

Discrete Optimization of Ship Structures from the Viewpoint of Practical Design

by Guoqiang Zhou*, *Member* Hisashi Nobukawa*, *Member*
Fengxiang Yang**

Summary

In practical design of ship structures, it is obvious that some structural elements, for instance, longitudinals, must be chosen from a given set of standard discrete values being available commercially. Furthermore, some integer parameters such as the number of transverse web frames in the midship segment should be considered as the design variables to be optimized. In view of those facts, we investigate especially how to deal with the above kinds of discrete design variables in this paper. A cargo ship with large hatch opening is taken as the numerical example. The structural construction cost of hold length between two transverse bulkheads is chosen as the objective function to be minimized by genetic algorithms. Besides the conventional variable design parameters, some particular parameters such as the serial number of standard shaped steel for longitudinals and for transverse frames, the number of longitudinals in a given length, the number of transverse web frames in the midship segment, etc., are also selected as the design variables from the viewpoint of practical design. For each design variable, we describe it by a specific length binary string with a corresponding relationship to its available discrete values. Through the numerical analyses, it shows that the present method is a simple yet efficient method to deal with the discrete optimization of ship structures. The search efficiency of some genetic strategies such as the elitist preserving selection, the multi-point crossover and the uniform crossover is also investigated in this paper.

1. Introduction

Reference 1) developed a discrete method for optimization of ship structures by genetic algorithms based on the schema theorem. Numerical analyses demonstrate that the method can handle the optimization of ship structures with discrete design variables well. Similar to the traditional optimization method, the web depth and the longitudinal spacing of longitudinals are still chosen as design variables in that study. However, the web depth of longitudinals can not be taken as a design variable independently in practical ship design. In order to reduce the fabrication cost, longitudinals should be taken as a type of shaped steel element to be selected from a given set of standard discrete values being available commercially. Since the number of longitudinals in a given length must be an integer, it is also unreasonable that the longitudinal spacing is chosen as a design variable.

Moreover, transverse frame spacing in the midship segment is an important variable design parameter in

practical ship design. Considerable range is allowed by the classification societies regarding this design parameter. Generally, ship designers decide transverse frame spacing by experience. On determining it, designers can easily design other ship structures under the guidance of rules and regulations. Thus, the transverse frame spacing in the midship segment should be considered as a design variable to be optimized. However, if the transverse frame spacing is determined in advance, it is difficult to ensure that the number of transverse frames in the given midship segment is an integer. Therefore, we substitute the number of transverse frames for the transverse frame spacing as a design variable in this study.

In view of those facts, we pay special attention to dealing with the above kinds of discrete design variables in this paper. The genetic algorithm is taken as an optimization method. Different from the traditional optimization method, some particular parameters are taken as the design variable from the viewpoint of practical design in this study. For example, the serial number of standard shaped steel for longitudinals and that for transverse frames, rather than their web depths, are considered as the design variables. In addition, the number of longitudinals in a given length and the number of transverse web frames in the midship segment are taken as the integer design variables. The

* Faculty of Engineering, Hiroshima University

** Graduate School, Hiroshima University

Received 10th July 1997

Read at the Autumn meeting 14, 15th Nov. 1997

longitudinal spacing is no longer assigned as a design variable. For each design variable, we describe it by a specific length binary string with a corresponding relationship to its available discrete values. We take a cargo ship with large hatch opening as the numerical example. In this study, the search efficiency of the elitist preserving selection, the multi-point crossover and the uniform crossover is also investigated.

2. Establishment of Fitness Function

2.1 Objective Function and Design Constraints

The optimal design problem can be defined as:

$$\min f(\mathbf{X}) \quad \text{s.t. } g_i(\mathbf{X}) \geq 0 \quad i=1, 2, \dots, m \quad (1)$$

where $\mathbf{X}=(X_1, X_2, \dots, X_n)^T$ is a vector containing n design variables, $f(\mathbf{X})$ is the objective function to be minimized, $g_i(\mathbf{X})$ is the i th constraint and m is the number of constraints.

In optimum design of ship structures, the objective function can be the structural weight and/or the structural construction cost. In this study, a cargo ship with large hatch opening is taken as the numerical example. The structural construction cost of the hold length between two transverse bulkheads is taken as the objective function to be optimized by genetic algorithms. The total cost consists of the material cost and the welding cost. The material cost is further divided into the cost of plate steel and the cost of shaped steel.

For the numerical example, the design constraints are considered as follows

(1) Bending strength for

- transverse web frames
- transverse frames
- longitudinals on bottoms
- plates of bottoms and side hulls
- ship hull panels
- longitudinal girders and stringers

$$g(\mathbf{X})=1-\frac{\sigma_{\max}}{\sigma_p} \geq 0 \quad (2)$$

where $\sigma_{\max}$ is the maximum bending stress in a structural member and σ_p is the allowable bending stress.

(2) Shear strength for

- transverse web frames
- longitudinal girders and stringers
- ship hull panels

$$g(\mathbf{X})=1-\frac{\tau_{\max}}{\tau_p} \geq 0 \quad (3)$$

where $\tau_{\max}$ is the maximum shear stress in a structural member and τ_p is the allowable shear stress.

(3) Buckling strength for

- longitudinals on bottoms
- plates of bottoms and side hulls

$$g(\mathbf{X})=1-\frac{\sigma_{co}}{\sigma_{cr}} \geq 0 \quad (4)$$

where σ_{co} is the assuming applied buckling stress and σ_{cr} is the critical buckling stress of structural member.

(4) Allowable maximum horizontal deflection of hatch opening for

- transverse strength
- longitudinal bending and torsional strength

$$g(\mathbf{X})=1-\frac{d_{\max}}{d_a} \geq 0 \quad (5)$$

where $d_{\max}$ is the maximum horizontal deflection of hatch opening and d_a is the allowable horizontal deflection of hatch opening.

(5) Required midship sectional modulus

$$g(\mathbf{X})=1-\frac{W_{\min}}{Z_{\min}} \geq 0 \quad (6)$$

where $W_{\min}$ is the required midship sectional modulus evaluated by NK rule²⁾ and $Z_{\min}$ is the minimum midship sectional modulus.

In the above-mentioned design constraints, many factors such as deflections, stresses and forces are evaluated by structural analyses. Two processes of structural analyses are carried out for the cargo ship: One is the structural analysis for transverse strength, the other is the structural analysis for longitudinal bending and torsional strength. Both structural analyses were in detail described in Reference 3).

2.2 Fitness Function

Genetic algorithms do search procedure based on an unconstrained fitness function. As the same as the method of Reference 1), we make two transformations to form the unconstrained fitness function from the constrained minimization problem defined in Eq.(1).

First, the constrained minimization problem is transformed into an unconstrained one by a penalty term depending on the degree of constraint violation. The augmented objective function of unconstrained minimization problem with the penalty term is expressed as

$$\Phi(\mathbf{X})=f(\mathbf{X})\left(1-c_p \sum_{i=1}^m \{\min[0, g_i(\mathbf{X})]\}\right) \quad (7)$$

where $\Phi(\mathbf{X})$ is the augmented objective function of the unconstrained minimization and c_p is the influence coefficient of the penalty term.

In the second step, the augmented objective function $\Phi(\mathbf{X})$ is further converted into a corresponding fitness value as follows

$$F_j(\mathbf{X})=\Phi_{\max}(\mathbf{X})-\Phi_j(\mathbf{X}) \quad (8)$$

where $F_j(\mathbf{X})$ is the fitness function of the j th individual and $\Phi_{\max}(\mathbf{X})$ is the maximum value of the augmented function from all the individuals in the population.

3. Discrete Design Variables

3.1 Coding of Discrete Design Variables

Genetic algorithms work with a coding of design variables, rather than the design variables directly. To use genetic algorithms, we must first code the design variables into a finite length string. The most common coding method is to transform the design variables into a binary string of specific length. If the corresponding relationship between the discrete design variables and a binary string of finite length is established, genetic algorithms can deal with the discrete optimization problems efficiently. For example, as shown in Fig. 1, a

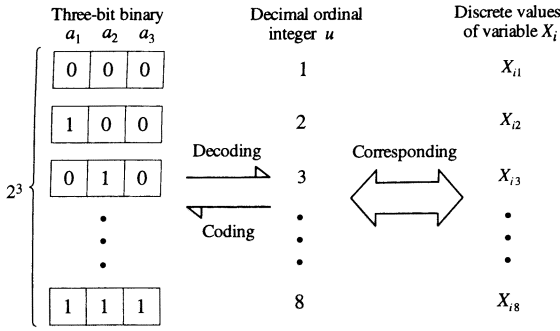


Fig. 1 Coding and decoding of discrete design variable

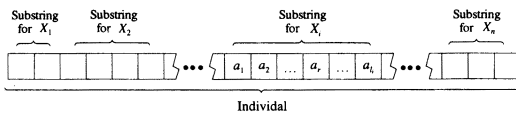


Fig. 2 A multivariable string

discrete design variable with eight discrete values can correspondingly be coded into a three-bit binary string to represent. Generally, a discrete design variable with 2^l discrete values can be represented by a binary string with the length l .

For a multivariable problem, a multivariable coding is constructed by simply concatenating the substring of each single variable into a complete string, as shown in Fig. 2.

3.2 Decoding of Discrete Design Variables

In order to evaluate the fitness function, we decode each binary substring of the discrete design variable in the multivariable coding into its corresponding discrete value respectively. Referring to the relationship shown in Fig. 1, we can get the following expression:

$$u = 1 + \sum_{r=1}^{l_i} a_r 2^{r-1} \quad (9)$$

where u is the decimal ordinal integer of the i th design variable X_i , l_i is the coding length of the binary substring for the i th design variable, a_r ($a_r \in \{0, 1\}$) is the r th allele's value in the binary substring of the i th design variable.

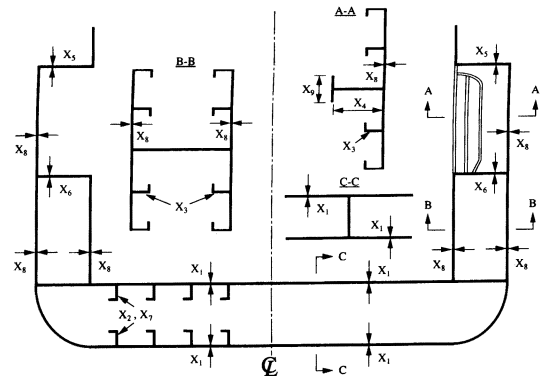
If a discrete design variable has the regular discrete values with a step between its maximum and minimum values, its discrete value can be further calculated by the equation:

$$X_i^u = X_{i \min} + \frac{X_{i \max} - X_{i \min}}{2^{l_i} - 1} (u - 1) \quad (10)$$

where X_i^u is the u th discrete value of the i th design variable, $X_{i \min}$ is the minimum value of the i th design variable and $X_{i \max}$ is the maximum one.

3.3 Design Variables for Numerical Example

Now, let us use some examples to describe how to deal with some special design variables in genetic



Note: X_2 is the variable for the serial number of standard shaped steel used by longitudinals on bottoms;
 X_3 is the variable for the serial number of standard shaped steel used by transverse frames;
 X_7 is the variable for the number of longitudinals on bottoms in the half bottom length;
 X_{10} is the variable for the number of transverse web frame spaces (unmarked in the figure).

Fig. 3 Design variables

algorithms. In this study, ten variable parameters of ship structural members for the numerical example of cargo ship are chosen as the design variables shown in Fig. 3. Different from those described in Reference 1), some particular parameters are considered as design variables in this study. The differences are described as follows:

- (1) The serial number of standard shaped steel for the bottom longitudinals and that for the transverse frames, rather than their web depths, are assigned as design variables.
- (2) The number of longitudinals in the half bottom length is taken as an integer design variable. The longitudinal spacing for the bottom longitudinals is no longer considered as a design variable.
- (3) The number of transverse web frames in the midship segment is selected as a new integer design variables to be dealt with.

The above four design variables are specially described in Fig. 3. The other design variables are kept as the same as those in Reference 1). All design variables are assumed to have the integer or discrete values in this study. Their detail information is listed in Table 1. By simply concatenating each single variable substring,

Table 1 Discrete values of design variables

Design Variable	Minimum	Maximum	Step	Substring Length
X_1 : thickness of bottom plates (cm)	1.00	1.75	0.05	4
X_2 : serial number of shaped steel for bottom longitudinals	3	6	1	2
X_3 : serial number of shaped steel for transverse frames	3	6	1	2
X_4 : depth of upper side transverse web frames (cm)	45.00	60.00	1.00	4
X_5 : thickness of deck girder plate (cm)	1.00	1.75	0.05	4
X_6 : thickness of side stringer plate (cm)	1.00	1.75	0.05	4
X_7 : number of longitudinals in half bottom length	6	9	1	2
X_8 : thickness of side hull plates (cm)	1.00	1.75	0.05	4
X_9 : flange width of upper side transverse web frames (cm)	10.00	17.50	0.50	4
X_{10} : number of transverse web frame spaces	12	18	2	2
Multivariable String Length				32

a multivariable string of thirty-two-bit length is constructed.

It is worth noticing that not all design variables have the regular discrete values with a step between their minimum and maximum values. As mentioned before, the serial number of standard shaped steel for longitudinals and that for transverse frames are assigned as the design variables. Their web depths are no longer taken as the design variable. It is appear that the serial number of standard shaped steel can be considered as a design variable with regular discrete values. However, the practical scantlings of standard shaped steel are the pure discrete values, that is, their scantlings can not be calculated by Eq. (10). For this instance, it is necessary to supply an additional list of discrete values that the design variable can take. In this study, the inverted unequal leg angles are taken as the shaped steel for longitudinals and for transverse frames. The available discrete values of standard inverted unequal leg angles for ship building are listed in Table 2. Therefore, the scantlings of standard shaped steel are practically decided by the relationship shown in Fig. 4 in the optimization process. Each ordinal number represents a group of scantling data of the standard inverted unequal leg angles, that is, each ordinal number corresponds to a group of discrete scantling data rather than a single one. In addition, the number of longitudinals in the half bottom length, rather than the longitudinal spacing, is taken as the integer design variables. The number of transverse web frames in the midship segment is also considered as an integer design variable.

Table 2 Available discrete values of inverted unequal leg angles for ship building

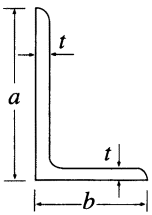
Serial No.	Sizes ($a \times b \times t$) (mm)	
1	100 × 75 × 7	
2	100 × 75 × 10	
3	125 × 75 × 7	
4	125 × 75 × 10	
5	150 × 90 × 9	
6	150 × 90 × 12	

Table 3 Practical transverse and longitudinal spacing lengths

Number of transverse web frame spaces	12	14	16	18
Length of transverse spacing (cm)	320.00	274.29	240.00	213.33
Number of longitudinals in bottoms	6	7	8	9
Length of longitudinal spacing (cm)	75.00	65.63	58.33	52.50

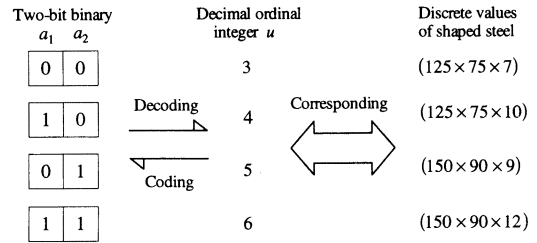


Fig. 4 Relationship between the binary coding and the discrete values for shaped steel

For these two integer design variables, their practical spacing lengths are shown in Table 3. These spacing lengths can also be calculated by the expression :

the spacing length = the given length / the number of spacings

4. Optimization Algorithms

4.1 Genetic Operators

Genetic algorithms get power from the genetic operators. The search procedure in genetic algorithms is essentially executed by three basic but very important operators: reproduction, crossover and mutation. The following genetic operators and strategies are used in this study.

(1) Reproduction operator

There are many ways to achieve effective reproduction. One simple strategy to select individual strings in the population is based on the following probability

$$p_j = \frac{F_j(\mathbf{X})}{\sum_{i=1}^{n_p} F_i(\mathbf{X})} \quad (11)$$

where p_j is the selection probability of the j th individual strings for reproduction and n_p is the size of population.

The above selection strategy is called the roulette wheel selection. There is another usual selection strategy named the elitist preserving selection. Its basic algorithm is: supposed that $\mathbf{X}^*(t)$ is the elitist individual, that is, the best individual generated up to generation t , $\mathbf{X}^*(t)$ will be included to the subsequent generations as the $(n_p + 1)$ th member until a new elitist individual better than $\mathbf{X}^*(t)$ is found to substitute for it. Its advantage is that the structure of the elitist individual up to generation t will not be destroyed by the subsequent genetic operators unless the elitist individual can be substituted by a better elitist individual generated in the subsequent generations. However, the selection strategy is easy to converge prematurely due to the elitist individual to reproduce quickly in large amounts.

(2) Crossover operator

Crossover is a primary operator in the mating process which generates the offsprings or new generations. In this paper, three kinds of crossover operators are investigated: the one-point crossover, the two-point crossover and the uniform crossover. All of them proceed in

two steps. In the first step, individual strings in the mating pool generated by the reproduction operator are mated at random. The difference among them is the operations of the second step. Assumed that l is the length of the string, each pair of individual strings undergoes crossing over respectively by the three cross-over operators as follows:

One-point crossover: an integer position i along the string is decided uniformly at random between 1 and the string length less one, $[1, l-1]$. Two new strings are created by swapping all characters between position $i+1$ and l inclusively, as shown in Fig. 5(a).

Two-point crossover: two integer positions i and j ($i < j$) along the strings are decided uniformly at random between 1 and the string length less one, $[1, l-1]$. All characters between position $i+1$ and j are inclusively exchanged to generate two new strings, as shown in Fig. 5(b).

Uniform crossover⁴⁾: A mask binary string with the length l is generated randomly. The bit values in two parent strings cross over or not depending on the bit value in the same position of the mask. Two new strings are generated by exchanging the characters in the position which corresponding bit value of the mask is 1, and keeping the characters in the position which corresponding bit value of the mask is 0, as shown in Fig. 5(c).

(3) Mutation operator

Mutation is the occasional random alteration of an allele's value in a new string. If a mutation operation is executed, it means changing a 1 to a 0 and vice versa in a binary coding.

4.2 Numerical Algorithms

The flowchart of genetic algorithms is shown in Fig. 6. Its basic algorithms are described as follows:

Step 1: With a well-defined fitness function and coding, an initial population of strings is randomly generat-

ed.

Step 2: By decoding the individual strings of the initial population, the solution for each specific instance is determined and the fitness value for this individual string is evaluated. This applies to all members of the population.

Step 3: According to the fitness values of individual strings, a randomized selection process named reproduction is implemented. Similar to the natural selection, the string with a higher fitness value has a higher probability to be reproduced.

Step 4: After reproduction, a crossover operation proceeds with a specified crossover probability. Two parent strings in the mating pool are selected randomly, then part of their bit information is exchanged to create two offspring strings according to a certain crossover scheme. The resulting strings are placed in the new population continuously until the population is filled

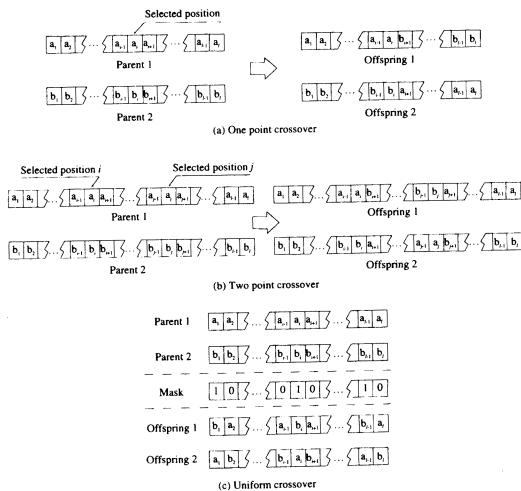


Fig. 5 Crossover operators

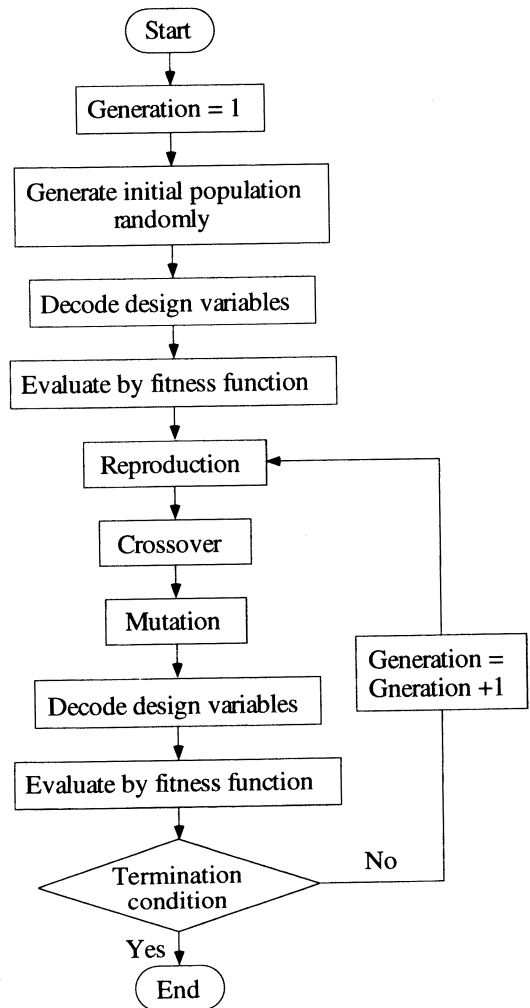


Fig. 6 Flowchart of genetic algorithms

with new progenies.

Step 5: Then mutation, a random walk, is applied to the whole string space with a small mutation probability. In a binary coding, it changes a particular bit of coded string from 0 to 1 or vice versa if a mutation operation happens.

Step 6: Decode all individual strings in the new generation and evaluate their fitness values respectively. If a given termination condition is satisfied, the transmission process is to end. Otherwise, go to step 3 to repeat the above transmission process from generation to generation until the termination condition is satisfied.

In this study, a maximum iteration number of generations is taken as the termination condition.

5. Numerical Analyses

5.1 Outline of Numerical Example

The principal dimensions of cargo ship for numerical analyses are:

Length B. P. = 70 m; Breadth (mld.) = 12.5 m;
Depth (mld.) = 6.8 m; Draught = 3.7 m;
Deadweight = 1,750 tonf.

The number of transverse web frame spacings within the hold length for the actual ship is sixteen. All structural members are assumed to be made of mild steel. The unit price per ton of plate steel is 90,000 yens and that of shaped steel is 105,000 yens. The price of fillet welding per unit welding length in meter with unit welding leg length in centi-meter is 4,200 yens.

Two load cases shown in Fig. 7 are considered for the numerical analyses. One is the hogging condition for the transverse strength analysis. The other is the load condition on the oblique wave for the longitudinal bending and torsional strength. Both load cases are evaluated on the basis of NK rules²⁾.

The allowable elastic stresses for the structural members are listed in Table 4. The other allowable stresses are considered as follows: the yield stress $\sigma_Y = 2\,800$ kgf/cm² and the assuming applied buckling stress $\sigma_{co} = 2\,000$ kgf/cm². The allowable horizontal deflection of hatch opening is taken as 12.0 mm.

A multivariable string of thirty-two-bit length is constructed for the numerical example. All design variables are assumed to have the integer or discrete values. Their detail information is illustrated in Table 1, Table 2 and Table 3. With the multivariable string

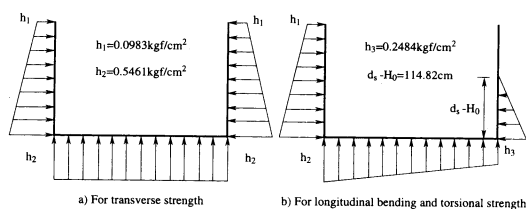


Fig. 7 Load conditions

coding, the example cargo ship is optimized by genetic algorithms. The maximum iteration number of generations for all numerical cases is assigned to be one hundred. Moreover, the penalty coefficient c_p is taken as 0.05, the population size n_p as 50 and the mutation probability p_m as 0.003 for all numerical cases.

5.2 Results of Optimum Design

First, the numerical example is optimized by a simple genetic algorithm in which the genetic strategies of the roulette wheel selection and the one-point crossover are used. The optimum design is obtained with the crossover probability $p_c = 0.8$. The generation history for the optimum design is plotted in Fig. 8 and the optimum results are listed in Table 5.

Compared with the values of the actual ship, the number of longitudinals on bottoms for the optimum design is smaller, but its corresponding serial number of standard shaped steel becomes larger. These results are similar to the optimum design obtained in Reference 1). However, there is an essential difference between them. If the longitudinal spacing is taken as a design variable, it is difficult to ensure that the number of longitudinals

Table 4 Allowable elastic stresses for structural members

Structural Member	Bending Stress (kgf/mm ²)	Shear Stress (kgf/mm ²)
Transverse web frames	18.0	8.5
Longitudinal girders	10.0	5.5
Longitudinals	10.0	---
Transverse frames	10.0	---
Side hull & bottom plates	20.0	---
Ship hull panels	18.0	8.5

Table 5 Results of discrete optimization

Item		Actual Ship	Optimum
Design	X ₁ (cm)	1.20	1.35
	X ₂ (no.)	4 (125 × 75 × 10)	6 (150 × 90 × 12)
	X ₃ (no.)	4 (125 × 75 × 10)	4 (125 × 75 × 10)
Variable	X ₄ (cm)	50.00	55.00
	X ₅ (cm)	1.40	1.00
	X ₆ (cm)	1.00	1.00
Values	X ₇ (num.)	7	6
	X ₈ (cm)	1.20	1.20
	X ₉ (cm)	15.00	17.00
	X ₁₀ (num.)	16	14
Cost Values (10 ³ yen)	Material	24,991	25,862
	Welding	24,783	20,282
	Total	49,774	46,144

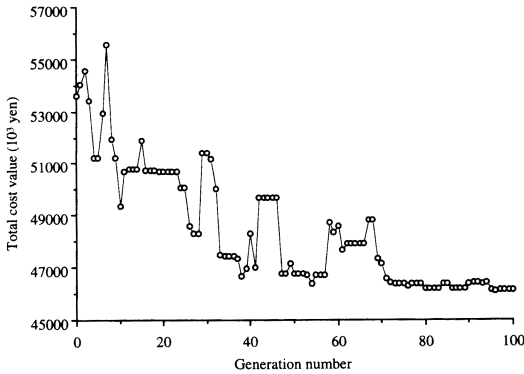


Fig. 8 Generation history of total cost value for the optimum design

in a given length is an integer exactly. As a result, the optimum design may be practically unfeasible. From the results illustrated in Reference 1), the optimum longitudinal spacing is 76.00 cm. It is near to the longitudinal spacing 75.00 cm which corresponding number of longitudinals is six, the minimum limit for the number of longitudinals in the half bottom length. Therefore, there is no integer number of longitudinals in the given bottom length corresponding to the longitudinal spacing 76.00 cm exactly. Similarly, it is also unreasonable to take the web depth of longitudinals as a design variable. If we fabricate longitudinals directly according to the optimum values of their web depths, rather than utilize the standard shaped steel being available commercially, it may cause the fabrication cost to increase. The optimum web depth for longitudinals on bottoms is 17.50 cm and that for transverse frames is 14.00 cm in Reference 1). According to the available discrete values of inverted unequal leg angles for ship building listed in Table 2, there is also no exact standard shaped steel corresponding to them.

In addition, the optimum value for the number of transverse web frame spaces, the new design variable in this study, is fourteen which is different from that for the actual ship. If the number of transverse web frame spaces is not taken as a design variable, it is obvious that several numerical analyses with different given number of transverse web frame spaces have to be done to find the optimum number of transverse web frame spaces.

The generation history of several active design constraints is shown in Fig. 9. Because the best individual in the feasible individuals is selected as the optimization solution at each generation, the design constraints are always satisfied in every generation (a little violation of design variables is allowed since the exterior penalty term is adopted in this study). In addition, it has been found that these design constraints became active as the optimization process converges. This is because the penalty term should tend towards zero as the optimiza-

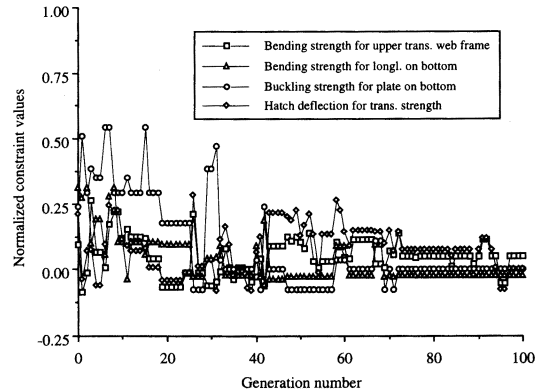


Fig. 9 Generation history of several active design constraints

tion process converges. In other words, according to the definition of Eq. (8), the fitness function will get its maximum value as the optimization process converges.

The above facts demonstrate that the present method can deal with the discrete design variables well whether they have the regular discrete values or not. Even some variable design parameters with pure discrete values, such as standard shaped steels for structural members, transverse spacing or longitudinal spacing in a given length, can be transformed into a certain form of design variable to be optimized. The distinguish to deal with the design variables with or without the regular discrete values is mainly that the description of their discrete values is different. If a design variable has regular discrete values with a step, its discrete value can be calculated by Eq. (10). Otherwise, we must supply an additional list of discrete values for the pure discrete design variable.

5.3 Analyses of Different Genetic Strategies

In genetic algorithms, there are many strategies and techniques to achieve effective search for optimization. In this paper, the search efficiency of the elitist preserving selection, the multi-point crossover and the uniform crossover is also investigated numerically. Their results are compared with those of the simple genetic algorithm described in Section 5.2.

(1) Reproduction operator

It is necessary to point out that the elitist preserving selection is used together with the roulette wheel selection for reproduction in this paper. As a preserving strategy, the elitist preserving selection is applied to select the best individual up to the present generation as the $(n_p + 1)$ individual for the next generation. Except this point, the other genetic strategies are the same as those used in the simple genetic algorithm.

In this study, the search efficiency of the elitist preserving selection with the crossover probability $p_c = 0.6, 0.7, 0.8$ and 0.9 is investigated. The generation histories for the four cases are illustrated in Fig. 10. For the

purpose of comparison, the optimum design of the simple genetic algorithm is also plotted in the same figure.

Since the elitist preserving strategy is utilized, it is obvious that the values of the objective function decrease monotonically. It means that the structure of the elitist individual up to the present generation will not be destroyed by the subsequent genetic operators unless a better elitist individual is found to substitute for it in the subsequent generations. This is an advantage of the elitist preserving selection over the other reproduction strategies. However, it is a pity that the results in the four cases with the elitist preserving selection are not so good for this numerical example. All of them are convergent prematurely comparing to the results of the simple genetic algorithm. It shows that this selection strategy is easy to cause a premature result due to the elitist individual to reproduce quickly in large amounts. This is a main shortcoming needed to overcome when the elitist preserving strategy is used.

(2) Crossover operators

The genetic operators of the two-point crossover and

the uniform crossover are analyzed numerically. The crossover probability p_c of optimum design is 0.7 for the case of the two-point crossover operator, while that is 0.6 for the case of the uniform crossover operator. The generation histories for these two cases and the optimum design of the simple genetic algorithm (with the one-point crossover operator) are plotted in Fig. 11. From these results, we observe that the case of the two-point crossover operator is a little better than that of the one-point crossover operators. However, the final results for these two cases are almost the same. The case of the uniform crossover operator is early to converge, but it is premature a little comparing to the other two cases.

From the results of the above numerical analyses, there is not remarkable improvement over the simple genetic algorithm to be observed when the genetic strategies of the elitist preserving selection, the multi-point crossover and the uniform crossover are used for this numerical example. These advanced genetic strategies can obtain good results sometimes in many circumstances. However, the simple genetic algorithm can also get enough satisfying results for ordinal optimization problems. This is a reason why the simple genetic algorithm is widely applied to solve varied optimization problems.

6. Concluding Remarks

In this study, we pay special attention to the discrete optimization problem from the viewpoint of practical design. Through the numerical analyses, it demonstrates that the present method can deal with the discrete design variables well whether they have the regular discrete values or not. Genetic algorithms use only fitness function information to do search. They are not limited by restrictive assumption about search space, such as continuity or existence of derivatives. This characteristic makes genetic algorithms possible to handle discrete design variables even though they may be with pure or integer discrete values. If a design variable has regular discrete values with a step, we can calculate its discrete value by Eq. (10). Otherwise, we must supply an additional list of discrete values for the pure discrete design variable.

It is important to select design variables properly in practical optimization design, or else it may result in an unfeasible optimum design. For example, if the longitudinal spacing is taken as a design variable directly, its optimum result may be an unfeasible one because the corresponding number of longitudinals in a given length may not be an integer. In this instance, we can substitute the number of longitudinals in a given length for the longitudinal spacing as a design variable. Similarly, some other variable design parameters with pure discrete values, such as standard shaped steels for structural members, can also be transformed into a design variable with an additional list of discrete values to be

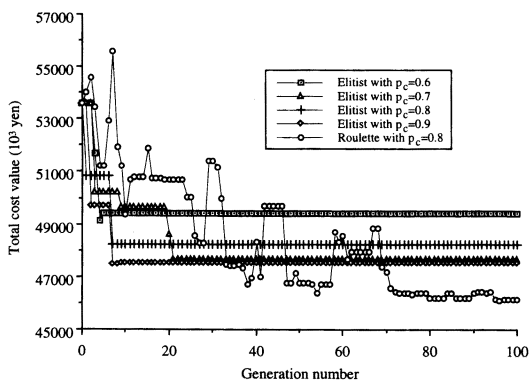


Fig. 10 Generation history of total cost value with different reproduction operators

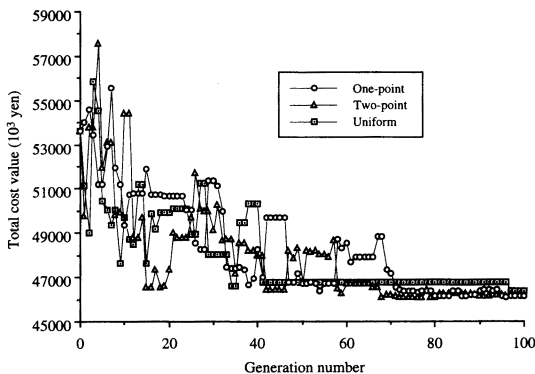


Fig. 11 Generation history of total cost value with different crossover operators

optimized.

The simple genetic algorithm is a simple yet powerful optimization method. It is enough to deal with the ordinal optimization problem effectively. In this optimization method, the roulette wheel selection and the one-point crossover are used.

As for further studies, multi-purpose design databases should be established to support discrete optimization design of manifold ship structures.

References

- 1) Nobukawa, H., and Zhou, G.: "Discrete Optimization of Ship Structures with Genetic Algorithms", J. Soc. Naval Arch. of Japan, Vol. 179, 1996, pp. 293-301.
 - 2) Nippon Kaiji Kyokai: Rules and Regulations for the Construction and Classification of Ships, Amendment, 1994 (in Japanese).
 - 3) Nobukawa, H., Zhou, G., Kitamura, M., and Kobayashi, S.: "Structural Optimization for Small Container Ship with Large Hatch Opening", Trans. West Japan Soc. of Naval Arch., Vol. 90, 1995, pp. 181-197.
 - 4) Syswerda, G.: "Uniform Crossover in Genetic Algorithms", Proc. of 3rd Inter. Conf. on Genetic Algorithms, Schaffer, J. D.(ed.), Morgan Kaufmann Publishers, Inc., 1989, pp. 2-9.
 - 5) Spears, W. M., and De Jong, K. A.: "An Analysis of Multi-Point Crossover", Foundations of Genetic Algorithms, Rawlins, G. J. E.(ed.), Morgan Kaufmann Publishers, Inc., 1991, pp. 301-315.
 - 6) Goldberg, D. E.: Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, Mass., 1989.
-