[7] Galileo Galilei. *Dialogues Concerning Two New Sciences*. Evanston, Ill. Northwestern University Press, 1950. Originally published in 1665.

[8] James M. Gere and William Weaver. *Analysis of Framed Structures*. D. Van Nostrand Company, Inc., 1965.

[9] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, Mass. : Addison-Wesley Publishing Co., 1989.

[10] David E. Goldberg and Manohar P. Samtani. Engineering optimization via genetic algorithm. In *Ninth Conference on Electronic Computation*, pages 471–82, New York, N.Y., 1986. ASCE.

[11] W. M. Jenkins. Towards structural optimization via the genetic algorithm. *Computers and Structures*, 40(5):1321–7, 1991.

[12] W. M. Jenkins. Plane frame optimum design environment based on genetic algorithm. *Journal of Structural Engineering*, 118(11):3103–13, November 1992.

[13] K. V. John and C. V. Ramakrishnan. Minimum weight design of trusses using improved move limit method of sequential linear programming. *Computers and Structures*, 27(5):583–91, 1987.

[14] S. F. Jóźwiak. Probability-based optimization of truss structures. *Computers and Structures*, 32(1):87–91, 1989.

[15] Sushil Louis and Gregory Rawlins. Designer genetic algorithms: Genetic algorithms in structure design. In Richard K. Belew and Lashon B. Booker, editors, *Fourth International Conference on Genetic Algorithms*, pages 53–60, University of California, San Diego, July 1991. Morgan Kauffman Publishers.

[16] D. T. Pham and Y. Yang. Optimization of multimodal discrete functions using genetic algorithms. In *Institute of Mechanical Engineers (Part D)*, pages 53–9, 1993.

[17] Kent Porter. Handling huge arrays. *Dr. Dobb's Journal of Software Tools for the Professional Programmer*, 13(3):60–3, March 1988.

[18] David Powell, Michael Skolnick, and S. Tong. Engeneous:domain independent, machine learning for design implementation. In J. David Schaffer, editor, *Third International Conference on Genetic Algorithms*, pages 151–9, George Mason University, June 1989. Morgan Kauffman Publishers.

[19] David Powell, Michael M. Skolnick, and S. Tong. Using genetic algorithms in engineering design optimization via non-linear constraints. In *Fifth International Conference on Genetic Algorithms*, pages 424–31, University of Illinois at Urbana-Champaign, July 1993. Morgan Kauffman Publishers.

[20] S. Rajeev and C. S. Krishnamoorthy. Discrete optimization of structures using genetic algorithms. *Journal of Structural Engineering*, 118(5):1233–50, May 1992.

[21] S. Rao, K. Sundararaju, B. Prakash, and C. Balakrishna. Multiobjective fuzzy optimization techniques for engineering design. *Computers and Structures*, 42(1):37–44, 1992.

[22] M. P. Saka and M. Ulker. Optimum design of geometrically nonlinear space trusses. *Computers and Structures*, 42(3):289–99, 1992.

[23] Marc Schoenauer and Spyros Xanthakis. Constrained ga optimization. In *Fifth International Conference on Genetic Algorithms*, pages 573–80, University of Illinois at Urbana-Champaign, July 1993. Morgan Kauffman Publishers.

[24] A. B. Templeman and D. F. Yates. A linear programming approach to discrete optimum design of trusses. In H. Eschenauer and N. Olholff, editors, *Optimization Methods in Structural Design*, pages 133–9. BI Wissenschaftsverlag, Mannheim, Germany, 1983.

[25] Andrew B. Templeman. Discrete optimum structural design. *Computers and Structures*, 30(2):511–8, 1988.

[26] A. R. Toakley. Optimum design using available sections. *Journal of the Structural Division. ASCE*, 94(ST 5):1219–33, 1968.

[27] Garret N. Vanderplaats. *Numerical Optimization Techniques for Engineering Design with Applications*. McGraw-Hill Book Company, 1984.

[28] Ming Zhou and Renwei Xia. An efficient method of truss design for optimum geometry. *Computers and Structures*, 35(2):115–9, 1990.

[29] D. M. Zhu. An improved templeman's algorithm for optimum design of trusses with discrete member sizes. *Engrg. Opt.*, 9:303–12, 1986.

Method	Weight	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
Rajeev	5613.84	33.50	1.62	22.00	15.50	1.62	1.62	14.20	19.90	19.90	2.62
CONMIN	5563.00	25.20	1.89	24.87	15.83	0.10	1.75	16.76	19.73	20.98	2.51
OPTDYN	5472.00	25.70	0.10	25.11	19.39	0.10	0.10	15.40	20.32	20.74	1.14
LINRM	6249.00	21.57	10.98	22.08	14.95	0.10	10.98	18.91	18.42	18.40	13.51
SUMT	5932.00	30.69	2.37	31.62	11.66	0.10	3.71	21.71	20.90	13.97	3.26
M-3	5719.00	25.84	3.07	26.42	12.77	0.10	3.43	19.34	19.17	18.76	4.42
M-5	5725.00	25.83	2.88	26.45	12.75	0.10	3.77	19.37	19.18	18.77	4.38
GRP-UI	5727.00	24.78	4.17	24.78	14.45	0.10	4.17	17.46	19.26	19.27	5.26
GENETIC	5586.59	30.00	1.62	22.90	13.50	1.62	1.62	13.90	22.00	22.00	1.62

Table 4: Comparison of our results (GENETIC) with other methods reported in the literature for the plane truss. For more detailed information about each one of these methods, see [2] and [20].

Method	Weight	A1	A2	A3	A4	A5	A6	A7	A8
Zhu	562.93	0.100	1.900	2.600	0.100	0.100	0.800	2.100	2.600
Rizz	545.16	0.010	1.988	2.991	0.010	0.010	0.684	1.676	2.662
Schmit	545.22	0.010	1.964	3.033	0.010	0.010	0.670	1.680	2.670
Rajeev	546.01	0.100	1.800	2.300	0.200	0.100	0.800	1.800	3.000
GENETIC	493.94	0.100	0.700	3.200	0.100	1.400	1.100	0.500	3.400

Table 5: Comparison of our results (GENETIC) with other methods reported in the literature for the space truss. Notice that Rizz and Schmit are continuous methods. For more detailed information about each method, refer to [29] and [20].

analysis programs to C in order to have more transportability (a C version of the SGA is already available). Until now, all our work has been done on IBM PCs. We wish to make our code generally available and are working to do so. Our code does not require an extremely fast computer, although a mathematical coprocessor is helpful.

Our technique could be extended to other engineering areas that present equally complex problems. For example, hydraulics presents many optimization problems requiring tedious and complex calculations.

6 Conclusions

GAs seem to be a good choice for discrete structural optimization. They offer several advantages that other techniques lack such as generality and the ability to deal directly with discrete search spaces. GAs performed well on a plane truss and 3-D truss problem as compared to more traditional techniques. We do not claim that GAs are the golden key to structural optimization that engineers have sought for nearly two centuries. There is no doubt, however, that GAs have potential for automating discrete structural optimization.

References

[1] Jasbir S. Arora. *Introduction to Optimum Design.* McGraw-Hill Book Company, 1989.

[2] Ashok Dhondu Belegundu. *A Study of Mathematical Programming Methods for Structural Optimization.* PhD thesis, University of Iowa, Dept. of Civil and Environmental Engineering, 1982.

[3] Andrzej Marek Brandt, Wojciech Dzieniszewski, Stefan Jendo, Wojciech Marks, Stefan Owczarek, and Zbigniew Wasiutynski. *Criteria and Methods of Structural Optimization.* M. Nijhoff Publishers, 1986.

[4] Bill P. Buckles and Frederick E. Petry. *Genetic Algorithms.* Technology Series. IEEE Computer Society Press, 1992.

[5] Carlos A. Coello. Análisis de estructuras reticulares por computadora (método de rigideces). Tesis de Licenciatura, 1991. (in Spanish).

[6] Kalyanmoy Deb. Optimal design of a welded beam via genetic algorithms. *AIAA Journal*, 29:2013–15, November 1991.

set of all possible cross-sectional areas (the S list) is fed into the program. Then, the constraints on maximum allowable stress and maximum deflection are provided. The program that does the structural analysis is executed separately to generate the stiffness matrix and its results are stored in a separate file. The GA is then executed and the user provides the size of population, number of generations, etc. After that, the program starts iterating, reading and rewriting the file that contains the results of the analysis, modifying it with values from the chromosomes at each generation. As progress is made, a simplified report is sent to the output device (a laser printer in our case) showing the current generation and the best solution found so far.

4 Comparison of Results

Tables 4 and 5 compare our results to techniques reported in the literature. The GA performed well, surpassing all but two of the other methods for the plane truss and surpassing all the other methods for the space truss.

Though the space truss has a smaller intrinsic search space its analysis takes more CPU time. The reason is that a space truss has more degrees of freedom—because there are more unknown forces acting on each of its nodes. Since we have to solve the structure for each chromosome in order to get its fitness, fitness evaluation has the potential to take too much time if the structure is sufficiently large, even though its search space could be relatively small.

It should be pointed out that Rajeev and Krishnamoorthy [20] use a genetic algorithm to optimize the two trusses presented in this work, but they make assumptions that considerably reduce the size of the search space based on imposing a lower and upper bound on each member's values. They assume that one design variable can take only 16 possible values. This assumption reduces the chromosome length to only 4 bits, and therefore the size of the search space gets small enough to find reasonable solutions using populations of 20, 30, and 40 individuals, running through no more than 20 generations. However, even without making those assumptions (i.e., dealing with a larger search space), we find superior results, though we use bigger population sizes and a larger number of generations.

Rajeev and Krishnamoorthy [20] proposed an interesting fitness function

$$F(x) = [\phi(x)_{max} + \phi(x)_{min}] - \phi(x) \qquad (3)$$

where $F(x)$ is the fitness of candidate solution x, $\phi(x)_{max}$ and $\phi(x)_{min}$ are respectively the maximum and minimum $\phi(x)$ over the population, $\phi(x)$ is

$$\phi(x) = f(x)(1 + KC) \qquad (4)$$

$$C = \sum_{j=1}^{m} c_j \qquad (5)$$

and m is the number of constraints, the c_j are the amounts by which each constraint is violated, $f(x)$ is the weight function (see Equation 1) and K weights the constraint violations. They found a value of K=10 suitable for the two problems addressed here.

They used $F(x)$ following Goldberg's suggestion of subtracting $\phi(x)$ from a large constant for minimization problems, so that all the fitness values are positive and individuals get fitness values according to their actual merit.

Following the same track, Jenkins [11] uses the mass of the structure as his objective function. Then, in order to present the problem in minimization form, he writes the objective function as

$$F(x) = F_{max} - f(x) \qquad (6)$$

where F_{max} is the mass of the heaviest possible structure. We took a simpler approach by considering just the inverse of the weight as our objective function, because we want to *minimize* the weight of the structure. This reduces the amount of computing and, as can be seen from the results, seems to work fine. In fact, even if the constant v is not used (see Equation 2) and we just drop the fitness value to zero when a constraint is violated, we still get similar results, though sometimes larger populations are required, to avoid an all-zero fitness initial population.

5 Future Work

Our goal is to develop an automated structural design system that uses GAs. The genetic algorithm approach is so general that the code developed here can be used with only minor modification to optimize the remaining framed structures (plane and space frames, plane grids and beams). We are trying to find a more realistic fitness function that doesn't require too much bookkeeping. In terms of implementation, work continues to overcome the memory and execution time constraints of personal computers. The programs used are far from optimal—much work remains to be done in terms of fine tuning the code. However, at the moment, the main concern is to translate the structural

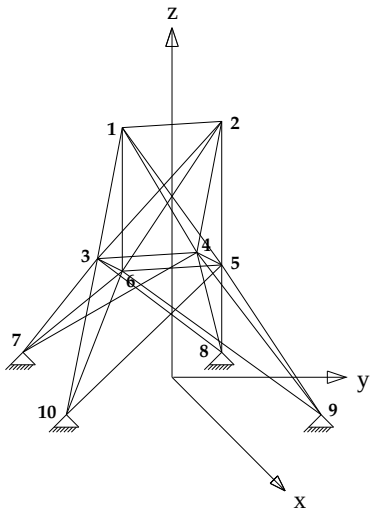

Figure 3: 25-bar space truss used for example No. 2.

Node	Fx (lbs)	Fy (lbs)	Fz (lbs)
1	1000	-10000	-10000
2	0	-10000	-10000
3	500	0	0
6	600	0	0

Table 1: Loading conditions for the 25-bar space truss shown in Figure 3.

> A space truss is similar to a plane truss except that the members may have any directions in space. The forces acting on a space truss may be in arbitrary directions, but any couple acting on a member must have its moment vector perpendicular to the axis of the member. The reason for this requirement is that a truss member is incapable of supporting a twisting moment.

Loading conditions are given in Table 1, member groupings are given in Table 2, and node coordinates are given in Table 3. The assumed data are: modulus of elasticity, $E = 1 \times 10^4$ ksi (6.89×10^4 MPa), $\rho = 0.10$ lb/in^3 (2,770 kg/m^3); $\sigma_a = \pm 40,000$ psi, $u_a = \pm 0.35$ in. The set of areas available for this truss is [20] S={0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 3.0, 3.2, 3.4} (in^2).

Group Number	Members
1	1-2
2	1-4, 2-3, 1-5, 2-6
3	2-5, 2-4, 1-3, 1-6
4	3-6, 4-5
5	3-4, 5-6
6	3-10, 6-7, 4-9, 5-8
7	3-8, 4-7, 6-9, 5-10
8	3-7, 4-8, 5-9, 6-10

Table 2: Group membership for the 25-bar space truss shown in Figure 3.

Node	X	Y	Z
1	-37.50	0.00	200.00
2	37.50	0.00	200.00
3	-37.50	37.50	100.00
4	37.50	37.50	100.00
5	37.50	-37.50	100.00
6	-37.50	-37.50	100.00
7	-100.00	100.00	0.00
8	100.00	100.00	0.00
9	100.00	-100.00	0.00
10	-100.00	-100.00	0.00

Table 3: Coordinates of the joints of the 25-bar space truss shown in Figure 3.

3 Implementation Details

We use a customized version of the SGA (Simple Genetic Algorithm) presented in Goldberg [9]. Dynamic memory management [17] is used instead of static arrays. Also, binary tournament selection is used instead of the roulette wheel selection. Finally, several of the original functions and procedures are rewritten.

The user provides a maximum number of generations as the stopping criteria. The crossover probability is approximately 0.80, and the mutation probability is 0.01. We don't deactivate the mutation rate as Rajeev and Krishnamoorthy [20] propose. To analyze the structures we used programs from [5].

The basic operation of the program is simple. The

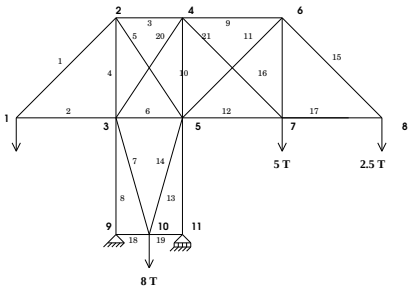

Figure 1: 10-bar plane truss used for example No. 1.

ure 1. Gere and Weaver [8] define a *plane truss* as follows:

> A plane truss is idealized as a system of members lying in a plane and interconnected at hinged joints. All applied forces are assumed to act in the plane of the structure, and all external couples have their moment vectors normal to the plane. The loads may consist of concentrated forces applied at the joints, as well as loads that act on the members themselves. For purposes of analysis, the latter loads may be replaced by statically equivalent loads acting at the joints. Then the analysis of a truss subjected only to joint loads will result in axial forces of tension and compression in the members. In addition to these axial forces, there will be bending moments and shear forces in those members having loads that act directly upon them. The determination of all such stress resultants constitutes the complete analysis of the forces in the members of a truss.

The objective of the problem is to minimize the weight of the structure, $f(x)$,

$$f(x) = \sum_{j=1}^{10} \rho A_j L_j \qquad (1)$$

where x is the candidate solution, A_j is the cross-sectional area of the jth member, L_j is the length of the jth member, and ρ is the weight density of the material. The assumed data are: modulus of elasticity, $E = 1 \times 10^4$ ksi (6.89×10^4 MPa), $\rho = 0.10$ lb/in^3 ($2,770$ kg/m^3), and vertically downward loads of 100 kips (445.374 kN) at nodes 2 and 4. Additionally, the truss is subject to the following set of constraints

$$\sigma_j \leq \sigma_a, \quad \text{for j = 1 to 10}$$
$$u_j \leq u_a$$

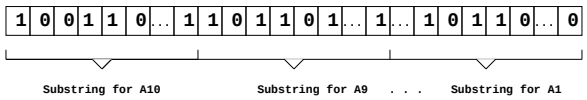

Figure 2: Binary representation of a chromosome.

where σ_j is the stress in member j, σ_a is the maximum allowable stress for all members, u_j is the displacement of each node (horizontal and vertical), and u_a is the maximum allowable displacement for all nodes.

The constraints can be expressed in normalized form as

$$\frac{\sigma_j}{\sigma_a} - 1 \leq 0; \quad \frac{u_j}{u_a} - 1 \leq 0$$

The fitness function used was

$$F(x) = 1/(f(x)[1000v + 1]) \qquad (2)$$

where v is the count of the number of constraints violated by a given solution. When there is no violation to the constraints, the fitness is simply the inverse of the weight. As constraints are violated, the fitness is lowered correspondingly. The constant 1000 was determined experimentally.

The constraints for this problem are as follows. The maximum displacement is 2 inches (50.8 mm) and the stresses are limited to ± 25 ksi (172.25 MPa). The list of discrete values, taken from the American Institute of Steel Construction Manual [1], is S={1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 13.9, 14.2, 15.5, 16.0, 16.9, 18.8, 19.9, 22.0, 22.9, 26.5, 30.0, 33.5} (in^2).

Since there are 10 design variables, and each can take any of the 42 available sections, the intrinsic size of the search space is 42^{10} ($\cong 10^{16}$). A 6 bit GA representation is used and the extra codes are assigned to random values taken from S. Thus each chromosome is 60 bits long (6 bits/truss x 10 trusses) as shown in Figure 2. The representation can easily be adapted when the number of bars in the truss changes. It also has the advantage of using relatively small amounts of computer memory, even for large trusses.

Optimization of a Space Truss

Our second example is the 25-bar space truss taken from Rajeev and Krishnamoorthy [20], shown in Figure 3. A *space truss* is defined in Gere and Weaver [8] as follows:

- GAs operate on multiple partial solutions simultaneously (sometimes called implicit parallelism), gathering information from a population of search points to direct subsequent search efforts. Their ability to maintain multiple partial solutions concurrently helps make GAs less susceptible to the problems of local maxima and noise.

These characteristics make GAs a good choice for structural optimization.

1.1 Previous Work

Goldberg and Samtani [10] appear to have first suggested the use of GAs for structural optimization. They considered the use of a GA to optimize a 10-bar plane truss. A few others have applied the technique to the design of welded beams [6], plane frames [12], a trussed-beam roof structure and a thin-walled cross-section [11], and generalized trusses [20].

Pham and Yang [16] presented interesting work on the optimization of multi-modal discrete functions using GAs. Powell, et al. [18] described a domain independent design optimization tool for engineers involved with iterative design called EnGENEous. This program uses expert systems and genetic algorithms to move from a domain independent system with no knowledge to a domain dependent system with knowledge. It has been used in the design of cooling fans, molecular electronic structure, and aircraft engine turbines, with the authors claiming an increase in engineer productivity by a factor of 10. Powell, et al. [19] showed the results of examining 10 engineering design optimization problems with regard to the comparative performance of GAs and numerical techniques.

Schoenauer and Xanthakis [23] presented a general method of handling constraints in genetic optimization, based on the Behavioural Memory paradigm. Instead of requiring the problem-dependent design of either repair operators (projection onto the feasible region) or penalty functions (weighted sum of constraint violations and the objective function), they sampled the feasible region by evolving from an initial random population successively applying a series of different fitness functions which embody constraint satisfaction. Only in the final step was the optimization restricted to the feasible region. The success of the whole process is highly dependent on the genetic diversity maintained during the first steps, ensuring a uniform sampling of the feasible region. They applied this scheme to test problems of truss structure optimization: a 10-bar (2D) and a 25-bar (3D) truss.

Louis and Rawlins [15] discussed the application of GAs to design structures, focusing on combinatorial circuit design problems—given a set of logic gates, design a circuit that performs a specified function.

There is a large amount of research in the area of structural optimization. Scientists have used many techniques, from gradient methods to sophisticated variations of Newton's method. Arora [1], Brandt, et al. [3], Vanderplaats [27] and Belegundu [2] presented good reviews of these methods. Also, Templeman [25] offers a review of several methods for the optimum design of trusses, emphasizing the problems of using them in a computer-aided design context.

Much recent work has been done on optimization of discrete structural systems. Toakley [26] proposed a model to optimize statically determinate trusses using a list of sizes of their members. Templeman and Yates [24] proposed a method to optimize statically indeterminate trusses that considers stress and displacement constraints. Zhu [29] introduced a modification to this method that considerably improves its performance. John and Ramakrishnan [13] proposed the use of sequential linear programming with a branch and bound algorithm for discrete structural optimization. Saka and Ulker [22] presented an algorithm for structural optimization of statically indeterminate space trusses subject to stresses, displacements, and cross-sectional constraints. Their algorithm is based on the use of an iterative linear analysis of the structure. Rao, et al. [21] considered the use of multiobjective fuzzy optimization techniques for engineering design and presented examples of plane and space trusses. Zhou and Xia [28] proposed a method for the configurational optimization of a truss subject to displacement, stress and buckling constraints under multiple load conditions. Their method generates a sequence of approximate convex problems which are solved by a dual method of convex programming. Finally, Jóźwiak [14] presented a probability-based approach in which the mean value of the structural mass is taken as the objective function.

2 Examples

We present two examples of the application of our method in this section. We concentrate on defining the problem and the inputs to the GA. We present our obtained results in a later section of the paper.

Optimization of a Plane Truss

Our first example is the 10-bar plane truss taken from Rajeev and Krishnamoorthy [20], shown in Fig-

Using Genetic Algorithms for Optimal Design of Trusses

Carlos A. Coello Coello Michael Rudnick Alan D. Christiansen

Department of Computer Science
Tulane University
New Orleans, LA 70118

Abstract

This paper presents a method for optimizing the design of plane and space trusses subject to a specified set of constraints. Our method is based upon a search technique using genetic algorithms. Traditional structural optimization techniques consider a continuous search space, and consequently lead to unrealistic solutions because structural members are not available in continuously varying sizes. A practical method should consider only the discrete values associated with commonly available materials. On the other hand, most modern structural optimization techniques, even when they consider a discrete search space, suffer a lack of generality, and tend to be limited to a certain kind of structure. Genetic algorithms remedy these two problems since they can deal with discrete search spaces and they are general enough to be easily extended to any kind of structure without substantial modifications. Our results show the genetic algorithm can provide very good solutions, often surpassing other complex and specialized techniques.

1 Introduction

Galileo appears to be the first scientist who studied the optimization of structures, as we can see in his work on the bending strength of beams [7]. Bernoulli, Lagrange, and Navier are just a few of the other great scientists who sought the "best" shapes for structural elements to satisfy the given strength requirements. As time passed, this discipline evolved and became an engineering area known as *Structural Optimization*, which seeks to determine the most economical geometrical shapes satisfying the constraints (e.g. stresses and deflections) imposed on the design.

Traditionally, the design of a certain structure has depended on the experience of an engineer. Consequently, designed structures have often been suboptimal. More recently, engineers have started using computers to automate structural design. However, their use has concentrated mainly in iterative design. In other words, engineers have been doing their same work, but now a lot faster and more accurately.

To what extent can computers assist engineers in a more sophisticated manner? Several researchers are developing techniques that may eventually lead to the fully automated design of structures. However, most of these techniques share some problems: due to their mathematical origin (most of them are linear programming techniques) they tend to treat structural optimization as a problem in which the search space is continuous, when it's really discrete. Only a small number of structural shapes are available in the market.

We focus on the use of genetic algorithms to optimize the design of plane and space trusses. The technique considers a discrete search space, yielding more realistic results than linear programming methods. Though some structural optimization techniques can deal with discrete search spaces, they suffer an inherent lack of generality and therefore can't be readily extended to other kinds of structures.

The genetic algorithm (GA), for its part, is problem independent. The code developed for this work can be reused to solve the remaining framed structures (plane and space frames, plane grids and beams) with little change. Finally, the GA has performance comparable to existing techniques, sometimes even surpassing them.

GAs differ from traditional search techniques in several ways [4]:

- GAs don't require problem specific knowledge to carry out a search.
- GAs use stochastic instead of deterministic operators and appear to be robust in noisy environments.