A COUPLED NEWTON-KRYLOV SOLVER FOR IMPROVED CHAD CACHE UTILIZATION AND PERFORMANCE *

Thomas R. Canfield, Tai-Hsin Chien, Henry M. Domanus,
Adrian M. Tentner, Constantine P. Tzanos, Richard A. Valentin,
and David P. Weber
Argonne National Laboratory, 9700 S. Cass Ave., Argonne, IL 60439-4844

Abstract

CHAD (Computational Hydrodynamics for Advanced Design) is a computer program that has been developed to analyze flows in automotive and defense applications. Extensive performance analysis of the CHAD computer program indicated a need to address cache memory use to increase computational performance. Several strategies have been adopted to achieve this goal: simultaneous solution of the coupled Navier-Stokes equations, data clustering, and data ordering. A coupled Newton-Krylov solver has been incorporated into a version of the CHAD program, resulting in consistent improvement in run times that varies from 50% to 200%. Further work will be required to tune the solver for optimal performance. In addition, experiments with data cluster and reordering indicate a potential for performance improvement.

Introduction

CHAD (Computational Hydrodynamics for Advanced Design) was originally developed under the Super Computing Automotive Applications Partnership (SCAAP) with the United States Councel for Automotive Research (USCAR) and five U.S. Department of Energy laboratories: Argonne, Lawrence Livermore, Los Alamos, Oak Ridge, and Sandia National Laboratories. It computes three-dimensional fluid flows with chemical reactions and fuel sprays. It is the successor to the KIVA code, which has become a standard research computer program for device-level modeling of internal combustion engines. CHAD is intended for use in modeling automotive design applications, including combustion, interior airflow (HVAC), under-hood cooling and exterior flows.

The CHAD computer program offers improved technology for device-level simulation of combustion processes in terms of flexible modeling and increased accuracy. The current version of the program will model internal combustion engines (ICEs) using spray models and species transport. However, higher-fidelity ICE simulation with CHAD will require two types of algorithmic improvements. Improvements in the numerical methods will lead to increased computational performance in existing models and higher resolution in terms of the number of computational cells. Improvements in the physical modeling will enable higher accuracy to be realized.

Control Volume Formulation

The integral conservation equations for mass, momentum, and energy have the following generic form:

$$\frac{d}{dt}\int_V \rho Q \, dV + \int_A \rho Q(\mathbf{u}-\mathbf{v})\cdot\mathbf{n}\, dA = \int_A \rho D_Q \nabla Q \cdot \mathbf{n}\, dA + \int_V S_Q \, dV,$$

where Q is the transport variable, ρ is fluid the density, $\mathbf{u}$ is the fluid density, $\mathbf{v}$ is the mesh veloc-

*Work supported by U.S. Department of Energy under Contract W-31-109-Eng-38.

ity, D_Q is the diffusivity, and S_Q is the source. In CHAD, the discretization of this equation requires that the variables be colocated at the vertices. This is accomplished through a combination of streamline upwinding and explicit node coupling to control differencing of transport terms. Details of this approach are given in [4].

The least complex option in CHAD is the solution of the nonlinear Navier-Stokes equations. In this case there are five sets of equations corresponding to $Q : (1, h, u, v, w)$. The resultant finite difference equations are rearranged into five sets of nonlinear residual equations involving the $5N$ independent vertex variables:

$$R_{i\nu}(q_{j\mu}^{n+1}, q_{j\mu}^{n}, t, \Delta t) = 0,$$

where the latin indices i and j denote the N independent field variables $p, T, u, v,$ and w.

For known $q_{i\nu}^{n}$, given time t and time step Δt, the solution of

$$R_{i\nu}(q_{j,\mu}^{n+1}) = 0$$

is sought.

Segmented Solver

CHAD uses a segregated strategy for the iterative solution of the residual equations (see Figure 1). In this approach the residual equations associated with mass, momentum, and enthalpy variables are solved sequentially while holding nonassociated variables fixed. The associated variables are updated between each solve, and the process is repeated until the overall system has converged. This solution strategy has been used in many CFD codes primarily because it requires less CPU memory. The method does not follow the steepest descent of the Jacobian for coupled system, but can be accelerated in some cases with judicious updates of the non-associated variables between the segregated solves.

A version of CHAD has been instrumented with PETSc [2] logging functions. Timings for a typical problem are given in Table 1. Timing studies indicate that CHAD spends most of time solving the residual equations. The measurements indicate that the largest amount of time is spent

Iterate:
(1) *Solve momentum equations* $\rightarrow$
$\Delta u, \Delta v, \Delta w$ *and* $\Delta \rho$
(2) *update*
u, v, w *and* ρ
(3) *Solve enthalpy equation* $\rightarrow$
$\Delta h, \Delta T$ *and* $\Delta \rho$
(4) *update*
h, T *and* ρ
(5) *Solve pressure equation* $\rightarrow$
$\Delta p, \Delta T, \Delta u, \Delta v, \Delta w$ *and* $\Delta \rho$
(6) *update*
p, T, u, v, w *and* ρ
(7) *Break if converged*

Figure 1: Outline of CHAD's Segregated Solver.

solving the pressure equation. For compressible flows, this takes about 30% of the total time. For incompressible flows, CHAD is much less efficient. CHAD approximates incompressible fluids by introducing an artificially large bulk modulus. With the same geometry and boundary condition, CHAD spends 65% of the time solving the pressure equation.

It was conjectured that more accurate solution of the pressure equation might improve the overall performance of CHAD and possibly accelerate the convergence of the segregated solver. Experiments were performed to test this conjecture. In the test code the original SOR solver for the pressure equations was replaced with GMRES from PETSc. The modified code was able to solve the pressure equation much more accurately, but the iteration counts for the segregated solver did not decrease. For this reason no further refinements were made to the modified pressure solver.[1]

FUN3D

Recent experience with the NASA code, FUN3D [1], suggests that substantial increase in performance can be achieved by using an alternative strategy of solving the coupled nonlinear Navier-

[1]We note that the most current version of CHAD use GMRES in place of SOR for the segregated solves.

Phase	Compressible Time (sec)	Incompressible Time (sec)
Total Time	232.06	571.88
Hydro-Loop	129.76	469.03
Pressure Solve	66.44	417.82
Pressure Residual	40.55	359.02
Momentum Solve	36.22	26.35
Energy Solve	11.47	11.00
Momentum Residual	15.70	8.36
Energy Residual	2.71	2.96

Table 1: Performance of CHAD's segregated solver on a problem with 10648 vertex control volumes running on 8 processors of an SGI Origin 2000 with 250 MHz R10000 IP27 processors with 256 MB of local memory. *Note:* The times are inclusive.

Stokes equations [3]. This approach, when combined with data restructuring and reordering, can lead to a fivefold increase in overall performance.

CHAD is similar to FUN3D in many respects. Both codes use unstructured grids. They both use a node centric finite volume formulation of the Navier-Stokes equations. Evaluation of integrals over the vertex cells uses an edgewise construction in both codes. Some of the underlying data structures are similar. For these reasons CHAD might benefit by adopting a similar solution strategy.

There are fundamental differences between the two codes. FUN3D is written in FORTRAN 77 whereas CHAD is written in FORTRAN 90. CHAD uses some of the advanced features of FORTRAN 90. The parallel implementation and the data distribution are different. For example, FUN3D shares vertex data on processor boundaries in so-called *ghost cells*. Whereas the vertex data in CHAD is assigned to a unique processor and is not shared. The data is shared in a gather operation where values from the distributed vertex arrays are placed into edge terminus arrays. Figure 2 illustrates how these approaches differ in the calculation of the average of vertex quantities associated with the edge terminuses onto the median mesh points associated with the cell faces.

The FUN3D approach to calculation of median quantities uses less memory and allows for reuse of data fetched into local cache memory. The CHAD approach involves a vector operation

```
for all edges, i
      m(i)=(v(edge(1,i))+v(edge(2,i)))/2.0

for all edges, i
      m(i)=(vt(1,i)+vt(2,i))/2.0
```

Figure 2: FUN3D (*top*) and CHAD (*bottom*) approach to calculating and average value on the median mesh.

that uses more memory and inhibits reuse. The difference in the approach is subtle. How this may affect overall performance is difficult to measure. In rather simplistic loop experiments we have found the FUN3D approach to be slightly more efficient. The results depend on many factors, however, including the order of the accessing the data, the size of the data, and the CPU architecture. To appreciate how cache utilization can effect performance refer to Figure 3. As long as everything involved in a calculation is available in cache (*cache hit*), a high level of performance is achieved. As soon as memory outside of cache has to be accessed (*cache miss*), however, performance degrades.

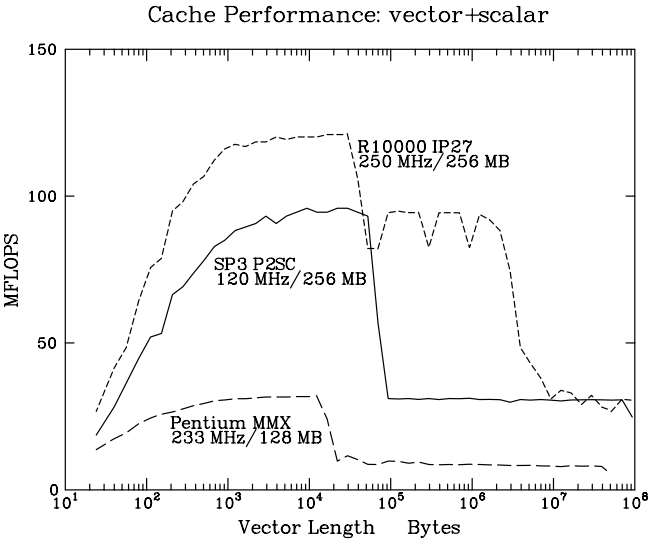

Figure 3: Cache performance on the SGI Origin 2000, IBM SP3, and NEC laptop computer using a simple *vector+scalar* loop.

Coupled Newton Solver

The new coupled equation solver that has been implemented in CHAD solves the combined residual equations for mass, momentum, and enthalpy. These residual equations are solved using the nonlinear equation solver available in PETSc. The combined routines eliminate redundant calculations and reduce the communications time by a factor of two in the solution phase of a CHAD calculation.

During each solve an inexact Newton iteration is used in the solution of nonlinear residuals:

$$R(q) = 0,$$

where q is updated by the approximation

$$q^{k+1} = q^k - [R'(q^k)]^{-1} R(q^k), k = 0, 1, \cdots.$$

Starting with an initial guess, q^0, and assuming $R'(q^k)$ is nonsingular, the Newton iteration is implemented by the following two steps:

$$\begin{array}{lrcl} \text{Solve} & R'(q^k)\Delta q^k & = & -R(q^k) \\ \text{Update} & q^{k+1} & = & q^k + \Delta q^k \end{array}$$

The method is inexact because $R'(q)$ is usually approximated rather than computed exactly.

Table 2: Comparison of segregated solver and coupled solver times for CHAD with various values of SNES convergence parameter. Problem is a simple duct with 16000 cells run for 20 time steps on 16 processors of the SGI Origin 2000.

	CPU Time (sec)	SNES Tolerance
Segregated Solver	285.84	–
Coupled Solver	192.35	10^{-3}
	185.93	10^{-4}
	193.37	10^{-5}
	193.97	10^{-6}
	183.31	10^{-7}
	212.39	10^{-8}

The early results are quite encouraging. In initial studies we have scaled the residuals, a strategy that is roughly equivalent to approximating the Jacobian with a diagonal matrix. Table 2 gives the timing studies for one of our simple test cases.

Summary

We have successfully implemented a coupled Newton-Krylov solver in CHAD using PETSc. We have demonstrated speedup with the new method. Further studies are anticipated with various types of preconditioners and approximate Jacobians.

References

[1] Anderson, W. K., and D. L. Bonhaus,"An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," Computers in Fluids 23, no. 1, 1–21, 1994.

[2] Baly, S., W. D. Gropp, L. C. McInnes, and B. F. Smith, The Portable Extensible Toolkit for Scientific Computing, version 2.0.26, http://www.mcs.anl.gov/petsc, 1999.

[3] Keyes, D. E., D. K. Kaushik, and B. F. Smith,"Perspectives for CFD on Petaflops Systems," CFD Review, ed. M. Hafez, World Scientific, Singapore, pp. 1079–1096, 1998.

[4] O'Rourke, P. J., and M. J. Sahota,"A Variable Explicit/Implicit Numerical Method for Calculating Advection on Unstructured Meshes," J. Computational Physics 143, 312–345, 1998.