A 3-D Compressible Navier-Stokes Code using Parallel Fortran

Lyle N. Long[1]
The Pennsylvania State University
University Park, PA 16802

Introduction. This note describes a new computer program written in CM-Fortran (Fortran-90) to solve the 3-D Navier-Stokes equations on the massively parallel Connection Machine (CM-2). Performance and programming comparisons will be made to similar codes written in the past in *Lisp for the CM-2 and in Fortran-77 for vector computers.

Navier-Stokes. The form of the compressible Navier-Stokes equations being solved are:

$$\frac{\partial}{\partial t}\iiint_V \begin{bmatrix}\rho\\ \rho\mathbf{u}\\ \rho E\end{bmatrix} dV = -\iint_S \begin{bmatrix}\rho u_n\\ \rho \mathbf{u} u_n + p\mathbf{n} - \mathbf{T}\\ \rho H u_n - \mathbf{T}\cdot\mathbf{u} + \mathbf{q}\cdot\mathbf{n}\end{bmatrix} dS$$

where ρ, $\mathbf{u}$, E, T, and $\mathbf{q}$ are the density, velocity, total energy, traction, and heat transfer; respectively. The traction is defined as

$$\mathbf{T} = \mu n_j\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + n_i(\mu_b - \frac{2}{3}\mu)\frac{\partial u_i}{\partial x_i}\delta_{ij}$$

and n_i is the unit normal to the surface, p is the pressure, μ is the dynamic viscosity, and μ_b is the bulk viscosity. These equations are applied to each cell in the computational domain, where V and S refer to the volume and surface area of each cell.

The above equations are integrated in time using a multi-stage Runge-Kutta scheme. This is the same algorithm that is used in the Lockheed/USAF Three-Dimensional Euler/Navier-Stokes Aerodynamic Method (TEAM) [1] and in the NSCM code [2,3] written in *Lisp. The current program uses structured grids, while NSCM can use either structured or unstructured grids. The present code is written in a very modular fashion, making it easy to change boundary conditions and grid topology.

The adaptive dissipation scheme of Jameson is used, where second order dissipation is turned on in regions of large pressure gradient. Fourth order dissipation is used elsewhere.

Boundary Conditions. The present code, while written for general curvilinear grids, has been applied here to C-H type grids. These grids require four different types of boundary conditions: Solid surface, Far-field, Symmetry, and Wake region. "Ghost" cells are added to every edge of the grid. This allows a minimum of operations to be performed while the other processors are idle and then the cell face between the ghost cell and the boundary cell can be treated just as the interior part of the grid. The wake region requires special treatment because the grid wraps back upon itself. In Fortran-90 this amounts to performing the following operation:

```
G(:, 1, :) = G(NX:1:-1, 2, :)
```

which is an array reversal in one dimension. Using a grey-code mapping in PARIS was no more efficient than simply using the Fortran statement. The far-field boundary conditions used are essentially the same as those used in the TEAM [1] and NSCM codes [2,3]. For subsonic flow, a locally one-dimensional Riemann invariant problem is solved to achieve the correct propagation of the disturbances. These complicated far-field boundary conditions are very expensive on SIMD computers because they require numerous, lengthy calculations (such as x**y) and while these are being performed the interior grid points (or processors) are idle. In the present implementation, the far-field boundary conditions accounted for up to 20% of the CPU time.

Residual Smoothing. Residual smoothing is also used in the code to speed convergence to a steady-state. On an NxNxN grid, this requires solving $3N^2$ tridiagonal systems, which requires roughly $3N^2(16N-1)$ floating point operations per stage. The current code uses the library routine from Thinking Machines Corp. (CMSSL) to solve the tridiagonal systems. This routine uses parallel cyclic reduction and achieves about 200 Mflops on a 64k CM-2, but has a tolerance option that allows one to get an approximate solution to a tridiagonal system relatively quickly.

Results. Viscous (laminar) flow over a rectangular wing with an NACA 0012 airfoil section was simulated at subsonic, transonic, and supersonic

[1] Assistant Professor, Member AIAA