An Overture Overview

Bill Henshaw

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

www.llnl.gov/casc/Overture

Presented at the Sixth DOE ACTS Collection Workshop, August 2005.

Acknowledgments

Supported by

Department of Energy, Office of Science
- MICS Program: Mathematical, Information, and Computational Sciences
- SciDAC: Scientific Discovery through Advanced Computing
  - TSTT: Terascale Simulation Tools and Technology Centre

Current Overture developers

Kyle Chand
Bill Henshaw

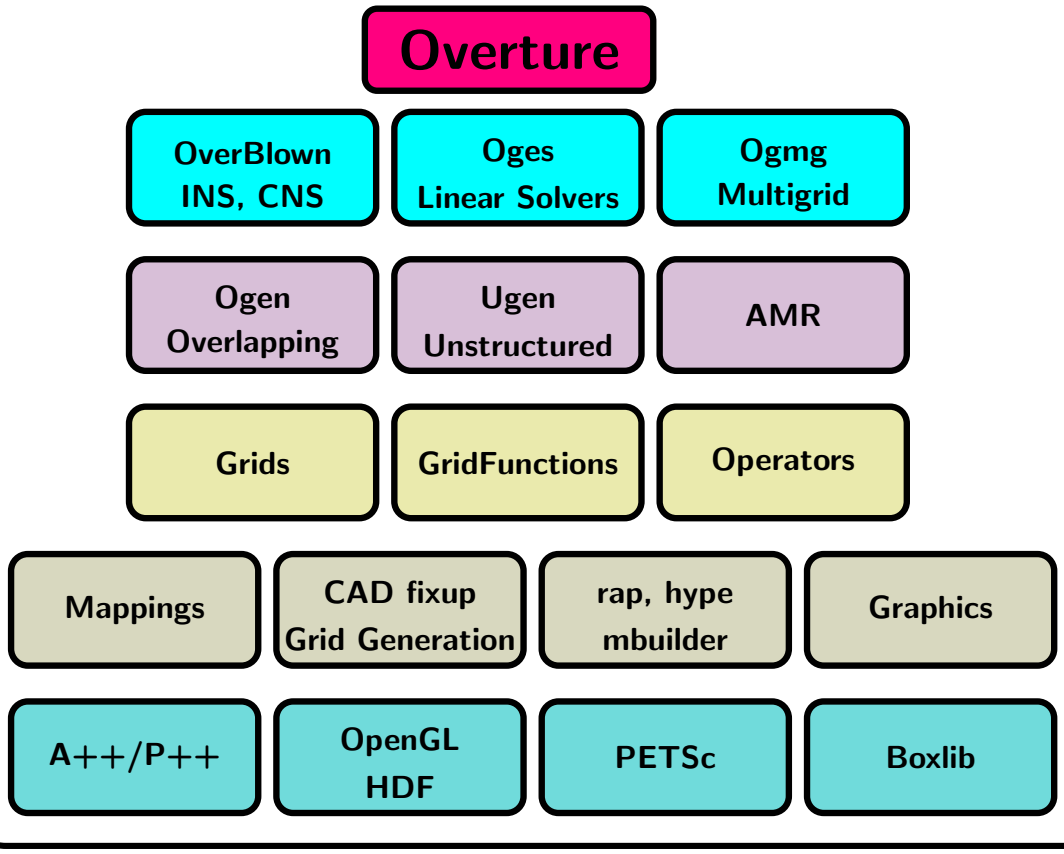
Collaborators

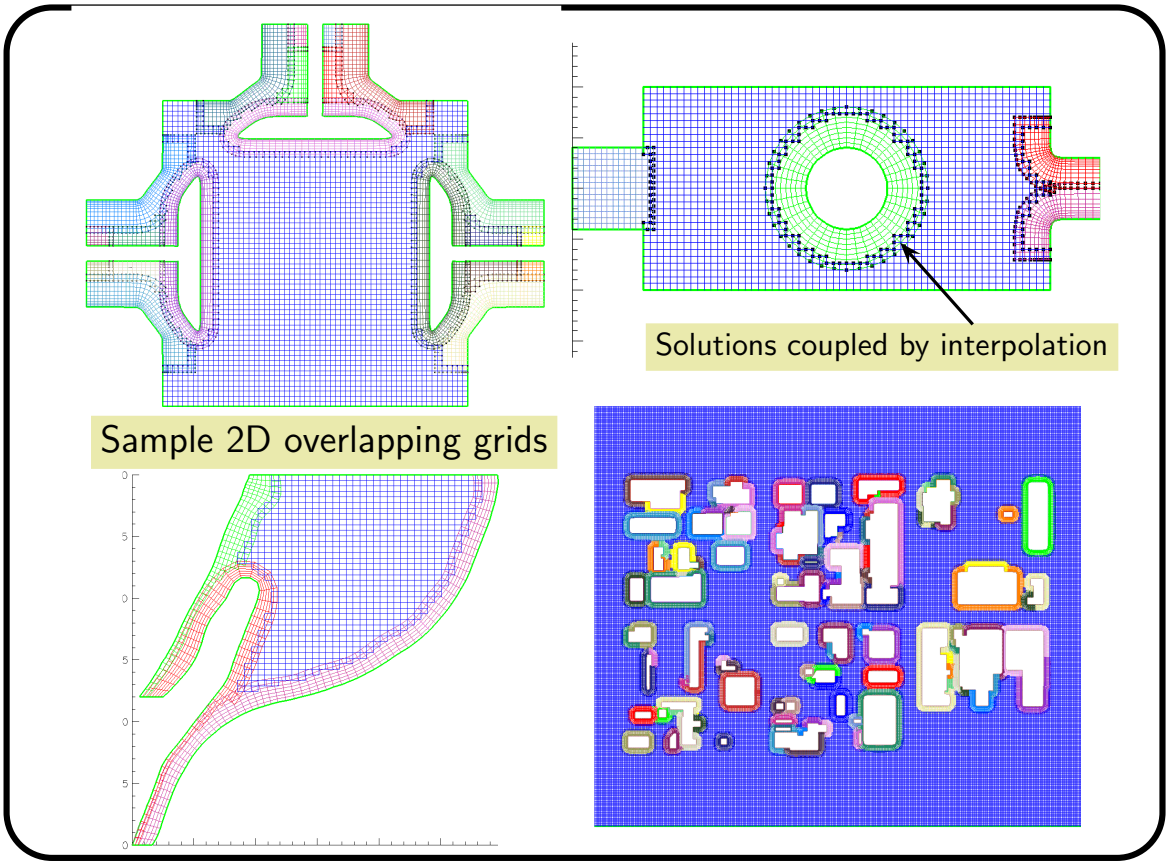
Petri Fast (LLNL)
Don Schwendeman (RPI)
Jeff Banks (RPI)

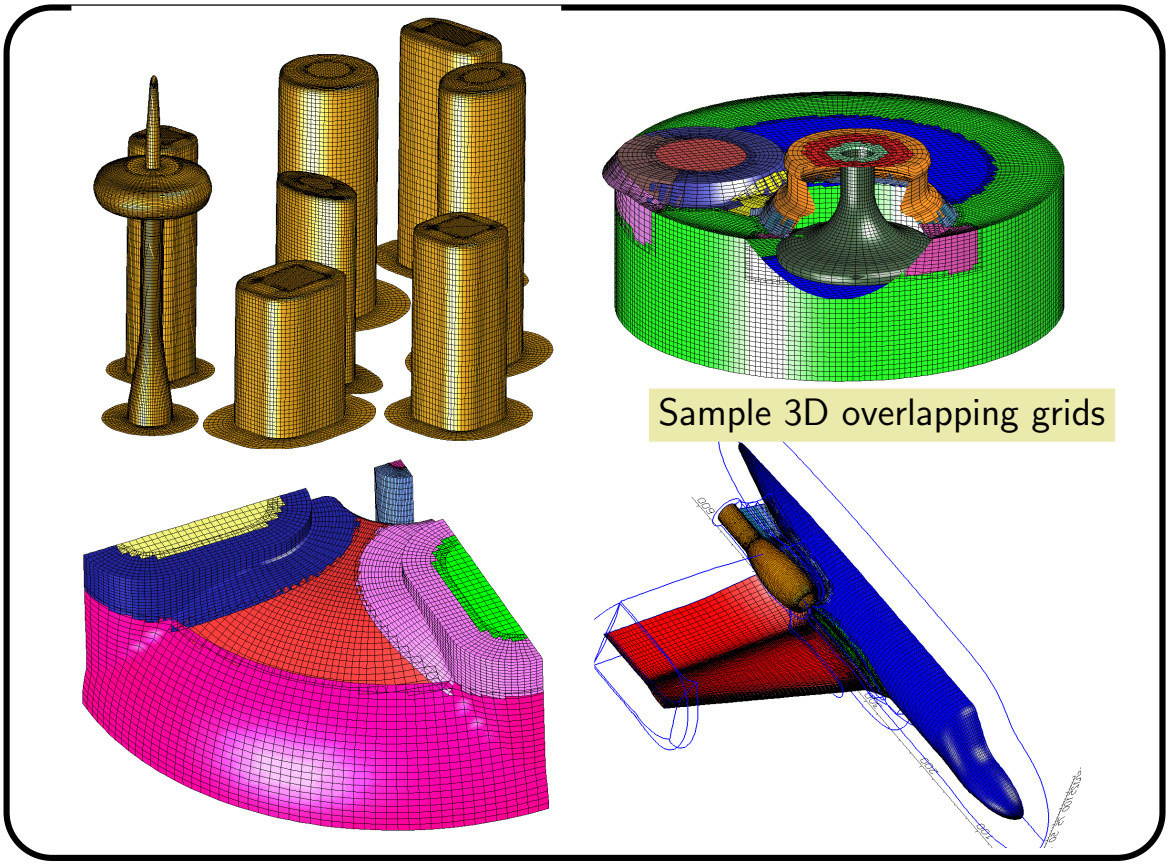
Overture is a collection of C++ classes that can be used to solve partial differential equations on structured, overlapping and hybrid grids.

Key features:

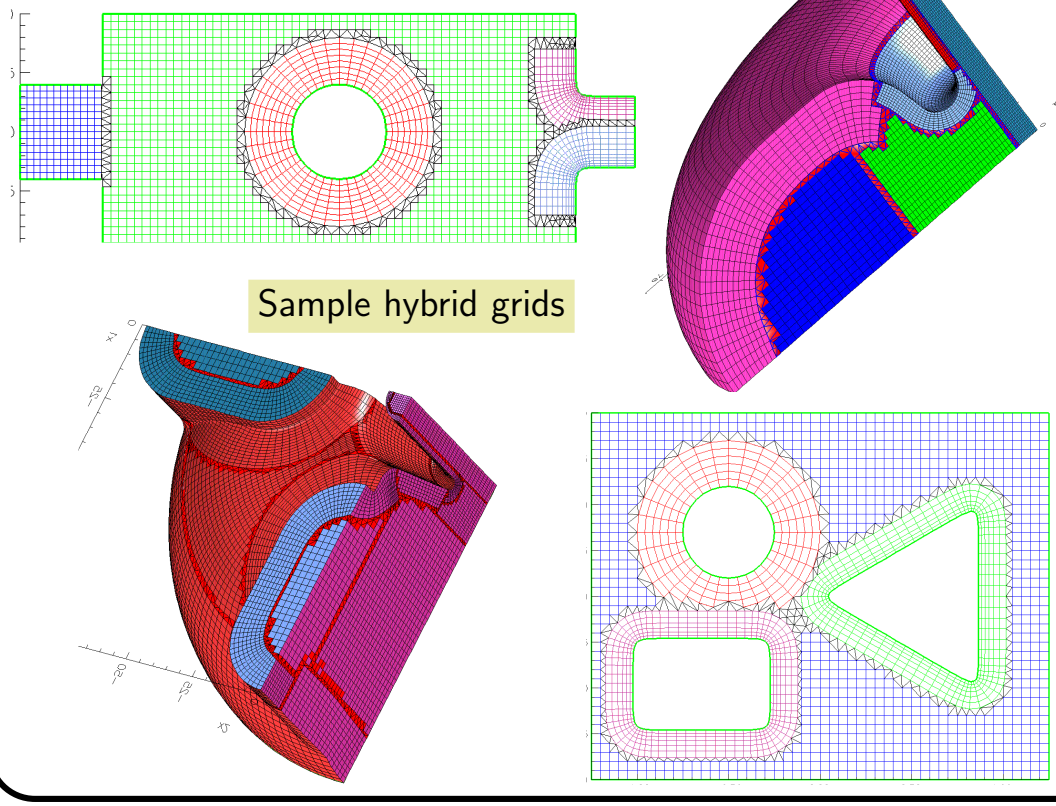
- provides a high level interface for rapid prototyping of PDE solvers.
- built upon optimized C and fortran kernels.
- provides a library of finite-difference operators: conservative and non-conservative, 2nd, 4th, 6th and 8th order accurate approximations.
- support for moving grids
- support for block structured adaptive mesh refinement
- extensive grid generation capabilities
- CAD fixup tools
- interactive graphics and data base support.
- PDE solvers built upon Overture include:
  - OverBlown: incompressible Navier-Stokes, compressible Navier-Stokes, reactive Euler equations.
  - Max: time domain Maxwell's equations solver: fourth-order accurate, parallel.



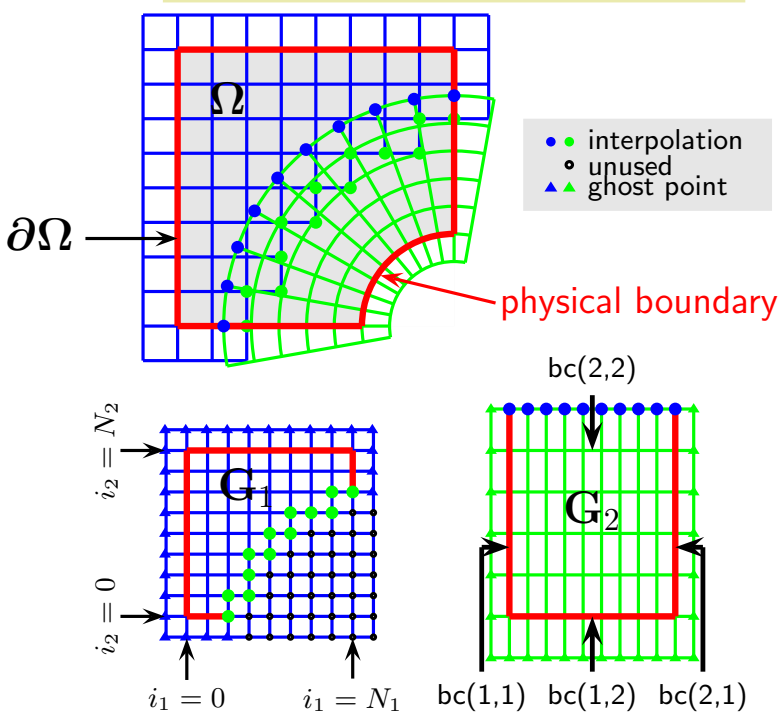

Solutions coupled by interpolation
Sample 2D overlapping grids


Sample 3D overlapping grids

Sample hybrid grids



Components of an Overlapping Grid



Overture supports a high-level C++ interface (but is built mainly upon Fortran kernels):

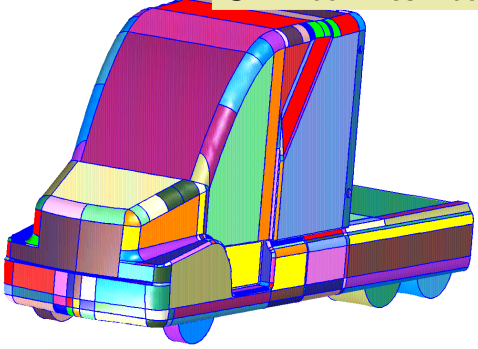
Solve $u_t + au_x + bu_y = \nu(u_{xx} + u_{yy})$

```
CompositeGrid cg;  // create a composite grid
getFromADataBaseFile(cg,"myGrid.hdf");
floatCompositeGridFunction u(cg);  // create a grid function
u=1.;
CompositeGridOperators op(cg);  // operators
u.setOperators(op);
float t=0, dt=.005, a=1., b=1., nu=.1;
for( int step=0; step<100; step++ )
{
  u+=dt*( -a*u.x()-b*u.y()+nu*(u.xx()+u.yy()) ); // forward Euler
  t+=dt;
  u.interpolate();
  u.applyBoundaryCondition(0,dirichlet,allBoundaries,0.);
  u.finishBoundaryConditions();
}
```

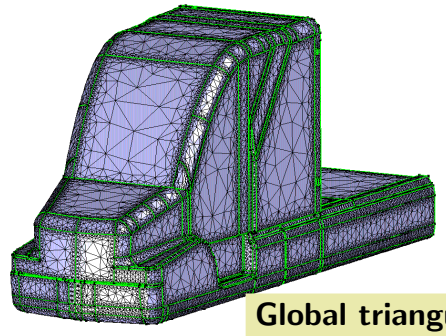

Current Projects with Overture

- Hybrid (unstructured) grid generation and algorithms (Kyle Chand)
  - the overlap region is replaced by an unstructured grid (advancing front algorithm).
  - a stabilized DSI scheme for Maxwell's equations on hybrid grids.
- Deforming boundaries in incompressible flow (Petri Fast).
- Multigrid solvers for elliptic problems on overlapping grids.
  - robust coarsening and adaptive smoothing techniques.
  - second- and fourth-order accurate, Dirichlet and Neumann boundary conditions.
- Incompressible Navier-Stokes solvers for overlapping grids.
  - fourth-order accurate time accurate solver.
  - line-implicit pseudo-steady state solver.
- High speed reactive flow and adaptive mesh refinement (with Don Schwendeman (RPI))
- An overlapping grid solver for the time dependent Maxwell's equations.
  - fourth-order accurate, parallel

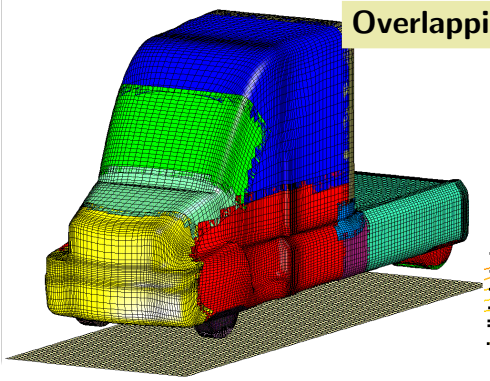
CAD to Mesh to Solution with Overture



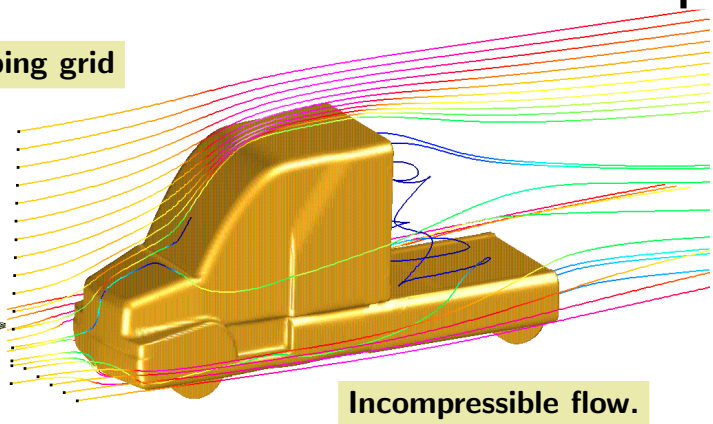
Cad fixup



Global triangulation



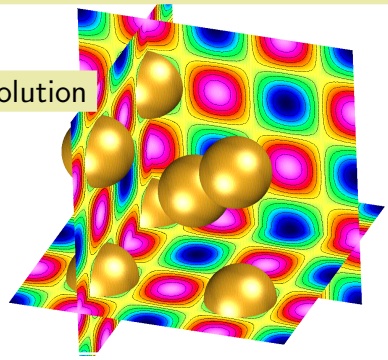
Overlapping grid

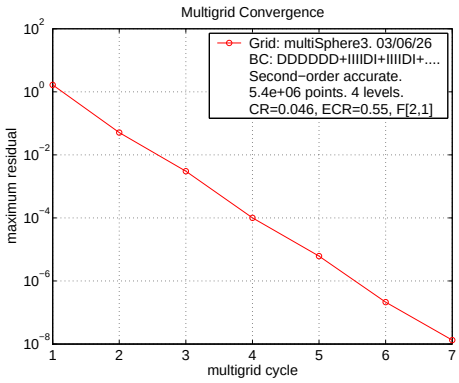


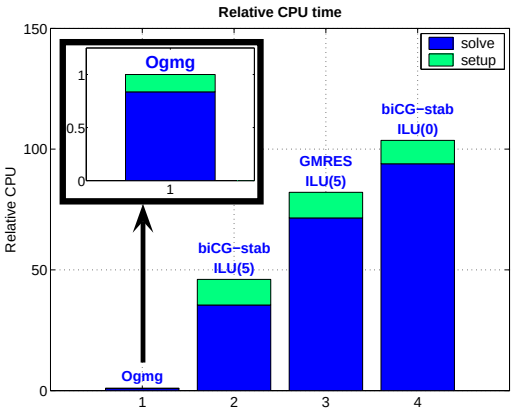
Incompressible flow.

Multigrid solution to Poisson's equation, 5.4 million grid points

solution



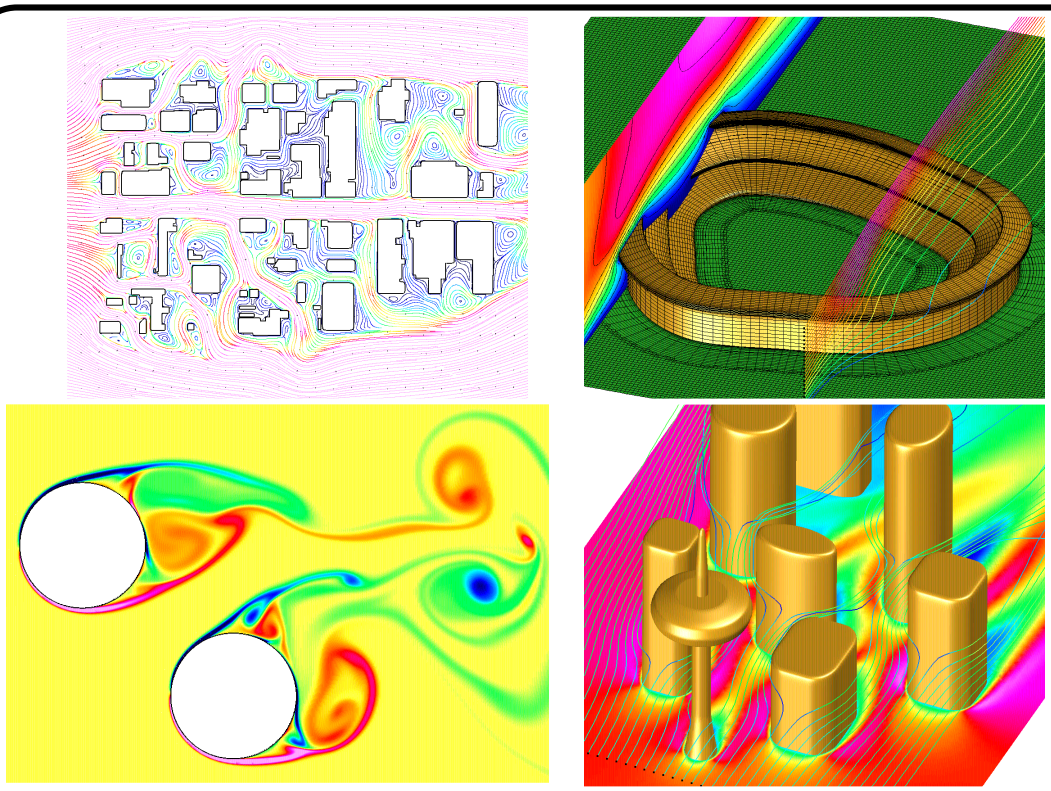




Mesh independent convergence rates

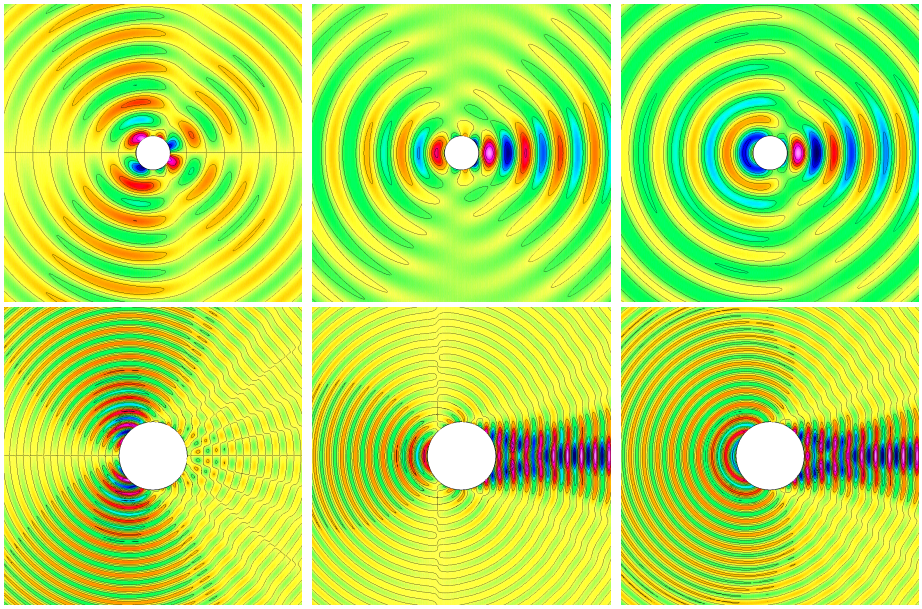
New adaptive MG for overlapping grids.

In comparison to Krylov solvers multigrid is an order of magnitude faster and uses an order of magnitude less storage



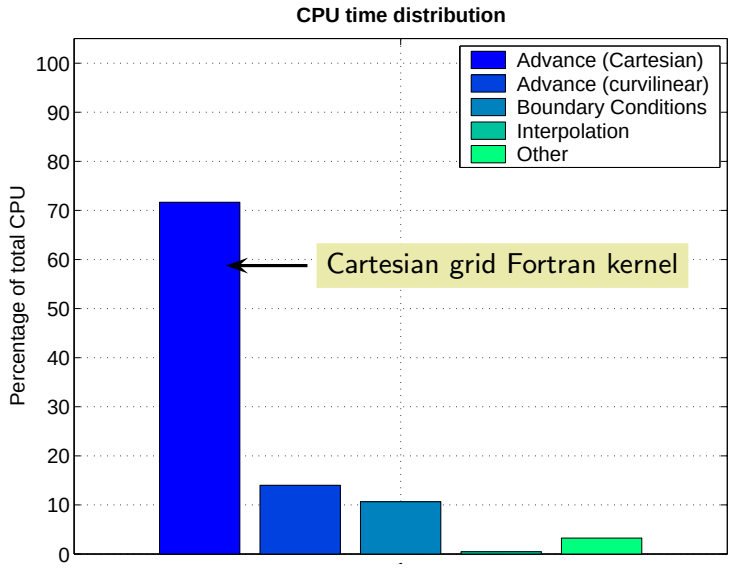
Incompressible flow computations with OverBlown.

A Parallel 4th-order accurate solver for the time-dependent Maxwell equations



Scattering of a plane wave by a cylinder. Top: scattered field E_x, E_y and H_z for $ka = 1/2$. Bottom: scattered field E_x, E_y and H_z for $ka = 5/2$

Performance of overlapping grid codes can approach that of Cartesian grid codes.

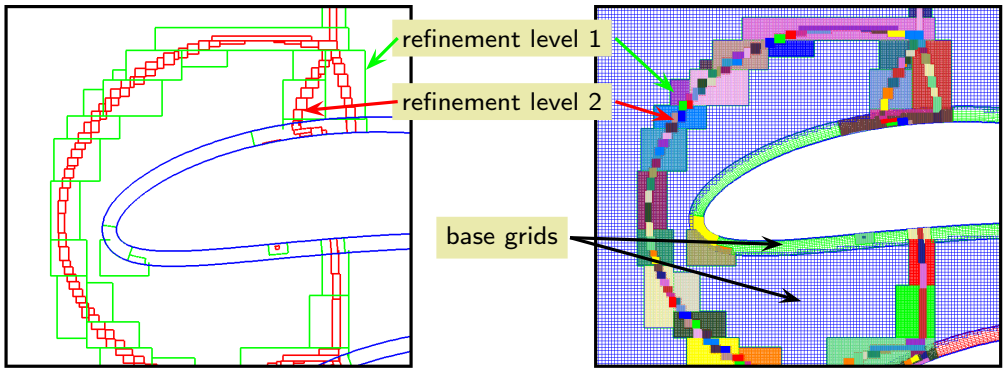


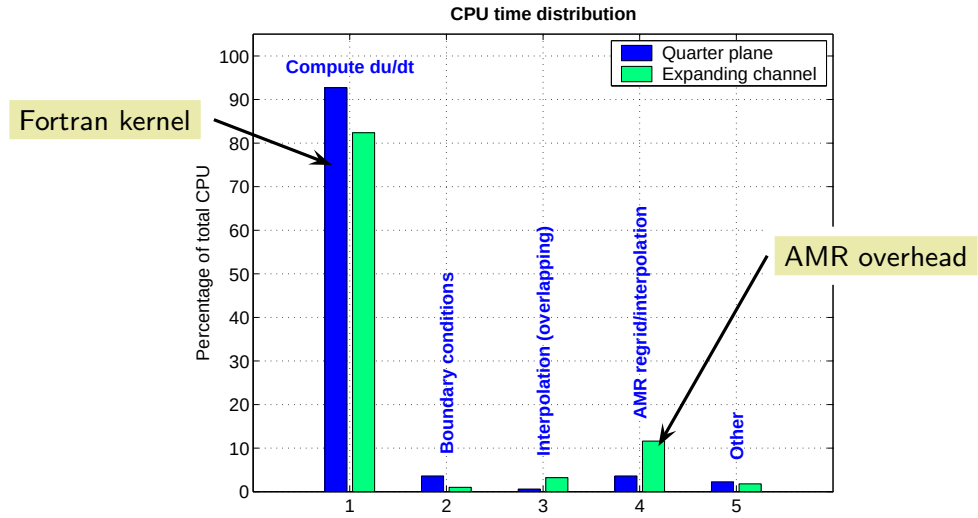
Performance of the Maxwell solver (serial).

Two-dimensions, 3.8 million grid-points.

Block Structured Adaptive Mesh Refinement and Overlapping Grids

- ♢ Refinement patches are generated in the parameter space of each component grid (base grid).
- ♢ Refinement patches are organized in a hierarchy of *refinement levels*.
- ♢ Error estimators determine where refinement is needed.
- ♢ AMR grid generation (Berger-Rigoutsos algorithm) builds refinement patches based on the error estimate.
- ♢ refinement grids may interpolate from refinement grids of different base grids.
- ♢ The key issue is efficiency.



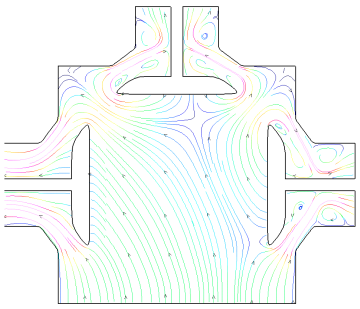


	Quarter plane	Expanding channel
time steps	12418	21030
seconds per step	14.94	13.96
grids (min,ave,max)	$(2, 57, 353)$	$(5, 274, 588)$
points (min,ave,max)	$(2.0e5, 9.2e5, 1.9e6)$	$(1.2e5, 6.4e5, 1.3e6)$

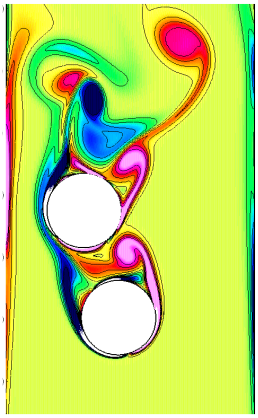
Overlapping grid AMR performance on two detonation problems.

Moving Overlapping Grids

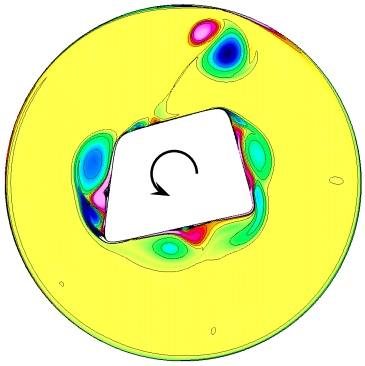
- ◇ Boundary fitted component grids are used to discretize each moving body.
- ◇ Grids move at each time step according to some governing equations.
- ◇ Overlapping connectivity information is updated by Ogen (interpolation points, discretization points, unused points).
- ◇ Solution values at **exposed points** are interpolated at previous time levels.
- ◇ Issue: Detection and treatment of collisions – elastic/in-elastic collisions
- ◇ Issue: Bodies that get very close – how should the grids interpolate



Moving valves (INS)

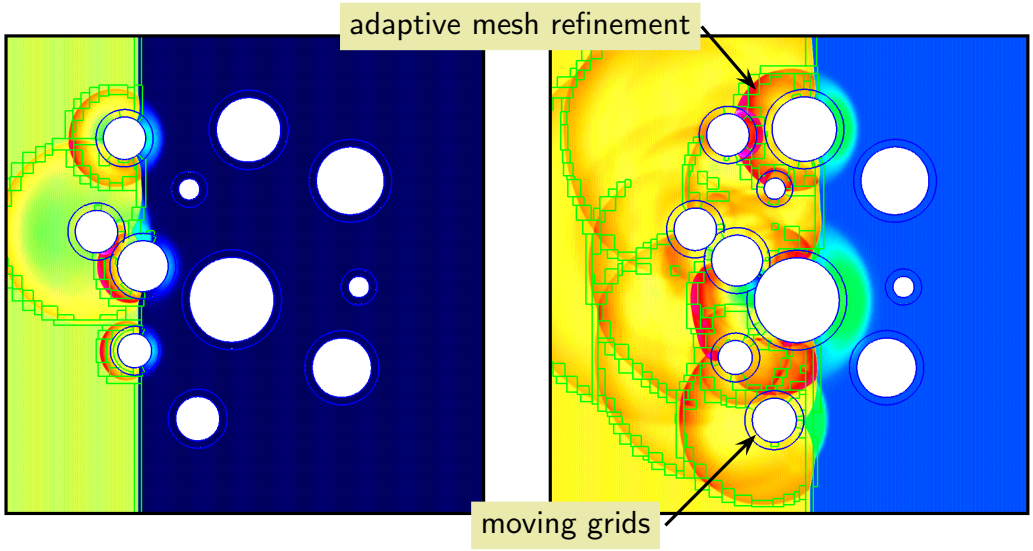


Falling cylinders (INS)

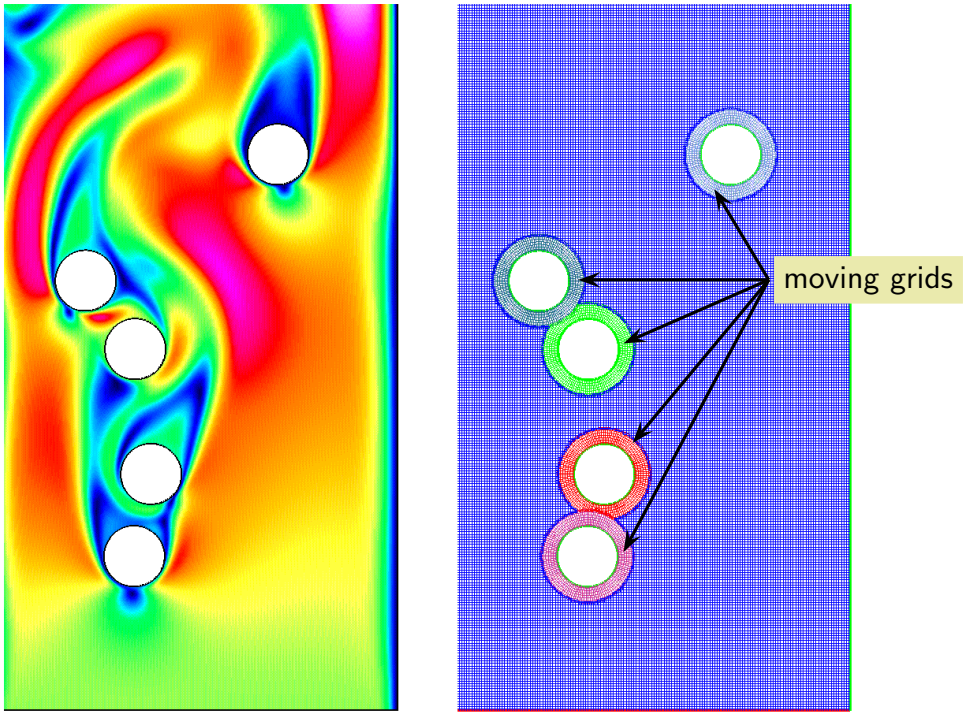


Rotating body (INS)

Moving geometry and AMR

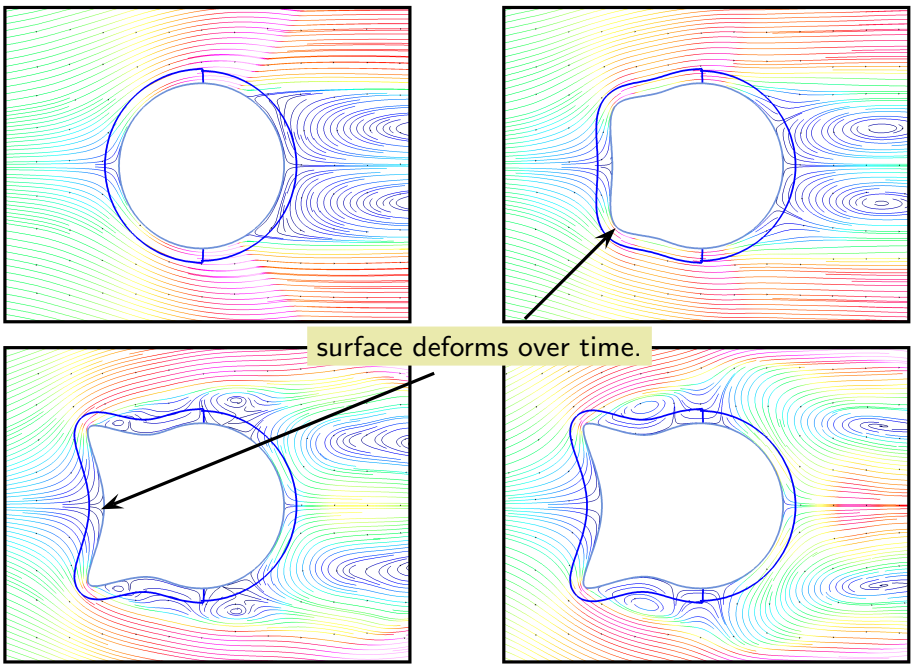


A shock hitting a collection of cylinders (density).



Falling cylinders in an incompressible flow

Modeling Deforming Geometry with Overlapping Grids

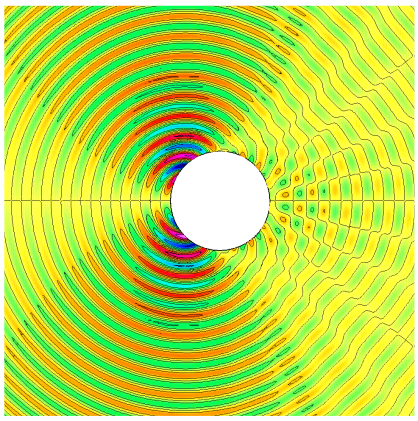


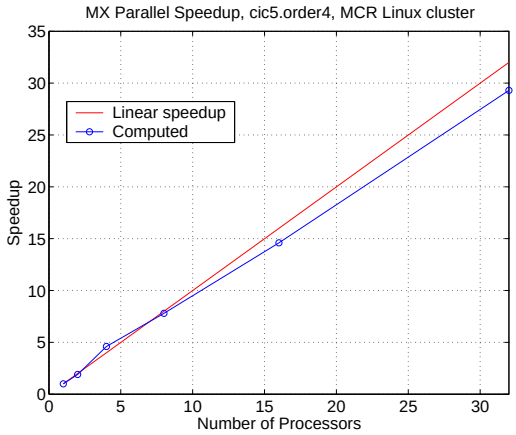
Streamlines of a compressible flow around a deforming boundary.

The model for distributed parallel computing in Overture

- ◇ Grids can be distributed across one or more processors.
- ◇ Distributed parallel arrays using P++ (K. Brislawn, B. Miller, D. Quinlan)
- ◇ P++ uses Multiblock PARTI (A. Sussman, G. Agrawal, J. Saltz) for block structured communication with MPI (ghost boundary updates, copies between different distributed arrays)
- ◇ A special parallel overlapping grid interpolation routine is used for overlapping grid interpolation.

Parallel scaling of the Maxwell solver – preliminary results



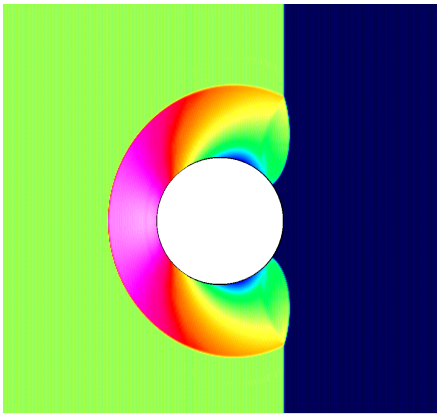


Fourth-order accurate

2D scattering from a cylinder

Fixed size problem, 3.8 million grid-points

Euler equations: preliminary parallel results



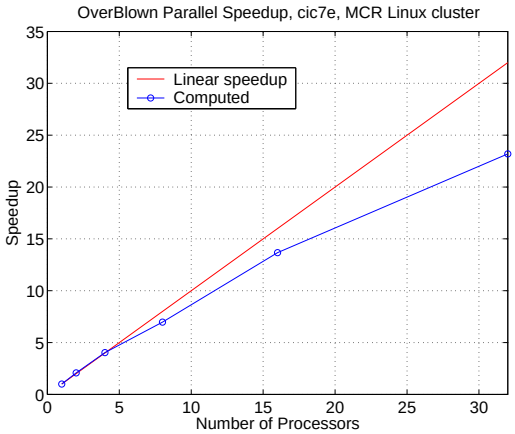
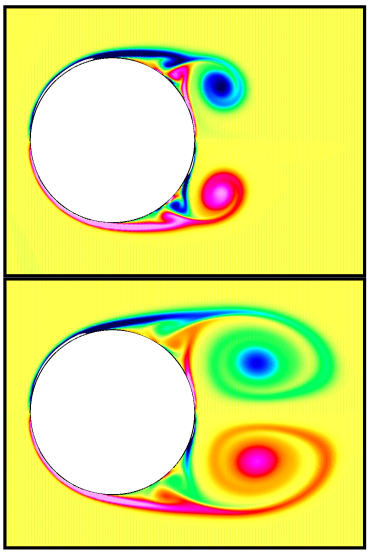


Figure 1: Left: the computation of a shock hitting a cylinder (density). Right: parallel speedup for this problem, keeping the problem size fixed (4 Million grid points), on a linux cluster (Xeon processors).

Incompressible Navier-Stokes: preliminary parallel results



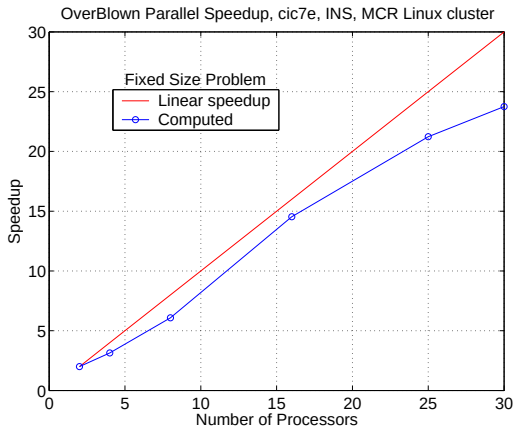


Figure 2: Left: impulsively started cylinder in an incompressible flow (vorticity). Right: parallel speedup keeping the problem size fixed (4 Million grid points), on a linux cluster (Xeon processors). The pressure equation is solved with algebraic multigrid (Hypre).