領域分割法を用いた定常流れ問題の安定化有限要素解析
A Stabilized Finite Element Method of Stationary Flow Problems Using a Domain Decomposition Method

Hiroshi KANAYAMA, Department of Intelligent Machinery and Systems, Kyushu University,*
Daisuke TAGAMI, Department of Intelligent Machinery and Systems, Kyushu University,
Leong Yew SIAH, Department of Intelligent Machinery and Systems, Kyushu University,
Takahiro ARAKI, Department of Intelligent Machinery and Systems, Kyushu University.
*6-10-1, Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan, Email:<kanayama@mech.kyushu-u.ac.jp>.

A parallel computation system with an iterative domain decomposition method is developed for advection-diffusion problems. Using BiCGSTAB, the interface problem is solved implicitly under the constraints of temperature equivalence and heat flux continuity on the interface. A stabilized finite element method is used to analyze an advection-diffusion problem in each subdomain. The hierarchical domain decomposition method is introduced for parallel processing. The present method is successfully applied to a three-dimensional advection-diffusion problem. Numerical results show that the iterative procedure converges with a high CPU efficiency.

1. INTRODUCTION

The objective of this paper is to develop a system capable of analyzing large-scale advection-diffusion problems. As the scale and complexity of the numerical simulation problems escalate, conventional finite element method is inefficient because of its long computational time as well as the hard disk resources (i.e. memory) required. Parallel computing, which concurrently uses a number of processors has been recognized as the remedy to large-scale computer simulations.

In the present paper, a parallel computation using hierarchical domain decomposition method (HDDM; see Yagawa and Shioya [1], [2], [3]) is explained. HDDM has been successful in solving a structural problem of 1 billion degrees of freedom [1]. The method proposed in this paper, uses BiCGSTAB [5] to solve the interface problem implicitly while stabilized finite element method is used for advection-diffusion problems in the subdomains. Its implementation on parallel processing environment and numerical results are presented and discussed.

2. FUNDAMENTAL EQUATIONS

In this section, we first state the governing equations for the steady state advection-diffusion problem. Then, we introduce the stabilized finite element method used in the domain decomposition method.

2.1 Problem Statement

We consider Ω as a three-dimensional bounded domain as shown in Figure 1. The boundary of Ω consists of three parts: Γ_T denotes a temperature boundary, Γ_Q denotes a heat flux boundary, and Γ_C denotes a convective heat transfer boundary.

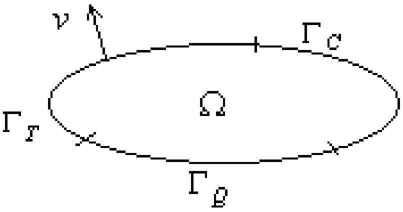

Fig. 1. Domain and boundaries.

The temperature in the domain, θ [K] is governed by the following equations:

$$\begin{cases} q = -\lambda \nabla \theta & \text{in } \Omega, \quad (1a) \\ \rho C u \cdot \nabla \theta + \operatorname{div} q = f & \text{in } \Omega, \quad (1b) \\ \theta = \theta_o & \text{on } \Gamma_T, \quad (1c) \\ q \cdot v = Q_o & \text{on } \Gamma_Q, \quad (1d) \\ q \cdot v = \alpha_c (\theta - \theta_c) & \text{on } \Gamma_C, \quad (1e) \end{cases}$$

where q [W/m^2] denotes the temperature gradient, f [W/m^3] is the internal heat generation, u [m/s] denotes the given flow velocity, ρ [kg/m^3] denotes the density, C [J/(kgK)] denotes the specific heat, and λ [W/(mK)] denotes the heat conductive coefficient.

θ_o [K] is a temperature set on Γ_T, Q_o [W/m^2] is a heat flux, α_c [W/(m^{2}K)] is a heat convective coefficient, θ_c [K] is a outside temperature, and v is the unit normal vector to the boundary. ρ, C, α_c, λ are assumed to be positive.

2.2. Fundamental Equations for Domain Decomposition Method

For simplicity, we consider a two-subdomain problem. The original domain is fictitiously partitioned into two non-overlapping subdomains, $\Omega^{(1)}$ and $\Omega^{(2)}$.

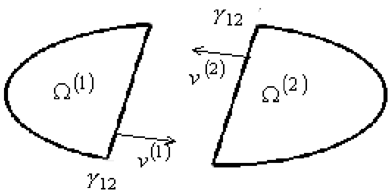

Fig. 2. Domain decomposition (two-subdomain case).

As shown in Figure 2, γ_{12} which is called interface, is the inter-subdomain boundary between $\Omega^{(1)}$ and $\Omega^{(2)}$, where $\nu^{(k)}$ (k = 1, 2) denotes the unit normal vector to $\partial\Omega^{(k)}$. The continuity of temperature and equivalence of heat flux at γ_{12} is preserved as shown below:

$$\theta^{(1)} = \theta^{(2)} = \mu \qquad \text{on } \gamma_{12}\ , \tag{2a}$$

$$q^{(1)} \cdot \nu^{(1)} = q^{(2)} \cdot \nu^{(2)} \qquad \text{on } \gamma_{12}\ . \tag{2b}$$

The domain is decomposed into tetrahedra, where $\Im_h$ represents the partition of $\overline{\Omega}$ into tetrahedra. Let us define X_h , $\Theta_h^k(g)$ and Θ_h^k as follows:

$$X_h := \left\{\eta_h \in H^1(\Omega) \cap C^o\left(\overline{\Omega}\right);\ \eta_h \in R_1(K), K \in \Im_h\right\},$$
$$\Theta_h^{(k)}(g) := \begin{Bmatrix} \eta_h^{(k)};\ \exists\ \eta_h \in X_h \text{ s.t. } \eta_h^{(k)} = \eta_h \text{ in } \overline{\Omega}^{(k)}, \\ \eta_h^{(k)} = g \quad \text{on } \Gamma_T^{(k)} \cup \gamma_{12} \end{Bmatrix},$$
$$\Theta_h^{(k)} := \Theta_h^{(k)}(0),$$

where $R_1(K)$ denotes the space of polynomials of degree 1 defined in $K \in \Im_h$, $H^1(\Omega)$ denotes the space of functions in $L^2(\Omega)$ with derivatives up to the first order, and $C^o\left(\overline{\Omega}\right)$ represents the space of continuous functions defined in $\overline{\Omega}$.

Using the stabilized finite element method proposed by Franca et. al.[4], the approximate advection-diffusion problem in the subdomain becomes the following:

For $g_0^{(k)} = \begin{cases} \theta_o & \text{on } \Gamma_T^{(k)}, \\ \mu_h & \text{on } \gamma_{12}\ , \end{cases}$

find $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_0^{(k)}\right)$ such that

$$B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) = F_h\left(\eta_h^{(k)}\right), \quad \eta_h^{(k)} \in \Theta_h^{(k)},$$

where

$$\begin{aligned} B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) &:= \left(\rho C\ (u \cdot \nabla\theta_h^{(k)})\ , \eta_h^{(k)}\right) \\ &+ \left(\lambda\nabla\theta_h^{(k)}\ , \nabla\eta_h^{(k)}\right) + \left(\alpha_c^{(k)}\theta_h^{(k)}, \eta_h^{(k)}\right)_{\Gamma_C} \\ &+ \sum_{K \in \Im_h} \left(\rho C(u^{(k)} \cdot \nabla\theta_h^{(k)}\ ), \tau_K^{(k)}\left(u^{(k)} \cdot \nabla\eta_h^{(k)}\right)\right)_K , \end{aligned}$$

$$\begin{aligned} F_h\left(\eta_h^{(k)}\right) &:= \left(f^{(k)}\ , \eta_h^{(k)}\right) - \left(Q_o^{(k)}, \eta_h^{(k)}\right)_{\Gamma_Q} \\ &+ \left(\alpha_c^{(k)}\theta_c^{(k)}, \nu_h^{(k)}\right)_{\Gamma_c} \\ &+ \sum_{K \in \Im_h} \left(f^{(k)}\ , \tau_K^{(k)}\left(u^{(k)} \cdot \nabla\nu_h^{(k)}\right)\right)_K . \end{aligned}$$

We also employ $(\cdot,\cdot)$ and $(\cdot,\cdot)_K$ to denote the L^2 -inner product in Ω and the L^2 -inner product in element K, respectively. The stability parameter $\tau_K^{(k)}$ is defined as follows:

$$\tau_K^{(k)} := \min\left\{\frac{h_K^{(k)}}{2|u|_K^{(k)}}, \frac{h_K^{(k)2}}{12\lambda/\rho C}\right\},$$

where $h_K^{(k)}$ represents the diameter of tetrahedral element K and $|u|_K^{(k)}$ denotes the maximum absolute value of the velocity.

2.3 ITERATIVE DOMAIN DECOPOSITION METHOD

The domain decomposition method (DDM) used in this research is such that the solution in the whole domain is obtained through the finite element analysis (FEA) in the subdomains. The FEA in subdomains are performed in parallel under the constraints of temperature continuity and heat flux equivalence on the interfaces among subdomains. This is done by an iterative algorithm. In this case, BiCGSTAB is used. For another formulation of DDM, see Quarteroni and Valli [7].

Let us define an asymmetric operator A as follows:

$$\begin{aligned} A(\mu_h) := -\Big(q_h^{(1)}(0,0,0,0,\mu_h) \cdot \nu^{(1)} \\ + q_h^{(2)}(0,0,0,0,\mu_h) \cdot \nu^{(2)}\Big)\Big|_{\gamma_{12}} . \end{aligned} \tag{3}$$

$q_h^{(k)}(0,0,0,0,\mu_h)$ in the above equation is defined by using $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_1^{(k)}\right)$ which is the solution for the following equation:

$$B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) = 0\ , \ \text{for } \eta_h^{(k)} \in \Theta_h^{(k)}\ ,$$

such that $g_1^{(k)}$ equals to zero on $\Gamma_T^{(k)}$, but takes the value of μ_h on γ_{12} , while $q_h^{(k)}(0,0,0,0,\mu_h) = -\lambda\nabla\theta_h^{(k)}$.

The interface problem is defined as the following:

$$\begin{aligned} A(\mu_h) = \Big(q_h^{(k)}(f, Q_o, \theta_o, \theta_c, 0) \cdot \nu^{(1)} \\ + q_h^{(k)}(f, Q_o, \theta_o, \theta_c, 0) \cdot \nu^{(2)}\Big)\Big|_{\gamma_{12}} . \end{aligned} \tag{4}$$

$q_h^{(k)}(f, Q_o, \theta_o, \theta_c, 0)$ in the above equation is defined by using $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_2^{(k)}\right)$ which is the solution of

$$\begin{aligned} B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) &= \left(f^{(k)}\ , \eta_h^{(k)}\right) \\ &- \left(Q_o^{(k)}, \eta_h^{(k)}\right)_{\Gamma_Q} + \left(\alpha_c^{(k)}\theta_c^{(k)}, \nu_h^{(k)}\right)_{\Gamma_c} \\ &+ \sum_{K \in \Im_h} \left(f^{(k)}\ , \tau_K^{(k)}\left(u^{(k)} \cdot \nabla\nu_h^{(k)}\right)\right)_K , \quad \text{for } \eta_h^{(k)} \in \Theta_h^{(k)}, \end{aligned}$$

such that $g_2^{(k)}$ equals to θ_o on $\Gamma_T^{(k)}$, but takes the value of zero on γ_{12} , while $q_h^{(k)}(f, Q_o, \theta_o, \theta_c, 0) = -\lambda\nabla\theta_h^{(k)}$.

As mentioned above, advection-diffusion problems produce an asymmetric problem on the interface. Consequently, BiCGSTAB algorithm is used to solved the interface problem, and the algorithm is described as follows:

```
Set $\mu_h^0 = 0$;
  Compute $r_h^0$;
    $\beta^{-1} := 0$;
  for n=0, 1, 2, …;
    $p_h^n = r_h^n + \beta^{n-1}\left(p_h^{n-1} - \zeta^{n-1} A\left(p_h^{n-1}\right)\right)$;
    $\alpha^n = \dfrac{\left(r_h^0, r_h^n\right)}{\left(r_h^0, A\left(p_h^n\right)\right)}$;
    $t_h^n = r_h^n - \alpha^n A\left(p_h^n\right)$;
    $\zeta^n = \dfrac{\left(A\left(t_h^n\right), t_h^n\right)}{\left(A\left(t_h^n\right), A\left(t_h^n\right)\right)}$;
    $\mu_h^{n+1} = \mu_h^n - \alpha^n p_h^n - \zeta^n t_h^n$;
    $r_h^{n+1} = t_h^n - \zeta^n A\left(t_h^n\right)$;
    if $\dfrac{\| r_h^{n+1} \|}{\| r_h^0 \|} \le \varepsilon$, break;
    $\beta^n = \dfrac{\alpha^n}{\zeta^n} \dfrac{\left(r_h^0, r_h^{n+1}\right)}{\left(r_h^0, r_h^n\right)}$;
End .
```

The initial residual, r_h^0 is defined by

$$r_h^0 := -\left(q_h^{(1)}\left(f, Q_o, \theta_o, \theta_c, \mu_h^0\right) \cdot \nu^{(1)} + q_h^{(2)}\left(f, Q_o, \theta_o, \theta_c, \mu_h^0\right) \cdot \nu^{(2)}\right)\Big|_{\gamma_{12}} . \tag{5}$$

$q_h^{(k)}\left(f, Q_o, \theta_o, \theta_c, \mu_h^0\right)$ in the right hand side of (5) is defined by using $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_3^{(k)}\right)$ which is the solution for the following equation:

$$B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) = \left(f^{(k)}, \eta_h^{(k)}\right) - \left(Q_o^{(k)}, \eta_h^{(k)}\right)_{\Gamma_Q} + \left(\alpha_c^{(k)} \theta_c^{(k)}, v_h^{(k)}\right)_{\Gamma_c} + \sum_{K \in \Im_h} \left(f^{(k)}, \tau_K^{(k)}\left(u^{(k)} \cdot \nabla v_h^{(k)}\right)\right)_K, \quad \text{for } \eta_h^{(k)} \in \Theta_h^{(k)},$$

such that $g_3^{(k)}$ equals to θ_o on $\Gamma_T^{(k)}$, but takes the value of μ_h^0 on γ_{12}, while $q_h^{(k)}\left(f, Q_o, \theta_o, \theta_c, \mu_h^0\right) = -\lambda \nabla \theta_h^{(k)}$.

It is to be noted that, $A\left(p_h^n\right)$ is defined as the following:

$$A\left(p_h^n\right) := -\left(q_h^{(1)}\left(0,0,0,0, p_h^n\right) \cdot \nu^{(1)} + q_h^{(2)}\left(0,0,0,0, p_h^n\right) \cdot \nu^{(2)}\right)\Big|_{\gamma_{12}} . \tag{6}$$

$q_h^{(k)}\left(0,0,0,0, p_h^n\right)$ in the right hand side of (6) is defined by using $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_4^{(k)}\right)$ which is the solution of

$$B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) = 0, \quad \text{for } \eta_h^{(k)} \in \Theta_h^{(k)},$$

such that $g_4^{(k)}$ equals to zero on $\Gamma_T^{(k)}$, but takes the value of p_h^n on γ_{12}, while $q_h^{(k)}\left(0,0,0,0, p_h^n\right) = -\lambda \nabla \theta_h^{(k)}$.

Similarly, $A\left(t_h^n\right)$ is defined as the following:

$$A\left(t_h^n\right) := -\left(q_h^{(1)}\left(0,0,0,0, t_h^n\right) \cdot \nu^{(1)} + q_h^{(2)}\left(0,0,0,0, t_h^n\right) \cdot \nu^{(2)}\right)\Big|_{\gamma_{12}} . \tag{7}$$

$q_h^{(k)}\left(0,0,0,0, t_h^n\right)$ in the right hand side of (7) is defined by using $\theta_h^{(k)} \in \Theta_h^{(k)}\left(g_5^{(k)}\right)$ which is the solution of

$$B_h\left(\theta_h^{(k)}, \eta_h^{(k)}\right) = 0, \quad \text{for } \eta_h^{(k)} \in \Theta_h^{(k)},$$

such that $g_5^{(k)}$ equals to zero on $\Gamma_T^{(k)}$, but takes the value of t_h^n, while $q_h^{(k)}\left(0,0,0,0, t_h^n\right) = -\lambda \nabla \theta_h^{(k)}$.

Under the constraints as mentioned in equations (2a) and (2b), this algorithm described above enables us to compute the solution without the need of forming explicitly the stiffness matrix of *A*. Computations of (5), (6) and (7) are performed independently through the respective FEA in the subdomains, thus a high parallel computational efficiency may be achieved. In the practical computation, $q_h^{(k)} \cdot \nu^{(k)}$ in (5), (6), (7) is given by the integral over each subdomain based on the Gauss divergence theorem.

3. HIERACHICAL DOMAIN DECOMPOSITION METHOD

HDDM (see [1], [2], [3]), as shown in Figure 3, divides the whole domain into parts, which are further decomposed into smaller domains called subdomains.

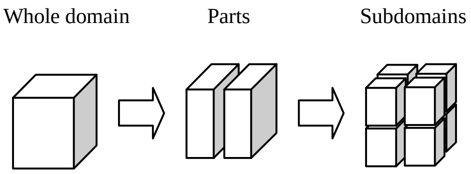

Fig. 3. Hierarchical domain decomposition.

HDDM provides a better parallel computational platform as processors are divided into 3 groups, “Grand Parent”, “Parent” and “Child” (see [2]). Figure 4 illustrates the practical implementation of the hierarchical organized processors in this research.

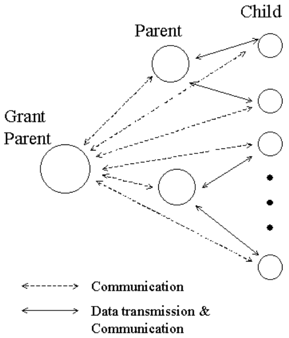

Fig. 4. Grand Parent-Parent-Child Processors – data flow and communication.

One processor is assigned as “Grand Parent”, a few as “Parent”, and the other as “Child”. The number of processors assigned as Parent is the same as that of parts. The role of Grand Parent is to organize all the communications between processors. Parent prepares mesh data, manages FEA results, and coordinates the BiCGSTAB iteration, including convergence decision for BiCGSTAB. Parent send data to Child, where FEA is carried out. After the FEA, Child sends the result to Parent.

Workload at processors is an important issue in parallel computing. There are two basic ways to handle the workload; they are static load balancing and dynamic load balancing.

In static load balancing, ideally, all processors would have the same workload. If the subdomains are distributed evenly over the Childs, and if the whole domain is not partitioned evenly over the subdomains, this might result in different times to completion for the FEA. Some Childs might need to wait for others to complete. To overcome this problem, computational load should be balanced. If the computational time is proportional to the degrees of freedom (DOF), this can be achieved by dividing subdomains into a similar DOF.

Unlike static load balancing, in dynamic load balancing, the workload of Child is unknown. Child would request for work when it is idle. Workload is balanced during program execution. However, the computational time increases as the stiffness matrix in each subdomain has to be re-constructed in every iteration.

The workload balancing used in this method is a combination of static and dynamics load balancing, called Hybrid Load Balancing. Initially, workloads (i.e. subdomains) are assigned to idling Childs (i.e. hence, dynamics). Once a Child receives data from a particular Parent, the Child is “officially” assigned to that particular Parent (i.e. hence, static).

4. NUMERICAL RESULTS

In this section, we examine the effectiveness of the iterative domain decoposition method discussed in the previous section. The numerical test is essentially of pure advection flows as it is subjected to a very high Peclet number.

Case I - Comparison between HDDM and DDM.

We consider a cubic model of 1 x 1 x 0.015625 [m^3]; where $-0.5 \leq x,\ y \leq 0.5$ [m], and $0.0 \leq z \leq 0.015625$ [m]. Only temperature setting boundary condition is considered in this problem. Figure 5 illustrates the boundary conditions on the *x*-*y* plane of the problem.

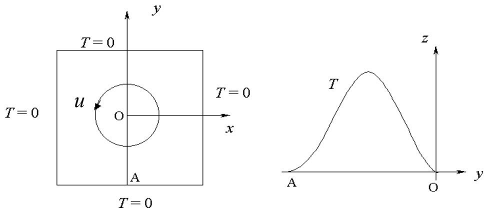

Fig.5. Boundary conditions on the *x*-*y* plane.

These boundary conditions are applied identically along the *z*-axis. Within the domain, the flow velocity components are given by *u* = (-*y*, *x*, 0) as shown in Figure 5. The temperatures along the external boundary are set to be 0 [^{o}C]. Along the internal boundary OA, the temperature *T* [^{o}C] is set to be the following:

$$T = \frac{1}{2}\left(\cos\left(4\pi y + \pi\right) + 1\right), \quad -0.5 \leq y \leq 0\ .$$

The material properties of the model are: ρ =1 [kg/m^3], *C*=1 [J/(kgK)], and λ =10^{-6} [W/(mK)]. Internal heat generation is not considered (i.e. *f* = 0[W/m^3]) in this problem.

The temperature is approximated by tetrahedral linear elements. The domain is decomposed into 120 subdomains, where the total DOF of the mesh is 49,923; while the total interface DOF is 7,875. The convergence criterion for BiCGSTAB is set to be less than 10^{-7}.

HDDM and DDM were used in Case I. 11 processors (Alpha 533MHz) were used in the HDDM analysis; where 1 processor as the Grand Parent, 1 processor as the Parent and the remaining 9 processors as the Childs. In the DDM analysis, only 1 processor (Alpha 533Mhz) was used.

Table 1. Computational data.

Type	No. of Iterations	Computational Time [hours]	Average CPU Efficiency [%]
HDDM	625	6.57	94.96
DDM	770	32.70	100

Table 1 shows the number of iterations, computational time as well as the average CPU efficiency of the Child, which is defined by

$$\frac{\text{Working Time}}{\text{Working Time} + \text{Idling Time}} \text{ x } 100\,[\%]\,.$$

Table 1 indicates that the computational time of HDDM is approximately 1/5 that of the DDM. An average CPU efficiency of 94.96% is achieved in HDDM.

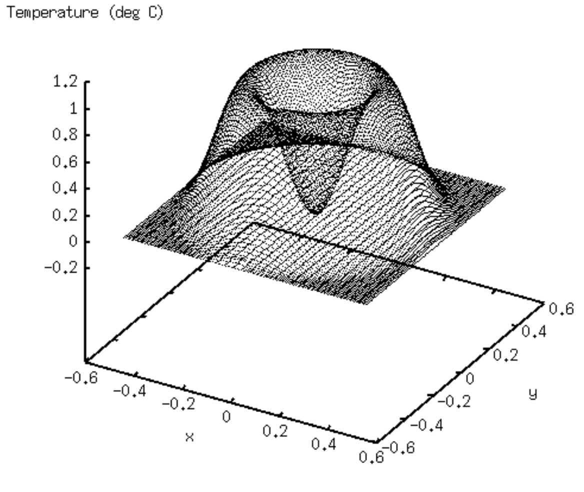

Fig. 6. Temperature distribution on the *x*-*y* plane (Case I).

Figure 6 illustrates the three-dimensional temperature distribution on the *x*-*y* plane. As advection is dominant, the temperature distributed along boundary OA is seen to be rotating around *z*-axis. Figure 7 shows the temperature distribution along the line *x*=0 and *z*=0. The relative error between the HDDM and DDM results is shown in the same figure as well. The relative error is defined by

$$\frac{|\theta_{HDDM} - \theta_{DDM}|}{|\theta_{HDDM}|} \text{x}\,100\,[\%],$$

where $|\theta_{HDDM} - \theta_{DDM}|$ denotes the absolute value of the difference between the HDDM result and the DDM result at each nodal point, and $|\theta_{HDDM}|$ is the absolute temperature of the HDDM result at each nodal point. The maximum relative error between HDDM and DDM results on the line *x*=0 and *z*=0, is 5.5x10^{-4}%; while the maximum relative error in the model is 4.1x10^{-3}%.

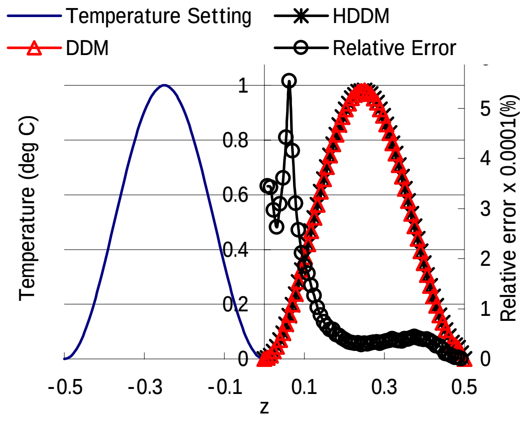

Fig. 7. Temperature distribution on *x*=0 & *z*=0 (Case I)

Figure 8 shows the relative residual history of BiCGSTAB. The HDDM and the DDM do not produce the same relative residual curve. This is because they are two different methods, where the sequences of subdomain computation are different. However, the relative residuals of both methods converge to below 10^{-7}.

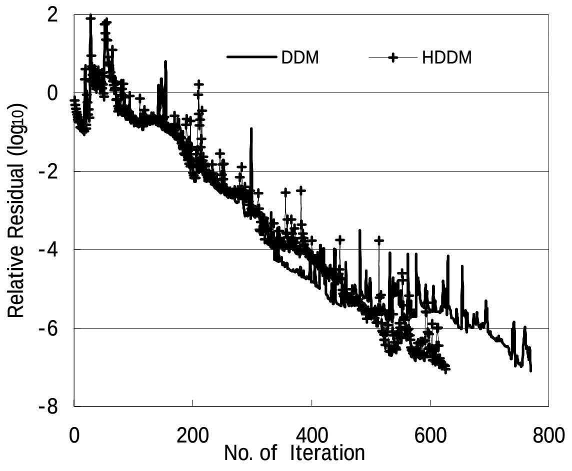

Fig. 8. Relative residual history (Case I).

Case II – Finer Mesh Analysis.

The same problem was further analyzed by using a finer mesh; where a cubic of 1 x 1 x 1 m^3 was considered. Figure 9 shows the finer mesh, where the model is decomposed into 3 parts and 1800 subdomains (i.e. 600 subdomains per part). The total DOF of the model is 357,911, the total number of elements is 257,250, and the total interface DOF is 143,855.

Using this mesh, HDDM analysis was performed on 19 Alpha 533Hz processors; where 1 processor as the Grand Parent, 3 processors as the Parents and the remaining 15 processors as the Childs. The BiCGSTAB convergence was set to be below 10^{-7}. It took 1,302 iterations to converge and the total computational time is 90.6 hours. The average CPU efficiency of the Child in Case II is 98.22%.

The residual history curve is shown in Figure 10. Although the relative residual is oscillating, the BiCGSTAB procedure converges. Figure 11 illustrates the temperature contour. The temperatures set along boundary OA can be clearly seen to be revolving around the *z*-axis in this figure.

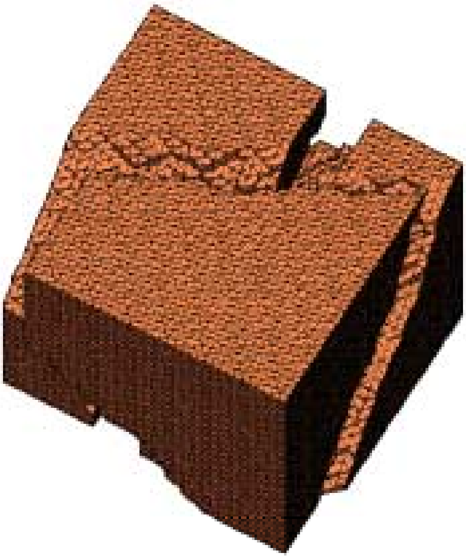

Fig. 9. Mesh and its decomposition (Case II).

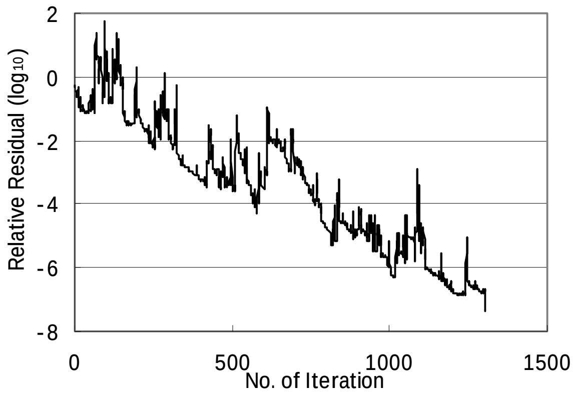

Fig. 10. Relative residual history (Case II).

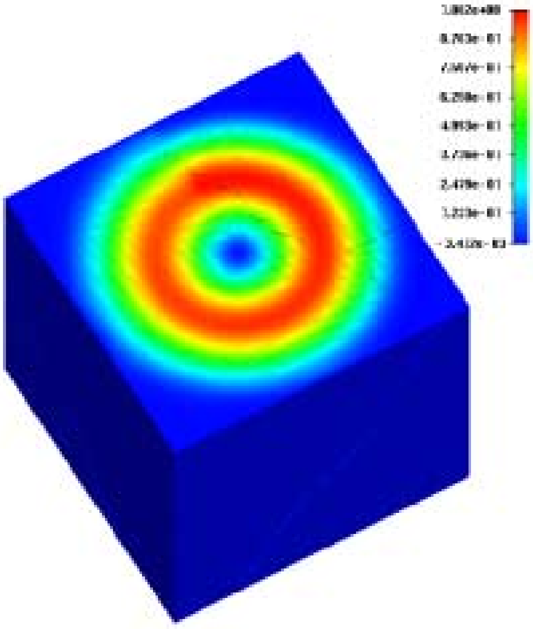

Fig. 11. Temperature contour (Case II).

5. CONCLUSIONS

The parallel computation system using an iterative domain decomposition method is successfully developed in the present study. The hierarchy domain decomposition method is adopted as the fundamental algorithm for the parallel processing. The system is capable of analyzing advection-diffusion problems using the stabilized finite element method. Numerical results show that the system achieves a high CPU efficiency.

Acknowledgment

We would like to express our gratitude to ADVENTURE Project [6] Committee for offering us the permission to use their Compaq Alpha Clusters in the numerical computations.

References

[1] Yagawa, G. and Shioya, R., *Iterative Domain Decomposition FEM with Preconditioning Technique for Large Scale Problem*. ECM'99, Progress in Experimental and Computational Mechanics in Engineering and Material Behaviour, (1999), pp. 255-260.

[2] Yagawa, G. and Shioya, R., Super Parallel Finite Element Analysis, (In Japanese), Asakura Shoten, (1998).

[3] Yagawa, G. and Shioya, R., *Parallel Finite Elements on a Massively Parallel Computer with Domain Decomposition*, Computer Systems in Engineering, 4 (1993), pp. 495-503.

[4] Franca, L., Frey, S. and Hughes, T., *Stabilized Finite Element Methods: I. Application to the Advective-Diffusive Model*, Computer Methods in Applied Mechanics and Engineering, 95 (1995), pp.253-277.

[5] Fujino, S. and Chang, S. L., The Mathematics for Iterative Algorithm, (In Japanese), Asakura Shoten, (1996), pp. 48.

[6] ADVETURE Project (Sponsored by the Japan Society for the Promotion of Science, a JSPS-RFTF Program), <http://adventure.q.t.u-tokyo.ac.jp/>.

[7] Quarteroni, A. and Valli, A.., Domain Decomposition Methods for Partial Differential Equations, Clarendon Press, 1999.