Optimization of Space Structures with Fuzzy Constraints Via Real Coded Genetic Algorithm(RCGA)

A. Behravesh[a], M.M. Ebadi[*], V. Rashtchi[b]

Abstract— **Most conventional optimization methods contain the problems with constraints having crisp numbers. But in most practical engineering problems, the constraints contain allowable non crisp numbers that make using fuzzy optimum design inevitable. In this article the combination of α-cut fuzzy method with evolutionary algorithms such as Genetic Algorithm (GA) is investigated. Also some schemes to increase the speed of Real Coded Genetic Algorithm (RCGA) are used. The performance characteristics of the above methods are investigated by two space structures. The examples show that using fuzzy programming increase the efficiency of RCGA.**

***Keywords*—fuzzy programming, membership degree, parametric nonlinear programming, Genetic Algorithm.**

1 INTRODUCTION

The problems in the real world often have complex construction due to the imprecision and uncertainty inherent in defining and perceiving them. It is possible to accurately model the systems which have less complexity with mathematical equations. If there is enough data from the systems that their complexity is a little more, we will be able to solve them with free models such as neural network. However, in order to model the systems that have high complexity, less data, ambiguous and imprecise information, Fuzzy set theory is the best way.

Fuzzy set theory was introduced by Zadeh in 1965. Since then it has been developed and applied in different branches of sciences [1]. In crisp or Conventional set theory the value of each sentence can be true or false but in fuzzy set theory the value of each sentence can be a number between zero and one. For example, in structural design problems the stress (σ) in the members must be limited by allowable values. In classical set theory it means that $\sigma = \sigma^U$ is acceptable but $\sigma = \sigma^U + \Delta\sigma$ $(\Delta\sigma > 0)$ even for very low values of $\Delta\sigma$ is unacceptable, but in fuzzy set theory we can accept it with a membership degree [7]. The fuzzy set based optimization and the concepts of fuzzy constraints, fuzzy objective and fuzzy decision were introduced by Bellmann and Zadeh [1]. Since in most structural design problems, the input data and parameters are fuzzy and the behavior of structure is non linear, it is necessary to develop fuzzy mathematical programming for optimum design [3].

After modeling imprecision in formulation of structural optimization problem, Genetic Algorithm (GA) can be used for solving the problem [4]. GA is a search and optimization method based on principles of genetics and natural selection. Genetic algorithm was introduced by John Holland in 1975 and developed by one of his students, Goldberg (1989). The advantage of this method to other methods is its ability to find the global minimum or maximum with continuous or discrete variables without using the derivatives of cost function [5]. Genetic algorithm is used directly only for solving unconstrained optimization problems, so for solving constrained problems we should transform them to unconstrained problems by penalty function method [6].

For the first time Wang and Wang [7] applied α–cut method for solving fuzzy structural optimization problem. In this method, there is an optimal solution for each value of α. This is the base strategy for solving structural optimization problems in which the allowable stresses and displacements are in fuzzy form.

Rao, Sundaraju and Prakash [8], have used λ formulation for solving fuzzy structural optimization problem. Contrary to α–cut method, λ formulation gives a unit optimum answer and is often used in multi objective optimization problems.

Sarma and Adeli [4], combined fuzzy λ formulation with binary genetic algorithm for solving fuzzy optimization problem.

Soh and Yang [2], have used fuzzy logic for handling the GA operators in size and shape optimization of structures.

Shih, Chi and Hsiao [3], have used multiple α cut method (single - double and multiple cuts methods) for solving fuzzy structural optimization problems with fuzzy constraints.

In this article α–cut method is used to transform fuzzy optimization problem to crisp parametric programming problem, then an optimum RCGA is used for solving the obtained problem. Two space structures with continuous

Manuscript received March 3, 2009.
a- A. Behravesh
Full Professor, Department of Civil Engineering, Islamic Azad University Mahabad branch, daneshgah Street, Mahabad, Iran, P. O. Box: 59135-443.
∗Corresponding author: M.M. Ebadi, Postgraduate Student, Department of Civil Engineering, Islamic Azad University Mahabad Branch, Mahabad, Iran and Member of YRC (Young Researchers Club of Islamic Azad University), No 952, 4th Street, 3rd phase, Mahdavy Avenue, Shahrak Karemandan, Zanjan, Iran, Postal Code:4513914595. Tell: +98 (241) 4240444, Cell Phone: +98 (912) 3411592
E-mail: mohammadmehdi.ebadi@gmail.com
b- V. Rashtchi
Assistant Professor, Department of Electrical Engineering, University of Zanjan, Daneshgah Road (Tabriz road km 5), Zanjan, Iran, P. O. Box:313.
E-mail: vrashtchi @ yahoo.com.

sizing variables and fixed topology are presented to demonstrate the robustness of the method. The programs for analyzing the structures and RCGA are developed in (MATLAB 6.5) software.

2 FUZZY CONCEPT

Assume X is a set of elements and A is a fuzzy subset of X then the fuzzy set A can be defined as a set of ordered pairs, each with the first element from **X** and the second element from the interval **[0,1]** [9].

$$\tilde{A}=\{(x,\mu_{\tilde{A}}(x)) \mid x\in X\} \quad (1)$$

$\mu_{\tilde{A}}(x)$ is the membership function or membership degree of element **x** in fuzzy set $\tilde{A}$, the membership degree maps the elements of **X** to the interval **M= [0,1].**

$$\mu_{\tilde{A}}(x) : X \rightarrow [0,1] \quad (2)$$

3 NONLINEAR PROGRAMMING WITH FUZZY CONSTRAINTS

The basic concepts and procedure of conventional linear programming with fuzzy constraints (FLP) can be applied to nonlinear programming problems with fuzzy inequality constraints (FNLP) [3].

The general model of nonlinear programming with fuzzy resources can be formulated as [3]:

$$\begin{aligned} &\text{find} \quad X = [x_1, x_2, ..., x_n]^T \\ &\min \ f(X) \\ &\text{S.t} \quad g_i(X) \le \tilde{b}_i \qquad i=1,2,\ldots,m \\ &X^L \le X \le X^U \end{aligned} \quad (3)$$

In (3) **f(x)** and $g_i(x)$ are fitness function and ith inequality constraints, respectively. The fuzzy number $\tilde{b}_i \forall i$ is located in the fuzzy region of $[b_i , b_i+P_i]$ with the fuzzy tolerance P_i.

4 α- CUT METHOD

In the Verdagay's method for fuzzy mathematical programming, the trapezoidal membership function is considered for fuzzy constraints. Fig.1 shows the trapezoidal membership function and (4) is its equation [3]:

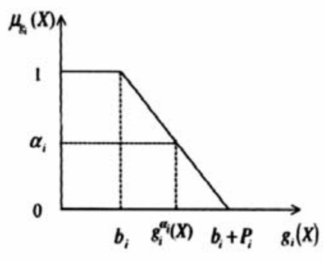


Fig. 1. Trapezoidal membership function for $\mu_{g_i}(x)$

$$\mu_{g_i}(X) = \begin{cases} 1 & \text{if} \ g_i(X) < b_i \\ 1-\dfrac{g_i(X)-b_i}{p_i} & \text{if} \ b_i \le g_i(X) \le b_i + p_i \\ 0 & \text{if} \ g_i(X) > b_i + p_i \end{cases} \quad (4)$$

In fact (4) represents the degree of satisfaction of the ith constraints with the vector **X**. Also it shows that by increasing the resources i.e., by going to the base of trapezoidal, the degree of satisfaction of constraint decreases. (4) is equivalent to the following formulation [3]:

$$\begin{aligned} &\min \ f(X) \\ &\text{S.t} \\ &X \in X_\alpha \end{aligned} \quad (5)$$

Where $\alpha\in[0,1]$ and $X_\alpha = \{x_i \mid \mu_{g_i}(X) \ge \alpha \ \forall_i , X \ge 0\}$ by substituting (4) into (5) we will obtain the following parametric nonlinear optimization problem.

$$\begin{aligned} &\min \ f(X) \\ &\text{S.t} \\ &g_i(X) \le b_i + (1-\alpha)P_i \\ &X^L \le X \le X^U \end{aligned} \quad (6)$$

5 SOLVING PARAMETRIC NONLINEAR PROGRAMMING PROBLEM

The obtained parametric programming problem (6), can be solved for different values of α by reliable methods. In this article we have used an optimum RCGA for solving the crisp problem.

6 GENETIC ALGORITHM (GA)

GA is one of the methods which may be used to solve an optimization problem. This algorithm is based on natural selection using random numbers and does not require a good initial estimate [10].

6.1 Binary Genetic Algorithm

GA in the binary form works with binary string. Each string which is called chromosome is the member of population and the GA's operators that are inspired from the natural selection guide the population to the evolution or in other words, maximize the fitness function. A simple genetic algorithm consists of three operators [11]-[12]:

1- Reproduction
2- Crossover
3- Mutation

Fig. 2 indicates the different steps of the algorithm with three operators. The process of each operator is detailed bellow:

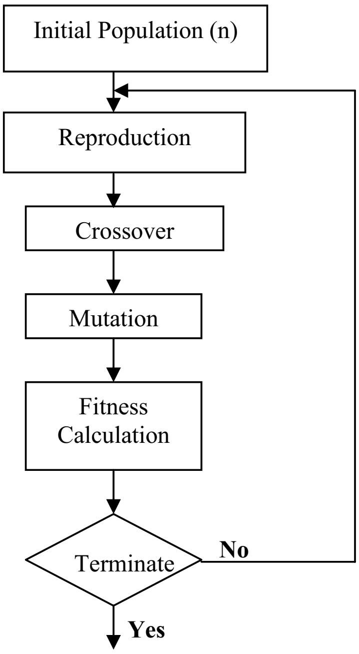


Fig. 2. simple flowchart of a GA

Reproduction

The reproduction operator copies each chromosome proportional to its fitness function in the mating pool so the chromosome with the best fitness function will be copied more than the rest in the new population [11]-[12].

Crossover

The crossover operator works on two chromosomes and produces two new offspring that will inherit some characteristics of their parents. In this process two chromosomes in the mating pool are selected in pairs with (p_c) probability, then another random number determines the crossover point on the chromosome. Finally all bits of these two strings are exchanged between each other [11]-[12]. Fig. 3 shows the process of this operator. Crossover point can be selected more than once on the chromosome; the process is called multi point crossover [11]-[12].

chromosome $\mathbf{A_1}$ = 0 1 1 0 | 1
chromosome $\mathbf{A_2}$ = 1 1 0 0 | 0
Crossover
chromosome A'_1 = 0 1 1 0 0
chromosome A'_2 = 1 1 0 0 1

Fig. 3. Crossover operator

Mutation

The mutation operator causes random changes in the people of population. In the process the chromosomes with probability of (p_m) are selected for mutation from the population, then a random number determines the position of mutation point and the bit in this place is complemented. The process is shown in Fig. 4 [11]-[12]:

chromosome **A1** = 1 0 0 1 ***1*** 0 0 1
mutation
chromosome A'_1 = 1 0 0 1 ***0*** 0 0 1

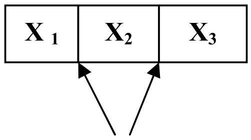
Fig. 4. Mutation operator

6.2 Real Coded Genetic Algorithm (RCGA)

Since in structural optimization problems with continuous variables we need to work on real numbers we have used RCGA. In RCGA contrary to the binary method we do not need to decode the variables also, less processing memory is used [5]. In this method each chromosome is defined as an array of real numbers with the mutation and crossover operators working as shown in Fig. 5 [13]:

The mutation can change the value of a real number randomly and the crossover can take place at the boundary of two real numbers [13].

$$\overline{\mathbf{X}} = (\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3})$$



Possible crossover position

$$x_i \xrightarrow{\textbf{Mutation}} x_i = x_i + \text{rand}\left(-\frac{X_{imax}}{2}, \frac{X_{imax}}{2}\right)$$

X_{imax} is the maximum possible value for Xi

Fig. 5. mutation and crossover operators in RCGA

7 Transforming Constrained Optimization to Unconstrained

Since GA is used only for solving unconstrained optimization problems, it is necessary to transform the constrained problem to the unconstrained optimization problem. Here we have used a quadratic penalty function as shown in (7) [6]:

$$\min \varphi(A) = \frac{1}{Lf}\sum_{1}^{N}\rho_i l_i A_i + \beta\{\sum_{1}^{N}[(\frac{|\sigma_i|}{|\sigma_i^a|} - 1)^+]^2 + \sum_{1}^{M}[(\frac{|\delta_i|}{|\delta_i^a|} - 1)^+]^2\} \quad (7)$$

$$(\frac{|\sigma_i|}{|\sigma_i^a|} - 1)^+ = \max(\frac{|\sigma_i|}{|\sigma_i^a|} - 1, 0)$$

$$(\frac{|\delta_i|}{|\delta_i^a|} - 1)^+ = \max(\frac{|\delta_i|}{|\delta_i^a|} - 1, 0)$$

$$\sigma_i^a = \sigma_i^L \quad \text{when} \quad \sigma_i < 0$$
$$\sigma_i^a = \sigma_i^U \quad \text{when} \quad \sigma_i \geq 0$$
$$\delta_i^a = \delta_i^L \quad \text{when} \quad \delta_i < 0$$
$$\delta_i^a = \delta_i^U \quad \text{when} \quad \delta_i \geq 0$$

where the last term is the penalty function, L_f is the normalizing factor, β is the penalty coefficient, M is the number of degrees of freedom, σ_i is stress in member i, δ_i is the displacement in the direction of degree of freedom i and σ_i^a, δ_i^a are allowable stresses and displacements, respectively.

8 METHODS FOR INCREASING THE SPEED OF RCGA (OPTIMUM RCGA)

Typically in structural optimization problems with GAs, a population (pop) of many individuals with a high crossover rate (p_c) and very low mutation rate (p_m) is used [6]-[14]-[15]. Following the typical condition, RCGA with pop = 40, p_c = 0.7, p_m = 0.01 was tested for two sample space structures:

8.1 Sample structures:

The samples contain 25 and 72-bar space truss under two load conditions [16]-[17]. Displacement method is used for analyzing the structure. The value of β (penalty coefficient) is increased by constant 10 in each 10 iteration and a value of 1000 is used for normalizing factor L_f.

8.1.1 The 25 bar space truss:

The 25 bar transmission tower that is shown in Fig. 6 has been optimized by different researchers [16]-[17]. In this example the material density is 2779.48 kg/m^3 and the modulus of elasticity is 68900 MPa. The structure is subjected to two loading conditions that are shown in Table 1. There are 25 members that are divided into 8 groups:

(1) A_1 , (2) A_2-A_5 , (3) A_6-A_9 , (4) A_{10}-A_{11} , (5) A_{12}-A_{13} , (6) A_{14}-A_{17} , (7) A_{18}-A_{21} , (8) A_{22}-A_{25}.

Table 1

Loading conditions for 25 bar space truss

Node	Case1(kN)			Case2(kN)		
	Px	Py	Pz	Px	Py	Pz
1	0.0	89.0	-22.25	4.45	44.5	-22.25
2	0.0	-89.0	-22.25	0.0	44.5	-22.25
3	0.0	0.0	0.0	2.225	0.0	0.0
4	0.0	0.0	0.0	2.225	0.0	0.0

The stress limits of the members are listed in Table 2. All nodes in all directions are subjected to the displacement limits of ±0.00889m. The lower bound of cross-sectional areas is given as 0.00000645m^2.

Table 2
Member stress limits for 25 bar space truss

Variables	Compressive stress limits (MPa)	Tensile stress limits (MPa)
1	241.78	275.6
2	79.855	275.6
3	119.23	275.6
4	241.78	275.6
5	241.78	275.6
6	46.569	275.6
7	47.947	275.6
8	76.354	275.6

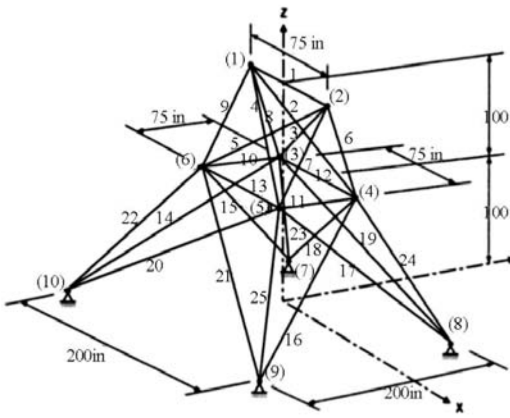


Fig. 6. 25 bar space truss structure (1in=0.0254m)

8.1.2 The 72 bar space truss:

The 72 bar spatial truss which is shown in Fig. 7 has been studied by different researchers [16]-[17]. In this example the material density is 2779.48 kg/m^3 and the modulus of elasticity is 68900 MPa. The structure is subjected to two loading conditions, in the first loading condition, node 17 is subject to 22.25 kN, 22.25 kN, -22.25 kN along the x, y and z directions, respectively. In the second loading condition, nodes 17, 18, 19 and 20 are subjected to -22.25 kN along the z axis. allowable stress for both tension and compression are 172.25 MPa. The upper most nodes are subjected to the displacement limits of ±0.00635 m. The minimum permitted cross-sectional area of each member is 0.00000645m^2. The structure has 72 members that are divided into 16 groups which are following:

(1) A_1-A_4, (2) A_5-A_{12}, (3) A_{13}-A_{16}, (4) A_{17}-A_{18}, (5) A_{19}-A_{22}, (6) A_{23}-A_{30}, (7) A_{31}-A_{34}, (8) A_{35}-A_{36}, (9) A_{37}-A_{40}, (10) A_{41}-A_{48} , (11) A_{49}-A_{52}, (12) A_{53}-A_{54}, (13) A_{55}-A_{58}, (14) A_{59}-A_{66}, (15) A_{67}-A_{70}, (16) A_{71}-A_{72}.

From the results the speed of convergence was not desirable. For achieving more speeds, a variety of sets of conditions with 40<pop<100, 0.6<p_c<0.9 and 0.001<p_m<0.1 were used.

Table 3

Optimal design for different values of α in 25 bar space truss

α	1	0.8	0.6	0.4	0.2	0.0
$A_1(cm^2)$	0.7845	0.2548	1.1374	0.2787	0.0922	0.1012
$A_2(cm^2)$	13.888	15.938	14.456	10.507	8.8295	9.6432
$A_3(cm^2)$	18.047	16.819	20.630	17.128	17.779	14.022
$A_4(cm^2)$	0.0980	0.0703	1.8626	0.1103	0.0909	0.1567
$A_5(cm^2)$	0.1393	0.0806	0.3238	0.4509	0.0729	0.0651
$A_6(cm^2)$	5.1061	6.4384	6.5287	4.3525	3.6711	3.8350
$A_7(cm^2)$	10.527	10.637	9.1799	9.1328	9.2179	8.9973
$A_8(cm^2)$	16.807	17.459	17.077	13.744	13.025	12.832
W(N)	2438.64	2305.52	2188.11	2070.21	1967.70	1876.96

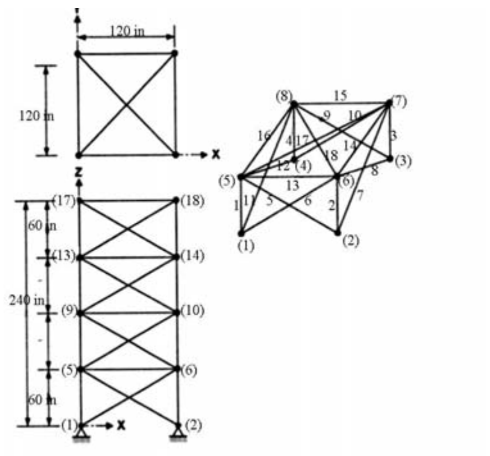


Fig. 7. 72 bar space truss structure (1in=0.0254m)

Although in all these cases convergence occurred and the parameters were obtained with enough accuracy, we did not have desirable increase in speed of the algorithm and the number of iteration remained around 9000 times. For increasing the speed of convergence in the algorithm following observations were made [13]:

1- Increasing p_m .

2- Decreasing p_c (An increase in pc causes the uniformity of the population, and the algorithm loses its efficiency).

3- Decreasing the population (Although the increase in pop decrease the number of iterations, the computation time per iteration increases and this altogether decreases the speed of the algorithm).

Regarding these facts the following conditions are used:

pop=6
p_c=0.25
p_m=0.9

After each iteration, the weakest individual in the new generation is replaced by the strongest in the old generation [13].

Table 4

Optimal design for different values of α in 72 bar space truss

α	1	0.8	0.6	0.4	0.2	0.0
$A_1(cm^2)$	14.039	11.676	9.9928	10.724	12.262	11.893
$A_2(cm^2)$	3.0150	2.8569	2.5053	2.7756	2.8640	2.0588
$A_3(cm^2)$	0.0812	0.1348	0.1606	0.0696	0.0832	0.1264
$A_4(cm^2)$	0.0690	0.0748	0.0825	0.1600	0.1193	0.1406
$A_5(cm^2)$	7.7985	7.5604	7.0423	8.6908	8.4076	6.3932
$A_6(cm^2)$	3.4402	3.5634	3.0388	2.4485	3.0679	2.8917
$A_7(cm^2)$	0.0832	0.0941	0.1948	0.1600	0.0690	0.3826
$A_8(cm^2)$	0.0909	0.0954	0.3709	0.1258	0.0935	0.0832
$A_9(cm^2)$	4.3189	3.5363	3.3834	3.6408	2.2988	2.2304
$A_{10}(cm^2)$	3.6660	2.3562	2.6653	2.3498	2.7666	2.3169
$A_{11}(cm^2)$	0.1922	0.2258	0.2296	0.0787	0.1458	0.2780
$A_{12}(cm^2)$	0.8206	0.5671	1.0561	1.1058	0.4922	0.3664
$A_{13}(cm^2)$	1.2342	1.0761	0.8716	0.8510	0.9219	0.9297
$A_{14}(cm^2)$	3.0563	4.3499	4.0647	3.1550	2.0775	2.9498
$A_{15}(cm^2)$	3.2524	3.0298	2.4543	2.5730	1.8233	1.9923
$A_{16}(cm^2)$	3.5756	4.0247	4.2815	3.8363	3.1343	3.4821
W(N)	1655.07	1593.35	1497.24	1407.74	1353.34	1298.72

ISBN:978-988-18210-1-0

After performing the above modifications, the number of iteration is reduced from 9000 to 1800 and the number of structural analysis is reduced from 360000 to 10800 times. In fact the speed of convergence is increased about 33 times. Since the last step has an important effect in speed up, they are explained in the following paragraph:

8.1.3 Placing the strongest of old generation

Since the process is highly randomized, to preserve the characteristics of the old generation, or in other words, to prevent the extinction of the old generation, after each iteration and the execution of the crossover and mutation operators the strongest individual from the previous population replaces the weakest one in new generation [13].

Now it is possible to use the obtained optimum RCGA for solving the (6) with different values of α in the two sample structures. The trapezoidal membership function with the tolerance of (Pi=0.3bi)[4] is used for stress and displacement constraints in sample structures. Tables 3 and 4 show the optimal solution of the algorithm for different values of α. Figs. 8 and 10 provide the convergence history for different values of α in 25 and 72 bar samples, respectively. As it can be understood by Figs. 8 and 10 (Fig. 11 is the magnified form of Fig. 10 which shows the obtained results clearly), reducing the values of α, decreases the weight of structures. Also we have drawn the weight of structure curve for different values of α, in Figs. 9 and 12.

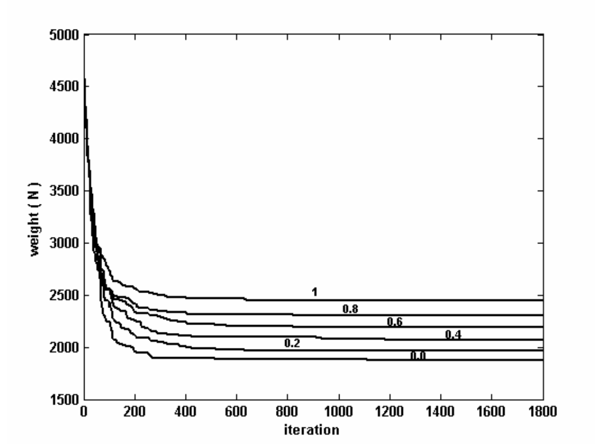


Fig. 8. Convergence history for different values of α in 25 bar space truss

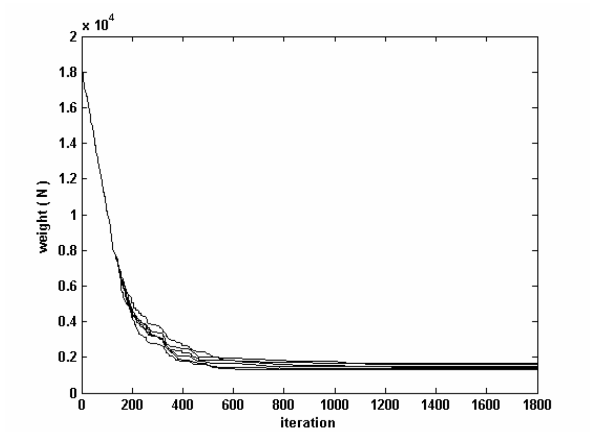


Fig. 10. Convergence history for different values of α in 72 bar space truss

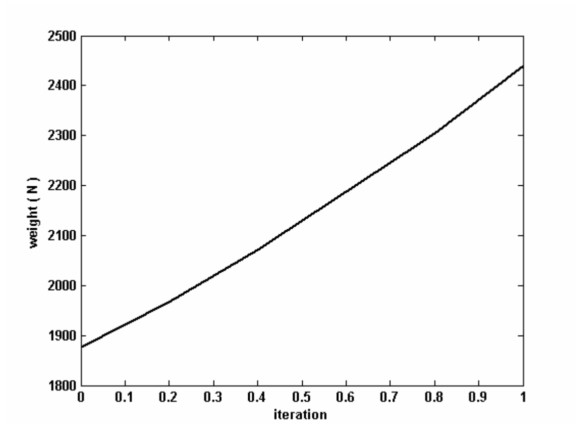


Fig. 9. Variations of weight with different values of α in 25 bar space truss

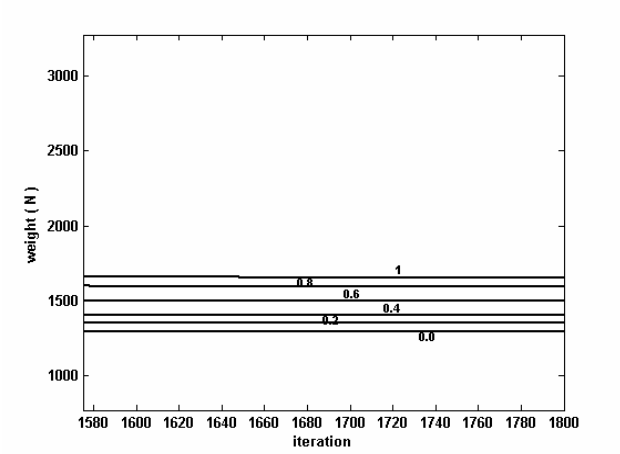


Fig.11. magnified form of Fig.10

Table 5

Variation of weight with iterations for α=1 in 25 bar space truss

iters	0	200	400	600	800	1000	1200	1400	1600	1800
W(N)	4812.2	2554.3	2475.5	2462.2	2448.8	2445.3	2442.6	2442.2	2442.2	2438.64

ISBN:978-988-18210-1-0

Table 6

Variation of weight with iterations for α=1 in 72 bar space truss

iters	0	200	400	600	800	1000	1200	1400	1600	1800
W(N)	18330	4596	2490	1971	1817	1768	1661	1660	1658	1655

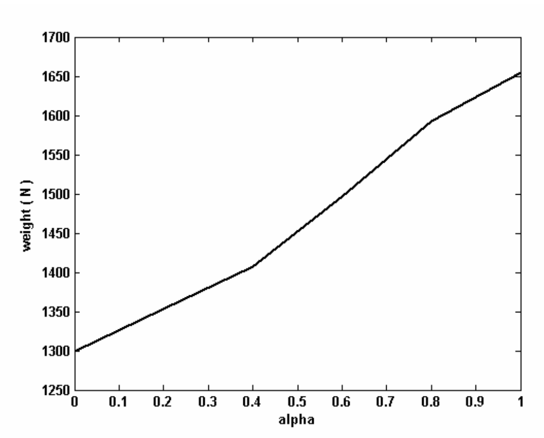


Fig. 12. Variations of weight with different values of α in 72 bar space truss

The variation of weight with iterations for α=1 in 25 and 72 bar space trusses, are shown in Tables 5 and 6, Respectively.

9 CONCLUSION:

In this article fuzzy Real Coded Genetic Algorithm was used for optimizing the structures. Also, some points were used to increase the speed of RCGA. The method was examined for two sample space structures (25 and 72 bar space structure). It is concluded that by integrating fuzzy programming with RCGA we can increase the efficiency of GA and the results will be better than those obtained when the constraints are in the crisp form (α=1). Also it is observed in all examples, that by reducing the values of α, the weight of structure will decrease. If we draw the weight of structure curve for different values of α, it gives a fuzzy solution for the optimization problems in which the allowable values are the functions of α parameters, and can be developed for different structures.

References:

[1] Y. Xiong, SS. Rao, "Fuzzy nonlinear programming for mixed-discrete design optimization through hybrid genetic algorithm". Fuzzy sets and systems., Vol. 146, 2004, pp. 167-186.
[2] CK. Soh, J. Yang, "Fuzzy controlled genetic algorithm search for shape optimization". Computing In Civil Engineering., 10(2), 1996, pp. 143-150.
[3] CJ. Shih, CC. Chi and JH. Hsiao, "Alternative α-level-cuts methods for optimum structural design with fuzzy resources". Computers & Structures, 81, 2003,pp. 2579-2587.
[4] KC. Sarma, H. Adeli, "Fuzzy genetic algorithm for optimization of steel structures". Journal of Structural Engineering, 126(5), 2000, pp.596-604.
[5] RL. Haupt, SE. Haupt, "Practical genetic algorithms". New Jersey: John Wiley & Sons, 2004.
[6] H. Adeli, NT. Cheng, "Integrated genetic algorithm for optimization of space structures". Journal of Aerospace Engineering, 6(4), 1993, pp. 315-328.
[7] GY. Wang, WQ. Wang, "Fuzzy optimum design of structures". Eng Optim, 8, 1985, pp. 291-300.
[8] SS. Rao, K. Sundaraju, BG. Prakash and C. Balakrishna, "Multiobjective fuzzy optimization techniques for engineering design". Computers & Structures, 42(1), 1992, pp. 37-44.
[9] C. Carlsson, R. Fuller, "Fuzzy reasoning in decision making and optimization". New York : Springer Verlag , 2002.
[10] V. Rashtchi, E. Rahimpour and E. Mashhadi Rezapour, "Using a genetic algorithm for parameter identification of transformer R-L-C-M model". Electrical Eng, 88, 2006, pp. 417-422.
[11] Z. Michalewicz, "Genetic Algorithms + Data Structures Evolution Programs". New York: Springer Verlag, 1992.
[12] DE. Goldberg, "Genetic Algorithms in search, optimization and machine learning". Addition Wesley, 1989.
[13] A. Raie, V. Rashtchi, "Using a genetic algorithm for detection and magnitude determination of turn faults in an induction motor". Electrical Eng, 84, 2002, pp. 275-279.
[14] N. Ali, K. Behdinan, Z. Fawaz, "Applicability and viability of a GA based finite element analysis architecture for structural design optimization". Computers & Structures, 81, 2003, pp. 2259-2271.
[15] K. Deb, S. Gulati, "Design of truss- structures for minimum weight using genetic algorithms". Finite elements in analysis and design, 37, 2001, pp. 447-465.
[16] LJ. Li, ZB. Huang, F. Liu and QH. Wu. "A heuristic particle swarm optimizer for optimization of pin connectd structures". Computers & Structures, 85, 2007, pp. 340-349.
[17] KS. Lee, ZW. Geem, "A new structural optimization method based on the harmony search algorithm". Computers & Structures, 82, 2007, pp. 781-798.
[18] Ebadi MM. MS.c. Thesis, Department of Civil Engineering, Islamic Azad University Mahabad Branch, 2008.