Parallel algorithm for BEM analysis

N. Kamiya[a], H. Iwase[b], E. Kita[c]
[a]*School of Information and Sciences, Nagoya University, Nagoya 464-01, Japan*
[b]*Graduate School of Human Informatics, Nagoya University, Nagoya 464-01, Japan*
[c]*Graduate School of Engineering, Nagoya University, Nagoya 464-01, Japan*

Abstract

Two algorithms for the boundary element parallel computation are presented in this paper. The first is that the adaptive iteration scheme, composed of sample point error analysis and mesh refinement by extended error indicator proposed by the authors, is divided into partial parallel routines with two workstations connected by LAN. The second is the domain decomposition boundary element analysis on cluster computing with a few workstations. The idea, however, is similar to that in the finite element domain decomposition scheme for the parallel computation, some fundamental considerations are carried out for the identification of the boundary condition on the fictitious internal boundaries and iteration schemes for the final solution. Three different schemes, with respect to the convergence and stability, are compared for simple example problems.

Introduction

These days, parallel computation is widely employed for various numerical calculations. In the finite element methods, for example, the data separation and exchanging data between processors is popular; domain decomposition is typical scheme, *e.g.*[1]. Davies[2,3], in the boundary element method, used transputer and distributed array computers in order to distribute boundary nodes to each processor for data decomposition, parallel computation and experienced high computational efficiency. Ramesh *et al.*[4] solved charge evolution problem. Hribersek *et al.* [5] solved the Navier-Stokes problem with boundary-domain formulation with the aid of the Schwarz method. Avrashi *et al.* [6] tried to solve two-dimensional elastic problem by domain decomposition using the boundary strip method. Cluster computing with a few workstations connected by Local Area Network (LAN) can realize a virtual parallel computer resource. Performance of this system is closely connected to communication speed and

therefore is fitted for coarse grain problems.

In this study, we try to perform cluster parallel computing for increase of computational efficiency in the boundary element analysis. The first part deals with the adaptive iteration scheme by dividing the entire process into the boundary element analysis and error analysis processes. The second parts is devoted to the domain decomposition boundary element analysis for the parallel solver using three kinds of renewal methods for the variables on the fictitious internal boundaries.

Cluster computing circumstance

A few workstations connected by Local Area Network can construct a computer system available for parallel computation with data communication between them, which is often called the cluster computing system. In the ordinary UNIX operating system, communication by TCP/IP requires higher level technology, but recently message passing libraries which makes the cluster computing circumstance easier have been developed. Figure 1 is the cluster system constructed here for the stated object in our laboratory; one workstation is the host machine controls other m workstations (submachines) connected by ethernet cable. The task is divided into a few parts and each submachines treats its different parts. Data communication between host and submachines are performed by PVM installed in the host. Performance of these workstations is in general different because they are open for users having access account.

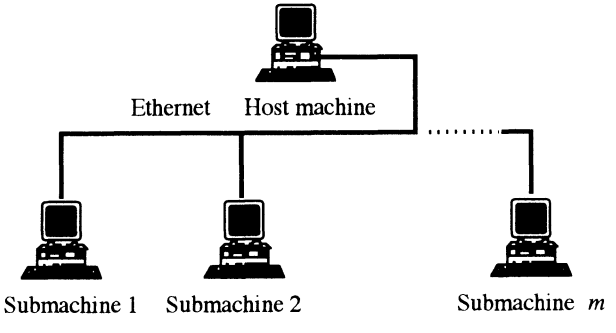


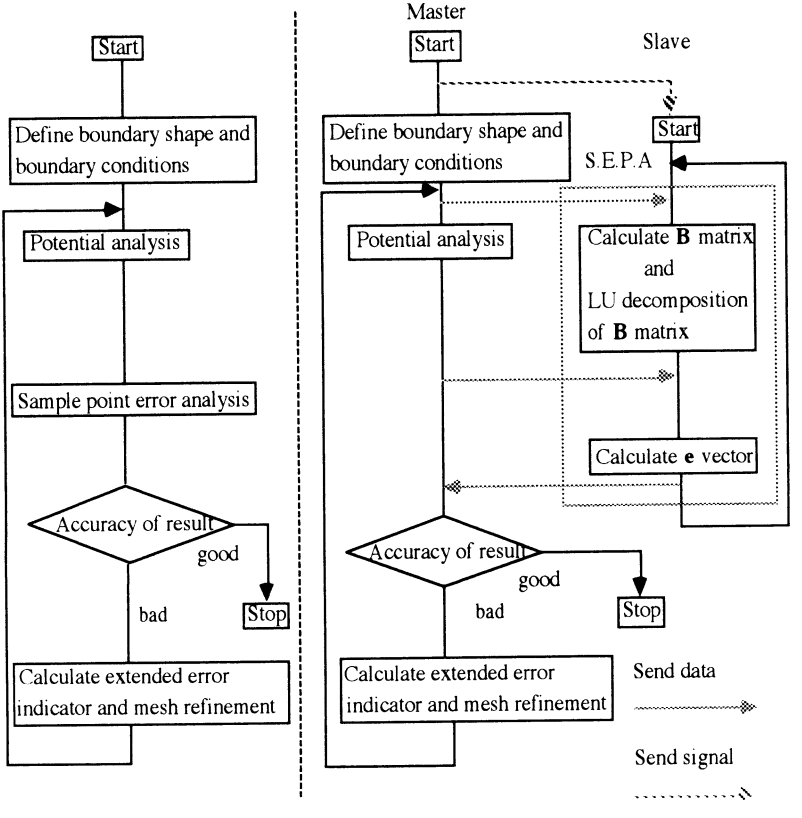
Fig. 1 Cluster computing system

Adaptive parallel computing

Kawaguchi and Kamiya [7] proposed an adaptive boundary element mesh refinement scheme by their own theoretical approach called sample point error analysis and mesh refinement by extended error indicator. It is an adaptive iteration process starting from initial very rough boundary element distribution, if possible, with minimum number of elements sufficient for definition of the given problem. The computation process for the above-stated adaptive boundary

element scheme is depicted left side in Fig. 2. The process consists of the boundary element analysis, the sample point error analysis, computation of the extended error indicator and mesh refinement. The iterative process is repeated to reach the satisfactory solution. The execution is used to be done in serial manner. We mention that, for the above computation process, we need two different matrices **A** and **B** for the boundary element analysis and error estimation, respectively.

By investigating the serial process shown left side in Fig. 2, we can partially perform the process in parallel by interchanging data, while the entire process cannot be performed independently. As the primary step, we will employ two workstations system (host and one submachine; called here *master* and *slave* system). Right side of Fig. 2 is a prototype of the partially parallel process developed here, in which the slave works for the sample point error analysis.



(a) Calculation process of adaptive BEM by sample point error analysis.

(b) Calculation process of parallel implementation of adaptive BEM by sample point error analysis.

Fig. 2 Flow chart of serial and parallel process.

The error analysis in the slave drives by a starting signal from the master, which receives mesh data and computes the matrix **B** for the error analysis. During this period, the master determines an approximate solution by the boundary element analysis and then sends the result to the slave. The slave receives the data of the result and computes the element-wise errors. Further, the master checks the error level and, if they are not satisfactory, determines the extended error indicator. The mesh is refined in the master if the solution is not sufficiently correct. The process is repeated until the acceptable solution is obtained.

Computation by example problem

Here, we consider a problem with two-dimensional L-shaped domain governed by the Laplace differential equation [7] (Fig. 3). Initial reference data are obtained using *one workstation* (HP9000 (model 712/60)) in *serial* computation. Seven iterations are performed from the initial boundary element discretization; one linear element for each side (total six elements). Figure 4 shows respective computer times required for the boundary element analysis and the sample point error analysis in each iteration. Computer time increases with progress in iteration with increased number of elements due to *h*-version adaptation. Required times for the boundary element analysis and for the adaptive error analysis are almost equal in the example considered, which indicates that, in the element adaptation process, computer time for the latter is not small. The stated analysis of computation process promotes the parallel computation for the main two subprocesses.

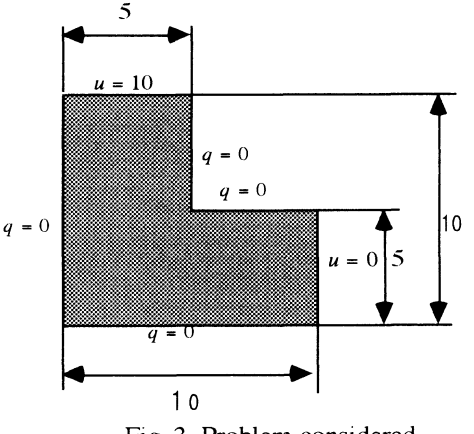


Fig. 3 Problem considered

Workstations employed for the present project are DEC5000 (model 200), HP9000 (models 712/60 and 712/80). Their performance is different each other. Parallel computations are carried out using three different combinations of the master and slave workstations as indicated in Table 1. Computation times in the parallel processing are compared with corresponding time in the serial processing using only one workstation (master workstation only). Improvement in computational efficiency is declared in Fig. 5 for each case.

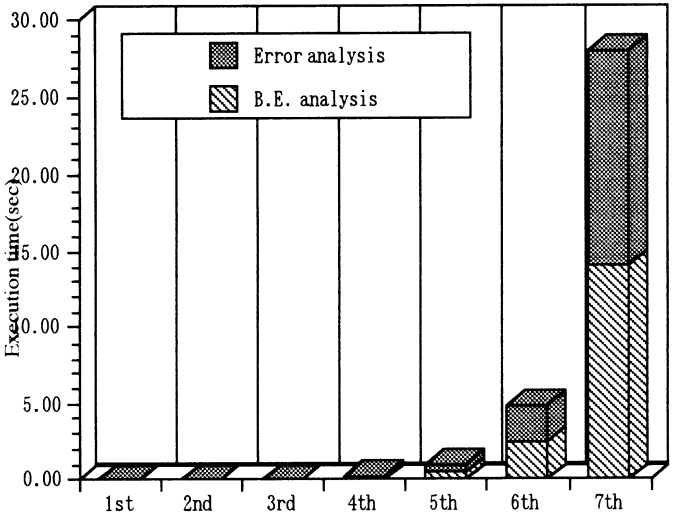


Fig. 4 Computation times for B.E. analysis and for error analysis in each iteration.

	Master	Slave
Case 1	DEC5000	HP9000(80)
Case 2	DEC5000	HP9000(60)
Case 3	HP9000(60)	HP9000(80)

Table 1. Combination of master and slave workstations

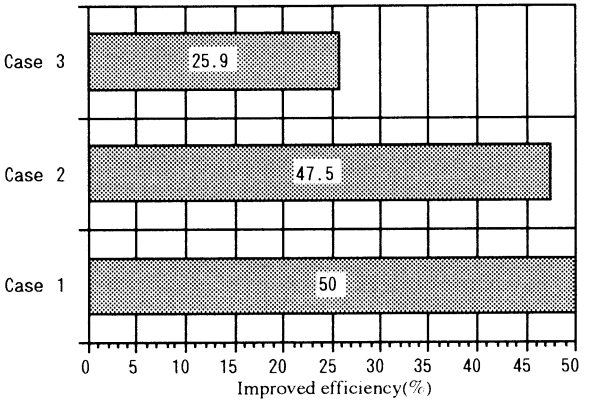


Fig.5 Improved efficiency(%)

The highest improvement is gained for the case 1, in which the difference of the performances of the master and slave is most pronounced. As predicted, since the computer load for the error analysis is conspicuous compared with the whole computation, more improvement in efficiency is realized when the performance of the slave is higher. Figure 6 represents precise time histories for the seventh iteration in the master and slave. Although, in the parallel processing, time for communication and waiting is added, it is more efficient than by the serial processing.

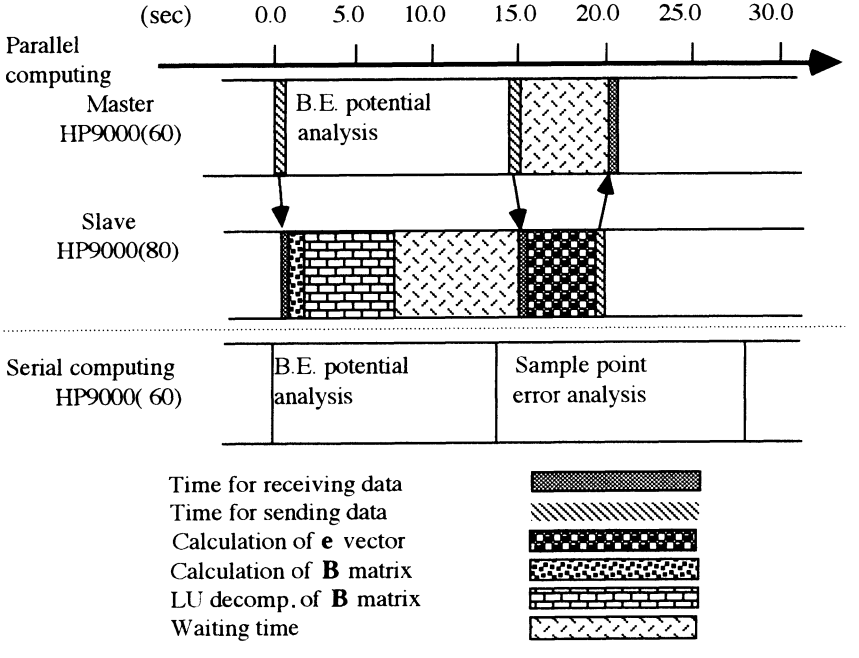


Fig. 6 Time chart of 7th adaptive iteration.

Domain decomposition for parallel computing

Domain decomposition and/or subregion scheme in the boundary element analysis is well-known for the domain composed of different kinds of materials and for improvement accuracy of solution and computational efficiency in complicated geometry problems. For the first, it is natural since each domain is governed by respective differential equation. For the latter, this is done by the reduced size of the system matrices in the discretized integral equation in the analysis. Contrary to the stated advantages, we mention that the system coefficient matrices become different for respective subdomain partition; topology of subdomain distribution affects the construction of the entire matrices. This is inconvenient for software users and for implement of general purpose boundary element software. Then we try here to perform the domain decomposition boundary element analysis in parallel, in cluster workstations

system mentioned above. For each subdomain one submachine takes in charge of simple and formal task and the entire flow of computation is controlled by the host machine. In this case, the boundary conditions on the fictitious internal boundaries must be specified and are renewed iteratively.

As the renewal of the boundary conditions on the fictitious internal boundaries, we employ the following three methods; Uzawa method (referred to Method (1)), Schwarz Neumann-Neumann method (Method(2)), and Schwarz Dirichlet-Neumann method (Method (3)). The methods via Schwarz are familiar for the boundary-value problems with help of overlapped regions. In what follows, we explain these with the domain Ω_k under consideration and Ω_l its adjacent domain.

Method (1): prescribed values; u^k and u^l
renewal scheme; $u^k_{s+1} = u^k_s - \alpha(q^l_s + q^k_s)$ and $u^l_{s+1} = u^k_{s+1}$

Method (2): prescribed values; q^k and q^l
renewal scheme; $q^k_{s+1} = q^k_s + \beta\left(u^l_s - u^k_s\right)$ and $q^l_{s+1} = -q^k_{s+1}$

Method (3): prescribed values; u^k and q^l
renewal scheme; $u^k_{s+1} = u^k_s + \gamma\left(u^l_s - u^k_s\right)$ and $q^l_{s+1} = -q^k_s$

where u and q are the potential and flux respectively for the potential-type problem. The subscript s denotes iteration number and α, β and γ are constants. It should be noticed that for the Schwarz methods one often encounters the case of the Neumann data boundary condition along the whole boundary, from which we keep away.

Example of three-subdomains decomposition

Figure 7 is a simple example problem for a single domain governed by the Laplace differential equation. For the domain decomposition parallel computing, the original domain is divided into three subdomains. Since on the upper and lower real boundaries the Neumann data are given, it is inappropriate to prescribe the Neumann data on the fictitious boundaries Γ_{12} and Γ_{23}. Then we employ following two combination methods:

(a) Method (2) on Γ_{12} and Method (1) on Γ_{23},
(b) Method (3) on Γ_{12} and Method (1) on Γ_{23}.

The boundaries (real and fictitious) of each subdomain are discretized 24 constant elements. Using initial guess 0 on the entire fictitious boundaries, we computed the iteration as shown in Fig. 8, values on the middle of the fictitious boundaries. The constant appearing in the renewal schemes are taken as for (a) $\beta = 0.08$, $\alpha = 0.4$ and (b) $\gamma = 0.4$, $\alpha = 0.4$ determined by

trial-and-error. Figure 8 indicates rapid convergence and favorable stability for the both combinations of the renewal schemes.

Conclusion

We constructed workstations cluster computing system for the parallel schemes in the boundary element solution. Two algorithms were tried: the first is for the boundary element adaptive iteration using the sample point error analysis and mesh refinement by the extended error indicator;the second is for the domain decomposition boundary elements for the ordinary subregion analysis. These not only accelerate computation efficiency in the boundary element analysis but also make it easier to implement software for the subregion boundary elements.

References

1. Yoshioka, A., Yagawa, G., Yoshimura, S., and Soneda, N., Development of Network Computing for Supercomputational Mechanics, *Trans. Japan Soc. Mech. Eng.*, (**A**), **57** (1991) 1964-1972 (in Japanese)

2. Davies, A.J., The boundary element method on a transputer network, *Proc. Boundary Elements XIII*, eds., Brebbia, C.A. and Gipson, G.S., pp. 985-994, Elsevier, 1991.

3. Davies, A.J., The boundary element method for magnetostatic problems using a distributed array processor, *Proc. Boundary Elements in Mechanical and Electrical Engineering*, eds., Brebbia, C.A. and Chaudouet-Miranda, A., pp. 407-416, Springer-Ver., 1990.

4. Ramesh, P.S., Shaw, J.G., and Lean, M.H., Simulation of charge transport in a distributed computing environment, *Proc. Boundary Element Technology VII*, eds., Brebbia, C.A., and Ingber, M.S., pp. 851-861, Elsevier, 1992.

5. Hribersek, M., and Skerget, L., Domain Decomposition for Vorticity Transport Equation in Boundary Domain Integral Method, *Boundary Element Technology X*, eds. Aliabadi, M.H., Brebbia, C.A., Dular, P., and Nicolet, A., pp. 161-168, Comp. Mech. Pub., 1995.

6. Avrash, J., Michael, O., and Rosenhouse, G., Domain Decomposition Technique Using the Boundary Spectral Strip Method, *Boundary Element Technology X*, eds. Aliabadi, M.H., Brebbia, C.A., Dular, P., and Nicolet, A., pp. 203-210, Comp. Mech. Pub., 1995.

7. Kawaguchi, K., and Kamiya, N., An Adaptive BEM by Sample Point Error Analysis, *Eng. Anal. Bound. Elem.*, **9** (1992) 255-262.

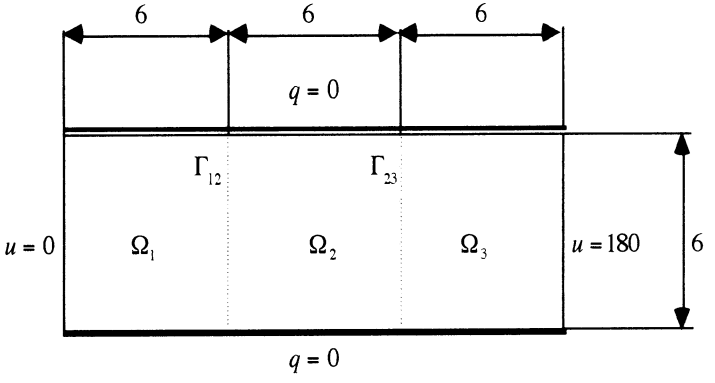
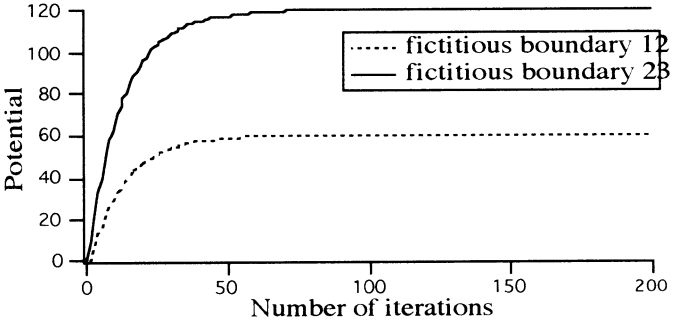
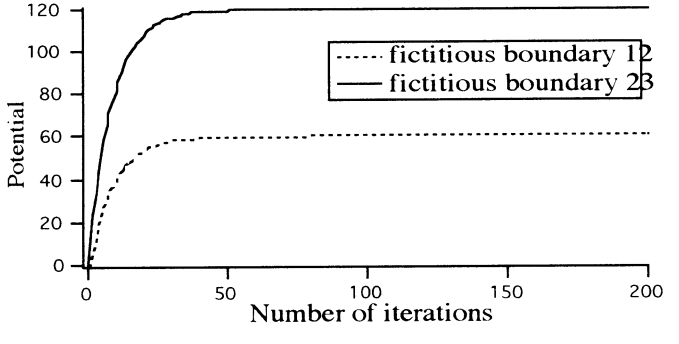


Fig. 7 Three-decomposed domain



(a) Combination of Methods (2) and (1)



(b) Combination of Methods (3) and (1)

Fig. 8 Convergence of solution in combined methods