A modified elitist genetic algorithm applied to the design optimization of complex steel structures.

María Belén Prendes Gero[a]*, Antonio Bello García[a], Juan José del Coz Díaz[b]

[a] Construction and Production Engineering Department, Oviedo University, Mieres Campus, 33600 Mieres, Spain
[b] Construction and Production Engineering Department, Oviedo University, Viesques Departmental Building nº 6, 33204 Gijón, Spain
* Corresponding author. E-mail address: belen@mieres.uniovi.es

Abstract.

This paper shows the implementation of an elitist genetic algorithm that, when applied to steel structures, is able to obtain structural elements with minimum weight and satisfy the safety factors or coefficients (Ultimate Limit States) of the applicable building code. To this end, a modified objective function has been defined that considers the constraints established by these coefficients. In addition, the codification of the design variables has been modified so that all of them have the same probability of initial selection; a selection operator has been implemented to consider the dispersion of the individuals within the population as well as a crossover operator that interchanges the sections assigned to the structural elements without their prior modification. The final result is a robust genetic algorithm that is simple from a mathematical point of view and is able to work with complex structures under different load and constraint conditions.

Key words: Genetic algorithm, elitist, steel structures, minimum weight, ultimate limit states.

1. Introduction.

The continuous development of genetic algorithms (GA) has led to the obtaining of so-called elitist GA[1], the aim of which is to avoid the best individual of a population failing to obtain offsprings within the following generation. To do so, they copy the best individual from the present population in the new one, normally achieving a speed increase in the obtaining of the optimal individual.

The elitist GA implemented in this study does not save a single individual, but a percentage of the best individuals according to the elite probability, thus achieving a greater speed of convergence.

This algorithm is applied to steel structures, with the aim of obtaining individuals of minimum weight in the structural elements, but which fulfill the safety factors established by the selected building code.

With this aim in mind, a modified objective function has been defined that considers the constraints of these coefficients. In addition:

- The *codification of the design variables* has been modified, achieving the same probability of initial selection in all of these.
- A *selection operator* denominated *aptitude* has been implemented, considering the dispersion of the individuals in the population.
- A crossover operator denominated *phenotype crossover* has been developed that only exchanges the section type assigned to the structural elements, without modifying it previously.

Besides, the design of a two-dimensional articulated structure was implemented in order to test the performance of the developed elitist algorithm and the suggested modifications. The structure (Fig. 1) with seven nodes and ten beams collected in five groups (collection of beams with the same section), is subjected to three loads of 40 tn on the nodes 2, 3 and 4. The steel properties used are: modulus of elasticity = 20,58 kN/m^2, specific weight = 76,44 kN/m^3 and yield stress = 254800 kN/m^2. In addition, this study has produced to the modification of the penalty factor (see 2.4.1).

2. Elitist genetic algorithm.

2.1. *Encoding the design variables*.

In structural optimization, the tendency is to use as design variables the section of the structural elements codified by means of chains of bits, whose length is evaluated using Eq. 1.

$$\lambda = 2^n \tag{1}$$

where λ is the number of sections of the commercial catalogue represented and n the integer of bits needed to represent λ sections of this one.

Very often, the integer deduced from Eq. 1, n, enables us to represent more sections than those existing in the catalogue, meaning that integers will exist with empty positions. In order to avoid this situation, these positions are filled up by the first sections of the catalogue, yielding a different initial selection probability in these.

The codification of design variables implemented in this work[2] means that all the sections have the same probability of initial selection. This codification is represented in Fig. 2, where the lengths of chain of n bits present n_{pos} possible sections as opposed to the n_{ex} existing sections in the commercial catalogue.

The proposed solution consists in checking whether the number assigned randomly to each design variable has an empty position assigned to it or, on the contrary, it has a real section assigned to it. In the former case, the design variable with an empty position is generated again until a real section is assigned to it, while in the latter case this section will now become part of the population.

In this way, it not only assures that all the sections assigned to the structural elements belong to the catalogue of commercial sections but also that all of them have the same probability of initial selection.

The sections randomly assigned to the design variables of the example (Fig. 1) are selected amongst 2835 sections from the Spanish building code, NBE EA-95[3]. This number produces chains of 12 bits with 4096 possible sections and 1261 empty positions.

This elevated number of sections, 2835, can produce unfeasible structures from a constructuve point of view and a major number of evaluations. But it is compensated with numerous viable and different designs where the designer will be able to select the most economic or the easiest to construct. Also it is compensated by the use of sections that habitually are not considered in some positions and nevertheless have enough resistance for them.

On the other hand, the designer selects the sections of an individual into the initial population and the number of groups to collect the beams of the structure. This will reduce the time of convergence because the individual will be next to the optimum one and a minor number of goups will reduce the number of evaluations, but always in function of the designer experience.

2.2. *"Aptitude"* selection operator.

In the most frequent reproduction found in the bibliography[4], an individual is selected to form part of the new generation based on its fitness, independently of whether this is very far or not from the average. This can give rise to isolated individuals or "strangers", though with high aptitudes, also having a high number of offsprings, thus vastly altering subsequent generations.

For this reason, a reproductive operator denominated *aptitude*[2] has been implemented (Fig. 3) that considers the population dispersion. In this operator:

- A new function is defined on the basis of the modified objective function (see 2.4), denominated *aptitude function*.
- The value of the aptitude function or *aptitude* of all the individuals is obtained and those whose values are lower than the average are eliminated.
- A *probability of rejection* is defined for the surviving individuals, of inverse value to aptitude.

The new population is thus created from the best individuals of the previous population, avoiding isolated elements and increasing the speed of the GA in the search for the optimal individual.

To study the efficiency of the *aptitude*, it has been compared with the *roulettewheel* selector[5]. For it, trial runs with both selectors and with different combinations of parameters (P_{mut}, P_c, P_e) have been carried out. In Fig. 4 is represented the variation of the average weight of the optimum individuals and the average number of function evaluations against the population size for P_{mut} = 0,5%; P_c = 70% and P_e = 30%.

This study shows that an increase of the population size, decreases the average weight in the two selectors until reaching population sizes of 40 individuals. In that point the value of the average weight barely differs between the two selectors. On the contrary, an increase in the population size produces an increase in the average of the number of evaluations. In both cases the values obtained with the roulettewheel selector are over the ones obtained with the aptitude selector. This signifies worse optimum individuals and a greater time of convergence, i.e., a greater computational consume for the roulettewheel selector.

2.3. "*Phenotype*" crossover operator.

In general, GAs use crossover operators that interchange bits, randomly assigning the crossover points[2]. This entails not only an exchange of information, but also an alteration in the design variables.

This modification is avoided by means of the so-called *phenotype* crossover. In this operator, the crossover point is located between two phenotypes or design variables from two individuals termed parents. Two new strings, the children, are created swaping all the characters between the selected position and the overall length of the parents strings (Fig. 5).

To test the behaviour of the phenotype crossover, trial runs with *one-point*[5], *two-points*[5] and phenotype crossover have been carried out (Fig. 6).

The study let to say that for population sizes over 60 individuals, none of the crossover operators has a considerable effect in the results. On the contrary, for lower populations the results depend on the values of the parameters. In every trial the phenotype crossover was more stable opposite to one-point ant two-points crossover. Besides, it carries out less number of evaluations, so much for upper as lower populations of 60 individuals, what is transformed into faster speed of convergence and smaller computational consume.

2.4. Formulation of the problem.

The function used as a measure of effectiveness of the design is denominated the *objective function, merit function* or *cost function*. This function may be formulated from a simple objective $f_1(x)$ or multiple objectives as shown in Eq. 2.

$$F(x) = \{f_1(x), f_2(x), \ldots, f_p(x)\} \quad (2)$$

Optimization with more than one objective is denominated *multicriterion optimization*[6] and is the most general case of structural optimization, where the weight, displacements, stresses, loads or some combination of these can be used as an objective function. Different ways exist to reduce the number of functions, but the most widely used one is:

- To choose an objective function that analyzes the total weight of the structure and considers the imposed limits (stresses in each member, displacements in the nodes, critical loads).

The variables used are discrete, i.e. they take precise values from a list or commercial catalogue, which can be mathematically expressed as (Eq. 3):

$$\begin{aligned} & x = \left(x_1^T, x_2^T, x_3^T, ..., x_j^T\right) \qquad j = 1,2,3,..,J \\ & x_{i,j} \in D_j \\ & D_j = \left(d_{j,1}, d_{j,2}, ..., d_{j,\lambda}\right) \end{aligned} \tag{3}$$

where the vector of the design variables x is divided into x_j sub-vectors, whose component $x_{i,j}$ take values from a D_j catalogue, and in which i is the number of design variables in each sub-vector and λ is the number of sections in each catalogue.

The limits that the design variables take are given by Eq. 4.

$$\frac{G_s(x)}{\tilde{G}_s(x)} \le 1 \quad s = 1,2,...,s \tag{4}$$

where $G_s(x)$ is the calculated value of the constraint, $\tilde{G}_s(x)$ is its limited value and s is the number of inequality functions.

In the optimization problem, not all the constraints are function of a term but may be functions of several terms (Eq. 5).

$$\frac{G_{s,1}(x)}{\tilde{G}_{s,1}(x)} + \frac{G_{s,2}(x)}{\tilde{G}_{s,2}(x)} + ... + \frac{G_{s,m}(x)}{\tilde{G}_{s,m}(x)} \le 1 \tag{5}$$

where m is the number of terms in the constraint function.

Considering the expression of the objective function (Eq. 2), of the design variables (Eq. 3) and of their constraints (Eq. 4), the optimization problem may be expressed mathematically according to Eq. 6, whose interpretation is:

The aim of structural optimization, and in particular GAs, is to obtain the sections of the structural elements that minimize an objective, subject to certain limits or constraints.

$$
\begin{array}{lll}
\textit{Minimize} & F(x) & \\
\textit{Constraints} & \dfrac{G_s(x)}{\tilde{G}_s(x)} \le 1 & s = 1,2,\dots,s \\
& x = \left(x_1^T, x_2^T, \dots, x_j^T\right) & j = 1,2,\dots,J \\
& x_{i,j} \in D_j & \\
& D_j = \left(d_{j,1}, d_{j,2}, \dots, d_{j,\lambda}\right) &
\end{array}
\qquad (6)
$$

These constraints may be classified in two types: explicit and implicit. Explicit constraints are analyzed without a simulation system. In contrast, implicit constraints require analysis and verification of the designs, such as for instance the allocation of areas to the sections.

Several methods of adjusting the constraints exist[7]:

- Using operators specialized in viability.
- Using only viable solutions.
- Penalizing the solutions that violate one or more constraints.

Specialized operators only work with explicit constraints and are useful for cost problems. The second and third methods can be used with implicit or explicit constraints, or a combination of both. In the second method, the members of the population that violate one or more constraints are eliminated; this can be ineffective in large problems with few viable solutions with respect to nonviable ones. The most appropiate method is therefore the penalizing of the members of a population that have one or more violations, although difficulties exist when applying penalty functions as they are usually dependent problems.

In general, the problems covered by GA are of the restricted optimization type, and for this reason the optimization problem must be transformed into nonrestrictive problems. In this case, the penalty is based on the transformation method represented in (Eq. 7).

$$\textit{Minimize} \quad \overline{F}(x,r) = F(x) + \overline{P}(r,G(x),H(x)) \tag{7}$$

where $\overline{F}(x,r)$ is the modified objective function, $F(x)$ is the objective function and $\overline{P}(r,G(x),H(x))$ is the penalty term defined as a function of the penalty coefficient r and the constraint functions $G(x)$ and $H(x)$.

The method is defined by means of penalty parameters, the rules that update these parameters and constraint functions.

As mentioned above, the aim of the implemented elitist GA is to obtain steel structures of minimum weight that fulfill the safety factors established by the applicable building code. This may be mathematically expressed according to (Eq. 8).

$$\begin{aligned} &\textit{Objective function to minimize} && F(x) = \rho \cdot \sum_{s=1}^{n_{bar}} x_s \cdot L_s \\ &\textit{Constraints} && G_s(x) \le 1 \quad H_s(x) \le 1 \quad \ldots \quad T_s(x) \le 1 \end{aligned} \tag{8}$$

where $F(x)$ is the objective function (the weight of the structure analyzed), defined on the basis of ρ the density of the material, x_s the area of the section and L_s the length of n_{bar} bars that compose the structure; and the constraints, the limit values that the safety factors calculated in each bar $G_s(x)$, $H_s(x)$, ..., $T_s(x)$ may reach.

Applying the mathematical expression of the implemented GA (Eq. 8) to the transformation of (Eq. 7), the problem will be expressed mathematically according to (Eq 9).

$$\overline{F}(x,r) = F(x) + \sum_{s=1}^{n_{bar}} \left[r_1 \cdot G_s(x) + r_2 \cdot H_s(x) + \ldots + r_{n_c} \cdot T_s(x) \right] \quad (9)$$

where n_c is the number of safety factors established by the applicable building code.

The addition of another term to the penalty function in order to consider the displacements of each node as a design criteria is not so simple as it seems. Though our analysis program give us every node displacements, each structural member can be submitted to two different verifications. One local to each member and a global one where a maximum displacement (x,y,z) is considereted for the whole problem. The latter would led to a new penalty function that it is being studied by the authors.

Of the terms that define the modified objective function, the easiest to obtain is the weight, since it is directly defined on the basis of the geometric data of the structure, the characteristics of the material assigned to the bars and the properties of the sections assigned to these.

In contrast, in order to obtain the second term it is necessary to define the penalty coefficient and to carry out an analysis of the structure. In this way, the stresses and moments that define the safety factors and therefore the constraints of the problem will be obtained. The analysis of the structure and the verification of the safety factors can be carried out by means of a program of conventional analysis. In this study, the analysis was carried out using the Escal3D[8] program, capable of obtaining the safety factors established by Spanish, French, American and European building codes. Specifically, the Spanish bulding code[3] was considered, which defines safety factors such as the quotient between the calculated value and the maximum allowed value of:

axial stress, shear stress, shear and bending stress, bending, Von Mises stress, buckling by compression, buckling by compression and bending, buckling by torsion, buckling by torsion and bending[9].

Although the displacements in each node is very important in the design of an steel structure, in the Eq. 9 only the stress terms have been considered like a first aproximation to the optimum structure.

2.4.1. Penalty factor.

The *safe bar* concept was considered for the definition of the penalty coefficient. A safe bar is the one whose safety factors are equal to or lower than one[9]. In addition, if the coefficient is far below unity the bar is considered *oversized.* That is to say, there exists another section with a smaller area that, if assigned to this bar, provides coefficients closer to unity, thus diminishing the weight of the structure.

In contrast, if the calculated safety factor is greater than unity, the bar is not able to support the stresses and moments calculated in it. In this case, it will be necessary to look for another section whose resistant properties are able to support these stresses and moments.

In accordance with the *safe structure* concept, the penalty coefficient is defined as the value which multiplied by the safety factor calculated in a bar, increases this coefficient if it is different from one and maintains it constant if it is equal to one. The sum of the penalized coefficients of all structural elements will be the penalty term of the modified objective function.

Therefore, the penalty term increases both, the weight of structures with bars that do not fulfill some of the safety factors and the weight of structures with oversized bars, distancing them, in both cases, from the sought-after minimum weight.

Initially the penalty coefficient was assigned the value of 1000 for safety factors greater or lower than one and the value of 1 for safety factors equal to one (Eq. 10).

$$r(c)=\begin{cases} c & if \quad c=0 \quad or \quad c=1 \\ c\cdot 1000 & if \quad c\neq 0 \quad and \quad c\neq 1 \end{cases} \qquad c=G_S(x) \tag{10}$$

As can be observed in Fig. 7a, this first adjustment did not differentiate between the penalty that was assigned to the structures that contained bars with coefficients close to unity and the one that was assigned to the structures whose bars presented coefficients far from this value. Moreover, after successive analyses and trial runs in the two-dimensional structure (Fig. 1), it was demonstrated that coefficients of less than one were less penalized than those above unity, resulting oversized structures like optimum individuals.

Then an exponential distribution was assigned to the penalty coefficient in an attempt to penalize those coefficients that were far from unity more, favoring the search for minimum weight in zones close to one (Eq. 11).

$$r(c)=\begin{cases} 0 & if \quad c=0 \\ e^{2-c} & if \quad 0<c<1 \\ 1 & if \quad c=1 \\ e^{c} & if \quad c>1 \end{cases} \qquad c=G_S(x) \tag{11}$$

In this case, those individuals with safety factors far from unity were penalized more strongly, but no distinction was made between oversized structures and nonvalid structures, since this penalization affected values lower than one in the same way as values above one (Fig. 7b). This meaned that oversized structures with coefficients far below unity but with a very great structural weight had similar values of the modified objective function to less heavy structures with safety factors higher than unity.

On the other hand, the growth of an exponential distribution was so fast that values of coefficients higher than unity produced excessively large penalty coefficients from a computational point of view.

Taking all the above considerations into account, it was decided to carry out two types of adjustment of the penalty coefficient on the basis of whether the calculated safety factor was lower than unity or not. In the former case, an exponential distribution was followed, thus favoring the individuals with coefficients close to unity. In the later case, a linear distribution with penalty values much higher than the above values was followed in order to avoid the equality of weighting between oversized structures and nonvalid structures (Eq. 12), (Fig. 7c).

$$r(c) = \begin{cases} 0 & if \quad c = 0 \\ e^{2-c} \cdot 10 & if \quad 0 < c < 1 \\ 1 & if \quad c = 1 \\ c \cdot 1000 & if \quad c > 1 \end{cases} \qquad c = G_S(x) \tag{12}$$

2.4.2. Trials.

In order to test the suggested improvements (in addition to the modification of the penalty factor), and investigate the effect of tunning GA parameters[10] (N_p, P_{mut}, P_c, P_e) on the performance of the developed elitist algorithm, the domain of each parameter has been set as follows:

- The population size N_p varies from 20 to 140.
- The elite probability P_e varies from 0,0% to 90%.
- The crossover probability P_c varies from 10% to 90% (subjected to $P_c + P_e \leq 100\%$).
- The mutation probability P_{mut} varies from 0,1% to 4%.

The study has been carried out using five runs for each combination of the domain aforementioned, on the structure shown in the section 1.

The conclusions can be summarised as follows (Fig. 8):

- When the value of N_p increases, the average weight of the best individuals decreases, although the average number of evaluations increases too. It can be observed that solutions with a good performance of the developed GA are obtained with a value of N_p between 60 and 100 individuals.
- A value of P_{mut} ranging between 1% and 3% and a value of P_e elite ranging between 10% and 30% give better solution achieved within a reasonable average number of function evaluations and coefficients next to one.
- An increase of the P_c until values of 80%, with the same probability of elite, decreases the average weight.

After carrying out all the trials aforementioned, the best individual was found in 100 runs with the following parameters: Np = 100, Pe = 30%, Pc = 70%, $Pmut$ = 1%.

The steel sections obtained in this individual and its minimum weight are given in Table 1. This table also includes the solution obtained from a conventional design approach, using a commercial program of structural analysis. It can be seen that the minimum weight solution is about 9,3% lighter than the conventional design.

An advantage of the computational model presented in this paper is that one can quickly find several minimum weight soluctions using different grouping members. A cost comparison of various solutions will then yield the optimum solution.

2.5. Flow of the elitist GA.

The flow of the implemented elitist GA is represented in Fig. 9. It can be seen how each new population is formed by three types of individuals obtained from the surviving population (Fig. 10):

- *Elite individuals*, selected between the best individuals of the current population without mutation and whose number N_e is the result of multiplying the elite probability P_e and the population size N_p.
- *Crossover individuals*, selected between the surviving individuals in function of their probability of rejection with mutation and whose number N_c is the result of multiplying the crossover probability P_c and the population size $N_{p.}$
- *Random individuals*, whose number is equal to the difference between the total number of individuals in the population and the sum of elite and crossover individuals.

4. Conclusions.

A modified elitist GA has been developed composed of a crossover operator denominated *phenotype crossover* and a selection operator named *aptitude* that increase its speed. In addition, the codification of the design variables has been modified so that all of these have the same probability of initial selection, and a modified objective function has been defined that, when applied to steel structures, diminishes the weight of the structure according to safety factor constraints.

The implemented elitist GA is a robust method of optimization of little mathematical complexity that is adequate for designers. It does not need prior information about the objective function or the constraint functions and can work with complex structures under different load and constraint conditions. In addition, it permits the use of commercial sections catalogues as design variables and is able to apply the engineer experience in creacting groups of identical section, selecting these variables and their relation with the structural members.

References.

1 Davis, L. "Handbook of Genetic Algorithms". Van Nostrand Reinhold, New York, 1991.

2 Mahfouz, S. Y. "Design Optimization of Structural Steelwork. Design Optimization of steel frame structures according to the British codes of practice using a genetic algorithm". Thesis. Department of Civil and Environmental Engineering. University of Bradford, UK, 1999.

3 Norma Básica NBE EA-95. "Estructuras de Acero en Edificación. Norma Básica de la Edificación NBE EA-95". Madrid Vicente, Ediciones, 1996. ISBN 84-87440-86-X.

4 Bartschi Wall, M. "A Genetic Algorithm for Resource-Constrained Scheduling". Thesis. Department of Mechanical Engineering. Massachusetts Institute of Technology, 1993.

5 Bartschi Wall, M. "Galib: A C++ Library of Genetic Algorithm Components". Department of Mechanical Engineering. Massachusetts Institute of Technology, 1996.

6 Álvarez, L. F. "Design Optimization based on Genetic Programming. Aproximation model building for design optimization using the response surface methodology and genetic programming". Thesis. Department of Civil and Environmental Engineering. University of Bradford, UK, 2000.

7 Topping, B. H. V.; Papadrakakis, M. "Advances in Structural Optimization". Civil – Comp Press, paperback 246 pages, 1994. ISBN 0948749-27-X.

8 Bello García, A.; Ordieres Meré, J.; Del Coz Díaz, J. J.; Suárez Domínguez, F. J.; Felgueroso Fernández, D.; Álvarez Fernández, M. "Aprendizaje Interactivo mediante el programa de análisis estructural avanzado ESCAL3D". Journal of Constructional Steel Research. Editorial Elsevier Science, 1998. Vol 46, Pag. 273-275.

9 Argüelles Álvarez, R.; Argüelles Bustillo, R.; Arriaga Martitegui, F.; Atienza Reales, J. R. “Estructuras de Acero. Cálculo, Norma Básica y Eurocódigo”. Bellisco. Ediciones Técnicas y Científicas, Madrid, 1999. ISBN 84-930002-8-0.

10 Mahfouz, S. Y.; Toropov, U. U.; Westbrook, R. K. “Improvementes in the performance of a genetic algorithm: application to steelwork optimum design”. Proceedings of 7th AIAA/USAF/NASA/ISSMO. Symposium on Multidisciplinary Analysis and Optimization, 1998. Pag. 2037-2045.

Tables

Number of group	Sections (E.G.A.)	Weight (kg)	Sections (conventional design)	Weight (kg)
1	HEA300	1519,14	HEA300	1519,14
2	HEA300	773,50	HEA300	773,50
3	LSI150x15	345,08	IPN240	378,60
4	HEB220	488,49	IPE550	715,15
5	IPN400	1935,72	IPE550	2145,45
Weight (kg)		5061,93		5531,84

Table 1. Best individual against conventional design.

Legends to illustrations.

Figure 1. Structure analysed.

Figure 2. Encoding the design variables.

Figure 3. Selection operator: “*aptitude*”.

Figure 4. Aptitude selector against roulettewheel selector.

Figure 5. Phenotype crossover in binary representation.

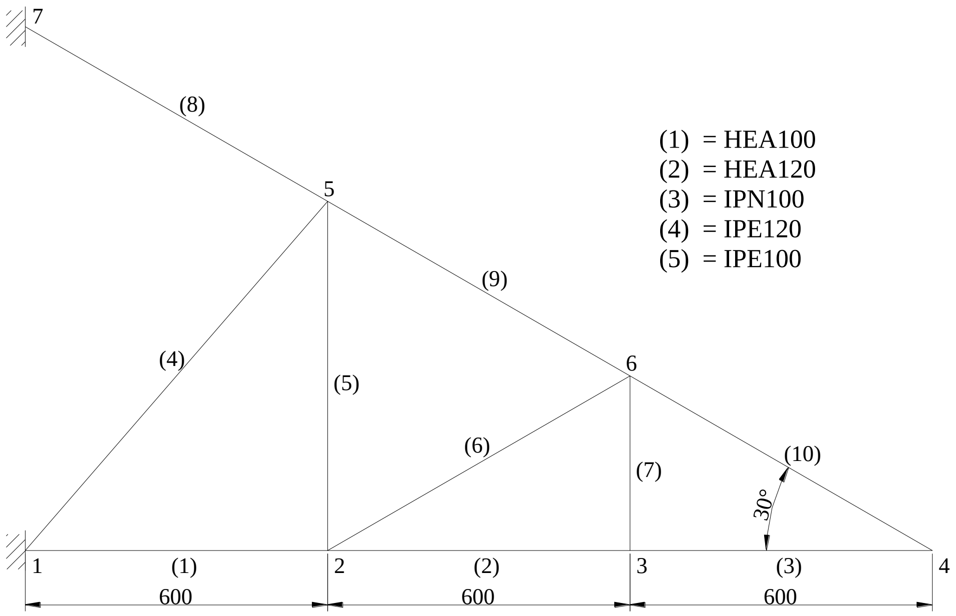
Figure 6. Phenotype crossover against one-point and two-points crossover.

Figure 7. Penalty coefficients.

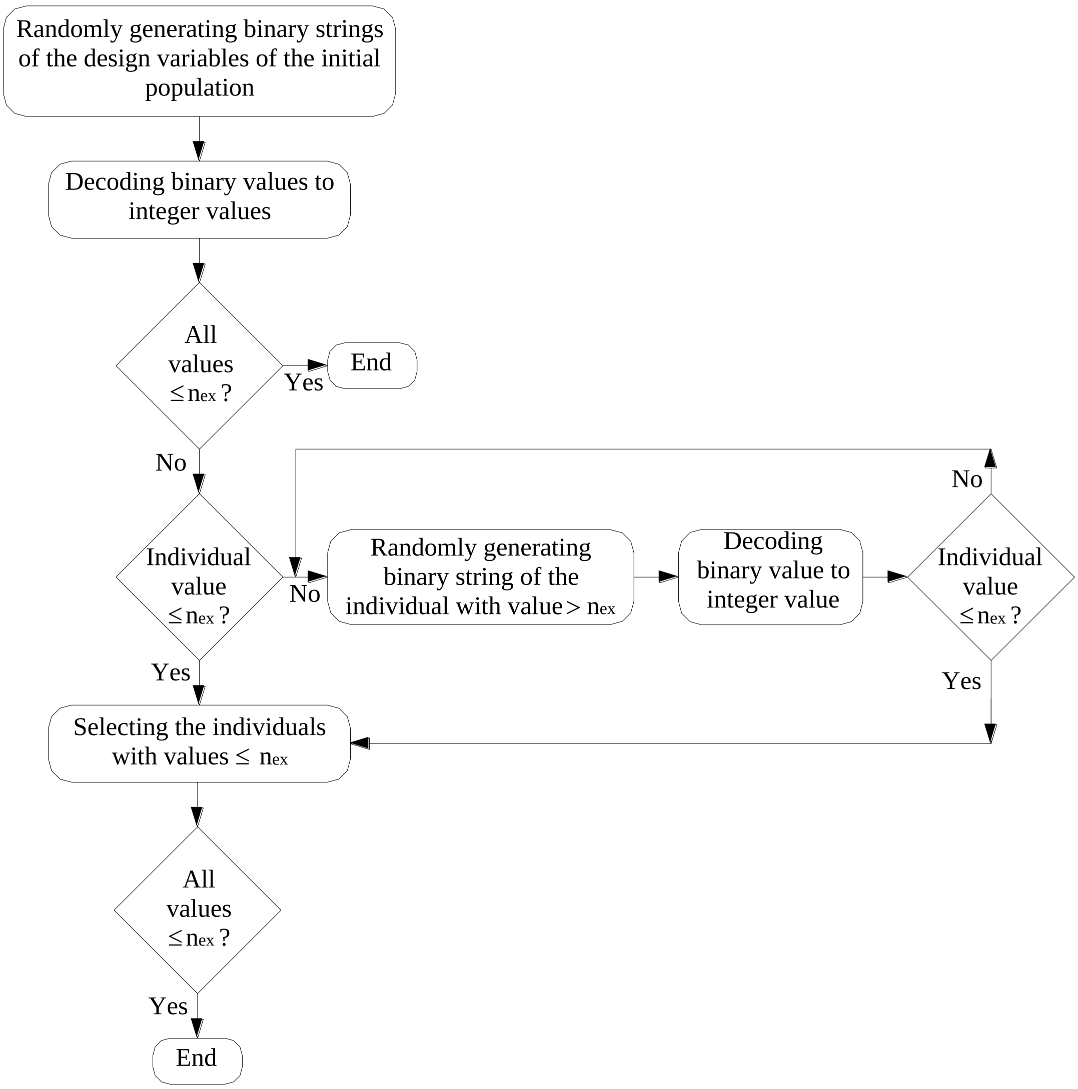
Figure 8. Tunning of parameters.

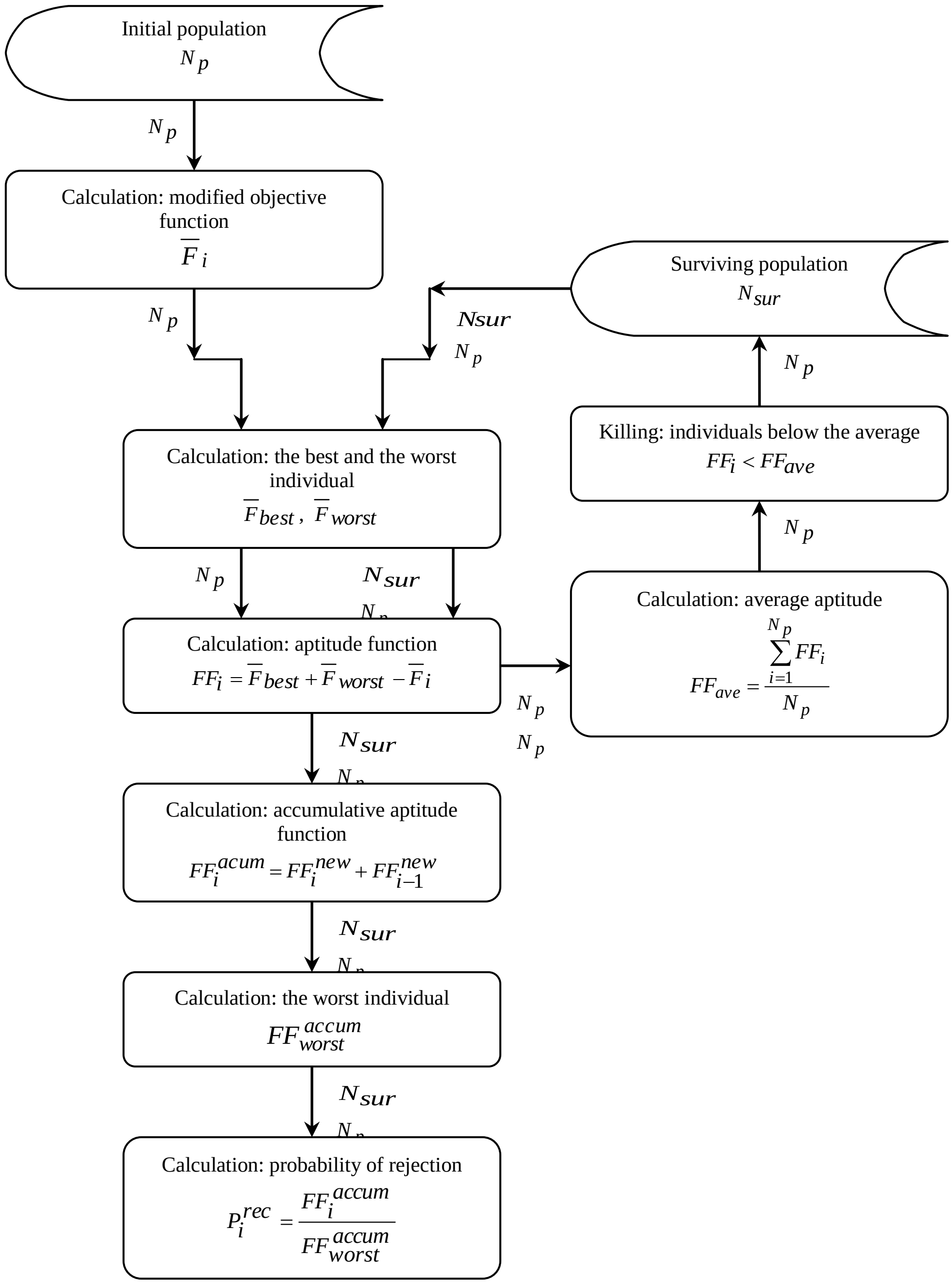
Figure 9. Elitist GA flow.

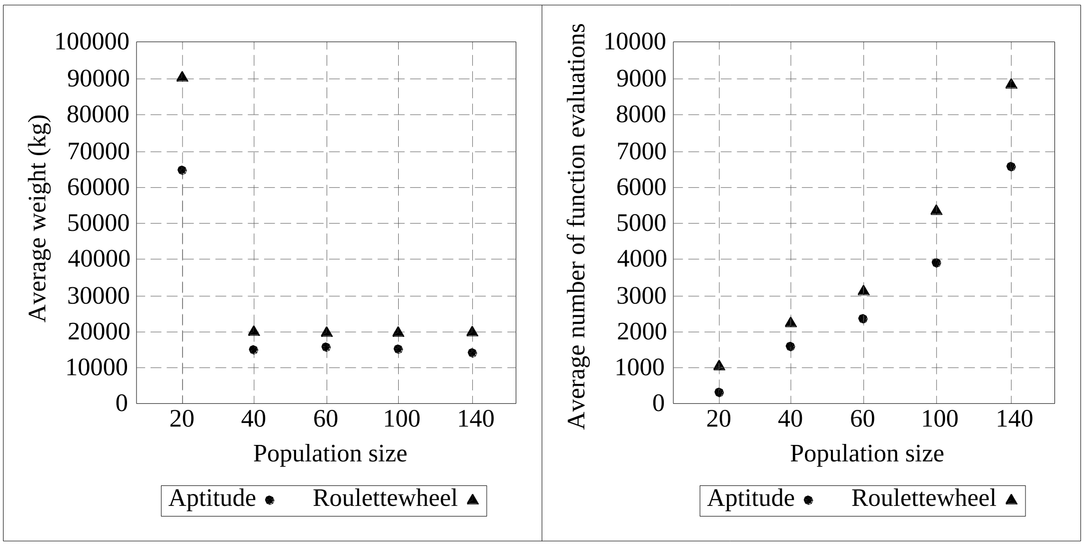
Figure 10. New populations with elitist GA.



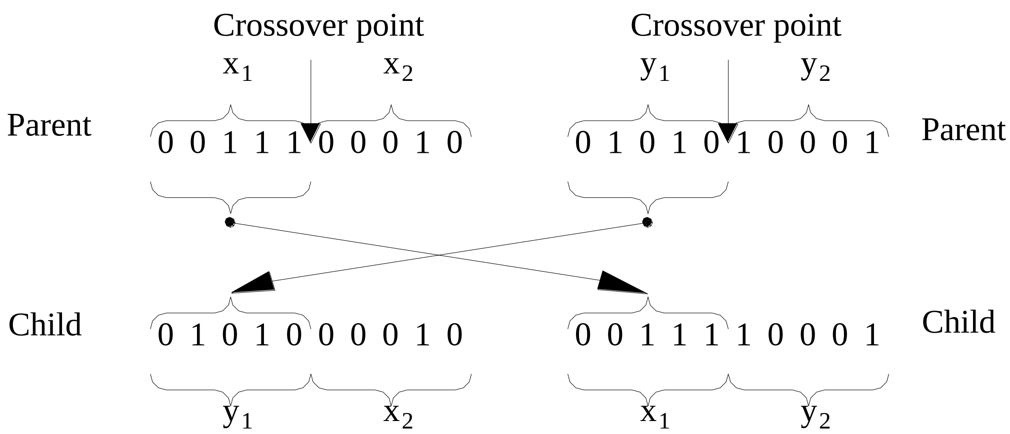
7
(8)
5
(9)
6
(10)
(4)
(5)
(6)
(7)
30°
1
(1)
2
(2)
3
(3)
4
600
600
600
(1) = HEA100
(2) = HEA120
(3) = IPN100
(4) = IPE120
(5) = IPE100


Randomly generating binary strings of the design variables of the initial population
Decoding binary values to integer values
All values ≤ n_{ex} ?
Yes
End
No
Individual value ≤ n_{ex} ?
No
Randomly generating binary string of the individual with value > n_{ex}
Decoding binary value to integer value
Individual value ≤ n_{ex} ?
No
Yes
Yes
Selecting the individuals with values ≤ n_{ex}
All values ≤ n_{ex} ?
Yes
End

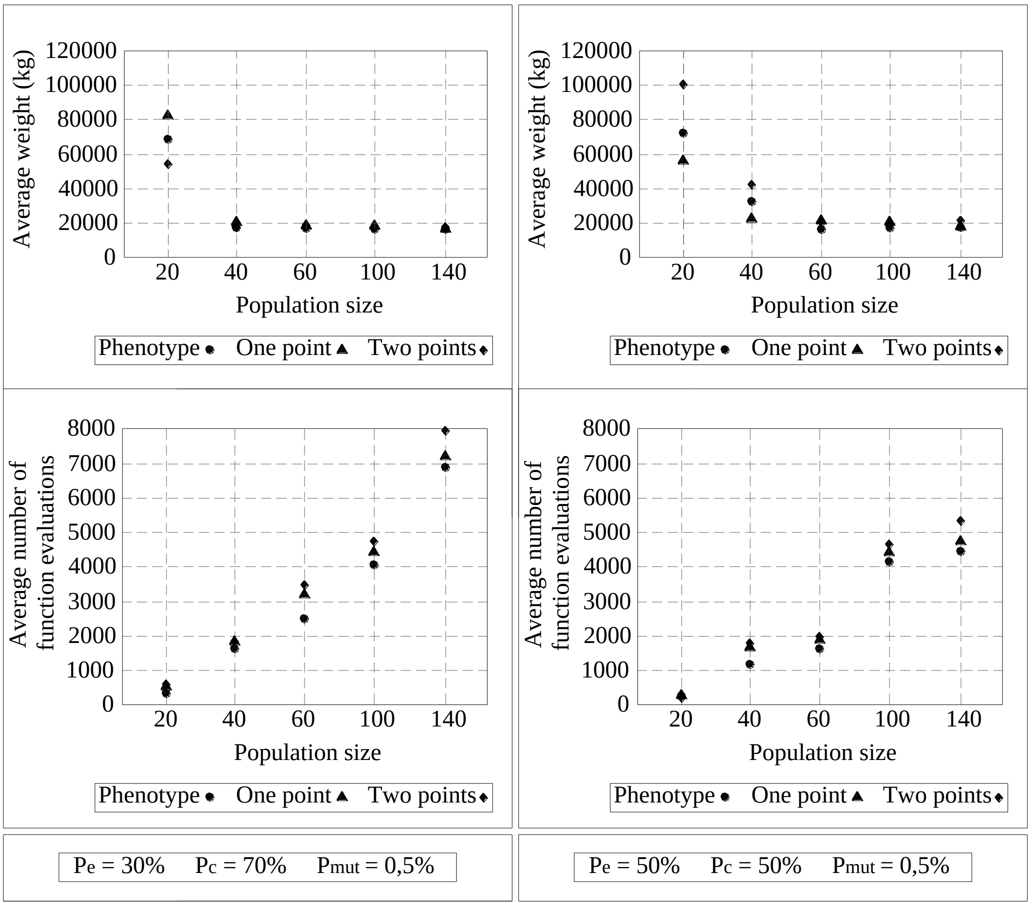

Initial population
N p
Calculation: modified objective function
F i
Surviving population
N sur
Nsur
Killing: individuals below the average
FFi < FFave
Calculation: the best and the worst individual
F best , F worst
Calculation: aptitude function
FFi = F best + F worst − F i
Calculation: average aptitude
FFave = Σ FFi / N p
Calculation: accumulative aptitude function
FFi acum = FFi new + FFi−1 new
Calculation: the worst individual
FF worst accum
Calculation: probability of rejection
Pi rec = FFi accum / FF worst accum

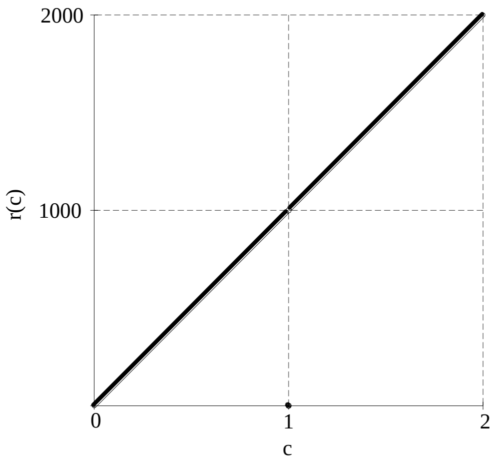


100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
Average weight (kg)
20
40
60
100
140
Population size
Aptitude
Roulettewheel
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Average number of function evaluations
20
40
60
100
140
Population size
Aptitude
Roulettewheel

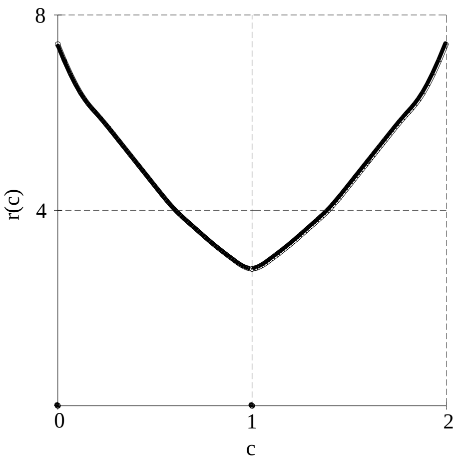


Crossover point
Crossover point
x1
x2
y1
y2
Parent
0 0 1 1 1 0 0 0 1 0
0 1 0 1 0 1 0 0 0 1
Parent
Child
0 1 0 1 0 0 0 0 1 0
0 0 1 1 1 1 0 0 0 1
Child
y1
x2
x1
y2

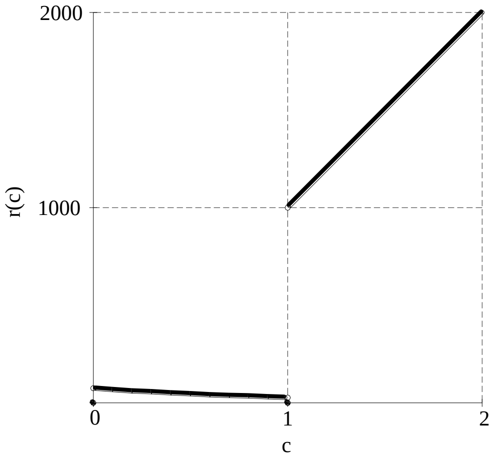

Average weight (kg)
120000
100000
80000
60000
40000
20000
0
20
40
60
100
140
Population size
Phenotype
One point
Two points
Average number of function evaluations
8000
7000
6000
5000
4000
3000
2000
1000
0
Pe = 30%
Pc = 70%
Pmut = 0,5%
Pe = 50%
Pc = 50%
Pmut = 0,5%



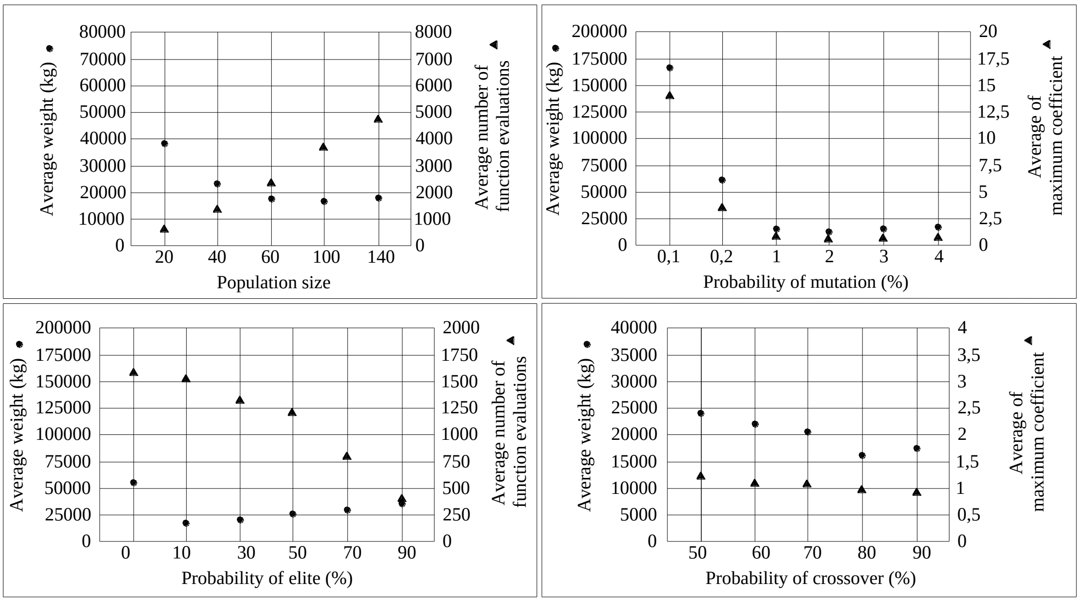
a. - Lineal penalty coefficient



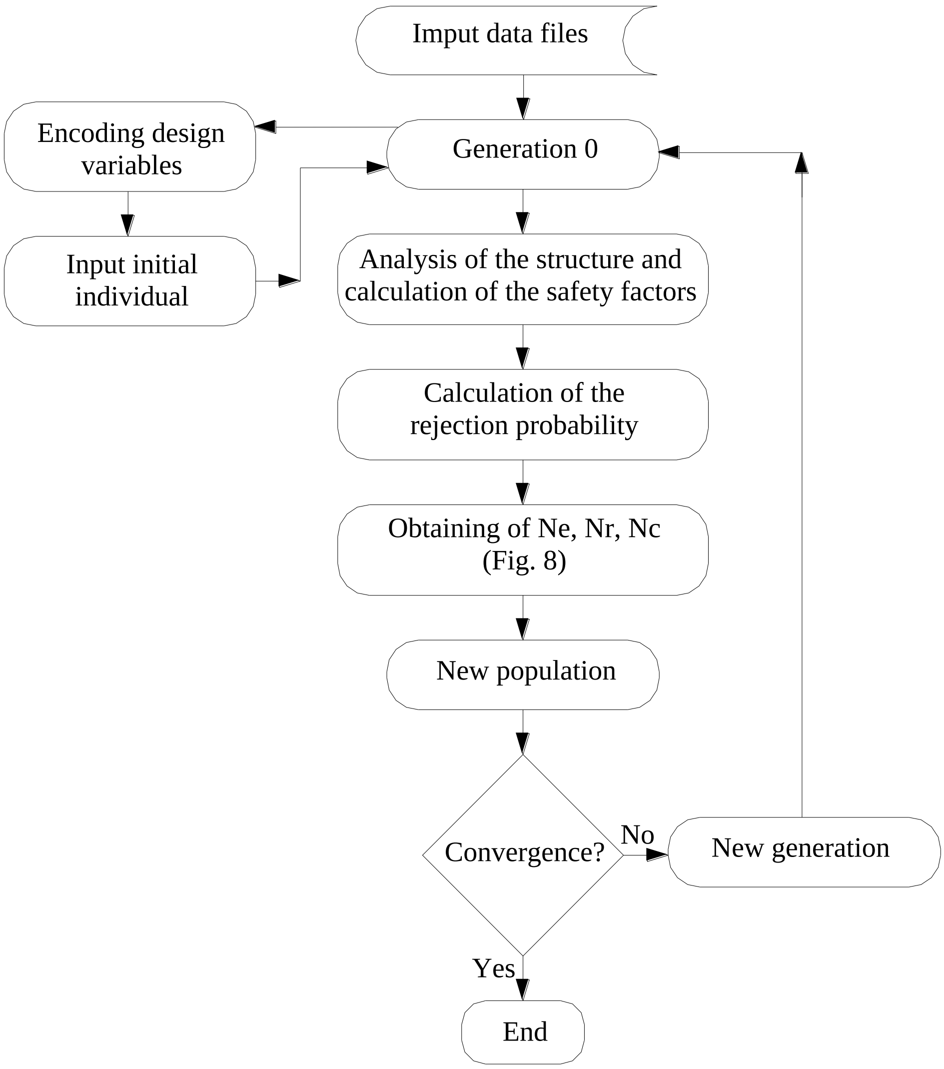
b. - Exponential penalty coefficient

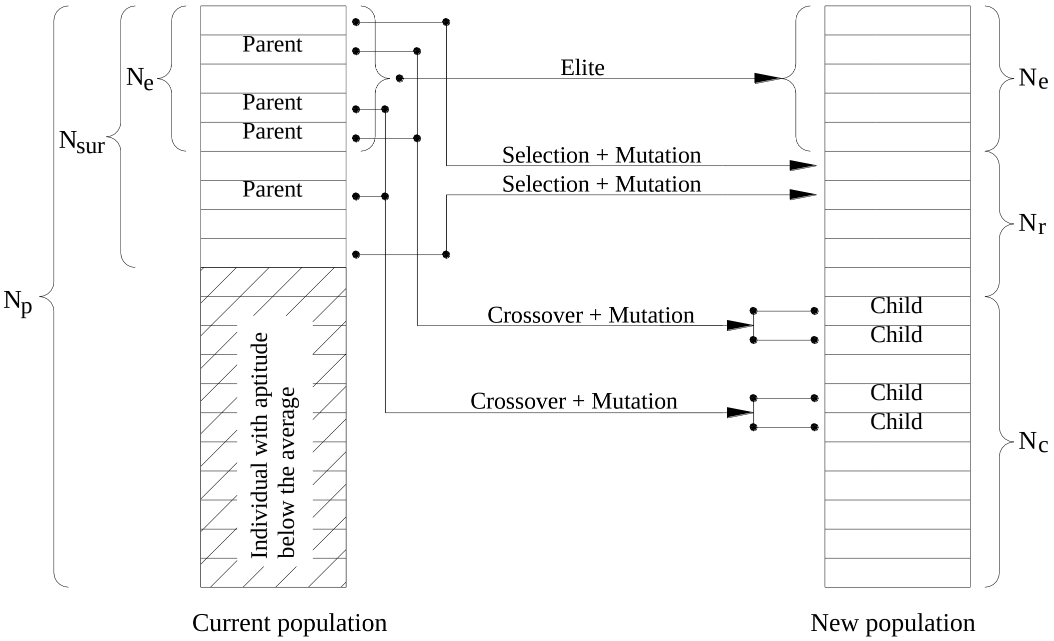


c. - Encoding penalty coefficient



Average weight (kg)
Average number of function evaluations
Population size
Average maximum coefficient
Probability of mutation (%)
Probability of elite (%)
Probability of crossover (%)


Imput data files
Encoding design variables
Generation 0
Input initial individual
Analysis of the structure and calculation of the safety factors
Calculation of the rejection probability
Obtaining of Ne, Nr, Nc (Fig. 8)
New population
Convergence?
No
New generation
Yes
End



Parent
Parent
Parent
Parent
Elite
Selection + Mutation
Selection + Mutation
Crossover + Mutation
Crossover + Mutation
Child
Child
Child
Child
N_e
N_{sur}
N_p
N_e
N_r
N_c
Individual with aptitude below the average
Current population
New population