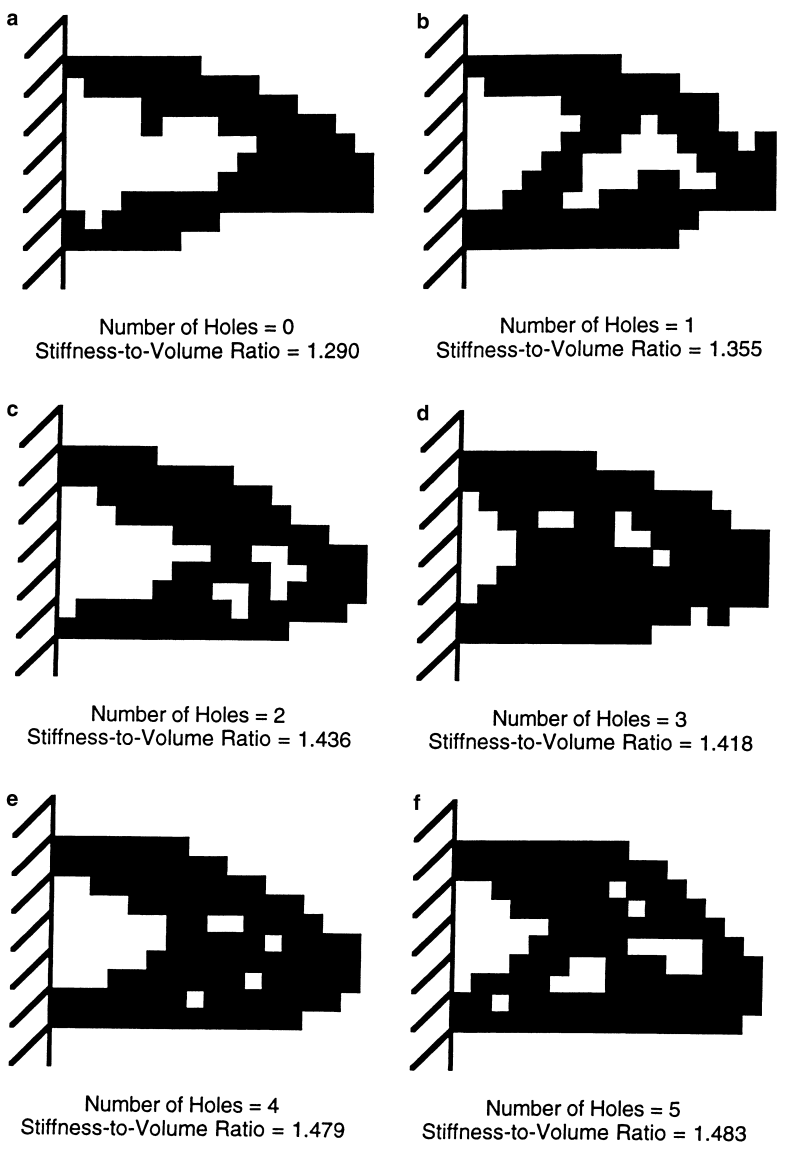
a
Number of Holes = 0
Stiffness-to-Volume Ratio = 1.290
b
Number of Holes = 1
Stiffness-to-Volume Ratio = 1.355
c
Number of Holes = 2
Stiffness-to-Volume Ratio = 1.436
d
Number of Holes = 3
Stiffness-to-Volume Ratio = 1.418
e
Number of Holes = 4
Stiffness-to-Volume Ratio = 1.479
f
Number of Holes = 5
Stiffness-to-Volume Ratio = 1.483

Fig. 10. Topologies combining high stiffness-to-volume ratio with few internal holes [9].

extent to which a population is speciated and use this measure to probabilistically encourage mating of reasonably similar designs (i.e., intraspecies mating).

Goldberg and Richardson [17] suggest the use of sharing functions as a means to cause speciation. Chromosomes share their ‘raw’ fitness function payoff with all other chromosomes in the entire population. The sharing with each chromosome is inversely related to the ‘distance’ between the pairs of chromosomes. Distance can be computed in the genotype or phenotype space, and is used in a sharing function. Sharing function values range from zero, for two chromosomes that are very distant, to one for two chromosomes that have no distance between them (i.e., they are the same). Such a sharing function allows a fitness reduction due to sharing with all chromosomes to be computed by summing the pairwise sharing function values and dividing the raw fitness value by this sum. This can be stated as follows, with x representing chromosomes, d representing distance, and s representing the sharing function:

$$\text{fitness}_{\text{shared}}(x_i) = \frac{\text{fitness}_{\text{raw}}(x_i)}{\sum_{j=1}^{n} s(d(x_i, x_j))}. \tag{7}$$

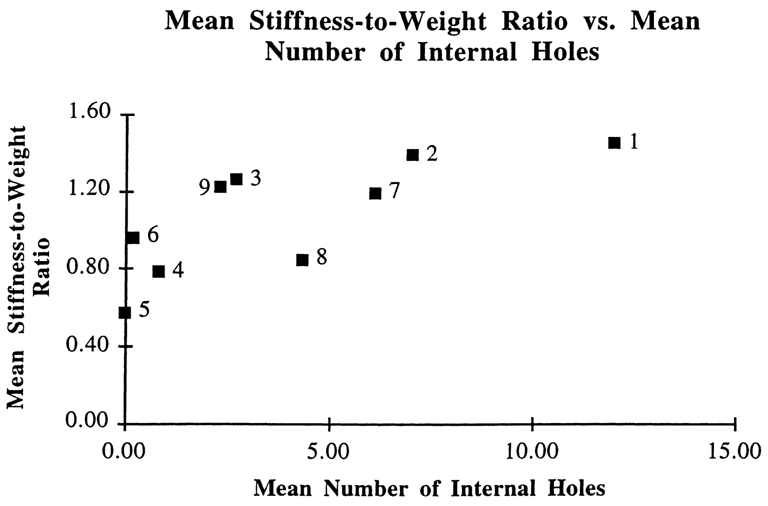

Fig. 11. Mean fitness function performance [9].

Such a shared fitness will evolve chromosomes that cluster about peaks in numbers proportional to the amount of fitness function value available at those peaks. These clustered subgroups are analogous to species exploiting ecological niches. Note, however, that reducing the fitness in no way precludes two very dissimilar chromosomes from being mated, so long as (reduced) fitness is the only criterion for choosing parents. This would be like two very different species trying to mate! It has been shown that restricting mating to similar chromosomes can improve GA performance (see, e.g., [13]), and this technique proves useful in our practical application.

This example considers a beam supported at each end and subjected to a vertical load at its midpoint. The design domain used was an 8×21 discretization, as shown in Fig. 12. The lower two nodes of each end element were constrained to have zero displacement, and the point load was applied to the node at the center of the element at the midpoint of the beam, shown in black in Fig. 12. The goal of this optimization was to minimize the structure's weight while maintaining a displacement less than the specified value ($d_{\max}$) at the load point. Material properties for this example were those of steel ($E = 200$ GPa, $v = 0.3$). The size of each element in the design domain is 1 cm $\times$ 1 cm $\times$ 1 mm thick. The magnitude of the applied load was set to 10 kN, and $d_{\max}$ was 0.01 m.

Unlike the previous examples, here we employ a two-dimensional binary array chromosome, as it was found to perform as well or better for the problem considered. First, connectivity analysis is performed to identify disconnected material. Then, a normalized mass is computed as the percentage of (connected) elements in which material is present. The connectivity analysis also checks to see if the point of load application is connected to the fixed nodes by material. If the loaded element is not connected to either material constraint element, the structure's fitness is given by (8a); if it is connected to only one material

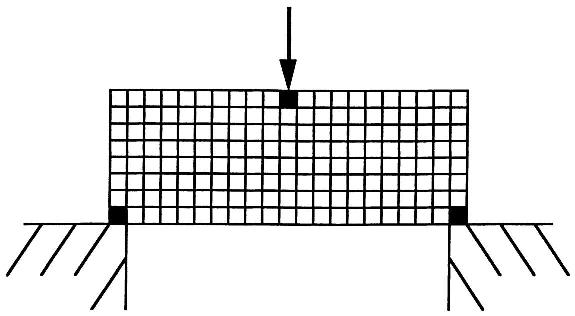

Fig. 12. Beam design domain [14].

constraint element, (8b) is used. If the load point is connected to both material constraint elements, the fitness score is calculated according to (8c) if the displacement constraint is violated and by (8d) if it is not. These equations provide the raw fitness score of a chromosome. When fitness sharing is used, the reduced fitness score is calculated as explained previously.

$$\text{fitness} = 5 \times \text{mass}, \tag{8a}$$

$$\text{fitness} = 50 \times \text{mass}, \tag{8b}$$

$$\text{fitness} = 1000 - 500 \times (d - d_{\max}), \tag{8c}$$

$$\text{fitness} = 1100 - \text{mass} \times 100. \tag{8d}$$

In order to perform sharing-based speciation and mating restriction during evolution, a sharing function is needed for the binary material/void representation. The sharing function we have chosen, called the Jaccard Coefficient (see [37, p. 143]), is applied to the genotype designs, after performing connectivity analysis. Comparing two material/void design arrays on an element-by-element basis, first determine the following intermediate parameters:

a = number of corresponding elements that are both material,
b = number of corresponding elements, where exactly one is material.

The Jaccard similarity coefficient is then computed as follows:

$$C_{\mathrm{J}} = \frac{a}{a+b}. \tag{9}$$

Note that the coefficient ranges from a value of zero for no material/material matches to one for an identical material/material match over the domain.

We also use this sharing function to recognize the formation of species during the evolution and use this information to restrict mating between excessively dissimilar chromosomes. This is done by using cluster analysis techniques (again, see [37]) to partition the population into subsets of similar chromosomes at each generation. The first step is to compute the ‘resemblance’ between each pair of designs. This is done by computing the Jaccard coefficient for each pair. The pair that is most similar (i.e., highest Jaccard coefficient value) is ‘clustered’ and treated as a single entity as long as its similarity is above a chosen threshold. The most similar pair of designs in the new set (now reduced in number by one) is found next, and the process continues as such until no further clustering is possible.

It is possible to measure the ‘tightness’ of a clustering by comparing the actual similarities between pairs of designs with the similarities that were used in the clustering process. These similarities differ, as described above, when clusters are treated as a single point in the computation of the Jaccard coefficient (the average distance to the designs in the cluster is used). Intuitively, the tightness of the clustering is related to these differences: how much was the data ‘distorted’ in order to consider it clustered? An aggregate value for this notion is provided by the Pearson product-moment correlation coefficient, $r_{x,y}$ [1] (see [37]) which varies from zero to one. Only pairs that were clustered (i.e. nearer than the threshold) contribute to the computation of the Pearson coefficient.

During an evolutionary optimization, the value of $r_{x,y}$ is monitored to determine the extent to which sharing-based speciation has partitioned the population. Note that initial randomly generated populations can be very well speciated (in a non-useful manner) if the chromosomes are mutually dissimilar. $r_{x,y}$, therefore, typically drops in value in the early generations and then grows to a higher value. When niche formation has sufficiently progressed, interspecies mating is increasingly unlikely to produce improved

[1] The process we have just outlined is actually an abbreviated version of a standard clustering process as outlined in ([37, Chapter 2]). In the complete process there is no clustering threshold and *all point pairs are eventually clustered.* $r_{x,y}$ for such a full clustering is referred to as the ‘cophenetic correlation coefficient’.

designs. We take this into account by using $r_{x,y}$ as another probabilistic factor on the occurrence of mating. After mating restriction is started, for each generation $r_{x,y}$ is computed with a threshold value of C_J. Each time a pair of parents is selected, there is an $r_{x,y}$% chance that this pair must be in the same species (defined by C_J). If this probabilistic test fails, the pair is discarded and a new pair is chosen for testing. In this way, the likelihood of interspecies mating decreases as $r_{x,y}$ increases. A typical threshold value for C_J is 0.85. In our tests, we have initiated mating restriction after a preset number of generations (halfway through a run). Over the second half of a run, $r_{x,y}$ typically ranges in value from 0.70 to 0.90.

Several GA runs were done with different settings. Runs were done with no fitness sharing or mating restrictions and with sharing but no mating restriction. Runs with fitness sharing and restricted mating were also done, with populations of different sizes. In each case, the GA was run for 200 generations, and clustering was done with a threshold value of 0.85. When restricted mating was implemented, it was started after 100 generations. Figs. 13–15 show typical results from these GA runs. Note in Fig. 15, as a result of the sharing-based speciation, that the largest cluster is characteristic of the best designs found. Table 1 summarizes the results. Note that each entry in this table represents the average result from multiple GA runs done with the same parameter settings.

5.5. Example 5: real number chromosomes

All of the examples shown so far have used a binary material/void representation, which has led to a discretized and precisely defined arrangement of material. GAs can also operate with chromosomes made up of real numbers [45]. Using such a representation, we have optimized structures in which the material has a continuously-variable density. Fig. 16 shows a structural design domain made up of 144 planar triangular finite elements with 85 finite element nodes. The density of the material is allowed to vary from zero to one at each node and the density within elements is interpolated from the element nodal densities. A real-numbered chromosome of length 85 was used to represent the nodal densities and a quadratic crossover specifically devised for real-numbered chromosomes (see [1]) was used along with Gaussian mutation. As all nodes of the domain were encoded, the symmetry of the domain was not exploited. The weight of the structure was minimized subject to a displacement constraint for horizontal and vertical loading cases.

Fig. 17 shows the results (attributable to Adewuya and Jakiela, to date unpublished) for a tensile horizontal loading case (left) and a vertical loading case (right). Although the finite element edges clearly influence the solution, a continuously-variable density, like homogenization solutions, is evident.

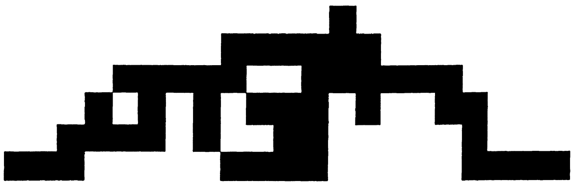
Fig. 13. Typical result with no fitness sharing and no mating restriction [14].

Table 1
Results of beam optimization [14]

	Population size	Best fitness score	Mass (%)	Number of clusters
No fitness sharing	60	1072	28	1
Fitness sharing, unrestricted mating	60	1069	31	8
Fitness sharing, restricted mating	60	1071	29	7
Fitness sharing, restricted mating	75	1071	29	10
Fitness sharing, restricted mating	90	1071	29	10

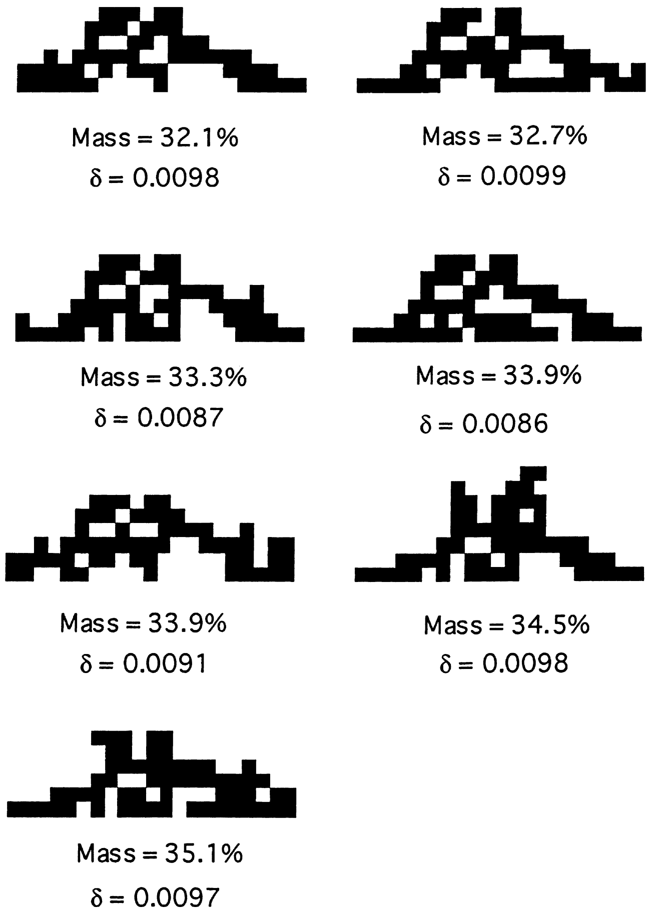

Fig. 14. Results of GA run with fitness sharing and no restricted mating [14].

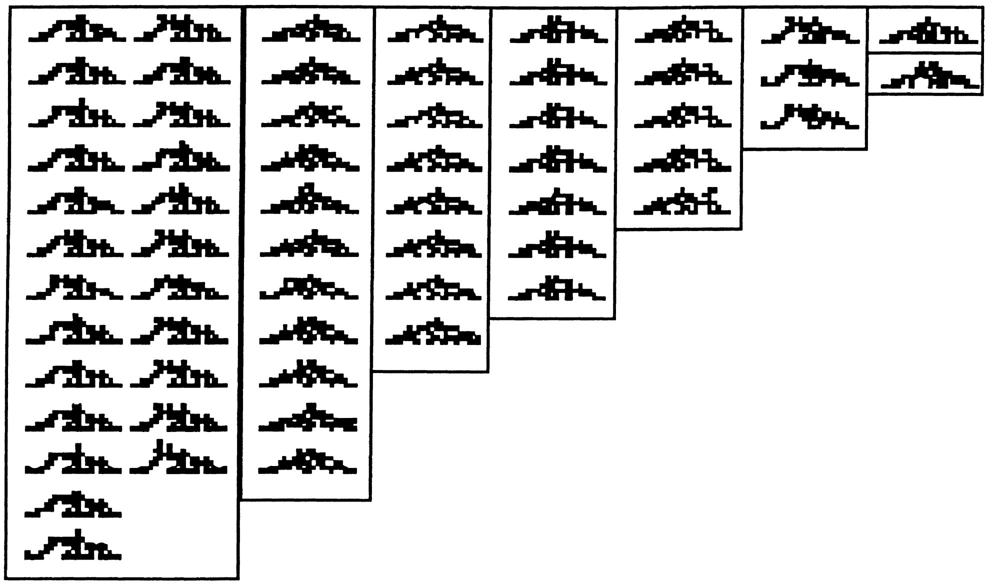

Fig. 15. Final population of GA with fitness sharing and restricted mating. Individuals grouped by cluster. Largest cluster represents best characteristic design [14].

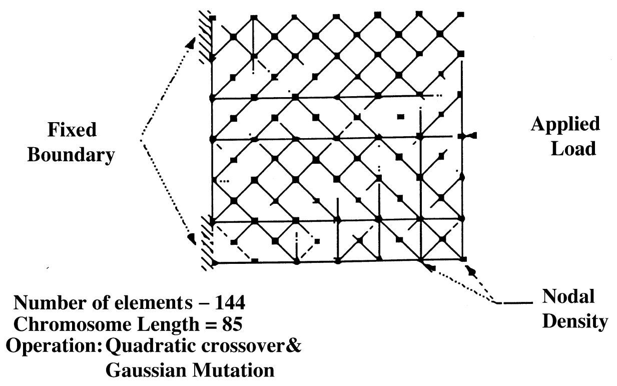

Fig. 16. Continuously variable density design domain.

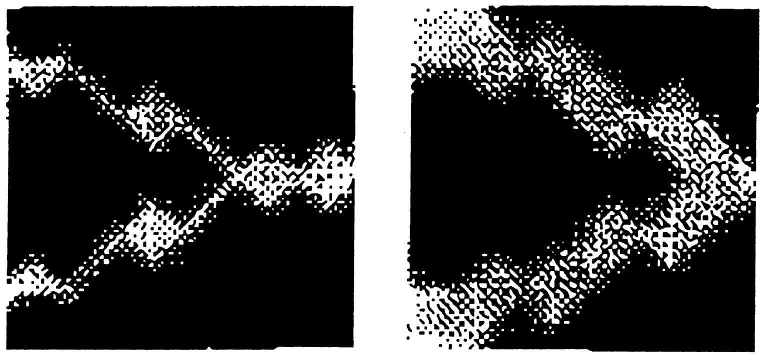

Fig. 17. Structures generated with continuously variable density design domain. Tensile horizontal loading (left) and vertical loading (right).

6. Discussion

In closing, we note that because they require a large number of function evaluations, GA solutions are computationally expensive in many cases and computationally prohibitive in some cases. Their corresponding advantage is that, since they only require zero'th order function evaluations, they can be applied to problems for which little is known about the nature of the design domain. Other researchers have applied them to domains that require similarly expensive function evaluations involving physical simulation, notably antenna design [2], with excellent results. We have begun to investigate techniques to reduce the required number of function evaluations [44].

Specifically, with regard to structural optimization, the examples included here demonstrate both the advantages and disadvantages of GA's. For relatively small problems, the GA-based solutions still require an arguably impractical amount of computation. Still, the versatility and ease of application of a GA is clear: the same representation and solution technique can be robustly applied to a variety of objective functions. Our recommendation would be to continue investigating the use of GA's for structural optimization, with emphasis on two specific topics. One is the acceleration or reduction of expensive function evaluations. Parallelization of the structural analyses would allow a global search as was demonstrated in Example 4. Approaches that use approximations to reduce the number of function evaluations, such as response surface techniques, would be similarly useful. The other topic would be the use of GA's for mixed variable optimizations. An advantage of GA's is that they can very easily be applied to problems with both discrete and continuous variables. In a structural optimization context, the discrete variables could describe high level topological changes and the continuous variables could represent more localized parameters such as sizes and densities. In this regard, 'hybridizing' a GA with other structural optimization techniques could be useful. A GA performing a discrete search, for example, could search for the best topological conditions for a related homogenization optimization.

Acknowledgements

This work was supported by the National Science Foundation with a Presidential Young Investigator's grant (DDM-9058415). Matchable funds for this grant were provided by Schlumberger Technology. These sources of support are gratefully acknowledged. Additionally, the software was developed using the computational facilities of the Computer-Aided Design Laboratory at the Massachusetts Institute of Technology, Department of Mechanical Engineering. This support is also gratefully acknowledged. Finally, Mark Jakiela gratefully acknowledges the support he currently receives as the Hunter associate professor of mechanical design at the Washington University in St. Louis.

References

[1] A. Adewuya, New methods in genetic search with real-valued chromosomes, M.S. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, 1996.
[2] E. Altshuler, D. Linden, Wire-antenna design using genetic algorithms, IEEE Antennaa and Propagation Magazine 39 (2) (1997) 33–43.
[3] G. Anagnostou, E. Ronquist, A. Patera, A computational procedure for part design, Computer Methods in Applied Mechanics and Engineering 97 (1992) 33–48.
[4] J. Baker, Reducing bias and inefficiency in the selection algorithm, in: Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms, Massachusetts Institute of Technology, July 1987 pp. 14–21.
[5] M. Bendsoe, A. Diaz, N. Kikuchi, Topology and generalized layout optimization of elastic structures, in: M. Bendsoe, C. Soares (Eds.), Topology Design of Structures, NATO ASI Series, 1993, pp. 159–205.
[6] M. Bendsoe, N. Kikuchi, Generating optimal topologies in structural design using a homogenization method, Computer Methods in Applied Mechanics and Engineering 71 (1988) 197–224.
[7] M. Bremicker, M. Chirehdast, N. Kikuchi, P. Papalambros, Integrated topology and shape optimization in structural design, Mechanics of Structures and Machines 19 (1991) 551–587.
[8] C. Chapman, Structural topology optimization via the genetic algorithm, M.S. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, 1994.
[9] C. Chapman, M. Jakiela, Genetic algorithm-based structural topology design with compliance and topology simplification considerations, ASME Journal of Mechanical Design 118 (1) (1996) 89–98.
[10] C. Chapman, M. Jakiela, Genetic algorithm-based structural topology design with compliance and manufacturability considerations, in: Proceedings of the ASME 20th Design Automation Conference: Advances in Design Automation, DE-vol. 69(2), American Society of Mechanical Engineers, New York, 1994, pp. 309–322.
[11] C. Chapman, K. Saitou, M. Jakiela, Genetic algorithms as an approach to configuration and topology design, ASME Journal of Mechanical Design 116 (4) (1994) 1005–1012.
[12] C. Chapman, K. Saitou, M. Jakiela, Genetic algorithms as an approach to configuration and topology design, in: Proceedings of the ASME 19th Design Automation Conference: Advances in Design Automation, DE-vol. 65(1), American Society of Mechanical Engineers, New York, 1993, pp. 485–498.
[13] K. Deb, D. Goldberg, An investigation of niche and species formation in genetic function optimization, in: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 42–50.
[14] J. Duda, Speciation, clustering, and other genetic algorithm improvements for structural topology optimization, M.S. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, 1996.
[15] J. Duda, M. Jakiela, Generation and classification of structural topologies with genetic algorithm speciation, ASME Journal of Mechanical Design 119 (1) (1997) 127–131.
[16] D. Goldberg, Genetic Algorithms in Search Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[17] D. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Earlbaum Associates, Hillsdale, NJ, 1987, pp. 41–49.
[18] D. Goldberg, M. Samtani, Engineering optimization via genetic algorithm, in: Electronic Computation – Proceedings of the Ninth Conference on Electronic Computation, American Society of Civil Engineers, University of Alabama at Birmingham, February, 1986, pp. 471–482.
[19] J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Transactions on Systems, Man, and Cybernetics SMC 16 (1) (1986) 122–128.
[20] D. Grierson, W. Pak, Discrete optimal design using a genetic algorithm, in: M. Bendsoe, C. Soares (Eds.), Topology Design of Structures, NATO ASI Series, 1993, pp. 89–102.
[21] P. Hajela, E. Lee, C. Lin, Genetic algorithms in structural topology optimization, in; M. Bendsoe, C. Soares (Eds.), Topology Design of Structures, NATO ASI Series, 1993, pp. 117–133.

[22] P. Hajela, Genetic algorithms in automated structural synthesis, in: B. Topping (Ed.), Optimization and Artificial Intelligence in Civil and Structural Engineering, vol. 1, Kluwer Academic Publishers, Dordrecht, 1992, pp. 639–653.

[23] P. Hajela, Genetic search – an approach to the non-convex optimization problem, AIAA Journal 28 (7) (1990) 1205–1210.

[24] W. Jenkins, Structural optimization with the genetic algorithm, The Structural Engineer 69 (24) (1991) 418–422.

[25] W. Jenkins, Towards structural optimization via the genetic algorithm, Computers and Structures 40 (5) (1991) 1321–1327.

[26] R. Haber, C. Jog, M. Bendsoe, Variable-topology shape optimization with a control on perimeter, in: Proceedings of the ASME 20th Design Automation Conference: Advances in Design Automation, DE-vol. 69(2), American Society of Mechanical Engineers, New York, 1994, pp. 261–272.

[27] R. Haftka, R. Grandhi, Structural shape optimization–a survey, Computer Methods in Applied Mechanics and Engineering 57 (1986) 91–106.

[28] J. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, Michigan, 1975.

[29] E. Jensen, Topological structural design using genetic algorithms, Doctor of Philosophy Thesis, Purdue University, 1992 (November).

[30] S. Kirkpatrick, C. Gelatt, M. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.

[31] U. Kirsch, Structural Optimization: Fundamentals and Applications, Springer, Berlin, 1993.

[32] C. Lin, P. Hajela, 1993, Genetic search strategies in large scale optimization, AIAA Paper #93-1585, IAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, La Jolla, CA, April, 1993.

[33] S. Rajeev, C. Krishnamoorthy, Discrete optimization of structures using genetic algorithms, Journal of Structural Engineering 118 (5) (1992) 1233–1250.

[34] R. Richards, S. Sheppard, Learning classifier systems in design optimization, in: Proceedings of the 1992 Design Theory and Methodology Conference, DE-vol. 42, American Society of Mechanical Engineers, Scottsdale, Arizona, 1992, pp. 179–186.

[35] P. Papalambros, M. Chirehdast, An integrated environment for structural configuration design, Journal of Engineering Design 1 (1990) 73–96.

[36] H. Rodrigues, Personal communication, 1993.

[37] C. Romesburg, Cluster analysis for researchers, Lifetime Learning Publications, Belmont, CA, 1984.

[38] E. Sandgren, E. Jensen, Automotive structural design employing a genetic optimization algorithm, SAE Technical Paper #920772, in: Proceedings of the SAE International Congress and Exposition, Detroit, February, 1992.

[39] E. Sandgren, E. Jensen, J. Welton, Topological design of structural components using genetic optimization methods, in: Sensitivity Analysis and Optimization with Numerical Methods, AMD-vol. 115, Proceedings of the Winter Annual Meeting of the American Society of Mechanical Engineers, Dallas, TX, 1990, pp. 31–43.

[40] J. Schaffer, R. Caruana, L. Eshelman, R. Das, A study of control parameters affecting online performance of genetic algorithms for function optimization, in: Proceedings of the Third International Conference on Genetic Algorithms, George Mason University, June, 1989, pp. 51–60.

[41] N. Shankar, P. Hajela, Heuristics driven strategies for near-optimal structural topology development, in: B. Topping (Ed.), Artificial Intelligence and Structural Engineering, Civil-Comp Press, 1991, pp. 219–226.

[42] H. Watabe, N. Okino, A study on genetic shape design, in; Proceedings of the Fifth International Conference on Genetic Algorithms, University of Illinois at Urbana–Champaign, July, 1993, pp. 445–450.

[43] G. Strang, R. Kohn, Optimal design in elasticity and plasticity, International Journal for Numerical Methods in Engineering 22 (1986) 183–188.

[44] J. Zhang, M. Jakiela, Heuristically skipping function evaluations in genetic algorithms, Submitted for review to IEEE Transactions on Evolutionary Computation, 1998.

[45] L. Davis, Hybridization and numerical representation, in L. Davis (Ed.), The Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991, pp. 61–71.