Genetic Algorithm Optimization of Semi-Rigid Steel Structures

Alexandre A. Del Savio[1], Sebastião A. L. de Andrade[1], Pedro C. G. da S. Vellasco[3] and Luiz Fernando Martha[2]

[1] PhD Student, Civil Engineering Department,
PUC-RIO - Pontifical Catholic University of Rio de Janeiro, Brazil.
[2] Associate Professor, Civil Engineering Department,
PUC-RIO - Pontifical Catholic University of Rio de Janeiro, Brazil.
[3]Associate Professor, Structural Engineering Department,
UERJ - State University of Rio de Janeiro, Brazil.

Abstract

The main purpose of the present paper is to describe an investigation made on the optimization of the flexural capacity of steel structures by varying the stiffness and resistance of its associate semi-rigid joints. This optimisation process is made using a genetic algorithm developed and implemented in a structural analysis program, FTOOL/SRC, [1,2]. This algorithm was inspired in the optimum-use algorithm, incorporating the problem domain's knowledge and leading to the development of a user-friendly system with consistent solutions. This paper describes in detail the formulation steps and procedures, and presents examples of semi-rigid steel portal frames. The purpose of the examples was to validate and prove the genetic algorithm efficiency towards an efficient bending moment distribution within the investigated semi-rigid steel structure. This purpose was fulfilled by varying the stiffness and the capacity of the adopted semi-rigid joints.

Keywords: steel structure optimisation, semi-rigid joints, structural engineering, genetic algorithm, computational intelligence, structural steel joint design.

1 Introduction

Traditionally, structural design is made by trial-and-error procedures guided by design specifications and by the structural engineer's experience and intuition. Meanwhile, continuous research on this field is being done to develop simple design tools aimed to help the structural designer to perform repetitive and time-consuming tasks.

Among the structural optimisation techniques, math programming and genetic algorithms have been frequently used to generate economic and efficient structures. Genetic algorithms are often used for this purpose due to their intrinsic abilities to deal with problems without precise mathematical models, being capable of

performing global searches, representing the solutions in a simple form and allowing a natural and easy implementation of rules and auxiliary heuristics.

The complete steel frame optimization involves a significant number of design variables that should be considered together with countless restrictions. To exemplify these aspects it is possible to mention the steel profile geometry variations (height, flanges, etc.), vertical and horizontal displacement restrictions, etc.

The main goal of the present investigation is to develop and implement an optimization methodology for the design of 2D steel frames, in which the main variables are associated to the structural joint stiffness. The aim is to determine an ideal bending moment distribution that could lead to an efficient steel profile, avoiding situations such as the one depicted in Fig. 1, related to the use of hinged beam-to-column joints where the joint stiffness is zero.

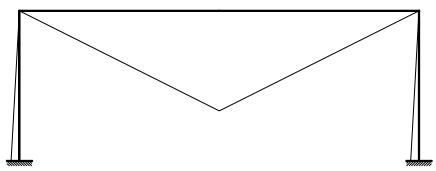


Figure 1: Steel frame (with hinged beam-to-column joints) bending moments.

2 Semi-Rigid Connections

Structural joints play a fundamental role in the global steel structures behaviour. The actual joint response is usually situated between two extremes: pinned and rigid. The rigid behaviour implies no change in the angle of members adjacent to the joint. On the other hand, flexible joints indicate that no bending moment transfer will occur among members connected by the joint.

In the intermediate case, semi-rigid joints, the bending moment to be transferred among members will be a function of their relative rotation change. Consequently, in global structural analysis the joints not only affect the structural displacements but also interfere in the distribution and magnitude of their internal forces and/or moments in the entire structure. These aspects indicate that structural modelling, including semi-rigid joint effects, becomes imperative in order to represent the actual structural behaviour of current joints found in the design practice.

3 Formulation/Modelling of the Optimisation Problem

The investigated problem can be summarized in an optimization of the bending moment distribution present in steel portal frames. This optimization is made by varying the joint stiffness and is expressed by the expression:

$$f_{value\ \ close}(k_1, k_2, ..., k_n) = \sum_{e=1}^{nelem} \frac{M_{ei}}{M_{ej}} = nelem \tag{1}$$

where k is the semi-rigid joint stiffness, n is the number of semi-rigid joints, *nelem* is the number of structural bars and M_i *and* M_j are the bar element bending moments on the initial, i, and final, j, nodes. This function represents an optimization problem concerning the determination of a value close to the number of structural bars, *nelem*. In order to change this search algorithm into a minimization procedure, Eq. (1) is rewritten:

$$f(k) = f_{\min}(k_1, k_2, \ldots, k_n) = ABS\left(nelem - \sum_{e=1}^{nelem} \frac{M_{ei}}{M_{ej}} \right) \tag{2}$$

Using this approach, the problem is now formulated based on a minimization of the objective function. This is made by determining M_{ei} and M_{ej} that minimise the *f(k)* function, where M_{ei} and M_{ej} depend on the semi-rigid joint stiffness (k_1, k_2, ...,k_n, variables). In order to better define the problem, additional restrictions were introduced on the stiffness search space. These restrictions, which can be defined by the user, limit the acceptable search space region, thus eliminating the infinite search space problem.

4 FTOOL/SRC Structural Joint Modelling

Figure 2 presents the FTOOL/SRC [1,2] graphical interface used to model the semi-rigid joints. In this figure it is possible to identify the spring icons representing the joints as well as their initial stiffness values.

More details about the FTOOL/SRC program, the type of structural analysis implemented and available elements used to represent the semi-rigid joints can be found in Del Savio et al. [1,2].

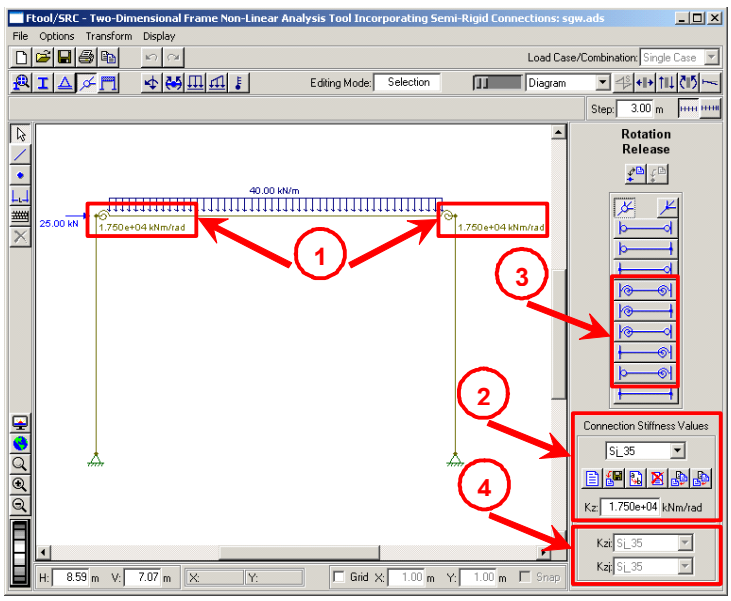


Figure 2: Semi-rigid joint pre-processing.

5 Genetic Algorithms

Genetic algorithms were initially formulated by John Holland at the end on the 60's inspired by Darwin's evolution theory presented in the famous *The Origin of Species* [5]. Genetic algorithms constitute a versatile and robust class of tools used to solve optimisation problems, despite the fact that they could not be strictly considered as function minimisers, [6].

The use of genetic algorithms to minimise functions differs from the usual mathematical programming techniques in that they use a population of individuals, deal with possible codified solutions, employ probabilistic transition rules and do not require additional information about the objective function.

The solution-search algorithm can be used in non-convex spaces, even in isolated sets, with non-convex and non-differentiable objective functions capable of dealing simultaneously with integer, real and logical variables. Genetic algorithms are not easily locked to local minima, as is the case of mathematical programming algorithms [5]. When these characteristics are applied to structural design, they can lead [7] to non-conventional solutions, generally not foreseen by structural engineers because they are non-intuitive solutions.

6 Implementation Details

This section describes the implementation details of a genetic algorithm based on Michalewicz [4], for the problem presented in previous sections. This algorithm incorporates the problem domain's knowledge, resulting in a user-friendly system. The initial structural modelling generates a seed for the genetic algorithm, leading to more consistent solutions.

6.1 Representation Scheme and Decoding

Several representation types are possible, such as binary, real (floating point), etc. The selection of an appropriate representation type depends on the characteristics of the search space.

The present investigation has adopted the real number representation, because it performs better in optimisation problems with variables on a continuous domain [10]. This is more relevant in large domains, where the binary representation requires the adoption of large chromosomes.

The real number representation achieves faster performances by eliminating further decoding stages. The real representation is also more precise (depending on the computer) and has a property according to which two points that are close in the representation space are also close in the problem space. This fact implies that the chromosome should be defined by a list of semi-rigid joint stiffness, where the number of joints n defines the chromosome's magnitude order, i.e.:

$$Chromosome \rightarrow C_i \rightarrow (k_1^i, k_2^i, ..., k_n^i) \rightarrow (k_m^i) \tag{3}$$

where i varies from 1 up to the number of elements of the population and m varies from 1 up to n.

6.2 Fitness Function

The fitness function is responsible for selecting individuals and determining their quality. In the present case this function is directly related to the objective function that requires minimization, i.e. the self-evaluation function, Eq.(2).

6.3 Selection

The genetic algorithm selection process selects individuals for reproduction. The selection is based on the individuals' fitness, i.e., more apt individuals have a larger probability of being chosen for reproduction. Two selection techniques were implemented: *relative fitness* selection and *linear normalization* selection.

The first implemented selection technique establishes that the probability p_i of the individual i in the population being selected for reproduction is proportional to its *relative fitness*, i.e.:

$$p_i = \frac{f_i}{\sum_{i=1}^{m} fi} \tag{4}$$

where $f_i = f(k_i)$ is assumed to be positive, and the population contains m individuals.

Two drawbacks can be identified in this process: the appearance of super-individuals and tight competition. The first case occurs when an individual presents much better fitness than the others, thus determining premature convergence. The second case happens when individuals present very similar fitness, leading to what corresponds to a very low selection pressure and a consequent stagnation of the search algorithm.

In order to minimize these deficiencies, a second selection method by *linear normalization* was also implemented. In this method the individuals are initially classified according to their fitness. Later, these fitness values are changed according to the individual relative position. The best/worst individuals are associated to *max* and *min* fitness values, these two values being determined by the user. The other individuals have fitness values linearly distributed between *min* and *max* according to their relative position ($i=1$ corresponds to the worst individual).

$$A_i = \min + \frac{(\max - \min)}{n-1} \times (i-1) \tag{5}$$

The *max* and *min* values define the increment constant, a technique variable. As the increment constant increases, there is also an increase in the selection pressure over the best individuals. This method reduces the domain of super-individuals and increases the selective pressure over individuals with similar fitness values.

6.4 Genetic Operators – Real Codification

With the representation of the population elements defined, the following step consisted in constructing the genetic operators that act on the genotypes to produce new individuals. Such operators are basically of two types: recombination (crossover) and mutation.

6.4.1 Recombination Operators

Crossover is the primary operator in GAs and the key to their success [8]. This led to the implementation recombination operators like the *arithmetic crossover* and others. The *arithmetic crossover* performs a linear recombination of two genitors as defined by two chromosomes:

$$C_1 \rightarrow (k_1^1, k_2^1, ..., k_n^1) \text{ and } C_2 \rightarrow (k_1^2, k_2^2, ..., k_n^2) \tag{6}$$

The linear recombination between C_1 and C_2 produces the new generation, C^*_1 and C^*_2,

$$C_1^* \rightarrow (k_1^{1*}, k_2^{1*}, ..., k_n^{1*}) \text{ and } C_2^* \rightarrow (k_1^{2*}, k_2^{2*}, ..., k_n^{2*}) \tag{7}$$

that are defined by

$$k_m^{1*} = \lambda k_m^1 + (1-\lambda) k_m^2 \text{ and } k_m^{2*} = \lambda k_m^2 + (1-\lambda) k_m^1 \tag{8}$$

where m varies from 1 up to n (number of semi-rigid joints) and λ is defined as a random value present in a 0 to 1 interval.

6.4.2 Mutation Operators

Due to the real codification, several alternatives arise for the development of specific mutation operators in the investigated problem. Two types of mutation were selected and implemented: *real mutation* and *self-adjusted mutation*.

Real mutation randomly selects an element (k_m^i) in the chromosome (C_i), having its value changed to $k_m^{i*} \in [lower, upper]$ (a user-defined range of values) if the probability check is true.

The *self-adjusted mutation* searches for a close solution based on random changes in both directions (+-e):

$$C_i \rightarrow (k_1^i, k_2^i, ..., k_n^i) \rightarrow C_i^* \rightarrow (k_1^{i*}, k_2^{i*}, ..., k_n^{i*}) \tag{9}$$

where k_m^{i*} is defined by:

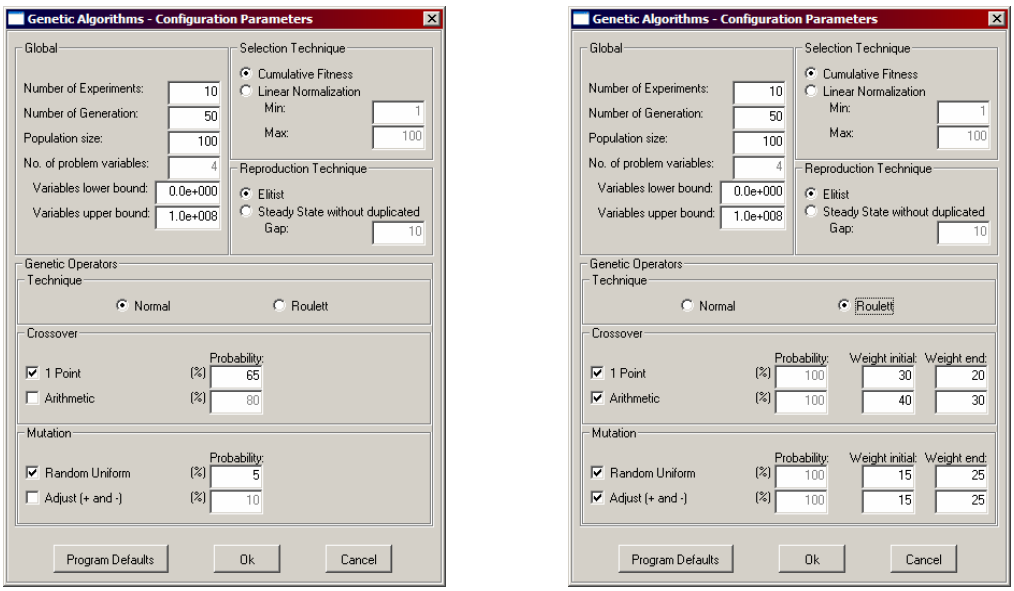
$$k_m^{i*} = \begin{cases} k_m^i + \Delta(\max - k_m^i) & \textit{if the chosen bit is equal to zero} \\ k_m^i - \Delta(k_m^i + \min) & \textit{if the chosen bit is equal to one} \end{cases} \tag{10}$$

where *max* and *min* are user-defined boundaries for the domain of the k_m^i variable, *m* varies from 1 up to *n* variables (number of semi-rigid joints) and Δ is a random number $\in$ [0,1]. This strategy makes the tuning magnitude vary randomly according to the Δ value.

6.4.3 Operator Roulette

The operator roulette technique was implemented in the system. In this strategy, all operators (mutation and crossover) are selected and applied according to the weight parameterization technique that linearly varies throughout the evolution.

With this technique, using independent operators, several genetic operators can be incorporated to the system. However, not all operators are used at the same time and with the same intensity. For example, it is intuitive that the crossover operator's rate is larger in the first generations, where the population is still scattered on the search space, thus allowing a faster search by taking advantage of promising solutions. After several generations, the individuals tend to present, in most cases, similar characteristics. An increase in the mutation rate at this stage of the evolution will lead to a dispersion of the population by bringing new genetic material for the development of the fittest individuals.



(a) Genetic Operators, Normal Technique

(b) Genetic Operators, Roulette Technique

Figure 3: Genetic algorithm's configuration parameters

The operator roulette randomly chooses one operator for mutation or for recombination related to each new individual created in the reproductive process.

The initial and final weights of the operators, as well as the interpolation rate, are user-defined algorithm parameters (Fig. 3).

In the genetic algorithm configuration strategy (Fig. 3), if the operator roulette technique is chosen (Fig. 3b), all operators are automatically selected to be presented to the roulette, with an application probability of 100%. The probability field shows the operator application probability. The fields related to weight initial and weight end indicate the initial and final weights that are linearly interpolated along the evolution. The weights, in a certain evolution, are used as the probability of selecting an operator. The global sum of all the weights is constant at every step and equal to 100%.

6.5 Reproduction

Reproduction techniques determine the individual substitution criteria in a population to produce the next generation. In this investigation the elitist population change and the population partial replacement without duplicated individuals were implemented.

6.5.1 Elitist replacement of the entire population

In this technique, all chromosomes are replaced. The fittest chromosome of the current population is copied to the next, after mutation and crossover. The use of this technique reduces the random effect of the selection process and guarantees that the best individual of the future generations is better or equal to its equivalent in the previous one [10].

6.5.2 Partial replacement of the population without duplicated individuals

The technique of partially replacing the population without duplicated individuals was created from the standard partial population replacement technique. In this case *n* individuals are generated to replace the worst individuals in the current population. The number of replaced individuals is known as GAP (see section 6.5.3).

The population partial replacement without duplicated individuals inherits the elitism property from the standard partial population replacement. This implies that the population tends to be kept constant, thus allowing the use of less conservative operators such as uniform crossover [10]. By not allowing the presence of duplicated individuals, this technique guarantees the better performance of the GA intrinsic parallelism, according to which *n* different points in the search space are evaluated at every cycle. However, this implies an overhead for the detection of duplicated individuals and for the creation of new elements.

The standard partial population replacement technique and the partial population replacement without duplicated individuals are denominated by some authors [10] as *Steady-state* and *Steady-state without duplicates*, respectively. However, according to Wu, Chow and Mitchell [8][9], *Steady-state reproduction* is only related to the replacement of one or two individuals at a time.

6.5.3 Generation Gap - GAP

GAP is a parameter that controls the percentage of the population to be replaced at every generation. The GAP value in traditional GAs is equal to 1.0 (100%), indicating that all of the population is replaced at every generation. Additionally, the population structure could be corrupted and fit individuals fail to be transferred to the next generation, leading to a decrease in the convergence rate [8].

According to Wu and Chow [8], to avoid problems related to the use of a GAP equal to 1.0 (100%) a small GAP should be used, i.e.:

$$GAP = \frac{2}{N_P} \tag{11}$$

where N_p is the number of individuals in the population. This means that only two individuals are selected for reproduction and the two newly generated individuals replace the worst individuals in the population. This strategy leads to a significant reduction in the number of evaluations of the objective function and, consequently, reduces the number of structural analyses to be performed by the FEMOOP program [3].

Generally these two newly-created individuals have a higher fitness than the worst individuals in the current population. Additionally, even if a large population could still be kept, the algorithm should still have sufficient power for performing the evolution.

Even with the previously described GAP value defined by Wu and Chow [8], a variable GAP strategy was implemented here to allow the user to have more flexibility. The GAP value defined in the graphical interface presented in Fig. 3 (reproduction technique: Steady-state without duplicated) is represented in terms of the percentage of individuals to be replaced at every generation.

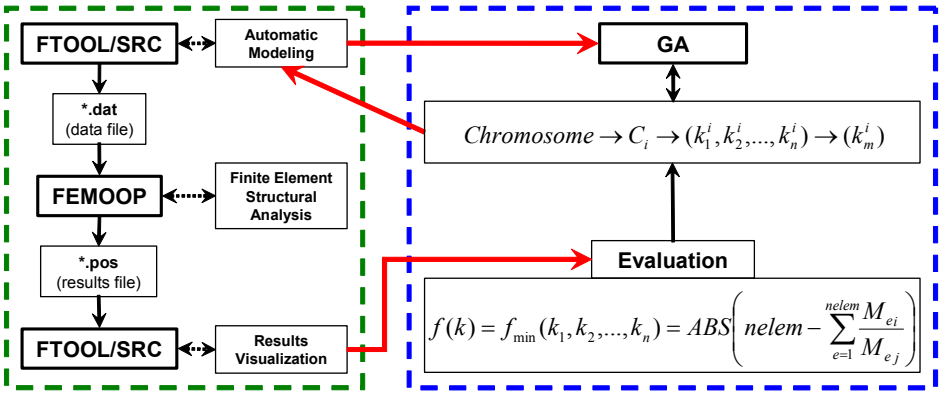


Figure 4: Evaluation algorithm for each individual.

7 Individual Evaluation Process

Figure 4 depicts the algorithm procedure for each individual. The evaluation of each individual consists in the automatic modeling of the structure made with the FTOOL/SRC program, with the design variables defined by the chromosome (Eq. 4). From this modeling a neutral file is created to be transferred to the finite-element structural analysis program FEMOOP [3]. This program calculates the internal stresses, forces and displacements, and transfers them to another neutral file. This new file is used by the FTOOL/SRC [1,2] program, which presents its results graphically and determines the bending moment distribution at the bar elements, making the chromosome evaluation (Eq. 3) and directing these values to the genetic algorithm.

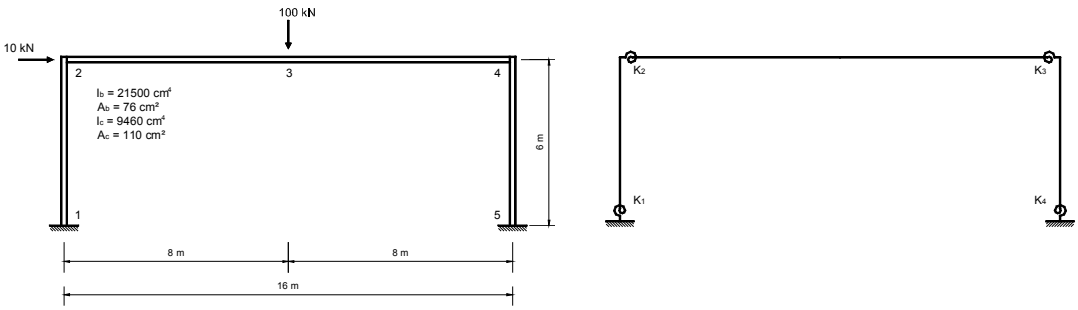
8 Numerical Examples

In this section, two numerical optimization examples of the implemented genetic algorithm are presented. The first example is related to a simple semi-rigid steel frame while the second involves a vierendeel girder system, both semi-rigid.

Both examples adopted: ten experiments with 50 generations each; a population of 50 individuals; the selection technique including an accumulated relative evaluation; steady-state reproduction with a GAP value of 50; and the operator roulette technique. The initial/end weights of the operators were, respectively, 30-10, 40-30, 15-10 and 15-50, for a single-point crossover, an arithmetic crossover, a real mutation and a self-adjusted mutation. The limits of the search domain for the problem variable were, for the semi-rigid steel frame, zero (hinge) and 1.0e+08 (rigid), while the vierendel beam example used zero as the lower limit and two upper limits of 1.0e+16 and 1.0e+08.

8.1 Semi-Rigid Steel Frame

The optimization was made on a 16 meter span frame, 6 meters high, subjected to a 100 kN vertical load at the beam centre span and to a 10 kN horizontal load located at the top of the first column, as represented in Fig. 5(a).

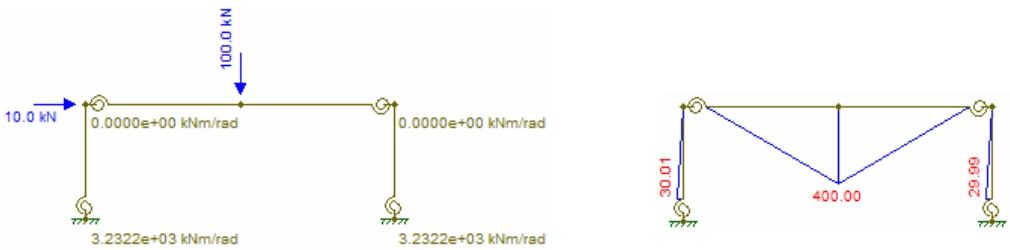


(a) Frame supports and loads. (b) Idealized structural model.

Figure 5: First steel frame example.

This structure was modelled varying its joint stiffness values. Figure 5(b) depicts the idealized structural model with springs to represent the joint elements. In this model k_2 and k_3 represent the beam-to-column joint stiffness, and k_3 and k_4 represent the base plate connection stiffness. Initially a pinned configuration was used in the beam-to-column joint while the base plate adopted an intermediate stiffness related to the column stiffness, EI_c/L_c, where I_c and L_c represent the column inertia and length.

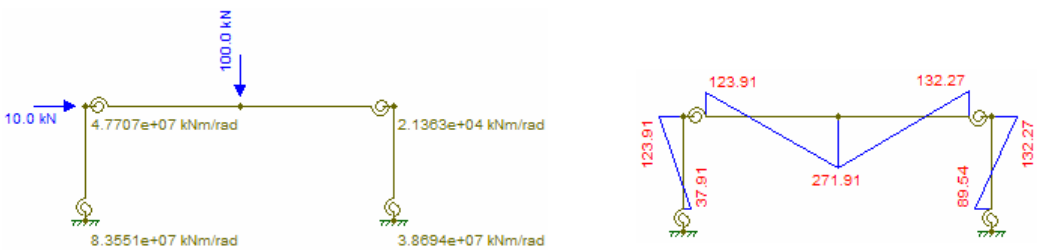
These joint stiffness values were used as the seeds of the genetic algorithm. Figure 6(a) illustrates the semi-rigid joint initial modelling and Fig. 6(b) depicts the associate frame bending moment distribution.



(a) (b)

Figure 6: (a) Initial semi-rigid joint stiffness values; (b) associate bending moment distribution.

Figure 7 shows the GA final results in terms of the optimised bending moment distribution and associate semi-rigid joint stiffness values.



(a) (b)

Figure 7: (a) Final GA semi-rigid joint stiffness values; (b) associate bending moment distribution.

A simple comparison can be made between the initial modelling values (Fig. 6) and the final GA results (Fig. 7). A significant improvement can be clearly seen. The initial bending moment values at the centre span and supports changed from 400 kNm and zero in the initial configuration to 271.91 kNm (centre span) and 123.91 kNm and 32.27 kNm (left and right supports) in the final GA results, as can be seen in Fig. 7(b).

A random exhaustive search procedure was used to calibrate the efficiency of the GA results. An average performance curve was produced for the exhaustive search procedure with the same number of iterations used in the GA method already described. These results are presented in Fig. 8.

When the GA results are compared with the random search procedure (Fig. 8), the GA method converged, at the 26th generation, to a value of 0.295 while no convergence was found with the other method. This fact demonstrates that the GA method has a better performance than the random search procedure.

8.2 A Vierendeel Girder Semi-Rigid Structural System

The second example concerned the optimisation of a 12-meter span vierendeel structural system, one meter height. The columns (vertical elements) and beams (horizontal elements) were, respectively, HEB240 and IPE240 steel profiles. Figure 9 presents (a) the rigid joint initial modelling used as seeds of the GA, and (b) the associate frame bending moment distribution.

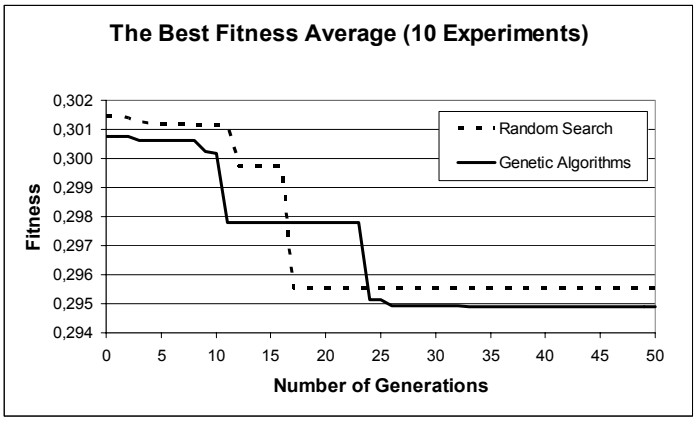
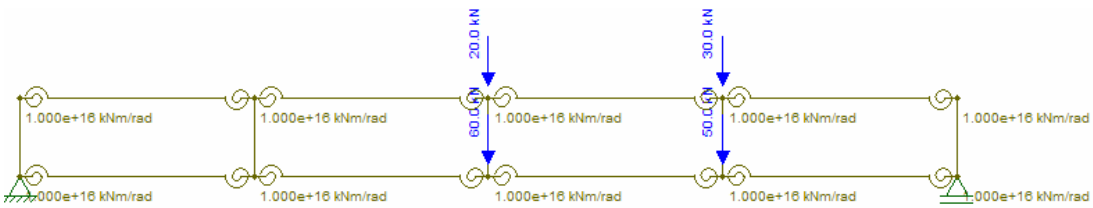
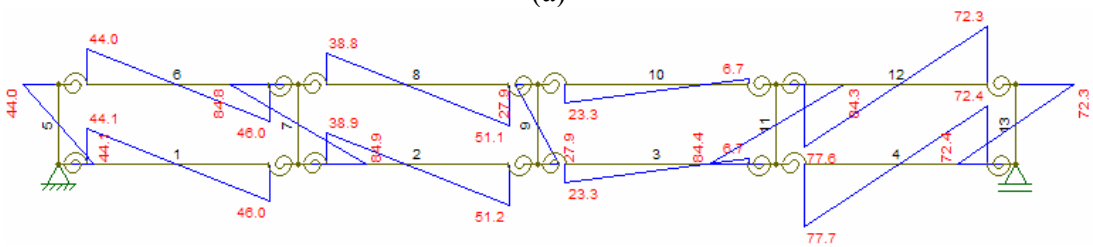


Figure 8: Genetic algorithm and random search comparison (ten experiments).



(a)



(b)

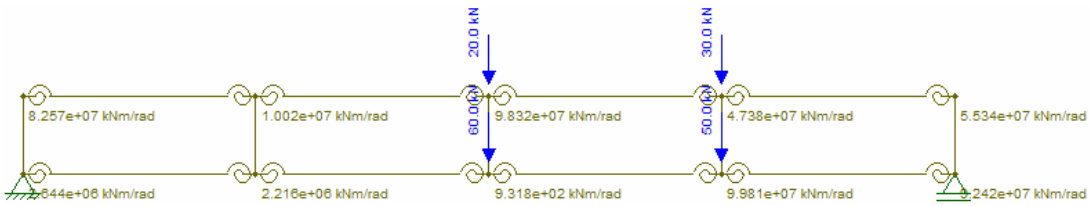
Figure 9: (a) Initial semi-rigid joint stiffness values; (b) associate bending moment distribution.

Figure 10 depicts the GA final results in terms of the optimised bending moment distribution and associate semi-rigid joint stiffness values. These results were obtained, as previously mentioned, by varying the joint stiffness values. Figures 11

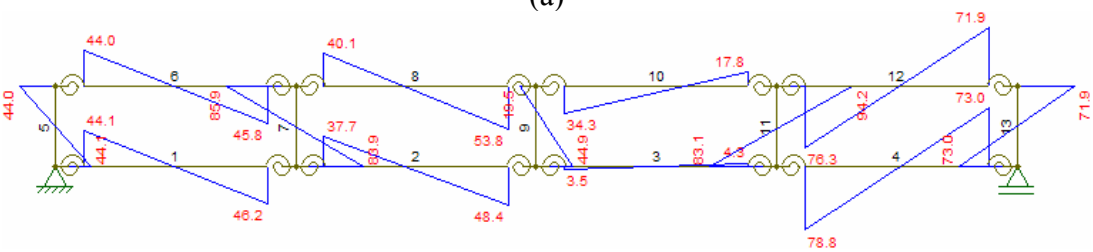
and 12 illustrate the GA performance in terms of an average curve of the best evaluations in ten experiments and of the best evaluations in 100 individuals associated to one experiment.

When the initially rigid modelling (Fig. 9a) is compared to the GA final results (Fig. 10a) it is evident that the GA method led to semi-rigid joint configurations with associate stiffness values significantly smaller than the rigid joint values. The new joint stiffness configuration led to a considerable improvement of the bending moment distribution. Detailed results of these comparisons are presented in Tab. 1.

A close inspection of Fig. 11 indicates that the search for the best average evaluations is influenced by the limits of the variable search space. When the search space was limited by zero and 1.0e+16, the GA method led to an average evaluation of 3.539. Alternatively, when the search space was limited by zero and 1.0e+8, the GA method conducted to an average evaluation of 0.668.



(a)



(b)

Figure 10: (a) Final GA semi-rigid joint stiffness values; (b) associate bending moment distribution.

	Stiffnes of Joints (kNm/rad)				Bending Moments (kNm)			
	Rigid		Semi-Ridid		Rigid		Semi-Ridid	
Element	Node i	Node j	Node i	Node j	Node i	Node j	Node i	Node j
1	1,000E+16	1,000E+16	2,644E+06	8,911E+07	-44,10	46,00	-44,10	46,20
2	1,000E+16	1,000E+16	2,216E+06	7,786E+07	-38,90	51,20	-37,70	48,40
3	1,000E+16	1,000E+16	9,318E+02	6,448E+07	23,30	-6,70	3,50	-4,30
4	1,000E+16	1,000E+16	9,981E+07	9,242E+07	77,70	-72,40	78,80	-73,00
5	1,000E+16	1,000E+16	1,000E+16	1,000E+16	44,10	-44,00	44,10	-44,00
6	1,000E+16	1,000E+16	8,257E+07	3,006E+07	-44,00	46,00	-44,00	45,80
7	1,000E+16	1,000E+16	1,000E+16	1,000E+16	-84,80	84,90	-85,90	83,90
8	1,000E+16	1,000E+16	1,002E+07	4,561E+07	-38,80	51,10	-40,10	53,80
9	1,000E+16	1,000E+16	1,000E+16	1,000E+16	-27,90	27,90	-19,50	44,90
10	1,000E+16	1,000E+16	9,832E+07	6,270E+07	23,30	-6,70	34,30	-17,80
11	1,000E+16	1,000E+16	1,000E+16	1,000E+16	84,30	-84,40	94,20	-83,10
12	1,000E+16	1,000E+16	4,738E+07	5,534E+07	77,60	-72,30	76,30	-71,90
13	1,000E+16	1,000E+16	1,000E+16	1,000E+16	72,30	-72,40	71,90	-73,00

Table 1: Comparison between initial modelling (Fig. 9) and final GA results (Fig. 10).

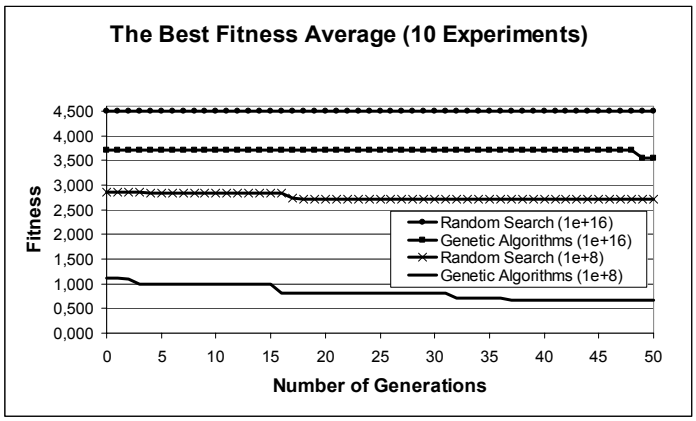


Figure 11: Comparison between genetic algorithm and random search (ten experiments).

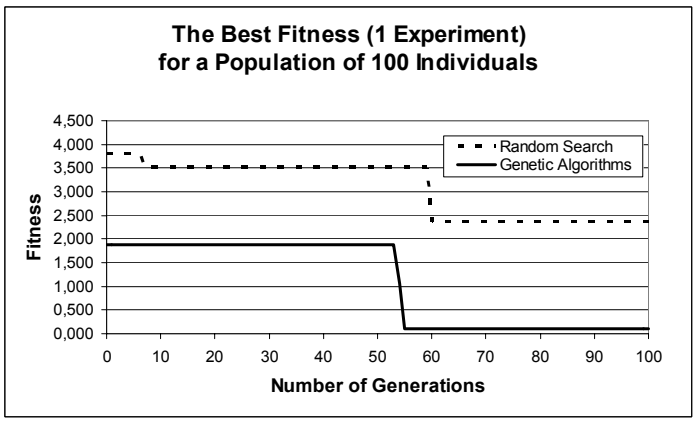


Figure 12: Comparison between genetic algorithm and random search (one experiment, for a population of 100 individuals).

In the comparisons made in Figs. 11 and 12 the GA demonstrated a superior performance than the random search procedure. When the best evaluation achieved with the use of the random search procedure (2.364) compared with the GA method (0.113), a difference of 2.251 is found (Fig 12). If this process is repeated for the results of Fig. 11, differences of 2.042 and 0.969 were found for the search space with superior limits of 1.0e+08 and 1.0e+16. These results, as well as the results of the previous example, validate the GA implementation as well as the adopted methodology used in the problem evaluation function.

9 Conclusions

The present investigation is centered in a non-conventional semi-rigid joint stiffness optimisation procedure where individuals were evaluated to generate an optimum bending moment distribution for a structure. This search methodology, varying the joint stiffness values, was applied to two different examples. Three important points deserve to be highlighted:

1) The best solution was always associated to the genetic algorithm when compared to a random search procedure. The difference between these two methods is amplified when the variable search space is increased. This difference was also magnified by increasing the structural problem complexity.
2) A significantly better bending moment distribution was achieved when the objective function adopted in the present implementation was globally applied to the investigated problems.
3) The second example (vierendel structural system) pointed out that the semi-rigid solution performance, found with the GA method, led to better results than the usually adopted rigid solution. The semi-rigid joints created a balanced moment distribution with smooth transition between the sagging and the hogging moment regions.

An optimum bending moment distribution is a direct consequence of an adequate choice of the structural joint stiffness. This is usually a complex procedure when made manually due to the tedious and significant amount of work required to define a joint stiffness combination that could lead to a balanced bending moment distribution. The use of genetic algorithms considerably simplified this process, leading to the determination of bending moment distributions that could generate efficient and economic steel structures.

The results of the examples here presented indicated that the implemented GA procedure and associated methodology are efficient, representing an important contribution to the development of steel frame structural design with semi-rigid joints.

The future steps of this investigation will contemplate the implementation of some evaluation functions usually adopted in practice, which generally minimises the structure weight and consequently its final costs. Additionally, new restrictions will be incorporated to the system, related for instance to displacements and stresses present in structural elements.

References

[1] A.A. Del Savio, S.A. de Andrade, P.C.G.S. Vellasco, L.F. Martha, *"Non-Linear Joint Finite Element Formulation for Semi-Rigid Steel Structures Analysis"*, submitted for publication, Adv. in Engineering Software, 2004.

[2] A.A. Del Savio, S.A. de Andrade, P.C.G.S. Vellasco, L.F. Martha, *"A Non-Linear System for Semi-Rigid Steel Portal Frame Analysis"*, Proceedings of the Seventh International Conference on Computational Structures Technology - CST2004, CD-Rom, 12 pages, 2004.

[3] L.F. Martha, E. Parente Jr., *"An Object-Oriented Framework for Finite Element Programming"*, Proceedings of the Fifth World Congress on Computational Mechanics, IACM, Vienna, Austria, 2002.

[4] Z. Michalewicz, *"Genetic Algorithms + Data Structures = Evolution Programs"*, Springer-Verlag, Berlin Heidelberg New York, 1996

[5] H.J.C. Barbosa, *"Algoritmos Genéticos para Otimização em Engenharia: Uma Introdução aos Algoritmos Genéticos"*, Lecture Notes, 2ª Escola de Verão em Computação Científica, Laboratório Nacional de Computação Científica – LNCC and UFJF, Rio de Janeiro, Brazil, 1997 (in Portuguese).

[6] K.A. De Jong, *"Genetic algorithms are NOT function optimizers"*, In FOGA-92, Foundations of Genetic Algorithms, Vail, Colorado, pp. 24-29, July 1992.

[7] E. Michielssen, D.S. Weile, *"Electromagnetic system design using genetic algorithms"*, G.Winter, et al, ed, Genetic Algorithms in Engineering and Computer Science, pp.345-369, John Wiley & Sons, 1995.

[8] S.J. Wu, P.T. Chow, *"Steady-State Genetic Algorithms for Discrete Optimization of Trusses"*, Computers & Structures, vol. 56, no. 6, pp. 976-991, Pergamon Press, UK, 1995.

[9] M. Mitchell, *"L.D. Davis, Handbook of Genetic Algorithms"*, Artificial Intelligence, 100, pp. 325-330, Elsevier Science B.V. Press, 1998.

[10] M.A.C. Pacheco, *"Computação Evolucionária: Algoritmos Genéticos"*, Lecture Notes, ICA: Lab. de Inteligência Computacional Aplicada, Electrical Eng. Dept. – PUC-Rio, Rio de Janeiro, Brazil, 2004 (in Portuguese).