A New Genetic Algorithm

For Continuous Structural Optimization

by

Xiaosu Ding

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2015 by the
Graduate Supervisory Committee:

Keith Hjelmstad, Chair
Narayanan Neithalath
Subramaniam Rajan

ARIZONA STATE UNIVERSITY

May 2015

ABSTRACT

In this thesis, the author described a new genetic algorithm based on the idea: the better design can be found at the neighbor of the current best design. The algorithm are described, including the rebuilding process from Micro-genetic algorithm and the different crossover and mutation formation. Some popular examples, including two variable function optimization and simple truss models are used to test this algorithm. In this study, the new genetic algorithm is proved as able to find the optimized results like other algorithms. Besides, the author also tried to build one more complex truss model. After tests, the new genetic algorithm can produce a good and reasonable optimized result. Form the results, the rebuilding, crossover and mutation can do the work as designed. At last, the author also discussed two possible points to improve this new genetic algorithm: the population size and the algorithm flexibility. The simple result of 2D finite element optimization showed that the effectiveness could be better, with the improvement of these two points.

To My Parents

ACKNOWLEDGMENTS

Frist, I would like to take this opportunity to express my gratitude to my advisor, Professor Keith Hjelmstad. His guidance and advice makes this work possible, and also inspire and encourage me throughout my future.

I would like to appreciate the Professors Rajan. The knowledge in his class is a big part of this thesis. And his advice about the genetic algorithm really helps me to understand my algorithm.

I am very thankful to my friends and colleges. With their assistance, I can make my work and my life easier.

At last, I want to appreciate my parents, whom this thesis is dedicated to. Without their supports, I cannot finish this thesis, like that I cannot finish anything in my 26 years of life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

CHAPTER 1

INTRODUCTION

In this paper, a new continuous genetic algorithm will be described and tested. In the first chapter, some background information is introduced to help understand the new genetic algorithm. The background information includes the details of basic genetic algorithm and the development of genetic algorithm.

1.1 The details of basic genetic algorithm

In this section, the formation of basic genetic algorithm from [2] is described. Basically, as shown in [6], a standard genetic algorithm should have the following process:

1. The initial population with the fitness
2. Evaluate all the designs (fitness calculation)
3. If not converged, go to loop

   3.1 Selection: select the individuals randomly from population as parents

   3.2 Crossover and Mutation: produce the new individuals by crossover and mutation

   3.3 Evaluate the new individuals (fitness calculation)

   3.4 Replace some individuals with new designs
4. End the calculation if converged.

1.1.1 Design representations

Traditionally, binary coding is widely implemented in genetic algorithm: one certain of binary string to present one variable. For example, to design one column, there are 15 kinds of formed steel columns. A four-digit binary string is enough for this column. So Column No. 1 could be:

$$0 \quad 0 \quad 0 \quad 0 \tag{1.1}$$

And the last one could be:

$$1 \quad 1 \quad 1 \quad 0 \tag{1.2}$$

All the variables will be coded in the same manner. The total length of binary strings for one design is shown as:

$$N_B = N_v \cdot N_b \tag{1.3}$$

Where $N_B$is the total binary strings length of the design; $N_v$is the number of variables; $N_b$is the binary length of one variable.

The binary coding is extremely suitable for the discrete variables and has been proved effective in practical calculation by [7]. In fact, for discrete variables, the genetic algorithm even has the advantage over the gradient based optimization.

The discussion of the continuous variables for the new genetic algorithm is shown in chapter 2.

1.1.2 Crossover and mutation

The crossover and mutation are the soul of genetic algorithm to make the optimization process like the natural evolution. For every iterations, the generation will operate the crossover and mutation to find the better designs. In this way, the population is refreshed by the new and better designs and will be dominated by one optimized design at the end of the calculation.

For the crossover, the most simple and widely used method is one point crossover. For example, two designs, $P_1$and P_2 are selected to operate the crossover and C_1 and C_2 are the products, shown as follows:

$$P_1\text{:}\ 1 \quad 1 \quad 0 \quad 1$$
$$P_2\text{:}\ 1 \quad 0 \quad 1 \quad 1$$

$$C_1\text{:}\ 1 \quad 1 \quad 1 \quad 1$$
$$C_2\text{:}\ 1 \quad 0 \quad 0 \quad 1$$

Figure 1. Crossover Process

In this way, as the binary strings are changed, the combination of variables is changed. If the crossover point is allowed to happen inside of one variable, this variable is changed as well. Therefore, the basic idea of using crossover is to make the better combination from current strings. Finally, the crossover will pull the population to the converged.

Mutation is used to maintain the diversity of the population and prevent the calculation pre-convergence. It is very easy to operate mutation for the binary strings, shown as follows:

$$M_1 = 1 \quad 1 \quad 0 \quad 1$$

$$M_2 = 1 \quad 0 \quad 0 \quad 1$$

Figure 2. Mutation Process

The second binary digit is mutated from 1 to 0.

In this way, mutation will create the new variable (the new string for one variable) from the outside of the population, to increase the creativity and diversity.

1.1.3 Constraints handling

For the genetic algorithm, the penalty function is used to control the constraint violations. Basically, the following equation is displayed as follows

$$\emptyset = \mathrm{W} + \mathrm{V} \tag{1.4}$$

Where $\emptyset$ is the fitness; W is the objective; V is the penalty function.

There are many ways to form a penalty function. Static penalty, adaptive penalty and dynamic penalty are the most widely used. All of these methods will add some values on the objective, according to the responses of calculation.

The smallest fitness means the best design with small objective and little penalty.

This is the summary of traditional genetic algorithm. It will be a big help to understand the rest of this paper.

1.2 The development of genetic algorithm

The genetic algorithm for structural optimization used to draw a number of attentions after [1] and [2] descried its basic idea. In the past 25 years, the genetic algorithm was developed a lot.

The very initial researches used, like [2], the basic genetic algorithm to do the sizing optimization of ten bar model, whose major efforts lied in describing and proving that the genetic algorithm could find out the best designs.

Moreover, researches like [3] [4] [5] who published in 1990s began to focus on the discussion of genetic algorithm parameters and trying more complex problems. [3], for example, tried to discuss the formation of penalty function. In [4], the author provided "precision" and made a continuous problem as a discrete one. Moreover, [4] also proved that the genetic algorithm can solve a simple shape and topology problem.

In the 21st century, according to a number of researches studying structural optimization of genetic algorithm, some bench marks such as 10 bar truss model and 18 truss model are used to test the new algorithm. In this paper, some models will be tested for the new algorithm to compare with other genetic algorithms. So more introductions and details are shown in chapter 3.

Moreover, researchers were likely to invent a better genetic algorithm for the continuous variable problems. One of the most important studies is harmony search, [8] and [9]. In this study, Geem built the genetic algorithm based on the idea of music harmony. Basically, for the bench marks of truss like models, Geem found the optimized designs although the most of recent studies had similar results. This part of discussion will be shown in chapter 3.

Besides, recently, researchers pay some attentions to topology optimization of two dimensional (2D) finite element. One of them is [10], whose idea and algorithm are proved successful to do some discrete topology optimization of 2D solid model.

1.3 Motivation of this research

In this study, a new genetic algorithm, based on the idea: better designs found in the neighbor of the best design, is described and tested. The author believes that this algorithm is a good addition of genetic algorithm map.

Moreover, in 25 years, genetic algorithm for structural optimization was improved, but all the algorithms were focused on some simple models. This could limit the improvement of genetic algorithm, so the author tried to show the more complex model in this study and use this model to test the algorithm.

As right now, the genetic algorithm cannot optimize the 2D solid model with continuous variables. In this study, the author also wants to discuss about the probability of the new genetic algorithm doing this 2D solid optimization problem.

1.4 The structure of the rest of this paper

In chapter 2, the new genetic algorithm (New GA) is described. In this section, the differences between the new and the traditional one will be explained and discussed. In chapter 3, the new genetic algorithm will be tested in some models. Some good functions and some simple bench mark models are optimized. All the results will be analyzed and compared. In chapter 4, the new complex truss model for test is described and the new algorithm will be used to optimize this model. The results will be analyzed. In chapter 5, the conclusions and future work are discussed.

.

CHAPTER 2

THE FORMATION OF NEW GENETIC ALGORIHTM

The new genetic algorithm in this study is based on the idea that a better design found in the neighbor of the best design. So the standard genetic algorithm mentioned in chapter one is modified in two points.

2.1 The rebuilding based on micro genetic algorithm

The genetic algorithm cannot improve the design effectively at the latter part of computation. For this problem, the mutation in the traditional genetic algorithm is invented to prevent the convergence to the local optimum. However, another method for making the designs to jump from the local optimum is discussed in [11]. This method called micro genetic algorithm was adopted in this study. As described in [11], when the computation cannot improve the designs effectively, the algorithm did not try to prevent the convergence but stopped the current iterations, rebuilt the new population based on the current designs and restarted the genetic algorithm process. In [12] and [13], the micro genetic algorithm is proved to be able to help the optimization process by jumping out from the local optimal point. For example, in [13] the author proved that micro genetic algorithm can solve a complex dynamic optimization problem with a population of only five individuals.

Similarly, in this study, the idea of micro genetic algorithm is used and recreating the new population called rebuilding. Every generation computed by the genetic algorithm is called one iteration. When the convergence criterion is met, the population restarts. Then all of the iterations between two restarts called one loop are optimizing.

After one loop ends, the new population is built by the best design of this loop. The basic idea of rebuilding is to change every variable of the best design with a multiplier and, that is,

$$x_i^n = c \cdot x_i^b \tag{2.1}$$

where, x_i^n is variable Number i of the new design and x_i^b is variable Number i of best design in the previous loop. c is a multiplier, randomly selected from a number group $\mathbf{N}$. In this study, we recommend that the number group with the range r and the sparseness s, and the numbers in the group are uniformly distributed controlled by these two parameters.

For example, if one number group has $r = [-0.5,1.5]$ and $s = 4$, then

$$c \in \mathbf{N} = \{-0.5,0,0.5,1.0,1.5\} \tag{2.2}$$

When building the initial population for the first loop, the user could choose to provide a design for rebuilding like gradient based optimization. This design can be regarded as the best design of loop zero. Or, the initial population can be built randomly, like the standard genetic algorithm.

The best design of the previous loop will be included in the new built population to make sure the computation is stable, which is called the elite strategy.

2.2 The new formation of mutation and crossover

In this study, the crossover and mutation is formed by the real values rather than binary coding.

2.2,1 Crossover

For crossover, two designs are selected randomly from the population. Some random variables in one design are replaced by those of the other design, to form the new one, that is,

$$\begin{cases} X_1 = \{x_1^1, x_k^1, \dots, x_l^1, \dots, x_n^1\} \\ X_2 = \{x_1^2, x_k^2, \dots, x_l^2, \dots, x_n^2\} \\ X_{new} = \{x_1^1, x_k^2, \dots, x_l^2, \dots, x_n^1\} \end{cases} \tag{2.3}$$

where, $X_1$and $X_2$are the selected designs; X_{new} is the new design; x_k^1andx_l^1 are the variables of Number k and l, respectively.

2.2.2 Mutation

For mutation, one design is selected randomly, whose random variables will be mutated in the similar way with rebuilding:

$$x_i^m = m \cdot x_i^p \tag{2.4}$$

The mutation process is:

$$\begin{cases} Y_1 = \{y_1, \dots, y_i^p, \dots, y_j^p, \dots, y_n\} \\ Y_{new} = \{y_1, \dots, y_i^m, \dots, y_j^m, \dots, y_n\} \end{cases} \tag{2.5}$$

where, i means variable Number i is selected for mutation; $y_i^m$is the mutated value of the selected variable; $y_i^p$is the original value of the selected variable; m is the mutation multiplier selected from a mutation number group $\mathbf{M}$. The mutation number group also has the range r_m and the sparseness s_m. In this study, we recommend that $r_m$as 0.0001 and s_m as 20.

2.2.3 The test for survival

After the new designs are produced, the fitness is calculated. For rebuilding, the designs will be ranked via their fitness: larger fitness means worse design. For crossover and mutation, only if it has a smaller fitness value than the worst fitness in the population does the new design survive and replaces the current worst fitness individual.

2.3 The algorithm process

The algorithm details are shown below:

Step 1: Choose one start design for the program, or start with a random population

Step 2: Start the new loop:

Step 2.a: Create the new population, loop over all designs in the population:

For each design, every variable is built by (2.1), based on the start design or the previous loop best designs.

Step 3: Start the new iteration:

Step 3.a: Operate the crossover for the current iteration:

For each time of crossover, two designs are selected randomly to produce a new design, as described in 2.2.1. The new design fitness is calculated for the test of survival, as described in 2.2.3.

Step 3.b: Do mutation for the current iteration:

For each time mutation, one design is selected randomly and some random variables are mutated by (2.4) to build one new design. The new design fitness is calculated for the test of survival, as described in 2.2.3.

Step 4: End the current iteration: if maximum iteration number or other criterion is met; go to step2; if not, go to step 3.

Step 5: End the current loop: if maximum loop or other criterion is met, go to step 6; if not, go to step 3.

Step 6: Output the result.

CHAPTER 3

THE OPTIMIZATION OF FUNCTIONS AND SIMPLE TRUSS MODELS

In this section, there are two kinds of tests for the new genetic algorithm. Part one is the basic function optimization with two variables. Part two is some simple popular truss models, also used by other papers.

3.1 Two variables functions optimization

In this part, we used the new genetic algorithm to do function optimization. For plotting easily, all functions have two variables.

3.1.1 Basic formation of functions optimization

The optimization program is built in the way discussed in section 2, but some points are valuable to mention for the particular function optimization. First, the population size N_p is set as 10 to ensuring the computation stable, instead of 4 or the square of variable number. The total loop number N_l is 50; the iteration number for one loop N_i is 20; the probability of crossover P_c and mutation P_m is 0.8 and 0.1, respectively.

So the total number of function evaluation can be calculated by the equation below:

$$N_t = N_l \cdot (N_i \cdot (P_c + P_m) + 1) \cdot N_p \tag{3.1}$$

For one run, the total evaluation number is about 9500 for the current genetic algorithm formation. The start point is (-0.8,-0.8) for all four functions.

For rebuilding, the range $r_n = [-2,2]$ and the sparseness $s_n = 20$. For mutation, the range $r_m = [0.999,1.001]$ and the sparseness $s_m = 20$.

As these are unconstrained optimization problems, no penalty functions are built.

3.1.2 Functions optimization results

The first function is:

$$\mathrm{F}(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7 \qquad (3.2)$$

The convergence history of this function is shown in figure 3. If 1.0×10^{-7} is the tolerance, the program used about 30 loops to finish the convergence. The optimal point is (0.05, 0).

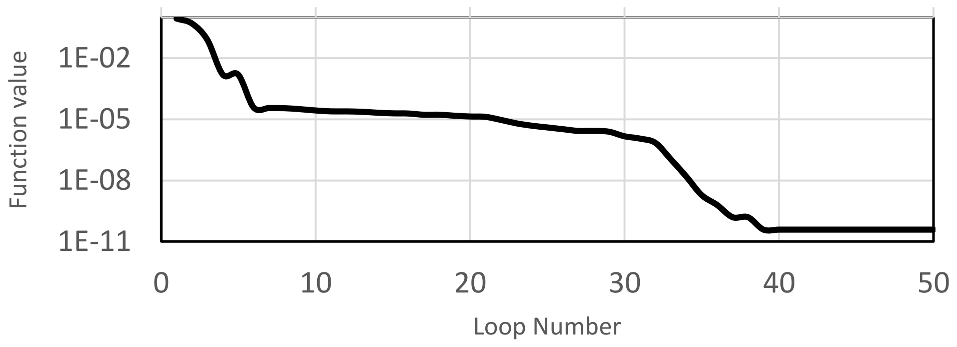

Figure 3. Convergence of Function (3.2)

The second function is:

$$\mathrm{F(x,y)} = -(\cos(2\pi\mathrm{x}) + \cos(2.5\pi\mathrm{x}) - 2.1) \times (2.1 - \cos(3\pi\mathrm{y}) - \cos(3.5\pi\mathrm{y})) \ (3.3)$$

The convergence history of this function is shown figure 4. Compared to the theoretical value in table 1, the program used about 45 loops to find the optimum. The optimal point is (0.4388, 0.3856)

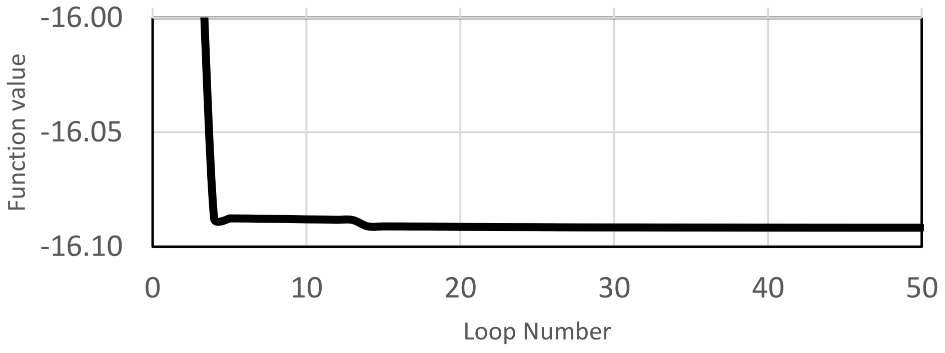

Figure 4. Convergence of Function (3.3)

The third function is:

$$f(x,y) = (y - \frac{5.1x^2}{4\pi^2} + \frac{5}{\pi}x - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x) + 10 \qquad (3.4)$$

The convergence history of this function is shown in figure 5. Compared with the theoretical value, the program used 45 loops to find the optimum. The optimal point is (3.1133, 2.2918).

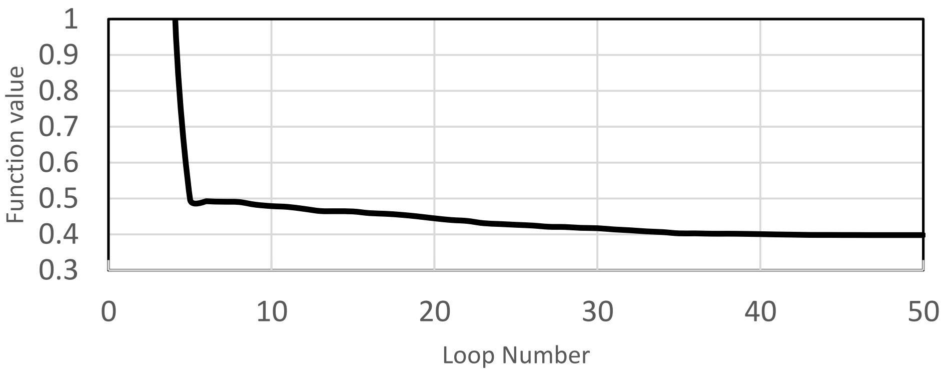

Figure 5. Convergence of Function (3.4)

The forth function is:

$$f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + (-4 + 4y^2)y^2 \qquad (3.5)$$

The optimization process for this function is shown below. It seems that the program used 25 loops for this test. The optimization map and function plot is shown below, as well. The optimal point is (0, -0.7072)

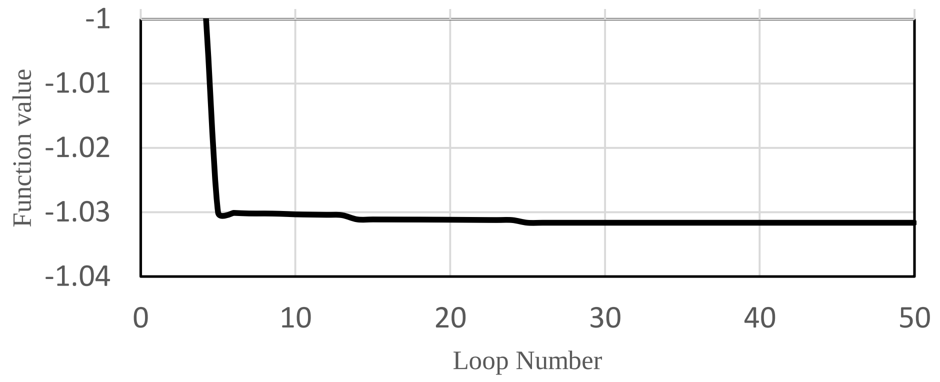

Figure 6. Convergence of Function (3.5)

All results for comparison are from [14]. From Table 1, the new genetic algorithm is able to locate the optimum and is similarly effective with the standard genetic algorithm (SGA).

One could criticize that the new genetic algorithm depends on the start points, and if the start points are totally wrong, the program could not find the optimum. In fact, the standard genetic algorithm will face this challenge. For example, for the first function, in [14], the search space is

$$-1.28 \leq x_1, x_2 \leq 1.27 \tag{3.6}$$

The search space is based on one's guess. So it would become a problem if basic knowledge tells us that the optimum lives in the region as

$$-128 \leq x_1, x_2 \leq 127 \tag{3.7}$$

When the search region is changed for the standard genetic algorithm, the population size and the evaluation number could increase by 10 times. A large search domain could happen in a complex and unknown structural optimization problem. So for both algorithms, the ability to find the optimum quickly in a large search space is very important. A start point far from the optimum is used to prove it.

Table 1

Summary of Function Optimization

Function	theoretical minimum	minimum (SGA)	Function evaluations (SGA)	Minimum (New GA)	function evaluations (New GA)
3.2	0	4.41E-06	4000	4.03E-06	5000
3.3	-16.09172	-16.09171	4000	-16.09172	8500
3.4	0.39789	0.39789	8000	0.39789	8500
3.5	-1.03163	-1.03163	6000	-1.03163	4250

For the final test of function optimization, the new genetic algorithm is used at (-128,-128) as the start variables. The other parameters remain the same with the previous tests. The convergence process is shown in figure 7.

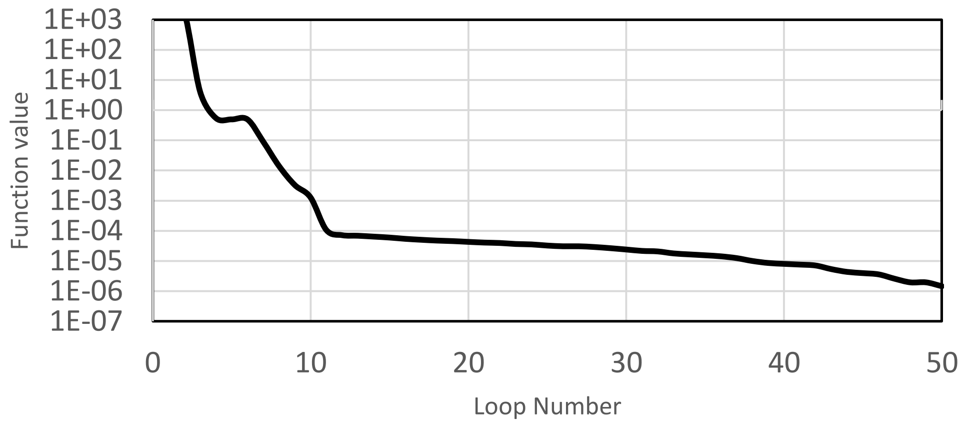

Figure 7 Function (3.2) Convergence with Bad Start

Figure 8 showed the function (3.2) contour plot every ten loops. The x-axis and y-axis mean two variables, respectively. The spots in the figure are the plots of all the individuals loops. The figure at the upper-left corner has a very large domain, which indicates at the beginning of the calculation, the program had to use some efforts to find the neighbor of the optimum. After 10 loops, the program just focused on finding the optimum in a small region, so the final four plots have the same domain. It proved that the new genetic algorithm can locate and find the optimum fast even if the start value is bad.

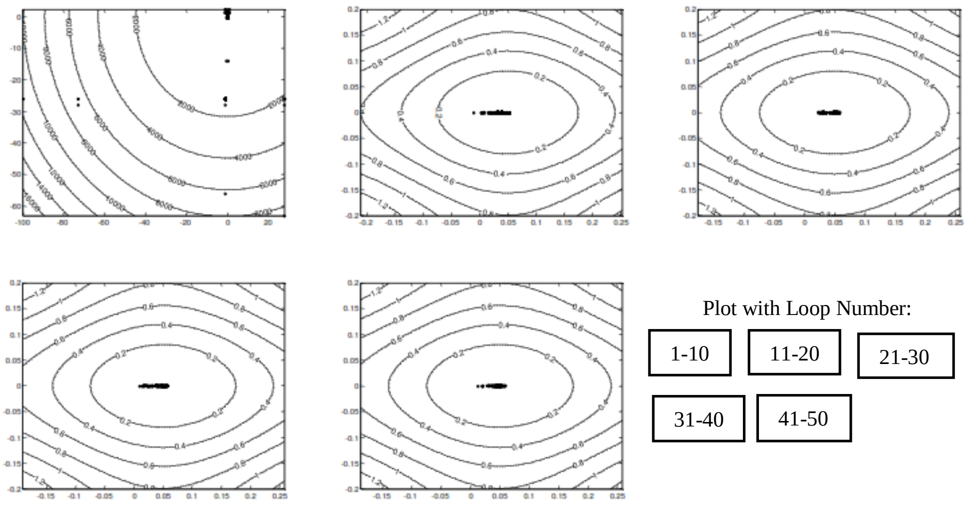

Figure 8. Fucntion (3.2) indivaduals plots with bad start point

3.2 Sizing optimization for simple models

For the second part of this section, some popular structural models are used to test the new genetic algorithm.

Basically, the structural optimization formation in this part is similar with the first part, but there are some different points.

First, the population size N_p is different and calculated with the equation below:

$$N_p = N_v^2 \tag{3.8}$$

In the (3.8), N_v is the number of variables.

Second, the structural optimization is a constrained optimization so the penalty function is important. In this study, the penalty function is:

$$P = \sum_{i=1}^{n} c \cdot g_i \quad \begin{cases} g_i = 0, & if r_i \le r_0 \\ g_i = \frac{r_i}{r_0} - 1 & if\ r_i > r_0 \end{cases} \tag{3.9}$$

where, c is penalty constant, set as 1000, in this study; r_i is the structural response, like displacement and stress; r_0 is the constraint values, allowable displacement or allowable stress. So the penalty is a static penalty function.

3.2.1 Ten bar model

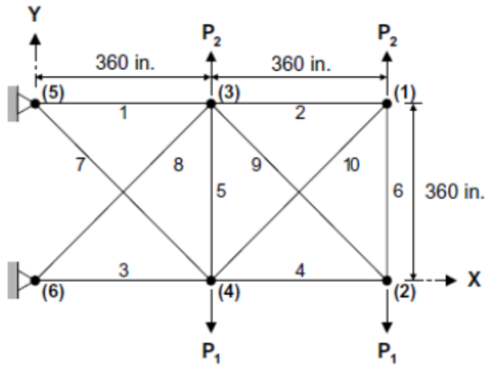

Figure 9. Ten bar model

The ten bar model is the first example to test the new genetic algorithm, shown in figure 9. As the most popular model, it is analyzed by many researchers, such as Schmit and Farshi [15], Schmit and Miura [16], Venkayya [17], Dobbs and Nelson [18], Rizzi [19], Khan and Willmert [20], Lee and Geem [9].

Two cases are tested, case one: $P_1 = -150kips$, $P_2 = 50kips$ and case two: $P_1 = -100kips$, $P_2 = 50kips$. The bar model has simply ten sizing variables; the objective is the total weight; for one member, the model is subjected to the stress constraint $\pm$25ksi and the displacement is $\pm$2in or ± 0.167ft.

The material density is $0.1lb/in^2$ and the elastic modulus is 10,000 ksi; each bar has the minimum size of $0.1in^2$. The results for two cases are shown in table 2 and table 3:

Table 2

Ten bar design comparison of case one

Method	1	2	3	4	5	6	7	8	9	10	Objective
Schmit & Farshi [15]	24.29	0.1	23.35	13.66	0.1	1.969	12.67	12.54	21.97	0.1	4691.84
NEW SUMT [16]	23.55	0.1	25.29	14.36	0.1	1.97	12.39	12.81	20.41	0.1	4676.96
CONMIN [16]	23.55	0.176	25.2	14.39	0.1	1.967	12.4	12.86	20.41	0.1	4684.11
Venkayya [17]	25.19	0.363	25.42	14.33	0.417	3.144	12.08	14.61	29.26	0.513	4895.6
Dobbs & Nelson [18]	25.81	0.1	27.23	16.65	0.1	2.024	12.78	14.22	22.14	0.1	5059.7
Rizzi [19]	23.53	0.1	25.29	14.37	0.1	1.97	12.39	12.83	20.33	0.1	4674.92
Khan & Willmert [20]	24.72	0.1	26.54	13.22	0.108	4.835	12.66	13.78	18.44	0.1	4792.52
Geem [9]	23.25	0.102	25.73	14.51	0.1	1.977	12.21	12.61	20.36	0.1	4668.81
New algorithm	24.62	0.11	23.36	13.74	0.1	0.94	14.36	9.12	20.23	0.1	4514.4

From the results, the new genetic algorithm is able to find the optimized design, like the other algorithm. For both cases, the total model evaluations number is about 20,000.

Table 3.

Ten bar design comparison with case two

Method	1	2	3	4	5	6	7	8	9	10	Objective
Schmit & Farshi [15]	33.43	0.1	24.26	14.26	0.1	0.1	8.388	20.74	19.69	0.1	5089
NEW SUMT [16]	30.67	0.1	23.76	14.59	0.1	0.1	8.578	21.07	20.96	0.1	5076.85
CONMIN [16]	30.57	0.369	23.41	14.73	0.1	0.1	8.547	21.11	20.77	0.1	5107.3
Venkayya [17]	30.42	0.128	23.41	14.91	0.1	0.1	8.696	21.08	21.08	0.32	5084.9
Dobbs & Nelson [18]	30.5	0.1	23.29	15.43	0.1	0.21	7.649	20.98	21.82	0.1	5080
Rizzi [19]	30.73	0.1	23.29	14.73	0.1	0.1	8.542	20.95	21.84	0.1	5076.66
Khan & Willmert [20]	30.98	0.1	24.17	14.81	0.1	0.406	7.547	21.05	20.94	0.1	5066.98
Geem [9]	30.15	0.102	22.71	15.27	0.102	0.544	7.541	21.56	21.45	0.1	5057.88
New algorithm	31.2	0.1	22.88	15.05	0.1	1.0326	5.78	22.53	20.23	0.1	5032

3.2.2 Seventeen bar model

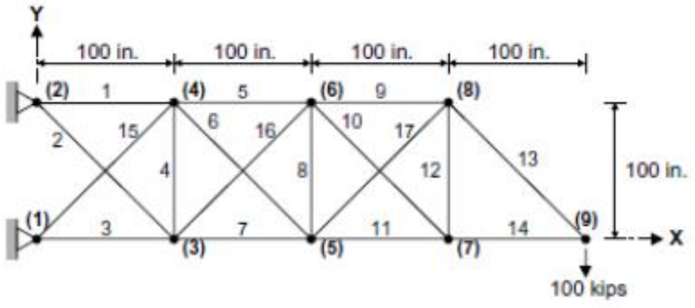

Figure 10. 17 bar model

The seventeen bar model is analyzed by Khot and Berke [21], Adeli and Kumar [24], Geem [9]. For designs, the objective is the total weight; the design variables are the member size for all of the members; the stress constraint for every member is $\pm 50\text{ksi}$ and displacement constraint is $\pm 2\text{in}$.

For one single load case, the load is in y direction and placed at node 9, which is -100kip. The material density is $0.268 lb/in^2$ and the elastic modulus is $30{,}000$ kis. The results are shown in Table 4. The minimal size is $0.1 in^2$.

Table 4.

Summary of 17 bar optimization

variables	Khot & Berke [21]	Adeli & Kumar [24]	Geem [9]	This study
1	15.93	16.029	15.821	15.89671
2	0.1	0.107	0.108	0.113516
3	12.07	12.183	11.996	12.19178
4	0.1	0.11	0.1	0.1
5	8.067	8.417	8.15	8.121617
6	5.562	5.715	5.507	5.581791
7	11.933	11.331	11.829	11.85274
8	0.1	0.105	0.1	0.1
9	7.945	7.301	7.934	7.977783
10	0.1	0.115	0.1	0.1
11	4.055	4.046	4.093	4.062786
12	0.1	0.101	0.1	0.1
13	5.657	5.611	5.66	5.650435
14	4	4.046	4.061	3.991059
15	5.558	5.152	5.656	5.574801
16	0.1	0.107	0.1	0.1
17	5.579	5.286	5.582	5.49158

From the result table above, the design produced by the new genetic algorithm is similar with other algorithms. The total model evaluations number is about 20,000 in this study.

3.2.3 twenty-two bar model

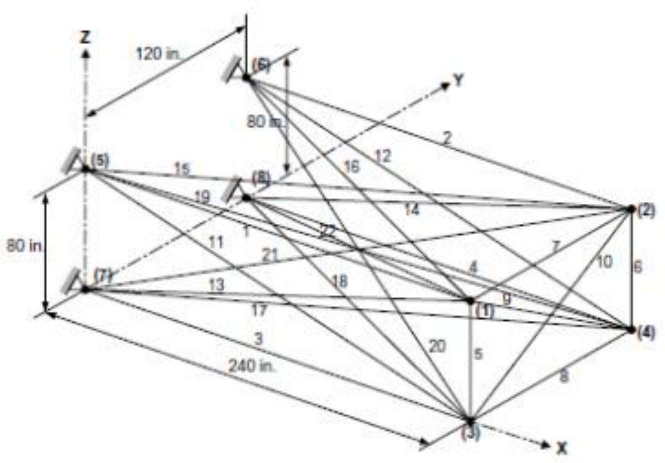

Figure 11. 22 bar model

The twenty-two bar model is shown in figure 11. The constraint conditions are the same with the previous studies, but this study we consider three load conditions for one design. The load conditions are shown in table 5.1, 5.2 and 5.3.

Table 5.1

Load condition 1 for 22 bar model

node	x	y	z
1	-20	0	-5
2	-20	0	-5
3	-20	0	-30
4	-20	0	-30

For the sizing optimization, the 22 members are parted as seven groups, shown in table 5 and the results are shown in Table 6.

Table 5.2

Load condition 2 for 22 bar model

Node	x	y	z
1	-20	-5	0
2	-20	-50	0
3	-20	-5	0
4	-20	-50	0

Table 5.3

Load condition 3 for 22 bar model

Node	x	y	z
1	-20	0	35
2	-20	0	0
3	-20	0	0
4	-20	0	-35

Obviously, the results from this study are totally different from the previous two researches. However, the model designed by other algorithms have a displacement, two times larger than the displacement constraint. So it is unfair to compare these two results, but it is fair to say that the new genetic algorithm can produce a feasible design.

Table 6

Summary of 22 bar model optimization

Variable	Member group	Geem [9]	This study
1	1,2,3,4	2.588	5.4517953
2	5,6	1.083	2.15690557
3	7,8	0.363	0.1
4	9,10	0.422	0.9403604
5	11,12,13,14	2.827	4.56553742
6	15,16,17,18	2.055	6.42514959
7	19,20,21,22	2.044	4.93652054
Load case		displacement	displacement
Case 1		4.8828	2.00108994
Case 2		2.3064	1.02920926
Case 3		4.9212	2.00734688

3.2.4 Twenty-five bar model

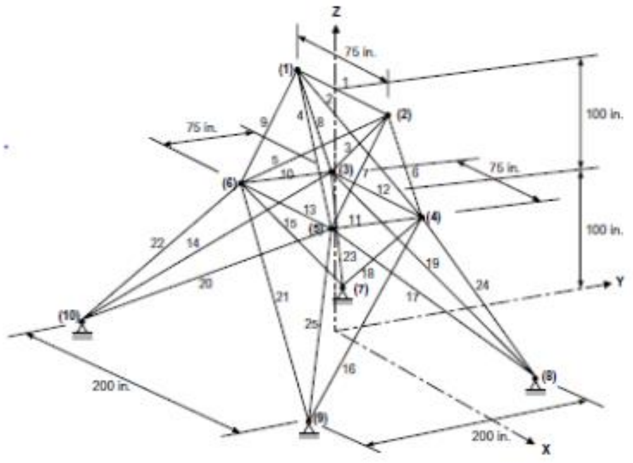

Figure 12. 25 bar model

The 25 bar model, shown in figure 12 is very popular model for testing the GA and was analyzed by Schmit and Farshi [15], Schmit and Miura [16], Venkayya [17], Dobbs and Nelson [18], Rizzi [19], Khan and Willmert [20], Lee and Geem [9], Gallatly and Berke [21], Templeman and

Winterbottom [22], Chao [23], Adeli and Kamal [24], John [25], Saka [26], Fadel and Clitalay [27], Stander [28], Xu and Grandhi [29] and Lamberti and Pappalettere [30].

Table 7.1

Load condition 1 for 25 bar model

Node	x	y	z
1	0	20	-5
2	0	-20	-5
3	0	0	0
6	0	0	0

Table 7.2

Load condition 2 for 25 bar model

Node	x	y	z
1	1	10	-5
2	0	10	-5
3	0.5	0	0
6	0.5	0	0

The model is under two load conditions, shown in Table 7.1 and 7.2, so for each evaluation, the model is analyzed twice for each load condition. The 25 members are parted as eight groups to do the sizing optimization. The objective is total weight. For constraints, the displacement constraint is placed on all the nodes in the three directions as $0.35\ in$.

The elastic modulus and the material density is $10{,}000ksi$ and $0.1lb/in^2$, respectively. And the minimum size is taken as $0.01in^2$

Table 8.1

Summary of 25 bar model design, part one

Variable	Member	Schmit & Farshi [15]	Venkayya [17]	Rizzi [19]	Khan & Willmert [20]	Templeman [22]	Chao [23]
1	1	0.01	0.028	0.01	0.01	0.01	0.01
2	2,3,4,5	1.964	1.964	1.988	1.755	2.022	2.042
3	6,7,8,9	3.033	3.081	2.991	2.869	2.938	3.001
4	10,11	0.01	0.01	0.01	0.01	0.01	0.01
5	12,13	0.01	0.01	0.01	0.01	0.01	0.01
6	14,15,16,17	0.67	0.693	0.684	0.845	0.67	0.684
7	18,19,20,21	1.68	1.678	1.677	2.011	1.675	1.625
8	22,23,24,25	2.67	2.624	2.663	2.478	2.697	2.672

Table 8.2

Summary of 25 bar model design, part two

variable	member	Saka [20]	Stander [22]	Adeli & Kamal [24]	Geem [9]	This study
1	1	0.01	0.01	0.01	0.047	0.04778
2	2,3,4,5	2.085	2.043	1.986	2.022	1.97185
3	6,7,8,9	2.988	3.003	2.961	2.95	2.94177
4	10,11	0.01	0.01	0.01	0.01	0.01
5	12,13	0.01	0.01	0.01	0.01	0.01055
6	14,15,16,17	0.696	0.683	0.806	0.688	0.60812
7	18,19,20,21	1.67	1.623	1.68	1.657	1.61073
8	22,23,24,25	2.592	2.672	2.53	2.663	2.48596

According to the results in Table 8.1and 8.2, the design from the new genetic algorithm is similar with other methods. So the new GA can find the optimized design in 20,000 evaluations of models.

3.3 Conclusions

For the function optimization, the new genetic algorithm was proved to be able to find the optimum in an acceptable speed, compared with the other algorithms. Especially, the search ability of the new genetic algorithm is proved as a great one, because it can search with a bad and far away start point.

For the simple truss optimization part, similar with other results, the new genetic algorithm can also find the optimized solution. In one example, the results in other algorithms obviously, violated the constraint. As the result, for the basic models, the new genetic algorithm is successful to produce the good results.

These bench mark models are not enough to test one genetic algorithm. There are two problems on them: first, these models are displacement driven, which have the active displacement constraint and very small stress. So stress constraint of the current optimization problem is non-active one. Second, the current optimization is only sizing optimization. The shape optimization is also a very important part in the optimization problems.

In the next section, we will introduce one complex model and try to optimize it in both of sizing and shape optimization with the new genetic algorithm.

CHAPTER 4

THE COMPLEX TRUSS MODEL

Like mentioned above, the widely used truss models can not cover all the needs for testing the genetic algorithm. So a more complex truss model is described and used to test the new genetic algorithm in this section.

4.1 The truss tower

In general, one possible design of the truss tower is shown in figure 13. It has a very simple structure, with three levels and three rings formation. The limitation is simple, as well. Only the height and the top radius are the fixed numbers. The total height of the tower is 20000 meters (65615 *feet*). The radius of top ring means the distance of the vertex and the center point of the top surface.

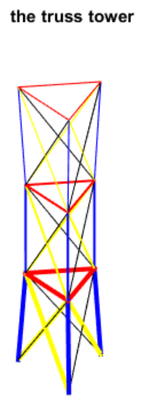

Figure 13. The truss tower model

For example, figure 14 shows the top surface of the truss tower. Obviously, the radius is from vertex to the origin, fixed as 10000 *feet*. Expect for these two parameters, other values to form this structure are based on shape or sizing variables, described in the next part.

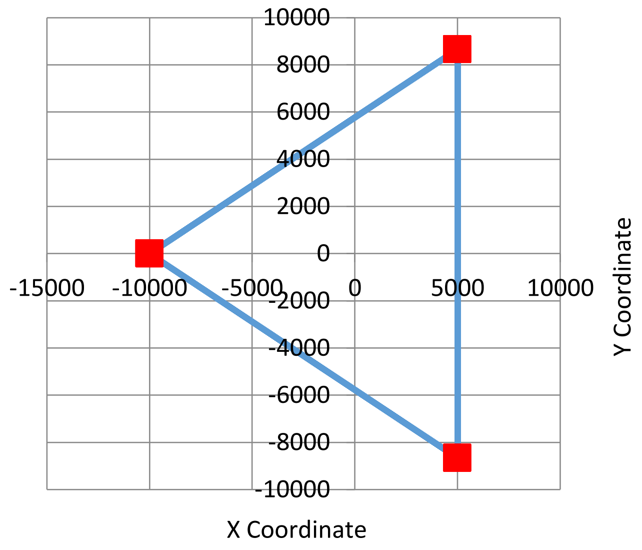

Figure 14. Top ring

4.1.1 Optimization problem formation

Basically, one three level and three ring truss tower will be optimized as a sizing and shape optimization. First, for sizing optimization, all the members will be optimized for their best member area. For the symmetric property and convenient calculation, these members are parted as several groups by position and levels. The look of the 3 level tower is shown in figure 13 to make the member groups clear. From the figure 13, obviously, there are four member groups (four colors) in one level, one vertical member (blue), two diagonal members (yellow and black) and one ring (red). So in total, for 3 level and 3 rings tower, 12 groups of member size will be optimized so there are 12 variables.

Moreover, as the height of the tower is extremely tall, buckling is a big issue. Considering this issue, all the members are likely to be designed as a fractional one. In this study, the fractional design is simplified as radius variable for the member group. So the member could be designed with a larger radius than the solid radius to avoid the local buckling. The calculation about the local buckling will be showed in structural analysis part. In total for the sizing optimization, 24 variables are optimized.

For shape optimization, two variables, the height and the ring radius in one level are optimized considering the symmetric property. The height variable means the z-axis value of one node on the level, and the radius, like described for top radius, is the distance between one vertex and the center point. As the top radius is fixed and the level 0 (ground) radius is free to design, one 3 levels, 3 rings tower, has 6 shape optimization variables. Therefore, for the simplest tower, 30 variables are to be optimized.

4.1.2 Structural parameters of the tower

In this section, the structural analysis about the truss tower is introduced, including the structural parameters, the force formation and the analysis of local buckling.

First, as a truss model, truss finite element method is used for structural analysis. As mentioned, the height and the top radius $h = 65619f$ and $R_t = 10000f$. The elastic modulus is $E = 29000ksf$. The material density is $\rho = 0.492pcf$. The ultimate stress for the material is $\sigma_u = 90ksi$ and the allowable stress is $\tau_a = 0.7$.

There are three kinds of external forces applied on the tower, payload, self-weight and wind load. The payload $P = 4 \times 10^8 lb$ could be simply seen as the nodal forces, averagely distributed at each nodes of the top ring. So for the 3 ring model, every top node will take $P_n = \frac{P}{3}$ as the node forces.

The other two forces are applied on the members, so they could be replaced by equal nodal forces based on the finite element method. Like other truss models, member weight is distributed at two end points equally.

The wind load is a very important part because of the tower height. Wind force is the product of wind area and wind pressure.

As the member could be fractional, the wind area is calculated as:

$$A_w = 0.7 \cdot \alpha \cdot R_L \cdot L_m \tag{4.1}$$

where A_w is wind are of one member; α is the cosine direction value between wind and member; L_m is member length.

On the other hand, the wind pressure is simplified as the wind pressure of the middle point of one member. So the wind pressure for this member is calculated as:

$$P_w = 0.5 \cdot \rho \cdot v_m^2 \tag{4.2}$$

where, P_w is the wind pressure; ρ is the air density; v_m is the wind speed.

The air density function is:

$$\rho = d_a \cdot (1 - \rho_b \cdot h_m)^{\eta} \tag{4.3}$$

$d_a = 0.7647 \times 10^{-4}$, as the air density constant on the ground; $\rho_b = 0.68756 \times 10^{-5}$ and $\eta = 4.2576$ as the air density constants.

The wind speed is calculated by two functions because the top wind speed is at the height $h_s = 38000\, feet$. So the wind speed is given by:

$$v_m = \begin{cases} v_0(1 + b(1-y) + c(1-y)^2) & y \leq 1 \\ v_0\left(\frac{1}{[1+e(1-y)^f]^g}\right) & y \geq 1 \end{cases} \tag{4.4}$$

Where v_0 is peak wind speed at h_s . y is the ratio of member height and h_s. Other constants are shown the table as:

Table 9.

Summary of wind speed constant

wind speed constant	b	c	e	f	g
value	-1.4601	0.5202	160	1.6	0.25

The local stability analysis is different to the most structural optimizations. In general, the member radius R_L only needs to meet Euler buckling equation. The critical force for a compression column can be calculated by:

$$F_c = \frac{\pi^2 EI}{L^2} \tag{4.5}$$

Where F_c is the critical force; L is column length and I is inertia moment.

In this study, as the member is fractional and not defined by any shapes, the Rankine Gordon formula is used. By avoiding to calculate the inertia moment, the critical stress by stability is:

$$\sigma_c = \frac{F}{A} = \frac{\pi^2 E}{(L/R_L)^2} \tag{4.6}$$

On the other hand, the allowable stress is given above,

$$\sigma_a = \sigma_u \tag{4.7}$$

According to the Rankine Gordon formula, we have

$$\frac{1}{\sigma_{max}} = \frac{1}{\sigma_a} + \frac{1}{\sigma_c} \tag{4.8}$$

The allowable stress for local stability is

$$\sigma_a^L = \sigma_{max} = \frac{\sigma_c \cdot \sigma_a}{\sigma_a + \sigma_c} \tag{4.9}$$

Obviously, the allowable stress for compression member is the function of member radius. In fact, there are two ways to improve the compression member if one is overstressed. The first is to enlarge the size and the other one is to enlarge the radius.

4.2 optimization problem formation

The aim of this problem is to find the optimized design with acceptable stress and displacement. The objective is the total weight, that is:

$$W = \sum_{i=1}^{n} L_i \cdot \rho \qquad (4.10)$$

Where, W is the objective; L_i is one member length; ρ is material density.

Like the simple truss models, two kinds of constraints are applied, stress and node displacement. For stress, the stress ratio value is calculated as:

$$R_s = \frac{\sigma_m}{\sigma_a} \qquad (4.11)$$

Where, R_s is the stress ratio and the constraints for all the member is 0.7; $\sigma_m$is the absolute value of one member stress ratio; σ_a is the allowable stress and if the member is in compression, this value is σ_a^L by the Rankine Gordon Formula.

For displacement constraint, the largest displacement will be monitored, and this constraint is smaller than $600f$. The two constraints are summarized in the table 10 below.

Table 10

Summary of Constraints

constraint	**stress ratio**	**displacement**
position	all members	maximum value
value	0.7	600 *f*

4.3 The genetic algorithm formation

In this section, the new genetic algorithm is described for the truss tower. The crossover is the same with that of the simple truss models. The mutation is programmed with these two parameters: $r_m = 0.0001$ and $s_m = 20$. The rebuilding has two parameters: $r = [0.75,1.25]$ and $s = 20$. The loop number $N_l = 20$ and the iteration number $N_i = 100$. The population size is $N_p = 900$.

As the truss tower is an unknown structure, the initial population is produced randomly. Like the idea of rebuilding, when the region and the number of possible variables are given, all the variables for the new population will be created randomly and easily.

Two penalty functions are used for displacement and stress. The penalty for displacement penalty is created as:

$$P_d = \begin{cases} c_d \cdot (d - d_0)^2 \; if\; d \geq d_0 \\ 0 \qquad\qquad\quad ifd \leq d_0 \end{cases} \tag{4.12}$$

where, $P_d$is displacement penalty; c_d is the penalty constant for displacement, as 10^{12}; d is the absolute value of the largest displacement. d_0 is displacement constraint.

On the other hand, the penalty function for stress is linear function

$$P_s^n = \begin{cases} c_1 \cdot (R_s^n - 0.7) & ifR_s^n > 0.7 \\ c_2 \cdot (R_s^n - 0.5) & ifR_s^n > 0.5 \\ 0 & ifR_s^n \leq 0.5 \end{cases} \tag{4.13}$$

where, P_s^n is stress penalty of stress; R_s^n is the stress ratio; $c_1$and c_2 are penalty constants by:$c_1 = 50 \cdot W_b$ and $c_2 = 0.01 \cdot W_b$; W_b is the weight with the smallest fitness from the previous loop and the initial value is 10^{12}.

As the result, the total value of stress penalty will be:

$$P_s^t = \sum_i^n P_s^i \tag{4.14}$$

4.4 The result and discussion

4.4.1 The research on convergence plot

Figure 15 showed the total weight is improved by loops. In general, the objective is improved loop by loop, from 2.39×10^{12} to 6.7×10^{10}. The loop 20 can improve the objective by 6% , from 7.1×10^{10} to 6.7×10^{10}. It means that at the end of calculation, the objective can be improved effectively, indicating the convergence.

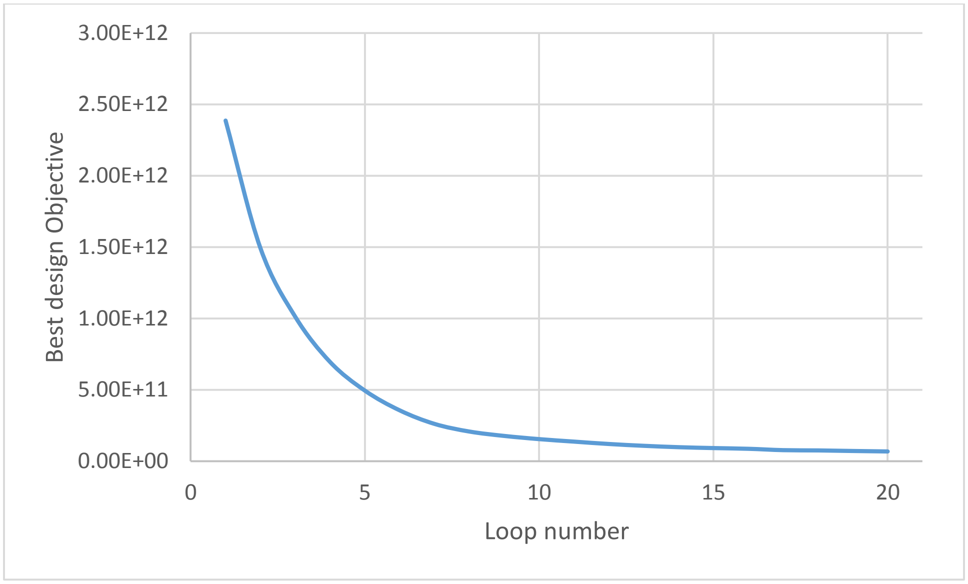

Figure 15. Convergence of truss tower by loops

Figure 16 showed the iteration details of optimization. In this study, the penalty constant is relatively large, so in this plot, the objective and fitness of the best design are very similar values. It means that most searches are done in the feasible domain.

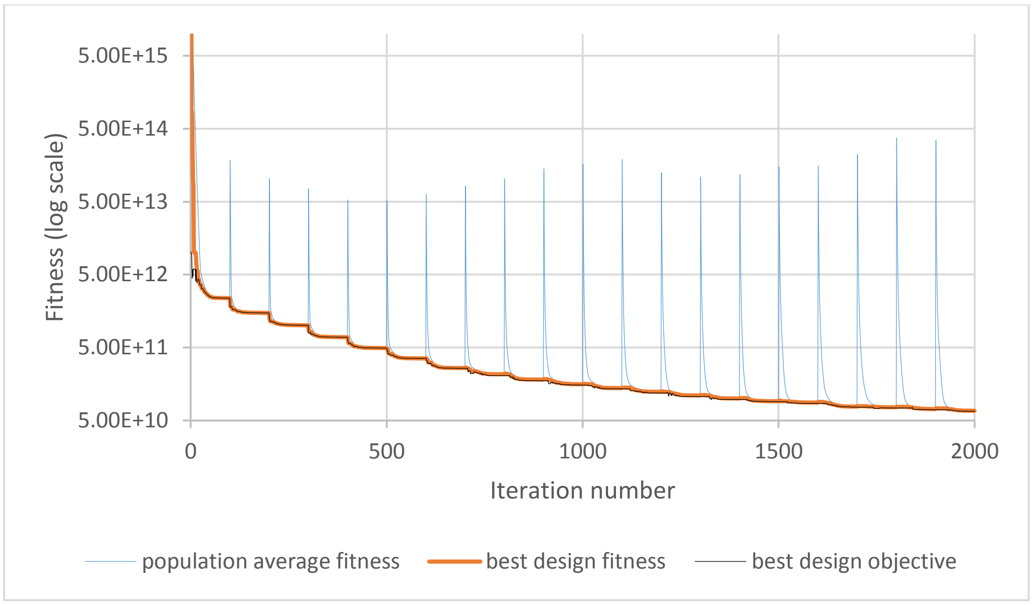

Figure 16. Convergence of truss tower by iterations

When one new loop starts, the average fitness in the population is enlarged a lot. So perhaps the rebuilding gave the new population some good variables but these good variables should be recombined for the best design. It is the mutation and crossover to make the good variables combine together and perhaps, to explore for the new variables to make the fitness smaller.

The loop 5 convergence plot, in figure 17 makes the argument above clearer. After the first iteration, the crossover and mutation help to improve the objective by 10%. The effective improvement lasts for 40 iterations. After that, the calculation is not effective and the population could tend to be converged.

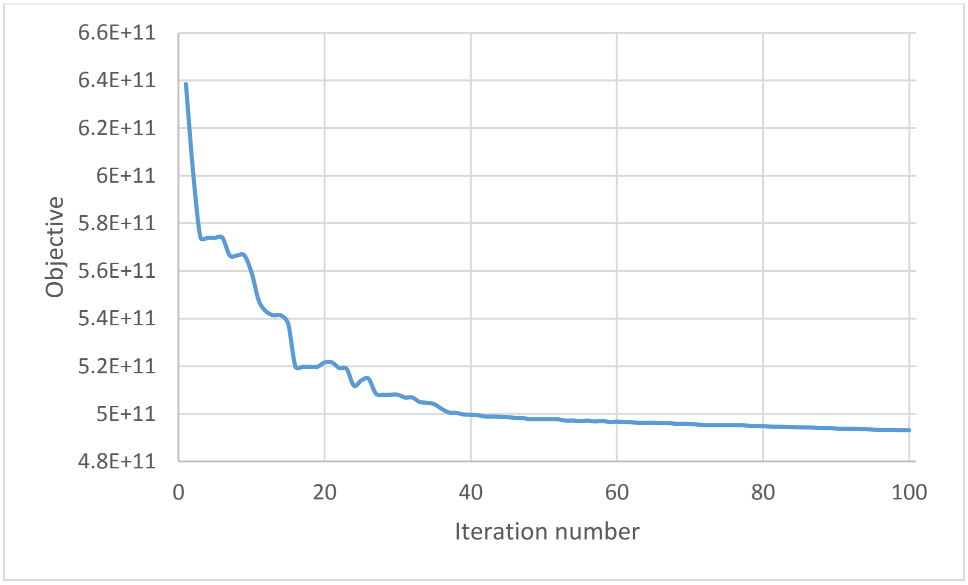

Figure 17. Objective convergence in loop 5

So from the convergence plot, the new genetic algorithm, with rebuilding, and the new type of crossover and mutation is able to optimize the complex truss model.

4.4.2 The structural plot

In this section, the tower shape plots are discussed. Figure 18 showed the best designs of loop 5, 10, 15, 20. Red bars are overstressed with $R_s \geq 0.85$; green bars have the stress ratio as $R_s = 0.5$. So if the member color has some red and green the stress is in good domain. If $R_s \leq 0.2$, the member is colored by blue. If the member color is some blue and green, the member size is wasted.

From the view of stress ratio basically all the designs are feasible and the design of loop 20 saves the largest amount of material. Because the columns as the largest weight contributors are all in the good stress domain. Perhaps some diagonal members are in blue, as the stress ratio is under 0.2. But their material cannot be saved because the other members in the same member group have a high stress ratio. The stress of loop 20 design, in table 11 can prove it.

Secondly, the tower shape are improved with iterations. The designs of loop 5 and loop 10 have very short columns.

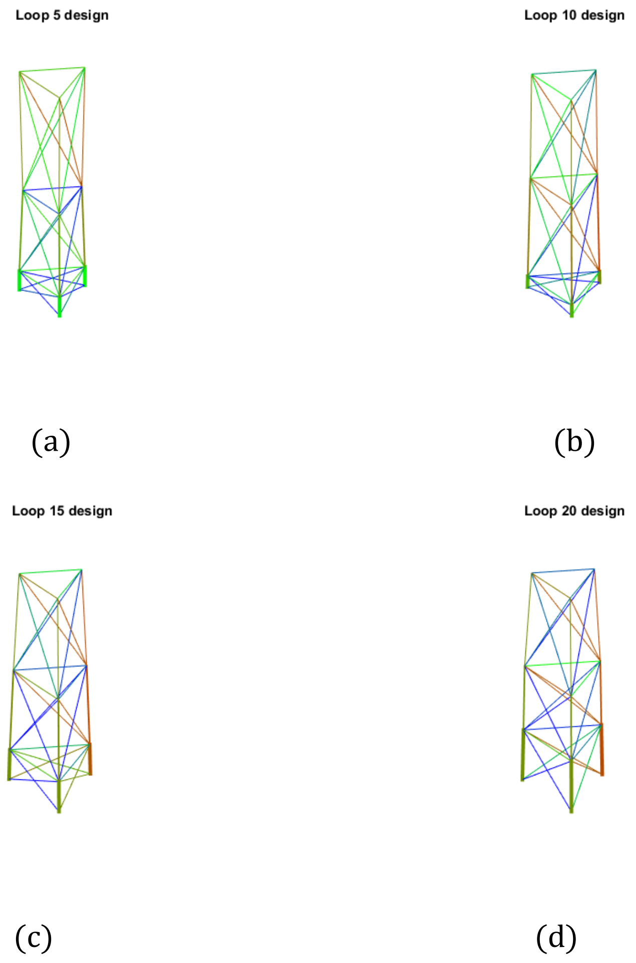

Figure 18. Truss tower design by loops; (a) is loop 5 design; (b) is loop 10 design; (c) is loop 15 design; (d) is loop 20 and final design

It is not a good idea because one short column means that the tower need another very long column to reach the height. For the long column, thus, the allowable stress will decrease by a large amount, by the Rankine Gordon formula.

The reason why the program preferred the short first level, could be to make a better objective. As the penalty constants are large enough, most of designs have no violations of the constraints. So the program tended to find the best way to cut the weight, the objective. Thus, when decreasing the first level height, the members with largest weight are shortened. But as the program goes, the short level designs will die out because of the local stability.

Table 11

Stress Ratio by member group

Member Group	Position	Stress Ratio
1	level 1 blue	0.716228832
2	level 1 yellow	0.702230164
3	level 1 black	0.688775323
4	level 1 red	0.535007048
5	level 2 blue	0.714533597
6	level 2 yellow	0.701560233
7	level 2 black	0.714533597
8	level 2 red	0.702699349
9	level 3 blue	0.714825532
10	level 3 yellow	0.70569566
11	level 3 black	0.703717178
12	level 3 red	0.640835472

4.4.3 Conclusions

Basically, the new genetic algorithm can optimize the complex truss tower from a random initial population. The rebuilding and the new formation of crossover and mutation can play the role as designed.

The optimization result is reasonable, from the shape and stress point of view. Because most of members, including all the columns are on the edge of the constraints and the tower shape is good for horizontal members to stay in the tensile condition. In this way, their allowable stress will not decrease.

CHAPTER 5

CONLUSIONS AND FUTURE WORK

5.1 Conclusions

In this paper, the new genetic algorithm is described and discussed. The formation of this genetic algorithm can present the basic idea: the better design found from the neighbor of the good designs. Moreover, the operations of genetic algorithm including the rebuilding, mutation and crossover are straightforward enough for others to build the new genetic algorithm

Some examples of two variable function optimization problems are tested for the new genetic algorithm, which proves that the algorithm can do the simple function optimization problems and can find the optimal solution with a bad start.

The simple truss model is used for testing the new algorithm as well. With most of results, the new algorithm can be proved to find the same optimized designs with other algorithms. Although there is one problem with different results, the results from the other algorithms could not be correct. For the speed to find the optimized result, the new genetic could provide an acceptable one, although most of the previous researches hardly had enough information to compare.

One new model, the tall tower, is discussed in the last part of this study. This model is complex as supposed to test the genetic algorithm. Additionally, as a sizing and shape optimization problem, this model could cover all of needs for testing the performance of optimization algorithm doing the truss model.

So in conclusion, the new genetic algorithm can do the optimization for the most of bench mark models from the previous paper in the effective way.

However, there is a lot of work to do, for this genetic algorithm or for the every kind of genetic algorithm for structural optimization problems.

5.2 Future work

In fact, to do structural optimization research, the genetic algorithm is not a popular tool. [31] described a lot about the advantages that a gradient based optimization has over a genetic algorithm.

Basically, the fatal problem of the genetic algorithm is not able to find the optimized results in an acceptable speed. This problem becomes more serious when considering a 2D topology optimization problem. So the future work is to improve the speed of genetic algorithm to find the optimized.

Based on the experiences of this study, there are two points to improve the effectiveness of the genetic algorithm, which is the size of population and the usage of the genetic algorithm flexibility. These two points are not studied deeply for genetic algorithm and perhaps are necessary to understand thoroughly.

First, the size of population is extremely important for the effectiveness of the genetic algorithm. In genetic algorithm, the individuals are picked randomly so the population size should be large enough to make sure the computation stable. In other words, under the same conditions, the algorithm should produce the similar results despite the noise, which means the population controls the randomness for the whole computation. It could become the criterion of population size.

Although lacking of researches on this topic in genetic algorithm, right now the population size could be calculated conservatively.

A 2D topology problem is built to discuss about the influence of these two improvement ways. Generally, all the elements at the left are fixed and other three sides are free. There is one point force at coordinate (30, 0), which is -10000. The 20×20 model has 400 elements will be optimized with the thickness of every elements, so in total, this is a sizing (topology) problem with 400 variables, figure 19.

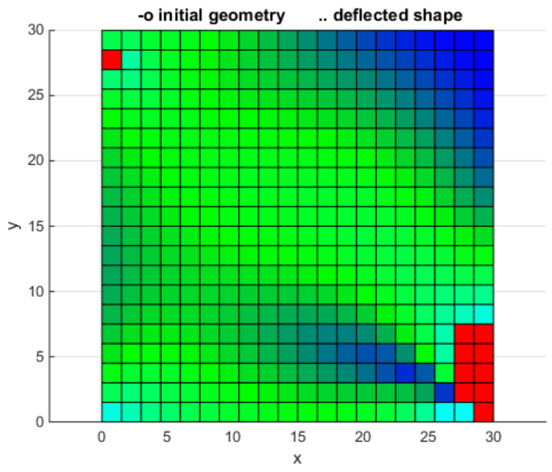

Figure 19. The initial design with stress ratio plot

For the new genetic algorithm if the value number is 400, the population size is about the square of value number, 160000. As an algorithm with $N_l = 20$ and $N_i = 50$, the total number of model evaluation is about $160{,}000{,}000$. It is impossible to use the new genetic algorithm to do the 2D topology optimization. However, if the population size is 400 other than 160000, the calculation speed will increase dramatically.

In the previous study, the rebuilding, crossover and mutation is based on randomness and has the same amount and probability to increase or decrease. However, in this study, to make the program more effective, some information from the best design from the previous loop or iteration is used, for example, the stress of every element.

In this study, the stress ratio is used in the rebuilding. The new member thickness is calculated by:

$$a_{n+1}^{m} = a_{n}^{m} \cdot c_s \cdot \sqrt{\tau_m} \tag{5.1}$$

Where, a_{n+1}^{m} is the new thickness of member M; $a_{n}^{m}$is the old thickness of member M. c_s is the multiplier from a number group: [0.8,0.85,0.9,0.95,1.0,1.05,1.1,1.15,1.2] σ_m is the old member stress ratio, which can be calculated by:

$$\tau_m = \frac{\sigma_m}{\sigma_a} \tag{5.2}$$

Where, σ_m is the stress of member M and σ_a is the allowable stress.

In this way, the stress ratio will partly control the new variable, to produce more favorable individuals, compared with randomly producing. The result is shown in the figure 20. In the plot, if it is blue, the block has less thickness; if it is green, the block has more thickness. Therefore, the design is reasonable.

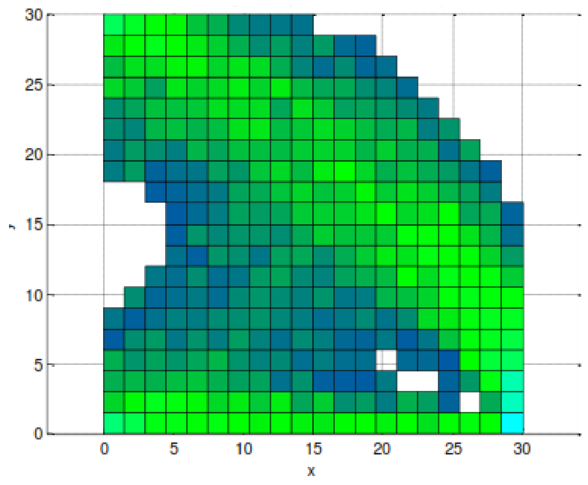

Figure 20. Final result of 2D finite element optimization

The idea of flexibility, using the experiences of the previous designs could be regarded as the good addition for the genetic algorithm.

In fact, the model is not optimized very well. The effective and result could be better, if the two improvement points, like the population size and the flexibility are studied deeply enough. And the genetic algorithm will be used more widely if so.

REFERENCES

[1] J.H. Holland, “*Adaption in natural and artificial systems*” Ann Arbor, MI: University of Michigan Press, 1975

[2] D.E. Goldberg, “*Genetic algorithm in search, optimization and machine learning*” Addison-Wesley, 1989

[3] S. Rajeev, C.S. Krishnamoorthy, “Discrete optimization of structures using genetic algorithm”, *J. Struct. Eng*, 1992.118:1233-1250

[4] S.D. Rajan, “Sizing, Shape, and topology design optimization of truss using genetic algorithm”, *J. Struct. Eng*, 1995.121:1480-1487

[5] C. Camp, S. Pezeshk, G. Cao, “Optimized design of two-dimensional structures using genetic algorithm”, *J. Struct. Eng*, 1998. 124:551-559

[6] R. Kicinger, T. Arciszewski, K.D. Jong, “Evolutionary computation and structural design: a survey of the-state-of-the-art”, *Computer and structures* 2005. 83 1943-1978

[7] S.Y. Chen, S.D. Rajan, “A robust genetic algorithm for structural optimization”, *Structural Engineering & Mechanics Journal*, 2000. Vol 10, No 4: 313-336

[8] K. S. Lee, Z.W. Geem, “A new structural optimization method based on the harmony search algorithm”, *Computers and structures*, 2004. 82: 781-798

[9] K. S. Lee, Z.W. Geem, “A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice Comput”, *Methods Appl. Mech. Engrg*, 2005. 194: 3902-3933

[10] G.C. Luh and C.Y. Lin, “Structural topology optimization using ant colony optimization algorithm”, *Applied soft computing*, 2009. 9:1343-1353.

[11] K. Krishnakumar, “Micro-Genetic Algorithm for Stationary and Non-stationary function optimization”, *Intelligent Control and Adaptive Systems*, 1989. SPIC Vol. 1196

[12] J. Lee, S. Kim, H.S. Park, “Optimum design of cold-formed steel columns by using micro genetic algorithm”, *Thin-Walled Structures* 2006. 952-960

[13] Q.S. Li, X.K. Zou, J.R. Wu and Q. Wang, “Integrated wind-induced response analysis and design optimization of tall steel buildings using Micro-GA”, *Struct. Design Tall Spec. Build,* 2011. 20: 951-971

[14] Y.D. Kwom, S.B. Kwon, S.B. Jin, J.Y. Kim, “Convergence enhanced genetic algorithm with successive zooming method for solving continuous optimization problems”, *Computers and Structures,* 2003, 81: 1715-1725

[15] L.A. Schmit, B. Farshi, “Some approximation concepts for structural synthesis”, *AIAA J* 1974. 12(5): 692-699

[16] L.A. Schmit and H. Miura, “Approximation concepts for efficient structural synthesis” *NASA,* 1976. CR-2552

[17] V.B. Venkayya, “Design of optimum structures”, *Comput. & Structures,* 1971.1(1-2): 265-309

[18] M.W. Dobbs, R.B. Nelson, “Application of optimality criteria to automated structural design”, *AIAA J,* 1976.14(10): 1436-1443

[19] P. Rizzi, “Optimization of multi-constrained structures based on optimality criteria”, *AIAA/ASME/SAE 17th structures, structural dynamics, and materials conference,* 1976

[20] M.R. Khan, K.D. Willmert, “An optimality criterion method for large-scale structures”, *AIAA J* 1979.17(7):753-61

[21] N.S. Khot and L. Berke, “Structural optimization using optimality criteria methods”, *New directions in optimum structural design,* 1984

[22] A.B. Templeman and S.K. Winterbottom, “Structural design by geometric programming”, *AGARD conference,* preprint-123, Milan; 1973

[23] N.H. Chao, S.J. Fences and A.W. Westerberg, “Application of reduced quadratic programming technique to optimal structural design”, *New directions in optimum structural design,* New York, 1984

[24] H. Adeli and O. Kamal, “Efficient optimization of space truss”, *Comput & Structures,* 1986. 24(3): 501-11

[25] K.V. John, C.V. Ramakrishnan, K.G. Sharma, “Minimum weight design of truss using improved move limit method of sequential linear programming”, *Comput & Structures,* 1987;27(5): 583-591

[26] M.P. Saka, “Optimum design of pin-jointed steel structures with practical applications”, *J Struct Eng,* ASCE 1990. 116(10):2599-2620

[27] G.M. Fadel, S. Clitalay, “Automatic evaluation of move-limits in structural optimization”, *Struct Optimization* 1993. 6:233-237

[28] N. Stander, J.A. Snyman, J.E. Coster, “On the robustness and efficiency of the S.A.M algorithm for structural optimization”, *Int J Numer Methods Eng,* 1995. 38: 119-135

[29] S. Xu, R.V. Grandhi, “Effective two-point function approximation for design optimization”, *AIAA J,* 1998. 36(12): 2269-2275

[30] L. Lamberti, Pappalettere, “Comparison of the numerical efficiency of different sequential linear programming based on algorithms for structural optimization problems”, *Comput & Structures,* 2000. 76:713-728

[31] O. Sigmund, “On the usefulness of non-gradient approaches in topology optimization”, *Struct Multidisc Optim* 2011. 43:589-596