AIAA JOURNAL
Vol. 37, No. 1, January 1999

Optimal Structural Design Based on Self-Similarity

Kazuyuki Hanahara* and Yukio Tada†
Kobe University, Kobe 657-8501, Japan

We discuss an application of self-similarity to structural systems from the viewpoint of optimal design. A general recursive formulation that describes the state of a self-similar structure is provided. On the basis of the formulation, it is also shown that the optimization process can be formulated in a similar recursive manner. In this study, the resource-based approach is adopted as the optimal design procedure; it is implemented based on the proposed recursive formulation. Minimum weight design of two-dimensional truss structures is conducted to demonstrate the feasibility of the proposed approach. The computational cost and the storage requirement are evaluated; the efficiency of the proposed recursive approach and suitability to the parallel computation are also discussed.

Introduction

SELF-SIMILARITY is one of the conspicuous characteristics of nature. Many natural systems exhibit self-similarity in their shape, structure, and sometimes also in their behavior.[1,2] An ordinary tree demonstrates the self-similarity of a natural system well. A branch is a part of a tree but has shape and topology similar to that of the entire tree. It is also well known that the branch has the similarity not only in its appearance but also in its structure and biological function to be a small independent tree itself in an adequate condition. In this article we discuss an application of self-similarity to structural systems from the viewpoint of optimal design. A general recursive formulation of such a self-similar structure is given, and an optimization approach corresponding to the formulation is proposed from the viewpoint of the resource-based approach.[3]

A two-dimensional truss structure is adopted as an example of the optimal design based on self-similarity. Required state variables for the optimization such as mass, stiffness, and so on are expressed in the recursive manner. A number of calculations are conducted as a part of the feasibility study, and the obtained optimal designs are discussed in the context of self-similarity.

The computational cost and the storage requirement for the proposed recursive optimization are evaluated through the truss structure designs. They are compared with the case of the nonrecursive approach based on the same resource-based algorithm, and the advantages of the proposed recursive approach are investigated. The proposed approach is expected to have the intrinsic suitability to parallel computing. A brief evaluation of the computational cost improvement is also discussed.

Recursive Formulation of Self-Similar Structural System

Self-similarity is defined as "the property of objects whereby magnified subsets hold similarity to the whole and to each other."[1] One typical example of such a self-similar structure in nature is an ordinary tree; that is, a branch of a tree has a similar geometry to the entire tree and can be regarded as a small-scale tree itself, which has a number of smaller branches. Another good example is a quilt such as shown in Fig. 1. Figures 1b and 1c demonstrate its self-similarity well; that is, a number of nine-patch quilts as that shown in Fig. 1b make up a larger nine-patch quilt shown in Fig. 1c. In this section we introduce a general formulation of a self-similar structure. A simplified notation based on the formulation is also given from the readability point of view.

General Formulation

A self-similar structure consists of a certain number of similar but small-scale substructures. Such a hierarchical structure can be suitably formulated in terms of a recursive function, that is, a function that refers the function itself. The mathematical formulation of self-similar structure is given in the following general form:

$$\boldsymbol{R}^N = \boldsymbol{R}\left(\boldsymbol{R}_1^{N-1}, \ldots, \boldsymbol{R}_M^{N-1}\right) \tag{1}$$

$$\boldsymbol{R}^n_{i_{N-1}\ldots i_n} = \boldsymbol{R}\left(\boldsymbol{R}^{n-1}_{i_{N-1}\ldots i_n 1}, \ldots, \boldsymbol{R}^{n-1}_{i_{N-1}\ldots i_n M}\right) \qquad (n = N-1, \ldots, 2) \tag{2}$$

The superscript, such as n, $n-1$ or $N-1$, expresses the scale order of the substructure (the number of hierarchy layers); the scale order of the entire structure is denoted by N. Subscript string $i_{N-1}\ldots i_n$ is the identifier for each substructure of scale order n, where i_n indicates that it is the i_nth immediate substructure of the substructure $i_{N-1}\ldots i_{n+1}$ whose scale order is $n+1$. The state variable of the substructure $i_{N-1}\ldots i_n$ is expressed as $\boldsymbol{R}^n_{i_{N-1}\ldots i_n}$. For example, $\boldsymbol{R}^3_{12}$ is the state variable of the substructure 12 of scale order 3, which is the second immediate substructure of the substructure 1 of scale order 4, which is the first immediate substructure of the entire structure of scale order 5.

A substructure of scale order 1 is the primitive component and represented by

$$\boldsymbol{R}^1_{i_{N-1}\ldots i_1} = \boldsymbol{R}^1\left(\boldsymbol{D}_{i_{N-1}\ldots i_1}\right) \tag{3}$$

where the vector $\boldsymbol{D}_{i_{N-1}\ldots i_1}$ represents the design variable of primitive component $i_{N-1}\ldots i_1$. Constant M describes the number of substructures of scale order $n-1$, which make up a substructure of scale order n; it is referred to as the coordination number. Accordingly, the material entity of the entire self-similar structure of scale order N becomes an assemblage of M^{N-1} primitive components. For the example of the nine-patch quilt shown in Fig. 1, the coordination number is $M = 9$. The nine-patch quilt shown in Fig. 1b has 9 blocks, and the larger one shown in Fig. 1c has $9^2 = 81$ blocks.

Equations (1) and (2) represent the self-similarity by means of the recursive function $\boldsymbol{R}$. Although the structure has M^{N-1} design variables as a whole, the two functions $\boldsymbol{R}$ and $\boldsymbol{R}^1$ describe all of the structural properties irrespective of the scale.

Introducing Simplified Notation

The formulation for a self-similar structural system introduced earlier is a generalized and detailed one; however, the notation such as $\boldsymbol{R}^n_{i_{N-1}\ldots i_n}$ uses many subscripts and is cumbersome. From the viewpoint of readability, we introduce a simplified notation listed in Table 1, which suppresses the use of the subscripts. On the basis of this simplified notation, Eqs. (1–3) are rewritten as the following two equations:

$$\boldsymbol{R}^n = \boldsymbol{R}\left(\boldsymbol{R}_1^{n,-1}, \ldots, \boldsymbol{R}_M^{n,-1}\right) \qquad (n = N, \ldots, 2) \tag{4}$$

$$\boldsymbol{R}^1 = \boldsymbol{R}^1(\boldsymbol{D}) \tag{5}$$

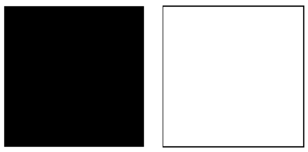
Received March 30, 1998; revision received Sept. 15, 1998; accepted for publication Sept. 16, 1998. Copyright © 1998 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

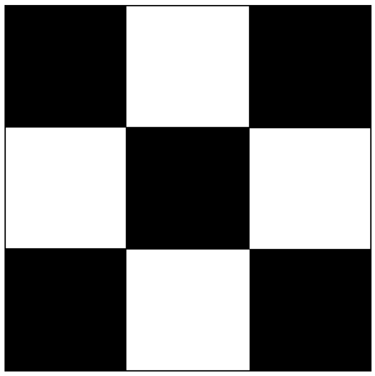
*Lecturer, Department of Computer and Systems Engineering, Rokkodai, Nada. Member AIAA.

†Professor, Department of Computer and Systems Engineering, Rokkodai, Nada.

Table 1 Simplified notation

Simplified	Original	Comments
Basic rules		
$\boldsymbol{R}^n$	$\boldsymbol{R}^n_{i_{N-1}\dots i_n}$	The identifier subscripts $i_{N-1}\dots i_n$ are omitted
$\boldsymbol{R}^{n,-1}_i$	$\boldsymbol{R}^{n-1}_{i_{N-1}\dots i_n i}$	Relative notation for an immediate substructure referring to $\boldsymbol{R}^n$ $(=\boldsymbol{R}^n_{i_{N-1}\dots i_n})$
$\boldsymbol{R}^n_{(a)}$	$\boldsymbol{R}^n_{i_{N-1}\dots i_n(a)}$	Element a of $\boldsymbol{R}^n$ $[\boldsymbol{R}^n_{(a)} \neq \boldsymbol{R}^{n,-1}_a]$
$\boldsymbol{D}$	$\boldsymbol{D}_{i_{N-1}\dots i_1}$	Variable for a primitive component
Examples		
$\boldsymbol{R}^{n,-1}_{i(a)}$	$\boldsymbol{R}^{n-1}_{i_{N-1}\dots i_n i(a)}$	Element a of $\boldsymbol{R}^{n,-1}_i$
$\boldsymbol{R}^{n-1,-1}_i$	$\boldsymbol{R}^{n-2}_{i_{N-1}\dots i_{n-1} i}$	Relative notation for subtruss of scale order $n-2$, referring to $\boldsymbol{R}^{n-1}$ $(=\boldsymbol{R}^{n-1}_{i_{N-1}\dots i_{n-1}})$
$\boldsymbol{R}^{n,-2}_{ij}$	$\boldsymbol{R}^{n-2}_{i_{N-1}\dots i_n ij}$	Relative notation for subtruss of scale order $n-2$, referring to $\boldsymbol{R}^n$ $(=\boldsymbol{R}^n_{i_{N-1}\dots i_n})$
$\boldsymbol{D}_{(a)}$	$\boldsymbol{D}_{i_{N-1}\dots i_1(a)}$	Element a of $\boldsymbol{D}$ $(=\boldsymbol{D}_{i_{N-1}\dots i_1})$


a) Colored and plain blocks ($N = 1$)


b) Nine-patch quilt ($N = 2$)

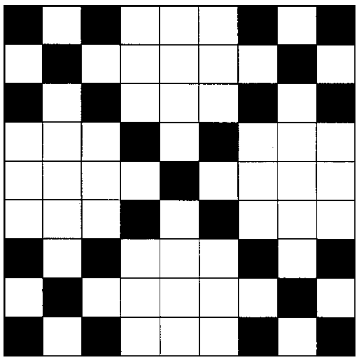

c) Larger nine-patch quilt ($N = 3$)

Fig. 1 Nine-patch quilt as a self-similar structure (M = 9).

Note that Eq. (5) expresses not a relation for all of the primitive components but a relation for every primitive component in this simplified notation. Unless otherwise stated, this simplified notation is used in the following.

Recursive Optimization Approach

Framework in General Form

The criterion function that evaluates the state of the entire self-similar structure can be formulated in the following general form:

$$g = g(\boldsymbol{R}^N) \tag{6}$$

We also introduce the following variable $\boldsymbol{Q}^N$, which represents the information required for the improvement of the structure and is transferred to all of the substructures and the primitive components:

$$\boldsymbol{Q}^N = \boldsymbol{Q}^*(\boldsymbol{R}^N, \boldsymbol{T}) \tag{7}$$

where $\boldsymbol{T}$ denotes the mechanical or geometrical condition assigned to the entire structure. Equation (7) represents only the general form; the contents of the information $\boldsymbol{Q}^N$ depend on the criterion function as well as the optimization algorithm.

Referring to Eqs. (4) and (5), the general framework of the optimization process for the self-similar structure with respect to the design variables $\boldsymbol{D}$ can be written in the following recursive form:

$$\boldsymbol{Q}^{n,-1}_i = \boldsymbol{Q}_i\left(\boldsymbol{Q}^n, \boldsymbol{R}^{n,-1}_1, \dots, \boldsymbol{R}^{n,-1}_M\right)$$

$$(i = 1, \dots, M, \quad n = N, \dots, 2) \tag{8}$$

$$\boldsymbol{D} \leftarrow \boldsymbol{Q}^1(\boldsymbol{Q}^1, \boldsymbol{D}) \tag{9}$$

The variable $\boldsymbol{Q}^n$ is the intermediate information used for the improvement of the corresponding substructure. The information $\boldsymbol{Q}^N$ obtained as Eq. (7) is then distributed to all of the substructures and the primitive components, by means of the function $\boldsymbol{Q}_i$ $(i = 1, \dots, M)$ in Eq. (8). Information $\boldsymbol{Q}^1$ obtained at a primitive component determines the new value of the corresponding design variable $\boldsymbol{D}$ by means of the function $\boldsymbol{Q}^1$ as Eq. (9).

Many of the optimization approaches, including the following resource-based approach, require iterative calculations. State variables of the substructure and the entire structure are updated by the function $\boldsymbol{R}^1$ and $\boldsymbol{R}$ based on the improved design variables $\boldsymbol{D}$ of all of the primitive components, as Eqs. (5) and (4); the information $\boldsymbol{Q}^N$ is calculated based on the updated state variable $\boldsymbol{R}^N$ as Eq. (7) and again distributed to the primitive components through the substructures for further improvement. The iterative process continues until convergence.

Resource-Based Optimal Design

Structural optimization taking account of not only shape but also topology is an important field of recent engineering design. Energy-based or functional-based approaches typified by such methods as the homogenization method are promising candidates of this field.[4] One of the different approaches is the resource-based approach used for discrete structures[3]; that is developed based on an object-oriented standpoint and is expected to be suitable for dealing with a structure consisting of a number of parts. From this point of view, we adopt this resource-based approach for the recursive optimization.

The essence of the approach is summarized as follows: 1) Each of the structural elements evaluates its critical resource with respect to the local constraints under the given condition. 2) Each element reduces its resource to the critical. 3) Repeat steps 1 and 2 until the entire structure attains a certain equilibrium state under the given condition.

From the viewpoint of this resource-based algorithm, the following abstract quantities are assigned to the variables g, $\boldsymbol{R}^n$, and $\boldsymbol{Q}^n$ that appear in Eqs. (4–9). 1) Criterion g represents the total resource of the entire structure. 2) The state variable $\boldsymbol{R}^n$ consists of two parts. The first part is the resource used by the corresponding substructure, and the second part denotes its performance. 3) Variable $\boldsymbol{Q}^n$ denotes the activity or contribution of the corresponding substructure under the given condition. The concrete examples of these quantities are given in the next section.

The resource-based recursive optimization is performed in the following procedure:

1) Set an initial value to design variables $\boldsymbol{D}$.

2) Calculate all of the state variables (i.e., resource and performance) of primitive components, substructures, and the entire structure, $\boldsymbol{R}^1$, $\boldsymbol{R}^n$, and $\boldsymbol{R}^N$, based on the current design variables as Eqs. (4) and (5).

3) Calculate the activity of the entire system $\boldsymbol{Q}^N$ under the given condition $\boldsymbol{T}$ as Eq. (7).

4) Assign activity to all of the primitive components through the substructures as Eq. (8).

5) For every primitive component, update the design variable $\boldsymbol{D}$ so as to achieve the minimum resource based on the corresponding required activity $\boldsymbol{Q}^1$ [Eq. (9)].

6) Repeat steps 2–5 until convergence.

It is noted that the local constraint condition of the optimization problem is implicitly implemented in the function $\boldsymbol{Q}^1$ and evaluated at step 5 in the procedure.

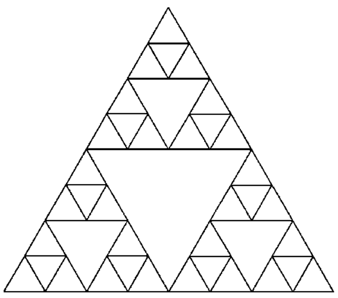
Minimum Weight Design of Truss Structure

Topology of Self-Similar Truss

The recursive approach requires the self-similarity of the structure. Reports published earlier deal with an adaptive truss structure[5] or a mechanical system[6] that has the explicit self-similarity and the topology as shown in Fig. 2a. For the truss structure shown in Fig. 2a, it is immediately possible to describe the structural property or behavior in the recursive fashion expressed as Eqs. (4) and (5) by using the three vertices as the representative nodes of the triangular truss at every scale. In this case the size of the variables and the detailed expression of the recursive functions become completely the same irrespective of the scale order of the truss. However, a truss structure having such a topology as shown in Fig. 2a is not generally applicable for an engineering use.

On the other hand, the truss structure shown in Fig. 2b has a relatively general topology. Self-similarity of the truss is not clear. It is, however, possible to introduce self-similarity to the truss structure to some extent as a matter of form, regarding it as an assemblage of four small-scale square trusses that are also assemblages of smaller square trusses. In this case, to describe the recursive functions in the form as expressed in Eqs. (4) and (5), the compatibility at the connection of the trusses of smaller scale order must be taken into account. Therefore, the state variables contain not only the quantities about the corner nodes but also the quantities about the internal nodes at the connection of the smaller trusses. Accordingly, the size of the variables and the detailed expression of the recursive functions become slightly different in accordance with the scale order of the truss.

In this study we deal with two-dimensional truss structures of such a topology as that shown in Fig. 2b. The primitive components shown in Fig. 3 are adopted, where L denotes the number of segments along


a) Explicitly self-similar truss ($N = 4$ and $M = 3$)

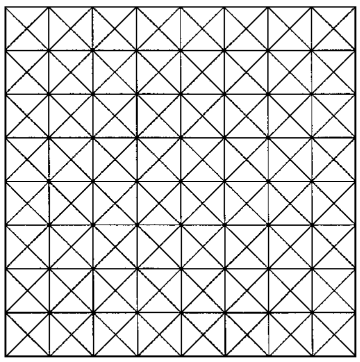
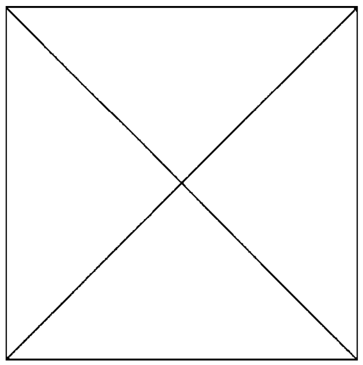
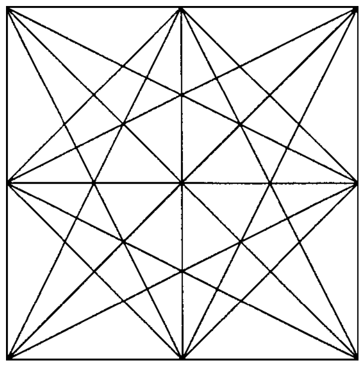
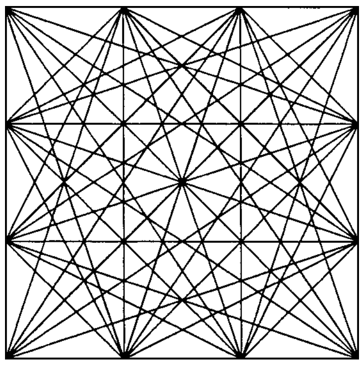

b) Truss structure of general topology ($N = 4$ and $M = 4$)

Fig. 2 Self-similar truss structure.


a) $L = 1$


b) $L = 2$


c) $L = 3$

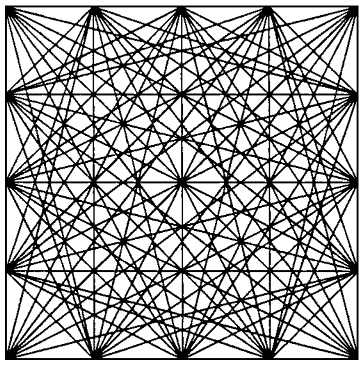

d) $L = 4$

Fig. 3 Primitive component of truss structure.

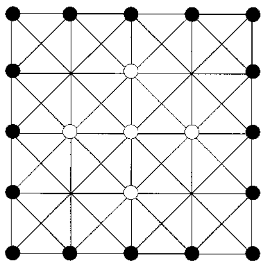


Fig. 4 Example of truss to be optimized ($N = 3$, $M = 4$, and $L = 1$): ●, peripheral nodes, and ○, internal nodes (peripheral nodes at the immediate subtruss of scale order, 2).

a side of the square truss. All peripheral nodes within a primitive component are assumed to be interconnected by truss members. Figure 4 shows an example of a truss structure to be optimized, whose scale order $N = 3$, coordination number $M = 4$, and number of segments in the primitive $L = 1$.

Formulation of Weight-Minimization Problem

The proposed resource-based recursive approach is applied to the following problem based on the aforementioned two-dimensional trusses: minimize total mass with respect to cross section of members subject to member stress constraints. The resource, performance, and activity for the resource-based optimization are now expressed by the following concrete quantities, respectively: resource—total mass, the sum of the mass of all constituent truss members; performance—peripheral nodal stiffness matrix, the stiffness matrix with respect to the peripheral nodes of the truss only, and activity—peripheral nodal displacements, the deformation assigned to the truss.

The mechanical and geometrical conditions are imposed on the peripheral nodes of the truss; that is, a part of the peripheral nodes experiences given nodal force and the other part is geometrically constrained. The cross sections of truss members are selected as the design variables and the stress of each truss member is the local constraint condition. The buckling of members and the body force such as the gravitational force are not taken into account in this study.

The state variable is represented as

$$\boldsymbol{R}^n = \{m^n, \boldsymbol{K}^n_{(P)}\} \tag{10}$$

where m^n denotes the total mass of the corresponding subtruss and $\boldsymbol{K}^n_{(P)}$ is its peripheral nodal stiffness matrix. The design variable is represented as

$$\boldsymbol{D} = \{A_{(1)}, \ldots, A_{(J)}\} \tag{11}$$

where $A_{(j)}$ is the cross-sectional area of member j of the corresponding primitive component and $J = {}_{4L}C_2$ is the number of truss members within a primitive component. [All of the $4L$ peripheral nodes are interconnected by truss members. Notation ${}_nC_r$ expresses the combination of n objects taken r at a time; that is, ${}_nC_r = n!/r!(n-r)!$] The information variable is represented as

$$\boldsymbol{Q}^n = \{\boldsymbol{U}^n_{(P)}\} \tag{12}$$

where $\boldsymbol{U}^n_{(P)}$ is the peripheral nodal displacement vector of the corresponding subtruss.

The recursive function $\boldsymbol{R}$ for the problem is then expressed in terms of m^n and $\boldsymbol{K}^n_{(P)}$ as

$$m^n = \sum_{i=1}^{M} m_i^{n,-1} \tag{13}$$

$$\boldsymbol{K}^n_{(P)} = \underline{\boldsymbol{K}}^n_{(PP)} - \underline{\boldsymbol{K}}^n_{(PI)}\left[\underline{\boldsymbol{K}}^n_{(II)}\right]^{-1}\underline{\boldsymbol{K}}^n_{(IP)} \tag{14}$$

where $M = 4$ for all of the examples considered here and

$$\underline{\boldsymbol{K}}^n = \begin{bmatrix} \underline{\boldsymbol{K}}^n_{(PP)} & \underline{\boldsymbol{K}}^n_{(PI)} \\ \underline{\boldsymbol{K}}^n_{(IP)} & \underline{\boldsymbol{K}}^n_{(II)} \end{bmatrix} \tag{15}$$

is obtained as the superposition of the peripheral nodal stiffness matrices of the immediate subtrusses, $\boldsymbol{K}^{n,-1}_{i(P)}$ $(i = 1, \ldots, M)$. The subscripts in the parentheses of the submatrices in Eq. (15) describe that their rows or columns are corresponding to the peripheral nodes (P) or to the internal nodes (I) of the corresponding subtruss of scale order n. It is noted that these are the peripheral nodes of the immediate subtrusses of scale order $n-1$. Figure 4 shows the peripheral and internal nodes of a subtruss of scale order 3 in the case of $L = 1$.

The function $\boldsymbol{R}^1$ that describes the behavior of primitive components is represented by m^1 and $\boldsymbol{K}^1_{(P)}$ in terms of $\boldsymbol{D}$ as

$$m^1 = \sum_{j=1}^{J} \rho A_{(j)} L_{(j)} \tag{16}$$

$$\boldsymbol{K}^1_{(P)} = \Lambda^T \boldsymbol{K}^L \Lambda \tag{17}$$

$$\boldsymbol{K}^L = \operatorname{diag}\left[\frac{EA_{(j)}}{L_{(j)}} \quad (j = 1, \ldots, J)\right] \tag{18}$$

where ρ and E are the material density and Young's modulus. (For the primitive components adopted in this study as shown in Fig. 3, all of the truss nodes are peripheral.) The length $L_{(j)}$ of member j and the coordinate transformation matrix Λ, which is the same matrix as the one generally used in the finite element method analysis of a truss structure, are common to all of the primitive components in this example.

The criterion function is simply expressed as

$$g(\boldsymbol{R}^N) = m^N \tag{19}$$

The function $\boldsymbol{Q}^*$ that calculates the information variable for the entire structure is written in terms of the nodal force acting on the free peripheral nodes $\boldsymbol{F}_{(\sigma)}$ as

$$\boldsymbol{U}^N_{(P)} = \begin{bmatrix} \boldsymbol{U}^N_{(\sigma)} \\ \boldsymbol{U}^N_{(u)} \end{bmatrix} \tag{20}$$

$$\boldsymbol{U}^N_{(\sigma)} = \left[\boldsymbol{K}^N_{(\sigma)}\right]^{-1} \boldsymbol{F}_{(\sigma)} \tag{21}$$

$$\boldsymbol{U}^N_{(u)} = 0 \tag{22}$$

where $\boldsymbol{U}^N_{(\sigma)}$ is the part of the peripheral nodal displacement vector $\boldsymbol{U}^N_{(P)}$ for the free nodes that experience the nodal force $\boldsymbol{F}_{(\sigma)}$ and $\boldsymbol{U}^N_{(u)}$ is the part for the fixed nodes. The stiffness matrix $\boldsymbol{K}^N_{(\sigma)}$ corresponding to the free nodes is obtained as the submatrix of the peripheral nodal stiffness matrix of the entire truss structure $\boldsymbol{K}^N_{(P)}$.

The recursive function $\boldsymbol{Q}_i$ $(i = 1, \ldots, M)$ that distributes the information required for optimization is expressed in terms of $\boldsymbol{U}^n_{(P)}$, which is the peripheral nodal displacements of the corresponding subtruss, as

$$\boldsymbol{U}^{n,-1}_{i(P)} = \begin{bmatrix} \text{Collection of displacements corresponding to} \\ \text{the peripheral nodes of } i\text{th subtruss within } \boldsymbol{U}^n \end{bmatrix} \tag{23}$$

where

$$\boldsymbol{U}^n = \begin{bmatrix} \boldsymbol{U}^n_{(P)} \\ \boldsymbol{U}^n_{(I)} \end{bmatrix} \tag{24}$$

and $\boldsymbol{U}^n_{(I)}$ is the internal nodal displacements of the subtruss obtained as

$$\boldsymbol{U}^n_{(I)} = \left[\underline{\boldsymbol{K}}^n_{(II)}\right]^{-1} \underline{\boldsymbol{K}}^n_{(IP)} \boldsymbol{U}^n_{(P)} \tag{25}$$

It is noted that a nodal displacement vector $\boldsymbol{U}^n$, for a subtruss of scale order n, consists of the peripheral nodal displacements of its immediate subtrusses, $\boldsymbol{U}^{n,-1}_{i(P)}$ $(i = 1, \ldots, M)$.

Modification of a design variable $\boldsymbol{D}$, represented by the function $\boldsymbol{Q}^1$, is expressed in terms of the peripheral nodal displacements of the corresponding primitive component $\boldsymbol{U}^1_{(P)}$. The member force vector $\boldsymbol{P} = [P_{(1)}, \ldots, P_{(J)}]^T$ is calculated as

$$\boldsymbol{P} = \boldsymbol{K}^L \boldsymbol{\Lambda} \boldsymbol{U}^1_{(P)} \tag{26}$$

The design variable is then updated so as to minimize the cross section against the member force as follows:

$$A_{(j)} \leftarrow \left| \frac{P_{(j)}}{\underline{\sigma}} \right| \qquad (j = 1, \ldots, J) \tag{27}$$

where $\underline{\sigma}$ is the allowable stress of the material. The constraint condition of the optimization problem is implicitly implemented as Eq. (27).

Unification of Multiple Design Variables

For the self-similar truss structure discussed here, neighboring primitive components have different design variables assigned to identical members from the viewpoint of the entire truss. In other words, such truss members have multiple design variables over the neighboring primitive components. A kind of unification process is required after the recursive optimization to obtain the truss design in the general form, that is, the design expressed in terms of nodes, each of which has a unique nodal position, and members, each of which uniquely connects two of those nodes.

The unification for a member of multiple design variables is performed so as to meet the compatibility and equilibrium conditions. In the case of a truss consisting of members of a uniform material discussed earlier, all cross sections corresponding to an identical member are accumulated to obtain a single cross section as a unique member. It is noted that the equality of such multiple design variables corresponding to an identical member is not guaranteed.

Reduction of Computational Cost

The formulated approach deals with a large number of design variables, that is, $M^{N-1}{}_{4L}C_2$ truss members; for example, the truss shown in Fig. 4 has 96 design variables. To reduce the computational cost, the following optional processes are employed during the implementation of the basic resource-based recursive optimization procedure described earlier.

1) A truss member that attains a sectional area less than a specified limit at step 5 is regarded as degenerated and will be ignored in the subsequent calculations.

2) A truss node where all of the connected members are found degenerated at step 2 is regarded as degenerated and will be ignored in the subsequent calculations.

3) A primitive component all of whose members are found degenerated at step 2 is regarded as degenerated and will be ignored in the subsequent calculations.

4) A subtruss all of whose constituent smaller trusses are found degenerated at step 2 is regarded as degenerated and will be ignored in the subsequent calculations.

The degeneration of a truss member is performed by setting its cross-sectional area to a number that is small enough not to affect the result and large enough to avoid an unexpected singularity at the numerical computation of the stiffness matrix in Eq. (14).

It is noted that one of the advantageous points of the proposed recursive optimization approach is the simple and effective reduction of computational cost based on the self-similar hierarchical structure.

Examples of Truss Structure Design

The conventional structural design problem shown in Fig. 5 is adopted for the demonstration of the feasibility of the proposed optimization approach. The truss structure is supported by the left-hand wall at the two fixed nodes and holds a downward load $F = 100$ kN at a distance $l = 1$ m from the wall by the center node at the right-hand end. The values of material density, allowable stress, and Young's modulus are assumed to be 10,000 kg/m^3, 100 MPa, and 200 GPa, respectively. The fundamental requirement for the proposed approach is not the square shape of the truss but the topology and the recursive configuration of the truss. Not square but rectangular primitive components are adopted in the following examples to meet the specified positions of fixed and loaded nodes for different numbers of segments and scale orders L and N.

Table 2 Recursive optimization results

Design			Members		Weight,	Iterations	CPU time	t/T,
Fig. 6	N	L	Initial[a]	Final[b]	kg	T	t, s	s
a)	3	2	448	64	44.77	371	20	0.054
b)	4	2	1,792	230	44.53	2,471	952	0.385
c)	5	2	7,168	532	44.27	4,417	15,710	3.557
d)	4	1	384	86	47.76	1,527	75	0.049
e)	4	3	4,224	428	44.07	5,298	6,730	1.270
f)	4	4	7,680	544	43.81	3,694	12,561	3.400

[a]Number of design variables is equal to the initial number of members.
[b]Members of adjacent primitive components that share both nodes are unified after the convergence.

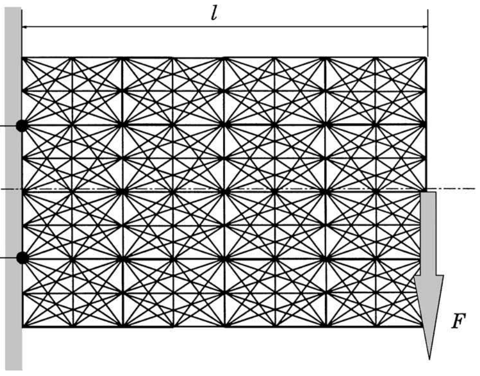


Fig. 5 Initial truss structure (N = 3, M = 4, and L = 2).

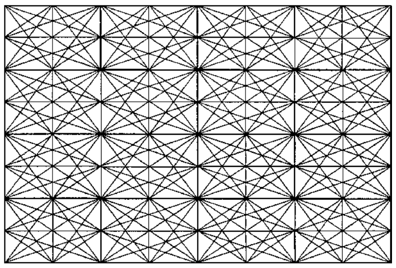
The initial value for the sectional area of members is 1 mm^2. The limit sectional area of members to be degenerated is 1×10^{-3} mm^2, and the value assigned for the degenerated members is 1×10^{-9} mm^2. The convergence of the optimization process is determined by either one of the following two conditions: one is to attain the specified value for the change in total mass of less than 1×10^{-9} kg for two successive iterations, and the other is the occurrence of no additional degenerated members for 1000 successive iterations.

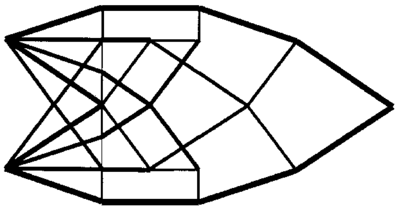
Figure 6 and Table 2 show the optimization results for various initial conditions. Figures 6a–6c show the difference among various scales ($N = 3$, 4, and 5 and $L = 2$), and Figs. 6d, 6b, 6e, and 6f show the difference among various numbers of segments ($L = 1$, 2, 3, and 4 and $N = 4$). Compared with the weight of 61.67 kg achieved by the simple two-bar truss structure, the optimality of these results is clearly confirmed. Comparison of Figs. 6c and 6e is interesting because the optimization result of Fig. 6e is better than that of Fig. 6c, even though its number of design variables is less than that of Fig. 6c. These results demonstrate well the importance of the selection of the primitive component.

The calculations shown in Table 2 are performed in the recursive manner based on the implicit self-similarity of the truss. However, the operation actually performed may be done based on the nonrecursive resource-based approach, which deals with the entire truss structure as a whole without any partitioning. The optimization results obtained based on such a nonrecursive approach are shown in Table 3. The results are practically the same as the results based on the recursive approach in Table 2. Nevertheless, the resultant optimal truss structures in Fig. 6 seem to exhibit a few similar patterns of various scales. For example, in the obtained structure shown in Fig. 6f, we see some small patterns similar to the shape and topology of the entire truss in various scales. This indicates that the self-similar approach has the possibility to be a more efficient optimization algorithm by taking these self-similar patterns into account.

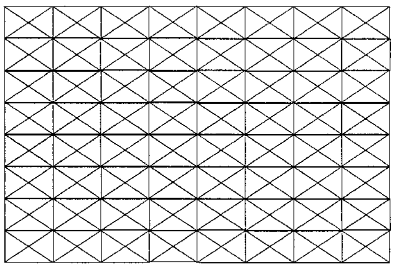
Computational Cost and Storage Requirement

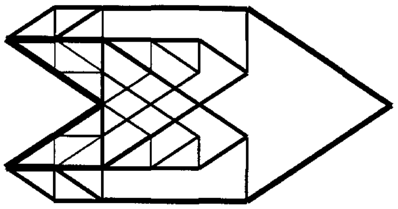
The required computational cost and storage capacity for the recursive and nonrecursive approaches are roughly evaluated and


(Initial structure)

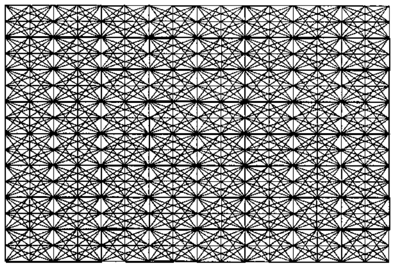

(Obtained structure)

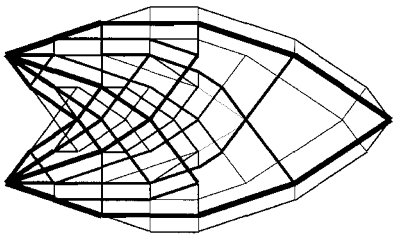
a) ***N*** **= 3,** ***L*** **= 2, total weight: 44.77 kg**


(Initial structure)

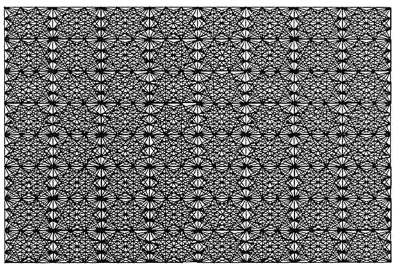

(Obtained structure)

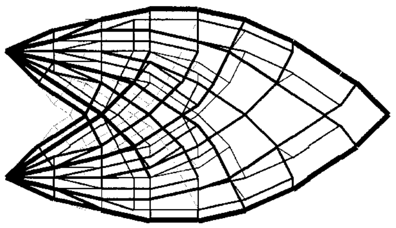
d) ***N*** **= 4,** ***L*** **= 1, total weight: 47.76 kg**


(Initial structure)

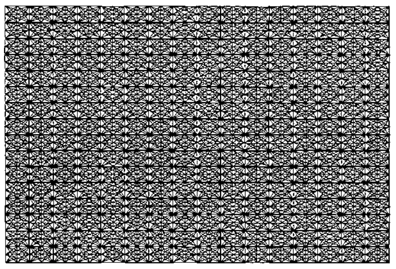

(Obtained structure)

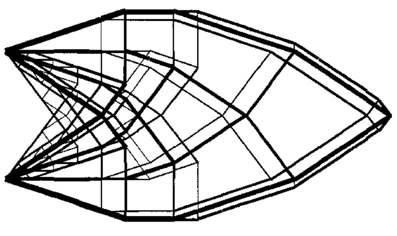
b) ***N*** **= 4,** ***L*** **= 2, total weight: 44.53 kg**


(Initial structure)

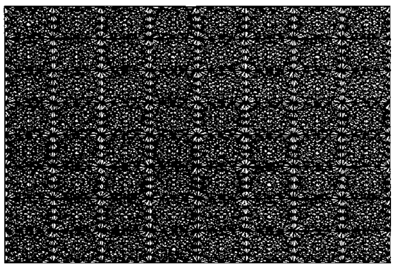

(Obtained structure)

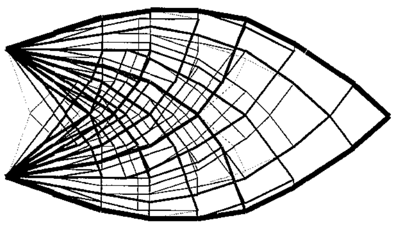
e) ***N*** **= 4,** ***L*** **= 3, total weight: 44.07 kg**


(Initial structure)


(Obtained structure)

c) ***N*** **= 5,** ***L*** **= 2, total weight: 44.27 kg**


(Initial structure)


(Obtained structure)

f) ***N*** **= 4,** ***L*** **= 4, total weight: 43.81 kg**

Fig. 6 Minimum weight trusses for various scales (*N*) and numbers of segments (*L*) (*M* = 4).

Table 3 Nonrecursive optimization results

Design			Members		Weight,	Iterations	CPU time,	t/T,
Fig. 6	N	L	Initial[a]	Final	kg	T	t, s	s
a)	3	2	424	66	44.77	370	12	0.032
b)	4	2	1,680	230	44.53	2,471	1,489	0.603
c)	5	2	6,688	532	44.27	4,416	331,539	75.08
d)	4	1	328	106	47.76	529	25	0.047
e)	4	3	4,056	428	44.07	5,296	18,951	3.578
f)	4	4	7,456	544	43.81	3,693	47,215	12.78

[a]The difference from the recursive approach is due to the unifying process performed before the optimization.

compared through the optimal designs of a truss structure calculated in the preceding section. The suitability of the proposed recursive optimization to parallel computing is also pointed out.

Computational Cost

For the resource-based recursive approach, the computational power is dominantly consumed at the calculation of Eqs. (14), (21), and (25). The computational cost of such square matrix operations is roughly approximated as $\mathcal{O}(D^3)$, where D denotes the dimension of matrix. Because the recursive approach only takes account of the peripheral nodes, the dimension of the stiffness matrix is $2^n 4L$ for a two-dimensional subtruss of scale order n. The computational cost can then be approximated as $\mathcal{O}(2^{3n}64L^3)$ and rewritten in the following form:

$$C(n) = 2^{3n}\alpha \tag{28}$$

where $C(n)$ is the computational cost for the matrix operation of each subtruss of scale order n and α is the proportional coefficient. The total computational cost in each iteration is then evaluated as

$$C_{\text{Total}}(N) = \sum_{n=1}^{N} 4^{N-n}C(n) = \sum_{n=1}^{N} 4^{N-n}2^{3n}\alpha$$
$$= 2^{2N}\alpha\sum_{n=1}^{N} 2^n \simeq 2^{2N}2^{N+1}\alpha = 8^N 2\alpha \tag{29}$$

Equation (29) states that the computational cost of each iteration is proportional to 8^N. In other words, the computation for the self-similar truss of scale order $N+1$ costs eight times more than that for the case of scale order N. Comparing the values of t/T in Table 2 for Figs. 6a–6c, it is confirmed that this approximation is reasonable.

On the other hand, the nonrecursive approach takes account of all nodes of the truss structure; that is, the dimension of the stiffness matrix is approximated as $2^{2N}L/2$, and the computational cost can be evaluated as $\mathcal{O}(2^{6N}L^3/8)$ and rewritten as

$$C'_{\text{Total}}(N) = 2^{6N}\beta = 64^N\beta \tag{30}$$

where β is the proportional coefficient for the nonrecursive case. The equation states that the computational cost for each iteration is proportional to 64^N and the computation for the truss of scale order $N+1$ costs 64 times more than that for the case of scale order N. Comparison of the values t/T for Figs. 6a–6c in Table 3 does not precisely meet this approximation; however, it can be confirmed that the evaluation is qualitatively correct and the computational cost of the nonrecursive approach becomes higher far more rapidly than the case of the recursive approach, in accordance with the increase of scale order N.

The computational cost evaluation conducted here is highly approximate because it does not take into account schemes for reducing computational cost such as using banded-matrix solution methods. However, it can be expected that the proposed recursive approach is useful to reduce a significant amount of computational cost in the case of the resource-based optimization such as demonstrated in the preceding section.

Storage Requirement

The required storage capacity for the recursive approach is evaluated here in terms of the capacities of the stiffness matrices. The number of elements in the stiffness matrix for a subtruss of scale order n is expressed as

$$S(n) = (2^n 4L)^2 = 2^{2n}16L^2 \tag{31}$$

because the dimension of the stiffness matrix is $2^n 4L$. The total number of elements of all of the stiffness matrices is then calculated as

$$S_{\text{Total}}(N) = \sum_{n=1}^{N} 4^{N-n}S(n) = 4^N 16L^2 \sum_{n=1}^{N} 1 = 4^N N 16L^2 \tag{32}$$

Equation (32) states that the storage requirement for the optimization of a truss of scale order N is proportional to $4^N N$.

On the other hand, the number of elements in the stiffness matrix used in the nonrecursive approach for a truss of scale order N is expressed as

$$S'_{\text{Total}}(N) = [2^{2N}(L/2)]^2 = 16^N(L^2/4) \tag{33}$$

because the dimension of the stiffness matrix is $2^{2N}L/2$. According to the equation, the storage requirement is proportional to 16^N in the case of the nonrecursive approach.

Note again that the evaluation conducted here is an approximate one; however, the proposed recursive approach is also expected to reduce the storage requirement for the truss with large-scale order N in these cases.

Suitability to Parallel Computation

One of the important properties of the proposed optimization approach for a self-similar structure is the intrinsic suitability to parallel computing, coming from the recursive formulation, which naturally has the divide-and-conquer property.[7] Currently, planning an application of such a parallel computing approach is under consideration. In this paper, we discuss a brief evaluation of computational cost improvement for the proposed recursive optimization based on parallel computing.

For the sake of simplicity, we assume 4^H parallel processors having the same performance. Because of the divide-and-conquer property, the calculation for the 4^H subtrusses of scale order $N-H$, including the calculation for their lower subtrusses, is expected to be performed in a fully parallel manner; the corresponding computational cost is the same as that for a single entire truss of scale order $N-H$, that is, $C_{\text{Total}}(N-H)$. (In this section the term "computational cost" just expresses the required computation time even in the case of parallel computation.) Referring to Eqs. (28) and (29), the total cost in the case of the 4^H parallel processors can then be evaluated as

$$C^{4**H}_{\text{Total}}(N) = C_{\text{Total}}(N-H) + \sum_{n=0}^{H-1} \frac{1}{\gamma(4^{H-n})}C(N-n) \tag{34}$$

where $\gamma(p)$ expresses the acceleration coefficient of an ordinary (nonrecursive) matrix operation by means of p parallel processors; that is, $\gamma(p) = p$ in the case for which the linear acceleration is expected, and $\gamma(p) = 1$ in the case for which no acceleration is expected.

For example, if we use four processors ($H = 1$), the computational cost is evaluated as

$$C^{4**1}_{\text{Total}}(N) = 8^{N-1}2\alpha + \frac{1}{\gamma(4)}8^N\alpha = \frac{\gamma(4)+4}{8\gamma(4)}8^N 2\alpha \tag{35}$$

Compared with Eq. (29), the obtained cost is reduced to one-quarter in the case of linear acceleration as a matter of course and five-eighthseven in the case of no acceleration. This indicates that on the basis of the recursive approach about a 37% speedup is assured without any difficulties by means of four parallel processors.

Concluding Remarks

In this paper we discussed an optimization approach for a structural system based on self-similarity. The mathematical formulation of a self-similar structure is described in the general form. The recursive optimization corresponding to the formulation is proposed, and the resource-based approach is applied in this context.

The proposed optimization is applied to the two-dimensional truss design problem. Introduction of implicit self-similarity to the truss of general topology is discussed; the obtained results demonstrate the feasibility of the proposed optimal design approach. The computational cost and the storage requirement are approximately evaluated, and the advantage of the proposed approach is confirmed. The suitability of the recursive optimization to parallel computing is also discussed.

References

[1]Mandelbrot, B. B., *The Fractal Geometry of Nature*, W. H. Freeman and Co., New York, 1977, Chap. 6.

[2]Vicsek, T., *Fractal Growth Phenomena*, World Scientific, Singapore, 1989, Chap. 1.

[3]Miki, M., "Object-Oriented Optimization of Discrete Structures," AIAA Paper 94-1595, April 1994.

[4]Bendsøe, M. P., *Optimization of Structural Topology, Shape, and Material*, Springer–Verlag, Berlin, 1995, Chap. 1.

[5]Hanahara, K., and Sugiyama, Y., "The Application of Self-Similarity to a Large Scale Adaptive System: Proposal of Recursive Architecture," *Smart Materials and Structures*, Vol. 4, 1995, pp. 186–194.

[6]Kokaji, S., "A Fractal Mechanism and Decentralized Control Method," *Proceedings of the USA–Japan Symposium on Flexible Automation*, Vol. 2, Inst. of Systems, Control and Information Engineers, Kyoto, Japan, 1990, pp. 1129–1134.

[7]Sedgewick, R., *Algorithms*, 2nd ed., Vol. 1, Addison–Wesley, Reading, MA, 1988, Chap. 5.

A. Chattopadhyay
Associate Editor