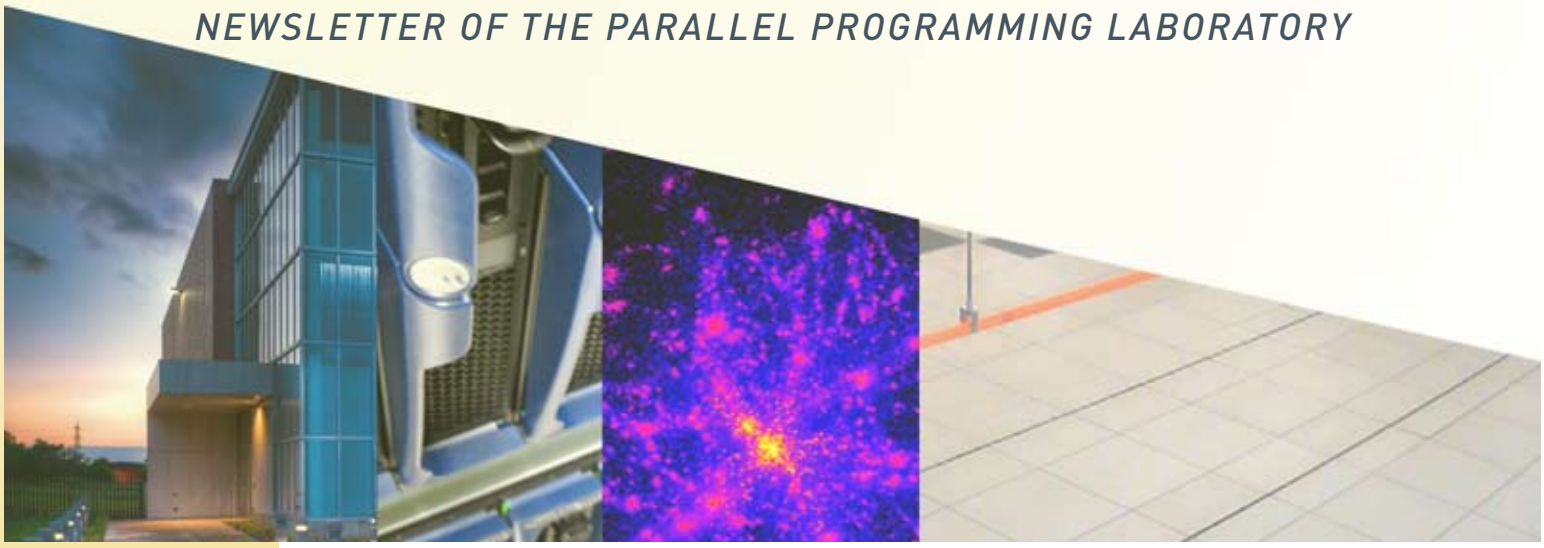
CS@ILLINOIS

Parallel Views

NEWSLETTER OF THE PARALLEL PROGRAMMING LABORATORY



PPL & Blue Waters

In August 2007, NSF announced plans to fund the development of "the world's most powerful leadership class supercomputer". Three classes of applications were identified to assess the performance of candidate systems: a molecular dynamics application, a lattice QCD application, and a turbulence application. For molecular dynamics, NSF stated the required performance levels in terms of NAMD (see page 4 for details), thus acknowledging the importance of NAMD to the HPC community. As co-developers of NAMD, this decision cemented the involvement of the Parallel Programming Lab (PPL) in the Blue Waters project.

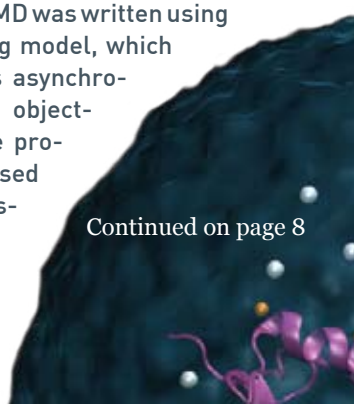
With the immense computational resources of the Blue Waters system, NAMD will enable the simulation of increasingly complex molecular systems at the atomic scale. At the time NSF put forth its initial solicitation for hardware designs in 2006, molecular systems containing between 1 and 3 million atoms were considered large. Currently, NAMD is simulating molecular systems with over 20 million atoms. When the Blue Waters supercomputer becomes operational, scientists will be able to simulate particle systems with over 100 million atoms.

As simulations scale to hundreds of thousands of processors, new performance bottlenecks begin to emerge. For example, the long range electrostatic calculation phase can become a bottleneck with its many-to-many communication pattern. To combat this issue, we are exploring a new multi-level summation algorithm for scaling long distance electrostatic calculations. We are also enhancing NAMD to take advantage of the collective communication operations supported by the Blue Waters interconnection network.

In addition to scaling the electrostatic calculation in NAMD, PPL is also tuning the performance of other portions of the application. With molecular systems on the scale of 10s or 100s of millions of atoms, all aspects of the program must be parallelized so that they are sufficiently scalable. For instance, parallel I/O is required to speed up the startup code and the data output portions of NAMD. The serial sections of the code are also being modified so they can take advantage of specific features of the Power 7 processor, such as support for the VSX SIMD instruction extension.

PPL's contribution to the Blue Waters project is not limited to NAMD alone. NAMD was written using the Charm++ programming model, which provides features such as asynchronous communication and object-based virtualization to the programmer. The model is based on an efficient runtime sys-

Continued on page 8



From The Director's Desk

It is with great pleasure and pride that we present the first newsletter from PPL.

It was approximately twenty years ago that I decided to call my research group the "Parallel Programming Laboratory" (PPL). We had only three or four students at that time! With hard work by generations of students and staff, PPL has become a large enterprise with a broad research agenda that does justice to our name. At PPL, we work on developing novel parallel programming abstractions backed by adaptive runtime techniques that automate tasks such as resource management and fault tolerance. We create collaborative, interdisciplinary applications, some of which are in production use on supercomputers all over the world and have scaled to tens of thousands of processors. The scope of our work and plans, and the number of people we affect, directly or indirectly, justifies the need for a regular newsletter. We intend to bring you feature stories about current research, recent accomplishments, information about collaborations, new and old, and much more, on a periodic basis.

For our first feature, we provide an overview of Charm++ and its history. We highlight three of our more successful parallel applications, ChaNGa, NAMD and OpenAtom. We also bring news about our visitors and award-winning PPL students. In future issues, we hope to continue with news about our inter-disciplinary collaborations and involvement in research projects. We are excited to bring the newsletter to you and we hope you will find it informative and useful.

- Laxmikant (Sanjay) Kale

What is Charm++?

The Parallel Programming Laboratory (PPL) at University of Illinois at Urbana-Champaign is a research group led by Prof. Kale aimed at multiple facets of parallel computing. The group has been called PPL since 1990, although it was founded when Prof. Kale came to Illinois in 1985. One of the signature products of PPL is a parallel programming system called Charm++. It forms a foundation for much of the research in the group.

We believe that a parallel programming system should provide abstractions aimed toward creating an optimal division of labor: the programmers should do what they can do best, while the system should automate what it is best at. Identifying what to do in parallel is something people are very good at, and systems (such as parallelizing compilers) are not quite so good at. On the other hand, resource management is much too tedious for a programmer, and it gets worse as machines grow more complex and applications grow more sophisticated – but a runtime system can automate it effectively. The design of Charm++ allows the programmer to specify decomposition of the computation into explicitly parallel work and data units, while empowering an intelligent adaptive runtime system (RTS) to control assignment of these units to processors, and even to change these assignments as the computation evolves. These units are abstracted as data-driven C++ objects known as chares.

This object-based design also respects locality of data-reference, which is critical to good performance on modern machines. Its cost model is clear to the programmer: access within the object is local and inexpensive, and access to any other object needs to go via asynchronous method invocations, and is clearly seen as more expensive.

The chares in a computation are organized into multiple indexed collections. Each chare is identified by the ID of the collection it belongs to, along with its unique index within the collection. Charm++ supports multiple index-structures including dense or sparse multi-dimensional arrays (of chares), as well as indexing by strings or bit-vectors.

Charm++ is a C++-based system, and each Chare is defined by a regular C++ class. The only extension is the presence of an interface file that describes the signature of methods of each Chare class that may be remotely invoked from other chares. The system uses these interface files to create code for automatically packaging method invocations into messages.

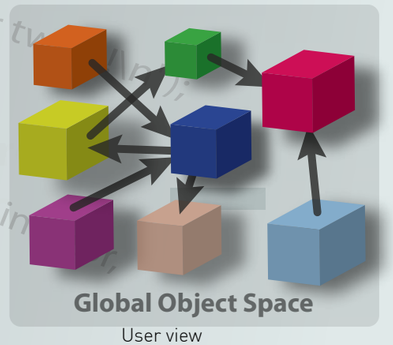
Charm++ supports a global object space, so programmers don't typically deal with processors; communication is expressed in terms of asynchronous method invocations (think of them as messages) sent towards other objects, named solely via their ID and index. So, if the runtime moves an object from one processor to another, the calling object (and the programmer) doesn't need do anything special.

There are typically many chares on a given processor. The system schedules them using a message-driven user-level scheduler. This adaptively overlaps computation and communication without any need for extra programming.

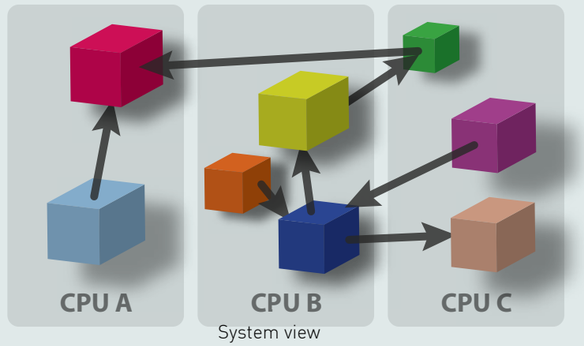
Continued on page 7

The History of Charm++

Charm++ evolved from its earlier incarnation, a library called ChareKernel, developed in 1988. Charm developed, over the next two years, into a C-based system with syntactic extensions that were supported by a simple translator with the Chare Kernel as its back end. Even though Charm was based on C, its structure was clearly object-based, with chares as message-driven objects. So it was natural, with the increasing popularity of C++, that 1992-93 saw development of a C++ based version, which was, obviously, named Charm++. This was the release 4.0 in Fall 1993. The concept of indexed collections of chares, the *chare-arrays*, was developed in 1995; and it was refined into almost the current form, with scalable support for migrations, insertions, deletions by year 2000. The system has been continually improved, with new ports, libraries and new functionality, such as fault tolerance, made available to the community via regular releases. The current release stands at 6.2.2.



User view



System view

APPLICATIONS CORNER

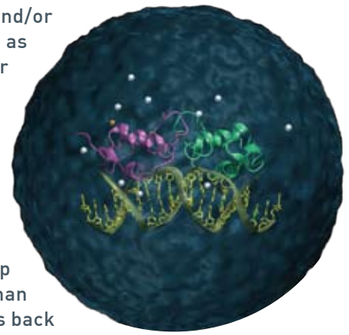
A review of PPL's top 3 parallel applications

The members of PPL collaborate with several computational science and engineering groups for cross disciplinary research efforts. These efforts have produced a number of parallel scientific applications. A few of them are described below.

NAMD

http://charm.cs.illinois.edu/research/moldyn/

NAMD (NAnoscale Molecular Dynamics) is a parallel molecular dynamics (MD) code designed for highly scalable simulations of large biomolecular systems. Typical NAMD simulations include all-atom models of proteins, lipids, and/or nucleic acids as well as explicit solvent (water and ions) and range in size from 10,000 to 10,000,000 atoms. The NAMD collaboration between Klaus Schulten's Theoretical and Computational Biophysics Group (TCBG) of the Beckman Institute and PPL dates back to more than fifteen years ago.



NAMD employs the prioritized message-driven execution capabilities of the Charm++ parallel runtime system, allowing excellent parallel scaling on both massively parallel supercomputers and commodity workstation clusters. It was awarded a Gordon Bell Prize in the 'Special' category at Supercomputing (SC) 2002. NAMD is distributed free of charge as both source code and pre-compiled binaries by TCBG. NAMD development is primarily funded by the NIH through the Resource for Macromolecular Modeling and Bioinformatics. NAMD is a popular MD package used widely at various supercomputing centers. NAMD has been downloaded by over 36,000 registered users, over 8,000 of whom have downloaded multiple releases. NAMD 2.7 was released on October 15, 2010 and features various optimizations for new architectures including NVIDIA CUDA GPU acceleration of nonbonded force evaluation.

OpenAtom

http://charm.cs.illinois.edu/OpenAtom

OpenAtom is a long standing (2001), cross-discipline, collaborative project to produce an efficient highly scalable fine grained implementation of the Car-Parrinello method for first principles molecular dynamics. By combining the physics of Glenn Martyna and Mark Tuckerman's PINYMD, with the event driven execution model and adaptive runtime of Charm++, the OpenAtom project has been able to achieve higher levels of productive parallelization (32 water molecules on 8,192 cores) than traditional schemes.

Recently in 2008, the project added Klaus Schulten (TCBG, Illinois) and Jack Dongarra (UT) as PI's to integrate with NAMD to provide QM/MM (quantum mechanics/molecular mechanics) features and optimize performance on ORNL/NICS's Cray machines. Once complete, this will provide the higher accuracy of QM modeling for the active regions of molecular systems modeled by NAMD.

The OpenAtom team is expecting to issue a new release 2.0 in 2011. In addition to QM/MM, the new features in this release will include Path Integrals, K-Points, Spin Orbitals, Tempering, high accuracy Van Der Waals, and Excited States.

ChaNGa

http://charm.cs.illinois.edu/research/cosmology/

Cosmological simulators are becoming increasingly important in the study of the formation of galaxies and large scale structures. The more general study of the evolution of interacting particles under the effects of Newtonian gravitational forces, also known as the N-Body problem, has been extensively reported in literature. Hierarchical methods for such simulations have been adopted for quite some time by astronomers. Most of those codes, however, do not scale effectively on modern machines with thousands of processors, due to load imbalance and communication overheads.

To address these scalability issues, PPL, in collaboration with Thomas Quinn, University of Washington, developed ChaNGa (Charm++ N-body Gravity solver), which had its first public release in February 2007. ChaNGa uses one of several available decomposition schemes to achieve an efficient distribution of particles over migratable objects. Through the use of techniques such as software tree caching and overlap of communication with useful force computation work, it has been able to accommodate large scale simulations of hundreds of millions of particles on over 20,000 cores. More recently it has also shown good performance on a cluster of a few hundred GPUs.

The Estrogen Receptor image above was made with VMD and is owned by the Theoretical and Computational Biophysics Group, NIH Resource for Macromolecular Modeling and Bioinformatics, at the Beckman Institute, University of Illinois at Urbana-Champaign.

PPL VISITORS

During 2009-2010, The Parallel Programming Laboratory hosted several visitors from different institutions around the world.

Research scientist Dr. Sathish Vadhiyar, from the Supercomputer Education and Research Centre of the Indian Institute of Science, Bangalore, was here during the summer researching scientific computational applications and developing enabling frameworks to apply computer science techniques to different scientific domains, including molecular dynamics, computational astronomy and climate modeling.

Eduardo Rodrigues, a Ph.D. Student from Federal University of Rio Grande Do Sul, Brazil, was with us for a year, working on dynamic load balancing in scientific applications, specifically weather forecasting codes. His research with PPL has resulted in a few papers, including one that was accepted at HiPC 2010 in Goa, India in December.

Dr. Yunchun Li of the Network and Information Center of Beihang University, China, is currently conducting research on parallel computing, focusing on the possible uses of Charm++ with embedded systems. Dr. Li will be leaving PPL at the end of 2010.

Ph.D. candidate Xavier Besseron from the Laboratory of Informatics of Grenoble (LIG), France, was in Urbana for a month researching domain decomposition applications with Kaapi/Charm++/MPI and fault-tolerance. The work he did added validation and additional results to his thesis. Dr. Besseron has successfully defended and is now working as a Postdoctoral fellow with Ohio State University.

Charm++ Workshop and 20th Anniversary of PPL

PPL's Workshop on Charm++ and its Applications will be held from April 18 through 20, 2011, at the University of Illinois. This annual meeting, gearing up for its ninth year, is an opportunity for interested groups to discuss research accomplishments in Charm++ and its applications, new development, and plans for the coming year. It is also an ideal time for new and prospective users to visit PPL, and get perspectives on the best ways to exploit Charm++'s unique features in building their applications. With its collaborative atmosphere, the workshop is a chance to present, learn about, and discuss work in progress related to the Charm++ ecosystem.

This upcoming workshop will feature a keynote address by Professor Jack Dongarra, Distinguished Professor at the University of Tennessee, Knoxville. PPL will also host a programming competition, focused on developing a load balancing strategy for a set of benchmark applications. Participants will harness Charm++'s object-based decomposition and adaptive runtime system to compete for the best performance.

Previous years' workshops have featured talks on a wide variety of topics, ranging from runtime implementation issues, parallel programming languages, debugging and analysis tools, to many application domains and more. Attendees can expect to hear about exciting new capabilities resulting from PPL's collaborations in enhancing NAMD, OpenAtom, and ChaNGa (featured on page 4), and how Charm++ facilitated their development. Other applications have included weather modeling, operations research, 3D rendering and visualization, and fundamental building blocks like parallel linear algebra and sorting.

PPL invites anyone interested in attending or submitting an abstract for a presentation to visit our web site, http://charm.cs.illinois.edu/charmWorkshop. We look forward to seeing you!

Awards and Honors

Members of the PPL group have been awarded a number of honors and awards over the past year. From outstanding undergraduate researchers to distinguished paper awards, the PPLers are aiming high (and hitting the mark) with their research.

Abhinav Bhatele successfully defended his thesis titled *Automating Topology Aware Mapping for Supercomputers* in August 2010. Dr. Bhatele has been the recipient of several awards due to his research: third prize in the ACM Student Research Competition at SC 2008 for contention studies on supercomputers, David J. Kuck Outstanding MS Thesis Award in 2009, and as mentioned later in this article, a Distinguished Paper Award, presented at EuroPar 2009 for significant improvements to a production scientific application (OpenAtom). Based on this thesis work, Abhinav was also selected as one of two George Michael Memorial High Performance Computing Fellows for 2009, in an international competition, by ACM/IEEE and the SC conference organizers. He will present his resulting research at SC'10 in New Orleans.

Edgar Solomonik was a PPL research assistant when he was picked as a finalist for the Computing Research Association's (CRA) Outstanding Undergraduate Researcher award for 2010. Solomonik was a senior in his 2nd year of study when he was awarded the honor.

The Computing Research Association presents the CRA Outstanding Undergraduate Researcher award yearly and this year it is sponsored by Mitsubishi Electric Research Labs. MicrosoftResearch and Mitsubishi Electric Research Labs are sponsors in alternate years. This represents the second time a PPL undergraduate earned this award. Ekaterina Gonina was awarded an Honorable Mention in 2008.

Edgar received the CS department Best Undergraduate Research Project Award for the 2008-2009 academic year and his research paper titled, "Highly Scalable Parallel Sorting," was presented at IPDPS (IEEE International Parallel and Distributed Processing Symposium) in April, 2010 in Atlanta, Georgia. After graduating with honors from Illinois, he went to pursue a PhD at University of California, Berkeley.

A Case Study of Communication Optimizations on 3D Mesh Interconnects by Abhinav Bhatele, Eric Bohm, and Laxmikant V. Kale was picked as a Distinguished Paper at Euro-Par 2009. The paper was one of four that was given this honor during the 2009 conference. Bhatele was on hand at the conference in Delft, The Netherlands, to present their research.

The paper presents research on exploiting the topology of supercomputers to improve application performance. Performance improvements of up to two times are presented in the paper. These gains were achieved using an API the authors developed to obtain topology information on 3D torus machines like IBM Blue Gene and Cray XT. Using the API and topology aware mapping, the object communication graph is embedded on the processor topology graph. The group is currently working on automating techniques developed in this paper and adding them to the Charm++ runtime.

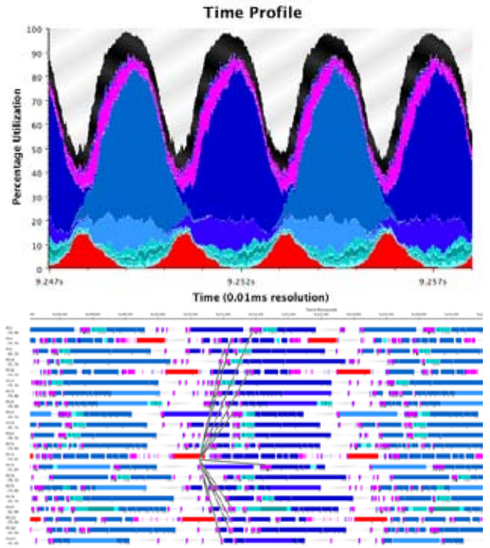
David Kunzman's "Toward a Framework for Abstracting Accelerators in Parallel Applications: Experience with Cell" was picked as one of four finalists for the best student paper honor at SC '09. The paper is a result of Kunzman's thesis research, which is aimed at harnessing the power of future supercomputers consisting of a heterogeneous collection of accelerators, along with manycore and multicore chips.

Recent PPL Alumni

This past year, four graduate students with PPL successfully defended their Ph.D. theses. Chee Wai Lee, Abhinav Bhatele, Isaac Dooley and Filippo Gioachin graduated from Illinois and are pursuing a variety of post-graduate careers. Lee is currently a post-doctoral researcher with the University of Oregon. Bhatele is doing a post-doc with Profs. Bill Gropp and Kale. Dooley accepted a position with Two Sigma, a financial corporation in Manhattan, New York. Gioachin is now based in Singapore and working for HP Labs.

Tools for Charm++ Development

Tools are important to the utility of a parallel programming system. Two PhD students at PPL submitted their dissertations during the past year on topics relating to tools: Chee Wai Lee's thesis on scalable performance analysis, whose results are being incorporated into Projections, a rich performance analysis tool with a long history in the group, and Filippo Gioachin's thesis on scalable debugging techniques, which are incorporated into a relatively new tool called CharmDebug.



Charm++ programs may be linked with a tracing library that produces detailed logs (or optionally, summary profiles) during execution. No user-level instrumentation is necessary. A tool called Projections can then be used to visualize various aspects of performance and to carry out certain kinds of analysis automatically. Projections has evolved over a decade's use in analyzing performance of production applications on large supercomputers. It has some unusual features that take advantage of the fine-grained application-level data that is available to the Charm++ runtime system, such as the separation of idle-time and overhead in communication operations. One can view a detailed timeline, as well as various profile and histogram displays. Projections supports advanced scalable analysis techniques, such as outlier analysis via k-means clustering. A recently added feature is live visualization of running programs, which relies on scalable data collection using a client-server interface without significant interference to the application performance.

CharmDebug is a specialized parallel debugger that understands Charm++ constructs and provides online access to runtime data structures. One can freeze a running computation, attach to individual nodes, and set conditional breakpoints. It provides support for tracking memory-corruption bugs, as well as non-deterministic bugs via a sophisticated record-replay system, which allows controlled analysis of the conditions leading to a bug on a given processor. CharmDebug was recently enhanced with the capability to analyze speculative message delivery, a technique that can uncover potential race conditions in a parallel program.

Charm++ continued from page 3

The ability to peek at the schedule's queue also allows the system to prefetch objects closer to the CPU. This has been exploited by our implementation on the Cell processor. The message-driven execution also supports efficient composability, since objects from multiple independent modules can interleave their execution on a processor, thus overlapping idle time in one with useful computation in another.

The most impressive benefits of this programming model arise when the RTS exercises its ability to migrate objects across processors. Dynamic load balancing is achieved by measurement based strategies that migrate objects away from overloaded processors. Evacuating processors in the face of impending failure leads to a proactive fault tolerance strategy. One can also shrink or expand the sets of processors allocated to a job, by migrating objects and adjusting runtime structures accordingly.

Another striking feature of Charm++ is its multiple strategies for reactive fault tolerance. Many of these strategies are supported in its production versions, as long as the job schedulers allow them. The simplest strategies support automatic checkpointing to disk with restart on a different number of processors. An in-memory checkpointing scheme goes further: it automatically detects failure, and restarts processes from their checkpoints stored in local or remote memories. An even more advanced experimental message-logging strategy sends only the failed processor back to its checkpoint, and *parallelizes* its recovery by sending the recovering objects to different processors. This allows an application to make progress even when MTBF falls below the checkpoint period!

The benefits of the Charm++ system are also available to MPI users via Adaptive MPI (AMPI), an implementation of the MPI standard on top of Charm++. Charm++ is a good substrate for developing novel higher level languages, because of automation of resource management and interoperability. We are developing a few such languages which elegantly capture specialized communication patterns: Charisma for static data flow, and MultiPhase Shared Arrays (MSA) for disciplined, deterministic and efficient use of globally shared arrays.

The system, its tools, documentation and papers about it are available at http://charm.cs.illinois.edu.

PPL & Blue Waters continued from page 1

tem that enables the dynamic overlap of computation and communication and provides automated load balancing and fault tolerance support for applications. This combination of features has enabled many complex applications written in Charm++ to scale to tens of thousands of processors in the past. However, scaling to machines the size of Blue Waters presents a new set of challenges. We are tackling these by introducing new features into the runtime system. These include SMP optimizations for intra- and inter-node communication, the automated placement of objects according to machine topology information, scalable hierarchical load balancing schemes, and the optimal usage of individual cores on a single node based on real-time measurements from program execution. Members of PPL have also started collaborations with researchers from DOE laboratories and other universities to define common runtime infrastructure that runtime systems of multiple programming models can use interopearably.

PPL is also responsible for the development of BigSim, a tool used to model and analyze the performance of applications running on future machines with large numbers of processors before they even exist. As a case-study, BigSim has been used to simulate and study the performance of NAMD on the entire Blue Waters system comprising more than 300,000 processing cores. The simulation itself was carried out on an existing cluster—with a much smaller number of processors. This is a powerful approach to the tuning of application performance, saving developers the expense of doing repeated and expensive performance tests on the actual machine. We have also integrated IBM's SystemSim sequential simulator into the BigSim tool chain, allowing BigSim to make accurate predictions of sequential pieces of code executing on future processor architectures before they have been implemented in actual hardware (such as on Power 7 before IBM manufactured a chip).

The development of accurate system simulators, such as BigSim, will enable scientists to better predict the performance of HPC applications on future machines and enable architects to design machines that are well-suited to target applications.

Department of Computer Science
University of Illinois • College of Engineering
201 North Goodwin Avenue
Urbana, IL 61801-2302

Non-profit Organization
US POSTAGE
PAID
Permit No. 75
CHAMPAIGN IL 61820

Members of the Parallel Programming Laboratory, May, 2010.