ROBUST AND EFFICIENT PARALLEL ITERATIVE SOLVERS FOR ULTRA LARGE SCALE STRUCTURAL ANALYSIS

東京大学・環境学専攻　吉村忍（Shinobu YOSHIMURA）
Institute of Environmental Studies, University of Tokyo
九州大学・知能機械システム部門　塩谷隆二（Ryuji SHIOYA），
荻野正雄（Masao OGINO）
Department of Intelligent Machinery & Systems Engineering,
Kyushu University

1. INTRODUCTION

Various general-purpose computational mechanics systems have been developed in the last three decades to quantitatively evaluate mechanical / physical phenomena such as deformation of solid, heat transfer, fluid flow and electromagnetics. Nowadays such systems are regarded as infrastructural tools for the present industrialized society. The existing systems, however, can not be used with massively parallel processors (MPPs) with the order of 100-10,000 processing elements (PEs), as they were developed for single-processor computers. Neither can the current systems be used in heterogeneous parallel and distributed computing environments such as the Grid. Owing to the fact, they can deal with only medium scale problems with millions degrees of freedom (DOFs) at most.

The ADVENTURE project [1, 2] was one of the research projects in the "Computational Science & Engineering" field selected for the "Research for the Future (RFTF)" Program sponsored by the Japan Society for the Promotion of Science (JSPS) [3,4] during 1997-2002. The project is continuously going on as an open source software development project. In the project we have been developing an advanced general-purpose computational mechanics system named ADVENTURE since August 1997. The system was designed to be able to analyze a three-dimensional finite element model of arbitrary shape with 10-100 million DOF mesh, and additionally to enable parametric and non-parametric shape optimization [5, 6]. The first version of the ADVENTURE system has been released from the project website [1] as open source software since March 2002. About 1,800 registered users in academia and industries are now using the programs, while one private company has developed and released its commercial version named ADVENTUREcluster [5, 6].

Domain-decomposition-based parallel algorithms are implemented in pre-processes (domain decomposition), main processes (system matrix assembling and solutions) and post-process (visualization), respectively. Especially the hierarchical domain decomposition method with a preconditioned iterative solver (HDDM) [9-12] is adopted in two of the main modules for solid analysis and thermal conduction analysis, named ADVENTURE_Solid and ADVENTURE_Thermal. The employed preconditioner is the Balancing Domain Decomposition (BDD) type method [13-18]. To efficiently solve a coarse space problem derived from equilibrium conditions for singular problems associated with a number of subdomains appeared in the BDD formulation, an incomplete factorization based parallel direct solver is employed. The ADVENTURE_Solid has been successfully implemented on a single PC, PC clusters and massively parallel processors such as Hitachi SR8000/MPP [2, 11, 12, 19]. Recently, this solid analysis module has been implemented with minor modification on the Earth Simulator consisting of 256 nodes, i.e. 2,048 vector-type PEs of theoretical peak performance of 16 TFLOPS, and succeeds in solving an elastostatic problem of a nuclear pressure vessel model of 100 million DOFs in 8.5 minutes with 5.1 TFLOPS, which is 31.8 % of the peak performance and over 80% parallel efficiency.

2. OVERVIEW OF ADVENTURE SYSTEM

The ADVENTURE system consists of pre-, main- and post-processing modules and design modules that can be used in various kinds of parallel and distributed environments [1, 2]. The system employs a hierarchical domain decomposition method (HDDM) [9-12] based massively parallel algorithm as one of the major solution algorithms in order to handle a huge-scale finite element model over 10-100 million DOFs efficiently. The system employs module-based architecture and consists of 19 modules. The pre-process modules include the surface patch generator which converts geometry model data into a collection of triangular surface patch data, named ADVENTURE_TriPatch, a tetrahedral mesh generator [20, 21], i.e. ADVENTURE_TetMesh, an attachment tool of boundary conditions and material properties onto the mesh, i.e. ADVENTURE_BCtool, and a domain decomposer of a finite element model, i.e. ADVENTURE_Metis. The kernels of the ADVENTURE_Metis are a graph partitioning tool METIS and its parallel version ParMETIS developed in the University of Minnesota [22, 23]. The main process modules, i.e. solvers include an implicit elastic-plastic analysis module named ADVENTURE_Solid [11-12, 19] which enables large-deformation and implicit dynamic analyses, a thermal conductive analysis module named ADVENTURE_Thermal, a thermal-fluid analysis module named ADVENTURE_Fluid, a magnetic analysis module named ADVENTURE_Magnetic [24], an explicit impact analysis module named ADVENTURE_Impact, and a rigid plastic analysis module named ADVENTURE_Forge. The post process module named ADVENTURE_Visual is for parallel visualization of analysis results [25]. Figure 1 shows the configuration of the ADVENTURE modules.

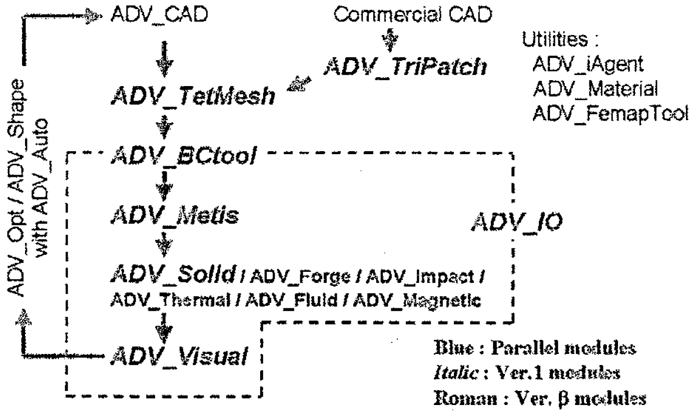

Fig.1 Configuration of ADVENTURE modules

3. PARALLEL ALGORITHMS IMPLEMENTED IN ADVENTURE_Solid

One of the key technologies of the ADVENTURE_Solid is the HDDM, which enables parallel finite element calculations on various kinds of computing environments [9-12]. Basically in the HDDM, force equivalence and continuity conditions among subdomains are satisfied through iterative calculations such as the Conjugate Gradient (CG) method. Therefore it is indispensable to reduce the number of iterations by adopting some appropriate preconditioning technique especially for solving large-scale ill-conditioned problems. The Neumann-Neumann algorithm (N-N) [13] is known as efficient domain decomposition preconditioner for unstructured subdomains. However, its convergence deteriorates with the increasing number of subdomains due to lack of a coarse space problem which takes care of global propagation of error. The Balancing Domain Decomposition (BDD) based N-N algorithm proposed by Mandel [14] shows that the equilibrium conditions for the singular problems on subdomains result in simple and natural construction of a coarse space problem and that its construction is purely algebraic. The BDD has been applied to solve various phenomena [15, 16]. There are also several researches on parallelization of the BDD and also the FETI (Finite Element Tearing and Interconnecting) [26-30]. However, most problems solved there are still medium scale ones such as sub-millions to one million DOFs. As the DOFs of the coarse space problem is directly related to the number of subdomains, it is indispensable to consider the parallelization of the solution process of the coarse space problem as well when solving large-scale problems. The Salinas system [31] , which employed the FETI-DP method [30], is succeeded in solving large-scale problems such as over 100 million DOF mesh of optical shutter model [32]. It shows good performance but does not seem to include load-balancing techniques. In the present study, an incomplete parallel direct method and the HDDM are adopted.

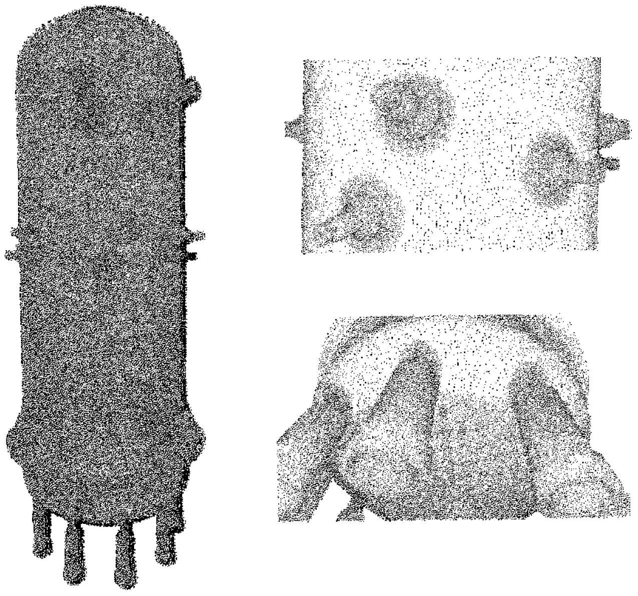

Fig.2(a) 35 million DOF mesh of ABWR model

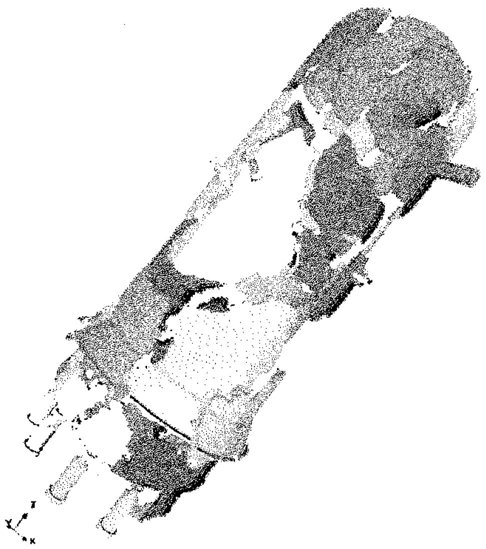

Fig.2(b) Part decomposition of ABWR vessel model

3.1 H3ierarchical Domain Decomposition Method (HDDM)

In Domain Decomposition Methods (DDM), an analysis model, i.e. a finite element mesh with boundary conditions and material properties, is subdivided into a number of subdomains. The HDDM employs a hierarchical technique to implement the DDM on various parallel computers. In the HDDM, a group of processing elements (PEs) are subdivided into the following three sub-groups : one Grand Parent PE (Grand), several Parent PEs (Parent or Parents), and many Child PEs (Child or Children). At the same time, the analysis model is subdivided into some 'parts' whose number is the same as the number of the Parents. Each part is further subdivided into a number of subdomains, the number of which can be much larger than that of the Children. Figure 2(a) shows a 35 million DOFs mesh for an Advanced Boiling Water Reactor (ABWR) vessel model, generated by the ADVENTURE_TriPatch and the ADVENTURE_TetMesh. Figure 2(b) illustrates an example of the hierarchically decomposed mesh generated by the ADVENTURE_Metis. Here only 128 parts decomposition is shown and each part is divided into subdomains. Owing to the HDDM algorithm, large-scale analysis data can be easily handled by increasing the number of the Parents. The main roles of the three kinds of processors are summarized as follows. The Grand manages all PEs, i.e. synchronization and calculation of the sum of vectors spread over a number of Children. Each Parent stores mesh data and material properties of subdomains, sends / receives subdomains data to / from Child, and iterates loops of the CG method. Each Child performs finite element calculations of the subdomains received from the Parent, and sends analyzed data back to the Parent. Figure 3(a) shows the schematic data flow among PEs.

According to the design concept of the HDDM, most computation is assigned to the Children, while most communication occurs in between Parents and Children. Varying the number of Parents and Children for different kinds of parallel computers, the present HDDM-based system can easily achieve high parallel performance. In the HDDM architecture, thanks to the dynamic load balancing technique among Child processors, high parallel performance can be achieved even in heterogeneous computer environments. However in this mode, an amount of data communication between Child and Parent tends to

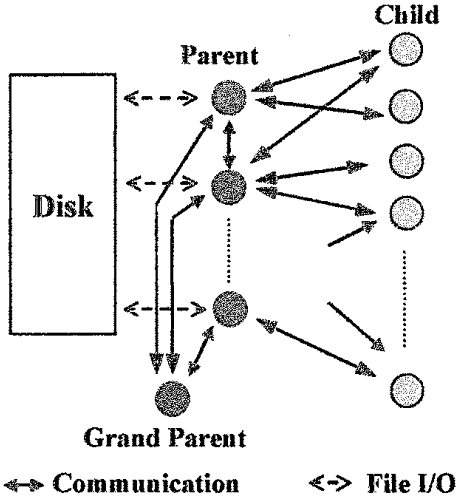

Fig.3 (a) Schematic data flow in h-mode

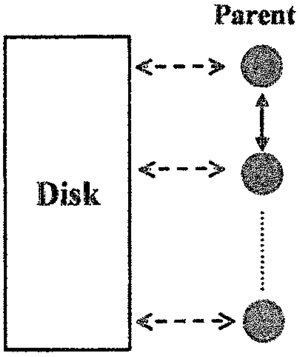

Fig.3(b) Schematic data flow in p-mode

be large. To reduce such data communication among Children and Parents, it is useful to adopt static load balance. This analysis mode shown in Figure 3(b) is called parallel processors mode (p-mode), while the original analysis mode as shown in Figure 3(a) is named hierarchical processors mode (h-mode)

3.2 B3alancing Domain Decomposition (BDD)

The BDD algorithm is based on the DDM with a preconditioned iterative solver. After eliminating interior DOFs of local subdomain matrices, the problem to be solved is reduced onto the interface DOFs of subdomains. The reduced matrix is so-called Schur complement. The reduced problem is also called the interface problem, and is to be solved by a preconditioned iterative method. There are two main methods as preconditioner, i.e. local subdomain correction and coarse grid correction in a coarse space. Main elemental calculations appeared in the BDD algorithm are described as follows :

a) $\boldsymbol{Ku} = \boldsymbol{f}$ is a linear algebraic system to be solved, where $\boldsymbol{K}$ is the global stiffness matrix, assumed to be positive definite.

b) $\boldsymbol{S}\boldsymbol{u}_B = \boldsymbol{g}$ is the reduced system, where $\boldsymbol{S} = \sum_{i=1}^{N} \boldsymbol{R}_i^T \boldsymbol{S}_i \boldsymbol{R}_i$ is the Schur complement, assumed to be positive definite, and $\boldsymbol{S}_i$ is the local Schur complement of i-th subdomain $i = 1,\ldots,N$, assumed to be positive semi-definite.

c) $\boldsymbol{R}_i$ is the matrix of the global to local DOFs corresponding to interface mapping for i-th subdomain, assumed to satisfy $\boldsymbol{R}_i \boldsymbol{R}_i^T = \boldsymbol{I}$.

d) $\boldsymbol{D}_i$ is a weighting matrix for i-th subdomain, assumed to form decomposition of unity and satisfy $\sum_{i=1}^{N} \boldsymbol{R}_i^T \boldsymbol{D}_i \boldsymbol{R}_i = \boldsymbol{I}$.

e) $\boldsymbol{Z}_i$ is the local coarse space of i-th subdomain, that contains all potential local singularities.

f) $\boldsymbol{R}_0$ is the weighted restriction from the global to coarse DOFs, defined by $R_0^T = \left[R_1^T D_1^T Z_1, \ldots, R_N^T D_N Z_N\right]$.

g) $\boldsymbol{P}$ is the S-orthogonal projection onto the coarse space, defined by $\boldsymbol{P} = \boldsymbol{QS},$ *where* $\boldsymbol{Q} = \boldsymbol{R}_0^T \boldsymbol{S}_0^{-1} \boldsymbol{R}_0$, and $\boldsymbol{S}_0 = \boldsymbol{R}_0 \boldsymbol{S} \boldsymbol{R}_0^T$ is a coarse grid operator, assumed to be positive definite.

Various domain decomposition methods contain a process of solving a reduced system using iterative methods such as the preconditioned CG method. At each step, the DDM or the HDDM requires to solve the following auxiliary problem :

$$Mz = r \,, \tag{1}$$

where M is a symmetric positive definite matrix called preconditioner and r is a residual vector in each iterative step. The BDD preconditioned operator is described [14,16] by :

$$M^{-1}S = P + (I - P)\left(\sum_{i=1}^{N} T_i\right)(I - P)^T \,, \tag{2}$$

where T_i is the local subdomain correction and $I - P$ is the coarse grid correction. If $P^T r = 0$, which means a residual vector has no components of the coarse space, Eq. (22) can be simplified as :

$$M^{-1}S = (I - P)\left(\sum_{i=1}^{N} T_i\right) . \tag{3}$$

The original BDD employs the N-N type algorithm as local subdomain correction with a two-level weighted sum of the inverses of S_i matrices [14]. To calculate the inverse of them, the Moore-Penrose pseudo-inverse or some regularization is required since S_i matrices are typically singular. However the Moore-Penrose pseudo-inverse takes high computational cost, while the regularization is less accurate. To overcome both issues of computation cost and accuracy simultaneously, we choose the diagonal scaling preconditioner for S_i as local subdomain correction. Since the local subdomain correction is applied subdomain-wise, its parallel algorithm is basically compatible to the HDDM.

3.3 Parallelization of BDD

The coarse grid correction for elastic problems becomes the Galerkin projection onto a coarse space derived from rigid body mode. For parallelizing the BDD, we decompose the BDD algorithm into the following two stages : construction of the coarse grid operator S_0, and operation of the coarse grid correction $I - P$. At first, the coarse grid operator S_0 is applied in subdomain-based blocks, and then its process is completely parallelized subdomain-wise. That is, S_0 is described by :

$$
\begin{aligned}
S_0 &= R_0\left(\sum_{k=1}^{N} R_k^T S_k R_k\right)R_0^T \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} R_{0i} R_k^T S_k R_k R_{0j}^T \quad , \qquad (4)
\end{aligned}
$$

where R_{0i} is the weighted restriction from the global to the local coarse DOFs of subdomain i, defined by $R_{0i} = \left[Z_i^T D_i^T R_i\right]$. Figure 4(a) schematically illustrates the data flow in parallel construction of the coarse grid operator for the HDDM.

Secondly, the coarse grid correction is applied to solve a linear system equation whose coefficient matrix is derived from the coarse grid operator. It should be noted here that this process can not be parallelized subdomain-wise. Thus we employ the following method. The coarse grid correction is implemented in each iteration with its own right-hand side vector. Here in order to reduce computation time, an LU factorized coarse matrix in the first iteration can be kept, and then the forward elimination and the backward substitution of the coarse system are applied after the second iteration.

The number of DOFs of the coarse space is directly related to the number of subdomains, and tends to become large when solving large-scale problems. To solve the coarse space problem, we adopt an incomplete factorization based parallel direct method. The coarse grid operator is factorized incompletely. In general, such an incompletely factorized operator is used together with some iterative calculation to compensate the

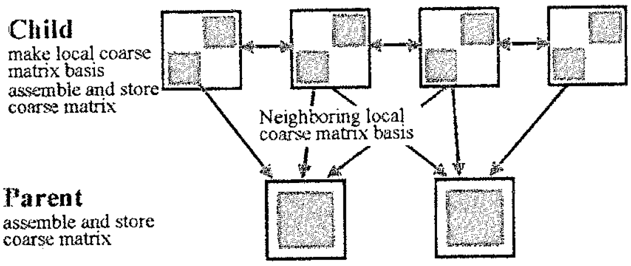
Child
make local coarse matrix basis
assemble and store coarse matrix
Neighboring local coarse matrix basis
Parent
assemble and store coarse matrix

Fig.4(a) Data flow of constructing coarse grid correction

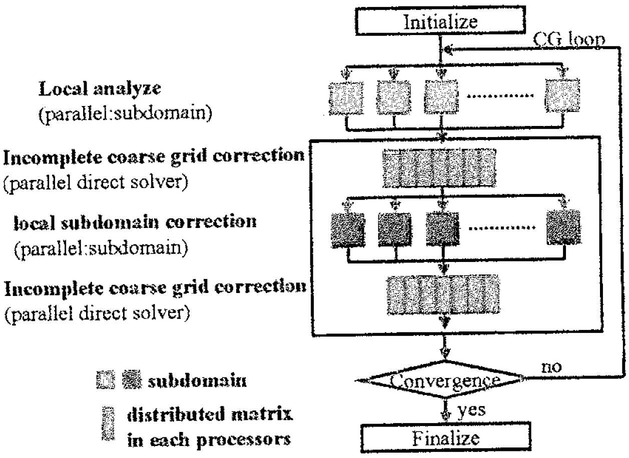

Fig.4(b) Analysis flow of the BDD with imcomplete coarse grid correction incompleteness. In this system, however, the coarse space problem is solved by the incompletely factorized operator without iterations. This incomplete factorization process decreases computation cost for each iteration and improves parallel efficiency, but may increase the number of iterations. Finally computation time is expected to be reduced. In the original BDD preconditioner of Eq. (33), the coarse grid correction is implemented after local subdomain correction in each iteration. However, in the present BDD preconditioner of Eq. (22), the coarse grid correction is applied to the CG residual vector before local subdomain correction. Figure 4(b) shows the analysis flow of the present parallel BDD algorithm.

4. NUMERICAL EXPERIMENTS

4.1 Elastostatic Analysis of ABWR Vessel Model with 35 Million DOFs on SR8000/MPP

This section first describes an elastostatic stress analysis for a precise model of an advanced boiling water reactor (ABWR) vessel with a 35 million DOFs unstructured mesh as shown in Figure 2(a). Size of fine elements placed in nozzle corners and internal pump junctions is about 2mm, while that of average elements is about 10mm. As boundary conditions, the bottom surface of its skirt portion is fixed, and a static gravitational force is applied to the vessel in the horizontal direction, imitating a seismic loading condition. Such a complex shaped and large-scale thin structure with less constraint often results in an ill-conditioned system matrix. Most iterative solution methods fail in attaining convergence when solving such ill-conditioned problems. The ADVENTURE_Solid successfully overcomes this problem owing to the employment of the BDD-based preconditioner. Figure 5 shows convergence histories of force imbalance measured at the interface of subdomains, i.e. residual norm, plotted against the number of iterations. The calculations are performed on Hitachi SR8000/MPP consisting of 1,024PEs whose theoretical peak performance is 1.8TFLOPS. There are two lines in the figure. HDDM denotes the result obtained using the HDDM with a simple diagonal-scaling preconditioner, while BDD denotes that of the HDDM with the BDD preconditioner. As for the HDDM case, the residual norm doesn't

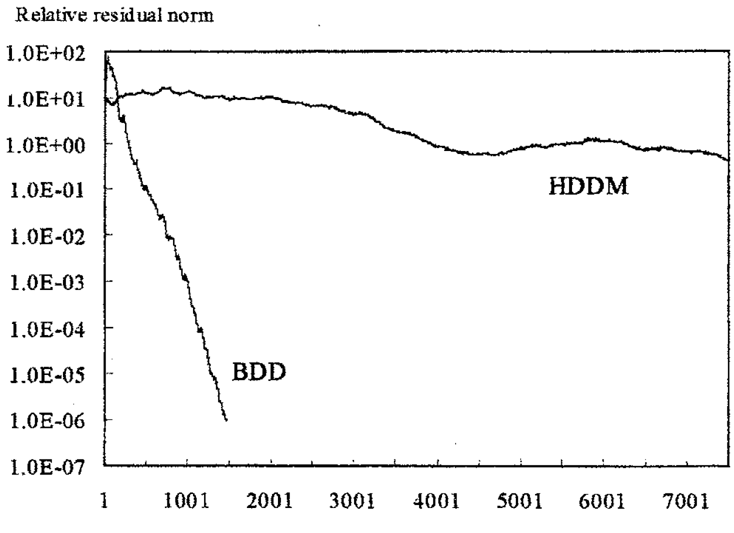

Fig.5 History of residual norm on SR8000/MPP with 1,024 PEs

decrease even below 1.0e-1 though spending 5 hours for over 7,000 iterations. On the other hand, in the BDD case, the calculation is successfully converged with about 1,400 iterations in about 20 minutes. As this model is divided into about 30,000 subdomains, the number of DOFs of the coarse space becomes about 180,000. This coarse space problem is solved in only 30 seconds with the incomplete factorization based parallel direct method.

Figure 6 shows total computational time vs. the number of processors employed. It can be evaluated from the figure that the scalability in total computation time is less than 99.7%, referring the value for the 128 PE case. The scalability in the total computation time is affected by the increased number of CG iterations, which is caused due to the employment of the incomplete factorization. It is anticipated that the present system still has some potential to attain further speed-up by improving the incomplete factorization based parallel direct method. Figure 7 shows calculated stress distribution and enlarged deformation of the ABWR vessel subjected to the quasi-static seismic loading. By solving a whole reactor vessel with a sufficiently large scale finite element mesh, various local stress concentration regions can be clearly indicated, and precisely evaluated.

4.2 Elastostatic Analysis of Pressure Vessel Model with 100 Million DOF on the ES

The ADVENTURE_Solid is implemented on the Earth Simulator [33] consisting of 256 nodes, i.e. 2,048 PEs with 4TB of main memory, whose theoretical peak performance is 16 TFLOPS. The second problem is an elastostatic stress analysis of a simplified pressure vessel model with 100 million DOFs unstructured mesh. Its mesh size is listed in Table 1. As a boundary condition, the bottom surface of the vessel is fixed, and a static gravitational force is applied to the vessel in the horizontal direction, being similar to the previous problem. Although we do not show convergence histories of relative residual, (1) HDDM with BDD and N-N preconditioner (denoted as BDD) and (2) HDDM with BDD and diagonal-scaling preconditioner (denoted as BDD-DIAG) demonstrate excellent performance in convergence. By considering the performance results, it is concluded here that the diagonal scaling is sufficient as local subdomain correction in the BDD method. The analysis model is divided

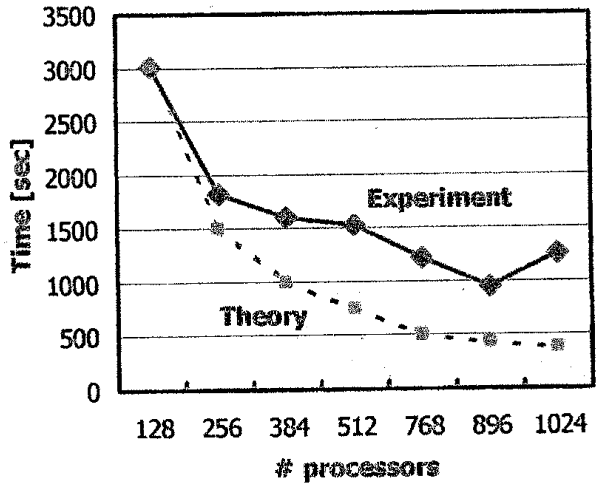

Fig.6 Scalability in total computation time

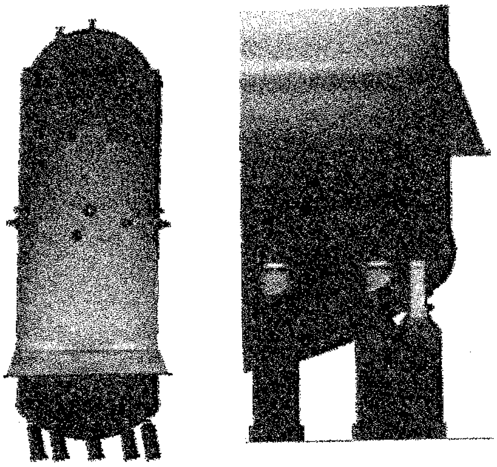

Fig.7 Stress distribution and deformation (x 4,000) of ABWR vessel subjected quasi-staic seismic loading

into 34,816 subdomains and then the number of DOFs of its coarse space is 208,896. The LU factorization of the coarse grid operator is calculated in only 20 seconds. As the result, the present system successfully achieved 5.1TFLOPS, which is 31.8% of the peak performance. The calculation time is only 8.5 minutes. Parallel ratio over 99.9% is achieved, and then parallel efficiency exceeds 80% not only for computation time per iteration but also for total computation time.

Table 1: Mesh size for a simplified vessel model

Number of elements	25,084,456
Number of nodes	34,772,634
Total degrees of freedom	104,195,500

In the DDM algorithm, the number of subdomains employed influences performance of the calculation to some extent. In the case of SR8000/MPP, we have the quasi-optimum number of DOFs per subdomain from our experience, that is 600. Considering that the ES has vector-type processors and larger memory space compared with SR8000/MPP, the larger numbers of subdomains are tested. It is estimated from some preliminary results that the semi-optimum number of subdomains for the ES is about 3,000DOFs/subdomain.

5. CONCLUSIONS

We have been developing an advanced general-purpose finite element analysis system, named ADENTURE, which is designed to be able to analyze a model of arbitrary shape with a 10-100 million DOF mesh. After the overview of the ADVENTURE system is briefly reviewed, the parallel solution algorithm, i.e. the Hierarchical Domain Decomposition Method with Balancing Domain Decomposition based preconditioner is explained. The ADVENTURE_Solid has been successfully implemented on a single PC, PC clusters and massively parallel processors such as Hitachi SR8000/MPP and the Earth Simulator. The present system successfully achieved 5.1TFLOPS, which is 31.8% of the peak performance fo the Earth Simulator consisting of 256 nodes, i.e. 2,048 PEs. The calculation time of elastostatic analysis with 100 million DOF mesh is only 8.5 minutes. Parallel ratio over 99.9% is achieved, and then parallel efficiency exceeds 80% not only for computation time per iteration but also for total computation time. It is clearly concluded from those results that the ADVENTURE system on the Earth Simulator can be used for virtual mockup tests of large-scale and complex artifacts such as nuclear pressure vessels.

Acknowledgments

The authors would like to thank the Earth Simulator Center for giving us the opportunity to use the Earth Simulator. They would also like to thank Mr. T. Fukuda at Tokyo Electric Power Company, Incorp. for providing us an image model of ABWR vessel. This work is performed as a part of the ADVENTURE project [1]. The authors would also like to thank all the members of the ADVENTURE project.

References

[1] http://adventure.q.t.u-tokyo.ac.jp

[2] S. Yoshimura, R. Shioya, H. Noguchi and T. Miyamura, Advanced general-purpose computational mechanics system for large-scale analysis and design, Journal of Computational and Applied Mathematics, 49 (2002) 279-296.

[3] Report on Computational Science and Engieering, JSPS-RFTF Program 2001-2002, (2002).

[4] http://proton.is.s.u-tokyo.ac.jp/cse/index-e.html/

[5] S. Yoshimura, N. Oshio, H. Kawai, H. Azegami and K. Takeuchi, Large-scale parallel shape optimization using hierarchical domain decomposition with traction method, Journal of the Simulation Technology, 20-4 (2001) 296-302 (in Japanese).

[6] B. H. Dennis, I. N. Egorov, H. Sobieczky, G. S. Dulikravich, S.Yoshimura, Thermoelasticity optimization of 3-D serpentine cooling passages in turbine blades, International Journal of Turbo & Jet-Engines, 21-1 (2004) 57-68.

[7] M. Suzuki, H. Akiba, S. Yoshimura and G. Yagawa, Analysis of stress intensity factor of piping using large scale analysis code ADVentureCluster, Trans. 16th SMiRT, G05/5, Washington D.C., CD-ROM, (2001).

[8] M. Suzuki, T. Ohyama, H. Akiba, H. Noguchi and S. Yoshimura, Development of fast and robust parallel coarse-grid based CG solver for large scale finite element analyses, Trans. JSME, 68A-671 (2002) 1010-1017 (in Japanese).

[9] G.Yagawa and R.Shioya, Parallel finite elements on a massively parallel computer with domain decomposition, Computing Systems in Engineering , 4 (1994) 495-503.

[10] R.Shioya and G.Yagawa, Parallel finite elements of ten-million DOFs based on domain decomposition method, WCCM IV Computational Mechanics -New Trends and Applications-IV 11 (1998) 1-12.

[11] R. Shioya and G. Yagawa, Parallel finite element analysis of 100 million DOFs based on the hierarchical domain decomposition method, Trans. JSCES, 3 (2001) 201-206 (in Japanese).

[12] T. Miyamura, H. Noguchi, R. Shioya, S. Yoshimura and G. Yagawa, Elastic-plastic analysis of nuclear structures with millions of DOFs using the hierarchical domain decomposition method, Nuclear Engineering & Design, 212 (2002) 335-355.

[13] Y. H. DeRoeck and P. LeTallec, Analysis and test of a local domain decomposition preconditioner, 4th International Symposium on Domain Decomposition Methods (1991) 112-128.

[14] J. Mandel, Balancing domain decomposition, Communications on Numerical Methods in Engineering , 9 (1993) 233-241.

[15] P. LeTallec and M. Vidrascu, Generalized Neumann-Neumann preconditioners for iterative substructuring, 9th International Symposium on Domain Decomposition Methods, (1996) 413-425.

[16] P. LeTallec, J. Mandel and M. Vidrascu, A Neumann-Neumann domain decomposition algorithm for solving plate and shell problems, SIAM J. Number. Math., 35 (1997) 836-867.

[17] R. Shioya, H. Kanayama, D. Tagami and M. Ogino, 3D large scale structural analysis using a balancing domain decomposition method, Trans. JSCES, 2 (2000) 139-144 (in

Japanese).

[18] R. Shioya, M. Ogino, H. Kanayama and D.Tagami, Large scale finite element analysis with a balancing domain decomposition method, Key Engineering Materials, 243-244 (2003) 21-26.

[19] T. Miyamura and S. Yoshimura, Parallel stress analyses of ancient architecture Pantheon on PC cluster, Trans. Architectural Institute of Japan, 55 (2001) 95-102 (in Japanese).

[20] G. Yagawa, S. Yoshimura and K. Nakao, Automatic mesh generation of complex geometries based on fuzzy knowledge processing and computational geometry, Integrated Computer-Aided Engineering 2 (1995) 265-280.

[21] S. Yoshimura, H. Nitta, G. Yagawa and H. Akiba, Parallel automatic mesh generation of nuclear structures with ten-million nodes, Trans. 15th SMiRT, Seoul, II (1999) 21-28.

[22] G. Karypis and V. Kumar, Multilevel k-way partitioning scheme for irregular graphs, Technical Report TR 95-064, Department of Computer Science, University of Minnesota, (1995).

[23] G. Karypis and V. Kumar, Parallel multilevel k-way partitioning scheme for irregular graphs, Technical Report TR 96-036, Department of Computer Science, University of Minnesota, (1996).

[24] H. Kanayama, R. Shioya, D. Tagami and M. Saito, Numerical analysis of 3D eddy current problems by the hierarchical domain decomposition method, Trans. JSCES, 3 (2001) 151-156 (in Japanese).

[25] S. Shoui, S. Yoshimura, H. Akiba, T. Ohyama and G. Yagawa, Parallel visualization of finite element solutions with ten million DOFs using PC cluster, Proceedings of European Congress on Computational Methods in Science and Engineering (ECCOMAS2000), Balcelona, CD-ROM, (2000)

[26] M. Vidrascu, Remarks on the implementation of the generalized neumann-neumann algorithm, 11th International Conference on Domain Decomposition Methods, (1998) 485-493.

[27] P. R. Amestoy, I. S. Duff, J. -Y. L'Excellent and P. Plechac, PARASOL. An Integrated programming environment for parallel sparse matrix solvers, High-Performance Computing, (1999) 79-90.

[28] F.-X. Roux and C. Farhat, Parallel implementation of the two-level FETI method, 9th International Conference on Domain Decomposition Methods, (1997)

[29] J. Mandel, R. Tezaur and C. Farhat, A scalable substructuring method by lagrange multipliers for plate bending problems, SIAM Journal of Numerical Analysis, 36 (1999) 1370-1391.

[30] M. Lesoinne and K. Pierson, FETI-DP : An efficient, scalable and unified dual-primal FETI method, 12th International Conference on Domain Decomposition Methods, (1999) 421-428.

[31] http://endo.sandia.gov/9234/salinas

[32] M. Bhardwaj, K. Pierson, G. Reese, T. Walsh, D. Day, K. Alvin, J. Peery, C. Farhat and M. Lesoinne, Salinas : A scalable software for high-performance structural and solid mechanics simulations, Technical Papers of SC2002, (2002).

[33] http://www.es.jamstec.go.jp/