An Efficient Method by Introducing Concept of Generalized Design Variables for Optimal Structural Design via GAs

by Hisashi Nobukawa*, *Member* Fengxiang Yang**, *Stnudent Member*
Mitsuru Kitamura*, *Member* Guoqiang Zhou*, *Member*

Abstract

In this paper, a new efficient method to solve the structural optimization problems with the static and dynamic constraints using Genetic Algorithms (GAs) was proposed. With this method, the static equilibrium equation and dynamic equation have no need to be solved by conventional methods resulting in saving the huge computing time which accounts for the most part of the computation in structural optimization. In order to achieve this goal, the concept of generalized design variables was introduced. The number of the variables becomes larger when the new method is applied to real-world engineering problems. To save the computing storage, in this paper, the floating point representation to the string of solution was used. Since many problems reach their optimal point on or near the boundary of constraints, the boundary mutation was introduced to speed up the convergence of the method. To improve the fine local tuning capabilities of this method, the non-uniform mutation was also used. The effect of the boundary mutation and non-uniform mutation on the performance of the GA was examined. A simple numerical example was given to illustrate applicability of this method.

1. Introduction

The calculus-based optimization techniques proceed the search from one point to a better one. For the structural optimization problems, most of the algorithms requires a large number of structural re-analyses. This repeated analyses tend to be too expensive for practical problems. It consumes the most part of computing time even though they always operate on one point every step.

GAs are fundamentally different from the traditional optimization techniques. It appears that they have some advantages[3] in some aspects. For example, they do not require any calculations of the gradient or Hessian matrix of the objective function and constraints; they converge to the global optimal point more easily; they can handle the discrete problems; etc. In the last few years, there has been a growing effort to apply GAs to the structural optimizations[1),2),5),6),10)]. However in the application to structural optimization, traditionally the structural analyses have to be done in order to introduce the static and dynamic constraints[5),6),10)]. As pointed out in Reference [10], the iteration number of structural re-analyses in GA method is much larger than that in the multiplier method and consequently takes more computing time. The reason for this is that GAs manipulate a population of points in search space every generation. It is quite difficult to apply the GAs to large-scale structures because of this weakness of time-consuming. It seems that GAs are inefficient compared to the calculus-based optimization techniques.

Up to the present, it is short of efficient methods to reduce the copious amount of computation time spent on the structural re-analyses. In this paper, the attempt was made to eliminate the need to solve the static and dynamic equations by the conventional methods. First, the concept of generalized design variables was created. Then, the penalty method was introduced into the GA. Based on these, the static and dynamic equations can be included in the new objective as penalty functions. These equations have no need to be solved by the conventional methods. GAs will push the equations to be satisfied with generations. This is one of the most significant features of GA methods. Therefore in the optimization process, only the structural stiffness and mass matrices are necessary to be formed. They should satisfy some conditions—the static and dynamic equations which govern the behavior of structure. If they don't satisfy the conditions, they will receive penalties according to their degrees of violations. As the results, with the generations, the optimal point can be found at which the governing equation of structures must be satisfied. This new method is quite simple and efficient for calculation.

* Faculty of Engineering, Hiroshima University
** Graduate School, Hiroshima University

Received 10th July 1998
Read at the Autumn meeting 12, 13th Nov. 1998

2. Illustration of Problem

The design variables X describes the structure such as the sizes of members of structure. Suppose the displacements of nodes and the eigenvalues of the structure are Y and ξ respectively.The objective is to find the design variables X to minimize the cost function of structure under the static and dynamic constraints as the following:

$$\min f(X) \tag{1}$$

subject to constraints as below:

(1) Static bending and shear stress constraints

$$\sigma_{i,max}(Y) \leq \sigma_{i,0} \qquad i=1, 2, \cdots, n \tag{2}$$

(2) Dynamic constraints

$$\xi_i(X) \geq \omega_0^2 \quad \text{for all } i \tag{3}$$

$$\xi_i(X) \leq \omega_1^2 \quad \text{for the first } I \text{ eigenvalues} \tag{4}$$

$$\xi_i(X) \geq \omega_2^2 \quad \text{for all } i>I \tag{5}$$

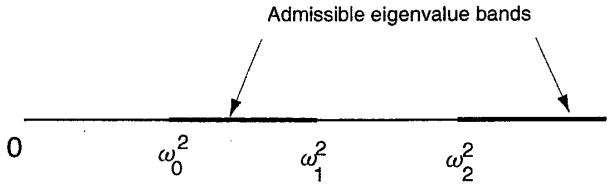

Fig. 1 Graphical representation of frequency constraints

(3) Design variable constraints

$$x_{i,min} \leqq x_i \leqq x_{i,max} \qquad i=1, 2, \cdots, q \tag{6}$$

In general, the maximum bending and shear stresses in each element can be calculated by solving the static equilibrium equation as below:

$$K(X)Y=P \tag{7}$$

$\xi_i(X)$ in formulae (3) to (5) can be calculated by solving the eigenvalue problem as below:

$$K(X)U=\xi_i M(X)U \tag{8}$$

3. GA Method with Conventional Re-analyses of Structure

Genetic Algorithms (GAs)[3),4),9)] are powerful and broadly applicable stochastic search and optimization techniques based on principle from evolution theory. Recently, Genetic Algorithms have received considerable attention regarding their potential as a novel optimization technique[1),2),5),6),10)]. As for the structural optimization problems, it can be known from Section 2 that they involve a large number of constraints. In GAs, constraint handling[5)] can be done by penalty methods[9)] which use penalty functions as an adjustment to the optimized objective function. Therefore, a constrained problem is transformed to an unconstrained problem by associating a penalty with all constraint violations. Thus the formula (1) above is transformed into optimization of the function:

$$F(X)=f(X)+\sum_{i=1}^{p}\delta_i\Phi_i(X) \tag{9}$$

where p is the total number of constraints. δ_i is a penalty coefficient. $\Phi_i(X)$ is a penalty term related to the i-th constraint $(i=1, \cdots, p)$.

Traditionally, in the applications to the structural optimization problems, only the static and dynamic constraints, and the domain constraints represented by the inequalities (2) to (6) are introduced into the new objective function with penalty terms when penalty method is used. The penalty terms of the static and dynamic constraints can be written as

$$\Phi_i^s(X)=\begin{cases}0 & \sigma_{i,max} \leq \sigma_{i,0} \\ |\sigma_{i,max}-\sigma_{i,0}|^m & \sigma_{i,max} > \sigma_{i,0}\end{cases} \tag{10}$$

$$\Phi_j^d(X)=\begin{cases}0 & \textit{in admissible bands} \\ |\xi_i-\omega_k^2|^m & \textit{in forbidden bands}\end{cases} \tag{11}$$

From Eq. 10 and Eq. 11, it is known that if the constraints are not satisfied, namely the stresses of structure are larger than allowable stresses, or the eigenvalues of structure fall in the forbidden band, the candidate design will receive penalties. The stresses and eigenvalues have to be calculated for every candidate design at every step. This procedure is natural for designers because with the design variables fixed, the stresses and eigenvalues are also fixed. Therefore the exploration of the solution space is limited in the field of design variables. This method has been applied to some real-world engineering problems[5),6),10)] successfully. However it is more time-consuming than the calculus-based optimization techniques. This weakness limits its application to the large-scale real-world engineering problems.

4. GA Method without Conventional Re-analyses of Structure

The method described above has been investigated by several authors. As already stated above, the problem of this method is time-consuming caused by the structural re-analyses. To overcome this shortcoming, a new method is proposed as below.

Displacement of nodes Y, eigenvector U and eigenvalue ξ above describe the behavior of the structure under certain conditions. They can not be chosen by the designer. Therefore they are not be selected as design variables generally. In fact, they are the function of the design variables X. They can be calculated from Eq. 7 and Eq. 8. Since it is much time-consuming to solve these equations, especially Eq. 8, we left these equations unsolved, instead include them into the penalty function. This makes us extend the concept of "design variables". Here the dependent variables Y, U, ξ can be considered independent first, the dependent relationship can be carried out by the penalty functions of Eq. 7 and Eq. 8. In this study, the generalized design variable vector Z is defined as below:

$$Z=\begin{Bmatrix}X\\Y\\\xi\end{Bmatrix} \tag{12}$$

The structural optimization problem can be rewritten as the following:

$$\min f(\boldsymbol{Z}) \tag{13}$$

subject to constraints:

(1) Static constraints

a) bending and shear stress constraints

$$\sigma_{i,max}(\boldsymbol{Z}) \leq \sigma_{i,0} \quad i=1, 2, \cdots, n \tag{14}$$

b) equality constraints

$$K(\boldsymbol{Z})\boldsymbol{Y}-\boldsymbol{P}=\boldsymbol{0} \tag{15}$$

(2) Dynamic constraints

$$|\boldsymbol{K}(\boldsymbol{Z})-\xi_i\boldsymbol{M}(\boldsymbol{Z})|=0 \tag{16}$$

(3) Design variable constraints

$$z_{i,min} \leq z_i \leq z_{i,max} \quad i=1, 2, \cdots, q \tag{17}$$

Because $\boldsymbol{Z}$ includes the state variables $\boldsymbol{Y}$ and $\boldsymbol{\xi}$, their bound constraints should be determined according to some experience. For the displacements of nodes, the design regulation has such requirement. For the natural frequency constraints, we know the forbidden band of frequency before designing. According to this, a boundary value of the constraints above can be determined.

Traditionally, Eq. 7 and Eq. 8 are not considered as constraints. Every time, when the objective with penalty terms is calculated, the equations have to be solved. Since GAs operate on a population of candidate designs for a certain number of generations, the static and dynamic analyses will be done for *pop—size*×*generation* times. This leads to the GA much time-consuming compared to calculus-based optimization techniques. To overcome this drawback, in this study, we use the generalized design variables as the design variables. Here Eq. 7 and Eq. 8 for structural analyses are rewritten as Eq. 15 and Eq. 16 which are considered as constraints. They are included in the penalty function directly as below:

For convenience, assume

$$\{\bar{p}_1, \bar{p}_2, \cdots, \bar{p}_{dof}\}^T=\boldsymbol{K}(\boldsymbol{Z})\boldsymbol{Y}-\boldsymbol{P}$$

then the penalty term of the static equilibrium equation can be formed as

$$\Phi_{se}=\sum_{i=1}^{dof}|\bar{p}_i|^m \tag{18}$$

The penalty term for dynamic eigenvalue equation is

$$\Phi_{de}=|\boldsymbol{K}(\boldsymbol{Z})-\xi_i\boldsymbol{M}(\boldsymbol{Z})|^m \tag{19}$$

As a result, the equations have no need to be solved directly by the conventional methods. When the equality constraints are not satisfied in an approximate way, the chromosome will be penalized. Obviously when the penalties approach zero, the GA has the chance to find the global lowest point within the design variables field. With the generations, the GA can find the set of design variables which make the objective reach the lowest point and, at the same time, the constraints Eq. 15 and Eq. 16 are satisfied. At the final, the optimal point of $\boldsymbol{Z}$ can be found. The iteration process is self-correcting to the equality constraints.

5. Penalty Method for Structural Optimization

The optimization problem described above can be generalized as follows:

$$\min f(\boldsymbol{Z})$$

subject to constraints:

$$g_i(\boldsymbol{Z}) \leq 0.0 \quad i=1, \cdots, m$$

$$h_j(\boldsymbol{Z})=0.0 \quad j=1, \cdots, p$$

The inequality constraints include Eq. 14 and Eq. 17. The equality constraints include Eq. 15 and Eq. 16. When penalty method is used to handle these constraints, the objective function with penalty terms is the same as Eq. 9. For the constraints above, the penalizing function can be written as the following form:

$$\Phi_i(\boldsymbol{Z})=\begin{cases}0 & g_i(\boldsymbol{Z})\leq 0\\ |g_i(\boldsymbol{Z})|^m & g_i(\boldsymbol{Z})>0\end{cases} \tag{20}$$

$$\Phi_j(\boldsymbol{Z})=\begin{cases}0 & |h_j(\boldsymbol{Z})|\leq \varepsilon\\ |h_j(\boldsymbol{Z})|^m & |h_j(\boldsymbol{Z})|>\varepsilon\end{cases} \tag{21}$$

where m can be 1 or 2. ε is the criteria used to avoid oscillation in constraint violations from one iteration to the next. $\boldsymbol{Z}$ include the design variables $\boldsymbol{X}$, the displacements of nodes $\boldsymbol{Y}$ and the eigenvalue ξ_i.

6. Boundary Mutation and Non-uniform Mutation

In addition to the elitism, simple crossover and uniform mutation, which are commonly included in practical applications, the operators of boundary mutation and non-uniform mutation are introduced in this calculation.

Since the optimal solution in structural optimizations lies often on or near the boundary of the constraints, it is essential to introduce the operator of boundary mutation to hasten the rate of convergence. With this mutation, the mutated x_k is changed to be either right bound value or left bound value with equal probability.

The non-uniform mutation is incorporated in this study to improve the fine local tuning capabilities of this GA. It works as follows. Suppose variable x_k is selected for this mutation, the result of this mutation is:

$$x'_k=\begin{cases}x_k+\Delta(t, right(k)-x_k) & if\ \gamma=0\\ x_k+\Delta(t, x_k-left(k)) & if\ \gamma=1\end{cases} \tag{22}$$

where γ is a random binary digit. t denotes the number of generation. $right(k)$ and $left(k)$ are the right bound and left bound of x_k. Function $\Delta(t, y)$ is taken as the following form:

$$\Delta(t, y)=y\cdot r\cdot\left(1-\frac{t}{T}\right)^b \tag{23}$$

where r is a random number between 0 and 1, T is the maximal generation number, and b is the coefficient for this mutation determining the degree of non-uniformity. In this study, $b=2$.

7. Numerical Example

Structural model in Fig. 2 is taken as the example to test the method.

7.1 Optimization under Dynamic Frequency Constraints

The cross-sectional areas of the members determine the volume as well as the stiffness and mass of the structure when the material and geometry are fixed. In this study, b_1, b_2, b_3 are fixed as $0.1m$, $0.1m$, $0.18m$

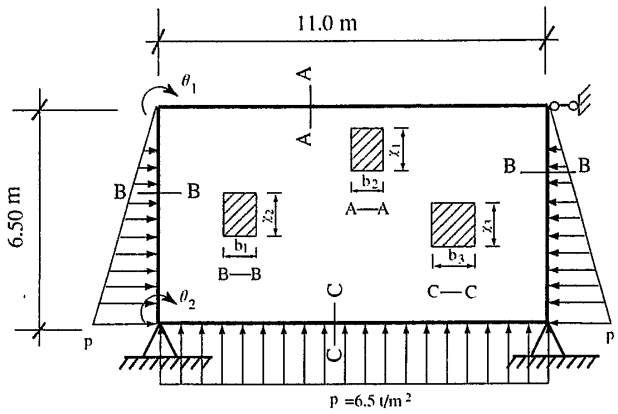

Fig. 2 Structumal model

respectively. x_1, x_2, x_3 are selected as the design variables shown in Fig. 2. The domain constraints of x_1, x_2, x_3 are from 0.1 to 0.7. The forbidden eigenvalue band for this example is [2000, 4000]. The goal here is to find out the 'best' chromosome—the set of generalized design variables, namely x_1, x_2, x_3, and the corresponding eigenvalue ξ_i, under the dynamic frequency constraints to minimize the volume of the structural members by using the proposed GA directly.

Four approaches shown in Table 1 were calculated and were run on *HP*715/100. All runs were performed with the following GA parameters: *pop-size*=50, the probability of crossover p_c=0.8, the probability of uniform mutation p_m=0.08, the probability of boundary mutation p_{bm}=0.06, the probability of non-uniform mutation p_{um}=0.05 and the coefficient for non-uniform mutation b=2. The results of the four approaches are summerized in Table 2 and Table 3. The "generation number" in Table 3 is the final generation number at which the optimization process is stopped according to the termination condition.

From Table 3 and Table 2, it can be observed that the computing time with the proposed method is reduced considerably. Approach B uses only 396 sec to reach the optimal point, by contrast, 1852 sec is spent by Approach A even though 867, the required generations to converge to the optimum by Approach B, is much larger than 212 by Approach A. From this, it should be known that the structural re-analyses take up the most part of computing time in structural optimization using

Table 1. Four GA Approaches

method	representation	structural re-analyses	boundary mutation	nonuniform mutation
A	binary	○	×	×
B	float	×	×	×
C	float	×	○	×
D	float	×	○	○

Table 2. Results of the four Approaches

method	x_1 (m)	x_2 (m)	x_3 (m)	ξ (rad/s)^2	objective (m^3)	% obj difference
exact	0.1000	0.1410	0.1000	4014.31	0.49130	0.0000
A	0.1019	0.1385	0.1033	4023.40	0.49662	1.0828
B	0.1090	0.1408	0.1000	4017.61	0.5010	1.9743
C	0.1036	0.1420	0.1000	4051.90	0.49656	1.0706
D	0.1000	0.1421	0.1000	4053.37	0.49276	0.2970

Table 3. Comparison of the four Approaches

method	CPU time (s)	string length	generation number	% obj difference	penalty (10^{-8})	% feasible
A	1852	57	212	1.0828	0.000	60
B	396	4	867	1.9743	1.221	0
C	108	4	233	1.0706	0.013	0
D	78	4	153	0.2970	5.121	0

classical GAs.

The results of Approach B and Approach C indicate that the boundary mutation makes a great contribution to convergence for this problem. Approach C with boundary mutation converges at generation 233. However Approach B find the optimal point at generation 867. From Table 3, it is also observed that the introduction of the operator of boundary mutation for this problem also improves the accuracy of the GA. The objective differences to the exact solution for Approach B and Approach C are 1.9743% and 1.0706% respectively. The non-uniform mutation is responsible for the precision. Obviously from the Table 3, the GA using the non-uniform mutation clearly outperforms the other one with respect to the accuracy of the found optimal solution. The objective difference to the exact solution is only 0.297% at iteration 153 for Approach D.

Floating point representation to solution saves the computing storage greatly. For this numerical example, in Approach A, there are three variables x_1, x_2, x_3. If the precision of five digits after the decimal point is required, the length of the binary solution vector is 57. However for the floating point representation used in the proposed method, although the number of the extended design variables x_1, x_2, x_3, ξ_i is larger, the length of the string to a chromosome is only 4.

All the methods converge to the optimal solution with a certain precision. In the proposed method, all the chromosomes are in the infeasible field. However the GA pushes them to the point which almost satisfies the constraints at the final generation. At final, the error is small enough and the more feasible point can be achieved when the number of generations increases. Table 3 shows that the sum of the penalties for any

method is very small when the GA approaches the optimal point. Table 4 represents the eigenvalue error caused by the equality constraints. ξ_0 is obtained by the proposed methods directly. ξ_1 is calculated from x_1, x_2, x_3 obtained by the GA methods, which are shown in Table 2. It is clear that the new method indeed forces the equality constraints to be satisfied with an accepted accuracy.

7.2 Optimization under Static Constraints

The same structure under the static constraints was investigated. The load is shown in Fig. 2. All b_1, b_2, b_3 are taken as $0.02m$. The allowable bending stress is $18kgf/mm^2$. x_1, x_2, x_3, θ_1, θ_2 are selected as the generalized design variables. The domain constraints of x_1, x_2, x_3 are the same as above. Both the domain constraints of θ_1, θ_2 are from 0 to 0.3×10^{-3}. Through the proposed method, the optimal point, a set of x_1, x_2, x_3, θ_1, θ_2, can be found automatically.

The same four approaches were carried out with the same GA parameters as before. The results are shown in Table 5, Table 6 and Table 7.

The computing time is also reduced greatly. However it is not difficult to understand that the efficiency of reduction of computing time to the static constraint problems is less than to the dynamic constraint problems when the system becomes complex. The reason behind this is that the dynamic analysis is more time-consuming than the static analysis. For the dynamic constraint problems, several eigenvalues must be calculated when the methods with structural re-analyses are used in order to make sure that no natural frequencies fall in the forbidden band. This needs to solve the eigenvalue problem, which results in a tremendous amount of computation to a large-scale system. However for the proposed method, just the calculation of the determinant of a matrix is needed. The eigenvalue is one of the generalized design variables. Only one variable is enough for any complex system. For the static constraint problems, there is no need to solve a system of linear equations. However the computation effort to do this is not as time-consuming as to solve a large-scale eigenvalue problem. Moreover the number of the generalized design variables in the static constraint problems becomes much larger than that in the traditional method because all the displacements of nodes are taken as the independent design variables.

The boundary mutation in this numerical example produces little effect. This operator is designed for the problems in which the optimal point is reached on or near the boundary of the constraints. In the dynamic constraint problem described above, three of four generalized design variables meet or approach their edges of domain constraints. For the static constraint problems, since the displacements are introduced into the generalized design variables, generally the optimal point has little possibility to be on or near their edges of domain constraints. It seems that the more the number of variables which approach the edges of their domain constraints at optimal point, the more useful this operator is. The operator of non-uniform mutation also enhances the accuracy of the proposed method from 0.5113% for Approach C to 0.3863% for Approach D shown in Table 6.

Table 7 shows the error of displacements. θ with superscript G represents the displacement obtained from the proposed method directly. θ with superscript X represents the displacement calculated from x_1, x_2, x_3

Table 4. Error of the eigenvalue

method	ξ_0	ξ_1	difference %	penalty (10^{-8})
A	4023.40	4023.40	0.000000	0.000
B	4017.61	4016.43	0.029379	1.221
C	4051.90	4051.90	0.000000	0.013
D	4053.37	4052.40	0.023936	5.121

Table 5. Results of the four Approaches

	x_1 (m)	x_2 (m)	x_3 (m)	σ_{max} kg/mm^2	obj m^3	error obj %	error σ %
exact	0.1000	0.1366	0.1386	17.990	.08801	0.0	0.0
A	0.1000	0.1368	0.1383	17.977	.08799	0.0227	0.072
B	0.1012	0.1380	0.1372	17.829	.08833	0.3636	0.895
C	0.1000	0.1364	0.1368	18.163	.08756	0.5113	0.962
D	0.1000	0.1370	0.1366	18.061	.08767	0.3863	0.395

Table 6. Comparison of the four Approaches

method	CPU time (s)	string length	generation number	% obj difference	penalty (10^{-10})	% feasible
A	85	57	179	0.0227	0.000	80
B	24	5	2772	0.0363	9.702	0
C	24	5	2675	0.5113	9.822	0
D	19	5	1912	0.3863	9.877	0

Table 7. Error of displacements

method	θ_1^G (10^{-5})	θ_2^G (10^{-4})	θ_1^X (10^{-5})	θ_2^X (10^{-4})	% difference θ_1	% difference θ_2
A	4.513	-1.724	4.513	-1.724	0.00	0.00
B	4.776	-1.725	4.694	-1.703	1.75	1.29
C	5.000	-1.776	4.600	-1.751	8.70	1.43
D	5.000	-1.773	4.585	-1.737	9.05	2.07

through structural analysis. It is observed that the differences between them are large, especially for θ_1. However the stresses and the optimal point are not sensitive to them for this example. The errors of the objective and stresses are all below 1% shown in Table 5.

8. Conclusions

1) The proposed method was successfully applied to structural optimization problems without the need to solve the structural equations by conventional methods. It seems that the GA provided in this paper has a huge potential to solve structural optimization problems in an efficient way. The calculation is simple and the search can reach the global optimum. All the work needed is to form the stiffness and mass matrices. After that, the new method can find the global optimal point automatically.

2) The proposed method save computing time significantly. This method overcomes the drawback of the traditional GA method in the applications to structural optimization. The floating point representation is suitable for real-world engineering problems.

3) The boundary mutation and non-uniform mutation improve the GA significantly for some of structural optimization problems. The boundary mutation speeds up the convergence of GAs considerably for the real-world engineering problems whose constraints are active at the target global optimum. From this study, we know that the GA with non-uniform mutation outperforms the ones without it.

4) The proposed method is more efficient to dynamic constraint problems than to the static constraint problems.

References

1) Tetsuo Okada, Isao Neki, Optimization of Ship Structural Design by Genetic Algorithm, J. Soc. Naval Arch. of Japan, (April, 1992), pp 259-266.
2) Isao Neki, Tetsuo Okada, Optimum Structural Design Using Genetic Algorithm and Finite Element Method, J. Soc. Naval Arch. of Japan, (July, 1995) pp 327-337.
3) Zbigniew Michalewicz, Genetic Algorithms+Data Structures=Evolution Programs, 3 rd rev. Springer-Verlag. 1996.
4) Goldberg, D. E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, Reading, Mass., 1989.
5) Guoqiang Zhou, Hisashi Nobukawa and Fengxiang Yang, Discrete Optimization of Ship Structures from the Viewpoint of Practical Design, J. Soc. Naval Arch. of Japan, (Nov. 1997), pp 551-559.
6) Hisashi Nobukawa, Yang Fengxiang, Mitsuru Kitamura, Zhou Guoqiang, Optimization of Engine Room Structure under Static and Dynamic Constraints Using Genetic Algorithms. J. Soc. Naval Arch. of Japan, (May. 1998), pp 315-322.
7) Jeffrey A. Joines, Christopher R. Houck, On the Use of Non-stationary Penalty Functions to Solve Nonlinear Constrained Optimization Problems with GA's. IEEE (1985), pp 579-584.
8) Michalewicz, Z. and Janikow, C., Handling Constraints in Genetic Algorithms, Proceeding of the Fourth International Conference on Genetic Algorithms, (1991), pp 151-157.
9) Mitsuo Gen, Runwei Cheng, Genetic Algorithms and Engineering Design, JOHN WILEY & SONS, INC. 1997.
10) Hisashi Nobukawa, Guoqiang Zhou, Discrete Optimization of Ship Structures with Genetic Algorithms, Journal of The Society of Naval Architects of Japan, Vol. 179, (1996), pp 293-301.