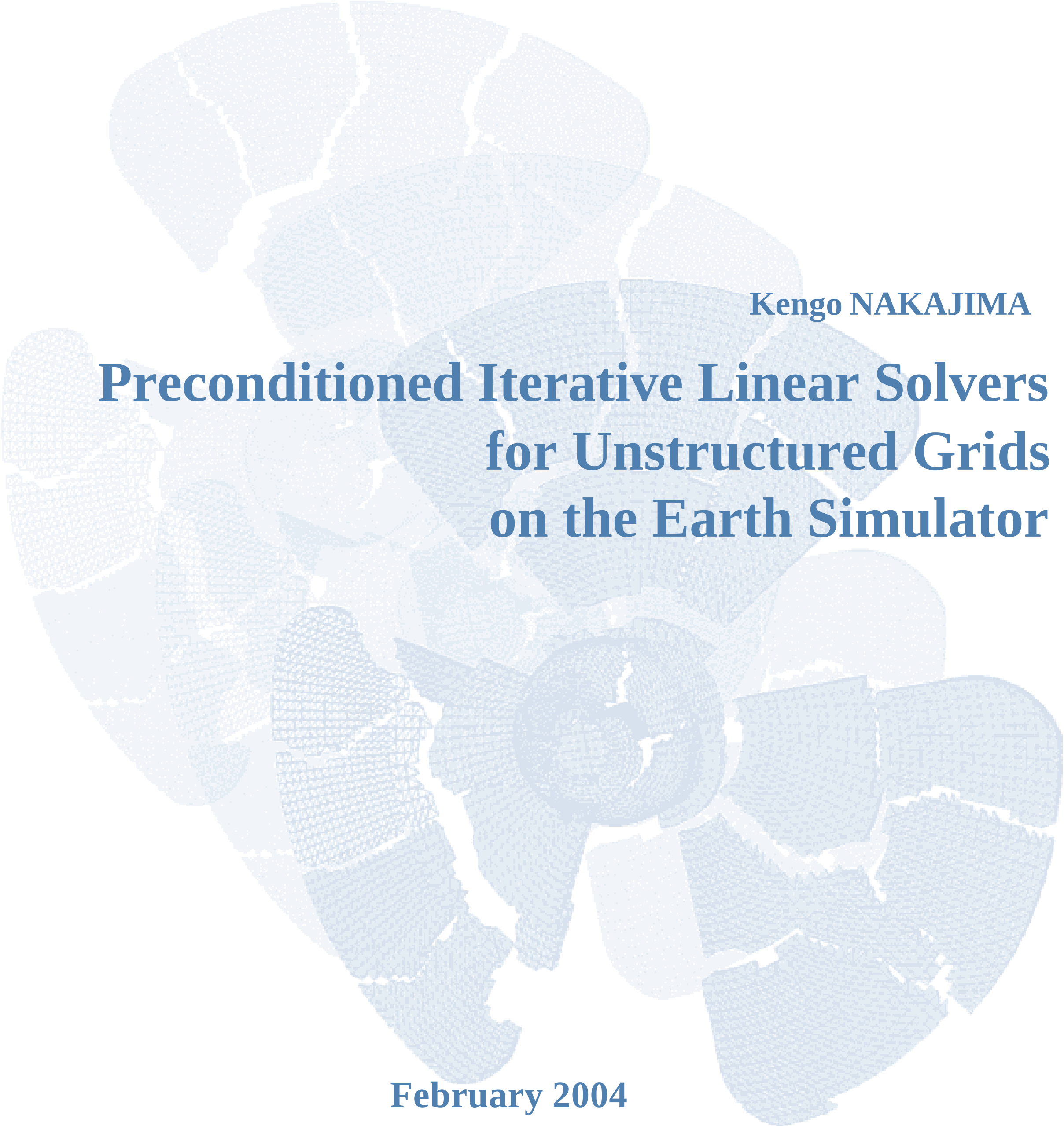
HPC-MW

MiddleWare for High-Performance Computing

Technical Report HPC-MW 004-001

Research Organization for Information Science & Technology

Institute of Industrial Science, the University of Tokyo Frontier Simulation Software for Industrial Science



Kengo NAKAJIMA

Preconditioned Iterative Linear Solvers for Unstructured Grids on the Earth Simulator

February 2004

HPC - MW

MiddleWare for High-Performance Computing

Technical Report HPC-MW 004-001

Research Organization for Information Science & Technology
Institute of Industrial Science, the University of Tokyo Frontier Simulation Software for Industrial Science

Kengo NAKAJIMA

Preconditioned Iterative Linear Solvers for Unstructured Grids on the Earth Simulator

February 2004

HPC-MW 2004-001

TITLE

Preconditioned Iterative Linear Solvers for Unstructured Grids on the Earth Simulator

AUTHORS

Kengo Nakajima (RIST)

CONTACT PERSON

Kengo Nakajima (nakajima@tokyo.rist.or.jp)

KEY WORDS AND PHRASES

Earth Simulator, SMP Cluster, Hybrid, Flat MPI, Unstructured Grid, Preconditioned Iterative Solvers

ABSTRACT

Efficient parallel preconditioned iterative linear solvers for unstructured grid have been developed for symmetric multiprocessor (SMP) cluster architectures with vector processors such as the Earth Simulator. Three types of preconditioning methods (ICCG, multigrid and selective-blocking for contact problems) have been developed and performance has been demonstrated on the Earth Simulator using flat-MPI and hybrid parallel programming models. Simple 3D linear elastic problems with more than 2.2x10^9 DOF have been solved using 3x3 block ICCG(0) method and PDJDS/CM-RCM reordering on 176 nodes of the Earth Simulator, achieving performance of 3.80 TFLOPS. Multicolor and RCM ordering provide excellent parallel and vector performance of the three preconditioned methods on the Earth Simulator.

REPORT DATE	**TOTAL NO. OF PAGES**
February 7th, 2004.	8

ANY OTHER IDENTIFYING INFORMATION OF THIS REPORT

submitted to HPC Asia 2004,
7th International Conference on
High Performance Computing and Grid in Asia Pacific Region
Omiya Sonic City, Tokyo Area, Japan
July 20-22, 2004.

Research Organization for Information Science & Technology (RIST)
2-2-54, Nakameguro, Meguro-ku Tokyo, 153-0061, Japan
TEL: +81-3-3712-5321 FAX: +81-3-3712-5552
http://geofem.tokyo.rist.or.jp/

Preconditioned Iterative Linear Solvers for Unstructured Grids on the Earth Simulator

Kengo Nakajima
Research Organization for Information Science and Technology (RIST)
nakajima@tokyo.rist.or.jp

Abstract

Efficient parallel preconditioned iterative linear solvers for unstructured grid have been developed for symmetric multiprocessor (SMP) cluster architectures with vector processors such as the Earth Simulator. Three types of preconditioning methods (ICCG, multigrid and selective-blocking for contact problems) have been developed and performance has been demonstrated on the Earth Simulator using flat-MPI and hybrid parallel programming models. Simple 3D linear elastic problems with more than 2.2×10^9 DOF have been solved using 3×3 block ICCG(0) method and PDJDS/CM-RCM reordering on 176 nodes of the Earth Simulator, achieving performance of 3.80 TFLOPS. Multicolor and RCM ordering provide excellent parallel and vector performance of the three preconditioned methods on the Earth Simulator.

1. Introduction

1.1. Preconditioned iterative methods

Solving linear equations with sparse coefficient matrices is a very important and expensive process in various types of scientific computing. Sparse linear solvers can be broadly classified as being either *direct* or *iterative*.

Direct solvers such as Gaussian Elimination are extremely robust. However, their memory requirements grow as a nonlinear function of the matrix size.

In contrast, iterative methods are memory scalable. Therefore they are the only choice for large-scale simulations by massively parallel computers. The rate of convergence of iterative methods strongly depends on the spectrum of the coefficient matrix. Hence, iterative methods usually involve a second matrix that transforms the coefficient matrix into a matrix with more favorable spectrum. The transformation matrix is called a *preconditioner*. According to [1], preconditioners can be divided roughly into following three categories:

I. Preconditioners designed for wide range of general classes of matrices (*e.g.* Jacobi, Gauss-Seidel, SOR, IC/ILU, and approximate inverse methods).
II. Preconditioners designed for broad classes of underlying problems such as elliptic PDE's (*e.g.* Multigrid and domain decomposition preconditioners).
III. Preconditioners designed for a specific matrix or underlying problem.

1.2. Present Work

In the present work, parallel iterative solvers with preconditioning for various types of applications on unstructured grids have been developed for the Earth Simulator. Following three types of preconditioners, each of which corresponds to each of the three categories of preconditioners in Section 1.1, have been considered:

I. Localized block ILU(0) preconditioning method for 3D solid mechanics.
II. Parallel scalable multigrid preconditioning method for 3D Poisson equations derived from incompressible Navier-Stokes solvers with adaptive meshes.
III. Selective blocking preconditioning method for 3D solid mechanics with contact conditions.

Local data structure in [2] and [3] has been applied. In this work, symmetric positive definite matrices are mainly treated which are derived from descritization of certain problems. Therefore, the Conjugate Gradient (CG) method is mainly used as iterative method [4].

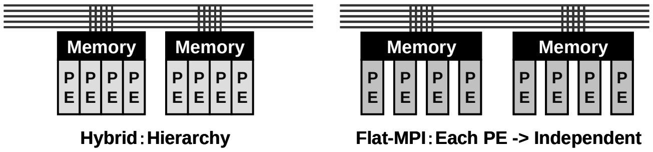
2. Reordering methods for parallel/vector performance on the Earth Simulator

2.1. Parallel programming models on SMP cluster architectures

Recently, symmetric multiprocessor (SMP) cluster architectures have become very popular as teraflop-scale parallel computers, such as the Accelerated Strategic Computing Initiative (ASCI) [5] machines and the Earth Simulator [6].

In order to achieve minimal parallelization overhead, a multi-level hybrid programming model [7-10] is often employed for SMP cluster architectures (Figure 1). 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, and fine-grain parallelism is obtained by loop-level parallelism inside each SMP node by compiler-based thread parallelization such as OpenMP.

Another often-used programming model is the single-level flat-MPI model [7-10] (Figure 1), in which separate single-threaded MPI processes are executed on each processing element (PE).



	Each PE	Intra NODE	Inter NODE
Hybrid	F90 + directives (OpenMP)		MPI
Flat-MPI	F90	MPI	

Figure 1. Parallel programming models for SMP cluster architectures [7-11]

The advantage of a hybrid programming model over flat-MPI is that there is no message-passing overhead in each SMP node. However, a hybrid approach usually requires more complex programming.

Although a significant amount of research on this issue has been conducted in recent years [7-11], 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 [7-11], although the efficiency depends on hardware performance (CPU speed, communication bandwidth, memory bandwidth), features of applications, and problem size [12].

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

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 [2,3,11], and each partition is assigned to one SMP node. On the contrast, each partition corresponds to each PE in the flat-MPI.

In this study, various types of preconditioners are ported to iterative solvers using both of flat-MPI and hybrid parallel programming models on the Earth Simulator.

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

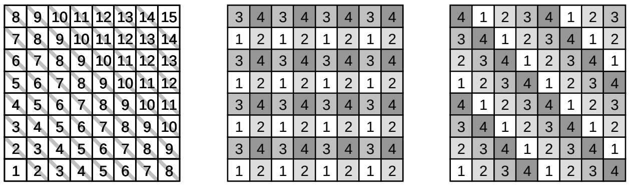
For unstructured grids, in which data and memory access patterns are very irregular, the reordering technique is very effective in achieving highly parallel and vector performance. In this study, a special reordering technique proposed by Washio et. al. [13,14] has been integrated with preconditioning methods and parallel iterative solvers.

2.2 Multicolor reordering

The popular reordering methods are reverse Cuthil-McKee reordering and multicolor reordering [15]. The reverse Cuthil-McKee (RCM) method (Figure 2(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 (Figure 2(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 [16], CM-RCM reordering (Figure 2 (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.



8	9	10	11	12	13	14	15
7	8	9	10	11	12	13	14
6	7	8	9	10	11	12	13
5	6	7	8	9	10	11	12
4	5	6	7	8	9	10	11
3	4	5	6	7	8	9	10
2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8

(a) RCM

3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2

(b) MC: 4 colors

4	1	2	3	4	1	2	3
3	4	1	2	3	4	1	2
2	3	4	1	2	3	4	1
1	2	3	4	1	2	3	4
4	1	2	3	4	1	2	3
3	4	1	2	3	4	1	2
2	3	4	1	2	3	4	1
1	2	3	4	1	2	3	4

(c) CM-RCM

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

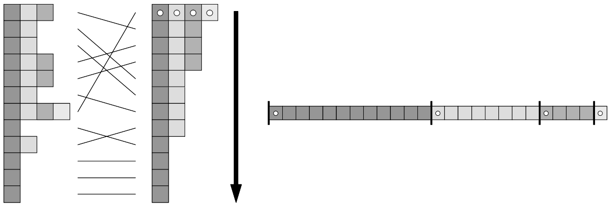
2.3 DJDS Reordering

The compressed row storage (CRS) [4,15] format originally used in GeoFEM is highly memory-efficient, however the innermost loop is relatively short due to matrix-vector operations. Loop exchange is then effective for obtaining a sufficiently long innermost loop for vector operations.

Descending-order jagged diagonal storage (DJDS) [11,13,14] is suitable for this type of operation and involves permuting rows into an order of decreasing number of non-zeros, as in Figure 3(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 Figure 3(b).

2.4. 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) [11].



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

(b)1D array of matrix coefficient

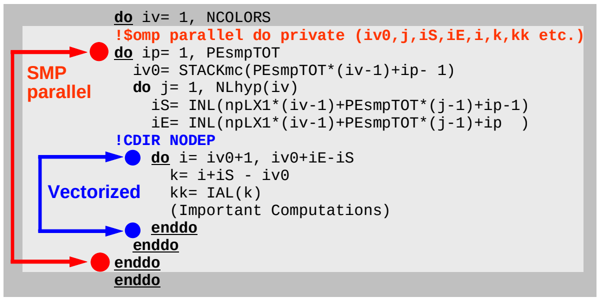
Figure 3. DJDS reordering for efficient vector/parallel processing

2.5 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:

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

Figure 4 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
```

Figure 4. Forward/backward substitution procedure using OpenMP and vectorization directives during ILU(0)/IC(0) preconditioning by PDJDS/MC reordering.

3. ICCG solvers for linear-elastic problems (Category-I)

The proposed reordering method is applied to simple applications in 3D solid mechanics, as described in Figure 5, which represent linear elastic problems with homogeneous isotropic material properties and boundary conditions.

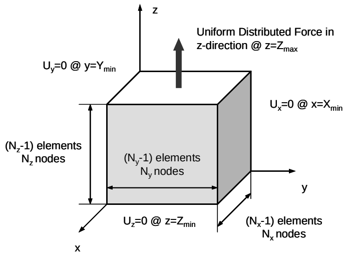

z
Uniform Distributed Force in z-direction @ z=Z_{max}
U_y=0 @ y=Y_{min}
U_x=0 @ x=X_{min}
(N_z-1) elements N_z nodes
(N_y-1) elements N_y nodes
y
U_z=0 @ z=Z_{min}
(N_x-1) elements N_x nodes
x

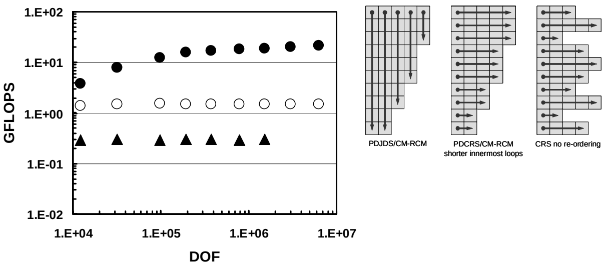
Figure 5. Example for 3D solid mechanics

Figure 6 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 (Figure 6):

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

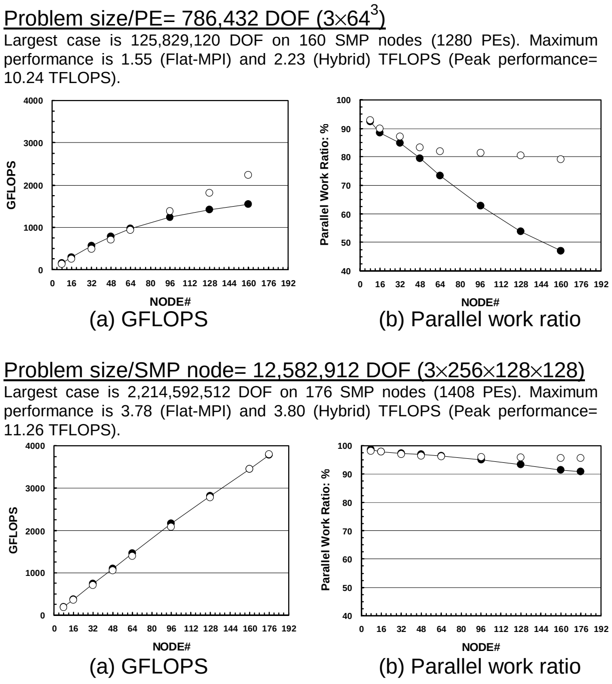
PDCRS/CM-RCM reordering is identical to PDJDS/CM-RCM except that the matrices are stored in a CRS manner. The length of the innermost loop is shorter than that for PDJDS. The elapsed execution time was measured for various problem sizes from 3×16^3 (12,288) DOF to 3×128^3 (6,291,456) DOF on a single SMP node of the Earth Simulator (8 PEs, 64 GFLOPS peak performance). Color number for CM (cyclic multicoloring) has been fixed as 99. PDJDS outperforms PDCRS for larger problems due to larger length of inner-most loops. On the Earth Simulator, 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 cases without reordering exhibit very poor performance of only 0.30 GFLOPS (0.47% of peak performance).

Figure 7 shows the results for large-scale problems in Figure 5 implemented up to 176 SMP nodes of the Earth Simulator (1,408 PEs, 11.26 TFLOPS peak performance). 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 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.



(BLACK Circles: PDJDS/CM-RCM, WHITE Circles: PDCRS/CM-RCM, BLACK Triangles: CRS no reordering).

Figure 6. Effect of coefficient matrix storage method and reordering for the 3D linear elastic problem with various problem sizes on the Earth Simulator with a single SMP node.



Problem size/PE= 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).

(a) GFLOPS (b) Parallel work ratio

Problem size/SMP node= 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).

(a) GFLOPS (b) Parallel work ratio

Figure 7. Problem size and parallel performance on the Earth Simulator for the 3D linear elastic problem using between 8 and 176 SMP nodes. (BLACK: Flat-MPI, WHITE: Hybrid).

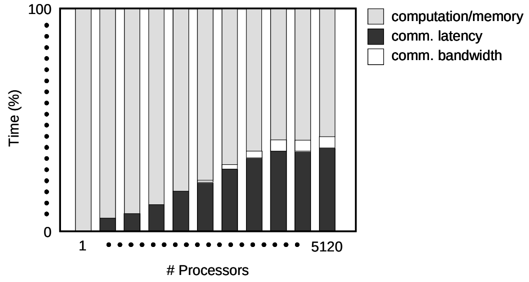


Figure 8. Performance estimation of a FVM code on the Earth Simulator based on the results described in [17].

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. In general, flat-MPI performs better than hybrid model for smaller numbers of SMP nodes, while the hybrid outperforms flat-MPI when a large number of SMP nodes are involved, especially if the problem size per node is small as shown in Figure 7. This is mainly because of the latency overhead for MPI communication. According to the performance estimation for finite-volume application in [17], a greater percentage of time is required by the latency component on larger processor counts, simply due to the available bandwidth being much larger (Figure 8). 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.

4. Multigrid preconditioning for Poisson equations (Category-II)

Multigrid is an example of scalable linear solver. Relaxation methods such as Gauss-Seidel can efficiently damp high-frequency error but low-frequency error is left. The multigrid idea is to recognize that this low-frequency error can be accurately and efficiently solved for on a coarser grid.

Multigrid algorithms tend to be problem specific and less robust than preconditioned Krylov iterative methods such as IC/ILU methods. Fortunately it is easy to combine the best features of multigrid and Krylov iterative methods into one algorithm, multigrid preconditioned Krylov iterative methods. The resulting algorithm is robust, efficient and scalable.

There are two basic multigrid approaches according to the methods for constructing coarse grids: *geometric* and *algebraic*. In geometric multigrid, the geometry of the problem is used to define the various multigrid components. On the contrary, algebraic multigrid methods use only the information available in the linear system of equations such as matrix connectivity.

The author developed a multigrid-preconditioned conjugate gradient iterative method (MGCG) for Poisson equations [18]. Target application was 3D incompressible thermal convection in the region between dual sphere surfaces. This type of geometry appears often in the simulations of earth sciences for both fluid earth (atmosphere and ocean) and solid earth (mantle and outer core). Solving Poisson equations for pressure-correction [18] is the most expensive process in the entire computation and convergence acceleration of this process is very critical issue for the total speed of the method. In this work, this procedure has been implemented to the Earth Simulator.

Semi-unstrucred prismatic grids generated from triangles on sphere surface are used. Meshes start from icosahedron and are globally refined recursively as in Figure 9. The grid hierarchy due to recursive refinement can be utilized for the coarse grid formation (Figure 10). In the current application, pressure and potential for pressure correction are defined at cell center. Therefore dependent variables are defined at cell center in this study.

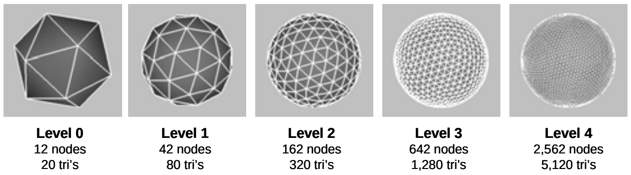


Figure 9. Surface triangle meshes generated from icosahedron

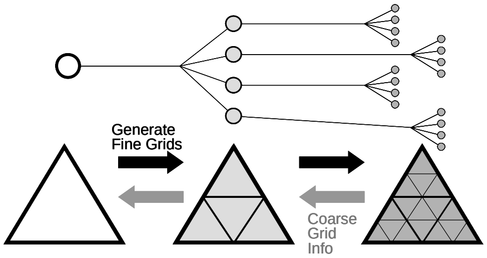


Figure 10. Grid hierarchy obtained by successive refinement

Features of the parallel MGCG (Multigrid preconditioned Conjugate Gradient) procedure are summarized as follows :

- V-cycle multigrid Preconditioned MGCG solver
- Gauss-Seidel smoothing.
- Semi-coarsening in lateral and *normal-to-surface* direction [18].
- Entire region is partitioned along radial (*normal-to-surface*) direction.
- Multilevel communication tables are defined at partition interfaces [18].

Most of the MGCG procedure is spent for smoothing process (more than 95%), therefore optimization of this process is the most critical issue. Multicolor ordering described in Chapter 2 has been applied to Gauss-Seidel smother, thus localized operations with continuous memory access and sufficiently long loops, which are suitable for vector processors have been achieved.

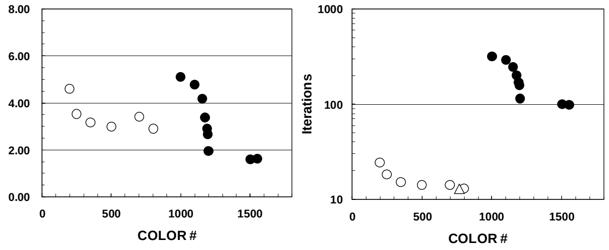
In the examples, problem size for 1 processor is fixed as 1,280 triangles (Level=3 in Figure 9) × 600 (layers) = 768,000 (cells) and computations using from 1 PE to 80 PEs have been tested. This corresponds to 768,000 to 61,440,000 cells. *Uniform* boundary condition described in [18] has been applied.

Figure 11 shows effect of multicolor ordering on computation time and iterations of Poisson solvers using a single CPU of the Earth Simulator. Ordering by many colors provides improvement of performance due to smaller number of iterations. It took more than 300 sec. for convergence by the original MGCG solvers [18] on the Earth Simulator, but optimized solver with multicolor ordering converges in 3 sec. (Figure 11 (a)).

Figure 12 shows GFLOPS rate according to color numbers. Both methods attained more than 20% of the peak performance (8 GFLOPS). Ordering with many colors provides faster convergence, but loop length is shorter. Therefore, GFLOPS rate becomes smaller for large number of colors. Figure 12 shows this feature, but computation time is shorter because iterations for convergence has been smaller, as shown in Figure 11. GFLOPS rate of MGCG is generally smaller than that of ICCG, because of shorter loop length in coarse level grids.

Figure 13 shows computation time up to 80 PEs of the Earth Simulator of MGCG (color number= 800) and ICCG (color number= 1500) solvers using flat-MPI parallel programming model. Problem size for 1 processor is fixed as 768,000 (cells) and maximum problem size on 80 PEs is 61,440,000 (cells). Computation time remains almost constant in MGCG cases when the number of PE is small but tends to increase along with the number of PEs. This is because that effect of Gauss-Seidel smoothing becomes localized and close to Jacobi smoothing as the PE number increases. But even in the 80 PE cases, MGCG is much faster than ICCG.

Figure 14 shows comparison between flat-MPI and *hybrid* parallel programming model using a single SMP node (8 PEs) of the Earth Simulator and Hitachi SR8000 in University of Tokyo [19] according to elapsed time for 10 MGCG cycles. 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. Moreover, the hybrid parallel programming model is much more sensitive to color number and innermost vector length than flat-MPI.



(a) Computation Time (b) Iterations

Figure 11. Effect of multicoloring on computation time and iterations of Poisson solver by ICCG and MGCG. Single CPU of the Earth Simulator. 768,000 cells. (BLACK: ICCG , WHITE: MGCG)

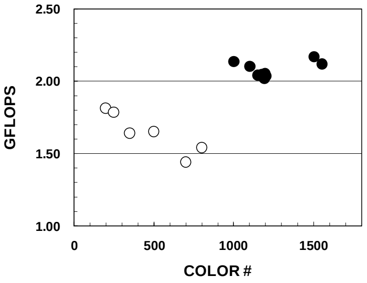


Figure 12. Effect of multicoloring on GFLOPS rate of Poisson solver by ICCG and MGCG. Single CPU of the Earth Simulator (peak performance: 8 GFLOPS). 768,000 cells. (BLACK: ICCG , WHITE: MGCG)

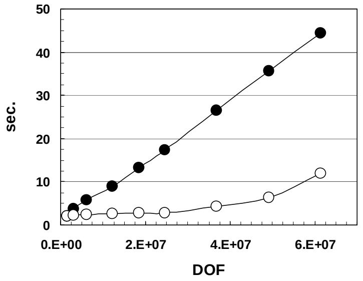
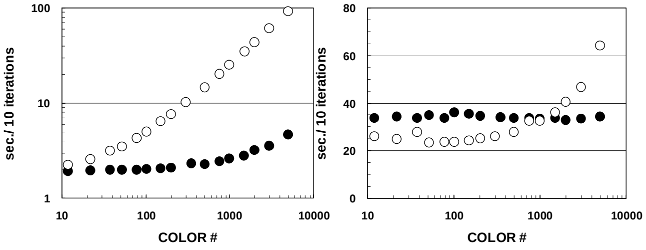


Figure 13. Elapsed computation time including communication by Earth Simulator for fixed problem size on each processor. 1280×600=768,000 cells/PE, 1-80 PEs (up to 61,440,000 cells). BLACK: ICCG(1500 colors), WHTE: MGCG(800 colors)).



(a) Earth Simulator (b) Hitachi SR8000

Figure 14. Elapsed time for 10 MGCG cycles on 1 SMP node (8 PEs) on the Earth Simulator and Hitachi SR8000. 6,144,000 cells. (BLACK: Flat-MPI, WHTE: Hybrid).

5. Selective-blocking preconditioning for contact problems (Category-III)

Final example is preconditioning method for contact problems. Boundary nonlinearity due to contact is very critical in wide range of scientific and engineering applications. For example, stress accumulation on plate boundaries (faults) is very important in estimating the earthquake generation cycle (Figure 15). In this type of applications, a large penalty number λ is usually introduced for constraint conditions around contact surfaces [20]. A large λ can provide an accurate solution, but the condition number of the coefficient matrices is large. Therefore, many iterations are required for convergence of iterative solvers [20].

In previous work [21], the author developed a robust preconditioning method, called *selective blocking*, for the simulation of contact with penalty constraints using parallel computers (Figure 16). 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 (Table 1).

In the present work, the reordering procedure described in Chapter 2 has been implemented to the selective-blocking preconditioning method. Special ordering method for contact problems has been developed.

Example tests using 10 SMP nodes of the Earth Simulator have been done for the 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 17 shows 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 163.4 GFLOPS (25.5% of peak performance, 640 GFLOPS) for the Southwest Japan has been obtained. As for the flat-MPI, the performance was 190.4 GFLOPS (29.8%) for the Southwest Japan. These results are as excellent as those in Figure 7 by ICCG solvers for simple geometries with homogeneous boundary conditions, shown in Chapter 3. 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.

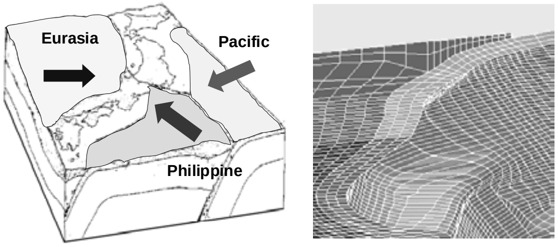


Figure 15. Plate boundaries (faults) around Japanese Islands and finite-element model

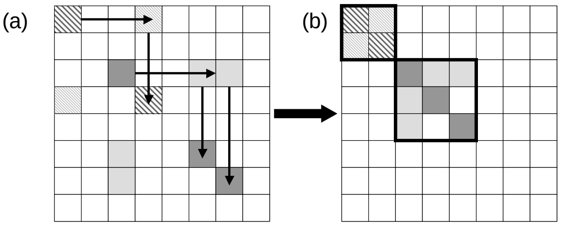
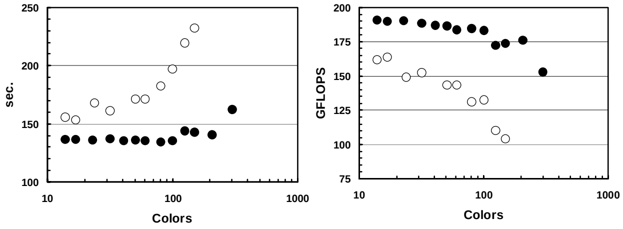


Figure 16. Procedure of the *selective blocking* : Strongly coupled elements are put into the same *selective block*. (a) searching for strongly coupled components and (b) reordering and selective blocking.

Table 1. 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 [20,21] (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+Solve (sec.)	Memory (MB)
BIC(0)	10^6	2590	252.3	59
BIC(1)	10^6	78	20.3	176
BIC(2)	10^6	59	30.8	319
SB-BIC(0)	10^6	114	13.0	67



(a)Elapsed time for the linear solver (b) GFLOPS

Figure 17. Performance of 10 SMP nodes of the Earth Simulator (peak performance= 640 GFLOPS) using SB-BIC(0) CG for the 3D elastic contact problem with MPC condition ($\lambda=10^6$) (Southwest Japan model, 23,301,006 DOF). (BLACK: Flat-MPI, WHITE: Hybrid)

6. Concluding Remarks

Parallel iterative linear solvers for unstructured grids with three different types of preconditioning methods have been developed on the Earth Simulator. In all cases, multicolor and RCM ordering techniques provide excellent parallel and vector performance on the Earth Simulator.

Flat-MPI and hybrid parallel programming models are competitive in most cases. Hybrid outperforms Flat-MPI when number of SMP node is large and problem size is not so large. This is because of the effect of communication latency in many MPI processes.

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.

Acknowledgements

This study is supported by “Solid Earth Platform for Large-Scale Computation” project funded by the Ministry of Education, Culture, Sports, Science and Technology, Japan (MEXT) through “Special Promoting Funds of Science & Technology” and “Frontier Simulation Software for Industrial Science” in the “IT Program” by MEXT.

References

[1] A.Greenbaum: *Iterative Methods for Solving Linear Systems*, SIAM, 1997.
[2] GeoFEM Web Site: http://geofem.tokyo.rist.or.jp/
[3] HPC-MW Web Site: http://hpcmw.tokyo.rist.or.jp
[4] R. Barrett, M. Berry, T.F. Chan, J.W. Demmel, J. Donato, J.J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H. van der Horst: *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, 1994.
[5] Accelerated Strategic Computing Initiative (ASCI) Web Site : http://www.llnl.gov/asci/
[6] Earth Simulator Center Web Site: http://www.es.jamstec.go.jp/
[7] Cappelo, F. and Etiemble, D. :"MPI versus MPI+OpenMP on the IBM SP for the NAS Benchmarks", *SC2000 Technical Paper*, Dallas, Texas, 2000.
[8] Djomehri, M.J. and Jin, H.H.: "Hybrid MPI+OpenMP Programming of an Overset CFD Solver and Performance Investigations", *NASA/NAS Technical Report* NAS-02-002, (2002). 16.
[9] Falgout, R. and Jones, J. : "Multigrid on Massively Parallel Architectures", *Sixth European Multigrid Conference*, Ghent, Belgium, September 27-30, 1999.
[10] 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.
[11] Nakajima, K.: "OpenMP/MPI Hybrid vs. Flat MPI on the Earth Simulator: Parallel Iterative Solvers for Finite Element Method", *WOMPEI 2003*, Tokyo, Japan, *Lecture Notes in Computer Science 2858*, pp.486-499, 2003.
[12] Rabenseifner, R.: "Communication Bandwidth of Parallel Programming Models on Hybrid Architectures", *WOMPEI 2002*, Lecture Notes in Computer Science 2327(2002), pp.437-448.
[13] 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.
[14] Washio, T., Maruyama, K., Osoda, T., Shimizu, F. and Doi, S. : "Efficient implementations of block sparse matrix operations on shared memory vector machines", *SNA2000*, 2000.
[15] Saad, Y. : *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.
[16] 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.
[17] 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*, 2002.
[18] K. Nakajima, “Parallel Multilevel Iterative Linear Solvers with Unstructured Adaptive Grids for Simulations in Earth Science”, *Concurrency and Computation: Practice and Experience*. pp.484-498, Vol.14, No.6-7, 2002.
[19] Information Technology Center, The University of Tokyo Web Site: http://www.cc.u-tokyo.ac.jp/
[20] Iizuka, M., Okuda, H. and Yagawa, G.: "Nonlinear Structural Subsystem of GeoFEM for Fault Zone Analysis", *Pure and Applied Geophysics*, Vol.157 (2000), pp.2105-2124.
[21] Nakajima, K. and Okuda, H. "Parallel Iterative Solvers with Selective Blocking Preconditioning for Simulations of Fault Zone Contact", *Preconditioning 2001*, USA, 2001, *Journal of Numerical Algebra with Applications (in press)*.