Parallel Multilevel Iterative Linear Solvers with Unstructured Adaptive Grids for Simulations in Earth Science

Kengo Nakajima

Department of Computational Earth Sciences, Research Organization for Information Science and Technology (RIST), Tokyo, Japan (e-mail: nakajima@tokyo.rist.or.jp, phone: +81-3-3436-5271, fax: +81-3436-5274).

Abstract

Preconditioned iterative solver is one of the most powerful choice such as IC (Incomplete Cholesky) or ILU (Incomplete LU) factorization method for large-scale scientific computation. But in these methods, iteration number until convergence increases as the problem size becomes larger. In multigrid solvers, the rate of convergence is independent of problem size and the number of iterations remains fairly constant. Multigrid is also a good preconditioner for Krylov iterative solvers. In this study, multigrid preconditioned conjugate gradient iterative method (MGCG) on parallel computers has been developed and applied to the Poisson equation in the region between dual sphere surfaces on semi-unstructured prismatic grids generated adaptively. Moreover this procedure has been applied to the grids with local refinement. Computational results on Hitachi SR2201 using up to 128 processors show good scalability of the developed method.

1. Introduction

In many large-scale scientific simulation codes, almost computation is spent for linear solvers. Preconditioned Krylov iterative solvers such as ICCG (Incomplete Cholesky Factorized Conjugate Gradient) [1] provides robust convergence for wide range of scientific applications. IC and ILU (Incomplete LU Factorization) include global dependent operations but they can be localized on parallel computation[2] and smooth convergence can been obtained. Problem is that ICCG type solver is not *scalable* which means that iteration number increases as the problem size becomes larger. This is very critical if we solve $>10^9$ DOF linear equations on $>10^4$ processors.

Multigrid method[3] is a very widely known scalable method. In multigrid solvers, the rate of convergence is independent of problem size and the number of iterations remains fairly constant. Multigrid is also a good preconditioner for Krylov iterative solvers. Multigrid solvers and preconditioners have been widely used in finite-difference methods with structured grids since mid 80’s but they are not popular in finite-element methods with unstructured grids. Recently, various types of multigrid methods for unstructured grids have been developed[4][5][6][7][8] both for parallel and serial computers.

In this study, multigrid preconditioned conjugate gradient iterative method (MGCG) on parallel computers has been developed and applied to the Poisson equation in the region between dual sphere surfaces on semi-unstructured prismatic grids generated adaptively based on the method in [8]. Moreover this procedure has been also applied to the grids with local refinement.

2. Multigrid Method : Overview

Multigrid is an example of scalable linear solver. Relaxation methods like Gauss-Seidel are used there. Gauss-Seidel relaxation methods 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. Recursive application of this idea to each consecutive system of coarse-grid equations leads to a multigrid V-cycle[3] (Fig.1). If the components of the V-cycle are defined properly, the result is a method that uniformly damps all error frequencies with a computational cost that depends only linearly on the problem size. In other words, multigrid algorithms are *scalable*.

Multigrid algorithms tend to be problem specific and less robust than preconditioned Krylov iterative 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

One of the most important issue in multigrid is how to construct coarse grids. There are two basic multigrid approaches: geometric[7][8] and algebraic[6]. In geometric multigrid, the geometry of the problem is used to define the various multigrid components. In contrast, algebraic multigrid methods use only the information available in the linear system of equations.

The convergence rate of standard multigrid methods degenerates on problems that have anisotropic discrete operators which appear in very thin meshes near the wall for Navier-Stokes computation. Error becomes smooth in the direction where the connection is strong but not in the direction where the connection is weak. One popular approach in order to deal with anisotropic operators is to use *semi-coarsening* where the multigrid coarsening is not applied uniformly to all of the coordinate directions. By selecting NOT coarsening the grid in a certain direction, the anisotropy can be reduced on the coarser grid. This makes it easier for the smoother to eliminate other components of the high frequency error on the coarse grid.

3. Parallel Multigrid Preconditioned Iterative Solvers

3.1 Applications

In this study, target application is 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). Semi-implicit pressure-correction scheme[9] is applied. In this method, momentum and energy equations are solved explicitly and pressure-correction Poisson equation :

$$\Delta\phi = \nabla \cdot u'$$

is solved for incompressible constraint. This Poisson equation part 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 study, the Poisson equation solver part is mainly considered.

Semi-unstrucred prismatic grids generated from triangles on sphere surface are used (Fig.2). Meshes start from icosahedron and are globally refined recursively as in Fig.3. The grid hierarchy due to recursive refinement can be utilized for the coarse grid formation (Fig.4). In the current application, velocity components and temperature are defined at cell corners and pressure and potential for pressure correction are defined at cell center. Therefore dependent variables are defined at cell center in this study.

3.2 Parallel MGCG solvers for Poisson Equations

Parallel MGCG (Multigrid preconditioned Conjugate Gradient) solver is developed in Fortran 90 with MPI[10]. Features of the procedure are summarized as follows :

(1) V-cycle Multigrid Preconditioned MGCG solver
(2) Gauss-Seidel and ILU(0) smoothing.
(3) Semi-coarsening in lateral and normal-to-surface direction.
(4) Entire region is partitioned along radial (normal-to-surface) direction.
(5) Multilevel communication tables are defined at partition interfaces
(6) Effect of local grid refinement can be considered.

Basically, Gauss-Seidel iterative method has been adopted as smoother, but ILU(0) factorization[1] has been also tested as smoother.

V-cycle method described in Fig.1 has been adopted. In each cycle, Gauss-Seidel procedures are repeated 5 times for both restriction (*fine-to-coarse*) and prolongation (*coarse-to-fine*) or until convergence has stagnated as is shown in Fig.5. The ILU(0) smoother has been implemented with additive Schwartz domain decomposition method[4] at each level. 2 iterations are applied at each multigrid level (1 smoothing + 1 domain decomposition + 1 smoothing).

Semi-coarsening method has been introduced in lateral and normal-to-surface directions. In order to keep the semi-unstructured grid feature in normal-to-surface direction, entire region is partitioned along radial direction.

Parallel multigrid cycle is executed until :

• Lateral direction :	Initial icosahedron (20 triangles)
• Normal-to-surface direction :	1 layer

on each PE. Furthermore, multigrid procedure on single PE has been continued until the number of layer is equal to 2. Equation on the coarsest grid with (20*2=40 cells) has been solved accurately by Gauss-Seidel method. These computations on single PE are very small and does not affect the parallel efficiency.

In parallel computation with unstructured/semi-unstructured grids using message passing library, communication tables among partitions should be explicitly defined by users[2]. In this study, multi level communication tables have been defined according to the multigrid procedure at each level (Fig.6). In this procedure, variables are defined at cell centers, therefore the entire region is partitioned in a *cell-based* manner. Cells are classified into the following three categories from the viewpoint of message passing :

- Internal cells (originally assigned cells)
- External cells (cells which form the matrix connectivity in the partition but are located outside of the partition)
- Boundary cells (*external cells* of other partitions)

Values in *boundary* cells in the partitions are *sent* to the neighboring partitions and they are *received* as *external* cells at the *destination* partition. This type of communication table is defined at each grid level.

3.3 Grid Adaptation

Adaptive methods in applications with unstructured meshes have evolved as efficient tools for obtaining numerical solution without a priori knowledge of the details of the nature of the underlying physics[8].

A dynamic adaptation algorithm developed for 3D unstructured meshes[8] by the author has been implemented. The algorithm is capable of simultaneous refinement and coarsening of the appropriate regions in the flow domain.

The adaptation algorithm is guided by a feature detector that senses regions with significant changes in flow properties, such as shock waves, separations and wakes. Velocity differences and gradients are used for feature detection and threshold parameters are set in order to identify the regions to be refined or coarsened. The details of the method for feature detection used in this study are described in [8]. In the present implementation, the feature detector marks edges.

The prisms are refined directionally in order to preserve the structure of the mesh along the normal-to-surface direction. The prismatic mesh refinement proceeds by dividing only the *lateral* edges and faces. Faces are refined either by *quadtree* or *binary* division. The resulting surface triangulation is replicated in each successive layer of the prismatic mesh as illustrated in Figure 7. As is seen from this figure, the prismatic mesh refinement preserves the structure of the initial mesh in the direction normal to the surface.

In order to avoid excessive mesh skewness, repeated *binary* divisions of prisms are not allowed. Furthermore, in order to avoid sudden changes in mesh size, the mesh refinement algorithm also limits the maximum difference in embedding level between neighboring elements less than two.

Recently, various types of multigrid methods for locally refined grid have been developed for block structured type grids in finite-difference methods. Typical procedure described in [3] is to utilize the grid hierarchy in adapted grid and to apply nested multigrid procedure for each adaptation level. This approach (*level-by-level method*) usually requires additional memory and computations for *fine* cells without adaptation (white triangles in Fig.8). In this study, we applied *direct jump method* where we start from fine grid with *full (deepest)* adaptation level and the go back directly to the 2nd globally finest grid level in multigrid procedure as is described in Fig.8.

4. Results

Developed methods are tested on Poisson equations for the region between dual sphere surfaces using Hitachi SR2201 parallel computer at the University of Tokyo[11] up to 128 processors. 2 types of problems have been considered. In both problems the following homogeneous Poisson equation has been solved :

$$\Delta\phi = 1$$

In the 1st application, problem size for 1 processor is fixed as 320 (triangles, Level=2 in Fig.3) X 900 (layers) = 288,000 (cells) and computations using from 2 PEs (processing element) to 128 PEs have been tested. This corresponds to 576,000 to 36,864,000 cells.

Inner radius of the sphere is 0.50 and thickness of the each layer has been fixed as 0.01. 2 types of boundary conditions have been applied as follows :

- **Uniform** : Dirichlet boundary condition (ϕ=0) for all triangles at outermost surface of the prisms
- **One-Patch** : Dirichlet boundary condition (ϕ=0) for just ONE triangle patch of the initial icosahedron at outermost surface of the prisms. Dirichlet B.C. has been applied to all children and grandchildren generated from this ONE of 20 triangles. This configuration provides much more *ill-conditioned* coefficient matrices than the *Uniform* cases.

Fig.9 (a) and (b) show the results (elapsed computation time including communication) for MGCG/GS(Gauss-Seidel) and ICCG computations in Application-I. Computation time remains almost constant in MGCG/GS 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[2] and close to Jacobi smoothing as the PE number increases. But even in the 128 PE cases, MGCG/GS is much faster than ICCG.

Fig. 10 (a) and (b) show the results for MGCG/GS, MGCG/ILU(0) and ICCG for Application-I up to 32 PEs. MGCG/ILU(0) shows more robust convergence than MGCG/GS especially for the cases with *One-Patch* boundary condition. Fig.11 (a) and (b) show the iterations until convergence for these 3 methods. Iteration number of MGCG/ILU(0) remains rather constant compared with MGCG/GS cases even if the number of PEs increases.

In the 2nd application, effect of local grid refinement and multgrid strategy (direct jump and level-by-level) mentioned in the previous section have been evaluated. Inner radius of the sphere is 0.50 and thickness of the each layer has been fixed as 0.01. 2 types of boundary conditions mentioned in the 1st application have been applied. Initial grid is the Level-2 grid in Fig.3 with 320 triangle surfaces. 3-level grid adaptations have been applied as is shown in Fig.12. At each adaptation level, number of the triangular surfaces is as follows :

•	1st-level:	532	triangles
•	2nd-level:	1508	
•	3rd-level:	4448	

In the 2nd application, layer number on each PE is fixed as 50 and same Poisson equation as the 1st application has been solved by MGCG/GS using 2-32 PEs. Fig.12 shows the computational time normalized by number of cells (=DOFs (degree of freedom)) on each processor according to problem size (= PE numbers) using Direct Jump Method. If the procedure is scalable, the curves should be flat. In the cases with Uniform boundary condition, the MGCG/GS with Direct Jump Method provides very nice scalability except for 32 PE case with 3-level adapted grids. In the cases with One-Patch boundary condition, efficiency goes down as the PE number increases. This is as similar as in the 1st application (Fig.9-11).

Fig.13 shows the comparison between Direct Jump and Level-by-Level multigrid strategies for adapted grid with Uniform boundary conditions. If the adaptation level is *shallow*, two methods are competitive, but at *deeper* levels of adaptation, the Direct Jump Methods shows much more excellent efficiency.

In both applications, communication overhead is very low (less than 1%) except the cases for initial grids with 320 triangles in the 2nd applications with 50 layers for 1 PE where computational load is rather low.

5. Concluding Remarks

Multigrid preconditioned conjugate gradient iterative method (MGCG) on parallel computers has been developed. V-cycle and semi-coarsening approach have been adopted for multigrid procedure. Both Gauss-Seidel and ILU(0) with additive Schwartz Domain Decomposition smoothers have been tested. Developed procedure has been applied to the Poisson equation in the region between dual sphere surfaces on semi-unstructured prismatic grids generated adaptively under various types of boundary conditions. Computational results on Hitachi SR2201 using up to 128 processors show good scalability of the developed method compared with ICCG solvers. Communication overhead is less than 1 % for sufficiently large problems. Efficiency of the Gauss-Seidel smoother becomes worth as the PE number increases because the procedure is localized. ILU(0) smoother is rather robust for cases with many PEs.

Developed procedure has been also applied to the grids with local refinement. 2-types of multigrid strategy (Direct Jump and Level-by-Level) have been compared. Direct Jump is much more efficient in cases with deeper level adaptation in spite of its simplicity.

In the next stage, further robust smoothers should be developed for more PE cases such as >1,000.

Acknowledgments

This study is a part of the project "Solid Earth Platform for Large Scale Computation" funded by the Ministry of Education, Culture, Sports, Science and Technology, Japan through its "Special Promoting Funds of Science & Technology".

Furthermore the author would like to thank Professor Yasumasa Kanada (Computing Center, The University of Tokyo) for fruitful discussions on high performance computing.

References

[1] Barrett, R., Bery, M., Chan, T.F., Donato, J., Dongarra, J.J., Eijkhout, V., Pozo, R., Romine, C. and van der Vorst, H. : Templates for the Solution of Linear Systems : Building Blocks for Iterative Methods, SIAM, 1994.

[2] GeoFEM Web Site : http://geofem.tokyo.rist.or.jp/

[3] Briggs, W.L., Henson, V.E. and McCormick, S.F. : A Multigrid Tutorial Second Edition, SIAM, 2000.

[4] Smith, B., Bjϕrstad, P. and Gropp, W. : Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations, Cambridge Press, 1996.

[5] .CASC (Center for Applied Scientific Computing, Lawrence Livermore National Laboratory) Web Site : http://www.llnl.gov/CASC/linear_solvers/

[6] Stüben, K. : Algebaic Multigrid (AMG) : An Introduction with Applications, GMD Report 53, GMD-Forschungstentrum Informationstechnik GmbH, 1999.

[7] Adams, M.F. and Demmel, J,W. : "Parallel Multigrid Solver for 3D Unstructured Finite Element Problems ", SC99 Proceedings, Portland, Oregon, USA, 1999.

[8] Parthasarathy, V., Kallinderis, Y. and Nakajima, K. : "A Navier-Stokes Method with Adaptive Hybrid Prismatic/Tetrahedral Grids", AIAA Paper 95-0670, 1995.

[9] Kallinderis, Y. and Nakajima, K. : " Finite Element Method for Incompressible Viscous Flows with Adaptive Hybrid Grids", AIAA Journal, Vol.32, No.8, pp.1617-1625, 1994.

[10] Gropp, W., Lusk, E. and Skjellum, A., : Using MPI, MIT Press, 1994.

[11] The University of Tokyo, Computer Center, Web Site : http://www.cc.u-tokyo.ac.jp/

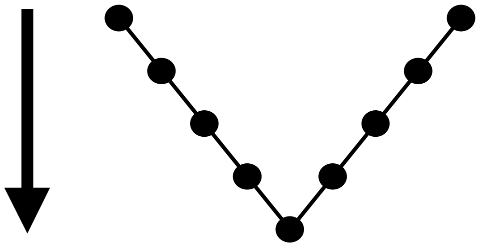


Fig.1 V-cycle for Multigrid Operation

The down-cycle of a multigrid V-cycle uses smoothers to damp oscillatory error components at different grid scales. The up-cycle corrects the smooth error components remaining on each grid level by using the error approximations on coarser grids[4]

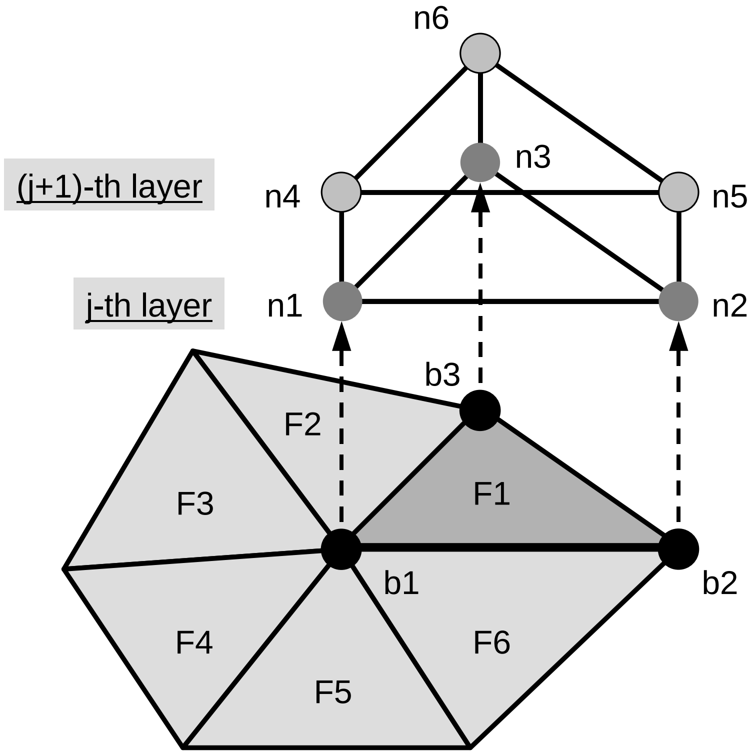


Fig.2 Prisms generated from Surface Triangles

The surface of the model is covered with triangles, which provide geometric flexibility, while the structure of the mesh in the direction normal to the surface provides thin prismatic elements suitable for the viscous region.

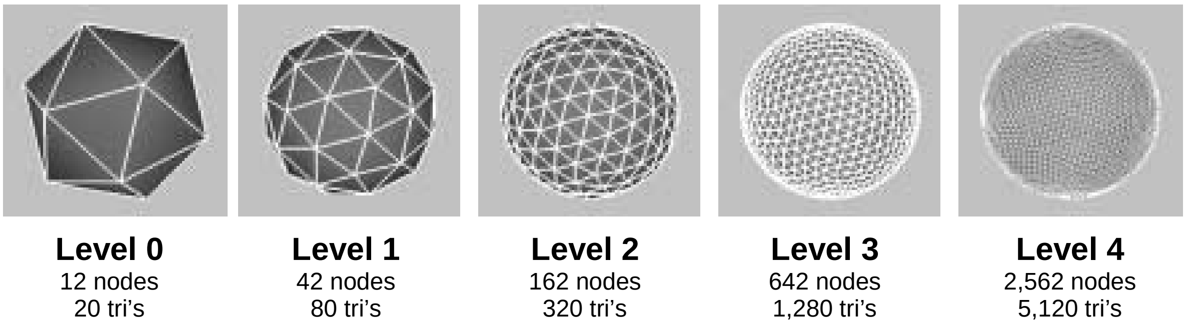
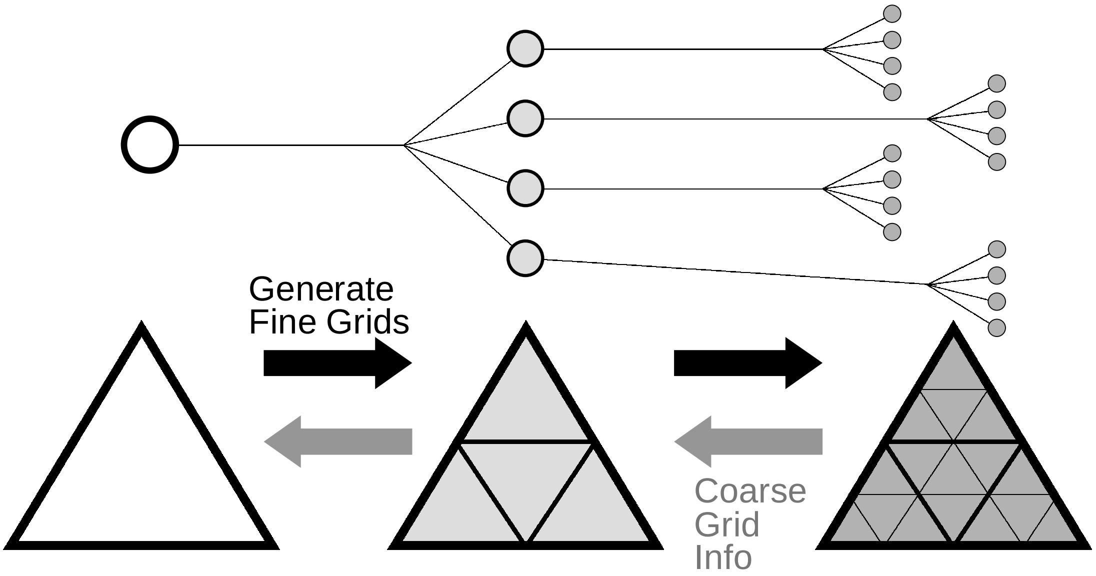
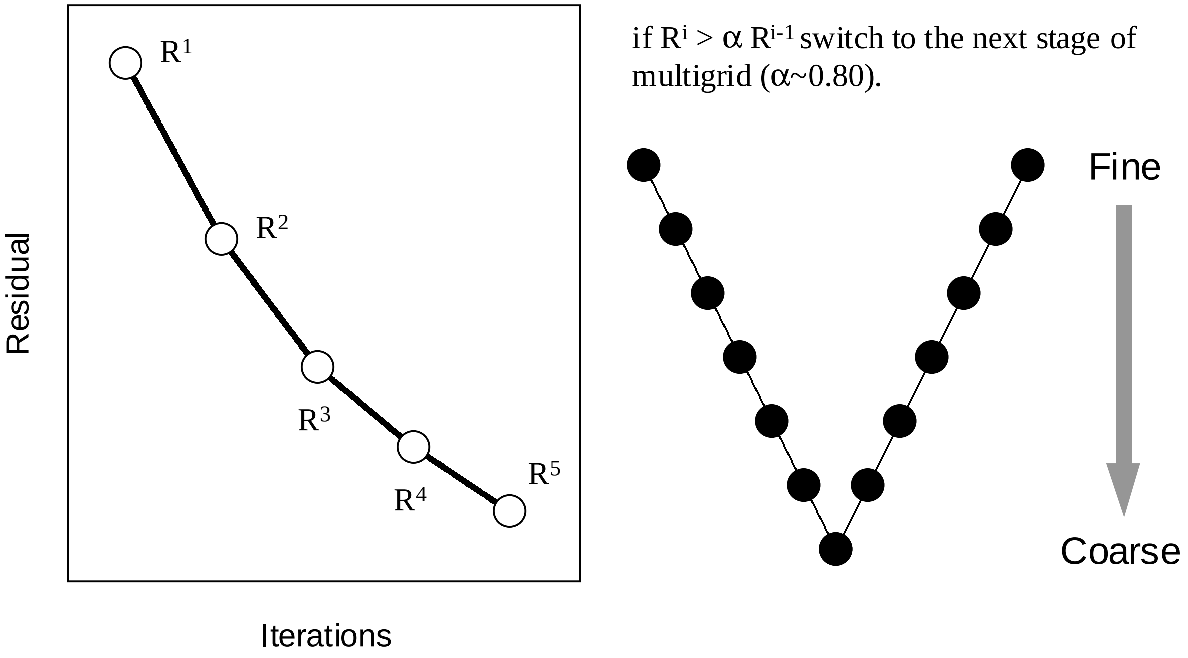


Fig.3 Surface triangle meshes generated from icosahedron

4 children generated from 1 parent triangle





if $R^i > \alpha R^{i-1}$ switch to the next stage of multigrid (α~0.80).

Fig.5 Smoothing Strategy at Each Restriction/Prolongation Stage

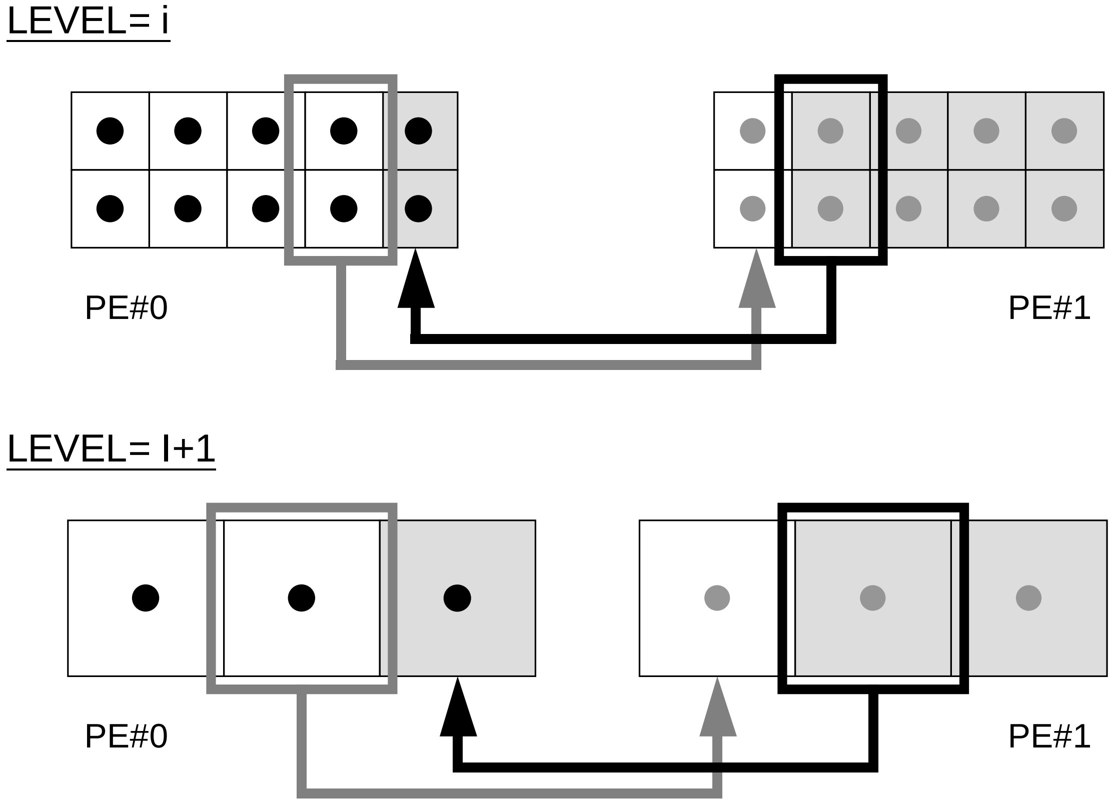


Fig.6 Multilevel Communication Table

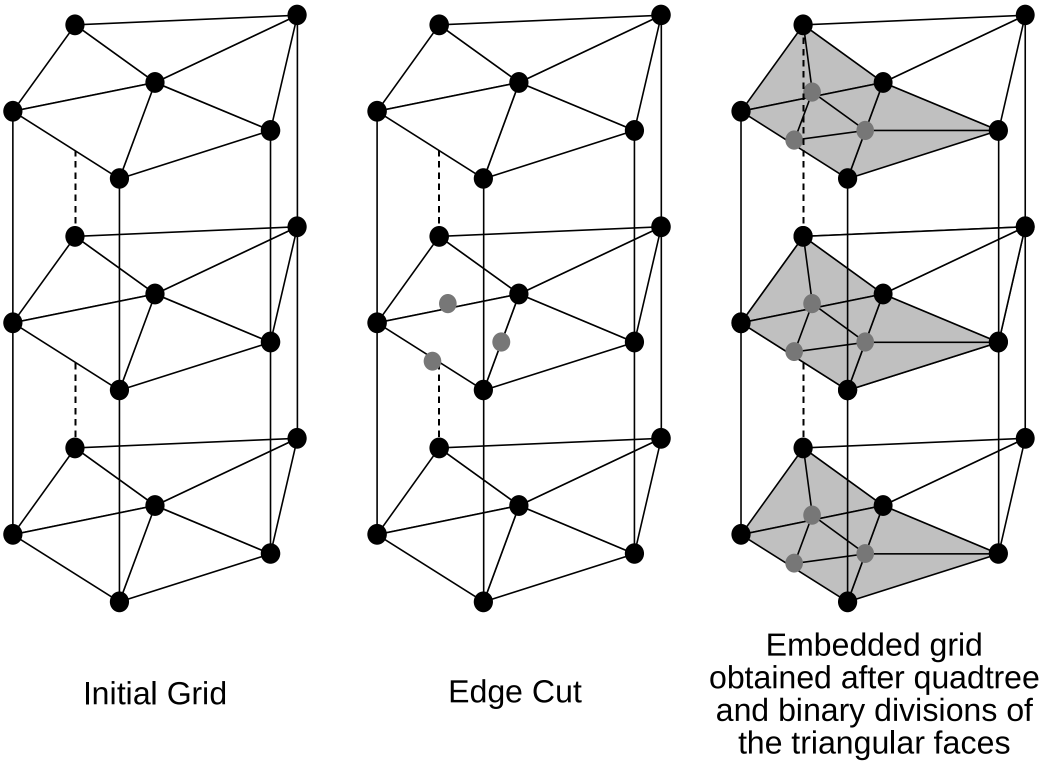

Initial Grid
Edge Cut
Embedded grid obtained after quadtree and binary divisions of the triangular faces

Fig.7 Directional Refinement of Prisms based on *Quadtree* and *Binary* Divisions of the Triangular Faces on the Wall

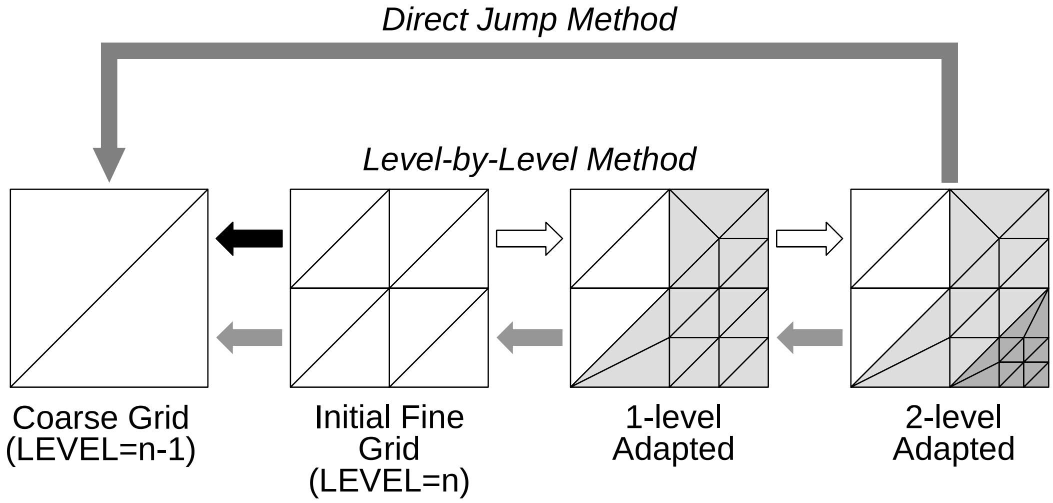
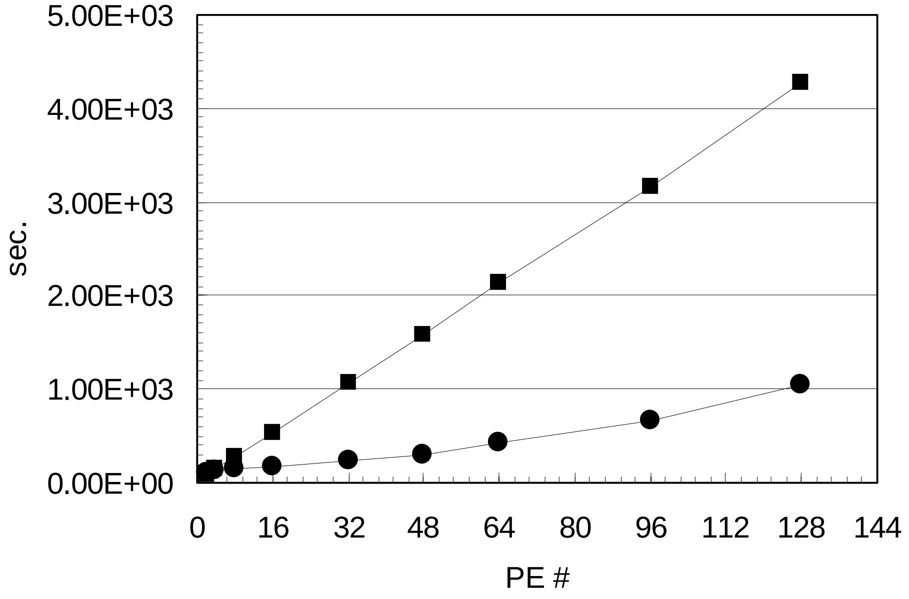
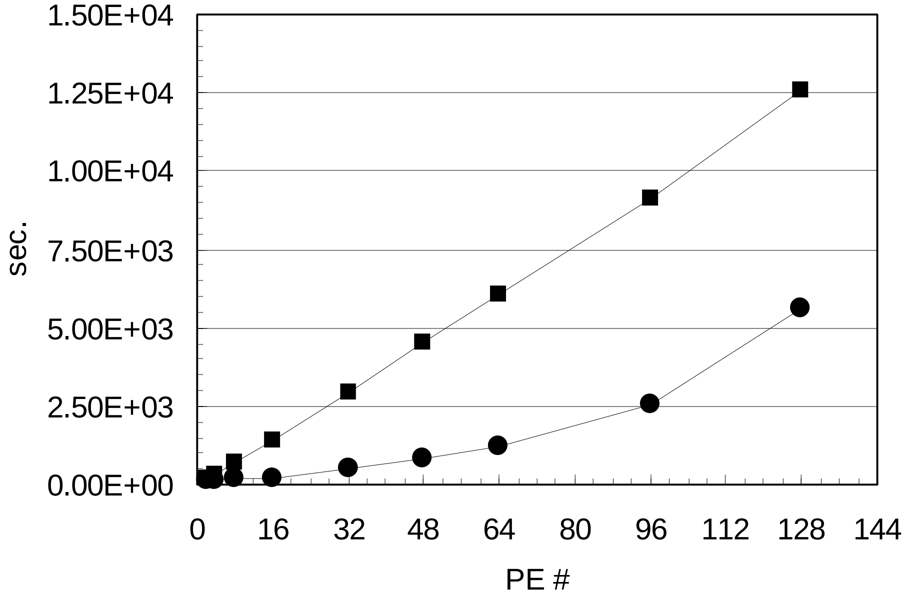


Fig.8 Multigrid Strategy for Locally Refined Grids Level-by-Level and Direct Jump Method

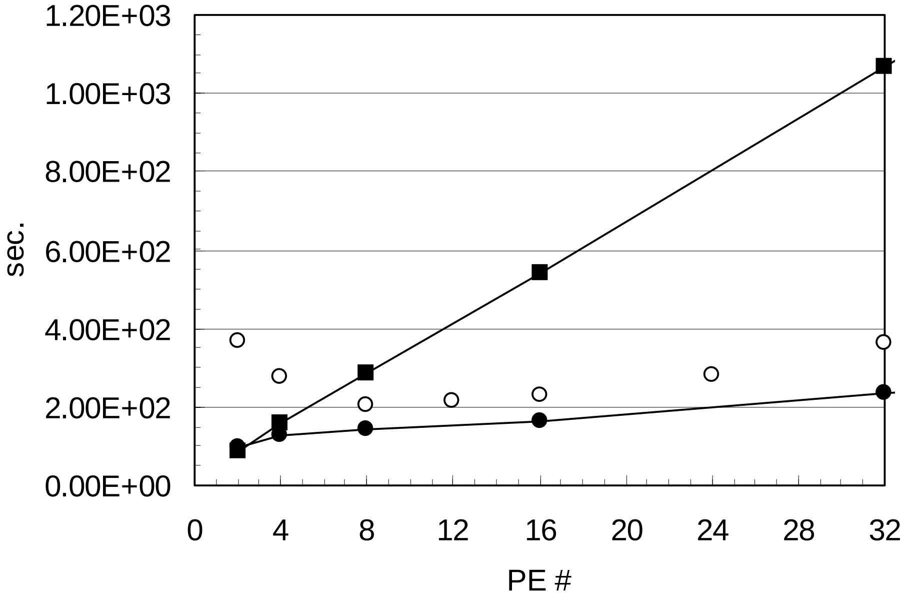


(a) Uniform Boundary Condition

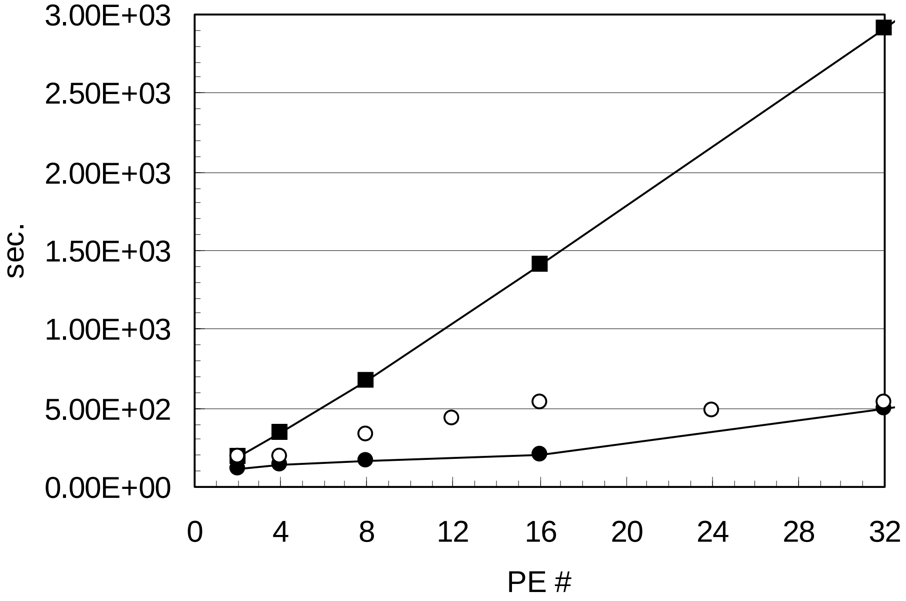


(b) One-Patch Boundary Condition

Fig.9 Results of Application-I (Elapsed computation time including communication by Hitachi SR2201 for fixed problem size on each processor. 320*900=288,000 cells/PE, 2-128 PEs (up to 36,864,000 cells). Black Square: ICCG, Black Circle: MGCG/GS)

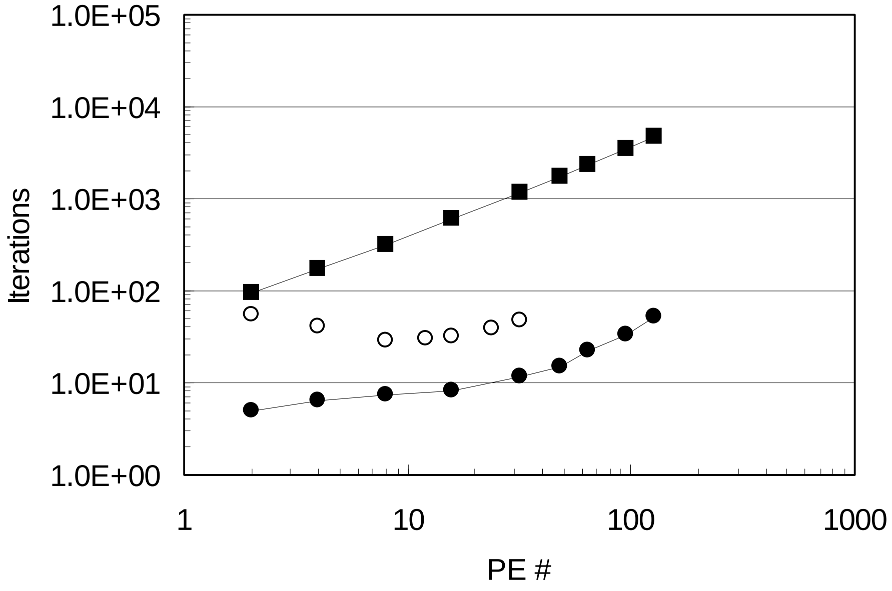


(a) Uniform Boundary Condition

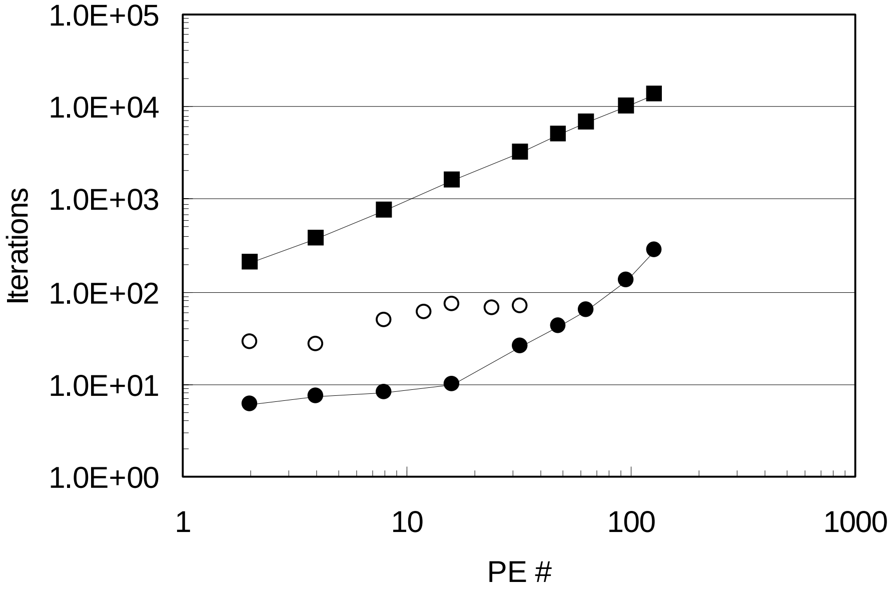


(b) One-Patch Boundary Condition

Fig.10 Results of Application-I (Elapsed computation time including communication by Hitachi SR2201 for fixed problem size on each processor. 320*900=288,000 cells/PE, 2-32 PEs (up to 9,216,000 cells). Black Square: ICCG, Black Circle: MGCG/GS, White Circle: MGCG/ILU(0))

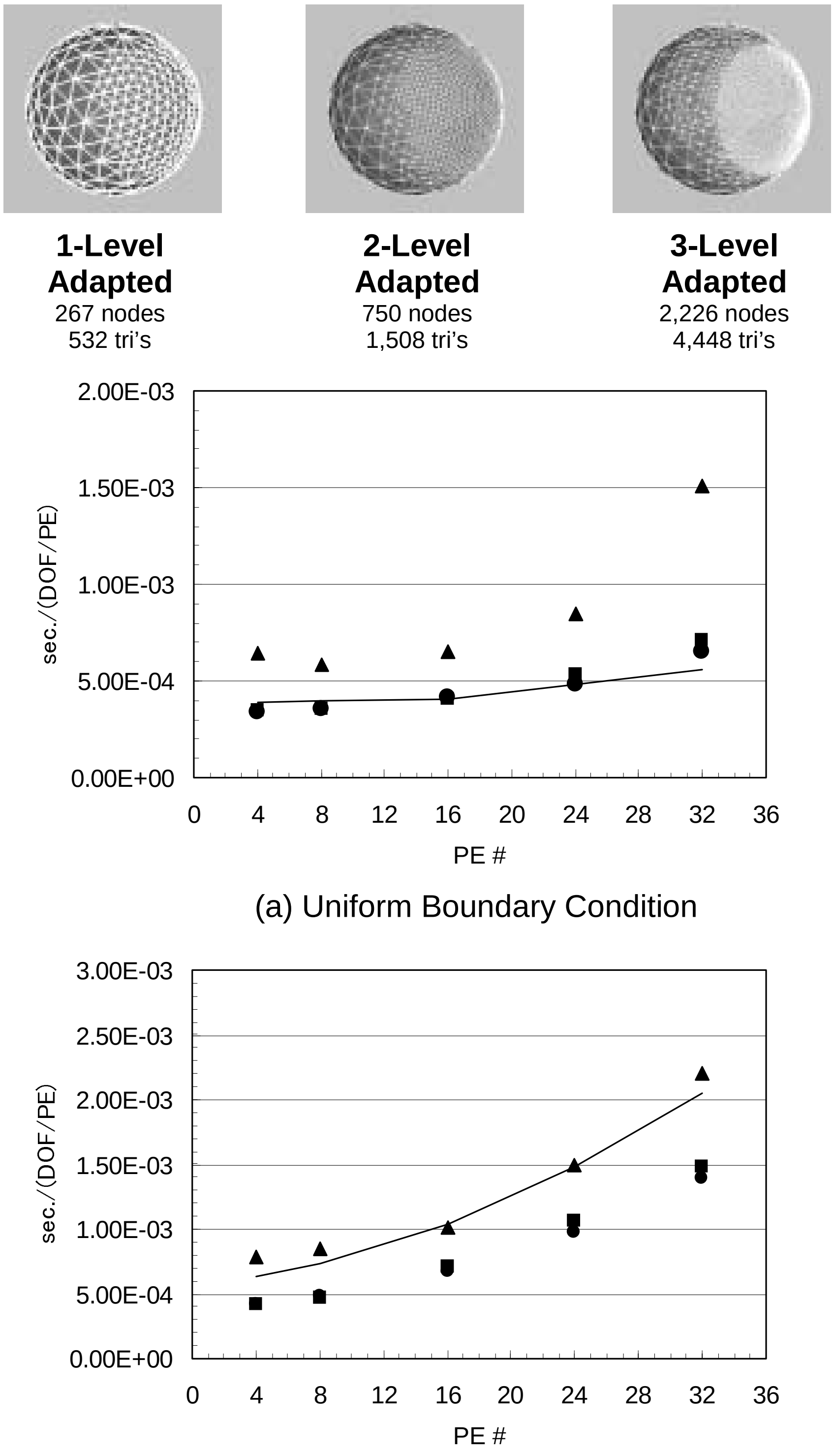


(a) Uniform Boundary Condition



(b) One-Patch Boundary Condition

Fig.11 Results of Application-I (Iteration number until convergence by Hitachi SR2201 for fixed problem size on each processor. 320*900=288,000 cells/PE, 2-32 PEs (up to 9,216,000 cells). Black Square: ICCG, Black Circle: MGCG/GS, White Circle: MGCG/ILU(0))



(a) Uniform Boundary Condition

(b) One-Patch Boundary Condition

Fig.12 Results of Application-II (Elapsed computation time including communication normalized by cell number/PE on Hitachi SR2201 for Locally Refined Grids using MGCG/GS. 50 layers/PE, 4-32 PEs (up to 7,116,800 cells). Solid Line: Initial Grid (320 triangles), Circle: 1-level adapted, Square: 2-level adapted, Triangle: 3-level adapted)

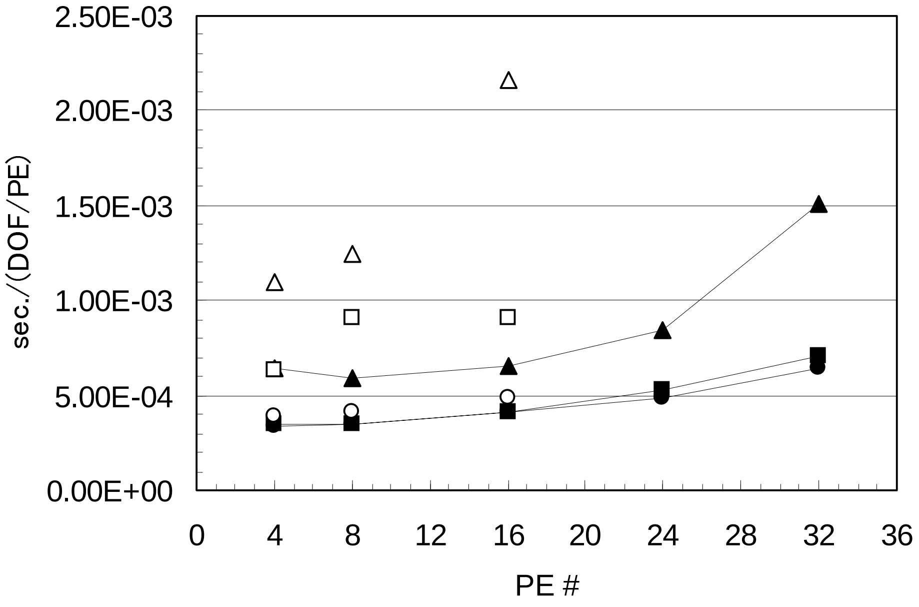


Fig.13 Results of Application-II (Elapsed computation time including communication normalized by cell number/PE on Hitachi SR2201 for Locally Refined Grids using MGCG/GS. 50 layers/PE, 4-32 PEs (up to 7,116,800 cells). Uniform Boundary Conditrions

Circle: 1-level adapted, Square: 2-level adapted, Triangle: 3-level adapted)

Black : Direct Jump Method, White: Level-by-Level Method