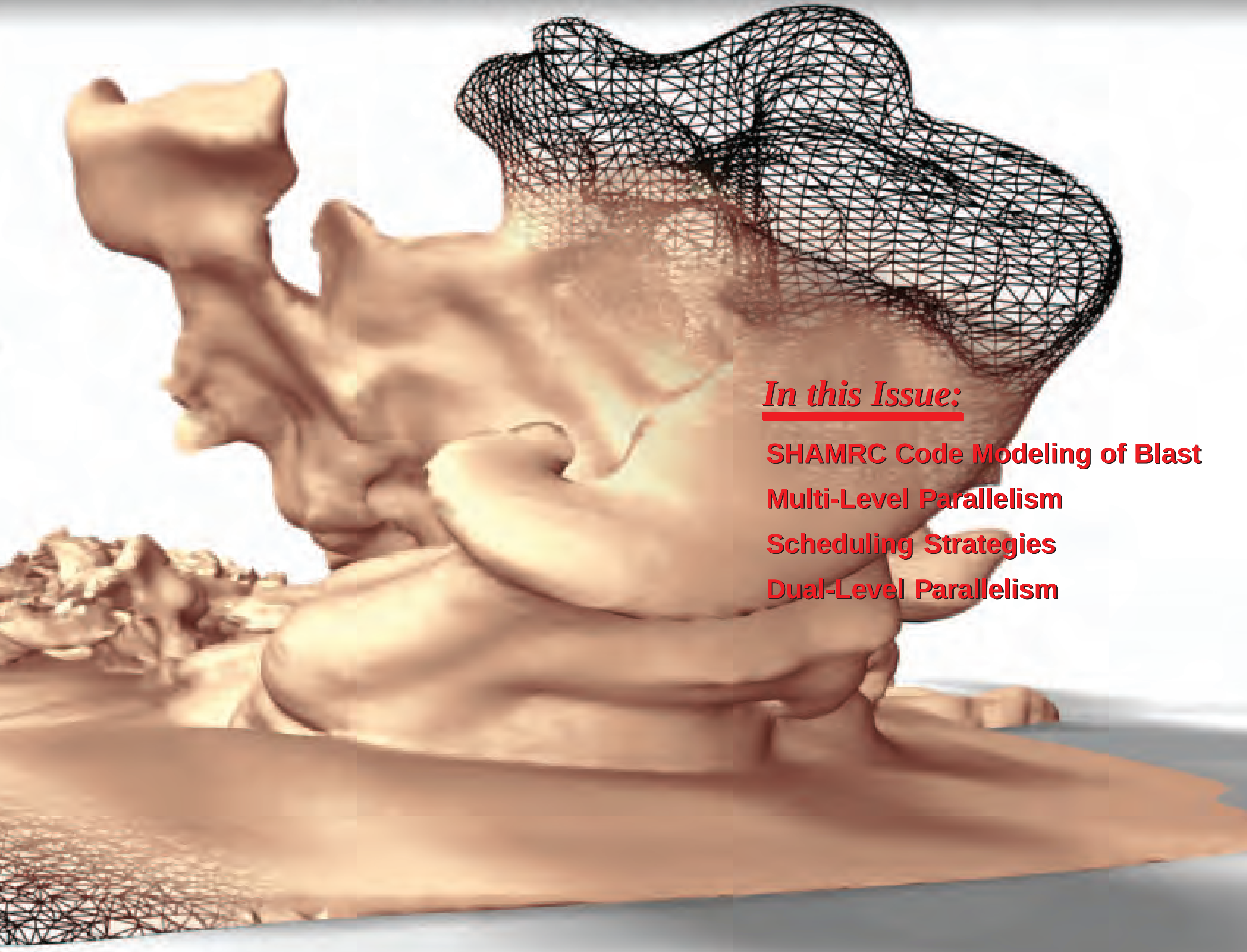
NEWSLETTER

FALL 2002

The Resource

U.S. ARMY ENGINEER RESEARCH AND DEVELOPMENT CENTER
INFORMATION TECHNOLOGY LABORATORY



In this Issue:

ERDC MSRC
Major Shared Resource Center

HPC Modernization Program

From the Director's chair

When I was first asked to talk about the theme of this year's supercomputing conference – Terabytes to Insights – what initially came to mind were the various supercomputers we have on the computer floor at the U.S. Army Engineer Research and Development Center (ERDC). What a great opportunity – I thought – to emphasize the fact that ERDC has the largest Cray T3E in the world, and the strength of the rest of our world-class high-performance computing (HPC) infrastructure! Our users want to get their jobs completed on time and the faster the better, so the speed and number of processors are a significant factor for them.

But the more I thought about it, the more I realized that my focus should really be on the scientific insight users obtain from their simulations.

High-performance computing is really about transformation of data – terabytes of data – into knowledge. In the Department of Defense (DoD), we are concerned with knowledge about global weather patterns; knowledge about future combat systems; knowledge about mobility and survivability; knowledge about troop deployment and tactical operations; and knowledge about the hundreds of other mission areas to which the DoD applies HPC each year.

Here at the ERDC Major Shared Research Center (MSRC), we provide the computers, networks, data storage, and expertise required to enable DoD scientists and engineers to solve problems that cannot be solved in any other way, or to solve them faster or more accurately than ever before. Doing this requires specialists trained in mathematics, physics, statistics, domain sciences, computer science, and data analysis as well as system architecture, design, and administration. These craftspeople turn our HPC infrastructure into the engine for transforming data into knowledge, and providing answers to tomorrow's questions.

Hundreds of researchers from all over our country are using ERDC's Compaq, SGI, and Cray systems to help shape the present and future for the DoD. As their simulations become more sophisticated, and as we provide them with an increasingly powerful infrastructure, the rate at which our users generate data will continue to grow. This is the age where data – not computation – drive how we provide service to our scientists and engineers.

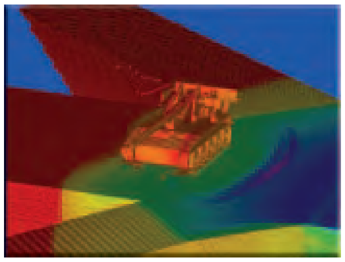
Providing the infrastructure and expertise to deal effectively with these data will differentiate those who only provide high-performance computers from those who enable scientific insight and help researchers transform their terabytes of data into knowledge about our world. This is the heritage of the ERDC MSRC, and this will be our mission for the future.

John West
Director, ERDC MSRC

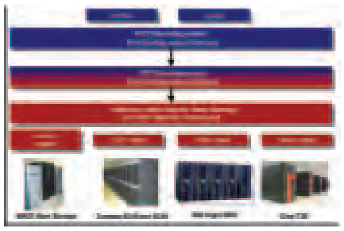
About the Cover:
Second-Order Hydrodynamic Automatic Mesh Refinement Code, SHAMRC, is used to model the heavy dust-laden flow resulting from a non-ideal air blast on a military vehicle using data from the Defense Threat Agency's Large Blast and Thermal Simulator at White Sands, New Mexico (see article, page 4). Cover design by the ERDC MSRC Scientific Visualization Center.

Features

Departments


p.4

p.16


p.20

p.27

John E. West Becomes New Director of the ERDC MSRC - Bradley M. Comes Joins HPC Program Office Staff

By Rose J. Dykes

In August 2002, John E. West stepped into his new role as Director of the ERDC MSRC. John has a long history of service in the MSRC – beginning as an undergraduate when he worked as a contract student in the ERDC Scientific Visualization Center performing studies in virtual reality and volume visualization. Upon completing degrees in electrical and computational engineering at Mississippi State Univeristy, John joined the staff of the ERDC Information Technology Laboratory (ITL) working in the Computational Science and Engineering (CS&E) group supporting studies related to high-performance computing (HPC).

John has served in a variety of functions in the Center, beginning as part of the source selection evaluation team for the original MSRC procurements. While working with the CS&E group, John was one of the principals on an early example of geographically distributed metacomputing as part of the largest finite element-based soil particle simulation ever performed. He has also published in virtual reality, parallel visualization, and numerical algorithm design. Most recently John served as Director of Scientific Computing for the MSRC.

After spending 20 years at ERDC, first as an engineer performing computational studies and later as the Director of the ERDC MSRC, Bradley M. Comes has joined the HPC Modernization Office as the HPC Centers Project Manager. This new position includes oversight of the 14 Centers supported by the HPCMP and oversight of technology insertions (new equipment and capabilities), user support, budgeting, performance monitoring, and compliance with DoD regulations associated with large data centers. The combined capability of the Centers exceeds 20 peak teraflops and serves more than 4,000 users. Additional requirements associated with the position involve cross-coordination with other HPCMP program elements as well as interfacing with other Federal HPC programs.

The ERDC MSRC wishes Brad well in his new position. He says, “The capability offered by the Program is a critical component contributing toward maintaining peace on the homeland as well as abroad. I am really excited about the expanded DoD community with which I will be working in my new position, as it affords me the privilege of working with the many dedicated individuals who are committed to this cause.”

off-campus

ERDC to Present Tutorial at SC2002 - The ERDC MSRC will conduct a tutorial entitled “Dual-Level Parallelism Techniques” at the SC2002 Conference, Baltimore, Maryland, 16-22 November 2002.

Jackson State University (JSU) Summer Institute on HPC

By Ginny Miller

Fifteen students from Minority Serving Institutions were introduced to high-performance computing (HPC) and its applications in science and engineering during the 6th annual JSU Summer Institute on HPC. The 2-week program was held 17-28 June.

Sponsored by the HPCMP Programming Environment and Training (PET) program, the summer institutes are intended to give underrepresented segments of the U.S. student population a first-hand look at research activities and the use of HPC. The summer institutes also foster interest in careers in HPC and the DoD in particular.

"Participants this year, all undergraduate students in mathematics, engineering, or science, were selected from Jackson State University, Texas Southern University, Alcorn State University, and North Carolina A&T University," said Reginald Liddell, PET Education Outreach and Training coordinator technologist at the ERDC MSRC.

"The students represented a good cross section of the country," Liddell said. "They were really interested and very hands-on. I think many of them came away with a genuine sense of what HPC is all about."

JSU serves as the host institution for the Summer Institute program. Since its establishment in 1997, the program has featured lectures and demonstrations given by MSRC staff and the PET university and onsite academic leads on the application of HPC to problems of national importance in science and engineering.

This year, ERDC representatives participated in 4 days of the institute and facilitated question-and-answer discussions, scientific visualization demonstrations, and a tutorial on Web site development techniques. The Engineering Research Center at Mississippi State University (MSU) also played an instrumental role, hosting students for a series of activities and demonstrations during the second week of the institute.

Another highlight of the institute was onsite laboratory tours at the ERDC, for which students were joined by the PET summer interns. The tours, on 21 June, included visits to the Coastal and Hydraulics Laboratory, the Environmental Laboratory, the Geotechnical and Structures Laboratory, and the Information Technology Laboratory.

Participants in the JSU Summer Institute stand in front of the Information Technology Laboratory during an ERDC tour.

SHAMRC Environment and Vehicle Loads Calculations for Non-Ideal Flow

By Joseph Crepeau, Charles Needham, and Robert Newell

Introduction

High-performance computing (HPC) resources have enabled scientists and engineers at Applied Research Associates (ARA), Inc., to perform high-fidelity Computational Fluid Dynamics (CFD) calculations in support of the Defense Threat Reduction Agency (DTRA) Non-Ideal Air Blast (NIAB) program. The NIAB program was designed to quantify the effects of NIAB on military vehicles over a wide range of geographical areas and scenarios. Of particular interest is the ability to simulate computationally the environment and vehicle loads from the exit jet produced by the DTRA Large Blast and Thermal Simulator (LB/TS) located at White Sands, NM. The calculations were run with the parallel version of Second-Order Hydrodynamic Automatic Mesh Refinement Code (SHAMRC) and leveraged HPC resources to produce timely and accurate results.

Results from the calculations are compared with experimental gauge recordings and high-speed photographic data. The comparisons are in excellent agreement. The results of the simulations demonstrate the viability of using SHAMRC to help predict equipment damage from non-ideal blast effects. Without the use of HPC facilities, these calculations could not have been completed within the given time constraints.

The Code

SHAMRC (pronounced shamrock) is a two-dimensional (2-D) and 3-D, finite difference, hydrodynamic computer code. It is used to solve a variety of airblast-related problems that include high explosive (HE) detonations, nuclear explosive (NE) detonations, structure loading, thermal effects on airblast, cloud rise, conventional munitions blast and fragmentation, shock tube phenomenology, dust and debris dispersion, and atmospheric shock propagation. Its capabilities and attributes include multiple geometries, nonresponsive structures, noninteractive and interactive particles, several atmosphere models, multimaterials, a large material library, a K-ε turbulence model, and water and dust vaporization. SHAMRC is second-order accurate in both space and time and is fully conservative of mass, momentum, and energy. It is fast because it employs a structured Eulerian grid and efficient because of the use of the preprocessor SRCLIB, which tailors the source code to each problem.

SHAMRC is a Government-owned, nonproprietary CFD code under export controls that has been supported and developed by DTRA over the past 30 years. The code has the capability to run with a single Eulerian grid or with the Automatic Mesh Refinement (AMR) option that divides the calculational domain into smaller Eulerian grids at multiple levels of refinement to provide high-resolution results. Both solution methods have been parallelized using the Message Passing Interface (MPI) library. The AMR capability of SHAMRC allows the efficient calculation of certain classes of problems that are otherwise intractable using the conventional single-grid method.

Initial single-grid versions of SHAMRC were parallelized in 3-D only, and the Eulerian grid was decomposed in only one dimension. The calculations reported here utilized this parallel model. Since that time, the code has been modified to perform automatic domain decomposition on the Eulerian grid in one, two, or three dimensions for both the 2- and 3-D versions of the code. The parallel version of SHAMRC has been verified on several HPC platforms.

SHAMRC exhibits good scalability. Figure 1 is a comparison of the "whiz factor" (the computational time per cell per cycle) between timing benchmarks and an ideal scalability curve. The chosen test problem was scalable, where the number of zones per processor was fixed over a range of 2 to 64 processors. The comparison shows nearly ideal scalability was achieved for up to 64 processors. The scalability behavior of the 3-D code is comparable to that of the 2-D code.

Calculational Models

Three 3-D calculations were run. The first two calculations computed the exterior environment produced by the exit jet from the LB/TS, which is used to simulate non-ideal airblast. High dynamic pressures and relatively low overpressures, with an extended positive duration of heavily dust-laden flow, characterize non-ideal airblast. Because the test section and access berms outside the LB/TS were constructed of sand, the sand that becomes entrained in the flow significantly

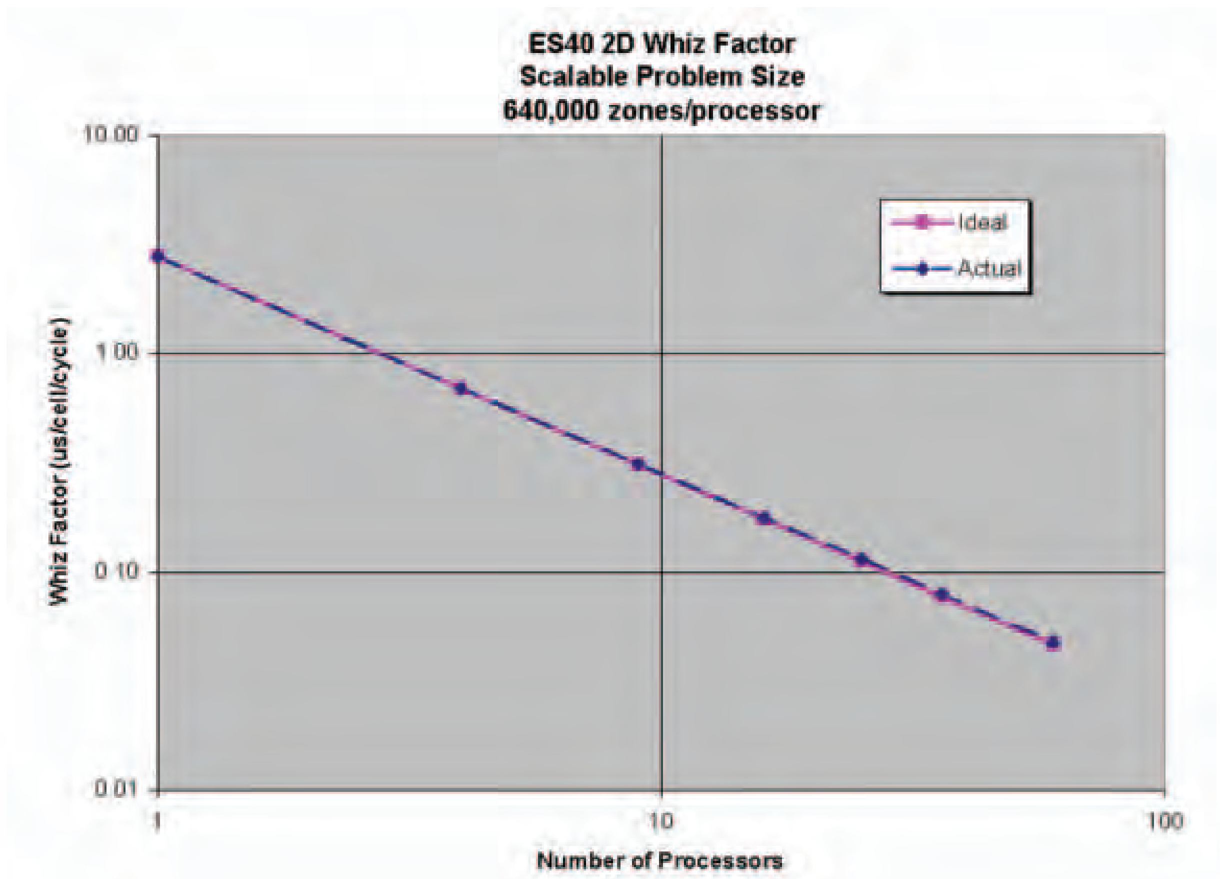


Figure 1. Scalability of 2-D SHAMRC for a scalable problem. Comparison of the calculated versus ideal whiz factor for up to 64 processors. Essentially ideal scalability is achieved.

modifies the airblast. The first environment calculation ignored the dust contribution. The second environment calculation included a dust injection model that is currently available in SHAMRC to simulate the effects of the dust scoured from the berm region.

All calculations modeled the last 25 meters of the shock tube and the structures and terrain exterior to the shock tube. The calculational grid for the environment calculations contained approximately 48 million zones. More than 5,000 monitoring points were placed in the calculation to record time-history data for comparison with experimental results. Figure 2 is a rendering of the calculational model used for the environment calculations. The blue dots represent the locations of "stations" placed in the calculational grid that record time-histories for several field variables.

The third calculation modeled a test of the non-ideal blast effects on an M-110A2 self-propelled Howitzer. It included a nonresponding model of the M-110A2 on a 3-cm resolution subgrid located 25 meters from the end of the shock tube and centered on the test berm. It also included the same dust injection model used in one of the environment calculations. The calculational grid contained more than 150 million zones. Stations were placed around and on the vehicle exterior at numerous locations, including those corresponding to the three experimental gauges. The station recordings were used to provide input to subsequent vehicle response calculations. Figure 3 shows the vehicle as modeled in the loads calculation. The structures outside the LB/TS were the same as in the environments calculations.

Dust Injection Model

For the two calculations containing dust, the $\beta\rho U$ dust injection model was used. The model injects dust into the zones adjacent to a defined surface, proportional to the local fluid density and velocity. This model was first used in a HULL calculation of an HE detonation over a dusty surface. The model is unchanged from its original implementation. The equation used for the mass of injected dust is as follows:

$$\Delta m_i = A_i \beta \rho_i (air) | U_i | \Delta t$$

where

Δm_i = mass injected into cell i over time-step Δt

A_i = cross-sectional area of the cell

β = dust injection constant (0.6667)

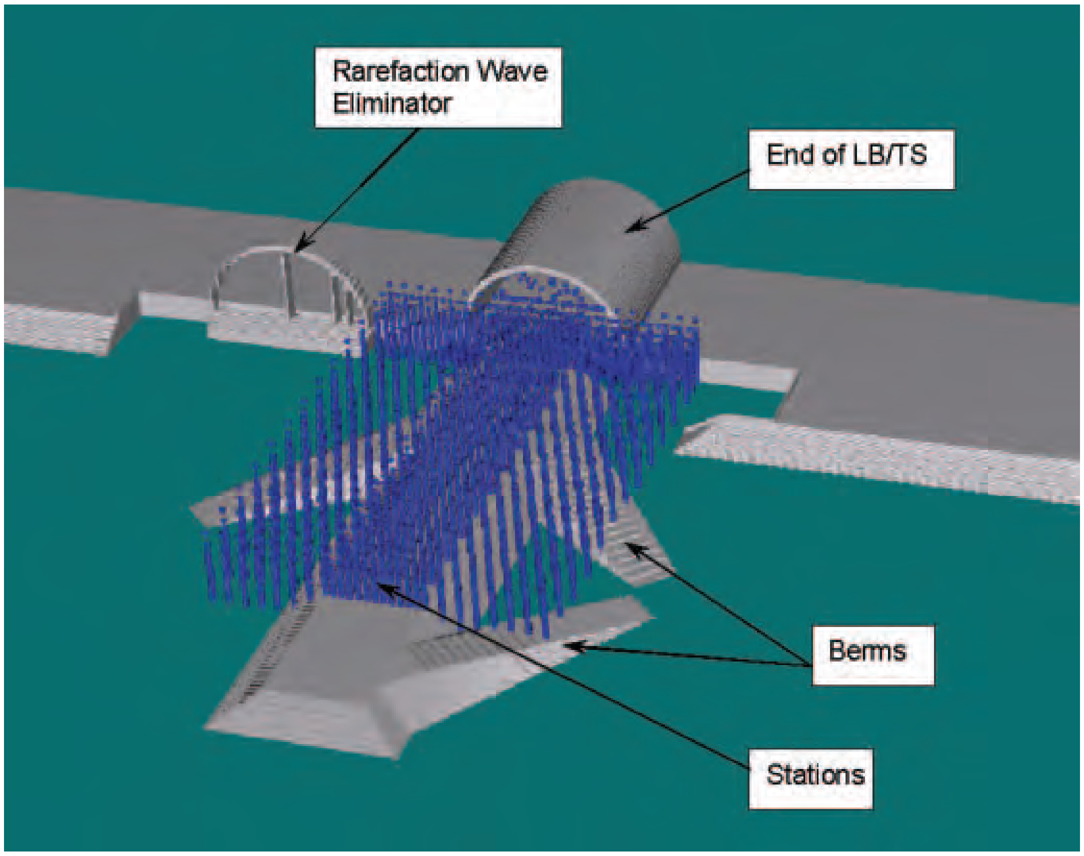


Figure 2. SHAMRC calculational model for the environment characterization simulations. The vehicle load calculation included the vehicle model at a distance of 25 meters from the end of the shock tube.

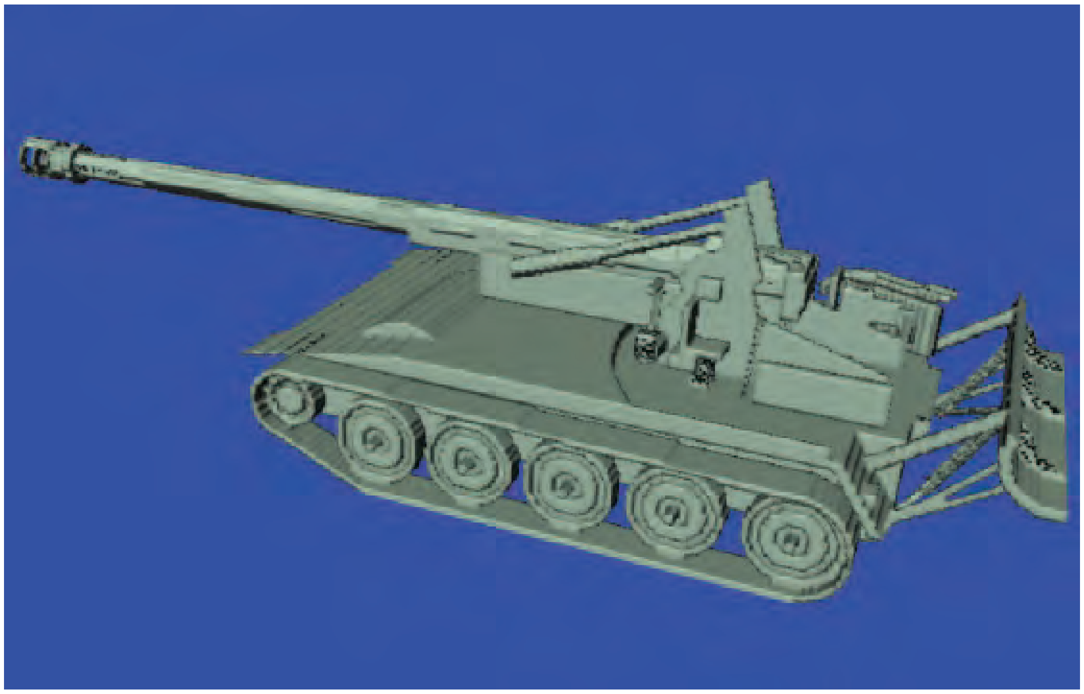


Figure 3. SHAMRC calculational model of an M-110A2 for the vehicle loads simulation.

ρ_i = air density
U_i = velocity of the cell (parallel to the surface)
Δt = time-step

The dust is injected into the cell and treated as a fluid with an initial velocity equal to 1/10th that of the velocity component parallel to the surface and at an angle of 45 degrees to the direction of the flow.

HPC Resources Used

Both the environment and loads calculations were run with the parallel version of SHAMRC. The smaller environment calculations were run on the Nirvana cluster at Los Alamos National Laboratories (LANL). The Nirvana cluster is a collection of 128-processor SGI Origin 2000 boxes. The calculational grid contained approximately 48 million zones and required 70 wall clock hours on 128 processors on the SGI Origin 2000.

The loads calculation was started on the LANL Nirvana cluster and was moved to the more powerful SGI Origin 3800 (Ruby) at the ERDC MSRC and completed. On both systems, the calculation was run on 128 processors. The calculation required a total of 440 wall clock hours to complete. Based on an expected efficiency of 43 percent for an identical size problem, the equivalent runtime for the serial code would be approximately 18,920 hours or 2.2 years.

Each calculation required numerous restart files in order to produce an animation when the calculations were completed. The restart files for the environments calculation with dust were slightly more than 1.3 GBytes per file. Eighty restart files were saved for a total of 104 GBytes of data. The 55 restart files for the loads calculation were just more than 4.3 GBytes per file for a total of 236 GBytes of data.

Results and Data Comparisons

The results of the environment and loads calculations are in excellent agreement with the experimental data. Figure 4 shows the pressure field on the vehicle and testbed surface at 390 ms.[1] A series of images like this was rendered to produce an animation of the pressure loads on the vehicle for the duration of the

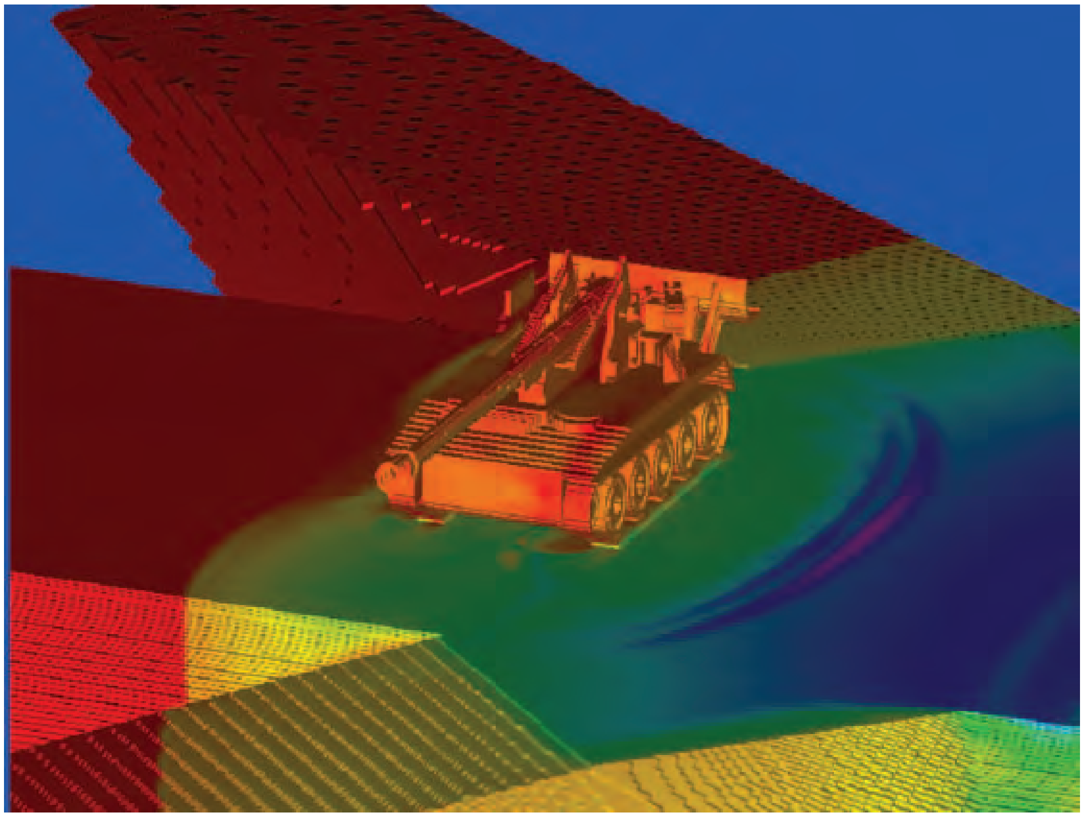


Figure 4. Pressure field on the vehicle and testbed surfaces at 390 ms. Low pressures are indicated by red, and high pressures are magenta. The shock is moving right to left over the vehicle. The reflected shock is visible on the upstream side of the vehicle. The incident shock has just traversed the entire vehicle.

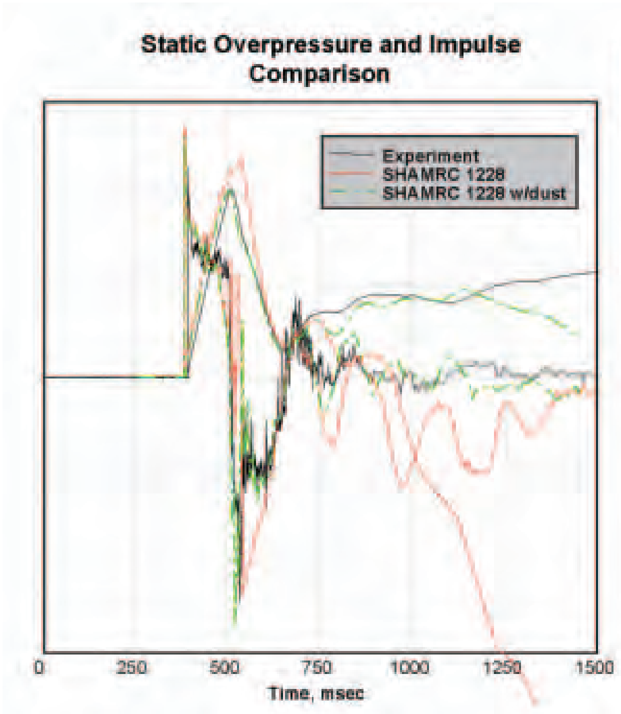
[1]Corrections made 05 May 03.

calculation. Figure 5 is a sample of overpressure waveform comparisons between the experimental data and the calculated data. The figure shows the SHAMRC environment calculation results from the calculations with and without dust overlaid on top of the experimental data trace. The waveform from the calculation without the dust (red dotted line) is in good agreement initially when the dust is not a factor. However, the arrival of a vortex at just more than 500 ms signals non-ideal flow and the calculated waveform begins to deviate substantially from the experiment. This is because of the presence of dust that has been injected into the flow by the vortex. The calculation with dust (green dashed line) is in excellent agreement with the experiment out to times beyond 1 second. The presence of the dust in the vortex modifies the overpressure in this region and fills in the negative phase of the waveform. After the vortex passes, the wild swings seen in the overpressure waveform without dust are reduced significantly and are in line with that observed experimentally. In addition, the overpressure impulse from the calculation with dust agrees extremely well with the experimental overpressure impulse out to late times. Such agreement between experimental data and the dusty flow calculation provided confidence in the predictive capability of SHAMRC for vehicle loads calculations.

Waveform comparisons from the vehicle loads calculation are presented in Figure 6 through Figure 8. Experimental gauges were placed on the vehicle at three locations (upstream, bottom, and downstream sides of the vehicle). Numerous stations were placed in the SHAMRC calculation on the surface of the vehicle model. These stations are indicated in the subfigures as blue dots. The location of the station corresponding to the experimental gauge is indicated as a red dot in the subfigures. The experimental data traces are in black, and the SHAMRC data traces are in blue.

Figure 6 is the comparison on the upstream side of the vehicle, above the tracks and not quite centered about the length of the vehicle. The calculated arrival time of the shock matches the experimental data exactly, as does the initial peak and decay behind the peak. The experimental data show the vortex arriving at about 440 ms. The calculation shows the vortex arriving 10 ms later. The overall waveform and the impulse are in very good agreement out to the time that the gauge breaks at 590 ms. Figure 7 compares the waveforms on the bottom of the vehicle. As with the upstream gauge, the overall agreement between the calculation and the experimental data is quite good. Figure 8 is a comparison on the downstream side of the vehicle. Again, the calculated arrival time is in excellent agreement with the experiment. Note that the pressure behind the initial shock from the experimental gauge increases to a peak and then decays into a strong negative phase. The calculated pressure waveform exhibits the same behavior and matches the rise to the peak and the decay into the negative phase. The overpressure impulse is also in agreement.

Photographic data were also available from the tests. A comparison of a single frame from the experimental video and from the calculation at 690 ms is shown



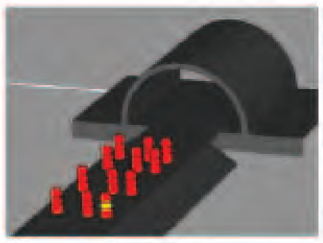
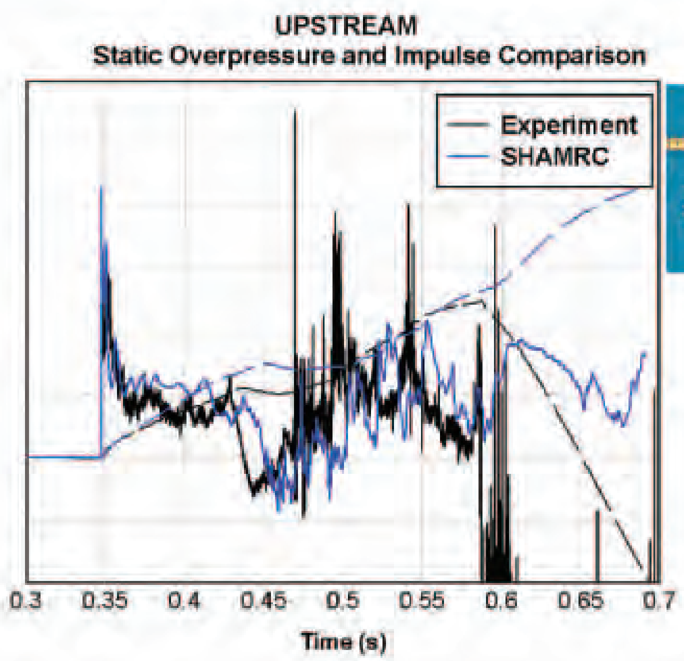


Figure 5. Waveform comparison between environment characterization test and SHAMRC. SHAMRC calculations with (green, dashed) and without (red, dotted) dust. The insert shows the location of this station in yellow.



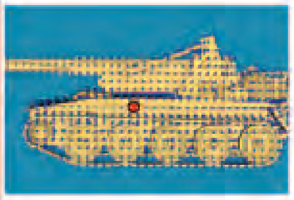
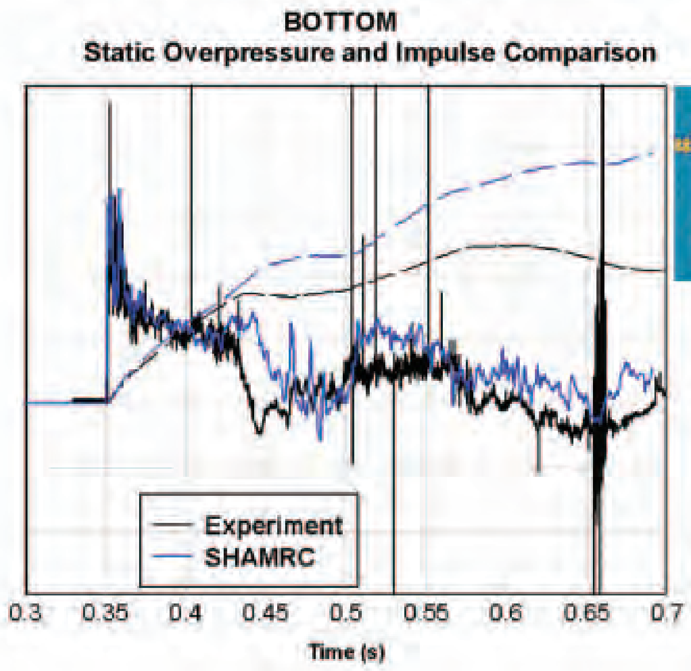


Figure 6. Overpressure waveform comparison between experimental gauge recording (black) and SHAMRC calculation (blue) on upstream side of vehicle. The insert shows the stations on the vehicle surface (blue dots) and the location of this station in red.



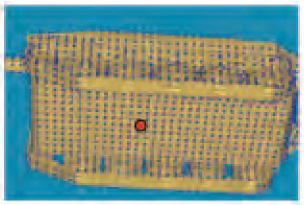
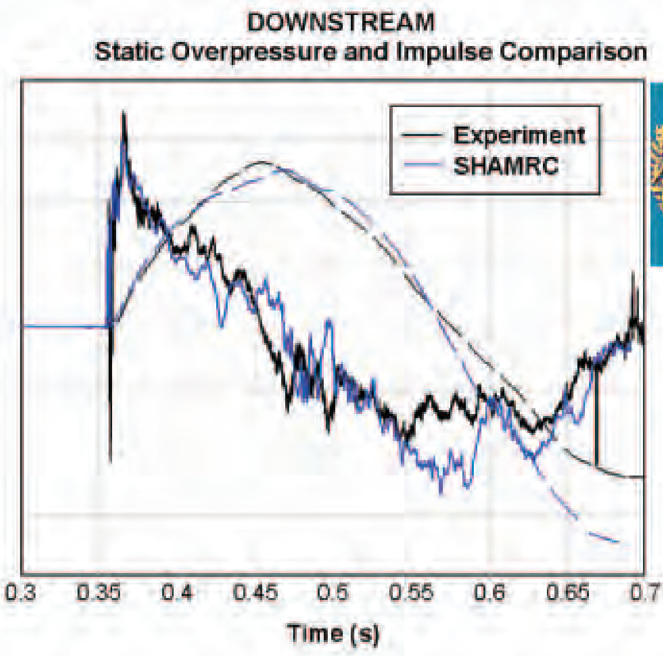


Figure 7. Overpressure waveform comparison between experimental gauge recording (black) and SHAMRC calculation (blue) on bottom of vehicle. The insert shows the stations on the vehicle surface (blue dots) and the location of this station in red.

in Figure 9. The image from the calculation is a 3-D rendering of a dust density isosurface at a value of 1.e-5 grams/cc. At this point in time, only the gun barrel and a small portion of the rear of the vehicle are visible. The agreement between the calculation and the test is remarkable. In fact, a time animation was produced from the calculation from essentially the same camera angle as that from the test. The authors worked closely with members of the ERDC MSRC Scientific Visualization Center (SVC) graphics team to produce the animation. The calculation shows the same overall dusty flow conditions as that seen in the test—that is, the relatively small dust injection into the flow until the arrival of the vortex, the height and angle of the dust in the flow, and the coverage of the vehicle by the dust in the flow.

Summary

HPC resources were successfully utilized to perform high-resolution SHAMRC calculations of the LB/TS exit jet environment and loads on a military vehicle. The environment calculations showed the importance of modeling the entrainment of dust in the flow field in



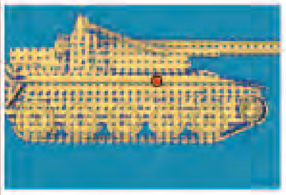


Figure 8. Overpressure waveform comparison between experimental gauge recording (black) and SHAMRC calculation (blue) on downstream side of vehicle. The insert shows the stations on the vehicle surface (blue dots) and the location of this station in red.

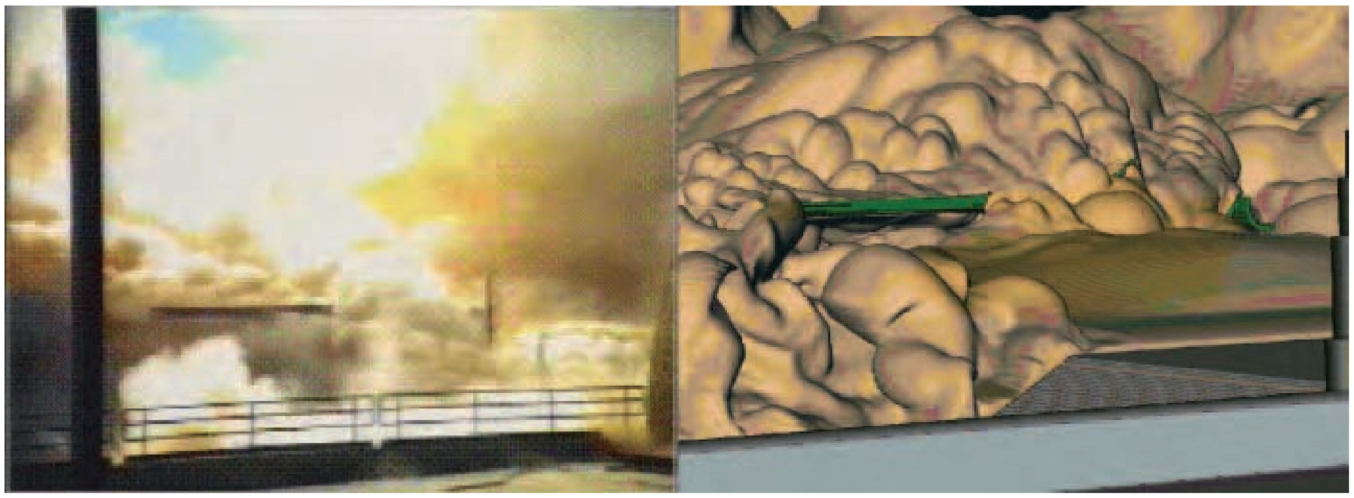


Figure 9. Comparison of test video with SHAMRC image at 690 ms.

order to properly characterize the exit jet environment. Comparisons of overpressure time-histories between the tests and calculated data for both the environment characterization and the vehicle loads tests are in excellent agreement. Comparisons were also made between photographic data from the test and a 3-D rendering of dust isosurfaces from the calculation and were in good agreement. The authors of this paper and members of the graphics support staff at the ERDC MSRC SVC produced an animation of the calculation. The resemblance between the test video and the calculation animation is striking.

The results of this effort would not have been possible without the use of HPC resources. This includes not only the computer time and data storage but also the use of the expertise of the graphics team at the ERDC MSRC. By leveraging HPC resources, it was possible to complete highly reliable simulations of LB/TS tests in a timely manner. The results of the simulations allowed scientists to better understand the complex phenomena associated with non-ideal airblast and their interaction with vehicles.

Multi-Level Parallelism (MLP) – An Alternate Parallel Programming Paradigm

By Drs. Fred T. Tracy and Daniel Q. Duffy and Robert W. Alter

As the hardware architectures of highly parallel machines have advanced, the programming methods necessary to effectively use these machines have become more sophisticated. Originally, most high-performance computers started out as strictly distributed-memory architectures, that is, a large number of individual processors with separate memory connected by a high-speed internal network. The Message Passing Interface (MPI) quickly became the dominant method of communicating among processes on distributed processing elements (PEs). The challenge with MPI is the need to manually distribute the work evenly over the PEs and to communicate among the PEs using broadcasts, reductions, sends, receives, etc.

Many of the newest architectures of massively parallel machines are built with large amounts of shared-memory space. That is, either within a node or across the entire machine, each processor can access the entire memory address space. Several vendors have made individual nodes with up to 100 or more processors sharing the same memory address space.

In its Origin line of computers, SGI has taken the concept of shared memory and applied it to the entire machine. Thus, computers with as many as 512 or 1,024 processors, which share the same memory address space, are becoming more and more common. For the SGI Origin series, each processor accesses memory in a Cache Coherent Non-Uniform Memory Access (cc-NUMA) fashion. In other words, though each process can access all the memory across the entire machine, the paths taken to get to all of the memory are of different lengths for each processor. Hence, a single process can access memory on its own node very quickly, but it must pay a latency price when accessing memory that physically resides on another node. Here, OpenMP directives have become a popular tool. Strategically placed directives before code such as FORTRAN do loops allow the system to partition the work for the user and avoid message passing. The challenge with OpenMP, however, is that this parallel paradigm does not typically scale to a very high number of PEs.

Within the last few years, a dual-level parallelism approach has offered developers a new tool for creating highly scalable codes. An obvious way to do that is to combine MPI processes with OpenMP threads (see Figure 1). What is the difference between a process and a thread? The difference between a process and a thread is that typically, a process contains the entire information about a program, its resources, and execution state, e.g., stacks and registers. Furthermore, a process has its own memory address space that cannot be accessed by other processes unless a shared-memory address space is declared by all the processes. MPI processes, by design, share no memory and must have explicit message-passing calls in order to communicate. Meanwhile, threads can easily share the same memory address space while maintaining thread-specific data. Furthermore, the latency of spawning threads is much less than the creation of heavier weight processes. Hence, given enough work to amortize the creation of threads, it becomes viable to have processes spawn threads during computationally intensive parts of applications.

Recently, a new technique has been developed specifically for shared-memory Origin architectures called Multi-Level Parallelism (MLP). Rather than use MPI processes and the associated MPI libraries to pass messages, MLP forks processes and allows developers to declare a shared-memory space into which are placed variables that need to be accessed by all the forked processes. Hence, no explicit message passing is necessary; simply copying to or reading from the appropriate variables in shared-memory space is enough. As with MPI message-passing synchronization, applications developers must take care to explicitly synchronize the reading and writing of shared variables. OpenMP threads are then used in the same way as with the MPI codes; the threads are placed at highly computationally intense sections of the application to provide the dual-level parallel approach.

The runtimes for a Gaussian Elimination code and the computational fluid dynamics (CFD) program, OVERFLOW, developed at NASA Ames will now be given for both MPI and MLP versions for performance comparisons. The machine used throughout this study is the 512-processor SGI Origin 3800 located at the ERDC MSRC.

The first example of performance testing is to solve the system of N simultaneous, linear equations

$$Ax = b \qquad (1)$$

using Gauss Elimination. The algorithm multiplies a given row by a value and adds it to other rows to achieve an equivalent set of equations in upper diagonal form. A back substitution process is then used to compute the solution. After converting the first column below the main diagonal term to zeroes, the system is given by

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ 0 & a'_{2,2} & a'_{2,3} & a'_{2,4} \\ 0 & a'_{3,2} & a'_{3,3} & a'_{3,4} \\ 0 & a'_{4,2} & a'_{4,3} & a'_{4,4} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{Bmatrix} \qquad (2)$$

Each row is assigned to a processor. Each time a column of zeroes below the main diagonal is done, all PEs need the leading row (row 1 for the first column, row 2 for the second column, etc.). Figure 2 shows how this is done in MPI using an MPI_BCAST, and Figure 3 shows how the shared variable, buff, is used to get the row to all PEs using MLP.

First, an MPI only version, an OpenMP only version, and three MLP versions with only one thread were considered. Figure 4 shows a log-log plot of speedup for the different versions using only the best MLP version. The OpenMP version does not scale after 8 PEs, and the MLP version is as good as the MPI version until approximately 64 PEs. Next, combinations of forked process and OpenMP threads are used to take full advantage of the MLP library. Figure 5 shows speedup for combinations of 16 for the best MLP version. For this application, only slight improvement is achieved by combining forked processes with OpenMP directives.

OVERFLOW is a CFD compressible Navier-Stokes flow solver for structured grids. It is extensively used at Government research centers and in industry to evaluate new aircraft designs. OVERFLOW is one of the largest consumers of NASA supercomputing cycles at the National Aerodynamic Simulator at the NASA Ames Research Center, which has developed the code over the past two decades. Large simulations of flow around complete aircraft configurations are routinely undertaken. OVERFLOW solves the viscous compressible flow Reynolds-average Navier-Stokes equations with turbulence modeling and plays an important role in aerodynamic design processes.

OVERFLOW can use single-block grids or Chimera overset structured grids. A typical solution approach is to solve for the fluid motion in the region of interest using a series of connected 3-D grids, or zones, that cover the region in a patchwork fashion. Typical problem sizes range from small calculations involving about one million points to simulations of an entire aircraft requiring tens of millions of points.

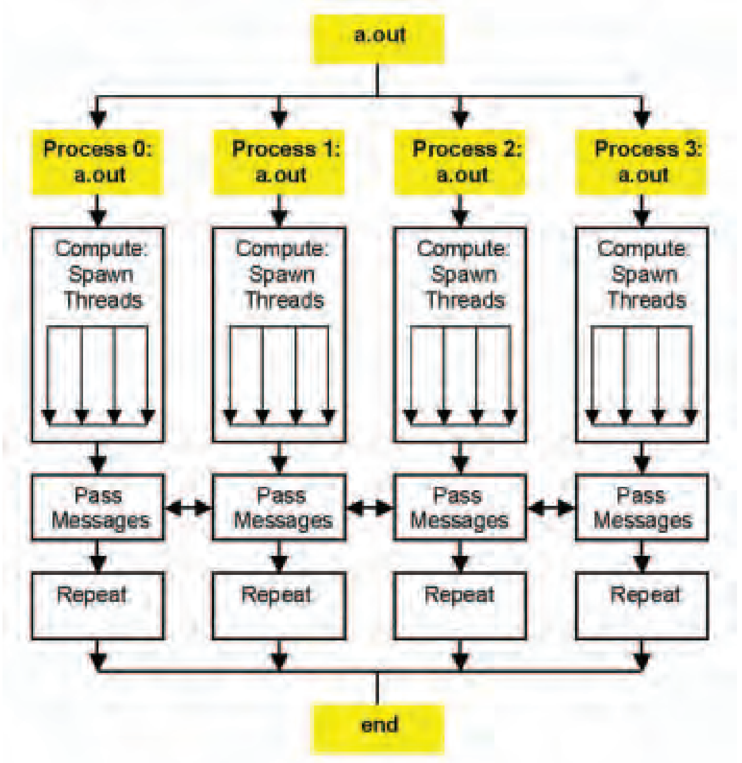
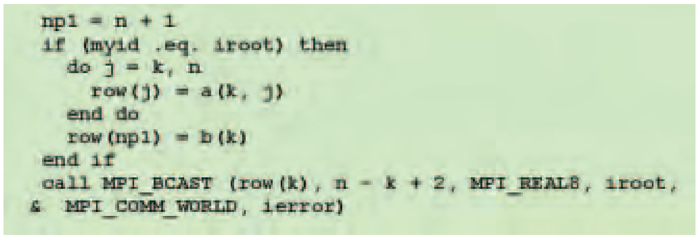


Figure 1. Four processes spawning four threads each



```
  np1 = n + 1
  if (myid .eq. iroot) then
    do j = k, n
      row(j) = a(k, j)
    end do
    row(np1) = b(k)
  end if
  call MPI_BCAST (row(k), n - k + 2, MPI_REAL8, iroot,
&  MPI_COMM_WORLD, ierror)
```

Figure 2. MPI row communication

To test the two versions of OVERFLOW, two test cases were obtained from Dennis Jespersen and James Taft of NASA Ames. The first test case (Trap Wing 7m) consists of 14 grids with approximately 7 million total grid points. A second test case with finer mesh resolution contains 32 grids with approximately 14 million grids points (Trap Wing 14m). To properly understand the performance of OVERFLOW, it is important to realize that the MPI and the MLP versions are two separate instances of the code. One of the most significant differences is that the work is partitioned differently: the MLP version is more sophisticated.

Figure 6 shows a log-log plot of the speedup for the smaller problem (Trap Wing 7m) while varying the total number of processors between 16 and 64 using the MPI time for 16 processors as the baseline. Also, the ratio of total processors to MPI forks is 8. For this small case, it is difficult to ascertain with any degree of certainty which method is more efficient. Figure 7 shows a linear plot of the speedup for the larger

```
npl = n + 1
if (myid .eq. iroot) then
  do j = k, n
    buff(j) = a(k, j)
  end do
  buff(npl) = b(k)
end if
call mlp_barrier (myid, noproc)
do j = k, npl
  row(j) = buff(j)
end do
```

Figure 3. MLP row communication

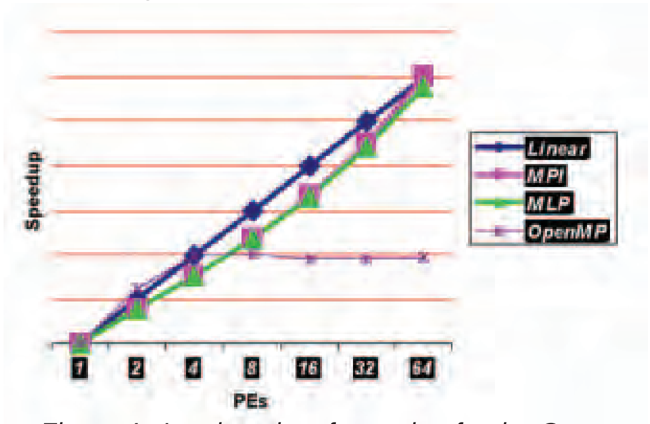


Figure 4. Log-log plot of speedup for the Gauss Elimination code

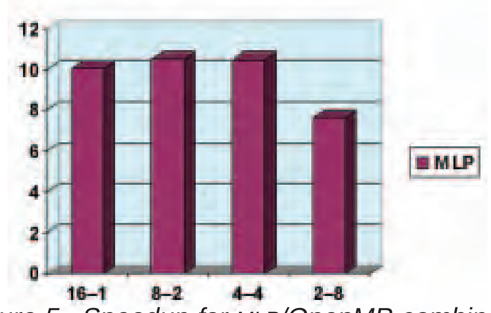


Figure 5. Speedup for MLP/OpenMP combinations

problem (Trap Wing 14m) while varying the total number of processors between 128 and 384 using the MPI time for 128 processors as the baseline. As before, the ratio of total processors to MPI forks is 8. In the larger case, the MLP version of the application clearly performs better than the MPI code.

The overall conclusions found in this study are as follows:

- The newly developed MLP library provides application programmers with another quality tool with which to create parallel codes.
- Typically, MLP is easier to implement than MPI and generally results in a significantly lower amount of code development. However, as with MPI message-passing synchronization, application developers must take special care to use barriers where appropriate to synchronize the reading and writing of data to shared memory.

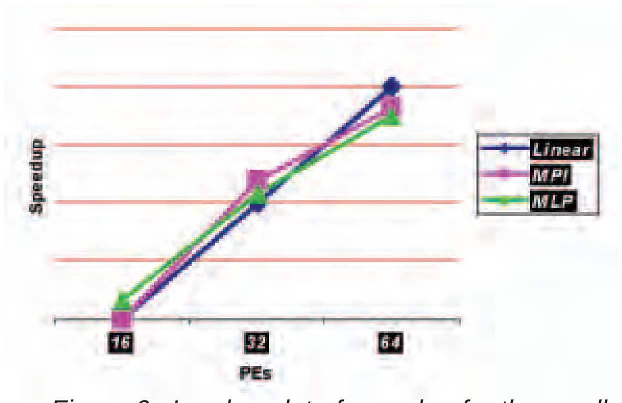


Figure 6. Log-log plot of speedup for the small problem

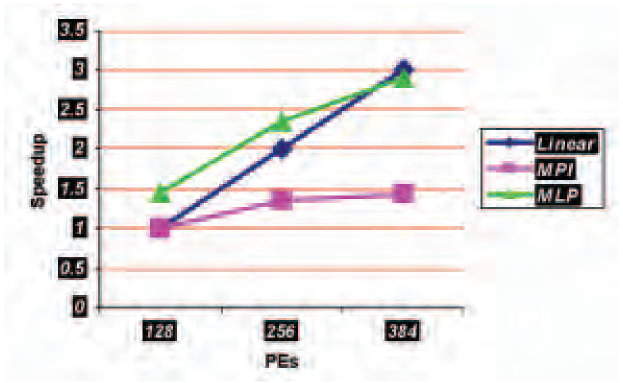


Figure 7. Linear plot of speedup for the large problem

- MLP performs well on both simple codes and large real-world engineering and scientific applications. Hence, MLP is versatile and can be used to solve a variety of problems.
- For some applications, MLP has greater capability to effectively load balance the work in a code than MPI.

References

MPI – The Complete Reference, Volume 1, The MPI Core, Marc Snir, Steve Otto, Steven Huss-Lederman, David Walker, and Jack Dongarra, The MIT Press, Cambridge, 1999; and *MPI – The Complete Reference, Volume 2, The MPI Extensions*, William Gropp, Steven Huss-Lederman, Andrew Lumsdaine, Ewing Lusk, Bill Nitzberg, William Saphir, and Marc Snir, The MIT Press, Cambridge, 1999.

OpenMP Standard, *http://www.openmp.org*.

Performance of the Overflow-MLP CFD Code on the NASA Ames 512 CPU Origin System, James R. Taft, NAS Technical Report, NAS-00-005, March 2000.

Overflow Gets Excellent Results on SGI Origin 2000, Jim Taft, NAS News, January-February 1998, Volume 3, Number 1.

Performance Implications of Process and Memory Placement Using a Multi-Level Parallel Programming Model on the Cray Origin 2000, Sheila A. Faulkner, NASA Ames Technical Report, *http://www.nas.nasa.gov/~faulkner/numa.html*.

Scheduling Strategies for Parallel Systems

By Dr. William A. Ward, Jr.

The order in which waiting jobs are scheduled on a parallel system is important for two reasons. First, it affects job wait time, probably the most important measure of quality service from a user perspective, and second, it affects system utilization, probably the most important performance measure from an administrative viewpoint. Many schemes have been proposed to schedule jobs, both on uniprocessors as well as multiprocessors, but nearly all involve assigning jobs a priority. Perhaps the simplest approach is to use wait time as the priority. More often, other job parameters, such as user-estimated runtime and requested number of CPUs, enter into the priority calculation.

In addition to the priority, a second feature that differentiates between scheduling algorithms is the action that is taken on the prioritized job list. One possibility is to repeatedly start the highest priority job until no more jobs can be initiated. Then when enough CPUs become available as jobs complete, the current highest priority job is launched. Although straightforward, this approach leads to long wait times and low utilization, particularly when the highest priority job needs many CPUs.

In this case, many CPUs may remain idle as the system waits for enough to become free to start the next job. One solution to this problem is to use checkpoint/restart to allow many jobs to periodically receive service and make progress. However, much time may be wasted swapping jobs, and in any case, many operating systems do not support this feature.

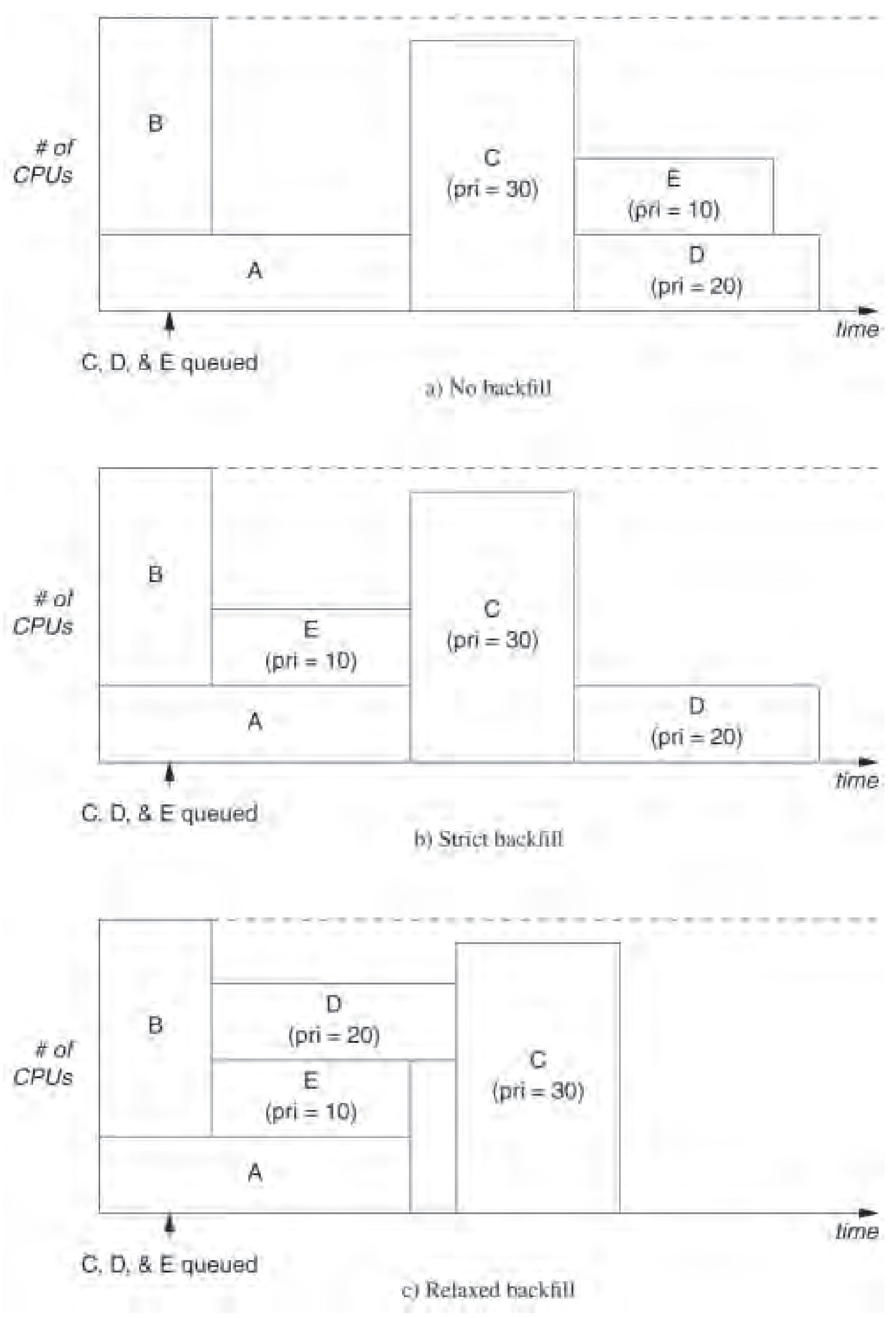


Figure 1. Schedule for running five jobs under three scheduling policies

Another, more feasible approach involves scheduling lower capacity jobs that will run using available CPUs as long as running them will not delay the earliest possible initialization of the current highest priority job. This is called "backfill," and typically results in lower wait times and higher system utilization.

Recently, Bill Ward, Carrie Mahood, and John West of the ERDC MSRC formulated a variant of this technique, called relaxed backfill. Here, a lower priority job may be backfilled as long as it does not delay the highest priority job "too much." The allowable delay is quantified by a parameter ω. If the earliest the highest job priority could start would be in 10 minutes, and $\omega = 1.5$, then a lower priority job would be started if it caused a delay of no more than $1.5 \times 10 = 15$ minutes. Figure 1 illustrates the method.

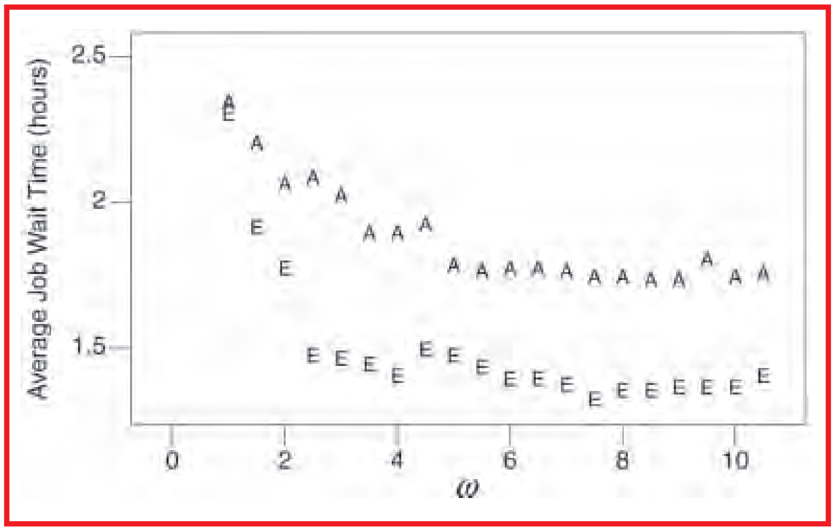


Figure 2. Average job wait times on the T3E versus ω. "E" denotes times for jobs scheduled based on user-estimated runtimes, and "A" denotes times for jobs scheduled based on actual runtimes. The rightmost points correspond to $\omega = \infty$.

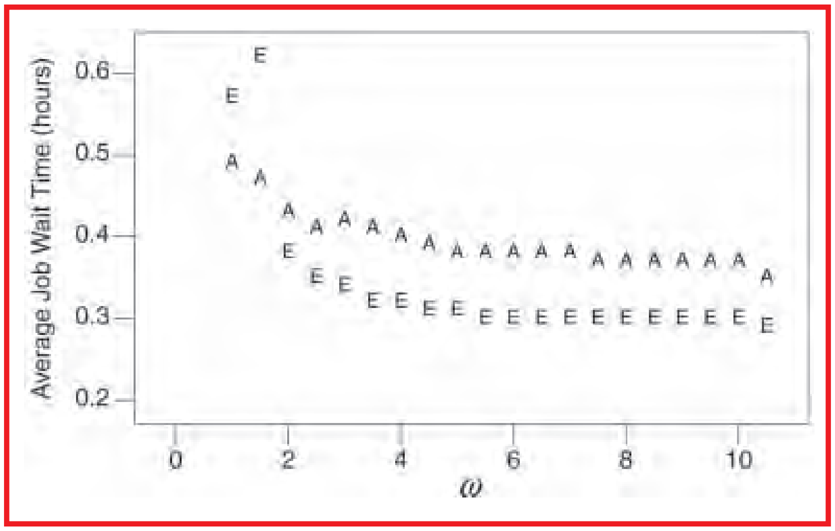


Figure 3. Average job wait times on the O3K versus ω. "E" denotes times for jobs scheduled based on user-estimated runtimes, and "A" denotes times for jobs scheduled based on actual runtimes. The rightmost points correspond to $\omega = \infty$.

To explore the merits of this approach, a scheduler simulator was written in Perl, and run logs from the ERDC MSRC Cray T3E and the SGI Origin 3800 were used to supply input data. Simulated runs, each covering several years of service, were performed for ω ranging from 1 to 10 in increments of 0.5, and using both user estimates for job runtime, as well as (in practice unavailable) exact values for job runtime.

As shown in Figures 2 and 3, relaxed backfill would have reduced average job wait time on both systems by a significant amount. Furthermore, it was discovered that relaxed backfill performs better in terms of average wait time than standard backfill as system utilization increases. A downside to the approach is that very large jobs (those requiring roughly half or more of the system's CPUs) suffer increased delays with increasing ω. The authors have several heuristics in mind that may remedy this shortcoming and intend to use the simulator to address them.

A more complete discussion of the method may be found in a paper to be published in the proceedings of the 8th Workshop on Job Scheduling Strategies for Parallel Processing.

Computational Fluid Dynamics (CFD)/Computational Structural Mechanics (CSM) Joint Workshop on Fluid-Structure Interaction

By Dr. Wayne Mastin

Dr. Bob Meakin, DoD Computational Fluid Dynamics Computational Technology Leader, gives a survey presentation entitled "A Perspective on Priorities and Emerging Computational Technologies."

The ability to efficiently and accurately calculate the solution of multidisciplinary problems is steadily increasing as computing power increases. The need to solve fluid-structure interaction problems is a primary motivating factor leading the development of multidisciplinary computational capabilities. The CFD/CSM Joint Workshop on Fluid-Structure Interaction provided an overview of the state of the art in computational modeling for fluid-structure interaction problems. The workshop also included information for those interested in developing an entry-level computational capability in this area.

The workshop was held at the ERDC MSRC on 30-31 July 2002. The event was sponsored by the Programming Environment and Training (PET) program and organized by Professor Bharat Soni of the University of Alabama at Birmingham (UAB) and Professor Tinsley Oden of the University of Texas at Austin (UT Austin). Professor Soni is Chair of the Department of Mechanical Engineering at UAB and leads all PET efforts in the CFD Computational Technology Area (CTA). Professor Oden is the Director of the Texas Institute for Computational and Applied Mathematics (TICAM) at UT Austin and plays a leading role in PET CSM activities. The workshop brought together leading experts in the fields of CFD and CSM, the two CTAs supported by the PET program through Component 3 hosted at the ERDC MSRC. Organizational support for the workshop was provided by Mr. Bob Athow, Government Technical Advisor at ERDC, Dr. Wayne Mastin, the component POC at ERDC, and Dr. Nathan Prewitt, CFD onsite lead at ERDC.

There were 42 participants at the workshop, including 12 local ERDC users from the Coastal Hydraulics Laboratory and Geotechnical and Structures Laboratory. Of the 30 visitors, 18 were from other DoD facilities. The other 12 visitors were from universities, NASA, and software vendors supporting DoD research and development activities.

On the first day of the conference the participants were welcomed to ERDC by Dr. Jeffery Holland, ERDC ITL Director. This was followed by the first Keynote Address, "N-Field Nonlinear Computational Aeroelasticity: From Theory to Practice," presented by Professor Charbel Farhat of the University of Colorado. There were also two presentations on the first day highlighting efforts of the HPCMP in the area of fluid-structure interaction. Dr. Bob Meakin, CFD CTA leader, gave a survey presentation titled "A Perspective on Priorities and Emerging Computational Technologies" and Dr. Raju Namburu, Common High Performance Scalable Software Initiative (CHSSI) Weapon-Target Interaction Portfolio Leader, described some of his team's work in the presentation titled "CHSSI Portfolios and Software Frameworks." There were six other presentations on the first day dealing with different tools and frameworks for solving fluid-structure interaction problems. The day concluded with a panel discussion led by Professor Oden. A workshop banquet on the evening of the first day provided an opportunity for further informal discussion.

Dr. Reid Melville, Air Force Research Laboratory at Wright-Patterson Air Force Base, presents a Keynote Address on "Problems, Solutions, Experiences in Fluid-Structure Interaction or Lessons My F-16 Taught Me."

Dr. Reid Melville, Air Force Research Laboratory (AFRL) at Wright-Patterson Air Force Base, led off the second day of the workshop with his Keynote Address on "Problems, Solutions, and Experiences in Fluid-Structure Interaction or Lessons My F-16 Taught Me." This was followed by eight presentations organized into two invited sessions. The first session dealt with methodology for solving multidisciplinary applications, and the second session presenters discussed fluid-structure interaction capabilities available in current software systems.

The participation in this inaugural workshop has verified that fluid-structure interaction is an area of significant interest to the DoD user community. Further activities in this area are planned for the future.

CFD/CSM Workshop attendees

2002 HPCMP PET Summer Intern Program Prepares Students

By Ginny Miller

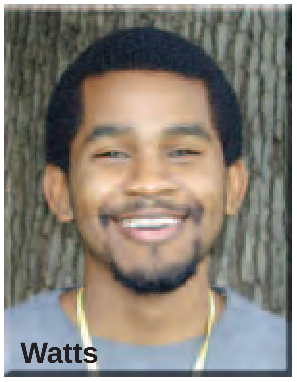
2002 HPCMP PET summer interns are (from left) Kristin Stechschulte, Owen Eslinger, Roderick Royal, Amber Overholser, and Marvin Watts

After 10 weeks spent at the ERDC MSRC, five college students credit the HPCMP PET Summer Intern Program for future employment options, a broader knowledge base, and real-world experience.

"This was my first internship in which I put my education to use," said Amber Overholser, a junior studying computer science and mathematics at Ashland University in Ashland, OH. "Now I will know what to expect (after college) in a computer-related atmosphere."

"The goal of the program is to encourage bright young students, especially minorities and women, to consider careers in HPC research within the DoD," said Dr. Wayne Mastin, the PET summer intern coordinator at the ERDC MSRC. During this year's program, from 3 June - 9 August, interns were matched with research mentors and gained valuable experience in areas such as computational science, information technology, computer science, scientific visualization, and computer simulation.

Performing work related to her college major is what Overholser, 20, most enjoyed about her internship. Under the supervision of Lee Higbie, a computational scientist with the ERDC MSRC's Computational Science and Engineering group, Overholser developed a graphical user interface program written in Java that compared two images and then magnified them to find the differences.

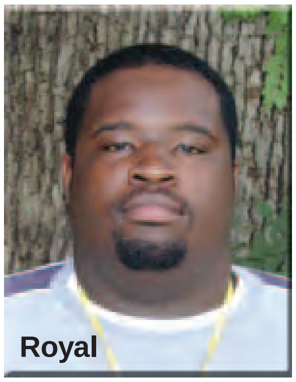


An aspiring research scientist in Computational Fluid Dynamics (CFD), Marvin Watts found he benefited most from the mentoring aspect of the summer internship program. "I enjoyed the exposure to people doing what I hope to do one day," said the 22-year-old, a systems science doctoral student at Clark Atlanta University who worked with Dr. Nathan Prewitt, the CFD onsite lead, on projects including a simulation of a long shore Rapidly Installed Breakwater.

"I think my experience as a PET intern will provide me with a broader knowledge base," Watts said.

Roderick Royal, 21, a senior computer science major at Stillman College in Tuscaloosa, AL, is hoping his internship helps him land a job. “Hopefully the experience will open up future employment options for me,” he said. Mentored by Dr. Mastin, Royal condensed and streamlined an extensive spreadsheet of training course topics germane to HPC software applications.

“What I liked most about being an intern at ERDC is the working environment,” said Royal, who heard about the PET program from a college professor. “The employees are very helpful, and it gave me a chance to work in my field of study.”



“My internships have given me contacts and a view into the experience of working in a lab,” said Owen Eslinger, a 4th-year graduate student at the University of Texas, Austin. Building on work begun as a PET intern in 2001, Eslinger was again paired with Dr. Fred Tracy (ERDC Information Technology Laboratory) and Dr. Stacy Howington (ERDC Coastal and Hydraulics Laboratory) on the ADH and FEMWATER projects.

Thanks to his internships, “I better understand the difference between the academic and the industrial world and how the two communicate,” said the 25-year-old, who plans to work in a national laboratory or industry after completing his Ph.D. in computational and applied mathematics.

Real-world experience is what Kristin Stechschulte sought from her internship. While assigned to the ERDC MSRC Scientific Visualization Center (SVC), Stechschulte, 21, a senior studying computer science and mathematics at Ohio Northern University in Ada, OH, worked with SVC lead Paul Adams and his staff to create an immersive data analysis application for the ImmersaDesk. She also worked with the Visualization Toolkit and wrote code for looping through and displaying images.

“I received advice that will be applicable in future situations,” Stechschulte said, adding that the experience “gave me a flavor of what working is like.”

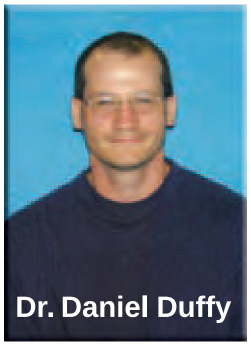
Throughout their internships, students were actively engaged in the Summer Lecture Series, receiving an introduction to the CFD, Computational Structural Mechanics (CSM), Environmental Quality Modeling and Simulation (EQM), and Climate/Weather/Ocean Modeling and Simulation (CWO) research topics from, respectively, CFD onsite lead Dr. Nathan Prewitt, CSM onsite lead Dr. Richard Weed, EQM onsite lead Dr. Jeff Hensley, and CWO onsite lead Dr. Phu Luong.

“The 2002 summer interns, selected from a pool of 41 applicants, were attentive during group meetings and appeared to adjust quickly to the ERDC work environment,” said PET Education Outreach and Training coordinator/technologist Reginald Liddell, who organized the summer intern activities. They arrived promptly and eager for work each day, expressed an appreciation for challenging projects, and also showed initiative for additional work assignments.

Students who apply for an intern position must be enrolled in an undergraduate or graduate program at a university or college and be pursuing a degree in science, engineering, or mathematics. Summer interns must also be planning to continue their education in the fall, should have completed their sophomore year by the beginning of the program, and must be U.S. citizens. All of the interns, who are expected to work 40 hours per week, receive a stipend based on their academic level.

The Practical Supercomputing Toolkit

By Dr. Daniel Duffy and David Sanders



Have you ever forgotten the command to submit a job on one of the high-performance computers throughout the Program? Is it qsub, bsub, or llsubmit? When you received a new account at a site, how easy was it to learn the ins and outs of the local mass storage solution? Is it msfput, msfget, or can I just copy files?

If you have had difficulty with any of the above issues, you are not alone. As the complexity of both the hardware and software throughout the Program grows, it has become increasingly difficult for users, both expert and novice, to easily and effectively make use of available resources.

Motivation and Goals

The goal of the Practical Supercomputing Toolkit (PST), often referred to as the Uniform Command Line Interface (UCLI), is to provide a uniform interface between site, machine, and batch queuing system commands to ease the difficulty of using HPCs. In essence, users will only have to learn a single set of commands that can be applied wherever the PST is installed. Hence, users will not have to recall what command is required to submit a job or access mass storage on a given machine at a given site (see Figure 1).

In general, the PST will provide a stand-alone software layer that can be installed piecemeal or in its entirety. The planned layers of the toolkit include the following:

- Tools for the Unification of SuperComputing (TUSC): a set of standards and utilities for unified command line interface for archiving and job queue submission.
- Source Editing Tools (SET): a set of standard routines for the preparation of source codes for execution.
- Management and Documentation (MD): routines used to install and maintain the PST.
- Application Cookbook Tools (ACT): a repository of sample codes and tutorials used to disseminate lessons learned.

The open source package is written entirely in Perl and is built on two types of modules, site-independent and site-dependent. As the name implies, only the site-

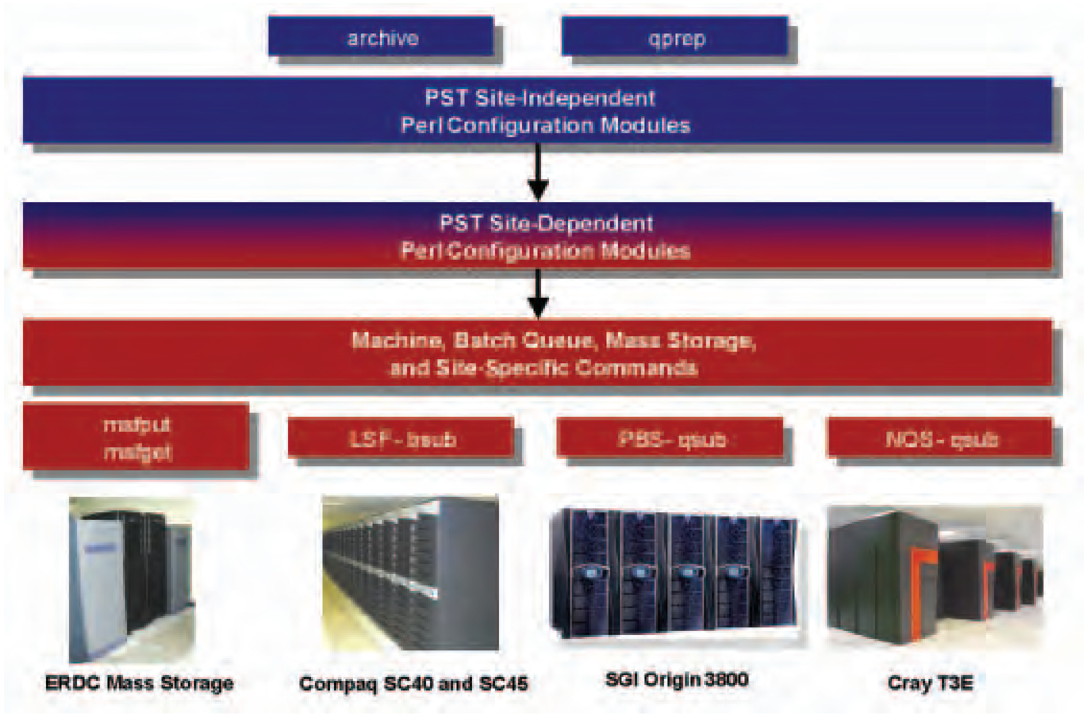

archive
qprep
PST Site-Independent Perl Configuration Modules
PST Site-Dependent Perl Configuration Modules
Machine, Batch Queue, Mass Storage, and Site-Specific Commands
msfput msfget
LSF - bsub
PBS - qsub
NQS - qsub
ERDC Mass Storage
Compaq SC40 and SC45
SGI Origin 3800
Cray T3E

Figure 1. The current PST contains qprep *for job submission and* archive *to access mass storage.*

dependent files need to be configured for any given installation. Many site-dependent files have already been created for a variety of architectures and batch queuing systems. Check the PST Web site, *http://www.pstoolkit.org*, for more information.

Available Commands

Currently, the two commands that are available in the TUSC layer of the PST are the following:

- *archive*: provides users with a consistent set of commands to the local archival storage system. For example, Table 1 shows the equivalent PST commands for the ERDC MSRC site-specific commands to access the mass storage system. Wherever the PST is installed, the *archive* commands may be used to manage mass storage accounts.

Table 1. The PST Translates Local Archival Storage System Commands into Common Instructions That May Be Used Wherever the PST Is Installed

ERDC MSRC Specific Command	PST Equivalent
msfput *filename*	archive get *filename*
msfget *filename*	archive put *filename*
msfmkdir *directoryname*	archive mkdir *directoryname*
msfls	archive ls

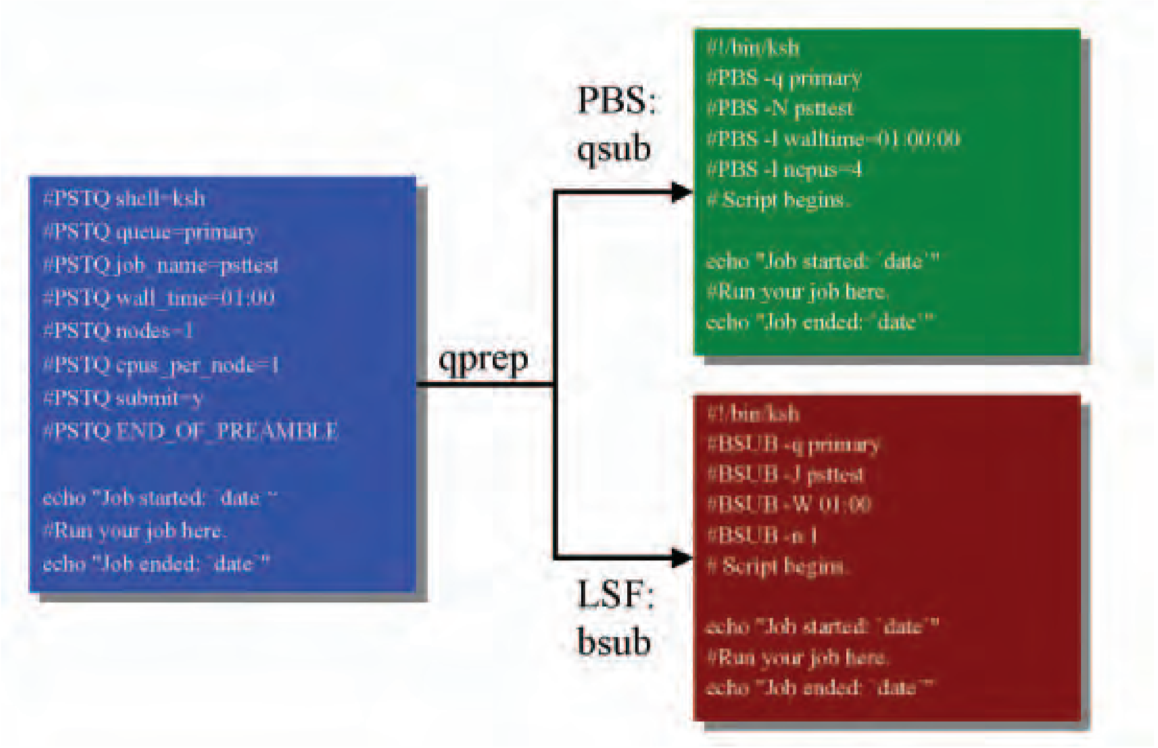
- *qprep*: provides users with a consistent set of commands to submit jobs to the local queuing system. This command translates a site-independent batch queuing script language into the site-specific batch queuing script. Figure 2 shows an example where a single, general batch script of PST directives can be submitted via the *qprep* command for both PBS and LSF. Wherever the PST is installed, this generic batch script can be submitted to the local batch queuing system using the *qprep* command. The syntax for this command is modeled after the typical way in which batch scripts are submitted

  $ qprep *batchscript*

The man pages for both *qprep* and *archive* contain much more information about the specifics of how to effectively use these commands.

History and Current Status of the PST

The PST came about as a solution to users' concerns about the disparity of commands across the Program and even within a single site. The development of the toolkit was originally funded as an HPCMP Corporate Initiative with the original core development team of Drs. Joseph Werne, Michael Gourlay, Chris Meyer, and Chris Bison at the Northwest Research Associates,



```
#PSTQ shell=ksh
#PSTQ queue=primary
#PSTQ job_name=psttest
#PSTQ wall_time=01:00
#PSTQ nodes=1
#PSTQ cpus_per_node=1
#PSTQ submit=y
#PSTQ END_OF_PREAMBLE

echo "Job started: `date`"
#Run your job here.
echo "Job ended: `date`"
```

```
#!/bin/ksh
#PBS -q primary
#PBS -N psttest
#PBS -l walltime=01:00:00
#PBS -l ncpus=4
# Script begins.

echo "Job started: `date`"
#Run your job here.
echo "Job ended: `date`"
```

```
#!/bin/ksh
#BSUB -q primary
#BSUB -J psttest
#BSUB -W 01:00
#BSUB -n 1
# Script begins.

echo "Job started: `date`"
#Run your job here.
echo "Job ended: `date`"
```

Figure 2. The qprep command may be used to translate a single generic batch queue script into PBS- or LSF-specific batch queue scripts. Note that the PBS script has requested 4 CPUs, the smallest number of allocatable CPUs under cpusets.

Inc., and the Colorado Research Associates Division. With the assistance of Dr. Aram Kevorkian, chairman of the Metacomputing Working Group, the development team has grown to include staff throughout the HPCMP. The staff members at the various sites are playing important roles by writing local configuration modules, beta testing, and, in some cases, even extending the toolkit. Through the hard work of many individuals, the toolkit is currently available for use on all machines at the four MSRCs, as well as the Arctic Region Supercomputing Center, the Maui High Performance Computing Center, and the Army High Performance Computing Resource Center.

Work continues on the PST with plans for a variety of extensions and additions. For example, definitions for machine-specific ways of issuing parallel jobs, i.e., standardizing mpirun, mpprun, and prun commands, are necessary to truly have a machine-independent batch queue script. Furthermore, methods for defining home directories and working directories are also needed. For the status of these extensions and other plans, check the PST Web site.

For More Information

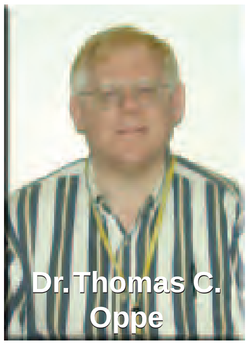
For more information or if you are interested in contributing to the toolkit, go to the Web sites listed in Table 2. The toolkit is written in Perl and is an open source product. Users who have developed solutions to general and unique problems encountered while using supercomputers are encouraged to contribute to the toolkit.

Table 2. Web Sites Containing Information About the PST

Web Site URL	Description
PSToolkit www.pstoolkit.org	Main page for the toolkit including downloads, documentation, mailing lists, etc.
qprep Quick Start Guide www.erdc.hpc.mil/faq/qsg/qrg/qprepqrg.pdf	The ERDC MSRC has developed a qprep quick reference guide.
Batch Script Generator www.erdc.hpc.mil/cgi-bin/batchwiz.cgi	For the HPCs at the ERDC MSRC, this cgi script will help users create a sample batch script for a given machine. Both the qprep and machine-specific scripts are created for the user.

Dual-Level Parallelism in High-Performance Computing

By Dr. Thomas C. Oppe



For several years, standards have existed for expressing parallelism in programs intended for HPC platforms: Message Passing Interface (MPI) and Parallel Virtual Machine (PVM) for passing explicit messages between independent UNIX processes, and OpenMP and pthreads for coordinating parallel work among threads within a single process. These parallel programming tools have arisen in the context of two different computer architectures in which memory is either distributed among the processors or shared by all the processors. In distributed memory architectures (see Figure 1), each processor, or CPU, has access to a modest amount of memory private to that processor. The CPU/memory modules are then connected to each other in a high speed network that has a particular topology. In distributed memory architectures, it is necessary to send messages between the memories of the individual processors in order to coordinate the parallel work such as sending portions of local arrays needed by other processors or synchronizing the processes to form a global sum from partial sums. The ERDC MSRC Cray T3E is an example of a distributed memory architecture having heterogeneous processing elements with differing amounts of memory and differing chip speeds. MPI, PVM, and SHMEM (a library for doing one-sided messaging) are all available on the Cray T3E.

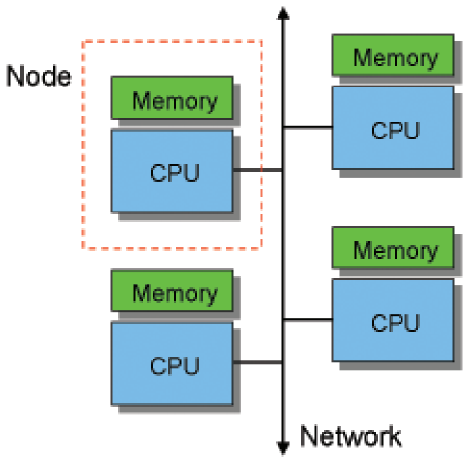


Figure 1. Distributed Memory Architecture

In shared memory architectures, on the other hand, all CPUs can access all the memory on the machine (see Figure 2). If all the memory is as easily accessible from any processor, the machine exhibits uniform memory access (UMA) and is called a symmetric multiprocessor (SMP). If some memory is "local" to the processor in some sense and much more quickly accessible than "remote" memory, then the machine exhibits nonuniform memory access (NUMA). The ERDC MSRC Origin 3800 (O3K) is an example of a NUMA architecture in which the CPUs are grouped together in nodes having four CPUs that share a local memory. Memory on other nodes is accessible from any node, but there is a performance penalty or latency involved in using this "remote" memory. Parallel efficiency of a program will suffer when the CPUs performing a computation become distant from the memory involved in the computation. Tools for doing shared memory parallel programming such as OpenMP and pthreads are available on the O3K, as well as MPI, PVM, and SHMEM.

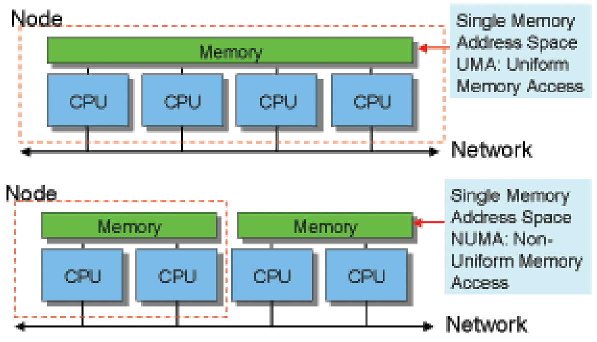

Node
Memory
CPU
CPU
CPU
CPU
Single Memory Address Space UMA: Uniform Memory Access
Network
Node
Memory
Memory
CPU
CPU
CPU
CPU
Single Memory Address Space NUMA: Non-Uniform Memory Access
Network

Figure 2. Shared Memory Architecture

In recent years, a new type of architecture has been developed that is a hybrid of distributed and shared memory architectures. This architecture, called Distributed-Shared Memory (DSM), consists of nodes that are locally SMP machines but have no memory that is shared among the nodes (see Figure 3). Thus message-passing tools such as MPI and PVM are used to coordinate computation between nodes, but shared memory tools such as OpenMP and pthreads can be used within each node. This architecture easily lends itself to a hybrid style of parallel computing that can exploit the lower latencies of lightweight thread processes within the node, but still uses MPI or PVM processes to communicate between nodes. Of course, MPI can still be used within a node, and many vendors supply MPI versions that have been optimized for intra-node communications. The ERDC MSRC Compaq SC40 and SC45 platforms are examples of DSM architectures in which each node comprises four processors that share a local memory.

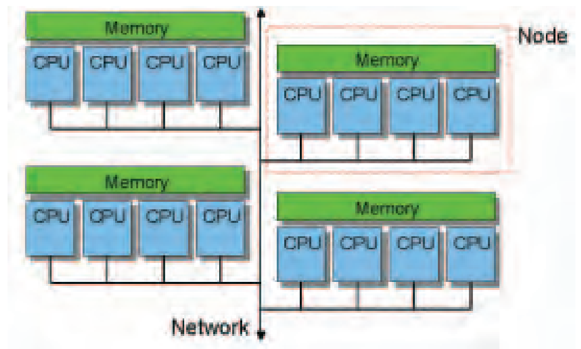


Figure 3. Distributed-Shared Memory Architecture

Many researchers have developed dual-level parallel programs that use MPI on the "outside" for parallelizing coarse-grain units of work and then use OpenMP to parallelize fine-grain computations such as loops on the "inside" (i.e., within each MPI process). One example of such a strategy for a modeling program is to divide the problem domain into subdomains using the techniques of domain decomposition or substructuring, and then assign each subdomain to an MPI process. As the number of MPI processes increases, the subdomains become smaller and the subdomain work and memory requirements of the MPI executable decrease. In addition, the number of messages to be sent increases but the size of the messages decreases. Since the time "t" to send a message of length "m" involves a startup latency "s", as in

$$t = s + c*m$$

where "c" is a constant, there will come a point when increasing the number of MPI processes will not decrease the elapsed running time of the application because of the latency costs of sending many short messages. In other words, the latency costs become significant in relation to the time needed to accomplish the subdomain computations. At this point, further reductions in the elapsed running time may be possible by parallelizing subdomain computations using OpenMP or pthreads, which do not involve sending messages and generally have lower latencies for thread creation and termination.

Some programs are "embarrassingly parallel" at the input data level. Such "task level parallelism" programs do independent calculations for each item of the input data set with little or no need to coordinate the calculations. MPI is often used to divide the independent tasks between the processors. This division of labor can be done in a static manner if the computations for each task take the same amount of time. If each task takes a different or unpredictable amount of time, the task assignment can be done dynamically at runtime using a "boss-worker" algorithm in which a "boss" MPI process assigns the next available unit of work to a "worker" process that has completed its previously assigned work. These "task-level parallelism" programs generally do little MPI communication and thus scale well because of low latencies. However, the MPI parallel speedup is limited to the number of independent items in the data set, which may be small. Also, since each MPI process is doing similar work, the size (memory requirement) of each MPI process does not decrease as the number of processes increases. Hence if, for example, a node has 4 GB total memory, and each MPI process requires more than 1 GB memory, it will not be possible to run four MPI executables on the node without the executables swapping in and out with each other. In either of these cases, further speedup can be obtained by placing fewer MPI processes on the node and using the remaining processors on the node to parallelize the "worker" computations using OpenMP. Typically, using OpenMP compiler directives to parallelize loops does not significantly increase the size of the MPI executable, making it possible to fully utilize the node resources without oversubscribing memory.

Given that a CPU is available, the question naturally arises of whether to use it for MPI or OpenMP work. The obvious choice is to use the parallelization strategy that achieves the greatest parallel speedup. However, there can be situations in which a dual-level approach is preferable to an MPI-only or to an OpenMP-only strategy. OpenMP parallel scalability is obviously limited to the number of processors on a shared-memory node (four in the case of the Compaq platforms). Even in the case of a large shared-memory machine such as the O3K, good OpenMP scalability requires the placement of threads so that the memory accessed by each thread is local to or near the processor running the thread. This may require a careful "first touch" programming style in which array portions to be accessed by particular threads are initialized by those threads.

On the Compaq platforms and the O3K (with the use of cpusets), MPI processes cannot become distant from corresponding memory, but MPI scalability may still be limited by several factors. There may be contention for memory or disk access on the node, or there may be resource contention between the nodes, such as caused by communication bandwidth limitations. Under certain accounting policies, it may be less expensive to use threads than MPI processes. Under other accounting policies, a user may be charged for the use of the full node, even if not all CPUs are being used. In this case, unused CPUs can be devoted to OpenMP work if no more MPI processes can be placed on the node. The convenience of adding parallelism incrementally to an existing MPI code by inserting OpenMP directives allows different parallelization strategies to be explored and adds flexibility to the application. Finally, an MPI program can be made into a de facto dual-level parallel code if calls are made to a threaded vendor-supplied scientific library. Many vendors supply threaded versions of the BLAS or LAPACK libraries, as well as threaded FFT routines and sparse linear system solvers, thus allowing additional speedups when running on the vendor platform.

In early August, Ms. Carrie Mahood and the author gave a tutorial on "Dual-Level Parallelism using MPI and OpenMP" at the Workshop on OpenMP Applications and Tools at the Arctic Region Supercomputing Center (ARSC) at The University of Alaska at Fairbanks, and they plan to deliver a similar tutorial in November at the Supercomputing 2002 conference in Baltimore, MD. In the ARSC tutorial, they described the motivation for using dual-level parallelism and computer architectures where such techniques are applicable, the differences between UNIX processes and threads, the basic OpenMP compiler directives, and several application case studies in which dual-level techniques have been implemented by the ERDC MSRC staff. These applications include the CGWAVE and STWAVE applications from the ERDC Coastal and Hydraulics Laboratory and the SARA-3D program from BBN Technologies for modeling problems in structural acoustics. An introduction to Multi-Level Parallelism (MLP), a new tool developed by Dr. James Taft of NASA Ames, was also presented. MLP combines UNIX forks with OpenMP threads on large shared memory platforms, such as the O3K, to address load-balancing issues more conveniently than the MPI/OpenMP strategy. The afternoon session of the tutorial was devoted to a hands-on laboratory for the attendees to experiment with applying several parallelization techniques to sample test codes.

An Evolving Benchmarking Strategy

By Carrie Mahood

Each year, the HPCMP uses a benchmark suite to assist in the selection of new HPC equipment. The goal is to use a suite of application codes that accurately reflects the total HPCMP workload. However, past efforts to accomplish this goal have resulted in many difficulties, for the suite creators and vendors alike, such as portability of the codes to different platforms, optimization, code distribution, etc. This year, the HPCMP has strived to reduce the complexity of this process.

There are three types of benchmark codes in the Technology Insertion 2003 (TI-03) benchmark suite: synthetic codes, application codes, and the newly added synthetic application codes. Synthetic codes are basic hardware and system performance tests aimed at calculating future performance of a system. Application codes are actual programs run on HPCMP machines, and are chosen by usage and CTA requirements. Though the name implies this type is a combination of the previous two, synthetic application codes are actually simplified versions of the application codes. It is hoped that synthetic application codes will replace application codes in the future.

The addition of the synthetic application codes to the benchmark suite is one of the ways the HPCMP is trying to simplify the benchmarking strategy. A further change is the combination of the vector and memory synthetic benchmarks. Finally, the application mixes have been removed, as the responsibility of obtaining a good scheduler has been passed to the usability team. This article will focus on the creation of the synthetic application codes.

There are many motivations behind synthetic applications. Using synthetic modules (discussed later) to model any application will reduce the manpower, and hence the budget, needed for benchmarking. Also, while classified codes are part of the HPCMP workload, they are obviously not permitted to be included in the benchmark suite, while their synthetic counterparts would be. There are difficulties in distributing the codes to vendors, such as very large programs, large input data files, and licensing, which the comparatively small synthetic application codes would not have. Finally, synthetic applications should be simple enough to resolve vendors' porting difficulties.

Two application codes, GAMESS and NLOM, were chosen for the creation of synthetic application codes. The Computational Science and Engineering (CS&E) group at the ERDC MSRC worked with NLOM, creating synNLOM. SynGAMESS was created by Instrumental, Inc.

The initial plan for this process of creating a synthetic application was to create several modules, which may be thought of as building blocks, of various types such as computation, I/O, communication, memory references, etc. The goal is to create a suite of parameterized modules that are generic enough to be assembled with specific parameters into a program that mimics any chosen application. This year, the CS&E group chose three module types: computation, I/O, and communication.

To create a synthetic application of NLOM, the first step was to profile NLOM. The actual NLOM code was "cleaned up" – all the machine-specifics and "if-defs" were removed, creating a "common" version of NLOM. Profiling NLOM was accomplished using several tools. Performance Bench was used to calculate (per subroutine) flops, total MPI bytes passed, and total MPI messages passed. Created by Instrumental, Inc., it uses Performance Application Programming Interface (PAPI) calls to obtain information generated by hardware counters during program execution. SGI's SSRUN and Perfex are tools that also give information such as flops and were run on the ERDC MSRC's Origin 3800 (O3K). Vampir Trace, also run on the O3K, gave a complete trace of NLOM, including subroutine calls, message-passing paths, sizes of messages, etc. Lastly, even good old-fashioned print statements were used to find patterns of subroutine calls and a path through the code.

One module of each of the three chosen types was created for this synthetic application. An I/O module was designed to read in two 1.2-GB files (akin to NLOM's fort.58 and fort.59) and write out a 1.3-GB output file. For the communication module, the HALO benchmark, which uses the same pattern of message passing as NLOM, was adapted. The eight most computationally intense routines in NLOM were selected, adapted, and combined into a computational module.

The last step, assembling the modules with appropriate input parameters into a main program, was then accomplished. Please see Figure 1 for a comparison of timing results for synNLOM and NLOM on multiple platforms.

The creation of synNLOM is a significant step toward the goal of using synthetic application codes instead of actual application codes. Though problems were encountered in the process, such as bugs in the HALO benchmark, time constraints, and overflow/underflow errors because the computational routines were basically calculating "nonsense," a great deal of knowledge was obtained that will make the following year's effort for a realistic yet simple benchmark suite even more of a reality.

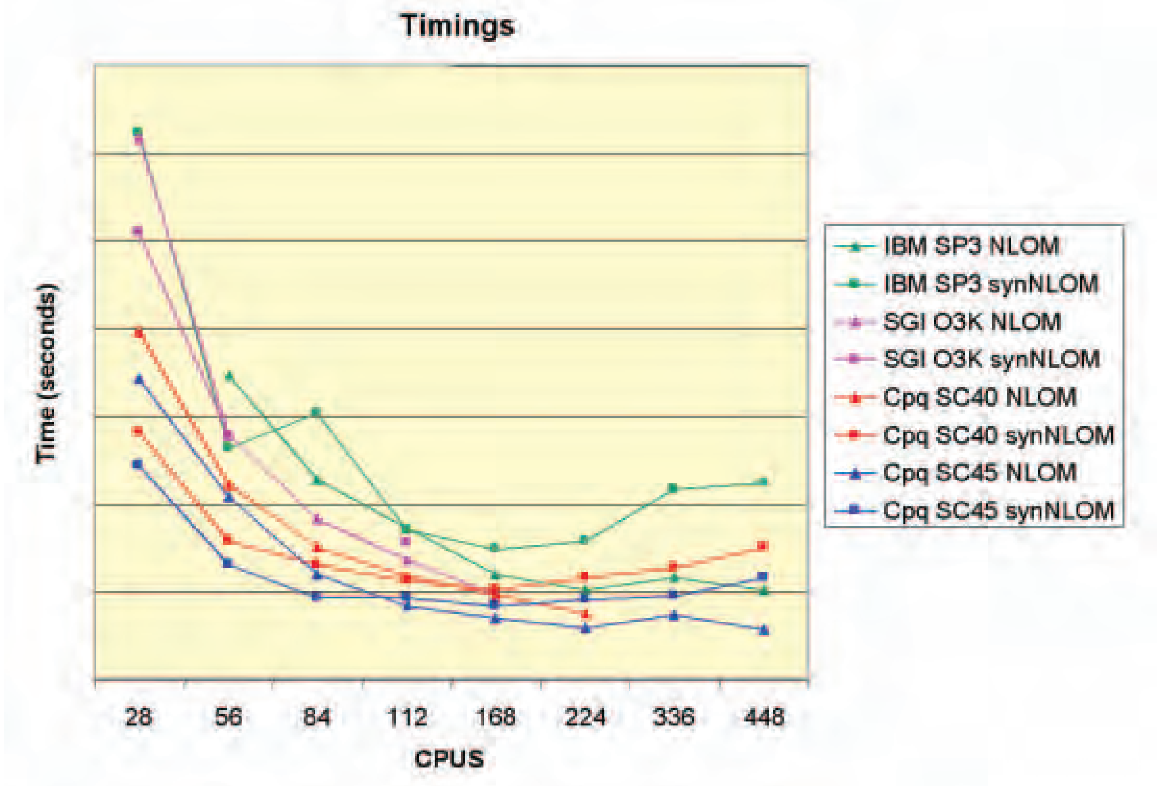


Figure 1. Runtime Comparison of synNLOM and NLOM

The ERDC MSRC welcomes comments and suggestions regarding *The Resource* and invites article submissions. Please send submissions to the following e-mail address:

msrchelp@erdc.hpc.mil

Retrospective –The 2002 DoD Users Group Conference

By David Stinson, Users Group 2002 Facilities Chair

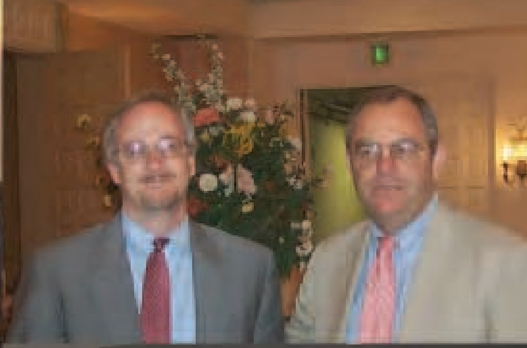
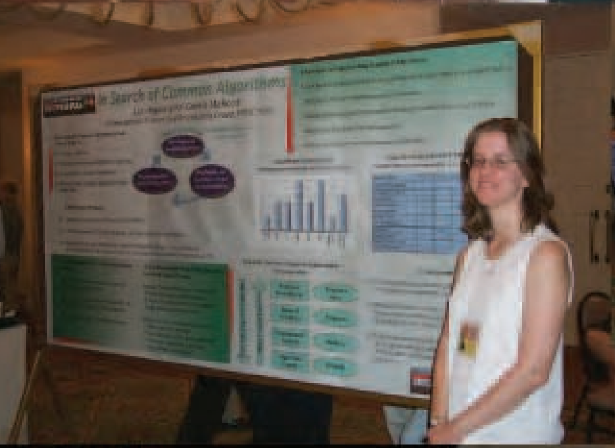
Nearly 400 attendees turned out for the 2002 DoD Users Group Conference in Austin, Texas. Almost everyone agreed that the food during the week was excellent, thanks to the DoubleTree catering staff and Catering by Rosemary. As in previous years, this conference provided an excellent opportunity to renew acquaintances and collaborate with others in the HPC Modernization Program.

About 170 attendees participated in the tutorials on Monday. The most popular tutorial, with about 50 attendees, proved to be the Comprehensive Introduction to Scientific Visualization jointly taught by the University of Texas, Texas Advanced Computing Center, and the ERDC MSRC. This was followed by the DoD Grid Computing Using Kerberized Globus tutorial also taught by the ERDC MSRC with about 30 in attendance. Many thanks to Steve Schraml for the excellent job he did as the Tutorials Chair.

On Tuesday, Mr. Steve Scherr, the General Chair, kicked off the day by welcoming everyone and introducing Dr. Charles Holland, Deputy Under Secretary of Defense, as the keynote speaker followed by the HPCMP Director, Cray Henry. Invited speaker, Mr. Robert Graybill, Program Manager for the Information Processing Technology Office, Defense Advanced Research Projects Agency, rounded out the morning. After an afternoon of technical papers, a record number turned out for the poster presentation that evening.

Wednesday, Steve Finn, the incoming chairman of the Users Advocacy Group and the conference Technical Program Chair welcomed attendees. Invited speakers on the morning agenda included Conrad Clyburn, U.S. Army Medical Research and Materiel Command, Dr. William Feiereisen, Department of Energy, Dr. Kyle Squires and Major Jim Forsythe from Arizona State University and the U.S. Air Force Academy, respectively, and Dr. Joseph Baum from SAIC. The evening social was held at the Texas State History Museum. A good crowd turned out to see the IMAX presentation "The International Space Station" and enjoy the Fajita buffet. Jeanie Osburn and Lynn Parnell warmed up the dance floor to the sounds of The Daddios, a local rock and roll band. Others took advantage of the three floors of exhibits, including a new one on Davey Crockett and visited the Texas Spirit Theater, a special effects theater where one could experience hurricanes and plagues of varmints.

Going on all day Thursday were four concurrent tracks of technical papers. By Thursday evening, many were ready to get out on the town and took the opportunity to visit some of the fine restaurants and hear some of the music in downtown Austin. Others took time to see and experience some of the local attractions such as the LBJ Presidential Library, watching bats at the Congress Street Bridge, and enjoying jet skiing at nearby Lake Travis. The conference, by all accounts, was a great success, and we are looking forward to next years conference in Bellevue, Washington.



In Search of Common Algorithms

Parallel MATLAB

Users Group Conference 2002

ERDC MSRC Team Member "Accepts the Challenge" of eCYBERMISSION Through the Army Ambassador Program

By Rose J. Dykes

David Stinson is one of five Army Ambassadors appointed from ERDC to represent the "Face of the Army" through the Army Ambassador Program. This program is an essential part of eCYBERMISSION—the Army's new venture with a Web-based science and math competition for middle school students that was launched this fall.

The Army decided to sponsor this unique science, math, and technology competition via the Internet because of decreased student interest throughout the Nation in science, math, and technology careers, thereby causing a human resource shortage in the future science and engineering workforce. Regarding eCYBERMISSION, the Chief of Staff, United States Army, Memorandum of April 27, 2001, Subject: A Science Fair for the Nation, directs as follows: "The Army will offer its resources, infrastructure, and personnel for logistic and administrative support, and its soldiers and laboratory civilians as mentors to students involved in the competitions…The initiative will support the President's commitment to education while enabling the Army to return something to American communities for the service of their children to the Nation."

In addition to his busy position as part of the team leading the ERDC MSRC Customer Assistance Center, David serves on several HPCMP Working Groups—represents the ERDC MSRC on the Users Advocacy Group; serves as a member of the Requirements Working Group (a group that looks at utilization metrics for machines across the Program to help in the selection of future HPC acquisitions); and has been a part of the System Administration Working Group, which is designed to help standardize user interfaces across the Program. With his civil engineering background, work experience, and strong desire for mentoring others, David will be an exceptional role model as he serves as an eCYBERMISSION Ambassador. He will "visit local schools to generate enthusiasm among teachers and students for the competition, promote eCYBERMISSION at key conferences and meetings, such as PTA meetings and Boy and Girl Scouts, present regional awards to students and participate in awards ceremonies, and gather and report feedback received." The time commitment for each Ambassador is 15 to 20 hours per month.

David Stinson (left), ERDC MSRC, discusses eCYBERMISSION with Ray Hume, Vicksburg Junior High School Principal

Computer Career Job Shadowing

By Rose J. Dykes

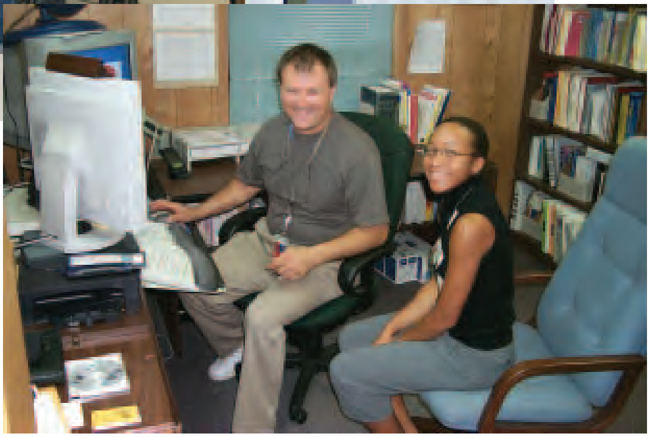
Students from Vicksburg High School participated in job shadowing at ERDC on May 2. The ERDC Information Technology Laboratory hosted 11 students who had specified computer careers.

David Stinson planned and coordinated the activities in the MSRC part of ITL for all 11 students. They visited the MSRC Scientific Visualization Center and viewed a three-dimensional glass-breaking demonstration. Also on the agenda was a tour of the ITL Joint Computing Facility.

One student, Stephanie Warren, had chosen the computer engineer field to shadow. David arranged for her to spend time with two members of the MSRC who served as mentors and talked with her about what their jobs involved and the prospects in their fields.

Lee Higbie, ERDC MSRC, and Stephanie Warren talked about implementing common algorithms

Computer career job shadowing students toured the Joint Computing Facility, along with Brantley Jones and Charlotte Glass (2nd and 3rd from left), ERDC ITL


Jerry Morris, ERDC MSRC, and Stephanie Warren discussed grid computing

Computer Sciences Corporation Invests in Mississippi Youth Programs

By Ginny Miller

Four Mississippi youth programs are getting a boost from Computer Sciences Corporation (CSC), the lead contract integrator for the ERDC MSRC, with financial gifts totaling nearly $4,000.

"It's a great way, especially in the post-9/11 environment, to rededicate ourselves to our children's future," said Doug Walker, ERDC MSRC Program Manager. "CSC is proud to be involved in enriching the lives of these young people. We also take pride in serving the DoD as partners with the ERDC MSRC."

CSC's monetary donations include $2,000 to the New Hebron Attendance Center library; $1,000 to a high school robotics team; $500 to the Red Carpet Bowl Classic, which in turn funds college scholarships; and $250 to a recreational athletic team.

"This contribution is very important to our school," said Susan Cliburn, library assistant at New Hebron Attendance Center, a public school serving grades K-8 in a town of nearly 400 people. "We are so desperate for books. You can never have too many."

Cliburn said the school would use the money it received from CSC to purchase books for the Accelerated Reader (AR) program. AR combines the use of great children's literature, with points assigned to different reading levels, with computer software to test comprehension. Students read the books at their own pace and are awarded prizes when their reading goals are reached.

Through AR, "We want the children to learn that reading is fun, that it helps your mind grow, that it can be your hobby and your best friend," Cliburn said. "We are so thankful to CSC for helping us accomplish this."

Travis Wayne Vance, a CSC employee working at the ERDC MSRC, is also grateful for CSC's contribution to the Red Carpet Bowl Classic. "The company is supporting local high school students who might not be able to afford college," Vance said, explaining that the bowl, established in 1962, awards scholarships each spring to seniors from St. Aloysius, Vicksburg, and Warren Central high schools. "That's one of the reasons I am proud to be a member of CSC's ERDC MSRC team."

CSC, one of the world's leading consulting and information technology services firms, also funded the purchase of monogrammed bat bags for the Vicksburg Barracudas, a baseball team of 12-year-olds, and assisted the robotics team that includes students from Vicksburg and Warren Central high schools.

Dr. Flavol Rester (left), principal of New Hebron Attendance Center, accepts a check for $2,000 from James Cliburn (right) of Computer Sciences Corporation. Looking on is Library Assistant Susan Cliburn (center)

Library Assistant Susan Cliburn reads to a group of first-graders at New Hebron Attendance Center. The school library recently received $2,000 from Computer Sciences Corporation

Greg Rottman, ERDC MSRC, and Department of the Army Intern Annabele Rosado in the Joint Computing Facility, August 21

John West (left), ERDC MSRC Director, with U.S. Senate Energy and Water Subcommittee Staffers Tammy Perrin, Roger Cockrell, and Jamee Plockmeyer (center), August 15. Dr. Jeffery Holland, ERDC ITL Director, and Dr. James Houston, ERDC Director are shown in background

(Left to right) John West, Dr. Michael J. O'Connor, Director of Research and Development, U.S. Army Corps of Engineers, and Timothy Ables, Assistant to the ERDC Director, August 2

COL Joseph Schroedel (left), Chief of Staff, Headquarters, U.S. Army Corps of Engineers, and David Stinson, ERDC MSRC, July 16

Former Mississippi Senator Grey Ferris (far left) along with family members including Vicksburg surgeon Dr. Gene Ferris (third from left) listen to Brad Comes, ERDC MSRC, present the history of HPC at ERDC, July 12

Members of the 326th Engineer Battalion, Fort Campbell, Kentucky, July 11

COL Ernie Huse (left), Senior Advisor for Reserve Affairs, and CPT Chris Boyd (right), Office of Reserve Affairs, U.S. Army Corps of Engineers, with David Stinson, July 10

COL (Ret.) Wayne Reynolds (right), Eastern Kentucky University, with Tom Biddlecome (left) and Paul Adams (center), ERDC MSRC Scientific Visualization Center, July 8

Members of Lower Mississippi River Forecasting Center, Slidell, Louisiana, with David Stinson, June 19

acronyms

Below is a list of acronyms commonly used among the DoD HPC community. You will find these acronyms throughout the articles in this newsletter.

ACT	Application Cookbook Tools
AFRL	Air Force Research Laboratory
AMR	Automatic Mesh Refinement
AR	Accelerated Reader
ARA	Applied Research Associates
ARSC	Arctic Region Supercomputing Center
cc-NUMA	Cache Coherent Non-Uniform Memory Access
CFD	Computational Fluid Dynamics
CHSSI	Common High Performance Scalable Software Initiative
CPU	Central Processing Unit
CSC	Computer Sciences Corporation
CS&E	Computational Science and Engineering
CSM	Computational Structural Mechanics
CTA	Computational Technology Area
CWO	Climate/Weather/Ocean Modeling and Simulation
DoD	Department of Defense
DSM	Distributed-Shared Memory
DTRA	Defense Threat Reduction Agency
EQM	Environmental Quality Modeling and Simulation
ERDC	Engineer Research and Development Center
HBCU	Historically Black Colleges and Universities
HE	High Explosive
HPC	High-Performance Computing
HPCMP	HPC Modernization Program
I/O	Input/Output
ITL	Information Technology Laboratory
JSU	Jackson State University
LANL	Los Alamos National Laboratories
LB/TS	Large Blast and Thermal Simulator
LSF	Load Sharing Facility
MD	Management Documentation
MLP	Multi-Level Parallelism
MPI	Message-Passing Interface
MSRC	Major Shared Resource Center
MSU	Mississippi State University
NASA	National Aeronautics and Space Administration
NE	Nuclear Explosive
NIAB	Non-Ideal Air Blast
NUMA	Nonuniform Memory Access
PAPI	Performance Application Programming Interface
PE	Processing Element
PET	Programming Environment and Training
PST	Practical Supercomputing Toolkit
PVM	Parallel Virtual Machine
SET	Source Editing Tools
SHAMRC	Second-Order Hydrodynamic Automatic Mesh Refinement Code
SAIC	Science Applications International Corporation
SMP	Symmetric Multiprocessor
SVC	Scientific Visualization Center
TICAM	Texas Institute for Computational and Applied Mathematics
TI-03	Technology Insertion 2003
TUSC	Tools for the Unification of SuperComputing
UAB	University of Alabama at Birmingham
UCLI	Uniform Command Line Interface
UMA	Uniform Memory Access
UT	University of Texas

training schedule

For the latest on PET training and on-line registration, please go to the On-Line Knowledge Center Web site:

https://tin2.wes.army.mil/okc/portal

Questions and comments may be directed to PET training at (601) 634-3131, (601) 634-4024, or *PET-Training@erdc.usace.army.mil*

ERDC MSRC Newsletter Editorial Staff

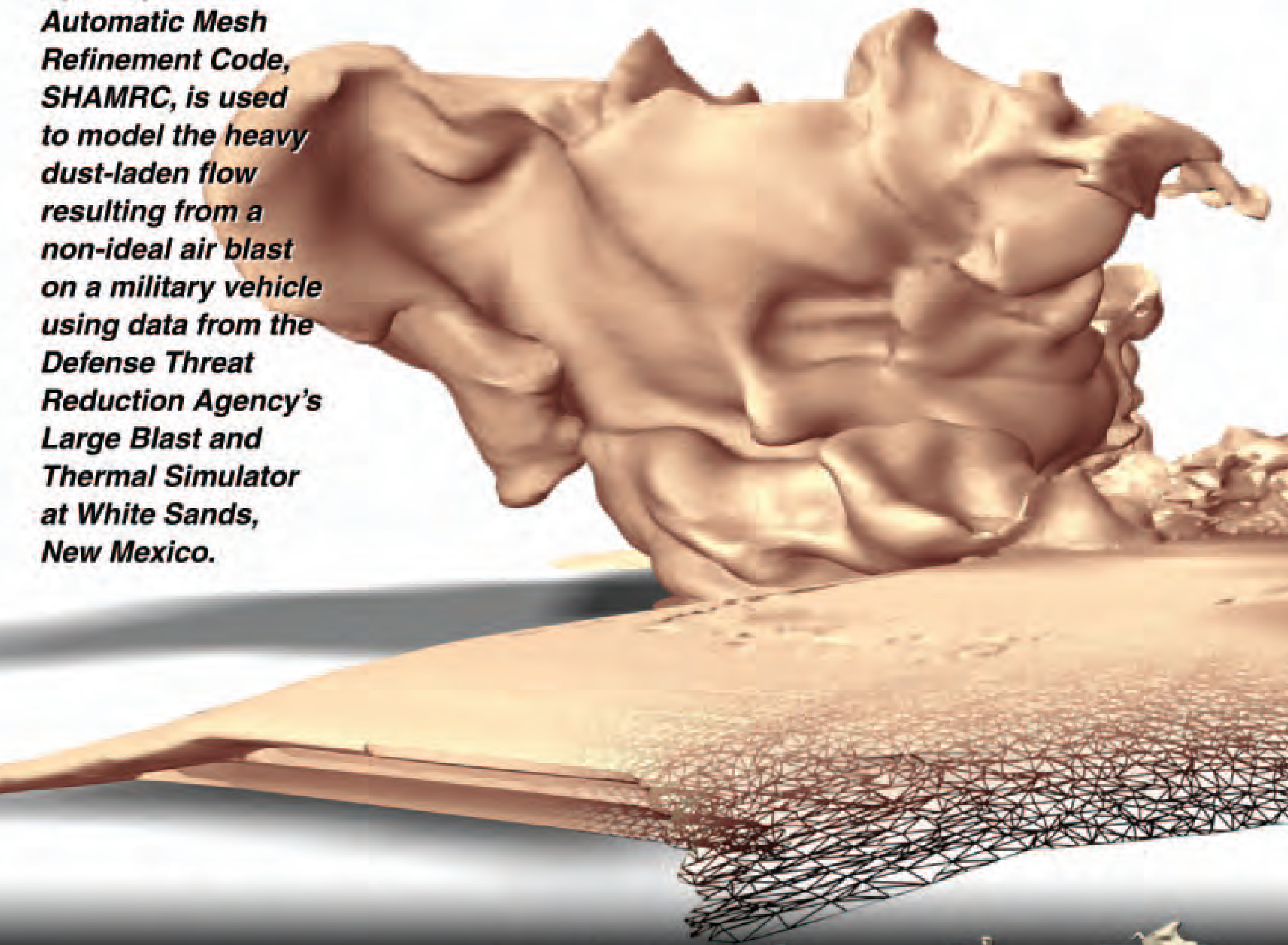
Interdisciplinary Coordinator
Mary L. Hampton

Outreach Coordinator
Rose J. Dykes

Chief Editors
Rose J. Dykes
Virginia “Ginny” Miller

Visual Information Specialist
Betty Watson

ERDC MSRC Customer Assistance Center
Web site: *www.erdc.hpc.mil*
E-mail: *msrchelp@erdc.hpc.mil*
Telephone: 1-800-500-4722



Second-Order Hydrodynamic Automatic Mesh Refinement Code, SHAMRC, is used to model the heavy dust-laden flow resulting from a non-ideal air blast on a military vehicle using data from the Defense Threat Reduction Agency's Large Blast and Thermal Simulator at White Sands, New Mexico.