International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 04 Issue: 07 | July -2017 www.irjet.net p-ISSN: 2395-0072

Cost Optimization Of Multistoried Rc Framed Structure Using Hybrid Genetic Algorithm

R.Regupathi[1]

[1]*Assistant Professor, Department of Civil Engineering, Government College of Engineering, Bodinayakanur, Tamil Nadu, India*

Abstract - *In general the optimization techniques enable designers to find the best design for the structure under consideration. This Project Cost Optimization of RC Beams and Columns using Hybrid Genetic Algorithm (GA) technique is presented. The structure is designed economically without impairing the functional purposes of the structural elements is supposed to serve and not violating provisions given in IS456-2000. Using the cross-sectional action effects as design variables, a Genetic Algorithm incorporated with a heuristic cost function is presented as an alternative to traditional cost functions for layout optimization of RC structures. Genetic Algorithm Program has been developed for the cost optimization of reinforced concrete beams and columns using MATLAB software. In order to validate the developed Hybrid GA for the proposed apartment building, conventional design of apartment building was analyzed using Staad Pro software and the results were compared.*

Key Words: Optimization, Genetic Algorithm, Design Variables, Conventional Design, Heuristic Cost

1. INTRODUCTION

Optimum design of structures has been the topic of many studies in the field of structural design. A designer's goal is to develop an "optimal solution" for the structural design under consideration. An optimal solution normally implies the most economic structure without impairing the functional purposes the structure is supposed to serve. There are some characteristics of RC make design optimization of these structures distinctly different from other structures. The cost structures which of RC structures is influenced by several cost items including the cost of concrete and reinforcement. Therefore, in case of RC structures, the minimum weight design is not necessarily the same as the minimum cost design. In fact, for RC structures the optimum cost design is a compromise between the consumption of concrete, reinforcement which minimizes the total cost of the structure and satisfies the design requirements. In the design optimization of RC structures the cross-sectional dimensions of elements and detailing of reinforcement, e.g. size and number of steel bars, need to be determined. Consequently, the number of design parameters that need to be optimized depends on cracking and durability requirements of RC structures. These requirements increases the number of design constraints of the optimization problem of RC structures.

The optimal design of three-dimensional reinforced concrete (RC) skeletal structures having members subjected to axial loads. The width, depth and area of longitudinal reinforcement of member sections are taken as the design variables. The optimality criteria (OC) method is applied to minimize the cost of the concrete, steel and formwork for the structure.

2. OPTIMIZATION TECHNIQUES

Optimization is a branch of mathematics which is concerned with obtaining the conditions that give the extreme value of function under given circumstances. An optimization problem can be mathematically stated as follows:

Find X = (x1, x2, . . ., an) which minimizes if (X) I =1, 2, . . ., no (2.1)

Subject to

g j (X) ≤ 0, j= 1, 2, . . ., ng(2.2)

He(X) = 0, k = 1, 2, . . ., ne (2.3)

xlm≤ xm≤ xum = 1, 2, . . ., ns (2.4)

Where is the vector of n design variables, if(X) is an objective or merit function, gj(X) and he(X) are the inequality and the equality constraints, respectively. These constraints represent limitations on the behavior or performance of the system. Therefore, they are called behavioral or functional constraints. Side constraints (2.4) restrict the acceptable range of potential solutions of the problem based on non-behavioral constraints. In this expression xlm, xm indium is the lower and upper limits on the design variable, respectively. In the above expressions no, nag, ne and ns are the number of objective functions, number of inequality, equality and side constraints, respectively. Depending on the specific choice of design variables, objective functions, and constraints, various types of optimization problems may exist.

2.1.Introduction to Genetic Algorithm

Genetic Algorithms were introduced by Holland in the 1960s. With the aid of his colleagues and students, he developed these Algorithms during the 1970s in University of Michigan. He summarized the results of these researches

in the book "Adaptation in natural and artificial systems" (Holland, 1975). Gases are numerical optimization techniques inspired by the natural evolution laws. A GA starts searching design space with a population of designs, which are initially created over the design space at random. In the basic GA, every individual of population (design) is described by a binary string (encoded form). GA uses four main operators, namely, selection, creation of the mating pool, crossover and mutation to direct the population of designs towards the optimum design. In the selection process, some designs of a population are selected by randomized methods for GA operations, for examples in creation of the mating pool; some good designs in the population are selected and copied to form a mating pool. The better (fitter) designs have a greater chance to be selected. Crossover allows the characteristics of the designs to be altered. In this process different digits of binary strings of each parent are transferred to their children (new designs produced by the crossover operation). Mutation is an occasional random change of the value of some randomly selected design variables. The mutation operation changes each bit of string from 0 to 1 or vice versa in a design's binary code depending on the mutation probability. Mutation can be considered as a factor preventing from premature convergence. A GA uses a discrete set of design variables in the optimization process. However, by defining the number of decimal digits for representation of continuous variables or step size between the sequential values of design variables this method can be applied to continuous problems as well.

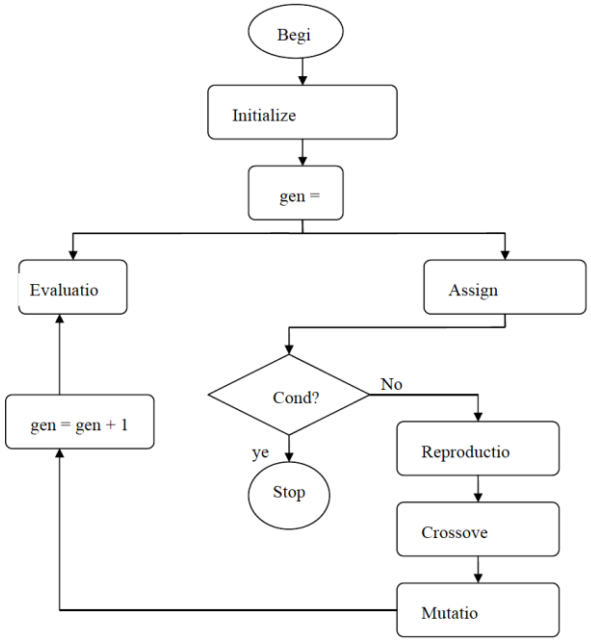

Fig.1 Flow Chart of Working Principle of GA

2.1 GA Procedure

GA is an iterative procedure that is motivated by the 'survival of the fittest' Principle of Darwinian Theory of natural evolution (Darwin, 1859). The flowchart of a simple GA is given in Figure 2. A GA begins with a randomly created population of solutions that are represented by a coded string of fixed length (chromosome). The solutions are decoded and evaluated according to a criterion, called the fitness function. Every solution is assigned a fitness value according to the obtained value of the fitness function for that solution. To produce a new generation of solutions (new population). Some solutions are first selected according to their fitness values to enter a mating pool. The mating pool is then filled up by cloning the individuals (solutions), which have been entered in the mating pool, in proportion to their fitness values (creation of the mating pool). Creation of a new population is implemented by repeating the crossover and mutation operations. In the crossover stage, two individuals are initially selected as parents and then some segments of encoded strings of parents are swapped to create two children. For the mutation, some children are selected randomly, and then some alleles of these children are altered at random. The fitness value of individuals in the new generation is then evaluated. If the termination conditions are satisfied, the process is terminated. Otherwise, the iterative process is repeated for a new generation.

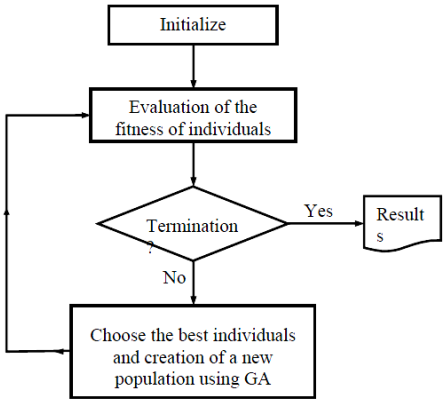

Fig.2 Flow Chart of a GA

2.2 Representation of GA

In GAs, design variables are usually encoded and represented in a string form. Two popular systems of coding are binary and real coding schemes. In the real number representation, each variable is represented as a conventional floating-point number. Binary and real coding 'differed mainly in the way of implementation of GA processes. In this project the basic binary coding method is used.

2.3 Components of GA

A typical genetic algorithm includes four processes: selection, creation of the mating pool, crossover and mutation. Depending on the problem and the selected representation scheme some of these processes can be applied. In the present research only the four main processes, i.e. Selection, creation of the mating pool, crossover and mutation have been used.

2.4 Method of Selection

In different stages of a GA some individuals are selected by a randomized technique for performing certain GA processes on them. The method of selection of individuals has a significant influence on driving the search towards a promising area and finding a good solution within a short time. However, decreasing the diversity of the population may cause premature convergence to a local optimum. Many selection methods have been discussed in the literature, among them roulette wheel (also referred to as fitness-proportional selection), Boltzmann selection, tournament selection, ranking selection (linear and non-linear) and steady state selection (Goldberg and Deb, 1991, Mitchell, 1998, Coley, 1999). In this project, roulette wheel selection method has been implemented. In the fitness-proportional selection, the selection probability for each individualism proportional to its fitness value. The process of randomized selection of individuals can be simulated using a roulette wheel in which the width of each slot corresponds tithe fitness of a specific individual of the population. The wheel is then spun and the individual that come under the marker of the wheel is selected. Whitley (1989) pointed out two weaknesses of the fitness-proportional selection, which are stagnation and premature convergence of the search. When the relative difference between fitness value of the individuals is small the search process stagnates. On the other hand, when the relative difference between the fitness values of the individuals is large, the fittest individuals dominate the creation of the next generation. Consequently, the search prematurely converges to a solution.

2.5 Creation of the mating pool

Creation of the mating pool is a process in which some individuals in a population are selected according to their fitness values as parents to form a mating pool. In this process, the fitter individuals have a greater chance to be selected to enter the matting pool. All individuals in a current population may initially be considered to enter the matting pool. However, a criterion may be determined for the acceptance or rejection of certain individuals for reproduction. The average fitness of the current population cane considered as an acceptance limit for individuals to enter the mating pool. Alternatively, it is possible to select a percentage of the existing population in the order of decreasing magnitude of their fitness values that can enter the mating pool (percentage of survivors). The second strategy for selection of individuals is implemented in the GA developed in this project.

2.6 Genetic Algorithm Operators

The Population Size defines the number of initial (feasible and infeasible) solutions which are created at the beginning of the GA model work. These initial solutions are created taking into account the variables' bounds which are stated in theca model. Limiting the population size to very little initial solutions will prevent the development of much better solutions since the interaction between the initial solutions are limited to a narrow range. Meanwhile increasing the population size enables the genetic algorithm to search more points and thereby obtain a better result, however, the larger the population size, the longer the genetic algorithm takes to compute each generation. So one can experiment with different settings for population size that return good results without taking a prohibitive amount of time to run.

Crossover enables the algorithm to extract the best genes from different individuals and recombine them into potentially superior children. Three types of crossover are found in the literature of the GA, two of them are considered in this study, namely: single point crossover and two point crossover. Mutation adds to the diversity of a population and thereby increases the likelihood that the algorithm will generate individuals with better fitness values. Different values of the mutation rate are considered in the models in order to examine the best mutation rate that will lead to the optimum solution.

In this stage a percent of the total population size is chosen in order to perform the crossover and mutation on it. In our models, this percent is chosen randomly and varies between (0.3 and 0.8), these different values are examined in order to get the best solution.

As GAs is an unconstrained optimization technique; it is necessary to transform the constrained design optimization problem to an unconstrained one. Several methods for handling constraints by GAs have been proposed Michalewics, (1995). Among them the rejecting strategy and methods based on a penalty approach can be named. In the rejecting strategy, any design that violates one or more constraints is not accepted for the involvement in the GA process to create a new population. In a penalty method, a Constrained Optimization problem is converted to an unconstrained problem by adding a penalty for each constraint violation to the objective function, the simple penalty function is a famous function of the penalty method. It converts the constrained nonlinear programming "NLP" problem to an unconstrained minimization problem by penalizing infeasible solutions:

$$P(x,R,r) = f(x) + \sum_{j=1}^{J} R_j \langle g_j(x) \rangle^2 + \sum_{k=1}^{K} r_k [h_k(x)]^2$$

The parameters j R and k r are the penalty parameters for inequality and equality constraints, respectively. The success of this simple approach lays in the proper choice of these penalty parameters. One thumb rule of choosing the penalty parameters is that they must be so set that all penalty terms are of comparable values with themselves and with the objective function values. This is intuitive because if the penalty corresponding to a particular constraint is very large compared to that of other constraints, the search algorithm emphasizes solutions that do not violate the former constraint. This way other constraints get neglected and search process gets restricted in a particular way. In most cases, most search methods prematurely converge to a suboptimal feasible or infeasible solution. Since a proper

choice of penalty parameters are the key aspect of the working of such a scheme, most researchers experiment with different values of penalty parameter values and find a set of reasonable values. Deb, (1998) proposed a technique for handling constraint optimization problems for Genetic Algorithms. The proposed method belongs to both second and third categories of constraint handling methods described by Michalewicz and Schopenhauer (1996). Although a penalty term is added to the objective function to penalize infeasible solutions, the method differs from the way the penalty term is defined in conventional methods. He devised the following fitness function, where infeasible solutions are compared based on only their constraint violation.

$$F(x) = \begin{cases} f(x), & \text{if } g_j(x) \geq 0, \quad \forall j = 1,2,\ldots\ldots,m, \\ f_{\max} + \sum_{j=1}^{m} \langle g_j(x) \rangle, & \text{otherwise} \end{cases}$$

The parameter max f is the objective function value of the worst feasible solution in the population. Thus, the fitness of an infeasible solution not only depends on the amount of constraint violation, but also on the population of solutions at hand. However, the fitness of a feasible solution is always fixed and is equal to its objective function value. The two penalty functions discussed above are used in this research.

The fitness function is a criterion for the evaluation of the goodness of each individual in a population. Since the fitness function is a figure of merit, and some of the selection methods require a positive fitness value, it must therefore take positive values. In typical cost optimization problems, minimization of some cost function is carried out rather than maximization of utility or profit.

3. OPTIMIZATION OF APARTMENT BUILDING USING HYBRID GENETIC ALGORITHM

3.1 Optimization of RC Beams

The chief task of the structural engineer is the design of structures. Design is the determination of the general shape and all specific dimensions of a particular structure so that it will perform the function for which it is created and will safely withstand the influences that will act on it through its useful life. These influences are primarily the loads and other forces to which it will be subjected. Beams are structural members carrying transverse loads that can cause bending moments, shear forces and in some cases torsion. Design of a beam starts with proportioning its sections to resist bending moment and choosing the required reinforcement. Once this is done, the chosen sections are checked and designed for shear and torsion. In order to limit deflections, the depth of the cross section is chosen to fulfill the IS456-2000 Code serviceability requirements.

3.1.1 Design Parameters

The two main parameters in optimizing the apartment building is Objective Function and Constraint Function.

3.1.2 Rectangular Beam Objective Function

The objective function for the simply supported reinforced concrete beam model was run to optimize the design of a rectangular cross section for this beam and loadings while satisfying the provisions of the IS456-2000 Code.

Function= (Cc*((bD + d') – Ast)*L) +(Cs*Ast*L*ṗ) +(Cf*(((2*D+b)*L));

3.1.3 Flanged Beam Objective Function

The objective function for the simply supported reinforced concrete beam model was run to optimize the design of a Flanged Beam cross section for this beam and loadings while satisfying the provisions of the IS456-2000 Code.

Function= (Cc*((bD + d') – Ast)*L) +(Cs*Ast*L*ṗ) +(Cf*(((2*D+b)*L));

3.1.4 Doubly Reinforced Beam Objective Function

Function=(Cc*(b*(d+35))-(Asc+(Ast))*L)+(Cs*(Asc) +(Ast)*L*ṗ*)+ (Cf**(2*(d)+ d')+x(1))*L);

3.1.5 Beam Optimization Example Problem

A rectangular RC simply supported beam carrying L.L of 12kN/m and D.L 6kN/m. Effective span of beam is 6m.Assume M20 & Fe415 combination. Assume thickness of masonry wall 230mm.

Table 1 Iteration Values of Beam Optimization

Iteration	F(x)
0	2.272500e+003
1	9.952500e+003
2	1.214794e+004
3	1.204933e+004
4	1.136024e+004
5	1.089663e+004
6	9.624273e+003
7	9.150973e+003
8	9.078070e+003
9	9.077771e+003
10	9.076790e+003
11	9.073803e+003
12	9.073061e+003
13	9.073062e+003
14	9.073062e+003

Here we used hybrid genetic algorithm concepts, with heuristic approach to find the constraint and non-linear optimization and the optimized results satisfied all our constraints as per IS456-2000. The conventional designed results and optimized results were checked with limit state of collapse and serviceability criteria.

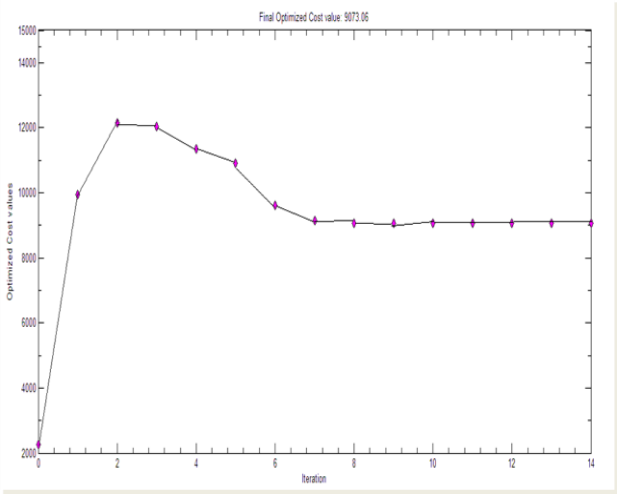

Fig. 3 Representation of Fitness Function for Beam

This above graph shows the optimized value for the beam, and this graph is plotted for the fitness function iteration values.

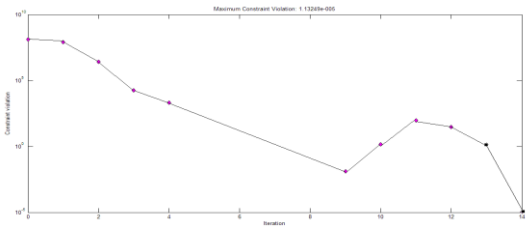

Fig. 4. Representation of Constraint Function for Beam

In Iteration process ,at first there will be a huge violation in satisfying the constraints, on further process of iteration the constraints are satisfied and at one point it reach the optimality characteristics.

3.1.6 Tuning of GA operators

Various values for each GA operator were tried bearing in mind the recommended Values of these operators. The operator was chosen and then the optimization models (the RC models) were run several times. Then, the best value for each operator is chosen (the best value is that which corresponds to the least cost among other values). The models were run again to check various values of another operator taking into account the best value of the preceding tuned operator.

3.1.7 Number of Iterations

The number of iterations was studied by keeping the values of other parameters fixed as follows
The Population size = 100, the Mutation rate = 0.15, the Percent of population not selected for mating = 0.5. It is clear that by using a very little number of iterations (< 50), the optimizer kept the value of the cost far away from the optimum (minimum). Therefore, the obtained solution is not the best possible one. This means that using a very little number of iteration may allow little application of the basic operators of the GA on the population. Therefore, the model will not be able to explore wide range of developed populations (solutions) and the model may get stuck to a suboptimal or infeasible solution. It can be said that as the number of iteration increases, the solution will enhance, this is true until the number of iteration reaches the value of 200 for both the single point and the two point crossover types, these results are the same when using both the simple and Deb penalty functions, beyond this value, any increase in the number of iteration will not significantly alter the best solution. The number of iterations of 200 is a moderate value that could result in the optimum solution in an acceptable time of run of the GA model for both the RC and PC models. It can also be seen that the type of crossover and the type of the penalty function has a small effect on the results of the GA model either on the trend of the graph or on the minimum cost obtained.

3.1.8 Population Size

The population size is studied by fixing the values of the other parameters
The No of iteration = 200, the Mutation rate = 0.15, the Percent of population not selected for mating = 0.5, The results of tuning the population size parameter, it is clear that by using a lower number for the population size (<50), this will keep the value of the cost away from the optimum, thus the obtained solution is not the best possible, so limiting the population size to very little initial solutions will prevent the development of much better solutions. Meanwhile increasing the population size enables the genetic algorithm to search more points and thereby obtain a better result. The population size increases, the solution enhances, this is true until the population size reaches the value of 50 for both the single point crossover case and the two point crossover case, after that any increase in the population size will not improve the best solution significantly. While the solution reaches its best value at a population size of 100 for the case of two point crossover, it reaches its best value at a population size of 200 for the case of single point crossover. Figures shows that while the solution reaches its best value at a population size of 100 for the case of single point crossover, it reaches its best value at a population size of 200 for the case of two point crossover. It can be said that the population size of 100 or 200 is a moderate value that could result in the optimum solution in an acceptable time of run of the GA model.