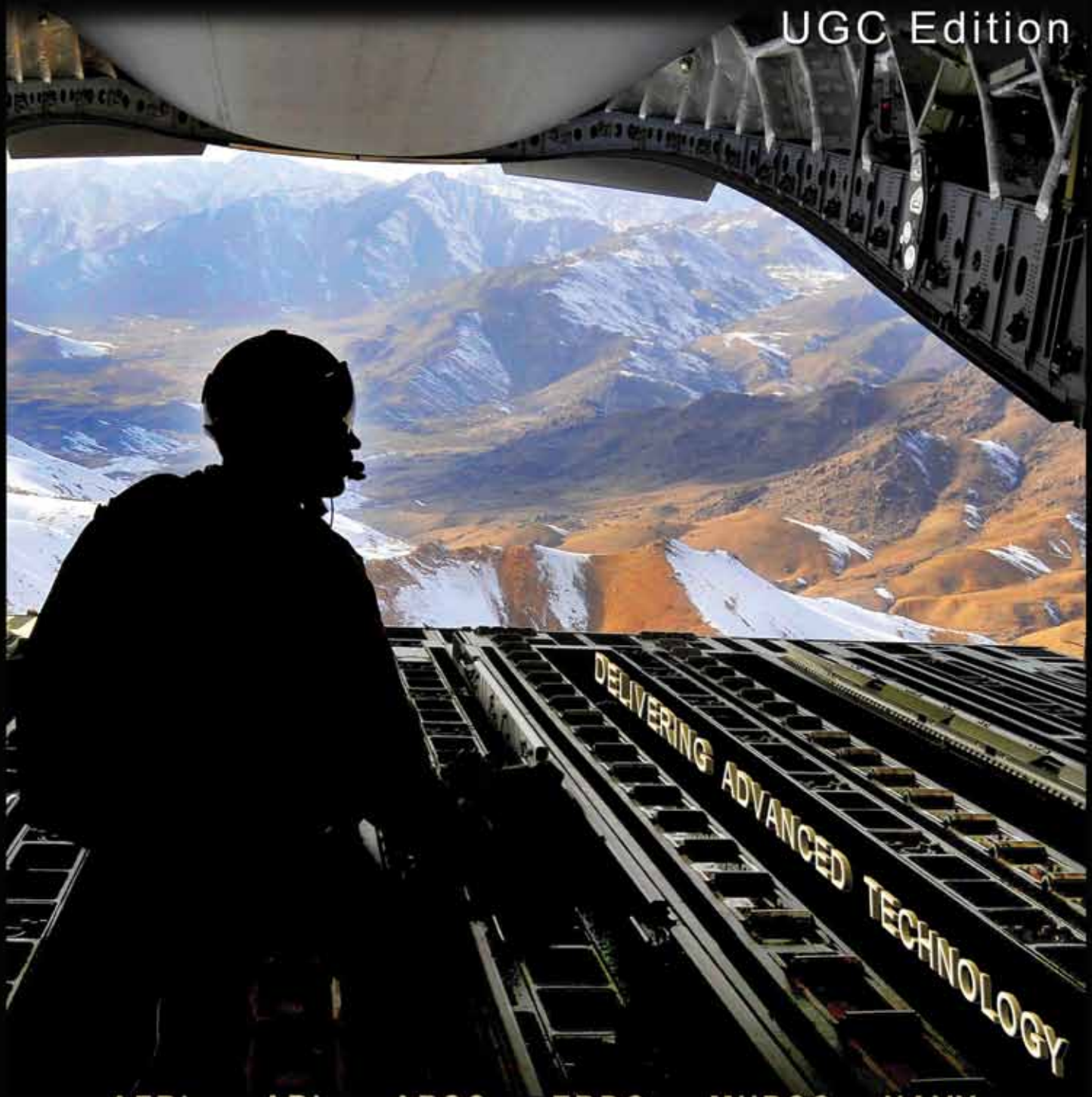
HPCinsights

DoD Supercomputing Resource Centers

Spring 2010

UGC Edition



AFRL • ARL • ARSC • ERDC • MHPCC • NAVY

SUPERCOMPUTING FOR THE WARFIGHTER

HPC Insights is a semiannual publication of the Department of Defense Supercomputing Resource Centers under the auspices of the High Performance Computing Modernization Program.

Publication Team

AFRL DSRC, Wright-Patterson Air Force Base, OH
Joan Henley
Chuck Abruzzino

ARL DSRC, Aberdeen Proving Ground, MD
Debbie Thompson
Brian Simmonds

ARSC DSRC, Fairbanks, AK
Debra Damron
Mary Haley

ERDC DSRC, Vicksburg, MS
Rose J. Dykes

MHPCC DSRC, Maui, HI
Jeff Schmidt
Betty Duncan

Navy DSRC, Stennis Space Center, MS
Christine Cuicchi
Lynn Yott

HPCMPO, Lorton, VA
Deborah Schwartz
Sheryl Caramanzana
Leah Glick

MANAGING EDITOR
Rose J. Dykes, ERDC DSRC

DESIGN/LAYOUT
Betty Watson, ACE-IT

COVER DESIGN
Chandra "Pat" Caldwell, ACE-IT

Contents

About the Cover: The picture on the front shows a U.S. Air Force Tech. Sgt. sitting on the ramp of a C-17 Globemaster III aircraft while flying over the mountains of Afghanistan after an airdrop mission in February 2010. *U.S. Air Force photo by Staff Sgt. Angelita Lawrence*

DoD Supercomputing Resource Centers
Making the Supercomputer Wait on the Scientist

By Brad Comes, HPC Centers Project Manager

Brad Comes
HPC Centers Project Manager

Hindsight is 20/20, and we won't know for sure until we can look back; but we believe the High Performance Computing Modernization Program (HPCMP) is at a tipping point. Historically, our challenge has been to keep pace with the computational demands of scientists and engineers in the Department of Defense (DoD). Fundamentally, this meant acquiring and deploying the latest commercially available high performance computers. Our computational capabilities have nearly doubled every year. Here's where the tipping point comes in. Recent forecasts from the HPCMP annual requirements surveys indicate that we're approaching the ability to meet the DoD's high performance computing "compute" requirements. What does this mean? Our first reaction was that we'll reduce the rate of deployment of new supercomputer capabilities. However, we surveyed the HPC user community, and the data revealed a requirement we never thought we would have had the luxury of addressing—to increase the productivity of the scientist versus the productivity of the computer. Our strategy to enhance science and engineer productivity has evolved with two initiatives—improved time-to-solution and seamless workflow.

In concept, improved time-to-solution is achieved by improving the traditional batch job submission environment to something approaching interactive computing. I say "something approaching" because interactive supercomputing is a ways off. But, we need to move in that direction. Doing this requires a change in the way we operate our supercomputers and measure success. Currently, people wait on the very precious supercomputer. We have to create an environment where the computers are waiting on the people. Historically, we have strived for 70-80 percent system utilization. The other 20-30 percent is overhead from downtime events and unavoidable gaps in job scheduling. Anything below 70-80 percent has been perceived as if we were doing something wrong: we didn't have enough users on the system; we had too much system downtime; we weren't efficiently scheduling jobs; and always looming in the wings is the inevitable conclusion—if you're only using 70 percent of what you have, why do you need more? We have to break the multiple decade mentality that we can't afford to let a CPU cycle go unused. In pursuit of significantly increasing the productivity of our highly educated, skilled, and expensive workforce, we need to instead strive for high scientist and engineer utilization rates. Scientists and engineers should not have to wait for the computers; the computers should be waiting for them—likely resulting in HPC system utilization rates at or below the 50th percentile. One service contributing to this goal that you'll find available by the time this is published is the Advance Reservation System (ARS); most all systems in the HPCMP will have an ARS capability for end-users to schedule a predetermined time to start a job. This is a first step toward making the computer wait on the scientist.

Supercomputers are the crown jewel in a computing infrastructure, but they have never been tightly integrated into their surroundings. The functionalities associated with desktop and server room computing exploded while supercomputers remained pretty much batch engines with command-line prompts. This has created gaps in the user's workflow that again significantly contributes to inefficiencies in end-user productivity. The HPCMP is focusing on closing this gap. The most significant near-term initiatives that will contribute toward this are the Storage Initiative (SI) and the Common Utility Enhancement Services (CUES). Services you can expect to see in Fiscal Year 2011 associated with the SI and CUES are 30-day working directories (versus the current 5), an information lifecycle management system for archived data, remote visualization services, and remote job management. Longer term projects include the delivery of interactive grid-generation capabilities, an extensive software development environment, a single sign-on capability, and Web-based portals that connect the desktop to the supercomputer.

The HPCMP's first reaction to our out-year projections of compute requirements versus compute capabilities suggested that we had achieved our mission of "HPC modernization." All we had to do now was retain our edge. However, upon a closer look, we realized that we had another huge opportunity—user productivity. As a result, we are now leveraging lessons learned from our challenges of the past into mature processes required to sustain the core HPC compute capability while we embark on this new challenge—making the supercomputer wait on the scientist.

Air Force Research Laboratory
DoD Supercomputing Resource Center

From the Director's Desk – Frank Witzeman

Frank Witzeman
Director, AFRL DSRC

The Technology Insertion 2010 (TI-10) award, delayed over 2 months, was announced with anticipated excitement within the AFRL DSRC—a new 43,712-core next-generation Cray system will be coming to Wright Patterson Air Force Base this summer! It has been a long time since a Cray system was operational in our facility, and we're looking forward to a renewed, successful partnership with Cray Inc. in providing over 400 teraflops/sec to our users.

I recall that it was quite a challenge in 1990 to exploit the latest high performance computing (HPC) technology to calculate the complex fluid dynamic phenomena over aircraft components (e.g., wing sections). Typical turnaround times were assumed on the order of months to get meaningful results. Ten years later in 2000, reasonable turnaround times of days were being realized for fully turbulent, transonic flow over entire aircraft configurations. For those lucky enough to get most of the HPC system (about 1000 processors at that time), the calculations could be done in a matter of hours. Now in 2010, we have more computational power than we could have imagined 10-20 years ago, and we might expect solutions in a matter of minutes. Despite the rapid advances in HPC technology, however, we are still experiencing a relatively constant time-to-solution for our most demanding computational work.

There exist a number of reasons why we're not accelerating time-to-solution. From one perspective, pre- and postprocessing capabilities have not experienced necessary developments targeted for HPC applications and architectures. Many of these capabilities are best suited for the desktop or workstation environments, so there's no driving force to bring them into the HPC realm. Another issue is related to user applications that do not scale well. The applications themselves consist of the input/output data ("models") and analysis software ("codes") that quickly reach limits in terms of communications overhead where the problem being solved can only be partitioned across a reasonable set of cores or nodes before the computational time becomes swamped by the interconnect communication time. Users prefer to stay below these limits, thereby reducing the size of their applications to perhaps relatively low core counts. A related aspect is that users themselves do not scale well because they can easily become overwhelmed by the amount of data generated from massive computational problems. It has become easier to piece together several applications in order to better manage smaller data sets than one enormous one (and get faster turnaround for each smaller application instead of waiting for one huge one).

More reasons have surfaced, such as the general lack of scalable software/codes, and they all need to be investigated to determine how we will move forward in HPC and supercomputing. Are we satisfied with our current paradigm, that is, to continue to operate increasingly larger systems to accommodate growing workloads of relatively small core-count jobs? Or, should we take a fresh look at how we can encourage and expand large core-count jobs, on the order of tens of thousands of cores? Do we expect a reasonable mix of small (some tiny by today's standards) and large jobs that stretch the imagination of our HPC architectures? For reference, about 92 percent of all jobs on our SGI Altix 9000-core *Hawk* system use less than 100 cores and about 3 percent use 400 cores or more. An expectation 10 years ago might have been that maybe a third of our jobs in 2010 would consist of at least 1000 cores because our HPC systems would be capable of handling the workload.

As I think back to the early days of the HPCMP and think ahead to the possibilities, I am eager to learn more about our users' thoughts on the issues of HPC and supercomputing. I am aware of some anecdotal evidence that there are users eager to run ten thousand 1-core jobs on the new architectures, as well as users ready to scale up single applications to 10,000 cores. We need to be prepared to handle the anticipated full spectrum of jobs, and we can only do that by staying intimately connected to the user community.

Army Research Laboratory
DoD Supercomputing Resource Center

From the Director's Desk – Charles J. Nietubicz

Charles J. Nietubicz
Director, ARL DSRC

Hello again. It always gives me great pleasure to talk to you through *HPC Insights* and briefly mention the exciting activities ongoing at the ARL DSRC and the new directions in which we are heading. The management and staff here are proud to play a part in the successful HPCMP and trust that we are providing you the current, and working on the anticipated, services that you need. Our collective success comes from the hard work, dedication, and vision of many people. The ultimate success is measured in what you, the HPCMP user community, are able to provide to the warfighter in sophisticated weapons development and evaluation, made possible in part through the resources of the HPCMP. However, while we have had much success, we cannot rest with the status quo and need to continually look for how we can modernize, improve, be responsive, and create a future environment that rivals all that has come before now. It was not long ago that an Ethernet-connected workstation (some of you may remember that thick yellow cable running from office to office) interfacing with Cray supercomputers (the HPC standard in the late 80s, early 90s) was considered leading edge, top of the line, king of the road. We have come a long way since then and made great progress in solving some complex problems, but I see the future as being even more exciting. What is that future? I believe we will develop a new HPC world focused on access to HPC by nonexpert users, i.e., those engineers and scientists who need answers, not a new education. How can this happen? Through concepts like HPC cloud computing; HPC from user desktop environments; access via Web and graphical user interfaces; projects like the DRE Portal; next-generation DREN and handheld devices; data storage and retrieval from anywhere, anytime; and knowledge through information and data fusion. Now, I did not say this would be easy. Stating a vision is easy; accomplishing that vision is often hard. As the question is sometimes asked (nonliterally), "How do you eat an elephant?" – the answer comes back, "One bite at a time." So, that is how we can go about achieving our vision for HPC. Some of the "bites" are included in this issue of *HPC Insights*.

In the fall issue, I described some initial efforts we were beginning with the HPCMP Office (HPCMPO) on the development of a transparent access to HPC for Matlab users. This project is called the DRE Portal project, and we can use your help. See the call for participation on page 4; if you see Dr. Pat Collins walking the halls at the User Advocacy Group meeting, please let him know your thoughts on this subject. We are also researching a Microsoft HPC capability. Stop us in the halls at the Users Group Conference to discuss. We would very much like to get your thoughts and ideas in this area.

You may have heard some initial comments about the new HPCMP Storage Initiative originally developed to replace the end-of-life tape library. The initiative has expanded to include much more than its original goal and now includes looking at a standard storage architecture, infrastructure upgrades, and utility servers throughout the HPCMP. A full team from all Centers, representatives from the HPCMPO, and industry and user representatives are involved. Please take a look at the article (page 33) that provides some of the emerging details.

A new and emerging technology is the potential use of accelerators in the HPC arena. Where and how they fit is not exactly known, but we are beginning to investigate. This is not unlike the investigations we undertook in the 2000 time frame with cluster computers. Hybrid systems comprised of multicore and many-core configurations is an interesting technology that needs to be examined and understood. We need to see what works and what does not. Dr. David Richie and others are exploring these questions and have provided a good summary in their article "Investigations of Algorithms for Hybrid Multicore/Many-Core Architectures" found on page 29.

We would encourage you to take a look at the articles I have mentioned and feel free to contact the ARL DSRC with any questions or comments.

DRE Portal

By Steve Thompson, ARL DSRC Sortware Engineer

Would you like to use the power of many processors without needing to learn Linux?

If so, you will be interested to know that the ARL DSRC and the HPCMPO are working on a portal that will bring the power of high performance computing (HPC) to your Desktop. Currently, we are in the pilot phase of development as we work to define possible technologies and specify requirements. Matlab is our initial test application, with other applications to be made available later.

This portal is designed to bring HPC into your environment and, therefore, make HPC easier to use. Our goal is to provide an environment where you can use your familiar desktop interface to run an application, such as Matlab, yet harness the power of multiple processors. This will provide the ability to complete your computations much faster or allow for multiple parametric studies in the same time frame. This portal will be known as the DRE (Defense, Research, and Engineering) Portal.

Does this sound interesting to you? If so, you can help us in two ways:

(1) We need feedback from DoD Matlab users (especially those not currently using the DSRCs) as to how much you would envision using this portal when it becomes available.

(2) We need interested users to help with initial testing and shakeout of the pilot system to ensure that it will be robust enough to handle your computational needs.

Another technology that the ARL DSRC is investigating is the Microsoft (MS) HPC Server. Microsoft has made a significant investment in developing a massively parallel HPC server version of its commercial Operating System. This investment is showing up in many commercial sites for technical applications and cloud configurations (reconfigurable servers). ARL will be standing up hardware resources in conjunction with MS in an effort to provide computational cycles to MS applications, make use of MS development tools, and allow for seamless desktop to supercomputer integration.

Please give us your feedback by e-mailing us at dreportal@arl.army.mil.

Arctic Region Supercomputing Center
DoD Supercomputing Resource Center

From the Director's Desk – Frank Williams

Frank Williams
Director, ARSC DSRC

The ARSC DSRC is one of three Centers in the HPCMP to receive new Cray supercomputers under the HPCMP Technology Insertion process for 2010.

With the installation of the ARSC DSRC's new 11,648 compute core Cray system this summer, DoD HPCMP users can access the newest in custom interconnect and chip technologies for their computational campaigns.

The 88,000-square-foot National Petascale Computing Facility in Champaign, IL, includes a 20,000-square-foot data center with an additional 10,000 square feet of raised floor for other infrastructure

The suite of new machines will be located in the Petascale Computing Facility (PCF) of the National Center for Supercomputing Applications (NCSA) at the University of Illinois.

Once installed, the systems will be remotely operated by the ARSC DSRC staff in Fairbanks and will eventually include a platform for Common Utility Enhancement Services (CUES). The ARSC DSRC is taking the lead in the programwide CUES effort aimed at increasing the functionality provided to users for postprocessing as well as an integrated archival storage system.

Locating the new ARSC DSRC systems in the PCF will provide a production-level demonstration of full-up remote operations that at the same time exceeds all of the high standards the ARSC DSRC requires for user satisfaction and data security.

The new ARSC DSRC supercomputer is considered by Cray to be one of its next-generation supercomputing systems, code-named *Baker*. It will feature a new interconnect chipset known as *Gemini* as well as enhanced system software to boost performance and productivity.

In addition to the ARSC DSRC supercomputer, PCF will house *Blue Waters,* www.ncsa.illinois.edu/BlueWaters/, a massive supercomputer funded by the National Science Foundation that will be capable of performing quadrillions of calculations every second.

The ARSC DSRC's partnership with one of the largest academic supercomputing centers in the U.S. will provide the HPCMP and its users with more opportunities for collaboration and shared use of technologies colocated at the National Center for Supercomputing Applications.

We also believe this venture will provide the Modernization Program with a venue conducive to developing strategic opportunities with other Federal agencies and especially the National Science Foundation. The NCSA Director Thom Dunning is very supportive of

The machine room of the National Petascale Computing Facility will house the ARSC DSRC TI-10 system, an 11,648-core next-generation Cray that will be operated from Fairbanks. Next year, NPCF will bring Blue Waters online as the first system of its kind to sustain one petaflop performance on a range of science and engineering applications

the partnership and is especially excited about working collaboratively to help the scientists and engineers supported by the DoD meet their research goals.

The ARSC DSRC is proud of its leadership role in connecting research, computing, and defense communities with the computers, data storage systems, high-speed networks, and next-generation experimental systems necessary for discovery in engineering and science.

ARSC DSRC
DoD Supercomputing Resource Center

Increasing Capabilities of Front-End Nodes to High Performance Computing Resources

By Dr. Greg Newby, Chief Scientist, ARSC DSRC

Three new types of nodes have been added to *Pingo,* the 3456-node Cray XT5 supercomputer at the ARSC DSRC. They are not computational nodes, administrative nodes, or storage nodes. While *Pingo*'s compute nodes have 8 CPU cores with 4 GB of memory each, these new nodes have 16 CPU cores, each with 8 GB of memory (128 GB total). Cray Inc. calls these external service login nodes (or esLogin, for short). They are early examples of what the HPCMP calls utility servers. Concepts and practices of utility servers are being developed by the HPCMP in a program referred to as Community Utility Enhancement Services (CUES).

Located on the University of Alaska Fairbanks campus, the ARSC DSRC is leading the CUES effort. Dr. Sergei Maurits, ARSC HPC Specialist, has gathered a team from the HPCMP DSRCs nationwide to examine requirements of utility servers, perform analysis of potential utility server solutions, and work toward a set of guidelines for utility servers within the HPCMP. This is a fast-moving initiative, since utility servers are slated for testing throughout 2010, with initial implementation early in FY2011. The CUES effort is a component of the HPCMP storage initiative (SI) and will leverage several new capabilities on the HPCMP computational resources the SI is introducing.

But what is a utility server? That question is not easy to answer, because there are different and sometimes competing views on utility server functionality. For example, compute nodes might not have sufficiently large memory or external network connectivity to run interactive visualization software. Another example is pre- and postprocessing of data sets, which could require long run times but only a single CPU.

Dr. Sergei Maurits, HPC Specialist at ARSC DSRC, is leading the programwide effort to develop concepts and practices of utility servers to enhance and improve the HPCMP users' experience

Other usage examples lead to application mixes that are not typical of HPC systems, such as database software and virtualization. Different hardware might also be a utility server component, such as graphics processing unit (GPU) accelerators or solid-state disk drives. A single utility server is unlikely to meet all of these needs, so Dr. Maurits and others are seeking to understand how different subsets of needs can be met and how utility servers can interact with mainstream batch-oriented HPC resources.

Pingo's esLogin nodes are early examples of utility servers that, among other things, greatly enhance capabilities for work that otherwise would typically land on cluster login nodes. This includes compiling and serial debugging, pre- and postprocessing, and some visualization and file staging. Compared with *Pingo's* original login nodes, the new esLogin nodes provide far more memory, as well as fast connectivity to *Pingo* and to the outside world.

The esLogin nodes also help to make *Pingo* more robust to user-caused system problems. Because other *Pingo* nodes, or *Pingo's* storage subsystem, or other elements do not depend upon these nodes, users can cause node slowdowns or even crashes without impacting the rest of the system. Because of the larger memory footprint, the esLogin nodes are better equipped to handle users' nonbatch workload.

Front and back image of the three esLogin nodes installed on Pingo, a Cray XT5 at ARSC DSRC

The esLogin nodes supplement and in many cases replace functionality of the login nodes that originally came with *Pingo*. The original nodes were comparable with the compute nodes. Similar to the process at other DSRCs, users could choose from several login nodes to do their work. While these nodes were effective for batch job submission and monitoring, as well as compilation and some pre- and postprocessing of data, some users found them to have insufficient capacity to meet all of their needs. Occasionally, heavy uses of the default login nodes would crash the nodes or result in noticeable slow-downs for other users – such as during large-scale visualization or parallel preprocessing of input data.

The new model is for users to select from either the original login nodes or the new esLogin nodes. These have a superset of the software on the original login and compute nodes, with additional visualization software. They run a complete Linux operating system, versus compute nodes, which have a reduced OS image that is optimized for large-scale parallel computing.

Dr. Maurits and the rest of the CUES team know that esLogin nodes address only a subset of utility server needs. But many of the structural elements are well-represented, including high-speed access to the cluster filesystem, the ability to interact with batch jobs, capability to get dedicated interactive access to compute nodes, ability to connect back to external systems for interactive X Window sessions, large memory and a symmetric multiprocessor architecture, and software geared towards analysis and pre-/postprocessing.

The demand for such nodes to be included on new HPC systems is reinforced by the TI-10 systems set to be installed by the end of FY2010. These new Cray systems include esLogin nodes rather than the more traditional login nodes found on *Pingo*. Moving forward within CUES and the SI, the CUES team will provide guidance on new HPC systems to be dedicated to utility server functions. Interaction and overlap with existing HPC systems are intended to be as seamless as possible, while allowing users to select the best resource for their jobs.

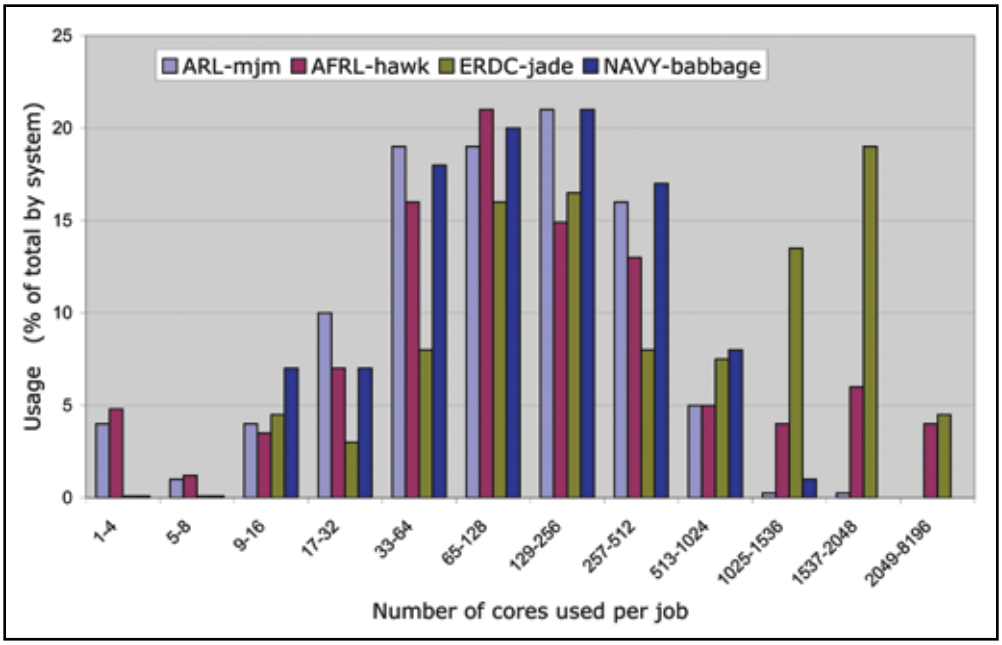
The ARSC DSRC has a tradition of early proof-of-concept and deployment of new technologies and services within the HPCMP. The ARSC DSRC's deep involvement in the SI and CUES and experiences with *Pingo*'s esLogin nodes will be beneficial to the HPCMP and its users as utility servers are deployed throughout the Program.

U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center

From the Director's Desk – Dr. Robert S. Maier

Dr. Robert S. Maier
Director, ERDC DSRC

Seven years ago, then-Director John West set a goal for the ERDC DSRC to become the "big jobs" Center by ensuring end-to-end support for jobs requiring a high core count. This goal has been largely realized, within the constraints of our common HPCMP queue policies. Jay Cliburn, Lockheed-Martin Technical Lead, documents the effect in the chart below, which plots the number of jobs versus job core count. A significant number of jobs are running at higher core counts. Our 16K-core SGI Altix, a Technology Insertion 2009 (TI-09) system, is already running an unprecedented number of 2000-core jobs. Our systems are growing in size. This year, the ERDC DSRC will install a 20K-core next-generation Cray system from TI-10. We expect users will scale up their jobs accordingly, and we will do our best to enable 4000-core jobs with routine fast turnaround. Users who run at higher core counts will receive special attention and support from our staff. It is a blessing that HPC systems are growing while their cost remains relatively stable. Industry continues to follow Moore's Law, with processing speeds growing exponentially in time. According to a recent *SciDAC* article, www.scidacreview.org/0904/html/multicore.html, even power consumption is not expected to grow as fast as processing speed. Nevertheless, electricity costs represent our most significant barrier to growth, in an era of flat operations budgets.



Greg Rottman is leading a study of energy conservation measures for DoD data centers. His focus includes the raised floor as well as the external infrastructure. Inside the data center, opportunities for conservation range from improving power distribution efficiency to more careful management of system cooling. Outside, opportunities include alternative energy sources, such as solar and geothermal. A guiding consideration in Greg's investigation is the need to work closely with system vendors to understand their roadmaps and trends in environmental requirements.

DoD provides financial assistance to the Services for energy conservation investments, through its MILCON funding line. Indeed, there is no better time for such investments than now. As operations budgets remain constant or shrink in real terms, increasing power costs will erode our ability to support new systems. Sound financial management is essential to meeting this challenge. The recognition of costs for infrastructure support, whether incurred by the Services or by the HPCMP, will help guide our investments and our plans for the future.

ERDC DSRC
DoD Supercomputing Resource Center

New Record Set on *Diamond* in Quantum Turbulence Simulations

By David Longmire, ERDC Information Technology Laboratory

Dr. George Vahala is an HPCMP Capability Application Project (CAP) enthusiast. He wrapped up his fifth CAP earlier this year and is looking forward to number six. "It's like letting loose a little boy in a candy store — with no supervision," said Dr. Vahala, Professor of Physics, College of William & Mary, Williamsburg, Virginia. "It's a wonderful, beautiful idea by the HPCMP. It's the greatest thing since chocolate!" he said.

CAPs are scheduled by the HPCMP before new systems go online for general use. Researchers compete for the opportunity of being the sole system user. The CAP gave Dr. Vahala and his team the opportunity to run their massively parallel code on an entire supercomputer for a month giving them the computing power to explore new areas of basic quantum physics research.

Dr. Vahala, joined in his research by his wife, Dr. Linda Vahala, College of Engineering & Technology, Old Dominion University, Norfolk, Virginia; Dr. Min Soe at Rogers State University; and Dr. Jeffrey Yepez, Air Force Research Laboratory, were able to set a new record in quantum turbulence simulations this year.

The CAP was conducted on the ERDC DSRC's new SGI Altix ICE 8200 named *Diamond*. "No one in quantum turbulence research has been able to run at this fine of a scale, so this is the first time that anyone has had this amount of grid resolution in a simulation," Dr. Vahala said.

"The CAP is the optimal time to use the machine. It allows us to run these huge grids; like on *Diamond,* we ran coupled Bose-Einstein condensate (BEC) states at a 4032^3 grid. We typically ran at 12,288 cores on *Diamond*," he said.

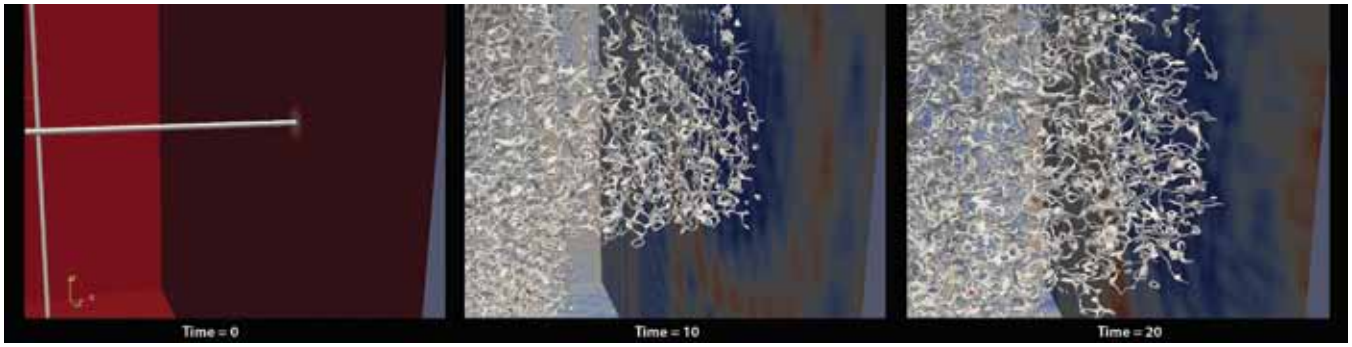
"Dr. Vahala ran 63 of the 12,288 core jobs during the CAP. After the *Diamond* CAP was completed, Dr. Vahala ran for more than 1.5 million hours and over 40 jobs using 4096 cores, the largest the queues will now allow."

— Bob Alter, ERDC DSRC HPC Service Center

"When you get huge grids, the amount of steps that you need scales, basically as the ratio of the grid squared; thus you have to run your simulations for much longer periods of time, and that's where the CAP comes into play. That's when you can utilize a very large number of cores with a time limit of a few weeks," Dr. Vahala said.

During the CAP, Dr. Vahala ran a total of 585 jobs and consumed 14.6 million cpu-hrs running his group's custom code called the QLG (quantum lattice gas) algorithm. The CAP runs generated more than 650 TBytes of data.

From the perspective of the DoD, the idea of the CAP is to first and foremost gain insight into challenging research problems that would lend themselves to the CAP computing environment; the Program Office also wants code that will push the system to find any problems or limitations. It is like taking a new ship on a shakedown cruise. "The DoD is very interested in the physics research data that we are getting, but it does help the systems people out. Systems people like us because we do shake it down," said Dr. Vahala.



*Quantum vortex simulation on a 4032^3 grid. The image series shows a small section of the volume data, bounded by three walls. In the first image (Time = 0), you see two of the quantum line vortices that are perpendicular to each other. The second image (Time = 10) shows the breakup of the line vortices. The third image (Time = 20) shows the small-scale vortex loops and strands. These are clearly visible utilizing the 4032^3 grid but would be missed in lesser resolution grid simulations. The images are from the recent CAP conducted on the ERDC DSRC's **Diamond**, an SGI Altix ICE 8200 system*

"There are two aspects of this research of particular interest to the DoD. One is generating codes that will run on quantum computers when they become available, as the usual CFD codes will not run on these future quantum computers; and the second is to be able to simulate the BECs and to predict new behaviors and uses. For example, there is a great impetus to experimentally develop atom interferometry using coupled BECs. This would permit the detection of changes in the gravitational fields or magnetic fields to extreme unheard-of precisions. For example, the DoD could use these interferometers in the future to detect underground tunnels," Dr. Vahala said.

So, Dr. Vahala's code makes it possible for him and his team to look at myriad basic physics problems during a CAP. "One of the biggest things is parallelization of a code so that if you have more processors, you can use them all. The problem is that most codes will choke after about three or four thousand processors. They can't utilize more because of the communication between the processors that is needed. It is like traffic pattern flow with traffic signals that are amok — the traffic is snarled by miscued traffic signals. A perfectly parallelized code is as though the synchronization is so good that one can go from one end of town to the other end without stopping. So, you are basically limited by the algorithm. Our algorithms don't suffer that. We have always found that things speed up with more processors, and the more processors you get, the greater the grid resolution that you can run," he said.

Analyzing the more than 650 TBytes of data generated during the CAP requires special requirements. Sean Ziegeler at the Navy DSRC worked with Dr. Vahala and his group to help them visualize their data. "Sean set it up so that I could visualize the data. He got ParaView up and running on *Diamond* and figured out how we could dump data on these huge 4032^3 grids without slowing the machine down, and then we were able to use ParaView to be able to start looking at the quantum vortices. Sean Ziegeler's work was absolutely critical. It permitted parallelized viewing of the vortices during the run so that we could see if we were barking up the wrong tree or had chosen a bad parameter regime. Basically, every data point on the 4032^3 grid is visible so you can see the fine structure. You would never be able to see this on a 1024^3 grid," Dr. Vahala said.

Dr. Vahala and his group were recently published in *Physical Review Letters.* Their published paper highlighted their research findings that directly resulted from CAPs on *EINSTEIN* at the Navy DSRC and *Diamond* at the ERDC DSRC. "We are gaining quite a bit of insight. What we managed to do both on *EINSTEIN* and *Diamond* was pull off the energy spectrum and see the $k^{-5/3}$ Kolmogorov Inertial Range. We were able to pull this out very nicely. It's like the classical cascades of energy. We were able to do this because we ran on such a huge grid. We also found what the spectral law was for the quantum cascade," Dr. Vahala said.

"The main thing that we found and write about in our paper was that we were able, using our quantum code and the 4032^3 grid, to handle the multiscale turbulence scales ranging from the classical turbulence regime right down to going into the quantum turbulence regime! This energy discovery was mainly found on the *EINSTEIN* CAP runs, but we did verify them on the CAP *Diamond* runs. Previously, other researchers had not seen this quantum cascade — partly because they introduced some extra damping terms into their simulations so as to see the classical Kolmogorov cascade. This destroyed the physics at the small scales, where the quantum cascade would live, and so they could not see the quantum cascade. The other reason is that their grids were too small anyway! Their codes hit mismatched traffic signals.

"So, for the first time the quantum cascades of energy have been identified in HPC simulation, and our research results indicate that the waves in these quantum cascades have a different structure than was theorized. We are looking further into this."

— Dr. Vahala

Classical turbulence studies are critical to the DoD and are utilized in designs for aircraft, ships, and other DoD systems. "Quantum turbulence is basically defined as vortices which are entangled. It's a spaghetti-type thing," Dr. Vahala said, "which is a thousand times smaller than classical turbulence." Understanding the basic physics of the quantum vortex is the objective of Dr. Vahala's team and their data simulations generated during the recent Diamond CAP at the ERDC DSRC.

Maui High Performance Computing Center
DoD Supercomputing Resource Center

From the Director's Desk – David Morton

David Morton
Director, MHPCC DSRC

Fiscal Year 2010 (FY10) is proving to be a year of significant growth for the MHPCC DSRC. Since the introduction in August 2009 of *Mana* (meaning power in the Hawaiian language), a new 9216-core Dell cluster, more than 500 user accounts are actively employing this HPC resource. The MHPCC DSRC has allocated more than 70,000,000 CPU hours for the HPCMP users for FY10. The MHPCC DSRC is also hosting nine high-priority HPCMP Challenge Projects with a combined allocation totaling more than 15,000,000 CPU hours.

The MHPCC DSRC is tasked to support the DoD HPCMP High Performance Computing Software Applications Institute for Space Situational Awareness (HSAI-SSA). One of nine HPCMP software institutes, the HSAI-SSA provides breakthrough capabilities by developing and transitioning HPC software applications. During the recent call for HPCMP Dedicated High Performance Computing Project Investment Award (DHPI) proposals, the AFRL DoD HSAI-SSA team proposed a system replacement for their previous DHPI systems, *Hoku* and *Polaris.* An HPCMP DHPI Award was granted. The MHPCC DSRC operations staff proposed the reuse of existing *Jaws* frames to support this requirement. For only eight percent of the potential awarded funds, the MHPCC DSRC was able to repurpose 640 cores of the existing *Jaws* system and one frame of DDN storage, upgrade the system memory to 8 GB/core, and standup the new system – *Kaku* (meaning barracuda in the Hawaiian language). This new *Kaku* HPC asset will further enable scientific and technological advances in Space Situational Awareness.

Continuous facility upgrades are ongoing to meet the growing demands of the MHPCC DSRC, which include the expansion of the Center's infrastructure and implementation of alternate green technologies (an R&D effort in advanced photovoltaic technologies). As the need for science and engineering continues to accelerate, the MHPCC DSRC continues to provide state-of-the-art technology and exceptional customer service. The MHPCC DSRC staff provide above and beyond support for special case needs. For years, the MHPCC DSRC has hosted advanced reservations, made exceptions for high-priority projects, and assisted users with codes, debugging, logins, batching, etc.

To ensure technological supremacy and to foster the flow of technology into warfighting systems, the MHPCC DSRC conducts R&D-oriented Directed Technical Tasks (DTTs) for the DoD and other government organizations. Two of the DTT Projects that the MHPCC DSRC supports recently received AFRL Directed Energy Directorate Awards. The AFRL's DoD HPCMP's HSAI-SSA received the 2009 R. Earl Good Award "For significant team contributions to the AFRL mission or image outside of AFRL and for accomplishments that have had a significant impact and enhanced the creditability of AFRL." AFRL's High Accuracy Network Determination System (HANDS) project received the 2009 International Award "For leveraged cooperative opportunities that provide mutual benefit in priority research areas that enhance and benefit the AF S&T capability."

The MHPCC DSRC continues to be a vibrant DoD organization dedicated to accelerate development and transition of the ever changing advanced defense technologies into superior warfighting capabilities.

MHPCC DSRC's Newest Supercomputer, *Mana*, Goes Green

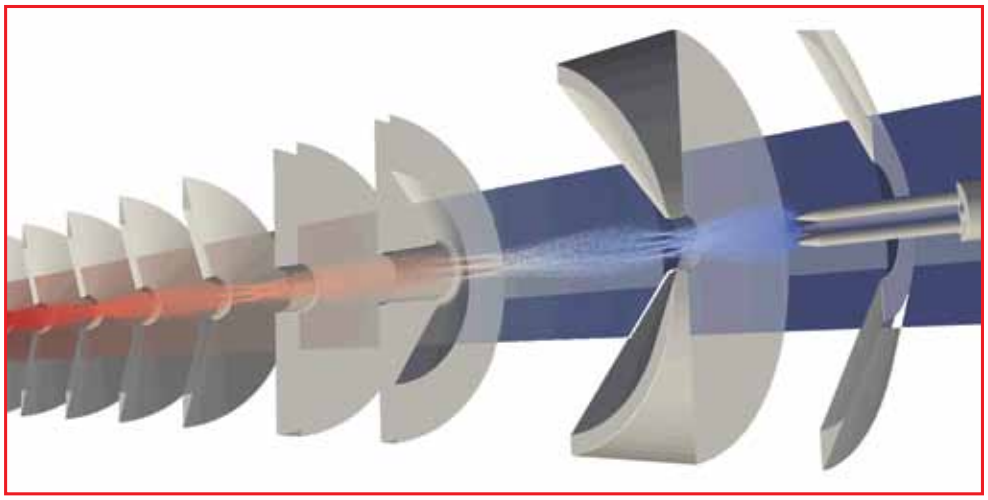
By Marie Greene, MHPCC DSRC Deputy Director

Mana utilizes the latest generation of Dell PowerEdge Servers. These servers come with an extensive collection of sensors that can automatically track thermal activity helping regulate temperatures and reduce energy consumption. The sensors are designed to automatically make adjustments to help reduce energy usage. The system is also deployed with variable speed fans mounted as part of Dell's latest blade chassis subsystem. The fans automatically sense the heat generated by the components in the chassis and increase the fan speed (and thus the overall power consumption) only when the systems are active. These heat-sensing fan subsystems will also save the MHPCC DSRC thousands of dollars in energy costs over the life span of the system and reduce our overall carbon footprint as a Center.

HPCMP Challenge Projects Have *Mana's* Lights Flashing

By Marie Greene, MHPCC DSRC Deputy Director

For FY10, the MHPCC DSRC is hosting nine HPCMP Challenge Projects allocating more than 15 million CPU hours on its newest supercomputer, *Mana*. Sponsors of the Challenge Projects include AFRL, the United States Air Force Academy (USAFA), the Army Research Laboratory (ARL), the Air Force SEEK EAGLE Office, and the Office of Naval Research (ONR). DoD Challenge Projects are large, computationally intensive projects. The guideline for consideration is a total computational resource requirement of at least 5 Habu-equivalent years annually, which is equivalent to approximately 2,500,000 processor-hours per year.



HPC modeling is used to enhance the quality of high-power lasers. The electron gun, shown in this figure, uses four 50 micron radius carbon nanotube (CNT) ropes to produce the electron beam through field emission. Four high aspect ratio CNT cathodes were used. The color of the particles represent the energy, blue being the least energetic, red being the most energetic. At the end of the cathode, the power transferred by the electron beam is 70 GW/m^2 (total power of 200 W and 30 microns in radius). Image courtesy of Nathaniel Lockwood, Civ USAF AFMC AFRL/RDHE

The FY10 HPCMP Challenge Projects that the MHPCC DSRC is currently hosting are as follows:

- Simulation of High Power Lasers (AFRL)
- Virtual Prototyping of Directed Energy Weapons (AFRL)
- Design of Energetic Ionic Liquids (AFRL)
- Aero-optical Distortions in Directed Energy Applications and Their Mitigation Using Feedback Flow Control (USAFA)
- Environmental Fate and Transport of Energetic Materials (ARL)
- Understanding and Designing Complex Ferroelectrics and Multiferroics from First Principles (ONR)
- Stability and Control Test and Evaluation Process Improvement Through Judicious Use of HPC Simulations (AF SEEK EAGLE Office)
- First Principles Studies of Ferroelectric Materials (ONR)
- Numerical Exploration of the Stable Atmospheric Boundary Level and Its Effect on Forecasting Battlefield Weather, Sensor Propagation and Diffusion, and Dispersion of Smoke and Other Agents (ARL)

MHPCC DSRC HPCMP DHPI Award Gives Rise to *Kaku* for the AMOS Site

By Chris Sabol, HSAI-SSA Director

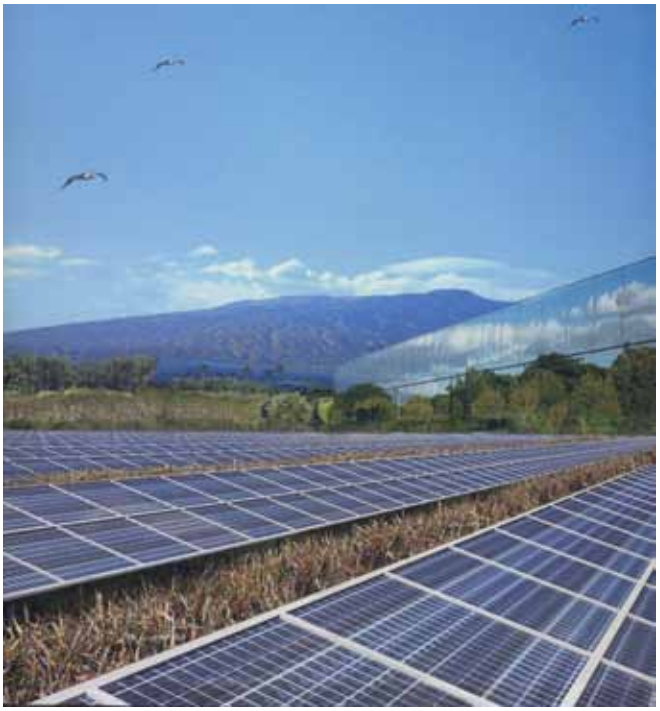
The Air Force Maui Optical and Supercomputing (AMOS) site recently received an HPCMP DHPI Award to support advanced image reconstruction. A system named *Kaku* was established at the MHPCC DSRC and replaced the previous distributed center dedicated Cray XD-1 systems named *Hoku* and *Polaris*. The system was provided to the High Performance Computing Software Applications Institute for Space Situational Awareness (HSAI-SSA) software developers on December 15 and to the Maui Space Surveillance System (MSSS) users on March 11.

The MHPCC DSRC, HSAI-SSA, and MSSS are all managed through the same AFRL branch at AMOS, and all three work together to deliver unmatched ground-to-space satellite imagery to the U.S. Strategic Command and other consumers. MSSS is home to the DoD's largest optical telescope and researches advanced image reconstruction techniques such as Physically Constrained Iterative Deconvolution (PCID), an algorithm developed to remove the effects of atmospheric blurring from a series of raw images. The HSAI-SSA transformed the PCID algorithm into a robust and efficient software application (three orders of magnitude speedup over the original research code), coupled it with a user-focused, Web-hosted run management system called the Advanced SPeckle Imaging Reconstruction Environment (ASPIRE), and delivered it back to MSSS image analysts for daily use. The MHPCC DSRC is home to the HSAI-SSA, most of its software application engineers, and the HPC resources utilized for PCID/ASPIRE RDT&E. *Kaku i*s comprised of 154 dual Woodcrest nodes (616 compute cores in total) reused from the previous MHPCC DSRC *Jaws* system and upgraded to 32 GB memory per node, 4 I/O nodes, and 2 login nodes. *Kaku* means barracuda in Hawaiian and is a skewed reference to its previous incarnation, *Jaws* (*Jaws* was actually named after one of the largest surf breaks in the world and not the fictional shark). The memory upgrades were required to support efficient operation of the PCID software with large data sets. All in all, *Kaku* enables a ~20x speedup in high-resolution image reconstruction for AMOS and opens the door for near real-time product delivery. The *Kaku* team included Kathy Borelli (contractor, KJS-Consulting), Scott Spetka (contractor with Rome Laboratory, AFRL/RITB), Major David Strong (AFRL/RDSM), and Bill Stevens, Ron Viloria, Bruce Duncan, Steve Gima, and Mike McCraney (all contractors with the University of Hawaii).

Maui Energy Improvement Initiative (MEII)

By Captain Joseph Dratz, MHPCC DSRC Deployed Systems Program Manager

MEII is an "American Recovery and Reinvestment Act"-funded, 15-month research effort to demonstrate the feasibility of providing photovoltaic (PV) power to Mana, MHPCC's supercomputer. The initiative will test traditional Silicon panels side by side with third-generation triple-junction optical concentrator PV modules. Additionally, the effort will install 100 kW of triple junction dual-axis tracking modules to demonstrate integration with the MHPCC Data Center. This effort will provide an environmental assessment along with cost and technical data that can be fed into a long-term renewable power solution for the MHPCC DSRC.



Solar photovoltaic panel

Navy DoD Supercomputing Resource Center

From the Director's Desk — Tom Dunn

Tom Dunn
Director, Navy DSRC

Providing the best computational capabilities for the Department of Defense and its high performance computing community has always been the cornerstone of the mission of the Navy DSRC. As we prepare locally for future capability enhancements, our staff continues to participate and provide leadership in numerous programwide ventures designed to broaden user access to our computational resources. These ventures include new storage infrastructure design, automated system monitoring, workload management enhancements, and leadership of automated baseline compliance-checking efforts. Having been one of the pioneer hosts of the Advance Reservation System (ARS), the Navy DSRC is now working to bring you the ARS on all of our unclassified computational systems. This system will allow users more options for running interactive and batch jobs along with current batch queuing options.

Ensuring maximum performance of our high performance computing (HPC) and storage resources is also of great importance to our team. Larger HPC platforms and more complex user jobs have led to local adjustments made to increase bandwidth to our archive servers. The Navy DSRC user outreach staff continues to look for ways to further improve their ability to assist users with challenges such as software installation and compilation, performance tool utilization, and debugging and porting applications between different architectures. Our network, security, and system administrators are vigilant in preserving system and network access and performance while anticipating future requirements both from the user community and the overall HPCMP perspective.

All of this effort rises from a simple concept: maintaining the highest possible customer support for our users, the high performance computing community, and the Department of Defense. Customer support is not simply a goal of the Navy DSRC; rather more importantly, it is the daily driving force instilled in our entire team.

Infrastructure Upgrades Prepare Navy DSRC for the Future

By Navy DSRC Staff

In early 2008, the Navy DSRC found that it would soon be welcoming *EINSTEIN,* a 12,876-core Cray XT5, and *DAVINCI,* a 5312-core IBM Power6 cluster, to its computational arsenal. Infrastructure upgrade requirements were immediately identified to accommodate the 43,800-lb *EINSTEIN* and the water-cooled *DAVINCI* in the main and secondary computer room facilities that serve the Center.

Updates to the main computer room facility required a phased approach, as the IBM Power5+ *KRAKEN* system still occupied a significant portion of the floor space there. A 73-foot wall was constructed to protect *KRAKEN* from construction debris during the 5-month-long Phase I process. The design called for the facility to provide up to 2500 kVA of power and 640 tons of cooling (1 chiller ton is defined as 12,000 Btu/h).

The unoccupied 4650 square feet was then upgraded from a 1250-lb/ft^2 rated floor to a 2000-lb/ft^2 rated 18-inch raised floor. The former water-based fire suppression system was upgraded to the FM-200 waterless system. Equipment on this main floor, including the test system and disk racks for *EINSTEIN,* is supported by one 60-ton chiller and two 320-ton chillers, eight computer room air conditioner (CRAC) units, three 2.2 megawatt (MW) generators, and five 625 kVA Uninterruptable Power Supplies (UPSs).

In November 2009, with *KRAKEN* retired and removed from the site, work began on Phase II of the facility upgrade to enhance the remaining portion of the main computer facility. This second phase upgraded 2796 square feet of raised floor and completed the replacement of the water-based fire suppression system. The Phase II area supports an additional six CRAC units and is also supported by the chillers, generators, and UPSs that service the Phase I area.

A secondary computer room facility supports the Center's remaining computational systems including *DAVINCI.* This 8578 square foot facility houses three IBM clusters on a 2000-lb/ft^2 rated 30-inch raised floor. Six hundred installed tons of cooling, of which approximately 300 tons are in use at present, and 10 CRAC units provide air and water cooling. The facility is supported by emergency generators and three 1 MW UPSs.

The planned Phase II computer room upgrades have recently come to a close, leaving the Center much more prepared to accommodate future HPC hardware acquisitions. The photographs below show Phase II work in process and then completed.

Work in Progress

Work Complete

Data Analysis and Assessment Center

From the Director's Desk – Dr. Michael Stephens

Dr. Michael Stephens
Director, DAAC

The DAAC is a programwide resource to assist all HPCMP users with their data visualization needs. Housed within the DAAC are several computer systems and expert staff dedicated to data visualization. The DAAC model of service is a three-tier approach. First we serve the community at-large by sharing our expertise through our Web site. This site has an ever growing number of tutorials and how-to's for a wide variety of data visualization challenges. Also at the community level, we answer users' questions specifically about visualization tools and techniques. In the second level of service, the collaborative level, users work with the DAAC staff to develop a workflow that users can then use to do their own data analysis. This often involves users sharing example data sets with the DAAC staff, thereby allowing more time to be spent exploring data analysis possibilities. In this way, both the users and DAAC grow in expertise. The users get a set of tools and knowledge on how to use them. DAAC gets exposure to a wide variety of data sets and visualization goals that, in turn, get shared with the entire community via our Web site. The last service level that the DAAC provides is for custom projects. These projects usually blend together data analysis with conceptual computer animation and/or video to tell a story about the user's HPC work. These are highly polished images and movies with narration, music, etc. Think Discovery Channel.

To learn more about the DAAC and how it can assist you, please visit our Web site: daac.hpc.mil.

ParaView Client-Server on Crays at the ERDC DSRC

By Randall Hand

Several Cray XT3, XT4, and XT5 systems are now in operation throughout the HPCMP, and several of you have migrated your codes to run in the unique Cray architecture. The three-tier architecture of the Crays (Login Nodes, Service Nodes, and Compute Nodes) causes problems for the typical client-server configurations used by visualization packages such as Kitware's ParaView. Just a subset of the problems include the following:

- Lack of Shared Library support on Compute Nodes.
- Lack of SSH support between various tiers.
- Queuing system constraints.

The Data Analysis and Assessment Center (DAAC) has been working with Kitware on these issues and has successfully compiled and deployed ParaView on various Cray systems throughout the program. Already in use by a select few pilot users, it is now available to all of you in the Program. The instructions below will help you connect your Windows Client (with the HPCMP Kerberos Kit installed) to a ParaView server running across the Cray backend. The same procedure will work for Linux, by replacing the Putty steps with standard SSH.

Windows Client Connecting to *Sapphire*

Before we begin, here are a few notes to start. In these instructions, I refer to the ERDC DSRC Cray named *Sapphire,* specifically node "sapphire05", but it can really be any node of *Sapphire,* so long as it is consistent. The same instructions have been tested on *Jade* (ERDC) and *Pingo* (ARSC) successfully.

Also, the currently deployed version is ParaView 3.6.1, fully MPI aware but with no experimental modules enabled.

To execute ParaView on the Crays

1. Fire up ParaView on your client. Click on the File Menu "Connect". "Add Server".
   - Give it a basic name, and select server type "Client/Server Reverse Connection". Leave the Host as Localhost and Port as 11111.
   - When you click Configure, on the next screen select "Manual" and save the connection.
   - Double click on the resulting connection, and ParaView should show a popup saying "Please wait while server starts.."
2. Pick a random number over 30,000; that will be your PORT1. Kerberize using the normal means, and then start the HPCMP putty.
   - Enter a hostname, sapphire05.
   - Expand the SSH area on the left, and select "Tunnels".
   - Enter your selected PORT1 for the "Source Port".
   - For a destination, enter "localhost:11111".
   - Click the "Remote" radio button, and click Add.
   - If done correctly, you should see a line read something like "R<PORT1> localhost:11111" appear.
   - Save this connection if you want, then click "Open" to connect.
3. You should be connected to *Sapphire* now... Copy `/usr/local/usp/ezViz/CNL/portfwd-sample.cfg` to your home directory. Edit the two numbers in that file so that the left number is another random number over 30,000 (we will call it PORT2), and the right number is PORT1. The result should look like the following:

   ```
   tcp { PORT2 { => localhost:PORT1 } }
   ```

4. Start up portfwd by executing

   ```
   /usr/local/usp/ezViz/CNL/portfwd -g -c <your new config file>
   ```

   It should then just sit there.

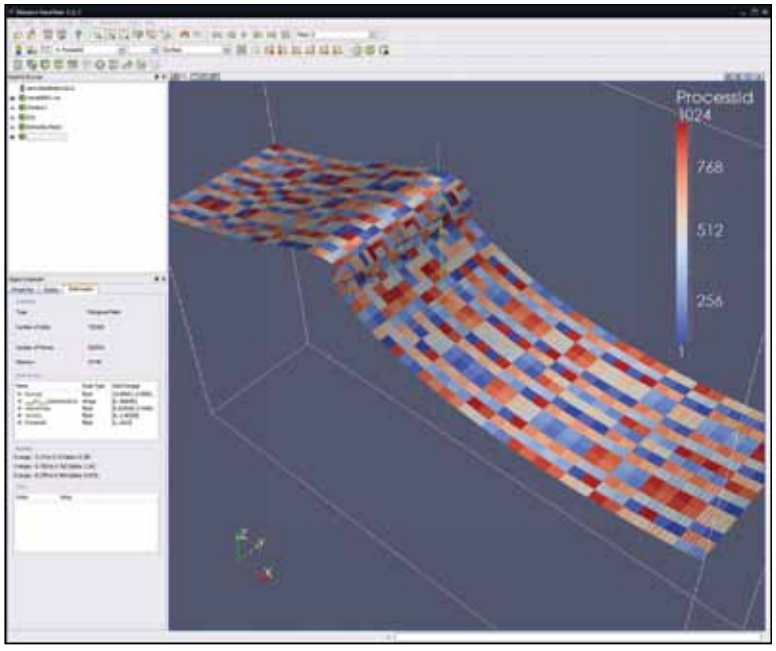


Figure 1. A 150-million cell run, visualized on 1024 processors of ***Jade*** *(ERDC)*

5. Fire up another terminal and SSH into *Sapphire* normally (any node).
6. Submit a queue job with the following command:
   ```
   aprun -n <number of nodes>  /usr/local/usp/ezViz/CNL/pvserver -rc -ch=sapphire05 -sp=PORT2
   ```

Linux Client Connecting to *Sapphire*

1. Fire up ParaView on your client. Click on the File Menu "Connect". "Add Server".
   - Give it a basic name, and select server type "Client/Server Reverse Connection". Leave the Host as Localhost and Port as 11111.
   - When you click Configure, on the next screen select "Manual" and save the connection.
   - Double click on the resulting connection, and ParaView should show a popup saying "Please wait while server starts.."
2. Pick a random number over 30,000; that will be your PORT1. Open a terminal, PKI Init, and then
   ```
   ssh -R PORT1:localhost:11111 username@sapphire05.erdc.hpc.mil
   ```
3. You should be connected to *Sapphire* now... Copy /usr/local/usp/ezViz/CNL/portfwd-sample.cfg to your home directory. Edit the two numbers in that file so that the left number is another random number over 30,000 (we will call it PORT2), and the right number is PORT1. The result should look like
   ```
   tcp { PORT2 { => localhost:PORT1 } }
   ```
4. Start up portfwd by executing
   ```
   /usr/local/usp/ezViz/CNL/portfwd -g -c <your new config file>
   ```
   It should then just sit there.
5. Fire up another terminal and SSH into *Sapphire* normally (any node).
6. Submit a queue job with the following command:
   ```
   aprun -n <number of nodes>  /usr/local/usp/ezViz/CNL/pvserver -rc -ch=sapphire05 -sp=PORT2
   ```

Linux Client Connecting to *Jade*

Jade has a slightly different configuration from *Sapphire* that complicates the network connectivity between the backend nodes and the client. As the backend nodes cannot directly connect to the login node via the public interface, you must take special precautions to correctly route between the two network interfaces.

1. Fire up ParaView on your client. Click on the File Menu "Connect". "Add Server".
   - Give it a basic name, and select server type "Client/Server Reverse Connection". Leave the Host as Localhost and Port as 11111.
   - When you click Configure, on the next screen select "Manual" and save the connection.
   - Double click on the resulting connection, and ParaView should show a popup saying "Please wait while server starts.."
2. Pick a random number over 30,000; that will be your PORT1. Open a terminal, PKI Init, and then
   ```
   ssh -R PORT1:localhost:11111 username@jade03.erdc.hpc.mil
   ```
3. You should be connected to *Jade* now... Copy /usr/local/usp/ezViz/CNL/portfwd-sample.cfg to your home directory. Edit the two numbers in that file so that the left number is another random number over 30,000 (we will call it PORT2), and the right number is PORT1. The result should look like
   ```
   tcp { PORT2 { => localhost:PORT1 } }
   ```
   Also edit the top line so that the "listen on" is the IP address of the "ss" interface for that login node. To get that IP address, simply execute '`/sbin/ifconfig ss`' on the node.
4. Start up portfwd by executing
   ```
   /usr/local/usp/ezViz/CNL/portfwd -g -c <your new config file>
   ```
   It should then just sit there.
5. Fire up another terminal and SSH into *Jade* normally (any node).
6. Submit a queue job with the following command:
   ```
   aprun -n <number of nodes>  /usr/local/usp/ezViz/CNL/pvserver -rc -ch=<ip address from step #3) -sp=PORT2
   ```

Unfortunately, the CNL nodes do not properly resolve hostnames, so you cannot set the "client host" (-ch) to something nice like "jade03"; you have to use the IP address of the internal interface (ss).

Thoughts and Notes

- When first testing this for accuracy and performance, use the Debug Queue with small processor counts (4 works well). You will not be able to load large data sets, but it is good enough to generate a simple Paraview Wavelet Source and benchmark.
- Users have been reporting problems attempting to submit jobs with the aprun command in the qsub script, but it works fine in Interactive when entered manually. This issue is currently under investigation.

Graphical User Interface to Create GAMESS Input Files from Within VMD

By Michael Lasinski, Army Research Laboratory, User Productivity Enhancement, Technology, and Training (PETTT) Program, Computational Biology, Chemistry and Materials Science (CCM) Onsite

Problem

The Applied Systems Biology project improves biological modeling capabilities in the Army, allowing researchers to explore concepts such as engineering bacteria to be capable of cleaning up firing range toxins. A significant portion of the project's research requires the investigation of molecular systems through the use of molecular dynamics (MD) simulations. Frequently, it is also necessary to simulate part of the molecular system in extreme detail at the quantum level. Despite current high performance computing (HPC) capabilities, such detail cannot be achieved by MD simulations alone, instead requiring the combination of an MD simulation and a quantum mechanics (QM) calculation known as a quantum mechanics/molecular mechanics (QM/MM) simulation. Typically, an MD code and a QM code are developed separately from one another. For QM/MM simulations, this is a significant issue because both codes have to interface with one another. In fact, several difficulties stem from the lack of an efficient and user-friendly interface to set up and run a QM/MM simulation. One such difficulty with setting up a QM/MM simulation is the inability to easily select atoms from the molecular system to be included in the QM calculation.

Methodology

As part of the Applied Systems Biology project, Margaret Hurley, Army Research Laboratory (ARL) Weapons and Materials Research Directorate, uses the Nanotechnology Molecular Dynamics (NAMD) tool from the University of Illinois to perform MD simulations on biomolecular systems. In future portions of the project, mixed quantum/classical simulations will also be performed. While mixed quantum/classical simulation is becoming a more accepted research tool, its usage is still limited to select practitioners, and job setup and analysis is still time-intensive. Quantum-oriented graphical user interfaces (GUIs) such as Gaussview do not have the necessary graphics capabilities to work easily with large biological systems. Visual Molecular Dynamics (VMD), also from the University of Illinois, is typically used to both view such biomolecular systems and to set up MD simulations to be performed using NAMD. VMD also has some ability to create input files for QM calculations. It is not possible, however, to select atoms from a displayed molecule and then place only those atoms into an input file for a QM calculation. Furthermore, there is no ability to add to the input file any means of dealing with the interface between the MD simulation and the QM calculation.

To address these issues, a GUI was written by the ARL/DoD Supercomputing Research Center (DSRC) User Productivity Enhancement, Technology Transfer, and Training (PETTT) Program Onsite Michael Lasinski that can be added to VMD as a Tcl/Tk plugin as shown below in Figure 1. This can be done on a local desktop or on a DSRC machine where VMD is installed, such as *MJM* (the Woodcrest cluster at the ARL DSRC). With this GUI, it is now possible to select atoms by clicking on them from within VMD in order to create a GAMESS (a QM code widely available on machines such as those at the ARL DSRC) input file. The user is able to easily select individual atoms or portions of a molecule to be added to the GAMESS input file. For example, with this GUI, it is now possible to quickly add only the atoms in the red box of the molecule shown in Figure 2. (See Figure 3 for a close-up view of those specific atoms.) In addition, the GUI provides the ability to include in the GAMESS input file common QM/MM interface options such as linking hydrogen atoms and quantum capping potentials. Finally, there are additional options available to further modify the GAMESS input file such as changing the type of QM

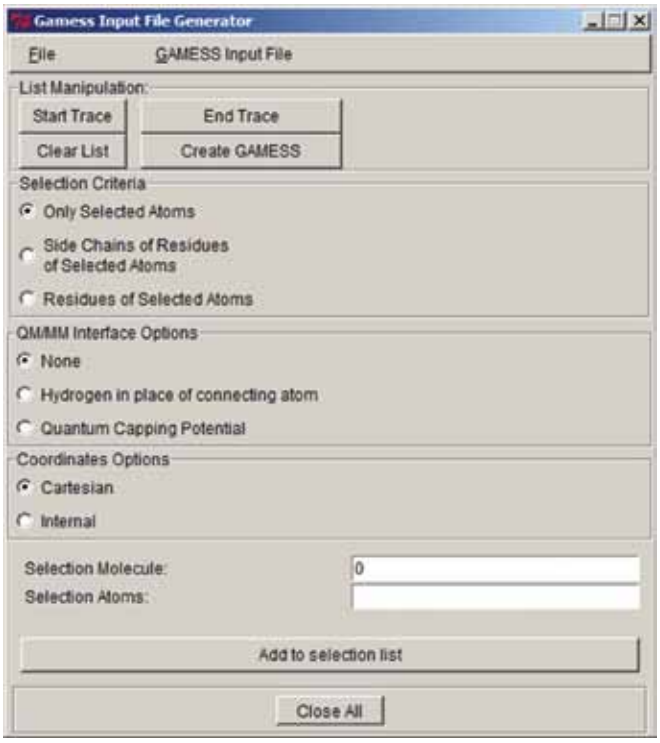


Figure 1. Initial GUI window

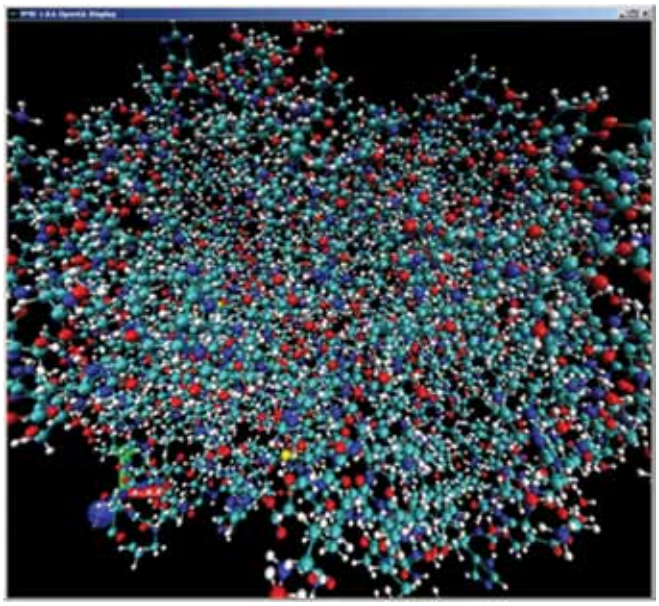


Figure 2. Protonated CPK representation of Human Deoxyhemoglobin loaded into VMD

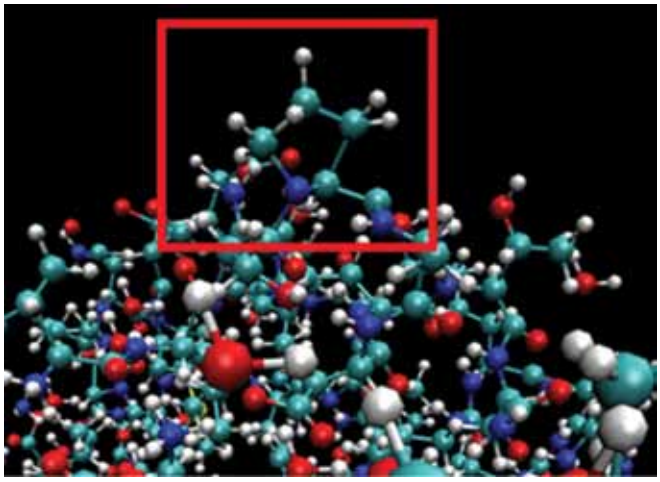


Figure 3. Red box represents atoms selected from the molecule to be included in a quantum mechanics calculation (zoomed in from Figure 2)

calculation that is performed, changing the basis function, and using effective core potentials.

With this in place, Hurley and incoming postdoctoral fellow Scott Pendley (ARL/WMRD) will be able to apply a novel QM/MM formulation that has shown great promise on nonbiological systems, but has not been tested on enzymes. As the immediate focus of the Applied Systems Biology project is biodegradation of energetic materials, initial tests will focus on the reduction of TNT by hydrogenase. This reaction has been studied experimentally but not well enough to be completely understood. Mixed QM/MM simulations will provide sufficient detail to confirm or deny certain molecular rearrangements that have hitherto been postulated based on experimental evidence but not proven. This type of analysis is *only* possible with mixed QM/MM methods, which have not been previously applied to this system.

Users Supported

This work supports Margaret Hurley (ARL/WMRD) by developing an interface to efficiently define the QM part of a QM/MM simulation by selecting atoms from a molecule loaded into VMD and putting them into a GAMESS input file. This GUI also provides options to handle the interface between the QM and MD part of the calculation in the GAMESS input file. By facilitating the generation of the GAMESS input file, this GUI is intended to be part of the larger effort to quickly and easily set up and run QM/MM simulations.

"This GUI will significantly speed up job submission for the QM/MM portion of the Systems Biology project and will allow my incoming postdoc to hit the ground running. Additionally, PETTT's expedited turnaround on this allowed me to focus on other portions of the work. This resulted in a briefing to ARL management with project milestones ahead of schedule."

— Margaret Hurley (ARL/WMRD)

DoD Impact

This work has wide application across the DoD by providing an efficient and user-friendly software tool that can be used to set up GAMESS input files and, ultimately, be part of a set of software tools that would better facilitate running QM/MM simulations. This tool will facilitate the use of these codes by users on ARL/DSRC platforms to perform these calculations. As an example, this GUI will specifically help to further the stated goal of developing in-house systems biology capabilities at ARL to model biological systems. As part of the Applied Systems Biology project, these QM/MM simulations of biological molecules will be used on a wide range of applications, the first of which will be the study of the breakdown of nitroaromatic compounds by bacteria. Nitroaromatics, which are a main component of current munitions, have toxicological and physical properties that make them an environmental hazard in areas of release, such as firing ranges and manufacturing plants. The development of bacteria that can effectively break down these compounds *in situ* would greatly facilitate range cleanup.

Acknowledgments

The authors thank the DoD High Performance Computing Modernization Program at the Army Research Laboratory for the funding of this effort.

User Interface Toolkit (UIT) v3.1

By Wes Monceaux, Keith Rappold, and Scotty Swillie, U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center

With the programwide move to CAC for Kerberos authentication, major changes were required within the User Interface Toolkit's (UIT) architecture to accommodate the physical interaction with the common access card (CAC) reader. Basically, this move broke the UIT's anywhere, anytime, Web-only model, and a serious "rethink" had to take place to ensure this tool remained available to the High Performance Computing Modernization Program (HPCMP) users and developers.

UIT version 3.0 was the first step toward addressing CAC support for UIT applications. This version of the UIT shifted functionality from the centralized UIT Web service to the user's desktop. Along with this shift came newer capabilities, but also some drawbacks. Multitiered, multiuser Web applications were unable to utilize UIT version 3.0 because of its desktop-centric nature. UIT version 3.1 is designed to address this shortcoming.

In order to understand what has changed, a quick overview of how Version 3.0 works is needed.

Version 3.0 Overview

Version 3.0 requires a one-time download of ezHPC (or another interface) and the UIT that comes prepackaged with a small Web server that is run accessible only from the local system. Developers can code against this application programming interface (API) the same way that it is coded against Version 2.0 with the exception of authentication. Authentication takes place via `pkinit` or `kinit` using the DSRC Kerberos client kit, just as it is done for any other application that requires Kerberos. From that point forward, API calls by a user interface to the UIT verify that the user has a ticket and then simply uses that ticket to perform actions on the user's behalf on the HPC machines.

UIT version 3.0 requires a user to have Java Runtime version 1.5+ and the DSRC Kerberos toolkit installed for their platform.

Here's how the process works:

1. Users use their SecurID card or their CAC to authenticate using the DSRC Kerberos `kinit`, `pkinit`, or similar command. Once authenticated, the ezHPC Java application running on the user's desktop then issues commands (using `ssh`) to interact with the HPC systems.
2. The HPC systems treat the `ssh` connections from the interface no differently than any other connections made by this user.

This version of the UIT effectively behaves just like any of the other DSRC tools (i.e., Filezilla, PuTTY, etc.) and utilizes the user's existing Kerberos tickets to interact with the HPC systems. The UIT is oblivious to how a user acquired his Kerberos ticket (CAC, HToken, or SecurID).

Centralized UIT information is still maintained in the UIT database. This includes all user-specific pIE data and the systems to which they have access. It also includes the configuration information house for each system to provide a uniform interface to them – i.e., queuing information, system availability, etc. Updates to the UIT will be downloaded and installed automatically when users log in.

Version 3.0 Benefits

Aside from CAC authentication, there are other benefits that came with the way Version 3.0 interacts with the API. Direct connections from the user's machine to the HPC machines are possible, allowing for better performance when manipulating large files. Additionally, since the API is now accessed locally, method calls respond much faster than before.

Benefits to moving UIT v3.0 in this direction are as follows:

- Use of CAC and any DSRC-supported authentication.
- Direct file transfer and communication between client and HPC Systems.
- Security architecture is greatly simplified.
- Users' Kerberos ticket life not limited to 1 hour (depending on which network they are on).
- Method calls respond faster than before, since the API is now accessed locally.

Version 3.1 Overview

As with any situation where trade-offs must be made, there are issues with Version 3.0 that needed to be addressed. The most pressing problem was that Web-site-type UIT clients that authenticate users by passing their credentials on to the UIT Web service do not work with the Web service model of Version 3.0. For this reason, UIT Version 3.1 was developed.

UIT Version 3.1 makes use of the HPCMP Single-Use Authentication Method (HSAM) tool to generate SecurID-like challenge/response codes with a user's CAC that can be used during kinit. The HSAM tool will become a standard part of the DSRC Kerberos

toolkit and will be needed by users of UIT Version 3.1 for CAC-based authentication.

UIT-Enabled Web Application Usage Scenario

Figure 1 gives an overview of how a user might use a UIT-based Web application.

1. A user will use his favorite Web browser to access a UIT-enabled Web application. The login screen will look like many common Web site login screens. Nothing about CAC is required to access the page. In fact, the Web browser itself need not be CAC-aware. The user will enter his Kerberos principal and password and click "Get Challenge".
2. The login process is not yet complete, as a challenge and response exchange must occur. The login Web page will now display the challenge code issued by the Kerberos system. The user must provide a corresponding challenge response to successfully log in.
3. HSAM is the tool used to generate a challenge response. A user must have the DSRC Kerberos toolkit that includes the HSAM tool installed on their system. The user will first place his CAC into the reader on their system. From a command prompt, the user will run the HSAM command (hsam) supplied with his Kerberos principal and the challenge code as arguments. He will then be prompted for his CAC pin code. Upon entering the correct pin, a response will be printed.
4. The user can now complete the login form on the Web application using the challenge response received from the HSAM command and clicking "Complete Login".
5. The user has now successfully generated a Kerberos ticket on the UIT Web service for use by this Web application to interact with the HPC systems on the user's behalf.

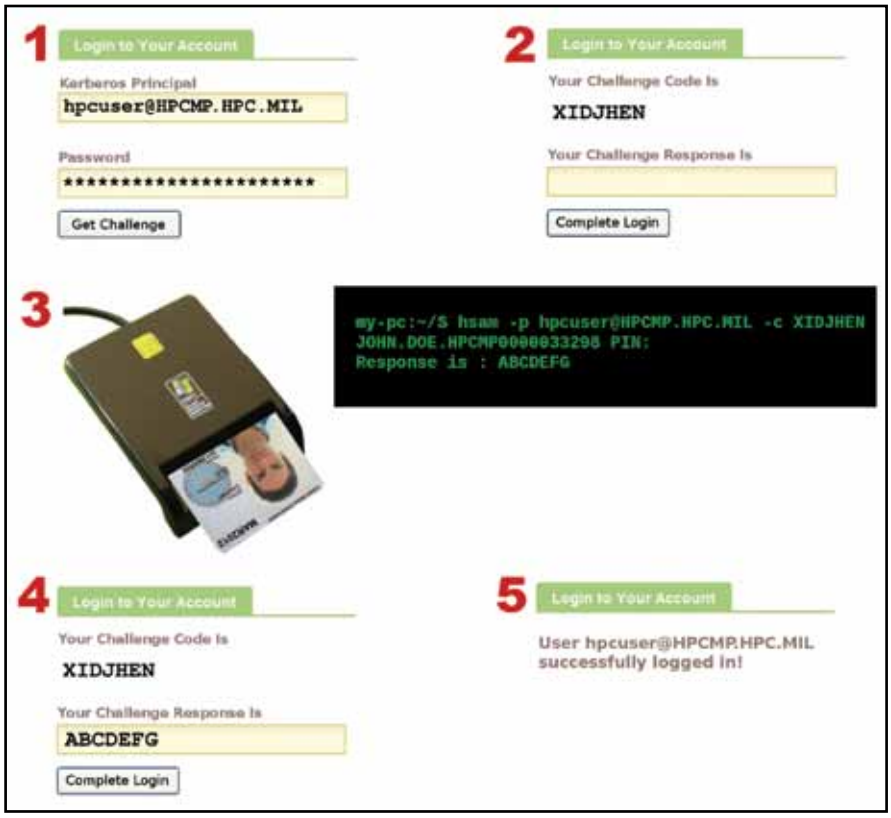


Figure 1. UIT-enabled Web application using CAC with HSAM

UIT Developer Options

The architecture of UIT Version 3.1 is nearly identical to that of 2.0. The primary changes are to authentication methods to permit this multistep challenge/response mechanism using HSAM. For developers, the UIT API has not changed significantly, as we try not to break application compatibility with the UIT unless absolutely necessary.

So which version of the UIT do you target? It depends on what kind of application you are writing. If it is a Web application or Web site, then version 3.1 is your only option. If you are developing a desktop application, you can use either version. Version 3.1 requires fewer things to install on a user's system, but version 3.0 is able to directly access the HPC systems from a user's desktop. For desktop application developers, there is certainly no reason you could not try both and see which satisfies all of your requirements best.

Conclusion

The benefits for Web applications are clear when comparing UIT Version 3.0 to Version 3.1. Several of the benefits (direct file transfer, etc.) from version 3.0 are mostly lost when moving to version 3.1 for desktop UIT clients, however. Ideally, we would like to combine the benefits of both versions. These are ideas that we are researching as we move the UIT forward.

Visit https://www.uit.hpc.mil for more information.

ASC Users to HPCMP Realm Consolidation

By AFRL DSRC Staff

In October 2009, the High Performance Computing Modernization Program Office (HPCMPO) announced the intent to merge users of the four Kerberos realms (ASC.HPC.MIL, ARL.HPC.MIL, NAVO.HPC.MIL, and WES.HPC.MIL) into the consolidated corporate realm HPCMP.HPC.MIL or the ARSC.EDU realm. For anyone unfamiliar with the ARSC.EDU realm, it is reserved for users awaiting completion of their National Agency Check (NAC) or for users only running on the open Arctic Region Supercomputing Center (ARSC) systems.

The consolidation project started with the consolidation of users in the ASC.HPC.MIL realm into the HPCMP.HPC.MIL realm. The Aeronautical Systems Center (ASC) (now the Air Force Research Laboratory (AFRL)) Kerberos Key Distribution Center (KDC) is physically maintained at the AFRL DoD Supercomputing Resource Center (DSRC), as is the HPCMP.HPC.MIL KDC. The close physical proximity and staff familiarity with both KDCs made the choice to consolidate the ASC realm first easy.

The ASC consolidation effort started in October 2009 and was completed in November 2009. AFRL transferred approximately 300 users in a 4-week time span with transfer groups consisting of 20 to 50 users. Transfers occurred twice a week.

To minimize the impact for the ASC-realmed users, a plan was created to automate many manual account processes, apply internal database updates, and monitor updates between the Portal for Information Environment (pIE) and all HPCMP-affiliated sites potentially affected by the realm change. This plan helped to maintain a less than 24-hour outage for the majority of the users.

The recent push toward consolidation will provide many user benefits. Consistent and reliable front-line Kerberos support is more easily provided if users are all realmed uniformly: password resets for users who need them can be done in approximately 15 minutes instead of a potential turnaround of 1 to 2 hours; all users needing to obtain a Kerberos ticket utilize the same set of IP addresses, minimizing the helpdesk turnaround time for new users needing this information.

Users should be aware that the IP addresses for the HPCMP Kerberos realm should be opened by their local network administrators. For a user's Kerberos ticket to forward to an HPC system appropriately, it may also be necessary to have allocated system IP addresses opened as well. All IP addresses can be obtained by contacting the Consolidated Customer Assistance Center (CCAC) at 1-877-222-2039 or help@ccac.hpc.mil.

External Login Servers Enhance *EINSTEIN* Capabilities

By Navy DoD Supercomputing Resource Center (DSRC) Staff

With over 12,000 cores, 25 TB of memory, and 500 TB of high-speed storage, the Cray XT5 *EINSTEIN* at the Navy DSRC is one of the most powerful supercomputers in the Department of Defense. One weakness that *EINSTEIN* has always possessed, however, is the inability to handle demanding interactive and pre/post-processing work on its login nodes. That has changed with the addition of four Cray external login (esLogin) servers. Each one of the esLogin servers provides an enormous increase in the amount of CPU, memory, and external network bandwidth of a current "built-in" login node.

Existing XT5 Login Nodes: Built on XT3 Technology

While the Cray XT5 platform is a relatively new technology, its service nodes (login, I/O, etc.) are essentially the same as what was designed for Cray XT3 systems in 2004. Each login node has the same high-speed Seastar 2+ interconnect as a compute node, but only has one dual-core 2.6 GHz AMD Opteron processor and 8 GB of RAM. This relatively small amount of memory, combined with the fact that the nodes have no locally attached disk to use for swap space, may seem like a bad design decision, but the "lean" design of the service blades actually works well for their originally intended purpose. XT systems were designed with the intention that users would only use service/login nodes to compile codes and submit PBS jobs, and the compute nodes (controlled by PBS/ALPS) would handle all of the demanding work. The reality, however, is that users create a much more varied workload on login nodes, doing anything from data pre-/postprocessing to interactive data analysis that cannot be done from compute nodes or PBS batch jobs.

Table 1 – Comparison of *EINSTEIN* nodes/servers

System	Login node	Compute node	esLogin server
CPU	Dual-core 2.6 GHz	2x Quad-core 2.3 GHz	4x Quad-core 2.4 GHz
RAM	8 GB	16 GB (32 GB on bigmem nodes)	128 GB
Swap	N/A	N/A	32 GB
Process Limits	2 GB	N/A	16 GB
External Network	2 x 1 Gb/sec Ethernet	N/A	1 x 10 Gb/sec Ethernet

Currently, limits are enabled that prevent a user process from using more than 2 GB of RAM on an *EINSTEIN* login node. This makes it nearly impossible to perform some tasks on the login nodes, but was necessary to prevent "runaway" user jobs from crashing the system by using all of a login node's available memory.

esLogin Servers: Designed for Pre-/ Postprocessing and Interactive Work

The High Performance Computing Modernization Program recently purchased esLogin servers for all Cray XT systems throughout the Program. The Navy DSRC is replacing the four currently available "internal" login nodes on *EINSTEIN* (EINSTEIN1.navo.hpc.mil – EINSTEIN4.navo.hpc.mil) with four esLogin servers. Each server is a Dell r905 chassis with four quad-core 2.4 GHz AMD Opteron processors, 128 GB of RAM, and 300 GB of locally attached RAID-1 storage. Each esLogin server is directly connected to *EINSTEIN* via a dedicated 10 Gb/sec Ethernet link. An additional 10 Gb/sec Ethernet link will connect each esLogin server to the DSRC network and DREN.

The esLogin servers share the same $HOME and $WORKDIR Lustre file systems as *EINSTEIN* and have the same programming environment, compilers, and application software as *EINSTEIN*. Programs compiled on the esLogin servers will be automatically optimized for the XT5 architecture and will run on *EINSTEIN* without any needing any modifications. Users will have access to the same PBS commands as on *EINSTEIN* and will be able to submit PBS jobs to *EINSTEIN* directly from the esLogin servers.

Since each esLogin server has significantly more processing power and memory than an "internal" *EINSTEIN* login node, user processes will be allowed to address up to 16 GB of memory. This gives users a friendlier environment to perform pre-/postprocessing work, data analysis, and other interactive work.

After completing internal testing and pioneer user evaluations, the esLogin servers at the Navy DSRC will be available to all users sometime during the second quarter of 2010.

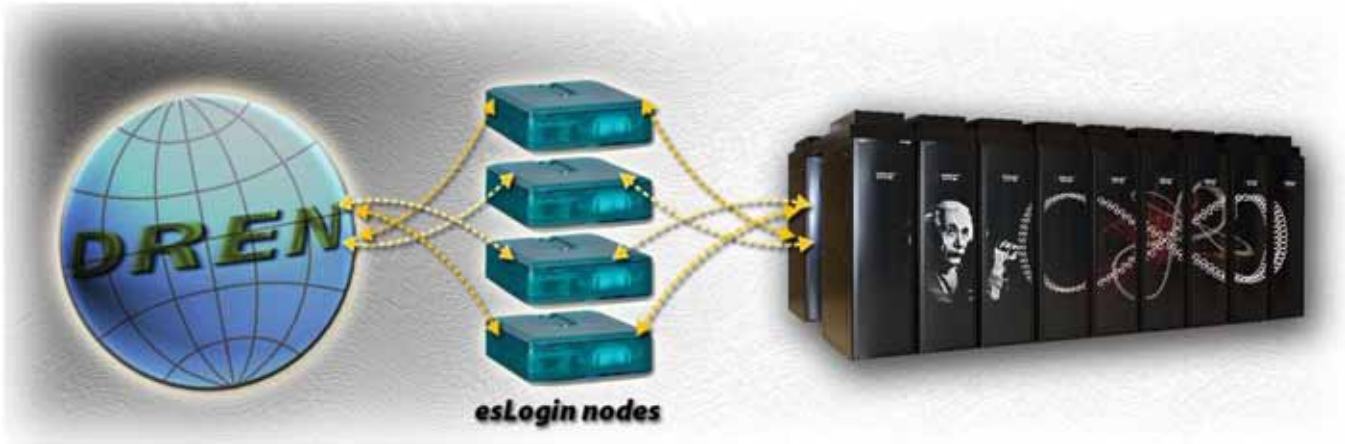


Figure 1. Illustration of esLogin connections to EINSTEIN and external network. All connections are 10 Gb/sec Ethernet

Python for Scientific and Parallel Computing

By Dr. Jose Unpingco, Air Force Research Laboratory, User Productivity Enhancement, Technology Transfer, and Training (PETTT) Program Signal/Image Processing Onsite

Introduction

Interest in high productivity languages has been accumulating recently and with good reason: they boost scientific productivity by letting scientists focus on science instead of computer programming. High productivity languages, such as MATLAB, facilitate interactive construction and visualization of computations by providing an intervening "interpreter layer" between the user and the hardware that both mitigates the underlying hardware specifics (e.g., libraries, memory, drivers) and provides an easy-to-use coding environment. This interpreter provides portability – code written in the high-level language can be run on any platform that provides the same interpreter.

Python is an open-source high productivity language that has matured into a serious platform for scientific computing and visualization. Python was originally designed as a portable, general-purpose, easy-to-learn, object-oriented "glue" language. It comes with a standard library that covers many common topics such as text processing, file I/O, and data compression. However, the real power of Python comes from the large number of freely available open-source Python modules that cover a wide variety of topics from basic numerical algebra to graph theory to complex data visualization. These modules provide syntactic support for arithmetic and mathematical operations and common algorithms including FFTs, numerical integration, optimization, special functions, computer algebra, interactive data visualization, publication-quality plotting, and modules to interface with numerous other programming languages (e.g., FORTRAN and C). Python runs on a variety of platforms – from high performance computers (HPCs) to graphic processing units (GPUs) to mobile telephones.

In this article, we will discuss Python for scientific and parallel computing, important open-source Python modules for numerics and visualization, the basic structure of the Python language, parallel computing using Python, and work at the Department of Energy and other government agencies that use Python as a framework for their scientific computing.

Python Usage

Let us examine some features of the Python language to understand how it works. The traditional "Hello world!" program in Python is the following one line:

```
print "Hello world!"
```

This program can be run interactively at the command prompt by typing

```
% python
```

which starts the interpreter

```
Python 2.4.3 (#1, Sep 17 2008,
16:07:08)
[GCC 4.1.2 20071124 (Red Hat 4.1.2-
41)] on linux2
Type "help", "copyright", "credits" or
"license" for more information.
>>>
```

Then, typing the program in the interpreter at the prompt (>>>),

```
>>> print "Hello world!"
```

gives,

```
Hello world!
```

The program can also be run in batch mode by saving it into a plain text file (say, `hello_world.py`) and running it as

```
% python hello_world.py
```

Key Scientific Modules

Python is a general-purpose language, and specialized computations are facilitated by separate Python modules. For example, the open-source `numpy` module provides the basic array and matrix types within Python. Python utilizes modules by "importing" them as in

```
import numpy
```

The following short program computes the eigenvalues of a diagonal matrix:

```
import numpy
m = numpy.eye(5)
numpy.linalg.eigvals(m)
```

Modules provide interactive help with the "help" command as in,

```
help(numpy.eye)
```

which produces the following output

```
eye(N, M=None, k=0, dtype=<type 'float'>)
  eye returns a N-by-M 2-d array where
the k-th diagonal is all ones,and ev-
erything else is zeros.
```

There are several visualization engines available in Python, depending on the kind and amount of data to be visualized. The best general-purpose and easy-to-use visualization package is `matplotlib`. The following example shows how to plot a simple function

using the `pylab` component of `matplotlib`.

```
import numpy
import pylab
x = numpy.arange(10)
y = x**2
pylab.plot(x,y,'o-')
pylab.show()
```

and the corresponding plot that is generated.

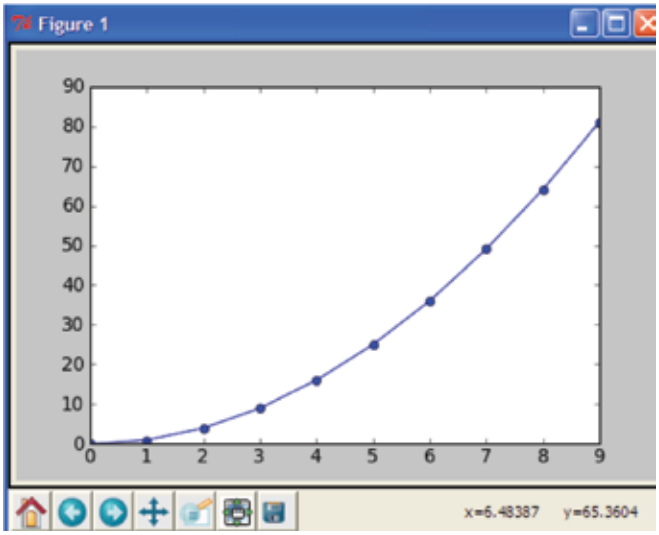


Figure 1. Simple graph created using Python's matplotlib module

As you can see, the syntax of the plotting command is similar to that of MATLAB. Three-dimensional and more complicated plots are also supported by `matplotlib`.

Browsing Scientific Python Using SAGE

Since the majority of Python programmers are Web developers, there is a well-developed Web toolkit for Python; the scientific Python community has capitalized on it by creating comfortable browser-based interfaces. The SAGE project is a consolidation of 100 of the best scientific open-source packages into a unified command-line and browser-based interface. The integrated packages include GNU GSL (Gnu Scientific Library), OpenOpt (numerical solvers), PARI/GP (Number theory), among others.

SAGE can be started on the command-line as shown,

```
cougar:1002> sage
----------------------------------------------------------------------
| Sage Version 4.3.1, Release Date: 2010-01-20
| Type notebook() for the GUI, and license() for information.
----------------------------------------------------------------------
sage:
```

where computations can be entered immediately at the `sage:` prompt, using the underlying integrated packages. To start the browser-based interface, you type `notebook()` and then navigate to the corresponding

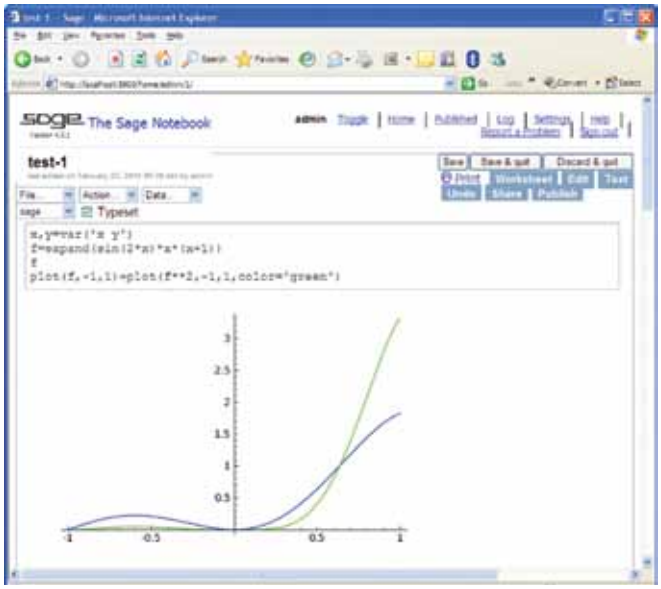


Figure 2. SAGE is a consolidation of over 100 of the best open-source scientific computing modules in a unified Web browser interface as shown. Computations entered in the browser generate corresponding results, including two- and three-dimensional graphics, which are also rendered in the browser

URL SAGE provides. As shown, computations can be entered directly into the Web page where the results are rendered alongside the input. The browser-interface also allows for high-quality LaTeX typesetting with integrated two- and three-dimensional graphics.

The advantage of running SAGE in the browser as opposed to the command-line is that it makes it easier to cut-and-paste rendered graphics or mathematical typesetting symbols and notation.

Parallel Computing in Python

Python supports both interactive parallel computing and full MPI parallel coding. Interactive parallel computing is provided by IPython wherein a corresponding set of separate Python processes is spawned on distinct compute nodes on an HPC. IPython then communicates with these processes to stage and manage parallel computations. For example, the following shows how the `MultiEngineClient` is imported and then used to connect to the separately running Python processes via the `mec` variable.

```
>>> from Ipython.kernel.client import *
>>> mec=MultiEngineClient()
```

Now, calculations can be automatically distributed by "mapping" a defined function over the available compute nodes (15 in this case).

```
>>> def f(x): return x**10
>>> mec.map(f,range(15)) # f is ap-
plied in parallel
```

```
0,1,1024,59049,1048576,9765625,60466176,
282475249,1073741824,3486784401L,
10000000000L,25937424601L,61917364224L,
137858491849L,289254654976L]
```

Note that IPython automatically takes care of scattering and distributing the data as well as gathering the resulting answer in the interactive interpreter for further processing. Subsequent calculations within IPython can utilize the full range of Python functionality, thus providing a clean integration with interactive parallel computing in a feature-rich development environment.

Full MPI bindings are available in Python via MPI4PY that is constructed on top of the MPI-1/MPI-2 specification. It supports point-to-point and collective communications of Python objects as well as optimized communications for specific Python objects such as numpy arrays. For example, the following is a short Python MPI program that broadcasts a Python dictionary:

```
from mpi4py import MPI
comm = MPI.COMM_WORLD
rank = comm.Get_rank()
if rank == 0:
   data = {'key1' : [7, 2.72, 2+3j],
           'key2' : ( 'abc', 'xyz')}
else:
   data = None
data = comm.bcast(data, root=0)
```

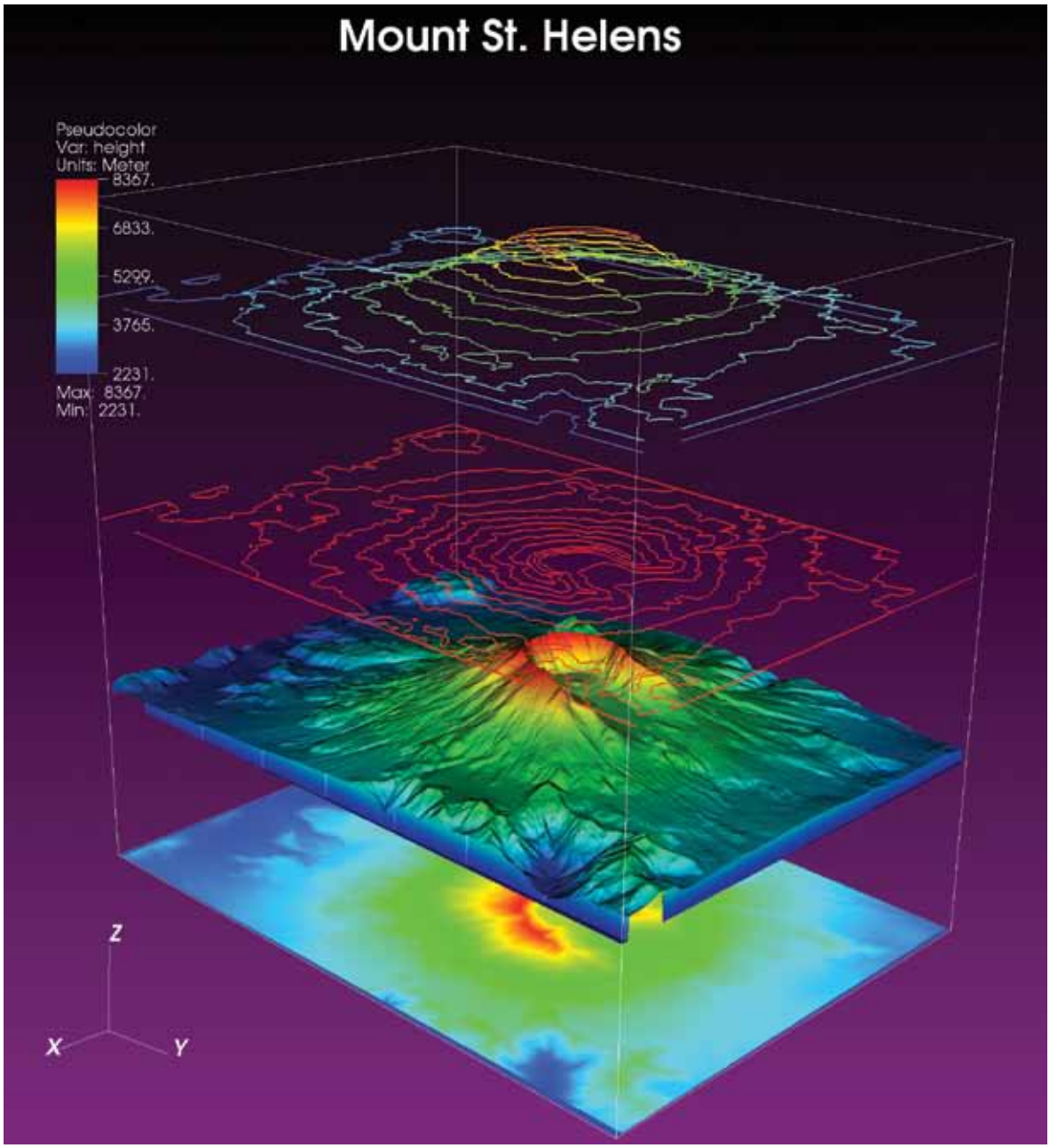


Figure 3. A topographic visualization of Mount St. Helens performed with Lawrence Livermore Lab's VisIt software

MPI4PY additionally provides accessing C/FORTRAN MPI codes (even both in the same process) in Python.

For using GPUs for data parallel applications, PyCUDA provides Python access to the full CUDA API from NVIDIA. This means NVIDIA GPUs can be programmed in Python instead of the native C++ of the CUDA libraries. An important feature of PyCUDA is that it provides automatic error checking and object cleanup that helps eliminate leaks and crashes. These features, as well as the comparative ease of programming in Python as opposed to C++, make PyCUDA an extremely productive way to get started programming GPUs. In fact, NVIDIA itself is internally developing the Python-based "copperhead" framework, which is a subset of Python that automatically produces CUDA code.

Federal Agencies Use Python for Science

Python is used extensively in industry by companies like Google and by many Federal research agencies such as the National Institute of Standards and Technology (NIST), Department of Energy (DOE) , and National Oceanic and Atmospheric Administration (NOAA). For example, Hank Childs from Lawrence Berkeley National Laboratory (LBNL) leads Visualization and Analysis for Very Large Data Sets (VisIt), a visualization and analysis tool designed for processing large data.

Tony Drummond (LBNL) leads PyACTS, a project that provides Python interfaces to the ACTS collection of high-performance codes (Aztec, Hypre, PETSc, SLEPc, ScaLAPACK, SUNDIALS, SuperLU, TAO, and OPT++). Bill Spotz of Sandia National Labs develops the PyTrilinos set of Python interface for over 40 of the Trilinos packages (e.g., phdMESH, Zoltan, PAMGEN, IFPACK). Jeff Whitaker at NOAA sponsors the Basemap visualization toolkit, which provides tools to draw graphics on a wide variety of map projections of the earth, including topological features and boundaries.

Summary

Python has developed into a serious platform for scientific computing and development and is used by industrial and government labs around the world. As an open-source product, Python has zero licensing fees with no license manager hassles. The lack of fees is obviously important for running on HPCs since commercial products usually base fees on the number of cores utilized. Readers interested in learning more about these tools can visit the SciPy Web site (scipy.org), which hosts a number of these tools and contains links and information to many more related projects. Additionally, many DoD Python courses are available on the Online Knowledge Center (OKC) and PETTT User Portal (PUP) sites. Python and related numerical packages are installed as part of the PETTT parallel tools runtime environment package (ptools-rte) on all DoD Supercomputing Resource Center (DSRC) systems in the $PET_HOME/pkgs/ptoolsrte directory. For further information or assistance with Python tools, please contact the author or help@pettt-ace.com.

Acknowledgments

The author would like to thank and acknowledge Dr. Aram Kevorkian (SSC-Pacific) for his seminal support of Python for scientific computing in the DoD and Dr. Sameer Shende (ParaTools, Inc.) for his initiative in making these tools available across the DSRCs.

Investigation of Algorithms for Hybrid Multicore/Many-Core Architectures

By David Richie, Brown Deer Technology; James Ross, High Performance Technologies, Inc.; and Dale Shires and Song Park, U.S. Army Research Laboratory

In support of the U.S. Army Research Laboratory's (ARL) basic and applied research into the use of asymmetric core computing for deployable and nontraditional high performance computing (HPC), researchers within ARL's Computational and Information Sciences Directorate (ARL/CISD) are developing and examining algorithms for hybrid multicore/many-core architectures. The initial investigation has focused on computational kernels representing broad classes of HPC applications, which are studied to assess the potential of this emerging technology and identify software development and optimization strategies for future HPC application development.

Background

General-purpose graphics processing units (GPGPUs) currently provide price/performance metrics for floating-point-intensive applications unmatched by competing technologies. As a result, technology primarily developed for graphics processing has been repurposed for HPC applications with the emergence of architectures comprised of conventional processors and GPGPU co-processors. These many-core processors provide massive chip-level parallelism and enable the construction of large-scale clusters with more cores per node than nodes in the overall system. This represents an inversion of the parallelism familiar to most HPC software developers.

The ability of software developers to reconcile this "parallelism at the bottom" within the traditional HPC software stack will be critical for exploiting these architectures to their full potential. Figure 1 shows a prototypical hybrid multicore/many-core compute node easily assembled using existing technology. The architecture is comprised of dual hexa-core processors coupled with an aggregate of 6400 stream processing cores distributed over four GPGPUs. The challenge for the software developer is programming this heterogeneous architecture, including effectively controlling the asynchronous computation and data transfer across distributed memory. Many-core processors are capable of executing thousands of threads concurrently, requiring careful synchronization and control.

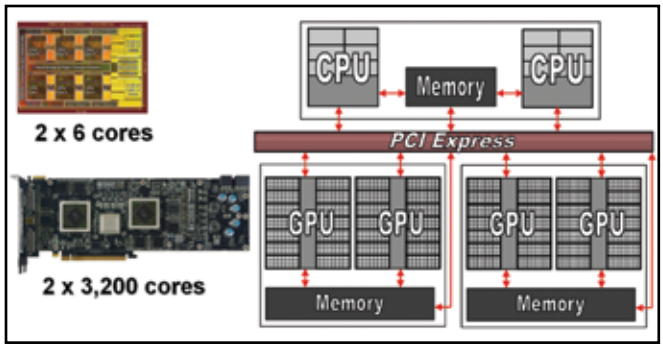


Figure 1. Hybrid Multicore/Many-core Compute Node. Using existing technology, a conventional dual quad-core compute node can be augmented with GPGPU coprocessors providing thousands of processing cores. The complexity of managing the compute and data flow on such a compute node presents challenges for HPC software developers

Investigation

The project investigated a broad range of computational kernels selected to provide sufficient complexity so as to prove nontrivial, yet general enough to cover the scope of core algorithms found in many HPC applications. The selected computational kernels included those representing particle-based simulations, grid-based finite-difference and lattice methods, multidimensional hierarchical search algorithms, and image processing algorithms. The algorithms exhibited different computational scaling, compute/communication ratios, and computational complexity.

The project employed the most relevant application programming interfaces (APIs) available for programming these hybrid multicore/many-core architectures. Specifically, the work employed MPI, OpenMP, explicit Pthreads, SSE, ATI Stream, and Nvidia CUDA. In many cases, these APIs were used in combination, reflecting the challenge of programming for heterogeneous computing platforms. Optimization strategies explored in the work ranged from coarse-grained decomposition across a multinode test cluster to assembly-level optimizations of critical loops on the many-core processors. The performance issues examined ranged from on-chip thread performance to communication latency and bandwidth.

Benchmarking techniques were used to determine the relative performance of the hardware and the efficiency of the implementations, which included techniques developed to address the nontrivial challenges of timing asynchronous operations inherent in these architectures. As an example, techniques were developed to measure relative timing between asynchronous memory transfers and kernel setup and execution on coprocessor devices.

Results

As an example of a particle-based algorithm, a canonical N-body simulation was implemented and benchmarked for both multicore and GPGPU architectures.

The N-body algorithm exhibits $O(N^2)$ compute and $O(N)$ communication requirements such that the size of the system provides a tunable parameter for the investigation of compute- vs. communication-bound problems. A single-GPGPU implementation of the algorithm was extended to perform distributed multi-GPGPU simulations over an eight-node AMD FireStream 9170 test cluster. All data were shared between compute nodes after each time-step. Excellent scaling was observed when the local compute load was great enough to hide the communication. Results showed a sustained performance of 1.4 TFLOPS, greater than 30 percent of the theoretical peak[1] efficiency for the platform.

As an example of a grid-based algorithm, a transmission line matrix (TLM) solver was implemented for a GPGPU that employed multiple grids per GPGPU and allowed the exchange of cell edges between multiple GPGPUs within a small test cluster. The TLM algorithm is generally memory-bound and requires relatively little computation. The code was set up to interlace the communication and computation. For sufficiently large problems where the compute to communication ratio was high, the code scaled nearly linearly over the eight GPGPUs and attained greater than a 130x speedup over a serial CPU implementation. Results from the GPGPU-accelerated TLM solver are shown in Figure 2.

As an example of a tree-based search algorithm, a quad-tree search algorithm was used to support a ray tracing method for calculating the ballistic trajectory field within an urban environment. A "first-hit" search was performed, and a simple fourth-order model was applied to each ray based on distance. The benchmark algorithms are representative of the kind found in applications ranging from ray-tracing-based visualization, wireless path-loss prediction, ballistic effects analysis, and data-analytic applications. The specific implementation of the quad-tree search algorithm was designed to be suitable for use with many-core processors, and specifically required no recursion, no pointers, and the use of a short stack. Compared with an optimized CPU implementation, the GPGPU-accelerated algorithm exhibited a 10x speedup. Results from a benchmark calculation are shown in Figure 3.

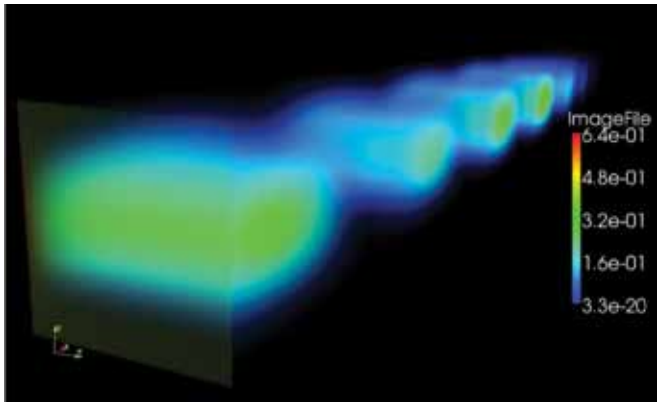


Figure 2. Electromagnetic power from a transmission line matrix (TLM) benchmark run on an AMD FireStream 9170

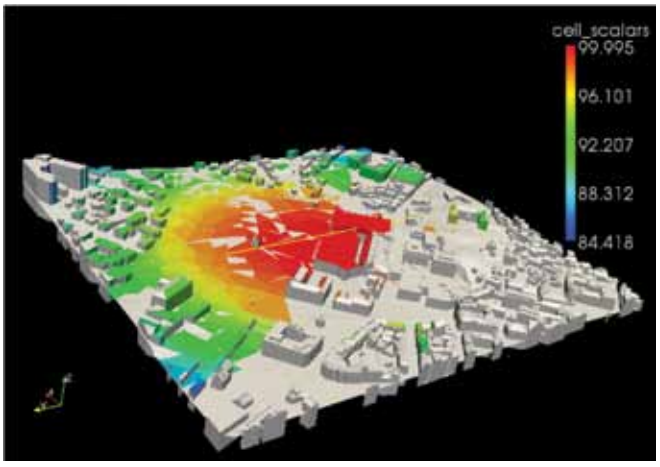


Figure 3. Ballistic Trajectory benchmark calculated on an ATI Radeon HD 4870. Shown is the threat probability from a location near the city center

Results of the project go beyond the specific algorithms developed within the project. The use of hybrid multicore/many-core compute nodes presents significant challenges for the software developer that can only be partially mitigated with tools and compilers targeting this technology. Many assumptions central to the design of HPC software must be revisited—for example, the use of heavy conditionals to reduce floating-point operations. Many-core processors exhibit an extraordinary capability for performing floating-point operations with comparatively weak capabilities for branching. When designing for these architectures, software developers must employ explicit speculative execution at the expense of unnecessary floating-point operations to achieve optimum performance. This is but one example of the lessons that must be learned in developing software for massively parallel multicore/many-core platforms. A central objective of the project is to develop and document these design rules necessary for the development of next-generation HPC software.

Conclusions

The emergence of hybrid multicore/many-core compute nodes as building blocks for future large-scale HPC platforms presents many challenges for addressing this new parallelism at the bottom of the hardware architecture. Many of these challenges are entirely new to HPC and can only be partially resolved with further

[1] The algorithm contains nontrivial floating-point operations such as square root functions, whereas the theoretical peak is based on a precise ratio of multiply-add operations.

development of tools and compilers. This new hardware will impact the design of future HPC software and holds the promise for both increased capability in traditional HPC applications and also entirely new opportunities in the area of deployable/mobile HPC and interactive HPC.

Ongoing efforts of the project include the evaluation of OpenCL[2] as an API for the parallel programming of many-core processors. Reconciling OpenCL within the traditional HPC software stack remains a critical unresolved challenge facing the HPC community. Increased focus in the area of algorithms will include issues of interdevice distributed computations within these heterogeneous architectures, including load-balancing, asynchronous execution, synchronization, and distributed-memory management. The anticipated results from these efforts will be techniques, algorithms, and design rules for future HPC software development on massively parallel multicore/many-core architectures.

[2] OpenCL is an industry standard for parallel programming heterogeneous computing platforms.

DoD impact

The results of the effort will impact DoD software development projects undertaken by the Army Research Laboratory Advanced Computing Laboratory as well as other High Performance Computing Modernization Program initiatives such as the Mobile Network Modeling Institute (MNMI). GPGPU-accelerated architectures are increasingly available to DoD HPC users at both DoD Supercomputing Research Centers and Associated Resource Centers and can be expected to appear in larger systems in the future. Developing programming models and strategies to efficiently use these resources will greatly enhance their effectiveness. This work will impact future DoD software development initiatives by enabling HPC applications to exploit hybrid computing platforms comprised of traditional multicore processors and many-core GPGPU coprocessors. Results from the project are already being used in support of the MNMI to design path loss prediction models that take advantage of GPGPU acceleration to provide greater fidelity in network simulation and emulation codes.

Parallel MATLAB Without Toolbox Licenses Using Standard MPI Implementations

By Peter G. Raeth, Air Force Research Laboratory, User Productivity Enhancement, Technology Transfer, and Training (PETTT) Program, Signal and Imaging Processing (SIP) Onsite; and Juan C. Chaves, Army Research Laboratory, PETTT Program, SIP Onsite

Introduction

MATLAB is widely used in academia, the Government, and industry (more than 1 million users by some estimates). It has emerged as an important tool used by DoD and its contractors in the development of new warfighting capability. The attraction of MATLAB is that it combines an easy-to-use scientific programming language with a common environment for prototyping, coding, and visualization. Traditionally, applications written in MATLAB are oriented to single-processor systems. A tool (bcMPI) now exists that readily enables parallel processing for MATLAB programs. bcMPI provides MATLAB wrappers for calls to industrial-strength, open-source MPI implementations such as MPICH and LAM-MPI. By applying the same parallel processing designs used in C, C++, Fortran, Java, and other popular languages, MATLAB users can achieve considerable throughput improvement.

Improving MATLAB Throughput

Originally written by the Ohio Supercomputing Center (OSC), the authors transformed bcMPI for general implementation and, under the auspices of the User Productivity Enhancement, Technology Transfer, and Training (PETTT) Program, transferred it to the supercomputing systems of the DoD High Performance Computing Modernization Program (HPCMP). This result is independent of operating system, does not require licensing of MathWorks' toolboxes, does not attempt to create another implementation of MPI, and uses network message passing instead of shared files.

To demonstrate bcMPI's effectiveness, we wrote MATLAB code for basic image convolution, assigning a new value individually to each pixel based on the weighted average of neighbors. We applied a 3x3 filter to a 2112x2816 grayscale image, and we applied strong scaling. As the number of cluster processes increased, the application was not changed. We compared *Falcon* (www.afrl.hpc.mil/customer/userdocs/falcon/hpxcguide.pdf), a Linux supercomputer located at the Air Force Research Laboratory DoD Supercomputing Resource Center (AFRL DSRC) (Wright-Patterson AFB), to a common Windows XP-Pro workstation network having eight nodes with two 3 GHz processors

on each node (only one process was run on each node.) The results showed a 3x throughput improvement between networks, without any attempt at optimization (Figure 1). *Falcon* reached a 5x improvement as the number of processes increased.

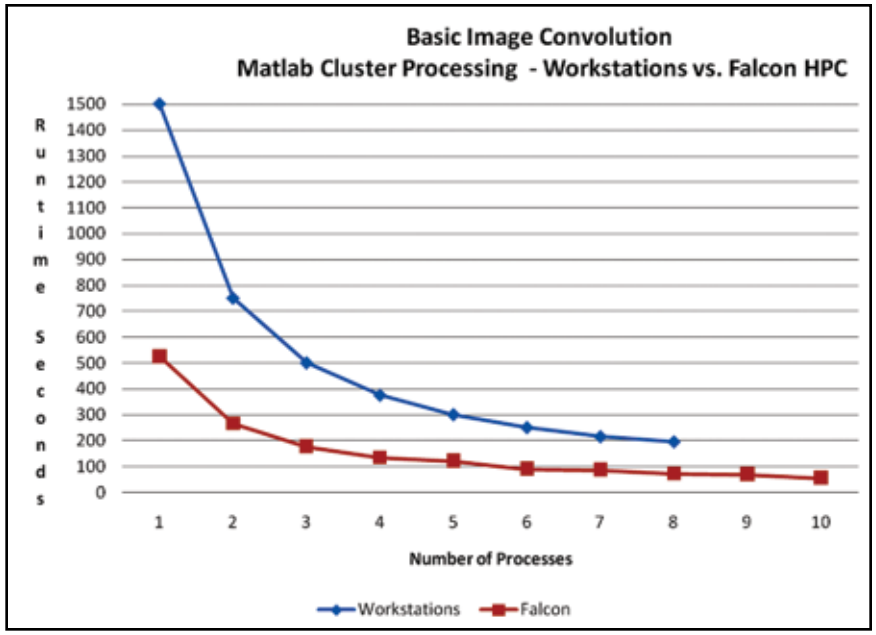


Figure 1. Basic image convolution throughput

The approach we take is illustrated in Figure 2. This approach works directly with the MATLAB graphical user interface (GUI) or with executables created by the MATLAB compiler. In fact, using the GUI, it is possible to step through source code on various processes while the program is running interactively. Wrapping MPI calls in .m files that call .mex files allows us to perform parallel processing directly, without reference to special vendor-specific tool boxes and MPI implementations. It also accommodates MATLAB's pass by value approach to sending variables to external modules.

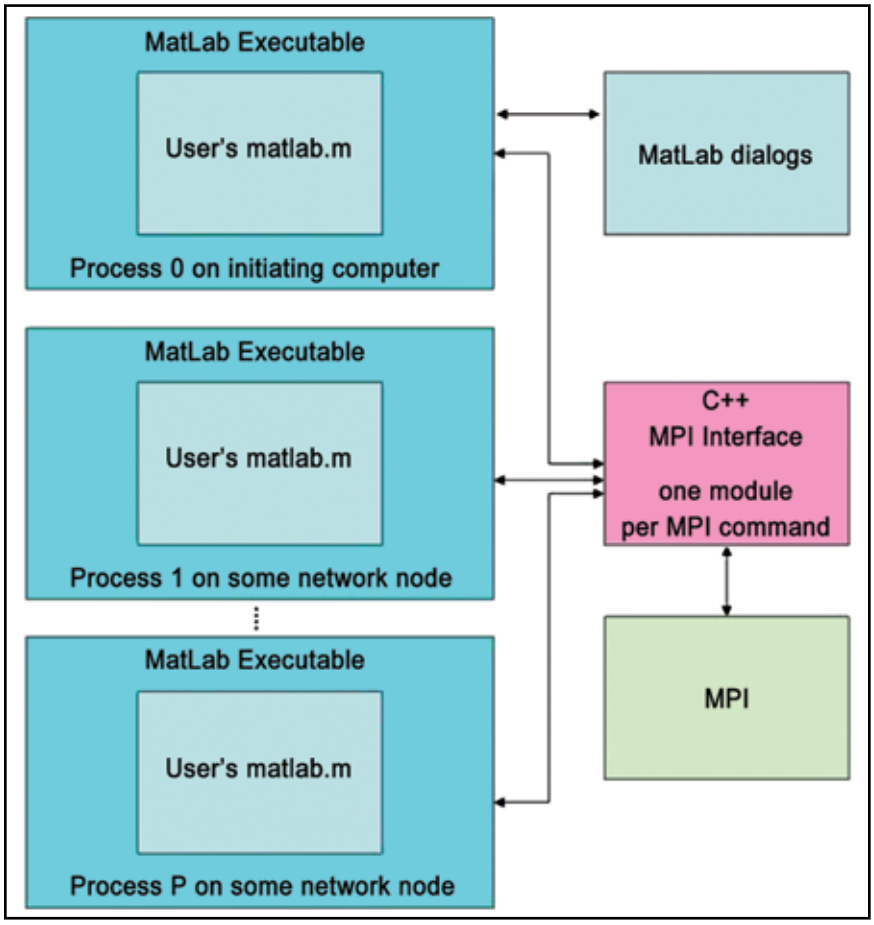


Figure 2. Approach to MATLAB parallel processing

Applications

An important application of bcMPI is currently underway at the Air Force Institute of Technology (AFIT) College of Engineering. They are showing through simulation and experimentation that scene contents not directly visible to either a camera or light source may yet be observed. Objects not directly viewable can be illuminated by light scattered off material in the light's direct path. Present efforts colocate a camera with a laser source to observe hidden scene components. Essentially, AFIT's work seeks to allow us to see around corners without having to physically look. This has many practical implications for urban combat and homeland security.

Present users of MATLAB on DoD HPC systems include the AFRL, Air Force Academy, Army Research Laboratory (ARL), Engineer Research and Development Center (ERDC), Naval Research Laboratory, Coast Guard Academy, and numerous companies supporting DoD weapons systems development.

Strong potential for porting bcMPI to Simulink has been demonstrated using a simple signal dampening application. The results were equally encouraging. With either MATLAB or Simulink, improved throughput makes it possible to go directly to production applications without a translation step. This removes a significant barrier to technology transition. Currently

installed on *Falcon* at the AFRL DSRC (Wright-Patterson AFB, OH), the DoD version of bcMPI can be installed at any DoD HPC Center that supports a modern version of MATLAB. Installation of bcMPI has just recently been completed on *MJM* and *Harold* (ARL) and *Diamond* (ERDC) and is now working on those three machines. This makes the tool freely accessible from anywhere via the user desktop and standard Internet communication. The authors have prepared a succinct user guide and introductory examples.

Acknowledgments

D.G. Adams and Terry McClure of the Consolidated Customer Assistance Center (CCAC, http://www.ccac.hpc.mil) were most helpful in the development of this new capability. There were many configuration issues to understand, and CCAC's pro-active expertise made all the difference.

About the Authors

Dr. Peter Raeth is a senior staff scientist in the Advanced Systems Group at High Performance Technologies, Inc. (HPTi). He is a PETTT Signal and Imaging Processing (SIP) Onsite at the AFRL DSRC supporting DoD HPC users in several domains, including computational fluid dynamics, laser reflection, and image processing. Publications include topics in sensor data processing, anomaly detection in streaming data, adaptive systems, and computational throughput improvement. He holds a Ph.D. from Nova Southeastern University.

Dr. Juan Carlos Chaves is a senior staff scientist in the Advanced Systems Group at High Performance Technologies, Inc. (HPTi). He is the PETTT Signal and Imaging Processing (SIP) Onsite at the ARL DSRC and the computational technology area lead. He has expertise in computational physics and extensive experience in high level languages (HLL), HPC productivity, and in the development of innovative solutions for complex DoD problems. He has provided PETTT training in parallel MATLAB and has been a regular contributor to the HPCMP Users Group Conferences. His current research interests are virtualization technologies applied to the solution of SIP problems on HPC systems, parallel and distributed computing, and SIP computational problems.

Launch of the Storage Initiative

By Reid Bingham, Army Research Laboratory DoD Supercomputing Research Center, Storage Initiative LM Project Manager

Background

In 2006 and 2007, the High Performance Computing Modernization Program (HPCMP) recognized that it was facing a growing budget-share competition between data storage requirements and the refresh of its computational inventory.

The Program took this challenge to the customer base and established a 140 percent compound annual growth rate (CAGR) ceiling with the condition that larger temporary data storage space (working space) and more robust data management tools become available for use by the customers.

To address the storage requirements of the HPCMP, work by a multi-Center Storage Initiative (SI) team commenced. Significant elements of the storage solution include Storage Lifecycle Management (SLM) tools, a data archive capability, a centerwide filesystem (CWFS) and the Common Utility Enhancement Services (CUES) interactive data analysis environment (also known as "Utility Server").

Launch

The SI implementation was launched in August 2009 with a contract awarded to General Atomics for the SLM and data Archive components. Subsequent to the contract award, the SI team expanded to include experts from the DoD Supercomputing Resource Centers (DSRCs), General Atomics and its partners, and Lockheed Martin, the technical services provider for the DSRCs. The team has since been working to design and implement the new storage capabilities. The following provides some insight into each of the components.

Storage Lifecycle Management Solution

The General Atomics COTS software is Nirvana Storage Resource Broker™ (SRB). SRB is in a category of software known in industry as digital asset management.

The primary interaction of the HPCMP customers with SRB will be via a command-line and/or a Java-based graphical user interface (GUI) allowing customers to view and manipulate their data inventory. The capabilities of SRB will be used to associate and maintain customer and system data attributes (metadata) relevant to the information contained in the files and/or the executable that generated the files. The most significant customer-visible capabilities of SLM include the following :

- Collections – The SRB equivalent of a directory is a Collection. Collections are views of the user's data inventory.

- Global namespace – A global, or federated, namespace is implemented by replicating the SRB metadata among all six Centers in the "federation." Users will be able to see all of their archived data across all Centers in one set of Collections.
- Federation customer identity – the global namespace construct also facilitates a customer having an identity at the "federation" level that is quasi-independent of the user accounts on individual servers and HPC systems.
- Files as data objects – an SRB data object has a single identity within the global name-space. For example, a customer can identify an object for residency at multiple DSRCs. The object can then be managed centrally as if it were one file. This feature will be particularly useful to customers that do business at multiple Centers but also have a preferred "home" Center for data archiving.
- Federation data – Data will now be federated across all DSRCs via the federated customer identity, the global namespace, and the files-as-data-objects feature. The DSRCs will coordinate their local policies for user account lifecycle and user data lifecycle.

Data Archive

Data archiving capability is similar to the way it has historically been in the DSRCs except that the data will be registered and can be managed by SRB. For example, the customer can assign metadata to an archived file indicating when it should be deleted. When that date arrives, it will automatically be deleted. Additionally, customers can query on other metadata attributes, e.g., project name and test case number, to easily identify what they are searching for.

Center-wide Filesystem (CWFS)

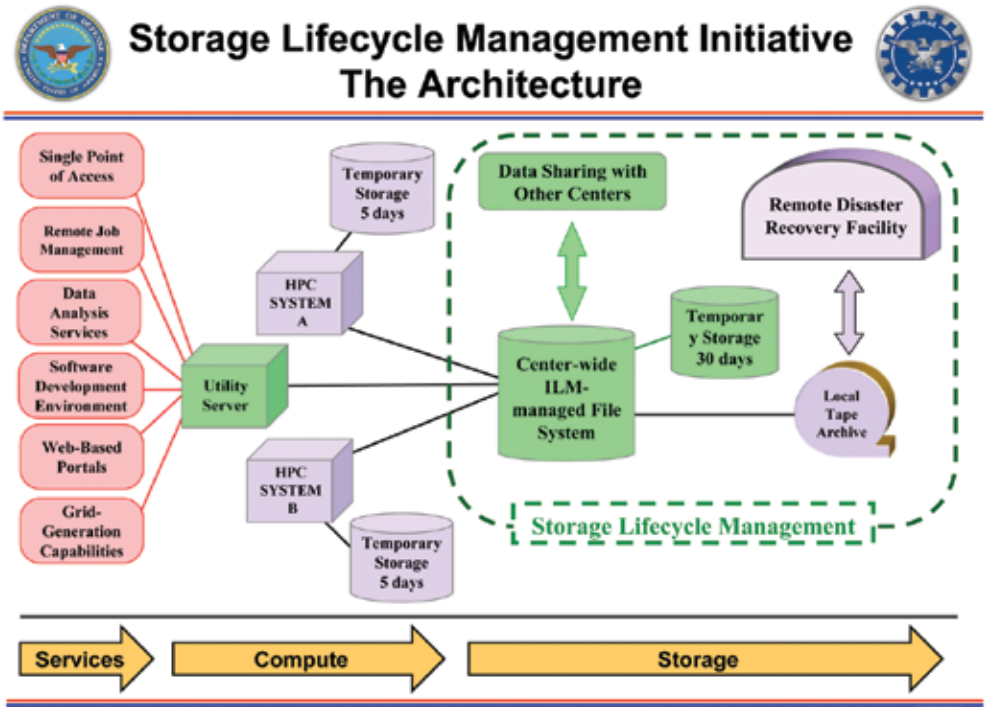
The current time-to-live workspace is approximately 5 days. The Center-wide filesystem at each Center is intended to provide a 30-day time-to-live. This should provide customers adequate time to analyze and prune their data prior to deleting it or committing it to the SLM archive.

Common Utility Enhancement Services (CUES)

The CUES is a large Linux-based system with access to the CWFS. This "engine" will provide the customer with a pre- and postprocessing environment local to their data (the CWFS). Services planned for early availability on the CUES include remote scientific visualization capabilities and remote job management. Services envisioned for the future include grid generation tools, software development tools, single sign-on and Web-based portals.

Putting the Pieces Together

The figure below shows a model envisioning a typical DSRC's access, compute, and storage environment. Input from industry, academia, other National laboratories, the DSRC staff, DREN staff, the User Advocacy Group, and users in general have shaped this new architecture for the DSRCs. The goals are improved user productivity and better management of the volumes of data the HPCMP stores. Achieving this is a team effort, and we look forward to working with the user community to mature it to best address their requirements.



New Web Site Unveiled to Guide Allocation Requests with TI-10 Benchmarking Times

By Laura Brown, Dr. Paul Bennett, Mark Cowan, Carrie Leach, and Dr. Tom Oppe, U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center (ERDC DSRC)

Every year the High Performance Computing Modernization Program (HPCMP) uses quantitative performance analysis results from the Application Benchmark Test Suite to evaluate vendor proposals submitted in the annual Technology Insertion (TI) process. The most recent insertion was TI-10. During the TI process, a suite of applications—a small cross section of all applications used on HPCMP machines—is run on all major systems within the program. Over the course of the TI cycle, it usually becomes clear that certain codes within the suite perform more efficiently on specific architectures. Our new Web site (URL: http://www.benchmarking.hpc.mil/) presents rankings for each architecture, based on its performance with each individual code in the TI-10 suite. The architectural rankings provided there are intended to help HPC users choose the most appropriate architecture given a target problem and are based on performances observed for the following TI-10 application test cases: Adaptive Mesh Refinement (AMR) Standard/Large, Air Vehicles Unstructured Solver (AVUS) Large, CTH Standard/Large, General Atomic and Molecular Electronic Structure System (GAMESS) Standard/Large, HYbrid Coordinate Ocean Model (HYCOM) Standard/Large, Improved Concurrent Electromagnetic Particle In Cell (ICEPIC) Standard/Large, and Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) Standard/Large.

Short Description of TI-10 Application Codes

AMR, an Eulerian fluid dynamics code, performs an algorithm on either adaptive or structured grids to solve hyperbolic equations modeling the impact of a gas-dynamic shock contacting a helium bubble. This C++ and FORTRAN code achieves parallelization through the use of MPI libraries and generates substantial amounts of I/O in the form of output binary files. The standard test case simulates a gas-dynamic shock contacting a helium bubble and calculates 80 time-steps with two levels of refinement. The large test case is similar and calculates 40 time-steps with four levels of refinement.

AVUS, another Eulerian fluid dynamics code, was developed by the Air Force Research Laboratory to calculate the 3-D viscous fluid flow and turbulence around air vehicles and projectiles. In general, this code reads in a large, unstructured grid file and uses ParMETIS to partition and perform computations on the grid. The AVUS large test case calculates 400 time-steps for fluid flow and turbulence over a grid numbering in excess of 31 million cells that models a supersonic/hypersonic vehicle "riding" a shock wave that forms below the vehicle. AVUS uses MPI to achieve parallelization and generates large amounts of I/O through process MPI-0. AVUS is an export-controlled code.

CTH, a structural mechanics code, was developed by Sandia National Laboratories to model complex, multidimensional scenarios involving large deformations or strong shock physics. This code uses second-order accurate Eulerian algorithms to solve mass, momentum, and energy conservation equations and has both static and adaptive mesh capabilities. CTH uses MPI for parallelization and uses NetCDF to aid with data array access. Both the standard and large test cases simulate a ten-material metal rod striking an eight-material plate at an oblique angle. The standard case uses AMR with six refinement levels, while the large case uses a fixed grid size with static decomposition. Both test cases generate large amounts of I/O, as each case outputs a 2- to 4-GB binary output file. Also, CTH writes one (approximately 10 MB) restart file for every MPI process during execution. CTH is also an export-controlled code.

GAMESS is an *ab initio* quantum chemistry code that calculates energy integrals using atom positions, electron orbitals, and elliptic equations. The code is currently maintained by Ames Laboratory and Iowa State University. The standard test case performs a density functional theory (DFT) computation to compute the nuclear gradient vector of a POSS (Polyhedral Oligomeric Silsesquioxane) molecule using Restricted Hartree-Fock calculation with self-consistent field wave functions. The standard test case is a sequence of iterations consisting primarily of three steps: (a) computation of approximately N^4 two-electron integrals, (b) diagonalization of a dense NxN matrix, and (c) evaluation of the gradient vector, which involves evaluation of two-electron integral gradients (similar to what is done in step (a)). Here N is equal to the number of basis functions in the calculation. Steps (a) and (b) are repeated until convergence is achieved, after which step (c) is done. The large test case similarly performs an MP2 computation that finds the nuclear gradient vector of a BC4 molecule using the same calculation method. Many of the initial steps in the large test case processing are similar to those in the standard test case; however, following convergence

(step (b) above), there is a rather complicated integral transformation step, which has an $O(N^5)$ computational requirement. This case uses 1.8 GB of memory per core, which makes it much more memory-intensive than the standard test case, which uses approximately 800 MB per core. Conversely, the standard case requires a significantly large amount of scratch space (9.7 GB versus the approximately 350 MB required for the large case). The communication protocol used (e.g., MPI, SHMEM, LAPI, sockets) to achieve parallelization is platform-dependent.

HYCOM was developed by the Naval Research Laboratory, Los Alamos National Laboratory, and the University of Miami as a structured grid-based ocean modeling code. The code makes use of a direct linear solver and LU factorization to solve linear systems. HYCOM uses MPI to achieve parallelization, although it can also accommodate MPI-2. The standard test case is a 26-layer, 24-hour model that represents the oceans as one global body of water with one-fourth-degree resolution and calculates 2160 time-steps. It uses approximately 0.75 GB of memory per processor, but generates large amounts of I/O (4 GB of scratch files and 11 GB of binary output files). The large test case is a similar global model that represents the oceans over an 18-hour period with one-twelfth-degree resolution. This test case calculates 864 time-steps. This case also generates large amounts of I/O (23 GB of scratch files and 23 GB of binary output) and uses approximately 0.9 GB of memory per processor.

ICEPIC, an Eulerian electromagnetics code, was developed by the Air Force Research Laboratory Directed Energy Directorate as a Particle-in-Cell plasma physics simulator. In this code, ions and electrons move under the influence of electromagnetic fields. These fields are calculated using Maxwell's equations on a structured, static grid. Both the standard and large test cases simulate electron motion and electromagnetic fields in a complex microwave device. The standard case calculates 2.2 million particles and 25 million cells in approximately 6000 time-steps. The large case calculates 12 million particles and 200 million cells in 9900 time-steps. Both cases use MPI to achieve parallelization and generate relatively little (150-300 MB) I/O. ICEPIC is an export-controlled code.

LAMMPS is an open-source classical molecular dynamics code that uses spatial domain decomposition to simulate molecular and atomic systems. This code uses MPI to achieve parallelization and generates relatively little I/O. It is currently being maintained by Sandia National Laboratories. The standard test case is an adaptation of the LAMMPS Rhodopsin Benchmark Model, an all-atom Rhodopsin protein in a solvated lipid bi-layer, which simulates 10,976,000 atoms in 1500 time-steps. This case makes use of FFTW, a library of subroutines that calculates multidimensional discrete Fourier transforms. The large test case is an adaptation of the Embedded Atom Model simulating a metallic copper solid with 108 million atoms in 2500 time-steps.

Of course, these codes are not the full complement of codes that run on HPC machines; however, they are chosen to be a representative sample of codes seeing daily use. They have been carefully selected by members of the user community to reflect average and stressful computational burdens similar to commonly run tasks across most of the computational technology areas (CTAs). Attempts are always made to remove or reduce significantly any functional redundancies between applications.

Benchmarking Methodology

Thirteen test cases (two per application except for AVUS, which ran with only the large test case) were executed at several processor counts on nine HPCMP systems. Times were recorded at predefined processor counts (typically, 64, 128, 256, and 384 for the standard case; 256, 512, 1024, and 1280 for the large). Using these times, a performance curve was constructed using a least-squares approach with a power-law model to compare individual performance with the performance observed on the DoD TI-10 baseline benchmarking system—the U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center (ERDC DSRC) Cray XT3 containing 2.6 GHz AMD Opteron processors under Catamount with the Cray SeaStar interconnect—at 1024 processors (regardless of the number of nodes actually present on the system under test). The full TI-10 test suite was executed on ERDC's recent acquisition—the SGI Altix ICE, code named DIAMOND. The relative performance of the applications for both the standard and large test cases are included on the Web site, along with the official TI-10 numbers.

All benchmarking executions were performed by the Computational Science and Engineering (CS&E) benchmarking team seated at the ERDC DSRC. The original purpose of this effort was to evaluate the performance of a quiescent system to eliminate the possibility of runtime variations resulting from system load at the time of execution. These systems, however, were in active use at the time the benchmarking runs were completed and could not be made quiescent. Therefore, for each application test case and architecture pair, the best time/performance datum from multiple executions was recorded under the theory that the best time was the most accurate estimate of the time for a dedicated execution on a quiescent system. The architectures

were then ranked for each application test case to highlight the relative suitability of individual architectures in executing different applications. It should come as no surprise that machines obtained under the more recent acquisition cycles demonstrate the best performance, as the vendors were challenged to outperform times from earlier platforms as part of the acquisition process. It is suggested to users that they try to determine which application in the test suite is most similar to their commonly executed programs. Then a quick review of the performance matrix on our new Web site should provide a strong indication of where an HPC user's allocation is most suitably housed.

For further information, visit our new Web site at http://www.benchmarking.hpc.mil/

Tuning Application Performance on the Cray XT Without Modifying the Code

By Dr. Thomas C. Oppe, U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center (ERDC DSRC)

Code optimization usually offers the greatest potential for improving runtime performance, but much can be done to improve a code's performance by changing the way the code is compiled and run. An experiment was conducted on *EINSTEIN,* the Cray XT5 at the Navy DoD Supercomputing Resource Center (DSRC), to explore the effects of using different compilers, using large TLB (translation look-aside buffer) page sizes, and different process placement strategies to improve the performance of six application codes taken from the Technology Insertion 2010 (TI-10) Application Benchmark Test Package. The codes used in this study are AMR, AVUS, CTH, HYCOM, ICEPIC, and LAMMPS, and each code was run with two data input sets, denoted "standard" and "large." The codes were compiled with two compilers, PGI (version 9.0.4) and PathScale (version 3.2.99), with the PGI options being

```
-fastsse -O3 -tp barcelona-64
```

and the PathScale options being

```
-O3 -OPT:Ofast -fno-math-errno -ffast-math
-funsafe-math-optimizations -march=barcelona
-mtune=barcelona -mcpu=barcelona -m64
```

The executables were linked using either the default size TLB pages or with huge TLB pages using the library "`-lhugetlbfs`" and the environment variables

```
export HUGETLB_MORECORE=yes
export MALLOC_TRIM_THRESHOLD_=1073741824
export MALLOC_MMAP_MAX_=0
```

and starting the MPI job with

```
aprun -m700mh -n <procs>  <executable>
```

The codes were run in batch mode under PBS with two process placement methods for assigning MPI processes to cores, namely the default "SMP-style" that places processes with consecutive rank numbers on nodes and a round-robin style that distributes consecutive processes across nodes. Each compute node on *EINSTEIN* contains two quad-core AMD Opteron processors and thus has eight cores. An example of an SMP-style distribution and a round-robin distribution of 24 processes on three nodes is shown below

Node	SMP-Style	Round-Robin
1	0 1 2 3 4 5 6 7	0 3 6 9 12 15 18 21
2	8 9 10 11 12 13 14 15	1 4 7 10 13 16 19 22
3	16 27 18 19 20 21 22 23	2 5 8 11 14 17 20 23

The environment variable `MPICH_RANK_REORDER_METHOD` is set to 0 for round-robin and either not set or set to 1 for the default SMP-style. In addition to these environment variables, all runs set the following environment variables to prevent the writing of core files, to display the MPI process placement, to display the version of MPICH used (version 4.0.1), and to prevent the printing of FORTRAN stop messages to improve scalability, respectively:

```
export CORE_ACTION_FIRST=KILL
export CORE_ACTION_OTHER=KILL
export MPICH_RANK_REORDER_DISPLAY=1
export MPICH_VERSION_DISPLAY=1
export NO_STOP_MESSAGE=1
```

All codes were compiled in pure MPI mode so that no OpenMP functionality in any of the codes was used. The elapsed times in seconds are given in Tables 1-3 with the minimum time indicated in red. The three columns under each compiler indicate the use of small TLB pages with the default SMP-style process mapping ("sml"), the use of large TLB pages with SMP-style process mapping ("lrg"), and the use of large TLB pages with a round-robin process mapping ("lrg-rr"). Since the use of large pages was generally beneficial, the runs using round-robin were limited to the executables linked with large pages. To reduce the presence of run-to-run variation in the times due to running in a production environment, each run was repeated at least once with the reported time being the minimum time achieved. The process counts used for each code and test case were the same with the exception of HYCOM, for which only a limited set of input files existed.

Table 1. AMR and AVUS Timings (seconds)

Code	Case	Procs	PGI			PathScale		
			sml	lrg	lrg-rr	sml	lrg	lrg-rr
AMR	std	64	9733	7767	8449	8524	7030	7302
	std	128	4922	4033	4151	4367	3684	3770
	std	256	2553	2101	2125	2265	1914	1920
	std	384	1784	1486	1462	1562	1330	1350
	std	512	1396	1174	1128	1229	1045	1020
AMR	lrg	256	8004	6592	6639	7051	5896	6053
	lrg	512	4241	3509	3468	3707	3149	3112
	lrg	768	2993	2515	2449	2638	2239	2219
	lrg	1024	2393	2008	1936	2089	1782	1702
	lrg	1280	2013	1695	1610	1757	1508	1425
AVUS	std	64	7710	7097	7086	7362	6961	6954
	std	128	3836	3528	3534	3649	3467	3478
	std	256	1901	1744	1740	1815	1720	1717
	std	384	1274	1169	1165	1220	1151	1153
	std	512	967	883	883	930	877	885
AVUS	lrg	256	6692	6273	6345	6586	6351	6422
	lrg	512	3494	3278	3324	3480	3343	3402
	lrg	768	2401	2253	2273	2423	2327	2357
	lrg	1024	1847	1737	1757	1883	1815	1849
	lrg	1280	1543	1452	1483	1585	1532	1569

Table 2. CTH and HYCOM Timings (seconds)

Code	Case	Procs	PGI			PathScale		
			sml	lrg	lrg-rr	sml	lrg	lrg-rr
CTH	std	64	6099	5877	6250	6031	5797	6209
	std	128	3325	3092	3269	3288	3054	3240
	std	256	1766	1644	1802	1752	1637	1788
	std	384	1255	1157	1279	1244	1152	1266
	std	512	933	881	985	934	880	981
CTH	lrg	256	5494	5173	4943	5529	5245	5008
	lrg	512	3125	2913	2737	3172	2934	2751
	lrg	768	2283	2104	1974	2301	2135	2006
	lrg	1024	1814	1667	1564	1829	1677	1591
	lrg	1280	1613	1399	1351	1555	1423	1362
HYCOM	std	59	6917	6908	7179	7105	7038	7154
	std	124	3340	3335	3412	3508	3425	3489
	std	250	1696	1685	1747	1746	1744	1872
	std	501	935	916	1029	951	936	1036
	std	1020	561	571	672	565	548	653
HYCOM	lrg	256	5225	5231	5234	5443	5443	5447
	lrg	504	2744	2716	2802	2793	2789	2849
	lrg	766	1997	2004	2054	1994	1995	2066
	lrg	1006	1600	1591	1625	1576	1583	1612
	lrg	1267	1322	1313	1375	1291	1273	1321

Table 3. ICEPIC and LAMMPS Timings (seconds)

Code	Case	Procs	PGI			PathScale		
			sml	lrg	lrg-rr	sml	lrg	lrg-rr
ICEPIC	std	64	8877	8652	8753	8029	7904	7955
	std	128	5784	5651	5707	5262	5103	5127
	std	256	4086	3931	3944	3796	3612	3617
	std	384	3314	3146	3170	3009	2936	2935
	std	512	2897	2833	2914	2738	2659	2712
ICEPIC	lrg	256	10649	10480	10396	9468	9347	9289
	lrg	512	5772	5607	5567	5114	4963	4952
	lrg	768	4073	3937	3951	3625	3498	3514
	lrg	1024	3152	3050	3083	3013	2720	2742
	lrg	1280	2621	2552	2569	2337	2257	2294
LAMMPS	std	64	10015	9683	9742	9592	9146	9236
	std	128	5152	4887	4995	4843	4648	4733
	std	256	2516	2454	2491	2386	2318	2366
	std	384	1729	1638	1700	1670	1553	1595
	std	512	1292	1291	1312	1226	1220	1239
LAMMPS	lrg	256	5212	4978	5036	5164	4951	5017
	lrg	512	2618	2555	2559	2606	2523	2541
	lrg	768	1766	1719	1724	1770	1700	1712
	lrg	1024	1295	1252	1276	1292	1244	1265
	lrg	1280	1048	1014	1031	1041	1006	1032

It was found that the choice of compiler, TLB page size, and the process placement strategy can each have a significant effect on the performance of particular codes. However, there was no single environment that benefitted all the codes for both test cases. The PathScale compiler generated faster executables except for AVUS_lrg, CTH_lrg, and HYCOM. The use of large pages helped all codes except possibly for HYCOM, for which many times were so close that it is difficult to draw conclusions. The use of a round-robin process placement strategy benefitted only AMR_lrg, CTH_lrg, and a few of the ICEPIC runs.

The information gained in this study can help the DoD DSRC user compile and run these codes for more efficient use of DoD HPC resources, as well as suggest relatively easy optimization techniques that may benefit other codes.

HPCMP Sustained Systems Performance Test

By Dr. Paul M. Bennett, U.S. Army Engineer Research and Development Center DoD Supercomputer Research Center (ERDC DSRC)

The sustained systems performance (SSP) test has been implemented on High Performance Computing Modernization Program (HPCMP) HPC systems comprising 2000 processing elements or more in order to quantitatively evaluate updates to system software, hardware repairs, job queuing policy modifications, and revisions to the job scheduler as necessary. The test incorporates codes used in the system acquisition cycle selected for proven migration capability to HPCMP HPC systems and nonempirical tests for numerical accuracy. Metrics such as code compilation times, queue wait times, benchmark job execution times, and total test throughput times are gathered and compared against metric data from previous tests to monitor the systems under test while minimizing impact to the users. Jobs failing to execute properly or executing in anomalously short or long times are investigated, and the results are reported to systems administrators and Center directors at each Center for appropriate actions.

The DoD Supercomputing Resource Centers (DSRCs) rely on SSP testing to monitor the quality of updates. For example, in accordance with direction from the HPCMP, the Air Force Research Laboratory (AFRL) DSRC implemented the PBS job scheduler on the SGI Altix 4700, with initial installation on three nodes. The AFRL DSRC requested that the SSP test performed at the close of FY2009 be conducted on those three nodes to test the implementation. Although most test cases performed as expected, one test case behaved oddly. Diagnosis of the cause led to detection of problems in the configuration of PBS, which were corrected prior to implementation of PBS systemwide. As another example, to check on the quality of updates to compilers and numerical libraries in early FY2010, the ERDC DSRC requested an SSP test on the Cray XT4 in Vicksburg, Mississippi. The test demonstrated improved performance for most SSP codes and detected no severe performance issues.

The frequency of the SSP test on systems procured in Technology Insertion 2009 (TI-09) and going forward has increased, with attendant changes in the number of test cases comprising the test. Specifically, while the SSP results will continue to be reported quarterly for all HPC systems under test (SUT) currently, all systems procured in TI-09 and later will be tested monthly. The constituent SSP codes for these systems number three instead of five as in the SSP-07 and SSP-08 suites, and there are only five test cases instead of ten. However, there are two new test cases that use synthetic benchmarks to monitor OS jitter and cache and memory bandwidth rates on core.

The SSP test continues to play an important role in monitoring the quality of HPC service that the HPCMP delivers to DoD users programwide. The results are used by the system administrators of the SUTs, the DSRCs themselves, and the system vendors to monitor the quality of HPC system updates and to perform corrective actions on the SUT as necessary.

MPI IO on the Cray XT Series

By Carrie Leach, U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center (ERDC DSRC)

"MPI IO" is the common name for the parallel I/O offered in MPI-2. In MPT, the Message Passing Toolkit, there are settings available to tune MPI IO to the Cray XT series and its Lustre file system.

Of particular interest for tuning MPI IO is an environment variable MPICH_MPIIO_CB_ALIGN that controls which algorithm is used to divide the I/O workload. The I/O is spread to aggregators. There are three values to choose for collective buffering:

0: Physical I/O boundaries and Lustre striping are disregarded. Each aggregator simply has its equal portion of the I/O workload.

1: As above, Lustre striping is disregarded. However, this algorithm tries to align the I/O requests (based on size) to the physical I/O boundaries themselves.

2: The I/O workload is divided into chunks to match the size of the Lustre striping, causing the same chunks to be paired with the same aggregators for every I/O access.

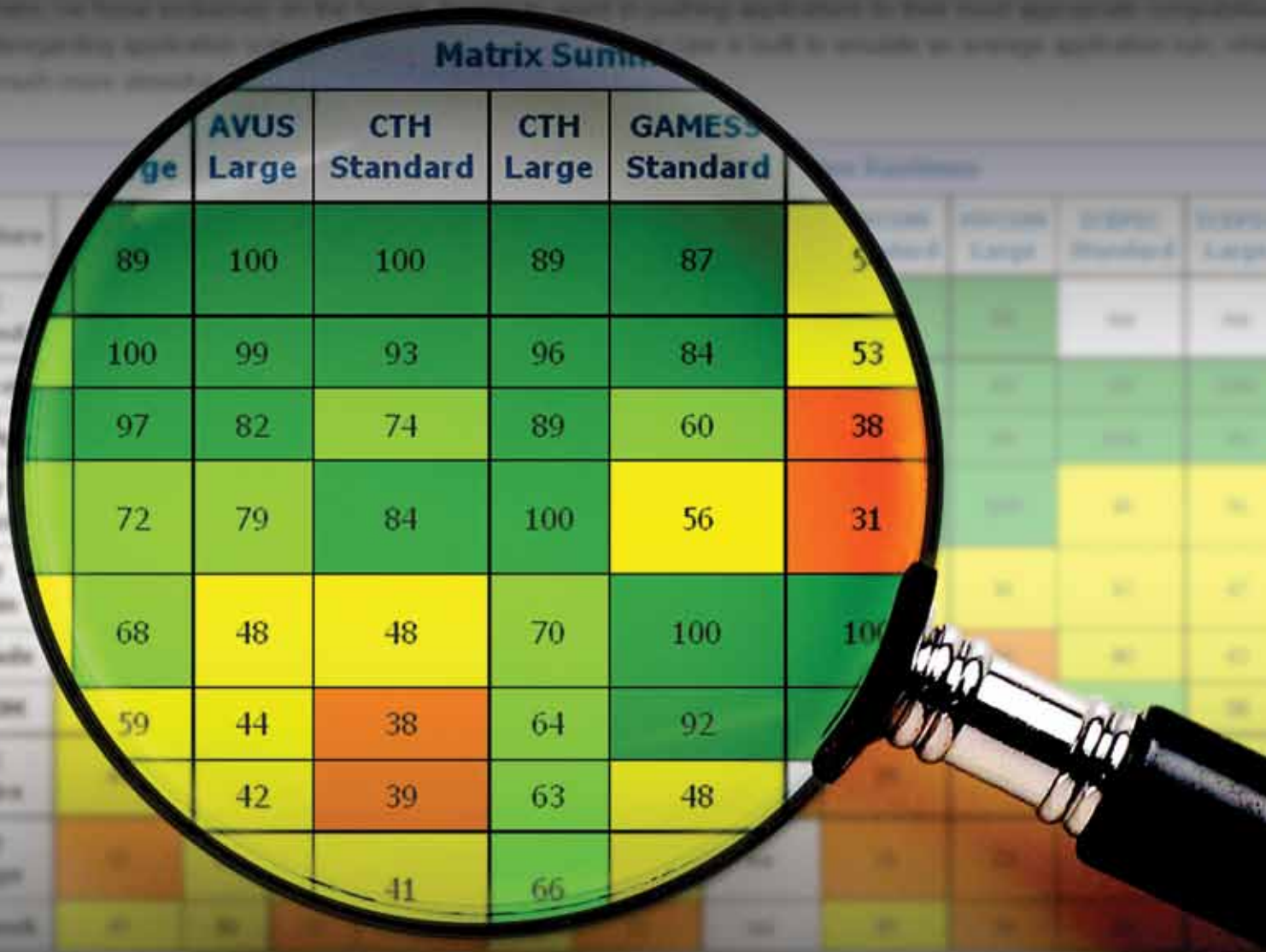
In previous versions of MPT, the default for the MPICH_MPIIO_CB_ALIGN environment variable was 0. With the current version of MPT, 2 is the default.

HYCOM, the HYbrid Coordinate Ocean Model, utilizes MPI IO. The TI-10 standard and large test cases were run at target core-counts on *Jade,* the Cray XT4 at the ERDC DSRC. These preliminary results show the efficiency of simply setting this environment variable to the appropriate value, 2. Values shown below are benchmark times in seconds. For further information, please see the Cray document "Getting Started on MPI I/O" at http://docs.cray.com.

MPICH_MPIIO_CB_ALIGN	2	1	0	n/a (no parallel I/O)
Std. test case, 250 cores (sec)	1906	1982	2315	1973
Lrg. test case, 1006 cores (sec)	1904	1968	1955	2031

Announcements

Making the Most of Your Allocation



- House your HPC allocation where it fits best
- We compare the codes--we compare the machines
- You decide!

www.benchmarking.hpc.mil

DoD High Performance Computing Modernization Program

SOLVING THE HARD PROBLEMS

SUPERCOMPUTING FOR THE WARFIGHTER

Accelerate development and transition of advanced defense technologies into superior warfighting capabilities by exploiting and strengthening US leadership in supercomputing, communications, and computational modeling.

2010 UGC Edition