Performance Engineering Research Institute
SciDAC-2 Enabling Technologies Institute
2008 Annual Report

2008-09-29

Investigators

Lead investigators from the original PERI consortium:

Argonne National Laboratory	Paul Hovland and Boyana Norris
Lawrence Berkeley National Laboratory	David Bailey and Kathy Yelick
Lawrence Livermore National Laboratory	Bronis de Supinski and Dan Quinlan
Oak Ridge National Laboratory	Pat Worley, Jeff Vetter and Phil Roth
Rice University	John Mellor-Crummey
University of California at San Diego	Allan Snavely
University of Maryland	Jeff Hollingsworth
University of North Carolina	Rob Fowler
University of Southern California	Bob Lucas and Jacqueline Chame
University of Tennessee at Knoxville	Jack Dongarra and Shirley Moore

Lead investigators from extended members of the consortium:

University of Chicago	Chris Daley
University of Oregon	Allen Malony and Sameer Shende
Portland State University	Karen Karavanic
North Carolina State University	G. Mahinthakumar
University of Utah	Mary Hall

Purpose

This document reports on work performed by the Performance Engineering Research Institute (PERI), an Enabling Technologies Institute of the Scientific Discovery through Advanced Computing 2 (SciDAC-2) program of the Department of Energy (DOE) Office of Science (SC).

Enhancing the performance of SciDAC applications on Petascale systems has high priority within the DOE SC. As we look to the future, achieving expected levels of performance on high-end computing (HEC) systems is growing ever more challenging due to enormous scale, increasing architectural complexity, and increasing application complexity. To address these challenges, PERI has implemented a unified, tripartite research plan encompassing: (1) performance modeling and prediction; (2) automatic performance optimization; and (3) performance engineering of high profile applications. The PERI performance modeling and prediction activity is developing and refining performance models, significantly reducing the cost of collecting the data upon which the models are based, and increasing model fidelity, speed and generality. Our primary research activity is automatic tuning of scientific software. This activity is spurred by the strong user preference for automatic tools and is based on previous successful activities such as ATLAS, which has automatically tuned components of the LAPACK linear algebra library, the highly successful FFTW library, and other recent work. Our third major component is application engagement, to which we are devoting 30% of our effort to work directly with SciDAC-2 applications. This last activity not only helps DOE scientists meet their near-term performance

goals, but also helps keep PERI research focused on the real challenges facing DOE computational scientists as they enter the Petascale era.

Over the course of the past two years, the original PERI consortium of ten institutions has been augmented with some additional researchers, either via subcontracts or because the researchers have changed institutions and their funding will be transferred to the new institution. Inasmuch as these participants have made significant contributions to the overall PERI goals, and are now being funded directly or indirectly through PERI funding sources, we have included them in this report. See the list above for these additional institutions and their lead investigators.

FY08 Accomplishments

As mentioned above, PERI has three primary activities: (a) application performance modeling, (2) automatic performance tuning, and (3) application engagement. A brief overview of our accomplishments in each of these areas during the past year of this project is summarized below. This is followed by a brief discussion of other activities, specifically management and outreach to the rest of the performance community. Since this report is a summary of the overall progress of the entire PERI effort, we direct the interested reader to the annual reports of each PERI institution for additional details. Finally, because the universities involved in PERI received a No-Cost Extension of three months in 2007, this report covers a fifteen month period from June 2007 through August 2008.

Application Performance Modeling

Previously, under SciDAC-1 PERC funding, PERI researchers established a performance modeling methodology to predict application run-times accurately and with error bounds as the application's input parameters are varied. Additional results in this direction have demonstrated techniques that also provide significant cost savings for computer architecture sensitivity studies, which vary machine parameters such as cache sizes and prefetch and branch prediction strategies. Under PERI, we are continuing this research direction and extending it to make accurate predictions of application scaling performance.

One key center of the PERI performance modeling activity is the PMaC lab at SDSC. During the past year, SDSC researchers have developed a new open-source network simulator, PSINS. PSINS consumes MPI message traces such as those generated by MPIDTrace, Vampir, and others. It then enables researchers to play "what if" by varying attributes of topology, network bandwidth and latency and the like, to see what the performance effect would be on full scale applications if the underlying machine parameters were changed. This tool largely replaces the functionality of Dimemas, a commercial and closed-source simulator developed at the University of Barcelona. Because it is open-source, the PSINS tool allows a larger team of programmers to work on the code base and to develop features needed for petascale forecasting. The source is developed in a modular, easy-to-extend, style; one may easily add new topologies or new trace formats.

In collaboration with LLNL, PMaC is working to improve the capabilities of PSINS dramatically. PSINS, like Dimemas, is currently limited to predicting performance with fixed input size and CPU count. This new research is developing *trace extrapolation*, a technique that takes as input a series of traces at increasing CPU counts and/or input sizes, and extrapolates to traces of larger problems. This allows detailed and accurate simulation at processor counts beyond those available in existing systems. The early results show substantial accuracy for forecasting strong scaling (input held constant) and weak scaling (input allowed to grow with CPU count). These trace extrapolation techniques for communication are building directly on LLNL's scalable MPI tracing mechanism. In the last year, together with North Carolina State University, LLNL has

extended this mechanism to track application timing information in a scalable fashion. The results of this effort were reported in an ICS 2008 paper [Ratn].

The PMaC lab has completed a series of performance predictions of S3D (a turbulent combustion simulation code from Sandia) using both input scaling (increasing the complexity of the chemistry) and CPU count scaling. These predictions (along with those of Hoisie's group at LANL) were presented at the 2007 Denver SciDAC meeting as part of a controlled experiment using different technologies, with the goal to develop robust models for risk assessment in procuring large DOE systems. Both groups achieved 95% accuracy on the common subset of tested problems [Snavely2008].

Portland State University Professor Karen Karavanic has decided to spend her sabbatical leave as a Visiting Research Scientist in the PMaC Lab. Karavanic will provide an installation of the PerfTrack data collection, storage, and analysis tool for the PERI architecture tiger team effort. PerfTrack provides the functionality needed to automatically collect metadata describing application build and runtime environments (including system software, libraries, and the underlying hardware), and to integrate this with performance data from a variety of measurement tools into a single data store accessible through a GUI. This will allow PERI researchers to quickly view sets of results and locate measured application runs with specific characteristics.

In other work in the application modeling and measurement area, PERI researchers at Rice University enhanced their HPCToolkit performance tools with the capability for collecting call-path profiles for highly optimized scientific programs. Call path profiling enables one to accurately associate costs with the context in which they occur. This work builds on FY07 work, where Rice showed how to use call path profiles to pinpoint the quantified causes of scalability bottlenecks in parallel codes to individual lines in a source program.

Collecting call path profiles on highly parallel programs in turn required developing support for binary analysis of optimized programs. HPCToolkit employs three flavors of binary analysis: analysis of load modules to recover function entry points for stripped code; on-the-fly analysis of optimized machine code for functions to determine how to unwind the call stack from any program counter location within that procedure; and post-mortem binary analysis of the program and its symbol table information to recover information about loops and inlined code so that measurements can be properly attributed. During FY08, with support from both PERI and CScADS, researchers at Rice developed binary analysis capabilities to support call path profiling on both the Cray XT and IBM Blue Gene leadership class architectures. Prototype versions of these tools are operational and have been used to study a range of INCITE and SciDAC codes including the S3D and GTC Joule codes, the MFDn code from the UNEDF project, and the Chombo framework, as well as codes in domains such as atmospheric modeling, shock and turbulence, and molecular dynamics. Kernel bugs on both the Cray XT and IBM Blue Gene systems have prevented full deployment of these tools. Collaboration between Rice, UTK, and engineers at Cray resolved these problems on XT. A modified kernel will be deployed soon. Rice and UTK are continuing to coordinate with IBM to rectify the problem on the BG/L architecture.

UTK researchers have recently developed significant new PAPI support for performance modeling. The PAPI team has released PAPI version 3.6.1, with support for newer processors relevant to the modeling work of PERI, including the quad core AMD Opteron Barcelona found in Jaguar and Franklin. They have standardized event names for events native to the processors, based on industry input through the perfmon2 interface. They are working with IBM to coordinate development efforts for PAPI on the BlueGene/P systems at Argonne and ORNL, and are engaged in discussions with IBM on the implementation of overflow processing on BG/P for the support of the HPCToolkit, which will play an important role in performance data collection

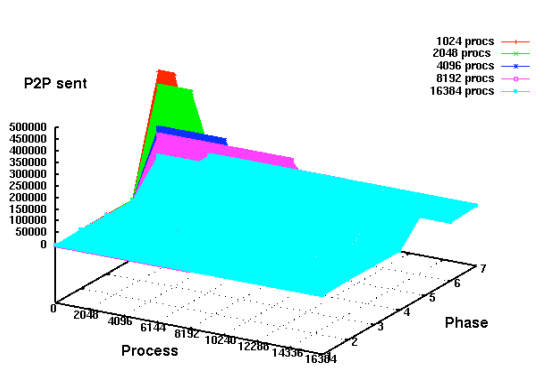
for this architecture. UTK has worked extensively with engineers at Cray to ensure that PAPI performs as expected on Cray's current and future operating system implementations on the XT3, XT4, and XT5 systems and their successors. UTK worked with PERI researchers at Rice to identify and to resolve issues surrounding fork/exec behavior while collecting performance profile data with PAPI. UTK has also developed an XML output format for PAPI events, making it easier for end-user tools to consume and to parse PAPI event information consistently.

UTK has also made additional improvements to PAPI. Through the use of PAPI_ITIMER and PAPI_set_opt(), PAPI can now use any itimer/signal combination for profiling/overflow. By using "software overflow emulation" and an event threshold of 1, users can now emulate gprof() style profiling and still use the PAPI_[s]profil() API. One can do this for real time (SIGALRM), virtual time (SIGVTALRM) or both (SIGPROF). This capability is very useful for profiling code that may spend large amounts of time waiting in system calls, where the program counter data returned from the sampling hardware is in kernel space and thus is dropped. Program counter data returned from this type of profiling are always in user-space. This improvement is useful to HPCToolkit. The new PAPI release has also improved the performance of PAPI_read_ts(), which is the fast version of PAPI_read() + PAPI_get_real_cyc() [PAPI]. This improvement lowers the data collection overhead for tools such as TAU and Scalasca.

In other longer-range research work, RENCI and LLNL researchers have developed mechanisms to identify the nature of application load imbalances and to capture detailed load balance statistics in a scalable fashion, based on wavelet transforms and LLNL's scalable MPI tracing mechanism. The mechanism provides an ability to understand how load differences evolve across all tasks in different code regions over a series of application time steps [Gamblin2008a].

LLNL researchers have continued to explore regression-based performance prediction techniques across large parameter spaces, such as architectural configurations, parallelism or even application input. Current foci include extrapolations to scaling behavior as well as reducing the volume of input data required and deriving optimization guidance from predictions. Recent work published at ICS 2008 demonstrated that the techniques can provide scaling predictions directly by extrapolating from timings of several input sets at smaller scales [Barnes].

In another research activity, ORNL has extended the Modeling Assertions (MA) framework to model communication patterns symbolically, as an alternative to regression and statistical models. These symbolic models can be used to generate synthetic trace files of any scale and for other analysis tasks, such as sensitivity analysis. Also, application developers can capture and represent hierarchical decomposition and distribution of their applications, which allows them to explore optimal mappings of their workload on a target system without actually executing it on that system. This methodology has been demonstrated on an INCITE fusion application called GYRO and the models could identify and validate the sub-communicator that is the scaling bottleneck. We have developed a synthetic trace generator based on these symbolic models for performance predictions using the MA models. We then inject these traces into a communication simulator to explore many combinations of workload and system configurations efficiently. A preliminary grammar has been defined and a simulator prototype has been developed. Figures 1a and 1b show some results of using MA for GYRO and AMBER.



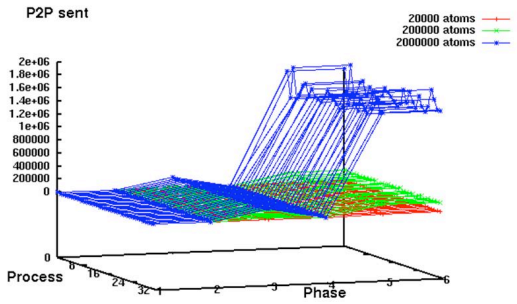


Figure 1a: GYRO communication growth rates generated with MA symbolic models.

Figure 1b: AMBER communications rates generated with MA symbolic models.

The latest MA version comes with a set of additional pre- as well as post-processing tools. The MA pre-processing toolset allows automatically insertion and deletion of pragma statements in front of MA function calls. It facilitates model generation and development processes for petascale applications. The MA post-processor implements parallelized validation algorithms, which enable fast validity check of complex performance modeling data. Very large data volumes can be verified due to the close interaction with state-of-the-art parallel production environments. The validation tool can efficiently handle MA traces with thousands of processes. An overview of the MA architecture is depicted in Figure 2.

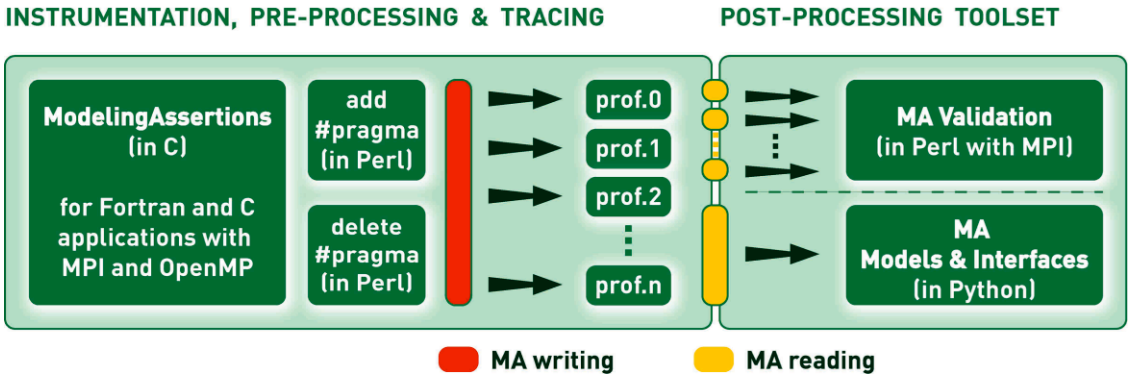


Figure 2: Modeling Assertions workflow.

Automatic Performance Tuning

In the area of automatic performance tuning, PERI researchers are developing whole program analysis that facilitate optimizations spanning multiple kernels, files, and procedure boundaries. This work utilizes empirical techniques on significantly larger scales than individual kernels. Initial efforts will demonstrate this methodology on a selection of SciDAC applications. The principal research focus will be in scaling the techniques developed for kernel level optimization to the larger applications and in leveraging work elsewhere within PERI to support the recognition and analysis of entire applications and identify opportunities for optimization. We have also begun to define the initial analysis and transformation support required to introduce custom optimizations to SciDAC applications. A number of transformation mechanisms are proposed to simplify the development of specific sorts of general transformations. This work will include a library of transformations that are easy to specify and to apply as part of custom transformations.

Within PERI, five different research groups are developing auto-tuning *tools* to address the challenges of automatic performance tuning. These projects have complementary strengths and can, therefore, be brought together to develop an integrated auto-tuning *system*. Towards that end, we are working to develop a common framework to allow auto-tuning tools to share information and search strategies. Through common application programming interfaces (APIs), we can evolve an auto-tuning system that brings together the best capabilities of each of these tools, and also engage the broader community of tool developers beyond PERI researchers.

PERI researchers have focused development of the interfaces on two portions of the auto-tuning process. Any compiler-based approach will apply code transformations to rewrite application code from its original form to one that more effectively exploits architectural features such as registers, caches, SIMD compute engines, and multiple cores. Commonly used code transformations include loop unrolling, blocking for cache, and software pipelining. Thus, we are designing a *transformation API* that will be input to the Transformation box in Figure 3. This API provides a *transformation recipe* that describes how to transform original source into an optimized source representation. By accepting a common transformation recipe, the auto-tuning system permits code transformation strategies derived by PERI compilers and tools (or users) to be implemented using any transformation and code generation tools, such as Chill (USC/ISI), LoopProcessor (LLNL) and POET (UTSA). The API supports the specification of unbound transformation parameters that are then tuned using search algorithms. The initial draft of the API includes a naming convention for specifying language constructs in source code and code transformations available in Chill and POET.

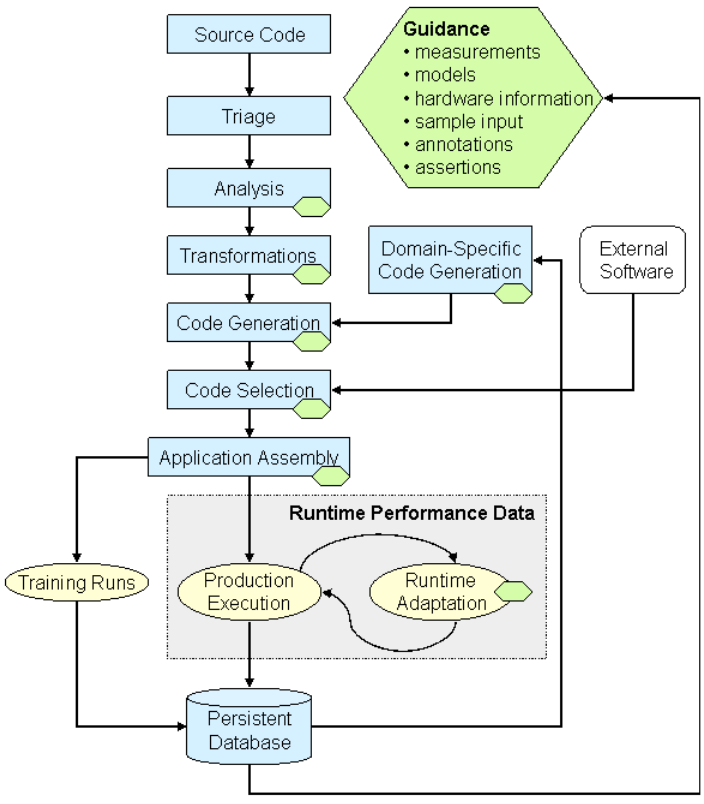


Figure 3: PERI automatic tuning workflow.

A *search API* provides input into the empirical optimization process of running a series of experiments on actual hardware to determine the best optimized implementation. The search API allows the auto-tuning tools to exchange information about their available tuning options and constraints on the search space, and to plug-in different search algorithms. The common framework will support both auto-tuning using training runs (and re-compilation) along with continuous optimization during production runs. For the search API, we are working on developing a simple and extensible language that standardizes the parameter space representation. Using the language, developers and researchers can expose tunable parameters to tuning frameworks. Relationships (ordering, dependencies, constraints and ranking) between tunable parameters can also be expressed.

Work at LLNL has focused on the compiler support for the analysis, transformation, and optimization of large scale Fortran 2003, C, and C++, OpenMP and UPC applications. Fortran work is a collaboration with LANL and uses their Open Fortran Parser (OFP) and with Rice. The work in ROSE includes support for mixed language applications (Fortran and C) and global analysis. ROSE also supports older versions of Fortran (F95, F90, F77, F66). Recent work has also provided Fortran support for code outlining, which is used to support whole program auto-tuning of large scale applications. ROSE is openly available at http://www.roseCompiler.org, with all documentation and access to the SVN repository. All work is tested and updated nightly on the web site. Language support in ROSE is ongoing; work has demonstrated the robustness of ROSE on a wide collection of tests, compiler test suites, and applications code (results are made available on the web site).

To understand the requirements of integrating auto-tuning tools developed independently by PERI researchers, we have been engaging in the integration of several PERI tools. Researchers at the University of Tennessee have integrated auto-tuning search (described below) with the ROSE LoopProcessor (LLNL) and the POET code generator (UTSA). Similarly, an initial integration of the Active Harmony system (UMD) and the CHiLL transformation framework (USC/ISI) is providing experience in how to integrate these separate tools effectively into an auto-tuning system. The remainder of this section describes the research being carried out by these groups, and where relevant, integration activities between the tools.

This University of Maryland team of Jeff Hollingsworth and Ananta Tiwari led the effort to develop a common search API, to allow auto-tuning systems to share information and search strategies. The framework provides a common API to allow auto-tuning systems to exchange information about their available tuning options, constraints on the search space, and to plug-in different search algorithms. The common framework supports both auto tuning using training runs (and re-compilation) along with continuous optimization during production runs.

The aforementioned framework unites the existing research being conducted by the members of the PERI team. However, to bring application developers and researchers from other institutions into this process of collaborative tuning, we are developing a simple and extensible language that standardizes the parameter space representation. Developers and researchers can use this language to expose tunable parameters to our tuning frameworks and to express relationships (ordering, dependencies, constraints and ranking) between tunable parameters. We have revised the parameter space language several times and will release it soon [Chame].

USC/ISI has developed and released CHiLL, a framework for composing high-level loop transformations designed to generate efficient code for complex loop nests. It supports an extensive collection of loop transformations for perfect and imperfect loop nests, including tiling, permutation and unroll-and-jam, thus lifting the burden of generating multiple intermediate steps from compilers or optimization tools. CHiLL uses the Omega test to manipulate integer

arithmetic and relies on polyhedral scanning provided by Omega's code generator. Recently we have integrated new features to CHiLL, including support for user input to the tool. Users can now relay information about the code and input data that could not be derived by static analysis alone, and this information often results in more efficient code generation [ISI_NEED: REF].

USC/ISI has updated the Omega Library, which is a main component of CHiLL, to:

- Improve internal code generation interface to support both string and SUIF outputs and be more easily extensible for new intermediate code formats.
- Improve internal function (*gist*) to handle integer modular constraints more gracefully.
- Merge duplicate if-conditions in generated code.
- Correct output/input variable substitution for non-unimodular mapping relations.
- Fix problems in Omega's calculator parsing and interactive interface.
- Enhance code generation for loops with strides larger than one.

A complete list of updates can be found at *http://www.isi.edu/~chunchen/omega*. Both CHiLL and version 2.0 of the Omega Library have been released and can be found, respectively, at *http://www.isi.edu/~chunchen/chill* *and* *http://www.isi.edu/~chunchen/omega* .

UMD and USC researchers have been working to integrate the Active Harmony system with the CHiLL framework. Users can customize the CHiLL optimizer via a process called recipes. The Active Harmony framework uses information about the allowable transformations and their parameters (e.g. loop unroll factors) to generate recipes, and exponentially evaluate them. Using Harmony's search algorithms we can generate and evaluate different optimizations that would have been prohibitively time consuming for a programmer to explore manually.

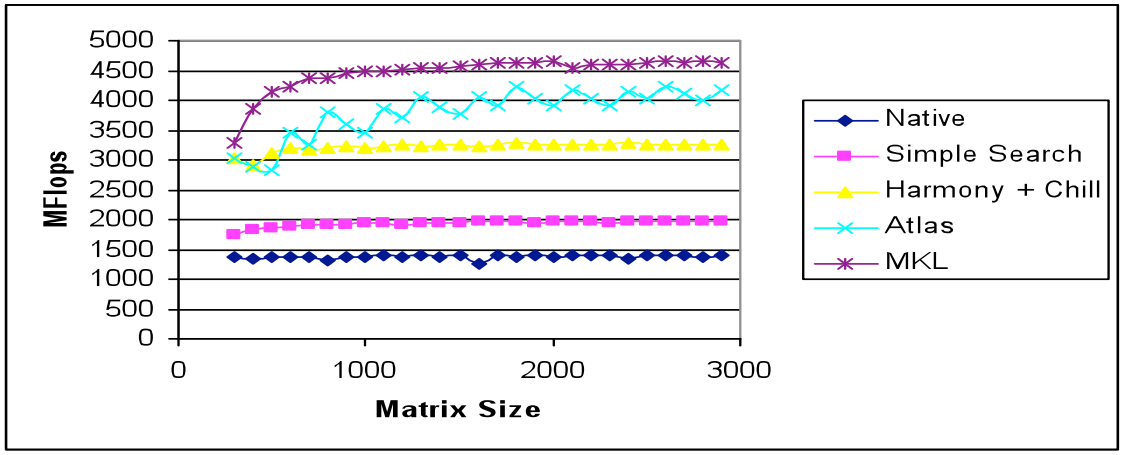


Figure 4: Performance of Matrix Multiply, ActiveHarmony+CHiLL versus ATLAS and MKL.

Figure 4 compares the performance (in terms of megaflops) for matrix multiply using our combined Active Harmony and CHiLL frameworks, on a 2.1GHz Intel CoreDuo. The results show that the fully automated compilation and tuning using Harmony plus CHiLL is able to exceed the base performance of Atlas (without custom assembly code). However, the approach is not yet able to reach the performance of MKL (which uses some explicit assembly code) or the full Atlas system (which also uses some assembly code).

A goal in PERI auto-tuning is to substitute different code generators and search engines easily in the tuning process. To that end, UTK has developed a simple syntax for specifying how to generate code to perform the evaluation of the code variants. From this specification, the test case generator produces C code and a Makefile that generate the timing executable. To permit

switching search techniques, the application-specific aspects of the evaluation process are described by a shell script, which has a simple specification of the search bounds and constraints embedded in comments. In this way, the search engine does not need to know the details of the code generator, build, or test case generation. This system permits evaluation of various search techniques to determine which one works best in an auto-tuning context. We have evaluated classical techniques such as genetic algorithms, simulated annealing, and particle swarm optimization, as well as ad-hoc techniques such as a modified orthogonal search.

As a simple example of an optimization search space, the Figure 5 shows optimization of square matrix-matrix multiplication (N=400), using an exhaustive search over two dimensions: block sizes up to 128 and unrolling up to 128. The x and y axes represent block size and unrolling amount, while the z axis represents the performance in Mflop/s of the generated code. The dimension size is relatively small to allow faster evaluation at each point while being large enough to ensure repeatable measurements using PAPI. The best results generally lie along the blocking axis with a low unrolling amount or along the diagonal with equal blocking and unrolling, but peaks also occur where the block size is evenly divisible by the unrolling amount. A block size 80 and unrolling amount 2 provides the best performance. This code variant ran at 1459 Mflop/s compared to 778 Mflop/s for the naive version compiled with gcc [Seymour2008].

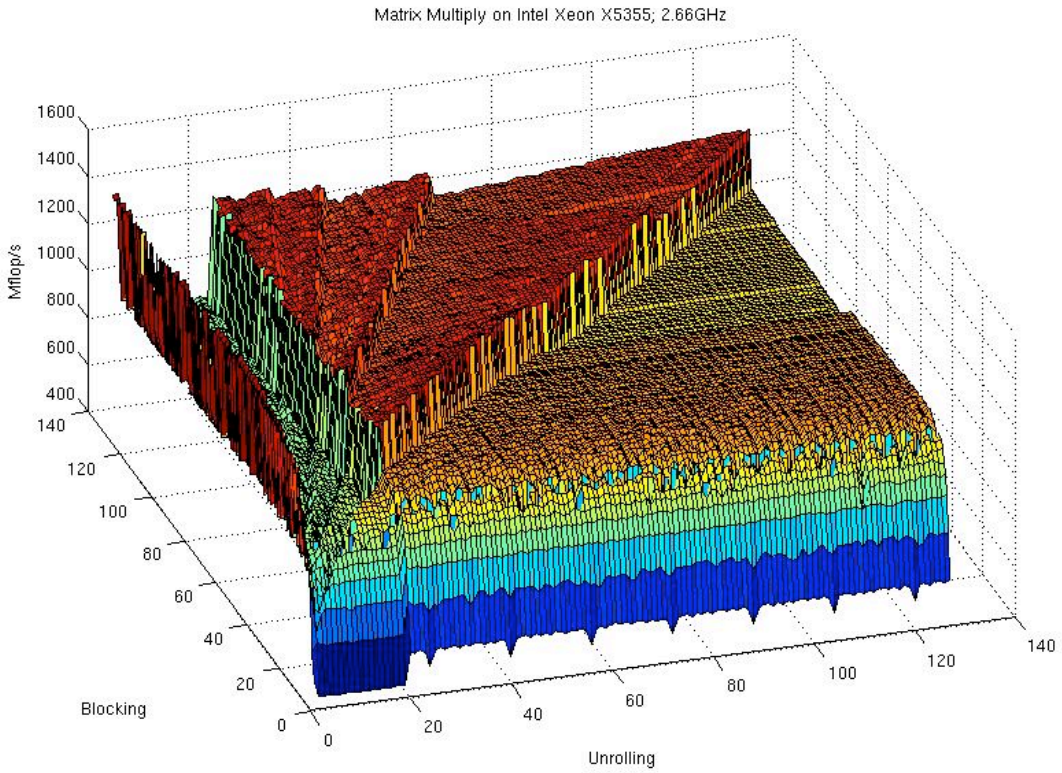


Figure 5: Performance of matrix multiplication with blocking and unrolling.

Recently UTK carried out a thorough comparison of the Nelder-Mead simplex, genetic algorithm, simulated annealing, particle swarm optimization (PSO), orthogonal search, and random search algorithms in terms of the performance of the best candidate found under varying time limits. We used matrix-matrix multiplication and matrix-vector multiplication as test cases, optimizing them for a 2.66GHz Intel Xeon X5355 running Fedora Core 6 and gcc 4.1.2. In addition to five code transformation options of separate blocking dimensions, an unrolling dimension, and a loop order

dimension, we added 16 dimensions of compiler optimizations to make the search more difficult. We found that PSO had a clear advantage in some experiments possibly because the swarming provides a local search functionality near the end of the cycle. Also, some of the dimensions have their best performance near the bounds and PSO tends to search theses areas well [Seymour-2008].

In other work, UTK developed an analytical model to estimate the cost of running an affinity-based thread schedule on a multicore system. The model has three components: an affinity graph model to describe the affinity relationship between threads, a memory hierarchy model to characterize the underlying shared memory architecture, and a cost model to estimate the cost of a particular thread schedule. We model the memory architecture as a generic tree structure, allowing a portable, architecture-aware optimization framework to find an optimized schedule for multi-threaded programs. Since the problem is NP-hard, we have designed a hierarchical graph partitioning algorithm to compute an approximate solution. The algorithm supports threads with data dependencies. We implemented the algorithm and applied it to Cholesky factorization. On the Intel Clovertown processor, the optimized version is 60 to 100% faster than the MKL routine as shown in Figure 6 [Song2007, Song2008].

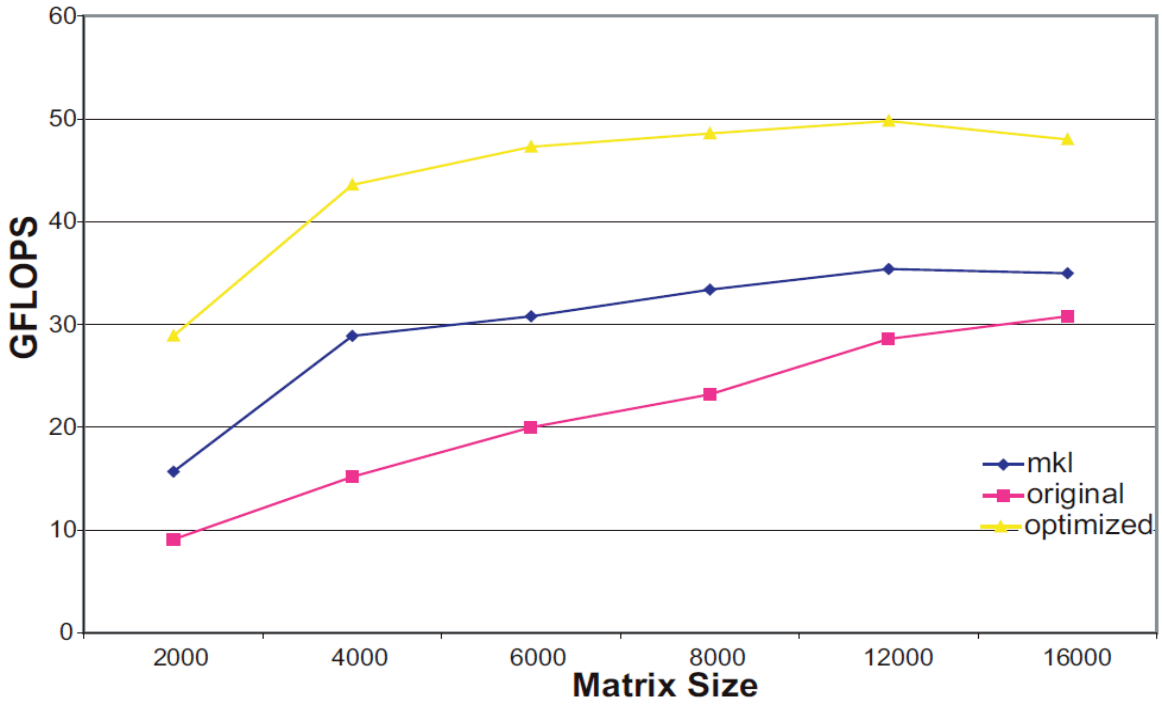


Figure 6: Performance of Cholesky factorization on Intel Clovertown

In a study of auto-tuning a sparse matrix-vector multiply (SpMV) kernel, researchers at LBNL compared performance on a dual-socket Intel Core2 Xeon, single- and dual-core AMD processors, a quad-core Opteron, a Sun Niagara2, and an IBM Cell Broadband Engine [Williams2007; Williams2008a; Williams 2008b]. This study found that these processors differed markedly in the effectiveness of various auto-tuning schemes on the SpMV kernel. In two subsequent studies, we analyzed similar but more sophisticated automatic optimizations for SpMV, the explicit heat equation PDE on a regular grid (Stencil), and a lattice Boltzmann application (LBMHD). This auto-tuning approach employs a code generator that produces multiple versions of the computational kernels using a set of optimizations with varying

parameter settings. The optimizations include TLB and cache blocking, loop unrolling, code reordering, software prefetching, streaming stores, and use of SIMD instructions.

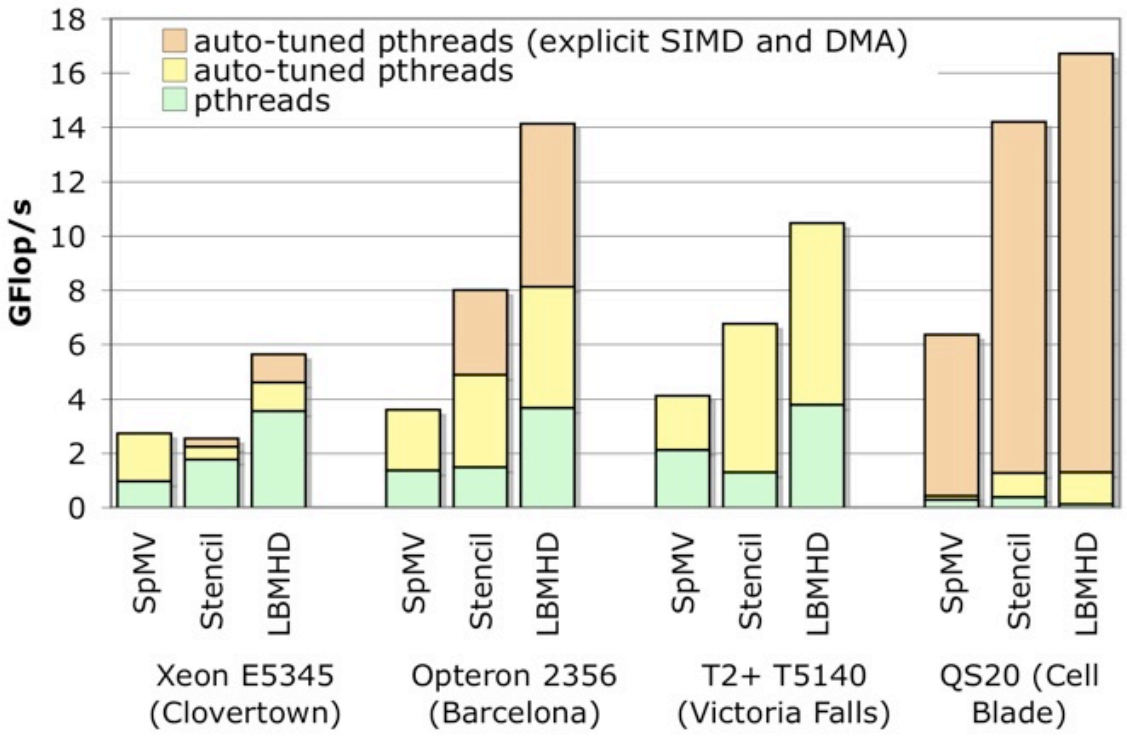


Figure 7: Performance of SpMV, Stencil and LBMHD on four platforms.

The impact of each optimization varied significantly across architecture and kernel, necessitating a machine-dependent approach to automatic tuning in this study. In addition, detailed analysis revealed performance bottlenecks for each computation on the various systems. As shown in Figure 7, the Cell processor offers the highest raw performance and power efficiency for these computations, despite having peak double-precision performance and memory bandwidth that is lower than many of the other platforms in our study. The key architectural advantage of Cell is explicit software control of data movement between the local store (cache) and main memory, which is a major departure from conventional programming. These studies demonstrate that considerable room for improvement in auto-tuning methods for the candidate architectures exists.

Figure 8 shows results on applying several of these automatic tuning techniques across a range of application kernels.

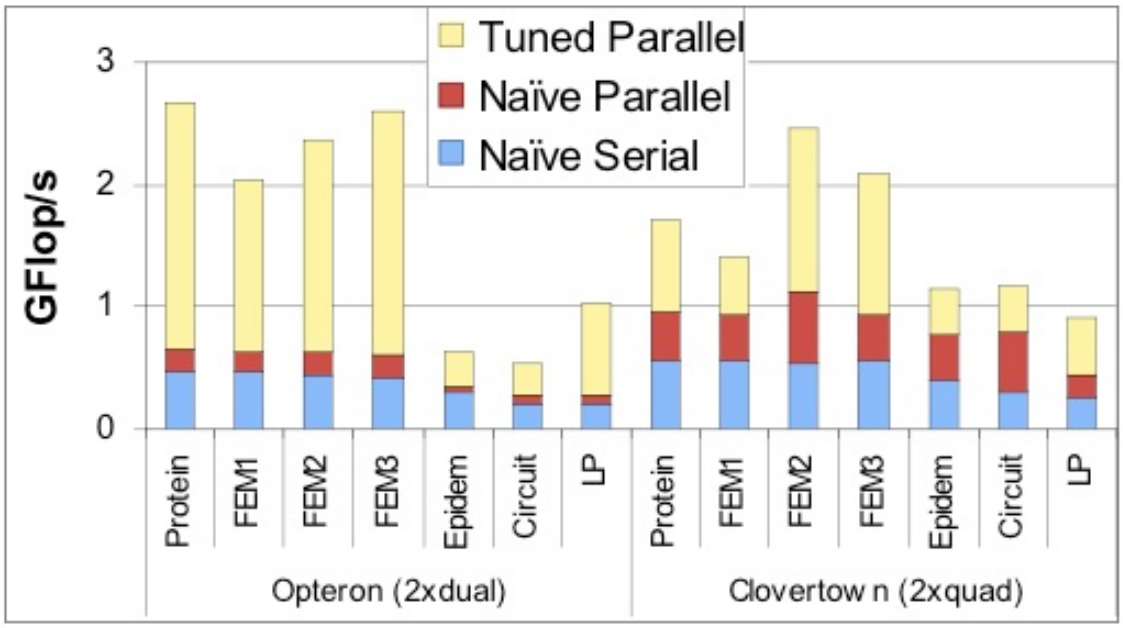


Figure 8: Tuned parallel applications on Opteron and Clovertown platforms.

Researchers at ANL continued development of component infrastructure for automated performance monitoring, modeling, and runtime adaptation in collaboration with the TASCS SciDAC Center. The goal of this joint effort is to develop component middleware for support of Computational Quality of Service (CQoS) for component applications that enables automated application configuration and runtime adaptation with the goal of minimizing time to solution.

During FY08 the Argonne group developed components for collecting performance data and associated application-specific metadata into a performance database; interfaces and components for managing the data in the performance database; initial comparator interfaces for defining data filters by defining comparison operations on arbitrary parameter sets (parameters can be both performance-related or based on metadata entries); and (in collaboration with the University of Oregon Performance Research Lab) an initial interface and component implementation wrapping PerfExplorer, a powerful performance analysis tool.

This work has enabled the automated application configuration for three applications: a quantum chemistry code (with Joe Kenny of Sandia), a combustion application (with Jaideep Ray of Sandia), and PETSc-based applications, such as a driven cavity code distributed. They are also beginning to define interfaces that would enable runtime adaptivity of these applications [Li].

Researchers at UNC/RENCI originally planned to explore several items regarding monitoring environmental factors and developing open source tools and models to predict impending failures. UNC has discontinued work in this area; instead, the UNC/RENCI team has focused on pressing performance issues related to emerging systems: scalable measurement and analysis on systems with tens or hundreds of thousands of nodes [deSupinski2008a ,Gamblin2008a, Gamblin2008b]; and performance measurement and analysis [Porterfield].

On-node power management is an important emerging issue. UNC/RENCI has performed experiments that confirm in the HPC domain studies conducted in the domain of mobile devices suggesting that often the most effective approach to energy conservation is to use the highest performance version of a program (best algorithm, most aggressive compiler optimization) to solve a problem as fast as possible and then either to move on to the next problem or to shut the

hardware off. However, other studies have suggested that dynamic voltage and frequency scaling (primarily on single core processors) could lower power consumption. In the past year RENCI researchers have initiated an effort to evaluate the opportunities for compiler optimizations rigorously, including insertion of power control directives, in the power vs. performance space.

Past studies have relied on the use of external power monitoring devices that are usually available only with a very coarse temporal granularity. RENCI researchers have therefore designed and constructed a prototype PowerMon, which is an internal device that sits between the power supply and system devices and that can provide frequent, detailed power measurements via a USB interface to either the system being measured or to an external machine. The prototypes could be manufactured for a parts cost of under $40 in quantities of 100 or more. The initial version used a software USB modem to supply 10 sets of 12 measurements a minute. A minor revision to the prototype increases this number to approximately 800 measurements per minute. A fully integrated revision of the design that will enable it to monitor additional devices should be ready by September 2008. Preliminary experiments with fine-grain PowerMon and hardware performance counter measurements on a multicore chip (Barcelona) indicate that adaptive runtime reduction of core voltage and frequency during phases of high memory latencies are effective for reducing energy consumption with negligible impact on performance.

Emerging multicore microprocessors increase the potential for contention for shared resources. Thus performance monitoring and analysis tools that focus solely on the behavior of each thread in isolation are likely to be less effective in identifying and diagnosing performance issues. RENCI have therefore embarked on a program of research into resource-centric tools for monitoring and analysis [Feng2008]. As a step in this direction, they have modified the Perfmon2 user-level tool “pfmon” to directly generate multicore profiles in “system-wide mode” that are compatible with HPCToolkit. Since this approach does not virtualize performance counters in PAPI, it is very reliable and imposes less overhead. The first real application for the resource-centric tool prototype is a LQCD computational kernel for computing the Wilson DSlash operator on AMD quad-core machines using MPI for inter-node communication. The two performance issues being addressed are thread scheduling/synchronization problems, and the interaction between SSE4 intensive computational loops and contention for shared memory [Fowler2008b].

UNC/RENCI is also hosting the database server for the PERI database(s) for providing feedback and guidance as illustrated in Figure 4.

Application Engagement

In order to ensure the relevance of this research and the usability of the resulting tools, PERI researchers are working closely with application scientists, focusing on DOE-funded computational science projects. PERI participants are proactive in communicating with the application groups: attending their meetings to make presentations, hosting individuals or groups at our sites to demonstrate tools, and communicating directly with the developers when optimizations important to their codes are identified. Such partnerships promote the widespread usage that is necessary for tools to gain broader use in the community. Pat Worley of ORNL has overall management responsibilities for engagement activities and for the liaison activities in particular. When not available or not sufficient through the computational science projects, computing resources for the PERI engagement activities on the ORNL and ANL systems are being provided by the Performance Evaluation and Analysis Consortium End Station (PEAC) INCITE project.

Applications Survey

As reported in the FY2007 report, PERI implemented a web-based application survey to gather information about code characteristics and performance goals and issues for SciDAC applications. This survey has been an important tool for identifying which computational science projects are good candidates for PERI engagement activities. For FY2008, the survey was modified to allow computational science projects to update the information submitted the previous year. PERI researchers periodically contact each computational science project that submitted a survey in FY2007 to check whether the survey still accurately represents the characteristics and performance needs of the project.

2007 Tiger Teams

In order to focus our performance tuning expertise on DOE's highest priorities, PERI has formed "tiger teams" to work directly with high-priority DOE application teams. At the beginning of the 2007 fiscal year, PERI consulted with its Office of Science Program Managers to select the application teams for that year's tiger team focus. That year PERI was directed to help the DOE Joule codes meet their milestones. After consulting with the PIs of the Joule code teams, we decided to focus on GTC and S3D. We include in this report some of those activities that crossed the FY2007-2008 boundary.

The PERI-GTC Tiger Team includes PERI researchers at UTK, Rice, and LBNL, as well as significant participation by the University of Oregon. It began by conducting joint conference calls between the Tiger Team and the GTC development team. The Oregon group profiled the original GTC code on several platforms including BG/L, an Opteron cluster (Jacquard at NERSC), and Cray XT3/4 (Jaguar at ORNL) using the TAU performance system and has started profiling GTCS on Jaguar. The developers identified the scatter-gather algorithm as a performance bottleneck in both versions of the code, and this bottleneck appears to be due to memory hierarchy performance issues. Using hand-tuning techniques such as common subexpression elimination and movement of statements outside loops, UTK has been able to get 10 percent speedup of the scatter.

In collaboration with GTC developers, PERI is investigating approaches for improving memory performance that involve changing the data layout. With support from both CScADS and PERI, researchers at Rice explored three strategies for improving locality and memory performance with GTC: transposing arrays of structures into structures of arrays (to improve spatial locality), fusing and blocking loops (to improve temporal locality), and using periodic sorting of ions in the plasma to translate physical proximity in space into memory locality. After these transformations, the shaped plasma version of the GTC code (known as GTS) now runs 21% faster on the Cray XT4 using test problems provided by the developers. The code changes were provided back to the GTC team. PERI researchers have also applied performance modeling techniques to understand GTC's memory performance better. The performance database working group (described below) has used GTC as an example code for its performance database interoperability effort and is collecting a comprehensive set of routine-level profile data [Adhianto; Marin2008a; Marin2008b].

The second PERI Tiger Team was organized to focus on S3D. For this project, SDSC developed a graphical user interface (GUI) to MetaSim tracer output. The tool thus shows expected cache rates of source code structures such as loops. Using the tool they discovered anomalously low cache hit rates of several important loops in S3D and then found better compiler flag combinations to improve their performance by at least 30%. This is an early instance of performance model guided tuning.

In addition, with HPCToolkit, Rice identified and improved several other performance issues in S3D. There was a compiler-generated loop copying 4D slices of a 5D array. We identified this problem with HPCToolkit's loop-level performance diagnosis, rewrote a few lines of code and removed 100% of this overhead, which improved overall S3D performance by 5.5%. The diffusive flux computation in S3D is the most memory intensive. We identified that this loop was making inefficient use of the memory hierarchy. We used LoopTool to unswitch conditionals out of the loops, fuse 3D loops arising from F90 vector statements, and unroll and jam two outer loops into the fused inner loops. The resulting loop nest was 66% faster on their 50 x 50 x 50 model problem. This change improves overall application performance 6.8%. Additionally, the heat flux computation performed a reduction that didn't fill the floating point pipeline. Minor transformations and massaging with LoopTool reduced execution time of this loop by 42%, providing a 1.73% improvement in overall performance.

Initial studies of S3D node performance for Sandia's Ethylene test case on quad-core Opteron processors using HPCToolkit indicated that almost 1/3 of the total execution time is spent in math library routines, such as exp, log, and pow. As S3D is applied to mixtures with more reactants, the performance of the math library routines will consume an increasingly larger fraction of application execution time [Tallent]. LBNL devised a custom exp routine, which ran faster; however, subsequent improvements in the vendor-supplied exp routine made this change unnecessary. Nonetheless, the S3D tiger team is working with the S3D team to ensure that their reaction rate code is vectorizable and that highly optimized math libraries are available on the leadership class machines.

More recently, the PERI S3D tiger team evaluated the scalability of S3D using the TAU Performance System, which has been developed at the University of Oregon [Mellor-Crummey2007; Mellor-Crummey2008]. We used TAU to automatically instrument S3D at the outer-loop level and to collect performance data on Jaguar at ORNL, Franklin at NERSC, and the Intrepid system at ALCF. TAU uses PAPI on all these systems to measure low-level hardware performance metrics such as the total number of floating point instructions executed in each loop and the number of data cache misses. The tiger team conducted experiments on Jaguar up to 12,000 cores using the S3D Ethylene test case. The profiles files collected were then uploaded to the PERI performance database maintained at RENCI. PERI uses TAU's PerfDMF database to collect and to disseminate performance data for performance modeling and analysis. TAU's PerfExplorer tool was used to identify scaling trends for S3D at the loop level, identifying two loops that exhibit poor floating point efficiency and account for significant runtime. The S3D tiger team is in the process of optimizing these loops. The collected performance data can be visualized using TAU's Java webstart enabled browsers at the PERI webpage http://www.peri-scidac.org/wiki/index.php/S3D_on_Jaguar.

Here are some details of PERI's 2007 tiger teams:

Tiger Team Coordinator: Bronis de Supinski (LLNL)

1. S3D/JOULE
Team Lead: Bronis de Supinski (LLNL)
Team: Bailey (LBNL); de Supinski, Vuduc (LBNL); Morris, Shende (UO); Fagan, Mellor-Crummey (Rice); Snavely, Wright (SDSC)
Application Team contacts: Chen (SNL); Sankaran (ORNL)

2. GTC/JOULE
Team Lead: Shirley Moore (UTK)
Team: Shan (LBNL); Moore, You (UTK); Mellor-Crummey, Jin, Marin (Rice)
Affiliates: Oliker (LBNL)
Application Team contacts: Ethier, Klasky, Lee, Wang (PPPL)

2008 Architecture Tiger Team

In the Spring of 2008, PERI was directed by DOE SC to conduct a study of affinity of twelve Petascale pioneering applications to different computer architectures. The intention is to use this information as input in the Office of Science's ten-year facilities planning. In response to this charge, PERI has formed an Architecture Tiger Team:

Team Lead: Bronis de Supinski (LLNL)

Team: Moore (UTK), Jagode (UTK and ORNL); Roth, Vetter, Worley (ORNL); Carrington, Karavanic, Olschanowsky, Tikir (SDSC); Bailey, Shan (LBNL); Shende, Malony (UO), Schulz (LLNL), Hovland, Norris, Riley (ANL), Mellor-Crummey (Rice)

The PERI Architecture Tiger Team decided to pursue a two-phased approach. The first phase is measuring the performance and scalability of the codes on today's leadership-class systems, the Cray XT-4 (LBNL and ORNL) and the IBM BG/P (ANL) which will enable better understanding of the codes and their affinity to today's systems. The second phase is using PERI modeling technology to predict the codes' relative performance, and hence affinity, for future systems. Measurement results for both the analysis and modeling efforts will be stored in a performance database. These future systems are to include the HPCS systems expected circa 2011.

DOE directed that the first three codes addressed be FLASH, GTC, and S3D. PERI contacted the Principal Investigators and was assured of their willingness to cooperate. Overall, the plan is for the Architecture Tiger Team to proceed in a series of stages, handling three applications of the twelve Pioneering applications (or others based upon consultation with OASCR) in each stage. Thus, the first stage will address both phases for the first three applications. The architecture team has received source code and input files from all three code teams, and as of this time is beginning the process of building and measuring them. The team is using the PEAC INCITE award as the source of computing cycles. Data collection to support the modeling efforts will follow shortly. PERI also hosted a brief meeting in Seattle, WA, after the 2008 SciDAC conference to introduce the participants and to provide a common understanding of our goals.

Recognizing that the Architecture Tiger Team required a significant increase in labor, DOE augmented LBNL's 2008 budget with $500,000. LBNL will redistribute this money to some of the other PERI institutions that are taking major roles in the Tiger Team. Specifically, this money will fund activities at the University of Oregon, for the first time as a formal part of PERI, and will enable increased participation of LBNL, Chicago, LLNL and SDSC. LBNL, which is adding

Hongzhang Shan and possibly Leonid Oliker to the project, will have a role in GTC, along with Shirley Moore from the University of Tennessee, and will manage the subcontracts to Oregon, ANL, LLNL and SCSD. Oregon will continue to provide performance analysis of the applications and will work on establishing the performance database used to collect these analysis results. Chris Daley of the University of Chicago is a member of the FLASH code team and will build on the initial FLASH measurement work of ANL's Katherine Riley. LLNL, which is adding Martin Schulz to the project, will lead interactions with S3D. Finally, SDSC is funding Karen Karavanic of Portland State University (visiting with SDSC), who will participate in the modeling activity and the performance database activity.

Liaisons

In order to allocate PERI engagement resources and disseminate PERI technology efficiently, PERI has also established some long-term relationships with projects having clearly identified performance needs and a desire to work with PERI. These interactions are defined and maintained by individual PERI personnel, who are assigned to be the liaisons between the science application project and PERI. The initial assignments were motivated by the information collected in the PERI Application Survey. The nature and number of liaison interactions have since evolved based on updates to the application surveys and requests from science application personnel, from DOE computing centers, and from DOE headquarters.

The nature of the interaction between PERI and a Science Application project is unique to each project and varies over time. Some of the PERI liaisons are engaged actively, helping enhance the performance of their colleague's codes, bringing in other PERI personnel resources and expertise as needed, and educating PERI researchers as to the important performance issues. Other interactions are more passive, with the liaison simply staying in touch, tracking performance needs, advising on performance issues, and looking for opportunities for PERI to contribute, but not directly engaged for the moment in tuning.

Current liaison assignments and the approximate nature of the interactions as of July, 2008 are presented below.

Active liaisons (close collaboration on well-defined performance issues):

1. Advanced Methods for Electronic Structure Application
   Application PIs: Fann (ORNL) and Harrison (ORNL)
   PERI liaison: Roth (ORNL)
2. Center for Plasma Edge Simulation
   Application PI: Chang (NYU)
   PERI liaison: Worley (ONRL)
3. Simulations of Turbulent Flows with Strong Shocks and Density Variations
   Application PI: Lele (Stanford)
   PERI liaison: de Supinski (LLNL)
4. Modeling Multiscale-Multiphase-Multicomponent Subsurface Reactive Flows using Advanced Computing
   Application PI: Lichtner (LANL)
   PERI liaison: Mahinthakumar (NCSU)
5. Linear Scale Electronic Structure Calculations for Nanostructures
   Application PI: Wang (LBNL)
   PERI liaison: Bailey (LBNL)

6. Hierarchical Petascale Simulation Framework for Stress Corrosion Cracking
   Application PI: Vashishta (USC)
   PERI liaison: Hall (Utah)
7. Community Petascale Project for Accelerator Science and Simulation
   Application PI: Spentzouris (Fermi)
   PERI liaison: Norris (ANL)

Passive liaisons (monitoring performance needs, providing advice or assistance as needed):

1. A Scalable and Extensible Earth System Model for Climate Change Science
   Application PI: Drake (ORNL) and Jones (LANL)
   PERI Liaison: Worley
2. Framework Application for Core-Edge Transport Simulations
   Application PI: Cary (TechX)
   PERI liaison: Worley (ORNL)
3. Hybrid Numerical Methods for Multiscale Simulations of Subsurface
   Biogeochemical Processes
   Application PI: Scheibe (PNNL)
   PERI Liaison: Mahinthakumar (NCSU)
4. Multidimensional Simulations of Core-Collapse Supernovas
   Application PI: Mezzacappa (ORNL)
   PERI Liaison: Fowler (UNC)
5. National Computational Infrastructure for Lattice Gauge Theory
   Application PI: Sugar (UC-Santa Barbara)
   PERI liaison: Fowler (UNC)
6. Devine (SNL) / Zoltan
   PERI liaison: Hovland/Norris (ANL)

Recent examples of PERI contributions to the performance diagnosis and optimization of science application codes through the liaison activities are as follows:

1. As noted above, PERI researchers at LBNL have worked with Lin-Wang Wang and colleagues on the "LS3DF" code, which is an advanced electronic structure code. Wang has garnered several INCITE awards on the NERSC system and also NCCS for running it on various materials science applications. LS3DF is notable in that it possesses a linearly scaling feature – linearly increasing the size of the problem domain can be achieved by a nearly linear increase in computational cost, whereas in other codes for similar applications, computational cost typically scales as n^3 or more. In addition, LS3DF is very well suited to highly parallel computing.

During the past year, PERI researchers at LBNL have worked with Wang and his colleagues to analyze the performance of LS3DF and to identify potential performance improvements. Among the bottlenecks noted was some load imbalance, as well as the usage of disk I/O to exchange data at one key step. Responding to this analysis, LBNL researchers assisted with a major revision of this code, including changing the design of the code to exchange data via the network in the system, rather than by I/O. Other improvements were made as well.

Before the revision, the code was limited to running on less than 2,000 nodes, and at only modest performance rates. Now the code has been tested on at least 30,720 computational cores of the Jaguar system at ORNL, achieving 60.3 TFlop/s, which is 23.4% of peak for this many cores. After making some additional improvements, these researchers were able to run it on 131,072

cores of the BlueGene/P system at ANL, achieving 107.5 Tflop/s, or 24.2% of peak. This is the first time an *ab initio* code of this class to achieve 100+ TFlop/s performance. A paper describing this code, performance analyses, changes made and results has been accepted for SC08 as a Gordon Bell Prize finalist in the "Special" category [Wang].

2. As part of the active liaison relationship with the project Hierarchical Petascale Simulation Framework for Stress Corrosion Cracking application, USC/ISI worked with Aiichiro Nakano's group at USC to optimize a quantum mechanical code. Four key computational kernels were identified that exhibited opportunities for improved performance. Two were replaced by linear algebra libraries, but the other two were identified as being well-suited for compiler optimization. Through this hand-tuning, the performance of the application improved by 19.8% on an Intel Core2Duo. The analysis also indicated that this code is an excellent candidate for further optimization from applying a number of automatic performance tuning technologies. Our next steps will be to leverage the Optimized Sparse-Kernel Interface (OSKI), developed by Berkeley's BeBOP team, for the optimization of the sparse-matrix kernel an then the development of a data-copy library that will include data layout optimizations targeting stencil computations.

3. NCSU has continued to track performance during the active development of the groundwater modeling code PFLOTRAN, identifying and diagnosing performance problems as they arise and communicating these results to the application developers. Recent results include the identification of scalability problems when using the MPI-IO for large processor counts on the Cray XT4 at ORNL. Significant performance improvement in the I/O portion was achieved by experimenting with Lustre settings, achieving nearly a factor of 10 improvement in I/O performance in some cases.

Other External Interactions

In addition to creating formal liaison relationships with SciDAC application code teams and forming Tiger Teams, PERI researchers have worked with a variety of other DOE-sponsored computational projects. Some of the highlights of this additional application engagement are presented below.

As mentioned above, work on the University of Oregon's TAU Performance System now receives PERI funding. In this regard, during the past year TAU has been successfully used to evaluate the performance of the following codes on Intrepid:

1. Quantum Lattice Gas (qlg) code (M. Soe, Rogers State University).
2. UNRES/MD molecular dynamics code (Adam Liwo, Cornell).
3. FronTier Front Tracking code (Tulin Kaman, SUNY Stony Brook).
4. DCA++ (Gonzalo Alvarez, ORNL).
5. Pelegant, Yusong (Wang, APS, ANL).
6. MFIX (Aytekin Gel, NETL).
7. DLPOLY (Shashi Adiga, ANL Material Science
8. AW (M. Dulak, DTU, J. Greeley, N. Romero, ANL)

Both directly and through the PEAC End Station, PERI interacts frequently with computer center staff at the ALCF, the NLCF, and NERSC, and with IBM and Cray. For example, PERI ports and maintains performance tools required by PERI research and engagement activities, and makes these available to the centers' users. PERI engagement activities of scaling and analyzing

Institute Management

PERI is a widely distributed project, so we go to great lengths to stay connected and coordinated. We have teleconferences, a meeting each year at the SC conference, and two annual PERI team meetings. Teleconferences are held approximately every two weeks. The precise schedule varies depending on the needs of the project as well as the availability of the PIs. PERI team meetings are scheduled every September and March for the five-year duration of SciDAC-2. Each of PERI's ten initial institutions will take a turn hosting the meeting. UNC hosted our third meeting Sep 19-21, 2007, and SDSC hosted the fourth meeting, Feb 24-26, 2008. A meeting was also held at SC|07 on Monday, Nov 12, 2007.

Robert Lucas of USC/ISI is the overall manager of the PERI institute, assisted by David H. Bailey of LBNL. This includes monitoring project activity, conducting biweekly telecons, overseeing twice-yearly project meetings, preparing reports, and representing PERI in various community activities. In particular, Bailey is overseeing a budget augmentation that has been granted to support the new Architecture Tiger Team activity.

Dan Gunter of LBNL has maintained and enhanced the PERI website and wiki. He has performed server administration to make sure that the wiki software, MediaWiki, has been kept up-to-date and secure. The website has been organized to follow DOE guidelines, and keeps a current list of liaison activities, mailing lists, and PERI publications. The wiki is used daily by PERI PIs, and Dan has organized the content to reflect the current activities of the PERI tiger team efforts clearly and to provide a "home page" for the PERI search, database, and transformations working groups. An extension of the PERI publications page is under way to support Mary Hall's annotated bibliography of auto-tuning publications.

Outreach to SciDAC-2 and the Performance Community

Although PERI includes many of the researchers working in performance evaluation, there are other projects in that area, some funded by DOE SC. PERI's intent is to be inclusive. We have established collaborative relationships with several other performance evaluation projects. These collaborations include access to our benchmark applications as well as substantive relationships. For example, we are also collaborating closely with the TASCS CET and CSCAPES Institute.

FY09 Plans

PERI's FY09 plans are still consistent with those we originally proposed, though we have encountered some schedule slippage due to the no-cost extension and resulting funding interruption that some of the PERI institutions received last summer. One significant exception is that we are redirecting a substantial amount of our modeling research due to Architecture Tiger Team effort. Here is a brief summary of our FY09 plans, presented institution-by-institution:

In FY09, Argonne will continue the development of open-source computational quality of service infrastructure (CQoS) for both component and non-component scientific applications. This is joint work between PERI and the TASCS SciDAC projects. In addition to the current extensive performance database interfaces, we will develop additional, simpler, interfaces to the performance database targeting high-level access by the application developers. We will define interfaces and implement components for automating the training phase of the auto-tuning process for components. Currently, the process of defining the smallest possible number of experiments required to gather performance data based on hardware counters is arduous and time-consuming and must be performed manually for each architecture. We will create infrastructure that automates at least some portions of the counter selection process, and then makes the results available for all users on the machine, so that one can easily minimize the number of training runs required prior to the analysis phase of each application. We will also expand the current PerfExplorer-based analysis interface to accommodate more general analysis algorithms not

necessarily based on the PerfExplorer infrastructure. In collaboration with the University of Oregon's Performance Research Lab, we will create robust versions of the current prototype analysis components, which are based on the machine learning technology in Weka [Weka]. We will lead the performance evaluation of FLASH applications on the Argonne's Blue Gene/P and ORNL's Cray XT3 architectures, including parallel scalability and single-node performance studies.

In FY09 LBNL will: (a) assist Lucas in overall PERI project management; (b) specifically oversee the PERI augmentation and the four institutions participating in the augmentation; (c) analyze GTC as part of the application tiger team activity; (d) continue efforts in tuning the LS3DF code (or possibly another key DOE application); (e) develop multicore automatic tuning capabilities and test these capabilities on several application kernels.

In FY09, LLNL will: (a) focus on the use of POET as an automated technique to generate kernels required to empirical tuning; (b) continue investigations into parametric performance models, specifically integrating these models with cross-platform modeling methodologies; (c) work closely with other PERI researchers to incorporate these techniques into integrated performance prediction tools that exploit complementary techniques; (d) apply these techniques to at least one SciDAC application; (e) continue various efforts to develop models of behavior within MPI applications; (f) extend our MPI tracing mechanism to capture additional communication patterns; (g) add capabilities that capture the performance distribution of MPI events; (h) continue to explore advanced techniques to capture complete traces including time stamps while preserving nearly constant scalability; (i) continue to lead the PERI tiger team activity.

In FY09 ORNL will: (a) continue development of open source interconnect simulator by extending the existing topology and routing configurations beyond Infiniband and Cray SeaStar to those interconnects needed by the architecture tiger team; (b) add I/O performance prediction capabilities to Modeling Assertions and interconnect simulator; (c) develop scalable symbolic models using PERI Modeling Assertions of the architecture tiger team applications in order to drive the simulator performance estimates; (d) compare simulations and models to empirical measurements on large scale systems; (e) develop sensitivity models for MA scalable symbolic models which easily identify scaling bottlenecks; (f) continue to provide liaison support to Climate, Fusion, and Materials Science application projects and, in collaboration with NCSU, with Groundwater application projects; (g) continue to lead PERI engagement activities; (h) continue to lead the PEAC INCITE project.

In FY09 NCSU will continue to provide liaison support for SciDAC groundwater application projects. Liaison activities will involve extensive performance analysis and optimization of the groundwater codes, PFLOTRAN (active liaison) and STOMP (passive liaison), on Cray XT4 (Jaguar) and BlueGeneP (Intrepid). As in the past, more emphasis will be placed on PFLOTRAN due to the maturity of the code and liaison status. NCSU will continue to have regular interaction with the application teams and PETSC developers (both codes use PETSC solvers) to communicate results and to obtain code updates. Since performance optimizations within PETSC are expected to be handled by the PETSC team, a major focus for FY09 will be the other aspects of the codes including I/O and user routines.

In FY09, Rice will continue to work with Cray and IBM to resolve operating system problems that cause statistical sampling using hardware counters to fail on the leadership computing platforms. Cray is expected to deploy fixes for this problem in Compute Node Linux 2.1 in Q408. IBM also expects to release a new Blue Gene OS release that will include fixes to support hardware counter-based sampling Q408. Rice plans to deploy HPCToolkit on the leadership computing platforms once the vendor OS issues are resolved. As part of this effort, Rice will continue to enhance capabilities to support call path profiling of fully optimized code in

HPCToolkit. Rice will also work to extend techniques for performance analysis of multithreaded programs to support analysis of OpenMP code as part of hybrid programs consisting of MPI+OpenMP. Finally, Rice will continue to work with SciDAC and INCITE application teams as part of engagement activities.

SDSC plans for FY09 include the following: (a) complete alpha testing of the PSINS open-source network simulator that will consume network traces such as those being generated/extrapolated by LLNL – this will enable predictions to processor counts beyond those currently available for tracing; (b) make a beta release of PSINS; (c) carry out fundamental R&D to extrapolate memory traces to larger inputs and processor count; (d) work closely with the rest of the Architecture Tiger Team, leveraging items a and c, to make performance predictions of future systems in the 2009-2011 timeframe; (e) extend our models to more futuristic systems.

The University of Maryland team will: (a) focus on efforts to integrate the PERI-wide auto-tuning framework with an emphasis on the empirical search component; (b) finish integration and start evaluation of the UMD developed Active Harmony and the USC/ISU Chill frameworks; (c) continue to lead development of the PERI wide search API and associated constraint language; and (d) support tiger teams efforts as appropriate.

The University of Oregon's activities in FY09 will continue support for using the TAU performance system in measurement and analysis of petascale applications, particularly as targeted by the Architecture Tiger Team efforts. UO will maintain close involvement with the performance measurement and analysis of S3D and GTC application. In addition, UO will continue to work with the PERI-DB group on the performance database integration efforts, with PerfDMF being used to create a reference implementation to store the Architecture Tiger Team performance data. UO will apply its PerfExplorer performance data mining tool for analysis of S3D and GTC performance data, including comparison, correlation, clustering, and factor analysis.

UNC/RENCI will: (a) focus on a new generation of measurement and analysis techniques that focus on contention for shared resources on multi-core systems; (b) develop an integrated power and performance measurement framework; (c) work on runtime power and performance adaption techniques; (d) continue work on scalable measurement and analysis; (e) host the PERI performance database server; and (f) apply the results of items (a), (b), and (c) in outreach efforts focusing on multi-core performance issues.

USC/ISI will: (a) provide overall PERI project management; (b) continue the development of the transformation API and make the API available to the broader autotuning community; (c) study a set of kernels of interest (stencil computations in a quantum chemistry code from SciDAC's Petascale Simulation Framework for Stress Corrosion Cracking application, stencil computations in PETSc, and kernels from GTC) to determine the requirements for autotuning and relevant capabilities of CHiLL; (d) in collaboration with Utah, work on the development of a data copy library with support for stencil computations; (e) in collaboration with Utah and Maryland, continue the integration of CHiLL with Active Harmony; (f) in collaboration with Utah, contribute to the regular software releases of CHiLL.

UTK will: (a) continue development and refinement of PAPI on leadership class machines to support PERI performance modeling and auto-tuning activities, (b) continue research on empirical search techniques for auto-tuning and integrate those techniques into the PERI auto-tuning framework, (c) continue research on optimization techniques for muticore architectures and integrate those techniques into the PERI auto-tuning framework, (d) continue active participation in the PERI tiger team and performance database efforts.

Utah will: (a) lead the PERI autotuning subgroup, organizing reports, posters and papers, and involving external collaborators; (b) integrate threading mechanisms into the autotuning compiler technology; (c) in collaboration with USC/ISI, work on the development of a data copy library with support for stencil computations; (d) in collaboration with USC/ISI and Maryland, continue the integration of CHiLL with Active Harmony; (e) in collaboration with USC/ISI, contribute to the regular software releases of CHiLL.

PERI-Funded Publications

[Adhianto] Laksono Adhianto, Sinchan Banerjee, Michael Fagan, Mark Krentel, Gabriel Marin, John Mellor-Crummey, Nathan Tallent, "HPCToolkit: Tools for performance analysis of optimized parallel programs," submitted to *Concurrency and Computation: Practice and Experience*, submitted, Aug 2008.

[Ahn] Dong H. Ahn, Dorian C. Arnold, Bronis R. de Supinski, Gregory L. Lee, Barton P. Miller and Martin Schulz, "Overcoming Scalability Challenges for Tool Daemon Launching," 2008 *International Conference on Parallel Processing (ICPP-08)*, Portland, OR, Sep 8-12, 2008.

[Alam2007a] S. R. Alam, N. Bhatia, and J. S. Vetter, "An Exploration of Performance Attributes for Symbolic Modeling of Emerging Processing Devices," *3rd International High Performance Computation Conference (HPCC)*, 2007.

[Alam2007b] S. R. Alam, N. Bhatia, and J. S. Vetter, "Sensitivity Analysis of Biomolecular Simulations using Symbolic Models," *7th International Conference on BioInformatics and BioEngineering*, Boston, 2007.

[Alam2007c] S. R. Alam, J. S. Meredith and J. S. Vetter, "Balancing Productivity and Performance on the Cell Broadband Engine," *IEEE Annual International Conference on Cluster Computing*, 2007.

[Alam2008a] S. R. Alam, P. Agarwal, H. Ong, S. Hampton and J.S. Vetter, "Impact of multicores on large-scale molecular dynamics simulations," *IEEE International Workshop on High Performance Computational Biology (HiCOMB)*, in conjunction with IPDPS, 2008, pp. 1-7.

[Alam2008b] S. R. Alam, R. F. Barrett, M. R. Fahey et al., "An Evaluation of the Oak Ridge National Laboratory Cray XT3," *International Journal of High Performance Computing Applications*, vol. 22, no.1, pg. 52-80, 2008.

[Alam2008c] S. R. Alam, R. F. Barrett, M. Eisenbach, M. R. Fahey, R. Hartman-Baker, J. A. Kuehn, S. W. Poole, R. Sankaran and P. H. Worley, "The Cray XT4 Quad-core : A First Look," *50th Cray User Group Conference*, Helsinki, Finland, May 5-8, 2008.

[Alam20008d] S. R. Alam, R. F. Barrett, M. Bast, M. R. Fahey, J. Kuehn, C. McCurdy, J. Rogers, P. C. Roth, R. Sankaran, J. S. Vetter, P. H. Worley and W. Yu, "Early Evaluation of IBM BlueGene/P," *SC'08*, Austin, TX, Nov. 15-21, 2008, to appear.

[Bailey] David H. Bailey, Robert Lucas, Paul Hovland, Boyana Norris, Kathy Yelick, Dan Gunter, Bronis de Supinski, Dan Quinlan, Pat Worley, Jeff Vetter, Phil Roth, John Mellor-Crummey, Allan Snavely, Jeff Hollingsworth, Dan Reed, Rob Fowler, Ying Zhang, Mary Hall, Jacque Chame, Jack Dongarra, Shirley Moore, "Performance Engineering: Understanding and Improving the Performance of Large-Scale Codes," *CT Watch Quarterly*, vol. 3, no. 4, pg. 18–23, Nov 2007.

[Barnes] Bradley Barnes, Barry Rountree, David K. Lowenthal, Jaxk Reeves, Bronis R. de Supinski and Martin Schulz, "A Regression-Based Approach to Scalability Prediction," *21nd International Conference on Supercomputing (ICS 2008)*, Kos, Greece, June 7-12, 2008.

[Basili] Victor R. Basili, Jeff Carver, Daniela Cruzes, Lorin Hochstein, Jeffrey K. Hollingsworth, Forrest Shull, Marvin V. Zelkowitz, "Understanding The High Performance Computing Community: A Software Engineer's Perspective", *IEEE Software*, Jul 2008.

[Bui] V. Bui, B. Norris, K. Huck, L. C. McInnes, L. Li, O. Hernandez, and B. Chapman. A Component Infrastructure for Performance and Power Modeling of Parallel Scientific Applications. *Proceedings of Component-Based High Performance Computing Workshop*, Oct 14-17, 2008, Karlsruhe, Germany, to appear.

[Carrington] Laura Carrington, Dimitri Komatitsch, Michael Laurenzano, Mustafa Tikir, David Michéab, Nicolas Le Goff, Allan Snavely and Jeroen Tromp, "High-Frequency Simulations of Global Seismic Wave Propagation Using SPECFEM3D_GLOBE on 62K Processors," *SC08*, Nov 2008, to appear.

[Chame] Jacqueline Chame, Chun Chen, Jack Dongarra, Mary Hall, Jeff Hollingsworth, Paul Hovland, Shirley Moore, Keith Seymour, Jaewook Shin, Ananta Tiwari, Sam Williams, Haihang You, "PERI Auto-tuning", *SCIDAC Review* 2008 (to appear).

[Curtis-Maury2007] Matthew Curtis-Maury, Karan Singh, Sally A. McKee, Filip Blagojevic, Dimitrios S. Nikolopoulos, Bronis R. de Supinski and Martin Schulz, "Identifying Energy-Efficient Concurrency Levels Using Machine Learning," *International Workshop on Green Computing (GreenCom'07)*, Austin, TX, Sep 17, 2007.

[Curtis-Maury2008] Matthew Curtis-Maury, Ankur Shah, Filip Blagojevic, Dimitrios S. Nikolopoulos, Bronis R. de Supinski and Martin Schulz, "Prediction Models for Multi-dimensional Power-Performance Optimizations on Many Cores," *17th International Conference on Parallel Architectures and Compilation Techniques (PACT-2008)*, Toronto, Canada, October 25–29, 2008.

[deSupinski2007] Bronis R. de Supinski, Jeff Hollingsworth, Shirley Moore and Patrick Worley, "Results of the PERI Survey of SciDAC Applications," SciDAC 2007, Boston, MA, June 24–28, 2007.

[deSupinski2008a] Bronis R. de Supinski, Robert J. Fowler, Todd Gamblin, Frank Mueller, Prasun Ratn and Martin Schulz, "An Open Infrastructure for Scalable, Reconfigurable Analysis," *International Workshop on Scalable Tools for High-End Computing (STHEC)*, Kos, Greece, Jun 7, 2008.

[deSupinski2008b] Bronis R. de Supinski, Rob Fowler, Todd Gamblin, Frank Mueller, Prasun Ratn and Martin Schultz, "An Open Infrastructure for Scalable, Reconfigurable Analysis, *International Workshop on Scalable Tools for High-End Computing (STHEC 2008)*, ACM/SIGARCH, Jul 2008.

[Dongarra2007] Jack Dongarra, Dennis Gannon, Geoffrey Fox, and Ken Kennedy, "The Impact of Multicore on Computational Science Software," *CTWatch Quarterly*, vol. 3, no. 1, Feb 2007.

[Dongarra2008] J. Dongarra, R. Graybill, W. Harrod et al., "DARPA's HPCS Program: History, Models, Tools, Languages," *Advances in Computers*, vol. 72, M. V. Zelkowitz, ed. London: Academic Press, Elsevier, 2008.

[Feng2008] Wu Feng, Robert J. Fowler, Mark K. Gardner, Song Huang, Allan Porterfield, "Multi-Source Event Generation And Analysis For Performance Understanding In Large-Scale Environments", *ORNL Fall Creek Falls Conference*, Sep 2008, poster.

[Fowler2008a] Robert J. Fowler, Todd Gamblin, Gopi Kandaswamy, Anirban Mandal, Allan K. Porterfield, Lavanya Ramakrishnan and Daniel A. Reed, "Challenges of Scale: When All Computing Becomes Grid Computing," in Lucio Grandinetti, editor, *High Performance Computing and Grids in Action, Advances in Parallel Computing*, IOS Press, Amsterdam, Mar 2008.

[Fowler2008b] Robert J. Fowler, Todd Gamblin, Allan K. Porterfield, Patrick Dreher, Song Huang and Balint Joo, "Performance Engineering Challenges: The View from RENCI," *Journal of Physics: Conference Series*, pg. 5, 2008, to appear.

[Fuerlinger] K. Fuerlinger and S. Moore, "Detection and Analysis of Iterative Behavior in Parallel Applications," *2008 International Conference on Computational Science (ICCS 2008)*, Krakow, Poland, Lecture Notes in Computer Science 5103, pg. 261-267, Jun 2008.

[Gamblin2008a] Todd Gamblin, Bronis R. de Supinski, Martin Schulz, Robert J. Fowler and Daniel A. Reed, "Scalable Load Balance Measurement for SPMD Codes," *SC2008*, Austin, Texas, November 15–21, 2008.

[Gamblin2008b] Todd Gamblin, Rob Fowler, and Daniel A. Reed, "Scalable Methods for Monitoring and Detecting Behavioral Classes in Scientific Codes," in *Proceedings of the International Parallel and Distributed Processing Symposium 2008*, Miami, FL, Apr 2008.

[Grbovic] Jelena Pjesivac-Grbovi´c, Thara Angskun, George Bosilca, Graham E. Fagg, Edgar Gabriel, and Jack J. Dongarra, "Performance Analysis of MPI Collective Operations," *Cluster Computing Journal*, vol. 10 (2007), pg. 127-143.

[Grobelny] E. Grobelny, D. Bueno, I. Troxel, A. D. George, and J. S. Vetter, "FASE: A Framework for Scalable Performance Prediction of HPC Systems and Applications," *Simulation*, vol. 83, no. 10, pg. 721-45, 2007.

[Huck] K. A. Huck, O. Hernandez, V. Bui, S. Chandrasekaran, B. Chapman, A. D. Malony, L. C. McInnes and B. Norris, "Capturing Performance Knowledge for Automated Analysis," *(SC'08),* 2008, to appear.

[Ipek] Engin Ipek, Sally A. McKee, Karan Singh, Rich Caruana, Bronis R. de Supinski and Martin Schulz, "Efficient Architectural Design Space Exploration via Predicitive Modeling," *ACM Transactions on Architecture and Code Optimization*, Vol. 4, No. 4, Jan 2008, pg. 1-33.

[Jagode] H. Jagode and J. Hein, "Custom Assignment of MPI Ranks for Parallel Multi-dimensional FFTs: Evaluation of BG/P versus BG/L," *2008 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA-08)*, Sydney, Australia, Springer, InderScience, Dec 10-12, 2008.

[Lee] Gregory L. Lee, Dong H. Ahn, Dorian C. Arnold, Bronis R. de Supinski, Matthew Legendre, Barton P. Miller, Martin Schulz and Ben Liblit, "Lessons Learned at 208K: Towards Debugging Millions of Cores," *SC'08*, Austin, Texas, Nov 15–21, 2008.

[Lee] Gregory Lee, Dong H. Ahn, Dorian Arnold, Bronis R. de Supinski, Barton P. Miller and Martin Schulz, "Benchmarking the Stack Trace Analysis Tool for BlueGene/L," *International Conference on Parallel Computing 2007 (ParCo 2007)*, Aachen, Germany, September 4–7, 2007.

[Lee] Gregory Lee, Martin Schulz, Dong Ahn, Andrew Bernat, Bronis R. de Supinski, Steven Ko and Barry Rountree, "Dynamic Binary Instrumentation and Data Aggregation on Large Scale Systems," *International Journal of Parallel Programming*, vol. 35, no. 3, Jun 2007, pg. 207-232.

[Li] Li Li, B. Norris, H. Johansson, L. McInnes and J. Ray, "Component Infrastructure for Managing Performance Data and Runtime Adaptation of Parallel Applications," *9th International Workshop on State-of-the-Art in Scientific and Parallel Computing(PARA08)*, May 13-16, 2008, Trondheim, Norway.

[Lively] C. Lively, S. R. Alam, V. Taylor and J. S. Vetter, "A Methodology for Developing High Fidelity Communication Models for Large-scale Applications Targeted on Multicore Systems," *20th International Symposium on Computer Architecture and High Performance Computing*, 2008.

[Marin2008a] Gabriel Marin, Guohua Jin, and John Mellor-Crummey, "Managing Locality in Grand Challenge Applications: A Case Study of the Gyrokinetic Toroidal Code," *SciDAC 2008*, *Journal of Physics: Conference Series* 125 012087.

[Marin2008b] G. Marin and J. Mellor-Crummey, "Pinpointing and Exploiting Opportunities for Enhancing Data Reuse, *International Symposium on Performance Analysis of Systems and Software*, April 2008.

[McCurdy] C. McCurdy, A. Cox, and J.S. Vetter, "Investigating the TLB Behavior of High-end Scientific Applications on Commodity Microprocessors," *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Austin, TX, 2008.

[Mellor-Crummey2007a] John Mellor-Crummey, Peter Beckman, Jack Dongarra, Ken Kennedy, Barton Miller and Katherine Yelick, "Software for Leadership-Class Computing," *SciDAC Review*, Fall 2007, pg. 36-45, to appear, available at http://www.scidacreview.org.

[Mellor-Crummey2007b] John Mellor-Crummey, "Harnessing the Power of Emerging Petascale Platforms," *SciDAC 2007*, *Journal of Physics: Conference Series* 78 (2007) 012048.

[Mellor-Crummey2007c] John Mellor-Crummey, Peter Beckman, Keith Cooper, Jack Dongarra, William Gropp, Ewing Lusk, Barton Miller, Katherine Yelick, "Creating Software Tools and Libraries for Leadership Computing," *CTWatch Quarterly*, Nov 2007.

[Michalakes2008] J Michalakes, J Hacker, R Loft, M O McCracken, A Snavely, N J Wright, T Spelce, B Gorda and R Walkup, "WRF Nature Run, *2008 Journal of Physics: Conference Series* 125 012022.

[Porterfield] Allan Porterfield, Robert Fowler and Mark Neyer, "MAESTRO: Dynamic Runtime Power Control," in *Workshop on Managed Multicore systems (MMCS)*, Boston, MA, Jun 2008.

[PAPI] PAPI 3.6.1 Release Notes, http://icl.cs.utk.edu/papi/

[Preissl] Robert Preissl, Thomas Koeckerbauer, Martin Schulz, Dieter Kranzlmueller, Bronis R. de Supinski and Daniel J. Quinlan, "Detecting Patterns in MPI Communication Traces," 2008 *International Conference on Parallel Processing (ICPP-08)*, Portland, OR, September 8-12, 2008.

[Preissl] Robert Preissl, Martin Schulz, Dieter Kranzlmueller, Bronis R. de Supinski and Daniel J. Quinlan, "Using MPI Communication Patterns to Guide Source Code Transformations," *Tools for Program Development and Analysis in Computational Science*, Krakow, Poland, June 23-25, 2008.

[Preissl] Robert Preissl, Martin Schulz, Dieter Kranzlmueller, Bronis R. de Supinski and Daniel J. Quinlan, "Using MPI Communication Patterns To Guide Source Code Transformations," poster at *SC2007*, Reno, NV, November 10-16, 2007.

[Ratn] Prasun Ratn, Frank Mueller, Bronis R. de Supinski and Martin Schulz, "Preserving Time in Large-Scale Communication Traces," *Twenty Second International Conference on Supercomputing (ICS 2008)*, Kos, Greece, June 7-12, 2008.

[Roth2007a] P. C. Roth, “Characterizing the I/O Behavior of Scientific Applications on the Cray XT,” *Petascale Data Storage Workshop*, held in conjunction with *SC07*, Reno, Nevada, Nov2007.

[Roth2007b] P. C. Roth and J. S. Vetter, “Intel Woodcrest: An Evaluation for Scientific Computing,” *8th LCI International Conference on High-Performance Clustered Computing*, 2007.

[Rountree] Barry Rountree, David K. Lowenthal, Shelby Funk, Vincent W. Freeh, Bronis R. de Supinski and Martin Schulz, “Bounding Energy Consumption in Large-Scale MPI Programs,” *SC2007*, Reno, NV, November 10-16, 2007.

[Schulz] Martin Schulz and Bronis R. de Supinski, “PnMPI Tools: A Whole Lot Greater Than the Sum of Their Parts,” *SC2007*, Reno, NV, November 10-16, 2007.

[Schulz] Martin Schulz and Bronis R. de Supinski, “Practical Differential Profiling,” *Euro-Par 2007*, Rennes, France, August 28–31, 2007.

[Seymour2008a] Keith Seymour, Haihang You, and Jack Dongarra, “Search Techniques for Empirical Code Optimization”, *3rd International Workshop on Automatic Performance Tuning*, Oct 1, 2008, Tsukuba International Congress Center, EPOCHAL TSUKUBA, Japan, submitted.

[Seymour2008b] Seymour, K., You, H., Dongarra, J. “A Comparison of Search Heuristics for Empirical Code Optimization,” *3rd international Workshop on Automatic Performance Tuning*, Tsukuba, Japan, October 1st, 2008.

[Snavely2008] Allan Snavely, Laura Carrington, Bronis de Supinski, Jeffrey Vetter, “Performance Modeling: Impact of Architecture Trends on Applications”, *ASCR Computer Science Research Principal Investigators Meeting* (poster), Denver, Mar 31 - Apr 2, 2008.

[Song2007] Song, F., Moore, S., Dongarra, J. "L2 Cache Modeling for Scientific Applications on Chip Multi-Processors," *Proceedings of the 2007 International Conference on Parallel Processing*, Xi'an, China, IEEE Computer Society, September, 2007.

[Song2008] Song, F., Moore, S., Dongarra, J., "Analytical Modeling for Affinity-Based Thread Scheduling on Multicore Platforms," UT-CS-08-626, August 12, 2008.

[Song2009] Fengguang Song, Shirley Moore, and Jack Dongarra, “Analytical Modeling for Affinity-based Thread Scheduling on Multicore Platforms”, *14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2009)*, Feb 14-18, 2009, Raleigh, North Carolina, submitted.

[Tabatabaee] Vahid Tabatabaee, Jeffrey K. Hollingsworth, “Automatic Software Interference Detection in Parallel Applications”, *SC'07*, Reno, NV, Nov 2007.

[Tallent] Nathan Tallent, John Mellor-Crummey, Laksono Adhianto, Michael Fagan, and Mark Krentel 2008. “HPCToolkit: Performance Tools for Scientific Computing,” *SciDAC 2008, Journal of Physics Conference Series* 125 012088.

[Tikir2007] M. Tikir, L. Carrington, E. Strohmaier, A. Snavely, “A Genetic Algorithms Approach to Modeling the Performance of Memory-bound Computations”, *SC’07*, Nov 2007, Reno, pg. 82-94

[Tilson] Jeffery L. Tilson, Mark S. C. Reed and Robert J. Fowler, “Workflows for Performance Evaluation and Tuning, in *2008 IEEE International Conference on Cluster Computing (Cluster 2008)*, pg. 8, Tsukuba, Japan, Sep 2008, IEEE.

[Wang] Lin-Wang Wang, Byounghak Lee, Hongzhang Shan, Zhengji Zhao, Juan Meza, Erich Strohmaier, David Bailey, "Linearly Scaling 3D Fragment Method for Large-Scale Electronic Structure Calculations," *SC'08*, to appear, Nov 2008, available at http://crd.lbl.gov/~dhbailey/ls3df-main.pdf.

[Weka] The Weka Machine Learning Project. http:// www. cs.waikato.ac.nz/~ml/index.html.

[Williams2007] S. Williams, L. Oliker, R. Vuduc, J. Shalf, K. Yelick and J. Demmel, "Optimization of Sparse Matrix-Vector Multiplication on Emerging Multicore Platforms," *SC07*, ACM/IEEE, Nov 2007.

[Williams2008a] S. Williams, J. Carter, L. Oliker, J. Shalf and K. Yelick, "Lattice Boltzmann Simulation Optimization on Leading Multicore Platforms," *International Parallel & Distributed Processing Symposium (IPDPS)*, to appear, 2008. WINNER: Best paper, applications track.

[Williams2008b] Samuel Williams, Kaushik Datta, Jonathan Carter, Leonid Oliker, John Shalf, Katherine Yelick, David Bailey, "PERI - Auto-tuning Memory Intensive Kernels for Multicore," *SciDAC 2008*, available at http://crd.lbl.gov/~dhbailey/dhbpapers/scidac08_peri.pdf.

[Wolf2007] Felix Wolf, Bernd Mohr, Jack Dongarra, Shirley Moore, "Automatic Analysis of Inefficiency Patterns in Parallel Applications," *Concurrency and Computation: Practice and Experience*, vol. 19, no. 11, pg. 1481-1496, Aug 2007.

[Wolf2008] F. Wolf, B. Wylie, E. Abraham, D. Becker, W. Frings, K. Fuerlinger, M. Geimer, M. Hermanns, B. Mohr, S. Moore, M. Pfeifer, Z. Szebenyi, "Usage of the Scalasca Toolset for Scalable Performance Analysis of Large-scale Parallel Applications," *2nd International Workshop on Tools for High Performance Computing*, Resch, Keller, Himmler, Krammer, Schulz, eds. Stuttgart, Germany, Springer, pg. 157-167, Jul 2008.

[Worley] P.H. Worley, "Early Evaluation of the IBM BG/P," *LCI International Conference on High Performance Clustered Computing*, Urbana, IL, Apr 29-May 1, 2008. Best paper award.

[Yu2007] W. Yu, S. Oral, J. Vetter and R. Barrett, "Efficiency Evaluation of Cray XT Parallel IO Stack," *Cray User Group Meeting (CUG 2007)*, Seattle, WA, 2007.

[Yu2008] W. Yu, J. S. Vetter and H.S. Oral, "Performance Characterization and Optimization of Parallel I/O on the Cray XT," *22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2008)*, 2008.

Students Graduated

Samuel Williams of UC Berkeley, who has made significant contributions to PERI (see references above) is finishing his studies as of September 2008.

Michael O. McCracken of UC San Diego, who used PERI modeling techniques in tuning WRF (see references above) is finishing his studies as of December 2008.