Early Evaluation of the Cray XT3 at ORNL

J. S. Vetter S. R. Alam T. H. Dunigan, Jr.
M. R. Fahey P. C. Roth P. H. Worley

Oak Ridge National Laboratory
Oak Ridge, TN, USA 37831

ABSTRACT: Oak Ridge National Laboratory recently received delivery of a Cray XT3. The XT3 is Cray's third-generation massively parallel processing system. The system builds on a single processor node—the AMD Opteron—and uses a custom chip—called SeaStar—to provide interprocessor communication. In addition, the system uses a lightweight operating system on the compute nodes. This paper describes our initial experiences with the system, including micro-benchmark, kernel, and application benchmark results. In particular, we provide performance results for important Department of Energy applications areas including climate and fusion. We demonstrate experiments on the partially installed system, scaling applications up to 3,600 processors.

KEYWORDS: performance evaluation; Cray XT3; Red Storm; Catamount; performance analysis; benchmarking.

1 Introduction

Computational requirements for many large-scale simulations and ensemble studies of vital interest to the Department of Energy (DOE) exceed what is currently offered by any U.S. computer vendor. As illustrated in the DOE Scales report [30] and the High End Computing Revitalization Task Force report [17], examples are numerous, ranging from global climate change research to combustion to biology.

Performance of the current class of HPC architectures is dependent on the performance of the memory hierarchy, ranging from the processor-to-cache latency and bandwidth to the latency and bandwidth of the interconnect between nodes in a cluster, to the latency and bandwidth in accesses to the file system. With increasing chip clock rates and number of functional units per processor, this dependency will only increase. Single processor performance, or the performance of a small system, is relatively simple to determine. However, given reasonable sequential performance, the metric of interest in evaluating the ability of a system to achieve multi-Teraop performance is scalability. Here, scalability includes the performance sensitivity of variation in both problem size and the number of processors or other computational resources utilized by a particular application.

EARLY EVALUATION: This paper contains preliminary results from our early delivery system, which is smaller in scale than the final delivery system and which uses early versions of the system software.

ORNL has been evaluating these critical factors on several platforms that include the Cray X1 [1], the SGI Altix 3700 [13], and the Cray XD1 [14]. This report describes the initial evaluation results collected on an early version of the Cray XT3 sited at ORNL. Recent results are also publicly available from the ORNL evaluation web site [25]. We have been working closely with Cray, Sandia National Laboratory, and Pittsburgh Supercomputing Center, to install and evaluate our XT3.

2 Cray XT3 System Overview

The XT3 is Cray's third-generation massively parallel processing system. It follows the successful development and deployment of the Cray T3D and Cray T3E [28] systems. As in these previous designs, the system builds upon a single processor node, or processing element (PE). The XT3 uses a commodity microprocessor—the AMD Opteron—at its core, and connects these processors with customized interconnect. In the case of the XT3, Cray has designed an ASIC (application specific integrated circuit), called SeaStar, to manage the communication fabric.

2.1 Processing Elements

As Figure 1 shows, each PE has one Opteron processor with its own dedicated memory and communication resource. The XT3 has two types of

PEs: compute PEs and service PEs. The compute PEs are optimized for application performance by running a lightweight operating system kernel—Catamount. In contrast, the service PEs run SuSE Linux and are configured for I/O, login, network, or system functions.

The XT3 uses a blade approach for achieving high processor density per system cabinet. On the XT3, a compute blade hosts four compute PEs (or nodes), and eight blades are contained in one chassis. Each XT3 cabinet holds three chassis, for a total of 96 processors per cabinet. In contrast, service blades host two service PEs and provide PCI-X connections for extensibility, such as I/O.

The ORNL XT3 uses Opteron model 150 processors. As Figure 2 shows, this model includes an Opteron core, integrated memory controller, three 16b 800 Mhz HyperTransport (HT) links, a L1 cache, and a L2 cache. The Opteron core has a 2.4 Ghz clock, three integer units, and one floating-point unit which is capable of two floating-point operations per cycle [2]. Hence, the peak floating point rate of this processor is 4.8 GFLOPS.

The memory structure of the Opteron contains a 64KB 2-way associative L1 data cache, a 64KB 2-way associative L1 instruction cache, and a 1MB 16-way associative, unified L2 cache. The Opteron has 64b integer registers, 48b virtual addresses, 40b physical addresses, sixteen 64b integer registers, and sixteen 128b SSE/SSE2 registers. The memory controller data width is 128b. Each PE has 2 GB of memory but only 1 GB is usable with the current kernel. The memory DIMMs are 1 GB PC3200, Registered ECC, 18 x 512 mbit parts that support Chipkill. The peak memory bandwidth per processor is 6.4 GBps.

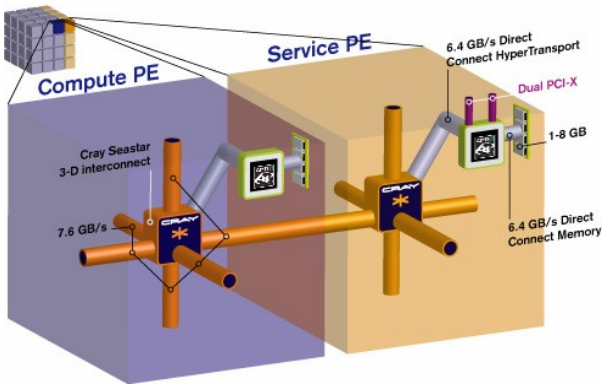

Figure 1: Cray XT3 Architecture (*Image courtesy of Cray*).

As a 100-series processor, the 150 does not support SMP configurations. Although it contains three HT links, none of these links support coherent HT. The benefits of supporting only uniprocessor configurations are realized in the memory subsystem because the 150 can have memory access latencies in the 50-60 ns range. In contrast, processors that support SMP configurations can have memory latencies that are considerably worse, due to the additional circuitry for coordinating memory accesses and managing the memory coherence across processors in the SMP. For comparison, current Intel processors use a separate chip—typically referred to as the 'Northbridge'—for the memory controller, which increases the latency for each memory access, in general.

The Opteron's processor core has a floating-point execution unit (FPU) that handles all register operations for x87 instructions, 3DNow! operations, all MMX operations, and all SSE and SSE2 operations. This FPU contains a scheduler, a register file, a stack renaming unit, a register renaming unit, and three parallel execution units. The first of these three execution units is known as the adder pipe (FADD); it contains a MMX ALU/shifter and floating-point adder. The next execution unit is known as the multiplier (FMUL); it provides the floating-point multiply/divide/square root operations and also an MMX ALU. The final unit supplies floating-point load/store (FSTORE) operations.

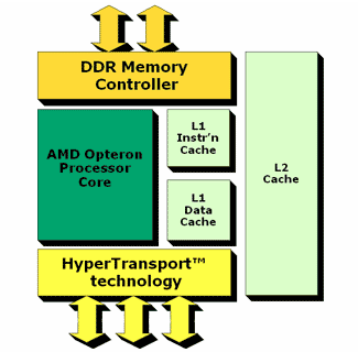

Figure 2: AMD Opteron Design (*Image courtesy of AMD*).

2.2 Interconnect

As Figure 1 illustrates, each Opteron processor is directly connected to the XT3 interconnect via a Cray SeaStar chip. This SeaStar chip is a routing and communications chip and it acts as the gateway to the XT3's high-bandwidth, low-latency interconnect. The PE is connected to the SeaStar chip with a 6.4 GBps HT path. The router in SeaStar provides six high-speed network links to connect to six neighbors in the 3D torus/mesh topology. Each of the six links has a peak bandwidth of 7.6 GBps. With this design, the Cray XT3 bypasses communication bottlenecks such as the PCI bus. The interconnect carries all message passing traffic as well as I/O traffic to the global parallel file system.

2.2.1 SeaStar

As described earlier, the SeaStar ASIC provides communication processing and routing facility on a single chip. Each communication chip is composed of:

- a **HyperTransport link** [3] --- this enables the chips inside of a computing system and network and communication devices to communicate with each other over parallel links without bus arbitration overheads.
- a **PowerPC 440 processor** --- the communications and management processor cooperates with the Opteron to synchronize and to schedule communication tasks.
- a **Direct Memory Access (DMA) engine** --- the DMA engine and the PowerPC processor work together to off-load message preparation and demultiplexing tasks from the Opteron processor.
- an **interconnect router** --- the router provides six network links to the six neighboring processors in the 3D torus topology. The peak bidirectional bandwidth of each link is 7.6 GB/s with a sustained bandwidth of around 4 GB/s.
- a **service por**t --- this port bridges between the separate management network and the Cray SeaStar local bus. The service port allows the management system to access all registers and memory in the system and facilitates booting, maintenance and system monitoring. Furthermore, this interface can be used to reconfigure the router in the event of failures.

2.3 Topology

The Cray XT3 sited at ORNL is currently a 40 cabinet system, with 3,748 compute PEs and 46 service PEs. These PEs are connected in a 10 x 16 x 24 (X x Y x Z) configuration with a torus in X and Z dimensions, and a mesh in the Y dimension.

Later this year, the system will be upgraded to 56 cabinets, totaling 5,212 compute processors and 82 service processors. It will be connected as 14 x 16 x 24 using a torus in X and Z with a mesh in Y. Depending on our experience as the full system is deployed, the Y dimension may also be converted to a torus.

2.4 System Software

The Cray XT3 inherits several aspects of its systems software approach from a sequence of systems developed and deployed at Sandia National Laboratories: ASCI Red [22], the Computational Plant [7, 26] (also known as Cplant), and Red Storm [5]. The XT3 uses a micro-kernel operating system on its compute PEs, a user-space communications library, and a hierarchical approach for scalable application start-up.

2.4.1 Operating Systems

The XT3 uses two different operating systems: Catamount on compute PEs and Linux on service PEs. Catamount is the latest in a sequence of micro-kernel operating developed at Sandia and the University of New Mexico, including SUNMOS [21], Puma [33], and Cougar. (Cougar is the product name for the port of Puma to the Intel ASCI Red system.) For scalability and performance predictability, each instance of the Catamount kernel runs one single-threaded process and does not provide services like demand-paged virtual memory that could cause unpredictable. Unlike the compute PEs, service PEs (i.e., login, I/O, network, and system PEs) run a full SuSE Linux distribution to provide a familiar and powerful environment for application development and for hosting system and performance tools.

2.4.2 Communication Library

The XT3 uses the Portals [8] data movement layer for flexible, low-overhead inter-node communication. Portals provide connectionless, reliable, in-order delivery of messages between processes. For high performance and to avoid unpredictable changes in the kernel's memory footprint, Portals deliver data from a sending process' user space to the receiving process' user space without kernel buffering. Portals support both one-sided and two-sided communication models. For flexibility, Portals support multiple higher-level communication protocols, including protocols for MPI message passing between application processes and for transferring data to and from I/O service PEs.

2.4.3 Scalable Application Launch

Like Cplant, the XT3 uses a hierarchical approach for scalable loading of parallel applications using the *yod* utility [5, 6]. On the XT3, launching a parallel application involves three steps:

1. *yod* determines the set of compute nodes allocated to the application;
2. *yod* delivers information about the application such as the user's environment and the application executable to the Process Control Thread (PCT) in the Catamount kernel running in the application's primary compute node; and
3. the PCT in the primary compute node multicasts the application information to the PCTs in the application's other compute nodes.

In the third step, a hierarchical communication structure (i.e., a multicast tree) is used for scalability. PCTs in

different branches of the tree can transmit messages in parallel to limit the latency for distributing job launch information.

2.5 Programming Environment

The Cray XT3 programming environment includes compilers, communication libraries, and correctness and performance tools [11]. The Portland Group's C, C++, and Fortran compilers are available. Cray-provided compiler wrappers ease the development of parallel applications for the XT3 by automatically including compiler and linker switches needed to use the XT3's communication libraries. The primary XT3 communication libraries provide the standard MPI-2 message passing interface and Cray's SHMEM interface. Low level communication can be performed using the Portals API (see Section 2.4.2). The Etnus TotalView debugger is available for the XT3, and Cray provides the Apprentice^2 tool for performance analysis.

2.6 Math Libraries

The primary math library is the AMD Core Math Library (ACML) version 2.5.0. It incorporates BLAS, LAPACK and FFT routines, and is designed to provide excellent performance on AMD platforms. This library is available both as a 32-bit library, for compatibility with legacy x86 applications, and as a 64-bit library that is designed to fully exploit the large memory space and improved performance offered by the new AMD64 architecture.

We also have installed and tested BLAS libraries for the AMD Opteron which were developed by Kazushige Goto [16].

3 Evaluation Overview

As a function of the Early Evaluation project at ORNL, numerous systems have been vigorously evaluated in the context of important DoE applications. Recent evaluations have included the Cray X1 [12], the SGI Altix 3700 [13], and the Cray XD1 [14].

The primary goals of these evaluations are to 1) determine the most effective approaches for using the each system, 2) evaluate benchmark and application performance, both in absolute terms and in comparison with other systems, and 3) predict scalability, both in terms of problem size and in number of processors. We employ a hierarchical, staged, and open approach to the evaluation, examining low-level functionality of the system first, and then using these results to guide and understand the evaluation using kernels, compact applications, and full application codes. The distinction here is that the low-level benchmarks, for example, message passing, and the kernel benchmarks are chosen to model important features of a full application. This approach is also important because a number of the platforms contain novel architectural features that make it difficult to predict the most efficient coding styles and programming paradigms. Performance activities are staged to produce relevant results throughout the duration of the system installation. For example, subsystem performance will need to be measured as soon as a system arrives, and measured again following a significant upgrade or system expansion.

3.1 Test Systems

For comparison purposes, performance data is also presented for the following systems:

- Cray X1 at ORNL: 512 Multistreaming processors (MSP), each capable of 12.8 GFlops/sec for 64-bit operations. Each MSP is comprised of four single streaming processors (SSPs). The SSP uses two clock frequencies, 800 MHz for the vector units and 400 MHz for the scalar unit. Each SSP is capable of 3.2 GFlops/sec for 64-bit operations.
- Cray XD1 at ORNL: 144 AMD 2.2Ghz Opteron 248 processors, configured as 72, 2 way SMPs with 4GB of memory per processor. The processors are interconnected by Cray's proprietary RapidArray interconnect fabric.
- Earth Simulator: 640 8-way vector SMP nodes and a 640x640 single-stage crossbar interconnect. Each processor has 8 64-bit floating point vector units running at 500 Mhz.
- SGI Altix at ORNL: 256 Itaninium2 processors and a NUMAlink switch. The processors are 1.5 GHz Itanium2. The machine has an aggregate of 2 TB of shared memory.
- HP/Compaq AlphaServer SC at Pittsburgh Supercomputing Center (PSC): 750 ES45 4- way SMP nodes and a Quadrics QsNet interconnect. Each node has two interconnect interfaces. The processors are the 1GHz Alpha 21264 (EV68).
- IBM p690 cluster at ORNL: 27 32-way p690 SMP nodes and an HPS interconnect. Each node has two HPS adapters, each with two ports. The processors are the 1.3 GHz POWER4.
- IBM SP at the National Energy Research Supercomputer Center (NERSC): 184 Nighthawk(NH) II 16-way SMP nodes and an SP Switch2. Each node has two interconnect interfaces. The processors are the 375MHz POWER3-II.

4 Microbenchmarks

The objective of microbenchmarking is to characterize the performance of the specific architectural components of the platform. We use both standard benchmarks and customized benchmarks. The standard benchmarks allow consistent and widespread historical comparisons across platforms. The custom benchmarks permit the unique architectural features of the system (e.g., global address space memory) to be tested with respect to the target applications.

Traditionally, our microbenchmarking focuses on the arithmetic performance, memory-hierarchy performance, task and thread performance, message-

passing performance, system and I/O performance, and parallel I/O. However, because the XT3 has a single processor node and it uses a lightweight operating system, we focus only on these areas:

1. Arithmetic performance, including varying instruction mix, identifying what limits peak computational performance.
2. Memory-hierarchy performance, including levels of cache and shared memory.
3. Message-passing performance, including intra-node, inter-node, and inter-OS image MPI performance for one-way (ping-pong) messages, message exchanges, and collective operations (broadcast, all-to-all, reductions, barriers); message-passing hotspots and the effect of message passing on the memory subsystem are studied.

Current, detailed microbenchmark data for all existing evaluations is available at our Early Evaluation website [25].

Table 1: STREAM Triad Performance.

Processor	Triad Bandwidth (GBps)
Cray XT3	5.1
Cray XD1	4.1
Cray X1 MSP	23.8
IBM p690	2.1
IBM POWER5	4.0
SGI Altix	3.8

4.1 Memory Performance

The memory performance of current architectures is a primary factor for performance on scientific applications. Table 1 illustrates the differences in measured memory bandwidth on the triad STREAM benchmark. The very high bandwidth of the Cray X1 MSP clearly dominates the other processors, but the Cray XT3's Opteron performs the best with respect to the other microprocessor-based systems.

Table 2: Latency to Main Memory.

Platform	Measured Latency to Main Memory (ns)
Cray XT3 / Opteron 150 / 2.4 Ghz	51.41
Cray XD1 / Opteron 248 / 2.2 Ghz	86.51
IBM p690 / POWER4 / 1.3 Ghz	90.57
Intel Xeon / 3.0 Ghz	140.57

As discussed earlier, the choice of the Opteron model 150 was motivated by low latencies to main memory. As Table 2 shows, our measurements revealed that the Opteron 150 has lower latency than the Opteron 248 configured as a 2-way SMP in the XD1. Furthermore, it has considerably less latency than either the POWER4 or the Intel Xeon, which both support multiprocessor configurations.

The memory hierarchy of the XT3 compute node is obvious when measured with the CacheBench tool [24]. Figure 3 shows that the system hits a maximum of 32 GBps when accessing vectors of data in the L1 cache. In the L2 cache, the maximum bandwidth is approximately 9 GBps. Finally, when data is accessed from main memory, the bandwidth drops to about 3 GBps, with the exception of the 'C memset function' which maintains a bandwidth of about 6 GBps. *Interestingly, we found that this 'read' bandwidth was limited by the complier's inability to optimize the benchmark loop. Additional manual unrolling of the loop generated results consistent with the other tests.*

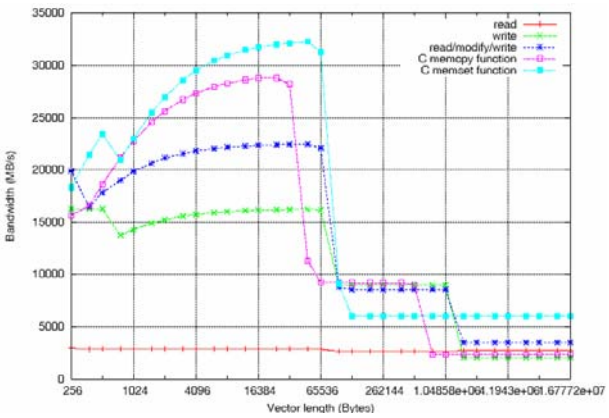

Figure 3: CacheBench results for XT3 compute node.

4.2 MPI

A very important part of system performance depends on the message passing performance. Latency and bandwidth provided through the Message Passing Interface (MPI) library [29] are particularly relevant because most contemporary applications are built on MPI.

Figure 4 and Figure 5 shows the latency and bandwidth for the MPI PingPong benchmark, respectively. We observe a latency of about 30 microseconds for a 4 byte message, and a bandwidth of about 1.1 GBps for messages over 1 MB.

Figure 6 and Figure 7 show the latency and bandwidth for the exchange benchmark, respectively, at 3,648 processors. This test separates all tasks into two groups, and then uses the MPI_SendRecv operation to transfer data between pairs of tasks, where the endpoints are in separate groups. As opposed to the PingPong operation, which transfers messages between only two tasks, the exchange benchmark has all pairs transferring messages at the same time. The average

latency of these transfers are higher, on the order of 90 microseconds for a 4 byte message. The bandwidth is also less than that for the PingPong test, but it reaches an average of nearly 1 GBps for an individual transfer, in the context of 1,824 simultaneous transfers.

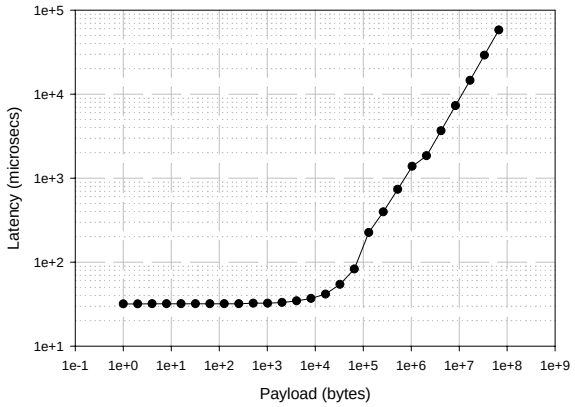

Figure 4: Latency of MPI PingPong.

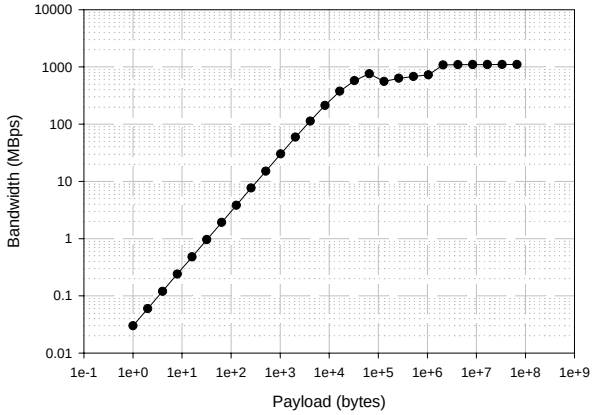

Figure 5: Bandwidth of MPI PingPong.

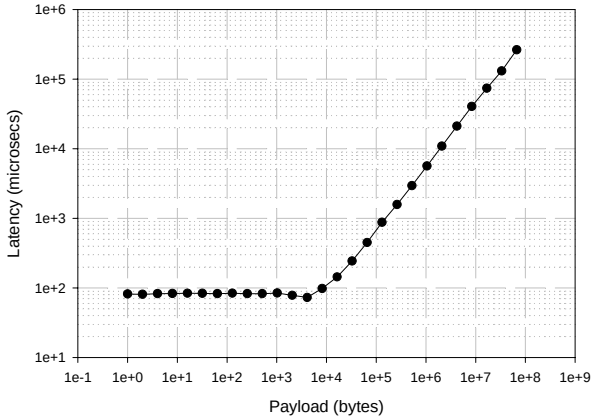

Figure 6: Latency of Pallas exchange operation.

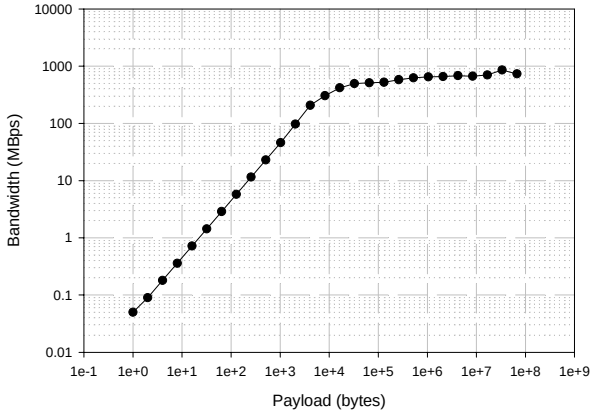

Figure 7: Bandwidth of Pallas exchange operation.

The latency for an Allreduce operation across 3,648 processors, as shown in Figure 8, is, on average, 600 microseconds for a 4 byte payload. The Allreduce operation is particularly important in large-scale scientific applications because it can be used multiple times on every timestep. Further, its blocking semantics also requires that all tasks wait for its completion before continuing, so latency for this operation is very important to good scaling.

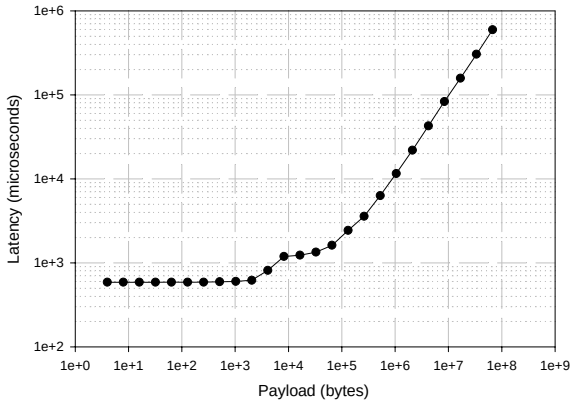

Figure 8: Latency for MPI_Allreduce of across 3,648 processors.

As mentioned earlier, we are using preliminary versions of the system software for these tests. We expect future versions of the software to improve both the latency and bandwidth of these MPI operations. In fact, two other sites are reporting latencies as low as 5 microseconds on MPI PingPong operations.

4.3 Scientific Operations

We use a collection of microbenchmarks to characterize the performance of the underlying hardware, compilers, and software libraries. The

microbenchmarks measure computational performance, memory hierarchy performance, and inter-processor communication. Figure 9 compares the double-precision floating point performance of a matrix multiply (DGEMM) on a single processor using the vendors' scientific libraries. The XT3 Opteron achieves 4 Gflops, about 83% of peak.

Figure 10 compares the vendor library implementation of an LU factorization (DGETRF) using partial pivoting with row interchanges. As expected, the X1 does very well for large matrix ranks; however, the XT3 and XD1 perform best for matrix ranks less than about 150.

Libraries undergo continuous optimization by their authors, so we constantly compare the performance of these libraries on common routines. A comparison of the ACML 2.5 and Goto libraries, as shown in Figure 11, on our current system shows small advantages to the Goto libraries.

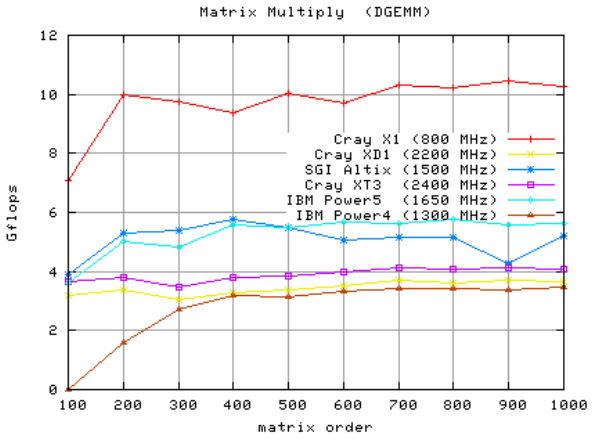

Figure 9: Performance of Matrix Multiply.

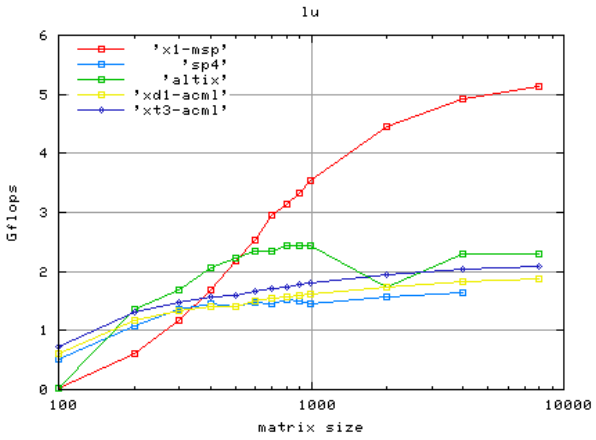

Figure 10: Performance of LU factorization.

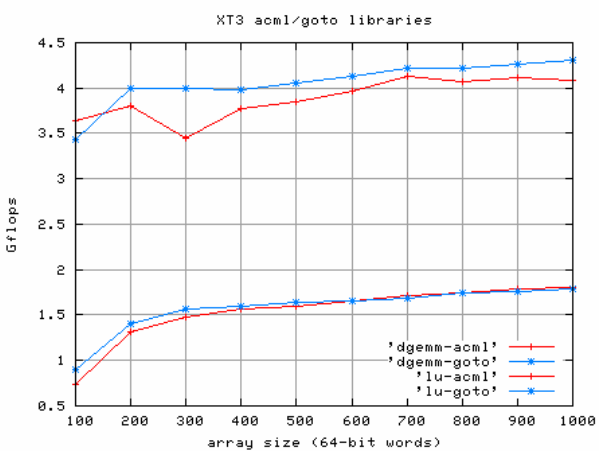

Figure 11: Comparison of ACML and Goto libraries on XT3.

In other testing, we compare vendor libraries with code generated by the optimizing FORTRAN compiler. Figure 12 shows the performance (Mflops) of Euroben *mod2b*, a dense linear system test, for both optimized FORTRAN and using the BLAS from the vendor library. In these tests, the advantages of the vendor libraries are clear when compared to the compiler optimized code.

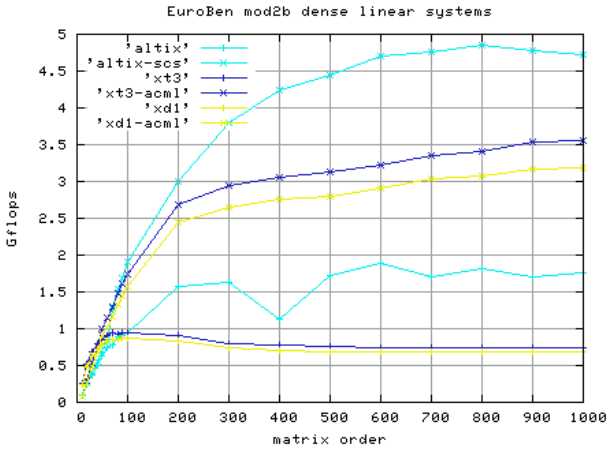

Figure 12: Performance of EuroBen mod2b.

Fast Fouier Transforms are another important kernl operation performed by many scientific and signal processing applications. Figure 13 compares a 1-D FFT using the FFTW benchmark [15]. Both the XD1 and XT3 perform well when compared to the SP3 and SP4, but the higher floating point rate of the Altix's Itanium allows it to generate higher performance. Alternatively, the vendors provide FFT libraries. Figure 14 plots 1-D FFT performance using the vendor library (-lacml, -lscs, -lsci or -lessl), where initialization time is not included. Again, the X1 does very well for long vectors, but the Opteron is competitive with other microprocessors.

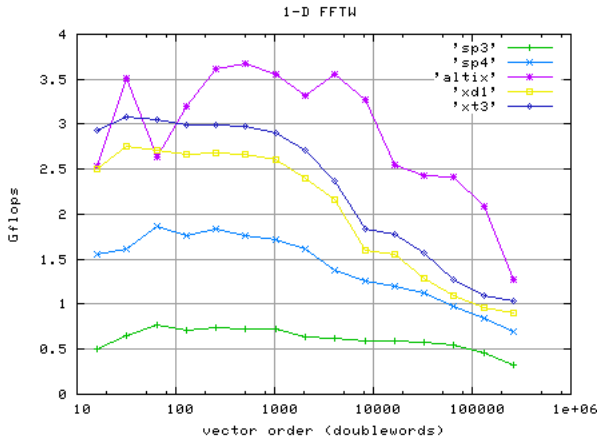

Figure 13: Performance of 1-D FFTW.

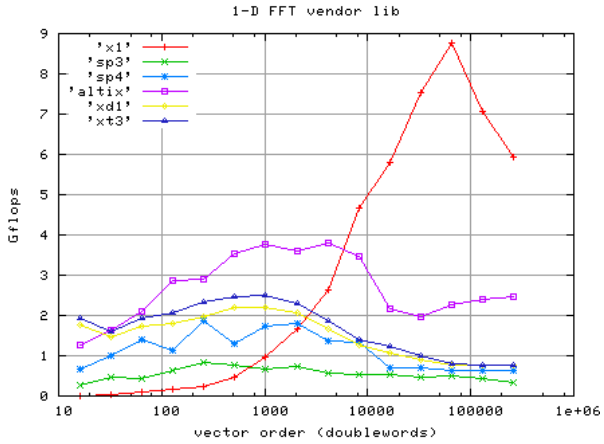

Figure 14: Performance of 1-D FFT using vendor libraries.

In general, our micro-benchmark results show the promise of the Cray XT3 compute nodes for scientific computing. Although the Cray X1's high memory bandwidth provided a clear benefit over the other systems we considered, and the SGI Altix and IBM Power5 systems gave better performance for several micro-benchmarks, the XT3 showed solid performance with these other systems, and in many cases, it performed better at very short vector lengths. Further, the benefits of using optimized libraries are very clear, given our performance comparisons.

4.4 HPC Challenge

The DARPA High Productivity Computing Systems program has recently sponsored the development of the HPC Challenge benchmark suite to emphasize the diverse application requirements of DARPA and its mission partners. Details and the latest version of the benchmark are available from the HPC Challenge website [20]. Initial versions (0.8b) of HPCC on our XT3 at 2,048 processors (9.8 TFLOPS) are producing an HPL result of 7.4 TFLOPS (75%), and a MPI Random Access result of 0.055 GUPS. Final numbers will be posted at the HPC Challenge website following installation of the final system.

5 Kernels

Next in our evaluation process, we focus on moderately-sized kernels that represent either common operations performed in scientific applications or operations extracted from target application codes. These kernels exercise multiple architectural features together, and provide a venue to examine the performance impact of a variety of coding styles and programming models. For example, on some systems, we have observed dramatic performance improvements by using UPC or Co-Array FORTRAN to implement communication operations in a critical application kernel.

The kernel-based evaluation is driven by the choice of the application codes. Whenever possible, standard kernels will be used, but profile data from the application codes will be the ultimate determinant. For example, it may be necessary to use code fragments extracted directly from the application code.

5.1 PSTSWM

The Parallel Spectral Transform Shallow Water Model (PSTSWM) [34] represents an important computational kernel in spectral global atmospheric models. As 99% of the floating-point operations are multiply or add, it runs well on systems optimized for these operations. PSTSWM exhibits little reuse of operands as it sweeps through the field arrays; thus it exercises the memory subsystem as the problem size is scaled and can be used to evaluate the impact of memory contention in SMP nodes. PSTWM is also a parallel algorithm testbed, and all array sizes and loop bounds are determined at runtime.

On the XT3, we used PSTSWM to analyze compiler optimizations, evaluate performance of the memory subsystem, and compare performance with other supercomputers. Figure 15 and Figure 16 show comparisons of optimization options. The comparisons are presented as computation rate versus horizontal resolution for two vertical resolutions. The problem sizes T5, T10, T21, T42, T85, and T170 are horizontal resolutions. Each computational grid in this sequence is approximately 4 times smaller than the next larger size. Although the two Figures are for 1 and 18 vertical levels, this aspect of the problem size does not change the compiler option comparison. PSTSWM manages its own heap, and all loop bounds and array sizes are

defined at runtime. From the figure, we see that this seems to limit some of the possible performance optimizations probably because the compiler is limited in what it can achieve.

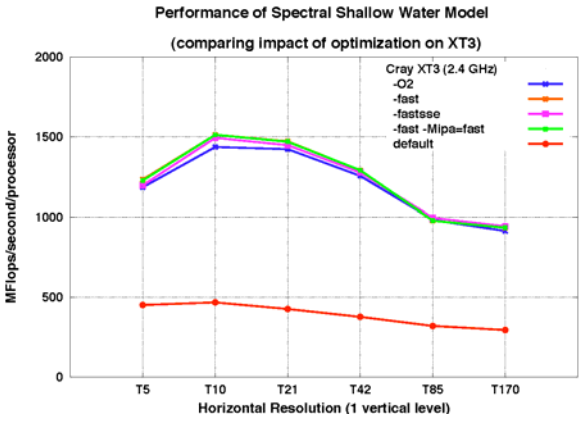

Figure 15: Impact of compiler optimizations on PSTSWM (1 level).

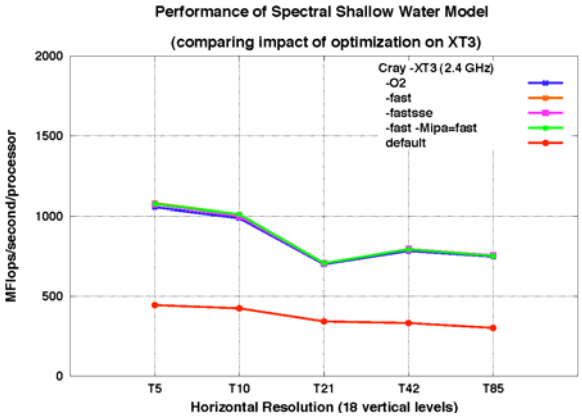

Figure 16: Impact of compiler optimizations on PSTSWM (18 levels).

Figure 17 compares performance across all problem sizes, showing impact of memory hierarchy.

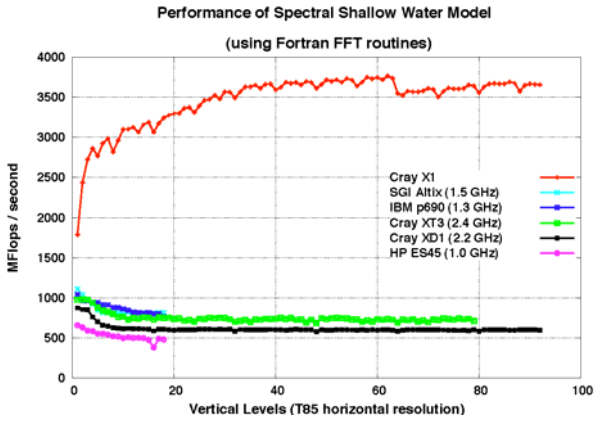

Figure 17: Performance of PSTSWM with varying numbers of vertical levels.

Most of the work is coupled most tightly horizontally, so additional vertical levels spreads data throughout memory, increasing access latency. Large problems drop from more than one Gop to 600 MFlops quickly as a function of number of vertical levels, but stay at the 600 MFlop rate from then on. Smaller problems drop more slowly, but to a lower asymptotic rate. The one and two processor per node comparison shows that this behavior is unaffected by both processors exercising memory simultaneously, so using both processors does not decrease memory performance or, in other words, increase memory contention.

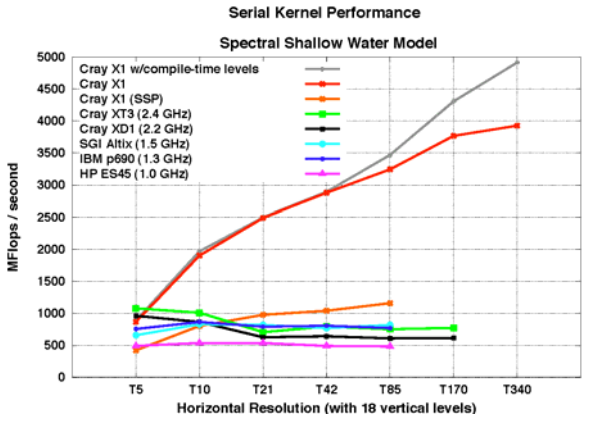

Figure 18: Performance of PSTSWM.

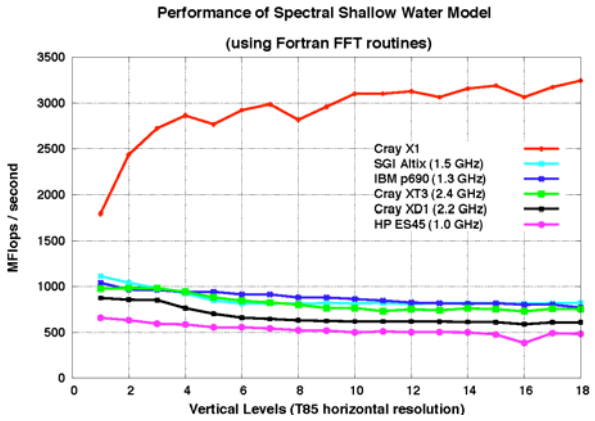

Figure 19: Single processor performance of PSTSWM for T85 while varying the number of vertical levels.

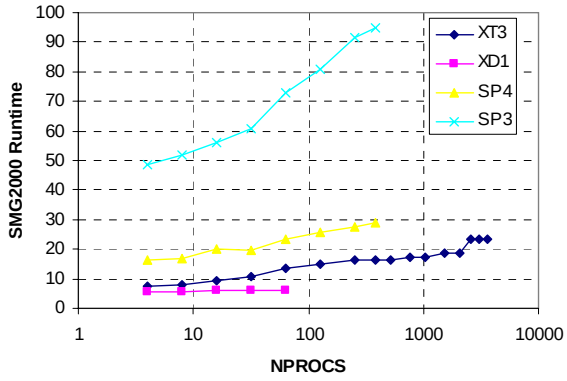

Figure 20: Performance of SMG2000.

Figure 18 shows a platform comparison. Note that for these tests, FFTW was used for FFTs were used instead of the vendor's optimized math library FFTs. The top chart in the figure compares single processor performance for various horizontal resolutions and a fixed 18 vertical levels. The X1's superior performance is due to the much higher processor/memory bandwidth. Figure 19 compares single processor performance with PSTSWM for T85 horizontal resolution and a range of numbers of vertical levels.

5.2 SMG2000

SMG2000 [9, 31, 32] is a parallel semicoarsening multigrid solver for the linear systems arising from finite difference, finite volume, or finite element discretizations of the diffusion equation on logically rectangular grids. The code solves both 2-D and 3-D problems with discretization stencils of up to 9-points in 2-D and up to 27-points in 3-D. Applications where such a solver is needed include radiation diffusion and flow in porous media. This benchmark includes both the setup of the linear system and the solve itself. Note that this setup phase can often be done just once, thus amortizing the cost of the setup phase over many timesteps. This trait is relatively common in implicit timestepping codes. For these experiments, we report only the solve time. These test scale the matrix size with the number of processors ('weak scaling').

As Figure 20 shows, the performance of SMG2000 shows the best performance on the Cray XD1, and, then, the XT3. Scalability on the XT3 is good out to 3,584 processors.

6 Applications

Because a system's behavior when running full applications is the most significant measure of its performance, we have investigated the performance and efficiency of applications relevant to the DOE Office of Science in the areas of global climate, fusion, chemistry, and bioinformatics. In addition to measures of performance and scalability common to evaluations of microprocessor-based MPP systems, the extent to which Cray compilers and tools can effectively utilize reconfigurable computing elements application codes will be investigated. The extent to which localized tuning can improve efficiency will also be investigated.

The evaluation team has worked closely with principal investigators who are leading the Scientific Discovery through Advanced Computing (SciDAC) application teams to identify important applications. As described above, initial steps in each domain was the detailed understanding of selected kernels and critical aspects of the system.

6.1 Parallel Ocean Program (POP)

The Parallel Ocean Program (POP) [18] is the ocean component of CCSM [4] and is being developed and maintained at Los Alamos National Laboratory (LANL). The code is based on a finite-difference formulation of the three-dimensional flow equations on a shifted polar grid. In its high-resolution configuration, 1/10-degree horizontal resolution, the code resolves eddies for effective heat transport and the locations of ocean currents.

We used a benchmark configuration (called x1) representing a relatively coarse resolution similar to that currently used in coupled climate models. The horizontal resolution is roughly one degree (320x384) and uses a displaced-pole grid with the pole of the grid

shifted into Greenland and enhanced resolution in the equatorial regions. The vertical coordinate uses 40 vertical levels with a smaller grid spacing near the surface to better resolve the surface mixed layer. Because this configuration does not resolve eddies, it requires the use of computationally intensive subgrid parameterizations. This configuration is set up to be identical to the actual production configuration of the Community Climate System Model with the exception that the coupling to full atmosphere, ice and land models has been replaced by analytic surface forcing.

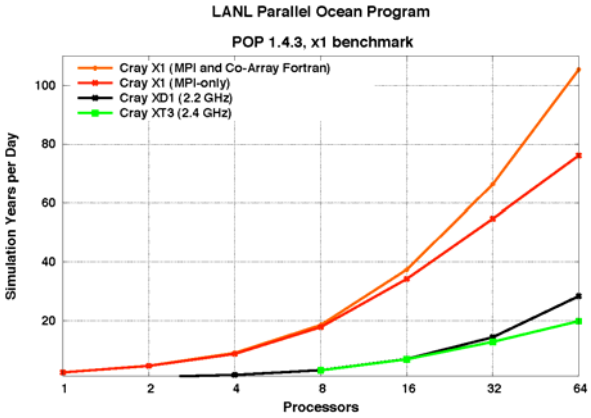

Figure 21: Performance of POP.

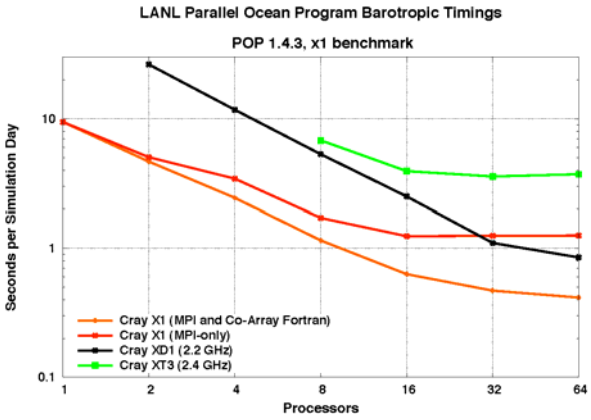

Figure 22: Performance of POP barotropic phase.

Figure 21 shows a platform comparison of POP throughput for the x1 benchmark problem. The XT3 performance is similar to that of Cray XD1. Figure 22 shows the performance of the barotropic portion of POP. This component is dominated by solution of 2D implicit systems using conjugate gradient solves and is known to scale poorly. Figure 23 shows the performance of the baroclinic portion of POP, which is known to scale well on many systems. The Cray XT3 did not scale as well as other systems we evaluated for the POP barotropic portion, perhaps due to known interconnect performance problems with the current system. On the other hand, the XT3 showed good scalability on the POP baroclinic portion.

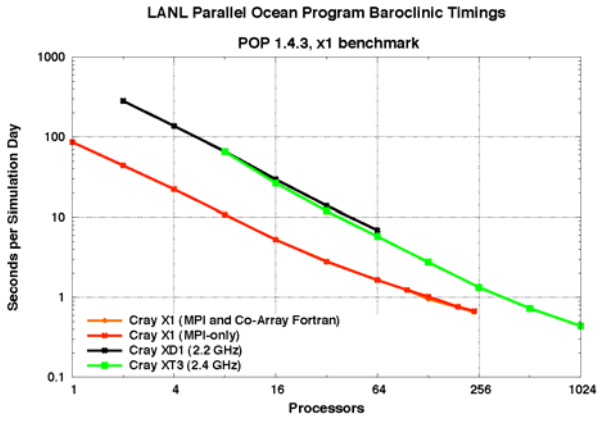

Figure 23: Performance of POP baroclinic phase.

6.2 GYRO

GYRO [10] is a code for the numerical simulation of tokamak microturbulence, solving time-dependent, nonlinear gyrokinetic-Maxwell equations with gyrokinetic ions and electrons capable of treating finite electromagnetic microturbulence. GYRO uses a five-dimensional grid and propagates the system forward in time using a fourth-order, explicit, Eulerian algorithm.

GYRO has been ported to a variety of modern HPC platforms including a number of commodity clusters. Since code portability and flexibility are considered crucial, only a single source is maintained. Ports to new architectures often involve nothing more than the creation of a new makefile.

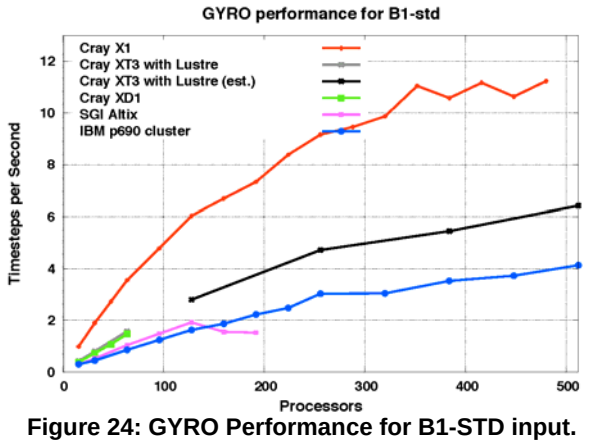

Figure 24: GYRO Performance for B1-STD input.

For our evaluation, we ran GYRO for two problems: B1-std and B3-gtc. The two problems differ in size and computational and communication requirements per node. The B1-std problem is smaller but requires more work per grid point than the B3-gtc

problem. GYRO tends to scale better for the B1-std problem (Figure 24) than the B3-gtc problem (Figure 25). The B3-gtc problem can use an FFT-based approach or a non-FFT approach.

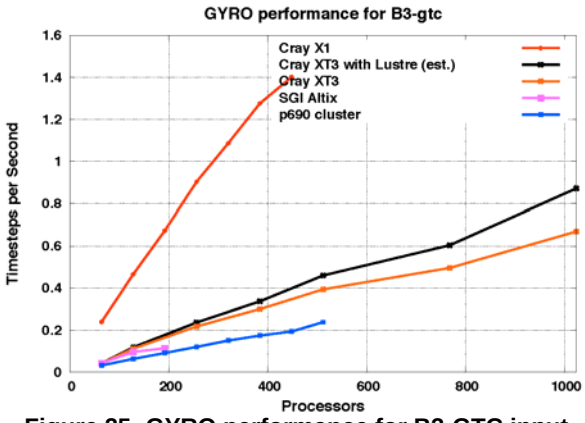

Figure 25: GYRO performance for B3-GTC input.

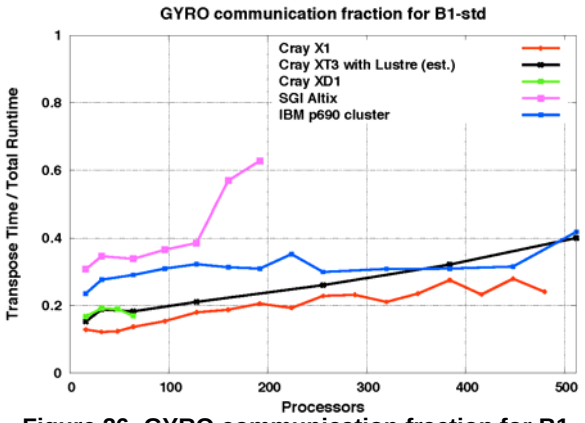

Figure 26: GYRO communication fraction for B1-STD input.

Figure 26 shows the communication fraction of the GYRO runtime for several different platforms. The excellent bandwidth of X1, XT3, and XD1 contribute to their low communication fraction on this strong scaling problem. The XT3 number is an estimate generated from actual GYRO experiments on the XT3 combined with timing estimates for IO activities on a smaller XT3 configured with Lustre. Figure 27 and Figure 28 show the differences in GYRO performance when using the different filesystems. As expected, both scaling and overall IO fraction for GYRO is much better for Lustre than for the scratch IO through Yod.

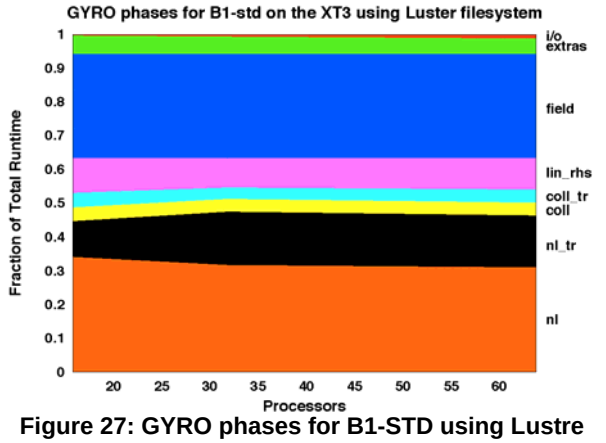

Figure 27: GYRO phases for B1-STD using Lustre filesystem.

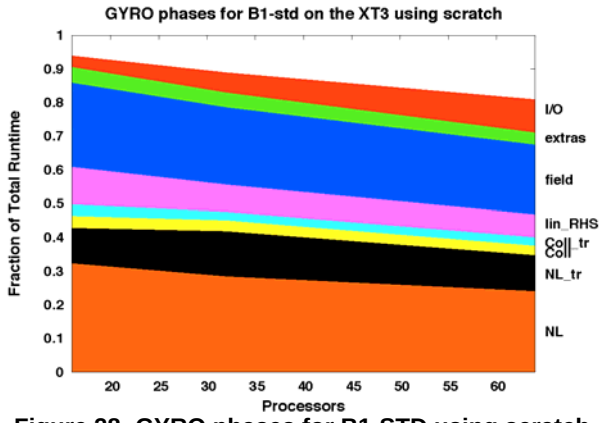

Figure 28: GYRO phases for B1-STD using scratch filesystem.

6.3 sPPM

sPPM [23, 31, 32] solves a 3-D gas dynamics problem on a uniform Cartesian mesh, using a simplified version of the Piecewise Parabolic Method. The algorithm makes use of a split scheme of X, Y, and Z Lagrangian and remap steps, which are computed as three separate sweeps through the mesh per timestep. Message passing provides updates to ghost cells from neighboring domains three times per timestep.

sPPM has been tested on numerous computer systems, and it is easy to scale the problem (weak scaling) to any number of processors. As we see in Figure 29, the scaling of sPPM scales very well across four platforms. On the XT3, scaling from 4 to 2,560 processors has 91.1% parallel efficiency. Because sPPM sends very large messages infrequently, MPI latency impacts performance less than bandwidth.

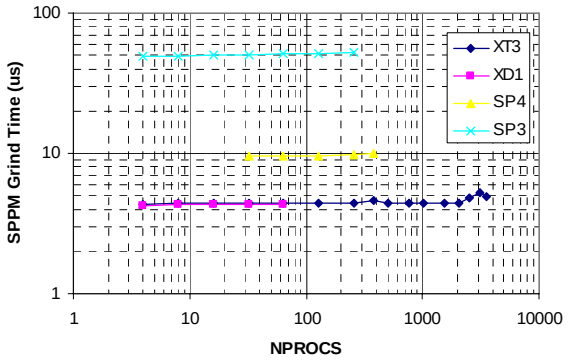

Figure 29: Performance of sPPM.

7 Future Performance Expectations

Because the Cray XT3 is a new product and the ORNL XT3 is only partially installed, we expect both hardware and software improvements as the system continues to be deployed. With our current version of software, we are measuring MPI unidirectional latencies of approximately 28 microseconds. We expect, and several Cray XT3 installations are reporting these same latencies on the order of 5 microseconds. With this goal in mind, we are using locally-developed predictive performance models to estimate the performance improvement on important DoE applications with this improved latency.

We predict the communication performance of two scientific codes, GYRO and POP, using LogGP models of communication. The LogGP parameters, L (latency), o (overhead), g (gap per message) are G (gap per byte) calculated using the logmpi software [19]. The MPI traces for the two applications are generated using parameterized simulation models of communication. LogGP models of communication define the latencies the communication latencies for point-to-point (*Ptp*), MPI_ALLREDUCE ($Allreduce_{binary}$), and MPI_ALLTOALL ($Alltoall_{linear}$) operations for a message size *m* as follows [27]:

$$Ptp = L + o_s + o_r + (m-1)G$$

$$Allreduce_{binary} = (\lceil \log_2(P+1) \rceil - 1) * (L + 3*o + (m-1)G + 2\gamma m) + (m-1)G + \max\{g, 3o + 2*\gamma m\}$$

$$Alltoall_{linear} = P*(L+2*o) + (P-1)*(P-1-\frac{P}{2})*(g+(m-1)G)$$

Figure 30 and Figure 31 show predicted runtimes for GYRO and POP, respectively, with 30µsec and 5µsec latencies. GYRO has a large number of MPI_ALLTOALL and MPI_ALLREDUCE operations. On the other hand, POP performs a large number of small nearest-neighbor operations: point-to-point and frequent MPI_ALLREDUCE call with 8 bytes messages. Hence, POP is comparatively more sensitive to communication latencies than GYRO because of small message sizes. We predict that a significant performance improvement will be achieved after the XT3 MPI latencies are reduced to 5µsec, particularly with large processor counts.

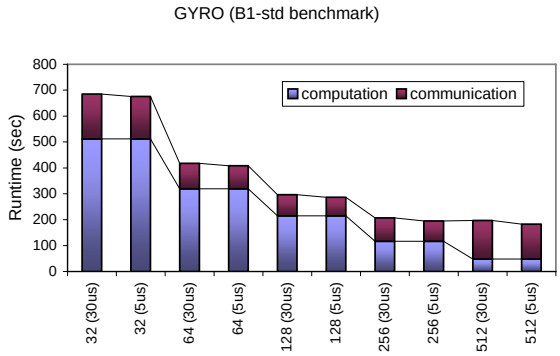

Figure 30: Performance prediction for GYRO.

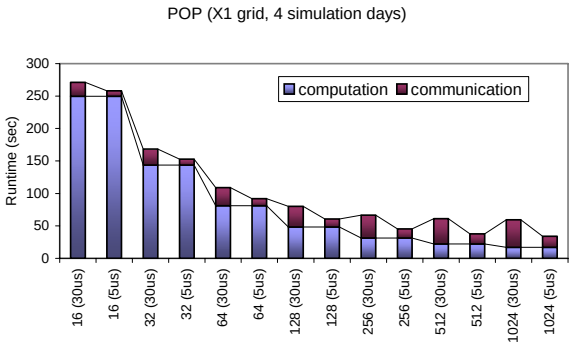

Figure 31: Performance prediction for POP.

8 Conclusions and Plans

Oak Ridge National Laboratory is in the process of receiving and installing a 5,200 processor Cray XT3. In this paper we describe our initial experiences with the system, including micro-benchmark, kernel, and application benchmark results. In particular, we provide performance results for important Department of Energy applications areas including climate and fusion. We demonstrate experiments on the partially installed system, scaling applications up to 3,600 processors. All of the components of the system are performing well with the exception of MPI latencies; we expect this

feature to improve dramatically with new software releases. Even given this issues, we have demonstrated that our system works well on several important applications at scale. We are eagerly anticipating the expansion to the final delivery size of 5,200 processors; we expect our application performance to improve as we receive additional firmware and software upgrades to the interconnect, compilers, and runtime libraries.

Acknowledgements

This research was sponsored by the Office of Mathematical, Information, and Computational Sciences, Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Batelle, LLC. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

Also, we would like to thank Jeff Beckleheimer, John Levesque, Nathan Wichmann, and Jim Schwarzmeier of Cray, and Don Maxwell of ORNL for all their assistance in this endeavor.

Authors

Sadaf R. Alam is a post-doctoral research associate in the Future Technologies group, Computer Science and Mathematics Division (CSMD) at the Oak Ridge National Laboratory. She has a PhD in Informatics from the University of Edinburgh, United Kingdom. Email: alamsr@ornl.gov.

Thomas H. Dunigan Jr. is a senior R&D staff member in the Computer Science and Mathematics Division of Oak Ridge National Laboratory. His research interests include the performance characterization and analysis of parallel computers and their communication subsystems; and computer and network security. Dunigan has a PhD in computer science from the University of North Carolina at Chapel Hill. E-mail: dunigan@ornl.gov.

Mark R. Fahey is a senior Scientific Application Analyst in the Center for Computational Sciences (CCS) at Oak Ridge National Laboratory. He is the current CUG X1-Users SIG chair. Mark has a PhD in mathematics from the University of Kentucky. E-Mail: faheymr@ornl.gov.

Philip C. Roth is an R&D staff member in the Computer Science and Mathematics Division of Oak Ridge National Laboratory. His research interests include tool automation and scalability, performance analysis and tuning, and systems software for high-end computing environments. Roth has a PhD in computer science from the University of Wisconsin-Madison. E-mail: rothpc@ornl.gov.

Jeffrey S. Vetter is a senior R&D staff member in the Computer Science and Mathematics Division of Oak Ridge National Laboratory, where he leads the Future Technologies Group. His research interests include experimental software systems and architectures for high-end computing. Vetter has a PhD in computer science from the Georgia Institute of Technology. He is a member of IEEE and the ACM. Also, Vetter is an Adjunct Professor in the College of Computing at Georgia Tech. E-mail: vetter@computer.org.

Patrick H. Worley is a senior R&D staff member in the Computer Science and Mathematics Division of Oak Ridge National Laboratory. His research interests include parallel algorithm design and implementation (especially as applied to atmospheric and ocean simulation models) and the performance evaluation of parallel applications and computer systems. Worley has a PhD in computer science from Stanford University. He is a member of the ACM and the Society for Industrial and Applied Mathematics. E-mail: worleyph@ornl.gov.

References

[1] P.A. Agarwal, R.A. Alexander *et al.*, "Cray X1 Evaluation Status Report," ORNL, Oak Ridge, TN, Technical Report ORNL/TM-2004/13, 2004, http://www.csm.ornl.gov/evaluation/PHOENIX/PDF/CRAYEvaluationTM2004-15.pdf.

[2] AMD, "Software Optimization Guide for AMD Athlon™ 64 and AMD Opteron™ Processors," Technical Manual 25112, 2004.

[3] D. Anderson, J. Trodden, and MindShare Inc., *HyperTransport system architecture*. Reading, MA: Addison-Wesley, 2003.

[4] M.B. Blackmon, B. Boville *et al.*, "The Community Climate System Model," *BAMS*, 82(11):2357-76, 2001.

[5] R. Brightwell, W. Camp *et al.*, "Architectural Specification for Massively Parallel Computers-An Experience and Measurement-Based Approach," *Concurrency and Computation: Practice and Experience*, 17(10):1271-316, 2005.

[6] R. Brightwell and L.A. Fisk, "Scalable Parallel Application Launch on Cplant," Proc. ACM/IEEE Conference on Supercomputing (SC 2001), 2001.

[7] R. Brightwell, L.A. Fisk *et al.*, "Massively Parallel Computing Using Commodity Components," *Parallel Computing*, 26(2-3):243-66, 2000.

[8] R. Brightwell, R. Riesen *et al.*, "Portals 3.0: Protocol Building Blocks for Low Overhead Communication," Proc. Workshop on Communication Architecture for Clusters (in conjunction with International Parallel & Distributed Processing Symposium), 2002, pp. 164-73.

[9] P.N. Brown, R.D. Falgout, and J.E. Jones, "Semicoarsening multigrid on distributed memory machines," *SIAM Journal on Scientific Computing*, 21(5):1823-34, 2000.

[10] J. Candy and R. Waltz, "An Eulerian gyrokinetic-Maxwell solver," *J. Comput. Phys.*, 186(545), 2003.

[11] Cray Incorporated, "Cray XT3 Programming Environment User's Guide," Reference Manual S-2396-10, 2005.

[12] T.H. Dunigan, Jr., J.S. Vetter *et al.*, "Performance Evaluation of the Cray X1 Distributed Shared Memory Architecture," *IEEE Micro*, 25(1):30-40, 2005.

[13] T.H. Dunigan, Jr., J.S. Vetter, and P.H. Worley, "Performance Evaluation of the SGI Altix 3700," Proc. International Conf. Parallel Processing (ICPP), 2005.

[14] M.R. Fahey, S.R. Alam *et al.*, "Early Evaluation of the Cray XD1," Proc. Cray User Group Meeting, 2005, pp. 12.

[15] M. Frigo and S.G. Johnson, *FFTW*, www.fftw.org, 2005.

[16] K. Goto, *High-Performance BLAS*, http://www.cs.utexas.edu/users/flame/goto/, 2005.

[17] High-End Computing Revitalization Task Force (HECRTF), "Federal Plan for High-End Computing," Executive Office of the President, Office of Science and Technology Policy, Washington, DC 2004.

[18] P.W. Jones, P.H. Worley *et al.*, "Practical performance portability in the Parallel Ocean Program (POP)," *Concurrency and Computation: Experience and Practice*(in press), 2004.

[19] T. Kielmann, H. Bal, and K. Verstoep, " Fast measurement of LogP parameters for message passing platforms," *Lecture Notes in Computer Science (IPDPS Workshops)*, 1800:1176-83, 2000.

[20] P. Luszczek and J. Dongarra, *HPC Challenge Benchmark*, http://icl.cs.utk.edu/hpcc/, 2005.

[21] A.B. Maccabe, K.S. McCurley *et al.*, "SUNMOS for the Intel Paragon: A Brief User's Guide," Proc. Intel Supercomputer Users' Group, 1994, pp. 245-51.

[22] T.G. Mattson, D. Scott, and S.R. Wheat, "A TeraFLOP Supercomputer in 1996: The ASCI TFLOP System," Proc. 10th International Parallel Processing Symposium (IPPS 96), 1996, pp. 84-93.

[23] A.A. Mirin, R.H. Cohen *et al.*, "Very High Resolution Simulation of Compressible Turbulence on the IBM-SP System," Proc. SC99: High Performance Networking and Computing Conf. (electronic publication), 1999.

[24] P.J. Mucci, K. London, and J. Thurman, "The CacheBench Report," University of Tennessee, Knoxville, TN 1998.

[25] Oak Ridge National Laboratory, *Early Evaluation Website*, http://www.csm.ornl.gov/evaluation, 2005.

[26] K. Pedretti, R. Brightwell, and J. Williams, "Cplant Runtime System Support for Multi-Processor and Heterogeneous Compute Notes," Proc. IEEE International Conference on Cluster Computing (CLUSTER 2002), 2002, pp. 207-14.

[27] J. Pjesivac-Grbovic, T. Angskun *et al.*, "Performance Analysis of MPI Collective Operations," Proc. 4th International Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems (PMEO-PDS 05), 2005.

[28] S.L. Scott, "Synchronization and Communication in the T3E Multiprocessor," Proc. Architectural Support for Programming Languages and Operating Systems (ASPLOS), 1996, pp. 26-36.

[29] M. Snir, S. Otto *et al.*, Eds., *MPI--the complete reference*, 2nd ed. Cambridge, MA: MIT Press, 1998.

[30] US Department of Energy Office of Science, "A Science-Based Case for Large-Scale Simulation," US Department of Energy Office of Science 2003, http://www.pnl.gov/scales.

[31] J.S. Vetter and F. Mueller, "Communication Characteristics of Large-Scale Scientific Applications for Contemporary Cluster Architectures," *Journal of Parallel and Distributed Computing*, 63(9):853-65, 2003.

[32] J.S. Vetter and A. Yoo, "An Empirical Performance Evaluation of Scalable Scientific Applications," Proc. SC 2002, 2002.

[33] S.R. Wheat, A.B. Maccabe *et al.*, "PUMA: An Operating System for Massively Parallel Systems," *Journal of Scientific Programming (special issue on operating system support for massively parallel systems)*, 3(4):275-88, 1994.

[34] D.L. Williamson, J.B. Drake *et al.*, "A Standard Test Set for Numerical Approximations to the Shallow Water Equations in Spherical Geometry," *Journal of Computational Physics*, 192:211-24, 1992.