ELSEVIER

International Journal of Solids and Structures 41 (2004) 2623–2641

INTERNATIONAL JOURNAL OF SOLIDS and STRUCTURES

www.elsevier.com/locate/ijsolstr

Parallelized structural topology optimization for eigenvalue problems

Tae Soo Kim [a], Jae Eun Kim [b,*], Yoon Young Kim [b,*]

[a] *Storage System Division, Samsung Electronics Co., Ltd., South Korea*

[b] *School of Mechanical and Aerospace Engineering and National Creative Initiatives Research Center for Multiscale Design, Seoul National University, San 56-1, Shillim-dong, Kwanak-gu, Seoul 151-742, South Korea*

Received 15 March 2003; received in revised form 11 November 2003

Abstract

A parallel topology optimization method is proposed to deal with large-scale structural eigenvalue-related design problems. While existing parallel methods have been employed mainly for finite element analysis, the present work deals with the entire parallel process of analysis and topology optimization using a parallel processing technique. In the framework of the physical domain decomposition method, the sensitivity analysis and design variable update are performed independently in each subdomain with minimum data communication among subdomains. A successive estimation strategy over design iterations is proposed to expedite the design optimization process. The preconditioned conjugate gradient method and the subspace iteration method are used as parallel solvers. Several numerical examples are considered to show the scalability and effectiveness of the present parallel approach in handling large-scale design problems.

© 2003 Published by Elsevier Ltd.

Keywords: Parallel processing; Topology optimization; Eigenvalue problem; Domain decomposition; Optimality criteria; Subspace iteration method; Successive estimation

1. Introduction

A major bottleneck to apply topology optimization in large-scale practical design problems is the extraordinarily large computation time. Typical optimizers usually require tens to hundreds of iterations, so it is a formidable task to carry out large-scale topology optimization only with a single CPU machine. To overcome this difficulty, parallelizing large-scale topology optimization will be a practical alternative. Current parallelization approaches may be classified into two categories: one category to parallelize numerical analysis itself by means of loop or array partitioning and the other to decompose a domain into a number of subdomains (Papadrakakis (1993, 1997)). The latter is now more popular and generally more effective, so the idea of the domain decomposition method will be used for the present parallelization.

* Corresponding authors. Tel.: +82-2-880-7130/7154; fax: +82-2-872-5431/883-1513.
E-mail addresses: jekim@idealab.snu.ac.kr (J.E. Kim), yykim@snu.ac.kr (Y.Y. Kim).

0020-7683/$ - see front matter © 2003 Published by Elsevier Ltd.
doi:10.1016/j.ijsolstr.2003.11.027

As far as numerical analysis itself is concerned, parallel analysis methods for linear static problems are now almost established (Bitzarakis et al., 1997; Farhat and Wilson, 1988; Johnson and Mathur, 1990; Papadrakakis and Bitzarakis, 1996; Su and Fulton, 1993). However, the research on parallelized methods for eigenvalue problems is still active (Hwang and Parsons, 1994; Katagiri and Kanada, 2001; Lang, 1999; Lui, 1998; Watkins et al., 1996). With regard to structural design optimization problems, various parallel processing ideas such as the parallelization of an optimization algorithm (Padula and Stone, 1998), the combination of the parallel finite element analysis with a serial optimization algorithm (El-Sayed and Hsiung, 1991), structural optimization based on the domain decomposition (Wang et al., 1996) have been proposed. Recently, parallel processing techniques have been applied to structural optimization problems using the zeroth-order optimizer such as genetic algorithms and evolutionary strategies (Thierauf and Cai, 1997; Topping and Leite, 1999).

Though there are many investigations on the parallelization of numerical analysis and some optimization algorithms, the parallelization of topology optimization has been rarely attempted. Borrvall and Petersson (2001) and Kim (2001) have worked out parallelized topology optimization. Borrvall and Petersson (2001) have parallelized the topology optimization for compliance minimization. They employed the preconditioned conjugate gradient method (PCG) and sequential convex programming in the domain decomposition setting. Kim (2001) has parallelized the static analysis by the iterative PCG solver and the eigenvalue analysis by the subspace iteration approach. As an optimization algorithm, Borrvall and Petersson (2001) use the method of moving asymptotes (Svanberg, 1987) while Kim (2001) employs the optimality criteria method.

In the present work, the design domain decomposition method will be employed for efficient parallelization. In the framework of domain decomposition, we will deal with eigenvalue-related topology optimization problems. For eigenvalue analysis, a parallel subspace iteration method effective for decomposed subdomains is proposed. If the parallel subspace iteration method uses the standard PCG method which will also be parallelized, it will take more computation time than a typical direct solver because of multiple right-hand sides. In order to overcome this difficulty, a successive estimation scheme especially suitable for topology optimization is employed. For optimization, we present the parallelized optimality criteria method. The proposed parallel strategies are summarized as boxes and several large-scale problems are considered to check various numerical aspects of the parallelization of topology optimization.

2. Parallelization of eigenvalue analysis

Since we will employ an iterative PCG solver with the subspace iteration method, we begin with the parallelization of the conjugate gradient method.

2.1. Parallelization of the finite element analysis by the conjugate gradient method

The most popular parallel finite element solver is perhaps the conjugate gradient method coupled with an appropriate preconditioner. This method consists mainly of matrix–vector and vector–vector multiplications, so its parallelization is rather straightforward. Fig. 1 illustrates a typical decomposition of a design domain into N_d subdomains, which will be used throughout the present work. To parallelize the solution process of the equilibrium equation $\mathbf{Ku} = \mathbf{F}$ using $\mathbf{u}$ that is defined in the decomposed domain, we decompose $\mathbf{K}$ and $\mathbf{F}$ as

$$\mathbf{K} = \sum_{s=1}^{N_d} \overline{\mathbf{K}}_s \tag{1}$$

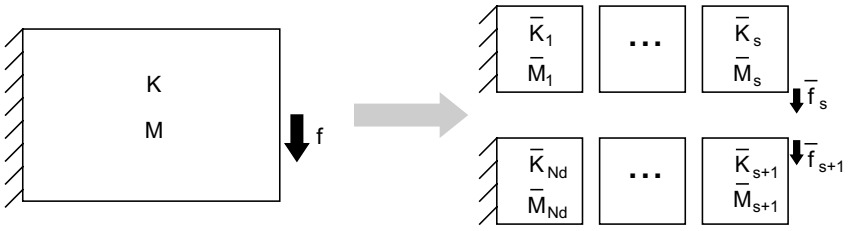

Fig. 1. Illustration of the design domain decomposition into subdomains.

$$\mathbf{F} = \sum_{s=1}^{N_d} \overline{\mathbf{F}}_s \tag{2}$$

where the stiffness matrix $\overline{\mathbf{K}}_s$ and the external load vector $\overline{\mathbf{F}}_s$ are assembled separately for each subdomain. In Eqs. (1) and (2), $\mathbf{K}$ is the global stiffness matrix and $\mathbf{F}$ is the external load vector. The global stiffness matrix is calculated as

$$\mathbf{K} = \sum_{e=1}^{N_e} \mathbf{k}_e = \sum_{e=1}^{N_e} \int_{\Omega_e} \mathbf{B}_e^T \mathbf{C}_e \mathbf{B}_e \, d\Omega \tag{3}$$

where $\mathbf{B}_e$ is the eth strain-interpolation matrix, $\mathbf{C}_e$ is the constitutive matrix and N_e is the number of finite elements used to discretize a domain. In addition, any preconditioning matrix employed for the preconditioned conjugate gradient (PCG) method can be also decomposed as

$$\mathbf{D}^{-1} = \sum_{s=1}^{N_d} \overline{\mathbf{D}}_s^{-1} \tag{4}$$

where the matrix $\mathbf{D}$ is the Jacobi preconditioner (see Angeleri et al., 1989; Coutinho et al., 1991).

The performance of the PCG algorithm is affected by two major factors. The first one is the condition number of the system matrix $\mathbf{K}$ and the other, the initial guess for $\mathbf{u}$. The initial guess for $\mathbf{u}$ is usually set equal to 0 in the PCG algorithm. However, we observe that the approximate solution at the previous optimization iteration shall be a good initial guess for the current optimization iteration in the case of topology optimization problems. This idea is implemented and the numerical performance based on it is compared with that by the standard PCG method using the initial guess $\mathbf{u} = \mathbf{0}$ at the beginning of every optimization iteration. It is noted that the optimization using the method requires much smaller numbers of the PCG iterations at later design iterations in comparison with the optimization not using it. The reduction in the PCG iterations are due to the fact that the topology and shape of the structure do not change much as the solution converges. Further research on improving this method may lead to a much faster iterative PCG solver especially suitable for design optimization. In the subsequent discussion, we will call the present method as the successive estimation method.

In the parallel PCG solver, most computations are performed in parallel at each of subdomains. However, the following two types of data communication among subdomains should be considered. The first type of communication is the communication occurring at nodes lying along the boundaries of adjacent subdomains. Some local quantities assigned on these nodes become global quantities after they are summed over all the subdomain boundaries along which the nodes lie. Fig. 2 depicts the summation operations along the boundaries. We will denote this type of communication by the operator $Comm_{\mathrm{bnd}}$, so we can write

$$\widehat{\boldsymbol{q}}_s = Comm_{\mathrm{bnd}}(\bar{\boldsymbol{q}}_s) \tag{5}$$

where the hatted quantities are global quantities.

The other type of communication is the communication required for the summation of some quantities $\boldsymbol{q}$ over the entire subdomains. This communication will be denoted by $Comm_{\text{sum}}$ as

$$\boldsymbol{q} = Comm_{\text{sum}}(\bar{q}_s) \tag{6}$$

where the operation $Comm_{\text{sum}}$ is the summation operator and defined differently depending on whether $\bar{\boldsymbol{q}}_{\text{s}}$ and $\boldsymbol{q}$ are scalars, vectors or matrices.

The two types of communication operations are implemented by using the Message Passing Interface (MPI) library (Malard, 1996; Message Passing Interface Forum, 1994) and the core of the parallelized PCG algorithm employing $Comm_{\text{bnd}}$ and $Comm_{\text{sum}}$ operators is summarized as Box 1. As shown in Box 1, every PCG iteration requires the communication operators several times and thus the subdivision pattern affects the numerical efficiency of the PCG algorithm.

Box 1. Parallelized PCG algorithm ("s" stands for the sth subdomain)

Set $\widehat{\mathbf{u}}_s^{(1)}$ ($=0$, frequently)
$\bar{\mathbf{r}}_s^{(1)} = \bar{\mathbf{f}}_s - \overline{\mathbf{K}}_s \widehat{\mathbf{u}}_s^{(1)}$
$\hat{\mathbf{r}}_s^{(1)} = Comm_{\text{bnd}}(\bar{\mathbf{r}}_s^{(1)})$
$\bar{\mathbf{z}}_s^{(1)} = \overline{\mathbf{D}}_s^{-1} \hat{\mathbf{r}}_s^{(1)}$
$\widehat{\mathbf{z}}_s^{(1)} = Comm_{\text{bnd}}(\bar{\mathbf{z}}_s^{(1)})$
$\widehat{\mathbf{p}}_s^{(1)} = \widehat{\mathbf{z}}_s^{(1)}$
$i = 1$
DO LOOP
  $\bar{\mathbf{b}}_s^{(i)} = \overline{\mathbf{K}}_s \widehat{\mathbf{p}}_s^{(i)}$
  $\alpha^{(i)} = \frac{Comm_{\text{sum}}(\bar{\mathbf{z}}_s^{(i)\mathrm{T}} \hat{\mathbf{r}}_s^{(i)})}{Comm_{\text{sum}}(\widehat{\mathbf{p}}_s^{(i)\mathrm{T}} \bar{\mathbf{b}}_s^{(i)})}$
  $\widehat{\mathbf{u}}_s^{(i+1)} = \widehat{\mathbf{u}}_s^{(i)} + \alpha^{(i)} \widehat{\mathbf{p}}_s^{(i)}$
  Convergence check:
    IF $\frac{Comm_{\text{sum}}(|\widehat{\mathbf{u}}_s^{(i+1)} - \widehat{\mathbf{u}}_s^{(i)}|)}{Comm_{\text{sum}}(|\widehat{\mathbf{u}}_s^{(i+1)}|)} < \varepsilon$, STOP
  $\bar{\mathbf{r}}_s^{(i+1)} = \bar{\mathbf{r}}_s^{(i)} - \alpha^{(i)} \bar{\mathbf{b}}_s^{(i)}$
  $\hat{\mathbf{r}}_s^{(i+1)} = Comm_{\text{bnd}}(\bar{\mathbf{r}}_s^{(i+1)})$
  $\bar{\mathbf{z}}_s^{(i+1)} = \overline{\mathbf{D}}_s^{-1} \hat{\mathbf{r}}_s^{(i+1)}$
  $\widehat{\mathbf{z}}_s^{(i+1)} = Comm_{\text{bnd}}(\bar{\mathbf{z}}_s^{(i+1)})$
  $\beta^{(i+1)} = \frac{Comm_{\text{sum}}(\bar{\mathbf{z}}_s^{(i+1)T} \hat{\mathbf{r}}_s^{(i+1)})}{Comm_{\text{sum}}(\bar{\mathbf{z}}_s^{(i)T} \hat{\mathbf{r}}_s^{(i)})}$
  $\widehat{\mathbf{p}}_s^{(i+1)} = \widehat{\mathbf{z}}_s^{(i+1)} + \beta^{(i)} \widehat{\mathbf{p}}_s^{(i)}$
  $i = i + 1$
END DO LOOP

2.2. Parallelization of the subspace iteration

While the parallelization strategy for linear static problems is almost established now, no unified parallel eigensolver has been established yet. Although the eigensolver parallelization by loop or array partitioning has been realized by Dongrra as SCALAPACK (Dongrra et al., 1994), such general-purpose library programs may not be very efficient for large-scale structural eigenvalue problems from the viewpoint of memory allocation and global system matrix assembly. So we use the domain decomposition approach to

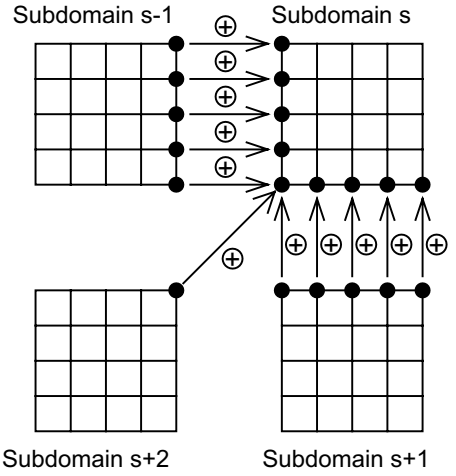

Fig. 2. Illustration for the role of the communication operator $Comm_{bnd}$ acting on nodes lying along subdomain boundaries.

enhance scalability and parallelization efficiency. To enhance the solution efficiency further, we propose a technique similar to the successive estimation method employed in the parallelized PCG solver.

In parallelizing the eigensolver, several approaches have been suggested. Papadrakakis and Yakoumidakis (1987) proposed a parallel PCG solver to minimize the Rayleigh quotient and Zhang and Moss (1994) suggested the parallelization of the linear equation analysis appearing in the eigensolution analysis procedure. In this work, however, we employ the subspace iteration method and parallelize it in the framework of the domain decomposition.

Before getting into the parallelization of the subspace iteration method, we write down the key equations in its serial version (see Bathe (1996) for more details)

$$\mathbf{K}\mathbf{x}_j = \lambda_j \mathbf{M}\mathbf{x}_j, \quad j = 1, 2, \ldots, p \tag{7}$$

Here, we assume that p eigensolutions are sought for. At the $(i+1)$th subspace iteration, we determine $\mathbf{P}^{(i+1)}$, $\mathbf{K}^{(i+1)}$ and $\mathbf{M}^{(i+1)}$ from the following equations:

$$\mathbf{K}\mathbf{P}^{(i+1)} = \mathbf{M}\mathbf{X}^{(i)} \tag{8}$$

$$\mathbf{K}^{(i+1)} = \mathbf{P}^{(i+1)\mathrm{T}}\mathbf{K}\mathbf{P}^{(i+1)} \tag{9a}$$

$$\mathbf{M}^{(i+1)} = \mathbf{P}^{(i+1)\mathrm{T}}\mathbf{M}\mathbf{P}^{(i+1)} \tag{9b}$$

In Eq. (8), $\mathbf{X}^{(i)}$ is the matrix consisting of q trial vectors at the ith iteration. We note that most of the computation time at the $(i+1)$th iteration step is spent on the calculation of the projection matrix $\mathbf{P}^{(i+1)}$ and the projection of $\mathbf{K}$ and $\mathbf{M}$ onto the q dimensional subspace. Therefore, we pay attention to the parallelization of them.

The parallel procedure to determine $\mathbf{P}^{(i+1)}$ from Eq. (8) is the same parallel PCG procedure as explained in the previous section. The Gaussian elimination-based direct solvers are generally efficient in handling multiple right-hand sides, $\mathbf{M}\mathbf{X}^{(i)}$, but they cannot be parallelized efficiently when the analysis domain is decomposed into subdomains. To parallelize the matrix computations involving $\mathbf{M}$ in Eqs. (8) and (9b), we decompose the mass matrix over $\mathbf{N}_d$ subdomains as

$$\mathbf{M} = \sum_{s=1}^{N_d} \overline{\mathbf{M}}_s \tag{10}$$

Box 2 describes the present parallelized subspace iteration method. The operators $Comm_{\text{sum}}$ in this case involves the summation of matrices. Note that at every iteration, the communication operator $Comm_{\text{sum}}$ is called only twice if not considering the parallel procedure to solve for $\overline{\mathbf{P}}_s^{(i+1)}$. From this point of view, the parallelized subspace iteration solver may be considered more efficient than the parallelized PCG solver.

Box 2. Parallelized subspace iteration method (described for the sth subdomain)

Set $\overline{\mathbf{X}}_s^{(1)} = ((\bar{\mathbf{x}}_1)_s(\bar{\mathbf{x}}_2)_s \cdots (\bar{\mathbf{x}}_q))^{(1)}$
(q initial guesses for eigenvectors)
$i = 1$
DO LOOP
Solve $(\overline{\mathbf{P}}_s^{(i+1)})$: (using parallel PCG solver)
$\mathbf{K}\mathbf{P}^{(i+1)} = \mathbf{M}\mathbf{X}^{(i)}$
$\overline{\mathbf{K}}_s^{(i+1)} = \overline{\mathbf{P}}_s^{(i+1)\mathrm{T}}\overline{\mathbf{K}}_s\overline{\mathbf{P}}_s^{(i+1)}$
$\overline{\mathbf{M}}_s^{(i+1)} = \overline{\mathbf{P}}_s^{(i+1)\mathrm{T}}\overline{\mathbf{M}}_s\overline{\mathbf{P}}_s^{(i+1)}$
Solve $(\boldsymbol{\lambda}, \mathbf{R}^{(i+1)})$:
$[Comm_{\text{sum}}(\overline{\mathbf{K}}_s^{(i+1)})]\mathbf{R}^{(i+1)} =$
$\lambda_k^{(i+1)}[Comm_{\text{sum}}(\overline{\mathbf{M}}_s^{(i+1)})]\mathbf{R}^{(i+1)}$
$\overline{\mathbf{X}}_s^{(i+1)} = \overline{\mathbf{P}}_s^{(i+1)}\mathbf{R}^{(i+1)}$
Convergence check:
IF $\frac{|\lambda_k^{(i+1)} - \lambda_k^{(i)}|}{|\lambda_k^{(i+1)}|} < \varepsilon (k = 1 \sim p)$, STOP
$i = i + 1$
END DO LOOP

2.3. The role of successive estimation in eigenvalue analysis

To improve the solution efficiency of the parallel PCG solver from Eq. (8), we also use the successive estimation method developed earlier. The role of the successive estimation method can be quite dramatic since multiple right-hand sides are solved simultaneously. To see this, we consider the eigenvalue analysis (not optimization) of the structure shown in Fig. 3(a) and compare the numerical performance of the parallel subspace iteration methods with and without the use of the successive estimation scheme.

To find the lowest seven eigensolutions of the problem in Fig. 3(a), we simply used the same number of the trial vectors. The analysis domain discretized by 5120 elements is decomposed into four subdomains; see Fig. 3(b) and (c). Tables 1 and 2 compare the analysis histories by the parallel subspace iteration methods with and without the use of the successive estimation scheme. The saving in the CPU time by the successive estimation scheme is quite significant. When no successive estimation is used in the PCG solver, the number of the PCG iterations remains unchanged after certain iteration numbers for each eigenmode. This is because the corresponding eigenvectors are almost converged after the iteration numbers. Therefore, the successive estimation scheme within every eigenvalue analysis at each design iteration step must be used to avoid unnecessary computations.

In the parallel eigenvalue analysis discussed above, the number q of the trial eigenvectors was the same as the number p of the desired eigenvectors. When q becomes larger than p, the number of the subspace

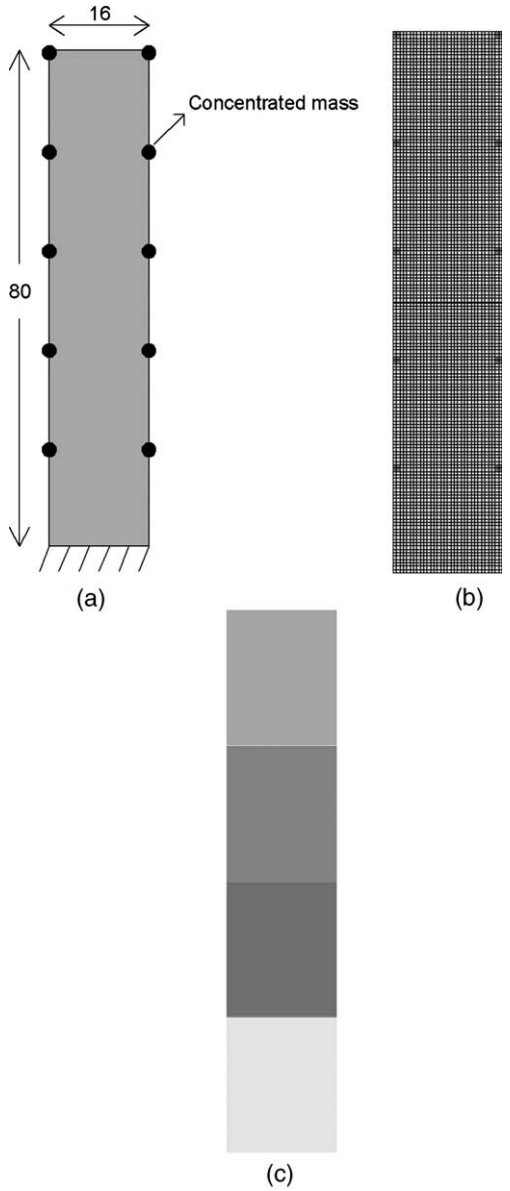

Fig. 3. (a) A structure model to be used for parallel eigenvalue analysis ($E = 2.0 \times 10^8$, $v = 0.3$, $\rho = 1.0$, point mass = 20); (b) the domain discretization by 5120 elements and (c) the domain division for parallel computing.

iterations can be substantially increased. So we use q which is equal to or larger than p by 1 or 2 in the subsequent design problems.

2.4. Parallelization efficiency in the eigenvalue analysis

The efficiency of the proposed parallel topology optimization algorithm will obviously depend on how the domain is decomposed. We will examine this aspect with the example shown in Fig. 4(a) where the domain is

Table 1
The iteration history for the parallel subspace iteration method not using the successive estimation scheme (the total elapsed time = 102 s)

Subspace iteration number	PCG iteration number							Sum
	First mode	Second mode	Third mode	Fourth mode	Fifth mode	Sixth mode	Seventh mode	
1	631	635	680	696	650	622	613	4527
2	542	345	671	656	507	669	690	4080
3	424	329	326	494	540	552	552	3217
4	424	329	203	412	424	547	478	2817
5	424	329	203	357	424	422	479	2638
6	424	329	203	357	424	421	462	2620
7	424	329	203	357	425	340	458	2536
8	424	329	203	357	377	292	457	2439
9	424	329	203	357	376	280	456	2425
10	424	329	203	357	376	269	426	2384
11	424	329	203	357	376	232	398	2319
12	424	329	203	357	376	231	373	2293
13	424	329	203	357	376	231	373	2293
14	424	329	203	357	376	231	373	2293
15	424	329	203	357	376	231	373	2293
16	424	329	203	357	376	231	373	2293
17	424	329	203	357	376	231	373	2293
Frequency (Hz)	5.05	27.10	40.83	64.30	106.05	121.51	148.88	45,760

Table 2
The iteration history for the parallel subspace iteration method using the successive estimation scheme (the total elapsed time = 58 s)

Subspace iteration number	PCG iteration number							Sum
	First mode	Second mode	Third mode	Fourth mode	Fifth mode	Sixth mode	Seventh mode	
1	631	635	680	696	650	622	613	4527
2	479	633	671	655	658	666	683	4445
3	2	290	504	659	531	683	686	3355
4	2	2	616	290	507	321	440	2178
5	2	2	2	104	304	318	436	1168
6	2	2	2	2	242	418	342	1010
7	2	2	2	2	145	316	264	733
8	2	2	2	2	140	179	270	597
9	2	2	2	2	88	178	198	472
10	2	2	2	2	2	98	265	373
11	2	2	2	2	2	83	198	291
12	2	2	2	2	2	2	146	158
13	2	2	2	2	2	2	211	223
14	2	2	2	2	2	2	64	76
15	2	2	2	2	2	2	53	65
16	2	2	2	2	2	2	42	54
17	2	2	2	2	2	2	2	14
Frequency (Hz)	5.05	27.10	40.83	64.30	106.05	121.51	148.88	19,739

divided into 1, 2, 4, 8, and 16 subdomains. In general, the important performance measure in parallelized algorithms is the parallel efficiency (Kumar et al., 1994) defined as $E(P) = T_{seq}/(P \times T(P))$ where P repre-

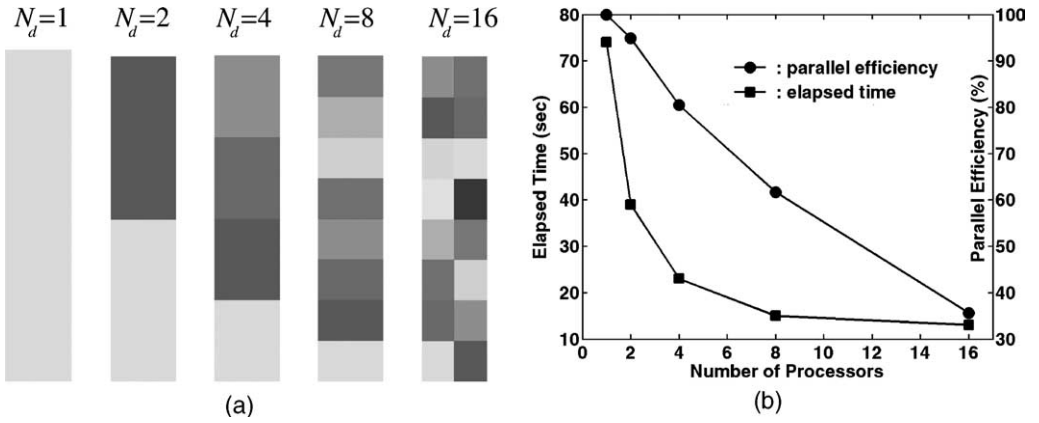

Fig. 4. (a) Domain decomposition of 2-dimensional structure (number of subdomains: 1, 2, 4, 8, 16) and (b) the relation between the elapsed time and the number of the used processors.

sents the number of used processors, T_{seq}, $T(P)$ are the runtime of the sequential algorithm and the parallel algorithm, respectively. The elapsed time and the parallel efficiency as the function of the process number are plotted in Fig. 4(b).

Some remarks may be made regarding the parallel efficiency shown in Fig. 4(b). When the number of subdomains in the domain decomposition increases, the number of the shared nodes by subdomains increases, so that the parallel efficiency drops. This phenomenon can also be explained by Amdahl's effect (Amdahl, 1967). According to Amdahl's effect, no parallelized program can exhibit the parallelization efficiency that is proportional to the number of processors because of the serial portion inside it.

3. Parallelization of topology optimization for free vibration problems

3.1. A brief overview of eigenvalue maximization problems

In the density function method, the topology optimization problems that maximize the structural eigenvalue within a prescribed amount of material can be described as:

$$\text{Minimize} \qquad f(\boldsymbol{\rho}) = -\ln(\lambda_1^{w_1}\lambda_2^{w_2}\cdots\lambda_{N_\lambda}^{w_{N_\lambda}}) \tag{11a}$$

$$\text{subject to a mass constraint,} \quad h(\boldsymbol{\rho}) = \sum_{e=1}^{N_e}\rho_e V_e - M_0 \leqslant 0 \tag{11b}$$

$$\text{side constraints} \quad \varepsilon \leqslant \rho_e \leqslant 1 (0 < \varepsilon \ll 1) \tag{11c}$$

where the objective function f is the weighted product of the first N_λ eigenvalues by the weighting factor w_j and N_e is the number of finite elements used to discretize a design domain and V_e the volume of each element. The functional form in Eq. (11a) was used by Kim and Kim (2002). The logarithmic function in Eq. (11a) for structural eigenvalue topology optimization problems serves to balance the contributions of several eigenvalues to the sensitivity. To see this, we write the sensitivity of f with respect to ρ_e

$$\frac{\partial f}{\partial \rho_e} = -\sum_{j=1}^{N_\lambda} w_j \frac{1}{\lambda_j}\frac{\partial \lambda_j}{\partial \rho_e} \tag{12}$$

The density array $\boldsymbol{\rho}$ is defined as $\{\rho_1, \rho_2, \ldots, \rho_e, \ldots, \rho_{N_e}\}^{\mathrm{T}}$ where ρ_e is the relative density variable assigned to the eth finite element.

In topology optimization based on the density method (Bendsøe and Sigmund, 1999), the constitutive matrix is penalized as

$$\mathbf{C}_e = \mathbf{C}_e(\boldsymbol{\rho}) = (\rho_e)^n \mathbf{C}_0 \tag{13}$$

where $\mathbf{C}_0$ is the constitutive matrix of a solid isotropic material and n is the penalty parameter (usually between 2 and 3).

The sensitivities of the eigenvalue λ_j and h with respect to the design variable ρ_e are given by (see Haftka and Gürdal, 1992)

$$\frac{\partial \lambda_j}{\partial \rho_e} = \frac{1}{\rho_e}[n(\mathbf{x}_j)^{\mathrm{T}} \mathbf{k}_e(\mathbf{x}_j) - \lambda_j(\mathbf{x}_j)^{\mathrm{T}} \mathbf{m}_e(\mathbf{x}_j)] \tag{14}$$

$$\frac{\partial h(\boldsymbol{\rho})}{\partial \rho_e} = V_e \tag{15}$$

where $\mathbf{k}_e$ and $\mathbf{m}_e$ are the element stiffness and mass matrices.

3.2. Parallelization of the optimality criteria algorithm

As an optimizer for topology optimization problems, various algorithms such Optimality Criteria (OC), Method of Moving Asymptotes (MMA, Svanberg, 1987) can be employed. Borrvall and Petersson (2001) used MMA for the parallel static topology optimization while Kim (2001) used OC for the parallel static and eigenvalue topology optimization. In this work, we employ the OC method and develop a parallelized version of the OC method for the present eigenvalue-related topology optimization problems.

To use the OC method, we consider the following Lagrangian function for the optimization problem stated as Eq. (11)

$$L(\boldsymbol{\rho}, \mu) = f(\boldsymbol{\rho}) + \mu h(\boldsymbol{\rho}) \tag{16}$$

where μ is a Lagrange multiplier and $h(\boldsymbol{\rho})$ is treated as an equality constraint.

Using the Karush–Kuhn–Tucker condition for Eq. (16) and some function approximations, one can obtain the well-known updating rule for the density design variable ρ_e (Ma et al., 1993; Bendsøe and Sigmund, 2003):

$$\rho_e^{(i+1)} = \left(-\frac{1}{\mu} \frac{\partial f/\partial \rho_e}{\partial h/\partial \rho_e} \right)^{\eta} \rho_e^{(i)} = (\Pi_e)^{\eta} \rho_e^{(i)} \tag{17}$$

where

$$\mu = \left(\frac{\sum_{e=1}^{N_e} ((-\partial f/\partial \rho_e)/(\partial h/\partial \rho_e))^{\eta} \rho_e^{(i)} V_e}{M_0} \right)^{1/\eta}$$

Note that $\partial f/\partial \rho_e$ and $\partial h/\partial \rho_e$ in Eq. (17) involve the quantities associated only with the eth element, which can be observed from Eqs. (14) and (15). Therefore, the design variable update rule in Eq. (17) can be performed independently for each subdomain. The data communication is then needed only to evaluate f and h for the convergence check. We use the following equations for the evaluation of h:

$$h(\boldsymbol{\rho}) = \sum_{s=1}^{N_{\mathrm{d}}} \bar{m}_s(\bar{\boldsymbol{\rho}}_s) - M_0 = \sum_{s=1}^{N_{\mathrm{d}}} \left(\sum_{e=1}^{(N_e)_s} \rho_e V_e \right) - M_0 = 0 \tag{18}$$

where $(N_e)_s$ denotes the number of finite elements used to discretize the subdomain.

To use the OC algorithm with the updating rule in Eq. (17) for eigenvalue problems, the shifting scheme is used. This means that the following form of the Lagrangian function $L(\boldsymbol{\rho}, \mu)$ is used with the objective function and the Lagrange multipliers replaced by $\tilde{f}$ and $\tilde{\mu}$, respectively:

$$L(\boldsymbol{\rho}, \mu) = f + \mu h = (f - \sigma h) + (\mu + \sigma)h = \tilde{f} + \tilde{\mu} h \tag{19}$$

where σ is adjusted to satisfy

$$\sigma > \left(\frac{\partial f / \partial \rho_e}{\partial h / \partial \rho_e} \right) \quad \text{for all } e = 1, \ldots, N_e \tag{20}$$

The sensitivity analysis and the shifting scheme are standard (see, Diaz and Kikuchi, 1992; Ma et al., 1995a,b; Kim and Kim, 2000, 2002), but are given here for the sake of completeness.

The parallelized OC method developed in this work is outlined as Box 3. While the parallel PCG solver requires both $Comm_{\text{bnd}}$ and $Comm_{\text{sum}}$, the parallel OC algorithm requires $Comm_{\text{sum}}$ only: see Boxes 1 and 3. From the viewpoint of the communication efficiency, therefore, the parallelization of the OC algorithm is more efficient than that of the PCG method. Since the present parallel algorithms are based on the single program multiple data (SPMD) processing strategy, good scalability is expected. This scalability aspect will be numerically examined in the next section.

Box 3. Parallelized OC method (described for the sth subdomain)

Set $\bar{\boldsymbol{\rho}}_s^{(1)}$ (uniform distribution with prescribed mass)
$i = 1$
DO
  Calculate f'_e and h'_e
  Set $S = \{e | 1 \leqslant e \leqslant (N_e)_s\}$
  $\widetilde{M}_0 = M_0$
  DO
    $\mu = \left(\dfrac{Comm_{\text{sum}}\left(\sum_{e \in S} (-f'_e/h'_e)^{\eta} \rho_e^{(i)} V_e \right)}{\widetilde{M}_0} \right)^{1/\eta}$
    $\rho_e^{(i+1)} = \left(-\frac{1}{\mu} \frac{f'_e}{h'_e} \right)^{\eta} \rho_e^{(i)}$
    Set $S = \{e | \rho_l < \rho_e < \rho_u\}$
    Set $S_l = \{e | \rho_e \leqslant \rho_l\}$
    Set $S_u = \{e | \rho_e \geqslant \rho_u\}$
    $\rho_e^{(i+1)} = \begin{cases} \rho_l & \text{if } (e \in S_l) \\ \rho_u & \text{else if } (e \in S_l) \\ \rho_e^{(i+1)} & \text{else} \end{cases}$
    Convergence check:
      IF $\frac{|Comm_{\text{sum}}(\bar{m}_s(\boldsymbol{\rho}^{(i+1)})) - M_0|}{M_0} < \varepsilon$, STOP
    $\widetilde{M}_0 = M_0 - Comm_{\text{sum}}\left(\sum_{e \in S_l} \rho_l V_e + \sum_{e \in S_u} \rho_u V_e \right)$
  END DO LOOP
  Apply move limits
  Convergence check:
    IF $\frac{Comm_{\text{sum}}(|\bar{f}_s(\boldsymbol{\rho}^{(i+1)}) - \bar{f}_s(\boldsymbol{\rho}^{(i)})|)}{Comm_{\text{sum}}(|\bar{f}_s(\boldsymbol{\rho}^{(i+1)})|)} < \varepsilon$, STOP
  $i = i + 1$
END DO LOOP

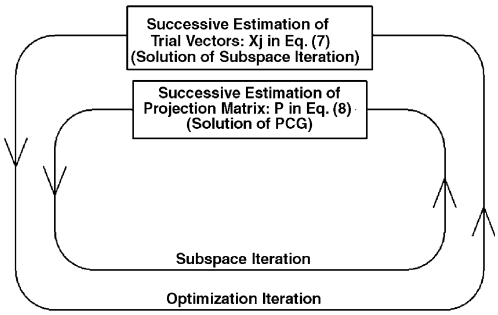

Fig. 5. The schematic description of the successive estimation scheme applied to both the analysis loop and the optimization loop.

In eigenvalue topology optimization problems, the successive estimation scheme is also used both in the iterative analysis loop and in the iterative optimization loop. The schematic description of the strategy is illustrated in Fig. 5. In the next section, we will examine the numerical performance of the present parallel strategy for eigenvalue topology optimization.

4. Numerical verification

In this section, we consider first a two-dimensional reinforcing problem of a cantilevered beam to check the numerical performance of the present parallel method and then solve two large-size optimization problems which would be formidable without a parallel optimization algorithm.

4.1. Case study 1: reinforcement of a cantilevered beam

Fig. 6(a) depicts the design problem for which we aim at maximizing the eigenfrequency of the structure. In this optimization problem, the lowest 4 eigenvalues are used in defining the objective function, Eq. (11) with $w_1 = w_2 = w_3 = w_4$. The mass constraint ratio is 30%. The lowest four eigenmodes for the structure having a uniform density distribution of $\rho = 0.3$ are shown in Fig. 6(b). The optimized results for the meshes consisting of 160×32 and 320×64 elements are shown in Fig. 6(c). In obtaining the result in Fig. 6(c) and the next examples, a checkerboard controlling filter is used (Sigmund, 1994). The obtained results are consistent with existing results and exhibit mesh dependence. Since the mesh dependence issue is now well understood (see, e.g., Sigmund and Petersson, 1998 or Bendsøe and Sigmund, 2003), this issue will not be discussed here. For the solution verification, the eigenfrequencies before and after optimization are compared in Table 3.

To check the parallelization efficiency, we divide the structure in Fig. 6(a) into subdomains shown in Fig. 7(a). Fig. 7(b) shows how the elapsed time varies with respect to the number of the used processors. The decrease in the elapsed time for the eigenvalue topology design optimization slows down as the number of processors (equal to the number of subdomains) increases, as expected; see Fig. 7(b). For this analysis, the mesh consisting of 160×32 elements is used and the successive estimation strategy proposed in this investigation is utilized. To see the effect of the successive estimation strategy on the solution speedup, the elapsed times to complete the design optimization with and without the use of the strategy are compared in Fig. 8. When the strategy is used, the elapsed time increases almost linearly to the optimization iteration number. However, the elapsed time increases rather rapidly, otherwise. This figure may demonstrate the

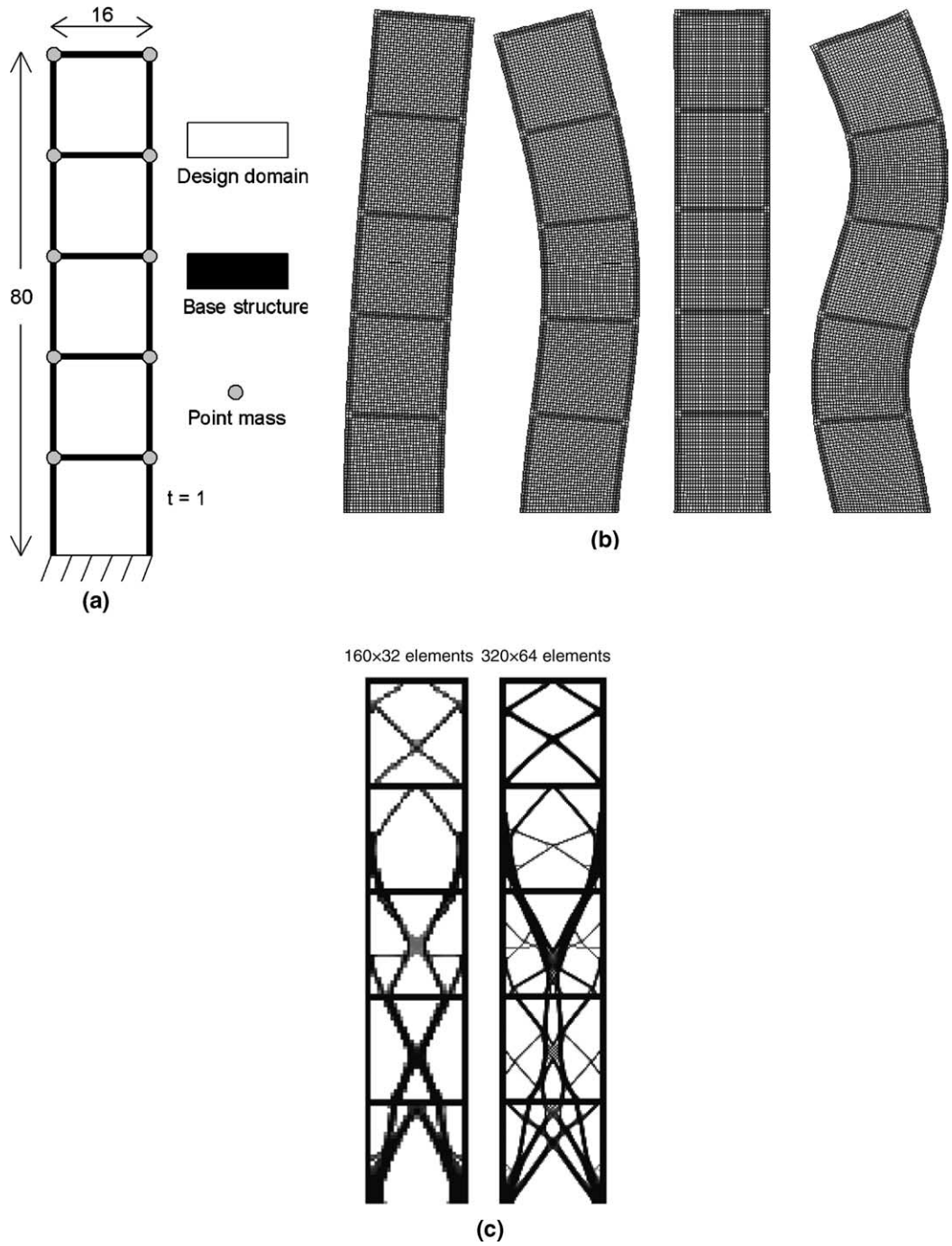

Fig. 6. (a) An eigenvalue topology design problem description ($E = 2.0 \times 10^8$, $v = 0.3$, $\rho = 1.0$, point mass = 20); (b) the lowest four eigenmodes of the structure having the uniform density of 0.3 in the design domain and (c) the optimized results for the mass constraint ratio of 30%.

Table 3
The eigenfrequencies before and after optimization for the problem depicted in Fig. 6(a)

Eigen frequency	Before optimization	After optimization	
		160×32 mesh	320×64 mesh
First	3.46 Hz	4.37 Hz (+26.3%)	4.40 Hz (+27.2%)
Second	12.88 Hz	17.78 Hz (+38.0%)	19.35 Hz (+50.2%)
Third	21.88 Hz	28.28 Hz (+29.3%)	29.65 Hz (+35.5%)
Fourth	26.01 Hz	34.98 Hz(+34.5%)	43.20 Hz (+66.1%)

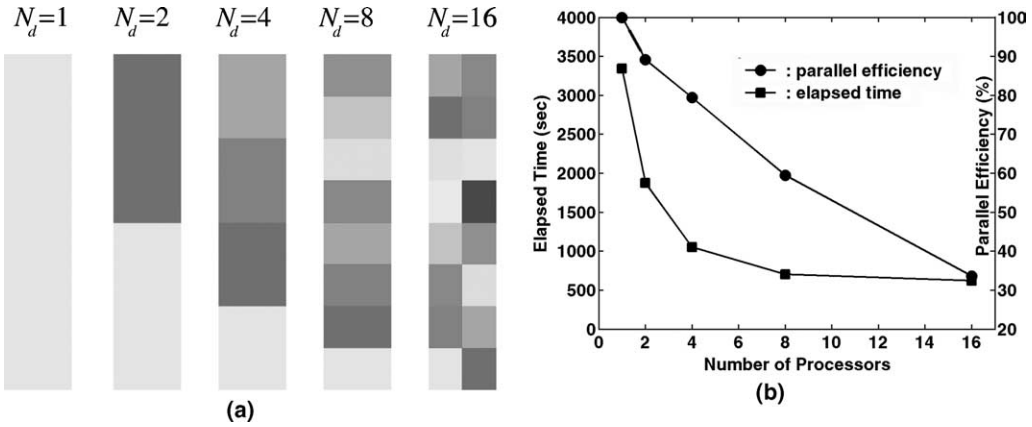

Fig. 7. (a) Various subdivisions of the design domain shown in Fig. 6(a) and (b) the relation between the elapsed time and the number of the used processors (for the 160×32 mesh).

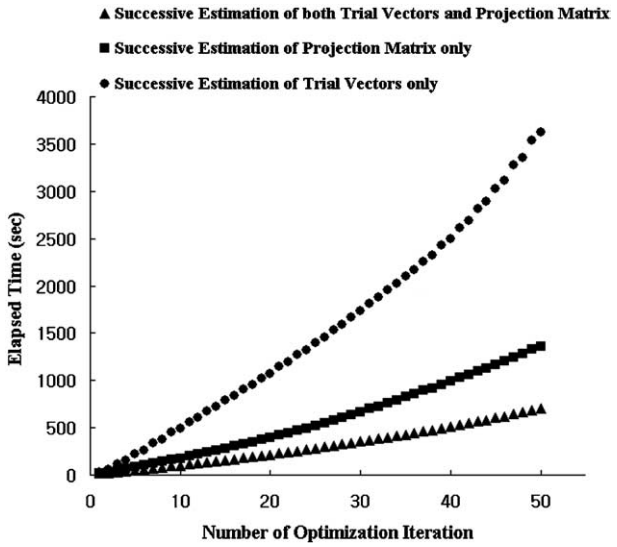

Fig. 8. The comparison of the elapsed times with and without the use of the successive estimation strategy (using eight processors, truncated at 50 iterations).

role of the successive estimation strategy also in design optimization. The reduction in the elapsed time by the proposed strategy is substantial.

4.2. Case study 2: three-dimensional elbow design

As a large-scale topology optimization problem, we consider the design of a three-dimensional cantilevered elbow with a rigid block attached to its one end. The problem is illustrated in Fig. 9(a). The design objective is to maximize the lowest 2 eigenvalues that are weighted equally in the objective function. The mass constraint ratio used is 15% and the design domain is discretized by 64,000 elements. This mesh size amounts to 213,003 degrees of freedom. The design domain is divided into 36 domains as shown in Fig. 9(b). Since this large-size design problem is not feasible to solve with a single or a few CPU's, we solve this problem only with 36 CPU's. The main motivation of this problem is to address the potential of the parallel eigenvalue topology optimization strategy in handling large-size design problems. The optimized result is shown in Fig. 9(c) and the eigenfrequencies before and after the optimization are compared in Table 4. It is apparent that the large-size problems can be solved satisfactorily within reasonable time if the number of CPU's for parallel optimization is sufficient.

4.3. Case study 3 (practical application): optical pickup bobbin design

As a practical large-scale topology optimization problem, we consider the design of an axial-type optical pickup bobbin design shown in Fig. 10. The optical lens will be mounted on the side of the cylindrical hole of radius $r_2 = 1.85$ mm. The bobbin moves along the small hole of radius $r_1 = 0.7$ mm and rotates around

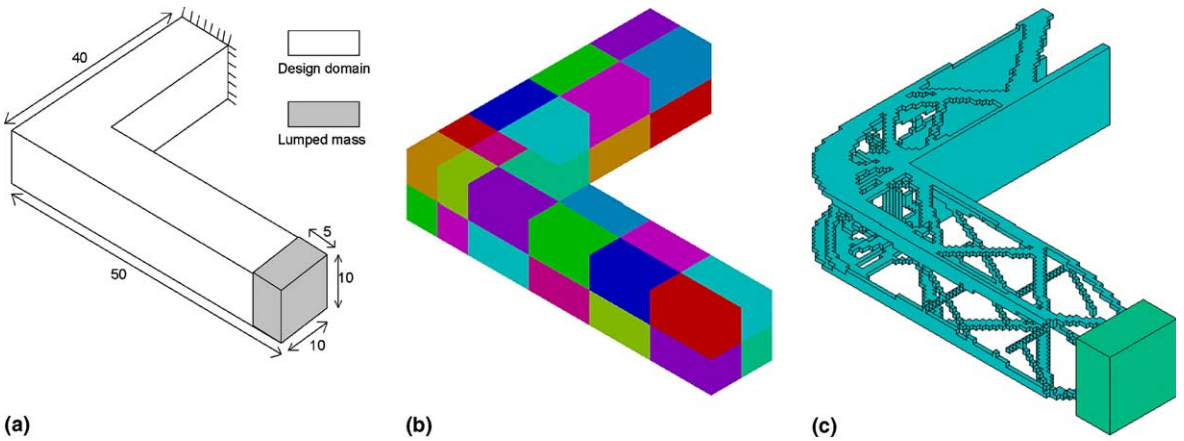

Fig. 9. (a) The eigenvalue maximization problem of an elbow structure ($E = 2.0 \times 10^8$, $v = 0.3$, $\rho = 1.0$); (b) the division of the design domain into 36 subdomains and (c) The optimized result.

Table 4
The eigenfrequencies before and after optimization for the problem depicted in Fig. 9(a)

Eigenfrequency	Before	After
First (Hz)	1.06	5.39
Second (Hz)	1.23	8.65
Number of optimization iterations		76

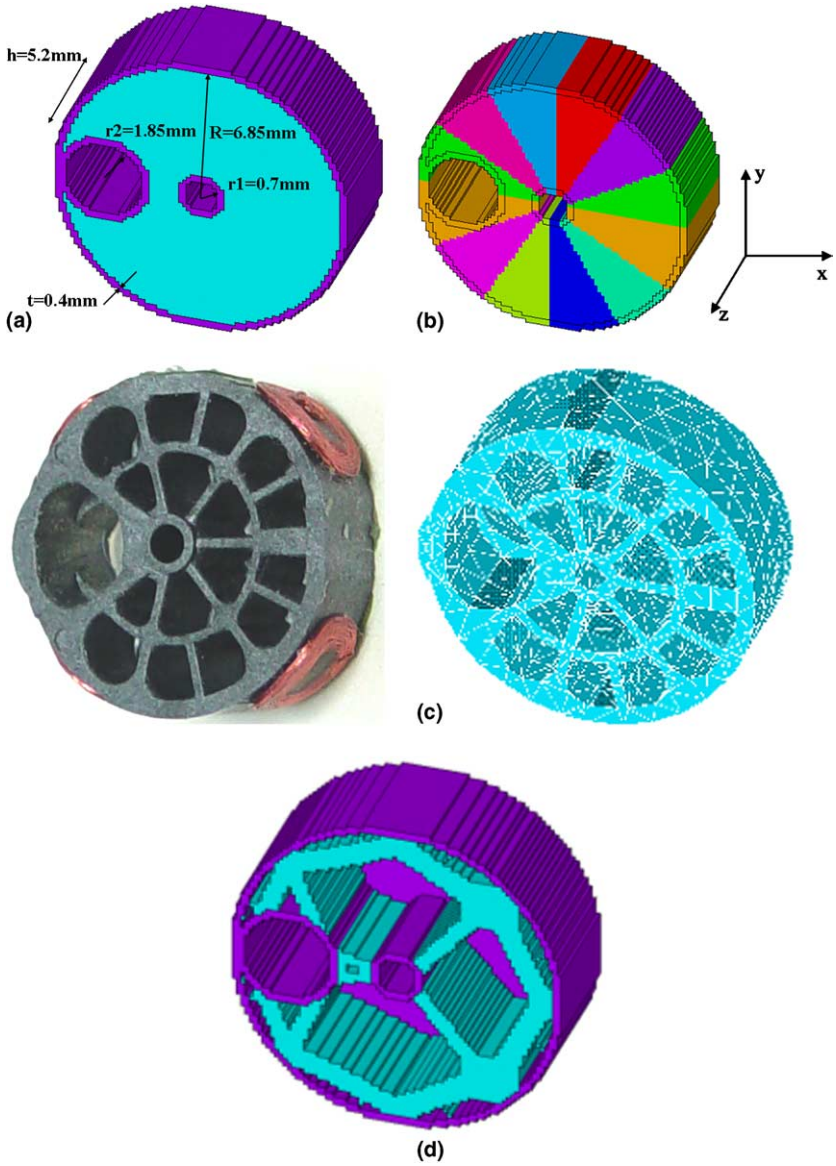

Fig. 10. (a) Model and design domain for an axial-type pickup actuator ($E = 29.4$ GPa, $v = 0.35$, $\rho = 1.49$ kg/mm^3); (b) the division of the design domain into 12 subdomains; (c) baseline design and (d) optimized design.

Table 5
The fundamental eigenfrequencies before and after optimization for the problem depicted in Fig. 10(a)

Quantity	Before	After
Fundamental eigenfrequency (kHz)	27.8	29.1
Mass (mg)	752	636
Number of optimization iterations		44

it. The design objective in this investigation is to maximize the fundamental eigenvalue of the bobbin itself while the mass constraint ratio is 40%. The detailed modeling technique of the axial-type bobbin is not given as this design example is used to show the effectiveness of the parallel topology optimization. The design domain is discretized by 135,807 elements which amount to 407,421 degrees of freedom. Fig. 10(b) shows a design domain and 12 subdomains for the parallelization. In this example, density design variables are linked along the z-axis (see Fig. 10(b)) for good manufacturability.

Fig. 10(c) and (d) show the initial baseline design and the optimized design by the present investigation. The fundamental eigenfrequencies before and after the optimization are also compared in Table 5. Table 5 shows that as a result of the topology optimization, the mass is reduced and the fundamental eigenfrequency is increased. Large-size design problems like this one would be difficult to handle without using the parallelized topology optimization method.

5. Conclusions

We developed a parallel topology optimization method in order to handle the large-size design optimization problems that may be encountered in practical optimization applications. In the frame of domain decomposition, both the numerical analysis and the optimization process were parallelized focusing on eigenvalue maximization problems.

Some contributions of the present work may be summarized as:

- The parallel optimality criteria algorithm based on domain decomposition is developed with which the design variables can be updated with the minimum communication between decomposed subdomains.
- A parallel subspace iteration based on the PCG solver was proposed for structural eigenvalue problems. To speed up the solution convergence, the successive estimation strategy for initial guesses is employed.
- The present parallel topology optimization method is successfully applied to large-size structural design problems which are otherwise formidable.

Acknowledgements

Most parallel computations in this work were carried out by the cluster system at Samsung Advanced Institute of Technology (SAIT) and IBM RS/6000 SP system (216 Gflops) at the Seoul National University Computer Center. We also thank the anonymous reviewers for their critical comments.

References

Amdahl, G., 1967. Validity of the single processor approach to achieving large-scale computing capabilities. AFIPS Conference Proceedings 30, 483–485.

Angeleri, F., Sonnad, V., Bathe, K.J., 1989. Studies of finite element procedures—an evaluation of preconditioned iterative solvers. Computers and Structures 32, 671–677.

Bathe, K.J., 1996. Finite Element Procedures. Prentice-Hall, Inc, Englewood Cliffs, NJ.

Bendsøe, M.P., Sigmund, O., 1999. Material interpolation schemes in topology optimization. Archive of Applied Mechanics 69, 635–654.

Bendsøe, M.P., Sigmund, O., 2003. Topology Optimization—Theory, Methods and Applications. Springer-Verlag, Berlin Heidelberg New York.

Bitzarakis, S., Papadrakakis, M., Kotsopulos, A., 1997. Parallel solution technique in computational structural mechanics. Computational Methods in Applied Mechanical Engineering 148, 75–104.

Borrvall, T., Petersson, J., 2001. Large-scale topology optimization in 3D using parallel computing. Computational Methods in Applied Mechanical Engineering 190, 6201–6229.

Coutinho, A.L.G.A., Alves, J.L.D., Landau, L., Ebecken, N.F.F., Troina, L.M., 1991. Comparison of Lanczos and conjugate gradients for the element-by-element solution of finite element equations on the IBM3090 vector computer. Computers and Structures 39, 47–55.

Diaz, A.R., Kikuchi, N., 1992. Solutions to shape and topology eigenvalue optimization problems using a homogenization method. International Journal of Numerical Methods in Engineering 35, 1487–1502.

Dongrra, J.J., et al., 1994. The design anf implementation of the SCALAPACK LU, QR, and Cholesky factorization routines. Technical Report, University of Tennessee.

El-Sayed, M.E.M., Hsiung, C.K., 1991. Optimum structural design with parallel finite element analysis. Computers and Structures 40, 1469–1474.

Farhat, C., Wilson, E., 1988. A parallel active column equation solver. Computers and Structures 28, 289–304.

Haftka, R.D., Gürdal, Z., 1992. Elements of structural optimization, 3rd ed. Kluwer Academic Publishers, London.

Hwang, T., Parsons, I.D., 1994. Parallel implementation and performance of multigrid algorithms for solving Eigenvalue problems. Computers and Structures 50, 325–336.

Johnson, S.L., Mathur, K.K., 1990. Data structures and algorithms for the finite element methods on a data parallel computer. International Journal of Numerical Methods in Engineering 29, 881–908.

Katagiri, T., Kanada, Y., 2001. An efficient implementation of parallel eigenvalue computation for massively parallel processing. Parallel Computing 27, 1831–1845.

Kim, T.S., 2001. Structural Topology Design Optimization Using Parallel Computers. Ph.D. Thesis. School of Mechanical and Aerospace Engineering, Seoul National University, Korea.

Kim, T.S., Kim, Y.Y., 2000. MAC-based mode-tracking in structural topology optimization. Computers and Structures 74, 375–383.

Kim, T.S., Kim, Y.Y., 2002. Multi-objective topology optimization of a beam under torsion and distortion. AIAA Journal 40 (2), 376–381.

Kumar, V., Grama, A., Gupta, A., Karypis, G., 1994. Introduction to Parallel Computing—Design and Analysis of Algorithms. The Benjamin/Cummings Publishing Company, Inc, Redwood City, CA.

Lang, B., 1999. Efficient eigenvalue and singular value computation on shared memory machines. Parallel Computing 25, 845–860.

Lui, S.H., 1998. Kron's method for symmetric Eigenvalue problems. Journal of Computational and Applied Mechanics 98, 35–48.

Ma, Z.D., Kikuchi, M., Hagiwara, I., 1993. Structural topology and shape optimization for a frequency response problem. Computational Mechanics 13, 157–174.

Ma, Z.D., Kikuchi, N., Cheng, H.C., Hagiwara, I., 1995a. Topological optimization technique for free vibration problems. Journal of Applied Mechanics 62, 200–207.

Ma, Z.D., Kikuchi, N., Cheng, H.C., 1995b. Topological design for vibrating structures. Computer Methods in Applied Mechanics and Engineering 121, 259–280.

Malard, J., 1996. MPI: a Message Passing Interface standard, history, overview and current status. Technology Watch Report, Edinburgh Parallel Computing Center.

Message Passing Interface Forum, 1994. MPI: a Message Passing Interface standard. International Journal of Supercomputer Applications and High Performance Computing, 8 (Special issue on MPI).

Padula, S.L., Stone, S.C., 1998. Parallel implementation of large-scale structural optimization. Structural Optimization 16, 176–185.

Papadrakakis, M., 1993. Solving large-scale problems in mechanics. John Wiley & Sons, Chichester.

Papadrakakis, M., 1997. Parallel solution methods in computational mechanics. John Wiley & Sons, Chichester.

Papadrakakis, M., Bitzarakis, S., 1996. Domain decomposition PCG methods for serial and parallel processing. Advances in Engineering Software 25, 291–307.

Papadrakakis, M., Yakoumidakis, M., 1987. On the preconditioned conjugate gradient method for solving $(A - \lambda B)X = 0$. International Journal of Numerical Methods of Engineering 24, 1355–1366.

Sigmund, O., 1994. Design of material structures using topology optimization. Ph.D. Thesis. Department of Solid Mechanics, Technical University of Denmark.

Sigmund, O., Petersson, J., 1998. Numerical instabilities in topology optimization: a survey on procedure dealing with checkerboards, mesh-dependencies and local minima. Structural Optimization 16, 68–75.

Su, P.S., Fulton, R.E., 1993. A parallel domain decomposition finite element method for massively parallel computers. Computing Systems in Engineering 4, 489–494.

Svanberg, K., 1987. The method of moving asymptotes-a new method for structural optimization. International Journal of Numerical Methods of Engineering 24, 359–373.

Thierauf, G., Cai, J., 1997. Parallel evolutionary strategy for solving structural optimization. Engineering Structures 19, 318–324.

Topping, B.H.V., Leite, J.P.B., 1999. Parallel simulated annealing for structural optimization. Computers and Structures 73, 545–564.

Wang, X., Williams, F.W., Kennedy, D., 1996. A parallel structural optimization method and its implementation on a Transtech Pyramid. Computer Methods in Applied Mechanics and Engineering 135, 381–392.

Watkins, W.J., Kennedy, D., Williams, F.W., 1996. Efficient parallel solution of structural Eigenvalue problems. Advances in Engineering Software 25, 189–281.

Zhang, H., Moss, W.F., 1994. Using parallel banded linear solvers in generalized eigenvalue problems. Parallel Computing 20, 1089–1105.