

## **Chapter 6     Parallel 3D Adaptive Navier-Stokes Solver in GeoFEM with Dynamic Load-Balancing**

Grid adaptation is a very useful method for applications with unstructured meshes but requires dynamic load-balancing for efficient parallel computation. In this chapter, a parallel 3D compressible Navier-Stokes code with adaptive hybrid meshes (epHYBRID) and parallel adaptation procedure (pADAPT) have been developed on the GeoFEM parallel platform. The DRAMA library has been integrated into the pADAPT module to solve the load-balancing problem. The entire code system has been evaluated under various types of conditions on Pentium clusters and Hitachi SR2201. Results show that DRAMA library provides accurate load-balancing for parallel mesh adaptation in pADAPT and excellent parallel efficiency in the Navier-Stokes computations in epHYBRID.

## 6.1 Introduction

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. But these methods cause severe load imbalance among processors in parallel computations. Recently, various types of methods for dynamic load-balancing in parallel mesh adaptation have been developed [9,85,105,114].

In this chapter, a parallel 3D compressible Navier-Stokes code with adaptive hybrid meshes (epHYBRID) and parallel mesh adaptation module (pADAPT) have been developed on the GeoFEM parallel platform. A repartitioning tool based on the DRAMA library [129] that provides dynamic load-balancing and complete data migration has been integrated into the pADAPT module. In the following section of this chapter, we outline the numerical method used in epHYBRID, and the parallel adaptation and load-balancing algorithm in pADAPT/DRAMA. Finally, the extended the GeoFEM data structures for parallel mesh adaptation are described.

The entire code system (Fig.6.1) has been tested with the simulation of the supersonic flow around a spherical body on Pentium cluster and Hitachi SR2201. Various types of repartitioning methods in the DRAMA library have been evaluated.

## 6.2 Parallel 3D Compressible Navier-Stokes Solver : epHYBRID

### 6.2.1 Outline

The epHYBRID code for parallel 3D compressible Navier-Stokes simulation is based on a sequential version of program which was originally developed for single CPU workstations by the author [89,91] for the simulation of the external flow around airplanes. An edge-based finite-volume method with unstructured prismatic/tetrahedral hybrid meshes suitable for complicated geometry is applied. The solution is marched in time using a Taylor series expansion following the Lax-Wendroff approach. Although the original program was written in Fortran 77, the newly developed parallel version is written in Fortran 90 to exploit its dynamic memory management features and uses the message passing interface (MPI) for communication.

In the hybrid mesh system, 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.6.2 and Fig.6.3). The outermost layer of the prismatic mesh is then used as the inner boundary surface for a tetrahedral mesh (Fig.6.3), which covers the rest of the computational domain. Tetrahedral meshes are also suitable for connecting different prismatic regions. Figure 6.4 shows an example of the hybrid meshes around a sphere.

### 6.2.2 Governing Equations

The Navier-Stokes equations for viscous fluid flow are written in the differential form as follows:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathcal{F} = \nabla \cdot \mathcal{R} \quad (6.1)$$

where  $\mathbf{U}$  is the state vector;  $\mathcal{F}$  comprises the convective flux vector components  $\mathbf{F}$ ,  $\mathbf{G}$  and  $\mathbf{H}$  in x, y, z- directions respectively;  $\mathcal{R}$  comprises the viscous flux vector components  $\mathbf{R}$ ,  $\mathbf{S}$  and  $\mathbf{T}$  in x, y, z- directions respectively. The state vector and the convective and viscous flux vectors are defined in terms of primitive variables.

The solution at any node N, at time level n+1 can be expressed in terms of the solution at time level n using a Taylor series expansion:

$$\mathbf{U}_N^{(n+1)} = \mathbf{U}_N^{(n)} + \delta \mathbf{U}_N^{(n)}$$

$$\delta \mathbf{U}_N^{(n)} = \mathbf{U}_N^{(n+1)} - \mathbf{U}_N^{(n)} = \left( \frac{\partial \mathbf{U}}{\partial t} \right)_N^{(n)} \Delta t + \left( \frac{\partial^2 \mathbf{U}}{\partial t^2} \right)_N^{(n)} \Delta t^2 + o(\Delta t^3) \quad (6.2)$$

The temporal derivatives in the preceding expression are evaluated in terms of spatial derivatives using the governing equations according to the Lax-Wendroff approach. The finite-volume method evaluates the integral averages of the temporal derivative terms in equation (6.2) over the control volume  $\Omega_N$  associated with node N.

### 6.2.3 Spatial Discretization with Mixed Elements

The spatial discretization proceeds by constructing a dual cell around each node N that represents the control volume over which the integral averages of the temporal derivatives are evaluated. The two-dimensional analogy of defining dual cells for different configurations in a triangular-quadrilateral hybrid mesh is illustrated in Fig.6.5. The duals are defined by connecting the midpoints of the edges and centroids of the triangular and/or quadrilateral faces that share the node. Dual cells for a three-dimensional hybrid grid are constructed along similar lines using the centroids of faces and cells with which each node is associated.

The integral average of the first-order temporal derivatives associated with the node N is written in discrete form following the governing equation (6.1):

$$\left( \frac{\partial \mathbf{U}}{\partial t} \right)_N = -\frac{1}{\Omega_N} \sum_{\mathbf{f}} (\mathcal{F} - \mathcal{R})_{\mathbf{f}} \cdot \hat{\mathbf{n}}_{\mathbf{f}} S_{\mathbf{f}} \quad (6.3)$$

where the summation  $\mathbf{f}$  is over all of the discrete faces of the dual mesh that constitute  $\partial\Omega_N$ . It is shown in [41,91] that the summation in equation (6.3) can be alternatively computed on an edgewise basis as:

$$\left( \frac{\partial \mathbf{U}}{\partial t} \right)_N = -\frac{1}{\Omega_N} \sum_{\mathbf{e}} (\mathcal{F} - \mathcal{R})_{\mathbf{e}} \cdot \hat{\mathbf{n}}_{\mathbf{e}} S_{\mathbf{e}} \quad (6.4)$$

where the summation  $\mathbf{e}$  is over all of the edges that share the node N. The term  $S_{\mathbf{e}}$  represents the dual-face area associated with each edge, and  $\hat{\mathbf{n}}_{\mathbf{e}}$  is the unit normal vector of the dual-face area  $S_{\mathbf{e}}$ . The  $S_{\mathbf{e}}$  are computed using the dual mesh

construction of Fig.6.5 and Fig.6.6, by accumulating the areas of each dual-mesh face that shares the edge. The finite volume scheme then proceeds by computing  $\delta\mathbf{U}$ s at the nodes by a global sweep over the edges and is thus, transparent to whether a node lies in the tetrahedral region, in the prismatic region, or at the interfaces.

The second-order temporal derivatives are evaluated along similar lines. The expression for the second-order derivatives at node N is given from [41,91] as follows:

$$\left(\frac{\partial^2 \mathbf{U}}{\partial t^2}\right)_N = -\frac{1}{\Omega_N} \int_{\partial\Omega_N} (\tilde{\mathbf{A}}_{n_x} + \tilde{\mathbf{B}}_{n_y} + \tilde{\mathbf{C}}_{n_z}) \frac{\partial \mathbf{U}}{\partial t} dS \quad (6.5)$$

where  $\tilde{\mathbf{A}} = \frac{\partial \mathbf{E}}{\partial \mathbf{U}}$ ,  $\tilde{\mathbf{B}} = \frac{\partial \mathbf{F}}{\partial \mathbf{U}}$ ,  $\tilde{\mathbf{C}} = \frac{\partial \mathbf{G}}{\partial \mathbf{U}}$  are the Jacobians of convective flux vectors. The Jacobians of flux vectors need to be computed to evaluate the second-order derivatives. However, only the convective flux vectors are considered in this step as the Jacobians of viscous flux vectors are too expensive to compute. Therefore, discretization of the viscous terms is first-order accurate in time and second-order accurate in space.

#### 6.2.4 Upwind-like Artificial Dissipation

The dissipation modeling in this work is formulated in such a manner as to simulate the implicit dissipation terms of the upwinding schemes without increasing the computation cost of the algorithm [41,89,91].

The numerical formula for the flux vector at any intermediate state I between two end states L and R can be expressed as:

$$\mathbf{F}_I = \frac{1}{2}(\mathbf{F}_L + \mathbf{F}_R) - \tilde{\mathbf{A}}_r(\mathbf{U}_R - \mathbf{U}_L) \quad (6.6)$$

where  $\tilde{\mathbf{A}}_r$  is Roe's matrix [41,89,91]. The dissipation terms are modeled so as to be similar to the second term of the above equation as this corresponds to the implicit smoothing term of the upwinding scheme. A simplified form of Roe's matrix [41,89,91] is obtained by replacing  $\tilde{\mathbf{A}}_r$  with  $\rho(\tilde{\mathbf{A}}_r) = |u| + c$ , the maximum eigenvalue of Roe's matrix. This ensures that the dissipation terms do not dwindle down to zero near the stagnation or the sonic points.

To extend this concept to 3D, *edge-based* operations are adopted for calculation of the artificial dissipation term. The contribution  $(\delta \mathbf{U}_0^n)_{s2}$  of shock smoothing terms to the change  $\delta \mathbf{U}_0^n$  at the node 0 is given as follows:

$$\begin{aligned}
 (\delta \mathbf{U}_0^n)_{s2} &= \frac{\Delta t}{\Omega_0} \sum_{e=1}^{Ne} (f(\mathbf{u}, S) + |c||S|)_e (\mathbf{U}_{N(e)} - \mathbf{U}_0) \\
 f(\mathbf{u}, S) &= |u S_x + v S_y + w S_z| \\
 |S|_e &= |S_x^2 + S_y^2 + S_z^2|_e^{1/2}
 \end{aligned} \tag{6.7}$$

where  $e$  denotes the connected edge. The shock smoothing term is evaluated similar to the viscous fluxes on an edge-wise basis. The fourth order smoothing contribution  $(\delta \mathbf{U}_0^n)_{s4}$  is computed in a similar fashion. Instead of the first difference of state vectors as used in equation (6.7), a difference of the accumulated first difference over the edges sharing a node is used for background smoothing in the flow high Reynolds number.

The change  $(\delta \mathbf{U}_0^n)_s$  at the node 0 due to second and fourth order smoothing is given by:

$$(\delta \mathbf{U}_0^n)_s = \sigma_2 (\Delta P) (\delta \mathbf{U}_0^n)_{s2} + \sigma_4 (1 - \Delta P) (\delta \mathbf{U}_0^n)_{s4} \tag{6.8}$$

The pressure switch  $\Delta P$  is used to turn the shock smoothing and the background smoothing on at the appropriate regions. The coefficients  $\sigma_2$ ,  $\sigma_4$  are empirical parameters that control the amount of shock and background smoothing. Their values are the smallest possible for which the method converges.

### 6.2.5 Local Time Stepping

The solution at each node is advanced in time using local time steps. A combination of the CFL and diffusion stability limitations is employed. The viscous-like smoothing term can have appreciable magnitude at shock regions, and therefore it is included in the diffusion limitation. The time-step restriction for the 1-D wave equation is  $\Delta t \leq \Delta x / (|u| + c)$ , while the restriction for the 1-D diffusion equation is  $\Delta t \leq (1/2) / (\Delta x^2 / \nu)$ , where in this case  $\nu = \mu / \rho + \sigma_2 \Delta P$ .

## 6.3 Parallel Mesh Adaptation and Dynamic Load-Balancing

### Module: pADAPT/DRAMA

#### 6.3.1 pADAPT

##### (1) Feature Detector

A dynamic adaptation algorithm developed for 3D unstructured meshes [41,89,90,91] has been parallelized on the GeoFEM parallel platform. 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 across the edges are used for feature detection and threshold parameters are set in order to identify the regions to be refined or coarsened [39,40]. This edge-based treatment is applied for both prisms and tetrahedra. The threshold values for the parameters are set based on the distribution of the parameters which is characterized by the average  $S_{ave}$  and the standard deviation  $S_{sd}$  of the respective parameters, where  $S$  is the detection parameter [39,40]. The following relations are used to set the threshold values for refinement.

$$S_{th} = S_{ave} + \alpha S_{sd} \quad (6.9)$$

The average and the standard deviation are defined as:

$$S_{ave} = \frac{1}{N_{edge}} \sum_{e=1}^{N_{edges}} |S_e|$$

$$S_d = \sqrt{\frac{1}{N_{edge}} \sum_{e=1}^{N_{edges}} (S_e - S_{ave})^2} \quad (6.10)$$

The value of the parameter  $\alpha$  is chosen empirically. The edges that have a detection parameter value greater than the threshold value are flagged to be refined. If very big value of  $\alpha$  is chosen, the grid may not be adapted at all. On the contrary, the grid can be refined through the entire domain for very small  $\alpha$ .



## (2) Prisms

A special type of adaptive refinement of prisms is applied in the present work in order to preserve structure of the mesh along the *normal-to-surface* direction. The detected triangular faces on the surface are divided, as shown in Fig.6.7. Then all prisms above these faces are directionally divided along the *lateral* directions. The cells are not divided along the third direction that is normal to the surface. In this way, grid interfaces within the prisms region are avoided and the structure of the grid along the *normal-to-surface* direction is preserved. Furthermore, such a division is not needed, since the points are distributed along that direction in a way that the viscous stresses are resolved [42]. Therefore, adaptation of the prisms reduces to adaptation of the triangular grid on the surface. The resulting surface triangulation is replicated in each successive layer of the prismatic mesh as illustrated in Fig.6.7. This results in a simpler and less expensive algorithm in terms of storage and CPU time compared to a 3-D adaptation algorithm.

Two types of division are applied. The first divides the triangular faces of the prisms into four smaller triangles (*quadtree*), while the second type divides them into two (*binary*), as shown in Fig.6.7. In the first case, the *parent* triangle is divided into 4 *children*, while it is divided into 2 *children* in the second case. If two edges of the triangle are to be refined, the third is also refined automatically to avoid stretching. Division of cells is also employed to divide *transition* cells at the interface between different embedded regions that contain *hanging* nodes in the middle of some of their edges due to refinement of neighboring cells. Furthermore the position of newly created surface nodes is corrected so that the original geometry of the surface should be kept. In addition, coarsening of the adapted prismatic grid is applied over regions where the embedded cells are no longer needed.

If grids are adapted at multi levels, there could be very *stretched* meshes. To avoid these situations, some rules are defined as follows:

- Only one level refinement/coarsening is allowed at one adaptation stage.
- If the *parent* cell is refined by *binary* division, all three edges should be divided at next refinement (Fig.6.7).
- If the maximum adaptation level difference of neighboring surface triangles around a node is more than 1, the coarser triangles will be refined by three-edge division, as shown in Fig.6.8. In this figure, the maximum adaptation level difference around node C is 2 following the second refinement and the adaptation level difference is reduced by refining the coarsest triangle.

- Grid coarsening is conducted in same manner. If the refinement and coarsening occur in same triangle (different edges), refinement procedure works over coarsening (refinement is always *stronger* than coarsening).

Also, adaptation may yield embedded regions that are slightly smaller than the features which are detected, which results in interfaces being located within or very close to the regions of relatively large gradients. In order to avoid such situations, the algorithm places extra rows (typically two) of embedded cells surrounding the detected regions. This extension of the embedded region is performed as follows:

- (1) All edges with large gradient and/or difference are refined.
- (2) Sweep through all the *active* (not *parent*) triangles and *flag* three edges of each triangle if at least one edge is refined at the current stage of adaptation.
- (3) *Refine* all the *flagged* edges.
- (4) Repeat Steps (2) and (3) number of specified times (typically two) to get sufficiently large embedded region.

To satisfy all of the above rules for grid smoothing, some *iterations* are required. Usually the number of iteration is less than 5.

### (3) Tetrahedra

The tetrahedral elements constitute the area of mesh dominated by inviscid flow features which do not exhibit the directionality as is generally seen in the viscous region. Hence, the tetrahedral meshes are refined isotropically.

The adaptation procedure for tetrahedra is very similar to that of prisms. The feature detector flags edges to be refined/coarsened. Figure 6.9 shows the following three types of tetrahedral cell division:

- One edge is refined. 2 children, *binary*.
- Three edges on the same face are refined. 4 children, *quadtree*.
- All six edges are refined. 8 children, *octree*.

After all edges are flagged, each tetrahedral cell is visited and the flagged edges are counted. Then, the cell is flagged for division according to the above three types. In all cases that are different from the three cases above, the cell is divided according to the

third type of division. If two edges on the same face are refined, the third on the surface is refined according to the second type of division.

To avoid *stretched* mesh, similar rules with prisms are applied. In order to avoid excessive mesh skewness, repeated *binary* and *quadtree* divisions of tetrahedra 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.

#### (4) Prisms/Tetrahedra Interfaces Treatment

The adaptation processes for prisms and tetrahedra are coupled through the outermost triangle surfaces of the prismatic grids, which coincide with tetrahedral triangular faces, as shown in Fig.6.10. The pairs of *interface* cells (between prisms and tetrahedra) are divided if one or both cells are flagged for division. In this way, additional mid-edge nodes are avoided. The procedure is as follows:

- (1) Visit all edges in prismatic region and flag edges to be refined/coarsened.
- (2) Visit all edges in tetrahedra region and flag refined/coarsened edges.
- (3) Visit the interface pairs of prisms/tetrahedra and flag both cells if at least one of them is flagged for division.
- (4) Repeat steps (1)-(3) if required.

Figure 6.11 shows the outline of the parallel mesh adaptation algorithm in pADAPT. Underlined functions use the DRAMA library and its data migration capability developed for this work.

### 6.3.2 DRAMA and Data Migration

The DRAMA library, originally developed within the European Commission funded project with the same name, supports dynamic load-balancing for parallel message passing, mesh-based applications. For a general overview see [8]. The library was evaluated with industrial FE codes and is further developed in ongoing research collaborations.

The core library functions perform a parallel computation of a mesh re-allocation that will re-balance the costs of the application code based on an adjustable, rich cost model. The DRAMA library contains geometric (RCB: Recursive Coordinate Bisection), topological (graph) and local improvement (direct mesh migration) methods and allows to use leading parallel graph partitioning packages such as METIS [138] and JOSTLE [136] through internal interfaces. DRAMA is open source, which is freely downloadable from the web-site in [129].

The DRAMA internal data structures have been designed to be suitable for adaptive applications (i.e. double numbering). The DRAMA library is a load-balancing tool that performs data migration for elements and nodes as described by the DRAMA mesh structure. It supports the application to complete the data migration by old/new and new/old element/node numbering relations. Especially for adaptive codes this is a considerable task involving the reconstruction of the entire grid hierarchy. Routines for mesh conversion and data migration have been developed to integrate the DRAMA library in the pADAPT module of the adaptive GeoFEM environment. The resulting code structure is shown in Fig.6.1

## 6.4 Distributed Data Structures for Parallel Mesh Adaptation

A proper definition of the layout of the distributed data structures is very important for the efficiency of parallel computations with unstructured meshes. Although the epHYBRID code adopts an edge-based formulation, the GeoFEM local data structures described in Chapter 2 which are node-based with overlapping elements [71,72,73,79,81,84,131] has been adopted here. This data structure with internal/external/boundary nodes and communication tables provides excellent parallel efficiency [28,79,81].

Some additional information for mesh adaptation and grid hierarchy has been added to the original static GeoFEM data structure. In order to conform with the DRAMA library interface and the data migration procedure, *double-numbering* of nodes, elements and edges has been implemented where items are identified by 2 types of ID (original partition and local ID) [129], instead of *single-numbering* where global ID for nodes and elements are used. Internal array for element connectivity is changed from 2D type to 1D compressed array with index array because both of prisms and tetrahedra appear in this work and 2D type array is not memory efficient.

Figures 6.12 and 6.13 show the examples of old and new data structure [76].

## 6.5 Examples

Numerical simulations of the supersonic flow ( $M=1.40$ ,  $Re=10^6$ ) around a sphere have been conducted under various types of configurations. In Fig.6.14, a spherical bow shock can be observed upstream the body. It shows the Mach number distribution in very coarse initial meshes, 1-level and 2-level adapted meshes. The shock is very sharply captured by 2-level adapted meshes. Computations are executed on the 32-processor LAMP Pentium cluster [137] operated by NEC-Europe and the 1024-processor Hitachi SR2201 computer at the University of Tokyo, Japan.

In these examples, grid adaptation is required only several times during entire computations. Therefore, computation time for grid adaptation and dynamic load-balancing is almost negligible compared to time for Navier-Stokes simulation. Computational and parallel efficiency of the grid adaptation, dynamic load-balancing and data migration have not been evaluated here.

### 6.5.1 Parallel Performance of epHYBRID without Adaptation

Parallel performance of epHYBRID code was evaluated using globally fine prismatic meshes without adaptation using 2 to 256 processors on both the LAMP cluster and the SR2201 computer. In these computations, the problem size for each processor was approximately kept fixed up to the 48 PE case. Ranging from 48 to 256 PEs, the entire problem size was held constant. GeoFEM's RCB method and MeTiS have been applied as initial static partitioning method.

The results are summarized in Table 6.1. The unit elapsed user execution time (including both computation and communication time) for each iteration stays almost constant up to 256 processor case and parallel efficiency of the epHYBRID is almost perfect.

### 6.5.2 Comparison of Repartitioning Methods (Tetrahedral Grids)

As is described in 6.3, the DRAMA library offers various types of repartitioning methods (for instance : graph-based (PARMeTiS or PJOSTLE) and geometry-based (RCB)). Here, we compare the effect of different repartitioning methods on the computational efficiency of the resulting meshes. The following repartitioning methods in the DRAMA library have been considered:

- No Repartitioning

- PJOSTLE
- PARMETIS k-way
- RCB Simple
- RCB Bucket (edgcut reduced)

The same problem described in 6.5.1 has been solved on 8 or 16 processors with purely tetrahedral meshes. The DRAMA options were set so that the partitioner would balance the number of internal nodes in each partition. Table 6.2.-6.4. show the resulting distributions and the corresponding elapsed time for epHYBRID (averaged for 1,000 steps). Fig.6.15 shows the partitioning after 2-level adaptation for the 8 processor case displayed by the parallel version of GPPView [124] tool developed within the GeoFEM project. Without repartitioning, load imbalance among the processors is severe especially after 2-level adaptation. Among the 4 repartitioning methods, PJOSTLE provided the best quality from the viewpoint of the performance of the epHYBRID code because resulting edge-cuts and edges in each partition are the fewest.

### 6.5.3 Comparison of Repartitioning Methods (Hybrid Grids)

Several cases were computed using hybrid grids on LAMP cluster. Description of the initial grid is as follows (Fig.6.16):

- 1,280 triangles, 642 nodes on the sphere surfaces
- 24 layers. Inner (closer to the sphere surface) 6 layers are for prisms and outer 18 layers are for tetrahedra, totally 76,800 cells (7,680 prisms and 69,120 tetrahedra) and 16,050 nodes.
- Divided into 16 regions by RCB (Recursive Coordinate Bisection).

We compare the effect of different repartitioning methods on the computational efficiency of the resulting meshes. The following repartitioning methods in the DRAMA library have been considered:

- No Repartitioning
- PJOSTLE
- RCB Simple
- RCB Bucket (edgcut reduced)

The DRAMA options were set so that the partitioner would balance the number of

internal nodes in each partition. Table 6.5. show the resulting distributions and the corresponding elapsed time for epHYBRID (averaged for 1,000 steps). Figures 6.17-6.19 show the partitioning after 1-level adaptation for the 16 processor case displayed by the parallel version of GPPView tool. Without repartitioning, load imbalance among the processors is severe after adaptation. Among the 3 repartitioning methods, PJOSTLE provided the best quality from the viewpoint of the performance of the epHYBRID code because resulting edge-cuts and edges in each partition are the fewest.



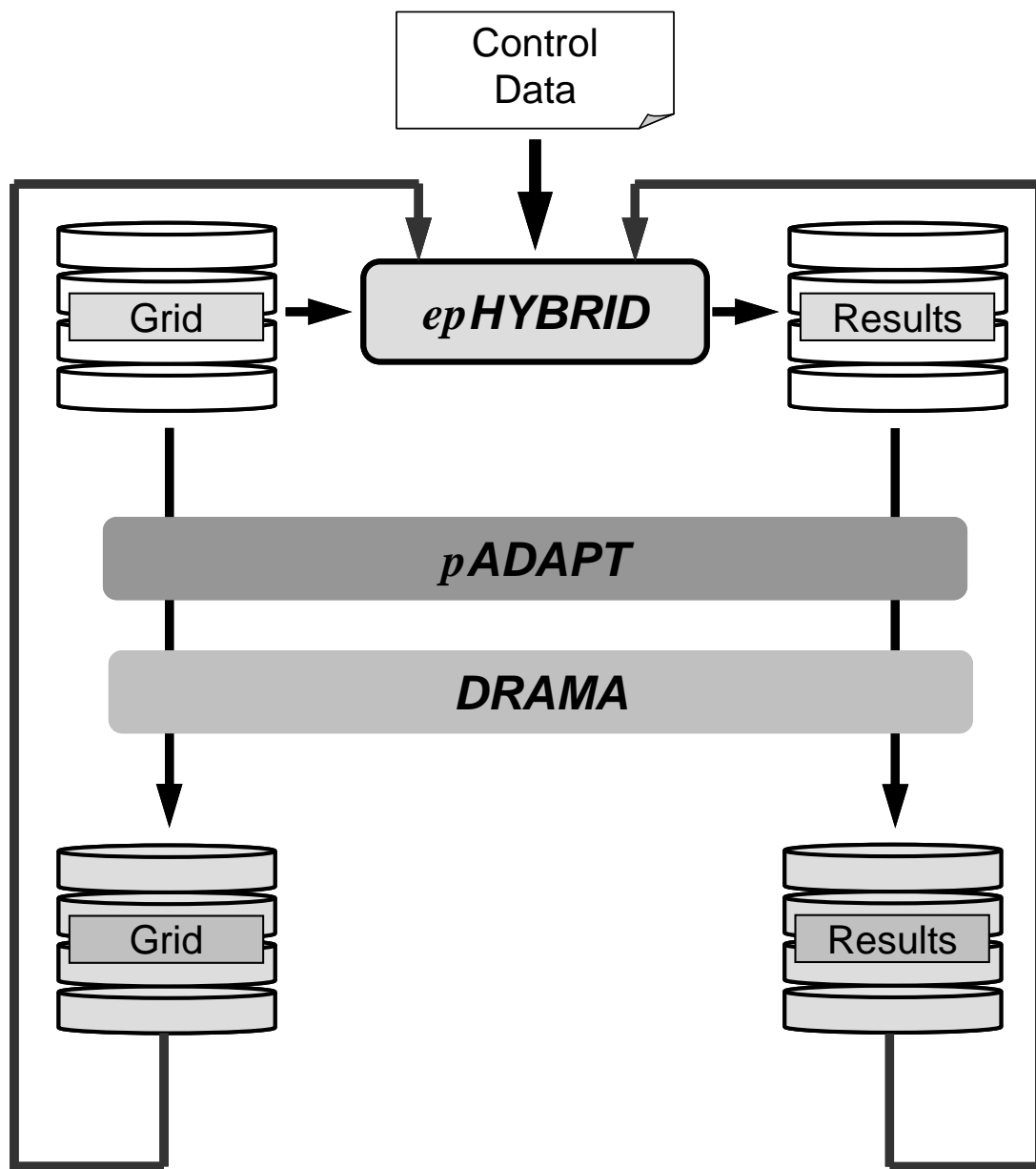
## 6.6 Summary

In this chapter, a parallel 3D compressible Navier-Stokes code with adaptive hybrid meshes (epHYBRID) and parallel mesh adaptation module (pADAPT) have been developed on the GeoFEM parallel platform with an extended data structure for grid adaptation and dynamic load-balancing.

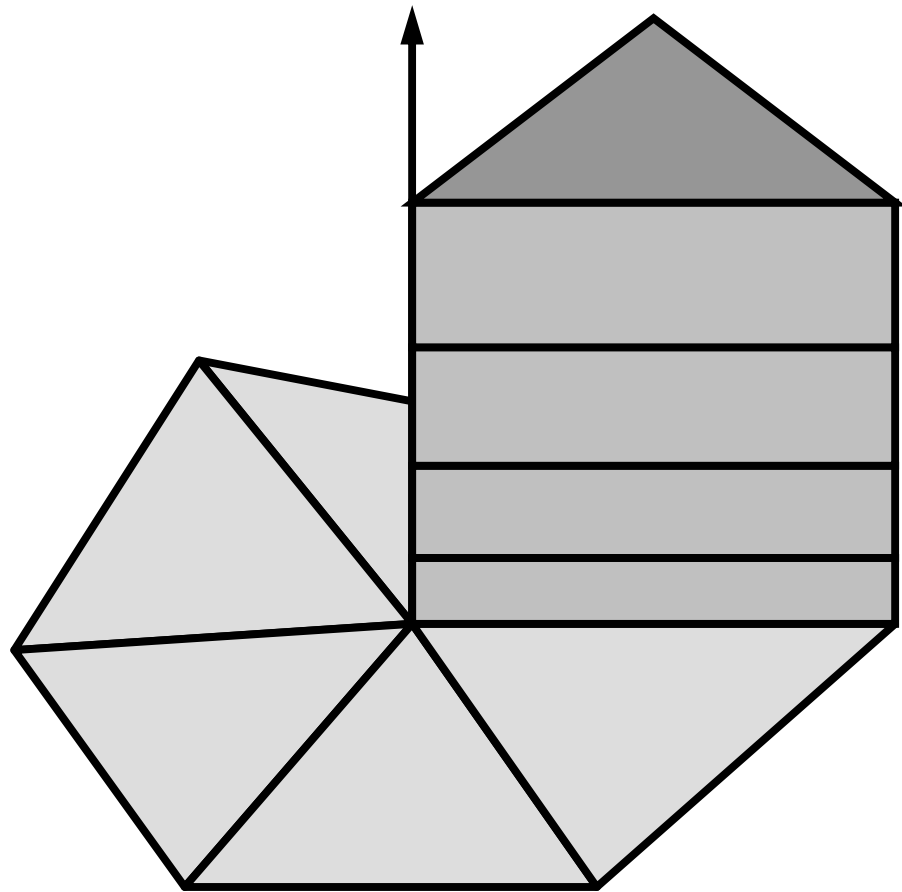
The DRAMA library has been integrated in the pADAPT module and the data migration procedure has been added. The entire code system has been tested with the simulation of the supersonic flow around a spherical body on a Pentium cluster and a Hitachi SR2201 computer. We found that the epHYBRID code with extended the GeoFEM data structure showed excellent parallel efficiency with dynamic load-balancing. Various types of repartitioning methods in the DRAMA library have been evaluated on both purely tetrahedral and hybrid meshes. Among these methods, PJOSTLE provided the best mesh partitioning quality from the viewpoint of the efficiency of the epHYBRID code.

Developed data structure with double-numbering proved to very flexible and efficient for processing distributed local data sets with parallel mesh adaptation and dynamic load balancing.

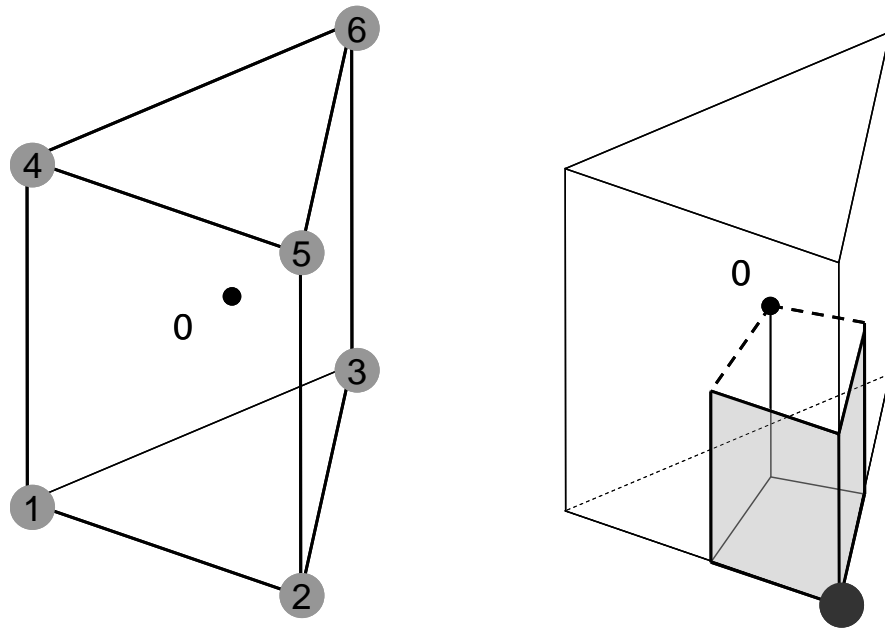




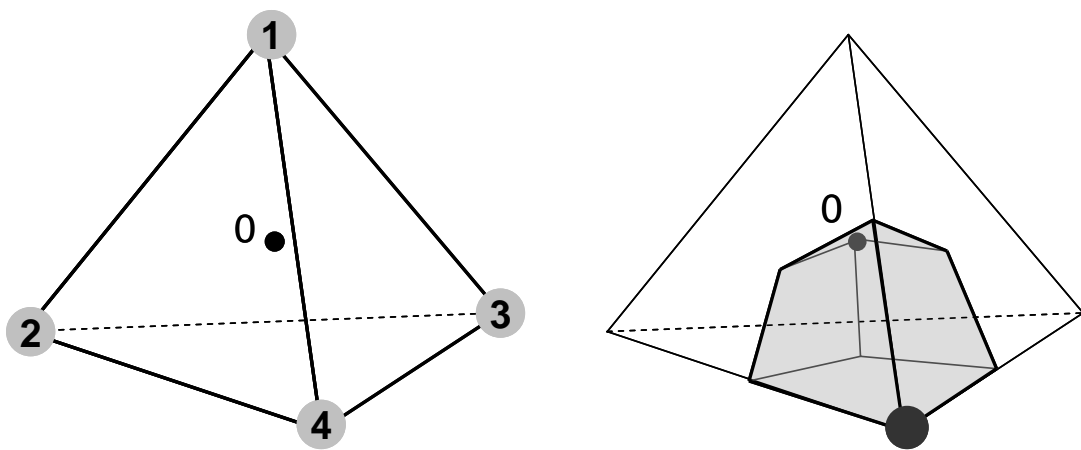
**Fig. 6.1** epHYBRID and pADAPT/DRAMA coupled system [76]



**Fig. 6.2** Prismatic meshes generated from surface triangles in the normal-to-surface direction [42,76,89,91]

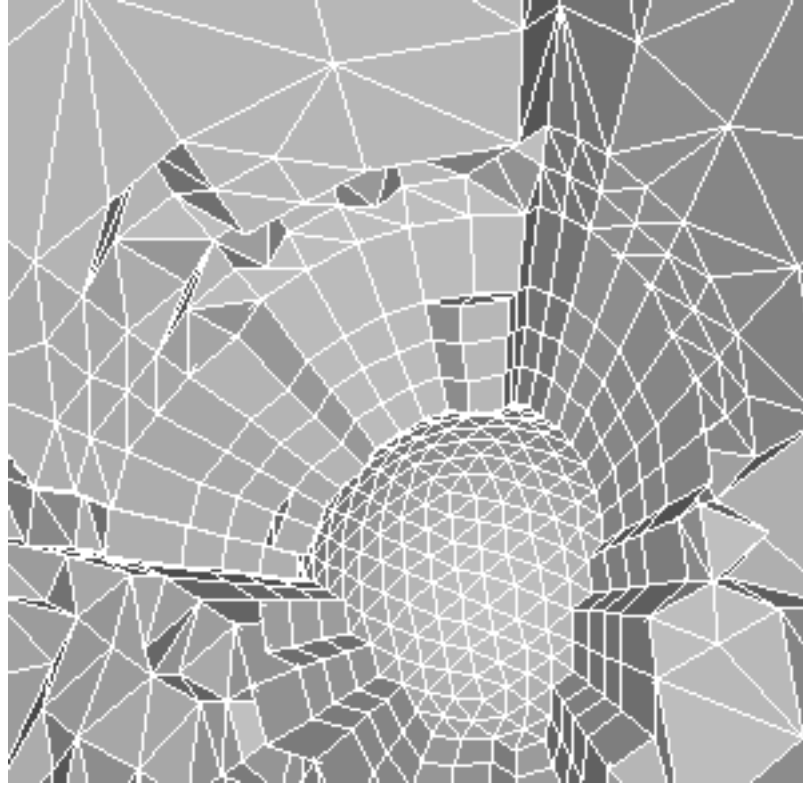


(a) Prism

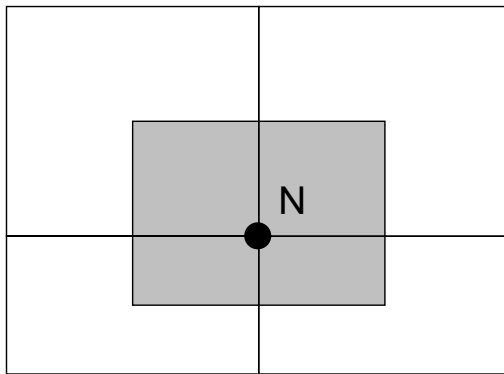


(b) Tetrahedron

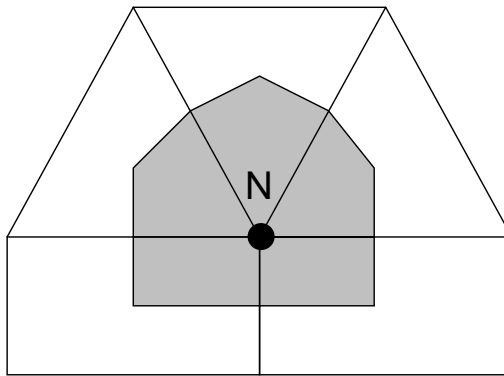
**Fig. 6.3** Prismatic and tetrahedral meshes and dual-cells [42,76,89,91]



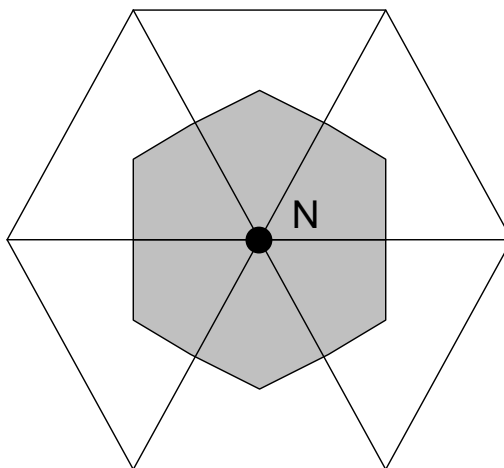
**Fig. 6.4** Example of the prismatic/tetrahedral hybrid meshes [76]



(a) Prismatic Region

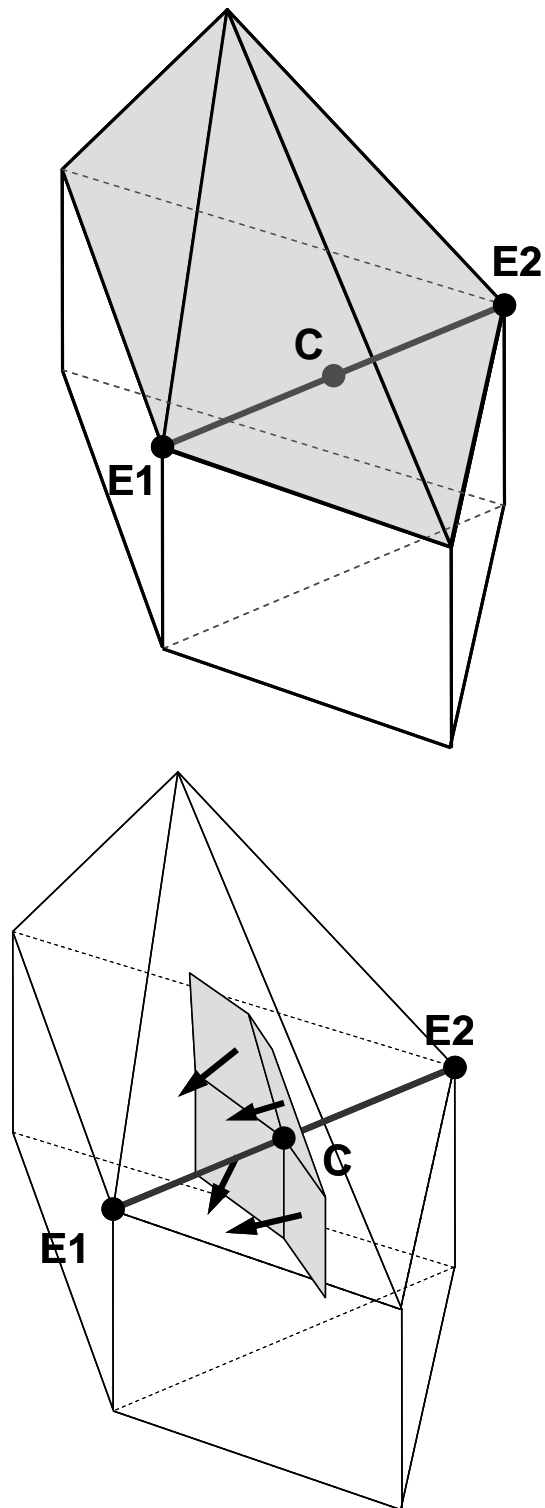


(b) Tetrahedral/Prismatic Interface



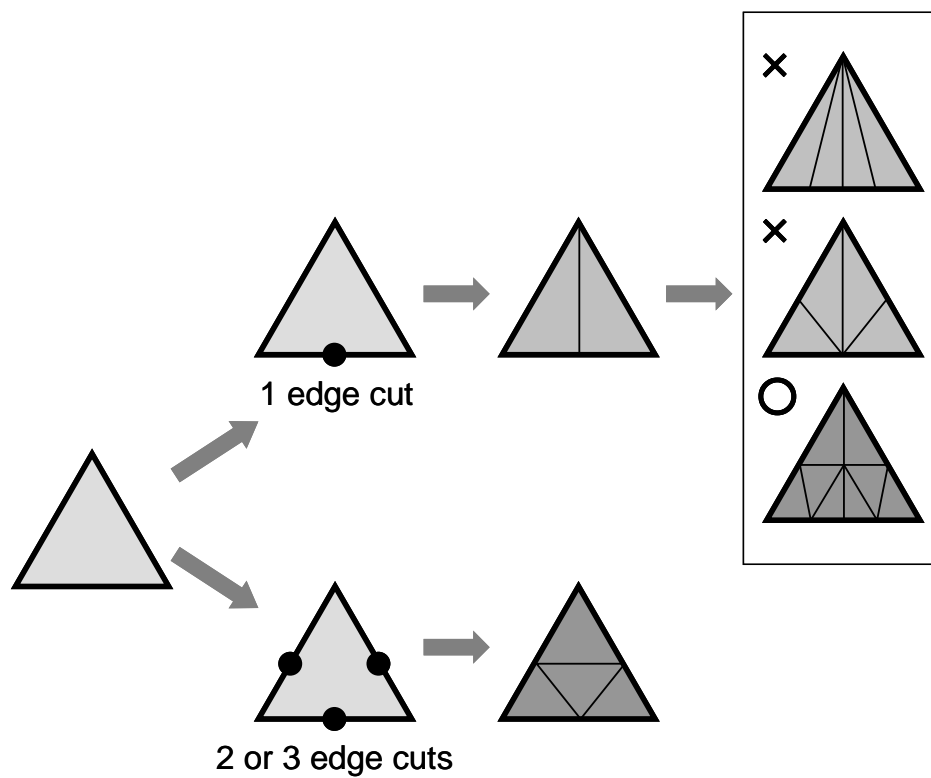
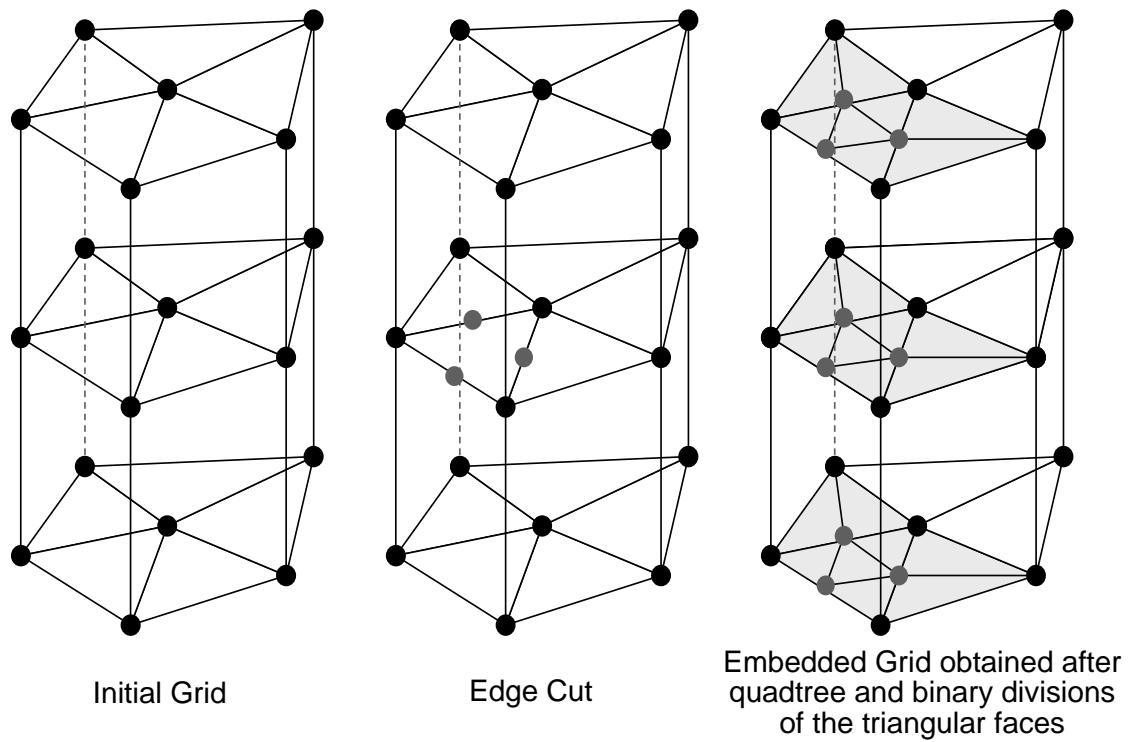
(c) Tetrahedral Region

**Fig. 6.5** Dual volume constructions for mixed-element topology. Two-dimensional analogies for dual mesh around a node in the (a) prismatic region, (b) tetrahedral-prismatic interface and (c) tetrahedral region [91]

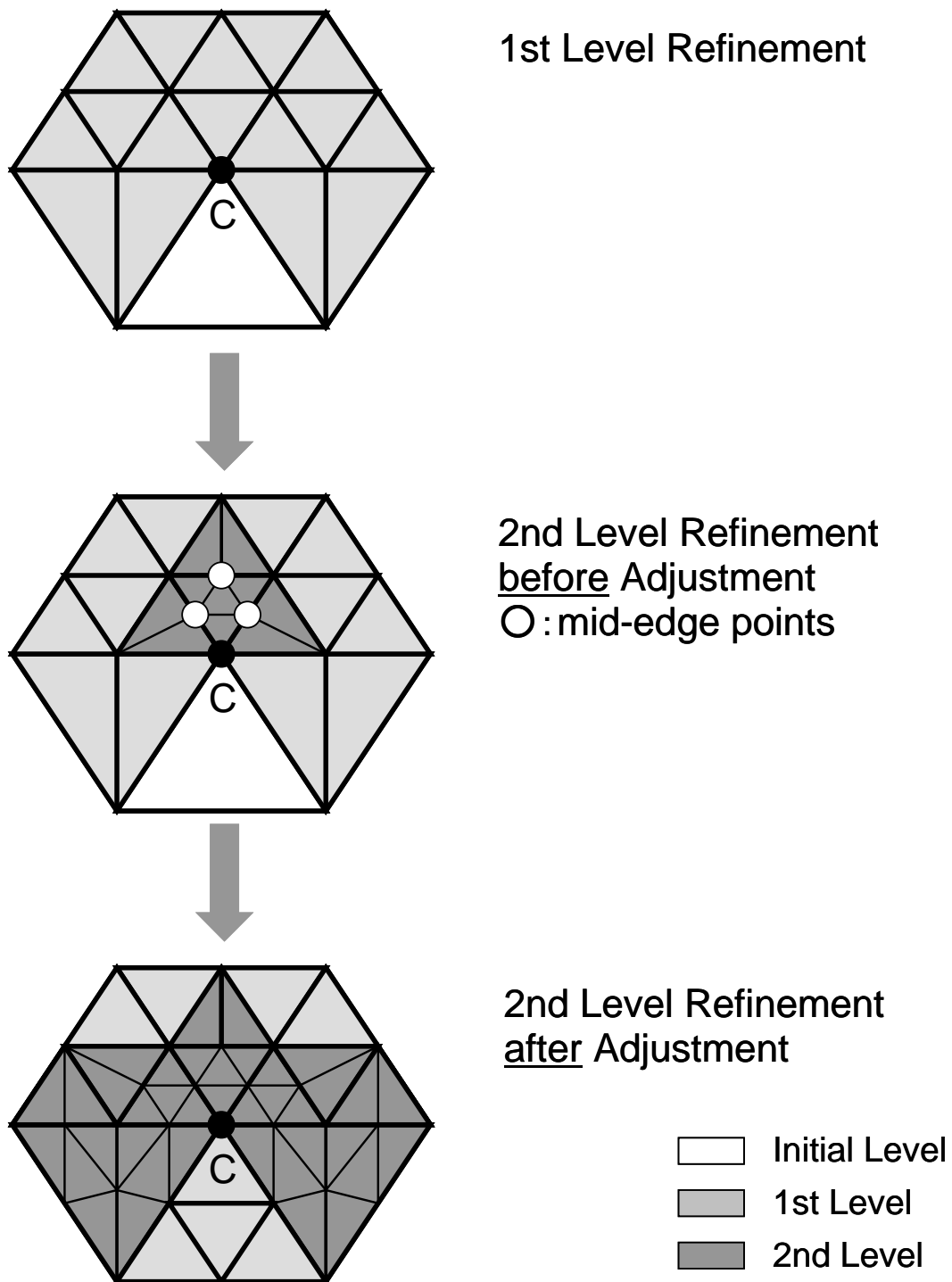


**Fig. 6.6** Edge-dual volume defined around the edges for computing the gradients of primitive variables at the edge centers [76,91]

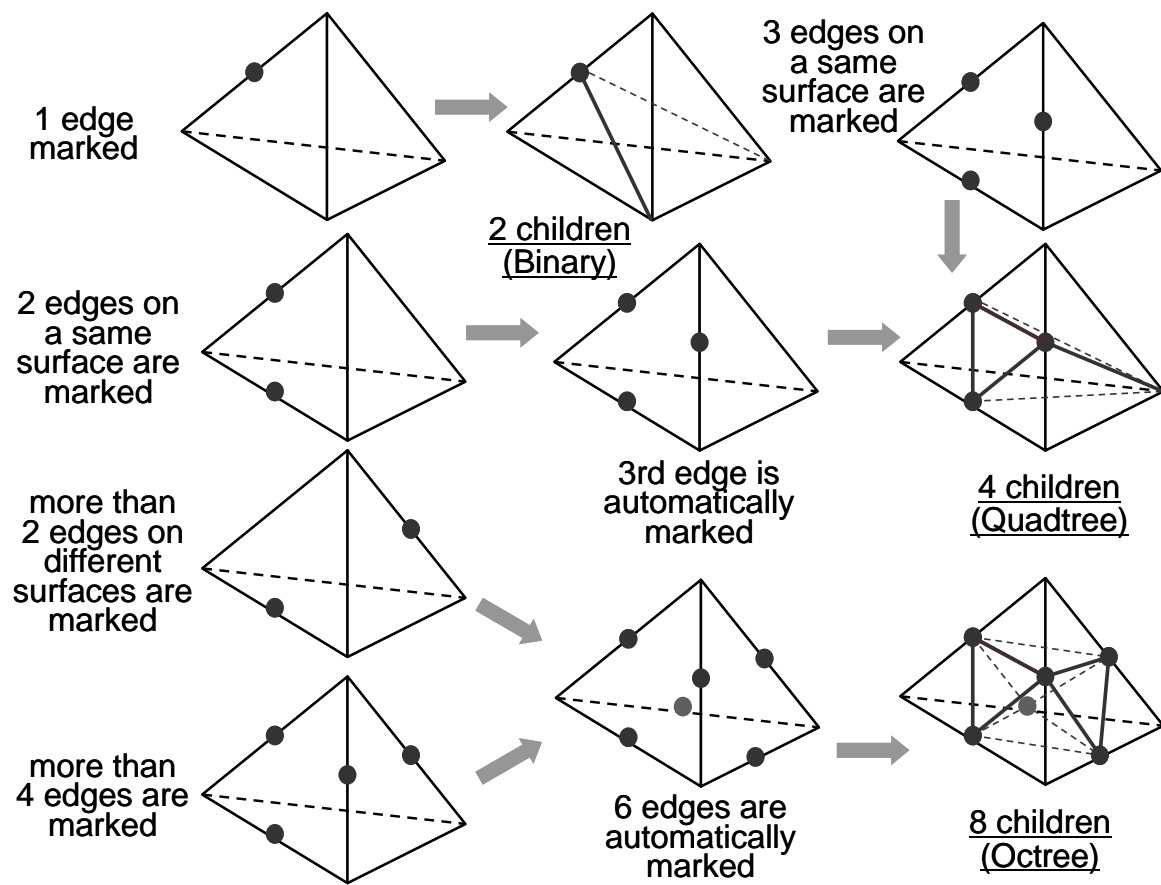




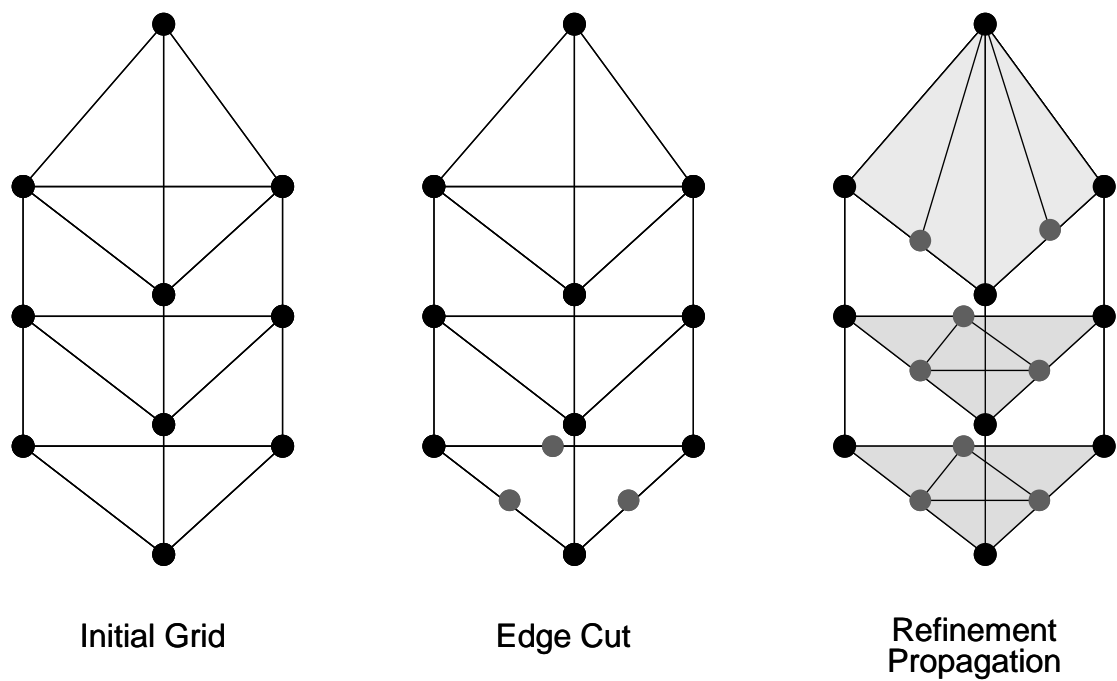
**Fig. 6.7** Directional refinement of prisms based on *quadtree* and *binary* divisions of the triangular faces on the wall [76,89]



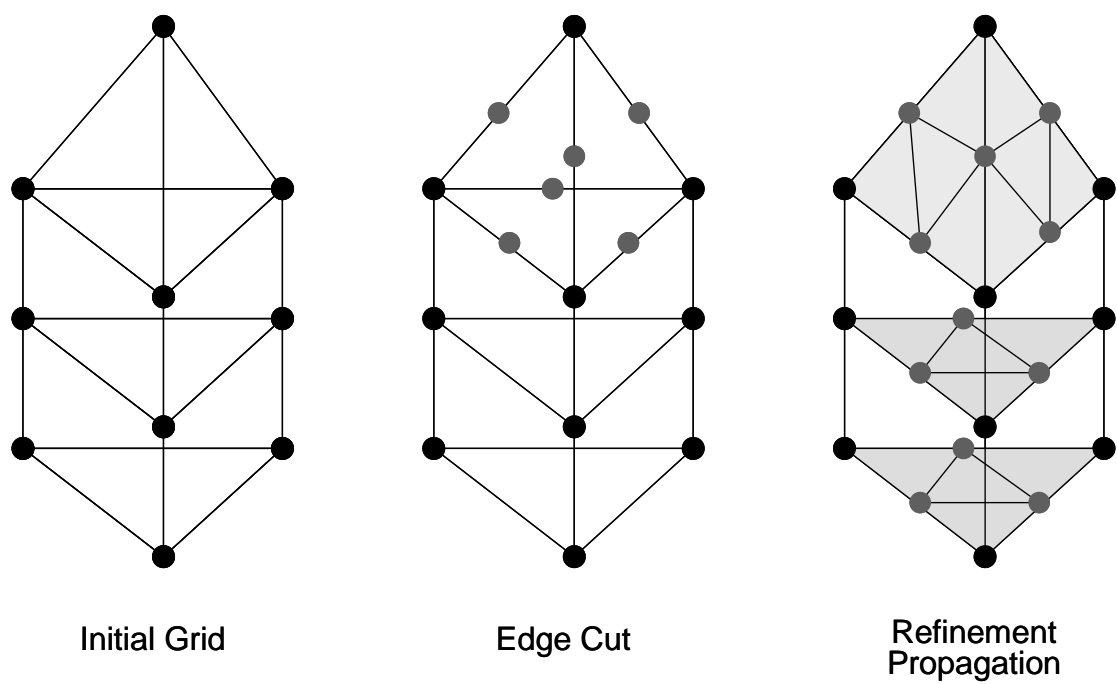
**Fig. 6.8** Procedure in for avoiding sudden changes in mesh size. The mesh refinement algorithm limits the maximum difference in embedding level between neighboring elements to less than 2 [76,89]



**Fig. 6.9** Refinement strategies for a tetrahedron (*binary*, *quadtree* and *octree*) [76,89]



(a) Propagation from prismatic region to tetrahedral region



(b) Propagation from tetrahedral region to prismatic region

**Fig. 6.10** Coupling of prismatic-tetrahedral adaptation at the interface [76,89]

- (1) Pre-Processing
  - reads original grid and result files
  - creates edges
  - defines INTERNAL edges and cells \*
  - creates edge/cell communication tables\*
- (2) Feature Detection
  - computes Velocity gradient/difference across the edges
  - computes average and standard deviation\*
  - MARKs edges which satisfy criterion
- (3) Extend Embedded Zones\*
- (4) Grid Smoothing\*
  - proper embedding patterns
  - adjusts cell embedding level around each node
- (5) New Pointers\*
- (6) New Communication Table\*
- (7) Load Balancing/Repartitioning by DRAMA Library
- (8) Data Migration
- (9) Output

**Fig. 6.11** Parallel mesh adaptation/dynamic load-balancing/data migration procedure in pADAPT/DRAMA coupled system (underlined items are added to the pADAPT module) [76]

- (a) Original data structure of GeoFEM with global node/element ID and 2D array for element connectivity

```

type local_mesh
  integer n_node
  real(kind=kreal),pointer:: node(:, :)
  integer n_elem
  integer,pointer:: elem_type(:)
  integer,pointer:: elem(:, :)
  integer n_internal
  integer,pointer:: global_node_id(:)
  integer,pointer:: global_elem_id(:)
end type local_mesh

```

total node #
node coordinates
total element #
element type
element connectivity
internal node #
global node ID
global element ID

- (b) Extended data structure of GeoFEM with double-numbering for node/element and compressed 1D array for element connectivity.

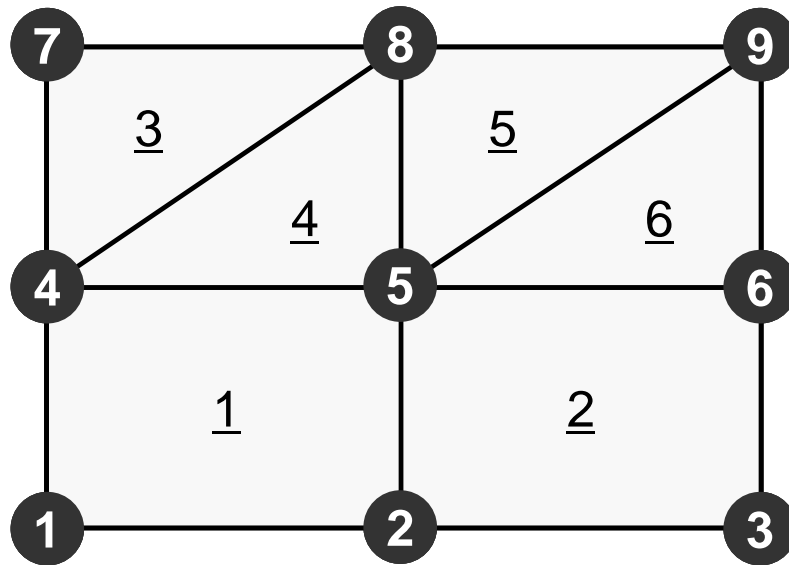
```

type local_mesh_new
  integer n_node
  real(kind=kreal),pointer:: node(:, :)
  integer n_elem
  integer,pointer:: elem_type(:)
  integer,pointer:: index_elem(:)
  integer,pointer:: ptr_elem(:)
  integer n_internal
  integer nelem_internal
  integer,pointer:: elem_internal_list(:)
  integer,pointer:: node_ID (:, 2)
  integer,pointer:: elem_ID (:, 2)
  integer,pointer:: CoarseGridLevels
  integer,pointer:: HOWmanyADAPTATIONS
  integer,pointer:: WhenIwasRefinedN(:)
  integer,pointer:: WhenIwasRefinedE(:)
  integer,pointer:: adapt_type(:)
  integer,pointer:: adapt_level(:)
  integer,pointer:: adapt_parent(:, 2)
  integer,pointer:: adapt_parent_type(:)
  integer,pointer:: adapt_child (:, 2)
  integer,pointer:: index_child(:)
end type local_mesh_new

```

total node #
node coordinates
total element #
element type
1D index for elem. connectivity
1D array for elem. connectivity
internal node #
internal elem.# (homeground)
internal element list (local ID)
home PE & local ID of nodes
home PE & local ID of elem.
how many coarsegrid level
how many adaptations
refinement history for node
refinement history for elem.
elem. refinement pattern
elem. refinement level
parent elem.: home PE & local ID
=(-1,0) if coarsest level
elem. refinement pattern of parent
child elem.: home PE and local ID
index for children

**Fig.6.12** Original and extended data structure of GeoFEM [76,131]



#### **OLD Data Structure**

```

elem_type(1~2)= 221
  elem(1,1~4) = 1,4,5,2
  elem(2,1~4) = 2,5,6,3
elem_type(3~6)= 211
  elem(3,1~3) = 4,7,8
  elem(4,1~3) = 4,8,5
  elem(5,1~3) = 5,8,9
  elem(6,1~3) = 5,9,6

```

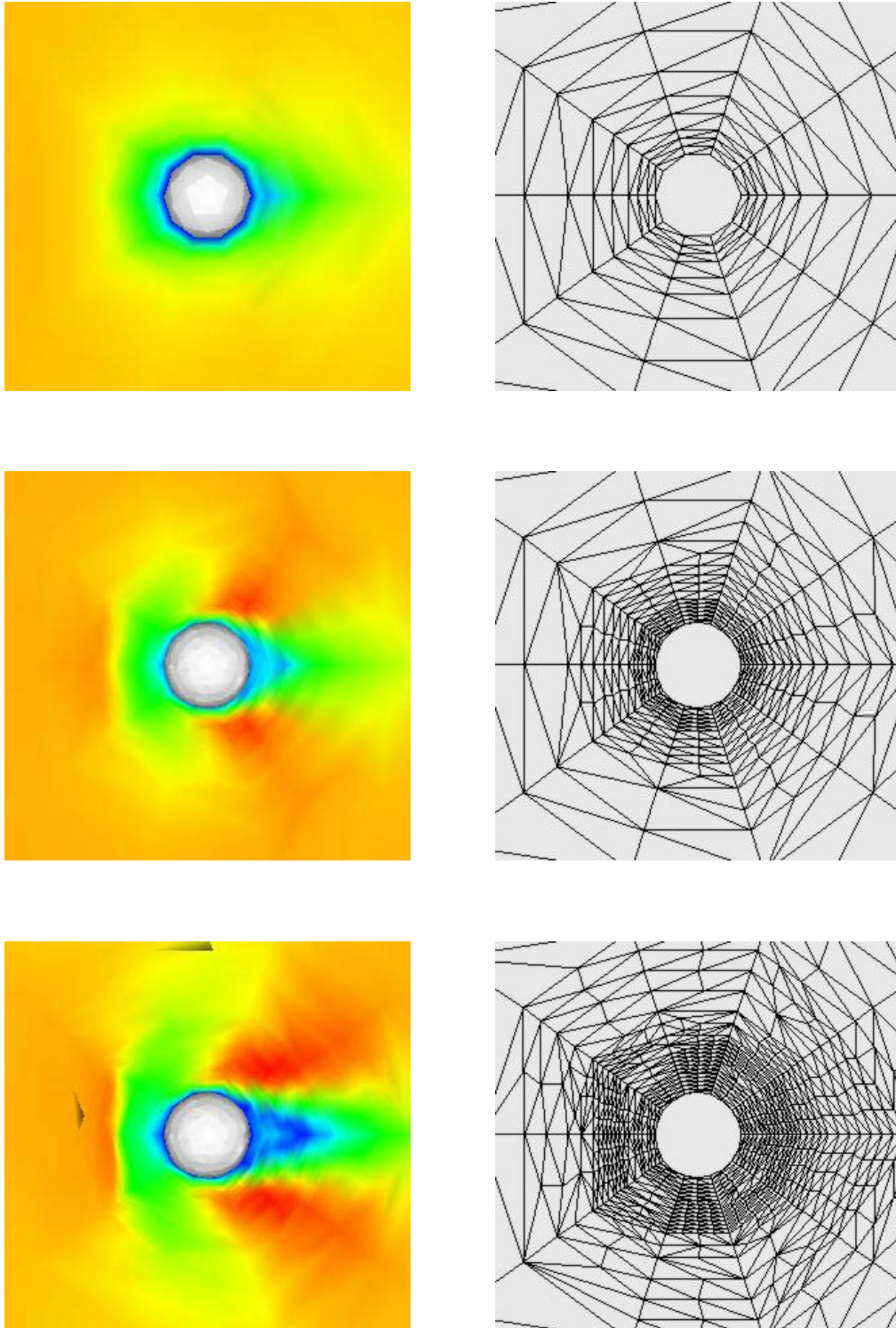
#### **NEW Data Structure**

```

elem_type(1~2)= 221
  index_elem(1)=4, ptr_elem(1~4)= 1,4,5,2
  index_elem(2)=8, ptr_elem(5~8)= 2,5,6,3
elem_type(3~6)= 211
  index_elem(3)=11, ptr_elem( 9~11)= 4,7,8
  index_elem(4)=14, ptr_elem(12~14)= 4,8,5
  index_elem(5)=17, ptr_elem(15~17)= 5,8,9
  index_elem(6)=20, ptr_elem(18~20)= 5,9,6

```

**Fig.6.13** Original and extended data structure of GeoFEM (Example) [76,131]



**Fig. 6.14** Supersonic flow around a spherical body ( $M=1.40$ ,  $Re=10^6$ ). Mach number distribution and meshes (a) Initial mesh (546 nodes, 2,880 tetrahedra), (b) 1-level adapted mesh (2,614 nodes, 16,628 tetrahedra), (c) 2-level adapted mesh (10,240 nodes, 69,462 tetrahedra) [76]



**Table 6.1** Hypersonic flow around a spherical body, 2-256 PE cases with globally fine prismatic meshes on the LAMP cluster system and Hitachi SR2201

PE #	Total Node #	P.M. (1*)	Total Edge Cut #	Max. Internal Node #	Max. Edge #	LAMP Time <sup>(2*)</sup> (μsec.) (Node/Edge)	SR2201 Time <sup>(2*)</sup> (μsec.) (Node/Edge)
2	33,306	R	2,518	16,653	66,553	144.7/36.21	95.48/23.89
4	64,050	R	9,585	16,013	65,799	144.9/35.26	118.0/28.72
8	133,146	R	15,347	16,644	67,268	150.2/37.16	135.8/33.59
16	256,050	R	52,904	16,004	67,048	171.8/41.02	108.7/25.95
32	532,505	R	136,975	16,641	71,306	-	123.2/28.75
48	778,278	M	110,106	16,700	68,399	-	124.6/30.41
64	778,278	M	127,621	12,525	51,735	-	135.7/32.86
80	778,278	M	142,461	10,021	41,765	-	158.7/38.12
128	778,278	M	179,060	6,262	26,251	-	127.8/30.48
256	778,278	M	247,155	3,131	13,458	-	130.9/30.47

(1\*) : Initial partitioning method : R-RCB, M-METIS

(2\*) : Elapsed execution time / step / (internal node or edge)

**Table 6.2** Hypersonic flow around a spherical body, 8 PE cases with 2-level adapted meshes (total : 10,240 nodes, 69,462 tetrahedra) on LAMP cluster system (initial mesh : 546 nodes, 2,880 tetrahedra)

Repartitioning Methods	Internal Node Number (min/max)	Total Edge Cut	Edge Number (min/max)	Time <sup>(1*)</sup> (sec.)
No Repartition	561/2,335	11,224	4,918/17,639	619
PJOSTLE	1,274/1,286	7,293	9,248/9,883	354
PARMeTiS k-way	1,267/1,293	7,679	9,258/10,222	363
RCB Simple	1,280/1,280	12,106	10,426/10,605	389
RCB Bucket	1,280/1,280	11,603	10,479/10,971	399

(1\*) : Elapsed execution time for 1,000 time steps (averaged)

**Table 6.3** Hypersonic flow around a spherical body, 16 PE cases with 1-level adapted meshes (total : 47,074 nodes, 306,236 tetrahedra) on LAMP cluster system (initial mesh : 16,050 nodes, 92,160 tetrahedra)

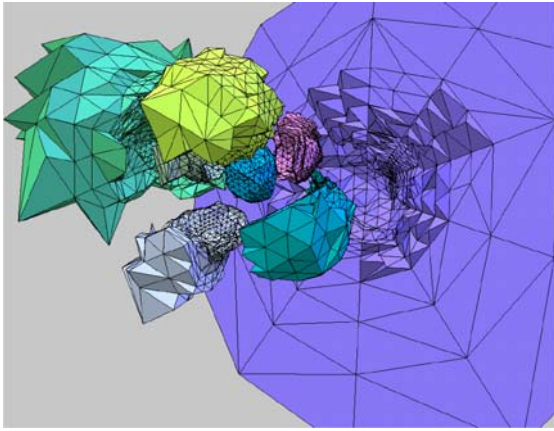
Repartitioning Methods	Internal Node Number (min/max)	Total Edge Cut	Edge Number (min/max)	Time <sup>(1*)</sup> (sec.)
No Repartition	1,343/6,351	39,888	10,576/48,495	1,683
PJOSTLE	2,929/2,961	25,085	21,089/22,233	874
PARMETIS k-way	2,905/2,984	26,274	21,201/22,630	880
RCB Simple	2,942/2,943	41,980	22,520/23,090	899
RCB Bucket	2,942/2,943	37,192	21,231/23,269	926

(1\*) : Elapsed execution time for 1,000 time steps (averaged)

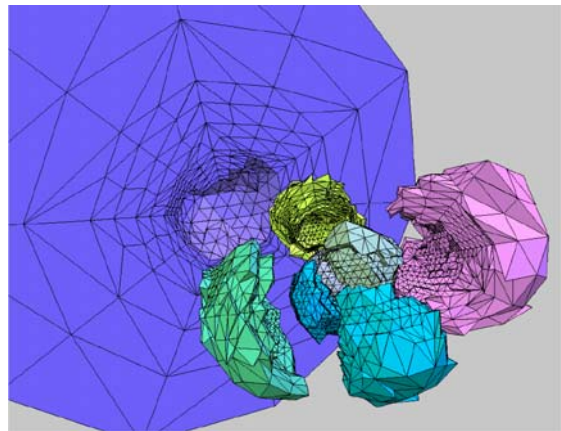
**Table 6.4** Hypersonic flow around a spherical body, 16 PE cases with 2-level adapted meshes (total : 163,537 nodes, 1,116,700 tetrahedra) on LAMP cluster system (initial mesh : 16,050 nodes, 92,160 tetrahedra)

Repartitioning Methods	Internal Node Number (min/max)	Total Edge Cut	Edge Number (min/max)	Time <sup>(1*)</sup> (sec.)
No Repartition	6,621/20,842	101,178	50,386/152,059	5,384
PJOSTLE	10,195/10,260	55,663	73,262/ 75,540	2,982
RCB Bucket	10,221/10,222	100,462	82,799/ 85,819	3,227

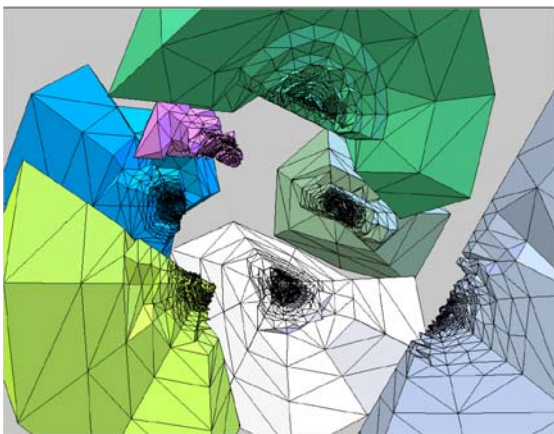
(1\*) : Elapsed execution time for 1,000 time steps (averaged)



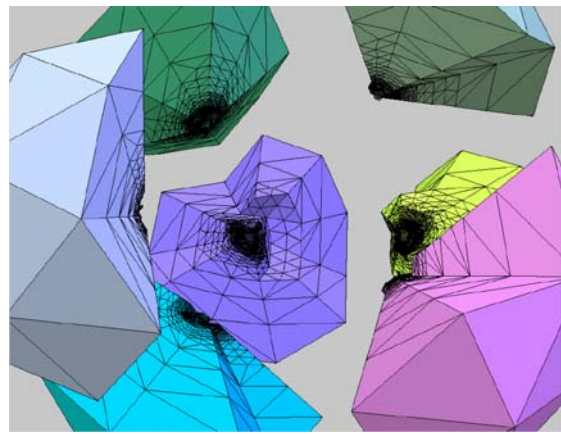
(a) PJOSTLE



(b) PARMETIS k-way

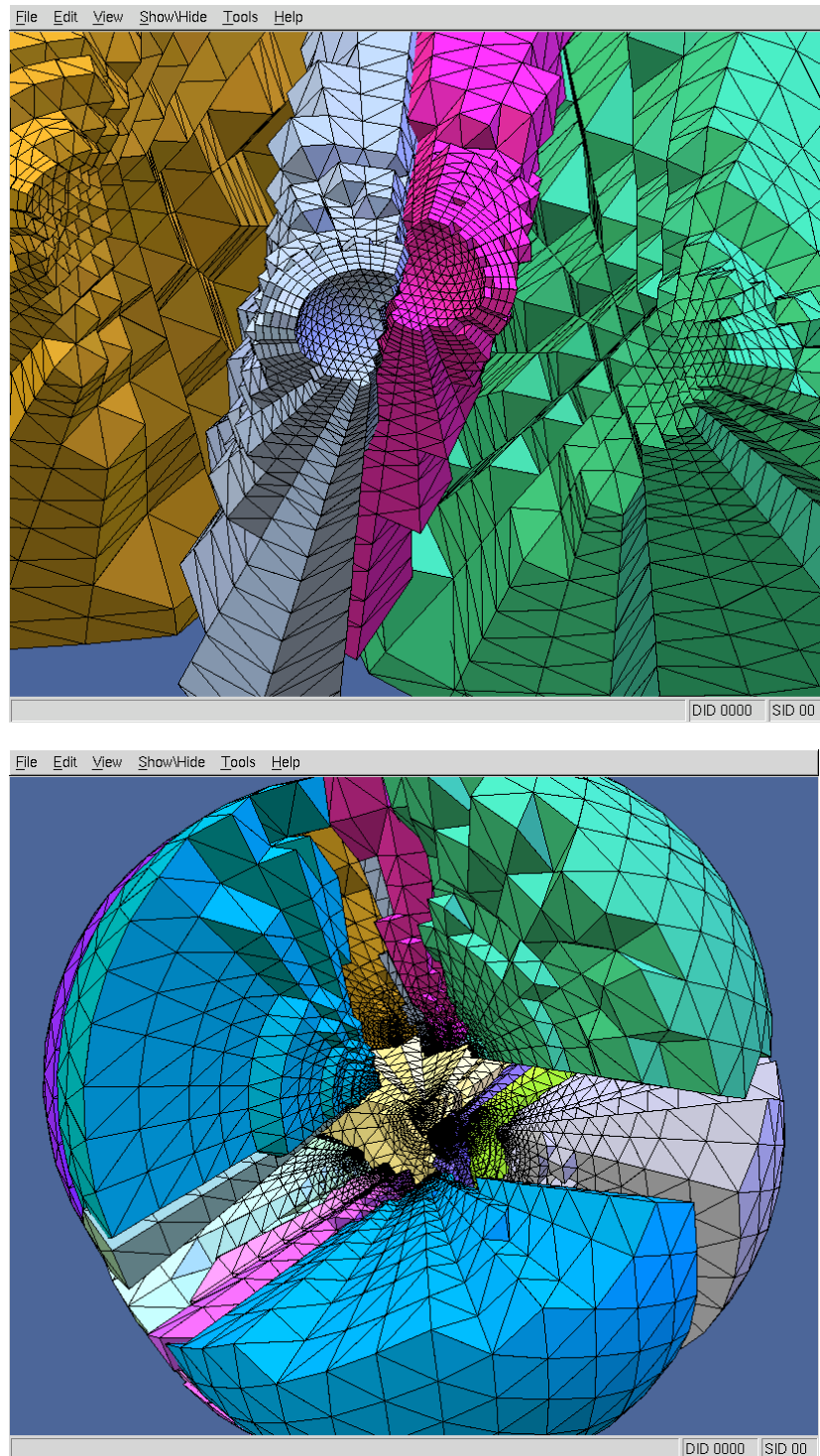


(c) RCB Bucket



(d) No Repartitioning

**Fig. 6.15** Repartitioned domains after 2-level adaptation with 8 processors (total : 10,240 nodes, 69,462 tetrahedra) displayed by GPPView [131] (a)PJOSTLE (b) PARMETIS k-way (c)RCB Bucket and (d) No Repartitioning (each partition is separately shown)

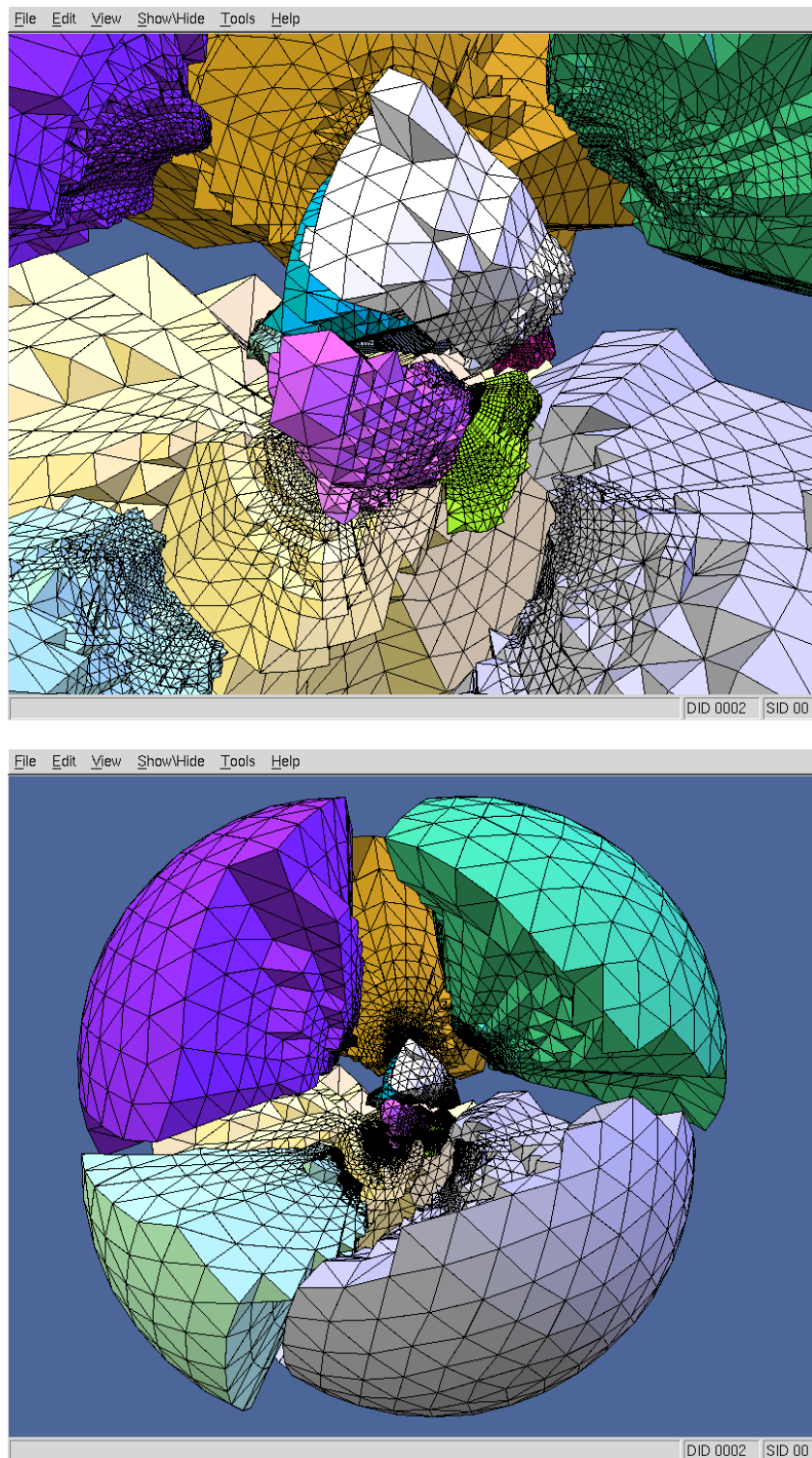


**Fig. 6.16** Initial hybrid grid with 16 partitions. 1,280 triangles, 642 nodes on the sphere surfaces. 24 layers, inner (closer to the sphere surface) 6 layers are for prisms and outer 18 layers are for tetrahedra, totally 76,800 cells (7,680 prisms and 69,120 tetrahedra) and 16,050 nodes.

**Table 6.5** Hypersonic flow around a spherical body, 16 PE cases with 1-level adapted meshes (total : 60,575 nodes, 336,660 cells) on LAMP cluster system (initial mesh : 16,050 nodes, 76,800 cells)

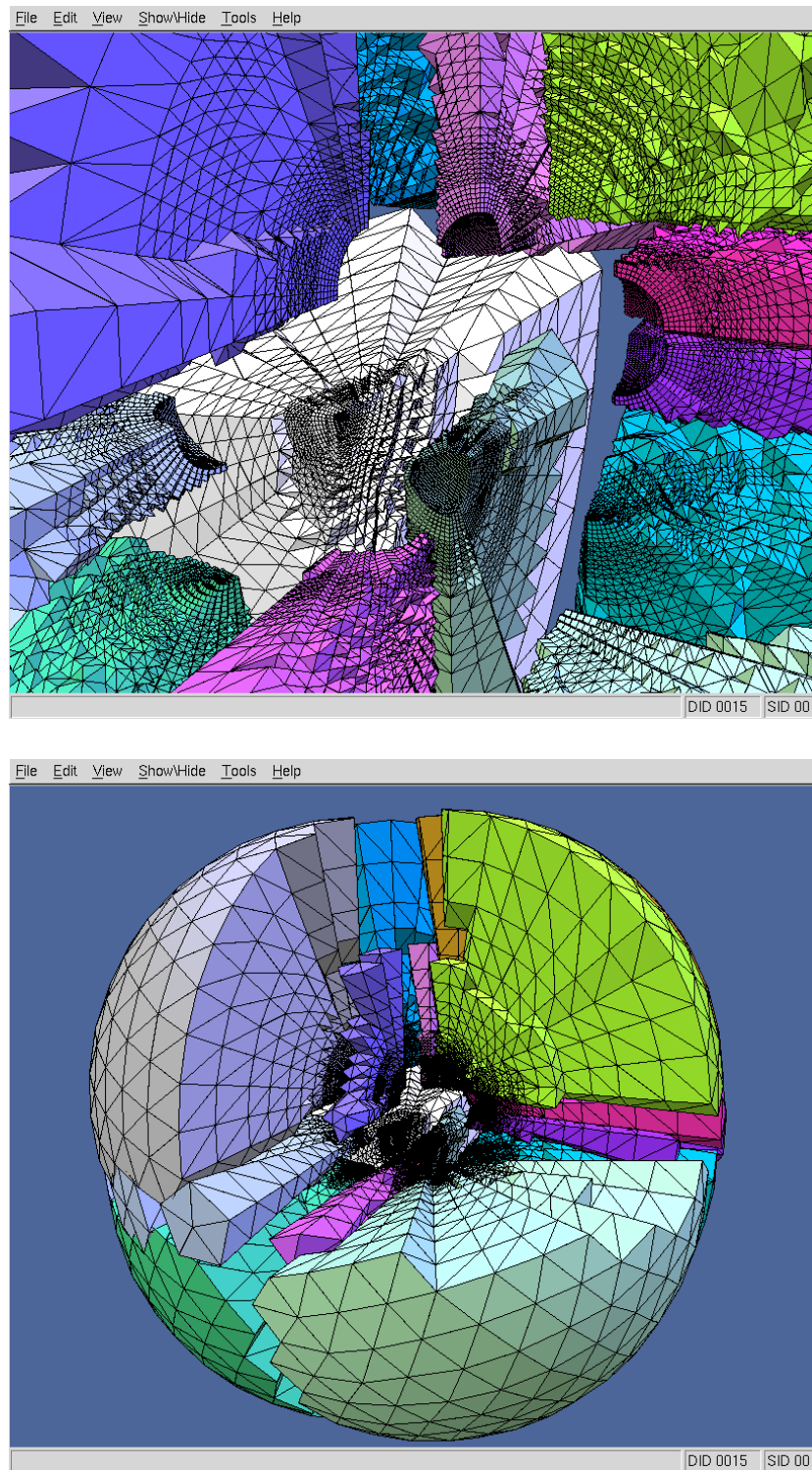
Repartitioning Methods	Internal Node Number (min/max)	Total Edge Cut	Edge Number (min/max)	Time <sup>(1*)</sup> (sec.)
No Repartition	1,992/6,029	42,601	15,612/37,922	1,396
PJOSTLE	3,772/3,810	31,439	20,297/28,418	992
RCB Simple	3,785/3,786	43,191	23,405/28,249	1,030
RCB Bucket	3,785/3,786	39,030	21,609/29,552	1,080

(1\*) : Elapsed execution time for 1,000 time steps (averaged)

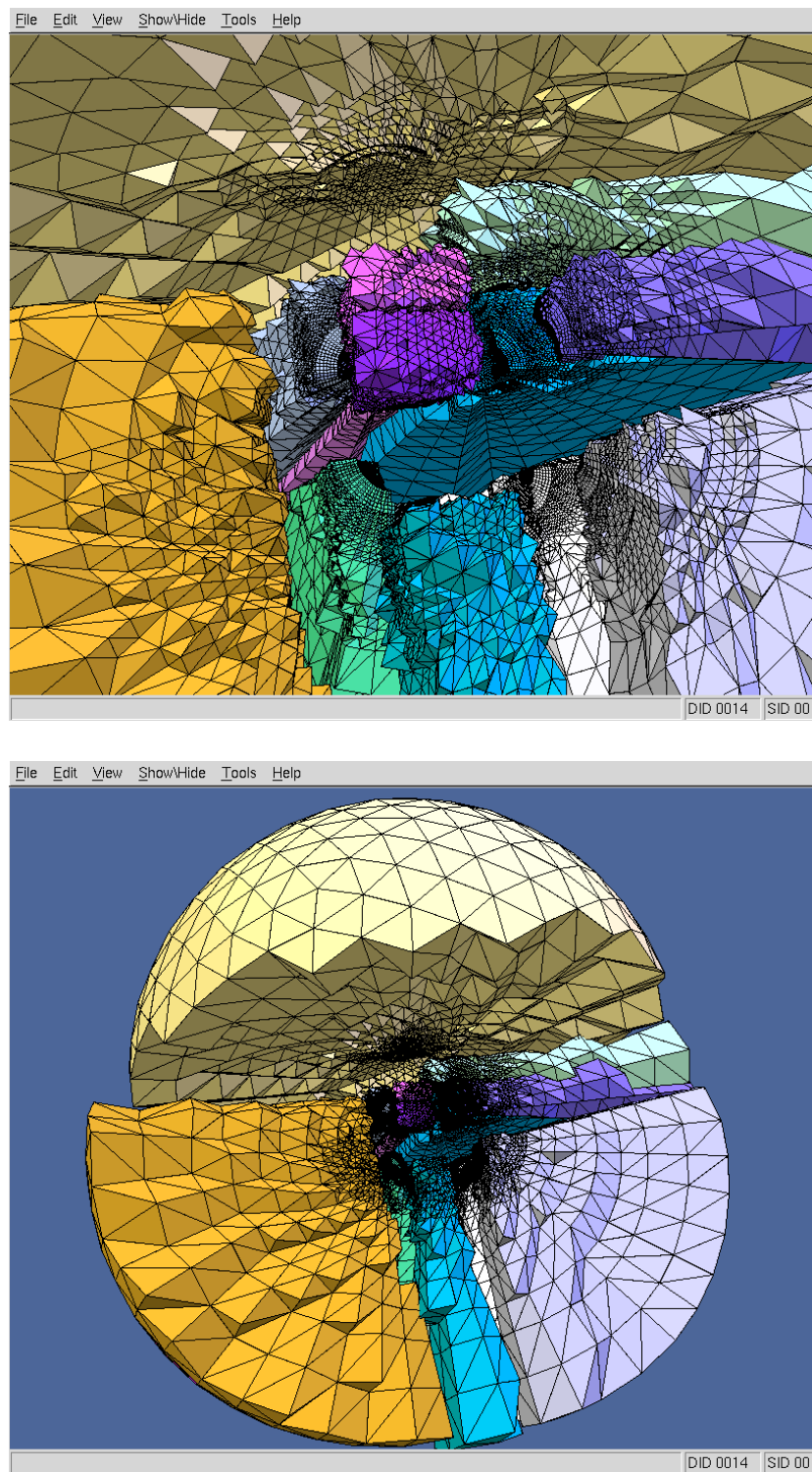


**Fig. 6.17** Repartitioned domains after 1-level adaptation with 16 processors by PJOSTLE (total : 60,575 nodes, 336,660 cells) displayed by GPPView [131] (each partition is separately shown)





**Fig. 6.18** Repartitioned domains after 1-level adaptation with 16 processors by RCB simple (total : 60,575 nodes, 336,660 cells) displayed by GPPView [131] (each partition is separately shown)



**Fig. 6.19** Repartitioned domains after 1-level adaptation with 16 processors by RCB bucket (total : 60,575 nodes, 336,660 cells) displayed by GPPView [131] (each partition is separately shown)



## **Chapter 7     Concluding Remarks**

This chapter presents summary and the main conclusions of the thesis, as well as recommendations for future work.

## 7.1 Summary of the Thesis

### Chapter 2

In Chapter 2, outline of local data structure according to node-based partitioning and parallel iterative solvers with localized preconditioning in GeoFEM was described. Well-designed local data structures with communication tables provide highly parallel efficiency that is greater than 95% for up to 1024 PEs on a Hitachi SR2201. The localized preconditioning method was shown to be stabilized by ASDD (additive Schwarz domain decomposition).

### Chapter 3

Chapter 3 describes general preconditioning methods, corresponding to category (I) in Section 1.2. In this chapter, an efficient parallel iterative method for unstructured grids was developed for the GeoFEM platform on SMP cluster architectures with vector processors such as the Earth Simulator. The method employs a *three-level hybrid* parallel programming model consisting of the three level hierarchy, MPI for Inter-SMP node, OpenMP for Intra-SMP node and vectorization for individual PE.

Simple 3D linear elastic problems with more than  $2.2 \times 10^9$  DOF were solved by  $3 \times 3$  block ICCG(0) with additive Schwarz domain decomposition and PDJDS/CM-RCM reordering on 176 SMP nodes of the Earth Simulator, achieving a performance of 3.80 TFLOPS (33.7% of peak performance). PDJDS/CM-RCM reordering provides excellent vector and parallel performance on SMP nodes. Without reordering, parallel processing of forward/backward substitution in IC/ILU factorization was impossible due to global data dependencies even in the simple examples in this study. Although the three-level hybrid and flat MPI parallel programming models offer similar performance, the hybrid programming model outperforms flat MPI in problems with a large number of SMP nodes.

The performance of PDJDS/CM-RCM reordering was also compared with PDJDS/MC. In a simple cubic geometry, PDJDS/CM-RCM usually converges faster than PDJDS/MC. However, when complicated geometries are involved with a large number of hyperplanes, in which case it is difficult to construct independent sets with sufficient loop lengths by CM, PDJDS/MC provides better performance in terms of GFLOPS rate and CPU time by guaranteeing sufficient loop length, even though PDJDS/CM-RCM requires fewer iterations for convergence.

The most appropriate reordering method should therefore be selected based on the

length of each hyperplane generated by RCM reordering.

## Chapter 4

Chapter 4 describes a problem-specific preconditioning method, corresponding to category (III) in Section 1.2. In this chapter, robust preconditioning and partitioning methods were developed for the simulation of fault-zone contact with penalty constraints using parallel computers. For symmetric positive definite matrices, block incomplete Cholesky factorization without inter-block fill-in, using *selective blocking* (SB-BIC(0)) has excellent performance, memory efficiency and robustness for a wide range of penalty parameter values even if meshes are distorted. Spectral condition number  $\kappa$  ( $\kappa = E_{\max}/E_{\min}$  where  $E_{\max}$  and  $E_{\min}$  are the largest and smallest eigenvalues, respectively, of  $[M]^{-1}[A]$ ) is a helpful parameter for the evaluation of convergence of the preconditioning methods. Usually, BIC(1) and BIC(2) requires fewer iterations for convergence than SB-BIC(0). However, the total computation time for SB-BIC(0) is lower as a result of the lower cost per iteration.

It is also shown that the partitioning method for elimination of edge-cuts in contact groups with load-balancing improves the convergence of parallel iterative solvers with localized preconditioning.

Parallel performance of the CG method with SB-BIC(0) preconditioning was evaluated using 16 to 128 PEs of a Hitachi SR2201 at the University of Tokyo using a flat MPI parallel programming model. Although the iteration number for convergence increases according to PE number due to locality of the preconditioner, this increase is only 11% from 16 PEs to 128 PEs and the speed-up ratio based on elapsed execution time including communication for 128 PEs, is higher than 120, as extrapolated from results for 16 PEs.

Furthermore, the developed method is vectorized and parallelized using OpenMP directives on one SMP node of the Earth simulator, and provides robust and smooth convergence and excellent parallel performance for both simple and complicated geometries with contact conditions.

The reordering method for SMP cluster architectures with vector processors described in Chapter 3 has been implemented to the selective blocking preconditioning using the MC reordering method. Special treatments for selective blocking, such as the introduction of dummy elements and the reordering of selective blocks according to block size, were implemented.

In cases involving several colors, fewer iterations are required for convergence, but the performance is worse due to the smaller loop length and greater overhead. In the

complicated Southwest Japan model, the number of iterations for convergence is not affected by the number of colors because there are many distorted elements in this model and the coefficient matrices are ill-conditioned.

Performance of 17.6 GFLOPS (27.5% of peak performance) for the simple block model and 18.6 GFLOPS (29.1% of peak performance) for the Southwest Japan has been obtained. Performance is about 60% if the reordering of selective blocks is not applied. The load-imbalance among PEs on the SMP node and the ratio of dummy off-diagonal components are not significant.

## Chapter 5

Chapter 5 describes multigrid preconditioner for Poisson equations, corresponding to category (II) in Section 1.2. A multigrid-preconditioned conjugate gradient iterative method for parallel computers has been developed, in which a V-cycle and semi-coarsening approach is adopted for the multigrid procedure. Extended local data structure based on that of GeoFEM has been developed for the multilevel parallel procedure. Two types of communication tables, one for node-based variables and the other for cell-based variables, have been defined. Both Gauss-Seidel and ILU(0) with additive Schwartz domain decomposition smoothers have been tested. Various combinations of parallel and serial smoothers have been applied. The proposed procedure was applied to Poisson equations in the region between two spherical surfaces on adaptively generated semi-unstructured prismatic grids under various boundary conditions. Computational results obtained on a Hitachi SR2201 parallel computer using up to 128 processors demonstrate the good scalability of the method, as compared to ICCG solvers. Excellent parallel performance provided by the developed data structure is also demonstrated.

Among the tested methods, MGCG/FGS (Full-Gauss-Seidel) provides the best performance up to 32 PEs, while MGCG/ILU-GSp (ILU-Gauss-Seidel-Parallel, parallel Gauss-Seidel is applied for the coarsest level of the grid) is relatively robust for computations across many PEs, although parallel performance is worse for cases involving many PEs due to the communications overhead of the single-stage parallel Gauss-Seidel procedure. In the cases with clustered mesh spacing in the radial direction, MGCG/ILU-GSp provided more very convergence compared to other methods. Generally, ILU-type smoothers provide more robust convergence than Gauss-Seidel-type smoothers, especially for ill-conditioned problems.

The proposed procedure was also applied to grids with local refinement, and 2 multigrid strategies (*direct jump* and *level-by-level*) were compared. The *direct jump*

method developed in this study was found to be much more efficient than the *level-by-level* method described in [11] for deeper-level adaptation despite the simplicity of the level-by-level method.

Finally, the proposed method was applied to 3D Navier-Stokes equations with thermal convection. CG solvers with multigrid preconditioning (MGCG/FGS and MGCG/ILU-GSp) provided much better performance than ICCG.

## Chapter 6

In Chapter 6, a parallel 3D compressible Navier-Stokes code with adaptive hybrid meshes (epHYBRID) and parallel mesh adaptation module (pADAPT) have been developed on the GeoFEM parallel platform with an extended data structure for grid adaptation and dynamic load-balancing. This data structure with double-numbering proved to very flexible and efficient for processing distributed local data sets with parallel mesh adaptation and dynamic load balancing.

The DRAMA library has been integrated in the pADAPT module and the data migration procedure has been added. The entire code system has been tested with the simulation of the supersonic flow around a spherical body on a Pentium cluster and a Hitachi SR2201 computer. We found that the epHYBRID code with extended GeoFEM data structure showed excellent parallel efficiency with dynamic load-balancing. Various types of repartitioning methods in the DRAMA library have been evaluated on both purely tetrahedral and hybrid meshes. Among these methods, PJOSTLE provided the best mesh partitioning quality from the viewpoint of the efficiency of the epHYBRID code.

## 7.2 Conclusions of the Thesis

In many large-scale scientific simulation codes using the finite-element method (FEM) and the finite-difference method (FDM), most computation is spent in solving linear equations with sparse coefficient matrices. For this reason, much of the scalable algorithm research and development is aimed at solving these large, sparse linear systems of equations on parallel computers. Sparse linear solvers can be broadly classified as either *direct* or *iterative*. Iterative methods are much more memory scalable than direct methods and are more suitable for parallel computing. But their convergence can be slow or they can fail to converge. The rate of convergence of iterative methods depends strongly 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*. The use of a good preconditioner improves the convergence of the iterative methods, sufficiently to overcome the extra cost of constructing and applying the preconditioner. Indeed, without a preconditioner the iterative method may even fail to converge.

In this thesis, the following three types of preconditioners of parallel iterative solvers for various types of applications on unstructured meshes using the GeoFEM platform for parallel finite-element methods:

- (I) **Localized block ILU(0) preconditioning method for 3D solid mechanics on SMP cluster type vector parallel computers, such as the Earth Simulator (Category I in Section 1.2, general preconditioners).**
- (II) **Parallel scalable multigrid preconditioning method for 3D Poisson equations derived from incompressible Navier-Stokes solvers with adaptive meshes (Category II in Section 1.2, preconditioners for broad classes of underlying problems).**
- (III) **Selective blocking preconditioning method for 3D solid mechanics with contact on SMP cluster type vector parallel computers (Category III in Section 1.2, preconditioners for specific problems).**

A proper definition of the layout of the distributed data structures is very important for the efficiency of parallel computations with unstructured meshes. Local distributed data structure of GeoFEM provides excellent parallel performance over 95%, even if the number of processors is over 1,000.

In order to achieve efficient parallel/vector computation for applications with unstructured grids, the following three matters 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 for achieving highly parallel performance and vector performance. In this study, various ordering methods have been tested for both simple and complicated geometries. Simple multicolor ordering usually provides sufficiently long loops for vector performance, although it usually requires more iterations for convergence than more sophisticated RCM ordering.

All of the developed preconditioning methods on the GeoFEM platform for its local data structure provide excellent parallel/vector performance up to  $> 1,000$  PEs and robustness for very ill-conditioned problems. The localized block ICCG(0) solver with special reordering strategy for unstructured mesh attained 3.80 TFLOPS for simple 3D linear elastic problem with  $2.2 \times 10^9$  DOF on 176 SMP nodes (1,408 PEs) of the Earth Simulator, corresponding to 33.7% of peak performance.

Parallel CG solvers with selective blocking preconditioning and special reordering developed in this study provided excellent performance on the Earth Simulator (29.1% of peak performance) and robustness for ill-conditioned matrices which appear in contact problems. Moreover, selective blocking preconditioning is memory efficient and requires only 25% of ILU(2) and 50% of ILU(1).

The parallel multigrid procedure with new local data structure provided excellent scalability and parallel performance of greater than 95% on a Hitachi SR2201 with 128 PEs. The *direct jump* method developed in this study for locally refined mesh is very simple, but was found to be much more efficient than the existing *level-by-level* method described in [11] for deeper-level adaptation. The effect of the parallel multilevel ILU smoother for ill-conditioned problems has been also evaluated.

These methods are very useful for wide range of scientific applications developed for SMP cluster type architecture which has become very popular for massively parallel computers in recent days.

Adaptive methods in applications with unstructured meshes have evolved as efficient tools for obtaining numerical solutions without a priori knowledge of the details of the nature of the underlying physics. However, these methods cause severe

load imbalance among processors in parallel computations. In this thesis, a parallel mesh adaptation method with dynamic load-balancing using DRAMA library [129] has been developed and implemented on a 3D compressible Navier-Stokes solver developed on the GeoFEM platform. The extended data structure of GeoFEM with mesh adaptation has been also proposed. This data structure with double-numbering proved to very flexible and efficient for processing distributed local data sets with parallel mesh adaptation and dynamic load balancing.



### 7.3 Future Study

In this study, three categories of parallel preconditioning method for large-scale problems have been developed on the GeoFEM platform. Moreover, an extended data structure for new methods has been proposed. Each of the developed methods demonstrated excellent performance and robustness for various types of complicated large-scale problems on massively parallel computers.

In the future, we will also examine:

- Large-scale applications with complicated geometry and physics on massively parallel computers such as the Earth Simulator will be performed using the newly developed method along with the GeoFEM platform.
- The current local data structure in GeoFEM is very simple but is not suitable for a wide range of applications and procedures. The newly developed data structure for mesh adaptation and multigrid provide the GeoFEM platform with flexibility for various types of applications.
- In this study, primarily ILU/IC and related preconditioning methods have been treated. Recently, the sparse approximate inverse method (SAI) [21,103,115,123] for preconditioning is expected to be applied as a global preconditioner in parallel computing. In the contact problems described in Chapter 4, infinitesimal and linear elastic deformation theory was assumed, although large slip and large deformation have to be considered in real simulations, where node location and the connectivity of contact groups can change dynamically. More robust preconditioning method and dynamic load-balancing methods will have to be developed for parallel computing in these types of models. According to [115], SAI is feasible for this type of problem and no contact information or repartitioning are required. Further study on the SAI method is required.
- Integration of the three categories of the preconditioning methods described in Section 1.2 is needed. Multigrid-based methods are scalable for large-scale problems but are not necessarily robust for problems with local constraints such as the contact simulations described in Chapter 4. An integrated method of multigrid and selective-blocking will provide both scalability and robustness for large-scale

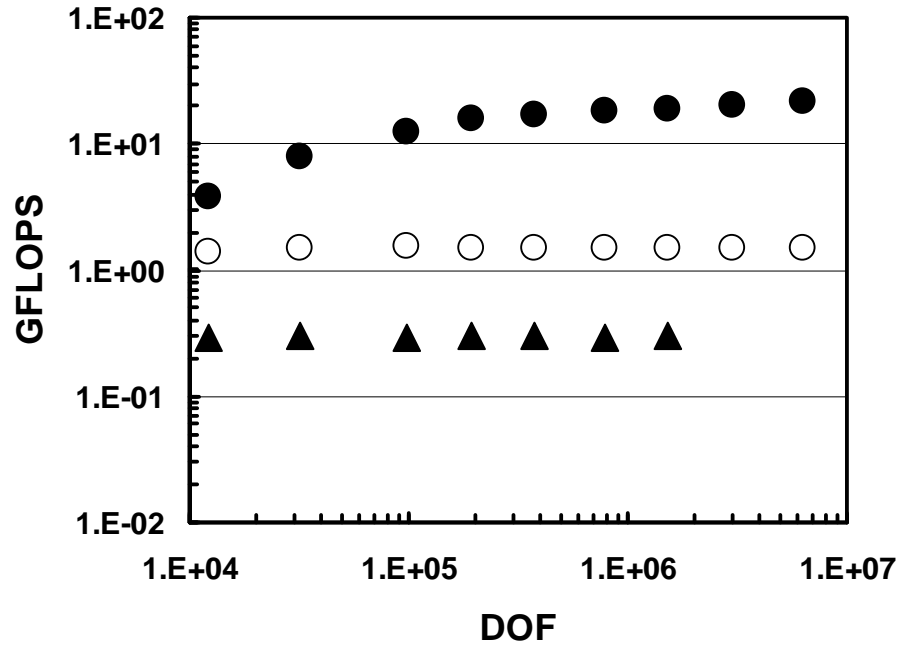
ill-conditioned problems.

- AMG (Algebraic Multigrid Method) is another expected method, but a number of AMG approaches suffer from parallelization problems. Hybrid algebraic-geometric multigrid methods have been successful. Proper data definition, including coarse mesh and a multilevel communication table, is required for this area of study. Moreover, very short loops in multigrid procedures at coarser levels of the grid reduce performance on vector processors. This problem is also solved by proper data definition.
- In this study, developed methods have been optimized primarily for vector processors, such as the Earth Simulator, or RISC processors with pseudo vector capability, such as the Hitachi SR2201 or SR8000. It is well-known that optimized code for vector processor is not necessarily optimum for RISC processors. Figure 7.1 shows an example. A block ICCG(0) solver with special ordering strategy, described in Chapter 3 optimized for vector processors were applied to a RISC processor. On vector processors, the differences among these three methods were significant, but it is very slight on a RISC processor. Moreover, performance decreases suddenly for larger problems due to cache overflow. This is a great disadvantage with respect to portability of the simulation codes for the high-performance computing environment. Recently, a project for *HPC middleware* (HPC-MW) [134] has started. HPC-MW is an infrastructure for developing optimized and reliable scientific simulation codes efficiently. In order to develop this HPC-MW, various types of scientific simulation methods such as FEM, FDM, FVM, BEM, Spectral Methods, MD and Particle Methods, should be investigated, and typical and common patterns for operations are extracted and each procedure will be optimized for various types of computers including vector/RISC processors, SMP parallel architectures and PC clusters. Source code developed on single-processor PCs is easily optimized on massively parallel computers by plugging-in the source code to the HPC-MW installed on the target computer (Fig.7.2). This HPC-MW will provide dramatic efficiency, portability and reliability in the development of scientific simulation codes. For example, the line number of the source codes is expected to be less than 1,000, and the duration of the development is expected to be 10% of previous development time. Moreover, under GRID environment where various types of computers are connected through networks, a virtual petaflops environment can be attained using a global operating

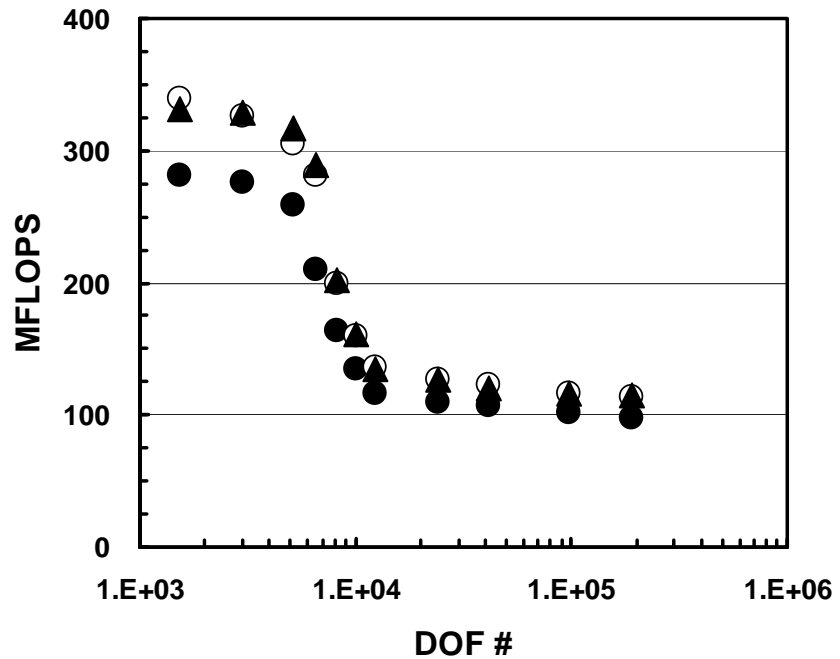
system and HPC-MW which is optimized for each hardware. Thus very large-scale simulation using world-wide resources (computer hardware, code, observed/computed data sets, etc.) is possible.



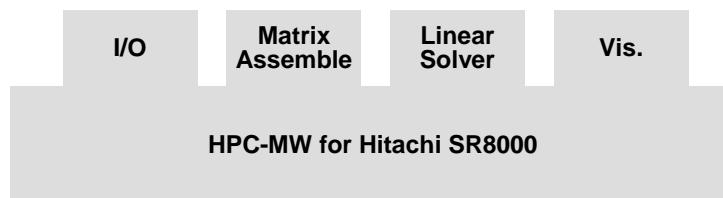
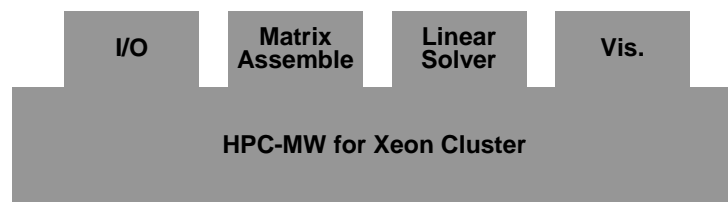
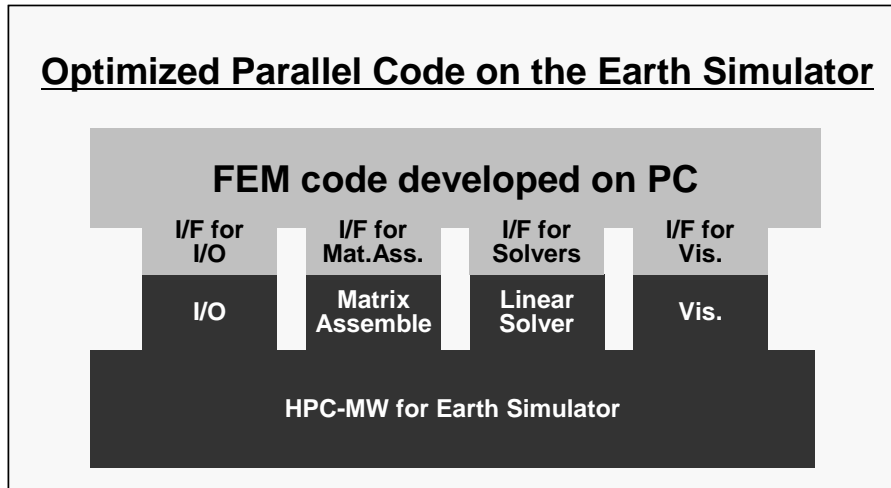
(a) Earth Simulator



(b) COMPAQ Alpha 21164/599 MHz



**Fig.7.1** Effect of coefficient matrix storage method and reordering for the 3D linear elastic problem in Fig.3.10 with various problem sizes on (a) Earth Simulator (1 SMP node) and (b) COMPAQ Alpha 21164/599 MHz (Single PE). (BLACK Circles: PDJDS/CM-RCM, WHITE Circles: PDCRS/CM-RCM, BLACK Triangles: CRS no reordering).



**Fig.7.2**      HPC-Middleware (HPC-MW) for Finite Element Method [134].

## References

- (1) M.F. Adams: *Multigrid Equation Solvers for Large Scale Nonlinear Finite Element Simulations*, UCB/CSD-99-1033, Ph.D. Thesis for University of California at Berkeley, 1999.
- (2) M.F. Adams and J.W. Demmel: *Parallel Multigrid Solver for 3D Unstructured Finite Element Problems*, SC99 Proceedings, Portland, Oregon, USA, 1999.
- (3) M.F. Adams: *A Distributed Memory Unstructured Gauss-Seidel Algorithm for Multigrid Smoothers*, SC 2001 Proceedings, Denver, Colorado, USA, 2001.
- (4) S.F. Ashby and R.D. Falgout: A Parallel Multigrid Preconditioned Conjugate Gradient Algorithm for Groundwater Flow Simulation, Nuclear Science and Engineering, Vol.124, pp.145-159, 1996.
- (5) V.A. Bandy, J.E. Dendy Jr. and W.H. Spangenberg: *Some Multigrid Algorithms for Elliptic Problems on Data Parallel Machines*, SIAM Journal of Scientific Computing, Vol.19, No.1, pp.74-86, 1998.
- (6) R.E. Bank and J. Xu: *The Hierarchical Basis Multigrid Method and Incomplete LU Decomposition*, Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations (D. Keyes and J. Xu, eds.), AMS, Providence, Rhode Island, USA, pp.163-173, 1994.
- (7) 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.
- (8) A. Basermann, J. Clinckemaillie, T. Coupez, J. Fingberg, H. Digonnet, R. Ducloux, J.M. Gratien, U. Hartmann, G. Lonsdale, B. Maerten, D. Roose, C. Walshaw,; *Dynamic load balancing of finite element applications with the DRAMA library*, Applied Mathematics Modelling, Vol.25, pp.83-98, 2000.
- (9) R. Biswas and R. Strawn: *A New Procedure for Dynamic Adaptation of Three-Dimensional Unstructured Grids*, AIAA 93-0672, 1993.
- (10) D. Braess: *Finite Elements : Theory, fast solvers and applications in solid mechanics*, Cambridge University Press, 1997.
- (11) W.L. Briggs, V.E. Henson and S.F. McCormick: *A Multigrid Tutorial Second Edition*, SIAM, 2000.
- (12) P.N. Brown, R.D. Falgout, and J.E. Jones: *Semicoarsening Multigrid on Distributed Memory Machines*, Lawrence Livermore National Laboratory

- Technical Report UCRL-JC-130720, 1998.
- (13) F. Cappelo, and D. Etiemble: *MPI versus MPI+OpenMP on the IBM SP for the NAS Benchmarks*, SC2000 Technical Paper, Dallas, Texas, 2000.
  - (14) R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald and R. Menon: *Parallel Programming in OpenMP*, Morgan Kaufmann Publishers, USA, 2001.
  - (15) L. Chen, I. Fujishiro, and K. Nakajima: *Parallel performance optimization of large-scale unstructured data visualization for the Earth Simulator*, Eurographics Workshop on Parallel Visualization and Graphics 2002, Blaubeuren, Germany, pp.133-140, 2002.
  - (16) A.J. Cleary, R.D. Falgout, V.E. Henson and J.E. Jones: *Coarse-Grid Selection for Parallel Algebraic Multigrid*, Lawrence Livermore National Laboratory Technical Report UCRL-JC-130893, 1998.
  - (17) W. Dahmen, S. Muller and T. Schlinkmann: *On an adaptive multigrid solver for convection-dominated problems*, SIAM Journal of Scientific Computing, Vol.23, No.3, pp.781-804, 2001.
  - (18) M.J. Dayde, J-Y. L'Excellent and N.I.M. Gould: *Element-by-Element Preconditioners for Large Partially Separable Optimization Problems*, SIAM Journal of Scientific Computing, Vol.18, No.6, pp.1767-1787, 1997.
  - (19) J.W. Demmel: *Applied Numerical Algebra*, SIAM, 1997.
  - (20) M.J. Djomehri and H.H. Jin: *Hybrid MPI+OpenMP Programming of an Overset CFD Solver and Performance Investigations*, NASA/NAS Technical Report (NASA Ames Research Center), NAS-02-002, 2002.
  - (21) J.J. Dongarra, I. Duff, D.C. Sorensen and H.A. van der Vorst: *Numerical Linear Algebra for High-Performance Computers*, SIAM, 1998.
  - (22) S. Ezure, H. Okuda and K. Nakajima: *Parallel Mesh Relocation, Parallel Finite Element Analysis, Large-Scale Simulation*, RIST/Tokyo GeoFEM Report 2002-012, 2002.
  - (23) R.D. Falgout and J.E. Jones: *Multigrid on Massively Parallel Architectures*, Sixth European Multigrid Conference, Ghent, Belgium, 1999.
  - (24) C. Farhat and M. Lesoinne: *Automatic Partitioning of Unstructured Meshes for the Parallel Solution of Problems in Computational Mechanics*, International Journal for Numerical Methods in Engineering, Vol.36, pp.745-764, 1993.
  - (25) C.A.G. Fletcher: *Computational Galerkin Method*, Springer-Verlag, 1984.



- (26) C.A.G. Fletcher: *Computational Techniques for Fluid Dynamics 1*, Springer-Verlag, 1988.
- (27) L.P. Franca, S.L. Frey and T.J.R. Hughes: *Stabilized finite element methods: II. The incompressible Navier-Stokes equations*, Computer Methods in Applied Mechanics and Engineering, Vol.99, pp.209-233, 1992.
- (28) K. Garatani, H. Nakamura, H. Okuda and G. Yagawa: *GeoFEM: High Performance Parallel FEM for Solid Earth*, Lecture Notes in Computer Science No.1593, pp.132-140, 1999.
- (29) A.Greenbaum: *Iterative Methods for Solving Linear Systems*, SIAM, 1997.
- (30) P.M. Gresho: *Some current CFD issues relevant to the incompressible Navier-Stokes equations*, Computer Methods in Applied Mechanics and Engineering, Vol.87, pp.201-252, 1991.
- (31) W. Gropp, E. Lusk and A. Skjellum: *Using MPI, Portable Parallel Programming with the Message-Passing Interface*, MIT Press, 1994.
- (32) M. Hafez and M. Soliman: *Numerical Solution of the Incompressible Navier-Stokes Equations in Primitive Variables on Unstaggered Grids*, AIAA Paper 91-1561 CP, 1991.
- (33) F.H. Harlow and J.E. Welch: *Numerical Calculation of Time-Dependent Viscous Incompressible Flow with Free Surface*, Physics of Fluids, Vol.8, pp.2182-2189, 1965.
- (34) D.G. Holmes and S.D. Connel: *Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Grids*, AIAA Paper 89-1932 CP, 1989.
- (35) D. Hysom and A. Pothen: *Efficient Parallel Computaion of ILU(k) Preconditioners*, NASA/CR-2000-210210, ICASE Report No.2000-23, 2000.
- (36) M. Iizuka, H. Okuda and G. Yagawa: *Nonlinear Structural Subsystem of GeoFEM for Fault Zone Analysis*, Pure and Applied Geophysics, Vol.157, pp.2105-2124, 2000.
- (37) O-P. Jacquotte: *Grid Optimization Methods for Quality Improvement and Adaptation*, Handbook of Grid Generation, pp.33-1 - 33-33, CRC Press, 1999.
- (38) A. Jameson, W. Schmidt and E. Turkel: *Numerical Solutions of the Euler Equations by Finite-Volume Methods Using Runge-Kutta Time-Stepping Schemes*, AIAA Paper 81-1259, 1981.
- (39) Y. Kallinderis: *Adaptation Methods for Viscous Flows*, Ph.D. Thesis, CFDL-TR-89-5, Dept. Aeronautics and Astronautics, MIT, May 1989.
- (40) Y. Kallinderis and J.R. Baron: *Adaptation Methods for a New Navier-Stokes Algorithm*, AIAA Journal, Vol.27, pp.37-43, 1989.

- (41) Y. Kallinderis and V. Parthasarathy: *An Adaptive Refinement Coarsening Scheme for 3-D Unstructured Meshes*, AIAA Journal, Vol 31, pp 1440-1447, 1993.
- (42) Y. Kallinderis and S. Ward: *Prismatic Grid Generation for 3-D Complex Geometries*, AIAA Journal, Vol. 31, pp.1850-1856, 1993.
- (43) Y. Kallinderis and K. Nakajima: *Finite Element Method for Incompressible Viscous Flows with Adaptive Hybrid Grids*, AIAA Journal, Vol.32, No.8, pp.1617-1625,1994.
- (44) V. Karlo and T. Tezduyar: *Parallel 3D Computation of Unsteady Flow around Circular Cylinders*, AHPCRC Preprint 96-074, Army High Performance Computing Research Center, 1996.
- (45) D.J. Kerbyson, A. Hoisie and H. Wasserman, *A Comparison Between the Earth Simulator and AlphaServer Systems using Predictive Application Performance Models*, LA-UR-02-5222, Los Alamos National Laboratory, USA, 2002.
- (46) S.W. Kim and T.J. Benson: *Comparison of the SMAC, PISO and Iterative-Advancing Schemes for Unsteady Flows*, Computers & Fluids, Vol.21, pp.435-454, 1992.
- (47) D.A. Knoll and W.J. Rider: *A Multigrid Preconditioned Newton-Krylov Method*, SIAM Journal of Scientific Computing, Vol.21, No.2, pp.691-710, 1999.
- (48) D.Y. Kwak: *V-cycle Multigrid for Cell-Centered Finite Differences*, SIAM Journal of Scientific Computing, Vol.21, No.2, pp.552-564, 1999.
- (49) J. Linden, G. Lonsdale, B. Steckel and K. Stüben: *Multigrid for the Steady-State Incompressible Navier-Stokes Equations: a Survey*, International Conference on Numerical Methods in Fluid Mechanics, Williamsburg, VA, 1988.
- (50) J. Liou and T.E. Tezduyar: *Clustered Element-by-Element Computations for Fluid Flow*, Parallel Computational Fluid Dynamics (Implementations and Results), edited by H.D.Simon, The MIT Press, 1992. pp.167-187.
- (51) F. Liu, S. Ji and G. Liao: *An adaptive Grid Method and Its Application to Steady Euler Flow Calculations*, SIAM Journal of Scientific Computing, Vol.20, No.3, pp.811-825, 1998.
- (52) I.M. Llorente and N.D.Melson: *Robust Multigrid Smoothers for Three Dimensional Elliptic Equations with Strong Anisotropies*, NASA/CR-1998-208700, ICASE Report No.98-37, 1998.

- (53) I.M. Llorente, B. Diskin and N.D.Melson: *Plane Smoothers for Multiblock Grids : Computational Aspects*, NASA/CR-1999-209331, ICASE Report No.99-17, 1999.
- (54) I.M. Llorente, B. Diskin and N.D.Melson: *Alternating Plane Smoothers For Multiblock Grids*, SIAM Journal on Scientific Computing Volume 22, Number 1, 2000.
- (55) R. Löhner: *The Efficient Simulation of Strongly Unsteady Flows by the Finite-Element Method*, AIAA Paper 87-0555, 1987.
- (56) R. Löhner: *A Fast Finite Element Solver for Incompressible Flows*, AIAA Paper 90-0398, 1990.
- (57) R. Löhner: *An Implicit Linelet-Based Solver For Incompressible Flows*, AIAA Paper 92-0668, 1992.
- (58) D.J. Mavriplis: *Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes*, NASA CR-1998-206910, ICASE Report No.98-6, 1998.
- (59) D.J. Mavriplis: *Directional Agglomeration Multigrid Techniques for High-Reynolds Number Viscous Flows*, NASA CR-1998-206911, ICASE Report No.98-7, 1998.
- (60) D.J. Mavriplis: *Large-scale Parallel Viscous Flow Computations Using an Unstructured Multigrid Algorithm*, NASA CR-1999-209724, ICASE Report No.99-44, 1999.
- (61) D.J. Mavriplis: *Parallel Performance Investigations of an unstructured mesh Navier-Stokes Solver*, NASA/CR-2000-210088, ICASE Report No.2000-13, 2000.
- (62) D.J. Mavriplis: *An Assessment of Linear Versus Non-linear Multigrid Methods for Unstructured Mesh Solvers*, NASA/CR-2001-210870, ICASE Report No.2001-12, 2001.
- (63) D.S. McRae and K.R. Laflin: *Dynamic Grid Adaptation and Grid Quality*, Handbook of Grid Generation, pp.34-1 - 34-33, CRC Press, 1999.
- (64) R.S. Montero, I.M. Llorente and M.D. Salas: *Semicoarsening and Implicit Smoothers for the Simulation of a Flat Plate at Yaw*, NASA/CR-2001-210871, ICASE Report No.2001-13, 2001.
- (65) P. de Montleau, J.M. Cela, S.M. Mpong and A. Godinass: *A Parallel Computing Mode for the Acceleration of a Finite Element Software*, International Workshop on OpenMP: Experiences and Implementations (WOMPEI 2002), Kyoto, Japan, Lecture Notes in Computer Science 2327, p.449-456, Springer, 2002.

- (66) E. Morano, D.J. Mavriplis and V. Venkatakrishnan: *Coarsening Strategies for Unstructured Multigrid Techniques with Application to Anisotropic Problems*, SIAM Journal of Scientific Computing, Vol.20, No.2, pp.395-415, 1998.
- (67) I. Moulitsas and G. Karyois: *Multilevel Algorithms for Generating Coarse Grids for Multigrid Methods*, SC 2001 Proceedings, Denver, Colorado, USA, 2001.
- (68) K. Nakajima: *Incompressible Navier-Stokes Methods with Hybrid Adaptive Grids*, M.S. Thesis, University of Texas at Austin, 1993.
- (69) K. Nakajima, Y. Kallinderis, I.A. Sibetheros, R.W. Miksad and K.F. Lambrakos: *A Numerical Study of the Hydrodynamics of Reversing Flows around a Cylinder*, Transaction of the ASME Journal of Offshore Mechanics and Arctic Engineering, Vol.116, No.4, pp.202-208, 1994.
- (70) K. Nakajima and Y. Kallinderis: *Comparison of Finite Element and Finite Volume Methods for Incompressible Viscous Flows*, AIAA Journal, Vol.32, No.8, pp.1090-1093.,1994.
- (71) K. Nakajima, H. Nakamura and T. Tanahashi: *Parallel Iterative Solvers with Localized ILU Preconditioning*, Lecture Notes in Computer Science 1225, pp.342-350, 1997.
- (72) K. Nakajima and H. Okuda: *Parallel Iterative Solvers with Localized ILU Preconditioning for Unstructured Grids*, IMACS Series in Computational and Applied Mathematics Volume 5: Iterative Methods in Scientific Computation IV, p.85-98, 1999.
- (73) K. Nakajima and H. Okuda: *Parallel Iterative Solvers with Localized ILU Preconditioning for Unstructured Grids on Workstation Cluster*, International Journal for Computational Fluid Dynamics, Vol.12, pp.315-322, 1999.
- (74) K. Nakajima: *Parallel Multilevel Iterative Solvers for 3D Incompressible Navier-Stokes Equations*, FEF 2000 (Finite Elements in Flow Problems), Austin, Texas, April, 2000.
- (75) K. Nakajima and H. Okuda: *Parallel Iterative Solvers for Simulations of Fault Zone Contact using Selective Blocking Reordering*, 2001 International Conference on Preconditioning Techniques for Large Sparse Matrix Problems in Industrial Applications (Preconditioning 2001), Tahoe City, CA, USA, 2001 (submitted to *Journal of Numerical Algebra with Applications* (in press)).
- (76) K. Nakajima, J. Fingberg and H. Okuda: *Parallel 3D Adaptive Compressible Navier-Stokes Solver in GeoFEM with Dynamic Load-Balancing by DRAMA Library*, HPCN Europe 2001, Amsterdam, Netherlands, Lecture Notes in

- Computer Science 2110, p.183-193, Springer, 2001.
- (77) K. Nakajima and H. Okuda: *Parallel Iterative Solvers with the Selective Blocking Preconditioning for Simulations of Fault-Zone Contact*, GeoFEM 2001-010, RIST/Tokyo, 2001.
  - (78) K. Nakajima: *Parallel Multilevel Iterative Linear Solvers with Unstructured Adaptive Grids for Simulations in Earth Science*, SSS2001 (Workshop on Scalable Solver Software: Multiscale Coupling and Computational Earth Science), Tokyo, 2001.
  - (79) K. Nakajima and H. Okuda: *Parallel Iterative Solvers for Unstructured Grids using Directive/MPI Hybrid Programming Model for GeoFEM Platform on SMP Cluster Architectures*, Concurrency and Computation: Practice and Experience. pp.411-429, Vol.14, No.6-7, 2002.
  - (80) 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.
  - (81) K. Nakajima and H. Okuda: *Parallel Iterative Solvers for Unstructured Grids using an OpenMP/MPI Hybrid Programming Model for the GeoFEM Platform on SMP Cluster Architectures*, International Workshop on OpenMP: Experiences and Implementations (WOMPEI 2002), Kyoto, Japan, Lecture Notes in Computer Science 2327, p.437-448, Springer, 2002.
  - (82) E.G. Ng, B.W. Peyton and P. Raghavan: *A Blocked Incomplete Cholesky Preconditioner for Hierarchical-Memory Computers*, IMACS Series in Computational and Applied Mathematics Volume 5: Iterative Methods in Scientific Computation IV, p.211-221, 1999.
  - (83) T. Oguni, T. Murata, T. Miyoshi, J.J. Dongarra and H.Hasegawa: *Matrix Computation Softwares* (in Japanese), Maruzen, 1991.
  - (84) H.Okuda, G.Yagawa, K.Nakajima and H.Nakamura; *Parallel Finite Element Solid Earth Simulator: GeoFEM*, WCCM V (Fifth World Congress on Computational Mechanics), Vienna, Austria, 2002.
  - (85) L. Oliker and R. Biswas: *Parallelization of a Dynamic Unstructured Application using Three Leading Paradigms*, SC99 Proceedings, Portland, Oregon, USA, 1999.
  - (86) L. Oliker, X. Li, P. Husbands and R. Biswas : *Effects of Ordering Strategies and Programming Paradigms on Sparse Matrix Computations*, SIAM Review, Vol.44, No. 3(2002), pp.373-393.
  - (87) C.W. Oosterlee and T. Washio: *An Evaluation of Parallel Multigrid as a Solver*

- and a Preconditioner for Singularly Perturbed Problems*, SIAM Journal of Scientific Computing, Vol.19, No.1, pp.87-110, 1998.
- (88) R.L. Panton: *Incompressible Flow*, John Wiley & Sons, 1984.
  - (89) V. Parthasarathy, Y. Kallinderis and K. Nakajima: *A Navier-Stokes Method with Adaptive Hybrid Prismatic/Tetrahedral Grids*, AIAA Paper 95-0670, 1995.
  - (90) V. Parthasarathy and Y. Kallinderis: *Directional Visous Multigrid Using Adaptive Prismatic Meshes*, AIAA Journal, Vol.33, No.1. pp.69-78, 1995.
  - (91) V. Parthasarathy and Y.Kallinderis: *Adaptive Prismatic-Tetrahedral Grid Refinement and Redistribution for Viscous Flows*, AIAA Journal, Volume 34, No.4. pp.707-716, 1996.
  - (92) S.V. Patankar: *Numerical Heat Transfer and Fluid Flow*, Hemisphere, 1980.
  - (93) S. Patankar and D. Spalding: *A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows*, Int. J. Heat and Mass Transfer, Vol.15, pp.1787-1806, 1972.
  - (94) R.M. Peric, G. Kessler and G. Scheuerer: *Comparison of Finite-Volume Numerical Methods with Staggered and Collocated Grids*, Computers & Fluids, Vol.16, pp.389-403, 1988.
  - (95) M.P. Prieto, R.S. Montero and I.M. Llorente: *A Parallel Multigrid Solver for Viscous Flows on Anisotropic Structured Grids*, NASA/CR-2001-211238, ICASE Report No.2001-34, 2001.
  - (96) T.H. Pulliam: *Artificial Dissipation Models for the Euler Equations*, AIAA Paper 85-0438, 1985.
  - (97) R. Rabenseifner: *Communication Bandwidth of Parallel Programming Models on Hybrid Architectures*, International Workshop on OpenMP: Experiences and Implementations (WOMPEI 2002), Lecture Notes in Computer Science 2327, pp.437-448, 2002.
  - (98) P. Raghavan, K. Teranishi and E.G. Ng: *Towards Scalable Preconditioning using Incomplete Cholesky Factorization*, 2001 International Conference on Preconditioning Techniques for Large Sparse Matrix Problems in Industrial Applications (Preconditioning 2001), Tahoe City, CA, USA, 2001.
  - (99) G.D. Raithby and G.E. Schneider: *Numerical Solution of Problems in Incompressible Fluid Flow: Treatment of the Velocity-Pressure Coupling*, Numerical Heat Transfer, Vol.2, pp.417-440, 1979.
  - (100) M. Raw: *Robustness of coupled algebraic multigrid for the Navier-Stokes equations*, AIAA paper 96-0297, 1996.

- (101) C.M. Rhie: *A Pressure Based Navier-Stokes Solver Using the Multigrid Method*, AIAA Paper 86-0207, 1986.
- (102) T. Sarpkaya and M. Issacson: *Mechanics of Wave Forces on Offshore Structures*, Van Nostrand Reinhold Company, 1981.
- (103) Y. Saad: *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.
- (104) Y. Shapira: *Multigrid for Locally Refined Meshes*, SIAM Journal of Scientific Computing, Vol.21, No.3, pp.1168-1190, 1999.
- (105) M.S. Shepard, J.E. Flaherty, C.L. Bottasso, H.L. de Cougny, C. Ozturan and M.L. Simone: *Parallel automatic adaptive analysis*, Parallel Computing, Vol. 23, pp.1327-1347, 1997.
- (106) H.D. Simon: *Partitioning of unstructured problems for parallel processing*, Computing Systems in Engineering, Vol.2, pp.135-148, 1991.
- (107) B. Smith, P. Bjørstad and W. Gropp: *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.
- (108) F. Sotiropoulos and S. Abdallah: *The Discrete Continuity Equation in Primitive Variable Solutions of Incompressible Flow*, Journal of Computational Physics, Vol.95, pp.212-227, 1991.
- (109) K. Stüben: *Algebraic Multigrid (AMG) : An Introduction with Applications*, GMD Report 53, GMD-Forschungszentrum Informationstechnik GmbH, 1999.
- (110) U. Tottemberg, C.Oosterlee, A.Schüller: *Multigrid*, Academic Press, 2001.
- (111) R.S. Tuminaro, J.N. Shadid and S.A. Hutchinson: *Parallel Sparse Matrix Vector Multiply Software for Matrices with Data Locality*, Sandia National Laboratories Technical Report SAND 95-1540J, 1995.
- (112) R.S. Tuminaro and C. Tong: *Parallel Smoothed Aggregation Multigrid : Aggregation Strategies on Massively Parallel Machines*, SC 2000 Proceedings, Dallas, Texas, USA, 2000.
- (113) V. Venkatakrishnan: *Parallel Implicit Methods for Aerodynamic Applications on Unstructured Grids, Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering*, pp.57-74, SIAM, 1994.
- (114) A. Vidwans, Y. Kallinderis and V. Venkatakrishnan: *Parallel Dynamic Load-Balancing Algorithm for Three-Dimensional Adaptive Unstructured Grids*, AIAA Journal, Vol.32, No.3, pp.497-505, 1995.
- (115) K. Wang, S-B. Kim, J. Zhang, K. Nakajima and H. Okuda: *Global and*

- localized parallel preconditioning techniques for large scale solid Earth simulations*, Technical Report No. 345-02, Department of Computer Science, University of Kentucky, 2002.
- (116) Z.J. Wang: *A fast nested multi-grid viscous flow solver adaptive Cartesian/Quad grids*, International Journal for Numerical Methods in Fluids, Vol.33, pp.657-680, 2000.
  - (117) T. Washio and C.W. Oosterlee: *Flexible Multiple Semicoarsening for Three-Dimensional Singular Perturbed Problems*, SIAM Journal of Scientific Computing, Vol.19, No.5, pp.1646-1666, 1998.
  - (118) T. Washio, K. Maruyama, T. Osoda, F. Shimizu, and S. Doi: *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, pp.42-49, 1999.
  - (119) T. Washio, K. Maruyama, T. Osoda, F. Shimizu, and S. Doi: *Efficient implementations of block sparse matrix operations on shared memory vector machines*, SNA2000 : The Fourth International Conference on Supercomputing in Nuclear Applications, Tokyo, Japan, 2000.
  - (120) J.M. Weiss, J.P. Maruszewski and W.A. Smith: *Implicit Solution of Preconditioned Navier-Stokes Equations Using Algebraic Multigrid*, AIAA Journal, Vol.37, No.1, pp.29-36, 1999.
  - (121) R. Wienands, C.W. Oosterlee and T. Washio: *Fourier Analysis of GMRES(m) Preconditioned by Multigrid*, SIAM Journal of Scientific Computing, Vol.22, No.2, pp.582-603, 2001.
  - (122) A.M. Wissink, R.D. Hornung, S.R. Kohn, S.S. Smith and N. Elliott: *Large Scale Parallel Structured AMR Calculations Using the SAMRAI Framework*, SC 2001 Proceedings, Denver, Colorado, USA, 2001.
  - (123) J. Zhang: *Sparse approximate inverse and multilevel block ILU preconditioning techniques for general sparse matrices*, Applied Numerical Mathematics, Vol.35, pp.67-86, 2000.
  - (124) J. Zhang: *Preconditioned Krylov subspace methods for solving nonsymmetric matrices from CFD applications*, Computer Methods in Applied Mechanics and Engineering, Vol.189, pp.825-840, 2000.
  - (125) S.L. Zhang: *GPBi-CG: Generalized Product-type methods based on Bi-CG for solving nonsymmetric linear systems*, SIAM Journal of Scientific Computing, Vol.18, No.2, pp.537-551, 1997.
  - (126) *Accelerated Strategic Computing Initiative (ASCI) Web Site:*



- <http://www.llnl.gov/asci/>
- (127) *AZTEC* Web Site: <http://www.cs.sandia.gov/CRF/aztec1.html/>
  - (128) *CASC (Center for Applied Scientific Computing, Lawrence Livermore National Laboratory)* Web Site: [http://www.llnl.gov/CASC/linear\\_solvers/](http://www.llnl.gov/CASC/linear_solvers/)
  - (129) *DRAMA* Web Site:  
<http://www.ccrl-nece.technopark.gmd.de/~drama/drama.html/>
  - (130) *Earth Simulator Center* Web Site: <http://www.es.jamstec.go.jp/>
  - (131) *GeoFEM* Web Site: <http://geofem.tokyo.rist.or.jp/>
  - (132) *Hitachi SR2201* Web Site:  
<http://www.hitachi.co.jp/Prod/comp/hpc/jpn/sr2.html>
  - (133) *Hitachi SR8000* Web Site:  
<http://www.hitachi.co.jp/Prod/comp/hpc/foruser/sr8000/>
  - (134) *HPC Middleware* Web Site (Frontier Simulation Software for Industrial Science, Institute of Industrial Science, The University of Tokyo):  
[http://www.fsis.iis.u-tokyo.ac.jp/hpc/index\\_e.html](http://www.fsis.iis.u-tokyo.ac.jp/hpc/index_e.html)
  - (135) *Information Technology Center, The University of Tokyo* Web Site:  
<http://www.cc.u-tokyo.ac.jp/>
  - (136) *JOSTLE* Web Site: <http://www.gre.ac.uk/jostle/>
  - (137) *LAMP* Web Site:  
<http://www.ccrl-nece.technopark.gmd.de/~maciej/LAMP/LAMP.html>
  - (138) *METIS* Web Site: <http://www-users.cs.umn.edu/~karypis/metis/>
  - (139) *MPI (Message Passing Interface) Forum* Web Site:  
<http://www.mpi-forum.org/>
  - (140) *NPB (NAS Parallel Benchmarks)* Web Site:  
<http://www.nas.nasa.gov/Research/Software/swdescription.html#NPB>
  - (141) *OpenMP* Web Site: <http://www.openmp.org/>
  - (142) *PETSc* Web Site: <http://www-fp.mcs.anl.gov/petsc/>
  - (143) <http://www6.tomshardware.com/cpu/00q4/001107/mobilecpu-19.html/>



## VITA

Kengo Nakajima was born in Okayama, Japan, on September 14, 1962, the son of Masao and Kazuko Nakajima. He graduated from the *University of Tsukuba High School at Komaba*, Japan in March 1981. He received a Bachelor of Engineering degree in Aeronautics from the *University of Tokyo*, Japan in March 1985. His thesis title is "*Active Flutter Suppression Method for a Cantilevered CFRP Wing*" supervised by Professor Kyohei Kondoh, Dr.Eng.

He worked for the *Mitsubishi Research Institute, Inc. (MRI)*, Japan as a research engineer in the area of nuclear engineering and computational science from April 1985 to June 1999.

While working for *MRI*, he entered the Graduate School of the *University of Texas at Austin* in the Fall of 1991. He received a Master of Science degree in Engineering at the Department of Aerospace Engineering and Engineering Mechanics in May 1993. His thesis title is "*Incompressible Navier-Stokes Methods with Hybrid Adaptive Grids*" supervised by Professor John Kallinderis, Ph.D. He worked as a research associate in TICOM (Texas Institute for Computational Mechanics) from June 1993 to January 1994.

He entered the *Research Organization for Information Science and Technology (RIST)*, Japan in July 1999. Since then he has been working for *GeoFEM* project and developing parallel programming models, parallel iterative linear solvers with preconditioning methods, mesh adaptation procedure with dynamic load balancing and various types of finite-element applications, such as 3D compressible/incompressible Navier-Stokes simulations, 3D groundwater flow and transport with convection and diffusion and unsteady tsunami simulations.