Parallel Iterative Solvers of GeoFEM with Selective Blocking Preconditioning for Nonlinear Contact Problems on the Earth Simulator

Kengo Nakajima*

Abstract

An efficient parallel iterative method with *selective blocking* preconditioning has been developed for symmetric multiprocessor (SMP) cluster architectures with vector processors such as the Earth Simulator. This method is based on a three-level hybrid parallel programming model, which includes message passing for inter-SMP node communication, loop directives by OpenMP for intra-SMP node parallelization and vectorization for each processing element (PE). This method provides robust and smooth convergence and excellent vector and parallel performance in 3D geophysical simulations with contact conditions performed on the Earth Simulator. The *selective blocking* preconditioning is much more efficient than ILU(1) and ILU(2). Performance for the complicated Southwest Japan model with more than 23 M DOF on 10 SMP nodes (80 PEs) of the Earth Simulator was 161.7 GFLOPS, corresponding to 25.3% of the peak performance for hybrid programming model, and 190.4 GFLOPS (29.8% of the peak performance) for flat MPI, respectively.

1. Introduction

1.1 Background

In 1997, the Science and Technology Agency of Japan (now, the Ministry of Education, Culture, Sports, Science and Technology, Japan) began a 5-year project to develop a new supercomputer, the Earth Simulator [1]. The goal has been the development of both hardware and software for earth science simulations. The present study was conducted as part of the research towards developing a parallel finite-element platform for solid earth simulation, named GeoFEM [2]. One of the most important applications of GeoFEM is the simulation of ground motion. Stress accumulation on plate boundaries (faults) is very important in estimating the earthquake generation cycle (Fig.1). We need very fine resolution (order of 10 meters) around zones with higher stress accumulation, therefore more than hundred millions of meshes are required for detailed simulations.

In ground motion simulations, material, geometric and boundary nonlinearity should be considered. Boundary nonlinearity due to *fault-zone contact* is the most critical. In GeoFEM, the augmented Lagrange method (ALM) and penalty method are implemented, and a large penalty number λ is introduced for constraint conditions around faults [3]. The nonlinear process is solved iteratively by the Newton-Raphson (NR) method. A large λ can provide an accurate solution and fast nonlinear convergence for the Newton-Raphson method, but the condition number of the coefficient matrices is large. Therefore, many iterations are required for convergence of iterative solvers (Fig.2).

* Senior Technical Staff, Research Organization for Information Science and Technology (RIST) (2-2-54 Naka-Meguro, Meguro-ku, Tokyo 153-0061, Japan, e-mail: nakajima@tokyo.rist.or.jp, phone: +81-3-3712-5321, fax: +81-3-3712-5552).

SC'03, November 15-21, 2003, Phoenix, Arizona, USA,

In previous work [4], the author developed a robust preconditioning method, called *selective blocking*, for the simulation of fault-zone contact with penalty constraints using parallel computers. For symmetric positive definite matrices, block incomplete Cholesky factorization without inter-block fill-in, using *selective blocking* (SB-BIC(0)) shows excellent performance, memory efficiency and robustness for a wide range of penalty parameter values, even if the meshes are distorted. The conjugate gradient (CG) method with SB-BIC(0) preconditioning was implemented to GeoFEM, and the parallel performance for both simple and complicated geometries was evaluated using 16 to 128 processing elements (PEs) of a Hitachi SR2201 at the University of Tokyo.

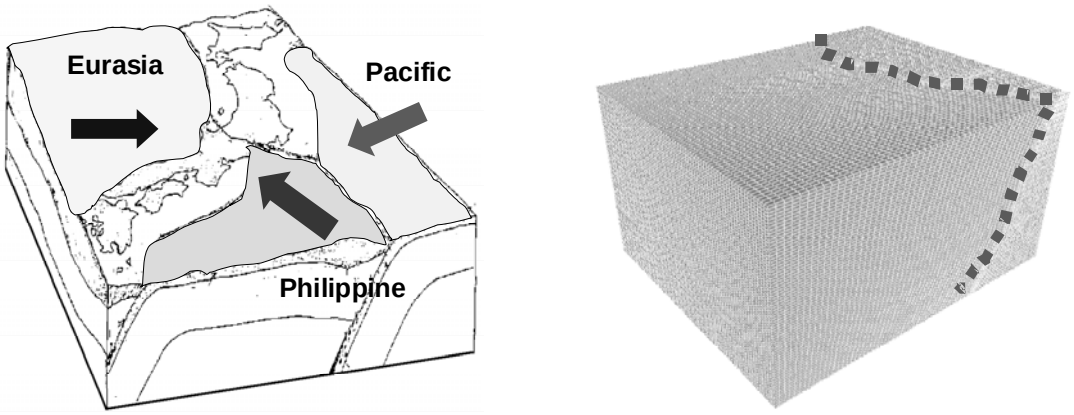


Fig. 1 Plate boundaries (faults) around Japanese Islands and an example of the finite element model

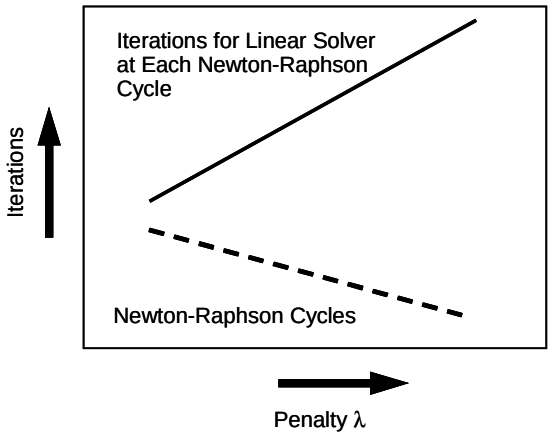


Fig. 2 Typical λ (penalty number)-iterations relationship in fault-zone contact computation without friction by ALM [4]. A large λ can provide an accurate solution and fast nonlinear convergence for the Newton-Raphson method, but the condition number of the coefficient matrices is large.

1.2 Overview of This Study

In the present work, GeoFEM and parallel iterative solvers with *selective blocking* preconditioning are ported to symmetric multiprocessor (SMP) cluster architectures with vector processors such as the Earth Simulator. The Earth Simulator has an SMP cluster architecture and consists of 640 SMP nodes. Each SMP node consists of eight vector processors. Peak performance of each PE is 8 GFLOPS, and the performance of the entire system is 40 TFLOPS. Each SMP node of the Earth Simulator has 16 GB of memory, which corresponds to a total of approximately 10 TB [1]. A hybrid parallel programming model is adopted with reordering methods for vector and parallel performance, and compared with flat MPI parallel programming model [5]. The parallel and vector performance of this method is demonstrated on the Earth Simulator.

In the following part of this paper, we give a brief overview of parallel iterative solvers in GeoFEM, selective blocking preconditioning, special reordering techniques for parallel and vector computation on the Earth Simulator and present the results of 3D applications in solid mechanics on the Earth Simulator.

2 Parallel Iterative Solvers in GeoFEM

2.1 Distributed Data Structures

GeoFEM adopts domain decomposition [2,6,7] for parallel computing where the entire model is divided into domains, and each domain is assigned to a PE. A proper definition of the layout of the distributed data structures is an important factor determining the efficiency of parallel computations with unstructured meshes. The local data structures in GeoFEM are node-based with overlapping elements, and as such are appropriate for the preconditioned iterative solvers used in GeoFEM [6].

Although MPI provides subroutines for communication among processors during computation for structured grids, it is necessary for users to design both the local data structure and communications for unstructured grids. In GeoFEM, the entire region is partitioned in a *node-based* manner and each domain contains the following local data:

- Nodes originally assigned to the domain
- Elements that include the assigned nodes
- All nodes that form elements but are from external domains
- A communication table for sending and receiving data
- Boundary conditions and material properties

Nodes are classified into the following three categories from the viewpoint of message passing:

- Internal nodes (originally assigned to the domain)
- External nodes (forming the element in the domain but are from external domains)
- Boundary nodes (*external nodes* of other domains)

Communication tables between neighboring domains are also included in the local data. Values on *boundary* nodes in the domains are *sent* to the neighboring domains and are *received* as *external* nodes at the *destination* domain. This data structure, described in Fig.3, and the communication procedure described in Fig.4 provide excellent parallel efficiency [2,5,6]. This type of communication occurs in the procedure for computing the matrix-vector product of Krylov iterative solvers described in the next subsection. The partitioning program in GeoFEM works on a single PE, and divides the initial entire mesh into distributed local data.

In GeoFEM, coefficient matrices for linear solvers are assembled in each domain according to FEM procedures. This process can be performed without communication among processors using the information of overlapping elements.

2.2 Localized Preconditioning

The incomplete lower-upper (ILU) and incomplete Cholesky (IC) factorization methods are the most popular preconditioning techniques for accelerating the convergence of Krylov iterative methods.

Of the range of ILU preconditioning methods, ILU(0), which does not allow fill-in beyond the original non-zero pattern, is the most commonly used. Backward/forward substitution (BFS) is repeated at each iteration. BFS requires global data dependency, and this type of operation is not suitable for parallel processing in which locality is of utmost importance. Most preconditioned iterative processes are a combination of (1)matrix-vector products, (2)inner dot products, (3)DAXPY (linear combination of vectors) operations [8] and vector scaling and (4)preconditioning operations

The first three operations can be parallelized relatively easily [8]. In general, preconditioning operations such as BFS represent almost 50 % of the total computation if ILU(0) is implemented as the preconditioning method. Therefore, a high degree of parallelization is essential for the BFS operation.

The localized ILU(0) used in GeoFEM is a *pseudo* ILU(0) preconditioning method that is suitable for parallel processors. This method is not a *global* method, rather, it is a *local* method on each processor or domain. The ILU(0) operation is performed locally for a coefficient matrix assembled on each processor by zeroing out components located outside the processor domain. This is equivalent to solving the problem within each processor with

zero Dirichlet boundary conditions during the preconditioning. This *localized* ILU(0) provides data locality on each processor and good parallelization because no inter-processor communications occur during ILU(0) operation. This idea is originally from the incomplete block Jacobi preconditioning method [7,8].

However, localized ILU(0) is not as powerful as the global preconditioning method. Generally, the convergence rate degrades as the number of processors and domains increases [2,5]. At the critical end, if the number of processors is equal to the number of degrees of freedom (DOF), this method performs identically to diagonal scaling.

Table 1 shows the results of a homogeneous solid mechanics example with 3×443 DOF solved by the conjugate gradient (CG) method with localized IC(0) preconditioning. Computations were performed on the Hitachi SR2201 at the University of Tokyo. Although the number of iterations for convergence increases according to the domain number, this increase is just 30% from one to 32 PEs.

Figure 5 shows the work ratio (real computation time/elapsed execution time including communication) for various problem sizes [5] of simple 3D elastic problems with homogeneous boundary conditions. In these computations, the problem size for 1 PE was fixed. The largest case was 196,608,000 DOF on 1024 PEs. Figure 5 shows that the work ratio is higher than 95% if the problem size for 1 PE is sufficiently large. In this case, code was vectorized and a performance of 68.7 GFLOPS was achieved using 1024 PEs. Peak performance of the system was 300 GFLOPS with 1024 PEs; 68.7 GFLOPS corresponds to 22.9% of the peak performance [5]. This good parallel performance is attributed largely to the reduced overhead provided by the use of communication tables as part of the GeoFEM's local data structure.

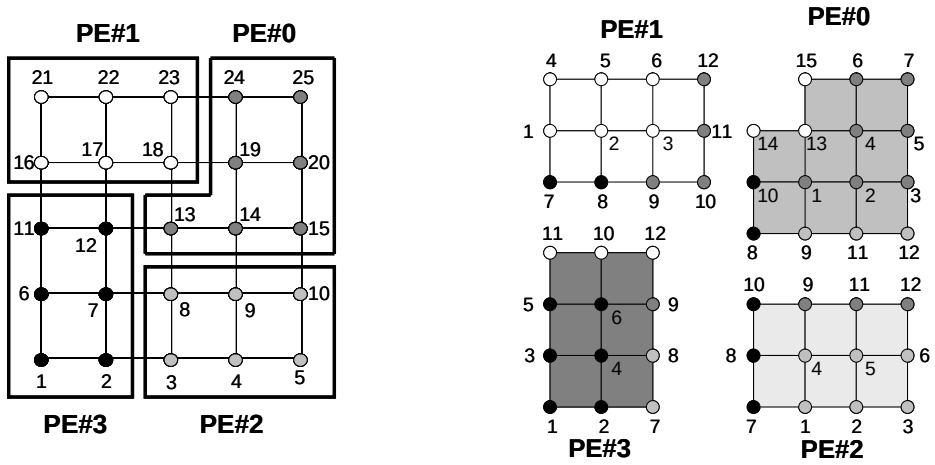


Fig.3 Node-based partitioning into four PEs [2,6].

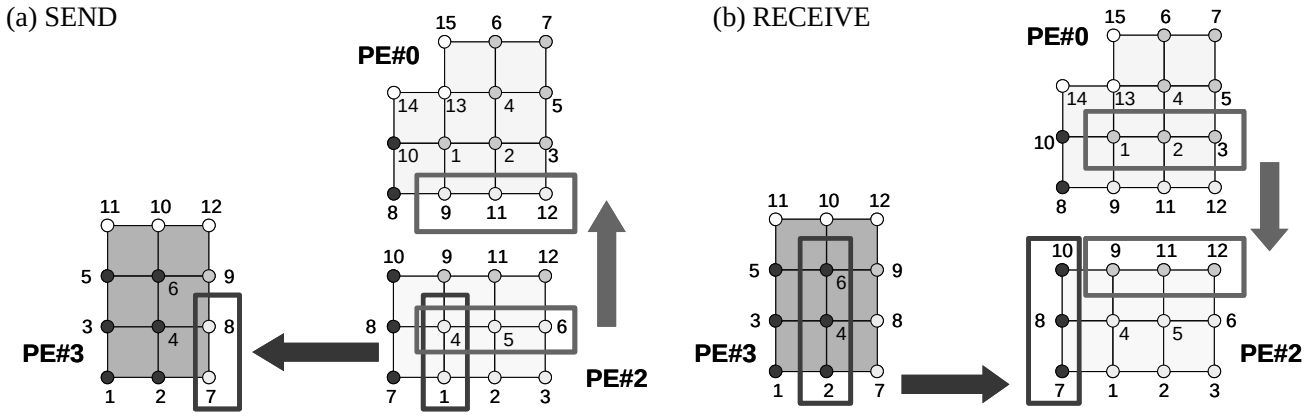


Fig.4 Communication among processors

Table 1 Homogeneous solid mechanics example with 3×44^3 DOF on Hitachi SR2201 solved by CG method with localized IC(0) preconditioning (Convergence Criteria $\varepsilon=10^{-8}$).

PE #	Iter. #	sec.	Speed Up
1	204	233.7	-
2	253	143.6	1.63
4	259	74.3	3.15
8	264	36.8	6.36
16	262	17.4	13.52
32	268	9.6	24.24
64	274	6.6	35.68

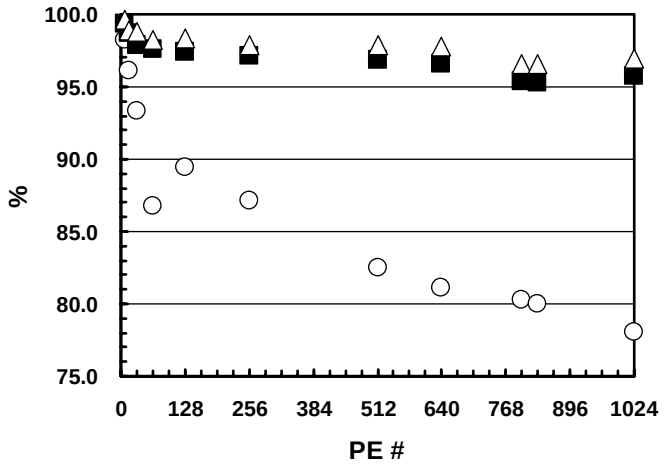


Fig.5 Parallel performance for various problem sizes for simple 3D elastic solid mechanics on Hitachi SR2201. Problem size/PE is fixed. Largest case is 196,608,000 DOF on 1024 PEs. (Circles: 3×16^3 (= 12,288) DOF/PE, Squares: 3×32^3 (= 98,304), Triangles: 3×40^3 (= 192,000)).

3. Selective Blocking

3.1 Robust Preconditioning Method for Ill-Conditioned Problems

The incomplete Cholesky (IC) and incomplete lower-upper (ILU) factorization methods are the most popular preconditioning techniques for accelerating the convergence of Krylov iterative methods. The typical remedies using an IC/ILU type of preconditioning method for ill-conditioned matrices, which appear in nonlinear simulations using penalty constraints, are as follows:

- Blocking
- Deep Fill-in
- Reordering.

In addition to these methods, a special method called *selective blocking* was also developed for contact problems in [4]. In the *selective blocking* method, strongly coupled finite-element nodes in the same contact group [3] coupled through penalty constraints are placed into the same large block (*selective block* or *super* node) and all of the nodes involved are reordered according to this blocking information. Full LU factorization is applied to each selective block. The size of each block is $(3\times NB)\times(3\times NB)$ in 3D problems, where NB is the number of finite-element

nodes in the selective block, which is shown in Fig.6. Thus, local equations for coupled finite-element nodes in contact groups are solved by means of a *direct* method during preconditioning.

Table 2 shows the convergence of CG solver with various types of preconditioning methods. The linear equations are derived from actual nonlinear contact problems in [3] and [4]. By introducing the 3×3 block, the CG solver preconditioned by block IC with no fill-in (i.e., BIC(0)), converges even when λ is as large as 10^6. *Deep fill-in* options provide faster convergence, but the SB-BIC(0) (i.e., BIC(0) preconditioning with *selective blocking* reordering) shows the best performance. SB-BIC(0) usually requires a greater number of iterations for convergence compared to BIC(1) and BIC(2), but the overall performance is better because the computation time for each iteration and set-up is much shorter. As is also shown in Table 2, because no *inter-block* fill-in is considered for SB-BIC(0), the memory requirement for this method is usually as small as that in BIC(0) with no fill-in. Only the *inter-node* fill-in in each *selective block* is considered in SB-BIC(0).

The CG solver with SB-BIC(0) preconditioning can be considered to be a hybrid of iterative and direct methods. Local equations for coupled finite-element nodes in contact groups are solved by means of a direct method during preconditioning. This method combines the efficiency and scalability of iterative methods with the robustness of direct methods.

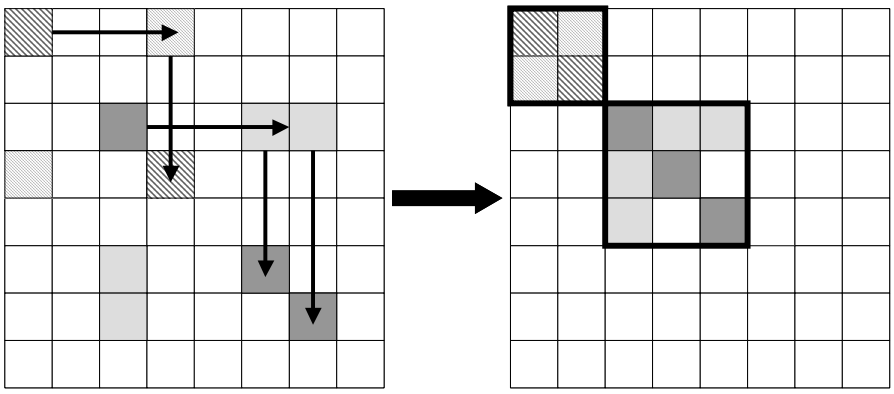
This idea of selective blocking is also related to the *clustered element-by-element method* (CEBE) described in [9]. In CEBE, elements are partitioned into clusters of elements, with the desired number of elements in each cluster, and the iterations are performed in a cluster-by-cluster fashion. This method is highly suitable for both vectorization and parallelization, if it is used with proper clustering and element grouping schemes. Any number of elements can be brought together to form a cluster, and the number should be viewed as an optimization parameter to minimize computational cost. The CEBE method becomes equivalent to the direct method when the cluster size is equal to the total number of elements. Generally, larger clusters provide better convergence rates because a larger number of fill-in elements are taken into account during factorization, but the cost per iteration cycle increases according to the size of the cluster, as shown in Fig.7. The trade-off between convergence and computational cost is not clear, but the results of examples in [9] show that larger clusters provide better performance.

In *selective blocking*, clusters are formed according to information about the contact groups. Usually, the size of each cluster is much smaller than that in a general CEBE method. If a finite element node does not belong to any contact groups, it forms a cluster whose size is equal to one in the selective blocking.

In Appendix A., the robustness of the preconditioning method was estimated according to the eigenvalue distribution of the $[M]^{-1}[A]$ matrix by the method in [4], where [A] is the original coefficient matrix and $[M]^{-1}$ is the inverse of the preconditioning matrix. According to the results, all of the eigenvalues are approximately constant and close to 1.00 for a wide range of λ values except for BIC(0). BIC(1) and BIC(2) provide a slightly better spectral feature than SB-BIC(0).

Table 2 Iterations/computation time for convergence ($\varepsilon=10^{-8}$) on a single PE of Intel Xeon 2.8 GHz by preconditioned CG for the 3D elastic fault-zone contact problem in [3] and [4] (83,664 DOF).: **BIC(n)**: Block IC with n-level fill-in, **SB-BIC(0)**: BIC(0) with the selective blocking reordering.

Preconditioning	λ	Iterations	Set-up (sec.)	Solve (sec.)	Set-up+Solve (sec.)	Single Iteration (sec.)	Memory Size (MB)
Diagonal Scaling	10^2	1531	<0.01	75.1	75.1	0.049	119
	10^6	No Conv.	-	-	-	-	
IC(0) (Scalar Type)	10^2	401	0.02	39.2	39.2	0.098	119
	10^6	No Conv.	-	-	-	-	
BIC(0)	10^2	388	0.02	37.4	37.4	0.097	59
	10^6	2590	0.01	252.3	252.3	0.097	
BIC(1)	10^2	77	8.5	11.7	20.2	0.152	176
	10^6	78	8.5	11.8	20.3	0.152	
BIC(2)	10^2	59	16.9	13.9	30.8	0.236	319
	10^6	59	16.9	13.9	30.8	0.236	
SB-BIC(0)	10^0	114	0.10	12.9	13.0	0.113	67
	10^6	114	0.10	12.9	13.0	0.113	



(a) (b)

Fig.6 Procedure of the *selective blocking* : Strongly coupled elements are put into the same *selective block*. (a) searching for strongly coupled components and (b) reordering and selective blocking.

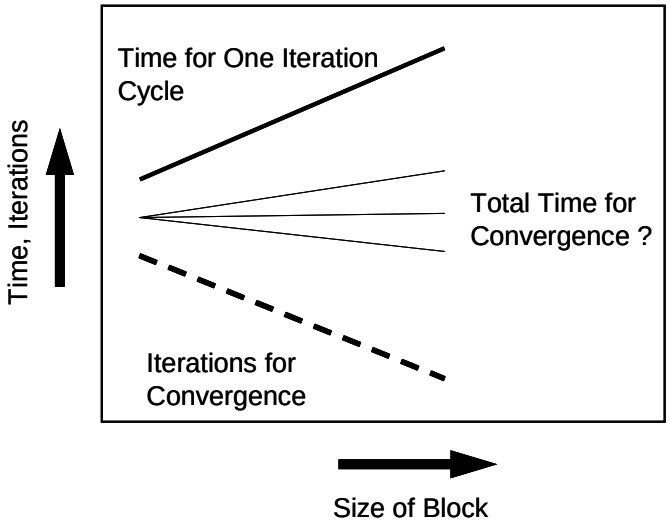


Fig.7 Trade-off between convergence and computational cost per on iteration cycle according to block size in CEBE type method. Based on [9].

3.2 Strategy for Parallel Computations

Localized ILU/IC [2,5,6] is an efficient parallel preconditioning method, but it is not robust for ill-conditioned problems. Table 3 (left side) shows the results by parallel CG solvers with localized preconditioning on a 8 PEs of Intel Xeon 2.8 GHz cluster using distributed matrices, for the problem described in Fig.1. According to the results, the number of iterations for convergence increases by a factor of 10 in $\lambda=10^6$ cases. This is because the *edge-cuts* occur at inter-domain boundary edges that are included in contact groups [2,5,6].

In order to eliminate these edge-cuts, a partitioning technique has been developed so that all nodes which belong to the same contact group are in the same domain. Moreover, nodes are re-distributed so that load-balancing among domains should be attained for efficient parallel computing (Fig.8).

In GeoFEM, there are several types of special elements for contact problems (types 411, 412, 421, 422, 511, 512, 521 and 522) [2]. Nodes included in the same elements of these types are connected through penalty constraints and form a contact group. In the new partitioning method, the partitioning process is executed so that these nodes in the same contact elements are on the same domain, or PE. These functions are added to the original domain partitioner in GeoFEM described in 2.

Table 3 (right side) shows the results obtained by this partitioning method. The number of iterations for convergence has been dramatically reduced for each preconditioning method although it is larger than that of the single PE cases shown in Table.2 due to localization.

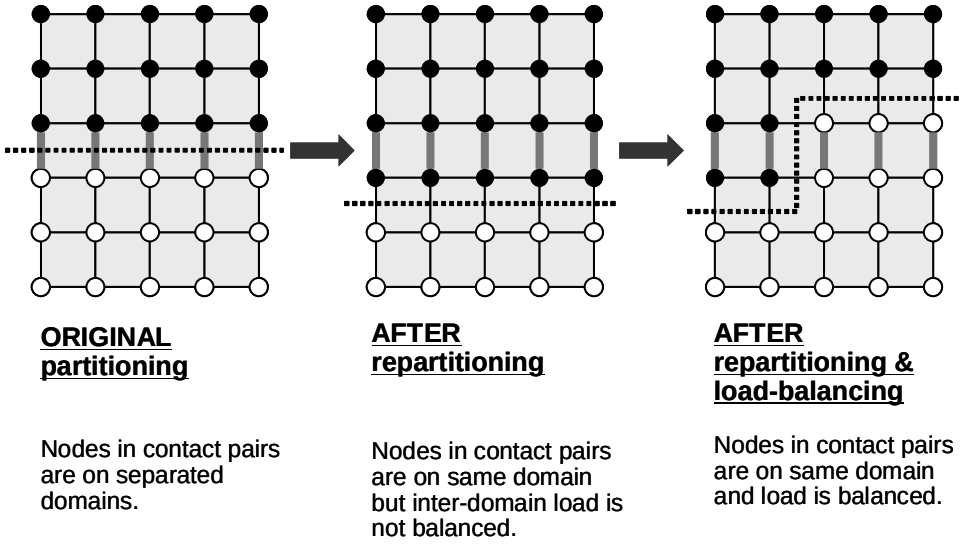


Fig.8 Partitioning strategy for the nodes in contact groups

Table 3 Iterations/computation time for convergence ($\varepsilon=10^{-8}$) on 8 PEs of Intel Xeon 2.8 GHz cluster by preconditioned CG for the 3D elastic fault-zone contact problem in [3] and [4] (83,664 DOF).: **BIC(n)**: Block IC with n-level fill-in, **SB-BIC(0)**: BIC(0) with the selective blocking reordering. Effect of repartitioning method in Fig.8 is evaluated.

		ORIGINAL Partitioning		**IMPROVED** Partitioning	
Preconditioning	λ	Iterations	Set-up+Solve (sec.)	Iterations	Set-up+Solve (sec.)
BIC(0)	10^2	703	7.5	489	5.3
	10^6	4825	50.6	3477	37.5
BIC(1)	10^2	613	11.3	123	2.7
	10^6	2701	47.7	123	2.7
BIC(2)	10^2	610	19.5	112	4.7
	10^6	2448	73.9	112	4.7
SB-BIC(0)	10^0	655	10.9	165	2.9
	10^6	3498	58.2	166	2.9

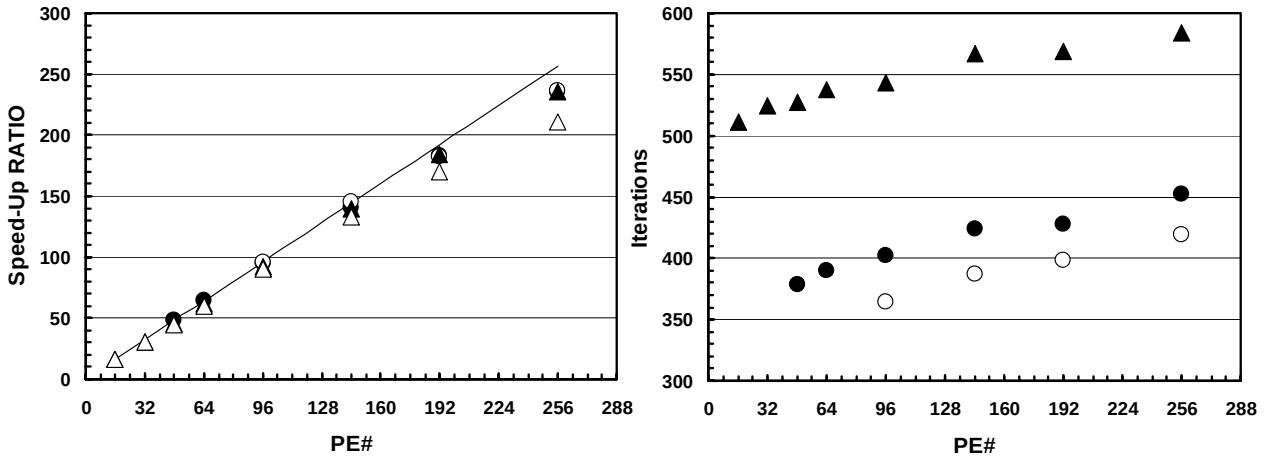
3.3 Large-Scale Computation

A large-scale computation was performed on the simple block model with 784,000 elements and 823,813 nodes (Total DOF= 2,471,439), which is shown later in Fig.23.

Linear elastic problem on the geometry was solved by parallel iterative solvers using various types of preconditioning methods with the MPC (multiple point constraint) conditions. Domains are partitioned according to the contact group information described in the previous chapter. Computations were performed using 16 to 256 PEs on a Hitachi SR2201 at the University of Tokyo.

Table 4 shows the results for various preconditioners. BIC(1), BIC(2) and SB-BIC(0) provide robust convergence but convergence of BIC(0) is very slow. SB-BIC(0) provides the most efficient performance, although the iteration number for convergence is larger than BIC(1) and BIC(2). Figure 9 and Table 4 show the parallel performance for the same problem solved using 16 to 256 PEs of Hitachi SR2201. BIC(1) and BIC(2) did not work if the PE number was small due to memory limitation. As shown in Table 4 and Fig.9, the iteration number for convergence increases according to PE number due to the locality of the preconditioning method, but this increase is very slight (only 14% increase from 16 PEs to 256 PEs for SB-BIC(0)). The speed-up ratio based on elapsed execution time including communication for 256 PEs, is 235 for SB-BIC(0), as extrapolated from the results obtained using 16 PEs.

The required memory size for each preconditioning method is compared in Table 4 for this problem. The memory size of each PE on the Hitachi SR2201 is 256 MB but only 224 MB of the memory is available for users. For example, BIC(2) does not function on 64 PEs because the required memory size is 14.4 GB but only 14.3 GB (224×64/1000=14.34) are available on 64 PEs. The required memory size for SB-BIC(0) is competitive with that of BIC(0), is less than 50% of that of BIC(1), and approximately 25% of BIC(2). The required memory size for SB-BIC(0) could change, according to the number of contact groups and the size of the matrix blocks by selective blocking, but the required memory size is much less than that of BIC(1) or BIC(2) because block-to-block fill-in is not considered in SB-BIC(0).



(a) Speed-up ratio (b) Iteration number for convergence

Fig.9 Parallel performance based on elapsed execution time including communication and iterations for convergence ($\varepsilon=10^{-8}$) on a Hitachi SR2201 with 16 to 256 PEs using preconditioned CG for the 3D elastinc contact problem with MPC condition ($\lambda=10^{6}$) in Fig.23 (2,471,439 DOF). (BLACK Circles: BIC(1), WHITE Circles: BIC(2), BLACK Triangles: SB-BIC(0), WHITE Triangles: BIC(0)).

Table 4 Iterations/elapsed execution time (including factorization, communication overhead) for convergence ($\varepsilon=10^{-8}$) on a Hitachi SR2201 with 256 PEs using preconditioned CG for the 3D elastic contact problem for simple block model with MPC condition in Fig.23 (2,471,439 DOF). Domains are partitioned according to the contact group information.: **BIC(n)**: Block IC with n-level fill-in, **SB-BIC(0)**: BIC(0) with the selective blocking reordering.

Precon-ditioning		16 PEs	32 PEs	48 PEs	64 PEs	96 PEs	144 PEs	192 PEs	256 PEs	Memory Size (GB)
BIC(0)	Iterations	14459	14583	15018	15321	15523	15820	16084	16267	
	sec.	13500	7170	4810	3630	2410	1630	1270	1230	3.10
	Speed-up	16	30	45	60	90	133	170	211	
BIC(1)	Iterations			379	390	402	424	428	452	
	sec.	N/A	N/A	236	175	119	81	62	48	8.39
	Speed-up			48	65	95	140	183	236	
BIC(2)	Iterations					364	387	398	419	
	sec.	N/A	N/A	N/A	N/A	212	140	112	86	14.4
	Speed-up					96	145	182	217	
SB-BIC(0)	Iterations	511	524	527	538	543	567	569	584	
	sec.	555	295	193	144	96	64	48	38	3.52
	Speed-up	16	30	46	62	92	139	185	235	

4. Reordering Methods for Parallel/Vector Performance on SMP Nodes

4.1 SMP Cluster Architecture and Hybrid Parallel Programming Model

Recent technological advances have allowed increasing number of processors to have access to a single memory space in a cost-effective manner. As a result, symmetric multiprocessor (SMP) cluster architectures have become very popular as teraflop-scale parallel computers, such as the Accelerated Strategic Computing Initiative (ASCI, currently "Advanced Simulation and Computing (ASC)") [10] machines and the Earth Simulator [1].

In order to achieve minimal parallelization overhead, a multi-level hybrid programming model [5,11,12,13,14] is often employed for SMP cluster architectures. The aim of this method is to combine coarse-grain and fine-grain parallelism. Coarse-grain parallelism is achieved through domain decomposition by message passing among SMP nodes using a scheme such as Message Passing Interface (MPI) [15], and fine-grain parallelism is obtained by loop-level parallelism inside each SMP node by compiler-based thread parallelization such as OpenMP [16].

Another often used programming model is the single-level flat MPI model [5,11,12,13,14], in which separate single-threaded MPI processes are executed on each processing element (PE). The advantage of a hybrid programming model over flat MPI is that there is no message-passing overhead in each SMP node. This is achieved by allowing each thread to access data provided by other threads directly by accessing the shared memory instead of using message passing. However, a hybrid approach usually requires more complex programming.

Although a significant amount of research on this issue has been conducted in recent years [5,11,12,13,14], it remains unclear whether the performance gains of this hybrid approach compensate for the increased programming complexity. Many examples show that flat MPI is rather better [5,11,12,13,14], although the efficiency depends on hardware performance (CPU speed, communication bandwidth, memory bandwidth), features of applications, and problem size [17].

In this study, selective blocking preconditioning is ported to iterative solvers using a three-level hybrid parallel programming model on the Earth Simulator. Individual PE of the Earth Simulator is a vector processor, therefore

third-level of parallelism for vector processing should be considered in addition to the two levels, OpenMP and MPI. Following three levels of parallelism are considered:

- Inter-SMP node MPI for communication
- Intra-SMP node OpenMP for parallelization
- Individual PE compiler directives for vectorization

In flat MPI approach, communication among PEs through MPI and vectorization for individual PE have been considered for the Earth Simulator. In the hybrid parallel programming model, the entire domain is partitioned into distributed local data sets [2,5,6], and each partition is assigned to one SMP node (Fig.10). On the contrast, each partition corresponds to each PE in the flat MPI.

In order to achieve efficient parallel/vector computation for applications with unstructured grids, the following three issues are critical:

- Local operations and no global dependency
- Continuous memory access
- Sufficiently long loops

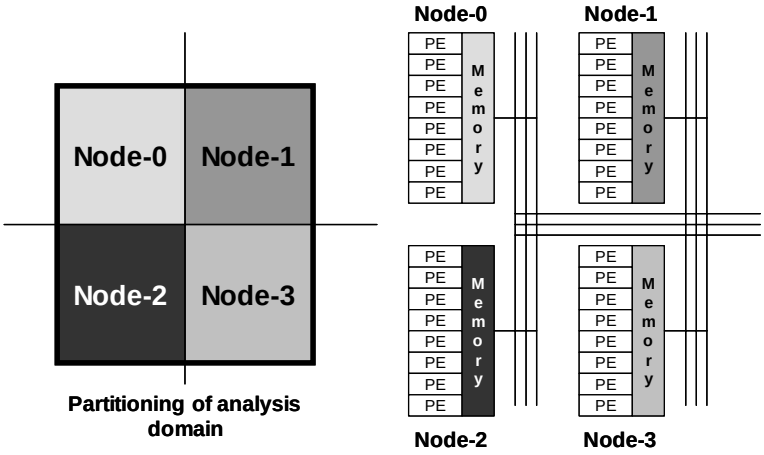


Fig.10 Domain partitioning for parallel FEM computation on SMP cluster architecture by hybrid parallel programming model. Each partition corresponds to an SMP node

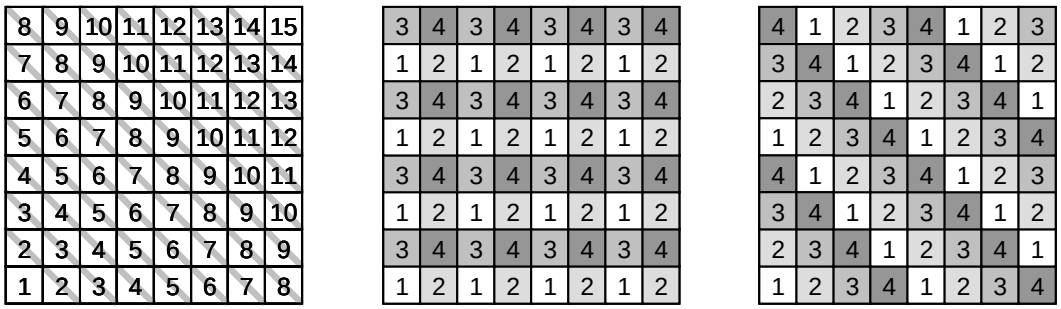
For unstructured grids, in which data and memory access patterns are very irregular, the reordering technique is very effective in achieving highly parallel and vector performance. In this study, a special reordering technique proposed by Washio et. al. [18,19] has been integrated with selective blocking preconditioning and parallel iterative solvers with localized preconditioning developed in the GeoFEM project [2,6] in order to attain local operation, no global dependency, continuous memory access and sufficiently innermost long loops.

4.2 Multicolor Reordering

The popular reordering methods are reverse Cuthil-McKee reordering and multicolor reordering [20]. The reverse Cuthil-McKee (RCM) method (Fig.11(a)) is a typical *level-set* ordering method. In Cuthill-McKee reordering, the elements of a level set are traversed from the nodes of the lowest degree to those of the highest degree according to dependency relationships, where the *degree* refers to the number of nodes connected to each node. In RCM, permutation arrays obtained in Cuthill-McKee reordering are reversed. RCM results in much less fill-in for Gaussian elimination and is suitable for iterative methods with IC or ILU preconditioning. Multicolor reordering (MC) is much simpler than RCM. MC (Fig.11(b)) is based on an idea where no two adjacent nodes have the same color.

In both methods, elements located on the same color (or level-set) are independent. Therefore, parallel operation is possible for the elements in the same color (or level-set) and the number of elements in each color (or level-set) should be as large as possible in order to obtain high granularity for parallel computation or sufficiently large loop length for vectorization.

According to [21], CM-RCM reordering (Fig.11(c)), which is a method combining cyclic multicolor and RCM (Reverse Cuthil-Mckee) reordering, provides fast and robust convergence for simple geometries, however, for complicated geometries in real-world applications, the number of level-sets may be extremely large, and constructing independent sets having a sufficiently large loop length is usually very difficult. Under these circumstances, classical multicolor reordering (MC) offers another option. Although MC usually provides slower convergence than CM-RCM and RCM, a sufficiently large loop length is guaranteed when a certain number of colors is specified. In the present work, the MC reordering method was adapted in order to achieve higher vector performance.



(a) RCM (b) MC: 4 colors (c) CM-RCM: 4 colors

Fig.11 Example of hyperplane/RCM, multicoloring and CM-RCM reordering for 2D geometry [5]

4.3 DJDS Reordering

The compressed row storage (CRS) [8] format originally used in GeoFEM is highly memory-efficient, however the innermost loop is relatively short due to matrix-vector operations, as shown in below:

```
do i= 1, N
  do j= 1, NU(i)
    k1= indexID(i,j);k2= itemID(k1)
    F(i)= F(i) + A(k1)*X(k2)
  enddo
enddo
```

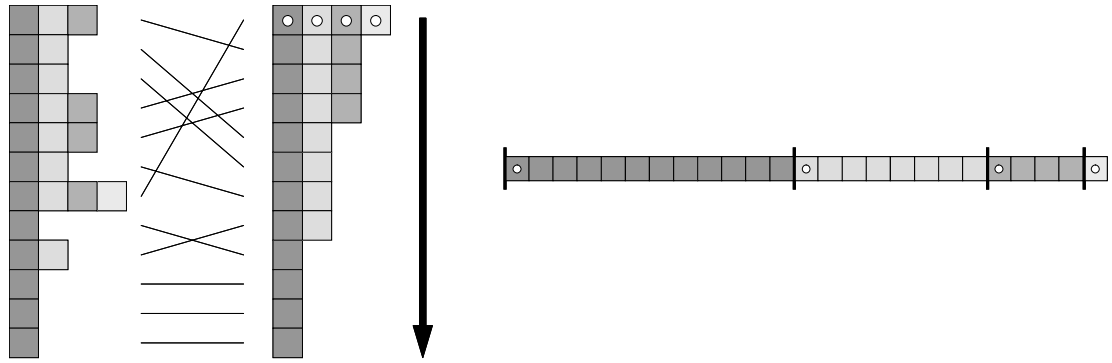
The following loop exchange is then effective for obtaining a sufficiently long innermost loop for vector operations:

```
do j= 1, NUmax
  do i= 1, N
    k1= indexID(i,j);k2= itemID(k1)
    F(i)= F(i) + A(k1)*X(k2)
  enddo
enddo
```

Descending-order jagged diagonal storage (DJDS) [5,18,19] is suitable for this type of operation and involves permuting rows into an order of decreasing number of non-zeros, as in Fig.12(a). As elements on the same color (or level-set) are independent, performing this permutation inside the color (or level-set) does not affect results. Thus, a 1D array of matrix coefficients with continuous memory access can be obtained, as shown in Fig.12(b).

4.4 Distribution over SMP Nodes : Parallel DJDS Reordering

The 1D array of matrix coefficients with continuous memory access and sufficiently long innermost loops is suitable for both parallel and vector computing. The loops for this type of array are easily distributed to each PE in an SMP node via loop directives. In order to balance the computational load across PEs in the SMP node, the DJDS array should be reordered again in a cyclic manner. The procedure for this reordering is called parallel DJDS (PDJDS) [5].



(a) Permutation of rows into order of decreasing number of non-zeros

(b) 1D array of matrix coefficient

Fig.12 DJDS reordering for efficient vector/parallel processing

4.5 Summary of Reordering Methods

The reordering procedures for increasing parallel/vector performance of the SMP cluster architecture described in this section are summarized as follows:

(1) MC reordering on the original local matrix for independent sets.
(2) DJDS reordering for efficient vector processing, producing 1D arrays of coefficients with continuous memory access.
(3) Cyclic reordering for load-balancing among PEs on an SMP node.
(4) PDJDS/MC reordering is complete.

Figure 13 shows the procedure for forward/backward substitution procedure using OpenMP and vectorization directives during ILU(0)/IC(0) preconditioning by PDJDS/MC reordering.

```
do iv= 1, NCOLORS
!$omp parallel do private (iv0,j,iS,iE,i,k,kk etc.)
do ip= 1, PEsmpTOT
  iv0= STACKmc(PEsmpTOT*(iv-1)+ip- 1)
  do j= 1, NLhyp(iv)
    iS= INL(npLX1*(iv-1)+PEsmpTOT*(j-1)+ip-1)
    iE= INL(npLX1*(iv-1)+PEsmpTOT*(j-1)+ip  )
!CDIR NODEP
    do i= iv0+1, iv0+iE-iS
      k= i+iS - iv0
      kk= IAL(k)
      (Important Computations)
    enddo
  enddo
enddo
enddo
```

SMP parallel

Vectorized

Fig.13 Forward/backward substitution procedure using OpenMP and vectorization directives during ILU(0)/IC(0) preconditioning by PDJDS/MC reordering.

4.6 Parallel/Vector Performance for Simple Geometries

(1) Preliminary Results

The proposed reordering method is applied to simple applications in 3D solid mechanics, as described in Fig.14, which represent linear elastic problems with homogeneous isotropic material properties and boundary conditions. In the following part of this section, parallel and vector performance evaluation of the proposed reordering methods in [5] is described. Both of flat MPI and hybrid parallel programming models are tested.

Governing equation in terms of displacement vector **x** (with three degrees of freedom at each node) is given as follows:

$$\mu\nabla^2\mathbf{x} + (\lambda + \mu)\nabla\nabla \cdot \mathbf{x} = 0,\ \mu = \frac{E}{2(1+\nu)},\ \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \tag{1}$$

where E: Young's modulus, ν: Poisson's ratio

For this problem, 3×3 Block ICCG(0) with PDJDS/CM-RCM reordering is applied with full LU factorization for each 3×3 diagonal block. In this section, PDJDS/CM-RCM (not PDJDS/MC) has been applied, because only simple geometries are considered. In each case, the number of colors for CM reordering was set to 99, corresponding to an average innermost loop length of (total number of FEM nodes)/(number of PEs or SMP nodes × 99 × NPE), where NPE is the number of PEs on each SMP node (=8 on the Earth Simulator).

As for the preliminary test, the increase in speed for a fixed problem size (3×128^3=6,291,456 DOF) using between 1 and 8 SMP nodes was evaluated. The speed-up rate for 8 SMP nodes was 6.36(Flat MPI) and 5.78 (Hybrid), which correspond to 79.5 % and 72.2% of the linear (ideal) speed-up, respectively. The performance for 1 node (8 PEs) was 23.4 GFLOPS (Flat MPI, 36.6% of the peak performance=64 GFLOPS) and 21.9 GFLOPS (Hybrid, 34.2%).

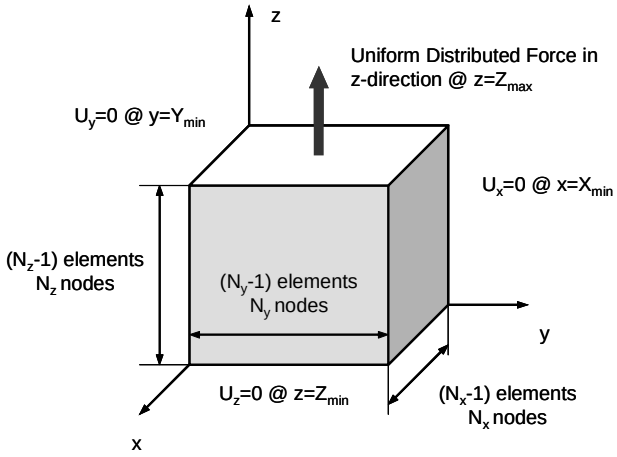


Fig.14 Problem definition and boundary conditions for 3D solid mechanics example cases.

(2) Effect of Reordering

Figure 15 shows the results demonstrating the performance on a single SMP node of the Earth Simulator by hybrid parallel programming model. In this case, the following three cases were compared (Fig.15):

- PDJDS/CM-RCM reordering
- Parallel descending-order compressed row storage (PDCRS)/CM-RCM reordering
- CRS without reordering

PDCRS/CM-RCM reordering is identical to PDJDS/CM-RCM except that the matrices are stored in a CRS manner [8] after permutation of rows into the order of decreasing number of non-zeros. The length of the innermost loop is shorter than that for PDJDS. The elapsed execution time was measured for various problem sizes from 3×16^3 (12,288) DOF to 3×128^3 (6,291,456) DOF on a single SMP node of the Earth Simulator (8 PEs, 64 GFLOPS peak performance, 16 GB memory). The difference between PDCRS and PDJDS for smaller problems is not significant, but PDJDS outperforms PDCRS for larger problems due to larger length of inner-most loops. On the Earth Simulator, the PDCRS performs at a steady 1.5 GLOPS (2.3% of peak performance), while the performance of PDJDS increases from 3.81 GFLOPS to 22.7 GFLOPS (from 6.0% to 35.5% of the peak performance) with problem size. The loop length provided by PDCRS is order of number of off-diagonal components for each

node, which is less than 30 in this case. On the contrast, average loop length provided by PDJDS is more than 2,500 for the case with 3×128^3 (6,291,456) DOF.

The cases without reordering exhibit very poor performance of only 0.30 GFLOPS (0.47% of peak performance). Without reordering, either of parallel and vector computations on a SMP node are impossible for the IC(0) factorization process even in the simple geometry examined in this study. This factorization process represents about 50% of the total computation time in CG solvers with IC(0) preconditioning, as was mentioned before. If this process is not parallelized, the performance decreases significantly.

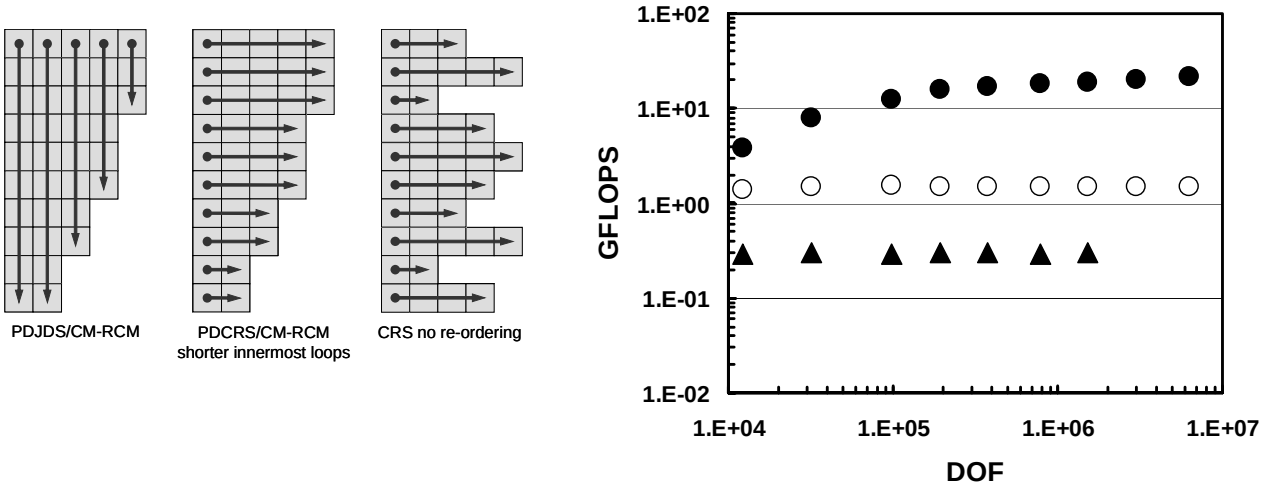
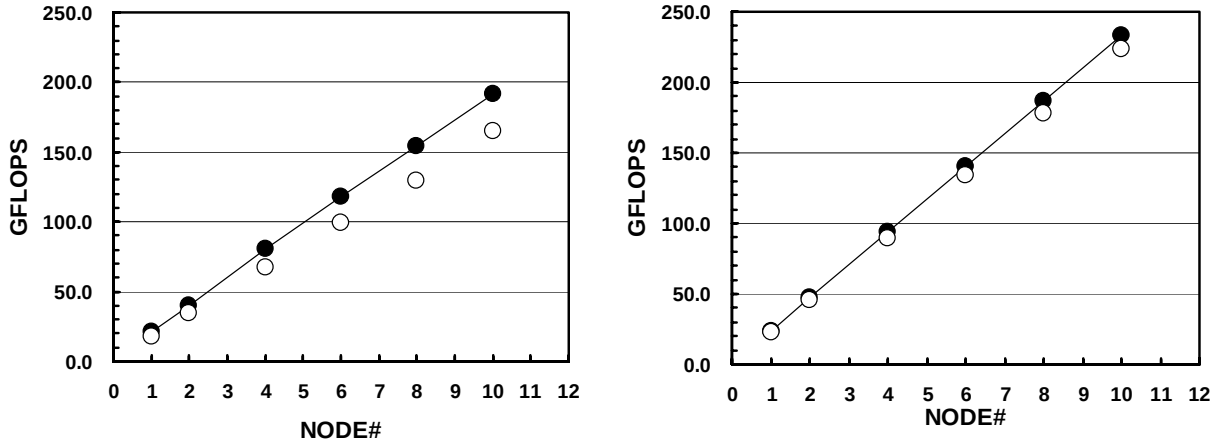


Fig.15 Effect of coefficient matrix storage method and reordering for the 3D linear elastic problem in Fig.14 with various problem sizes on the Earth Simulator with a single SMP node. (BLACK Circles: PDJDS/CM-RCM, WHITE Circles: PDCRS/CM-RCM, BLACK Triangles: CRS no reordering).

(3) Performance Evaluation for Large-Scale Problems

Figures 16-19 show the results for large-scale problems having simple geometries and boundary conditions as in Fig.14 implemented up to 176 SMP nodes of the Earth Simulator (1,408 PEs, 11.26 TFLOPS peak performance, 2.8 TB memory). Performance of the hybrid and flat MPI models were evaluated. The problem size for one SMP node was fixed and the number of nodes was varied between 1 and 176. The largest problem size was $176\times3\times128\times128\times256$ (2,214,592,512) DOF, for which the performance was about 3.80 TFLOPS, corresponding to 33.7 % of the total peak performance of the 176 SMP nodes. The parallel work ratio among SMP nodes for MPI is more than 90% if the problem is sufficiently large.

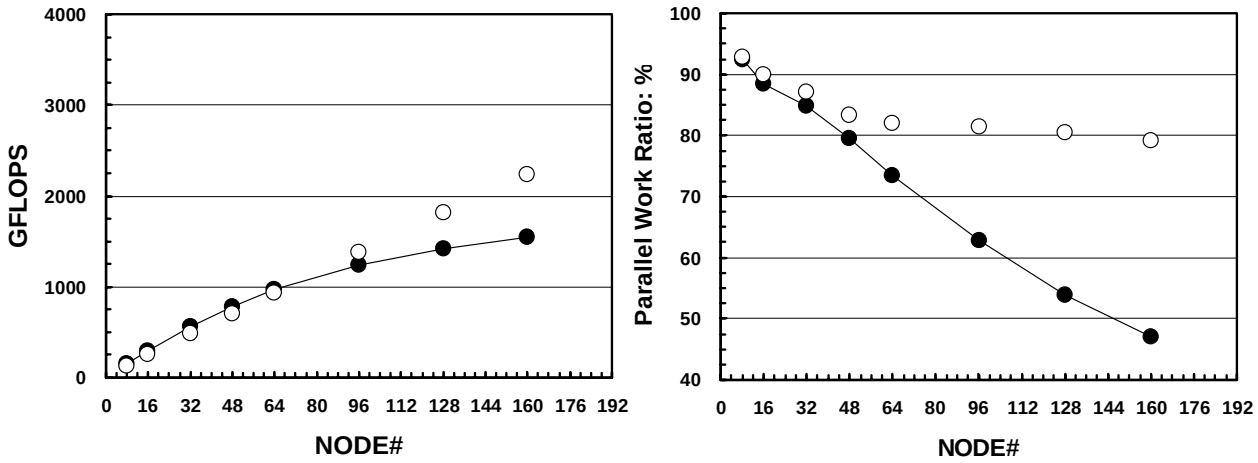
The performance of the hybrid model is competitive with that of the flat MPI model, and both provide robust convergence and good parallel performance for a wide range of problem sizes and SMP node numbers. Iterations for convergence in the hybrid and flat MPI are almost equal, although the hybrid converges slightly faster as shown in Fig.19(a). In general, flat MPI performs better the hybrid model for smaller numbers of SMP nodes as shown in Fig.16, while the hybrid outperforms flat MPI when a large number of SMP nodes are involved (Fig.17-19), especially if the problem size per node is small as shown in Fig.17 and Fig.19. This is mainly because of the latency overhead for MPI communication. According to the performance estimation for finite-volume application code for CFD with local refinement in [8], a greater percentage of time is required by the latency component on larger processor counts, simply due to the available bandwidth being much larger (Fig.20). Flat MPI requires eight times as many MPI processes as hybrid model. If the node number is large and problem size is small, this effect is significant.



(a) Problem size/SMP = 786,432 DOF (3×64^3). Largest case is 7,864,320 DOF on 10 SMP nodes (80 PEs). Maximum performance is 192 (Flat MPI) and 165 (Hybrid) GFLOPS.

(b) Problem size/SMP node = 12,582,912 DOF ($3\times256\times128\times128$). Largest case is 125,829,120 DOF on 10 SMP nodes (80 PEs). Maximum performance is 233 (Flat MPI) and 224 (Hybrid) GFLOPS

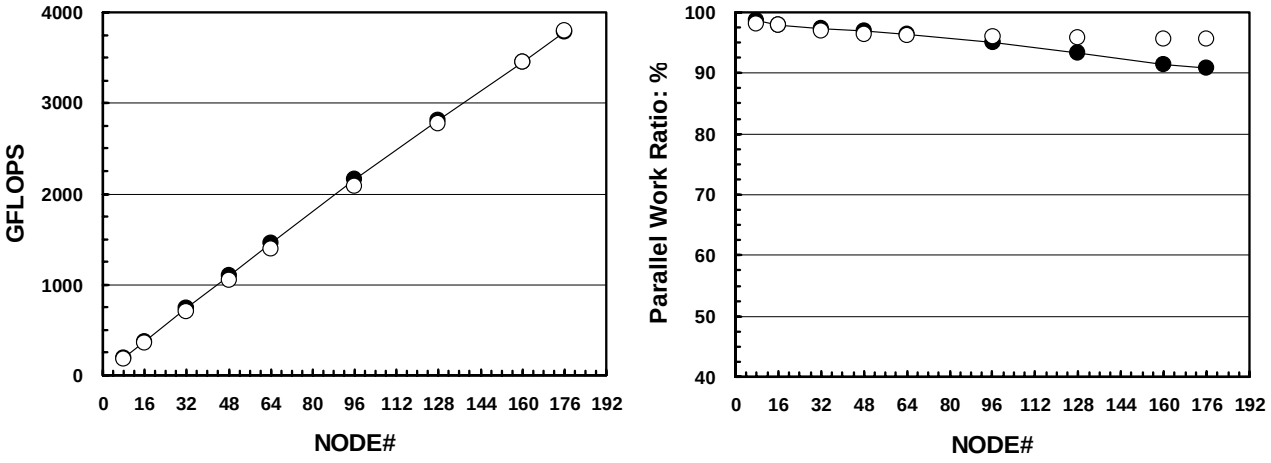
Fig.16 Problem size and parallel performance on the Earth Simulator for the 3D linear elastic problem in Fig.14 using between 1 and 10 SMP nodes. (BLACK Circles: Flat MPI, WHITE Circles: Hybrid). PDJDS/CM-RCM re-ordering.



(a) GFLOPS Rate

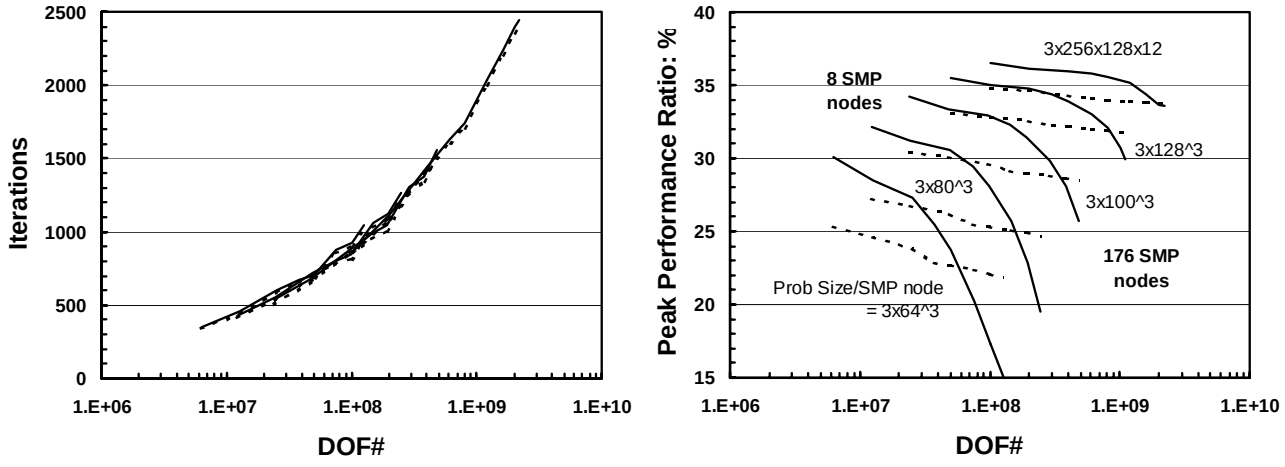
(b) Parallel Work Ratio

Fig.17 Problem size and parallel performance on the Earth Simulator for the 3D linear elastic problem in Fig.14 using between 8 and 160 SMP nodes. (a)GFLOPS rate and (b)Parallel work ratio. Problem size/PE is fixed as 786,432 DOF (3×64^3). Largest case is 125,829,120 DOF on 160 SMP nodes (1280 PEs). Maximum performance is 1.55 (Flat MPI) and 2.23 (Hybrid) TFLOPS (Peak performance= 10.24 TFLOPS). (BLACK Circles: Flat MPI, WHITE Circles: Hybrid). PDJDS/CM-RCM reordering.



(a) GFLOPS Rate (b) Parallel Work Ratio

Fig.18 Problem size and parallel performance on the Earth Simulator for the 3D linear elastic problem in Fig.14 using between 8 and 176 SMP nodes. (a)GFLOPS rate and (b)Parallel work ratio. Problem size/SMP node is fixed as 12,582,912 DOF (3×256×128×128). Largest case is 2,214,592,512 DOF on 176 SMP nodes (1408 PEs). Maximum performance is 3.78 (Flat MPI) and 3.80 (Hybrid) TFLOPS (Peak performance= 11.26 TFLOPS). (BLACK Circles: Flat MPI, WHITE Circles: Hybrid). PDJDS/CM-RCM reordering.



(a) Iterations for Convergence (b) Peak Performance Ratio

Fig.19 Problem size and parallel performance on the Earth Simulator for the 3D linear elastic problem in Fig.14 using between 8 and 176 SMP nodes. (a) Iterations for convergence and (b) Ratio to the peak performance. Problem size/SMP node is fixed (THICK lines: Flat MPI, DASHED lines: Hybrid). PDJDS/CM-RCM reordering.

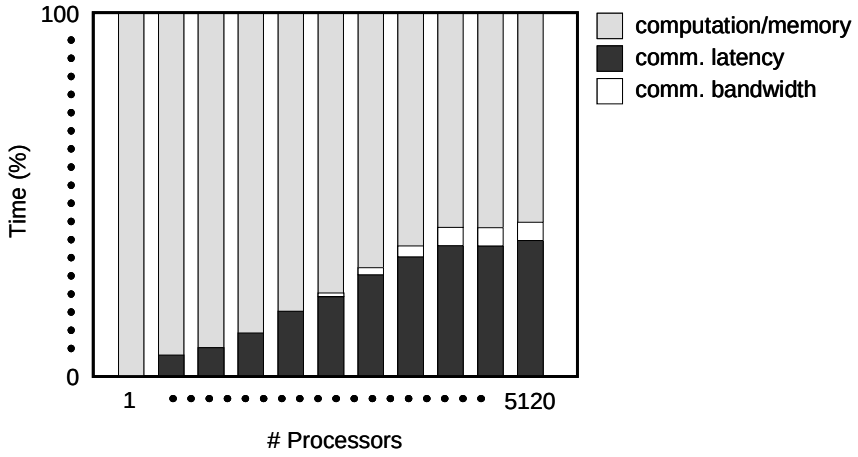
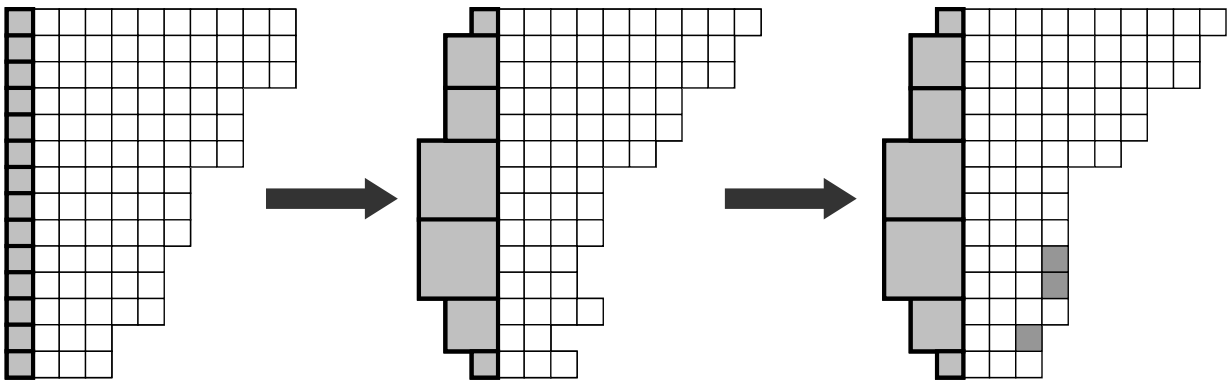


Fig.20 Performance estimation of a finite-volume application code for CFD with local refinement on the Earth Simulator. Based on the results described in [8]. A greater percentage of time is taken by the latency component on larger processor counts, simply due to its much larger available bandwidth.

4.7 Special Treatments for Selective Blocking

In selective blocking preconditioning, individual selective blocks (or super nodes) are computed independently; therefore, dependency among selective blocks should be considered at reordering for vector optimization. In this case, a load imbalance may occur because the size of each selective block differs according to the number of nodes in each contact group. Ordinary nodes which do not belong to any contact group are considered as a selective block of size 1. Currently, no special treatment for load-balancing is implemented.

Another problem is that, in DJDS reordering, the number of off-diagonal components may not reduce smoothly according to the size of the contact groups and the number of connected nodes, as shown in Fig.21. In this case, dummy elements are placed in order to maintain a smooth decrease in the number of off-diagonal components in descending order. If several dummy elements exist, efficiency and load balancing may be affected. Finally, block diagonal components for selective blocks are reordered according to the block size on each PE and for each color, as shown in Fig.22. Thus, *if* statements according to block size are eliminated from the full LU factorization procedure for each selective block during back/forward substitution.



(a) DJDS reordered elements

(b) Profile according to selective blocking

(c) Dummy components (shaded in gray color)

Fig.21 Dummy elements to maintain a smooth decrease in the number of off-diagonal components in descending order

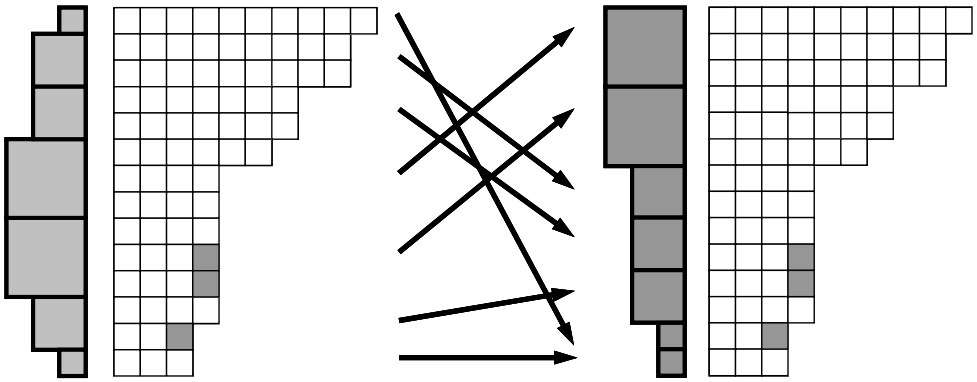


Fig.22 Reordering of selective blocks (super nodes) according to block size for full LU factorization.

5. Examples on the Earth Simulator

5.1 Overview

The efficiency and robustness of the *selective blocking* preconditioning and partitioning methods for simulations of fault-zone contact were evaluated in two types of 3D applications on the Earth Simulator.

Figure 23 shows the simple geometry and boundary conditions of an example model for 3D linear elastic solid mechanics. In this example, linear multiple point constraint (MPC) conditions were applied to the nodes of the contact groups in the following manner:

- The locations of nodes in each contact group are identical.
- All nodes in the contact groups are coupled tightly in any direction on the surfaces.
- Infinitesimal linear elastic deformation theory in solid mechanics was applied. Therefore nodes do not move, and the contact relationships have been kept during the simulation.
- A penalty constraint is applied to the nodes in the contact groups. 111-type element (Rod/Beam) in GeoFEM [2] is put in each contact group and very large stiffness corresponding to penalty is applied. Figure 24 shows the matrix operation of nodes in a contact group. Three components in x, y, and z directions are constrained through penalty.

A large penalty parameter provides a stronger constraint but the condition numbers of the coefficient matrices are larger. Therefore, the convergence of iterative solvers is usually slow if the penalty is large. The problem itself is linear elastic, but solving linear equations by iterative methods is as difficult as solving equations in nonlinear contact problems, such as those shown in previous sections. The definitions of the model and boundary conditions are as follows:

- Three zones of uniform material property for which the non-dimensional E (Young's modulus)=1.0, ν (Poisson's ratio)=0.30. Tri-linear (1st order) cubic hexahedral elements are used for spatial discretization.
- Uniform MPC conditions were imposed on the nodes along the boundary surfaces of the blocks. Therefore, the number of nodes in each contact group can be different, as shown in Fig.23(b).
- Symmetry boundary conditions were applied at the x=0 and y=0 surfaces.
- Free boundary conditions were applied at the $x=X_{max}$ and $y=Y_{max}$ surfaces.
- Dirichlet (fixed) boundary conditions were applied at the z=0 surface.
- A uniformly distributed load in the z-direction was applied at the $z=Z_{max}$ surface.

- If friction is not considered at fault surfaces, the coefficient matrix is symmetric positive definite; therefore, the CG method was adopted.

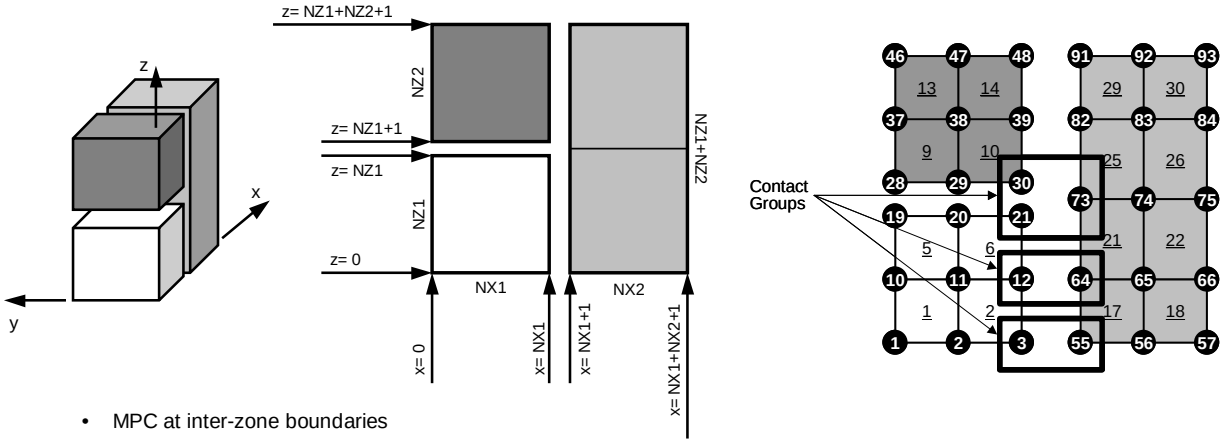
All of the meshes in this example are uniform cubes. Following three types of models are considered in this example:

Single SMP node test
- NX1=70, NX2=70, NY =40, NZ1= 70, NZ2= 70 (Fig.23 (a))
- Total Elements = 784,000, Total Nodes= 823,813, Total DOF= 2,471,439

Speed-up test
- NX1=70, NX2=70, NY =168, NZ1= 70, NZ2= 70 (Fig.23(a))
- Total Elements = 3,292,800, Total Nodes= 3,395,717, Total DOF= 10,187,151

Large-scale test
- NX1=300, NX2=300, NY =40, NZ1= 200, NZ2= 200 (Fig.23 (a))
- Total Elements = 9,600,000, Total Nodes= 9,909,823, Total DOF= 29,729,469

The second example, shown in Fig.25, is a more complicated model for earthquake simulation in the southwestern part of Japan [3]. This model consists of crust (dark gray) and subduction plates (light gray). There are 997,422 nodes (2,992,266 DOF) and 960,509 tri-linear (1st-order) hexahedral elements. Average size of mesh is order of 15 km. The same boundary conditions as those used in the model of Fig.23 were applied here. In this Southwest Japan model, a body force of -1.0 was applied in the z-direction rather than a surface force at the $z=Z_{max}$ surface, as in Fig.23, and no symmetry boundary conditions were applied in the x or y directions. In this example, the meshes are irregular, and some of the meshes are very distorted. The material property is linear and homogeneous (E=1.0, ν =0.30). The globally refined model with 7,767,002 nodes (23,301,006 DOF) and 7,620,057 elements was also tested using larger number of SMP nodes.



(a) Model and boundary conditions (b) Node, elements and contact groups

Fig.23 Description of the simple block model

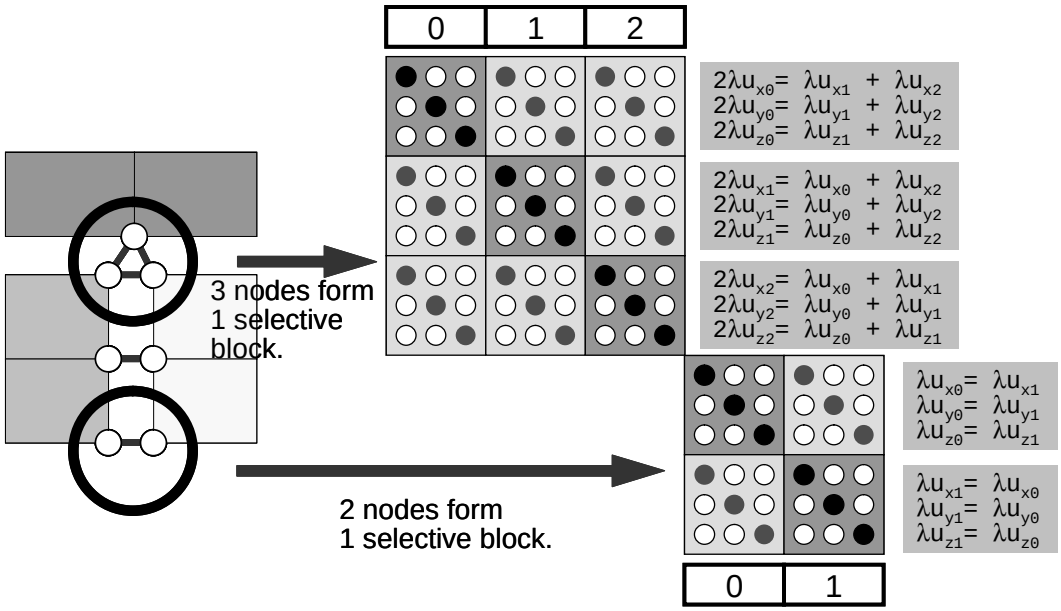


Fig.24 Matrix operation of nodes in a contact group [4]

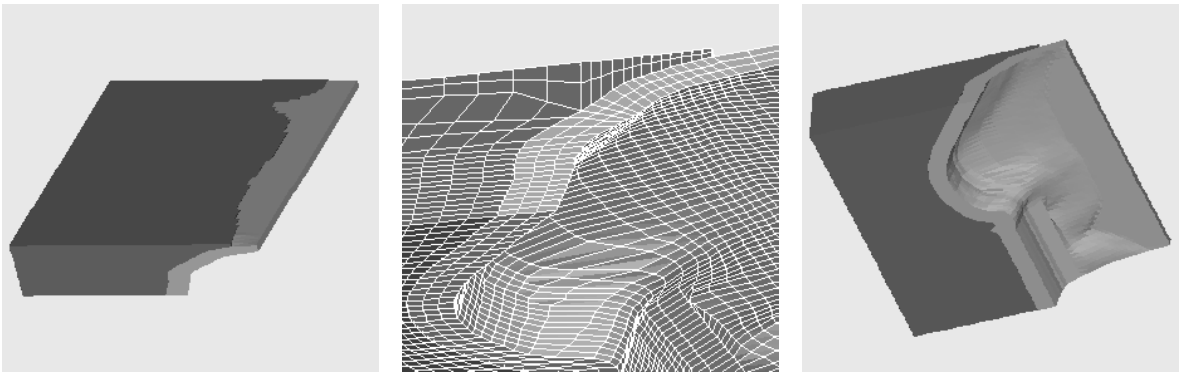


Fig.25 Description of the Southwest Japan model This model consists of crust (dark gray) and subduction plate (light gray). 997,422 nodes (2,992,266 DOF) and 960,509 tri-linear (1st order) hexahedral elements are included.

5.2 Results I (Single SMP Node)

Example tests using a single SMP node of the Earth Simulator have been done for the simple block model (NX1=70, NX2=70, NY =40, NZ1= 70, NZ2= 70 (Fig.23), 784,000 elements, 823,813 nodes, 2,471,439 DOF) and original Southwest Japan model (960,509 elements, 997,422 nodes, 2,992,266 DOF) for both of flat MPI and hybrid parallel programming model

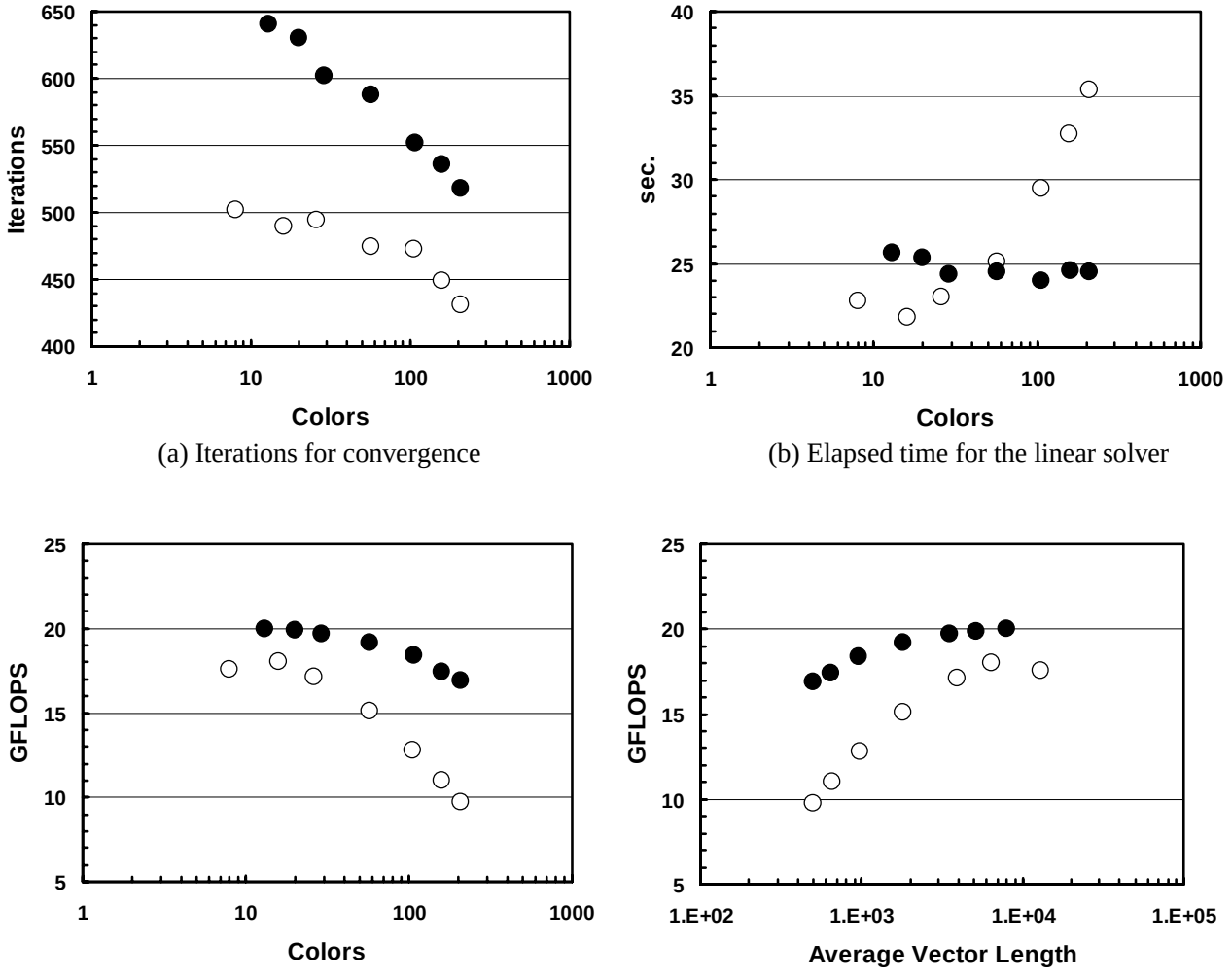
Figure 26 and 27 show the results. In the cases with many colors, fewer iterations are required for convergence, but the performance is worse due to the smaller loop length and greater overhead. In the Southwest Japan model, iterations for convergence are not affected by number of colors. This is because there are many distorted elements in this model and the coefficient matrices are ill-conditioned. In the hybrid parallel programming model, performance of 17.6 GFLOPS (27.5% of peak performance, which is 64 GFLOPS) for the simple block model and 18.6 GFLOPS (29.1% of peak performance) for the Southwest Japan has been obtained. This performance is as good as the results in Fig.16 by ICCG solvers for simple geometries with homogeneous boundary conditions, which is 20.5 GFLOPS for 3M DOF (32.1% of the peak performance). As for the flat MPI, the performance was 20.0 GFLOPS

(31.2%) for the simple block model and 20.1 GFLOPS (31.5%) for the Southwest Japan, respectively. Number of iterations for convergence is smaller in hybrid than in flat MPI due to the effect of local preconditioning, but performance based on GFLOPS rate is better in flat MPI. Another feature is that hybrid parallel programming model is much more sensitive to color number and innermost vector length than flat MPI. If the number of colors increases, effect of synchronization overhead in OpenMP increases, as is shown in Fig.13.

Figure 28 compares the performance with results obtained by the method without reordering the selective blocks according to block size, as shown in Fig.22. Performance is about 60% if this reordering is not applied. Figure 29 shows the load-imbalance among PEs on the SMP node and the ratio of dummy off-diagonal components. Load-imbalance is computed by the following method:

Load Imbalance (%) = 100 × (max. node # - min. node #) / average node #

The effect of load-imbalance and dummy elements is very small in both models and the effect is negligible in this computation.



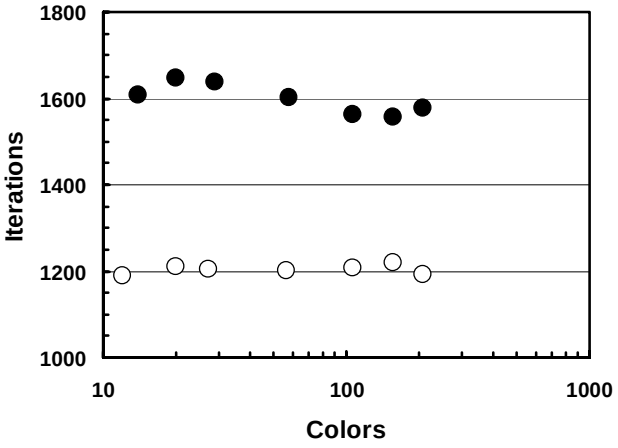
(a) Iterations for convergence

(b) Elapsed time for the linear solver

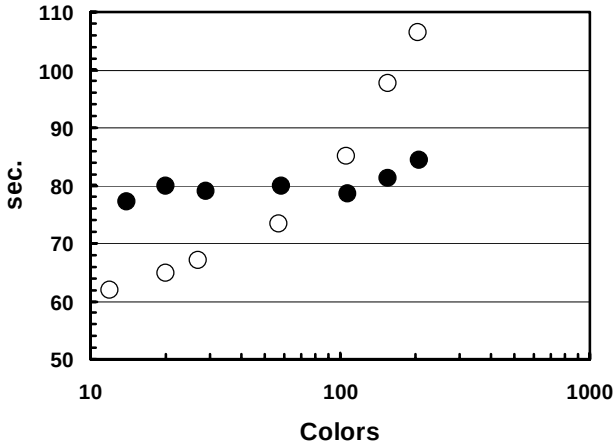
(c) GFLOPS rate (based on colors)

(d) GFLOPS rate (based on average vector length)

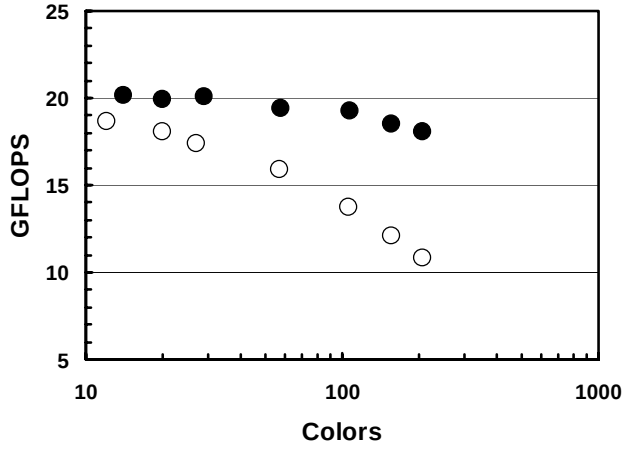
Fig.26 Performance on a single SMP node of the Earth Simulator (peak performance = 64GFLOPS) using SB-BIC(0) CG with PDJDS/MC reordering for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) in Fig.23 (Simple Block Model, 2,471,439 DOF). (a) Iterations for convergence, (b) Elapsed time for the linear solver, (c) GFLOPS rate (based on colors) and (d) GFLOPS rate (based on average vector length) (BLACK Circles: Flat MPI, WHITE Circles: Hybrid).



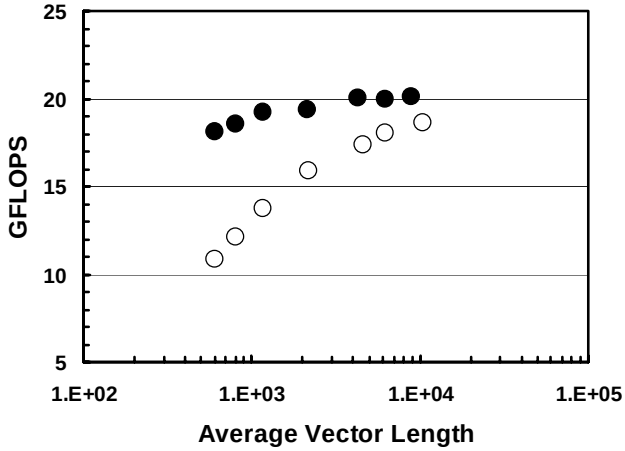
(a) Iterations for convergence



(b) Elapsed time for the linear solver

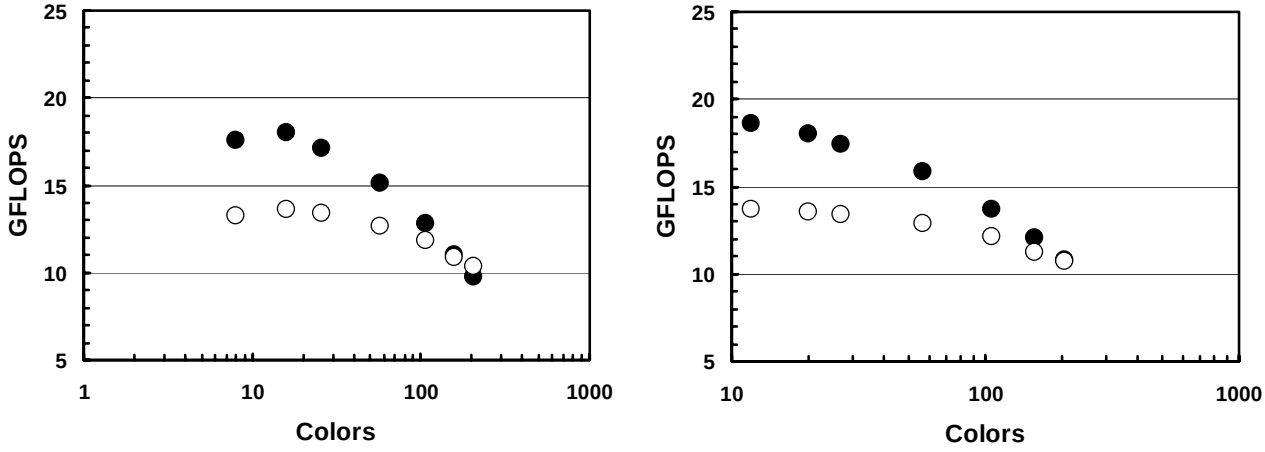


(c) GFLOPS rate (based on colors)



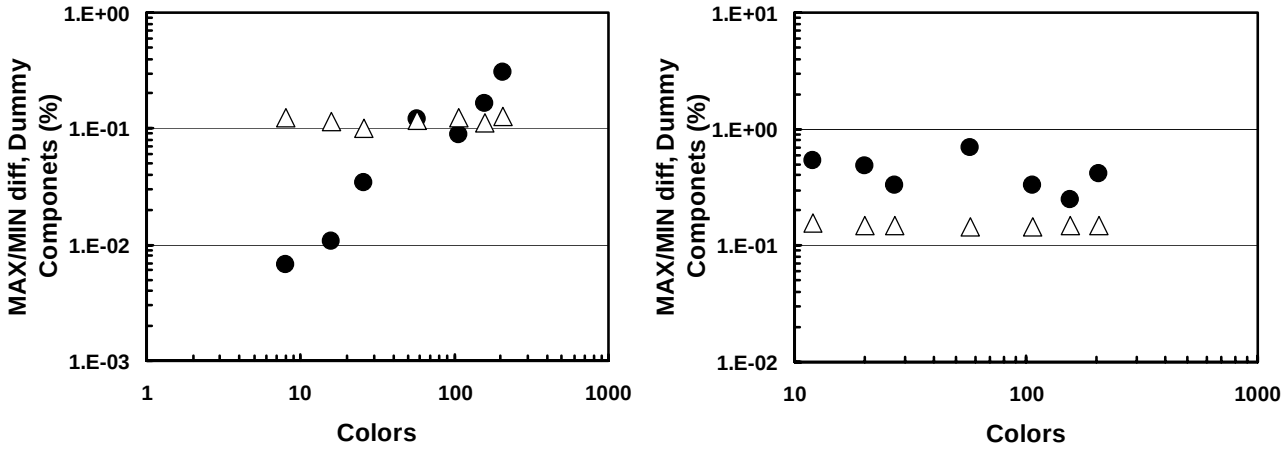
(d) GFLOPS rate (based on average vector length)

Fig.27 Performance on a single SMP node of the Earth Simulator (peak performance = 64GFLOPS) using SB-BIC(0) CG with PDJDS/MC reordering for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) in Fig.25 (Southwest Japan model, 2,992,266 DOF). (a) Iterations for convergence, (b) Elapsed time for the linear solver, (c) GFLOPS rate (based on colors) and (d) GFLOPS rate (based on average vector length) (BLACK Circles: Flat MPI, WHITE Circles: Hybrid).



(a) Simple Block (b) Southwest Japan

Fig.28 Effect of reordering of selective block on a single SMP node of the Earth Simulator (peak performance = 64GFLOPS) using SB-BIC(0) CG for the 3D elastic contact problem with MPC condition ($\lambda=10^6$). Hybrid parallel programming model. (a) Simple Block (b) Southwest Japan. (BLACK Circles: WITH reordering, WHITE Circles: WITHOUT reordering).



(a) Simple Block (b) Southwest Japan

Fig.29 Load imbalance on a single SMP node for selective blocking preconditioning. Hybrid parallel programming model. (a) Simple Block (b) Southwest Japan. (BLACK Circles: Load-imbalance among PEs on the SMP node, WHITE Triangles: Ratio of dummy off-diagonal components).

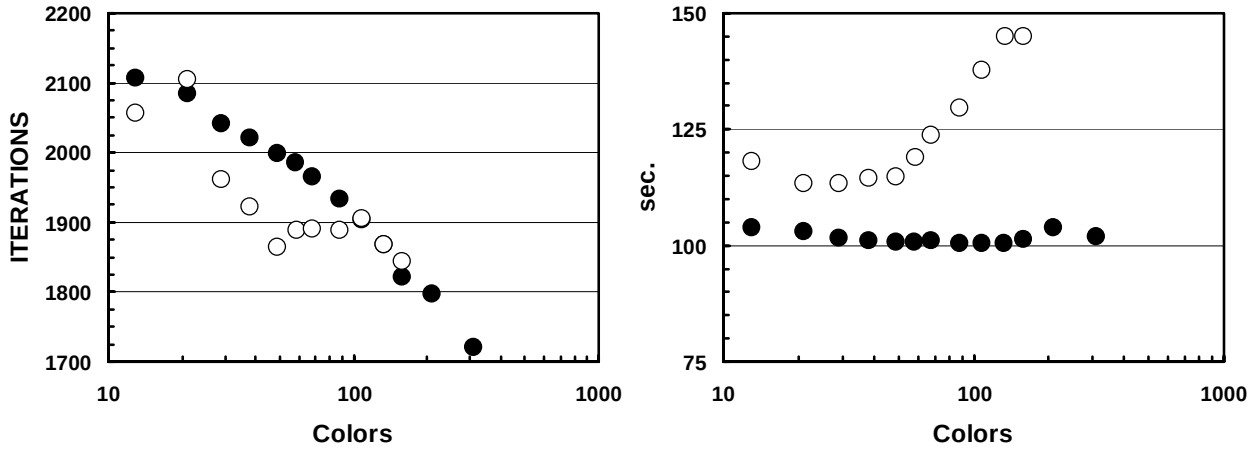
5.3 Results II (10 SMP Nodes)

Example tests using 10 SMP nodes of the Earth Simulator have been done for the simple block model (NX1=300, NX2=300, NY =40, NZ1= 200, NZ2= 200 (Fig.23), 9,600,000 elements, 9,909,823 nodes, 29,729,469 DOF) and globally refined Southwest Japan model (7,684,072 elements, 7,767,002 nodes, 23,301,006 DOF) for both of flat MPI and hybrid parallel programming model

Figure 30 and 31 show the results. In the cases with many colors, fewer iterations are required for convergence, but the performance is worse due to the smaller loop length and greater overhead. In the hybrid parallel programming model, performance of 178.4 GFLOPS (27.9 % of peak performance, 640 GFLOPS) for the simple block model and 163.4 GFLOPS (25.5% of peak performance) for the Southwest Japan has been obtained. As for the flat MPI, the performance was 195.0 GFLOPS (30.5%) for the simple block model and 190.4 GFLOPS (29.8%) for

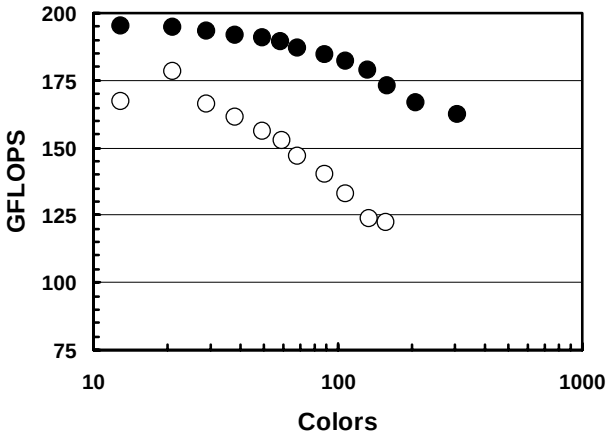
the Southwest Japan, respectively. These results are as excellent as those in Fig.19 by ICCG solvers for simple geometries with homogeneous boundary conditions, which are 196.1 GFLOPS for 30M DOF (30.6% of the peak performance) with hybrid programming model, and 218.9 GFLOPS (34.2%) with flat MPI. Number of iterations for convergence is smaller in hybrid than in flat MPI due to the effect of local preconditioning, but performance based on GFLOPS rate is better in flat MPI. Hybrid parallel programming model is much more sensitive to color number and innermost vector length than flat MPI due to the synchronization overhead.

Figure 32 shows the increase in speed-up for fixed problem size for the simple block model (NX1=70, NX2=70, NY =168, NZ1= 70, NZ2= 70 (Fig.23), 3,292,800 elements, 2,395,717 nodes, 10,187,151 DOF) using between 1 and 10 SMP nodes (8 and 80 PEs). Two cases with different number of colors (13 colors and 30 colors) have been evaluated. The speed-up ratio for 80 PEs from 16 PEs for hybrid parallel programming model (the case with 1 SMP node did not work for hybrid programming model due to memory shortage) was 64.9 (13 colors, 81.1% of the linear ideal speed-up) and 59.5 (30 colors, 74.4%). The speed-up ratio for 80 PEs from 8 PEs for flat MPI was 68.8 (13 colors, 86.0%) and 67.1 (30 colors, 83.9%). Parallel speed-up performance is better in the cases with a smaller number of colors.

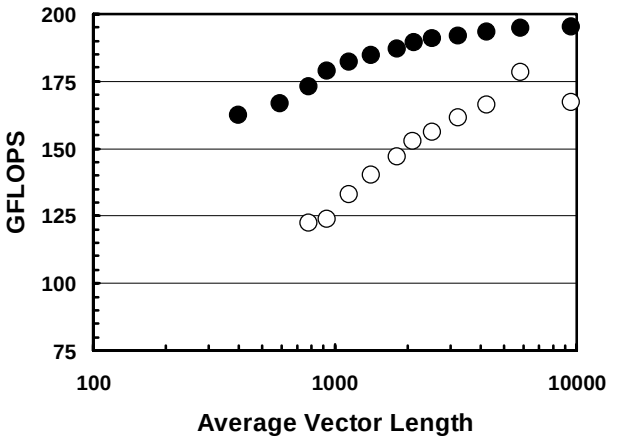


(a) Iterations for convergence

(b) Elapsed time for the linear solver

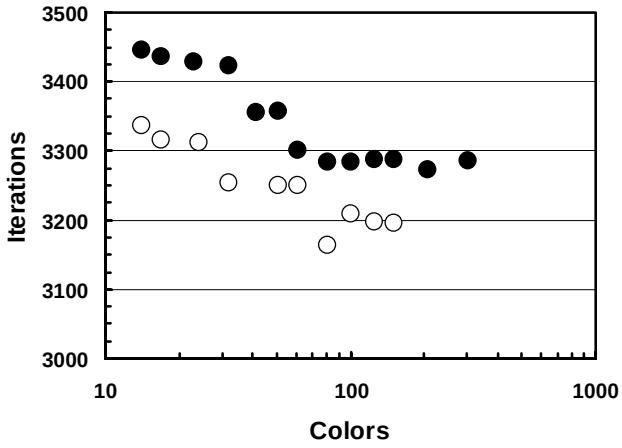


(c) GFLOPS rate (based on colors)

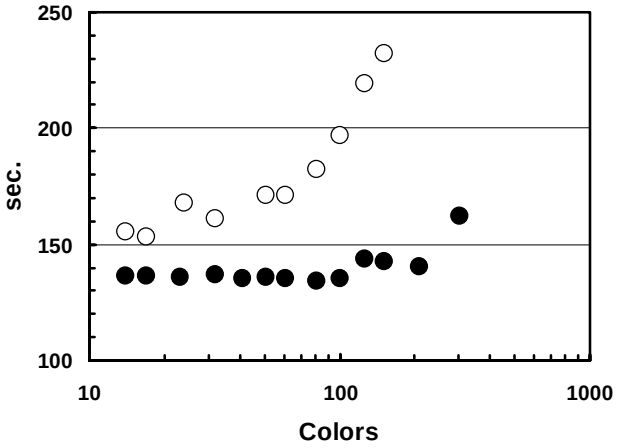


(d) GFLOPS rate (based on average vector length)

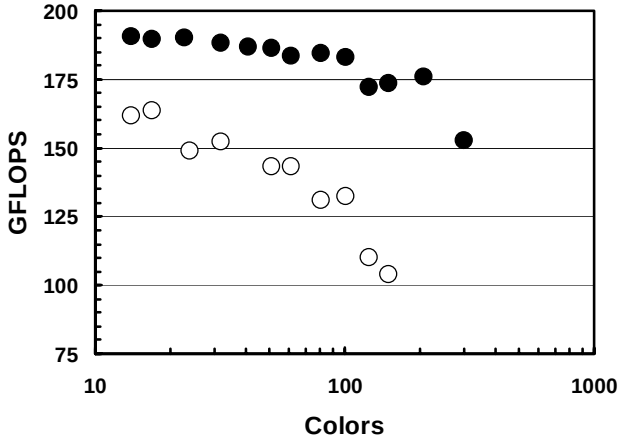
Fig.30 Performance on 10 SMP node of the Earth Simulator (peak performance = 640 GFLOPS) using SB-BIC(0) CG with PDJDS/MC reordering for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) in Fig.23 (Simple Block Model, 29,729,469 DOF). (a) Iterations for convergence, (b) Elapsed time for the linear solver, (c) GFLOPS rate (based on colors) and (d) GFLOPS rate (based on average vector length) (BLACK Circles: Flat MPI, WHITE Circles: Hybrid).



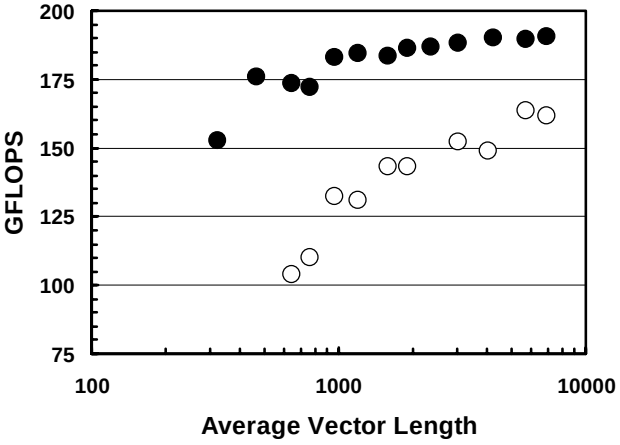
(a) Iterations for convergence



(b) Elapsed time for the linear solver

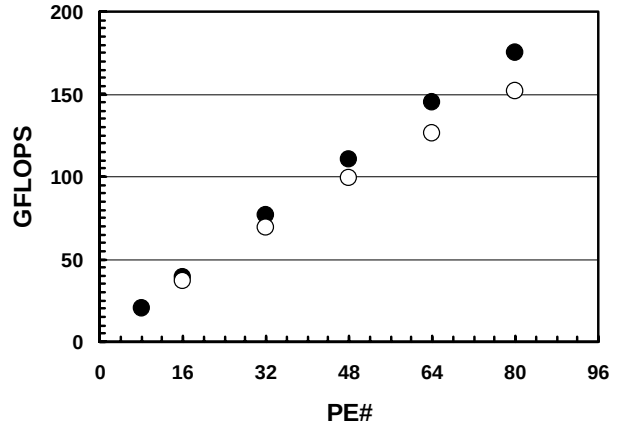


(c) GFLOPS rate (based on colors)

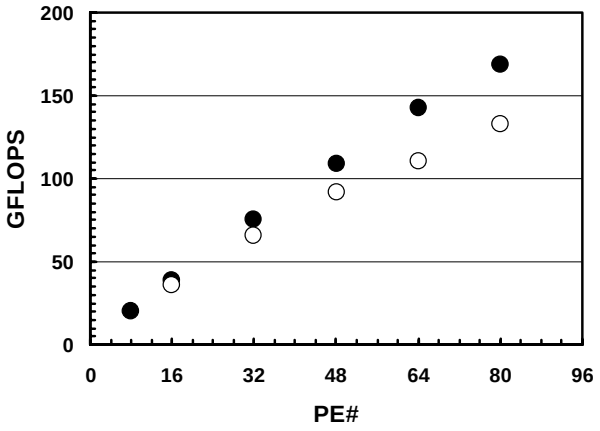


(d) GFLOPS rate (based on average vector length)

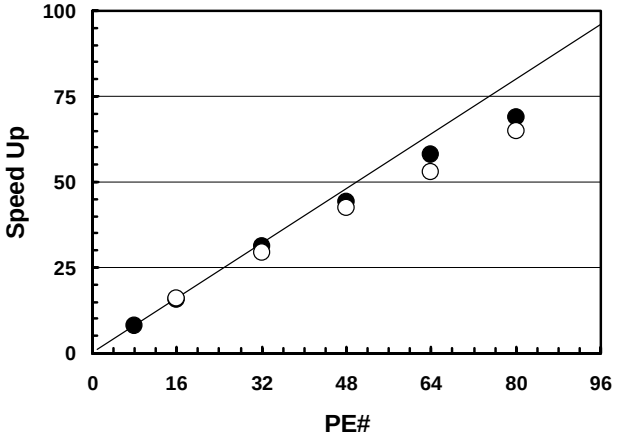
Fig.31 Performance on 10 SMP node of the Earth Simulator (peak performance = 640 GFLOPS) using SB-BIC(0) CG for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) in Fig.25 (Southwest Japan model, 23,301,006 DOF). (a) Iterations for convergence, (b) Elapsed time for the linear solver, (c) GFLOPS rate (based on colors) and (d) GFLOPS rate (based on average vector length) (BLACK Circles: Flat MPI, WHITE Circles: Hybrid).



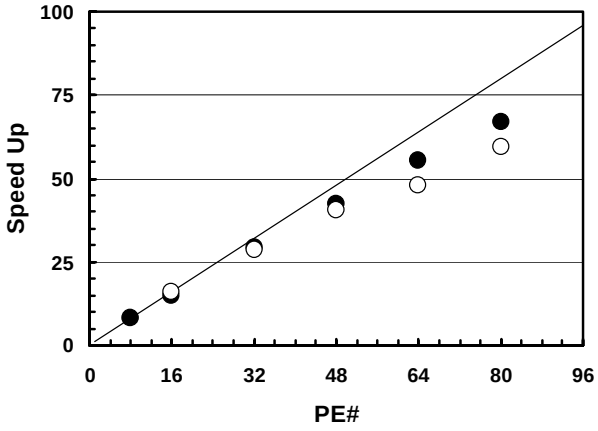
(a-1) GFLOPS rate (13 colors)



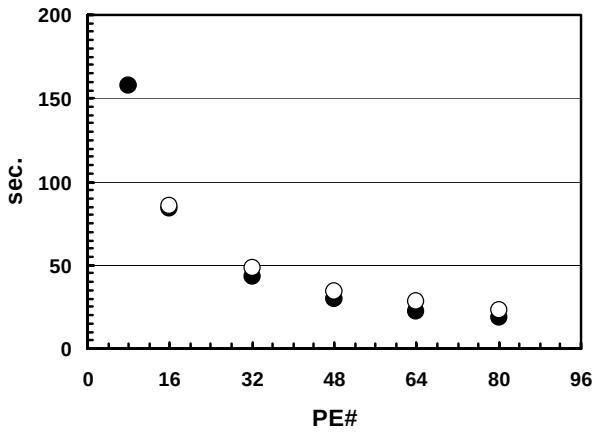
(a-2) GFLOPS rate (30 colors)



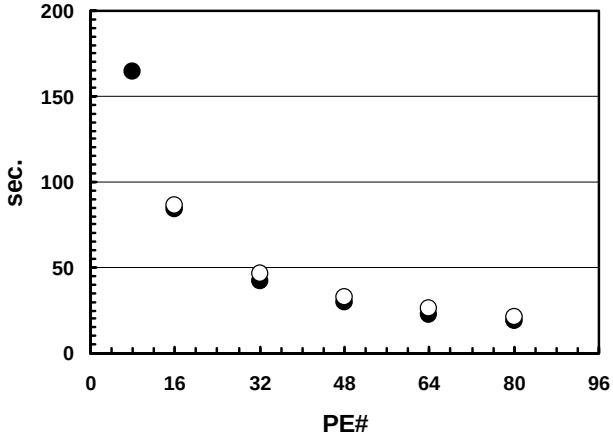
(b-1) Speed-up ratio (13 colors)



(b-2) Speed-up ratio (30 colors)



(c-1) Elapsed time for the linear solver (13 colors)



(c-2) Elapsed time for the linear solver (30 colors)

Fig.32 Parallel speed-up from 1 to 10 SMP node (8-80 PEs) of the Earth Simulator using SB-BIC(0) CG with PDJDS/MC reordering for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) in Fig.23 (Simple Block Model, 10,187,151 DOF). (a-1) GFLOPS rate (13 colors), (a-2) GFLOPS rate (30 colors), (b-1)Speed-up ratio (13 colors), (b-2)Speed-up ratio (30 colors), (c-1) Elapsed time for the linear solver (13 colors), and (c-2) Elapsed time for the linear solver (30 colors) (BLACK Circles: Flat MPI, WHITE Circles: Hybrid).

6. Concluding Remarks and Future Study

An efficient parallel iterative method with selective blocking preconditioning has been developed for the Earth Simulator using hybrid parallel programming model. The method is based on a three-level hybrid parallel programming model, which includes message passing for inter-SMP node communication, loop directives by OpenMP for intra-SMP node parallelization and vectorization for each processing element. Developed method provides robust and smooth convergence and excellent parallel performance in 3D geophysical simulations in GeoFEM for both simple and complicated geometries with contact conditions on the Earth Simulator. The *selective blocking* preconditioning is much more efficient than ILU(1) and ILU(2).

The reordering method for SMP cluster architectures with vector processors shown in [5] has been implemented with the selective blocking preconditioning using the MC reordering method. Special treatments for selective blocking are implemented, which include the introduction of dummy elements and reordering of selective blocks according to block size.

In the cases with many colors, fewer iterations are required for convergence, but the performance is worse due to the smaller loop length and greater overhead. Performance of the Earth Simulator is much affected by loop length. Usually, performance (GFLOPS rate and elapsed time) is better for a smaller number of colors even though more iterations are required for convergence. Number of iterations for convergence is smaller in hybrid than in flat MPI due to the effect of local preconditioning, but performance based on GFLOPS rate is better in flat MPI. Another feature is that hybrid parallel programming model is much more sensitive to color number and innermost vector length than flat MPI. If the number of colors increases, effect of synchronization overhead in OpenMP increases.

Performance of 17.6 GFLOPS (27.5% of peak performance, which is 64 GFLOPS) for the simple block model and 18.6 GFLOPS (29.1% of peak performance) for the Southwest Japan has been obtained on a single SMP node of the Earth Simulator using hybrid parallel programming model. As for the flat MPI, the performance was 20.0 GFLOPS (31.2%) for the simple block model and 20.1 GFLOPS (31.5%) for the Southwest Japan, respectively. Performance is about 60% in hybrid parallel programming model, if the reordering of selective blocks is not applied. Results show that the load-imbalance among PEs on the SMP node and the ratio of dummy off-diagonal component are not significant. Furthermore, the globally refined Southwest Japan model with more than 23 M DOF was tested on 10 SMP nodes (80 PEs) of the Earth Simulator. The best performance is 163.4 GFLOPS, corresponding to 25.5% of peak performance using hybrid programming model, and 190.4 GFLOPS (29.8%) by flat MPI. Parallel speed-up for fixed problem size between 1 and 10 SMP nodes was more than 80% of the linear ideal speed-up.

In future, large-scale simulations for ground motion will be conducted using more detailed realistic physical models with finer meshes, where mesh size is less than 1 km and a greater number of SMP nodes of the Earth Simulator. According to the current results, parallel and vector performance is as excellent as that in section 4.6 of this paper (Fig.16-19) by ICCG solvers of GeoFEM for simple geometries with homogeneous boundary conditions. Therefore, the *selective blocking* method is expected to show excellent parallel and vector performance in large-scale problems with hundreds of SMP nodes on the Earth Simulator, as ICCG solvers of GeoFEM attained 3.80 TFLOPS using 176 SMP nodes of the Earth Simulator (1408 PEs, 33.7% of the peak performance) for simple geometries with 2.2G DOF.

One of the most critical issues for large-scale computing is mesh generation. Mesh generation is easy for such simple geometry as shown in Fig.14, even if number of domain is more than 1,000, and total mesh number is more than 100M. But it is very difficult for complicated geometries, such as Southwest Japan model. Parallel scalable mesh generation method, as is described in [23] should be investigated and developed so that computational resources of the Earth Simulator would be efficiently utilized.

Selective blocking method provides robust and smooth convergence, and excellent parallel and vector performance in 3D geophysical simulations with ill-conditioned coefficient matrices. This method requires that all nodes in same contact groups are on same partition in parallel computation. This requirement might be very difficult for large-scale problems with complicated geometries and boundary conditions. Alternative method, such as multilevel method in [24], should be also considered in future study.

Acknowledgments

This study is a part of *the Solid Earth Platform for Large-Scale Computation* project funded by the Ministry of Education, Culture, Sports, Science and Technology, Japan through *Special Promoting Funds of Science & Technology*. The author would like to thank Mr. Shin'ichi Ezure (RIST) for providing finite-element mesh of the Southwest Japan model.

References

[1] Earth Simulator Center Web Site: http://www.es.jamstec.go.jp/
[2] GeoFEM Web Site: http://geofem.tokyo.rist.or.jp/
[3] Iizuka, M., Okuda, H. and Yagawa, G.: "Nonlinear Structural Subsystem of GeoFEM forFault Zone Analysis", *Pure and Applied Geophysics*, Vol.157 (2000), pp.2105-2124.
[4] Nakajima, K. and Okuda, H. "Parallel Iterative Solvers with Selective Blocking Preconditioning for Simulations of Fault Zone Contact", *2001 International Conference on Preconditioning Techniques for Large Sparse Matrix Problems in Industrial Applications (Preconditioning 2001)*, Tahoe City, CA, USA, 2001, *Journal of Numerical Algebra with Applications (in press)*.
[5] Nakajima, K.: "OpenMP/MPI Hybrid vs. Flat MPI on the Earth Simulator: Parallel Iterative Solvers for Finite Element Method", *International Workshop on OpenMP: Experiences and Implementations (WOMPEI 2003)*, Tokyo, Japan, *Lecture Notes in Computer Science 2858* (*in press*), Springer, 2003.
[6] Nakajima, K., Okuda, H.: "Parallel Iterative Solvers with Localized ILU Preconditioning for Unstructured Grids on Workstation Clusters", *International Journal for Computational Fluid Dynamics*, Vol.12 (1999) pp.315-322.
[7] Smith, B., Bjφrstad, P. and Gropp, W.: *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge Press, 1996.
[8] Dongarra, J.J., Duff, I.S., Sorensen, D.C., and van der Vorst, H.A.: *Numerical Linear Algebra for High-Performance Computers*, SIAM (1998).
[9] Liou, J. and Tezduyar, T.E.: "Clustered Element-by-Element Computations for Fluid Flow", *Parallel Computational Fluid Dynamics (Implementations and Results)*, edited by H.D.Simon, The MIT Press, 1992. pp.167-187.
[10] Accelerated Strategic Computing Initiative (ASCI) Web Site : http://www.llnl.gov/asci/
[11] Cappelo, F. and Etiemble, D. :"MPI versus MPI+OpenMP on the IBM SP for the NAS Benchmarks", *SC2000 Technical Paper*, Dallas, Texas, 2000.
[12] Djomehri, M.J. and Jin, H.H.: "Hybrid MPI+OpenMP Programming of an Overset CFD Solver and Performance Investigations", *NASA/NAS Technical Report (NASA Ames Research Center)*, NAS-02-002, (2002). 16.
[13] Falgout, R. and Jones, J. : "Multigrid on Massively Parallel Architectures", *Sixth European Multigrid Conference*, Ghent, Belgium, September 27-30, 1999.
[14] Oliker, L, Li, X., Husbands, P. and Biswas, R.: "Effects of Ordering Strategies and Programming Paradigms on Sparse Matrix Computations", *SIAM Review*, Vol.44, No. 3 (2002), pp.373-393.
[15] MPI Web Site : http://www.mpi.org/
[16] OpenMP Web Site : http//www.openmp.org/
[17] Rabenseifner, R.: "Communication Bandwidth of Parallel Programming Models on Hybrid Architectures", *International Workshop on OpenMP: Experiences and Implementations (WOMPEI 2002), Lecture Notes in Computer Science 2327* (2002), pp.437-448.
[18] Washio, T., Maruyama, K., Osoda, T., Shimizu, F. and Doi, S. : "Blocking and reordering to achieve highly parallel robust ILU preconditioners", *RIKEN Symposium on Linear Algebra and its Applications*, The Institute of Physical and Chemical Research, 1999, pp.42-49.

[19] Washio, T., Maruyama, K., Osoda, T., Shimizu, F. and Doi, S. : "Efficient implementations of block sparse matrix operations on shared memory vector machines", *SNA2000 : The Fourth International Conference on Supercomputing in Nuclear Applications*, 2000.
[20] Saad, Y. : *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.
[21] Doi, S. and Washio : "Using Multicolor Ordering with Many Colors to Strike a Better Balance between Parallelism and Convergence", *RIKEN Symposium on Linear Algebra and its Applications*, The Institute of Physical and Chemical Research, 1999, pp.19-26.
[22] Kerbyson, D.J., Hoisie, A. and Wasserman, H. "A Comparison Between the Earth Simulator and AlphaServer Systems using Predictive Application Performance Models", *LA-UR-02-5222, Los Alamos National Laboratory*, USA, 2002.
[23] Ezure, S., Okuda, H. and Nakajima, K. : "Parallel Mesh Relocation, Parallel Finite Element Analysis, Large-Scale Simulation", *RIST/Tokyo GeoFEM Report 2002-012*, 2002.
[24] Saad, Y. and Zhang, J. : "BILUTM: A Domain-Based Multilevel Block ILUT Preconditioning for General Sparse Matrices", *SIAM Journal on Matrix Analysis and Applications*, Vol.21, No.21 (1999), pp.279-299.

Appendix: Estimation of Robustness by Eigenvalue Analysis

A.1 Overview

The robustness of the preconditioning method was estimated according to the eigenvalue distribution of the $[M]^{-1}[A]$ matrix by the method in [8], where [A] is the original coefficient matrix and $[M]^{-1}$ is the inverse of the preconditioning matrix.

In a symmetric positive definite matrix, the spectral condition number κ is given by $\kappa=E_{max}/E_{min}$ where E_{max} and E_{min} are the largest and smallest eigenvalues, respectively, of $[M]^{-1}[A]$ [8].

A.2 Simple Block Model

Benchmarks for the simple block model, as shown in Fig.23, have been conducted for a wide range of penalty parameter values using various types of preconditioners on a single processor (COMPAQ Alpha 21164-600MHz). In the benchmarks, the following model is considered:

- NX1=20, NX2=20, NY =15, NZ1= 20, NZ2= 20 (Fig.4.8(a))
- Total Elements = 24,000, Total Nodes = 27,888, Total DOF = 83,664.

Table A.1 shows the results for convergence. BIC(0) does not converge within 1,000 iterations if λ is larger than 10^4. BIC(1), BIC(2) and SB-BIC(0) provide robust convergence for a wide range of λ values. SB-BIC(0) provides the most efficient performance although the iteration number for convergence is larger than that for BIC(1) and BIC(2).

Table A.2 shows E_{min}, E_{max} and κ derived from each preconditioning method for a range of penalty parameter values. According to Table A.2, all of the eigenvalues are approximately constant and close to 1.00 for a wide range of λ values except for BIC(0). BIC(1) and BIC(2) provide a slightly smaller κ than SB-BIC(0).

Table A.1 Iterations/CPU time (including factorization) for convergence ($\varepsilon=10^{-8}$) on a single PE of COMPAQ Alpha 21164/600MHz by preconditioned CG for the 3D elastic contact problem for simple block model with MPC condition in Fig.23 (83,664DOF).: **BIC(n)**: Block IC with n-level fill-in, **SB-BIC(0)**: BIC(0) with the selective blocking reordering.

Preconditioning	λ	Iter #	sec.
BIC(0)	10^2	388	202.
	10^4	> 1000	N/A
BIC(1)	10^2	77	89.
	10^6	77	89.
BIC(2)	10^2	59	135.
	10^6	59	135.
SB-BIC(0)	10^2	114	61.
	10^6	114	61.

Table A.2 Largest and smallest eigenvalues (E_{min}, E_{max}) and $\kappa= E_{max}/E_{min}$ of $[M]^{-1}[A]$ for a wide range of penalty parameter values: 3D elastic contact problem for simple block model with MPC condition in Fig.23 (83,664DOF).

Preconditioning		$\lambda=10^2$	$\lambda=10^6$	$\lambda=10^{10}$
BIC(0)	E_{min}	4.845568E-03	4.865363E-07	4.865374E-11
	E_{max}	1.975620E+00	1.999998E+00	2.000000E+00
	κ	4.077170E+02	4.110686E+06	4.110681E+10
BIC(1)	E_{min}	8.901426E-01	8.890643E-01	8.890641E-01
	E_{max}	1.013930E+00	1.013863E+00	1.013863E+00
	κ	1.139065E+00	1.140371E+00	1.140371E+00
BIC(2)	E_{min}	9.003662E-01	8.992896E-01	8.992895E-01
	E_{max}	1.020256E+00	1.020144E+00	1.020144E+00
	κ	1.133157E+00	1.134388E+00	1.134389E+00
SB-BIC(0)	E_{min}	6.814392E-01	6.816873E-01	6.816873E-01
	E_{max}	1.005071E+00	1.005071E+00	1.005071E+00
	κ	1.474924E+00	1.474387E+00	1.474387E+00

A.3 Southwest Japan Model

Benchmarks of the Southwest Japan model with complicated geometry (23,831 elements, 27,195 nodes, 81,585 DOF), as shown in Fig.25, have been conducted for a wide range of penalty parameter values using various types of preconditioners on a single processor (COMPAQ Alpha 21164-600 MHz).

Table A.3 shows the results for convergence. BIC(0) does not converge within 1,000 iterations if λ is larger than 10^4. BIC(1), BIC(2) and SB-BIC(0) provide robust convergence for a wide range of λ values but the iteration number for convergence increases in BIC(1) and BIC(2) as λ changes from 10^2 to 10^4.(BIC(1) from 201 to 259, BIC(2) from 176 to 232). SB-BIC(0) provides the most efficient performance although the iteration number for convergence is larger than both BIC(1) and BIC(2).

Table A.4 shows E_{min}, E_{max} and κ derived from the eigenvalue analysis of $[M]^{-1}[A]$ for each preconditioning method. In SB-BIC(0), E_{min}, E_{max} and κ remain constant for a wide range of λ values but κ increases in BIC(1) and BIC(2) when λ changes from 10^2 to 10^4. This corresponds to the increase in the iteration number for convergence in BIC(1) and BIC(2) between $\lambda=10^2$ and $\lambda=10^4$. In this example, the geometry is much more complicated than in the previous simple block model. Moreover, meshes are not uniform and some of the meshes are distorted. The

distortion of an individual mesh directly affects the components of the coefficient matrix [A] for linear equations and the eigenvalue distribution of $[M]^{-1}[A]$. SB-BIC(0) is robust under such conditions.

Table A.3 Iterations/CPU time (including factorization) for convergence ($\varepsilon=10^{-8}$) on a single PE of COMPAQ Alpha 21164/600MHz by preconditioned CG for the 3D elastic contact problem for Southwest Japan model with MPC condition in Fig.25 (81,585DOF).: **BIC(n)**: Block IC with n-level fill-in, **SB-BIC(0)**: BIC(0) with the selective blocking reordering.

Preconditioning	λ	Iter #	sec.
BIC(0)	10^2	344	172.
	10^4	> 1000	N/A
BIC(1)	10^2	201	192.
	10^4	256	237.
	10^6	256	237.
BIC(2)	10^2	176	288.
	10^4	229	360.
	10^6	230	361.
SB-BIC(0)	10^2	297	149.
	10^4	295	148.
	10^6	295	148.

Table A.4 Largest and smallest eigenvalues (E_{min}, E_{max}) and $\kappa= E_{max}/E_{min}$ of $[M]^{-1}[A]$ for a wide range of penalty parameter values: 3D elastic contact problem for Southwest Japan model with MPC condition in Fig.25 (81,585DOF).

Preconditioning		$\lambda=10^2$	$\lambda=10^4$	$\lambda=10^6$	$\lambda=10^{10}$
BIC(0)	E_{min}	1.970395E-02	1.999700E-04	1.999997E-06	2.000000E-10
	E_{max}	1.005194E+00	1.005194E+00	1.005194E+00	1.005194E+00
	κ	5.101486E+01	5.026725E+03	5.025979E+05	5.025971E+09
BIC(1)	E_{min}	3.351178E-01	2.294832E-01	2.286390E-01	2.286306E-01
	E_{max}	1.142246E+00	1.142041E+00	1.142039E+00	1.142039E+00
	κ	3.408491E+00	4.976580E+00	4.994944E+00	4.995128E+00
BIC(2)	E_{min}	3.558432E-01	2.364909E-01	2.346180E-01	2.345990E-01
	E_{max}	1.058883E+00	1.088397E+00	1.089189E+00	1.089196E+00
	κ	2.975702E+00	4.602277E+00	4.642391E+00	4.642800E+00
SB-BIC(0)	E_{min}	2.380572E-01	2.506369E-01	2.507947E-01	2.507963E-01
	E_{max}	1.005194E+00	1.005455E+00	1.005465E+00	1.005466E+00
	κ	4.222491E+00	4.011600E+00	4.009117E+00	4.009092E+00

A.4 Remarks

In both models (simple block and Southwest Japan), the spectral condition number of $[M]^{-1}[A]$ is a helpful parameter for the evaluation of the convergence of the preconditioning methods. In the simple block model, the spectral condition number of $[M]^{-1}[A]$ by BIC(1) and BIC(2) is usually smaller than that of SB-BIC(0) and the iteration number for convergence is smaller (Tables A.1 and A.2). In contrast, the Southwest Japan model provides a larger spectral condition number for BIC(1) and BIC(2) than in SB-BIC(0) when λ is larger than 10^4, but the iteration number for the convergence of BIC(1) and BIC(2) is smaller than that for SB-BIC(0) (Tables A.3 and A.4).