ISSN 1392 - 1207. MECHANIKA. 2007. Nr.2(64)

Use of genetic algorithms in topology optimization of truss structures

D. Šešok*, R. Belevičius**
**Vilnius Gediminas Technical University, Saulėtekio al. 11, 10223 Vilnius, Lithuania, E-mail: dsesok@one.lt*
***Vilnius Gediminas Technical University, Saulėtekio al. 11, 10223 Vilnius, Lithuania, E-mail: rb@fm.vtu.lt*

1. Introduction

Topology optimization of truss or frame systems is relevant and together complex technical problem, because the objective function is nonlinear and nonconvex.

The algorithms of variant calculations for the problems of practical size are not capable due to huge amount of calculations. For example, for truss structure consisting of 4 nodes (1+3)*0.5*3 = 6 different connection variants are possible. For more realistic problem, suppose having 100 nodes, there are (1+99)*0.5*99 = 4950 possible connections or about $1.25*10^{1490}$ different truss structure patterns. This is not acceptable for the design of optimal structure, because besides topology optimization the shape and sizing optimization is required here, i.e. the topology optimization procedure will be repeated several times.

The structure topology may be naturally described using discrete variables, therefore here the genetic algorithms are widely used [1, 2]. However, customizing and implementing genetic algorithm for the optimization of mechanical structure requires a lot of efforts of programmer [3].

Presently researchers have at their disposal a number of finite element packages (for example, ANSYS, ALGOR), which allow obtaining all mechanical characteristics of the structure: nodal displacements, stresses, reactive forces at supports, etc. The use of these standard packages in optimization program would considerably reduce the programming costs: now only the genetic algorithm has to be implemented and connected to the finite element package. One of the possible technologies how to link genetic algorithm with the package ANSYS in geometry optimization of frame structures is shown in [4]. The aim of this paper is twofold: first, to show the original implementation of genetic algorithm for topology optimization of truss or frame systems, and second, to show the technology of connection of genetic algorithm and standard finite element package ANSYS.

In the first part of the paper we define the optimization problem and describe it in terms of genetic algorithms. Later on the technology, which allows integrating the advantages of low-level programming language C++ and advantages of finite element package ANSYS thus reducing the programming costs, is described.

2. Problem formulation

As the objective function for topology optimization total mass of the structure is taken

$$M = \sum_{e=1}^{n} L_e \rho_e A_e \tag{1}$$

where L_e is the length of eth element, ρ_e is density, and A_e is cross-sectional area of the same eth element.

The most important property of any structure, e.g. a truss or other is its stability. Therefore the first set of constraints on the problem consists of nodal forces balance equation for all nodes of the structure

$$\sum_{j=1}^{k} \vec{F}_{ij} = 0 \tag{2}$$

where $\vec{F}_{ij}$ is jth force at ith node. This set of constraints for all applied external forces, internal forces arising in elements and reactive forces at supports is ensured by statical analysis with ANSYS.

In order to avoid meaningless solutions from engineering point of view, the constraints on maximum stresses in trusses of the structure have to be introduced

$$|\sigma_e| \le \sigma_{max} \tag{3}$$

where $|\sigma_e|$ is the stress in eth truss, and σ_{max} is maximum allowable stress.

We start the topology optimization from "excessive-connected" truss structure, thus covering all possible topologies of the structure. During optimization procedure the superfluous elements have to be excluded. Let the threshold value for this be

$$|\sigma_e| \ge \sigma_{min} \tag{4}$$

where σ_{min} is minimum allowable stress.

The constraints (3) and (4) are merged into one set

$$\sigma_{min} \le |\sigma_e| \le \sigma_{max} \tag{5}$$

3. Genetic algorithm

As the genetic algorithm will be used for the minimization of objective function, let us shortly describe the concept of genetic algorithms suggested by Holland in 1975 [5]. The main ideas of genetic algorithms simulate the ideas of Darwin theory: if a population of animals or plants lives in some ecosystem, a better chances to survive and to leave descendants have the individuals more adapted to the environment. The genes of individuals are influenced by crossover and mutation, and the worst individuals are eliminated during selection.

Genetic algorithms interpret "individual" as the string of bits, representing the design parameters for objective function. Additionally, the individual may contain other information, for example, on his parents. The superiority of individuals is estimated by their objective values.

Schematically the genetic algorithm is shown in Fig. 1. The algorithm is illustrated by the following simple example. Let four individuals form the initial population:
11111111
00000000
10101010
01010101
For instance, during selection for reproduction the sets of individuals 1 - 2, and 3 - 1 are taken. Then the following results of crossover will be obtained (Table 1).

The operator of mutation reverses the value of bit. Thus, if the individual is 11111111 and we know the third bit from the right side has to mutate, resulting individual will be 11111011. The probability of mutation is usually low. Say, if we have a population consisting of 10 individuals of 20 bits each, and the probability of mutation is 0.01, then during mutation only the values of 10*20*0.01=2 bits will be reversed.

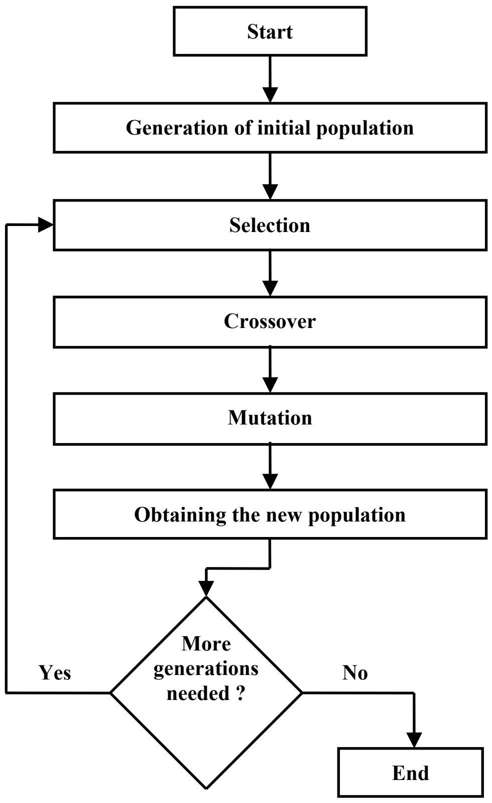
Start
Generation of initial population
Selection
Crossover
Mutation
Obtaining the new population
More generations needed ?
Yes
No
End

Fig. 1 Scheme of genetic algorithm

Table 1

Operation of crossover

Selected Individuals	Position of Crossover	Crossover	Resulting Individuals
11111111	3	11111|111	11111000
00000000	3	00000|000	00000111
10101010	4	1010|1010	10101111
11111111	4	1111|1111	11111010

4. Genetic algorithm for truss structures

Here we explain our coding of truss (or frame) structures for genetic algorithms. Let us have 4 immovable nodes and search for optimal connection of the nodes (Fig. 2).

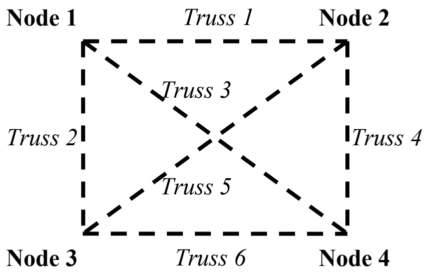

Fig. 2 Possible connections of the nodes

The external loads and supports are given as initial data and therefore, for the sake of simplification, will not be considered here.

As shown in Fig. 2, six possible trusses can connect the 4 nodes. Let us assign for a possible truss code "1" – if the element is presented in the structure, and "0" – if not. Then, after numbering the trusses (Fig. 2), the fully linked structure will be denoted by the following string of bits: 111111. The structures in Fig. 3 are coded 110010 and 011001 accordingly.

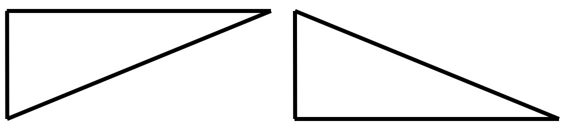

Fig. 3 Examples of truss structures

In terms of genetic algorithms, the string of bits is the chromosome, and every its bit is a gene. The whole truss structure is thus an individual. The essential property of an individual is that it includes also additional information necessary for the algorithm (Table 2); it is similar to one suggested in [6]). The individual data is rendered in C++ terms as the *struct* object (Fig. 4).

Table 2

Structure of individual data

INDIVIDUAL	
Parameter	**Value**
String of bits	Coded design parameter for objective function
Objective	Value of objective function
Parent 1	Number of the 1st parent in previous generation
Parent 2	Number of the 2nd parent in previous generation
Xsite	Position of crossover
Stresses	Stresses in each element (i.e., in gene)

```
Struct individual
{
 int A[lchrom]; // string of 0 and 1 (design parameter)
 double B[lchrom]; // stresses in each element
 double fitness; // value of merit function
 int parent1;
 int parent2;
 int xsite; // position of crossover
};
```

Fig. 4 C++ structure for individual

All the necessary information for the algorithm (total mass of truss structure, stresses in elements) is obtained with finite element package ANSYS; the interface rendering truss structure from individual data is written in C++. The scheme below shows the connection of genetic algorithm in C++ and package ANSYS trough intermediate code in APDL. APDL stands for *ANSYS Parametric Design Language*, a scripting language that can be used to build finite element models in terms of parameters (variables). APDL also encompasses a wide range of other features such as repeating a command, macros, if-then-else branching, do-loops, etc. [7].

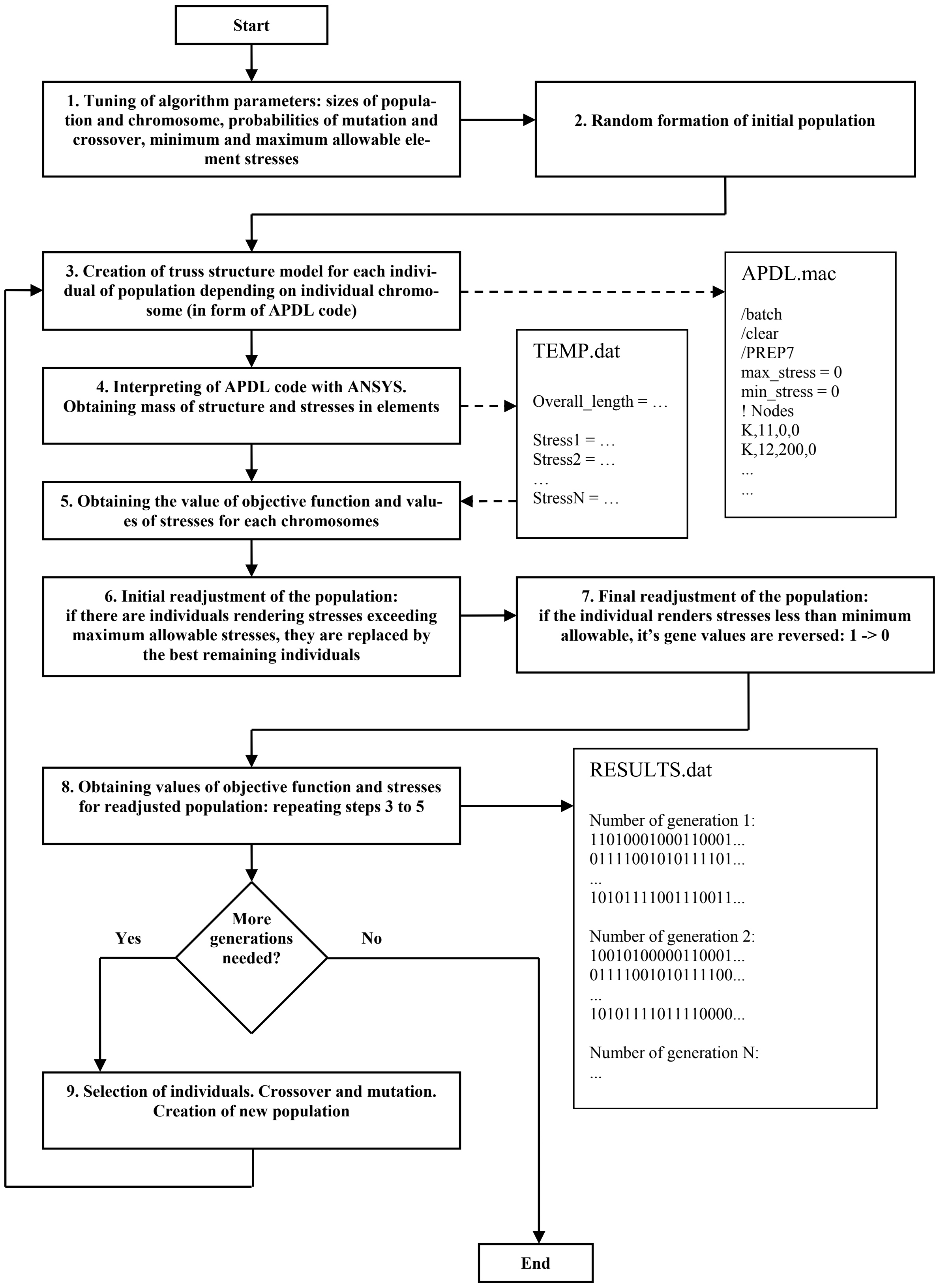
Start
1. Tuning of algorithm parameters: sizes of population and chromosome, probabilities of mutation and crossover, minimum and maximum allowable element stresses
2. Random formation of initial population
3. Creation of truss structure model for each individual of population depending on individual chromosome (in form of APDL code)
APDL.mac
/batch
/clear
/PREP7
max_stress = 0
min_stress = 0
! Nodes
K,11,0,0
K,12,200,0
...
...
4. Interpreting of APDL code with ANSYS. Obtaining mass of structure and stresses in elements
TEMP.dat
Overall_length = …
Stress1 = …
Stress2 = …
…
StressN = …
5. Obtaining the value of objective function and values of stresses for each chromosomes
6. Initial readjustment of the population: if there are individuals rendering stresses exceeding maximum allowable stresses, they are replaced by the best remaining individuals
7. Final readjustment of the population: if the individual renders stresses less than minimum allowable, it's gene values are reversed: 1 -> 0
8. Obtaining values of objective function and stresses for readjusted population: repeating steps 3 to 5
RESULTS.dat
Number of generation 1:
11010001000110001...
01111001010111101...
...
10101111001110011...
Number of generation 2:
10010100000110001...
01111001010111100...
...
10101111011110000...
Number of generation N:
...
More generations needed?
Yes
No
9. Selection of individuals. Crossover and mutation. Creation of new population
End

Fig. 5 Genetic algorithm: interface between C++ and ANSYS

5. Numerical results and discussion

For the sake of transparency of expected results the proposed technology was applied to the cantilever structure possessing 20 immovable nodes, three of them supported. The structure is loaded with concentrated load at the center (Fig. 6). The optimal shape of expected discrete structure is well known.

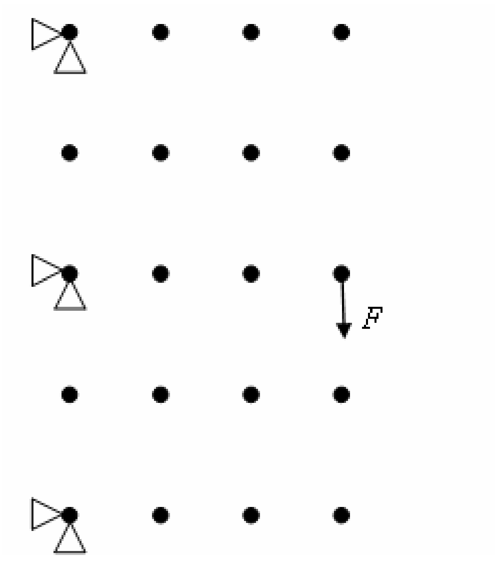

Fig. 6 Initial data of the structure

When the external force F = 280 kN, all trusses have the same cross-section area A = 3200 mm^2 and the same material with Young's modulus E = 200 GPa. Minimum allowable stresses in the elements are 2 N/mm^2, while maximum ones are 40 N/mm^2.

Experimentally tuning the genetic algorithm, the following control parameters were taken: the population size is 21 individuals, the size of each individual is 190 bits (i.e. the number of possible connections between all nodes), and the probability of mutation is 0.01. The individuals of initial population are created randomly, taking values 0 and 1 with equal probability. The number of generations is 10.

The life of each generation consists of 3 stages. In the first stage the current population is obtained from previous generation via selection, crossover and mutation. During the second stage the obtained individuals are analyzed and the best ones are selected for reproduction. If there are individuals with stresses exceeding maximum allowable stress, they are replaced by the best individual of the population. Finally, the elements with stresses below the minimum allowable stress are updated: the value of gene is reversed from 1 to 0.

The best individual of initial population (Fig. 7) has the chromosome
00110010010010000101110000000100000000000000100110
10000000001000000000100010000010100000001100000000
10000100000000000000011011010000000001001100100000
00000000110000101101000010010010000111010010010
01.

The value of objective function is 26316 mm (the total length of all structure elements; it is proportional to total mass of the structure).

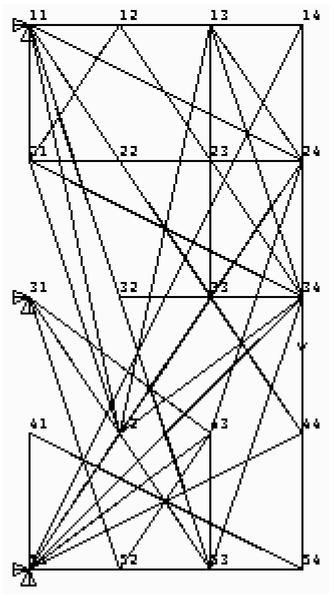

Fig. 7 The best individual of initial population

The best individual of the first generation has the objective value 19405 mm (Fig. 8).

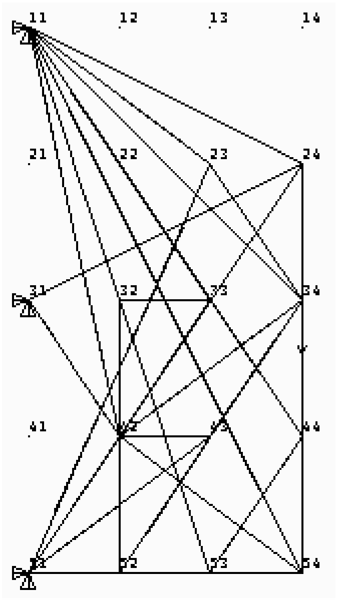

Fig. 8 The best individual of the first generation

During the second run of the algorithm any better solution is found; during the next three runs objective function value diminishes to 13679, 6058 and 5387 mm respectively. This best value remains through the whole 10 generations.

Inert character of objective function value is determined by the following peculiarity of the algorithm: always the best individual of previous generation becomes the first individual of the next generation.

The structures after third and fourth generations are depicted in Fig. 9.

The best individual found has the objective value of 5387 mm and chromosome
0000010000100100000000000000000000000000000000000
0000000000000000000000000000000000000000000000000
0000000001000010000000000000000000000000000000000

0000000000000000101000000000000000000100000000000 00. Only 8 genes of this chromosome have the value 1, i.e. only 8 elements remained in the truss structure (Fig. 10).

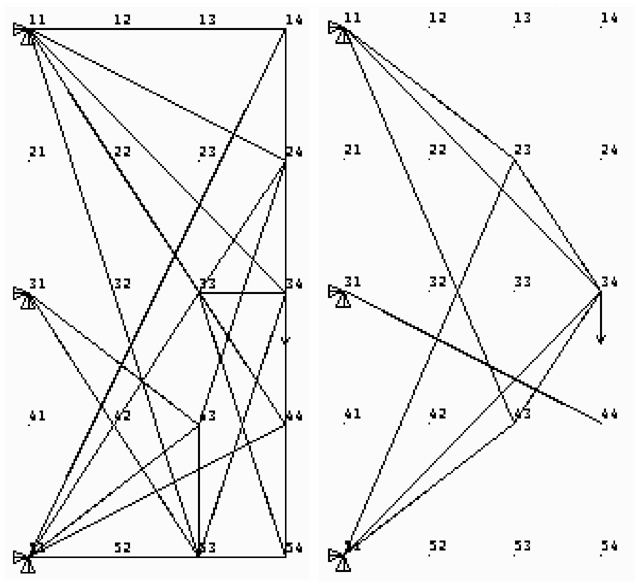

Fig. 9 The best individuals of the third and fourth generations

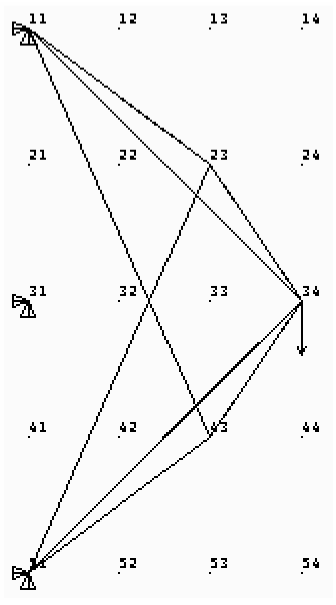

Fig. 10 The best solution

As shown in Table 3, the best individual is found already in 5th generation. Later on, also the average value of objective values of all generation stabilizes: all individuals approach the best solution. All individuals become nearly the same, therefore the crossover does not bring any improvements. Only the mutation operation brings small alterations on average objective value of the whole population. However, the probability of mutation is 0.01: 100 iterations are needed to change a required gene.

Graphically the history of objective value alteration is shown in Fig. 11.

Solution of the same problem with increased maximum allowable stresses (110 N/mm^2) after 8 iterations yields to the objective function value 1697 mm and the chromosome of only two nonzeroes genes (Fig. 12). This result coincides with theoretically known global solution of topology optimization of cantilever structure.

Table 3
Objective values of the best individual and all population

Number of population	Objective of best individual, mm	Average objective of all generation, mm
0	26316	31206
1	19405	25768
2	19405	21048
3	13679	16821
4	6058	13536
5	5387	10534
6	5387	5387
7	5387	5387
8	5387	5411
9	5387	5427
10	5387	5500

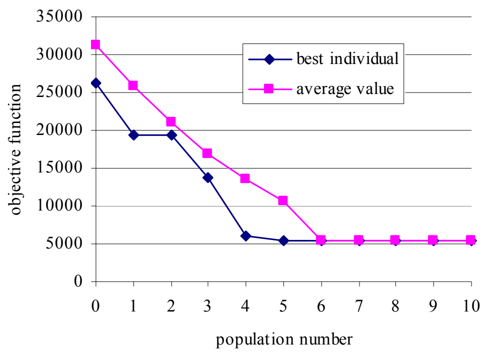

Fig. 11 Optimization history

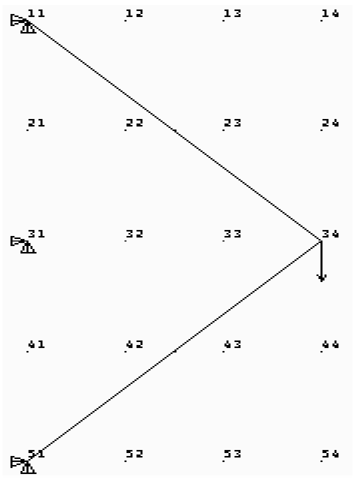

Fig. 12 The best solution with increased maximum allowable stresses

6. Concluding remarks

Genetic algorithms may be easily adapted to the topology optimization of truss and frame structures. Genetic algorithm cannot assure global solution of the problem, however, compared with other global optimizers, it yields to a reasonable solution in short time. For instance, for the given numerical example all the calculation of 10 populations on AMD Athlon 1.09 GHz processor, 1 GB RAM took 20 min, while the best solution is obtained after 12 min. (truss structure has about $1{,}57*10^{57}$ different possible patterns).

Solving the topology optimization problem with the proposed technology, it is necessary only to code the design parameters as string of bits; no additional information on objective function (sensitivity, continuity) is required. Moreover, it greatly reduces the costs of programming, as all the mechanical responses of constructed structure are obtained with commercially available finite element packages.

References

1. **Lingyun, W., Mei, Z., Guangming, W., Guang, M.** Truss optimization on shape and sizing with frequency constraints based on genetic algorithm.-Comput. Mech., 2005, v.35, p.361-368.
2. **Chaman, C., Saitou, K., Jakiela, M.** Genetic algorithms as an approach to configuration and topology design.-In Advances and Design Automation, ASME, 1993, v.65, p.485-498.
3. **Smith, J., Hodgins, J., Oppenheim, I.** Creating models of truss structures with optimization.-ACM Tracsactions on Graphics (SIGGRAPH 2002), 2002, 21(3), p.295-301.
4. **Puiša, R.** The adaptive stochastic algorithms for CAD-based structure optimization of mechanical parts.-Doctoral dissertation, Vilnius, 2005, p.129-144.
5. **Holland, J.H.** Adaptation in Natural and Artificial Systems.-Ann Arbor: The University of Michigan Press, 1975.-211p.
6. **Goldberg, D.** Genetic algorithms in search, optimization and machine learning.-Addison-Wesley, New-York, 1989, p.59-70.
7. ANSYS Release 8.0 Documentation. APDL Programmer's Guide. - http://www.ansys.com/ services/ss-documentation-manuals.asp.

D. Šešok, R. Belevičius

GENETINIŲ ALGORITMŲ TAIKYMAS OPTIMIZUOJANT STRYPINIŲ SISTEMŲ TOPOLOGIJĄ

R e z i u m ė

Straipsnyje aprašyta technologija, leidžianti optimizuoti strypinių sistemų (santvarų arba rėmų) topologiją taikant genetinius algoritmus. Kaip tikslo funkcija imama visa santvaros strypų masė, o apribojimų sistemą sudaro pusiausvyros ir įtempių sąlygos (t. y. reikalaujama, kad įtempiai strypuose neviršytų leistinųjų ir nebūtų mažesni už tam tikrą slenkstinę vertę). Tikslo funkcijos reikšmė apskaičiuojama naudojant ANSYS paketą, o genetinis algoritmas ir programos sąsaja su vartotoju sudaryti C++ kalba. Duomenimis tarp C++ ir ANSYS programų keičiamasi per tarpines rinkmenas. Pateikiama skaitinių pavyzdžių.

D. Šešok, R. Belevičius

USE OF GENETIC ALGORITHMS IN TOPOLOGY OPTIMIZATION OF TRUSS STRUCTURES

S u m m a r y

In this paper, a technology enabling to optimize the topology of truss or frame structures with genetic algorithms is presented. The objective function is total mass of the structure elements. Constrains consist of equilibrium equations and conditions on stresses in the elements: stresses cannot exceed allowable value and must be above some threshold value. The values of the objective function are calculated with package ANSYS. Genetic algorithm with interface is implemented in C++. Exchange of data between ANSYS and C++ goes through temporary files. Numerical examples are presented.

Д. Шешок, Р. Белявичюс

ИСПОЛЬЗОВАНИЕ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ ОПТИМИЗАЦИИ ТОПОЛОГИИ СТЕРЖНЕВЫХ КОНСТРУКЦИЙ

Р е з ю м е

В статье описана технология, позволяющая оптимизировать топологию стержневых конструкций (ферм), используя генетические алгоритмы. За целевую функцию взята общая масса стержней фермы, а граничные условия включают в себя требования равновесия и ограничения напряжений (т. е. требуется, чтобы напряжения в стержнях фермы не превышали допустимых и не были бы меньше определенного значения). Значение целевой функции вычисляется с помощью пакета ANSYS, а генетический алгоритм и диалог программы с пользователем реализованы на языке C++. Обмен данными между C++ и ANSYS программами осуществляется посредством промежуточного файла. В статье также приведены численные примеры.

Received January 10, 2007