

A COMBINATORIAL ALGORITHM FOR THE DISCRETE OPTIMIZATION OF STRUCTURES¹

Chai Shan (柴 山)² Sun Huanchun (孙焕纯)²

(Received Nov. 28, 1996)

Abstract

The definition of local optimum solution of the discrete optimization is first given, and then a comprehensive combinatorial algorithm is proposed in this paper. Two-level optimum method is used in the algorithm. In the first level optimization, an approximate local optimum solution \bar{X} is found by using the heuristic algorithm, relative difference quotient algorithm, with high computational efficiency and high performance demonstrated by the performance test of random samples. In the second level, a mathematical model of $(-1, 0, 1)$ programming is established first, and then it is changed into $(0, 1)$ programming model. The local optimum solution X^ will be from the $(0, 1)$ programming by using the delimitative and combinatorial algorithm or the relative difference quotient algorithm. By this algorithm, the local optimum solution can be obtained certainly, and a method is provided to judge whether or not the approximate optimum solution obtained by heuristic algorithm is an optimum solution. The above comprehensive combinatorial algorithm has higher computational efficiency.*

Key words discrete variables, structural optimization, combinatorial optimization, local optimum solution

I. Introduction

A majority of the structural optimum problems are discrete ones in engineering practice. Because of the basic reason, namely the discontinuity of design variables, the discrete optimization has two prominent difficulties:

(1) The models of discrete optimizations are nonconvex programmings, and we can not use the analytical mathematics but must use combinatorial mathematics to solve them. Because they belong to the so-called the NP hard problem of combinatorial optimization, it is very difficult to solve large scale problems. There are three kinds of combinatorial algorithm in common use:

① Accurate algorithm. This kind of algorithm can find the global optimum solution of problems, but its computational time is generally an exponential function of design variables. The evaluating standard for this kind of algorithm is its computational efficiency.

② Approximate algorithm. This kind of algorithm finds an approximate optimum

¹ Project supported by Natural Science Foundation of Shandong Province

² Dalian University of Technology, Dalian 116023, P. R. China

solution instead of an optimum solution, but this kind of algorithm can ensure that the relative error between the approximate optimum solution and the optimum solution does not exceed a definite ratio.

③ Heuristic algorithm. The basic idea of this kind of algorithms is not to find the optimum solution but to find an approximate solution in the permissible time. In general, this kind of algorithms is proposed on the base of studying the problems thoroughly, so it is called heuristic algorithm. The evaluating standard for the heuristic algorithm is the approximate degree of the approximate solution to the optimum solution.

In the above three kinds of algorithms, the accurate algorithm can find the optimum solution, but it is only suitable to the small scale problems as that was pointed out in reference [1]. In practice, the commonly used algorithms are heuristic algorithms proposed based on the characters of problems to be solved.

(2) The Kuhn-Tucker condition, local optimum condition in continuous optimization, is not suitable to discrete optimization, so people have no method to judge whether or not the solution obtained by heuristic algorithm is a local optimum solution. This is an unsolved problem in discrete optimization.

In order to overcome the above two difficulties, we first give the definition of local optimum solution in discrete optimization, and then propose a comprehensive combinatorial algorithm. The algorithm uses two-levels optimum method. The first level optimization finds an approximate local optimum solution \bar{X} by using the heuristic algorithm, relative difference quotient algorithm, with high computational efficiency and high performance demonstrated by the performance test of random samples. The second level establishes the mathematical model of $(-1, 0, 1)$ programming and then changes it into $(0, 1)$ programming model. The $(0, 1)$ programming model will be solved to find the local optimum solution \bar{X}^* by using delimitative and combinatorial algorithm [3] or relative difference quotient algorithm. If $\bar{X} \neq \bar{X}^*$, do iteration until $\bar{X} = \bar{X}^*$. Using this algorithm, we can obtain the local optimum solution certainly, and provide a method to judge whether or not the approximate optimum solution obtained by heuristic algorithm is an optimum solution. The above comprehensive combinatorial algorithm has higher computational efficiency.

II. Some Definitions

For the convenience of the following discussion, some definitions are given in this section.

Definition 1 The discrete near set of a point The discrete near points of point X are the neighbour discrete points of X in every coordinate direction. The set of them is:

$$S_e(X) = \bigcup_{i=1}^n \{P_i, Q_i\} \quad (2.1)$$

where P_i, Q_i are the lower and upper near points of point X along the i th coordinate respectively. $S_e(X)$ can be divided into two subsets, the subset of lower coordinate near points and that of the upper ones.

$$\left. \begin{aligned} S_{el}(X) &= \bigcup_{i=1}^n \{P_i\} \\ S_{eu}(X) &= \bigcup_{i=1}^n \{Q_i\} \end{aligned} \right\} \quad (2.2)$$

Define

$$E_i = \{x_i^{P_i}, x_i^X, x_i^{Q_i}\} \quad (2.3)$$

where $x_i^{P_i}, x_i^X, x_i^{Q_i}$ is the i th component of P_i, X, Q_i respectively.

Definition 2 The discrete neighbour set of a point The neighbour points of point X is defined as a set $S_u(X)$, which includes all the cross points of the discrete coordinate lines (planes or hyperplanes) of $\{x_i^{P_i}, x_i^X, x_i^{Q_i}, i=1, 2, \dots, n\}$ except the point X . Namely

$$S_u(X) = E_1 \times E_2 \times \dots \times E_n = \prod_{i=1}^n E_i \quad (2.4)$$

where $E_i \times E_j$ denotes the Descartes product E_i and E_j .

The geometrical relations between discrete near points and discrete neighbour points in the discrete space of two dimensions are shown in Fig. 2.

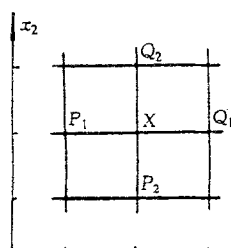


Fig. 1 The lower and upper near points of X

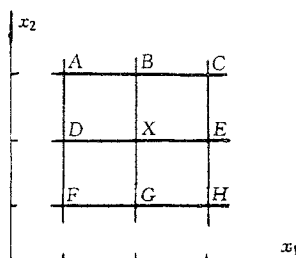


Fig. 2 Discrete near set and neighbour set of X

where

$$\begin{aligned} S_u(X) &= \{A, B, C, D, E, F, G, H\} & S_o(X) &= \{B, E, G, D\} \\ S_{ol}(X) &= \{D, G\}, & S_{eu}(X) &= \{B, E\} \end{aligned}$$

In general cases, if the number of design variables is n , based on the combinatorial theory, the combinatorial numbers are $N=3^n-1$ in discrete neighbour set $S_u(X)$ and $N=2n$ in discrete near set $S_o(X)$ respectively.

In discrete optimization, the constraint conditions are generally inequalities, namely

$$g_j(X) \leq 0 \quad (j=1, 2, \dots, m) \quad (2.5)$$

where, m is the number of constraint conditions of inequality, and $g_j(X)$ can be any arbitrary function.

Definition 3 The discrete feasible set In the set of discrete points, the set S_f of discrete points satisfying the constraint conditions is called the discrete feasible set

$$S_f = \{X | g_j(X) \leq 0, j=1, 2, \dots, m\} \quad (2.6)$$

Definition 4 Local pseudo optimum solution If $X^* \in S_f(X)$ and for all $X \in S_o(X^*) \cap S_f$, inequality (2.7) is tenable,

$$f(X^*) \leq f(X) \quad (2.7)$$

Then X^* is called a local pseudo optimum solution.

Definition 5 Local optimum solution If $X^* \in S_f$, and for all $X \in S_u(X^*) \cap S_f$, inequality (2.7) is tenable, then X^* is called a local optimum solution.

Definition 6 Global optimum solution If $X^* \in S_f$, and for all $X \in S_f$, inequality (2.7) is tenable, then X^* is called a global optimum solution. Namely, the local pseudo-optimum solution is the optimum solution in the set of discrete near neighbour points, the local optimum solution is that in the set of discrete neighbour points, and the global optimum solution is that in the set of all feasible points.

III. $(-1, 0, 1)$ Programming Model of Discrete Optimization of Structures

3.1 The mathematical model of discrete optimization of structures

The mathematical model of discrete optimization of structures subjected to the stress and displacement constraints is as follows:

$$\begin{aligned} \min \quad & W = \sum_{i=1}^n \rho_i l_i A_i \\ \text{s.t.} \quad & \sigma_{il} \leq [\sigma]_i \quad (l=1, 2, \dots, NL) \\ & \delta_{jl} \leq \bar{\delta}_j \quad (j=1, 2, \dots, NJ) \\ & x_i \in S_i \quad (S_i \in S) \end{aligned} \quad (3.1)$$

where, x_i is a design variable, and it is a set of geometrical properties of cross-section, $x_i = \{A_i, I_{yi}, W_{yi}, R_{yi}, I_{zi}, W_{zi}, R_{zi}, J_i\}$, S_i is allowable set of x_i and S is the set of S_i .

In the above model, stress constraints are local ones and displacement constraints are global ones. Two-levels algorithm is used to solve the above model. The first level deals with the local constraints and its mathematical model is as follows:

$$\begin{aligned} \min \quad & W = \sum_{i=1}^m \rho_i A_i \sum_{k \in G_i} l_k \\ \text{s.t.} \quad & \sigma_{il} \leq [\sigma]_i \quad (l=1, 2, \dots, NL) \\ & x_i \in S_i \quad (S_i \in S) \end{aligned} \quad (3.2)$$

where, i is the ordinal number of design variables after variable-linking, m is the number of design variables, and G_i is the set of design variables which have the same design variable x_i .

Use one dimensional search method to solve the above model and obtain the optimum solution $X_0^* = \{x_1^*, x_2^*, \dots, x_m^*\}^T$.

The second level deals with the global constraints. Take X_0^* as the lower bounds of discrete variables set and modify S_i and S to new sets S'_i and S' respectively. Because of the monotonic property of the stress constraints, the stress constraints can not be violated by increasing the sectional area. Therefore in the second level optimization, the stress constraints can be canceled. The mathematical model is as follows:

$$\min \quad W = \sum_{i=1}^m \rho_i A_i \sum_{k \in G_i} l_k$$

$$\begin{aligned} \text{s.t. } \delta_{jl} &\leq \bar{\delta}_j & (j=1, 2, \dots, NJ) \\ x_i &\in S'_i, & S'_i \in S' \end{aligned} \quad (3.3)$$

where

$$\delta_{jl} = \sum_{i=1}^m \sum_{k \in G_i} \delta_{kjl} \quad (3.4)$$

where, j is the ordinal number of displacement constraints, l is the ordinal number of load cases, and δ_{kjl} is the contribution of element k to the j th displacement under l th load case.

3.2 The $(-1, 0, 1)$ programming model of discrete optimization of structures

Assuming a known design point \bar{X} and according to the definition 4, if \bar{X} is an optimum solution in its discrete neighbour set $S_u(X)$, \bar{X} is a local optimum solution. If each corresponding design variable of \bar{X} is $\bar{x}_i (i=1, 2, \dots, m)$ and if P_i, Q_i are the lower and upper near points of point \bar{X} respectively, the $(-1, 0, 1)$ programming model of the discrete structural optimization is as follows:

$$\begin{aligned} \min \quad W &= \sum_{i=1}^m \rho_i A_i \sum_{k \in G_i} l_k \\ \text{s.t. } \delta_{jl} &\leq \bar{\delta}_j & (j=1, 2, \dots, NJ) \\ x_i &\in \bar{S}_i, & \bar{S}_i \in \bar{S}_u \end{aligned} \quad (3.5)$$

where, for each design variable $x_i \in \bar{S}_i = \{x_i^{P_i}, \bar{x}_i, x_i^{Q_i}\}$, three values can be taken. Let

$$x_i = \begin{cases} \bar{x}_i + x'_i (\bar{x}_i - x_i^{P_i}) & (x'_i \leq 0) \\ \bar{x}_i + x'_i (x_i^{Q_i} - \bar{x}_i) & (x'_i > 0) \end{cases} \quad x'_i \in \{-1, 0, 1\} \quad (3.6)$$

The mathematical model formula (3.5) can be changed into $(-1, 0, 1)$ programming model of design variable x'_i

3.3 Change $(-1, 0, 1)$ programming model into $(0, 1)$ programming model

For the convenience of computation, the mathematical model of (3.5) is changed into $(0, 1)$ programming model. Let

$$\left. \begin{aligned} x_i &= x_i^{P_i} + y_{i1} (\bar{x}_i - x_i^{P_i}) + y_{i2} (x_i^{Q_i} - x_i^{P_i}) \\ y_{i1} &\in (0, 1), y_{i2} \in (0, 1), y_{i1} + y_{i2} \leq 1 \end{aligned} \right\} \quad (3.7)$$

By expression (3.7), x_i can be represented as the linear function of y_{i1} and y_{i2} , and the number of design variables is increased from m to $2m$. While variable x_i in (3.5) is substituted by (3.6) and the ordinal numbers of design variables are renumbered, the $(-1, 0, 1)$ programming model can be changed into the following $(0, 1)$ programming model:

$$\begin{aligned} \min \quad W &= \sum_{r=1}^{2m} \rho_r A_r \sum_{k \in G_i} l_k \\ \text{s.t. } \delta_{jl} &\leq \bar{\delta}_j & (j=1, 2, \dots, NJ; l=1, 2, \dots, NL) \\ y_r &\in (0, 1) \end{aligned} \quad (3.8)$$

In formula (3.8), the renumbered design variable y_r must satisfy the relevant original constraint $y_{i1} + y_{i2} \leq 1$. Formally, formula (9.8) has $2^{2m} = 4^m$ combinations, but it only has 3^m combinations in practice because of the constraints $y_{i1} + y_{i2} \leq 1$.

IV. Computational Method

From the above analysis, we can see that there are two key problems in solving the mathematical model (3.1). One is to find an approximate local optimum solution \bar{X} , and the other is to search the optimum solution in discrete neighbour set $S_u(\bar{X})$. If \bar{X} is the optimum solution in $S_u(\bar{X})$, \bar{X} is a local optimum solution, or we must do iterations of optimization. The following methods are used to solve the above two key problems.

4.1 The relative difference quotient algorithm for finding the approximate local optimum solution

Reference [2] has proposed a heuristic algorithm, relative difference quotient algorithm, for a kind of discrete optimization problems. The algorithm has the advantages of higher computational efficiency, less iterative number of times and stable convergence. In order to test its precision, a random numerical test has been done by using some stochastic samples generated in a certain probability distribution. The results of numerical test with 100 random test samples of mathematical model (3.3) are that the average relative error of the approximate local optimum solution to the local optimum solution is 0.3%, maximum relative error of approximate local optimum solution to local optimum solution is 3.8%, and approximate local optimum solutions of 85% samples are equal to the corresponding local optimum solutions respectively. The results of test show that the relative difference quotient algorithm has a higher computational precision.

Firstly, change the constraint conditions of formula (3.3) into

$$\Delta_k = \bar{\delta}_j - \delta_{jl} \geq 0 \quad (k=1, 2, \dots, NJ \times NL) \quad (4.1)$$

and then divide the constraint conditions of formula (4.1) into two subsets. $G_1 = \{\Delta_k | \Delta_k < 0\}$ and $G_2 = \{\Delta_k | \Delta_k \geq 0\}$, where G_1 is a active subset and G_2 an inactive one. G_1 can be seen as a real vector defined in $R^{k_1} (k_1 \leq NJ \times NL)$, and take the 2 norm of G_1 as the united constraint function

$$Z = \|G_1\|_2 = \left(\sum_{k \in G_1} \Delta_k^2 \right)^{1/2} \quad (4.2)$$

Define the relative difference quotient as

$$\beta_i = \frac{\Delta Z / \Delta x_i}{\Delta W / \Delta x_i} = \frac{\Delta Z}{\Delta W} \quad (4.3)$$

where ΔZ and ΔW denote the forward differences of united constraint function and objective function respectively. The algorithm starts from the minimum point of objective function outside the feasible set and advances along the direction of minimum increment of objective function and maximum decrement of the united constraint function (the direction of minimum relative difference quotient) to find a better approximate optimum solution. Reference [2] has given the theory and computation method of the algorithm in detail.

4.2 The algorithm for solving (0, 1) programming

Reference [3] proposed a delimitative and combinatorial algorithm for a kind of (0, 1) programming. The basic idea of the algorithm is as follows. First, rearrange design variables

according to the decreasing order of the magnitude of coefficients of constrain function, find the least number r_0 of design variables which are not equal to 0 in the total feasible combinations, and take r_0 as the lower bound of combinatorial search. Because of the constraint of $y_{i1} + y_{i2} \leq 1$, the upper bound of combinatorial search is m , so that the number of

combinations has decreased to $\sum_{r=r_0}^m C(2m, r)$. Then, rearrange design variables according to the increasing order of the magnitude of coefficients of objective function, and determine the upper search bound according to the coefficients of objective function Such that the number of combinations to be searched can be decreased. The algorithm has a higher computational efficiency and can be used to solve middle scale problems.

To solve large scale problems, a relative difference quotient algorithm can be used. The basic idea of the algorithm is as the same as that in section 4.1 only the allowable set of each variable is $\{0, 1\}$. The number of combinations to be searched is not more than the number of design variables. Most important is that the algorithm can give error estimation, and when the error exceeds the allowable value, the algorithm can do modifications too. In general, a higher precision can be obtained by 0 order modification, or by 1 order modification at most. The computational efforts of 1 order modifications is $O(n)$.

4.3 The computational steps of combinatorial algorithm

We have discussed the original mathematical model of the discrete structural optimization, the $(-1, 0, 1)$ programming model and the corresponding $(0, 1)$ programming model, and have given some algorithms to solve the above models. The computational steps of combinatorial algorithm are as Fig. 3.

V. Examples

Two examples are given to show the contrast between the results of presented algorithm and that of other algorithms.

Example 1 The structural geometry and the loading cases of the frame with two storied and single span are shown in Fig. 4. The Young's modulus $E = 206.88 \text{ GN/m}^2$, the specific weight of material $\rho = 76999.34 \text{ N/m}^3$, allowable stress $[\sigma] = 163.86 \text{ MN/m}^2$. The upper limit of horizontal displacement of nodes is 25.4mm. All design variables have the same discrete set shown in table 1 and the calculated results are shown in table 2.

Example 2 A ten bars plane aluminium truss shown in Fig. 5 has 6 nodes and 10 design variables. The assigned data are $E = 68.96 \text{ GN/m}^2$, $\rho = 27150.68 \text{ N/m}^3$, $[\sigma] = 172.4 \text{ MN/m}^2$. Node 2 and node 4 are subject to a down force of 444.89kN respectively, and the allowable values of the vertical displacements of all movable nodes are less than 50.8mm. The lower limits of areas of all bars are 0.645 cm^2 and their initial values are taken as 64.5 cm^2 . Discrete set is $\{0.64516, 1.9355, 6.4516, 19.355, 32.258, 51.6128, 96.774, 109.677, 141.935, 154.838, 187.096, 200.000\}$, and the calculated results are shown in table 3.

Two heuristic algorithms, relative difference quotient algorithm and modified Templeman's algorithm were used to solve example 1, and they did not both obtain the local optimum solution. For example 2, two equal local optimum solutions are obtained simultaneously by the presented algorithm, but these two local optimum solutions are obtained by two heuristic algorithms, the relative difference quotient algorithm and the combinatorial algorithm for $(0, 1)$ programming.

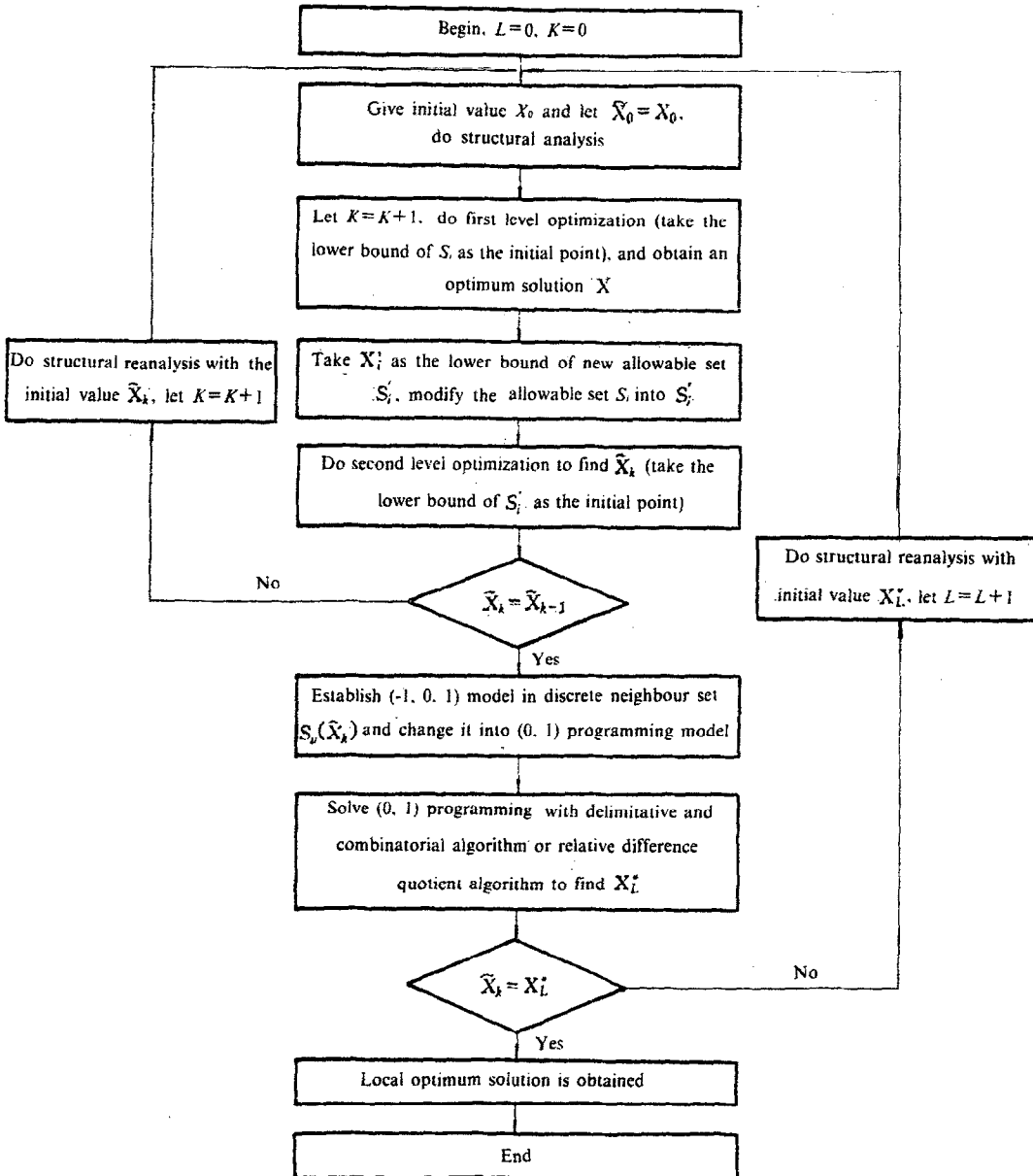


Fig. 3

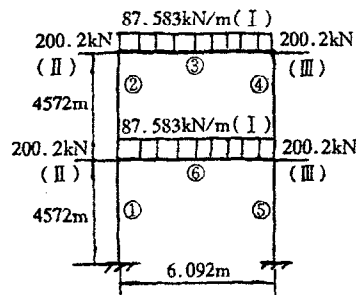


Fig. 4 The frame with two storied and single span

Table 1 Discrete set

	1	2	3	4	5	6	7	8	9
<i>A</i>	118.39	144.92	167.34	187.10	204.96	221.37	236.66	251.02	264.59
<i>W</i>	1690.2	2290.9	2842.5	3360.3	3852.7	4324.9	4780.5	5222.0	5651.4
<i>I</i>	41623	62435	83246	104058	124869	145681	166492	187304	208115

Table 2 Optimum results

	Sectional inertia moment <i>I</i>				Weight (N)	Number of reanalysis
	①⑤	②④	③	⑥		
Continuous variables	1183.13	64357	57227	174650	42296.8	4
Reference [4]	124869	62435	62435	187304	43219.3	
Reference [2]	146581	62435	62435	145681	42972.2	3
Local optimum solution	124869	62435	62435	166492	42534.4	5

Table 3 Calculated results

Element number	Sectional area of bars				
	Initial value	Reference [5]	Reference [6]	Local optimum solution	
1	141.9	187.096	200.000	187.096	200.000
2	141.9	0.64516	0.64516	0.64516	0.64516
3	141.9	154.838	141.935	154.838	141.935
4	141.9	96.7740	96.7740	96.7740	96.7740
5	141.9	0.64516	0.64516	0.64516	0.64516
6	141.9	0.64516	0.64516	0.64516	0.64516
7	141.9	51.6128	51.6128	51.6128	51.6128
8	141.9	141.935	141.935	141.935	141.935
9	141.9	141.935	141.935	141.935	141.935
10	141.9	0.64516	0.64516	0.64516	0.64516
Weight (kg)		2325.374	2325.374	2325.374	2325.374
No. of reanalysis		10	5	7	7

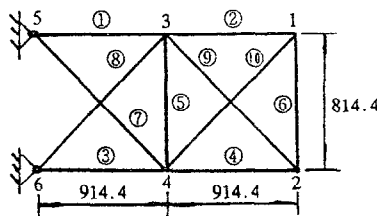


Fig. 5 Ten bars truss

References

- [1] J. S. Arora and M. W. Huang, Methods for optimization of nonlinear problems with

- discrete variables: a review, *Structural Optimization*, 8 (1994), 69~85.
- [2] Chai Shan, Wang Jian and Cao Xinzong, A direction-difference-quotient method of the optimal design of discrete variable, *Journal of Computational Structural Mechanics and Applications*, 3 (1994). (in Chinese)
 - [3] Chai Shan, A delimitative and combinatorial algorithm for a kind of (0, 1) programming and its application to discrete optimization of structures, *Engineering Mechanics*, 1 (1995). (in Chinese)
 - [4] Sui Yunkang, The optimization of beam-containing structure with discrete cross-sections and its computer implementation on plane frame structures, *Journal of Computational Structural Mechanics and Applications*, 3 (1987). (in Chinese)
 - [5] Xu Qiang and Sun Huanchun, Combinatorial algorithm to solve minimum weight design problems of truss structures with discrete variables, *Journal of Dalian University of Technology*, 6 (1991). (in Chinese)
 - [6] Chai Shan and Sun Huanchun, Difference quotient algorithm for finding the feasible set of (0, 1) programming of optimum design of structures with discrete variables, *Journal of Dalian University of Technology*, 5 (1995). (in Chinese)