Efficiency of genetic algorithms for optimal structural design considering convex models of uncertainty

S. Ganzerli
Dept. of Civil Engineering, Gonzaga University, Spokane, WA, USA

Paul DePalma & J. D. Smith
Dept. of Computer Science and Mathematics, Gonzaga University, Spokane, WA, USA

M. F. Burkhart
Dept. of Computer Science, University of Washington, Seattle, WA, USA

Keywords: Convex models, genetic algorithms, optimization, trusses, uncertainties.

ABSTRACT: The aim of this paper is to obtain the optimal design of trusses under uncertain static loads. Genetic algorithms are used to carry out the structural optimization and convex models are employed to handle the uncertainties. A study of the efficiency of genetic algorithms is performed. Using the uniform bound convex model affects the computational speed of the algorithm. In fact, it requires as many analyses as the number of uncertainties. However, it is shown that the problem still can be expressed as a polynomial time algorithm. Genetic algorithms prove to be most efficient when the initial population is chosen making an educated guess. A preliminary design is carried out first using integers and then switching to floating points. Finally, the efficiency of genetic algorithms is clear when using integer numbers for which convergence is more rapid than with real numbers.

1 INTRODUCTION

The focus of this paper is optimal design of trusses under uncertain load magnitudes. Convex models are implemented to handle the uncertain parameters and genetic algorithms (GA) are used for minimizing the structural volume.

Convex models have been proposed by Ben-Haim and Elishakoff (1990). They are a nonprobabilistic method and, together with probability and fuzzy sets, they compose the “uncertainty triangle” (Elishakoff, 1995). Since their discovery, convex models have been employed to address different kinds of uncertainties. In structural design, convex models have been implemented using the anti-optimization technique (Elishakoff. et. al, 1994). At each structural optimization cycle in order to minimize the structural cost, a complete optimal routine needs to be nested to maximize the load effects on the structure. This process is computationally quite expensive. In addition, it requires the constraints to be written as an explicit function of the design variables. A superposition method has been proposed by Ganzerli and Pantelides (2000). This method allows accounting for a large number of uncertainties and structures with many members. The superposition method is used in this paper. Ben-Haim et al. (1996) have shown the efficiency of convex models in identifying the worst-case-scenario due to uncertainties. In fact, for structures subjected to uncertain static loads, the maximum structural response due to the uncertain parameters cannot be identified simply by increasing each of the parameters to their maximum value. However, the convex model design proves resistant in terms of constraint violations. Recently, convex models have proven efficient in the study of thin-walled stiffened composite panels that are highly sensitive to geometrical imperfections (Elseifi et al., 1998). In this paper, a comparison between the convex models and a Monte Carlo simulation led to similar results but with an effort and cost reduction in favor of the convex models. Recently (Tonon et al., 2001) hybrid systems that combine the three methods available to deal with uncertainties, consisting of probability, fuzzy sets, and convex models, have been explored.

Traditional optimization techniques encompass: (1) mathematical programming, (2) optimality criteria, (3) approximation methods, and (4) steepest descent methods. Optimal structural design has been implemented for many years (Kirsh, 1981). In the traditional optimization, the domain is searched using the gradient of the objective function. A limitation arises when the function is not continuous and it is not possible to calculate its gradient. In the 1960s and 1970s Genetic Algorithms (GA) were conceived and developed by John Holland (1975). GA gained popularity after David Goldberg (1989), one of Holland's students, succeeded in solving a complicated problem presented in his dissertation. GA for optimal design of trusses have been extensively explored. In recent years, Rajeev and Krishnamoorthy (1997) have experimented with GA in the optimal design of trusses, and Ghasemi et al (1999) have demonstrated the suitability of GA to address large trusses with many uncertain variables. Optimization of large trusses using traditional algorithms was presented by Schemit, L.A. and Lai, Y.C. (1994). However, the advantage of GA consists in the possibility of considering discrete variables.

Only two publications exist that combine convex models with GA. Cho, M., and Rhee, S.Y. (2001) have studied the layup optimization for free edge strength utilizing GA for the optimal design and convex models to deal with uncertainties. Ganzerli and Burkhart (2002) have implemented GACON, a GA based optimization routine combined with the uniform bound convex model to deal with load uncertainties.

2 CONVEX MODELS OF UNCERTAINTY

In this paper convex models are employed in the structural design of trusses subject to an uncertain static load condition. The uniform bound convex model was chosen because of its easy implementation. To illustrate its application a simple structure in Fig. 1 is considered.

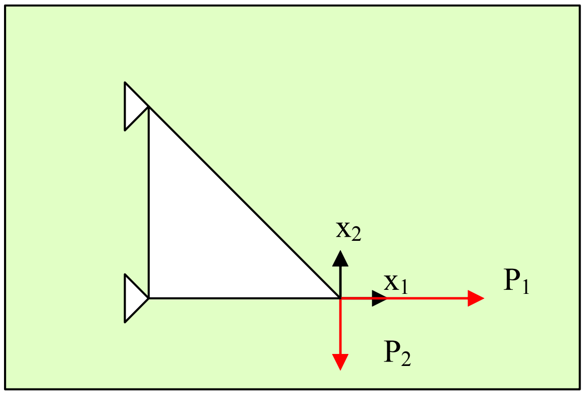

Figure 1. Three-bar truss.

The simple three bar truss has two degrees of freedom (x_1 and x_2) and is loaded by two static loads (P_1 and P_2). The latter represent the uncertain parameters. For the convex models to be applicable the uncertainties need to be bound within a convex set C. This can be represented as follows:

$$C = \begin{cases} P_1^L < P_1^n < P_1^U \\ P_2^L < P_2^n < P_2^U \end{cases} \tag{1}$$

where

P_i^L, P_i^n, and P_i^U are the lower, nominal and upper values respectively for the ith load.

The nominal values of the loads, named P_1^n and P_2^n, do not account for uncertainty. The upper and lower limits (P_i^L, P_i^U) are obtained by varying the nominal values by a given percentage β_i, as follows:

$$\begin{aligned} P_i^U &= P_i^n + \beta_i P_i^n \\ P_i^L &= P_i^n - \beta_i P_i^n \end{aligned} \tag{2}$$

The convex set can be represented graphically by a rectangle, if only two uncertainties are present (Fig. 2).

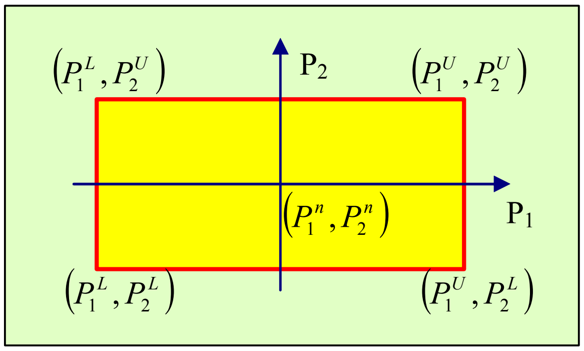

Figure 2. The convex domain.

The uncertainties are allowed to vary within the convex set. However, for a problem such as the one addressed in this paper, only the four corners of the convex model need to be considered. In fact, it has been demonstrated that the maximum effect due to the uncertainties should be found on the convex hull, in this case the perimeter. In particular, when the problem presents a linear relationship between the uncertain loads and the structural response (in terms of displacements and stresses) the maximum effect due to the uncertainty should be searched on the four corners of the rectangle (Elishakoff et al., 1994).

The convex structural response can be conveniently found through a superposition method. The convex portion is added to the nominal values of the structural response (displacements, forces and stresses) calculated when all the given nominal loads are acting on the structure. The latter consists of summing the nominal structural responses calculated with one load at a time and multiplying them by the per-cent of uncertainty for that load. The resulting expressions for the convex displacement and force vectors are:

$$\mathbf{x}_{\mathbf{con}} = \mathbf{x}(P_1^n,\ P_2^n) \pm \{\beta_1\, |\mathbf{x}(P_1^n)| + \beta_2\, |\mathbf{x}(P_2^n)|\} \tag{3}$$

$$\mathbf{F}_{\mathbf{con}} = \mathbf{F}(P_1^n,\ P_2^n) \pm \{\beta_1\, |\mathbf{F}(P_1^n)| + \beta_2\, |\mathbf{F}(P_2^n)|\} \tag{4}$$

where

- $\mathbf{x}_{\mathbf{con}}$ and $\mathbf{F}_{\mathbf{con}}$ are the convex displacements and internal forces

- $\mathbf{x}(P_1^n, P_2^n)$ and $\mathbf{F}(P_1^n, P_2^n)$ are the nominal displacements and internal forces calculated loading the structure with both P_1^n and P_2^n;
- $|\mathbf{x}(P_1^n)|$ and $|\mathbf{F}(P_1^n)|$ are the absolute values of the nominal displacements and internal forces calculated loading the structure with only P_1^n ($P_2 = 0$);
- $|\mathbf{x}(P_2^n)|$ and $|\mathbf{F}(P_2^n)|$ are the absolute values of the nominal displacements and internal forces calculated loading the structure with only P_2^n ($P_1 = 0$);
- β_1 and β_2 are the percent of uncertainty of P_1 and P_2 respectively.

In Eq. (3) the ± sign is in agreement with the sign of the first term. In other words, if $\mathbf{x}(P_1^n, P_2^n)$ are positive the plus/minus sign will turn into a plus sign and vice versa. This guarantees that the nominal displacements are always increased when uncertainty is present and the worst case scenario due to the uncertain parameters is captured by the equations.

Convex stresses can be obtained directly from the convex internal forces just by dividing the latter by the member cross sectional area (Wang, 1986).

In conclusion, to account for uncertainties in the structural design, it is sufficient to substitute the convex responses to the nominal ones in the structural analysis routine. One important note is that although the uniform bound convex model is implemented here jointly with genetic algorithms, it can also be introduced in the conventional (non optimal) design of structures.

3 GENETIC ALGORITHMS

GA is an optimization method that mimics the natural selection process (Haupt and Haupt, 1998). An initial population of individuals is randomly generated and ranked based on desired characteristics. Only the fittest individuals are selected to reproduce. Fitness is evaluated through a "cost function," the traditional name of the function to be optimized. For a constrained optimization, it is necessary to apply the constraints in terms of penalty functions to count against the cost of the individuals. In this way, an individual that violates the constraints will be ranked among the subset of the population that will not reproduce. Once the fittest individuals are selected, the mating process begins. Individuals have to be paired. A crossover operator needs to be implemented to establish which genes will be passed from the parents to the offspring. Once the process of reproduction is completed, the number of individuals will be the same as the one in the initial population. The percentage of the population retained for reproduction, and the number of offspring generated are design parameters. An important element is mutation. The offspring will contain genes inherited from the parents. Mutation guarantees that new genes are introduced in the gene pool. This ensures that some alleles (gene values) will be introduced that were not randomly generated at the beginning. The optimization process is sensitive to the mutation rate. If mutation is too fast the algorithm might quickly converge to a local optimal value. Mutation also affects the speed of convergence.

For the design problem considered in this paper the aim is to optimize trusses for cost (identified with the structural volume). The population is composed of trusses that represent the individuals. The ranking is based on cost and on penalty functions assigned to individuals (trusses) that violate the constraints. Stress and displacement constraints are accounted for. The design variables are the truss members cross sectional areas that will play the role of the genes. Their values are the alleles.

4 EXAMPLES

In this section three examples are presented. The trusses are of increasing complexity in terms of the number of members and nodes. The static loads are considered uncertain in magnitude and the uniform bound convex model is implemented. The scope of the optimal design is to minimize the structural volume subject to stress and displacement constraints. The design variables are the member cross-sectional areas.

4.1 *10-bar truss*

The following 10-bar truss (Fig. 3) is widely represented in the literature on optimization (Ganzerli and Burkhart, 2000).

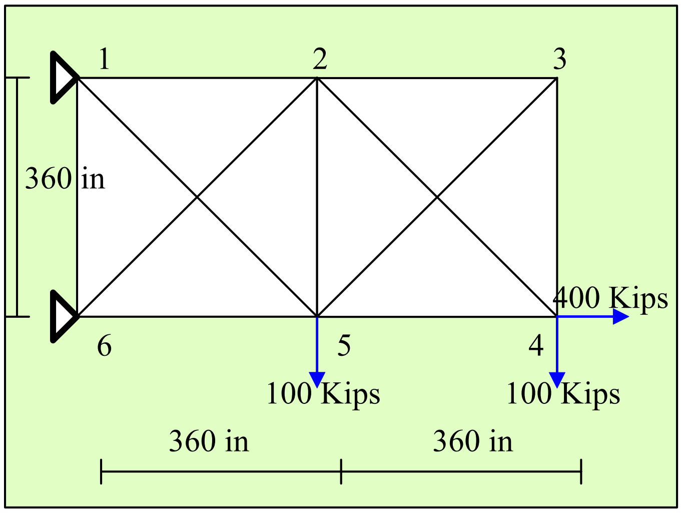

Figure 3. 10-bar truss.

The truss is aluminum (E = 10,000 Ksi). The uncertain loads all vary by 10 percent from their nominal values, represented in the Fig. 3. Constraints are imposed on stresses that are limited, for both tension and compression to a value of 25 Ksi for all members but member 2-4 for which the maximum stress is set at 75 Ksi. The displacements are limited to 5 in.

4.2 *19-bar truss*

Figure 4 represents a 19-bar truss with 11 nodes and 19 degrees of freedom. E = 10,000 Ksi. The vertical loads have an uncertainty of 20 percent and the horizontal loads have an uncertainty of 10 percent. The nominal values for the loads are given in Fig. 4. The constraints for the optimal design are imposed on stresses so that they can never exceed 25 Ksi for tension and compression, and on the horizontal displacement at node 11 that is limited to 3 in.

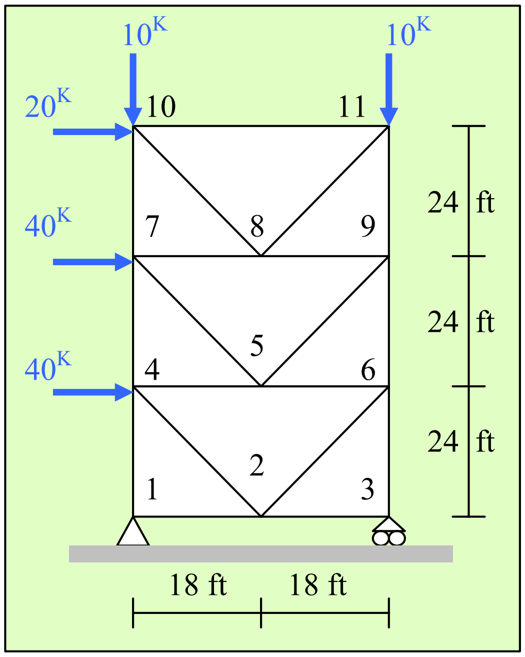

Figure 4. 19-bar truss.

4.3 *64-bar truss.*

Figure 5 represents a 64-bar truss with 28 nodes and 48 degrees of freedom. The aluminum truss has a Young's modulus equal to 10,000 Ksi. The length of all the horizontal and vertical members is 200 in. The load magnitudes (for which the nominal values are given in Fig. 5) are uncertain and their values are bounded as the convex models require. The 70 Kips loads have an uncertainty of 20 percent and the other loads have an uncertainty of 10 percent with respect to their nominal values. In this example, the allowable stresses for each member are set as ± 25 Ksi (for both tension and compression). Constraints are also imposed such that the vertical displacement at node 1 and the horizontal displacement at node 9 are less than 10 in.

This truss is presented in the literature with only deterministic loads. Ghasemi et al. (1999) reduce the variables (the member cross-sectional areas) to 19 by linking them. In this paper the 64 design variables are independent of each other.

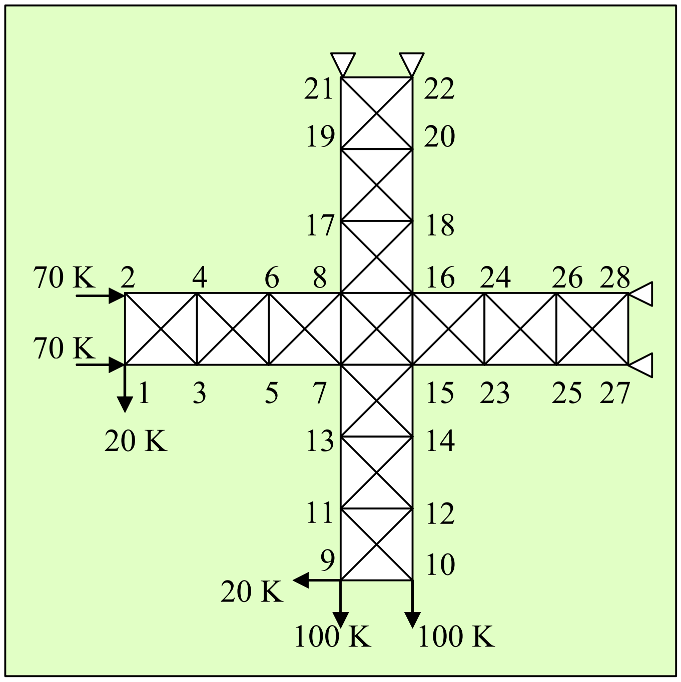

Figure 5. 64-bar truss.

4.4 *Results*

The optimal volumes for the three trusses described above are given in Table 1. The results are presented for both the integer optimization and the floating-points optimal design.

Table 1. Optimal results.

Truss	Optimal volume (integers)	Optimal volume (floating-points)
	in^2	in^2
10-bar	2.14 x 10^4	1.95 x10^4
19-bar	4.31 x 10^4	4.25 x 10^4
64-bar	5.19 x 10^4	4.07 x 10^4

As shown, GTO (Genetic Truss Optimizer), the code developed for this research, was successful in solving the posed problem of optimizing trusses subject to uncertain static loads. The program combines the GA routine with the structural analysis incorporating the uniform bound convex model. GTO was implemented in the C++ programming language and executed on a small Hewlett-Packard main frame running unix. Both the integer and floating point results are presented. The integer results are useful for two reasons. For a preliminary design, one could be satisfied with just an estimate of the sizes given by the integer variables. Additionally, the program converges faster if an educated guess is made and the initial population to obtain the floating–point solution is chosen based on the integer results.

A more detailed discussion on the benefits on using convex models and the robustness of a convex model design with respect to the uncertainties can be found in (Ganzerli and Pantelides, 2000). The aim of this paper is to analyze the algorithm itself and the feasibility of the implementation of the uniform bound convex model coupled with GA. From Eqs. (3) and (4) it can be understood how the computational effort is increased when using the convex model. In fact, the uniform bound convex model requires as many structural analyses as the number of uncertainties present. This important aspect will be addressed in the following section.

5 DISCUSSION AND FURTHER RESEARCH

A danger in using a computer to find an optimal or nearly optimal solution to a complex problem is that the problem under discussion might not yield a solution in reasonable time. The function associated with the cost of computing such problems grows exponentially relative to the size of the input parameters. The best known of these is the traveling sales person problem. Given a collection of cities, numbered 1 through n - 1, along with the distances between any two cities, the problem for the traveling sales person is to find a route of minimum cost. This problem can be represented with an n vertex graph. One vertex is the starting point, the others are the cities on the tour. The optimal solution is the collection of edges that, in sum, produces the shortest tour. Clearly, this is a permutation problem. There are n! candidate solutions. Somewhat surprisingly, an optimal solution cannot be found in reasonable time.

What is meant by reasonable time? Korsh (1986) offers a benchmark. Assume a computer that can carry out 10^6 basic operations (ops) per second. These include arithmetic operations, comparison, memory retrieval and so forth. Of course, these differ in cost from machine to machine, an objection that will be faced shortly. They also differ in cost among themselves. Since what is really being sought is an estimate of how bad things can get, assume that the hypothetical machine can carry out a million of the most expensive ops per second. If the computer were to run day and night for a year, it will have executed fewer than 10^{14} ops. Korsh suggests using this as a benchmark. Any solution requiring more computing power than a year's worth, is not reasonable. Somewhat surprisingly, 17! exceeds 10^{14}. By this measure of feasibility, the computation of the optimal seventeen-city tour is not feasible. One obvious objection is to find a faster computer, one that can carry out, say 10^9 ops per second. But this only increases the number of cities in the tour by two. The solution, if there is one, is not to be found in hardware.

Clearly what is important here is not the machine but the algorithm itself. Algorithms can be analyzed in two ways: *a posteriori* and *a priori* (Horowitz & Sahni, 1986). In an *a posteriori* analysis, the researcher collects run-time statistics about an algorithm as it executes on problems of varying sizes. In this analysis, the kind of machine makes a great deal of difference. But an *a posteriori* analysis makes sense only if it can be demonstrated in advance that the problem at hand will yield a solution in reasonable time. Thus, the problem of optimizing a truss using the convex model and a genetic algorithm is more fundamental than that of collecting run-time statistics. It is crucial to know if the approach to the problem is feasible by the definition. If it is not, then the next step is to determine if it falls into a class of problems, the NP-complete problems, for which there is no known polynomial time solution. The traveling sales person, in fact, is a member of this class (Garey & Johnson, 1979). Its membership was determined a priori, exactly the method that will be brought to bear on the truss problem. The method consists in finding a function on parameters from the problem description that provides an upper bound to the computing time of the algorithm.

Before proceeding, it is necessary to define, informally, two terms: polynomial time and exponential time algorithms. Suppose n represents some description of a problem's complexity. In the case of the traveling sales persons, this would be the number of cities on the tour. In the case of optimizing trusses with genetic algorithms there are several such parameters, most obviously the number of members, the number of nodes, the population size, and the number of generations that the algorithm must run through until it converges on a nearly optimal truss. Keeping things simple for the moment, a polynomial time algorithm is one whose run-time is bounded above by a polynomial of the form n^m. Notice that there is no m such that n^m forms an upper bound for 2^n. An informal way of saying this is that for sufficiently large n, $2^n > n^m$. We will call an algorithm exponential time if its *a priori* computing time is bounded below by 2^n. Not only does an exponential time algorithm increase more rapidly than a polynomial time algorithm, for large n the difference is dramatic and renders exponential time algorithms, by the definition, infeasible.

The good news is that the a priori run-time of the algorithm used to generate the results for this paper appears to be in the class of polynomial time algorithms. The cautious "appears to be" is used because the run-time depends functionally on the number of generations that the algorithm must produce. At this point in the research, several successful analyses of increasingly complex trusses have been made using a genetic algorithm and the convex model. If further research shows that the number of required generations increases exponentially with the complexity of the truss, another approach will be necessary. If on the other hand, the relationship can be shown to be one of polynomial time, as, indeed, it appears to be, the next task will be to refine the algorithm.

One possibility is to use explicit parallelism. For each generation, each of the trusses must undergo a structural analysis. Since these analyses are independent of one another, they are natural candidates for parallel programming. Now that machines running Linux may be inexpensively networked into clusters, such a solution is no longer the sole province of super-computing labs. Another is to explore the parallelism inherent in the individual computations themselves, through the use concurrent threads. This is a method used widely by makers of graphical user interfaces. Though the computational-intensive nature of structural analysis does not appear to offer the opportunity that graphical programming does to "steal" machine cycles, it is still an avenue worth investigating.

Now for the analysis. Any *a priori* analysis is, by its nature, a very rough measure of run-time efficiency. But since the task is only to decide if the genetic algorithm approach is feasible, it works quite nicely. The reason is that an *a priori* analysis will provide a worst-case measure. If it is possible to isolate the parameters on which the run-time of the algorithm depends, it can show that the computing time cannot exceed a function on those parameters. Computer programs spend most of their time in loops, that is, cycling through the same kind of calculation again and again. In the case of structural analysis using the convex model, one of the significant computing costs is matrix multiplication. It is well-known that the cost of matrix multiplication increases in proportion n^3, where n is the size of an n X n matrix. A significant cost of the genetic portion of the algorithm is the necessity of sorting the population of trusses at each generation. The rather sluggish, but simple, selection sort, is used in GTO. The run-time of selection sort can be shown to increase in proportion to the square of the number of items being sorted. Both of these algorithms, require what are known as nested loops, the first a loop within a loop within a loop, the second a loop within a loop. We have isolated several parameters that control loops. Using these, we have developed functions that describe the worst-case computing time.

One objection to this line of analysis, is that the computation that is performed within each of the many loops in the program varies widely. Some will perform one basic operation. Others will perform an integer multiple of a basic operation. Now, it is surely the case that among the loops, one will perform a collection of operations that are more expensive than the other loops. Call this collection B. If B is substituted for the cost of the operations in the other loops, the resulting function will have a value greater the actual cost of the functions.
If the function describing the algorithm can still be shown to be a polynomial time function, especially if it can be shown to be of relatively low order, then it is, at least theoretically, feasible.

Here, then, are the functions that describe GTO:
Let:
A = the total cost of running the genetic algorithm
T = the total cost of structural analysis per truss
C_n = the nominal cost for analyzing a single truss
C_c = the additional cost of analyzing a single truss using the convex model
P = the population size
G = the number of generations that the algorithm must produce and analyze
M = the number of members in the truss
N = the number of nodes in the truss
Then,

$$A = (G + 1)(2P^2 + 4PM + PT) + GP \qquad (5)$$

$$T = C_n + Cc \qquad (6)$$

$$C_n = 8N^3 - 24N^2 + 38N - 6M^2 + 9M - 68NM + 4NM^2 + 4MN^2 - 18 \qquad (7)$$

$$C_c = 32N^4 - 88N^3 + 96N^2 - 42N + M - 14NM - 48N^2M + 48N^3M + 8N^2M^2 - 6NM^2 \qquad (8)$$

These equations were obtained by a direct analysis of the algorithm that produced the results in this paper. The cost, here, is B, some integer multiple of the time it takes to perform a basic operation. Though these equations should be taken with a grain of salt, they do give some indication of how the run time of the algorithm will increase with an increasingly complex truss.

Given some constant, K, and sufficiently large N and M, it can be shown that:

$$T \leq KN^3(N + M) \qquad (9)$$

This is simply an informal statement of O-notation widely used in a priori analyses of computer algorithms (Horowitz and Sahni, 1986). Although, structural analysis using the convex model is very expensive computationally, it is still bounded above by 2^N and, so by the definition, a polynomial time algorithm. There is evidence to believe that G, the number of generations, does not increase exponentially with problem size. In fact, genetic algorithms were developed for just this purpose—to reduce the time necessary to find an a solution to certain problems from exponential to polynomial time. The price, of course, is that the solution is good enough, though not optimal. Only further research will uncover whether this is the case. If the hypothesis turns out to be correct, the algorithm--the structural analysis component and the genetic component—will improve with tighter programming, use of parallelism, and more investigation into the relationship between mutation factor, convergence percentage, and population size.

6 CONCLUSIONS

This paper has explored the use of the uniform bound convex model for studying uncertain static loads and the genetic optimal design of trusses. Implementing the uniform bound convex model increases the computational effort needed since, for each truss, it is required to perform as many structural analyses as the uncertain parameters present. Three examples of increasing complexity are solved and the GTO (Genetic Truss Optimizer) program developed for this research proves efficient in reaching a viable solution. The algorithm can produce both integer and floating-points results. The advantage in considering integer variables lies in being able to compute preliminary sizes for an initial design. Furthermore, the algorithm converges faster to an optimal solution using floating-points when the integer results are used as initial variables.

The *a priori* analysis of the GTO is performed to estimate the parameters on which the run-time depends. Using these parameters the functions were developed that describe the worst-case computing time. It is shown that, although the convex design is computationally expensive, it can still be considered a polynomial time algorithm and it will not increase exponentially with problem size.

7 APPENDIX

A Table for conversion factors from US to SI units is given below for the units used in this paper.

Table 2. Conversion factors.

inches (in)	millimeters (mm)	25.4
feet (ft)	meters (m)	0.305
pound force[a] (lb)	Newtons	4.448
Pounds per sq in[b] (psi)	Newtons per sq m (N/m^2)	6895

[a]Kips equal kilo-pounds = 1,000 lb
[b]Ksi equal kilo-pounds per square in = 1,000 psi

REFERENCES

Ben-Haim, Y., and Elishakoff, I. 1990. *Convex Models of Uncertainty in Applied Mechanics.* New York, NY: Elsevier.

Ben-Haim, Y., Chen, G., and Soong, T. T. 1996. Maximum structural response using convex models. *ASCE J. Engineering Mechanics.* Vol. 122: 325-333.

Cho, M., and Rhee, S.Y. 2001. Layup optimization by genetic algorithms for free-edge strength considering bounded uncertainty. *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference and Exhibit Technical Papers, Apr 16-19, 2001 Seattle, WA*. Vol 5 (No. 42): 3127-3136.

Elishakoff, I., Haftka, R. T., and Fang, J. 1994. Structural Design under Bounded Uncertainty Optimization with Anti-optimization. *Computers and Structures.* Vol. 53: 1401-1405.

Elishakoff, I. 1995. An Idea on the Uncertainty Triangle. *Editors Rattle Space, The Shock and Vibration Digest.* Vol. 22 (No. 10): 1.

Elseifi, M.A., Gurdal, Z., and, Nikolaidis, E. 1998. Convex and probabilistic models of uncertainties in geometric imperfections of stiffened composite panels. *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit and AIAA/ASME/AHS Adaptive Structures Forum. Part 2 (of 4)* Apr 20-23 1998 Long Beach, CA, USA: 1131-1140.

Ganzerli S. and Burkhart M.F. 2002. Genetic algorithms for optimal structural design using convex models of uncertainties. *Fourth International Conference on Computational Stochastic Mechanics (CSM4), Kerkyra (Corfu), Greece. June 9-12, 2002*.

Ganzerli, S., and Pantelides, C. P. 2000. Optimum structural design via convex model superposition. *J. of Computers and Structures*. Vol. 74 (No. 6): 639-647.

Garey, Michael & Johnson, David S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness,* Oxford, England: W.H. Freeman & Co.

Ghasemi, M.R., Hinton, E., and Wood, R.D. 1999. Optimization of Trusses Using Genetic Algorithms for Discrete and Continuous Variables. *Engineering Computations (Swansea, Wales).* MCB Univ. Press Ltd., Bradford, Engl. Vol. 16 (No. 3): 272-301,

Goldberg, D.E. 1989. *Genetic Algorithms in Search Optimization and Machine Learning.* New York: Addison-Wesley.

Haupt L.H. and Haupt S.E. 1998. *Practical Genetic Algorithms.* New York, NY: John Wiley & Sons, Inc.

Holland, J.H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press.

Horowitz, Ellis & Sahni, Sartaj, 1986. *Fundamentals of Computer Algorithms*, Rockville, MD: Computer Science Press.

Kirsch, U. 1981. *Optimum Structural Design.* McGraw-Hill, Inc., New York, NY.

Korsh, James F. 1986. *Data Structures, Algorithms, and Program Style*, Boston: PWS Computer Science.

Rajeev, S. and Krishnamoorthy C.S. 1997. Genetic algorithms-based methodologies for design optimization of trusses. *ASCE J. of Structural Engineering*. Vol. 123 (No 3): 350-358.

Schemit, L.A., Lai, Ye-Chen 1994. Structural optimization based on preconditioned conjugate gradient analysis methods. *International Journal for Numerical Methods in Engineering* Vol 37 (No 6): p 943-964.

Tonon, F., Bernardini, A., and Elishakoff, I. 2001. Hybrid analysis of uncertainty*: Probability, fuzziness and anti-optimization. Chaos, Solutions and Fractals* Vol. 12 (No. 8): p 1403-1414.

Wang, C.K. 1986. *Structural Analysis on Microcomputers.* New York, NY: Macmillan.