STUDI E RICERCHE - Vol.20, 1999.
Scuola di Specializzazione in Costruzioni in C.A. - Fratelli Pesenti.
Politecnico di Milano, Italia.

OPTIMAL LIMIT STATES DESIGN OF CONCRETE STRUCTURES USING GENETIC ALGORITHMS

Fabio Biondini (1)

ABSTRACT

The optimum design of concrete structures under static loads as regards both serviceability and ultimate limit conditions is presented. The procedure is oriented to the optimal design of R.C. frames but it is also suitable for other kind of structures. Design variables include either the shape and the dimensions of the concrete cross-sections, or the amount and location of the reinforcement. The objective of the design process is to minimise the structural cost of the system according to side and behavioural constraints. The optimal solution is achieved by using genetic algorithms, non deterministic heuristic search methods which formulation is based on some analogies with the principles which regulates the natural selection of the biological systems. The structural analyses needed for the solution process are performed by taking mechanical and geometrical non-linearities into account. The method of analysis is based on the finite element technique by means of a suitable composite beam element. An application to the optimisation of a R.C. frame shows the effectiveness of the presented procedure.

---

(1) PhD Student, Department of Structural Engineering, Milan University of Technology, Italy.
Post-Graduate at the School for Reinforced Concrete Structures "F.lli Pesenti.

1. INTRODUCTION

The structural design usually involves an iterative process in which the designer, based not only on his knowledge, but also on his experience and intuition, moves from a set of possible feasible and unfeasible solutions toward the feasible one which appears to be the most suitable for his purposes. Typically, the most economic design is searched for. This trial-and-error procedure may be tedious and computationally very expensive, especially when the structural system is far from the assumption of linear behaviour, which usually happens for concrete structures. Moreover, it doesn't give any guarantee as to both the optimality or the near-optimality of the adopted final solution. One way to overcome these drawbacks and limitations, is to frame the design problem within the field of the structural optimisation, which can provide very powerful analytical and numerical tools for a *direct* approach to the optimum structural design (Rao 1996).

Of course, interest about the practical application of the optimisation theory is progressively growing in many areas of structural engineering. Over the years a lot of research related to either deterministic or probabilistic design of reinforced and prestressed concrete structures has been developed (Kirsch 1993, Cohn 1994, Cohn & Dinovitzer 1994). Two main directions have been followed, the analytical way, based on optimality criteria (Adamu & Karihaloo 1995, Han *et al.* 1996), and the numerical way, based on mathematical programming methods (Kirsch 1983, Cohn & MacRae 1984, Kanagasundaram & Karihaloo 1990, Bontempi *et al.* 1998*a*, Biondini 1998). To this regard, the numerical methods certainly allow for a powerful structural synthesis, but their computational effectiveness is limited by the dimensions of the optimisation problem. On the contrary, when the dimensions of the problem increase, analytical approaches generally tend to be more efficient than the numerical ones, but, despite this, their practical application is usually restricted to very simple structures. Therefore, although a wide number of different techniques have been employed, not one of them appears fully satisfactory for a broad class of different structural problems. Moreover, since structural quantities are generally rounded over discrete intervals, additional drawbacks may arises when the design variables are not continuous but discrete-valued. In these cases exhaustive or enumerative search methods may be certainly used, but their practical applicability is limited to very small search spaces.

One of the most promising research fields which has been recently applied to structural optimisation deals with non deterministic heuristic methods which operate on the basis of some analogies with the growing and the evolutionary processes of biological systems (Steven *et al.* 1998). These methods have shown to be fairly robust and effective because, since only function values are usually required, they easily handle non convex problems with non differentiable functions and either discrete or continuous variables (Chong & Zak 1996, Rao 1996).

Genetic algorithms are search procedures which belong to this class of *heuristics from nature*. Based on the principles of the natural selection, they combine Darwin's principle of the survival-of-the-fittest and a non deterministic structured information exchange to build an efficient search mechanism (Goldberg 1989, Michalewicz 1992). Despite their heuristic nature, genetic algorithms operate according to strong theoretical foundations (Holland 1975) which assure a high level of performance. Such algorithms have been successfully applied to many classes of structural problems, especially dealing with truss (Jenkins 1991*a*, 1991*b*, Rajeev & Krishnamoorthy 1992, Dhingra & Lee 1994, Ohsaki 1995, Rajan 1995, Wu & Chow 1995*a*, 1995*b*, Galante 1996, Rajeev & Krishnamoorthy 1997, Shrestha & Ghaboussi 1998) and frame structures (Jenkins 1992, Grierson & Pak 1993).

In this paper, a systematic approach to the optimal design of concrete structures by using genetic algorithms is presented. In particular, the procedure is oriented to the optimal design of concrete frames but it is also suitable for other kinds of structures. Design variables include either the shape and the dimensions of the concrete cross-sections, or the amount and location of the reinforcement over the structure. Since the concrete dimensions and the location of the reinforcement are usually required as discrete quantities (f.i. whole centimetres) and the reinforcing bars are manufactured in standard sizes, the design variables are suitable to be treated as discrete-valued. The objective of the design process is to minimise the structural cost of the system according to side and behavioural constraints. In particular, both serviceability and ultimate limit conditions are considered to formulate the design constraints.

At first, the optimisation problem as regards the structural cost minimisation is formulated. Subsequently, the main features of the genetic algorithms as well as their theoretical foundations are exposed. Since the search technique requires the evaluation of functions which are implicitly related to internal quantities of the system (i.e. geometrical and material properties, stresses, strains, forces, displacements, etc.), repeated structural analyses are needed during the solution process. In this paper, the structural analyses are performed by taking mechanical and geometrical non-linearities into account. As regards this, the method of analysis, based on the finite element technique by means of a suitable composite beam element, is briefly recalled (Bontempi *et al.* 1995, Malerba 1998). Finally, an application to the optimisation of a reinforced concrete frame shows the effectiveness of the presented procedure.

2. OPTIMUM STRUCTURAL DESIGN

2.1. Formulation of the Optimization Problem

The problem of the optimum structural design consists of finding a set of design variables (i.e. geometrical and material properties, amount and location of the reinforcement, etc.) which accounts for assigned design constraints (on stresses, strains, forces, displacements, etc.) and optimises one or more given target requirement (structural cost, structural weight, strength, ductility, etc.). Therefore, from a mathematical point of view, the purpose of a one-target design process is to find a vector **x** which optimises the value of an objective function $f(\mathbf{x})$, according to either side constraints with bounds $\mathbf{x}^{-}$ and $\mathbf{x}^{+}$, or inequality $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$ and/or equality $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ behavioural constraints. Since, without any loss of generality, minimisation problems only may be considered, the previous optimisation problem is cast in the following form:

$$\min_{\mathbf{x} \in D} f(\mathbf{x}) \tag{1.a}$$

$$D = \left\{ \mathbf{x} \mid \mathbf{x}^{-} \leq \mathbf{x} \leq \mathbf{x}^{+},\ \mathbf{g}(\mathbf{x}) \leq \mathbf{0},\ \mathbf{h}(\mathbf{x}) = \mathbf{0} \right\} \tag{1.b}$$

which, in the general case, represent a non-linear programming problem.

2.2. Objective Function

As already mentioned, several quantities able to represent the structural performances may be chosen as target requirements for the optimal design. In this work, the adopted objective function is related to the total cost $C(\mathbf{x})$ of the structure. In particular, the sum of the costs of the component materials, concrete and reinforcing steel, is considered:

$$C(\mathbf{x}) = c_c V_c + c_s V_s \tag{2}$$

where $V_c(\mathbf{x})$, $V_s(\mathbf{x})$, are the total volumes of the concrete and of the steel, respectively, and c_c, c_s, are the corresponding unit costs. It is clear that additional cost components, i.e. cost of formwork and so on, may be considered in the present formulation. In any case, it should be noted that the function $C(\mathbf{x})$ represents a consistent criterion to compare different designs rather than the actual structural cost in a strict sense.

Since unit costs are usually not depending on the vector **x**, the optimisation problem is implicitly defined by the unit cost ratio $c = c_s / c_c$. Therefore, the objective function is assumed as follows:

$$f(\mathbf{x}) = V_c + cV_s \tag{3}$$

2.3. Constraints and the Penalty Function Method

The dimensions and the components of the vectors $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ which define the behavioural constraints are clearly depending on the particular design problem which has to be solved. However, at this stage, it is important to outline that direct search strategies of the optimal solution, like genetic algorithms, often cannot directly take behavioural constraints into account.

One way to handle such constraints is to penalise infeasible vectors $\mathbf{x} \notin D$ by adding to the objective function a penalty term $p(\mathbf{x}) \geq 0$, whose value is zero if no constraint violation occurs and is positive otherwise. In other words, the original constrained optimisation problem is transformed into the following unconstrained form:

$$\min_{\mathbf{x} \in E} j(\mathbf{x}) \tag{4.a}$$

$$E = \left\{ \mathbf{x} \mid \mathbf{x}^- \leq \mathbf{x} \leq \mathbf{x}^+ \right\} \tag{4.b}$$

being:

$$j(\mathbf{x}) = f(\mathbf{x}) + p(\mathbf{x}) \tag{5}$$

the new penalised objective function which has to be optimised within the side-bounded, obviously convex, feasible domain $E \supseteq D$.

Of course, the choice of a suitable penalty function $p(\mathbf{x})$ represents a critical point in establishing the effectiveness and the robustness of the search technique. Here, this function is defined as follows:

$$p(\mathbf{x}) = p_0 + \Delta p = p_0 + p_1 \mathbf{g}^T \boldsymbol{\Gamma} \mathbf{g} + p_2 \mathbf{h}^T \mathbf{h} \tag{6}$$

where the terms $p_i \geq 0$, with $i = 0,1,2,$ are penalty coefficients and $\Gamma(\mathbf{x})$ is a diagonal matrix whose k^{th} diagonal term $G_k(\mathbf{x})$ is given by the Heaviside operator applied to the corresponding inequality constraint:

$$G_k(\mathbf{x}) = H[g_k(\mathbf{x})] = \begin{cases} 0 & \text{, if } g_k(\mathbf{x}) \leq \mathbf{0} \\ 1 & \text{, if } g_k(\mathbf{x}) > \mathbf{0} \end{cases} \tag{7}$$

A proper value of the coefficient p_0 should be set in such a way that the worst feasible solution $\mathbf{x}_W \in D$ is better than the best unfeasible one $\mathbf{x}_B \notin D$, or $j(\mathbf{x}_W) < j(\mathbf{x}_B)$. This condition can be achieved at each iteration t by continuously updating its value during the search process, f.i. by assuming (Powell 1983, Michalewicz 1992):

$$p_0(\mathbf{x},t) = \begin{cases} 0 & \text{, if } \mathbf{x} \in D \\ \max\limits_{\mathbf{x} \in D} f(\mathbf{x},t) - \max\limits_{\mathbf{x} \notin D} [f(\mathbf{x},t) + \Delta p(\mathbf{x},t)] > 0 & \text{, if } \mathbf{x} \notin D \end{cases} \tag{8}$$

Although such an assumption should be itself enough to lead the procedure toward an optimal solution $\mathbf{x} \in D$, non zero constant values of the parameters p_1 and p_2 yield to a weighted measure of the behavioural constraint violations which assures a hierarchical arrangement of the explored set of possible solutions $\mathbf{x} \in E$. As will be pointed out, such classifications may assume an important role for those optimisation techniques which, like genetic algorithms, process contemporarily a population of possible solutions rather than a single candidate. Moreover, since some minor degree of constraint violations is often acceptable in design, a good enough unfeasible solution may be selected if no economic feasible solutions appear.

3. GENETIC ALGORITHMS

3.1. Introduction

Genetic algorithms are heuristic search techniques which belong to the class of stochastic algorithms, since they combine elements of deterministic and probabilistic search (Goldberg 1989, Michalewicz 1992). Their formulation is inspired by the principles of the natural selection which associate a well structured information exchange to the fundamental principle of the *survival-of-the-fittest* enunciated by Charles Darwin (1859) as the main evolutionary principle. For this reason, genetic algorithms borrow much of their terminology from genetics.

The search strategy works on a *population* of *individuals* subjected to an evolutionary process where individuals compete between them to survive in proportion to their *fitness* with the *environment*. In this process, population undergoes continuous reproduction by means of some *genetic operators* which, because of competition, tend to preserve best individuals. From this evolutionary mechanism, two conflicting trends appear: exploiting of the best individuals and exploring the environment. Thus, the effectiveness of the genetic search depends on a balance between them, or between two principal properties of the system, *population diversity* and *selective pressure*. These aspects are in fact strongly related, since an increase in the selective pressure decreases the diversity of the population, and vice versa.

The *structure* of an individual (*genotype*) consists of a number of *chromosomes* containing structured information about its meaning in the environment (*phenotype*). Usually, just individuals formed by a single chromosome are considered. A chromosome consists of units, called *genes*, whose locations within the chromosome itself are called *loci*. Finally, a gene can be in several states, called *alleles*.

In the following, some details about the translation of these evolutionary concepts in heuristic rules suitable for mathematical optimisation, are presented. The theoretical foundations of the genetic algorithms, which distinguish them from a purely random search, are also briefly recalled.

3.2. Population and Environment

With reference to the optimisation problem previously formulated, a population of *m* individuals belonging to the environment *E* represent a collection *X* of *m* possible solutions $\mathbf{x}_k \in E$:

$$X = \{ \mathbf{x}_1 \quad \mathbf{x}_2 \quad \dots \quad \mathbf{x}_m \} \tag{9}$$

each defined by a set of *n* design variables x_i^k :

$$\mathbf{x}_k^T = [x_1^k \quad x_2^k \quad \dots \quad x_n^k] \qquad , k = 1,\dots,m \tag{10}$$

Evolution starts from an *initial population* X_0 selected by means of a *seeding* process, where *m* individuals $\mathbf{x}_{0k} \in E$ are generated at random or heuristically deduced on the basis of some knowledge about the distribution of the fitness within the environment.

The population size *m* clearly influences the evolutionary process and then the effectiveness of the genetic search (Sven & Schmid 1993). In fact, if its value is too small, population diversity cannot be assured and the genetic algorithm may converge too quickly. Conversely, if it is too large, selective pressure cannot be adequately exerted and the waiting time for an improvement may be too long.

3.3. Fitness Function

The fitness of each individual **x** is measured by a scalar function $F(\mathbf{x}) \geq 0$ which increase with the adaptability of **x** to its environment *E*. Therefore, as regards the optimisation problem, the fitness function is related to the penalised objective function $j(\mathbf{x})$ as follows:

$$F(\mathbf{x}) = \frac{j(\mathbf{x}_B)}{j(\mathbf{x})} \tag{11}$$

In this way, the fitness of the best individual $\mathbf{x}_B$ in the current generation equals unity, or $F(\mathbf{x}) \in [0;1]$. Moreover, the already mentioned role played by a proper choice of the penalty coefficients p_1 and p_2, now clearly appears. In fact, a vectors **x** which violates some behavioural constraints is allowed for a propagation of its genes during reproduction in proportion to its degree of violation, so that higher genetic diversity of the population is preserved. However, to assure an actual balance between population diversity and selective pressure, some additional criteria are needed.

3.4. Scaling

The role of population diversity and selective pressure in the evolutionary process is now further examined. To this purpose, it is useful to introduce a *saturation ratio* $0 \leq S \leq 1$ defined as follows (Galante 1996):

$$S = \frac{\overline{F}}{F_{\max}} \tag{12}$$

where $\overline{F}$ and $F_{\max}$ are the average and the maximum fitness over the population *X*, respectively:

$$\overline{F} = \frac{1}{m}\sum_{k=1}^{m} F(\mathbf{x}_k) \qquad\qquad F_{\max} = \max_{k=1,\ldots,m} F(\mathbf{x}_k) \tag{13}$$

Saturation is very low if the current population contains a few extremely good individuals, so that their fitness is very high with respect to the fitness of the remaining population. In this case, since reproduction happens in proportion to the fitness, the resulting population tends to contain only copies of these super-individuals and the environment cannot be adequately explored. Lower values of saturation are then associated to a loss of population diversity. On the contrary, saturation results very high if the current population consists of individuals having approximately the same fitness. In this case, the population tends to be reproduced without alterations and it is difficult to exploit the best individuals. Higher values of saturation are then associated to a loss of selective pressure. Therefore, in both the mentioned cases, the search strategy may be not effective.

To reduce such problems, reproduction should be performed only after the saturation ratio has been tuned on a proper value, f.i. $S = 0.50$. This is achieved by *scaling* the fitness function to obtain a new function $\Phi(\mathbf{x}) \geq 0$ which is used to regulate the reproduction. In the following, a *linear scaling* is adopted:

$$\Phi(\mathbf{x}) = \Phi_0 + \Phi_1 F(\mathbf{x}) \tag{14}$$

where the coefficients Φ_0 and Φ_1 are chosen as a function of the saturation S_Φ so that the average fitness is mapped on itself, i.e. $\overline{\Phi} = \overline{F}$:

$$\Phi_0 = \frac{(S_\Phi F_{\max} - \overline{F})\overline{F}}{S_\Phi (F_{\max} - \overline{F})} \qquad \Phi_1 = \frac{(1 - S_\Phi)\overline{F}}{S_\Phi (F_{\max} - \overline{F})} \tag{15}$$

In this way, reproduction happens always with reference to a proper level of saturation. However, depending on the minimum fitness $F_{\min}$ over the population *X*:

$$F_{\min} = \min_{k=1,\ldots,m} F(\mathbf{x}_k) \tag{16}$$

linear scaling may lead to negative values of the scaled fitness $\Phi(\mathbf{x})$. For this reason, S_Φ has to be set no lower than the following critical value (Shrestha & Ghaboussi 1998):

$$S_{cr} = \frac{\overline{F} - F_{\min}}{F_{\max} - F_{\min}} \tag{17}$$

which leads to a zero scaled fitness for the worst individual $\mathbf{x}_W$ in the current population, i.e. $\Phi(\mathbf{x}_W) = 0$ if $S_\Phi = S_{cr}$. Alternatively, negative values can be avoided by adopting *non-linear scaling*, f.i. *exponential* or *power scaling*. Nevertheless, linear scaling with $S_\Phi \geq S_{cr}$ is shown to be good enough in most applications (Michalewicz 1992, Haslinger & Jedelský 1996, Shrestha & Ghaboussi 1998).

3.5. Coding

Genetic algorithms work with a coded representation $s(\mathbf{x})$ of the individuals $\mathbf{x}$ rather than the individuals themselves. In particular, they are usually coded by strings of ℓ characters (genes) arranged in successive positions $k=1,...,\ell$ (loci) and having some feature values s_k (alleles):

$$s(\mathbf{x})=\{ s_1 \quad s_2 \quad ... \quad s_\ell \} \tag{18}$$

Of course, the feature values s_k may belong to alphabets A_k having different cardinality z_k (Haslinger & Jedelský 1996):

$$A_k = \{ a_1 \quad a_2 \quad ... \quad a_{z_k} \} \qquad ,k=1,...,\ell \tag{19}$$

even if the same alphabet $A=A_k$ is usually adopted for each character *k*.

In any case, a real vector $\mathbf{x}$ can be coded as a constant length string $s(\mathbf{x})$ in many different ways. In this paper, each design variable x_i is coded in a sub-string $s_i(x_i)$ of ℓ_i characters:

$$s_i(x_i)=\{ s_{i1} \quad s_{i2} \quad ... \quad s_{i\ell_i} \} \qquad ,i=1,...,n \tag{20}$$

by using the binary alphabet ($z=2$):

$$A=\{ 0 \quad 1 \} \tag{21}$$

Such sub-strings are subsequently concatenated head-to-tail to form one long string $s(\mathbf{x})$ having total length:

$$\ell=\sum_{i=1}^{n} \ell_i \tag{22}$$

In particular, a *linear coding* is adopted as follows:

$$\mathbf{x}=\mathbf{x}^- + \mathbf{C}\,(\mathbf{x}^+ - \mathbf{x}^-) \tag{23}$$

where $\mathbf{C}(s)$ is a diagonal matrix whose i^{th} diagonal term C_i is given by:

$$C_i = \frac{1}{2^{\ell_i}-1} \sum_{k=1}^{\ell_i} (2^{\ell_i - k} s_{ik}) \tag{24}$$

In this way, since each binary string $s_i(x_i)$ of length ℓ_i can represent 2^{ℓ_i} different values, the domain of the corresponding design variable x_i is implicitly subdivided into $(2^{\ell_i}-1)$ intervals of amplitude:

$$\Delta_i = \frac{x_i^+ - x_i^-}{2^{\ell_i}-1} \qquad , i=1,...,n \tag{25}$$

Therefore, if the design variables x_i are discrete and their values are equally spaced over intervals of amplitude Δ_i, the string lengths ℓ_i consistent with the adopted linear coding are evaluated as follows:

$$\ell_i \geq \log_2\left(1+\frac{x_i^+ - x_i^-}{\Delta_i}\right) \qquad , i=1,...,n \tag{26}$$

where only a part of the available discrete values have to be allowed if strict inequality holds. Finally, it is outlined that nothing is lost in generalisation if some design variables are continuous. In fact, they can always be discretised in according to the previous requirements. However, if equally spaced intervals are not consistent with the nature of the design variables, *non-linear coding* is required.

3.6. Genetic Operators

As mentioned, the reproduction of the population during the evolutionary process is regulated by means of some genetic operators. In the following, three basic operators, *selection*, *crossover* and *mutation*, are briefly described.

Selection identifies the individuals of the current population who are selected as parents to reproduce offspring during the next generation. In particular, based the principle of the survival-of-the-fittest, individuals are selected according to their fitness with the environment in such a way that the best get more copies, the average stay even and the worst die off. Several methods are available for the selection process. Here, the roulette-wheel selection method is applied. At first, the probability of selection $p_S(\mathbf{x}_k)$ for each individual k is evaluated in proportion to its scaled fitness value $\Phi(\mathbf{x}_k)$:

$$p_S(\mathbf{x}_k) = \frac{\Phi(\mathbf{x}_k)}{\sum_{k=1}^{m} \Phi(\mathbf{x}_k)} \qquad , k = 1,...,m \tag{27}$$

Subsequently, the corresponding distribution of the cumulative probability $P_S(\mathbf{x}_k)$ is derived:

$$P_{Sk}(\mathbf{x}_k) = \sum_{i=1}^{k} p_{S\,i}(\mathbf{x}_i) \qquad , k = 1,...,m \tag{28}$$

Finally, a random number $R \in [0;1]$ is generated *m* times and each time the individual *k* such that:

$$P_S(\mathbf{x}_{k-1}) \leq R \leq P_S(\mathbf{x}_k) \tag{29}$$

is selected for the next generation. In this way, as mentioned, the resulting population, whose size *m* is unchanged, tends to consist of multiple copies of higher fitness individuals, single copies of average fitness individuals and no copies of lower fitness individuals.

Crossover is the primary operator in genetic algorithms and works by selecting pairs of individuals for mating. Since this combination implies an information exchange, offspring generated by crossover are not generally simple copies of the parents. Selection for crossover is assumed to happen with a predetermined probability p_C, so that the expected individuals who undergo mating are mp_C. In particular, a random number $R \in [0;1]$ is generated *m* times and each time the corresponding individual is selected for crossover if $R \leq p_C$. Subsequently, for each couple of selected parents, depending whether single-point or multiple-point crossover is used, one or more crossover location $I = 1,...,\ell$ are randomly selected. The resulting sub-strings are then mutually interchanged to create two new offspring which must replace the parents in the next population. For example, a single-point crossover in position *k* for the following couple of parents:

$$\begin{cases} s'(\mathbf{x}) = \{\, s'_1 \quad s'_2 \quad ... \quad s'_k \mid s'_{k+1} \quad ... \quad s'_\ell \,\} \\ s''(\mathbf{x}) = \{\, s''_1 \quad s''_2 \quad ... \quad s''_k \mid s''_{k+1} \quad ... \quad s''_\ell \,\} \end{cases} \tag{30}$$

results in the following couple of offspring:

$$\begin{cases} s'(\mathbf{x}) = \{\, s'_1 \quad s'_2 \quad ... \quad s'_k \mid s''_{k+1} \quad ... \quad s''_\ell \,\} \\ s''(\mathbf{x}) = \{\, s''_1 \quad s''_2 \quad ... \quad s''_k \mid s'_{k+1} \quad ... \quad s'_\ell \,\} \end{cases} \tag{31}$$

With analogous criterion, a two-point crossover in positions *i* and *k*:

$$\begin{cases} s'(\mathbf{x}) = \{\, s'_1 \;\; \ldots \;\; s'_i \mid s'_{i+1} \;\; \ldots \;\; \ldots \;\; s'_k \mid s'_{k+1} \;\; \ldots \;\; s'_\ell \,\} \\ s''(\mathbf{x}) = \{\, s''_1 \;\; \ldots \;\; s''_i \mid s''_{i+1} \;\; \ldots \;\; \ldots \;\; s''_k \mid s''_{k+1} \;\; \ldots \;\; s''_\ell \,\} \end{cases} \tag{32}$$

leads to the following couple of offspring:

$$\begin{cases} s'(\mathbf{x}) = \{\, s'_1 \;\; \ldots \;\; s'_i \mid s''_{i+1} \;\; \ldots \;\; \ldots \;\; s''_k \mid s'_{k+1} \;\; \ldots \;\; s'_\ell \,\} \\ s''(\mathbf{x}) = \{\, s''_1 \;\; \ldots \;\; s''_i \mid s'_{i+1} \;\; \ldots \;\; \ldots \;\; s'_k \mid s''_{k+1} \;\; \ldots \;\; s''_\ell \,\} \end{cases} \tag{33}$$

and so on for higher degrees of mating. Multiple-point crossover is usually adopted because, especially for very long strings, it tends to provide a greater explorative power during the genetic search than single-point crossover. In addition, a high probability of crossover p_C is typically set to allow a large amount of information exchange among individuals before they are discarded from the population.

Mutation operates at the level of the single character by altering the corresponding feature value. In particular, using a binary alphabet, the character is mutated from 0 to 1 and vice versa. Selection for mutation is assumed to happen with a predetermined probability p_M, so that the expected characters who undergo alteration are ℓp_M for each individual, or $m\ell p_M$ over the whole population. In particular, a random number $R \in [0;1]$ is generated ℓ times for each individual and each time the corresponding character is selected for mutation if $R \leq p_M$. Although mutation is usually considered as a secondary operator, it actually gives an important contribution to the search effectiveness. In fact, mutation introduces new genetic features at the gene level, which cannot be produced by simple selection or crossover. The higher diversity so enforced leads to extend the search in the neighbourhood of the altered individuals and allows to restart the evolutionary process when a too high saturation level is reached. However, to avoid the algorithms behaving as a random search method, a low mutation probability p_M is typically set.

Of course, depending on the nature of the problem, the performances of the genetic algorithm may be improved by introducing additional genetic operators (Michalewicz 1992).

3.7. Generation Gap

The genetic operators just described lead to replace the whole population at each generation. However, depending on the population size, this may lead to either a large number of fitness evaluation, or a disruption of the current genetic structure. In both cases, effectiveness of the search process may be substantially reduced. To avoid these drawbacks, a *steady-state* approach, where only a few individuals of the population are replaced at each generation, can be used. This is achieved by applying the genetic operators to *mG* individuals rather than to the whole population, being $G \leq 1$ a *generation gap* which control the percentage of the population to be replaced (Wu & Chow 1995*b*).

3.8. The Schema Theorem

The mathematical theory of genetic algorithms has been developed by using the concept of *schema* (Holland 1975). A schema *H*(**x**) is a similarity template which represents a set of strings *s*(**x**) having the same feature values s_k at certain *fixed* positions *k*. They are usually represented by strings constructed over alphabets A_k^* having cardinalities $(z_k + 1)$:

$$A_k^* = \{ a_1 \quad a_2 \quad \dots \quad a_{z_k} \quad * \} \qquad , k = 1, \dots, \ell \tag{34}$$

where the meta character $*$ within a schema *H*(**x**) represents all the possible feature values $s_i \in A_i$ at certain *don't care* positions *i*. For example, by considering strings constructed over the binary alphabet *A*={0 1} and having length $\ell = 5$, the schema:

$$H(\mathbf{x}) = \{0 \quad * \quad 1 \quad * \quad 1\} \tag{35}$$

matches the following set of four strings:

$$\begin{cases} s(\mathbf{x}) = \{0 \quad 0 \quad 1 \quad 0 \quad 1\} \\ s(\mathbf{x}) = \{0 \quad 1 \quad 1 \quad 0 \quad 1\} \\ s(\mathbf{x}) = \{0 \quad 0 \quad 1 \quad 1 \quad 1\} \\ s(\mathbf{x}) = \{0 \quad 1 \quad 1 \quad 1 \quad 1\} \end{cases} \tag{36}$$

Of course, the schema:

$$H(\mathbf{x}) = \{ s_1 \quad s_2 \quad \dots \quad s_\ell \} \tag{37}$$

collects only one string, while the schema:

$$H(\mathbf{x})=\{*\quad *\quad \ldots\quad *\} \tag{38}$$

represents all the strings $s(\mathbf{x})$. In general, a schema $H(\mathbf{x})$ constructed over a same alphabet A^* of cardinality z matches a set of z^d different strings, being $d \leq \ell$ the number of don't care characters. Conversely, a string $s(\mathbf{x})$ of length ℓ allows for $(z+1)^\ell$ different schemata, or is matched by z^ℓ different schemata. Thus, genetic algorithms implicitly process schemata whose number, for a population of size m, is between z^ℓ and mz^ℓ.

The *fitness* $F(H, t)$ of a schema $H(\mathbf{x})$ is defined as the average fitness of the $N(H,t)$ individuals $\mathbf{x}$ of the population matched by the schema at the generation t. Since selection operates according to the fitness distribution, in the next generation, an above average schema tends to receive an increasing number of individuals, a below average schema tends to receive a decreasing number of individuals and an average schema tends to stay on the same level. However, the evolutionary process is also affected by other properties of schemata. The *order* $o(H)$ of the schema $H(\mathbf{x})$ is the number of the fixed positions present in the schema, while its *defining length* $d(H)$ is the distance between the outermost fixed positions. Since crossover and mutation modify the genetic structure of the individuals, the probability of survival of a schema during evolution depends on its defining length and order, respectively. In particular, a lower bound on the expected number of individuals matched by a particular schema during the generation $(t+1)$ is given by (Goldberg 1989):

$$N(H,t+1) \geq N(H,t)\frac{F(H,t)}{\bar{F}}\left[1-p_C\frac{d(H)}{\ell-1}-p_M\,o(H)\right] \tag{39}$$

This inequality represents the fundamental law of the genetic algorithms and leads to formulate the following Schema Theorem: *short, low-order, above-average schemata receive an exponentially increasing number of individuals in subsequent generations of a genetic algorithm*. Schemata with these properties are usually called building blocks and the tendency to allocate exponentially increasing numbers of individuals to building blocks in parallel is the essence of the genetic search (*implicit parallelism*).

The Schema Theorem clarifies the influence of coding on the effectiveness of the search. Of course, the binary alphabet, besides leading to the maximum number of schemata per bit of information (Dhingra & Lee 1994), allows for the shortest building blocks.

3.9. Termination Criteria

The flow chart of Figure 1 briefly resumes the steps of the solution procedure. Of course, the genetic algorithm may be stopped when a predetermined number of generations t_∞ is reached:

$$t \le t_\infty \tag{40}$$

However, this criterion requires some a priori information about the expected length of the search, which is not usually available. For this reason, two additional criteria, which implicitly consider the chance for significant improvements, are considered too (Michalewicz 1992, Grierson & Pak 1993).

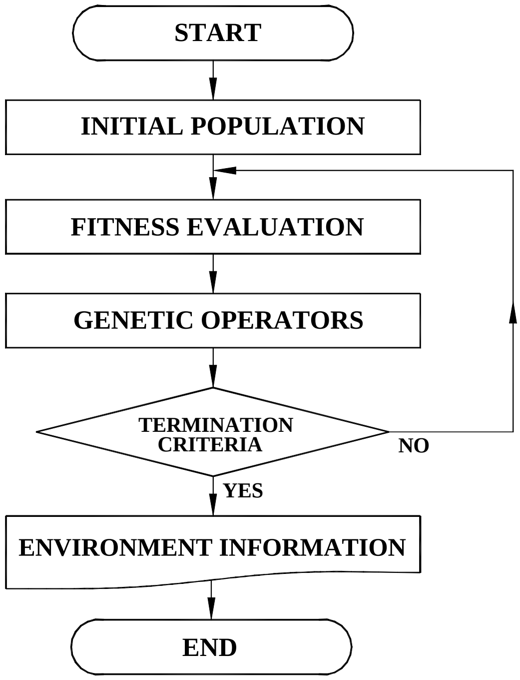


Figure 1 - Flow chart of a genetic algorithm.

The first one is based on the structure of the individuals (genotype) and measures the convergence of the population towards the optimum by checking the number of converged characters N_C, where a character is considered converged if it assumes the same feature value of the corresponding character of the current best individual. According to this criterion, if the number of converged characters N_C exceeds a predetermined percentage C over the population, the search process is terminated:

$$N_C \le m\ell C \tag{41}$$

The second criterion is based on the meaning of a particular individual (phenotype) and measures the convergence of the population by checking the progress made with respect to the current best individual in a predefined number of generations t_1. If no progress exists over such an evolutionary interval, the search process is terminated:

$$t - t_0 \leq t_1 \tag{42}$$

where t_0 is the last improvement generation.

The previous termination criteria can also be associated to an *elitist* strategy (Grierson & Pak 1993), where the selection procedure is modified to enforce the preservation of the best individual at each generation (Michalewicz 1992). In this way, the convergence is always checked with respect to the maximum fitness individual over the whole evolutionary process.

4. ANALYSIS AND DESIGN OF R.C. FRAMES

4.1. Structural Model and Non-Linear Analysis

In most cases, R.C. structures should be analysed by taking material and, eventually, geometrical non-linearities into account if realistic results under all load levels are needed. In this work, a two-dimensional structure is modelled using a R.C. beam finite element whose formulation, based on the Bernoulli-Navier hypothesis, deals with such kinds of non-linearities (Bontempi *et al.* 1995, Malerba 1998). In particular, both material $\mathbf{K}'_M$ and geometrical $\mathbf{K}'_G$ contributes to the element stiffness matrix $\mathbf{K}'$ and the nodal forces vector $\mathbf{f}'$, equivalent to the applied loads $\mathbf{f}'_0$, are derived by applying the principle of the virtual displacements and then evaluated by numerical integration over the length *l* of the beam:

$$\mathbf{K}' = \mathbf{K}'_M + \mathbf{K}'_G \qquad \mathbf{K}'_M = \int_0^l \mathbf{B}^T \mathbf{H} \mathbf{B} dx \qquad \mathbf{K}'_G = \int_0^l N\, \mathbf{G}^T \mathbf{G}\, dx \tag{43}$$

$$\mathbf{f}' = \int_0^l \mathbf{N}^T \mathbf{f}'_0\, dx \tag{44}$$

$$\mathbf{N} = \begin{bmatrix} \mathbf{N}_a & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_b \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} \partial \mathbf{N}_a / \partial x & \mathbf{0} \\ \mathbf{0} & \partial^2 \mathbf{N}_b / \partial x^2 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} \mathbf{0} & \dfrac{\partial \mathbf{N}_b}{\partial x} \end{bmatrix} \tag{45}$$

where N is the axial force and $\mathbf{N}$ is a matrix of axial $\mathbf{N}_a$ and bending $\mathbf{N}_b$ displacement functions. In the following, the shape functions of a linear elastic beam element having uniform cross-sectional stiffness **H** and loaded only at its ends are adopted (Przemieniecki 1968). However, due to material non-linearity, the cross-sectional stiffness distribution along the beam is non uniform even for prismatic members with uniform reinforcement. Thus, **H** has to be computed for each section by integration over the area of the composite element, or by assembling the contributes of both concrete $\mathbf{H}_c$ and steel $\mathbf{H}_s$:

$$\mathbf{H} = \int_A \mathbf{D}\, dA = \int_{A_c} \mathbf{D}_c\, dA_c + \sum_k \mathbf{D}_{sk} A_{sk} = \mathbf{H}_c + \mathbf{H}_s \tag{46}$$

being A_c, A_{sk}, the areas of the concrete and of the k^{th} reinforcement bar and $\mathbf{D}_c, \mathbf{D}_{sk}$, the corresponding matrixes which account for both stiffness and geometrical location of the materials. In this way, after the constitutive laws for concrete and steel are specified, the matrix **H** of each section can be computed under all load levels.

The equilibrium conditions of the beam element are derived from the already mentioned principle of the virtual work. Thus, by assembling the stiffness matrix **K** and the vectors of the nodal forces **f** with reference to a global coordinate system, the equilibrium of the whole structure can be formally expressed as follows:

$$\mathbf{Ks} = \mathbf{f} \tag{47}$$

where **s** is the vector of the nodal displacements. It is worth noting that the vectors **f** and **s** have to be considered as total or incremental quantities depending on the nature of the stiffness matrix **K**=**K**(**s**), or if a secant or a tangent formulation is adopted.

4.2. Limit States and Load Multipliers

Based on the general concepts of the R.C. design, the structural performances should generally be described with reference to a specified set of limit states, as regards both serviceability and ultimate conditions, which separate desired states of the structure from undesired ones (Bontempi *et al.*, 1998*b*).

Splitting cracks and considerable creep effects may occur if the compression stresses s_c in concrete are too high. Besides, excessive stresses s_s in reinforcing steel can lead to unacceptable crack patterns. The following limitations on the stress level in the materials, account for adequate durability at the serviceability stage:

$$-s_c \le -a_c f_c \qquad |s_s| \le a_s f_{sy} \tag{48.a}$$

where a_c and a_s are suitable reduction factors of the limit strengths f_c and f_{sy}. Excessive displacements **s** may also involve loss of serviceability and then have to be limited within assigned bounds $\mathbf{s}^-$ and $\mathbf{s}^+$:

$$\mathbf{s}^- \leq \mathbf{s} \leq \mathbf{s}^+ \tag{48.b}$$

When the strain in concrete e_c or in reinforcing steel e_s reaches a limit value e_{cu} or e_{su}, the collapse of the corresponding cross-section occurs. So, the following ultimate conditions have to be verified:

$$-e_c \leq -e_{cu} \qquad\qquad |e_s| \leq e_{su} \tag{49.a}$$

However, the collapse of a single cross-section doesn't necessarily lead to the collapse of the whole structure, the latter is caused by the loss of equilibrium arising when the reactions **r** requested for the loads **f** can no longer be developed. This condition can be formally stated as follows:

$$\mathbf{f} \leq \mathbf{r} \tag{49.b}$$

Since these limit states refer to internal quantities of the system, a check of the structural performance through a non-linear analysis needs to be carried out at the load level. To this aim, it is useful to assume:

$$\mathbf{f} = \mathbf{g} + l\,\mathbf{q} \tag{50}$$

where **g** is a vector of dead loads and **q** a vector of live loads whose intensity varies proportionally to a unique multiplier $l \geq 0$. With this position, the safe domains at both serviceability and ultimate states can be formulated as follows:

$$S = \left\{\ l \ \middle|\ -s_c \leq -a_c f_c,\ \ |s_s| \leq a_s f_{sy},\ \ \mathbf{s}^- \leq \mathbf{s} \leq \mathbf{s}^+ \ \right\} \tag{51.a}$$

$$U = \{\ l \ |\ -e_c \leq -e_{cu},\ \ |e_s| \leq e_{su},\ \ \mathbf{f} \leq \mathbf{r}\ \} \tag{51.b}$$

and the previous limit conditions can be synthetically restated as:

$$l \leq l_S = \max_{l \in S} l \tag{52.a}$$

$$l \leq l_U = \max_{l \in U} l \tag{52.b}$$

being l_S and l_U the limit multipliers which define the failure loads.

5. APPLICATION

The presented procedure is applied to the optimal design of the two-bay one-story R.C. frame shown in Figure 2, with $L = 8.00\,\text{m}$ and $H = 6.00\,\text{m}$.

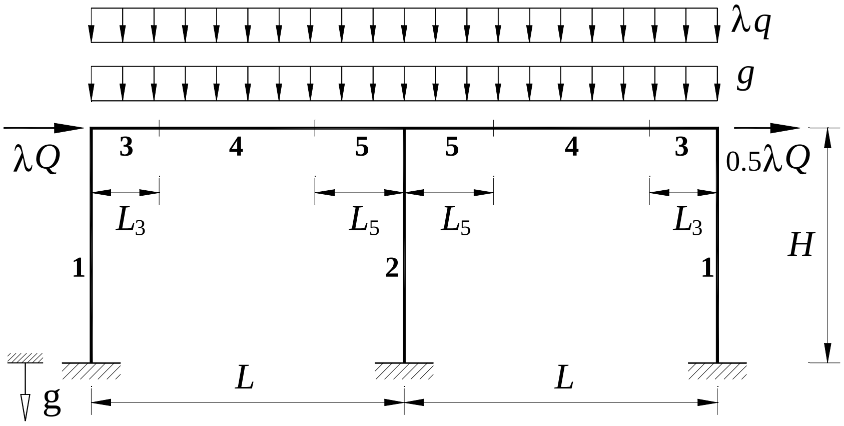


Figure 2. Two-bay one-story R.C. frame.

The non-linear constitutive laws adopted for concrete and steel are (Figure 3):

$$s_c = \begin{cases} \dfrac{kh - h^2}{1 + (k-2)h} & , e_{cu} \le e_c \le 0 \\ E_{c0} e_c & , 0 < e_c \le e_{cr} \\ f_{ct} & , e_{cr} < e_c \le e_{ctu} \end{cases} \qquad s_s = \begin{cases} E_s e_s & , |e_s| \le e_{sy} \\ f_{sy} \dfrac{e_s}{|e_s|} & , e_{sy} < |e_s| \le e_{su} \end{cases} \tag{53.a}$$

being:

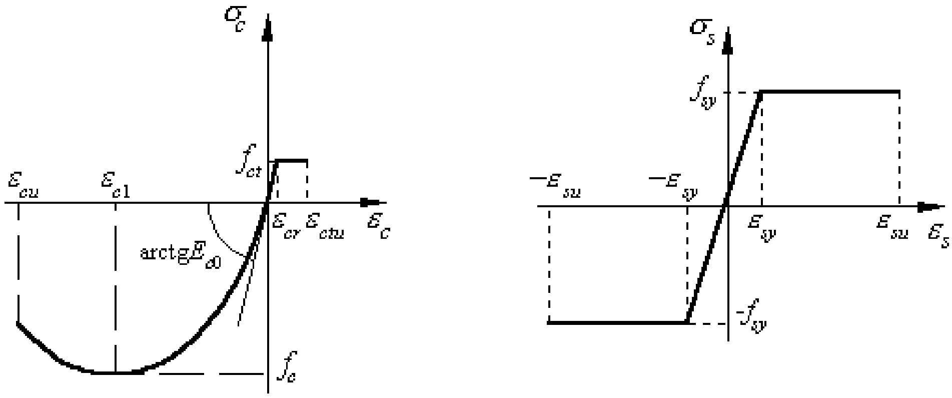
$$k = \frac{E_{c0} e_{c0}}{f_c} \qquad h = \frac{e_c}{e_{c0}} \qquad e_{cr} = \frac{f_{ct}}{E_{c0}} \qquad e_{sy} = \frac{f_{sy}}{E_s} \tag{53.b}$$

Such diagrams are completely defined by the following material properties:

$$f_c = -40\,\text{MPa} \qquad f_{ct} = 2.9\,\text{MPa} \qquad E_{c0} = 35\,\text{GPa} \tag{54.a}$$

$$e_{c0} = 2‰ \qquad e_{cu} = 3.5‰ \qquad e_{ctu} = 2e_{cr} \tag{54.b}$$

$$f_{sy} = 500\,\text{MPa} \qquad E_s = 210\,\text{GPa} \qquad e_{su} = 1\% \tag{54.c}$$



(a) (b)

Figure 3 - Stress-strain diagrams of the materials: (a) concrete; (b) steel.

Besides the effect of gravity g (weight density $g = 25$ kN/m^3), the frame is subjected to the distribution of dead and live loads in Figure 2, with:

$$g = 5\ \text{kN/m} \qquad q = 10\ \text{kN/m} \qquad Q = 20\ \text{kN} \tag{55}$$

In particular, the intensities of the live loads vary proportionally to a unique load multiplier l.

Five different cross-sections, arranged as shown in Figure 2, are adopted. The cross-sections are rectangular shaped and reinforced as follows: four steel bars at the corners for sections 1 and 2, two steel bars at the top and four steel bars at the bottom for section 4, four steel bars at the top and two steel bars at the bottom for the sections 3 and 5. In each section, the steel bars are placed at 50 mm from the nearest edge. Let b_i, h_i, the width and the height of the cross-section $i = 1,...,5$, respectively, and f_i^{top}, f_i^{bot}, the diameter of the top and the bottom reinforcing bars in such sections, respectively. By assuming:

$$b = b_1 = b_2 = b_3 = b_4 = b_5 \qquad h = h_3 = h_4 = h_5 \qquad f_1 = f_1^{top} = f_1^{bot} \qquad f_2 = f_2^{top} = f_2^{bot} \tag{56}$$

a total of $n = 14$ design variables are considered:

$$\mathbf{x} = [\, b \quad h \quad h_1 \quad h_2 \quad f_1 \quad f_2 \quad f_3^{top} \quad f_3^{bot} \quad f_4^{top} \quad f_4^{bot} \quad f_5^{top} \quad f_5^{bot} \quad L_3 \quad L_5 \,]^T \tag{57}$$

being L_3 and L_5 the length of the parts of the beams having section 3 and 5, respectively (Figure 2).

i	x_i	x_i^- [mm]	x_i^+ [mm]	Δ_i [mm]	ℓ_i [–]	x_{opt} [mm]
1	b	200	500	20	4	280
2	h	300	920	20	5	520
3	h_1	300	600	20	4	300
4	h_2	300	600	20	4	320
5	f_1	12	26	2	3	18
6	f_2	12	26	2	3	20
7	f_3^{top}	12	26	2	3	20
8	f_3^{bot}	12	26	2	3	12
9	f_4^{top}	12	26	2	3	12
10	f_4^{bot}	12	26	2	3	16
11	f_5^{top}	12	26	2	3	20
12	f_5^{bot}	12	26	2	3	16
13	L_3	350	3500	50	6	2950
14	L_5	350	3500	50	6	2150

Table 1. Design variables x_i, lower x_i^- and upper x_i^+ side constraints, length ℓ_i of the binary coded string $s_i(x_i)$ and optimal values x_{opt}.

Besides the side constraints listed in Table 1, the following behavioural constraints are imposed:

$$l_S \geq 1 \tag{58.a}$$

$$l_U \geq 2 \tag{58.b}$$

$$b \leq h_i \leq 2b \tag{58.b}$$

$$0.5\% \leq r_i \leq 6\% \quad , i=1,...,5 \tag{58.b}$$

where $r_i(\mathbf{x})$ is the geometric ratio of the reinforcement of the section *i* and the serviceability limit multiplier $l_S(\mathbf{x})$ is defined by assuming:

$$a_c = 0.45 \qquad a_s = 0.80 \qquad s^+ = -s^- = 25\,\text{mm} \tag{59}$$

The objective function $f(\mathbf{x})$ is defined by adopting a unit cost ratio $c=20$, while the penalty function $p(\mathbf{x})$ is constructed by choosing $p_1 = p_2 = 100$. The structural analyses needful to the evaluation of the penalised objective function $j(\mathbf{x})$ are performed by taking mechanical and geometrical non-linearities into account. The frame is discretised by 17 composite beam elements having lengths $l = 2.00\ \mathrm{m}$. The area of each concrete cross section is subdivided into 3 rectangular sub-domains having the same height. The numerical integration over the composite cross sections are then carried out in each sub-domain by using a 4 points Gauss-Legendre integration rule. The discrete contributes of the areas of the steel bars are directly added to the contribute of the area of concrete. The numerical integration along each element is instead performed by using a 7 points Gauss-Lobatto integration rule. The loads are incrementally applied until collapse by assuming $\Delta l = 0.1$.

An optimal solution is achieved by means of a genetic algorithm according to the following genetic properties:

$$m=50 \qquad \ell=53 \qquad S=0.50 \tag{60.a}$$

$$p_C=0.80 \qquad p_M=0.01 \qquad G=0.10 \tag{60.b}$$

$$C=0.80 \qquad t_1=300 \qquad t_\infty=1000 \tag{60.c}$$

Figure 4 shows the evolution of the objective function $f(\mathbf{x})$ and of its penalised value $j(\mathbf{x}) \geq f(\mathbf{x})$ with respect to the best individual of the population, while Figure 5 shows the evolution of the actual level of saturation S. The optimal value $f(\mathbf{x}_{opt}) = j(\mathbf{x}_{opt}) = 4.666\ \mathrm{m}^3$ is reached at the generation $t_{opt} = 902$, with the optimal values of the design variables listed in Table 1. The five optimal cross-sections are also shown in Figure 6. It is outlined that the optimal design complies with all the imposed behavioural constraints. In particular, serviceability and ultimate limit states are associated to the following limit multipliers:

$$l_S = 1.0 \tag{61.a}$$

$$l_U = 2.7 \tag{61.b}$$

The limit condition $l = l_S$ is also highlighted on the load-displacement diagram at the top of the right column shown in Figure 7.

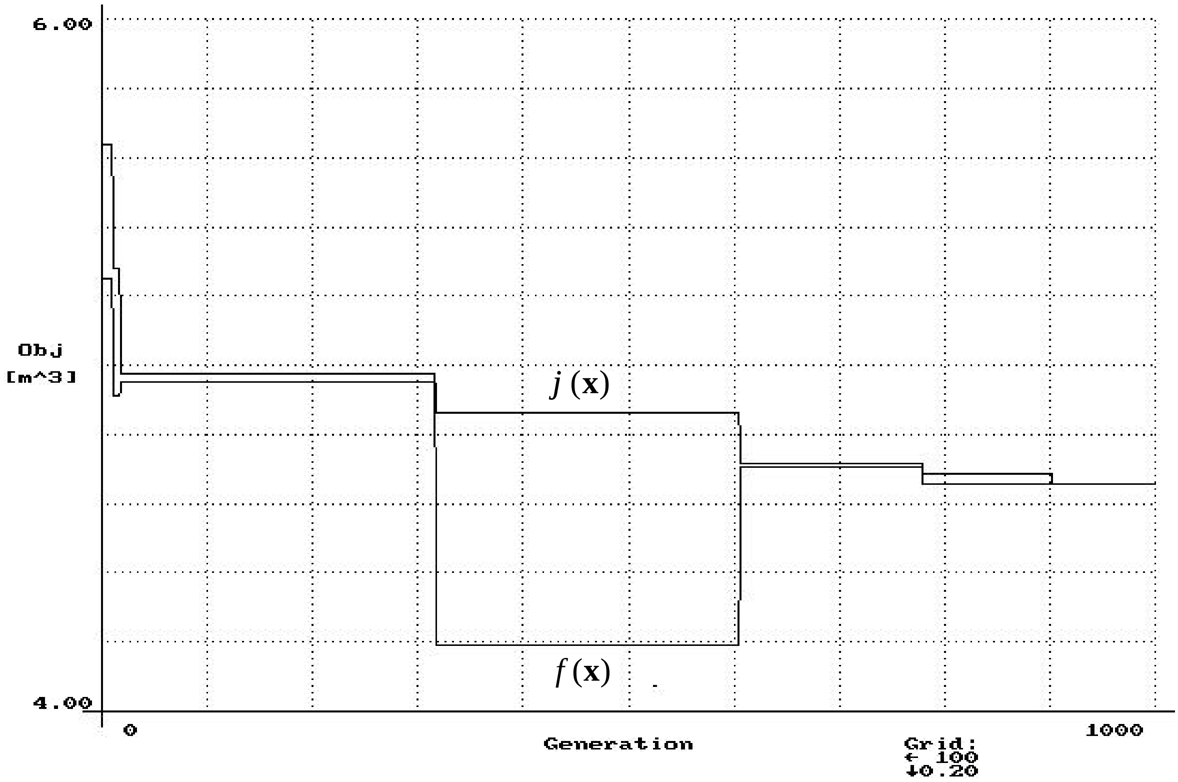


Figure 4. Evolution of the objective function $f(\mathbf{x})$ and of its penalised value $j(\mathbf{x}) \geq f(\mathbf{x})$ for the best individual in the population.

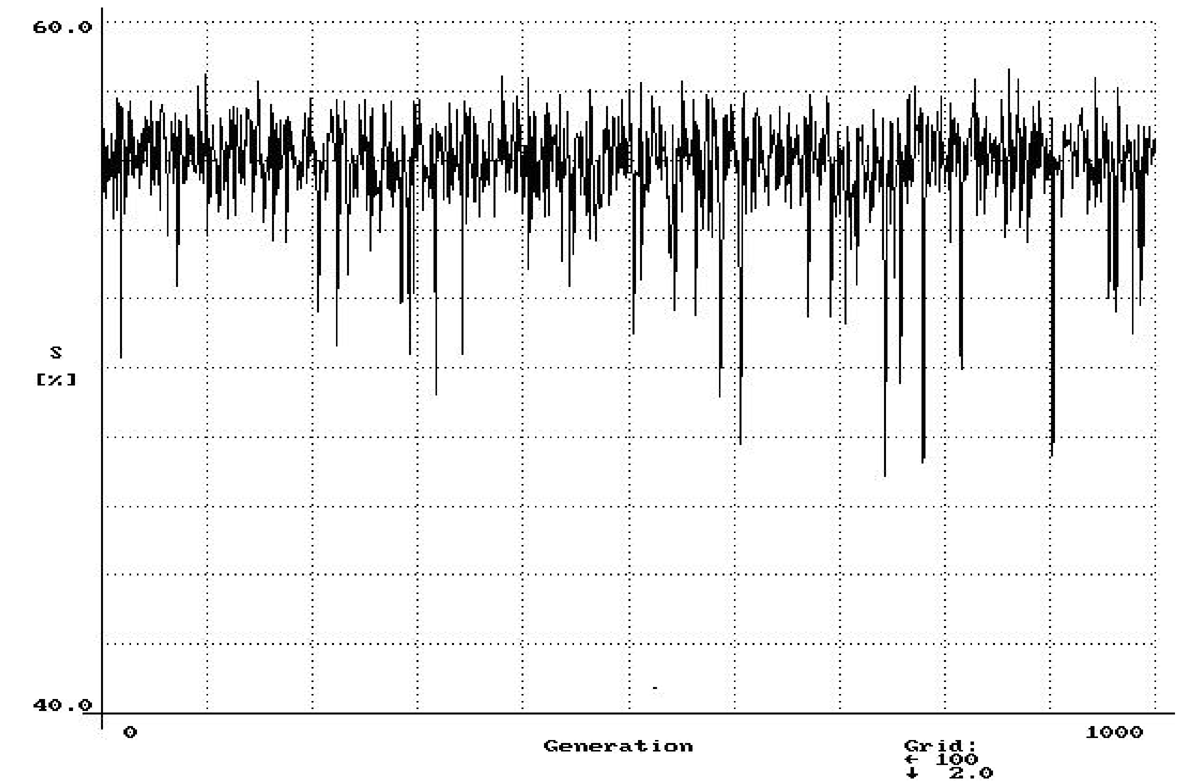


Figure 5. Evolution of the actual saturation ratio *S*.

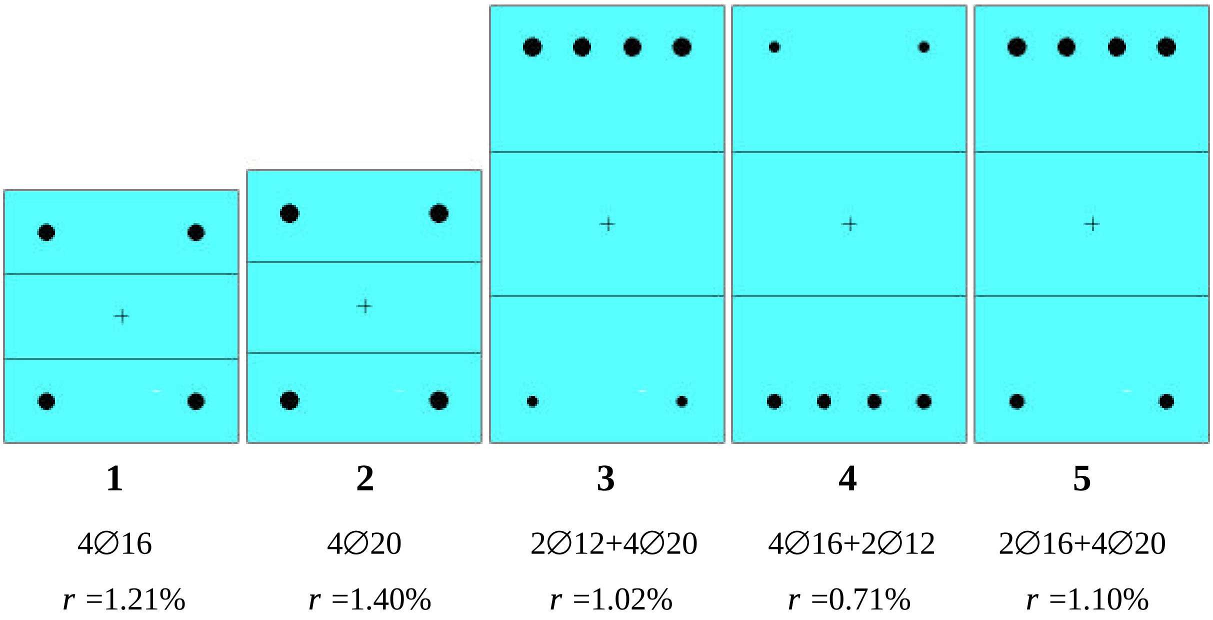


Figure 6. Optimal cross sections.

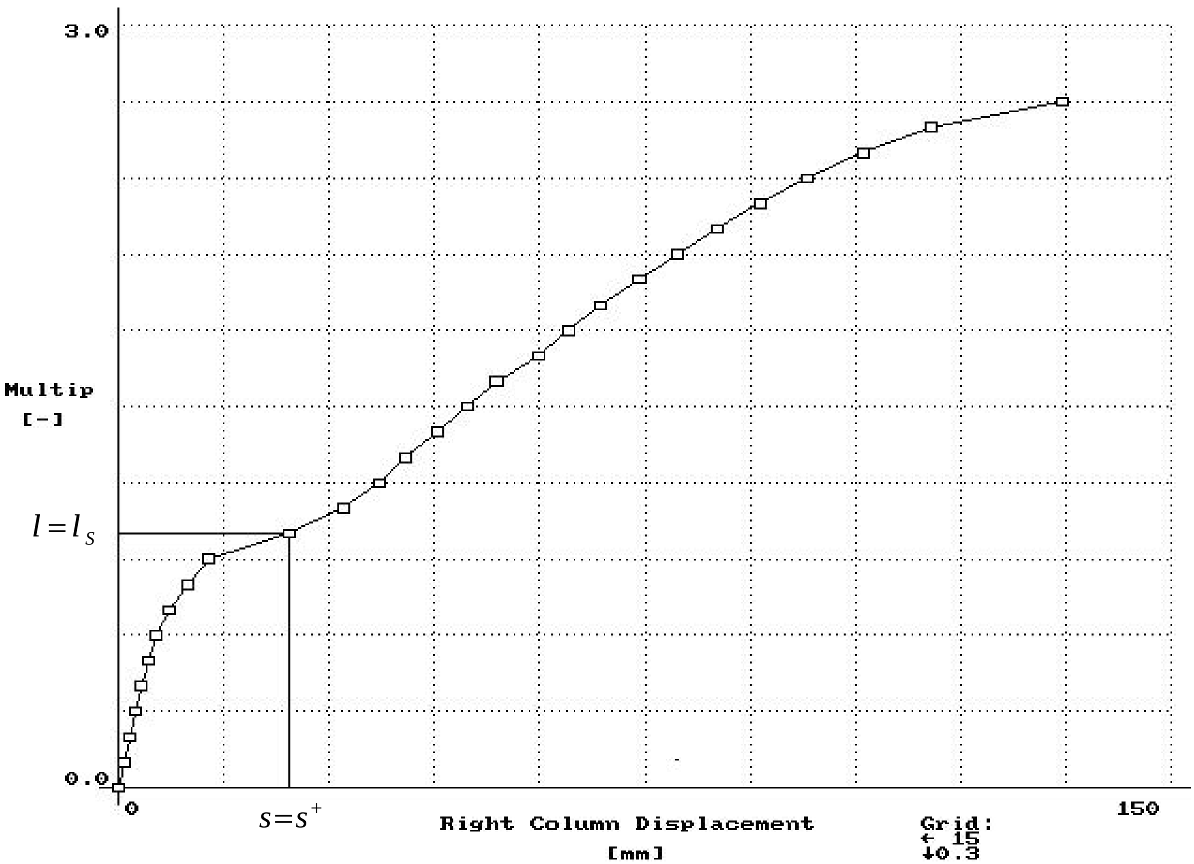


Figure 7. Load multiplier-lateral displacement diagram at the top of the right column for the optimal solution.

6. CONCLUSIONS

Genetic algorithms, non deterministic heuristic search techniques based on some analogies with the principles of natural selection of biological systems, have been applied to the optimal limit states design of concrete structures. The main differences between genetic algorithms and more traditional search methods can be summarised in the following points.

- *Population*: genetic algorithms process contemporarily a population of design points rather than a single point. This higher exploration of the search space gives more chances to avoid local optimum.
- *Fitness*: genetic search needs only objective function values, while derivatives and other auxiliary information are not requested. Thus, non differentiable objective functions can be easily handled.
- *Coding*: genetic algorithms work with a coded representation of the design variables rather than the design variables themselves. In this way, discrete design variables, as well as continuous variables with specified precision, can be considered.
- *Randomised operators*: genetic search evolves by means of stochastic rules rather than through deterministic paths. A higher robustness is then pursued during the exploitation of the best solution.
- *Schemata*: genetic algorithms implicitly process schemata, or higher level information branched into the population. This leads to an implicit parallelism of the search which assures a high effectiveness.

The optimisation of a R.C. frame has been considered. The shape and the dimensions of the concrete cross-sections as well as the amount and location of the reinforcement have been adopted as design variables. The minimisation of the structural cost of the system has been chosen as main objective of the design process, with respect to side and behavioural constraints. In particular, by taking the actual non-linear structural behaviour into account, both serviceability and ultimate limit states have been assumed as design constraints. The results show the effectiveness of the procedure and highlight the utility of the optimisation methods in practical design problems.

ACKNOWLEDGEMENTS

The author wishes to gratefully acknowledge Prof. Franco Bontempi and Prof. Pier Giorgio Malerba for their valuable advice. The present work is supported by MURST 97 research funds of the University of Udine.

APPENDIX I - REFERENCES

Adamu, A., Karihaloo, B.L. 1995. Minimum Cost Design of RC Frames Using the DCOC Method. Part I: Columns under Uniaxial Bending Actions. Part II: Columns Under Biaxial Bending Actions. *Structural Optimization*, **10**: 16-39.

Biondini, F. 1998. Optimal Prestressing in Framed Structures. *Studi e Ricerche*, Scuola di Specializzazione in Costruzioni in Cemento Armato, Politecnico di Milano, **19**: 59-98 (in Italian).

Bontempi, F., Malerba, P.G. & Romano, L. 1995. A Direct Secant Formulation for the Reinforced and Prestressed Concrete Frames Analysis. *Studi e Ricerche*, Scuola di Specializzazione in Costruzioni in Cemento Armato, Politecnico di Milano, **16**: 351-386 (in Italian).

Bontempi, F., Biondini, F., Malerba, P.G. 1998*a*. Thickness Optimization of Plane Prestressed Concrete Elements. *Structural Optimization*. in Steven, G.P., Querin, O.M., Guan, H. & Xie, Y.M. (Eds.), Sydney: Oxbridge, 285-292.

Bontempi, F., Biondini, F. & Malerba, P.G. 1998*b*. Reliability Analysis of Reinforced Concrete Structures Based on a Monte Carlo Simulation. *Stochastic Structural Dynamics*. Spencer, B.F. Jr, Johnson E.A. (Eds.), Rotterdam: Balkema, 413-420.

Chong, E.K.P., Zak, S.H. 1996: *An Introduction to Optimization*. New York: John Wiley & Sons.

Cohn, M.Z., MacRae, J. 1984, Optimization of Structural Concrete Beams. *ASCE Journal of Structural Engineering*, **110**(7): 1573-1588.

Cohn, M.Z., Dinovitzer, A.S. 1994. Application of Structural Optimizaton. *ASCE Journal of Structural Engineering*, **120**(2): 617-650.

Cohn, M.Z. 1994. Theory and Practice of Structural Optimization. *Structural Optimization*, 7: 20-31.

Darwin, C. 1859. *On the Origin of Species by means of Natural Selection or the Preservation of Favored Races in the Struggle for Life* (*The Origin of Species*). London: Murray.

Dhingra, A.K., Lee, B.H. 1994. A Genetic Algorithm Approach to Single and Multiobjective Structural Optimization with Discrete-Continuous Variables. *Interantional Journal for Numerical Methods in Engineering*, **37**: 4059-4080.

Galante, M. 1996. Genetic Algorithms as an Approach to Optimize Real-World Trusses. *Interantional Journal for Numerical Methods in Engineering*, **39**: 361-382.

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

Grierson, D.E., Pak, W.H. 1993. Optimal Sizing, Geometrical and Topological Design Using a Genetic Algorithm. *Structural Optimization*, **6**: 151-159.

Han, S.H., Adamu, A. & Karihaloo, B.L. 1996. Minimum Cost Design of Multispan Partially Prestressed Concrete Beams Using DCOC. *Engineering Optimization*, **26**: 35-59.

Haslinger, J., Jedelský, D. 1996. Genetic Algorithms and Fictitious Domain Based Approaches in Shape Optimization. *Structural Optimization*, **12**: 257-264.

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

Jenkins, W.M. 1991*a*. Structural Optimisation with the Genetic Algorithm. *The Structural Engineer*, **69**: 418-422.

Jenkins, W.M. 1991*b*. Towards Structural Optimization Via the Genetic Algorithm. *Computers & Structures*, **40**(5): 1321-1327.

Jenkins, W.M. 1992. Plane Frame Optimum Design Environment Based on Genetic Algorithms. *ASCE Journal of Structural Engineering*, **118**(11): 3103-3112.

Kanagasundaram, S., Karihaloo, B.L. 1990. Minimum Cost Design of Reinforced Concrete Structures. *Structural Optimization*, **2**: 173-184.

Kirsch, U. 1983. Multilevel Optimal Design of Reinforced Concrete Structures. *Engineering Optimization*, **6**: 207-212.

Kirsch, U. 1993. *Structural Optimization: Fundamentals and Applications*. Heidelberg: Springer.

Malerba, P.G. (Ed.) 1998. *Limit and Nonlinear Analysis of Reinforced Concrete Structures*. Udine: CISM (in Italian).

Michalewicz, Z. 1992. *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin: Springer.

Ohsaki, M. 1995. Genetic Algorithm for Topology Optimization of Trusses. *Computers & Structures*, **57**(2): 219-225.

Powell, M.J.D. 1983. Variable Metric Methods for Constrained Optimization. *Mathematical Programming: The State of the Art*, (Eds. A. Bachem, M.Grötschel and B. Korte), Berlin: Springer, 288-311.

Przemieniecki, J.S. 1968. *Theory of Matrix Structural Analysis*. New York: McGraw Hill.

Rajan, S.D. 1995. Sizing, Shape and Topology Design Optimization of Trusses Using Genetic Algorithms. *ASCE Journal of Structural Engineering*, **121**(10): 1480-1487.

Rajeev, S., Krishnamoorthy, C.S. 1992. Discrete Optimization of Structures Using Genetic Algorithms. *ASCE Journal of Structural Engineering*, **118**(5): 1233-1250.

Rajeev, S., Krishnamoorthy, C.S. 1997. Genetic Algorithms-Based Methodologies for Design Optimization of Trusses. *ASCE Journal of Structural Engineering*, **123**(3): 350-358.

Rao, S. S. 1996. *Engineering Optimization. Theory and Practice*. New York: John Wiley & Sons.

Shrestha, S.M., Ghaboussi, J. 1998. Evolution of Optimum Structural Shapes Using Genetic Algorithms. *ASCE Journal of Structural Engineering*, **124**(11): 1331-1338.

Steven, G.P., Querin, O.M., Guan, H. & Xie Y.M. (Eds.) 1998. *Structural Optimisation*. Sydney: Oxbridge.

Sven, G., Schmid, L.J. 1993. Parametric Studies with Genetic Algorithms. *Computational Mechanics*, (Eds. S. Valliappan, V. Pulmano & F. Tin-Loi), Rotterdam: Balkema, 987-990.

Wu, S.-J., Chow, P.-T. 1995*a*. Integrated Discrete and Configuration Optimization of Trusses Using Genetic Algorithms. *Computers & Structures*, **55**(4): 695-702.

Wu, S.-J., Chow, P.-T. 1995*b*. Steady-State Genetic Algorithms for Discrete Optimization of Trusses. *Computers & Structures*, **56**(6): 979-991.

APPENDIX II – PROGETTO OTTIMALE AGLI STATI LIMITE DI STRUTTURE IN CALCESTRUZZO MEDIANTE ALGORITMI GENETICI

Si presenta una procedura per il progetto ottimale di strutture in calcestruzzo soggette a carichi di tipo statico, con riferimento a stati limite di esercizio e ultimi. La procedura è orientata al progetto ottimale di telai in C.A., ma può essere impiegata anche per altri tipi di strutture. Si assumono come variabili di progetto sia la forma e le dimensioni della sezione di calcestruzzo, sia il quantitativo e la posizione delle armature. L'obiettivo del processo progettuale è quello di minimizzare il costo strutturale del sistema nel rispetto di limiti sia sulle variabili di progetto, sia sul comportamento strutturale. La soluzione ottimale è ottenuta mediante algoritmi genetici, ovvero mediante tecniche di ricerca di tipo euristico e non deterministico la cui formulazione si basa su alcune analogie con i principi che regolano la selezione naturale dei sistemi biologici. Le analisi strutturali necessarie per il processo di soluzione sono eseguite tenendo conto delle non linearità meccaniche associate al comportamento dei materiali e delle non linearità geometriche associate agli effetti del secondo ordine. Il metodo di analisi si basa sulla tecnica degli elementi finiti e fa uso di un elemento di trave composito. L'applicazione della procedura di ottimizzazione ad un telaio in C.A. mostra le potenzialità della procedura proposta.