

# Parallel Multistage Preconditioners Based on a Hierarchical Graph Decomposition for SMP Cluster Architectures with a Hybrid Parallel Programming Model

Kengo Nakajima

Department of Earth and Planetary Science, The University of Tokyo  
7-3-1 Hongo, Bunkyo-ku, Tokyo 112-0002, Japan  
nakajima@eps.s.u-tokyo.ac.jp

**Abstract.** In this work, the Parallel Hierarchical Interface Decomposition Algorithm (PHIDAL) and a hybrid parallel programming model were applied to finite-element based simulations of linear elasticity problems in media with heterogeneous material properties using parallel preconditioned iterative solvers. Reverse Cuthill-McKee reordering with cyclic multicoloring (CM-RCM) was applied for parallelism on each SMP node through OpenMP. The developed code has been tested on the IBM p5-575 and the Tsubame Grid Cluster using up to 512 cores. Preconditioners based on PHIDAL provide a superior scalable performance and robustness on both architectures in comparison to conventional block Jacobi-type localized preconditioners.

## 1 Introduction

### 1.1 Parallel Programming Models on SMP Cluster Architectures

In order to achieve minimal parallelization overheads on SMP (symmetric multiprocessors) clusters, a multi-level *hybrid* parallel programming model is often employed (Fig. 1). In this method, coarse-grained parallelism is achieved through domain decomposition by message passing among SMP nodes, while fine-grained parallelism is obtained via loop-level parallelism inside each SMP node using compiler-based thread parallelization techniques, such as OpenMP [1,2]. Another often-used programming model is the single-level *flat MPI* model (Fig. 1), in which separate single-threaded MPI processes are executed on each processing element (PE) [1,2]. The efficiency of each model depends on hardware performance (CPU speed, communication bandwidth, memory bandwidth, and the balance between these), application features, and problem size [3].

In previous works [1,2], the author developed an efficient parallel iterative solver for finite-element applications on the Earth Simulator (ES) [4] using a three-level hybrid parallel programming model with MPI, OpenMP and vectorization. Multicolor-based reordering methods were applied to distributed data sets on each SMP node in order to achieve an optimum parallel performance of Krylov iterative solvers with ILU/IC (Incomplete LU/Cholesky Factorization) preconditioners. The developed



Fig. 1. Parallel programming models on SMP cluster architecture [1,2]

method attained 3.8 TFLOPS with 176 SMP nodes of ES, corresponding to more than 33% of the peak performance (11.3 TFLOPS) [1]. Because of the relatively high sustained memory bandwidth of ES, flat MPI and hybrid parallel programming models are competitive, but the *hybrid* model outperforms *flat MPI*, if the number of processors exceeds 1,000. In general, *hybrid* parallel programming models provide a better performance the larger the number of nodes.

For cases using many colors, fewer numbers of iterations are required for convergence, because there are fewer incompatible nodes [5]. However, performance declines for these cases due to the smaller loop length and greater overhead on vector processors. For the ES, the hybrid parallel programming model was found to be very sensitive to the number of colors.

### 1.2 Parallel Preconditioned Iterative Solvers

Block Jacobi-type localized preconditioners are widely used for parallel iterative solvers [1,2]. They provide excellent parallel performance for well-defined problems, although the number of iterations required for convergence gradually increases according to the number of processors. However, this preconditioning technique is not robust for ill-conditioned problems with many processors, because it ignores the global effect of external nodes in other domains [1,2]. The generally used remedy is the extension of overlapped elements between domains [6,7]. Usually, this approach reduces the number of iterations required for convergence, but at the expense of requiring additional computations and communications, and so the final elapsed time of the computation is not necessarily reduced.

Table 1 shows the convergence of a parallel iterative solver with block Jacobi-type localized preconditioners for 3D linear elasticity problems in media with heterogeneous material properties on AMD Opteron clusters with 64 cores. In these computations the GPBi-CG (Generalized Product-type methods based on Bi-CG) solver [8]

**Table 1.** Convergence of parallel iterative solver using a block Jacobi-type localized preconditioner (GPBi-CG solver with localized SGS preconditioner) applied to 3D linear elasticity problems in media with heterogeneous material properties with 1,000,000 elements (3,090,903 DOF). AMD Opteron cluster with 64 cores

Depth of Overlap	Iterations for Convergence	sec.	Average Size per Core	
			Communication Table	Non-Zero Components in Coef. Matrix
0	1004	94.0	3,474	410,009
1	940	99.9	3,474	468,015
2	909	115.5	7,116	496,670

with localized SGS (Symmetric Gauss-Seidel) preconditioner [1,2] has been applied. The minimum and maximum values of Young’s modulus are  $10^{-3}$  and  $10^3$ , respectively, with an average value of 1.0. It may be seen that the extension of overlapped elements provides better convergence, but at the expense of increasing the elapsed computation time somewhat. *Selective-overlapping* [7] may improve this situation, but the benefits of developing more general preconditioners for scalable and robust computations are clear.

### 1.3 Overview of PHIDAL

The Parallel Hierarchical Interface Decomposition Algorithm (PHIDAL) algorithm provides robustness and scalability for parallel ILU/IC preconditioners [9]. PHIDAL is based on defining “*hierarchical interface decomposition (HID)*”. The HID process starts with a partitioning of the graph, with one layer of overlap. The “stages” are defined from this partitioning, with each stage consisting of a set of vertex groups. Each vertex group of a given stage is a *separator* for vertex groups of a *lower* stage. The incomplete factorization process proceeds by “stage” from lowest to highest. Due to the separation property of the vertex groups at different stages, this process can be carried out in a highly parallel manner. In [9], the concept of *connectors* (small connected sub-graphs) of different *levels* and *keys* are introduced for the purposes of applying this idea to general graphs as follows:

- Connectors of *level-1* ( $C^1$ ) are the sets of interior points. Each set of interior points is called a *sub-domain*.
- A connector of *level-k* ( $C^k$ ) ( $k > 1$ ) is adjacent to  $k$  *sub-domains*
- No  $C^k$  is adjacent to any other connector of level- $k$
- $Key(u)$  is the set of sub-domains (connectors of level-1,  $C^1$ ) connected to vertex  $u$ .

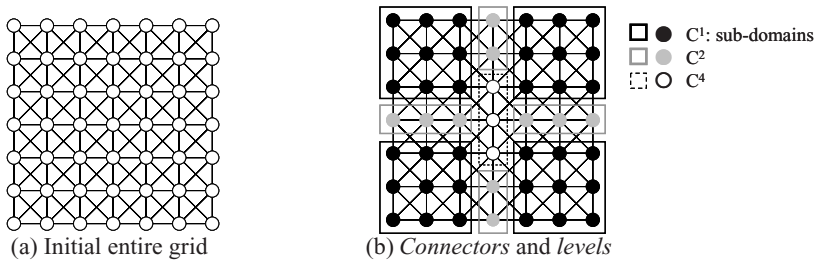


Fig. 2. Partitioning of a 9-point grid into 4 sub-domains

Fig.2 shows the example of the partition of a 9-point grid into 4 domains. In this case, there are 4 connectors of level-1 ( $C^1$ , sub-domain), 4 connectors of level-2 ( $C^2$ ) and 1 connector of level-4 ( $C^4$ ). Note that different connectors of the same level are not connected directly, but are separated by connectors of higher levels. These properties provide the block structure of the coefficient matrix  $A$  through reordering the unknowns by this decomposition. If the unknowns are reordered according to their level numbers, from the lowest to highest, the block structure of the reordered matrix would

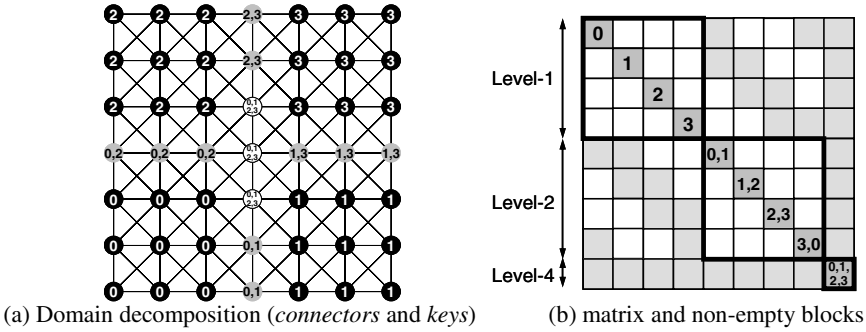


Fig. 3. Domain/block decomposition of the matrix according to the HID reordering

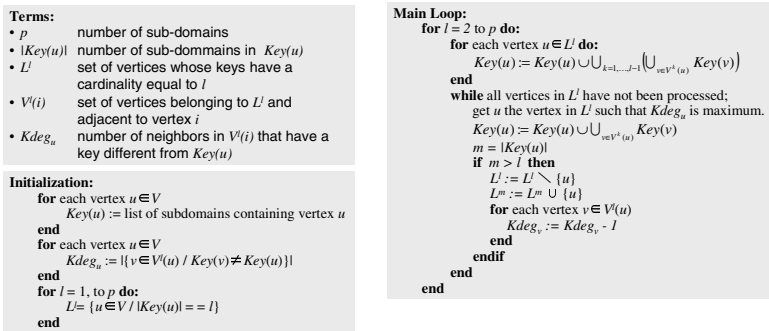


Fig. 4. Algorithms for HID processes [9]

be as shown in Fig.3. This block structure leads to natural parallelism if ILU/IC decompositions or forward/backward substitution processes are applied. Fig.4 provides algorithms for constructing independent connectors [9]. Thus, PHIDAL-based ILU/IC preconditioners can consider the global effect of external domains in parallel computations, and are expected to be more robust than block Jacobi-type localized ones.

### 1.4 Overview of the Present Work

In the present work, a hybrid parallel programming model has been applied to finite-element based simulations of 3D linear elasticity problems in media with heterogeneous material properties using parallel preconditioned iterative solvers based on “PHIDAL”. Reverse Cuthill-McKee reordering with cyclic multicoloring (CM-RCM) has been applied for parallelism in ill-conditioned problems on each SMP node through OpenMP. The developed code is tested on the IBM p5-575 [10] and the TSUBAME Grid Cluster [11] using up to 512 cores. The rest of this study is organized as follows: In section 2 we outline the details of the present application, and describe the reordering procedures. In section 3 preliminary results of the computations are described, while some final remarks are offered in section 4.

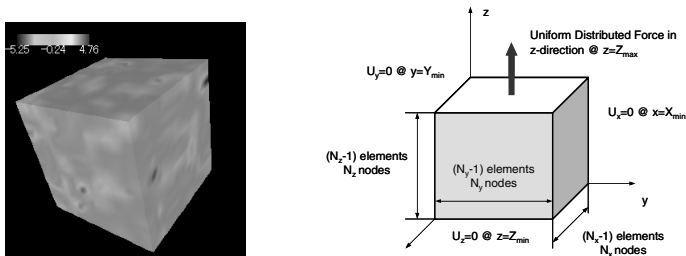
## 2 Implementations

### 2.1 Finite-Element Applications

In the present work, linear elasticity problems in simple cube geometries of media with heterogeneous material properties (Fig.5(a)) are solved using a parallel finite-element method (FEM). Tri-linear hexahedral elements are used for the discretization. Poisson’s ratio is set to 0.25 for all elements, while a heterogeneous distribution of Young’s modulus in each element is calculated by a sequential Gauss algorithm, which is widely used in the area of geostatistics [12]. The minimum and maximum values of Young’s modulus are  $10^{-3}$  and  $10^3$ , respectively, with an average value of 1.0. The boundary conditions are described in Fig.5(b). The GPBi-CG (Generalized Product-type methods based on Bi-CG) solver with SGS (Symmetric Gauss-Seidel) preconditioner (SGS/GPBi-CG) was applied. Since the present application concerns linear elasticity problems, the coefficient matrices for this application are symmetrical positive definite. Although GPBi-CG is designed for general unsymmetrical matrices, this method is adopted in the present study due to its robustness for ill-conditioned problems [1,2]. In SGS preconditioning, the original matrix  $A$  is used as a preconditioning matrix, therefore no factorization processes occur.

The code is based on the framework for parallel FEM procedures of GeoFEM [13], and the GeoFEM’s local data structure is applied. The local data structures in GeoFEM are node-based with overlapping elements [13]. In FEM-type applications, most communication between processors occurs via information exchange at domain boundaries. The ratio of communication to computation is usually small [1,2]. In the present work, GeoFEM’s original partitioner for domain decomposition has been modified so that it can create a distributed hierarchical data structure for PHIDAL. Each *sub-domain* (interior vertices, connectors of level-1) is assigned to an individual SMP node, which corresponds to each MPI process in the *hybrid* parallel programming model [1,2]. Higher-level connectors are distributed to each domain so that load-balancing can be attained and communications can be minimized. Fig.6 shows an example of the final partition of a 9-point grid into 4 domains.

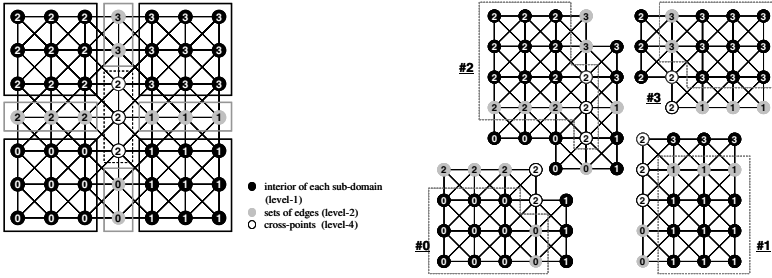
In each local data set, vertices are renumbered according to their level numbers, from the lowest to highest. During forward/backward substitution processes in SGS preconditioning, global communications are required for consistency, when computations at each *level* have been completed. Hierarchical communication tables for these special communications are also created by the modified partitioners.



(a) Heterogeneous distribution of material property

(b) Boundary conditions

**Fig. 5.** Simple cube geometries with heterogeneity as domains for 3D linear elasticity problems



(a) Final partition (number's correspond to ID of partition (0-3))

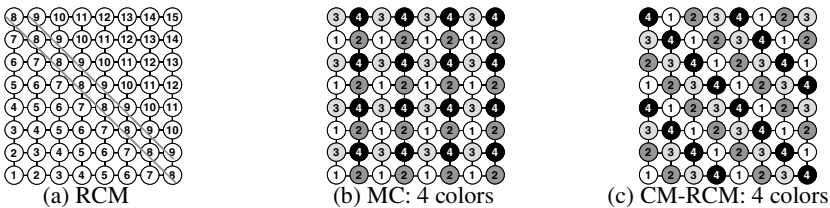
(b) Distributed local data sets with external vertices

**Fig. 6.** The final partition of a 9-point grid into 4 domains

### 2.2 Reordering Procedures

In the *hybrid* parallel programming model, multithreading, such as OpenMP, is applied to each partitioned domain defined in the previous section, where each domain corresponds to a single SMP node. The reordering of vertices in each domain allows the construction of local operations without global dependency but with continuous memory access, in order to achieve optimum parallel performance of Krylov iterative solvers with ILU/IC preconditioners. In previous studies [1,2], the author adopted multicoloring as a reordering method, mainly because a highly parallel performance and load balancing can easily be obtained. In the present work, a Reverse Cuthill-McKee (RCM) type approach is introduced. The RCM method is a typical level set reordering method [1,5,14], which is a *reversing* of the Cuthill-McKee reordering (CMK). In CMK reordering, the vertices adjacent to a visited vertex are traversed from lowest to highest degree, where the *degree* of a vertex refers to the number of vertices connected to it. Multicoloring (MC) is much simpler than RCM. MC is based on an idea in which no two adjacent vertices have the same color. In both methods, vertices of the same color (or level set) are independent. Therefore, parallel operation is possible for the vertices of each color, and the number of vertices of each color should be as large as possible for high granularity in the parallel computation.

RCM (Fig. 7(a)) reordering facilitates a faster convergence of iterative solvers with ILU/IC preconditioners than MC reordering, but leads to irregular numbers of vertices in each level set. For example, in Fig. 7(a), the 1st level set is of size 1, while the 8th level set is of size 8. In contrast, the MC method uses a uniform number of



**Fig. 7.** Example of RCM, MC and CM-RCM coloring for reordering on a 5-point grid [1,2]

vertices of each color (Fig. 7(b)). However, it is widely known that the convergence of iterative solvers with ILU/IC preconditioners is rather slow if the vertices are reordered by MC [5]. Convergence can be improved by increasing the number of colors due to the consequent reduction in the number of incompatible local graphs [5], but this reduces the number of vertices of each color. The solution to this trade-off is RCM with cyclic-multicoloring (CM-RCM) [15]. In this method, further renumbering in a cyclic manner is applied to vertices that are reordered by RCM. Figure 7(c) shows an example of CM-RCM reordering. In this case, there are 4 colors: the 1st, 5th, 9th and 13th groups in Fig. 7(a) are classified into the 1st color. There are 16 vertices in each color. In CM-RCM, the number of colors should be large enough to ensure that vertices of the same color are independent.

Figure 8 shows the convergence of the SGS/GPBi-CG solver with MC and CM-RCM reordering on a single AMD Opteron core for small problems (29,791 elements, 98,304 DOF) in elastic media with (a) homogeneous and (b) heterogeneous distributions of material properties. In both cases, CM-RCM yields a faster convergence. In

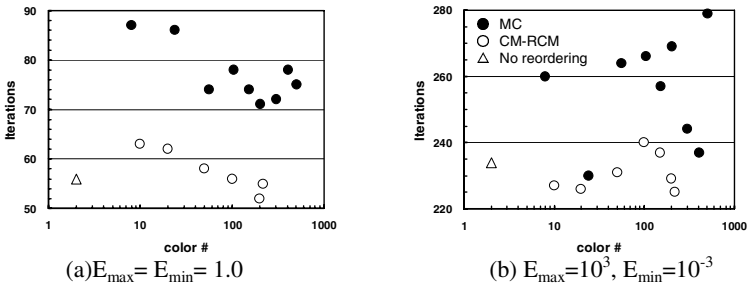


Fig. 8. Convergence of SGS/GPBi-CG on a single core for a 3D linear elasticity model with a heterogeneous distribution of Young’s modulus ( $E_{max}$ : maximum,  $E_{min}$ : minimum), for 29,791 elements and 98,304 DOF

```

do lev= 1, LEVELtot
do ic= 1, COLORTot(lev)
!$omp parallel do private(ip,i,SW1,SW2,SW3,isL,ieL,j,k,X1,X2,X3)
do ip= 1, PEsmpTOT
do i= STACKMc(ip-1,ic,lev)+1, STACKMc(ip,ic,lev)
SW1= WW(3*i-2,R); SW2= WW(3*i-1,R); SW3= WW(3*i ,R)
isL= INL(i-1)+1; ieL= INL(i)
do j= isL, ieL
k= IAL(j)
X1= WW(3*k-2,R); X2= WW(3*k-1,R); X3= WW(3*k ,R)
SW1= SW1 - AL(9*j-8)*X1 - AL(9*j-7)*X2 - AL(9*j-6)*X3
SW2= SW2 - AL(9*j-5)*X1 - AL(9*j-4)*X2 - AL(9*j-3)*X3
SW3= SW3 - AL(9*j-2)*X1 - AL(9*j-1)*X2 - AL(9*j )*X3
enddo
X1= SW1; X2= SW2; X3= SW3
X2= X2 - ALU(9*i-5)*X1
X3= X3 - ALU(9*i-2)*X1 - ALU(9*i-1)*X2
X3= ALU(9*i ) * X3
X2= ALU(9*i-4) * ( X2 - ALU(9*i-3)*X3 )
X1= ALU(9*i-8) * ( X1 - ALU(9*i-6)*X3 - ALU(9*i-7)*X2)
WW(3*i-2,R)= X1; WW(3*i-1,R)= X2; WW(3*i ,R)= X3
enddo
enddo
!$omp end parallel do
enddo

call SOLVER_SEND_RECV_3_LEV(lev,...): Communications using
enddo Hierarchical Comm. Tables.

```

Fig. 9. Forward substitution process of preconditioning written in FORTRAN and MPI with OpenMP directives, global communications using hierarchical communication tables occur at the end of the computation of each level

general, convergence is faster the larger the number of colors. The relationship between convergence and number of colors is not monotonic in heterogeneous cases, because only information for connected graphs is considered in the reordering procedures. In the present work, CM-RCM with 10 colors will be applied for performance evaluation. This reordering procedure has to be applied to the vertices at each *level*. The number of vertices in connectors of level 2 and above is usually relatively small. In the present work, the number of colors at each level is specified so that the number of vertices at each level is at least 10. Figure 9 shows *forward substitution* process of preconditioning written in FORTRAN and MPI with OpenMP directives. Global communications using hierarchical communication tables, mentioned in the previous chapter, occur in the end of the computation at each level.

### 3 Examples

#### 3.1 Hardware Environment

PHIDAL was applied to a parallel FEM code with an SGS/GPBi-CG solver using a *hybrid* parallel programming model. The FEM code solves for simulations of 3D linear elasticity problems in media with heterogeneous material properties, as shown in Fig.5. The developed code has been tested on the IBM p5-575 of the National Energy Research Scientific Computing Center at Lawrence Berkeley National Laboratory (*bassi*) [10], and the TSUBAME Grid Cluster at the Global Scientific Information and Computing Center at Tokyo Institute of Technology [11]. Table 2 presents a summary of the architectural characteristics of the each supercomputer.

The IBM p5-model 575 (IBM p5-575) at NERSC/LBNL [10] is a POWER5-based super scalar system, with 111 SMP nodes, 888 processors, and 3.5 TB memory. Total peak performance is 6.75 TFLOPS. Each node is connected via IBM's "Federation" HPS (High-Performance Switch).

The TSUBAME Grid Cluster at Tokyo Institute of Technology [11] is a scalar system with 655 SunFire X4600 nodes, where each node has 8 sockets (16 cores) of AMD's dual-core-Opteron at 2.4 GHz, connected through Coherent HyperTransport. The overall system has 10,480 cores, and 21.4 TB memory. Total peak performance is 50.4 TFLOPS. SMP nodes are connected through Infiniband 4x/Voltaire ISR 9288 switch. In the present work, only 8 of the cores on each SMP node have been used.

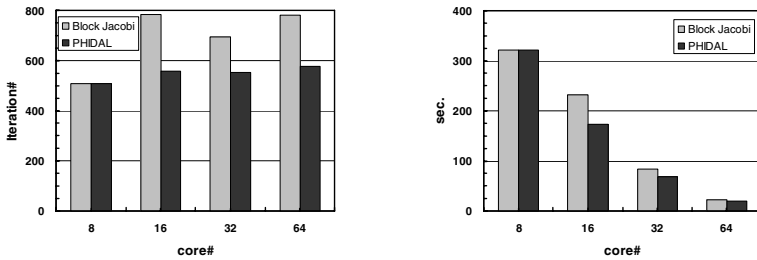
**Table 2.** Main architectural features of the IBM p5-575 and the TSUBAME Grid Cluster

	IBM p5-575 [10]	TSUBAME [11]
Core#/node	8	16
Clock rate (GHz)	1.90	2.40
Peak performance/PE (GFLOPS)	7.60	4.80
Memory/node (GB)	32	32
Peak Memory BW (GB/sec/node)	100	100
Network BW (GB/sec/node)	4.00	2.50
MPI Latency ( $\mu$ sec)	< 5.0	< 4.0

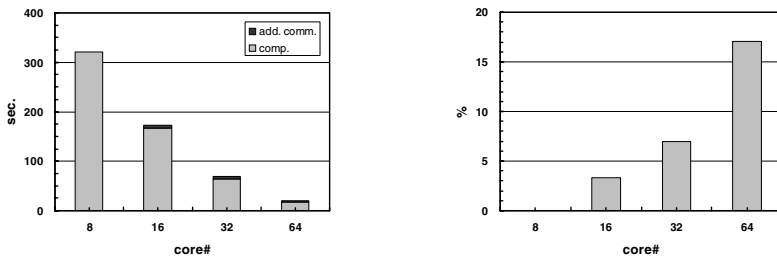
#### 3.2 Results on the IBM p5-575

Performance of the developed code has been evaluated using between 1 and 8 SMP nodes (8 and 64 cores) of the IBM p5-575. In this evaluation, a strong scaling test has

been applied, where the entire problem size is fixed at 1,594,323 DOF (512,000 elements). The hybrid parallel programming model has been applied using 8 threads. Figure 10 shows a comparison between block Jacobi-type localized preconditioner and the present PHIDAL-based one. In the case with 8 cores (= a single SMP node), there are no MPI communications, and “Block Jacobi” and “PHIDAL” provide identical results. As the number of core increases, the number of iterations required for convergence of the “Block Jacobi” increases significantly, but that of the “PHIDAL” stays constant. Both of the methods achieve excellent scalability, but PHIDAL achieves a better performance, because “PHIDAL” can consider the global effect of external domains in parallel computations, and are expected to be more robust. Figure 11 shows the effect of additional communications in SGS/GPBi-CG with PHIDAL. As shown in Fig.9, the PHIDAL algorithm requires additional communications at the end of forward/backward substitution processes at each level in the preconditioning. The ratio of the additional communications to entire computation time for the linear solver increases according to the number of cores. The ratio is 17% for 64 cores.



**Fig. 10.** Convergence (number of iterations required for convergence, and elapsed computation time for the linear solver) of SGS/GPBi-CG on the IBM p5-575 for the 3D linear elasticity model with a heterogeneous distribution of Young’s modulus ( $E_{max}=10^3$ ,  $E_{min}=10^{-3}$ ), for 512,000 elements and 1,594,323 DOF, using the *hybrid* parallel programming model



(a) Computation and additional communications (b) Ratio of additional communications

**Fig. 11.** Convergence of SGS/GPBi-CG with PHIDAL on the IBM p5-575 for the 3D linear elasticity model with a heterogeneous distribution of Young’s modulus ( $E_{max}=10^3$ ,  $E_{min}=10^{-3}$ ), for 512,000 elements and 1,594,323 DOF, using the *hybrid* parallel programming model

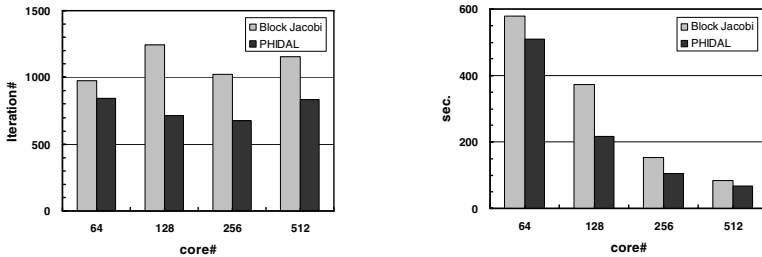
### 3.3 Results on TSUBAME

The performance of the developed code has been evaluated using between 8 and 64 SMP nodes (64 and 512 cores) of the TSUBAME Grid Cluster. A strong scaling test

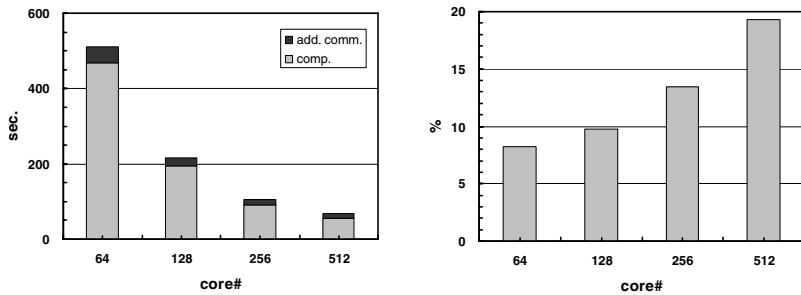
has been applied, where the entire problem size is fixed at 6,440,067 DOF (2,097,152 elements). Both a *flat MPI* and a hybrid parallel programming model have been applied, where 8 threads have been used in the latter.

Each plate of Fig. 12 shows the performance of the “Block Jacobi” and “PHIDAL” for the hybrid parallel programming model. Although both of the methods achieve excellent scalability, PHIDAL achieves the superior performance. Figure 13 shows effect of additional communications in SGS/GPBi-CG with PHIDAL. The ratio of the additional communications to entire computation time for the linear solver increases according to number of cores; the ratio is 19% for 512 cores.

Figure 14 shows a comparison between *flat MPI* and the *hybrid* parallel programming models for PHIDAL. In general, flat MPI outperforms the hybrid parallel programming model, mainly because of the efficiency of memory. However, as the number of cores increases, the problem size for each core (or SMP node) decreases, and the performance of hybrid parallel programming model improves markedly. The effect of additional communications is significant in *flat MPI*, and the ratio of these communications to the entire solver process is more than 40% at 512 cores.

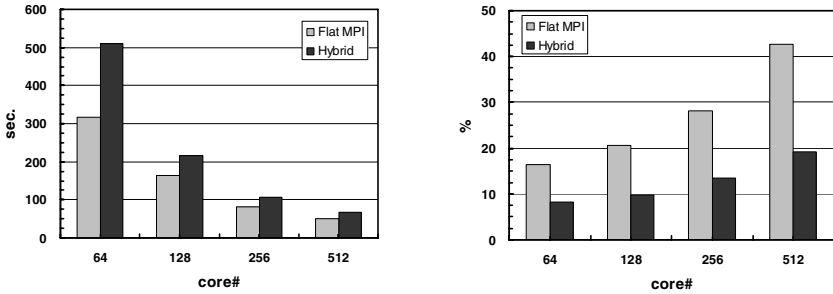


**Fig. 12.** Convergence (number of iterations for required for convergence, and elapsed computation time for linear solver) of SGS/GPBi-CG on TSUBAME for the 3D linear elasticity model with a heterogeneous distribution of Young’s modulus ( $E_{max}=10^3$ ,  $E_{min}=10^{-3}$ ), for 2,097,152 elements and 6,440,067 DOF, using the *hybrid* parallel programming model



(a) Computation and additional communications (b) Ratio of additional communications

**Fig. 13.** Convergence of SGS/GPBi-CG with PHIDAL on TSUBAME for the 3D linear elasticity model with a heterogeneous distribution of Young’s modulus ( $E_{max}=10^3$ ,  $E_{min}=10^{-3}$ ), for 2,097,152 elements and 6,440,067 DOF, using the *hybrid* parallel programming model



(a) Elapsed computation time for linear solvers (b) Ratio of additional communications

**Fig. 14.** Convergence of SGS/GPBi-CG with PHIDAL on TSUBAME for the 3D linear elasticity model with a heterogeneous distribution of Young's modulus ( $E_{\max}=10^3$ ,  $E_{\min}=10^{-3}$ ), for 2,097,152 elements and 6,440,067 DOF

## 4 Summary and Future Works

In this work, "PHIDAL (Parallel Hierarchical Interface Decomposition Algorithm)" using a hybrid parallel programming model has been applied to a finite-element solution of simulations of linear elasticity problems in media with heterogeneous material properties using parallel preconditioned iterative solvers. Reverse Cuthill-McKee reordering with cyclic multicoloring (CM-RCM) has been applied for parallelism on each SMP node through OpenMP. PHIDAL-based ILU/IC preconditioners can consider the global effect of external domains in parallel computations, and are expected to be more robust than block Jacobi-type localized ones.

The developed code has been tested on the IBM p5-575 and the TSUBAME Grid Cluster using up to 512 cores. A preconditioner based on PHIDAL provides a superior scalable performance on both architectures than a conventional block Jacobi-type localized preconditioner, although the PHIDAL-based approach requires additional communications at each level.

A comparison between flat MPI and hybrid parallel programming models for PHIDAL on TSUBAME for strong scaling shows that flat MPI is much better if the number of cores is small, but these two methods are more closely matched as the number of cores increases. The effect of additional communications is significant in cases with many cores, especially for the flat MPI parallel programming model. PHIDAL-based preconditioning with the hybrid parallel programming model is expected to be a good choice for excellent scalable performance and robustness for implementations on more than  $10^4$  cores on SMP cluster architectures and clusters of multi-core processors. A CM-RCM reordering provides a more robust convergence than MC reordering, but the relationship between number of colors and convergence is not straightforward in ill-conditioned cases with heterogeneity. Further investigation of effective reordering techniques for ill-conditioned problems is important.

In the present work, no fill-in processes have been considered in PHIDAL procedures. The results of using PHIDAL in comparison with conventional block Jacobi-type localized preconditioning have therefore not been so significant. More robust preconditioning methods based on PHIDAL may be developed by considering fill-ins

inside and between connectors. Moreover, the developed methods may further be evaluated on various types of SMP cluster architectures and clusters of multi-core processors.

## Acknowledgements

This work is supported by the 21st Century Earth Science COE Program at the University of Tokyo, and CREST/Japan Science and Technology Agency. The author would like to thank the Global Scientific Information and Computing Center at Tokyo Institute of Technology, and the National Energy Research Scientific Computing Center at Lawrence Berkeley National Laboratory, for use of their computer resources.

## References

1. Nakajima, K.: Parallel Iterative Solvers of GeoFEM with Selective Blocking Preconditioning for Nonlinear Contact Problems on the Earth Simulator. In: ACM/IEEE Proceedings of SC2003 (2003)
2. Nakajima, K.: The Impact of Parallel Programming Models on the Linear Algebra Performance for Finite Element Simulations. Lecture Notes in Computer Science 4395, 334–348 (2007)
3. Rabenseifner, R.: Communication Bandwidth of Parallel Programming Models on Hybrid Architectures. In: Zima, H.P., Joe, K., Sato, M., Seo, Y., Shimasaki, M. (eds.) ISHPC 2002. LNCS, vol. 2327, pp. 437–448. Springer, Heidelberg (2002)
4. Earth Simulator Center: <http://www.es.jamstec.go.jp/>
5. Doi, S., Washio, T.: Using Multicolor Ordering with Many Colors to Strike a Better Balance between Parallelism and Convergence. In: Proceedings of RIKEN Symposium on Linear Algebra and its Applications, pp. 19–26 (1999)
6. Washio, T., Hisada, T., Watanabe, H., Tezduyar, T.E.: A Robust and Efficient Iterative Linear Solver for Strongly Coupled Fluid-Structure Interaction Problems. *Computer Methods in Applied Mechanics and Engineering* 194, 4027–4047 (2005)
7. Nakajima, K.: Parallel Preconditioning Methods with Selective Fill-Ins and Selective Overlapping for Ill-Conditioned Problems in Finite-Element Methods. Lecture Notes in Computer Science 4489, 1085–1092 (2007)
8. Zhang, S.L.: GPBi-CG: Generalized Product-type methods based on Bi-CG for solving nonsymmetric linear systems. *SIAM Journal of Scientific Computing* 18, 537–551 (1997)
9. Henon, P., Saad, Y.: A Parallel Multistage ILU Factorization based on a Hierarchical Graph Decomposition. *SIAM Journal for Scientific Computing* 28, 2266–2293 (2007)
10. National Energy Research Scientific Computing Center, Lawrence Berkeley National Laboratory, <http://www.neresc.gov/>
11. TSUBAME Grid Clusterm, <http://www.gsic.titech.ac.jp/>
12. Deutsch, C.V., Journel, A.G., GSLIB,: Geostatistical Software Library and User's Guide, 2nd edn. Oxford University Press, Oxford (1998)
13. GeoFEM, <http://geofem.tokyo.rist.or.jp/>
14. Saad, Y.: Iterative Methods for Sparse Linear Systems. 2nd edn. SIAM (2003)
15. Washio, T., Maruyama, K., Osoda, T., Shimizu, F., Doi, S.: Efficient implementations of block sparse matrix operations on shared memory vector machines. In: Proceedings of The 4th International Conference on Supercomputing in Nuclear Applications (SNA2000) (2000)