Porting FUN3D to Distributed Memory Parallelism

David E. Keyes and Dinesh K. Kaushik, ICASE and Old Dominion University;
Barry F. Smith, Argonne National Laboratory; and W. Kyle Anderson, NASA Langley Research Center

While much research in parallel computer science is oriented toward language, software, and architectural environments to support future code, porting valuable legacy codes to contemporary parallel environments remains an important objective. This is particularly true since parallelism is the only means by which massive amounts of memory can be cost-effectively brought to bear on legacy applications that need to outgrow the vector Crays for which they were created. NASA's mission to support computational design and optimization requires that computational fluid dynamics (CFD) and other types of analyses be routinely extended to higher fidelity models on finer grids. Moreover, since optimization depends on derivatives of the solutions to the analysis problem, it is advantageous to employ analysis techniques that work directly with the Jacobian. This focuses considerable interest on parallel implicit algorithms for elliptic PDE-based simulations of all kinds.

Implicit methods for elliptic problems are notoriously difficult to parallelize scalably since by their mathematical nature, they seem to require--and are often naively implemented to require--either long recurrences or frequent all-to-all communication. In fact, most legacy solvers for CFD or structural mechanics or Helmholtz codes suffer poor scalability due to starvation if ported with full respect for the data dependencies in the sequential algorithm. Nevertheless, there are considerable investments in the models represented in these codes. The high value-added contributions from the engineering community--the subroutines that express the residuals and Jacobians of the physical conservation laws--will take many years to replace with equally trusted code. There is ample concurrency in these subroutines, typically on the scale of the number of gridpoints. A natural challenge, then, is to harvest the discretization "smarts" of legacy codes in the context of a parallel nonlinear solver.

FUN3D is one such valuable legacy code, integral to present and anticipated aerodynamic design work in NASA and industry, and has been pulled into the parallel environment by means of the PETSc library from Argonne National Laboratory (ANL). (See http://www.mcs.anl.gov/petsc/.) FUN3D is a tetrahedral vertex-centered unstructured grid code developed at NASA-Langley for compressible and incompressible Euler and Navier-Stokes equations. (See the FUN home page, which includes many illustrations of the code's use, at http://fmad-www.larc.nasa.gov/~wanderso/Fun/fun.html.) The group's parallel experience to date is with the incompressible Euler subset of FUN3D, but nothing in the algorithm or discrete connectivity changes for the other cases. Since the researchers have started with the submodel of FUN3D with the fewest floating point operations per gridpoint per global flux balance operation, parallel performance should only improve as the fidelity and work per gridpoint rise.

The library-based approach to parallel computing offers a great deal for both legacy code parallel ports and new code development, since library developers can complement the expertise of the engineering community by embedding in the libraries expertise in parallel solvers. In contrast, parallel compilers and programming languages can at best give parallel expression to algorithms known to the applications programmer. The parallel nonlinear solver introduced into FUN3D is Newton-Krylov-Schwarz (NKS). In addition, a pseudo-timestepping heuristic is used to robustify the Newton method in the early iterations. This algorithmic combination is abbreviated PsiNKS. From comparisons of PsiNKS to full approximate scheme (FAS) multigrid on two-dimensional unstructured grid Euler problems, it is known that it is very close to multigrid's optimal complexity when both are sequentially implemented. The table below shows that PsiNKS is also encouragingly scalable.

Newton	**Krylov**	**Schwarz**
asymptotically quadratic nonlinear solver	spectrally adaptive accelerator	parallelizable preconditioner

The parallelization of legacy FUN3D was the vision of PETSc co-author Barry F. Smith (ANL), FUN3D author W. Kyle Anderson (NASA Langley Research Center), and David E. Keyes (ICASE and Old Dominion University), and was launched during a discussion at an adaptive unstructured grid workshop (funded, like this project, by DOE, NASA, and NSF) at Argonne in September 1996. The main implementation effort was carried out by Old Dominion University graduate student and ICASE intern Dinesh K. Kaushik over the next five months of part-time effort, which included studying FUN3D and its mesh preprocessor, learning the MeTiS partitioner (see http://www.cs.umn.edu/~karypis/metis/metis.html), and adding and testing some new unstructured functionality in PETSc.

In addition to parallelizing FUN3D, Kaushik restructured the code from vector to cache orientation. This attention to cache locality accounts for a reduction of a factor of nearly eight in wall clock time in serial execution mode alone. Approximately 3,300 of 14,400 F77 lines of FUN3D were retained. Some of the code that has been cut is related to viscous effects and turbulence modeling, and will be restored in future work. PETSc solvers replaced the rest. The parallel effort has not yet incorporated several key features of a practical production CFD code, including parallel I/O, parallel visualization, restarting and checkpointing, and grid sequencing (the most primitive form of multigrid, which is essential for nonlinear problems). Nevertheless, the group has learned many lessons that are being codified into PETSc's support for unstructured problems, which should allow the next legacy port to require significantly less effort.

Since the PETSc version of FUN3D is built upon the Message Passing Interface (MPI) (see http://www.mcs.anl.gov/mpi), it ports without change of source code to almost all of today's distributed memory and distributed-shared memory machines. Some early performance results for fixed-size and fixed-memory-per-node problems spanning three orders of magnitude in gridpoint density for an ONERA M6 wing are given in a forthcoming ICASE TR presently available at http://www.icase.edu/~keyes/papers/cfdr97.ps.

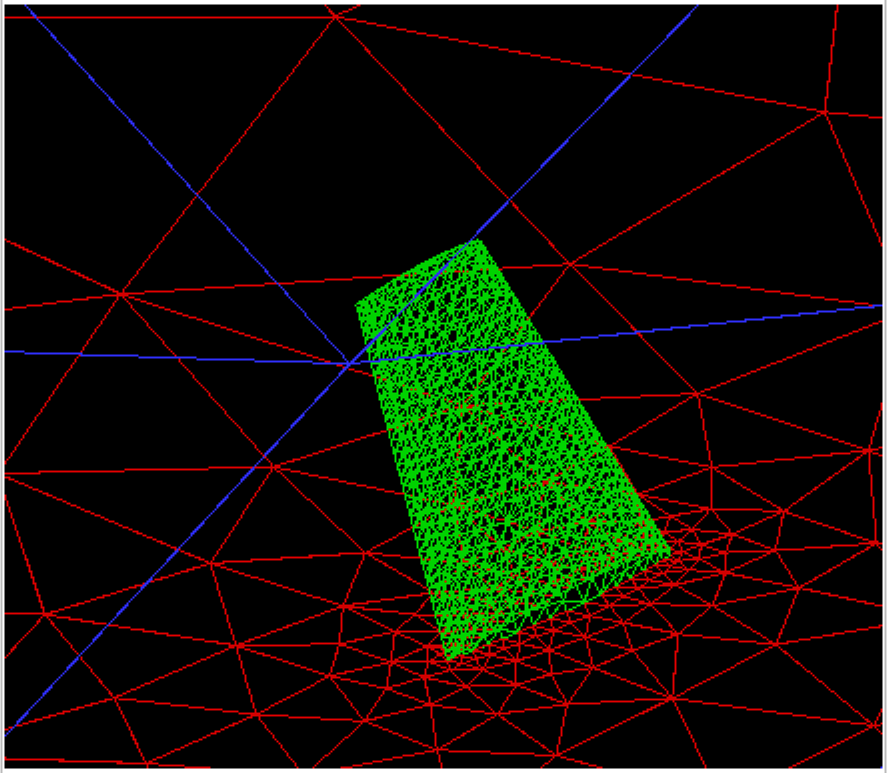
The table below, from this report, is for the largest case run to date, on a tetrahedral grid of 2,761,774 vertices. With four components per gridpoint, the (very sparse and nonsymmetric) Jacobian matrices of the resulting Newton problems have dimension 11,047,096. A surprising observation is that the restricted Additive Schwarz domain decomposition preconditioner loses very little algorithmic scalability when the problem is divided, for the sake of concurrency creation, into as many subdomains as the 512 available processors of the Cray T3E at NERSC.

This is part of the story behind the excellent fixed-size scalability of the method, as shown in the slow growth of iterations (from 164 to 171) over a four-fold range in processor number below. The other part of the story is the inspector/executor implementation model in PETSc's unstructured parallel vector gather-scatters and global reduction operations.

no. procs	no. its	exec. time	speed up	overall efficiency	sustained Mflop/s per proc.	sustained Glop/s overall
128	164	6,048s	1.00	--	66.7	8.5
256	166	3,242s	1.87	93%	64.8	16.6
512	171	1,881s	3.34	83%	62.6	32.1

Note that this scaling is over a range of only four (from 128 to 512 nodes) since the researchers were not permitted to possess queues of smaller numbers of processors for sufficient lengths of time to run the problem to completion, and it must run out of core for processor numbers much fewer than 128. The grid was generated just for this large-memory test by Dimitri Mavriplis of ICASE, and may be the largest unstructured grid upon which a Newton-based nonlinearly implicit CFD solver has been run.

The fall-off in efficiency to 83% at 512 nodes is due primarily to load imbalance between the unstructured gridblocks, leading to idleness at global reduction synchronizations, which accounts for 12% of execution time at 512 nodes. Though the number of owned vertices is very carefully balanced, the number of non-owned (ghosted) vertices varies from processor to processor, and is the principal reason for the imbalance of local work and local communication. The group is now in the process of addressing this through weighted partitioning based on finer work estimates, in hope of solving problems several times more finely resolved (e.g., to model viscous boundary layers on multiply slotted wings) on the Teraflops-scale machines coming into being under the auspices of the Accelerated Strategic Computing Initiative (ASCI) program today.



The surface triangulation of the wing (green), the symmetry root plane (red) and the farfield (blue) is shown for a relatively coarse grid above.

[Other Issues of *PCR*] [**Back to PCR**] [CRPC Home Page]