Software for the Parallel Adaptive Solution of Conservation Laws by Discontinous Galerkin Methods

J. E. Flaherty[1], R. M. Loy[2], M. S. Shephard[1], and J. D. Teresco[1]

[1] Scientific Computation Research Center (SCOREC)
and Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180

[2] Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439

Abstract. We develop software tools for the solution of conservation laws using parallel adaptive discontinuous Galerkin methods. In particular, the Rensselaer Partition Model (RPM) provides parallel mesh structures within an adaptive framework to solve the Euler equations of compressible flow by a discontinuous Galerkin method (LOCO). Results are presented for a Rayleigh-Taylor flow instability for computations performed on 128 processors of an IBM SP computer. In addition to managing the distributed data and maintaining a load balance, RPM provides information about the parallel environment that can be used to tailor partitions to a specific computational environment.

1 Introduction

By concentrating the computational effort in regions where solution resolution would otherwise be inadequate, adaptive finite element methods (FEMs) provide a reliable, robust, and time- and space-efficient means of solving problems involving partial differential equations [4]. Portions of the finite element mesh may be refined or coarsened (h-refinement), be moved to follow evolving phenomena (r-refinement), or use methods of different order (p-refinement) to enhance resolution and efficiency. In addition to adaptivity, parallel computation is essential for solving large three-dimensional problems in reasonable times. The discontinuous Galerkin (DG) method [5,6] provides an effective means of solving conservation laws on unstructured meshes in a parallel computing environment (§2). The discontinuous basis can capture shock waves and other discontinuities with accuracy, and the compact (nearest neighbor) stencil minimizes interelement communication. This stencil, furthermore, remains compact with high-order polynomial bases, which is (virtually) essential for unstructured mesh computation.

Reusable software libraries allow finite element problems to be solved without concern for the details of the underlying mesh structures, adaptive procedures, or parallelization. We are developing such libraries to support parallel adaptive finite element computation [10,11,26]. The conventional array-based data representations used for fixed-mesh computation are not well suited for adaptivity by h- or p-refinement [1]. However, alternative structures complicate the automatic (compiler) detection of parallelism. We describe (§3) software to manage distributed mesh data and to provide information about the computational environment by explicit parallelism achieved by message passing using the Message Passing Interface (MPI) [19]. Partitioning and dynamic load balancing algorithms distribute the computation across the processors by a domain decomposition of the spatial (or space-time) mesh.

The DG software is applied to a Rayleigh-Taylor flow instability (§5). This computation represents a preliminary step in the study of thermonuclear flashes on astrophysical bodies, such as neutron stars and white dwarves [18]. Two-dimensional studies [13,14] have shed light on the instability, but the phenomenon is three-dimensional, and such computations are essential for understanding. The problem is, however, quite complex, and the results presented here are only a first step in this direction.

2 The Discontinuous Galerkin Method

We consider three-dimensional conservation laws of the form

$$\mathbf{u}_t(\mathbf{x},t) + \sum_{i=1}^{3} \mathbf{f}_i(\mathbf{x},t,\mathbf{u})_{x_i} = 0, \quad \mathbf{x} \in \Omega, \quad t > 0, \tag{1a}$$

with initial conditions

$$\mathbf{u}(\mathbf{x},0) = \mathbf{u}^0(\mathbf{x}), \quad \mathbf{x} \in \Omega \cup \partial\Omega, \tag{1b}$$

and appropriate well-posed boundary conditions. For the Euler equations, the vector $\mathbf{u}$ specifies the fluid's density, momentum components, and energy. The subscripts t and x_i, $i = 1,2,3$, denote partial differentiation with respect to time and the spatial coordinates. Finite difference schemes for (1), such as the Total Variation Diminishing (TVD) [25,27] and Essentially Non-Oscillatory (ENO) [24] methods, usually achieve high-order accuracy by using a computational stencil that enlarges with order. However, a wide stencil makes the methods difficult to implement on unstructured meshes and limits efficient implementation on parallel computers. Finite element methods have stencils that involve only their neighboring elements regardless of the method order. This allows them to model problems with complex geometries and leads to efficient parallelization. We discretize (1) using a DG finite element method [3,5,6] with a piecewise-continuous spatial basis of polynomials relative to a tetrahedral element Ω_j, $j = 1,2,\ldots,J$, of the mesh on Ω. This basis has a more compact stencil than customary finite element approximations and involves communication only across element faces.

The numerical approximation $\mathbf{U}$ of $\mathbf{u}$ is discontinuous on $\partial\Omega_j$; thus, the flux $\mathbf{f}(\mathbf{u})$ required by the DG method is ambiguous there. It is customarily specified by a "numerical flux" function $\mathbf{h}(\mathbf{U}_j^+, \mathbf{U}_j^-)$ that depends on the solution states $\mathbf{U}_j^+$ and $\mathbf{U}_j^-$ on the interior and exterior, respectively, of $\partial\Omega_j$. Several numerical flux functions are possible [6,24]. Here, we use the method of Colella and Woodward [7,8] to compute an approximate solution to the Riemann problem at $\partial\Omega_j$. This method is based on a Newton's method algorithm of Van Leer [28] but makes the simplifying assumption that wave speeds are the same for both shocks and rarefactions. For efficiency, conditions within rarefactions are computed by linear interpolation to avoid the evaluation of a rational power. Once the ambiguity on $\partial\Omega_j$ has been resolved, the flux may easily be computed. Because of a required iteration, the Colella and Woodward flux is 2–3 times more expensive to evaluate than Van Leer's flux vector splitting [9,17,29], but it offers much greater resolution at contact discontinuities.

Computations with polynomial degrees $p > 0$ require flux or solution limiting to preserve a monotonic behavior near discontinuities. Biswas et al. [3] describe an adaptive solution limiting that avoids "flattening" the solution near smooth extrema and maintains the expected $O(h^{p+1})$, $h = \max_{1\leq j\leq J} \operatorname{diam}(\Omega_j)$, L^1 convergence rate when solutions are smooth [3,6]. The results in §6 use a piecewise-constant ($p = 0$) basis with explicit Euler integration in time; hence, limiting is unnecessary.

3 Rensselaer Partition Model

The *Rensselaer Partition Model* (RPM) [26] provides distributed mesh data structures and information about the parallel computational environment in which a program is executing. The basic mesh data structures in RPM are provided by the *SCOREC Mesh Database* (MDB) [1]; however, many of the ideas may be applied to other systems. MDB includes operators to query and update a mesh data structure consisting of a full mesh entity hierarchy: three-dimensional *regions*, and their bounding *faces*, *edges*, and *vertices*, with bidirectional links between mesh entities of consecutive order. Regions serve as finite elements in three dimensions, while faces are finite elements in two dimensions or interface elements in three dimensions. The full entity hierarchy allows efficient mesh modification during h-refinement [22] and facilitates p-refinement [21] by allowing attachment of degrees of freedom to the mesh entities and by providing necessary geometric information. Mesh entities have an explicit *geometric classification* relative to a geometric (CAD) model of the problem domain. This allows the mesh to remain correct with respect to the geometry during h- or p-refinement. Mesh entities are stored with the geometry, so inverse classification information (retrieval of all mesh entities classified on a given model entity) is readily available. This is useful, for example, when applying a boundary condition on a model face. Rather than visiting all faces in the mesh and querying each to check whether it is on the desired boundary, a list of the needed entities is traversed directly.

Each entity in a distributed finite element mesh is uniquely assigned to a *partition*. Each partition is assigned to a specific *process*, with the possibility that multiple partitions may be assigned to a single process. “Process” in this context refers to an address space. The model is hierarchical, with partitions assigned to a *process model* and processes assigned to a *machine model*.

The machine model represents the computational environment: the processing nodes and their network interconnections. The process model maps processes to the computer and maps interprocess communication to intercomputer networks or, perhaps, to a shared-memory interface. Partitions know the mesh entities that they contain, and mesh entities know their partition assignments (*partition model classifications*).

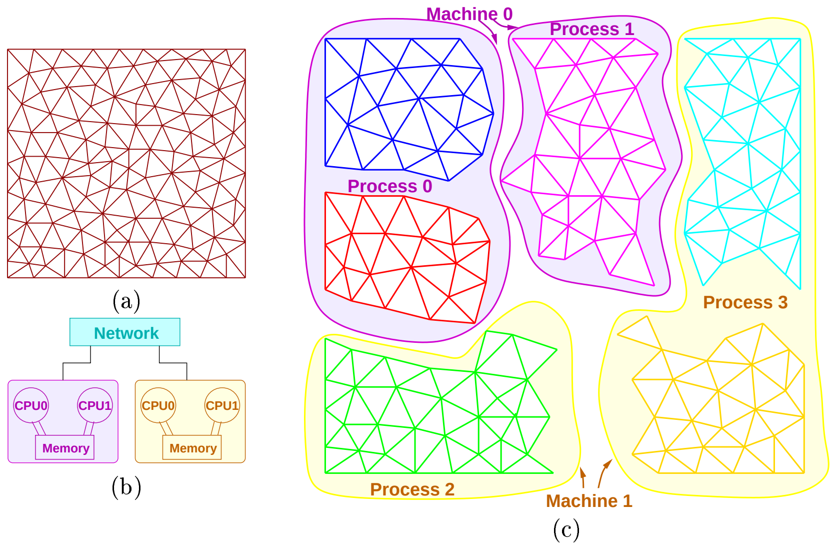

Fig. 1. A sample two-dimensional mesh (a), target parallel environment in which the mesh is to be partitioned (b), and partitioning of the mesh and assignment to processes and machines for the parallel environment (c).

In Figure 1, we see a sample two-dimensional mesh (a) and a target parallel environment consisting of two 2-way SMP workstations connected by a network (b). Figure 1(c) shows a partitioning of this mesh and the assignment of those partitions to the processes and machines of the target environment. Six partitions are created and assigned to four processes, since four processors are available. Two processes are assigned two partitions, while the other two are assigned only a single partition. The four processes are further assigned to the available machines, two to each.

Mesh entities are replicated only when on a partition boundary that is also a process boundary. Figure 2 shows two-dimensional examples of mesh faces that share a common edge across a partition boundary. The shared mesh edge is classified on the partition boundary in each case. On the left, the partition boundary is local to the process, so the mesh entity need not be replicated and is stored only with the partition boundary mesh. On the right, the partition boundary is also on the process boundary. This partition boundary is replicated in each process, so any mesh entity classified on this boundary must also be replicated.

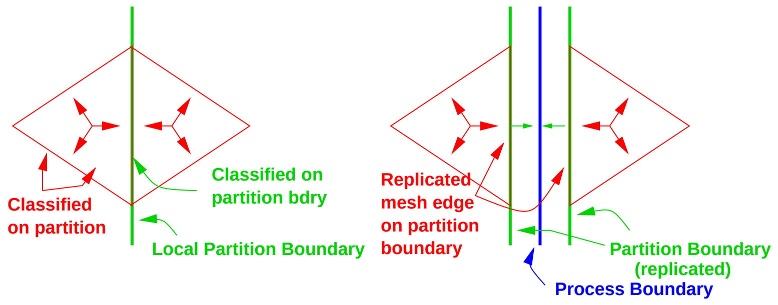
Classified on partition bdry
Classified on partition
Local Partition Boundary
Replicated mesh edge on partition boundary
Partition Boundary (replicated)
Process Boundary

Fig. 2. Partition classification of mesh entities at a same-process partition (left) and at a process-boundary partition (right).

4 Adaptive Methods

Adaptive spatial h-refinement [20,23] is edge based, using error indicators to guide enrichment. An element may be subdivided isotropically or anisotropically, according to predefined templates, depending on the number of its edges selected for refinement. Coarsening is performed when a convex polyhedron of elements request it. A central vertex is identified, the interior edges of the polyhedron are removed, and the polyhedron is remeshed to form fewer elements. Both refinement and coarsening are performed on distributed meshes. During refinement, interprocessor communication is required to update shared vertices, edges, and faces; however, element migration is not necessary. Coarsening requires that the entire polyhedron of elements lie on the same processor, so element migration may be required if the mesh near an interprocessor boundary is marked for coarsening.

With a wide range of element sizes, it is advantageous to use a local refinement method (LRM) [12,16] where spatially dependent time steps are based upon the Courant stability condition. In a given time period, a small number of large time steps will be taken on large elements, while the opposite will occur on small elements.

The time step for Ω_j is determined from the Courant condition as

$$\Delta t_j = \alpha \frac{r_j}{v_j}, \quad \alpha \leq 1, \tag{2}$$

where r_j is the radius of Ω_j's inscribed sphere and v_j is the maximum signal speed on Ω_j. For the Euler equations, v_j is the sum of the fluid speed and the sound speed. The parameter α is introduced to maintain stability in areas of mesh gradation. We empirically chose $\alpha = 0.65$, but a more thorough analysis is necessary.

Elements may take any stable time step; however, small differences in element sizes and shapes lead to minor differences in time steps. These differences, in turn, lead to time stepping many isolated elements, which causes additional flux evaluations and interpolations. Efficiency can, thus, be improved by rounding time steps down to the next lower (fractional) power of two. This time stepping also helps to organize the computation [15].

Temporal interpolation requires storage for solution data at the previous and current times. Additional space may be required so that the solution may be synchronized and interpolated to a common time for checkpointing or outputting. The interval between synchronization times is referred to as a *major* step. Each major step is composed of several smaller steps, each of which performs a single time step on elements that have the necessary data from their neighbors.

5 Rayleigh-Taylor Flow

The resulting software package implementing the parallel adaptive DG solution of the compressible Euler Equations is called *LOCO*. It has been built using the parallel structures and dynamic load balancing algorithms within RPM.

In collaboration with scientists at the University of Chicago and Argonne National Laboratory, we are working toward complete simulations of thermonuclear flashes on astrophysical bodies such as neutron stars and white dwarves. One crucial aspect of these simulations is the correct modeling of the flame front as it leaves the surface of a compact star in a deflagration stage. Because the relatively dense nuclear fuel lies above the less-dense nuclear ash, the front is subject to Rayleigh-Taylor instabilities. These dramatically alter the shape and area of the burn region and, consequently, the duration and strength of the nuclear flash. Large problems sizes are necessary to accurately model this phenomenon because fine-scale features can dramatically affect the large-scale features.

As a preliminary step, we solve a Rayleigh-Taylor instability problem in a rectangular parallelepiped ($x, y \in [0, 0.25], z \in [0, 1]$) containing an ideal gas with $\gamma = 5/3$. Initially, the gas in the top half of the domain has density $\rho = 2$ and that in the bottom half has $\rho = 1$; thus, the Atwood Number is $1/3$. The interface between the two regions is sharp. Pressure is unity at the top of the domain and increases toward the bottom with hydrostatic gradient ρg, where $g = 1$ is the acceleration of gravity acting in the z direction. Far-field conditions are applied on the sides, and the pressure is prescribed at the top and bottom to maintain the hydrostatic equilibrium. An initial single-mode sinusoidal velocity perturbation [30] is

$$V_z = -\epsilon_z \cos 8\pi x \cos 8\pi y \sin^\tau \pi z,$$

$$V_x = \epsilon_{xy} \sin 8\pi x \cos 8\pi y \cos \pi z \sin^{\tau-1} \pi z,$$

$$V_y = \epsilon_{xy} \cos 8\pi x \sin 8\pi y \cos \pi z \sin^{\tau-1} \pi z,$$

where

$$\epsilon_z = M_0 \sqrt{\gamma/2}, \qquad \epsilon_{xy} = -\epsilon_z \tau/16.$$

The velocity perturbation has magnitude $M_0 = 0.05$ and "tapers off" from the interface with a factor of $\tau = 6$. The planar cross terms V_x and V_y are used for consistency with other software and are of importance with incompressible flows.

6 Computational Results

The Rayleigh-Taylor problem was solved on 128 processors (32 4-way SMP nodes) of an IBM SP computer. Error indicator and refinement tolerances were chosen to detect the interface between the high- and low-density regions, and refined to a given edge length in that region. The initial mesh consisted of 234,421 regions. At $t = 0.28$, the mesh has been adaptively refined to 5,116,334 regions. Octree partitioning with a Morton traversal (OCTPART) and interprocessor boundary smoothing [11] was used to rebalance the computational load after each adaptive enrichment. Details regarding the parallel efficiency of OCTPART and other tools used in this computation are reported elsewhere [12,11].

Figure 3 shows the fluid density at $t = 0.28$ with (left) and without (right) mesh projections on a plane through the center of the domain. The instability is beginning; however, additional computation is necessary to see the complex flow that develops. The adaptive process has clearly concentrated the mesh in the interface zone. The interface is much more sharply defined than previous simulations, which employed a van Leer flux vector splitting rather than the Colella and Woodward fluxes.

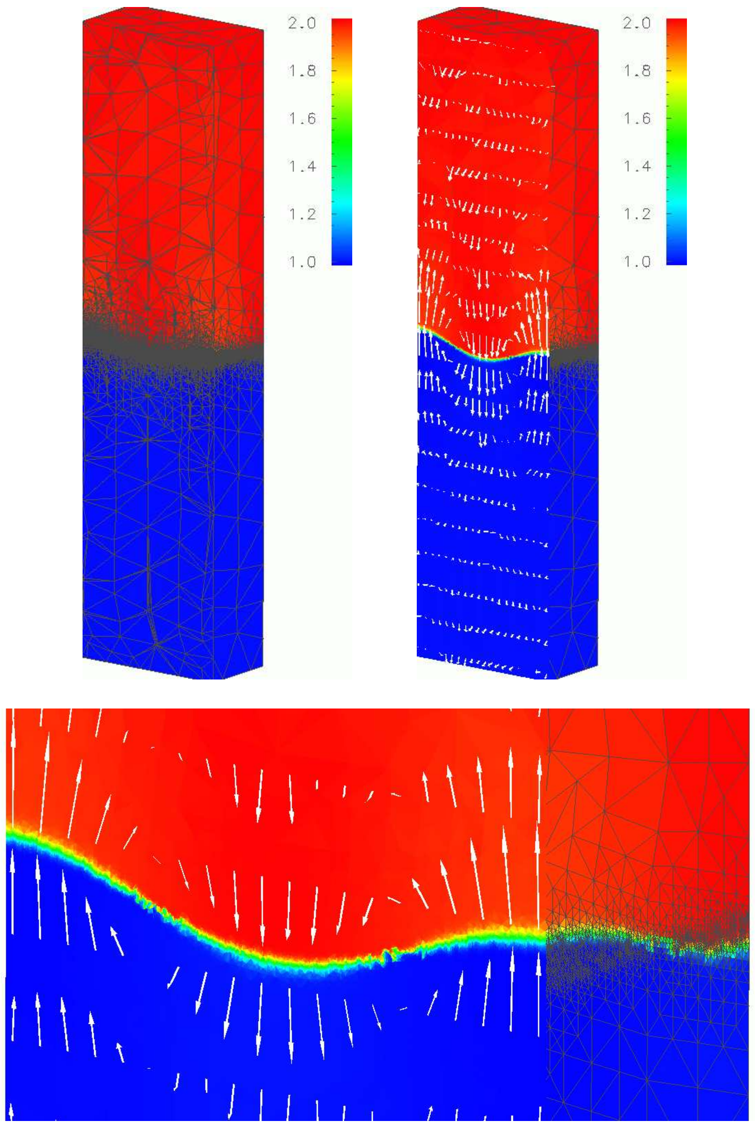

Fig. 3. Densities at $t = 0.28$ for a Rayleigh-Taylor flow projected on a plane through the center of the domain. Densities range from 1 (blue) to 2 (red). The projection on the upper left includes the mesh. Arrows shown on the cut plane at the upper right indicate velocity. The projection at the bottom is a closer view of the interface zone.

7 Discussion

The Rayleigh-Taylor problem is a complex and severe test of an adaptive solution procedure. Additional computations are ongoing. With meshes ranging into millions of regions, we need a hierarchical visualization system to examine the results. Such a system is under development using an octree decomposition of the spatial domain.

A higher-order basis would reduce the spurious diffusion of the piecewise-constant basis used here. This has been developed for two-dimensional flows [3] and is being incorporated into the three-dimensional software. As noted, higher-order requires limiting and the adaptive procedure of Biswas et al. [3] is being extended to unstructured meshes for this purpose. Estimates of discretization errors will be needed both to evaluate accuracy and to guide adaptive enrichment. Possibilities for these include use of superconvergence at Radau points [3] (although extending this idea to unstructured meshes presents a challenge) and the linear-problem estimates of Bey et al. [2]. Adaptive p- and hp-refinement procedures will be possible once these developments have been completed.

RPM is capable of handling the heterogeneities introduced by p-refinement. All of the load balancing procedures include capabilities to weight mesh entities. As noted, weighting due to the LRM was included here. While procedures to handle communications hierarchies are in place (§3), these have to be examined more closely and extended to account for memory hierarchies (cache utilization).

Acknowledgments

We thank Bruce Fryxell and Henry Tufo for many helpful discussions regarding the Rayleigh-Taylor problem and for providing software for the initial perturbation and Riemann flux evaluation.

Flaherty, Shephard, and Teresco were supported by the NSF under Grant ASC-9720227, by the USARO under Contract DAAG55-98-1-0200, by the DOE under Phase 1 ASCI Grant B341495, and by Simmetrix, Inc. Loy was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research Research, U.S. Department of Energy, under Contract W-31-109-Eng-38. The U.S. Government retains for itself, and others acting on its behalf, a paid-up, nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. Computer systems used include the IBM SP systems at the Maui High Performance Computing Center, the IBM SP at Rensselaer, and the IBM SP Blue-Pacific at Lawrence Livermore National Laboratory.

References

1. M. W. Beall and M. S. Shephard. A general topology-based mesh data structure. *Int. J. Numer. Meth. Engng.*, 40(9):1573–1596, 1997.

2. K. S. Bey, A. Patra, and J. T. Oden. *hp*-version discontinuous Galerkin methods for hyperbolic conservation laws: a parallel adaptive strategy. *Int. J. Numer. Meth. Engng.*, 38(22):3889–3907, 1995.
3. R. Biswas, K. D. Devine, and J. E. Flaherty. Parallel, adaptive finite element methods for conservation laws. *Appl. Numer. Math.*, 14:255–283, 1994.
4. K. Clark, J. E. Flaherty, and M. S. Shephard. *Appl. Numer. Math., special ed. on Adaptive Methods for Partial Differential Equations*, 14, 1994.
5. B. Cockburn, S.-Y. Lin, and C.-W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-Dimensional systems. *J. Comput. Phys.*, 84:90–113, 1989.
6. B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework. *Math. Comp.*, 52:411–435, 1989.
7. P. Colella. Glimm's method for gas dynamics. *SIAM J. Scien. Stat. Comput.*, 3(1):76–110, 1982.
8. P. Colella and P. R. Woodward. The piecewise parabolic (PPM) method for gas-dynamical simulations. *J. Comput. Phys.*, 54:174–201, 1984.
9. K. D. Devine, J. E. Flaherty, R. Loy, and S. Wheat. Parallel partitioning strategies for the adaptive solution of conservation laws. In I. Babuška, J. E. Flaherty, W. D. Henshaw, J. E. Hopcroft, J. E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75, pages 215–242, Berlin-Heidelberg, 1995. Springer-Verlag.
10. J. E. Flaherty, M. Dindar, R. M. Loy, M. S. Shephard, B. K. Szymanski, J. D. Teresco, and L. H. Ziantz. An adaptive and parallel framework for partial differential equations. In D. F. Griffiths, D. J. Higham, and G. A. Watson, editors, *Numerical Analysis 1997 (Proc. 17th Dundee Biennial Conf.)*, number 380 in Pitman Research Notes in Mathematics Series, pages 74–90. Addison Wesley Longman, 1998.
11. J. E. Flaherty, R. M. Loy, C. Özturan, M. S. Shephard, B. K. Szymanski, J. D. Teresco, and L. H. Ziantz. Parallel structures and dynamic load balancing for adaptive finite element computation. *Appl. Numer. Math.*, 26:241–263, 1998.
12. J. E. Flaherty, R. M. Loy, M. S. Shephard, B. K. Szymanski, J. D. Teresco, and L. H. Ziantz. Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. IMA Preprint Series 1483, Institute for Mathematics and its Applications, University of Minnesota, 1997. To appear, *J. Parallel and Dist. Comput.*
13. B. Fryxell. *Personal communication.*
14. B. Fryxell, E. Müller, and D. Arnett. Hydrodynamics and nuclear burning. *Max-Planck-Institut für Astrophysik Report 449*, 1989.
15. W. L. Kleb and J. T. Batina. Temporal adaptive Euler/Navier-Stokes algorithm involving unstructured dynamic meshes. *AIAA J.*, 30(8):1980–1985, 1992.
16. R. M. Loy. *Adaptive Local Refinement with Octree Load-Balancing for the Parallel Solution of Three-Dimensional Conservation Laws.* PhD thesis, Computer Science Dept., Rensselaer Polytechnic Institute, Troy, 1998.
17. R. A. Ludwig, J. E. Flaherty, F. Guerinoni, P. L. Baehmann, and M. S. Shephard. Adaptive solutions of the Euler equations using finite quadtree and octree grids. *Computers and Structures*, 30:327–336, 1988.
18. A. Malagoli. http://www.flash.uchicago.edu/scientific.htm. URL, 1997.
19. Message Passing Interface Forum, University of Tennessee, Knoxville, Tennessee. *MPI: A Message Passing Interface Standard*, first edition, 1994.

20. L. Oliker, R. Biswas, and R. C. Strawn. Parallel implementaion of an adaptive scheme for 3D unstructured grids on the SP2. In *Proc. 3rd International Workshop on Parallel Algorithms for Irregularly Structured Problems*, Santa Barbara, 1996.
21. M. S. Shephard, S. Dey, and J. E. Flaherty. A straightforward structure to construct shape functions for variable p-order meshes. *Comp. Meth. in Appl. Mech. and Engng.*, 147:209–233, 1997.
22. M. S. Shephard, J. E. Flaherty, C. L. Bottasso, H. L. de Cougny, C. Özturan, and M. L. Simone. Parallel automatic adaptive analysis. *Parallel Comput.*, 23:1327–1347, 1997.
23. M. S. Shephard, J. E. Flaherty, H. L. de Cougny, C. Özturan, C. L. Bottasso, and M. W. Beall. Parallel automated adaptive procedures for unstructured meshes. In *Parallel Comput. in CFD*, number R-807, pages 6.1–6.49. Agard, Neuilly-Sur-Seine, 1995.
24. C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *J. Comput. Phys.*, 27:1–31, 1978.
25. P. K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J. Numer. Anal*, 21:995–1011, 1984.
26. J. D. Teresco, M. W. Beall, J. E. Flaherty, and M. S. Shephard. A hierarchical partition model for adaptive finite element computation. To appear, *Comp. Meth. in Appl. Mech. and Engng.*, 1998.
27. B. Van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *J. Comput. Phys.*, 23:276–299, 1977.
28. B. Van Leer. Towards the ultimate conservative difference scheme. V. A second order sequel to Godunov's methods. *J. Comput. Phys.*, 32:101–136, 1979.
29. B. Van Leer. Flux vector splitting for the Euler equations. ICASE Report 82-30, ICASE, NASA Langley Research Center, Hampton, 1982.
30. Y.-N. Young, H. Tufo, and R. Rosner. On the miscible Rayleigh-Taylor instability: Mixing layer width scaling in 2 and 3-d. In progress, 1999.