UCRL-PROC-214613

LAWRENCE LIVERMORE NATIONAL LABORATORY

Solving Fluid Flow Problems on Moving and Adaptive Overlapping Grids

W.D. Henshaw

August 15, 2005

Parallel CFD 2005
Washington, DC, United States
May 24, 2005 through May 27, 2005

Disclaimer

Solving Fluid Flow Problems on Moving and Adaptive Overlapping Grids

William D. Henshaw [1]

Centre for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551.

Abstract

Solution of fluid dynamics problems on overlapping grids will be discussed. An overlapping grid consists of a set of structured component grids that cover a domain and overlap where they meet. Overlapping grids provide an effective approach for developing efficient and accurate approximations for complex, possibly moving geometry. Topics to be addressed include the reactive Euler equations, the incompressible Navier-Stokes equations and elliptic equations solved with a multigrid algorithm. Recent developments coupling moving grids and adaptive mesh refinement and preliminary parallel results will also be presented.

Key words:
PACS:

1. Introduction

The are many interesting problems that involve the solution of fluid dynamics problems on domains that evolve in time. Examples include the motion of valves in a car engine and the movement of embedded particles in a flow. The numerical solution of these problems is difficult since the discrete equations being solved change as the domain evolves. The problems can be especially hard when there are fine scale features in the flow such as shocks and detonations.

In this paper an approach will be described that uses composite overlapping grids to resolve complex geometry, moving component grids to track dynamically evolving surfaces and block structured adaptive mesh refinement (AMR) to efficiently resolve fine scale features.

The numerical method uses composite overlapping grids to represent the problem domain as a collection of structured curvilinear grids. This method, as discussed in Chesshire and Henshaw [1], allows complex domains to be represented with smooth grids that can be aligned with the boundaries. The use of smooth grids is particularly attractive for problems where the solution is sensitive to any grid induced numerical artifacts. This approach can also take advantage of the large regions typically covered by Cartesian grids. These Cartesian grids can be treated with efficient

Email address: `henshaw1@llnl.gov` (William D. Henshaw).

[1] This research was supported under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

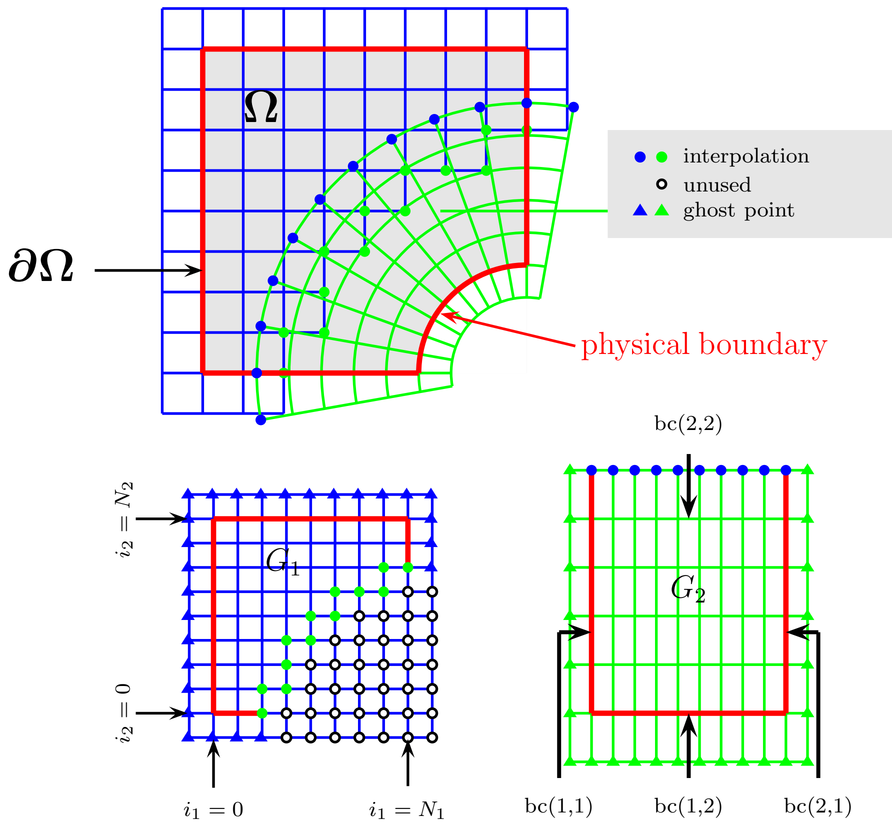

Fig. 1. The top view shows an overlapping grid consisting of two structured curvilinear component grids. The bottom views show the component grids in the unit square parameter space. Grid points are classified as discretization points, interpolation points or unused points. Ghost points are used to apply boundary conditions.

approximations leading to fast methods with low memory usage. Overlapping grids have been used successfully for the numerical solution of a variety of problems involving inviscid and viscous flows, see the references in [2,3] for example. The use of adaptive mesh refinement in combination with overlapping grids has been considered by Brislawn, Brown, Chesshire and Saltzman[4], Boden and Toro[5], and Meakin[6].

Figure 1 shows a simple overlapping grid consisting of two component grids, an annular grid and a background Cartesian grid. The top view shows the overlapping grid while the bottom view shows each grid in parameter space. In this example the annular grid cuts a hole in the Cartesian grid so that the latter grid has a number of unused points that are marked as open circles. The other points on the component grid are marked as discretization points (where the PDE or boundary conditions are discretized) and interpolation points. Solution values at interpolation points are generally determined by a tensor-product Lagrange interpolant in the parameter space of the donor grid. Ghost points are used to simplify the discretization of boundary conditions. In a moving grid computation one or more of the component grids will move, following the boundaries that move. As the grids move the overlapping connectivity information, such as the location of interpolation points, will be recomputed. In our work the grid generation is performed by the Ogen grid generator which has a specialized algorithm to treat moving grid problems efficiently. When adaptive mesh refinement is used on an overlapping grid, a hierarchy of refinement grids is added to the parameter space of each component grid. The locations of these refinement patches are determined by an appropriate error estimate. The software that we develop,

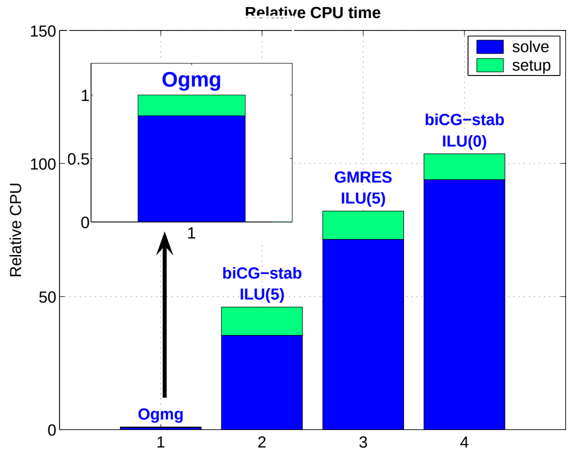

Fig. 3. In comparison to Krylov solvers the Ogmg multigrid solver is an order of magnitude faster and uses an order of magnitude less storage. These results are for the solution of a two-dimensional Poisson problem, for a cylinder in a square domain, using an overlapping grid with about 1 million grid points.

collectively known as the Overture framework, is freely available in source form [7].

2. Multigrid

A fast multigrid algorithm has been devised for solving elliptic boundary value problems on overlapping grids [3]. This method can be used to solve the implicit time-stepping equations and pressure equation in an incompressible Navier-Stokes solver, for example. In moving grid applications it is particularly important that the elliptic equation solver have a low startup cost since the equations will be changing at each time step. The Ogmg multigrid solver was developed to solve elliptic boundary value problems, in two and three space dimensions, of the form

$$
\begin{aligned}
Lu &= f \qquad \mathbf{x}\in\Omega\ , \\
Bu &= g \qquad \mathbf{x}\in\partial\Omega\ ,
\end{aligned}
$$

where L is chosen to be a second-order, linear, variable-coefficient operator and B is chosen to define a Dirichlet, Neumann or mixed boundary condition. The key aspects of the multigrid scheme for overlapping grids are an automatic coarse grid generation algorithm, an adaptive smoothing technique for adjusting residuals on different component grids, and the use of local smoothing near interpolation boundaries. Other important features include optimizations for Cartesian component grids, the use of over-relaxed Red-Black smoothers and the generation of coarse grid operators through Galerkin averaging. Numerical results in two and three dimensions show that very good multigrid convergence rates can be obtained for both Dirichlet and Neumann/mixed boundary conditions.

Figure 2 shows the solution and convergence rates when solving Poisson's equation on a region containing some spheres. The convergence rates are similar to the *text-book* convergence rates that one can obtain on single Cartesian grids.

Figure 3 presents a comparison of the multigrid solver to some Krylov based solvers for a two-dimensional problem with about 1 million grid points. The results show that that the multigrid solver can be much faster (over 45 times faster in this case) and also that the multigrid scheme has a low startup cost. Moreover, the multigrid solver uses about 10 times less memory in this case.

3. Solution of the reactive Euler Equations

The reactive Euler equations are solved on a domain $\Omega(t)$ whose boundaries, $\partial\Omega(t)$ may evolve in time. In two space dimensions the initial boundary value problem for the solution $\mathbf{u}=\mathbf{u}(\mathbf{x},t)$ is

$$
\begin{aligned}
\mathbf{u}_t+\mathbf{F}(\mathbf{u})_x+\mathbf{G}(\mathbf{u})_y &= \mathbf{H}(\mathbf{u}), \quad && \mathbf{x}\in\Omega(t)\ , \\
B(\mathbf{u}) &= 0, && \mathbf{x}\in\partial\Omega(t)\ , \\
\mathbf{u}(\mathbf{x},0) &= \mathbf{u}_0(\mathbf{x}),
\end{aligned}
$$

where

$$
\mathbf{u}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\\ \rho\mathbf{Y}\end{bmatrix},\quad
\mathbf{F}=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\\ \rho u\mathbf{Y}\end{bmatrix},
$$

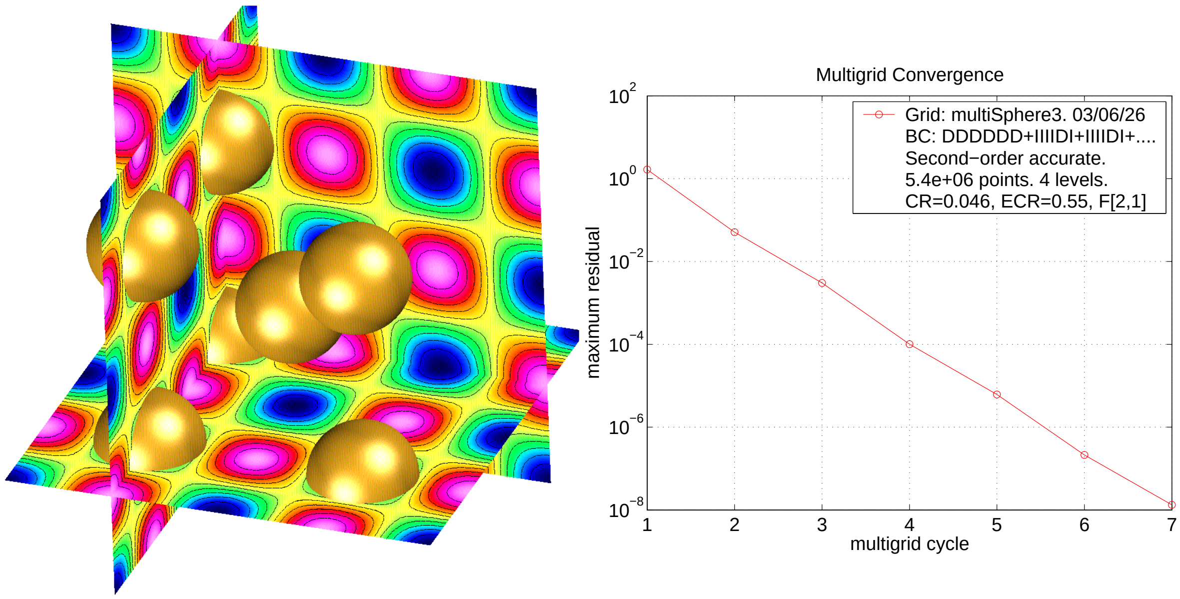

Fig. 2. Solution of Poisson's equation by the multigrid algorithm for a domain containing some spheres. The average convergence rate per F-cycle with 2 pre-smooths and 1-post smooth was about .046 .

$$\mathbf{G} = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ v(E+p) \\ \rho v \mathbf{Y} \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \rho \mathbf{R} \end{bmatrix}.$$

The state of the flow depends on the position $\mathbf{x} = (x, y) = (x_1, x_2)$ and the time t and is described by its density ρ, velocity $\mathbf{v} = (u, v)$, pressure p and total energy E. The flow is a mixture of m_r reacting species whose mass fractions are given by $\mathbf{Y}$. The source term models the chemical reactions and is described by a set of m_r rates of species production given by $\mathbf{R}$. The total energy is taken to be

$$E = \frac{p}{\gamma - 1} + \tfrac{1}{2}\rho\left(u^2 + v^2\right) + \rho q,$$

where γ is the ratio of specific heats and q represents the heat energy due to chemical reaction.

These equations are discretized, as part of the OverBlown solver, with a high-order accurate Godunov scheme coupled to an adaptive Runge-Kutta time stepper for the stiff source terms that model the chemistry [2]. Figure 4 shows results of a computation of a detonation diffracting around a corner. The detonation locally fails in the expansion region.

The motion of a rigid body $\mathcal{B}$ embedded in the flow is governed by the Newton-Euler equations. Let M^b be the mass of the body, $\mathbf{x}^b(t)$, and $\mathbf{v}^b(t)$ the position and velocity of the center of mass, $\mathcal{I}_i$ the moments of inertia, $\boldsymbol{\omega}_i$ the angular velocities about the principal axes of inertial, $\mathbf{e}_i$, $\mathbf{F}^b(t)$ the resultant force, and $\mathbf{G}^b(t)$ the resultant torque about $\mathbf{x}^b(t)$. The Newton-Euler equations are then

$$\begin{aligned} \frac{d\mathbf{x}^b}{dt} &= \mathbf{v}^b \ , \qquad M^b \frac{d\mathbf{v}^b}{dt} = \mathbf{F}^b \ , \\ \mathcal{I}_i \dot{\omega}_i - (\mathcal{I}_{i+1} - \mathcal{I}_{i+2})\omega_{i+1}\omega_{i+2} &= \mathbf{G}^b \cdot \mathbf{e}_i \ , \\ \dot{\mathbf{e}}_i &= \boldsymbol{\omega} \times \mathbf{e}_i \quad i = 1, 2, 3 \ , \end{aligned}$$

where the subscripts on $\mathcal{I}_i$ and ω_i are to be taken modulo 3 in the sense $\mathcal{I}_{i+1} := \mathcal{I}_{(i \bmod 3)+1}$. The force on the body will be a sum of body forces, $\mathbf{B}^b$, such as those arising from buoyancy, plus hydrodynamic forces on the boundary of the body, exerted by the fluid stresses,

$$\mathbf{F}^b = \mathbf{B}^b + \int_{\partial\mathcal{B}} \boldsymbol{\tau} \cdot \mathbf{n} \, dS \ .$$

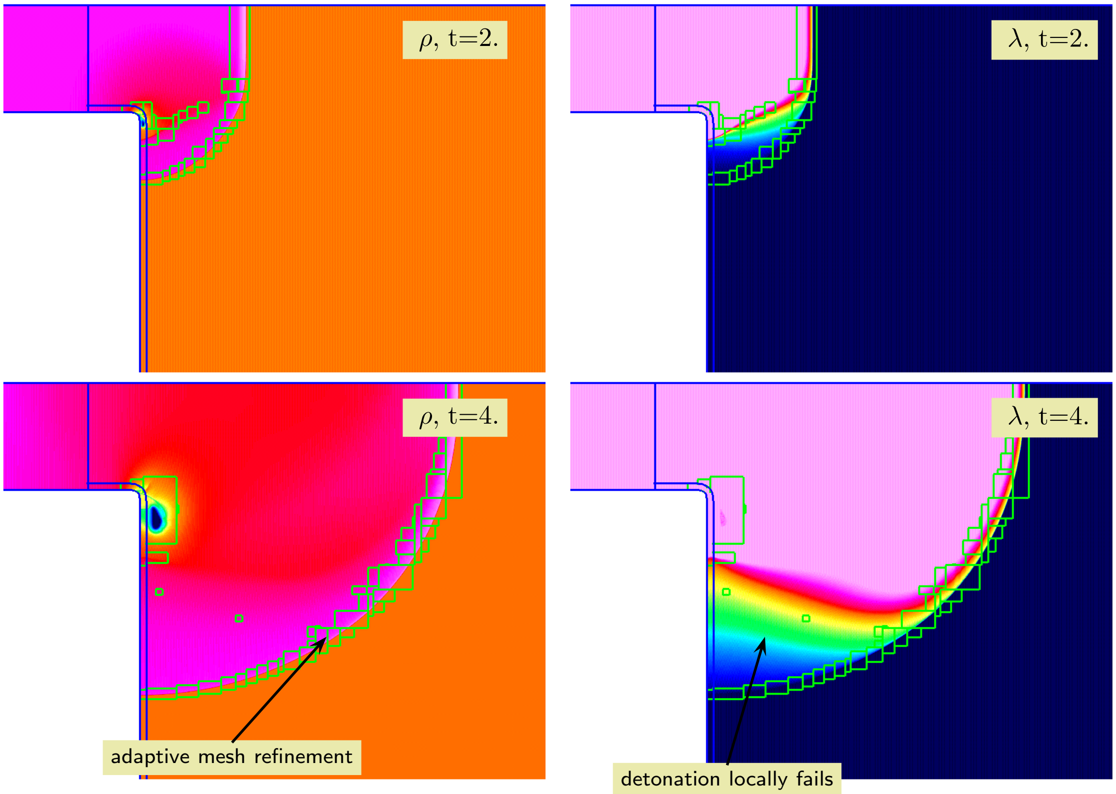

Fig. 4. Diffraction of a detonation by a corner. The computations were performed with an ignition and growth model and a JWL equation of state. The density and reaction progress variable are shown. The boundaries of the component base grids and of the AMR grids are also displayed.

Here the integral is over the surface of the rigid body, $\partial\mathcal{B}$. The torque $\mathbf{G}^b$ is given by

$$\mathbf{G}^b = \int_{\partial\mathcal{B}} (\mathbf{r} - \mathbf{x}^b) \times \; \boldsymbol{\tau} \cdot \mathbf{n} \, dS.$$

In the case of the Euler-equations, the stress tensor is simply $\boldsymbol{\tau} = -p\mathbf{I}$; the effects of viscosity are assumed to be negligible.

Figure 5 shows a computation of a shock hitting a collection of cylinders. The cylinders are rigid bodies that move due to the hydrodynamic forces. Adaptive mesh refinement is used in combination with moving grids. The grids around each cylinder move at each time step. The refinement grids move with their underlying base grid. The locations of all refinement grids are recomputed every few time steps. More details on this approach will be available in a forthcoming article.

4. Incompressible Flow

The incompressible Navier-Stokes equations are solved using the velocity-pressure formulation,

$$\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \nu\Delta\mathbf{u} \\ \Delta p + \nabla\cdot(\mathbf{u}\cdot\nabla\mathbf{u}) &= \alpha(\mathbf{x})\nabla\cdot\mathbf{u} \end{aligned}\right\} \quad \mathbf{x}\in\Omega \ ,$$

with boundary conditions

$$\left.\begin{aligned} B(\mathbf{u},p) &= 0 \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\partial\Omega \ ,$$

and initial conditions

$$\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0(\mathbf{x}) \quad \text{at } t=0 \ .$$

The term $\alpha(\mathbf{x})\nabla\cdot\mathbf{u}$ is used to damp the dilatation. The boundary condition $\nabla\cdot\mathbf{u} = 0$ is the additional

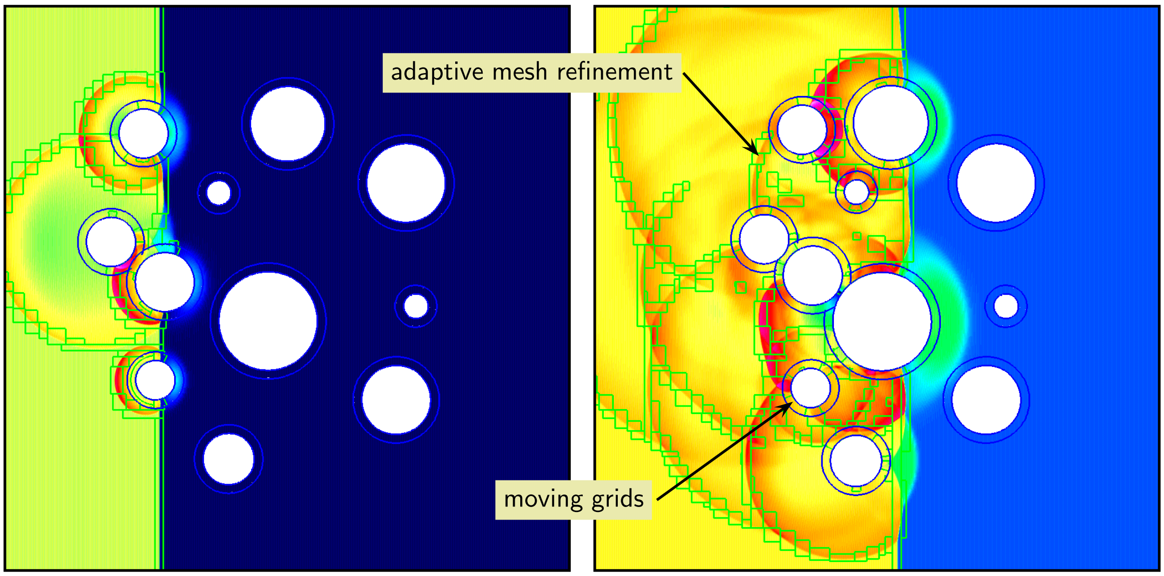

Fig. 5. Computation of a shock hitting a collection of cylinders, the density is shown at two different times. This computation illustrates the use of moving grids and adaptive mesh refinement. Each cylinder is treated as a rigid body that moves according to the forces exerted by the fluid. The boundaries of the component base grids and the AMR grids are also shown.

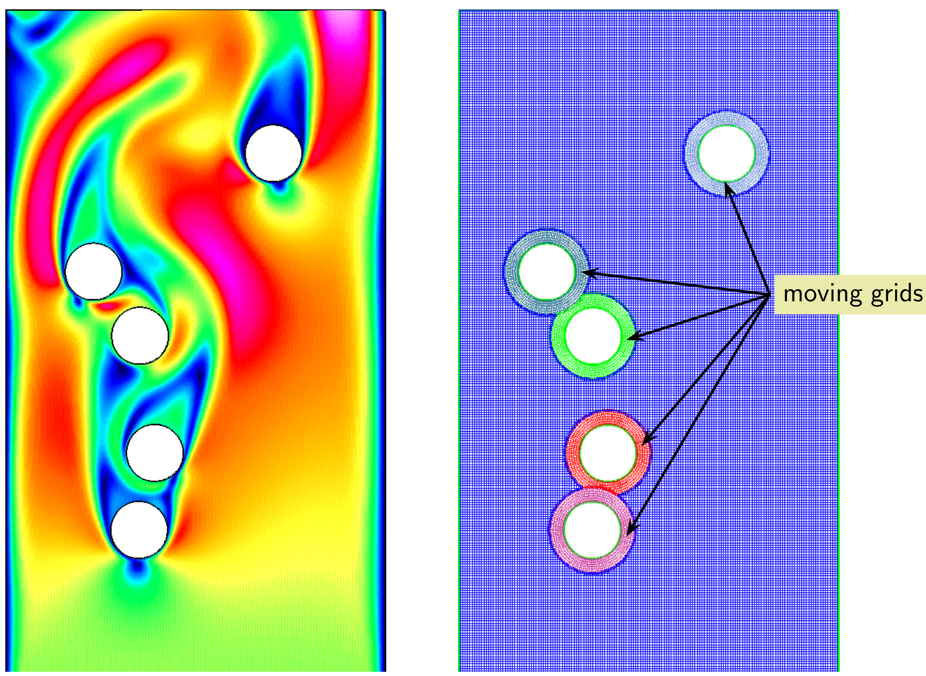

Fig. 6. Falling cylinders in an incompressible flow. Left: a contour plot of the magnitude of the velocity. Right: A coarsened version of the overlapping grid. The cylinders have a mass and move under the forces of gravity and the forces exerted by the viscous fluid. The annular grids are moved at each time step. A Poisson equation for the pressure is solved using the overlapping grid multigrid solver Ogmg.

pressure boundary condition needed for this formulation to ensure that the dilatation is zero everywhere. The numerical scheme is a split-step formulation. The velocity is advanced first in an explicit or implicit manner. The pressure is then determined. The scheme has been implemented using second-order and fourth-order accurate approximations using a predictor-corrector time stepping scheme. The discretization of the boundary conditions on no-slip walls is an important element of the scheme. This solution algorithm is also implemented in the OverBlown code. See [8,9] for further details.

Figure 6 shows results of a computation of some rigid cylinders falling through a channel containing a viscous fluid. The grids around each cylinder move at each time step according to the Newton-Euler equations of motion. The Ogen grid generator is used to update the overlapping grid connectivity information. The multigrid solver is used to solve the pressure equation. The Poisson equation changes at each time step but this equation can be treated efficiently with the multigrid solver. More details on the approach used to solve the incompressible equations with moving grids will appear in a future publication.

5. Parallel computations

In a distributed parallel computing environment, each component grid-function (representing the solution variables such as ρ, $\mathbf{u}$, p, etc.) can be distributed across one or more processors. The grid functions are implemented using a parallel distributed arrays from the P++ array class library[10]. Each P++ array can be independently distributed across the available processors. The distributed array consists of a set of serial arrays, one serial array for each processor. Each serial array is a multi-dimensional array that can be operated on using array operations. The serial array can also be passed to a Fortran function, for example. The serial arrays contain extra ghost lines that hold copies of the data from the serial arrays on neighbouring processors. P++ is built on top of the Multiblock PARTI parallel communication library [11] which is used for ghost boundary updates and copying blocks of data between arrays with possibly different distributions. Figure 7 presents a section of C++ code showing the use of P++ arrays.

A special parallel overlapping grid interpolation routine has been developed for updating the points on grids that interpolate from other grids, see Figure 1. Overlapping grid interpolation is based on a multi-dimensional tensor product Lagrange interpolant. In parallel, the Lagrange formula is evaluated on the processor that owns the data in the stencil (the donor points), the resulting sums are collected into a message and then sent to the processor that owns the interpolation point.

Figure 8 shows some preliminary parallel results from solving the Euler equations on an overlapping grid with the parallel version of OverBlown. The speed-up running on up to 32 processors is shown for a problem with a fixed number of grid points. The scaling is quite reasonable. The computations were performed on a Linux cluster.

6. Acknowledgments

The development of the reactive Euler equation solver was performed in collaboration with Professor Don Schwendeman. Thanks also to Kyle Chand and Jeffrey Banks for contributions to this work.

References

[1] G. Chesshire, W. Henshaw, Composite overlapping meshes for the solution of partial differential equations, J. Comp. Phys. 90 (1) (1990) 1–64.

[2] W. D. Henshaw, D. W. Schwendeman, An adaptive numerical scheme for high-speed reactive flow on overlapping grids, J. Comp. Phys. 191 (2003) 420–447.

[3] W. D. Henshaw, On multigrid for overlapping grids, SIAM Journal of Scientific Computing 26 (5) (2005) 1547–1572.

[4] K. Brislawn, D. L. Brown, G. Chesshire, J. Saltzman, Adaptively-refined overlapping grids for the numerical solution of hyperbolic systems of conservation laws, report LA-UR-95-257, Los Alamos National Laboratory (1995).

```
Partitioning_Type partition; // object that defines the parallel distribution
partition.SpecifyInternalGhostBoundaryWidths(1,1);

realDistributedArray u(100,100,partition); // build a distributed array
Range I(1,98), J(1,98);

// Parallel array operation with automatic communication:
u(I,J)=.25*( u(I+1,J) + u(I-1,J) + u(I,J+1) + u(I,J-1) ) + sin(u(I,J))/3.;

// Access local serial arrays and call a Fortran routine:
realSerialArray & uLocal = u.getLocalArray(); // access the local array
myFortranRoutine(*uLocal.getDataPointer(),...);
u.updateGhostBoundaries(); // update ghost boundaries on distributed arrays
```

Fig. 7. The P++ class library provides parallel multi-dimensional arrays. The class supports array operations with automatic communication. It is also possible to use Fortran or C kernels to operate on each local serial array.

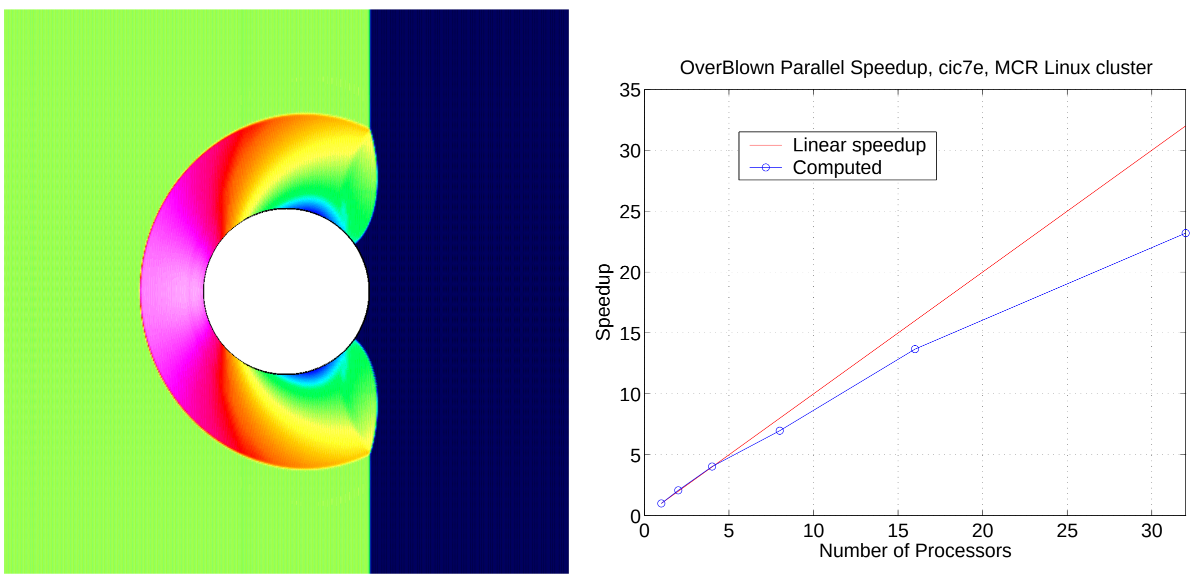

Fig. 8. Left: the computation of a shock hitting a cylinder (density). Right: parallel speedup for this problem, keeping the problem size fixed (4 million grid points), on a linux cluster (Xeon processors).

[5] E. P. Boden, E. F. Toro, A combined Chimera-AMR technique for computing hyperbolic PDEs, in: Djilali (Ed.), Proceedings of the Fifth Annual Conference of the CFD Society of Canada, 1997, pp. 5.13–5.18.

[6] R. L. Meakin, Composite overset structured grids, in: J. F. Thompson, B. K. Soni, N. P. Weatherill (Eds.), Handbook of Grid Generation, CRC Press, 1999, Ch. 11, pp. 1–20.

[7] D. Brown, K. Chand, P. Fast, W. Henshaw, A. Petersson, D. Quinlan, Overture, Tech. Rep. http://www.llnl.gov/CASC/Overture.html, Lawrence Livermore National Laboratory (2003).

[8] W. Henshaw, A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids, J. Comp. Phys. 113 (1) (1994) 13–25.

[9] W. D. Henshaw, N. A. Petersson, A split-step scheme for the incompressible Navier-Stokes equations, in: M. Hafez (Ed.), Numerical Simulation of Incompressible Flows, World Scientific, 2003, pp. 108–125.

[10] D. Quinlan, A++/P++ class libraries, Research Report LA-UR-95-3273, Los Alamos National Laboratory (1995).

[11] A. Sussman, G. Agrawal, J. Saltz, A manual for the Multiblock PARTI runtime primitives, revision 4.1, Technical Report CS-TR-3070.1, University of Maryland, Department of Computer Science (1993).