

RESEARCH ARTICLE

## Cost optimization of reinforced concrete frames using genetic algorithms

Bedilu Habte<sup>\*</sup>, Elias Yilma

*School of Civil & Environmental Engineering, Addis Ababa Institute of Technology, Ethiopia*  
*bedilu@web.de, elias.yilma@aait.edu.et*

---

ARTICLE INFO

Article history:

*Received: 11 September 2019*

*Accepted: 27 August 2020*

*Available Online: 27 December 2020*

Keywords:

*Structural Design Optimization*

*Decimal Encoding*

*Genetic Algorithms*

*Direct Stiffness Method*

*Reinforcement Detailing in Frames*

---

AMS Classification 2010:

*46N10, 65K10*

ABSTRACT

Cost optimization of reinforced concrete building frames using genetic algorithms is presented. Unlike previous works that used simplified discrete or continuous optimization models, this work considers constructability issues as well as the effects of shear and torsional actions in the design optimization of reinforced concrete frames. An integrated software system has been developed to implement the proposed optimization procedure using genetic algorithms. Examples have been incorporated in order to compare the results from the proposed study with that of a previous work which follows a different heuristic and with the traditional “design–check–revise” method. The structural design procedures recommended in the Eurocode-2 have been strictly followed in this work. Special emphasis has been given to structural analysis methods and studying computational efficiency of the developed framework. To improve the performance and computational complexity of the algorithm, the effect of genetic parameters such as mutation and crossover on the optimization process has been thoroughly studied. The method developed in this work proves to have a lot of advantages over the traditional “design–check–revise” paradigm and other heuristic methods.



---

### 1. Introduction

The cost optimization problem of reinforced concrete frames is a complex one to tackle. This stems from the fact that reinforced concrete structures are heterogeneous in that they are composed of both concrete and steel. This consequently imposes a contradictory constraint on the formulation of the objective functions for cost, since the quantities of concrete and steel required for a certain structure vary inversely. Considering the sheer number of design variables from the sizing to the reinforcements and their arrangements, calculus-based methods get ruled out for such tasks.

With the computational power of computers ever increasing, recent years have yielded substantial progress into non-deterministic search-based optimization methods for structural design problems. One such family of methods is genetic algorithms. Genetic algorithms (GAs) are meta-heuristic search algorithms based on the Darwinian theory of natural selection and genetics [1]. They are based on selecting solutions that are best fit to the problem at hand while maintaining a diverse set of possible solutions which change and evolve upon subsequent iterations.

Developed by John Holland [2] in the 1970s, GAs have been successfully applied in several structural design and optimization problems [3, 4, 5], including but not limited to cost [6, 7], weight [8] and topology optimization problems [9].

Most reinforced concrete structural optimization problems are treated as continuous in their function form, and the design variables can theoretically assume any positive real number. Hence, solutions usually consist of a set of real numbers. In structural design practice however, exact values are approximated into discrete sizes and quantities, which are subsequently used during construction. Discretization techniques through discrete longitudinal [10] and section-wise [11] reinforcement profiles as well as the use of reinforcement ratios [12] have been employed in previous works to get discretely optimized solutions for design problems and are directly compatible with GA formulations.

### 2. Genetic algorithms for structural optimization

GAs operate on a collection of individuals, usually called a population. Individuals are sets containing potential values of the structural design variables; they represent single points in the design search space or

---

<sup>\*</sup>Corresponding author

domain.

The process of genetic evolution generally consists of three stages: initialization, selection and generation. In order to make the selection and evolution process manageable, the information contained in the individual solutions is transformed into encoded strings of character sequences called chromosomes.

During initialization, an encoded and randomized set of  $n$  (?) individuals is generated. This randomization allows for the exploration of the search space and for maintaining diversity in the population [2,13]. Several modes of encoding design information into chromosomes have been proposed [3]. Binary digits (the dominant encoding type) were used in the early works of the use of genetic algorithms for truss optimization [8, 11, 14]. Later on, other studies [15] have adopted decimal and value encoding to successfully perform discrete optimization of structures. Dede et.al. [3] have shown that the type of encoding can not necessarily guarantee better solutions and that quality of solution depends on several factors such as nature of mutation, type of crossover and population size. The quality of solutions also depends on the size and complexity of the problem.

In the selection stage, a genetic operator called reproduction is applied, where a percentage of the population that consists of individuals that best fit the problem are selected while the rest of the individuals are discarded. The selection process varies among implementations like tournament selection as in [9, 11] and roulette selection as in [12, 15].

The generation stage consists of applying genetic operators on the selected individuals from the previous stage. Genetic operators form new individuals with mixed genetic materials and randomly modify existing individuals. The most common genetic operators used include reproduction, crossover and mutation.

Most attempts on structural optimization problems assumed only the most critical variables such as section type and material quantity. Later approaches tried to include variables such as classification of reinforcement as negative and positive [9]. Rajeev and Krishnamoorthy [11] have shown the inclusion of simple reinforcement detailing patterns as a design variable in the formulation of the optimization problem. This was achieved using reinforcement arrangements recommended in current construction practices or from guidelines found in detailing manuals. It should, however, be noted that the detailing arrangements taken into consideration did not reflect other detailing parameters such as anchoring, placement and development lengths, bends and overlaps. Additionally, as such arrangements take more complex forms, the number of design variables and thus, chromosome length, will also increase. This makes the decoding of the chromosomes complex, time-consuming and memory intensive. To alleviate this problem, grouping of similar structural elements has been used to reduce the size of chromosomes [11]. On the other hand,

Vidoso et.al. [10] have used a structural modeling scheme that incorporated symmetry in structures as an aid in reducing computational time and the number of design variables involved.

The models used to optimize plane portal frames in some previous works [10, 11, 12] can be extended to three dimensional portal frames. This involves the need for a three dimensional structural analysis, the addition of torsional forces, as well as biaxial bending effects to the design procedures. Reinforcement detailing for both the major and minor directions in shear and bending would also need to be added as design variables.

In structural optimization, a re-analysis of the structure would always be mandatory as the dimensions of the structural members are changed or modified. The most significant contribution of this research work is the development of a separate structural analysis module and its integration with the genetic algorithm component. Thereby, structural analysis is performed on every individual of the population in all the generations in order to obtain the actual internal action and member resistance. Additionally, this research addresses the problem of using the appropriate number of design variables without compromising the usability of the results, while, at the same time, minimizing the overall computational footprint of an optimization simulation.

### 3. Formulation of the optimization problem

#### 3.1. General

Given a three dimensional frame composed of  $H$  horizontal members,  $V$  vertical members, and  $l$  regions of critical internal action, general formulation for structural cost optimization is obtained by minimizing the total cost of concrete, reinforcing steel and formwork used, as shown in Eq. (1).

$$\text{Min } O = \sum_k \{ C_{c,k} + C_{s,k} + C_{f,k} \} \quad (1)$$

Subject to:

$$\left. \begin{aligned} M_{ED,k} &\leq M_{rd,k} & k \in H \\ V_{ED,k} &\leq V_{rd,k} & k \in H \\ T_{ED,k} &\leq T_{rd,s,k} & k \in H \\ \frac{N_k}{N_{u,k}} + \frac{M_{yk}}{M_{yu,k}} + \frac{M_{zk}}{M_{zu,k}} &\leq 1 & k \in (H \cup V) \\ \frac{V_{ED,k}}{V_{ED,max,k}} + \frac{T_{ED,k}}{T_{ED,max,k}} &\leq 1 & k \in (H \cup V) \\ A_{sk,min} &\leq A_{sk} \leq A_{sk,max} & k \in (H \cup V) \end{aligned} \right\} \quad (2)$$

Where the decision variables are the cross-sectional dimensions of the members (beams  $H$  and columns  $V$ ),  $b_k$ ,  $h_k$ , as well as the main and shear reinforcements provided in the elements, symbolically represented by  $A_{s,l,k}$  and  $A_{sv,k}$  in which  $k \in (H \cup V)$ ,  $l \in X_{f,k}$ ,

$X_{f,k}$  represents the set of all unique internal action regions; major and minor axes moments  $M$ , shear forces  $V$ , axial forces  $N$ , and torsional forces  $T$ . The subscripts  $ED(sd)$  and  $rd$  represent the design actions and the design resistances respectively.  $C_{c,k}$ ,  $C_{s,k}$ ,  $C_{f,k}$  represent the costs of concrete, reinforcing steel and formworks respectively.

In Eq. (1), the individual costs for concrete and reinforcing steel as well as form-work and scaffolding could be determined, respectively, using Eq. (3). This is based on the prevailing unit rates for the three components:  $c_c$ ,  $c_s$  and  $c_f$ .

$$\left. \begin{aligned} C_{c,k} &= c_c b_k h_k l_k & k \in (H \cup V) \\ C_{s,k} &= c_s \gamma_k b_k h_k l_k & k \in (H \cup V) \\ C_{f,k} &= \begin{cases} 2c_f (b_k + h_k) l_k & k \in V \\ c_{sc} b_k l_k + c_f (b_k + 2h_k) l_k & k \in H \end{cases} \end{aligned} \right\} \quad (3)$$

### 3.2. Structural analysis

Structural optimization demands a reanalysis of the structure specially when the basic dimensions of the structure are altered during the design stages. An independent finite-element program has been developed in this work for the elastic analysis of framed structures, the details of which can be found in [16]. Structural analysis takes up the most amount of time as compared to other processes, such as design or post-processing. The program developed in this work incorporates available programming techniques in order to boost its memory efficiency and speed; namely, the use of sparse matrices [17] for the storage of the associated structural data and the use of the conjugate gradient algorithm to iteratively solve the resulting stiffness equation. The analysis provides design values of the different actions on the members of the frame at critical sections.

### 3.3. Design checking as constraints

As provided in Eq. (2), the resistance of each member with respect to the internal actions needs to be higher than the corresponding design values in order for the structural design to be considered safe [18, 19].

#### 3.3.1. Check for bending

Based on the procedures described in [18], sections subjected to combined axial force and bending moments need to be designed for the combined stress interactions developed. For each element under biaxial bending, direct procedure of constructing interaction diagrams has also been used to check the available capacity of the sections. Interaction checks are made in one direction initiating uniaxial interaction function provided the criteria in Eq. (4) is satisfied.

$$\left. \begin{aligned} \frac{e_y}{h} / \frac{e_z}{b} &\leq 0.2 \\ \frac{e_y}{h} / \frac{e_z}{b} &\geq 5.0 \end{aligned} \right\} \quad (4)$$

Where  $e_y$  and  $e_z$  are the eccentricities along the respective axes. Any member that doesn't satisfy this condition is treated to behave in a biaxial manner. This case can be handled by constructing biaxial interaction diagrams for the current orientation  $\theta$  that the design moment in the major direction makes with the minor one. The fitness of a column is computed by evaluating how far the interaction contour containing the bending moment vs. axial compressive force coordinate (M-A coordinate) is from the interaction boundary. This distance has been normalized to an efficiency coefficient,  $\epsilon_{biaxial}$ , which normally ranges from 0 to 1. The farther the interaction coordinate from the contour, the larger the reserve capacity (i.e.,  $\epsilon_{biaxial}$  is low). If the M-A coordinate is located outside the interaction boundary (and  $\epsilon_{biaxial} > 1.0$ ), then the section has failed the check, necessitating a larger penalty to be applied according to the values suggested in Table 1.

**Table 1.** Penalty values for biaxial bending

Condition for penalty	Penalty magnitude
$\epsilon_{biaxial} = 0.0$	1.0
$0.0 < \epsilon_{biaxial} < 1.0$	$1.0 - \epsilon_{biaxial}$
$\epsilon_{biaxial} > 1.0$	4.0

#### 3.3.2. Check for shear

The design for pure shear in both the major and minor directions involves computing the resistance provided by concrete through strut action [19, 20]. Because the shear reinforcement spacing for the member has already been provided, the procedure is supposed to check the adequacy of these values. Finally, the penalty factor for shear capacity is computed for both major and minor directions based on the stress-vs-capacity ratio.

#### 3.3.3. Check for torsion

The procedure for checking torsional capacity is similar to that of pure shear. Additionally, members subjected to both shear and torsion require an interaction check given by Eq. (5).

$$\frac{V_{sd,i}}{V_{RD,max}} + \frac{T_{sd,i}}{T_{RD,max}} \leq 1 \quad (5)$$

The computation of efficiency and boundary checks follow the same mechanisms as discussed in biaxial interaction checks. The effect of torsional moments on a member needs to be reflected on both the longitudinal and shear reinforcements.

#### 3.3.4. Sizing constraints

In the design of concrete structures, the Eurocode dictates the following additional requirements to be fulfilled.

**Minimum width:** ... all beams and columns shall have a minimum width of 200mm assuming a 60 minutes duration of fire resistance [21]. Since the algorithm implemented in this study selects section sizes randomly, certain limits have been placed on these sizes, so as to avoid deep/narrow or thin/wide members; i.e, a  $b/h$  ratio of 0.5 has been taken as a limiting criteria.

**Maximum and minimum reinforcement ratio:** ... a minimum amount of reinforcement (shall) be placed in members that do not require strength reinforcement. Maximum reinforcement limits are provided such that the ease of constructability in placing reinforcement bars and pouring concrete is ensured [19].

#### 4. Genetic modeling

Genetic algorithms operate on collections of randomly generated strings of data, assembled from encoded values of the design variables, which later should be decoded in order for the data stored to be utilized. In this study, a single chromosome is used to represent all geometric and reinforcement data contained within a single structure. This includes the sizes (widths and heights) of each member in the structure, and flexural and shear reinforcements for critical regions of all members.

##### 4.1. Encoding member sizes

Section sizes for RC members are encoded discretely, where each integer represents a single width/height value in the individual chromosome. It is possible to obtain width-height combinations that would result in wide, deep or narrow sections; but these can be avoided by utilizing the sizing constraints. Starting from a value of 200mm, increments of 25mm have been applied up to the maximum dimension for both the breadth and height.

##### 4.2. Encoding reinforcements

Each member has two basic types of flexural reinforcements assigned to it: continuous reinforcements located at the top and bottom of the member, and additional reinforcements for regions of critical negative and positive bending stresses. Continuous reinforcements have a simple indexing scheme based on all available diameters of reinforcing bars, and two bars were used at the top and bottom sides of each member.

Four values are used in the encoding scheme: two indices for positive reinforcements in the Y- and Z- directions, and the remaining two assigned for negative reinforcements. Areas of reinforcements for positive and negative bending regions in a section could then be computed by adding up the reinforcement areas for the continuous and additional reinforcements.

Shear reinforcements for both beams and columns are provided in the form of stirrups. For convenience, 8mm bars have been used as stirrups with varying spacing for each member, where the corresponding indices in the

chromosomes represent discrete spacing values. In this work, the stirrup spacing ranges between 100 and 350mm, with alternating 20 and 30 mm increments in between.

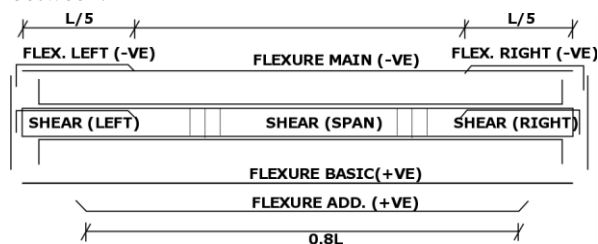


Figure 1. Reinforcement configuration for beams

#### 5. Genetic optimization

##### 5.1. Performance of a structure

The estimated performance of a structural design is based on two criteria: its structural performance under loading and the corresponding total cost. In order to evaluate how acceptable a design is on the basis of both these aspects, a combined expression that incorporates these values would be needed. This expression, called the fitness function, is the primary criteria through which the evolutionary selection process is carried out in this work.

##### 5.2. The fitness function

The fitness function evaluates the total performance of the structure and lets the genetic algorithms use its result as a means to identify whether that particular design is the best fit or not. The general expression is given by Eq. (6).

$$F = C \cdot (1 + p) \quad (6)$$

Where  $F$  is the fitness value,  $C$ , the total cost of the structure and  $p$ , the structural penalty. The fitness function acts as a weighted visualizer of the design's performance, whereas the penalty and cost parameters only illuminate part of this evaluation.

##### 5.3. Optimization procedure

The entire genetic optimization procedure is described in this sub-section. The optimization process starts by generating the population. After that, decoded values of the population are prepared to be used in the analysis, design and checking of the frame. Fitness values of each individual are then computed, followed by selection of best fit members. In the next step new individuals are generated using mutation and crossover.

##### 5.3.1. Population generation

As value-encoded individuals are used here, a series of individuals, each with an array of property indices (the chromosome), are generated randomly. Each index in the chromosome encodes a specific property such as a specific structural member's width or a specific diameter of reinforcement used in a single member.

For a single structure, 8 indices per each beam member,

and 5 indices per each column member, are needed for encoding. Thus, the total chromosome length for a structure that has  $n$  horizontal members and  $m$  vertical members is given by the expression in Eq. (7).

$$L = 8 \cdot n + 5 \cdot m \quad (7)$$

Once series of individuals are created, the next step is to get the actual decoded values from their chromosomes.

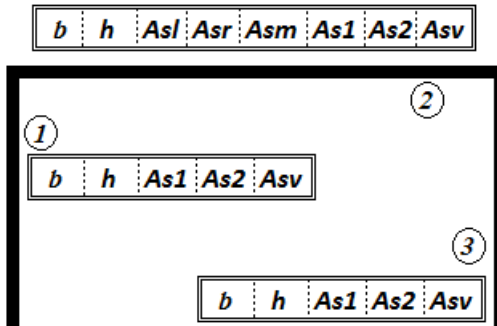


Figure 2. Member encoding representation

**5.3.2. Decoding individuals**

During the gene decoding process, an index is used to get the actual width value of the member from an array that has all possible values. For example, if the array [225,250,300,350,400,450,500] is representing all possible values of the width of a member, respectively; the indices 0, 1, 2, 3, 4, 5, and 6 stand for the respective encoded values of the width; whereas an index of 3 is decoded as a width value of 350mm.

Once all values for each variable have been encoded, frame structures are formed using the individual members.

**5.3.3. Evaluation of fitness**

After forming a series of structures corresponding to each individual, every structure needs to be evaluated based on the procedure outlined in section 5.2. In general, each structure is analyzed and evaluated for all constraints outlined in section 3.3. Throughout the evaluation process, penalty values corresponding to the constraint checks are added. The cost of the structure is also computed according to set unit rates for labor, material and formwork. Finally, the fitness value of every member of the population is computed.

**5.3.4 Selection**

Using the computed fitness values as a comparison metric, the best fit values (about the top 90%) from the set of evaluated structures are selected, while the rest are discarded.

**5.3.5 Generation**

The next step is to generate a set of new individuals by using the genetic material of the selected individuals. Two operations are used for this: mutation and crossover. Mutation randomly changes individual chromosomes, while crossover is used to mix the gene information between two randomly selected

individuals.

During mutation, a few randomly selected indices are changed to random values.

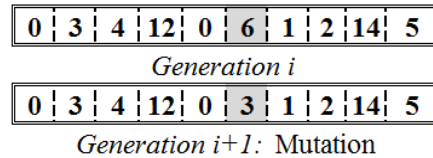


Figure 3. Mutation procedure

In this work, uniform crossover has been utilized where random indices between individuals get swapped to form new individuals.

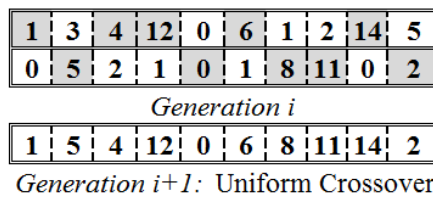


Figure 4. Uniform crossover

**5.4. Software implementation**

An integrated structural optimization software has been developed for the purpose of implementing the procedure proposed in this work. The program offers a user-interface to perform the structural analysis as well as the optimization of a reinforced concrete frame using genetic algorithm.

The software implementation for the entire structural optimization process, as developed in this work, is depicted in the flow chart shown in Figure 5.

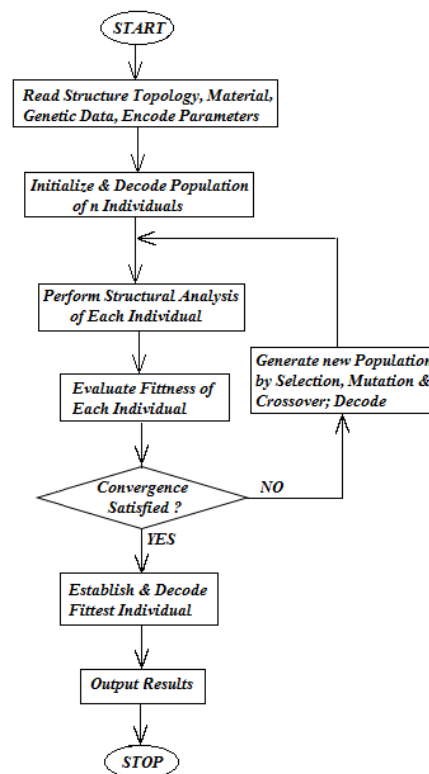


Figure 5. Structural optimization using genetic algorithm

**6. Verification**

Two example problems have been used to verify the outcomes of the genetic algorithm based design optimization procedure developed in this study. The first is comparing the result from this study with that of a previous work which used a different optimization procedure [10]. The second example considers a new simple space frame structure to compare the results from this study with that of an optimized design using the ETABS [22] software.

**6.1. Optimization of a plane frame**

The work on optimizing building frames done by Vidosa et. al. [10] has been used to compare the results with that obtained by the procedures developed in this work. The example consists of a 5-story 2-bay 2D frame (see Figure 6). In the benchmark, the structural optimization method used was “simulated annealing”, and the unit costs for the construction components are provided in Table 2.

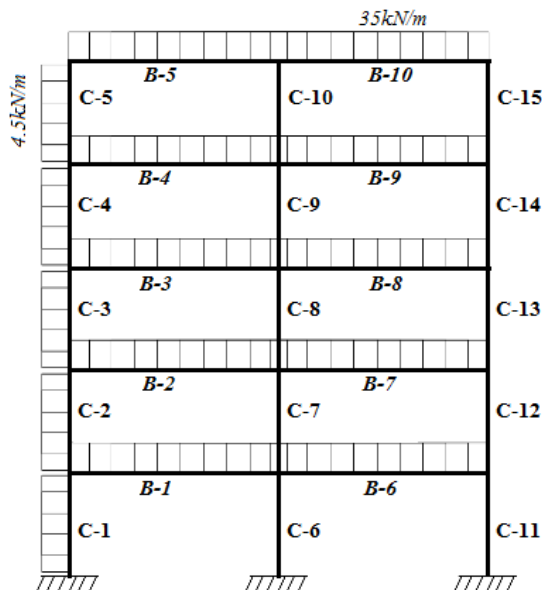
The problem used a combined vertical load of 35kN/m at each floor level (including self-weight), and a wind load magnitude of 4.5kN/m applied horizontally on columns of the left side. The simulation checked for ultimate limit states for flexure, shear and instability as well as for service conditions satisfying deflection based on the Spanish code standards. The optimization was treated as a discrete optimization problem with predefined reinforcement details and section sizing.

A cost of €4458.08 has been reported after using simulated annealing in the previous work; however, the actual cost computed by using the dimensions and following the procedure outlined in the report has been €4887.52. The simulated annealing (SA) procedure converged in 105,000 iterations and took 97 minutes to complete on a 3.2GHz processor.

**Table 2.** Unit cost (rate) for the components

Component	Units	Rate in €
500 Reinforcing steel	Kg	1.30
C35/45 Concrete	m <sup>3</sup>	112.13
Formwork for beams	m <sup>2</sup>	25.05
Formwork for columns	m <sup>2</sup>	22.75
Scaffolding for beams	m <sup>2</sup>	38.89

The genetic algorithm procedure has been performed for more than 40 runs using varied genetic parameter combinations. Population sizes varying from 40 up to 300 and mutation probabilities ranging from 1% up to 6% were used in the test runs. The optimum result was found using a mutation probability of 3.2%, a crossover probability of 50% and a population size of 250, while convergence was achieved in 200 iterations. Each simulation took about 44 seconds to complete. The best individual had a fitness value of 18439.86 with a cost of €4644.8 (as compared to €4887.52 for SA) and a total penalty value of 2.97.

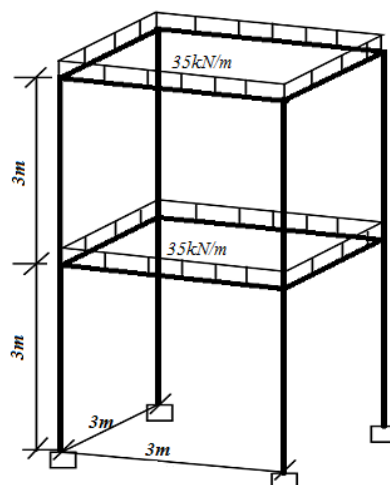


**Figure 6.** The 2-D frame and loading example

These results are close to those obtained in the research by Vidosa et. al. [11] with an improvement of about 5%. It can be observed that the program developed in this work gave the results within 0.55% of the time taken for the SA simulation to complete. Furthermore, the proposed procedure searched through a total of 250\*200=50,000 possible designs to arrive at a comparable solution with the one obtained from simulated annealing procedure, reducing the navigated search space by more than 50%.

**6.2. Optimization of a space frame**

The second example involves comparing the results obtained from the current work to that designed by the proprietary structural design package ETABS. The problem selected for comparison is a single-bay two-story 3D portal frame, as shown in Figure 7.



**Figure 7.** The 3-D frame and loading example

All longitudinal dimensions are 3m but material data and initial values have been assumed according to Table 2.

The auto-select feature of ETABS has been used for the automatic selection of concrete cross-sections from a predefined size set, instead of predefining initial cross-sectional dimensions as it is customary in the software. This entails defining all cross-sectional geometries used in the analysis, and including them to a list in the ETABS section definition dialog.

The genetic algorithm simulations were done for 35 runs and an optimal solution was found at a fitness value of 6921.22 and had a total cost of €1870.6. These results were obtained after 200 iterations, using a population size of 200, mutation probability of 7% and a crossover probability of 50%. The total number of solutions evaluated was  $200 \times 200 = 40000$ , and each simulation took 27 seconds to complete.

The ETABS auto-select design procedure resulted in a cost of €1871.57. The time benchmarks were not taken, since there were pauses between iteration confirmations, in addition to memory leaks that caused the software to freeze frequently. As it is evident, the results obtained by both software are close with an improvement of about 0.06%.

## 7. Effects of GA parameters

As the operation of genetic algorithms by itself is stochastic and unpredictable, it is important to study the effect of simulation parameters on speed, performance, convergence and most importantly, the quality of the solution. The parameters tested were population size, number of iterations and mutation probability. The 5-story by 2-bay frame used in the first example has been used here and was selected so as to have a comparable benchmark. To guarantee comprehensible results, each trial simulation has been done for a total of 5 runs. Loading configurations as well as material and cost parameters used in previous simulations have not been changed.

**Population size:** It is evident that if large population sizes are used, the probability of obtaining near optimal values is high since the initially generated population contains a large amount of genetic material that can be potentially used in the genetic algorithm progression (see Figure 8). Unlike mutation, the value of the population size varies in a single gradient fashion.

**Mutation:** Small mutation percentages can benefit the evolutionary process but misuse can result in problematic or even meaningless results. As shown in the data summary (see Figure 9), the larger the mutation value gets the larger the fitness value attained, resulting in unusable results, even though small costs were obtained for large mutations. On the contrary, a very low mutation probability can result in premature convergence with a large remnant of the original population unaltered in fitness.

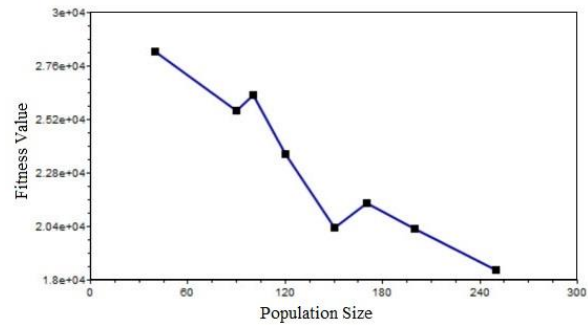


Figure 8. Effect of population size on fitness value

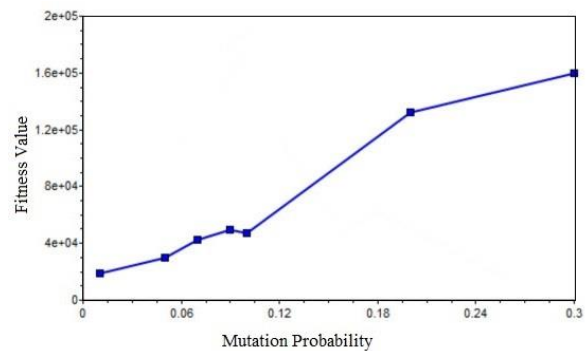


Figure 9. Effect of mutation on fitness value

**Evolution period:** The number of iterations used for a simulation is highly dependent on the population size, the mutation and crossover probabilities as well as the nature and complexity of the problem. Since large structures have several solutions and a wide search space available, the number of iterations used should be large enough so that convergence and uniformity among solutions may be achieved. On the other hand, the GA tends to converge rather soon for small structures, since they have a limited number of optimal solutions. Thus, in using large iteration magnitudes, the simulation might be exposed to divergence caused by mutation and crossover.

## 8. Performance profiling

This section quantitatively discusses the time and space complexity of genetic algorithms when applied to structural optimization of concrete frames. To profile the algorithm, regular portal frames of different sizes have been optimized and the resulting time and memory consumptions were recorded. Apart from the total degrees of freedom available, the total number of non-zero entries (NZE) in the structure stiffness matrix is a good indicator of the size of the problem. Table 3 gives a summary of the profiling results obtained for several size of frames optimized using the same genetic parameters.

### 8.1. Computation time

Estimating the total time vs. the size of the problem for increasing complexity in topology is problematic. This

happens mainly because of the variability in the time taken for structural analysis. Structural analysis takes 90-95% of the total time consumed for any particular simulation, with the rest 5-10% being mainly distributed between computation of internal actions and design check of members among other procedures. It can be seen from Table 3 that both memory consumption and simulation times increase as the structure gets larger and larger in topology.

**Table 3.** Simulation time for different frame sizes

Frame Size	DOF	$T^*_{iteration}$	$T^*_{Total}$	RAM(MB)
1x1x1	48	0.0063	4	423.4
2x2x2	162	0.045	16	460.5
3x3x3	384	0.09	49	539
5x5x5	1296	3	299	551
7x7x7	3072	15	1860	612
10x10x10	7986	136	12300	1019

$T^*$  measured in sec.

## 8.2. Memory usage

If the optimization model is applied for large problems, certain computational resources become increasingly critical. This is mainly due to the exponentially increasing number of non-zero entries in the Structural Stiffness Matrix (SSM) and its reduced variant. One such resource is the available RAM. The total utilized RAM shall be based on total RAM installed in the computer.

## 9. Conclusion

A design method for the cost optimization of two and three dimensional frames using genetic algorithms has been developed. The method proves to be a reliable design alternative in contrast to traditional trial-and-error methods. In addition, comparisons with other non-heuristic methods have shown that genetic algorithms prove to be formidable in speed and efficiency at navigating the design search space. The program developed in this work takes care of the effect of shear and torsional actions, reinforcement schedules and sizing constraints, thereby providing constructible design solutions.

Value encoding of individuals has proved to be well suited for structural optimization problems both in memory management as well as encoding and decoding schemes. The ease of changing key-value pairs also makes it a valuable contender to other encoding schemes such as binary encoding. The use of uniform crossover (in contrast to single point crossover) has achieved stronger variance between solutions, helping the algorithm efficiently navigate several regions of the search space.



It has been observed that the bulk of memory and computation time is consumed by structural analysis and the rest by the evaluation of the design algorithms. Even though most heuristic algorithms (including genetic algorithms) have exponential time and space

complexity, this work has shown that the amount of time taken per each iteration is drastically cut by using efficient evaluation techniques. Less than 1% of the time used by the Simulated Annealing method was required by the GA implementation to optimize the 2x5 frame example. A combination of multiple approaches have been utilized such as: the use of sparse matrices as a storage scheme, the use of the efficient matrix operations such as the conjugate gradient method for solving stiffness equations and the utilization of appropriate genetic encoding techniques. Through the use of such appropriate optimization formulation, the computation time and memory for the optimization procedure have been greatly reduced.

## References

- [1] Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Publishing Inc., Alabama.
- [2] Holland, J., Langton, C., & Wilson, S. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, Cambridge, Massachusetts.
- [3] Bekiroğlu, S., Dede, T., & Ayvaz, Y. (2009). Implementation of different encoding types on structural optimization based on adaptive genetic algorithm, *Finite Elements in Analysis and Design*, 45(11), 826–835.
- [4] Camp, C., Pezeshk, S., & Cao, G. (1998). Optimized Design of Two-Dimensional Structures Using a Genetic Algorithm, *Journal of Structural Engineering*, 13, 551–559.
- [5] Fang, X. (2007). *Engineering design using genetic algorithms*, Iowa State University, Iowa.
- [6] Aga, A.A.A. & Adam, F.M. (2015). Design Optimization of Reinforced Concrete Frames, *Open Journal of Civil Engineering*, 5, 74–83.
- [7] Mergos, P.E. (2016). *Optimal Design of Reinforced Concrete Frames According to EC8 and MC2010 with Genetic Algorithms*, City University London.
- [8] Prendes-Gero, M., Bello-García, A., Coz-Díaz, J., Suárez-Domínguez, F., & García P. (2018). *Optimization of steel structures with one genetic algorithm according to three international building codes*, Universidad de Oviedo EPSIG, Department of Construction, Asturias, Spain
- [9] Guerra, A., & Kioussis, P. D. (2006). Design Optimization of Reinforced Concrete Structures, *Computers and Concrete*, 3(5), 313–334.
- [10] González-Vidosa, F., Yepes, V., Alcalá, J., Carrera, M., Perea, C., & Payá-Zaforteza I. (2008). *Optimization of Reinforced Concrete Structures by Simulated Annealing*, School of Civil Engineering, Universidad Politécnica Valencia, Spain.
- [11] Rajeev, S., & Krishnamoorthy, C. S. (1998). Genetic



- Algorithm-based Methodology for Design Optimization of Reinforced Concrete Frames, *Computer-Aided Civil and Infrastructure Engineering*, 13(1), 63–74.
- [12] Najem, R.B., & Yousif, S. T. (2014). Optimum Cost Design of Reinforced Concrete Columns Using Genetic Algorithms, *Al-Rafidain Engineering*, 22(1), 112-141.
- [13] Koza, J. R. (1998). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, Massachusetts.
- [14] Mahfouz, S. (1999). *Design optimization of steel frame structures according to the British codes of practice using a genetic algorithm*, University of Bradford
- [15] Dede, T., Ayvaz, Y., & Bekiroglu, S. (2003). Optimization of Truss Structures using Value Encoding in a Genetic Algorithm, in B.H.V. Topping, (Editor), *Proceedings of the Seventh International Conference on the Application of Artificial Intelligence to Civil and Structural Engineering*, Civil-Comp Press, Stirlingshire, UK, Paper 38, 2003. doi:10.4203/ccp.78.38
- [16] Gere, J., Weaver, W. (1980). *Matrix Analysis of Framed Structures*, Van Nostrand Reinhold Company Inc., Wokingham, Berkshire.
- [17] Gundersen, G. (2002). *The Use of Java Sparse Arrays in Matrix Computation*, Master's Thesis, University of Bergen, Norway.
- [18] Beeby, A. W., & Narayanan, R. S. (2009). *Designers' guide to Eurocode 2: Design of Concrete Structures*, Thomas Telford Publishing.
- [19] European Committee for Standardization (2004). *Eurocode 2: Design of concrete structures - Part 1-1: General rules and rules for buildings*, Brussels, Belgium.
- [20] Mosley, B., Bungey, J., & Hulse, R. (2012), *Reinforced Concrete Design to Eurocode-2*, 7th Edition, Palgrave Macmillan Publishing, Houndmills, Hampshire.
- [21] European Committee for Standardization (2004). *Eurocode 2: Design of concrete structures - Part 1-2: General rules - Structural fire design*, Brussels, Belgium.
- [22] Computers and Structures Inc. (2019). ETABS – an integrated software package for the structural analysis and design of buildings, <https://www.csiamerica.com/products/etabs> Accessed 17 August, 2019.
- Bedilu Habte** has completed his B.Sc. and M.Sc. studies at Addis Ababa University in 1983 and 1989, respectively. He completed his PhD study in 2000, specializing in Informatics in Civil Engineering, from TU Darmstadt, Germany. For the next three years, he worked as a software developer and IT consultant for different companies in Germany. Since 2004 he is engaged in teaching and research activities as an Associate Professor at the School of Civil and Environmental Engineering of the AAU, in Ethiopia. As the chair of the Structural Mechanics division, he supervises MSc theses and conducts research offering computational solutions to civil engineering tasks. He has undertaken a research activity on E-learning at the Carnegie Mellon University, USA as a Fulbright scholar. He is a member of the Ethiopian Association of Civil Engineers and had also served in its editorial committee.
-  <https://orcid.org/0000-0002-8707-3514>
- Elias Yilma** is a structural engineering lecturer at Addis Ababa Institute of Technology. His research examines engineering, numerical and computational problems at the intersections of structural engineering and computational sciences. His research interests include applications of artificial intelligence in structural engineering problems and applied computational mechanics.
-  <https://orcid.org/0000-0001-5288-7334>

