INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2004; **59**:703–736 (DOI: 10.1002/nme.899)

An adaptive penalty scheme for genetic algorithms in structural optimization

Afonso C. C. Lemonge[1] and Helio J. C. Barbosa[2,*,†]

[1]*Departamento de Estruturas, Faculdade de Engenharia, Universidade Federal de Juiz de Fora, Juiz de Fora MG 36036 330, Brazil*
[2]*LNCC/MCT Rua Getúlio Vargas 333, Petrópolis RJ 25651 070, Brazil*

SUMMARY

A parameter-less adaptive penalty scheme for genetic algorithms applied to constrained optimization problems is proposed. Using feedback from the evolutionary process the procedure automatically defines a penalty parameter for each constraint. The user is thus relieved from the burden of having to determine sensitive parameter(s) when dealing with every new constrained optimization problem. The procedure is shown to be effective and robust when applied to test problems from the evolutionary computation literature as well as several optimization problems from the structural engineering literature. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: penalty methods; genetic algorithms; structural optimization

1. INTRODUCTION

Genetic algorithms (GAs) [1, 2], inspired by the Darwinian principles of evolution, are one of the most recent tools in structural optimization. From the early paper by Goldberg and Samtani [3] in 1986, where the classic ten-bar truss is minimized for weight, to the present day, several applications of GAs in structural optimization have appeared in the literature.

In solving a given problem, GAs encode all the variables corresponding to a candidate solution (the phenotype) in a data structure (the genotype) and maintain a population of genotypes which is evolved mimicking Nature's evolutionary process: solutions are selected—by a stochastic process that favours better solutions—and have their genetic material recombined /mutated by means of genetic operators. This gives rise to a new population

*Correspondence to: Helio J. C. Barbosa, LNCC/MCT Rua Getúlio Vargas 333, Petrópolis RJ 25651 070, Brazil.
†E-mail: hcbm@lncc.br

Contract/grant sponsor: CNPq; contract/grant number: 301233/86-1 and 475398/2001-7
Contract/grant sponsor: FAPEMIG; contract/grant number: TEC-692/99

Received 27 November 2002
Revised 24 January 2003
Accepted 30 May 2003

Copyright © 2003 John Wiley & Sons, Ltd.

containing improved solutions (with higher fitness). The process starts from an initial population and is repeated for a given number of generations or until some stopping criteria are met.

GAs are weak search algorithms which can be directly applied to unconstrained optimization problems, UOP(f, S), where one seeks for an element x belonging to the search space S, which minimizes (or maximizes) the real-valued objective function f. In this case, the GA usually employs a fitness function closely related to f. However, structural optimization problems involve several types of constraints and one is led to a constrained optimization problem, COP(f, S, b), where an $x \in S$ that minimizes (or maximizes) f in S is sought with the additional requirement that the Boolean function $b(x)$ (which includes all constraints in the problem) is evaluated as true. Elements in S satisfying all constraints are called feasible.

The straightforward application of GAs to COPs is not possible due to the additional requirement that a set of constraints must be satisfied. Several difficulties may arise:

- the objective function may be undefined for some or all infeasible elements,
- the check for feasibility can be more expensive than the computation of the objective function itself,
- an informative measure of the degree of infeasibility of a given candidate solution is not easily defined.

It is easy to see that even if both the objective function $f(x)$ and a measure of constraint violation $v(x)$ are defined for all $x \in S$ it is not possible to know in general which of two given infeasible solutions is closer to the optimum and thus should be operated upon or kept in the population. For instance, in a minimization problem, one can have $f(x_1) > f(x_2)$ and $v(x_1) = v(x_2)$ or a situation where $f(x_1) = f(x_2)$ and $v(x_1) > v(x_2)$ and still have x_1 closer to the optimum. As a result, several techniques have been proposed and compared in the literature in order to enable a GA to tackle COPs. However, it is important to note that most comparisons in the literature have been conducted in problems with constraints which can be written as $g_i(x) \geqslant 0$, where each $g_i(x)$ is a given *explicit* function of the independent (design) variable $x \in \mathbb{R}^n$. In fact, that has to do more with easier reproducibility of results than with practical significance. Although the available test problems attempt to represent different types of difficulties one is expected to encounter when dealing with practical situations, very often the constraints cannot be put explicitly in the form $g_i(x) \leqslant 0$. In engineering design problems most constraints are only known as complex implicit functions of the design variables. In structural optimization, in order to check if a certain constraint (the stress level at a particular component, for example) has been violated one is required to perform a whole computational simulation expending considerable computational resources.

The techniques for handling constraints within GAs can be classified either as *direct* (feasible or interior), when only feasible elements in S are considered, or as *indirect* (exterior), when both feasible and infeasible elements are used during the search process.

Direct techniques comprise: (a) the design of special *closed* genetic operators, (b) the use of special decoders, (c) repair techniques and (d) 'death penalty'.

In special situations, closed genetic operators (in the sense that when applied to feasible parents they produce feasible offspring) can be designed if enough domain knowledge is available [4]. Special decoders [5]—that always generate feasible individuals from any given genotype—have been devised, but no applications considering implicit constraints have been published.

Copyright © 2003 John Wiley & Sons, Ltd.

Repair methods [6, 7] use domain knowledge in order to move an infeasible offspring into the feasible set. However, there are situations when it is very expensive, or even impossible, to construct such a repair operator, drastically reducing the range of applicability of repair methods. The design of efficient repair methods constitutes a formidable challenge, specially when implicit constraints are present.

Discarding any infeasible element generated during the search process ('death penalty') is common practice in non-populational optimization methods. Although problem independent, no consideration is made for the potential information content of any infeasible individual.

Summarizing, direct techniques are problem dependent (with the exception of the 'death penalty') and actually of extremely reduced practical applicability.

Indirect techniques comprise: (a) the use of Lagrange multipliers [8], which may also lead to the introduction of a 'population of multipliers' and to the use of the concept of coevolution [9], (b) the use of fitness as well as constraint violation values in a multi-objective optimization setting [10], (c) the use of special selection techniques [11] and (d) 'lethalization': any infeasible offspring is just assigned a given, very low, fitness value [12].

For other methods proposed in the evolutionary computation literature see References [4, 13–16] and references therein.

Methods to tackle COPs which require the knowledge of constraints in explicit form have thus limited practical applicability. This fact, together with simplicity of implementation are perhaps the main reasons why penalty techniques, in spite of their shortcomings, are the most popular ones.

Penalty techniques, originating in the mathematical programming community, range from simple schemes (like 'lethalization') to penalty schemes involving from one to several parameters. Those parameters can remain constant (the most common case) or be dynamically varied along the evolutionary process according to an exogenous schedule or an adaptive procedure. Penalty methods, although quite general, require considerable domain knowledge and experimentation in each particular application in order to be effective.

In Reference [17] the authors developed a general penalty method for GAs which:

- handles inequality as well as equality constraints,
- does not require the knowledge of the explicit form of the constraints as a function of the decision/design variables,
- is free of parameters to be set by the user,
- can be easily implemented within an existing GA code and
- is robust.

In the present paper, the application of such technique to structural optimization problems is made through a very simple GA with

- binary coding,
- standard crossover and mutation operators,
- no GA parameter tuning and
- no use of the explicit form of the constraints in the construction of operators to use such knowledge.

It must be noted that further efficiency can be obtained by adding domain knowledge in the form of

- a better coding, including the use of real variables,

Copyright © 2003 John Wiley & Sons, Ltd.

- special operators,
- special decoders and
- repair operators.

In the proposed technique, in contrast with previous approaches where a single penalty parameter is used for all constraints in a given problem, an adaptive scheme automatically sizes the penalty parameter corresponding to *each* constraint along the evolutionary process.

In the next section, the penalty method and some of its implementations within genetic algorithms are presented. In Section 3 the proposed adaptive scheme is discussed, Section 4 presents numerical experiments with several test-problems from the literature and the paper closes with some conclusions.

2. PENALTY METHODS

A standard constrained optimization problem in R^n can be thought of as the minimization of a given objective function $f(x)$, where $x \in R^n$ is the vector of design/decision variables, subject to inequality constraints $g_p(x) \geqslant 0, \quad p = 1, 2, \ldots, \bar{p}$ as well as equality constraints $h_q(x) = 0, \ q = 1, 2, \ldots, \bar{q}$. Additionally, the variables may be subject to bounds $x_i^L \leqslant x_i \leqslant x_i^U$ but this type of constraint is trivially enforced in a GA and need not be considered here.

Penalty techniques can be classified as *multiplicative* or *additive*. In the multiplicative case, a positive penalty factor $p(v(x), T)$ is introduced in order to amplify the value of the fitness function of an infeasible individual in a minimization problem. One would have $p(v(x), T) = 1$ for a feasible candidate solution x and $p(v(x), T) > 1$ otherwise. Also, $p(v(x), T)$ increases with the 'temperature' T and with constraint violation. An initial value for the temperature is required as well as the definition of a function such that T grows with the generation number. This type of penalty has received much less attention in the evolutionary computation community than the additive type. In the additive case, a penalty functional is added to the objective function in order to define the fitness value of an infeasible element. They can be further divided into: (a) *interior* techniques, when a barrier functional $B(x)$—which grows rapidly as x approaches the boundary of the feasible domain—is added to the objective function

$$F(x) = f(x) + \frac{1}{k} B(x)$$

and (b) *exterior* techniques, where a penalty functional is introduced:

$$F(x) = f(x) + kP(x) \tag{1}$$

such that $P(x) = 0$ if x is feasible and $P(x) > 0$ otherwise (for minimization problems). In both cases (a) and (b), as $k \to \infty$, the sequence of minimizers of the UOP(F, S) converges to the solution of the COP(f, S, b).

At this point it is useful to define the amount of violation of the jth constraint by the candidate solution $x \in R^n$ as

$$v_j(x) = \begin{cases} |h_j(x)| & \text{for an equality constraint} \\ \max\{0, -g_j(x)\} & \text{otherwise} \end{cases}$$

Copyright © 2003 John Wiley & Sons, Ltd.

It is also common to design penalty functions that grow with the vector of violations $v(x) \in R^m$ where $m = \bar{p}+\bar{q}$ is the number of constraints to be penalized. The most popular penalty function is given by

$$P(x) = k \sum_{j=1}^{m} (v_j(x))^{\beta} \tag{2}$$

where k is the penalty parameter and $\beta = 2$. Although it is easy to obtain the unconstrained problem, the definition of the penalty parameter k is usually a time-consuming trial-and-error process.

Powell and Skolnick [18] proposed a method of superiority of feasible points where each candidate solution is evaluated by the following expression:

$$F(x) = f(x) + r \sum_{j=1}^{m} v_j(x) + \theta(t, x)$$

where r is a constant. The main assumption is that any feasible solution is better than any infeasible solution. This assumption is enforced by a convenient definition of the function $\theta(t, x)$. This scheme can be modified by the introduction of the tournament selection proposed by Deb [19] coupled with the fitness function:

$$F(x) = \begin{cases} f(x) & \text{if } x \text{ is feasible} \\ f_{\max} + \sum_{j=1}^{m} v_j(x) & \text{otherwise} \end{cases}$$

where $f_{\max}$ is the function value of the worst feasible solution. As observed in the previous conditions, if a solution is infeasible the value of its objective function is not considered to determine the fitness function and this value is replaced by $f_{\max}$. However, Deb's [19] constraint handling scheme—which also uses niching and a controlled mutation—works very well for his real-coded GA, but not so well for his binary-coded GA.

Among the many suggestions in the literature [13, 20, 21] some of them—more closely related to the work presented here—will be briefly discussed in the following.

2.1. Some methods in the literature

Besides the widely used case of a single constant penalty parameter k, several other proposals can be found in the literature.

2.1.1. Two-level penalties. Le Riche *et al.* [22] present a GA where two fixed penalty parameters k_1 and k_2 are used independently in two different populations. The idea is to create two sets of candidate solutions where one of them is evaluated with the parameter k_1 and the other with the parameter k_2. With $k_1 \gg k_2$ there are two different levels of penalization and there is a higher chance of maintaining feasible as well as infeasible individuals in the population and to get offspring near the boundary between the feasible and infeasible regions. The strategy can be summarized as: (i) create $2\times$pop individuals randomly (pop is the population size); (ii) evaluate each individual considering each penalty parameter and create two ranks; (iii) combine

Copyright © 2003 John Wiley & Sons, Ltd.

the two ranks in a single one with size pop; (iv) apply genetic operators; and (v) evaluate new offspring (2×pop), using both penalty parameters and repeat the process.

2.1.2. Multiple coefficients. Homaifar *et al.* [23] proposed different penalty coefficients for different levels of violation of each constraint. The fitness function is written as

$$F(x) = f(x) + \sum_{j=1}^{m} k_{ij}(v_j(x))^2$$

where i denotes one of the l levels of violation defined for the jth constraint. This is an attractive strategy because, at least in principle, it allows for a good control of the penalization process. The weakness of this method is the large number, $m(2l+1)$, of parameters that must be set by the user for each problem.

2.1.3. Dynamic coefficients. Joines and Houck [24] proposed that the penalty parameters should vary dynamically along the search according to an exogenous schedule. The fitness function $F(x)$ was written as in (1) and (2) with the penalty parameter, given by $k = (C \times t)^{\alpha}$, increasing with the generation number t. They used both $\beta = 1$ and 2 and suggested the values $C = 0.5$ and $\alpha = 2$.

2.1.4. Adaptive penalties. A procedure where the penalty parameters change according to information gathered during the evolution process was proposed by Bean and Hadj-Alouane [25]. The fitness function is again given by (1) and (2) but with the penalty parameter $k = \lambda(t)$ adapted at each generation by the following rules:

$$\lambda(t+1) = \begin{cases} (\frac{1}{\beta_1})\lambda(t) & \text{if } b^i \in \mathscr{F} \text{ for all } t-g+1 \leqslant i \leqslant t \\ \beta_2\lambda(t) & \text{if } b^i \notin \mathscr{F} \text{ for all } t-g+1 \leqslant i \leqslant t \\ \lambda(t) & \text{otherwise} \end{cases}$$

where b^i is the best element at generation i, $\mathscr{F}$ is the feasible region, $\beta_1 \neq \beta_2$ and $\beta_1, \beta_2 > 1$. In this method the penalty parameter of the next generation $\lambda(t+1)$ decreases when all best elements in the last g generations were feasible, increases if all best elements were infeasible and otherwise remains without change.

The method proposed by Coit *et al.* [26], uses the fitness function:

$$F(x) = f(x) + (F_{\text{feas}}(t) - F_{\text{all}}(t)) \sum_{j=1}^{m} (v_j(x)/v_j(t))^{\alpha}$$

where $F_{\text{all}}(t)$ corresponds to the best solution, until the generation t (without penalty), F_{feas} corresponds to the best feasible solution and α is a constant.

Schoenauer and Xanthakis [27] presented a strategy that handles constrained problems in stages: (i) initially, a randomly generated population is evolved considering only the first constraint until a certain percentage of the population is feasible with respect to that constraint; (ii) the final population of the first stage of the process is used in order to optimize with respect to the second constraint. During this stage, the elements that had violated the previous constraint are removed from the population, (iii) the process is repeated until all the constraints

Copyright © 2003 John Wiley & Sons, Ltd.

are processed. This strategy becomes less attractive as the number of constraints grows and is potentially dependent on the order in which the constraints are processed.

Hamida and Schoenauer [28] proposed an adaptive scheme using: (i) a function of the proportion of feasible individuals in the population, (ii) a seduction/selection strategy to mate feasible and infeasible individuals, and (iii) a selection scheme to give advantage for a given number of feasible individuals.

2.1.5. Other techniques. Runarsson and Yao [11] presented a novel approach where a good balance between the objective and the penalty function values is sought stochastically by a stochastic ranking. Using real-coding in an evolution strategy setting produced very good results.

Recently, Wright and Farmani [29] proposed a method that requires no parameters and represents the constraint violation by a single infeasibility measure.

Finally, inspired by the multiobjective optimization technique known as NPGA [30], Coello [31] proposed a dominance-based selection scheme to incorporate constraints into the fitness function of a GA. However, the resulting method hinges on the choice for its selection rate parameter S_r which is responsible for maintaining the required population diversity.

3. THE PROPOSED METHOD

In Reference [17] the authors introduced a method without any type of user-defined penalty parameter. An adaptive scheme was developed which uses information from the population such as the average of the objective function and the level of violation of each constraint during the evolution.

The fitness function proposed is written as

$$F(x) = \begin{cases} f(x) & \text{if } x \text{ is feasible} \\ \bar{f}(x) + \sum_{j=1}^{m} k_j v_j(x) & \text{otherwise} \end{cases}$$

where

$$\bar{f}(x) = \begin{cases} f(x) & \text{if } f(x) > \langle f(x) \rangle \\ \langle f(x) \rangle & \text{otherwise} \end{cases} \tag{3}$$

and $\langle f(x) \rangle$ is the average of the objective function values in the current population. In Figure 1 feasible as well as infeasible solutions are shown. Among the six infeasible solutions, the individuals #3, #4, #5 and #6 have their objective function values (represented by opened circles), less than the average objective function and, according to the proposed method, have $\bar{f}$ given by $\langle f(x) \rangle$. The solutions #1 and #2 have objective function values which are worst than the population average and thus have $\bar{f}(x) = f(x)$.

The penalty parameter is defined at each generation by

$$k_j = |\langle f(x) \rangle| \frac{\langle v_j(x) \rangle}{\sum_{l=1}^{m} [\langle v_l(x) \rangle]^2} \tag{4}$$

Copyright © 2003 John Wiley & Sons, Ltd.

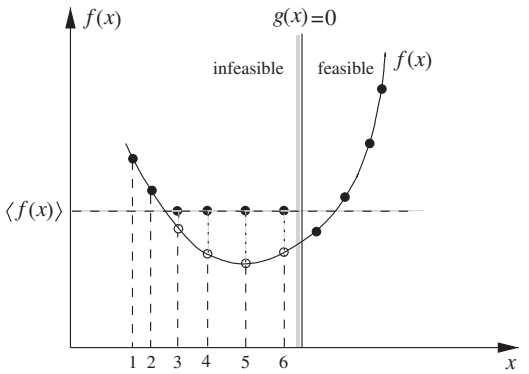

Figure 1. A pictorial description of the function $\bar{f}$.

and $\langle v_l(x)\rangle$ is the violation of the lth constraint averaged over the current population. Denoting by pop the population size, one could also write

$$k_j = \frac{|\sum_{i=1}^{\text{pop}} f(x^i)|}{\sum_{l=1}^{m}\left[\sum_{i=1}^{\text{pop}} v_l(x^i)\right]^2} \sum_{i=1}^{\text{pop}} v_j(x^i) \tag{5}$$

The idea is that the values of the penalty coefficients should be distributed in a way that those constraints which are more difficult to be satisfied should have a relatively higher penalty coefficient.

With the proposed definition one can prove the following:

Property A. An individual whose jth violation equals the average of the jth violation in the current population for all j, has its fitness value given by

$$F(\tilde{x}) = \begin{cases} f(\tilde{x}) + |\langle f(x)\rangle| & \text{if } f(\tilde{x}) > \langle f(x)\rangle \\ \langle f(x)\rangle + |\langle f(x)\rangle| & \text{otherwise} \end{cases}$$

Proof
In fact, let $\tilde{x}$ be such an element. By definition

$$F(\tilde{x}) = \bar{f}(\tilde{x}) + \sum_{j=1}^{m} \frac{|\langle f(x)\rangle|\langle v_j(x)\rangle}{\sum_{l=1}^{m}[\langle v_l(x)\rangle]^2} v_j(\tilde{x})$$

But, by hypothesis, $v_j(\tilde{x}) = \langle v_j(x)\rangle$ for all j leading to

$$F(\tilde{x}) = \bar{f}(\tilde{x}) + \frac{|\langle f(x)\rangle|}{\sum_{l=1}^{m}[\langle v_l(x)\rangle]^2} \sum_{j=1}^{m} [\langle v_j(x)\rangle]^2 = \bar{f}(\tilde{x}) + |\langle f(x)\rangle|$$

and the results follow from Equation (3). □

In the next section, several examples from the literature are considered in order to test the robustness of the proposed adaptive parameter-less scheme. It should be emphasized that

Copyright © 2003 John Wiley & Sons, Ltd.

the accuracy of the final results of the search depends also on other key components of the algorithm not considered here—such as coding, operators and selection scheme—besides the penalization procedure itself. Good results reported in the literature (such as in Reference [11]) are obtained with real coding, more sophisticated selection schemes and more effective genetic operators. The need for better operators in order to obtain good results in constrained continuous optimization has already been pointed out by Hamida and Petrowski [32].

4. NUMERICAL EXPERIMENTS

In order to investigate the robustness of the proposed penalty procedure, several optimization problems from the literature were tackled. The proposed adaptive parameter-less penalty scheme was introduced in our baseline GA and the resulting procedure is referred to as adaptive penalty method (APM) in the comparison of results. Our baseline binary-coded generational GA uses a Gray code, rank-based selection, and elitism (the best element is always copied into the next generation along with nine copies where one bit has been changed). Standard one-point, two-point and uniform crossover operators were applied with probabilities $p_c^1 = 0.16$, $p_c^2 = 0.32$ and $p_c^u = 0.32$, respectively. Mutation was applied bitwise to the offspring with rate $p_m = 0.03$.

It should be noted that no parameter tuning was attempted and the same set of values was applied to *all* test-problems in order to demonstrate the robustness of the procedure. Of course, changing those parameters in each case could lead to local performance gains. In all cases, the total number of objective function evaluations allowed was the same, or very close, to that of the other GAs.

4.1. Test-problem 1—the G-Suite

Before structural optimization problems are discussed, the 11 well-known G1–G11 test-functions presented by Koziel and Michalewicz [5] are considered here. The G-Suite has been repeatedly used as a testbed in the evolutionary computation literature and is made up of different kinds of functions and involves constraints given by linear inequalities (LI), non-linear equalities (NE) and non-linear inequalities (NI). A summary of those functions is presented in the Table I where the column 'opt' indicates minimization or maximization, 'n' shows the number of parameters, 'a' indicates the number of active constraints at the optimum solution and f^* denotes the known optimum value of f. More details of each of these problems are displayed in Tables II and III and the bounds for each parameter, in each function, are defined in Table IV. An extended discussion involving each one of these problems and different techniques from the evolutionary computation literature can be found in Reference [33].

Three set of experiments were performed in Reference [5] and have since then been often used for comparisons. In Experiment 1, using a population size of 70, 20 independent runs were performed with the maximum number of generations set to 5000. In Experiment 2, the maximum number of generations is set to 20 000. Finally, Experiment 3 consists of 10 independent runs (with 5000 generations) where the best solution obtained in the first experiment is introduced in the initial population. For all of the experiments each variable was coded with 20 bits.

Using the same number of function evaluations, the adaptive procedure proposed here (APM) is initially compared with the more complex homomorphous mapping procedure due to Koziel and Michalewicz [5]. Table V summarizes the comparison displaying the results for the worst

Copyright © 2003 John Wiley & Sons, Ltd.

Table I. Test-problem 1—Summary of the 11 test-functions, where 'opt' means minimization or maximization, 'n' is the number of parameters, LI means linear inequalities, NE, non-linear equalities, NI, non-linear inequalities, 'a' indicates the number of active constraints and f^* denotes the know optimum value of f.

Name	opt	n	Type of $f(x)$	LI	NE	NI	a	f^*
$G1$	min	13	Quadratic	9	0	0	6	−15.0
$G2$	max	20	Non-linear	0	0	2	1	0.803553
$G3$	max	10	Polynomial	0	1	0	1	1.0
$G4$	min	5	Quadratic	0	0	6	2	−30655.5
$G5$	min	4	Cubic	2	3	0	3	5126.4981
$G6$	min	2	Cubic	0	0	2	2	−6961.8
$G7$	min	10	Quadratic	3	0	5	6	24.306
$G8$	max	2	Non-linear	0	0	2	0	0.0958250
$G9$	min	7	Polynomial	0	0	4	2	680.63
$G10$	min	8	Linear	3	0	3	6	7049.33
$G11$	min	2	Quadratic	0	1	0	1	0.75

run, the best run, and the average of all runs. The best results in each case are indicated in **boldface**.

It can be noted that the technique of Reference [5] is not able to solve for function $G5$ and is only consistently superior to APM for function $G2$, while for functions $G1$, $G4$, $G5$, $G6$, $G9$ and $G10$, APM presents better results.

Next, the results of APM are compared to the results recently obtained by Wright and Farmani [29] who also used a binary coded GA. In Table VI it is seen that APM provides consistently better results for functions $G1$, $G4$, $G6$, $G9$, $G10$ and $G11$, comparable results in the other cases, and inferior results again for $G2$.

Summarizing, APM constitutes an effective technique for this test suite without requiring any parameter from the user.

4.2. Test-problem 2—the welded beam

This test corresponds to the design of the welded beam studied by Deb [19] and depicted in Figure 2.

The design variables are $\{h, l, t, b\}$, with bounds $0.125 \leqslant h \leqslant 10$, and $0.1 \leqslant l, t, b \leqslant 10$. The objective function to be minimized is the cost of the beam given as

$$c(h, l, t, b) = 1.10471h^2l + 0.04811tb(14.0 + l)$$

subject to

$$g_1(\tau) = 13\,600 - \tau \geqslant 0$$

$$g_2(\sigma) = 30\,000 - \sigma \geqslant 0$$

$$g_3(b, h) = b - h \geqslant 0$$

$$g_4(P_c) = P_c - 6\,000 \geqslant 0$$

$$g_5(\delta) = 0.25 - \delta \geqslant 0$$

Copyright © 2003 John Wiley & Sons, Ltd.

Table II. Test-problem 1—definition of functions G_1–G_7 of the G-Suite.

Function	Constraints
$G_1(x) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$	$2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leqslant 0$ $2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leqslant 0$ $2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leqslant 0$ $-2x_4 - x_5 + x_{10} \leqslant 0, \quad -8x_1 + x_{10} \leqslant 0$ $-2x_6 - x_7 + x_{11} \leqslant 0, \quad -8x_2 + x_{11} \leqslant 0$ $-2x_8 - x_9 + x_{12} \leqslant 0, \quad -8x_3 + x_{12} \leqslant 0$
$G_2(x) = \left\lvert \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right\rvert$	$0.75 - \prod_{i=1}^{n} x_i \leqslant 0, \quad \sum_{i=1}^{n} x_i - 7.5n \leqslant 0$
$G_3(x) = (\sqrt{n})^n \prod_{i=1}^{n} x_i$	$\sum_{i=1}^{n} x_i^2 - 1 = 0$
$G_4(x) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$	$0 \leqslant 85.334407 + 0.0056858x_2x5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 \leqslant 92$ $90 \leqslant 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 \leqslant 110$ $20 \leqslant 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 \leqslant 25$
$G_5(x) = 3x_1 + 0.000001x_1^3 + 2x_2 + 0.000002/3x_2^3$	$x_4 - x_3 + 0.55 \geqslant 0, x_3 - x_4 + 0.55 \geqslant 0$ $1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0$ $1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$ $1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0$
$G_6(x) = (x_1 - 10)^3 + (x_2 - 20)^3$	$-(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leqslant 0$ $(x_1 - 6)^2 - (x_2 - 5)^2 - 82.81 \leqslant 0$
$G_7(x) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$	$-105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leqslant 0$ $10x_1 - 8x_2 - 17x_7 + 2x_8 \leqslant 0$ $-8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leqslant 0$ $3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leqslant 0$ $5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leqslant 0$ $x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leqslant 0$ $0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leqslant 0$ $-3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leqslant 0$

Copyright © 2003 John Wiley & Sons, Ltd.

Table III. Test-problem 1—definition of functions G_8–G_{11} of the G-Suite.

Function	Constraints
$G_8(x) = \dfrac{\sin^3(2\pi x_1)\ \sin(2\pi x_2)}{x_1^3(x_1+x_2)}$	$x_1^2 - x_2 + 1 \leqslant 0$ $1 - x_1 + (x_2-4)^2 \leqslant 0$
$G_9(x) = (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 + 3(x_4-11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$	$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geqslant 0$ $282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geqslant 0$ $196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geqslant 0$ $-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geqslant 0$
$G_{10}(x) = x_1 + x_2 + x_3$	$-1 + 0.0025(x_4 + x_6) \leqslant 0$ $-1 + 0.0025(x_5 + x_7 - x_4) \leqslant 0$ $-1 + 0.01(x_8 - x_5) \leqslant 0$ $-x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \leqslant 0$ $-x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leqslant 0$ $-x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leqslant 0$
$G_{11}(x) = x_1^2 + (x_2-1)^2$	$x_2 - x_1^2 = 0$

Table IV. Test-problem 1—bound constraints for the functions of the G-Suite.

Function	Bound constraints
G_1	$0 \leqslant x_i \leqslant 1\ \ (i=1,...,9),\ 0 \leqslant x_i \leqslant 100\ \ (i=10,11,12),\ 0 \leqslant x_{13} \leqslant 1$
G_2	$0 \leqslant x_i \leqslant 10\ \ (i=1,...,n) n=20$
G_3	$0 \leqslant x_i \leqslant 1\ \ (i=1,...,n) n=10$
G_4	$78 \leqslant x_1 \leqslant 102,\ 33 \leqslant x_2 \leqslant 45,\ 27 \leqslant x_i \leqslant 45\ \ (i=3,4,5)$
G_5	$0 \leqslant x_1, x_2 \leqslant 1200,\ -0.55 \leqslant x_3, x_4 \leqslant 0.55$
G_6	$13 \leqslant x_1 \leqslant 100,\ 0 \leqslant x_2 \leqslant 100$
G_7	$-10 \leqslant x_i \leqslant 10\ \ (i=1,...,10)$
G_8	$0 \leqslant x_1, x_2 \leqslant 10$
G_9	$-10 \leqslant x_i \leqslant 10\ \ (i=1,...,7)$
G_{10}	$100 \leqslant x_1 \leqslant 10000,\ 1000 \leqslant x_i \leqslant 10000\ \ (i=2,3),\ 10 \leqslant x_i \leqslant 1000\ \ (i=4,...,8)$
G_{11}	$-1 \leqslant x_1, x_2 \leqslant 1$

Copyright © 2003 John Wiley & Sons, Ltd.

Table V. Test-problem 1—comparison of results for the G-Suite.

	Exp. #1 (this paper)			Exp. #1 (Koziel and Michalewicz)		
Function	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−14.0566	−14.7207	−14.4609
$G2$	0.6002978	0.7724640	0.7031971	**0.78427**	**0.79506**	**0.79176**
$G3$	0.9391756	0.9939078	0.9757277	**0.9917**	**0.9983**	**0.9965**
$G4$	**−30660.76**	**−30665.24**	**−30663.40**	−30617.0	−30662.5	−30643.8
$G5$	**6040.595**	**5126.571**	**5389.364**	—	—	—
$G6$	**−6961.779**	**−6961.796**	**−6961.789**	−4236.7	−6901.5	−6191.2
$G7$	42.01618	**24.86371**	29.86465	**38.682**	25.132	**26.619**
$G8$	**0.0725015**	**0.0958250**	**0.0926157**	0.0291434	**0.095825**	0.0871551
$G9$	**682.1562**	**680.7590**	**681.4076**	682.88	681.43	682.18
$G10$	**10002.93**	**7086.404**	**8161.997**	11894.5	7215.8	9141.7
$G11$	0.7579745	**0.75**	0.7503349	**0.75**	**0.75**	**0.75**
	Exp. #2 (this paper)			Exp. #2 (Koziel and Michalewicz)		
	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−14.6154	−14.7864	−14.7082
$G2$	0.6499022	0.7918570	0.7514353	**0.79119**	**0.79953**	**0.79671**
$G3$	**0.9983935**	1.000307	**0.9997680**	0.9978	**0.9997**	0.9989
$G4$	**−30664.91**	**−30665.51**	**−30665.29**	−30643.8	−30645.9	−30655.3
$G5$	**6040.595**	**5126.571**	**5389.347**	—	—	—
$G6$	**−6961.796**	**−6961.796**	**−6961.796**	−5473.9	−6952.1	−6342.6
$G7$	33.07581	24.85224	27.90973	**25.069**	**24.62**	**24.826**
$G8$	**0.0795763**	**0.0958250**	**0.0942582**	0.0291438	**0.095825**	0.0891568
$G9$	**681.6396**	**680.6678**	**680.9640**	683.18	680.91	681.16
$G10$	**9977.767**	**7080.107**	**8018.938**	9659.3	7147.9	8163.6
$G11$	**0.75**	**0.75**	**0.75**	**0.75**	**0.75**	**0.75**
	Exp. #3 (this paper)			Exp. #3 (Koziel and Michalewicz)		
	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−14.5732	−14.7184	−14.6478
$G2$	0.7729277	0.7780233	0.7741776	**0.78279**	**0.79486**	**0.78722**
$G3$	**0.9973952**	**0.9997135**	**0.9987124**	0.996	0.9978	0.997
$G4$	**−30665.25**	**−30665.51**	**−30665.39**	−30645.6	−30661.5	−30653.1
$G5$	**5126.571**	**5126.571**	**5126.571**	—	—	—
$G6$	**−6961.796**	**−6961.796**	**−6961.796**	−6390.6	−6944.4	−6720.4
$G7$	**24.86371**	**24.86130**	**24.86346**	26.182	25.09	25.545
$G8$	0.0918033	**0.0958250**	0.0940601	**0.0958246**	**0.0958250**	**0.0958248**
$G9$	**680.7590**	**680.7222**	**680.7461**	683.58	681.72	682.56
$G10$	**7082.667**	**7080.328**	**7081.146**	7685.8	7321.2	7498.6
$G11$	**0.75**	**0.75**	**0.75**	**0.75**	**0.75**	**0.75**

Copyright © 2003 John Wiley & Sons, Ltd.

Table VI. Test-problem 1—comparison of results for the G-Suite.

	Exp. #1 (this paper)			Exp. #1 (Wright and Farmani)		
Function	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−12.9519	−14.9996	−14.84
$G2$	0.6002978	0.7724640	0.7031971	**0.7205**	**0.79434**	**0.76739**
$G3$	0.9391756	0.9939078	0.9757277	**0.99027**	**0.99937**	**0.99812**
$G4$	**−30660.76**	**−30665.24**	**−30663.40**	−30261.6	−30624.1	−30547.915
$G5$	**6040.595**	**5126.571**	**5389.364**	—	5126.64487	—
$G6$	**−6961.779**	**−6961.796**	**−6961.789**	−6347.8	−6948.85	−6484.06
$G7$	42.01618	24.86371	**29.86465**	**37.98319**	**24.672**	31.52044
$G8$	**0.0725015**	**0.0958250**	**0.0926157**	0.0267	0.09588	0.089135
$G9$	**682.1562**	**680.7590**	**681.4076**	712.869	681.5615	688.05
$G10$	**10002.93**	**7086.404**	**8161.997**	10572.66	7298.136	8776.7699
$G11$	**0.7579745**	**0.75**	**0.7503349**	0.9884	**0.75**	0.8151
	Exp. #2 (this paper)			Exp. #2 (Wright and Farmani)		
	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−14.9081	−14.9996	−14.988
$G2$	0.6499022	0.7918570	0.7514353	**0.77091**	**0.79664**	**0.78465**
$G3$	**0.9983935**	1.000307	**0.9997680**	0.99807	**0.9994**	0.99902
$G4$	**−30664.91**	**−30665.51**	**−30665.39**	−30604.2	−30650.01	−30609.97
$G5$	6040.595	**5126.571**	5389.347	**5135.4409**	5126.6398	**5131.0404**
$G6$	**−6961.796**	**−6961.796**	**−6961.796**	−6347.8	−6961.37	−6657.81
$G7$	**33.07581**	24.85224	**27.90973**	37.9273	**24.6707**	30.927
$G8$	**0.0795763**	**0.0958250**	**0.0942582**	0.06413	0.09588	0.09246
$G9$	**681.6396**	**680.6678**	**680.9640**	689.6234	681.1982	684.413
$G10$	**9977.767**	**7080.107**	**8018.938**	10343.04	7152.832	8255.847
$G11$	**0.75**	**0.75**	**0.75**	0.90296	**0.75**	0.808
	Exp. #3 (this paper)			Exp. #3 (Wright and Farmani)		
	Worst	Best	Average	Worst	Best	Average
$G1$	**−15.00**	**−15.00**	**−15.00**	−14.9691	−14.9996	−14.9911
$G2$	0.7729277	0.7780233	0.7741776	**0.7812**	**0.8003**	**0.7845**
$G3$	0.9973952	**0.9997135**	0.9987124	**0.9989**	0.9994	**0.9991**
$G4$	**−30665.25**	**−30665.51**	**−30665.39**	−30604.2	−30650	−30609.025
$G5$	**5126.571**	**5126.571**	**5126.571**	—	5126.63997	—
$G6$	**−6961.796**	**−6961.796**	**−6961.796**	−6347.8	−6958.44	−6692.61
$G7$	**24.86371**	24.86130	**24.86346**	30.723	**24.61897**	27.3009
$G8$	0.0918033	**0.0958250**	0.0940601	**0.09314**	0.09588	**0.095328**
$G9$	**680.7590**	**680.7222**	**680.7461**	685.7403	681.162	683.4643
$G10$	**7082.667**	**7080.328**	**7081.146**	8357.847	7289.063	7788.533
$G11$	**0.75**	**0.75**	**0.75**	0.795	**0.75**	0.768

Copyright © 2003 John Wiley & Sons, Ltd.

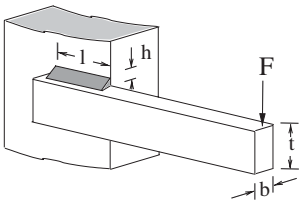

Figure 2. Test-problem 2 (welded beam). The designs variables are: h (width of the weld), l (length of the weld), t, and b (height and width of the beam, respectively). The load applied at the tip of the beam is $F = 6000$ lb.

The expressions for τ, σ, P_c, and δ involve the design variables and are given by

$$\tau = \sqrt{(\tau')^2 + (\tau'')^2 + l\tau'\tau''/\alpha}$$

$$\alpha = \sqrt{0.25(l^2 + (h+t)^2)}, \quad \sigma = \frac{504\,000}{t^2 b}$$

$$P_c = 64746.022\ (1 - 0.0282346t)tb^3$$

$$\delta = \frac{2.1952}{t^3 b}, \quad \tau' = \frac{6000}{\sqrt{2}hl}$$

$$\tau'' = \frac{6000(14 + 0.5l)\alpha}{2\{0.707hl(l^2/12 + 0.25(h+t)^2)\}}$$

Two experiments are conducted. The first one is used to evaluate the performance of the same baseline binary-coded GA using two different penalty schemes: (i) the standard penalty scheme—Equations (1) and (2)—using five constant penalty coefficients ($k = 10^2$, $k = 10^3$, $k = 10^4$, $k = 10^5$ and $k = 10^6$) and (ii) the APM proposed here.

Each variable was coded with 10 bits leading to a chromosome with 40 bits. The population size was set to 80, the maximum number of generations was 500, and 50 independent runs were performed.

A comparison of results in the Table VII shows: (i) that better values were found using the adaptive technique (APM) proposed here and (ii) the lack of any pattern in the results as the value of k is increased.

The second experiment compares APM with a more sophisticated real-coded GA proposed by Deb where, among other features, niching is used. Essentially, niching in a GA allows for a subdivision of the population in subpopulations where each one of them explores a potential region of the search space. Niching is used by Deb in order to maintain diversity in the population of candidate solutions.

The population size was again set to 80 in both cases. Two series of 50 runs were performed allowing for 500 and 4000 generations, respectively. Using our GA, each variable was coded with 30 bits generating a chromosome 120 bits long. Results for this test are shown in Table VIII.

Copyright © 2003 John Wiley & Sons, Ltd.

Table VII. Test-problem 2–cost of the welded beam found using constant penalty parameters versus APM within the same baseline GA.

	Best run	Worst run
$k = 10^2$	2.56822	2.63816
$k = 10^3$	3.09351	3.36340
$k = 10^4$	2.98612	3.04114
$k = 10^5$	2.47781	3.07576
$k = 10^6$	2.49419	3.24281
APM	2.39623	3.39956

Table VIII. Test-problem 2—values found for the cost of the welded beam—Deb's real-coded GA [19] versus APM.

			Reference [19]	
	Maxgenn	APM	W/o niching	With niching
Best	500	2.38352	2.44271	2.38119
	4000	2.38159	—	2.38119
Worst	500	3.73060	7.44425	2.64583
	4000	2.95533	—	2.38355

Table IX. Test-problem 2—details of the best solution for the cost of the welded beam found by APM: $c = 2.38159$.

Variables	Constraints
$h = 0.2442949$	$g_1(\tau) = 0.0004447$
$l = 6.2116738$	$g_2(\sigma) = 64.378068$
$t = 8.3015486$	$g_3(b, h) = 0.0000054$
$b = 0.2443003$	$g_4(P_c) = 0.0002553$
	$g_5(\delta) = 0.2342937$

It is clear that Deb's results only improve when the penalty scheme (besides the real coding) is augmented with niching—which in turn requires the definition of niching parameter(s)—while APM provides competitive results without requiring any parameter from the user. Table IX shows the details of the best solution for the cost of the welded beam found by APM: $c = 2.38159$.

4.3. Test-problem 3—the pressure vessel

The third problem has been studied exhaustively in the literature [34–37] and corresponds to the weight minimization of a cylindrical pressure vessel with two spherical heads as shown in

Copyright © 2003 John Wiley & Sons, Ltd.

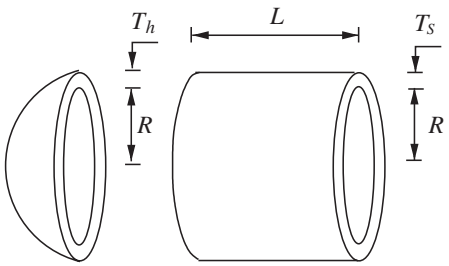

Figure 3. Test-problem 3—pressure vessel—(T_s) is the thickness of the pressure vessel, (T_h) is the thickness of the head, (R) is the inner radius of the vessel and (L) the length of the cylindrical component.

Table X. Test-problem 3—Weight found for the pressure vessel using constant penalty parameters for the baseline GA.

	Best run	Worst run		Best run	Worst run
$k = 10^2$	Not found	Not found	$k = 10^5$	6059.926	7332.905
$k = 10^3$	Not found	Not found	$k = 10^6$	6060.891	6838.765
$k = 10^4$	6059.746	8421.833	$k = 10^7$	6059.746	7274.531

Figure 3. The objective function involves four variables: the thickness of the pressure vessel (T_s), the thickness of the head (T_h), the inner radius of the vessel (R) and the length of the cylindrical component (L). Since there are two discrete variables (T_s and T_h) and two continuous variables (R and L), one has a non-linearly constrained mixed discrete-continuous optimization problem.

The bounds of the design variables are $0.0625 \leqslant T_s, T_h \leqslant 5$ (in constant steps of 0.0625) and $10 \leqslant R, L \leqslant 200$. The design variables are given in inches and the weight is written as

$$w(T_s, T_h, R, L) = 0,6224 T_s T_h R + 1.7781 T_h R^2 + 3.1661 T_s^2 L + 19.84 T_s^2 R$$

to be minimized subject to the constraints

$$g_1(T_s, R) = T_s - 0.0193R \geqslant 0$$

$$g_2(T_h, R) = T_h - 0.00954R \geqslant 0$$

$$g_3(R, L) = \pi R^2 L + 4/3\pi R^3 - 1\,296\,000 \geqslant 0$$

$$g_4(L) = -L + 240 \geqslant 0$$

The first two constraints establish a lower bound to the ratios T_s/R and T_h/R, respectively. The third constraint corresponds to a lower bound for the volume of the vessel and the last one to an upper bound for the length of the cylindrical component.

Using the same baseline GA (allowing for 70 000 function evaluations), the results of adopting constant penalty parameters (Table X) are compared with those obtained by APM (Table XI).

Copyright © 2003 John Wiley & Sons, Ltd.

Table XI. Test-problem 3—design variables, constraints and weight found for the pressure vessel.

Variables	Reference [36]	Reference [31]	APM
T_s	0.9375	0.8125	0.8125
T_h	0.5000	0.4375	0.4375
R	48.3290	42.097398	42.094658
L	112.6790	176.654047	176.684062
$g_1(T_s, R)$	0.005	0.000020	0.000073
$g_2(T_h, R)$	0.039	0.035891	0.035917
$g_3(R, L)$	3652.898	27.886075	2.929000
$g_4(L)$	127.321	63.345953	63.315938
w	6410.381	6059.946341	6060.187934

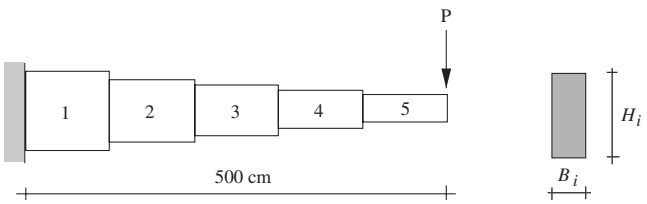

Figure 4. Test-problem 4—the cantilever beam—the values of B_i and H_i $(i = 1, 5)$, are the width and the height of the cross-section of the beam, respectively, and represent 10 design parameters.

It is important to note that using either $k = 10^2$ or 10^3 no feasible solutions are found, illustrating the difficulty in finding a constant penalty parameter and indicating that it is necessary to investigate several levels of penalty in order to find a feasible solution.

Next, a comparison of APM with other techniques in the literature is made.

A dominance-based selection scheme was recently proposed by Coello in Reference [31] and compared with other techniques. In that paper he shows that his results (the best results in a series of 30 runs and 80 000 function evaluations) are the best ones followed by those obtained by Deb using a genetic adaptive search (GeneAS) in Reference [36].

In Table XI, those results are displayed together with the ones obtained by APM in the best run (in a series of 30 runs and 80 000 function evaluations). It can be noted that: (i) all the solutions displayed in Table XI are rigorously feasible, (ii) the solutions found by APM and that found in Reference [31] represent the same design (from an engineering point of view) while that found in Reference [36] represents a very different design, and, finally, (iii) that the results from Reference [31] and APM are much better and differ in weight by less than 0.004%.

4.4. Test-problem 4—the cantilever beam

This test problem [38] corresponds to the minimization of the volume of the cantilever beam shown in Figure 4 where the load P is equal to 50 000 N. There are 10 design variables corresponding to the height (H_i) and width (B_i) of the rectangular cross-section of each of

Copyright © 2003 John Wiley & Sons, Ltd.

the five constant steps shown in Figure 4. The variables B_1 and H_1 are integer, B_2 and B_3 assume discrete values to be chosen from the set $\{2.4, 2.6, 2.8, 3.1\}$, H_2 and H_3 are discrete and chosen from the set $\{45.0, 50.0, 55.0, 60.0\}$ and, finally, B_4, H_4, B_5 and H_5 are continuous. The variables are given in centimetres and Young's modulus of the material is equal to 200 GPa.

The volume of the beam, to be minimized, is given by

$$v = 100 \sum_{i=1}^{5} H_i B_i$$

subject to

$$g_i(H_i, B_i) = \sigma_i \leqslant 14\,000\ \mathrm{N/cm^2}, \quad i = 1, \ldots, 5$$

$$g_{i+5}(H_i, B_i) = H_i / B_i \leqslant 20, \quad i = 1, \ldots, 5$$

$$g_{11}(H_i, B_i) = \delta \leqslant 2.7\ \mathrm{cm}$$

where δ is the tip deflection of the beam in the vertical direction.

In Table XII results obtained using different techniques are presented. The first four columns reproduce results presented by Thanedar and Vanderplaats [39]. The first column corresponds to using a continuous round-up method, the second column to an accurate discrete technique, the third column to a linear approximate discrete procedure, the fourth column to a conservative approximate discrete technique, and, finally, the fifth and sixth columns use a GA-based optimum structural technique (GAOS) proposed by Erbatur *et al.* [38]. This technique uses a multilevel approach which reduces the size of the search space for an individual in each successive level of the optimization process. Three levels were employed, each one allowing for 10 000 function evaluations. The results obtained using the APM in Table XII correspond to a population size of 70 and 500 generations. Twenty independent runs were performed.

The values in **boldface** denote constraint violations and the corresponding solutions, although with better results, (except for those presented in the first column), are infeasible.

It is interesting to note that starting from the optimal continuous solution, both rounding-up (first column) or rounding-off (not shown) procedures lead to infeasible, suboptimal solutions [39]. Also, the designs found by GAOS and APM are very similar and no solution in Table XII is both rigorously feasible and lighter than that found by APM.

In order to observe the performance of the APM when a smaller number (28 000) of function evaluations is allowed for, constraints are relaxed. In this case they are writen as $g_i(x) \leqslant \varepsilon$, where ε is a small tolerance. The first column of results shown in Table XIII were obtained considering $\varepsilon = 0.01$. In this case the APM found a competitive value for the objective function (64691.98) with a displacement (2.7099 in) greater than the allowable for this problem (2.7 in). The second column corresponds to no relaxation of constraints ($\varepsilon = 0$) and a higher weight is found, which is competitive in terms of number of function evaluations with the results achieved in Table XII.

In Table XIV, the performance of the baseline GA using a constant penalty coefficient is presented. It illustrates the difficulty of setting an effective penalty parameter which usually requires a potentially costly trial-and-error process. Competitive values for the weight of the beam begin to appear only when using parameters k greater than 10^5. However, in order to have this problem-dependent information, several runs had to be performed.

Copyright © 2003 John Wiley & Sons, Ltd.

Table XII. Test-problem 4—comparison of results for the cantilever beam—the first four columns present results from Thanedar and Vanderplaats [39] and the columns 5 and 6 were obtained with a GA-based optimum structural technique (GAOS) proposed by Erbatur *et al*. [38].

Var.	Continuous round up	Accurate discrete	Linear approx. discrete	Conserv. approx. discrete	GAOS level 1	GAOS level 2	APM
B_1	4	3	3	3	3	3	3
B_2	3.1	3.1	3.1	3.1	3.1	3.1	3.1
B_3	2.6	2.6	2.6	2.6	2.6	2.6	2.6
B_4	2.205	2.276	2.262	2.279	2.300	2.270	2.2894
B_5	1.751	1.750	1.750	1.750	1.800	1.750	1.7931
H_1	62	60	60	60	60	60	60
H_2	60	55	55	55	55	55	55
H_3	55	50	50	50	50	50	50
H_4	44.09	45.528	445.233	45.553	45.50	45.250	45.6256
H_5	35.03	34.995	34.995	35.004	35.00	35.000	34.5931
Volume	73555	64537	64403	64558	64815	64447	64698.56
g_1	9755.46	13888.89	13888.89	13888.89	13888.89	13888.89	13888.89
g_2	10752.69	12796.59	12796.59	12796.59	12796.59	12796.59	12796.59
g_3	11443.10	13846.15	13846.15	13846.15	13846.15	13846.15	13846.15
g_4	13997.89	12718.09	12964.26	12687.41	12600.87	12908.87	12589.61
g_5	13962.23	13998.17	13998.17	13990.97	13605.44	13994.17	13980.98
g_6	15.50	20.00	20.00	20.00	20.00	20.00	20.00
g_7	19.35	17.74	17.74	17.74	17.74	17.74	17.74
g_8	**21.15**	19.23	19.23	19.23	19.23	19.23	19.23
g_9	19.9954	**20.0035**	19.9969	19.9881	19.7826	19.9339	19.9289
g_{10}	**20.0057**	19.9971	19.9971	**20.0023**	19.4444	20.0000	19.2919
g_{11}	2.1349	**2.7026**	**2.7111**	**2.7015**	2.6960	**2.7094**	2.6999

Table XIII. Test-problem 4—comparison of results for the cantilever beam using a population size equal to 70 and a maximum number of generations equal to 400. ε is the tolerance used for constraint relaxation.

Variables	$\varepsilon = 0.01$	$\varepsilon = 0$	Constraints	$\varepsilon = 0.01$	$\varepsilon = 0$
B_1	3	3	g_1	13888.89	13888.89
B_2	3.1	3.1	g_2	12796.59	12796.59
B_3	2.6	2.6	g_3	13846.15	13846.15
B_4	2.3082	2.3239	g_4	12808.85	12804.95
B_5	1.8118	1.789	g_5	13941.99	12235.02
H_1	60	60	g_6	20.00	20.00
H_2	55	55	g_7	17.74	17.74
H_3	50	50	g_8	19.23	19.23
H_4	45.0488	44.9027	g_9	19.51	19.32
H_5	34.4626	35.5950	g_{10}	19.02	19.89
			g_{11}	**2.7099**	2.6999
Volume	64691.98	64853.26			

Copyright © 2003 John Wiley & Sons, Ltd.

Table XIV. Test-problem 4—optimum weight found for the cantilever beam using a constant penalty parameter within the baseline GA.

	Best run	Worst run		Best run	Worst run
$k = 10^2$	122987.14	184671.76	$k = 10^5$	64579.76	68222.76
$k = 10^3$	65472.09	190971.93	$k = 10^6$	64582.28	68395.94
$k = 10^4$	66086.14	78646.75	$k = 10^7$	64578.43	68313.08

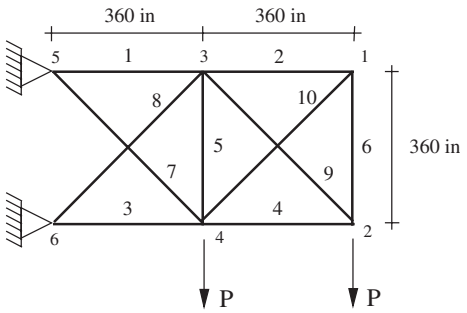

Figure 5. Test-problem 5—the ten-bar truss.

4.5. Test-problem 5—the ten-bar truss

This is the well-known test problem corresponding to the weight minimization of the ten-bar truss shown in Figure 5. The constraints involve the stress in each member and the displacements at the nodes. The design variables are the cross-sectional areas of the bars (A_i, $i = 1, 10$). The allowable stress is limited to ± 25 ksi and the displacements are limited to 2 in, in the x and y directions. The density of the material is 0.1 lb/in^3, Young's modulus is $E = 10^4$ ksi and vertical downward loads of 100 kips are applied at nodes 2 and 4.

Two cases are analysed: discrete and continuous variables. For the discrete case the values of the cross-sectional areas (in^2) are chosen from the set $\mathscr{S}$ with 42 options: 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, 11.50, 13.50, 13.90, 14.20, 15.50, 16.00, 16.90, 18.80, 19.90, 22.00, 22.90, 26.50, 30.00, 33.50. Using a six-bit string for each design variable, as in Reference [40], a total of 64 strings are available. In this way, the first 22 values of the set $\mathscr{S}$ (from 1.62 to 4.59) are listed twice (1.62, 1.62, 1.80, 1.80, etc.). For the continuous case the minimum cross-sectional area is equal to 0.1 in^2.

4.5.1. The continuous case. For the continuous case, several classical approaches are available and Table XV shows a comparison between some of the techniques from the literature [40–47] arranged according to decreasing optimum weight W found. Although such classical methods—when applicable—usually require a smaller number of function evaluations, for this problem they produced inferior results when compared to the APM. For the continuous case the APM used a population size equal to 200, 1400 generations and 20 independent runs were performed.

Copyright © 2003 John Wiley & Sons, Ltd.

Table XV. Test-problem 5—comparison of results from the literature and the present work where W is the final weight in lb—continuous case—ten-bar truss.

A_i	Reference [41]	Reference [42]	Reference [40]	Reference [43]	Reference [44]	Reference [45]	Reference [46]	Reference [47]
1	31.350	30.570	25.73	33.432	30.416	30.500	30.670	30.731
2	0.100	0.369	0.109	0.100	0.128	0.100	0.100	0.100
3	20.030	23.970	24.85	24.260	23.408	23.290	23.760	23.934
4	15.600	14.730	16.35	14.260	14.905	15.428	14.590	14.733
5	0.140	0.100	0.106	0.100	0.101	0.100	0.100	0.100
6	0.240	0.364	0.109	0.100	0.101	0.210	0.100	0.100
7	8.350	8.547	8.70	8.388	8.696	7.649	8.578	8.542
8	22.210	21.110	21.41	20.740	21.084	20.980	21.070	20.954
9	22.060	20.770	22.30	19.690	21.077	21.818	20.960	21.836
10	0.100	0.320	0.122	0.100	0.186	0.100	0.100	0.100
W	5112.00	5107.30	5095.65	5089.00	5084.90	5080.00	5076.85	5076.66

Table XVI. Test-problem 5—comparison of results from the literature and the present work where W is the final weight in lb—continuous case—ten-bar truss.

A_i	APM	Reference [48]	Reference [49]	Reference [50]	Reference [51]	Reference [52]	Reference [53]	Reference [54]
1	29.22568	30.0310	30.56091	30.520	31.28	32.9657	30.440	30.561
2	0.10000	0.1000	0.10000	0.100	0.10	0.1000	0.100	0.100
3	24.18212	23.2740	23.16999	23.200	24.65	22.7988	21.790	27.946
4	14.94714	15.2860	15.11224	15.220	15.39	14.1463	14.260	13.619
5	0.10000	0.1000	0.10000	0.100	0.10	0.1000	0.100	0.100
6	0.39463	0.5565	0.54910	0.551	0.10	0.7393	0.451	0.100
7	7.49579	7.4683	7.47048	7.457	7.90	6.3807	7.628	7.907
8	21.92486	21.1980	21.09910	21.040	21.53	20.9121	21.630	19.345
9	21.29088	21.6180	21.52714	21.530	19.07	20.9779	21.360	19.273
10	0.10000	0.1000	0.10000	0.100	0.10	0.1000	0.100	0.100
W	5069.086	5061.60	5060.920	5060.80	5052.00	5013.24	4987.00	4981.1

Table XVI shows another set of results, for the same case, including the results from APM and other References [48–54].

For the ten-bar truss problem, physical optimality criteria techniques were used in References [41, 44, 55]; CONstrained function MINimization—CONMIN was used in References [42, 54]; approximation concepts for structural optimization were used in Reference [43]; mathematical optimality criterion with Kunh–Tucker conditions was applied in the References [47, 45]; an NEW unconstrained sequential minimization technique—NEWSUMIT was used in Reference [46]; convex programming with Kuhn–Tucker conditions were adopted in Reference [50]; general geometric programming is used in Reference [51] and, finally, non-linear goal programming was tested in Reference [52].

In Reference [49] a real-coded steady-state GA is used considering a constant penalty function equal to 500 and the best solution was found using 40 000 function evaluations. In

Copyright © 2003 John Wiley & Sons, Ltd.

Table XVII. Test-problem 5—comparison of results on ten-bar truss, where 'W shown' is the weight of the structure presented in the cited reference and 'W evaluated' is weight actually evaluated for each structure if one uses the values of the optimal cross-sectional areas given in those references; (u_{y1}) and (u_{y2}) are the vertical displacements at nodes 1 and 2, respectively and the values in **boldface** are beyond the limits set to the problem–continuous case.

Reference	W shown	W evaluated	u_{y1}	u_{y2}
[41]	5112.00	5112.624	−1.99999	−1.98696
[42]	5107.30	5107.323	−1.99991	−1.99998
[40]	5095.65	5095.637	**−2.01151**	**−2.01368**
[43]	5089.00	5091.569	−1.99844	−1.99982
[44]	5084.90	5084.809	−1.99860	−1.99998
[45]	5080.00	5080.041	−1.99997	−1.98164
[46]	5076.85	5077.149	−1.99989	−1.99988
[47]	5076.66	5127.617	−1.98233	−1.97777
APM	5069.086	5069.086	−2.00000	−1.98937
[48]	5061.60	5061.659	−1.99999	−1.99149
[49]	5060.92	5060.920	−1.99999	−1.99147
[50]	5060.80	5060.926	−1.99996	−1.99138
[51]	5052.00	5052.628	**−2.01949**	**−2.00657**
[52]	5013.24	5013.237	**−2.01312**	−2.00000
[53]	4987.00	4999.215	**−2.02798**	**−2.01850**
[54]	4981.10	4981.093	**−2.04994**	**−2.06047**

Reference [53] a simple GA is implemented with binary coding, a population size of 200, 20 000 function evaluations and constant penalty parameters also adjusted by trial and error.

Table XVII shows the value of the weight of the optimum structure presented in each reference and the weight that is actually evaluated for each one if the values of the optimal cross-sectional areas given in those references are used.

It is interesting to observe the differences between these values in some references. Also, this table presents the displacements in the vertical direction of the node 1 (u_{y1}) and in the vertical direction of the node 2 (u_{y2}). It is easy to note that several solutions in this table have lower (better) weight but are infeasible with respect to those constraints. The results found by the APM in this problem are rigorously feasible and only slightly worse (0.16%) than those in References [48–50] which are also feasible.

As in the cantilever beam problem, constraints are now relaxed ($\varepsilon = 0.01$), and a smaller number (40 000) of function evaluations is allowed for, in 20 independent runs. The results found for the design variables are: $A_1 = 31.43271$, $A_2 = 0.10259$, $A_3 = 24.40457$, $A_4 = 14.49322$, $A_5 = 0.10019$, $A_6 = 0.13181$, $A_7 = 7.71714$, $A_8 = 20.72951$, $A_9 = 20.50630$ and $A_{10} = 0.10011$, leading to a weight, in the best run, equal to 5041.317 lbs. The displacements are $u_{y1} = -2.01999$ and $u_{y2} = -2.00354$, both greater than those imposed for this problem (2.0 in).

4.5.2. The discrete case. For the discrete case, not as many results are available in the literature as for the continuous case and some of them are presented in Table XVIII. A binary-coded GA was used in Reference [56] where a penalty function depends on a constant parameter. The results corresponding to a population size of 40, and 20 generations are displayed in

Copyright © 2003 John Wiley & Sons, Ltd.

Table XVIII. Test-problem 5—design variables and total weight found for the ten-bar truss–discrete case.

A_i	Reference [56]	APM$^+$	Reference [57]	APM*	APM	Reference [40]
1	33.50	26.50	33.50	30.00	33.50	33.50
2	1.62	1.62	1.62	1.62	1.62	1.62
3	22.00	26.50	22.00	22.00	22.90	22.00
4	15.50	16.90	14.20	16.90	14.20	13.90
5	1.62	1.80	1.62	1.62	1.62	1.62
6	1.62	1.62	1.62	1.62	1.62	1.62
7	14.20	13.50	7.97	11.50	7.97	7.97
8	19.90	22.00	22.90	22.00	22.90	22.90
9	19.90	19.90	22.00	22.00	22.00	22.90
10	2.62	1.99	1.62	1.80	1.62	1.62
w	5613.84	5619.662	5458.3	5572.60	5490.738	5447.54
W	5613.58	5619.662	5458.3	5572.60	5490.738	5493.36
u_{y1}	−1.87722	−1.88792	−1.97349	−1.91271	−1.95908	−1.96419
u_{y2}	**−2.00075**	−1.99917	**−2.01227**	−1.98513	−1.98892	**−2.00074**

Table XVIII. Using the same parameters of Reference [56], in 30 independent runs, the APM found 5619.662 lbs as shown in column APM$^+$ in Table XVIII and this solution is rigorously feasible.

In Reference [57] a GA with binary coding introducing the concept of 'rebirth' is discussed and for this problem a population size of 200, performing 23 200 function evaluations was used. Using 24 000 function evaluations (population size equal to 200, and 120 generations), the APM found, in 30 independent runs, a rigorously feasible solution (5572.60 lbs) as shown in the column APM* in Table XVIII.

Results in column 5 correspond to using APM with a population size of 200, 90 000 function evaluations, and 20 independent runs. The results for a GA with binary coding, used in Reference [40], with a constant penalty coefficient, relaxation of constraints, population size of 100, and 20000 function evaluations, are shown in the last column of Table XVIII.

Again, there are solutions with displacements (in **boldface**) greater than the limits imposed for this problem and these solutions are infeasible, obviously. Once again, the results found by the APM are rigorously feasible, as shown in Table XVIII.

From the practical engineering point of view, it is clear that the observed constraint violations may be considered negligible. However, although they lead to 'better' objective function values, they may potentially indicate a difficulty of the corresponding algorithm in achieving a good feasible solution.

4.6. Test-problem 6—the 25-bar truss

In this test problem, a truss with 25 bars shown in Figure 6 is submitted to weight minimization. The constraints require that the maximum stresses in the members remain in the interval $[-40, 40]$ ksi and that the maximum displacements at nodes 1 and 2 be limited to 0.35 in, in both the x and y directions. The design variables are cross-sectional areas of the bars to be chosen from the set with 30 different options (in square inches): 0.1, 0.2, 0.3, 0.4, 0.5, 0.6,

Copyright © 2003 John Wiley & Sons, Ltd.

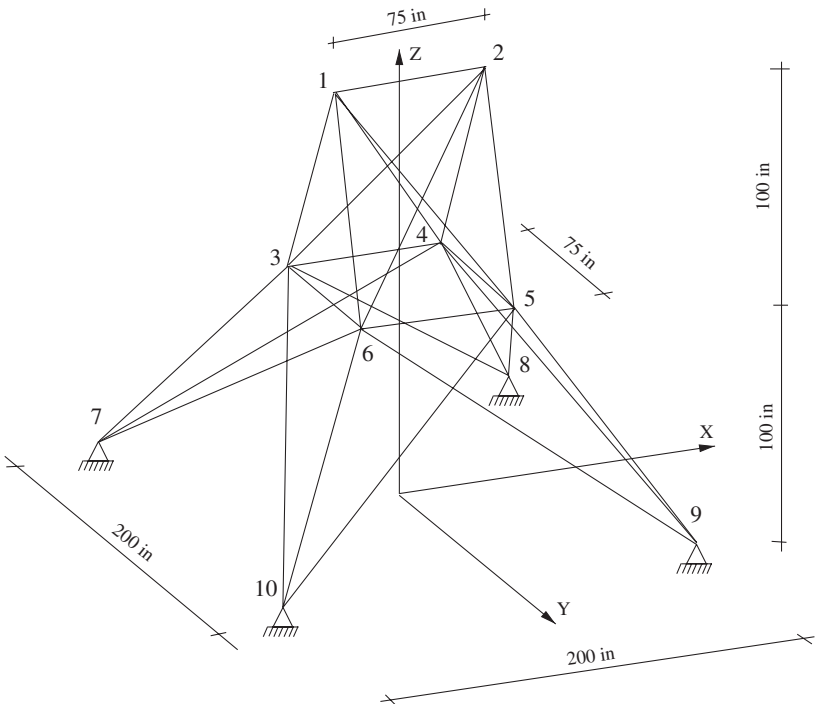

Figure 6. Test-problem 6—the 25-bar truss.

Table XIX. Test-problem 6—loading data for the 25-bar truss (kips).

Node	F_x	F_y	F_z
1	1	−10.0	−10.0
2	0	−10.0	−10.0
3	0.5	0	0
6	0.6	0	0

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. The design variables are linked in eight member groups detailed in Table XX. The material has density equal to 0.1 lb/in^3 and Young's modulus is equal to 10^4 ksi. The loading data is listed in Table XIX.

Table XXI presents several solutions found in the literature as well as the values of the vertical displacements u_{y1} and u_{y2}, at the nodes 1 and 2, respectively, since these are the hardest constraints observed for this problem. The results shown in the first column [56] were obtained using a binary-coded GA with a population size of 40, running for 20 generations. Using the same parameters, the APM found a weight of 486.743lb (column APM$^+$), better than the result found in Reference [56] as shown in Table XXI. Reference [38] uses a multilevel optimization approach (GAOS) and, for this (discrete) case, uses two levels with 10 000 function

Copyright © 2003 John Wiley & Sons, Ltd.

Table XX. Test-problem 6—member grouping for the 25-bar truss.

Group	Connectivities
A_1	1–2
A_2	1–4, 2–3, 1–5, 2–6
A_3	2–5, 2–4, 1–3, 1–6
A_4	3–6, 4–5
A_5	3–4, 5–6
A_6	3–10, 6–7, 4–9, 5–8
A_7	3–8, 4–7, 6–9, 5–10
A_8	3–7, 4–8, 5–9, 6–10

Table XXI. Test-problem 6—comparison of results for the 25-bar truss where W is the final weight in lb–discrete case; u_{y1} and u_{y2} are the displacements at the nodes 1 and 2, respectively. Results in the first two columns correspond to 800 function evaluations while those in the last 3 columns to 30 000, 40 000 and 20 000, respectively.

Variables	Reference [56]	APM$^+$	Reference [58]	Reference [38]	Reference [59]	APM
A_1	0.1	0.1	0.1	0.1	0.1	0.1
A_2	1.8	0.7	1.9	1.2	0.5	0.3
A_3	2.3	3.4	2.6	3.2	3.4	3.4
A_4	0.2	0.1	0.1	0.1	0.1	0.1
A_5	0.1	1.8	0.1	1.1	1.5	2.1
A_6	0.8	1.0	0.8	0.9	0.9	1.0
A_7	1.8	0.3	2.1	0.4	0.6	0.5
A_8	3.0	3.4	2.6	3.4	3.4	3.4
W	546.01	486.743	562.93	493.80	486.29	484.854
u_{y1}	−0.3481	−0.3493	−0.3486	−0.3499	−0.3495	−0.3498
u_{y2}	−0.3477	−0.3465	−0.3482	−0.3479	−0.3479	−0.3478

evaluations in each one of them. In Reference [59] a binary-coded steady-state GA is proposed using a constant penalty function and allowing for 40 000 function evaluations. The results of the APM presented in the last column of Table XXI correspond to a population size of 100, allowing for 20 000 function evaluations. An improved Templeman's algorithm was applied in Reference [58]. It is clear that: (i) all solutions are feasible, (ii) correspond to distinct designs and (iii) the technique proposed here provides the smallest weight.

4.7. Test-problem 7—the 52-bar truss

The 52-bar truss shown in Figure 7 is submitted to weight minimization. The loading condition is defined in Table XXII and the truss has the following material properties: Young's modulus equal to 2.07×10^5 MPa and density equal to 7860 kg/m^3. The allowable stresses in tension and compression are both set to 180 MPa. The members of the truss are linked in 12 groups defined in Table XXIII and the values of the cross-sectional areas are to be chosen from Table XXIV.

Copyright © 2003 John Wiley & Sons, Ltd.

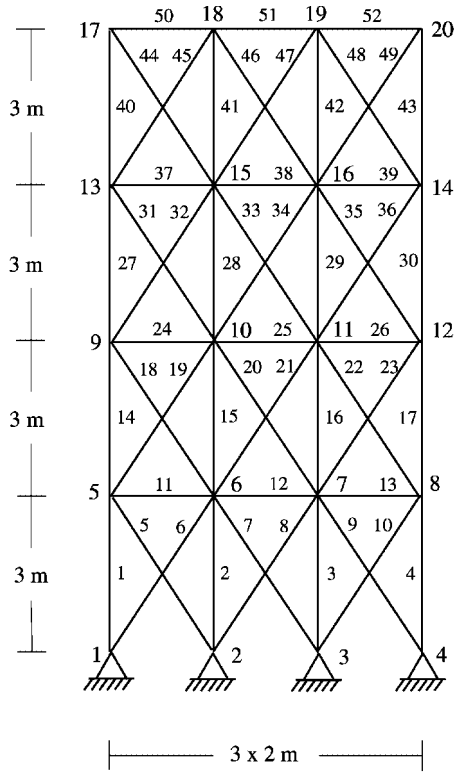

Figure 7. Test-problem 7—the 52-bar truss.

Table XXII. Test-problem 7—loading data for the 52-bar truss (kN).

Node	F_x	F_y
17	100.0	200.0
18	100.0	200.0
19	100.0	200.0
20	100.0	200.0

Table XXV presents two sets of results from Reference [59], one set from Reference [49] and another from the APM. The results from [59] correspond to experiments using a steady-state GA where crossovers of one point (the first column) and two points (the second column) were tested independently, performing 60 000 function evaluations in each case. The results from Referemce [49] were obtained by a generational GA with the same operators of APM and 20 000 function evaluations. The results from APM in this example uses a population size equal to 70, 250 generations in 20 independent runs. It is important to note that all the solutions in the Table XXV satisfy the whole set of constraints imposed for the problem. An interesting detail is noted from the results of the APM and Reference [49] since they both present essentially

Copyright © 2003 John Wiley & Sons, Ltd.

Table XXIII. Test-problem 7—member grouping for the 52-bar truss.

Group	Members
A_1	1, 2, 3, 4
A_2	5, 6, 7, 8, 9, 10
A_3	11, 12, 13
A_4	14, 15, 16, 17
A_5	18, 19, 20, 21, 22, 23
A_6	24, 25, 26
A_7	27, 28, 29, 30
A_8	31, 32, 33, 34, 35, 36
A_9	37, 38, 39
A_{10}	40, 41, 42, 43
A_{11}	44, 45, 46, 47, 48, 49
A_{12}	50, 51, 52

Table XXIV. Test-problem 7—cross-sectional areas available for the 52-bar truss.

Section	mm^2	Section	mm^2	Section	mm^2	Section	mm^2
1	71.613	17	1008.385	33	2477.414	49	7419.340
2	90.968	18	1045.159	34	2496.769	50	8709.660
3	126.451	19	1161.288	35	2503.221	51	8967.724
4	161.290	20	1283.868	36	2696.769	52	9161.272
5	198.064	21	1374.191	37	2722.575	53	9999.980
6	252.258	22	1535.481	38	2896.768	54	10322.560
7	285.161	23	1690.319	39	2961.284	55	10903.204
8	363.225	24	1696.771	40	3096.768	56	12129.008
9	388.386	25	1858.061	41	3206.445	57	12838.684
10	494.193	26	1890.319	42	3303.219	58	14193.520
11	506.451	27	1993.544	43	3703.218	59	14774.164
12	641.289	28	2019.351	44	4658.055	60	15806.420
13	645.160	29	2180.641	45	5141.925	61	17096.740
14	792.256	30	2238.705	46	5503.215	62	18064.480
15	816.773	31	2290.318	47	5999.998	63	19354.800
16	940.000	32	2341.191	48	6999.986	64	21612.860

the same optimum weight (1903.366392 versus 1903.366416) but with distinct values for some design variables.

4.8. Test-problem 8—the 72-bar truss

A 72-bar truss structure depicted in the Figure 8 is considered now. The design variables are the cross-sectional areas of the bars and the minimum value for each one is 0.1 in^2. The 72 design variables are linked in 16 groups detailed in Table XXVI. The constraints involve a maximum allowable displacement of 0.25 in at the nodes 1 to 16 along the x and y directions,

Copyright © 2003 John Wiley & Sons, Ltd.

Table XXV. Test-problem 7—comparison of results for the 52-bar truss where W is the final weight in kg.

Variables	Reference [59]	Reference [59]	Reference [49]	APM
A_1	3703.218	4658.055	4658.055	4658.055
A_2	2722.575	1161.288	1161.288	1161.288
A_3	1858.575	645.160	363.225	494.193
A_4	3206.445	3303.219	3303.219	3303.219
A_5	1008.385	1045.159	940.000	940.000
A_6	1008.385	494.193	641.289	641.289
A_7	2477.414	2477.414	2238.705	2238.705
A_8	1008.385	1045.159	1008.385	1008.385
A_9	388.386	285.161	494.193	363.225
A_{10}	2477.414	1696.771	1283.868	1283.868
A_{11}	1008.385	1045.159	1161.288	1161.288
A_{12}	1008.383	641.289	494.193	494.193
W	2294.521	1970.142	1903.36639	1903.36641

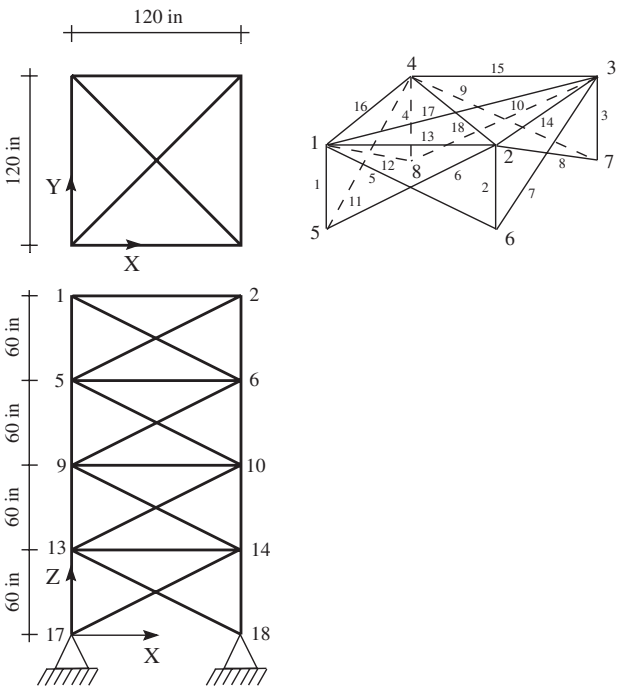

Figure 8. Test-problem 8—the 72-bar truss.

and a maximum allowable stress in each bar restricted to the range $[-25, 25]$ ksi. The density of the material is $0.1\,\mathrm{lb/in^3}$ and Young's modulus is equal to 10^4 ksi. Two load cases, defined in Table XXVII, are considered for this structure.

Copyright © 2003 John Wiley & Sons, Ltd.

Table XXVI. Test-problem 8—member grouping for the 72-bar truss.

Group	Members
A_1	1, 2, 3, 4
A_2	5, 6, 7, 8, 9, 10, 11, 12
A_3	13, 14, 15, 16
A_4	17, 18
A_5	19, 20, 21, 22
A_6	23, 24, 25, 26, 27, 28, 29, 30
A_7	31, 32, 33, 34
A_8	35, 36
A_9	37, 38, 39, 40
A_{10}	41, 42, 43, 44, 45, 46, 47, 48
A_{11}	49, 50, 51, 52
A_{12}	53, 54
A_{13}	55, 56, 57, 58
A_{14}	59, 60, 61, 62, 63, 64, 65, 66
A_{15}	67, 68, 69, 70
A_{16}	71, 72

Table XXVII. Test-problem 8—loading data for the 72-bar truss (kips).

Load case	Node	F_x	F_y	F_z
1	1	5	5	−5
2	1	0	0	−5
	2	0	0	−5
	3	0	0	−5
	4	0	0	−5

Table XXVIII presents a comparison of results with respect to the cross-sectional area of each bar and the final weight reached for the 72-bar truss. Concerning Table XXVIII, it must be noted that, according to our computations, based on the values shown in that table, the solutions marked with an asterisk are not rigorously feasible while the one marked with a '+' (although feasible) has an actual weight of 385.6189 lb. Only the solution presented in Reference [60] is both rigorously feasible and has a (1.51%) better result than APM.

References [41, 60] used traditional optimization techniques based in physical optimality criterion. Approximation concepts for structural optimization were used in Reference [43]. For this case, in 20 independent runs, the APM uses a population of 70 and 500 generations. The results in columns 4 and 5 [38], were obtained with the GA multilevel optimization approach GAOS. It is noted that when moving from 20 000 function evaluations (level 2, column 4) to 30 000 (level 3, column 5) the final weight is reduced, but the solution obtained is no longer rigorously feasible.

Finally, Table XXIX shows a summary of the results for the structural optimization test-problems where the best, average and worst objective function values found are listed together with the population size 'popsiz' and the maximum number of function evaluations, 'maxeval'.

Copyright © 2003 John Wiley & Sons, Ltd.

Table XXVIII. Test-problem 8—comparison of results for the 72-bar truss. W is the final weight in lb.

Var.	Reference [60]	Reference [41]	Reference [43]	Reference [38]	Reference [38]*	APM
A_1	0.161	0.1492	0.1585	0.155	0.161	0.15500
A_2	0.557	0.7733	0.5936	0.535	0.544	0.54534
A_3	0.377	0.4534	0.3414	0.480	0.379	0.27496
A_4	0.506	0.3417	0.6076	0.520	0.521	0.51853
A_5	0.611	0.5521	0.2643	0.460	0.535	0.60365
A_6	0.532	0.6084	0.5480	0.530	0.535	0.66607
A_7	0.100	0.1000	0.1000	0.120	0.103	0.10159
A_8	0.100	0.1000	0.1509	0.165	0.111	0.13008
A_9	1.246	1.0235	1.1067	1.155	1.310	1.19954
A_{10}	0.524	0.5421	0.5793	0.585	0.498	0.47368
A_{11}	0.100	0.1000	0.1000	0.100	0.110	0.10059
A_{12}	0.100	0.1000	0.1000	0.100	0.103	0.10945
A_{13}	1.818	1.4636	2.0784	1.755	1.910	1.95307
A_{14}	0.524	0.5207	0.5034	0.505	0.525	0.51653
A_{15}	0.100	0.1000	0.1000	0.105	0.122	0.10000
A_{16}	0.100	0.1000	0.1000	0.155	0.103	0.10105
W	381.2	395.97	388.63	385.76	383.12	387.036

The solutions marked with an asterisk are not rigorously feasible.

Table XXIX. Summary of results for the structural optimization test-problems, where 'popsiz' and 'maxeval' denote, respectively, population size used and maximum number of function evaluations allowed in APM.

	Best run	Average	Worst run	Popsiz	Maxeval
TP-2	2.38159	2.41718	2.95533	80	320000
TP-3	6060.188	6311.766	6838.939	200	80000
TP-4	64698.562	68107.046	73931.359	70	35000
TP-5[c]	5069.09	5091.43	5117.39	200	280000
TP-5[d]	5490.74	5545.48	5567.84	200	90000
TP-6	484.854	485.967	490.742	100	20000
TP-7	1903.366	2077.467	2383.755	70	17500
TP-8	387.036	402.588	432.953	70	35000

5. CONCLUSIONS

A new, simple, adaptive, parameter-less penalty scheme for the solution of constrained problems via genetic algorithms has been proposed. Its main feature, besides being adaptive and not requiring any parameter, is to automatically define, for each constraint, a different penalty coefficient which varies along the run according to the feedback received from the evolutionary process.

In all problems tested so far—including continuous and/or discrete variables, implicit, non-linear inequalities and equalities—the procedure has produced very good results: either the

Copyright © 2003 John Wiley & Sons, Ltd.

best available in the literature or very competitive ones. It must also be emphasized that such good results were obtained in spite of the use of a very simple baseline binary coded genetic algorithm. Good results in the literature are often obtained with real coding, more sophisticated selection schemes, more effective operators and/or by exploiting domain knowledge. The procedure is simple and can be easily implemented in any existing generational GA code.

Several well-known structural optimization test-problems were successfully solved and discussed in this paper which also points out good—but not rigorously feasible—solutions presented in the literature.

The main result of the paper is to present an effective and robust penalty method which relieves the user from the burden of having to determine, by trial and error, penalty—or any other—parameter(s) to deal with every new constrained optimization problem to be solved.

By not relying in specific domain knowledge, the proposed parameter-less adaptive penalty scheme can be applied to general constrained engineering optimization problems and, at the same time, allows for potential performance gains if any available domain knowledge is introduced in the design of the other components of the GA (such as the coding, the selection scheme and the genetic operators) which are also important in search effectiveness and efficiency.

ACKNOWLEDGEMENTS

The authors acknowledge the support received from CNPq (301233/86-1 and 475398/2001-7) and FAPEMIG (TEC-692/99). The authors would also like to thank the reviewers for the corrections and suggestions which helped to improve the quality of the paper.

REFERENCES

1. Holland JH. *Adaptation in Natural and Artificial Systems*. University of Michigan Press: Ann Arbor, 1975.
2. Goldberg DE. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley: Reading, MA, 1989.
3. Goldberg DE, Samtani MP. Engineering optimization via genetic algorithms. In *Proceedings of the 9th Conference on Electronic Computation*. ASCE: New York, NY, 1986; 471–482.
4. Schoenauer M, Michalewicz Z. Evolutionary computation at the edge of feasibility. In *Parallel Problem Solving from Nature—PPSN IV*, Voigt H-M, Ebeling W, Rechenberg I, Schwefel H-P (eds). Lecture Notes in Computer Sciences, Vol. 1141, Springer: Berlin, 1996; 245–254.
5. Koziel S, Michalewicz Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation* 1999; **7**(1):19–44.
6. Liepins GE, Potter WD. A genetic algorithm approach to multiple-fault diagnosis. In *Handbook of Genetic Algorithms*, Davis L (ed.). Chapter 17, Van Nostrand Reinhold: New York, 1991; 237–250.
7. Orvosh D, Davis L. Using a genetic algorithm to optimize problems with feasibility contraints. In *Proceedings of the First IEEE Conference on Evolutionary Computation*. IEEE Press: New York, 1994; 548–553.
8. Adeli H, Cheng N-T. Augmented Lagrangian genetic algorithm for structural optimization. *Journal of Aerospace Engineering* 1994; **7**(1):104–118.
9. Barbosa HJC. A coevolutionary genetic algorithm for constrained optimization problems. In *Proceedings of the Congress on Evolutionary Computation*. Washington, DC, USA, 1999; 1605–1611.
10. Surry PD, Radcliffe NJ. The COMOGA method: constrained optimization by multiobjective genetic algorithms. *Control and Cybernetics* 1997; **26**(3):391–412.
11. Runarsson TP, Yao X. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation* 2000; **4**(3):284–294.
12. van Kampen AHC, Strom CS, Buydens LMC. Lethalization, penalty and repair functions for constraint handling in the genetic algorithm methodology. *Chemometrics and Intelligent Laboratory Systems* 1996; **34**:55–68.

Copyright © 2003 John Wiley & Sons, Ltd.

13. Michalewicz Z, Schoenauer M. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation* 1996; **4**(1):1–32.
14. Hinterding R, Michalewicz Z. Your brains and my beauty: Parent matching for constrained optimization. In *Proceedings of the Fifty International Conference on Evolutionary Computation*. Alaska, 4–9 May 1998; 810–815.
15. Koziel S, Michalewicz Z. A decoder-based evolutionary algorithm for constrained optimization problems. In *Proceedings of the Fifth Parallel Problem Solving from Nature*, Bäck T, Eiben AE, Schoenauer M, Schwefel H-P (eds). Amsterdam, Lecture Notes in Computer Science, Springer: Berlin. 27–30 September 1998.
16. Kim J-H, Myung H. Evolutionary programming techniques for constrained optimization problems. *IEEE Transactions on Evolutionary Computation* 1997; **2**(1):129–140.
17. Barbosa HJC, Lemonge ACC. An adaptive penalty scheme in genetic algorithms for constrained optimization problems. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, Langdon WB, Cantú-Paz E, Mathias K, Roy R, Davis D, Poli R, Balakrishnan K, Honavar V, Rudolph G, Wegener J, Bull L, Potter MA, Schultz AC, Miller JF, Burke E, Jonoska N (eds). Morgan Kaufmann: New York, 9–13 July 2002; 287–294.
18. Powell D, Skolnick MM. Using genetic algorithms in engineering design optimization with non-linear constraints. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, Forrest S (ed.). Morgan Kaufmann: San Mateo, CA, 1993; 424–430.
19. Deb K. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering* 2000; **186**(2–4):311–338.
20. Michalewicz Z. A survey of constraint handling techniques in evolutionary computation. In *Proceedings of the 4th International Conference on Evolutionary Programming*. MIT Press: Cambridge, MA, 1995; 135–155.
21. Michalewicz Z, Dasgupta D, Le Riche RG, Schoenauer M. Evolutionary algorithms for constrained engineering problems. *Computers and Industrial Engineering* 1996; **30**(2):851–870.
22. Le Riche RG, Knopf-Lenoir C, Haftka RT. A segregated genetic algorithm for constrained structural optimization. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, Eshelman LJ (ed.). Pittsburgh, PA, July 1995; 558–565.
23. Homaifar H, Lai SH-Y, Qi X. Constrained optimization via genetic algorithms. *Simulation* 1994; **62**(4): 242–254.
24. Joines JA, Houck CR. On the use of non-stationary penalty functions to solve non-linear constrained optimization problems with GAs. In *Proceedings of the First IEEE International Conference on Evolutionary Computation*, Michalewicz Z, Schaffer JD, Schwefel H-P, Fogel DB, Kitano K (eds). 19–23 June 1994; 579–584.
25. Bean JC, Alouane AB. A dual genetic algorithm for bounded integer programs. *Technical Report* TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992.
26. Coit DW, Smith AE, Tate DM. Adaptive penalty methods for genetic optimization of constrained combinatorial problems. *INFORMS Journal on Computing* 1996; **6**(2):173–182.
27. Schoenauer M, Xanthakis S. Constrained GA optimization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, Forrest S (ed.). Morgan Kaufmann: Los Altos, CA, 1993; 573–580.
28. Hamida SB, Schoenauer M. An adaptive algorithm for constrained optimization problems. In *Parallel Problem Solving from Nature–PPSN VI*, Lectures Notes in Computer Science, vol. 1917. Springer: Berlin, 2000; 529–538.
29. Wright JA, Farmani R. Genetic algorithms: A fitness formulation for constrained minimization. In *Proceedings of the Genetic and Evolutionary Computation Conference—GECCO 2001*. Morgan Kaufmann: San Francisco, CA, 2001; 725–732.
30. Horn J, Nafpliotis N. Multiobjective optimization using the niched Pareto Genetic Algorithm. *Technical Report* 93005, University of Illinois, Illinois Genetic Algorithms Laboratory, 1993.
31. Coello CAC, Montes EM. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics* 2002; **16**:193–203.
32. Hamida SB, Petrowski A. The need for improving the exploration operators for constrained optimization problems. In *2000 Congress on Evolutionary Computation*. IEEE Service Center, San Diego, CA, USA, July 2000; 1176–1183.
33. Michalewicz Z, Fogel DB. *How to Solve It: Modern Heuristics*. Springer: Berlin, 1999.
34. Sandgren E. Nonlinear integer and discrete programming in mechanical design. In *Proceedings of the ASME Design Technology Conference*. Kissimee, FL, 1988; 95–105.

Copyright © 2003 John Wiley & Sons, Ltd.

35. Kannan BK, Kramer SN. An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design* (ASME) 1994; **116**(2):405–411.
36. Deb K. GeneAS: a robust optimal design technique for mechanical component design. In *Evolutionary Algorithms in Engineering Applications*. Dasgupta D, Michalewicz Z (eds.), Springer: Berlin, 1997; 497–514.
37. Coello CAC. Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry* 2000; **41**(2):113–127.
38. Erbatur F, Hasançebi O, Tütüncü I, Kilç H. Optimal design of planar and space structures with genetic algorithms. *Computers and Structures* 2000; **75**:209–224.
39. Thanedar PB, Vanderplaats GN. Survey of discrete variable optimization for structural design. *Journal of Structural Engineering–ASCE* 1995; **2**(121):301–306.
40. Ghasemi MR, Hinton E, Wood RD. Optimization of trusses using genetic algorithms for discrete and continuous variables. *Engineering Computations* 1997; **16**:272–301.
41. Gellatly RA, Berke L. Optimal structural design. *Technical Report* AFFDL-TR-70-165, Air Force Flight Dynamics Laboratory, AFFDL, 1971.
42. Schimit LA, Miura H. A new structural analysis/synthesis capability: access 1. *AIAA Journal* 1976; **14**: 661–671.
43. Schimit LA, Farshi B. Some approximation concepts in structural synthesis. *AIAA Journal* 1974; **12**: 692–699.
44. Venkayya VB, Khot NS, Reddy VS. Energy distribution in an optimal structural design. *Technical Report* AFFDL-TR-68-156, Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, 1969.
45. Dobbs MV, Nelson RB. Application of optimality criteria to automated structural design. *AIAA Journal* 1976; **14**:1436–1443.
46. Schimit LA, Miura H. Approximation concepts for efficient structural synthesis. *Technical Report* CR-2552, NASA, 1976.
47. Rizzi P. Optimization of multiconstrained structures based on optimality criteria. In *Proceedings of 17th Structures, Structural Dynamics, and Materials Conference*, King of Prussia, PA, 1976.
48. Haug EJ, Arora JS. *Applied Optimal Design*. Wiley: New York, 1979.
49. Lemonge ACC. Application of Genetic algorithms in Structural Optimization Problems. *Ph.D. Thesis*, Program of Civil Engineering–COPPE, Federal University of Rio de Janeiro, Brazil, 1999 (in Portuguese).
50. Haftka RT, Kamat MP. *Elements of Structural Optimization*. Martinus Nighoff: Dordrecht, 1985.
51. Adeli H, Kamal O. Efficient optimization of plane trusses. *Advances in Engineering Software* 1991; **13**(3): 116–122.
52. El-Sayed ME, Jang TS. Structural optimization using unconstrained non-linear goal programming algorithm. *Computers and Structures* 1994; **52**(4):723–727.
53. Galante M. Structures optimization by a simple genetic algorithm. In *Numerical Methods in Engineering and Applied Sciences*, CIMNE: Barcelona, Spain. 1992; 862–870.
54. Memari AM, Fuladgar A. Minimum weight design of trusses by BEHSAZ program. In *Advances in Structural Optimization*, Topping BHV, Papadrakakis M (eds). Civil-Comp Press: Athens, Greece, 1994; 179–185. (The Second International Conference on Computational Structures Technology.)
55. Khan MR, Willmert KD, Thornton WA. An optimality criterion method for large-scale structures. *AIAA Journal* 1979; **17**(7):753–761.
56. Krishnamoorty CS, Rajeev S. Discrete optimization of structures using genetic algorithms. *Journal of Structural Engineering* 1992; **118**(5):1233–1250.
57. Galante M. Genetic algorithms as an approach to optimize real-world trusses. *International Journal for Numerical Methods in Engineering* 1996; **39**:361–382.
58. Zhu DM. An improved Templeman's algorithm for optimum design of trusses with discrete member sizes. *Engineering Optimization* 1986; **9**:303–312.
59. Wu SJ, Chow PT. Steady-state genetic algorithms for discrete optimization of trusses. *Computers and Structures* 1995; **56**(6):979–991.
60. Venkayya VB. Design of optimum structures. *Journal of Computers and Structures* 1971; **1**(1–2):265–309.

Copyright © 2003 John Wiley & Sons, Ltd.