Abstract
Massively parallel computers have enabled the analyst to solve complicated flow fields (turbulent, chemically reacting) that were previously intractable. Calculations are presented using a massively parallel CFD code called SACCARA (Sandia Advanced Code for Compressible Aerothermodynamics Research and Analysis) currently under development at Sandia National Laboratories as part of the Department of Energy (DOE) Accelerated Strategic Computing Initiative (ASCI). Computations were made on a generic reentry vehicle in a hypersonic flowfield utilizing three different distributed parallel computers to assess the parallel efficiency of the code with increasing numbers of processors. The parallel efficiencies for the SACCARA code will be presented for cases using 1, 10, 50, 100 and 500 processors. Computations were also made on a subsonic/transonic vehicle using both 236 and 521 processors on a grid containing approximately 14.7 million grid points. Ongoing and future plans to implement a parallel overset grid capability and couple SACCARA with other mechanics codes in a massively parallel environment are discussed.

Introduction
The massively distributed parallel CFD code currently under development at Sandia National Laboratories as part of the Department of Energy (DOE) Accelerated Strategic Computing Initiative (ASCI) Program is called SACCARA (Sandia Advanced Code for Compressible Aerothermodynamics Research and Analysis). SACCARA is currently being developed from PINCA [1,2], a distributed parallel version of the commercial, finite volume, Navier-Stokes code, INCA [3], from Amtec, Inc. SACCARA solves the 2-D/Axisymmetric and 3-D Full Navier-Stokes equations for laminar and turbulent flows in thermo-chemical nonequilibrium. SACCARA is applicable from subsonic through hypersonic flows and can allow for perfect gas, equilibrium, and finite-rate chemistry. Standard zero-, one-, and two-equation turbulence models are also available. The code employs multiblock structured grids with point-to-point matchup at the block interfaces. The solution is driven to a steady state using the lower-upper symmetric Gauss Seidel (LU-SGS) or diagonal implicit solution advancement scheme based on a
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
DISCLAIMER

Portions of this document may be illegible electronic image products. Images are produced from the best available original document.
combination of the work of Yoon et al. [4,5] and Peery and Imlay [6]. The inviscid fluxes are evaluated using the flux vector splitting of Steger and Warming [7] or the symmetric TVD flux function of Yee [8]. The viscous terms employ a standard central differencing scheme.

Parallel Implementation of SACCARA

SACCARA can be run on a variety of shared memory or distributed memory parallel platforms as well as in a serial mode. The code can be compiled using the standard message-passing interface (MPI) [9] as well as Intel's native NX [10] parallel calls. Currently SACCARA can only allow for a single grid block per processor. Communication between the grids is handled through a layer of ghost cells at each block interface where pertinent information is updated explicitly after each global iteration step. All the input information, including the block interface information, is contained in a single input data file using a namelist format. The multiblock, structured grid is input in standard PLOT3D format [11] or in INCA format [3]. Both the input data file (IDATA) and input mesh file (IMESH) are the same for either serial or parallel operation.

The grids used in SACCARA are typically made using GRIDGEN [12] or any other multiblock structured grid generator. GRIDGEN has the ability to generate not only the IMESH file in PLOT3D format but also much of the IDATA file, including the block interface information. The blocking structure of the grid is generally created based on the topology of the geometry of interest, without regard to the number of processors that will be used to solve the problem. The DECOMP [1] code is used to subdivide the grid into the number of blocks that equals the total number of processors desired for parallel operation. DECOMP, which uses many of the SACCARA subroutines, was recently modified to include part of the BREAKUP [13] code. The inputs to DECOMP are the IDATA and IMESH files and the total number of desired processors. Given the original blocking topology, DECOMP will determine the optimal load-balanced system. DECOMP first determines the total number of grid points and divides it by the number of total processors to obtain an average number of grid points per processor. DECOMP then begins subdividing those blocks whose total number of grid points is greater than the average along the I, J, and K coordinate directions. Blocks with fewer than the average number of grid points are not subdivided. DECOMP then chooses the divisions in the coordinate direction which best minimizes the surface area to volume ratio of each new grid block to ultimately reduce communication time during the solution. In addition, DECOMP may slightly reduce or increase the total number of processors if it determines it can create a more load-balanced decomposition with a different number of total processors. Finally, DECOMP creates the new IDATA and IMESH files to be used as input to SACCARA for parallel execution.

Results

SACCARA computations have been made on a hypersonic flowfield surrounding a generic reentry vehicle utilizing three different computer platforms. The calculations were made on three machines: the ASCI Red (Intel) at Sandia National Laboratories, the ASCI Blue ID (IBM SP2), and ASCI Blue TR at Lawrence Livermore National Laboratories, using 1, 10, 50, 100, and 500 processors to assess the parallel efficiency of the code with increasing numbers of processors. The size of the computational grid was varied to maintain 13,000 grid points on each processor.
The ASCI Red machine has 4500+ nodes. Each node has two 200-MHZ Pentium Pro processors with 128 Megabytes of memory/node. The ASCI Blue ID machine had 128 nodes. Each node has one IBM 66-MHZ Power2 processor with 256 Megabytes of memory. The ASCI Blue TR machine, which recently replaced the ASCI Blue ID machine, has 158 nodes. Each node has four IBM 332-MHZ 604e processors with 512 Megabytes of memory/node.

The parallel efficiency is defined as the ratio of the main loop serial CPU time to the main loop parallel CPU time with the number of grid points per processor remaining fixed. The efficiency of the code (Figure 1) is seen to drop off as expected due to increasing communication overhead as more processors are used. In Figure 1 a direct comparison of parallel efficiency can be made between the ASCI Blue ID and the ASCI Blue TR machine. There are a number of factors that increase the parallel overhead of the ASCI Blue TR machine. A major factor is the ASCI Blue TR machine requirement that distributed parallel codes use the slow IP method to communicate between nodes. The IP communication method is approximately three times slower than IBM's high-speed switch in dedicated mode. There is a planned update to the ASCI TR Machine to modify the current method for running a distributive parallel code on the TR machines to allow the use of the high-speed switch. This modification should provide a substantial increase in the parallel efficiency.

Computations were also made on the subsonic/transonic vehicle shown in Fig. 2 on 521 processors (ASCI Red) and 236 processors (ASCI Blue). The entire grid contained approximately 14.7 million grid points. The computational grid size and point distribution was based on previous axisymmetric solutions of a nonfinned vehicle at 0° angle of attack. The axisymmetric solution was run on a sequence of grids to determine the solutions sensitivity to the grid resolution. The decomposed block structure on the surface and pressure contours are also shown in Fig. 2. The simulation of the subsonic/transonic vehicle in Figure 2 was performed at a Mach number of 0.8 and an angle of attack of 5°. The flowfield around the vehicle was assumed to be fully turbulent, and the one-equation, Baldwin Barth model was used to model the turbulent flowfield. The simulation was run on 236 processors and took 12.7 days of CPU time to reach a converged solution on the ASCI Blue TR machine. The simulation required 147,000 iterations to reach convergence and ran at approximately 23 Gflops. The solution was considered converged when the global L2 norm of the momentum residuals had dropped by five orders of magnitude and the surface quantities no longer changed. The SACCARA code utilizes a point implicit method in its solution algorithm. This method is very efficient and highly stable but is inherently slow to converge. The resulting 147,000 iterations to reach convergence for this type of problem is not unexpected. The aerodynamic coefficients resulting from the simulation are shown in Table 1. The aerodynamic coefficients are presented in the form of percent difference compared to wind tunnel data and parameterized coefficients based on flight test data. The wind tunnel data is from a test performed in Sandia's Trisonic Wind Tunnel (TWT). The parameterized coefficients were determined by fitting the aerodynamic coefficients until the accelerometer and rate gyro data obtained from trajectory simulations matched those obtained in flight testing.

The parameterized coefficients can be viewed as a set of aerodynamic coefficients that, when used in a trajectory simulation code, accurately reproduce the vehicle's dynamic motion in flight. The accuracy of the single set of parameterized coefficients is not known. However it is
instructive to see how these coefficients compare with wind tunnel measurements and the coefficients computed with SACCARA. Table 1 shows good agreement between the pitching moment ($C_m$) and normal force coefficient ($C_N$) measured in the wind tunnel and the coefficients predicted by the CFD simulation. There is a larger discrepancy between the parameterized normal force coefficient and the normal force coefficient predicted in the CFD simulation. The almost zero percent difference with the center of pressure ($X_{cp}/L$) indicates that ratio of pitching moment-to-normal force is consistent between the different data sets. The rolling moment ($C_l$) coefficient comparison in Table 1 shows good agreement with the parameterized flight test data but a larger percent difference with the wind tunnel data. The largest disagreement is with the total axial force coefficient ($C_A$). It is believed that most of the difference in the prediction of axial force coefficient is due to error in predicting the base pressure properly. The Balwin-Barth turbulent model is known not to accurately model the free shear layer in the base region. A 1 percent change in the average pressure acting on the base of the vehicle can result in a 20 percent change in the axial force coefficient. Additional work is planned to examine these results in detail and run additional simulations using the Spalart-Allmaras one-equation model and the k-ω two-equation turbulence. Comparing these solutions should provide insight into the base pressure predictions and sensitivity to different turbulence models.

### Table 1: Force and Moments

<table>
<thead>
<tr>
<th>Percent Difference from Computation</th>
<th>$C_A$</th>
<th>$C_N$</th>
<th>$C_m$</th>
<th>$C_l$</th>
<th>$X_{cp}/L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flight Test</td>
<td>56.0%</td>
<td>-9.54</td>
<td>5.5%</td>
<td>-7.67</td>
<td>.75%</td>
</tr>
<tr>
<td>Wind Tunnel</td>
<td>59.2%</td>
<td>2.4%</td>
<td>5.5%</td>
<td>-13.1%</td>
<td>.01%</td>
</tr>
</tbody>
</table>

### Ongoing and Future Work

SACCARA is continually being modified to handle new problems of interest. A multilevel overset, or Chimera, grid capability is being added to SACCARA to be used in a parallel environment. This capability will reduce the dependence on having point-to-point match up at the grid block interfaces, which can complicate the grid generation process on complex vehicle geometries. Overset grids will allow gridding of individual geometric or flowfield features without regard for the overall grid topology. This will simplify the multiblock structured grid generation as well as reduce the total number of grid points necessary to solve a particular problem. In addition, overset grids will allow the solution of moving or multibody problems, such as store separation, without having to regrid as one body is moved relative to the other.

Work is also being performed to couple SACCARA with other mechanics codes in a distributed parallel environment. This work has included coupling SACCARA’s compressible fluid mechanics capability to a parallel material thermal response code, COYOTE [14,15], for hypersonic reentry vehicle ablation prediction [16]. Ablation of an axisymmetric sphere-cone nose tip along a given trajectory using the coupled technique is shown in Figure 3. Current coupling between the codes has been achieved by exchanging surface interface information via file transfer. Ongoing work will allow these codes to be coupled using parallel calls. Finally,
plans are underway to couple both SACCARA and COYOTE to a six degree-or-freedom flight dynamics code for a full trajectory simulation of an ablating hypersonic vehicle.

References

Figure 1. Parallel efficiency of SACCARA on DOE ASCI machines.

Figure 2. Decomposed block structure and pressure contours on subsonic/transonic vehicle.
Figure 3. Ablated surface shapes at selected trajectory points for a sphere-cone re-entry vehicle (every other trajectory point shown).