Evaluation of the SGI Altix 3700 at Oak Ridge National Laboratory

Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN, USA 37831

Abstract. SGI recently introduced the Altix 3700. In contrast to previous SGI systems, the Altix uses a modified version of the open source Linux operating system and the latest Intel IA-64 processors, the Intel Itanium2. The Altix also uses the next generation SGI interconnect, Numalink3 and NUMAflex, which provides a NUMA, cache-coherent, shared memory, multi-processor system. In this paper, we present a performance evaluation of the SGI Altix using microbenchmarks, kernels, and mission applications. We find that the Altix provides many advantages over other non-vector machines and it is competitive with the Cray X1 on a number of kernels and applications. The Altix also shows good scaling, and its globally shared memory allows users convenient parallelization with OpenMP or pthreads.

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 [13] and the High End Computing Revitalization Task Force report [5], examples are numerous, ranging from global climate change research to combustion to biology.

It is incumbent on DOE to be aware of the performance of new or beta systems from HPC vendors that will determine the performance of future production-class offerings. It is equally important that DOE work with vendors in finding solutions that fulfill DOE's computational requirements.

Performance of the current class of HPC architectures is crucially 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 the increasing 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 a 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.

IBM and HP have dominated recent HPC procurements in the US. IBM HPC systems are clusters of 4 to 32 processor SMP nodes connected with a switch. Applications use MPI to communicate between the SMP nodes. Large IBM systems have been installed at a number of DOE sites, including LLNL, LBNL, and ORNL. While performance has been good, the next generation HPS switch technology was only recently delivered, significantly behind schedule. Performance results indicate that the HPS switch performs significantly better than the previous generation IBM switch, but does not perform as well as networks from some other HPC vendors. Additionally, it is possible to saturate the network with a modest number of SMP nodes. This will limit efficient scalability, especially as processors continue to grow faster. HP systems, installed at LANL and PNNL, are built around the Quadrics QsNet switch. The HP systems utilize small SMP nodes (2-4 processors), and application performance is highly dependent on the performance of the switch. The Quadrics switch continues to evolve, but the large HP systems are not well suited for applications with large memory requirements or random memory access patterns.

Both Cray Research and SGI have recently introduced systems that compare favorably with those from IBM and HP, with a number of unique characteristics that are attractive for certain classes of applications. The Cray system, a parallel vector system called the X1, is currently being evaluated at Oak Ridge National Laboratory (ORNL) and at Department of Defense (DOD) and other government sites [1]. A separate effort, documented here, is evaluating the SGI system, a (non-vector) parallel system called the Altix.

The SGI Altix system is a large shared memory system. During the evaluation, SGI demonstrated single systems with 512 processors and, just recently deployed a federated system totaling 10,000 processors. In contrast to previous SGI systems, the Altix uses a modified version of the open

source Linux operating system and the latest Intel IA-64 processor, the Itanium2. The Altix also uses the next generation SGI interconnect, the NUMAlink3, which is the natural evolution of the highly successful interconnect used in previous generation SGI systems. NUMAlink3 is also the interconnect used in the Cray X1. For SGI, the Altix is a combination of traditional strengths (large SMP nodes and fast interconnects) and risk (new processors and new operating system).

Performance of earlier revisions of the Intel Itanium2 were very interesting. The 1 GHz Itanium2 in the HP system at PNNL achieved 3.73 GFlop/sec, representing 93% of processor peak, on the Linpack 1000 benchmark and was 10% faster than the 1.45 GHz IBM Power4 processor on the SPECfp2000 benchmark. This Itanium2 processor also performed better than a large number of current AMD, HP, Compaq, SUN, MIPS, and IBM processors on a benchmark suite of computational chemistry kernels developed at Daresbury Laboratory in the UK, In particular, the Itanium2 was on average 20% faster than the 1.3 GHz Power4 for these benchmarks. The Itanium2 in the Altix differs from the earlier revisions in that it has as an integrated on-chip level 3 cache of size up to 6 MB and a clock rate 1.5 times faster.

This report describes the initial evaluation results collected on an SGI Altix system sited at ORNL. Researchers both within and outside the DOE Laboratory system were given access to this evaluation system and have contributed to this report. Results are also publicly available from the ORNL evaluation web site: http://www.csm.ornl.gov/evaluation. Other researchers at Daresbury [2] and Lawrence Berkeley Laboratory [9] also used the ORNL Altix in their performance studies. The CCS Altix was also featured in several articles including *Linux Magazine* [6].

2 Evaluation Methodology

The primary tasks of the evaluation were to:

- determine the most effective approaches for using the SGI Altix
- evaluate benchmark and application performance, and compare with similar systems from other vendors
- predict scalability, both in terms of problem size and in number of processors

We employed a hierarchical approach to the evaluation, examining low-level functionality of the system first, then using these results to guide and understand the evaluation using kernels and compact or full application codes.

Standard benchmarks were used as appropriate to ensure meaningful comparisons with other system evaluations; however, the emphasis of our evaluation is to study a number of different important DOE applications, as noted below.

Figure 1: SGI Altix 3700 at CCS.

The distinction here is that the low-level benchmarks, for example, message passing, and the kernel benchmarks were chosen to model important features of a full application. This is important in order to understand application performance and to predict scalability.

The end goal is to predict the expected sustained performance of multi-Terascale SGI Altix systems on large-scale simulations, using actual applications codes from environmental and life sciences. Specific sub goals are described in more detail below.

3 SGI Altix 3700 Overview

The initial offering of SGI Altix is a shared memory system made up of 2 processor nodes interconnected with the first implementation of the SN2 "scalable node architecture". This architecture supports 64 TB of addressable memory. At introduction, the SGI modification of Linux supported a single system image on up to 64 processors, but this increased to 256 processors during the course of the evaluation. Multiple Linux kernels reside in a Coherent Sharing Domain (CSD), which provides cache coherence for up to 512 processors. Multiple CSDs can reside within a single system, with the NUMAlink layer of SN2 providing high bandwidth and low latency communication within and between CSDs. Optimized communication libraries provide coherent access both within and between partitions defined by the OS images.

Unlike commodity Linux clusters, SGI's cache-coherent, shared memory, multi-processor system is based on NUMAflex, SGI's third-generation, non-uniform memory access (NUMA) architecture, which has proven to be a highly-scalable, global shared memory architecture based on SGI's Origin 3000 systems. In fact, the Altix 3000 uses many of the same components — called *bricks* — that the Origin uses. These bricks mount in racks and may be used in various combinations to construct a system balanced for a specific workload. SGI offers several different types of bricks. Table 1 lists these bricks.

Table 1: SGI Altix brick types.

Brick Type	Purpose
C-Brick	computational module housing CPUs and memory
M-Brick	Memory expansion module
R-Brick	NUMAflex router interconnect module
D-Brick	Disk expansion module
IX-Brick	Base system I/O module
PX-Brick	PCI-X expansion module

The Altix C-brick (Figure 2) consists of two nodes, each containing two Itanium 2 processors with their own cache. These Altix C-bricks are different from those in the Origin because the Origin C-bricks contain MIPS processors, while the Altix C-bricks contain Itaniums. The front-side buses of these processors are connected to custom ASICs referred to as *SHUBs*. The SHUBs interface the two processors to the memory DIMMs, to the I/O subsystem, and to other SHUBs via the NUMAflex network components. The SHUBs also interconnect the two nodes in a C-brick at the full bandwidth of the Itanium 2 front side bus (6.4 GB/sec).

The global shared memory architecture, implemented through SGI's *NUMAlink* interconnect fabric, provides high cross-sectional bandwidth and allows performance scaling not usually obtained on commodity Beowulf clusters. While some coarse grained applications scale just fine on Beowulf clusters, others need the high bandwidth and very low latency offered by a machine like the Altix. Still other applications are best implemented as shared-memory applications using many processors.

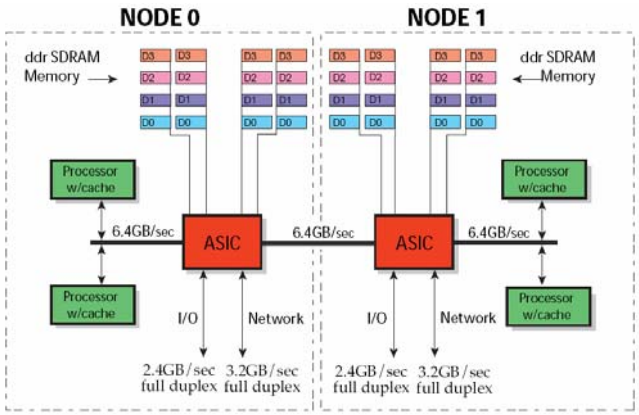

Figure 2: SGI Altix C-brick. ***(Image courtesy of SGI.)***

The Altix shipped with the Intel Madison Itanium processor, running at 1.3 GHz. The clock rate was later increased to 1.5 GHz. The Itanium has 3 levels of cache: 32 KB L1 (1 clock latency), 256 KB L2 (5 clock latency), and 6 MB L3 (14 clock latency). The L3 cache can sustain 48 GB/sec bandwidth to/from the processor. The memory system utilizes commodity DDR SDRAM DIMMs, achieving 10+ GB/sec bandwidth per node. The interconnect topology is a dual plane, quad bristled fat tree, capable of 800 MB/sec per processor in a bisection bandwidth test for up to 32 processors, and 400 MB/sec per processor for more than 32 processors. Additional configuration and operational information is available at http://www.ccs.ornl.gov/Ram/Ram.html.

Table 2: System configurations.

	SGI Altix	Alpha SC	IBM SP3	IBM SP4	HP XC	Cray X1
Name	Ram	LeMieux	Eagle	Cheetah	HPCS2	Phoenix
Proc	Itanium 2	Alpha EV67	POWER3-II	POWER4	Itanium 2	Cray X1
Interconnect	Numalink	Quadrics	Colony	Colony	Quadrics II	Cray X1
MHz	1500	667	375	1300	1500	800
Mem/Node	512GB	2GB	2GB	32GB	6GB	16GB
L1	32K	64K	64K	32K	32K	16K (scalar)
L2	256K	8MB	8MB	1.5MB	256K	2MB (per MSP)
L3	6MB	n/a	n/a	128MB	6MB	n/a
Proc Peak Mflops	6000	1334	1500	5200	6000	12800
Peak mem BW	6.4 GB/s	5.2GB/s	1.6GB/s	51 GB/s/MCM	6.4GB/s	26 GB/s/MSP

ORNL purchased a 256 processor Altix system with a total of 2 terabytes of shared memory, 12 TB of fiber channel attached disks, and a single system image (SSI) software needed to support 256 processors. The processors are 1.5 GHz Intel Madison Itanium2 processors with 6 MB of cache per processor. Initially, the system was configured to run 4 partitions of the Linux operating system. However, after working closely with SGI, the Altix is currently running a 256 processor single system image. Single applications run on all 256 processors using the NUMAlink interconnect for interprocessor communications between the multiple segments of the system.

Table 3: Experiment configurations.

Mnemonic	System	Programming model
X1-mpi	Cray X1 (Phoenix)	MPI
X1-ca	Cray X1 (Phoenix)	CoArray Fortran
Altix-mpi	SGI Altix (RAM)	MPI
Altix-omp	SGI Altix (RAM)	OpenMP
p690-mpi	IBM p690 (Cheetah)	MPI
p690-mpi	IBM p690 (Cheetah)	MPI

This evaluation also includes comparisons to other systems examined by CCS. Table 2 and Table 3 outline these system configurations. Interested readers can find more information about these platforms at the CCS website: http://www.ccs.ornl.gov/user/computers.html.

4 Microbenchmarks

The objective of microbenchmarking is to characterize the performance of the underlying architectural components of the SGI Altix. Both standard benchmarks and customized benchmarks are used. The standard benchmarks allow

component performance to be compared with other computer architectures. The custom benchmarks permit the unique architectural features of the Altix (e.g., large shared memory system utilizing Intel processors) to be tested with respect to the target applications. The architectural-component evaluation assesses the following:

- Arithmetic performance, including varying instruction mix, identifying what limits peak computational performance.
- Memory-hierarchy performance, including three levels of cache and shared memory. These tests utilize both System V shared memory and the SHMEM primitives. Of particular interest is the performance of the shared memory, and how remote accesses interact with local accesses.
- Task and thread performance, including performance of thread creation, locks, semaphores, and barriers. Of particular interest is how explicit thread management compares with the implicit control provided by OpenMP, and how thread scheduling and memory/cache management (affinity) perform.
- Message-passing performance, including intra-node, inter-node, intra-OS image, and inter-OS image MPI performance for one-way (ping-pong) messages, message exchanges, and aggregate operations (broadcast, all-to-all, reductions, barriers); message-passing hotspots and the effect of message passing on the memory subsystem are of particular interest.
- System and I/O performance, including a set of tests to measure OS overhead (context switches), virtual memory management, low-level I/O (serial and parallel), and network (TCP/IP) performance.
- Because of their importance to many application communities, we also assess the performance of parallel I/O, as well as the performance of standard MPI I/O benchmarks.

Detailed microbenchmark data are available from http://www.csm.ornl.gov/~dunigan/sgi/. For example, we used the EuroBen benchmark to evaluate hardware performance of add, multiply, divide, and square root, and the performance of the software intrinsics (exponentials, trigonometric functions, and logarithms). Other tests demonstrate how vector length and stride, compiler optimizations, and vendor scientific libraries affect performance. Figure 3 is a FORTRAN 1-D FFT from Euroben benchmarks. For this benchmark the Altix processor outperforms that of the IBM p690 and the Cray X1 for small to medium sized vectors.

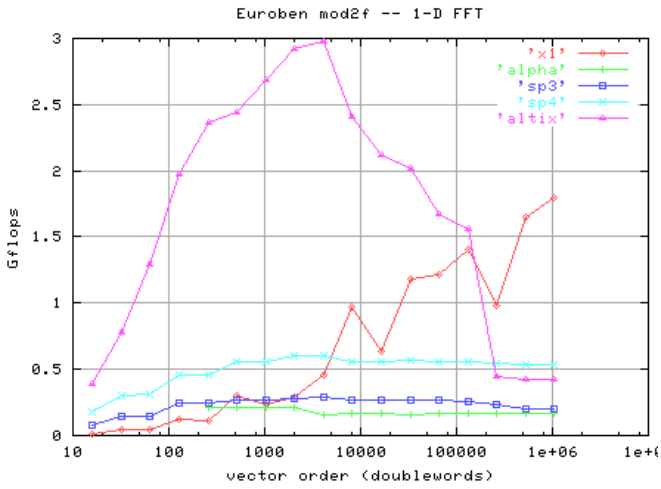

Figure 3: Euroben mod2f benchmark for 1-D FFT.

Our results also show that the Altix performs well on both sparse eigenvalue kernels and dense linear algebra kernels, achieving over 90% of peak for a matrix-matrix multiply (DGEMM). Figure 4 compares the performance of vendor math libraries for solving a dense linear system, demonstrating that the Altix is better than the IBM p690 and the Cray X1 for the middle range of matrix sizes.

The STREAMS and MAPS benchmarks show the high memory bandwidth the Altix achieves, and Figure 5 shows how the memory performance scales with the number of processors. (The NASA Ames 512-processor Altix is the fastest multiprocessor by this metric, providing over 1 TB/s of memory bandwidth for the STREAMS triad. (See http://www.cs.virginia.edu/stream/top20/Bandwidth.html.)

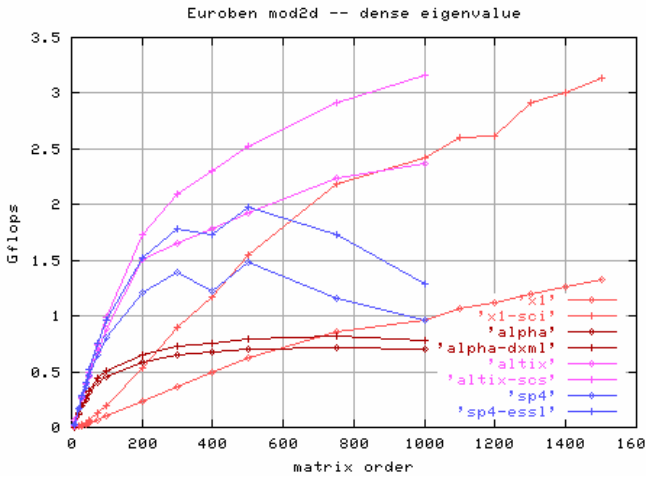

Figure 4: EuroBen mod2b benchmark for dense linear systems.

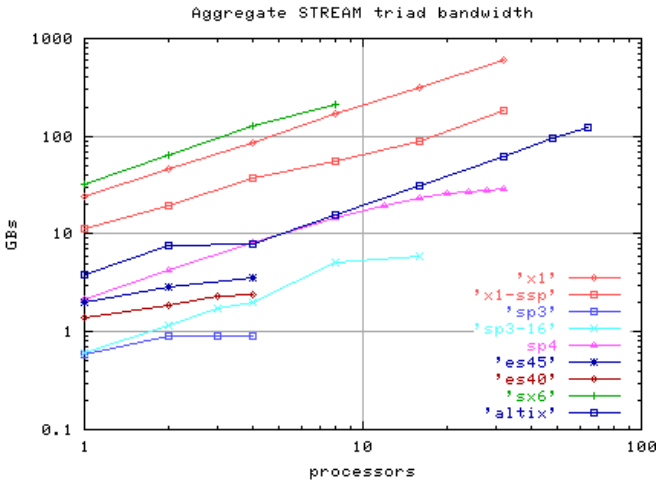

Figure 5: Aggregate STREAM triad bandwidth.

Our communication tests include the standard benchmarks (ParkBench and Euroben-dm) to measure latency and bandwidth as a function of message size and distance, as well as custom benchmarks that reflect common communication patterns. The Altix MPI latency is only 2.5 microseconds (us) compared to 7 us on the Cray X1 and IBM p690 (Federation).

Figure 6 compares the bandwidth when 2 processors a distance of 16 apart are exchanging messages using MPI and when 32 processors (16 pairs) are exchanging messages across the same distance. Figure 7 compares the bandwidth when 2 processors a distance of 64 apart are exchanging messages using MPI and when 128 processors (64 pairs) are exchanging messages. Note that on the IBM p690 cluster the first experiment is limited to processors in the same p690 shared memory node, while the second experiment requires communication across the HPS switch. Within the SMP node, the achieved IBM bandwidth is at least as good as that on the Altix.

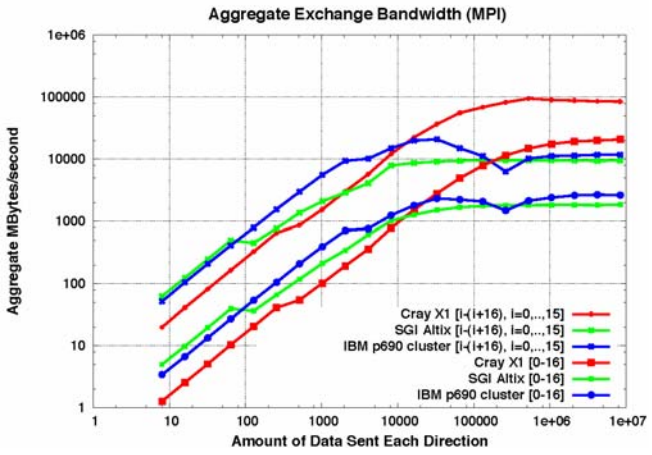

Figure 6: Aggregate Exchange Bandwidth (MPI) for distance of 16 processors.

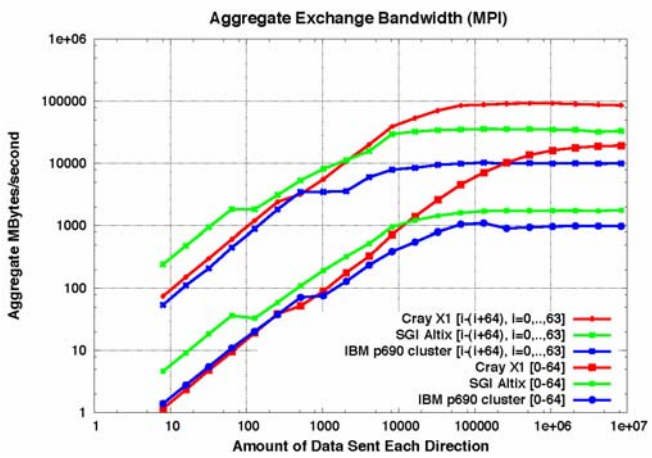

Figure 7: Aggregate Exchange Bandwidth (MPI) for distance of 64 processors.

In contrast, the Altix achieves much better aggregate bandwidth than the IBM when the IBM must communicate between p690s. While the Cray X1 achieves the best aggregate bandwidth for large messages, it reaches the same maximum for the two experiments, approximately 90 GBytes/sec. In contrast, the aggregate SGI bandwidth is still rising, achieving approximately 50% of the maximum X1 bandwidth in the second experiment. For small messages, the Altix achieves the best performance among the three systems.

Figure 8 and Figure 9 examine MPI communication performance as a function of physical distance between communicating processes. Except where noted, the cache is not invalidated before taking measurements. For small messages there is a performance advantage to communicating between physically neighboring processors, especially if in the same node. However, there is little performance sensitivity for larger distances.

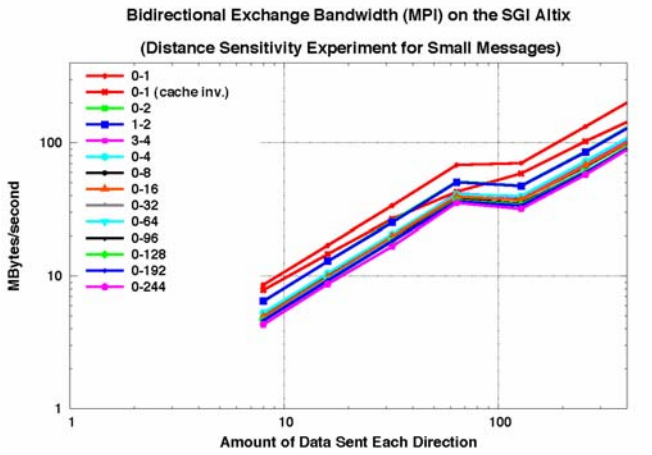

Figure 8: Distance sensitivity for small messages.

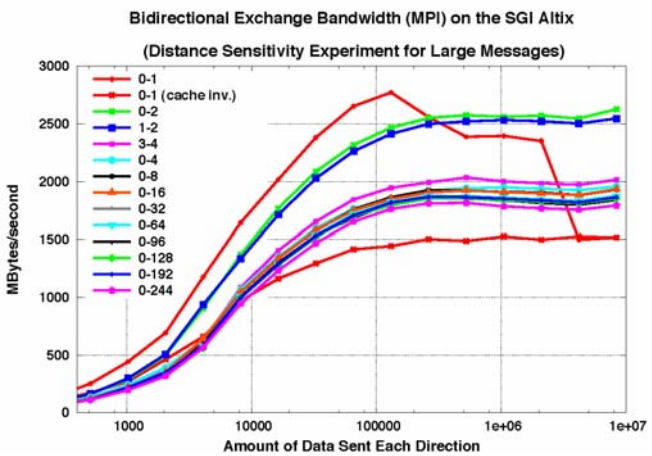

Figure 9: Distance Sensitivity for large messages.

In contrast, exchanging large messages between processors in the same 2-processor node is more expensive than when exchanging between processors not in the same node. This effect shows up sooner with cache invalidation than without. Exchanging large messages between processors in the same C-brick, but not in the same node, shows the highest performance. As with small messages, large message performance is relatively insensitive (<20%) to distance once the separation is greater than 4. Experiments with cache invalidation show similar behavior, except as noted above.

Figure 10 and Figure 11 examine the same issue for simultaneous exchange. Unlike Figure 6, the metric here is bandwidth per process pair, not aggregate bandwidth. For small messages there is again little performance sensitivity to distances greater than 4, and performance degradation compared to the distance experiments is <20%. For large messages contention does occur, with the performance observed by a single pair halved for 4 simultaneous exchanges, and reduced to 25% of the former bandwidth for 32 simultaneous exchanges. Note, however, that the aggregate bandwidth continues to increase, especially as the number of pairs (and the distance) increases.

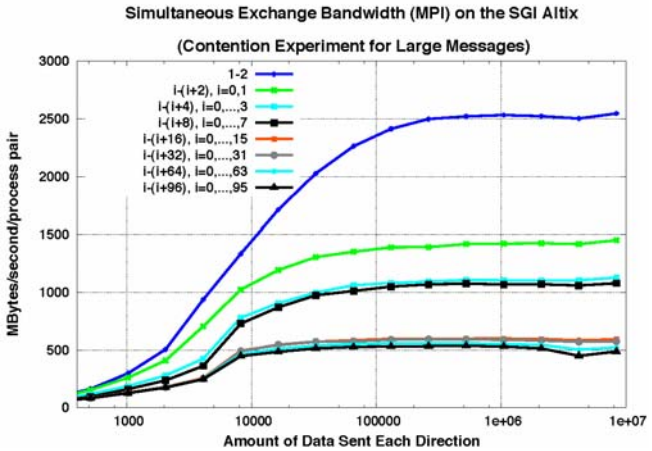

Figure 10: Contention for large messages.

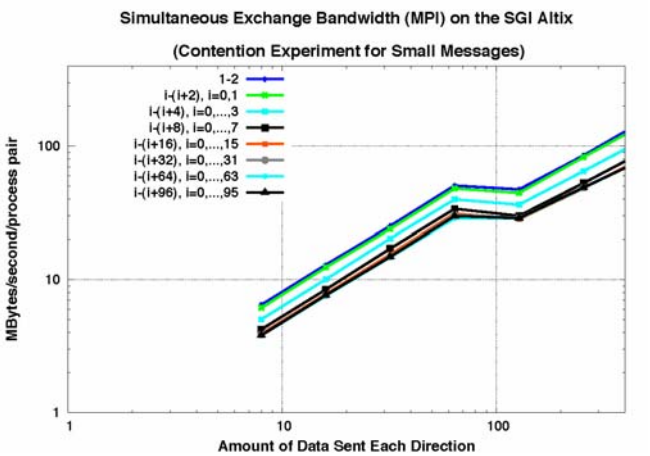

Figure 11: Contention for small messages.

Similar results hold when using SHMEM to implement the exchange instead of MPI. For small messages SHMEM performance is twice that of MPI, but sensitivity to distance and contention are qualitatively the same. For large messages SHMEM and MPI performance are nearly identical when the cache is invalidated first. However, without cache invalidation SHMEM performance is significantly better than MPI performance for all but the largest message sizes.

The exchange experiments were also used to determine the most efficient MPI communication protocol to use for an exchange. When enabling the single copy protocols, which are automatically used with MPI_SENDRECV, protocols using MPI_ISEND and MPI_RECV are the most efficient, but MPI_SENDRECV is typically one of the better performers.

When applications are scaled to larger processor counts or larger problems sizes, the ALLTOALLV and ALLREDUCE communication collectives are often the source of performance problems. The performance observed in the exchange experiments provides some information on the performance of the MPI_ALLTOALL collective.

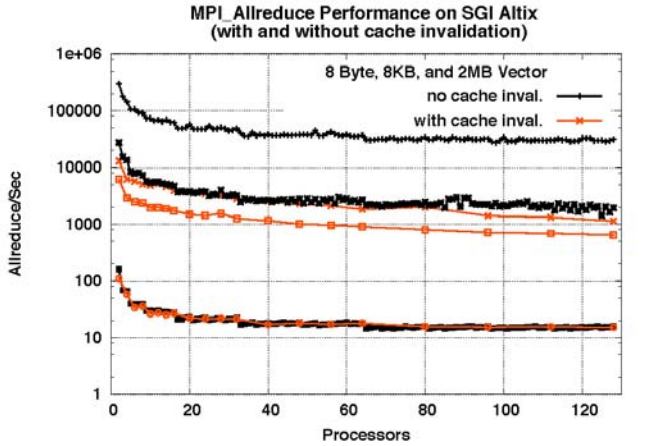

Figure 12: Performance of MPI_ALLREDUCE (SUM) for 8, 8K, and 2M bytes.

Figure 12 describes the performance of MPI_ALLREDUCE using the SUM operator on vectors of length 8 bytes, 8 Kbytes, and 2 Mbytes, both with and without cache invalidation. The metric is "allreduce/sec", so larger values imply better performance. Unlike the MPI point-to-point command experiments, cache invalidation has a large impact on the measured performance of MPI_ALLREDUCE, probably because the SGI implementation of MPI_ALLREDUCE takes advantage of shared memory primitives. Which assumption about the state of the cache is more realistic is application-specific.

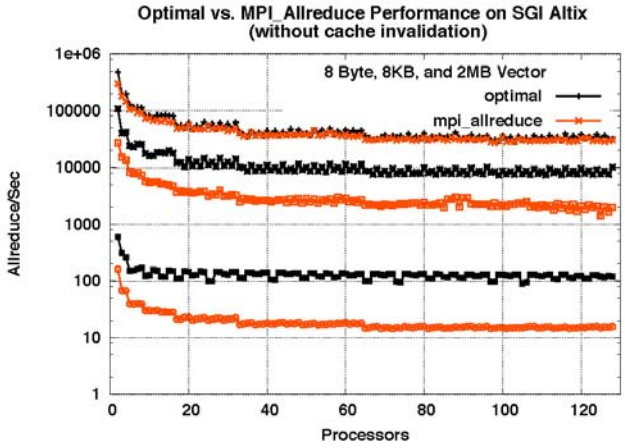

Figure 13: Performance of MPI_ALLREDUCE using different implementations.

Figure 13 compares the performance of MPI_ALLREDUCE with the best of a number of point-to-point implementations using both SHMEM and MPI-1 commands (including MPI_ALLREDUCE) without cache invalidation. The MPI collective is best for small vectors, but is suboptimal for large vectors.

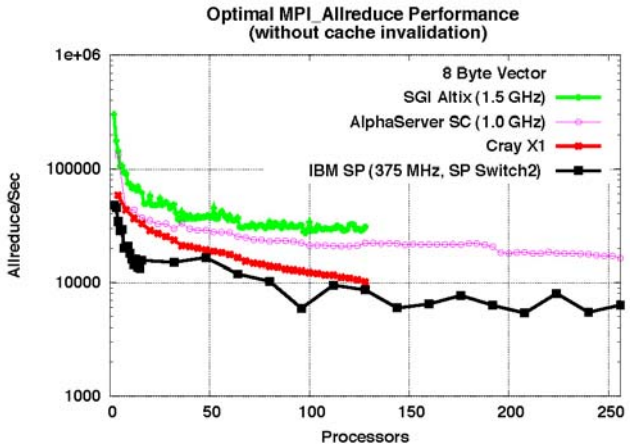

Figure 14: Performance of MPI_ALLREDUCE (8B) across platforms.

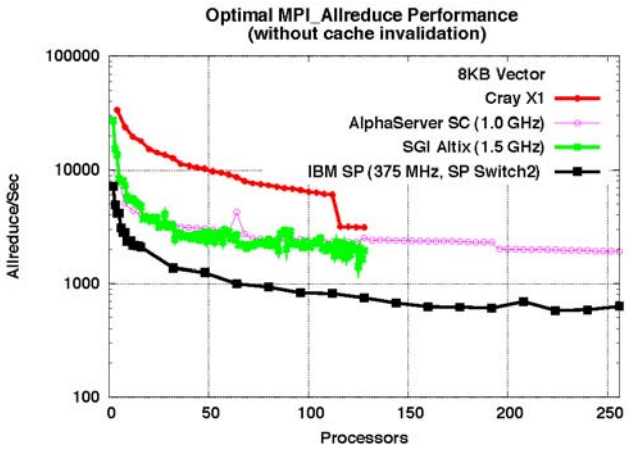

Figure 15: Performance of MPI_ALLREDUCE (8KB) across platforms.

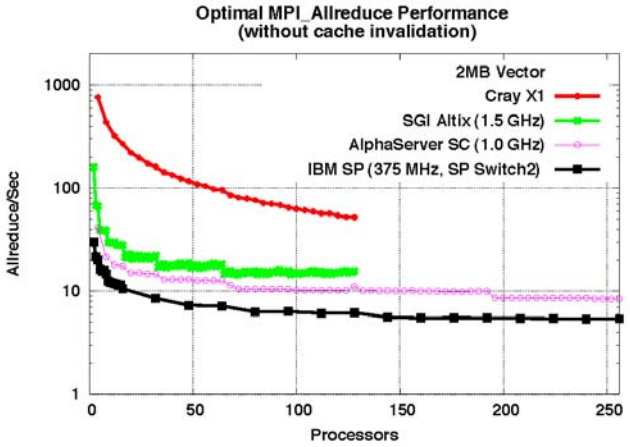

Figure 16: Performance of MPI_ALLREDUCE (2MB) across platforms.

Figure 14, Figure 15, and Figure 16 compare MPI_ALLREDUCE performance across a number of platforms for each of the three vector lengths. The Altix has the best MPI_ALLREDUCE for short vectors, but the Cray X1 has better performance for long vectors. In data not shown here the Altix and the X1 have identical performance when using the optimal ALLREDUCE algorithms for the 8Kbyte vector. However, the X1 has better performance for the 2 MByte vector even when using the optimal algorithms.

5 Kernels

The kernel benchmarks bridge the gap between the low-level microbenchmarks and the resource intensive application benchmarking. We used industry-standard kernels (ParkBench, NAS Parallel Benchmarks, Euroben) as well as kernels that we extracted from our scientific applications. We tested and evaluated single processor performance and parallel kernels with and without the vendor's parallel scientific library. We compared the performance of these kernels with other architectures and

have varied algorithms and programming paradigms (MPI, SHMEM, Co-Array Fortran, OpenMP). For example, Figure 17 compares the performance of the NAS multi-grid benchmark with various processor counts, architectures, and communication strategies.

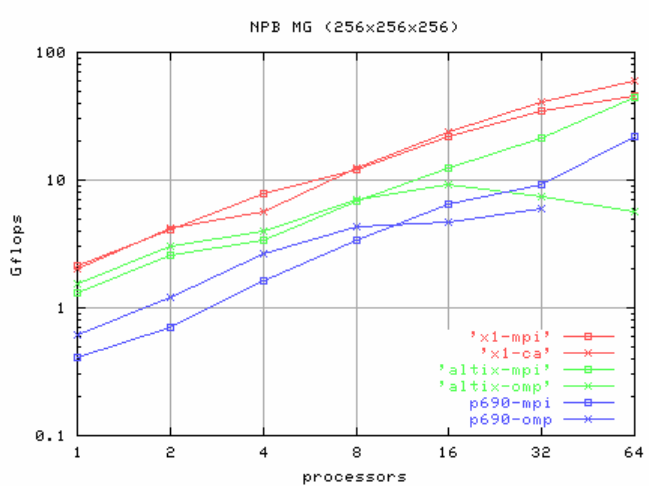

Figure 17: NPB MG benchmark.

We used a kernel representative of the dynamics algorithm used in the atmospheric component of the global climate model in an extensive comparative analysis. This kernel, the parallel spectral transform shallow water model (PSTSWM), supports different problem sizes, algorithms, and programming paradigms, and has been optimized on many parallel computer architectures. On the Altix we used PSTSWM to analyze compiler optimizations, evaluate math libraries, evaluate performance of the memory subsystem, compare programming paradigms, and compare performance with other supercomputers.

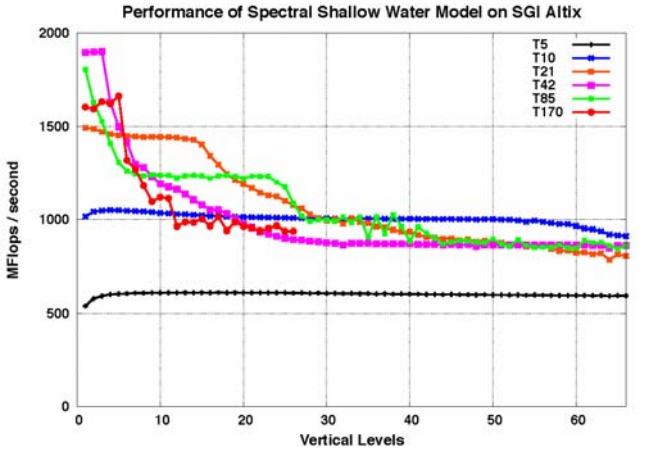

Figure 18: Sensitivity of PSTSWM performance to problem size on SGI Altix.

Figure 18 describes the sensitivity of PSTSWM performance to problem size on the Altix. 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. The X-axis is the number of vertical levels for a given horizontal resolution. Most of the problem coupling is in one or the other of the horizontal directions, and the vertical dimension simply controls the cache locality of certain phases of the computation. As the number of vertical levels increase, performance for the 4 largest problem sizes converges, slowly decreasing as the number of levels continues to increase. The performance curves look very similar to memory bandwidth curves used to illuminate the memory hierarchy, and we assume that the memory hierarchy is what is controlling the performance degradation as the number of vertical levels increases.

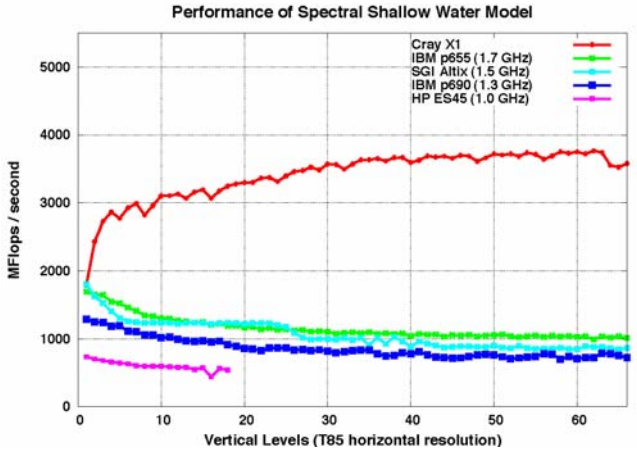

Figure 19: Performance of PSTSWM T85 across platforms.

Figure 19 compares the performance for the T85 problem on a number of HPC systems. These data show the advantage of the memory subsystem of the Cray X1 over that in the nonvector systems. However, the Altix performance holds its own compared to the other nonvector systems. These results are all for a single processor.

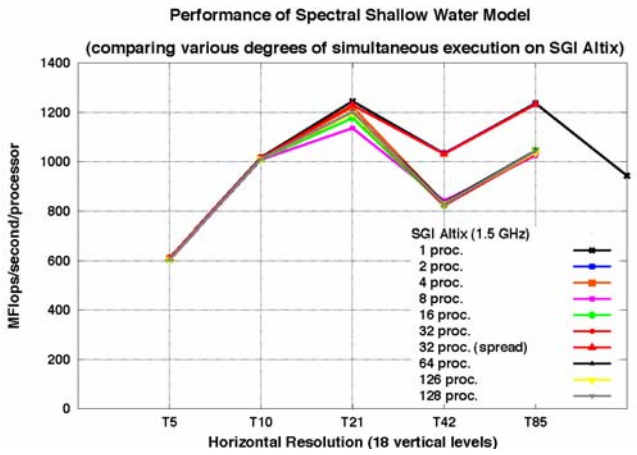

Figure 20: PSTSWM performance for different horizontal resolutions.

Figure 20 compares the performance for the different horizontal resolutions with 18 vertical levels when run on 1, 2, 4, …, 128 consecutive processors simultaneously.

Performance is identical when using 1 or 2 processors, or when using 32 processors when only every fourth processor is used. However, if all four processors in a C-brick are used, then there is contention for memory bandwidth and performance degrades for the larger problem sizes. This is the only situation where contention occurs, and performance does not continue to degrade as more processors are used.

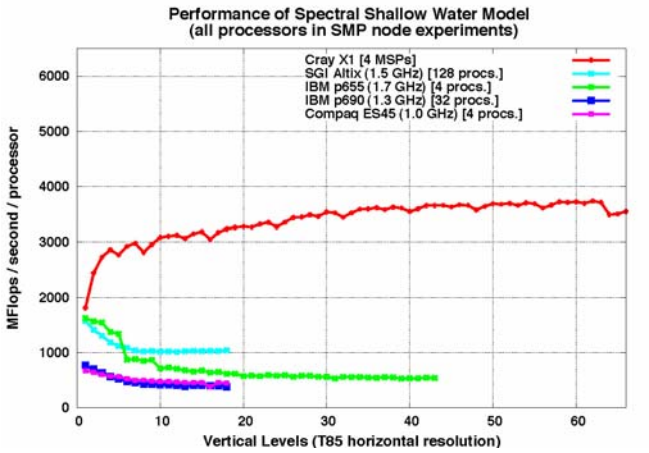

Figure 21: Effects of SMP node contention on PSTSWM.

Figure 21 compares the impact of contention when using all processors in an SMP node for problem T85 as the number of vertical levels increase. The Altix shows the least amount of performance degradation among the non-vector systems, indicating that the SGI memory subsystem scales very well for this type of memory contention benchmark.

6 Applications

6.1 Methodology

Two aspects of application benchmarking are emphasized in these preliminary results, identifying peak achievable performance and inter-platform comparisons. Standard benchmarks from, for example, NAS, PARKBENCH, SPEC-HPC, and SPLASH-2, have been run to allow comparison with published results for other platforms. Timings for kernels and production codes from the application areas were also measured and compared with existing timings on other machines.

6.2 Application Areas

ORNL computational scientists from a number of areas have expressed interest in working with the evaluation team to understand the promise of the SGI Altix architecture for their applications. While the emphasis is on computational biology and chemistry, applications from climate and fusion were also used in these initial studies.

6.3 Climate – POP

The Parallel Ocean Program (POP) is an ocean modeling code developed at Los Alamos National Laboratory that was developed to take advantage of high-performance computer architectures. POP is used on a wide variety of computers for eddy-resolving simulations of the world oceans [8, 11] and for climate simulations as the ocean component of coupled climate models. For example, POP is the ocean component of the Community Climate System Model (CCSM) [3], the primary model for global climate simulation in the U.S.

POP has proven to be a valuable tool for evaluating the scalability of HPC systems. It is comprised of two computational kernels, the baroclinic and the barotropic. POP parallelization is based on a two dimensional decomposition of the horizontal grid (leaving the vertical dimension undecomposed). Communication is required to update halo regions and to compute inner products in a conjugate gradient linear solver. The baroclinic phase scales very well on most platforms, with computation dominating communication until the processor count becomes large. In contrast the barotropic is dominated by a slowly converging iterative solution of a linear system used to solve a 2D elliptic problem. The linear system is solved using a conjugate gradient method, which requires halo updates to compute residuals and global reductions to compute inner products. The barotropic is very sensitive to communication latency, and the best that can be hoped is that the time spent in the barotropic does not grow with processor count for large processor counts.

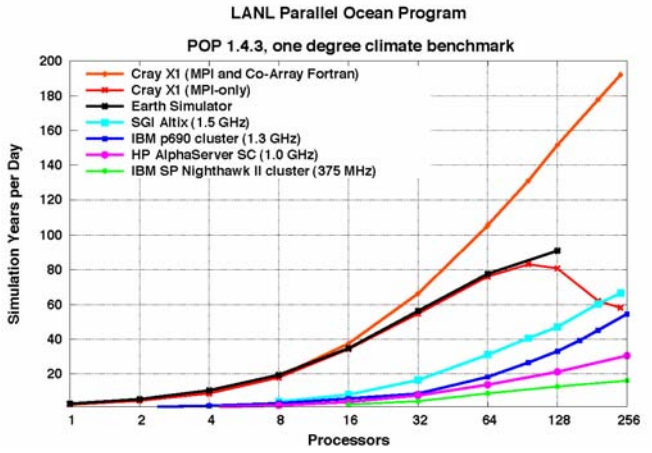

Figure 22: POP performance across platforms.

Figure 1 compares the performance of POP for a relatively small benchmark using a computational grid with a one-degree horizontal resolution. This problem size is the same as used in current coupled climate model simulations. POP has been vectorized to run on the Cray X1 and the Earth Simulator, and different versions of the code were run on the vector and non-vector systems to produce the data in this figure. On the Altix, optimizations included empirical determination of optimal domain decompositions and tuning of certain of the communication protocols. While the vector

systems were the best performers, the Altix showed the best performance of the non-vector systems and achieved 30% of the X1 performance. The Altix performance was better than that of the X1 when the X1 used only MPI. (An alternative implementation of POP using SHMEM instead of MPI did not improve POP performance on the Altix.)

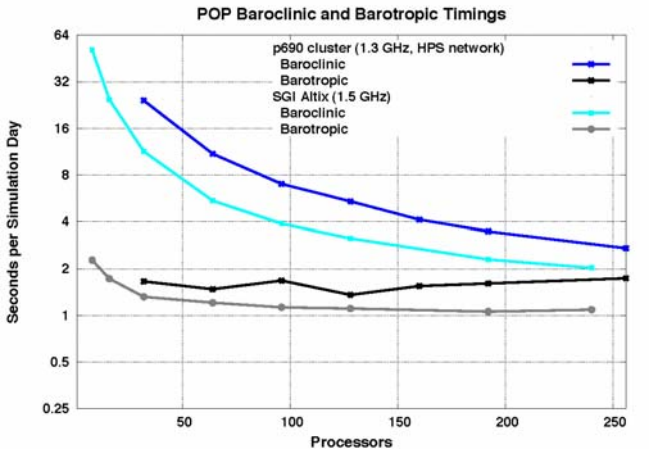

Figure 23: POP baroclinic and barotropic performance (p690 and Altix).

Figure 23 and Figure 24 compare the performance of the Altix on the baroclinic and barotropic phases with that of the IBM p690 cluster and the Cray X1, respectively. The Altix has a 50-100% advantage over the IBM system in the computation-bound baroclinic phase, and the barotropic phase scales better on the Altix than on the p690 cluster. This benchmark is computation bound on both systems out to 248 processors. The Altix is 3 times slower than the X1 on both the baroclinic and the barotopic at 248 processors, but is scaling equally well on both.

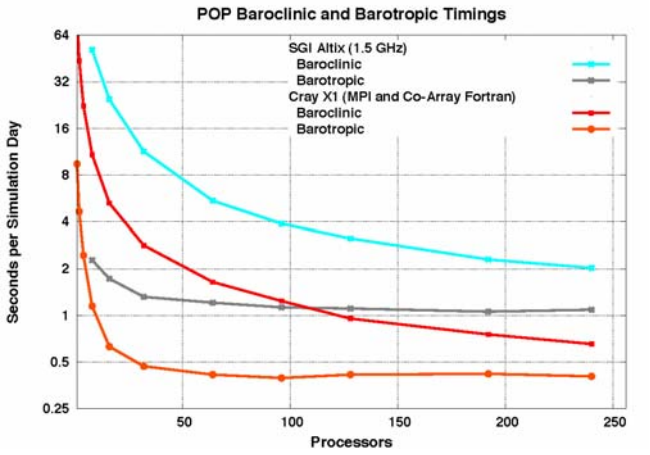

Figure 24: POP baroclinic and barotropic performance (Altix and X1).

6.4 Fusion – GYRO

GYRO is an Eulerian gyrokinetic-Maxwell solver developed by R.E. Waltz and J. Candy at General Atomics [Candy2003]. It is used to study plasma microturbulence in fusion research. GYRO uses the MPI_ALLTOALL command to transpose the distributed data structures and is more sensitive to bandwidth than to latency for large problem sizes.

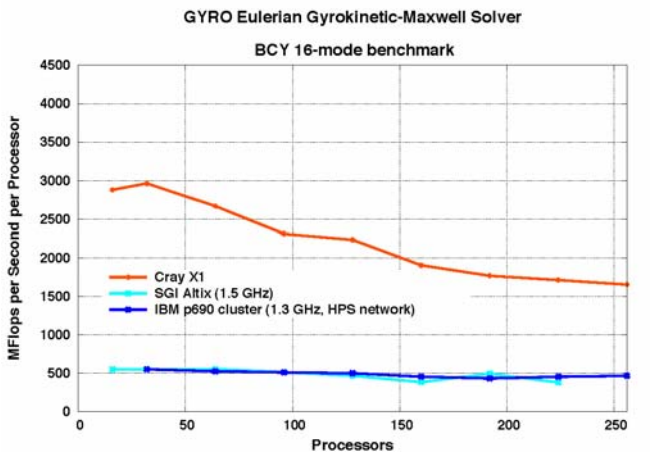

Figure 25: GYRO 16-mode performance across platforms.

Figure 25 and Figure 26 describe the per processor performance of two different benchmark problems: a small 16-mode problem labeled BCY and a large 64-mode problem labeled GTC. The metric, MFlops/sec/processor, is calculated using the same floating point operation count for each system, so performance between the different systems can be compared directly. This view of performance is useful in that it allows both raw performance and scalability to be displayed in the same graph.

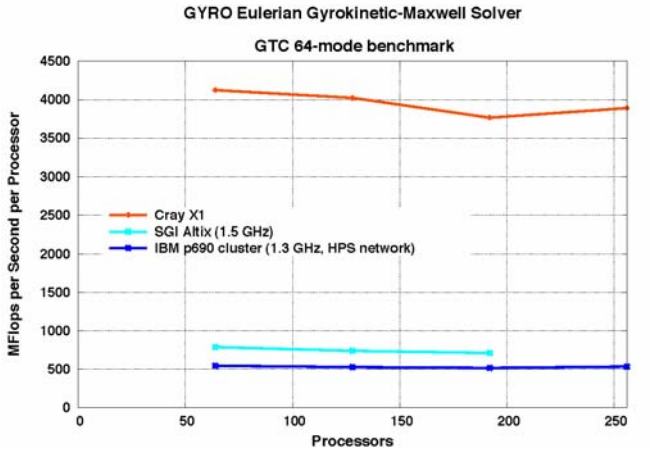

Figure 26: GYRO 64-mode performance across platforms.

Scalability on the non-vector systems is excellent for both benchmarks (and is also very good for the large benchmark on the Cray X1). Note, however, that while the Altix is 60% faster than the IBM on the GTC benchmark, it achieves essentially identical performance on the BCY benchmark.

Figure 27 depicts the fraction of the total time spent in MPI communication on each system for the two benchmark

problems. Since the total time differs for each system, these values cannot be compared directly. However, it is clear that time spent in MPI_ALLTOALL is impacting performance on the Altix to a much greater degree than on the other systems. For example, on the X1, communication is never more than 10% of the execution time, and the small and large problem sizes demonstrate similar percentages. On the p690 cluster, communication is a much smaller percentage of the time for the large problem size, indicating that the computational complexity is increasing faster than the communication complexity (and that the IBM HPS switch is able to handle the increased bandwidth demands).

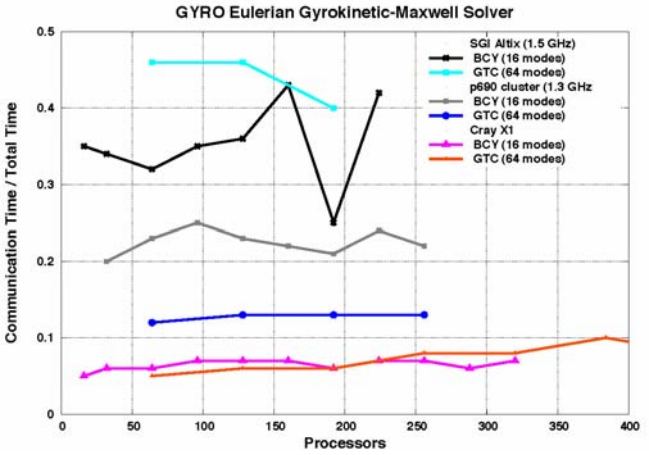

Figure 27: Communication fraction for GYRO.

In contrast, on the Altix the fraction of time spent in communication is higher for the larger benchmark, indicating that the interconnect is having difficulty communicating the large messages. It is unclear at this time whether there is a performance problem in the MPI_ALLTOALL or whether this is a limitation in the Altix network. The communication microbenchmarks indicated that the Altix network should support higher bandwidth rates than the IBM system. However, part of the MPI_ALLTOALL communicates within the p690 SMP node, and the IBM may have an advantage there.

6.5 Fusion – AORSA

Previous full-wave models for radio frequency (rf) heating in multi-dimensional plasmas have relied on either cold-plasma or finite Larmor radius approximations. These models assume that the perpendicular wavelength of the rf field is much larger than the ion Larmor radius, and they are therefore limited to relatively long wavelengths and low cyclotron harmonics. Recently, alternate full-wave models have been developed that eliminate these restrictions. These "all orders spectral algorithms" (AORSA) take advantage of new computational techniques for massively parallel computers to solve the integral form of the wave equation in multiple dimensions without any restriction on wavelength relative to orbit size, and with no limit on the number of cyclotron harmonics retained. These new models give high-resolution, two-dimensional solutions for mode conversion and high harmonic fast wave heating in tokamak geometry. In addition, they have been extended to give fully three-dimensional solutions of the integral wave equation for minority ion cyclotron heating in stellarator geometry. By combining multiple periodic solutions for individual helical field periods, it is possible to obtain complete wave solutions valid over the entire volume of the stellarator for arbitrary antenna geometry.

Figure 28 shows the scientific output of the problem computed on these systems, while Table 4 shows the performance of the major components of AORSA across four platforms.

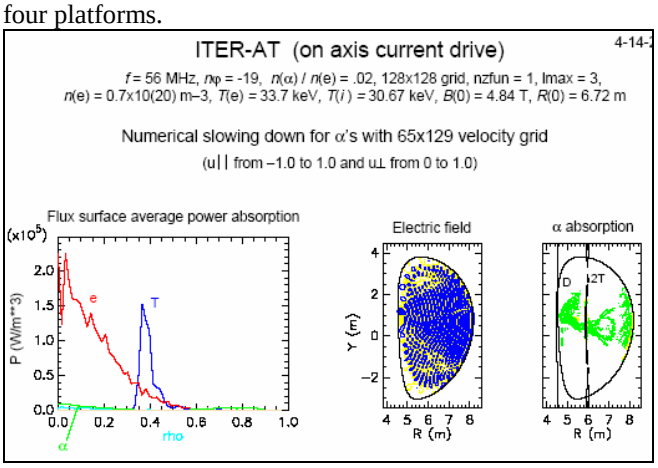

Figure 28: AORSA fusion results.

The overall performance of the SGI Altix is very competitive with the other three platforms at both the 64 and 128 processor experiments. The plasma current module performs the best on the Altix.

Table 4: AORSA performance results (minutes).

128 processors	Seaborg (IBM POWER3)	Cheetah-f (IBM POWER4)	RAM	Phoenix
Matrix load	207.7	37.3	10.2	13.7
ScaLAPACK	8.5	3.9	4.2	2.0
Plasma current	151.9	45.7	12.4	16.4
Absorption (wdot)	86.3	39.7	37.2	31.4
Total cpu time	456.1	127.4	64.2	64.1

64 processors	Cheetah-f (IBM POWER4)	RAM	Phoenix
Matrix load	74.4	34.1	27.4
ScaLAPACK	7.3	7.8	3.1
Plasma current	76.2	20.8	27.1
Absorption (wdot)	52.8	56.9	50.4
Total cpu time	211.4	122.1	108.5

6.6 Chemistry: NWChem

NWChem is a large code (nearly 1 million lines) with a wide variety of computational kernels that vary in the demands they place upon the underlying hardware. For example, the Gaussian-based DFT and the MD codes are sensitive to the latency of remote memory access, the CCSD(T) code requires efficient local matrix multiplication,

and the MP2 gradients and several other modules require high-performance I/O with both sequential and random access.

In contrast to the other applications codes in the evaluation report, NWChem uses a distributed, shared-memory programming model supported by the Global Arrays library. Point-to-point message passing is only used in third-party libraries, such as for parallel linear algebra or FFTs. The NWChem algorithms are typically based upon a non-uniform memory-access model. Shared data (in a global array) are assumed to be accessible by any process without the explicit involvement of any other process. This one-sided access is critical to realizing the efficiency and ease of programming demonstrated in NWChem.

The following benchmarks compare the HP Itanium2 system at PNNL and the SGI Altix Itnaium2 system at ORNL. As both of these systems utilize the same microprocessor the primary difference is the interconnect. The HP Itnaium2 is interconnected with Quadrics QSNET-2. QSNET-2 supports low latency (<3us) one-sided RDMA data transfers. The SGI supports global addressable memory via hardware. The Global Arrays library was ported and optimized for both approaches.

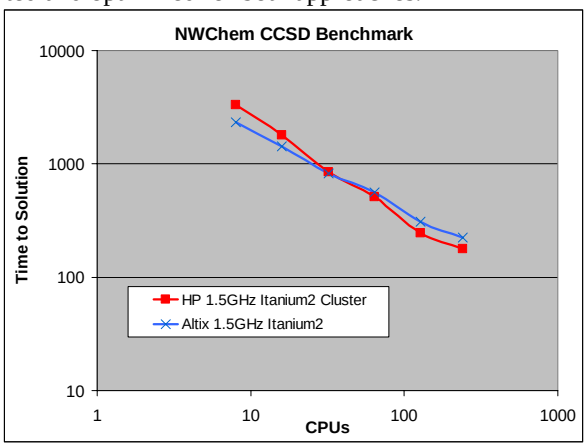

Figure 29: NWChem CCSD Benchmark.

The CCSD calculation, shown in Figure 29, is dominated by local matrix multiplications (DGEMM). The initial performance of the Alitx proved very promising and both the SGI and HP sustained over 20% sustained floating point performance for 8 CPU runs on this calculation.

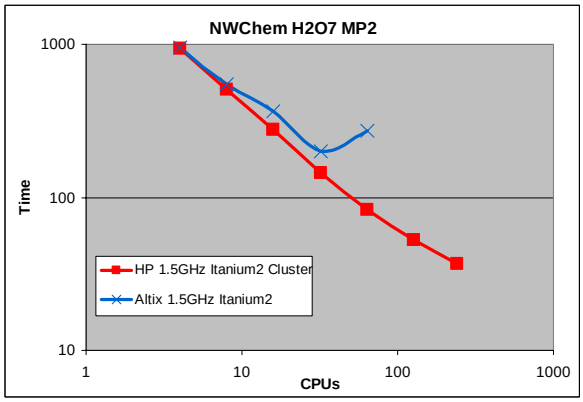

Figure 30: NWChem H207 MP2.

The MP2 gradients require high-performance I/O with both sequential and random access. As Figure 30 shows, the scalability of the IO subsystem on Altix hampered the ability to scale on the MP2 benchmark. While the Altix was only able to demonstrate a sustained 200MB/s aggregate performance the aggregate performance the HP cluster sustained 48GB/s over an equivalent 240 processors. This is because each node on the HP cluster has 7 disks to provide a 400MB/s scratch space. It is clear that for IO intensive applications that alternate configurations of the SGI system should be explored.

6.7 Biology: Molecular dynamics and Docking software.

Molecular dynamics (MD) and molecular docking simulations are being routinely used as an integral part of biological research. Given the importance of the MD and docking programs, several popular programs have already been ported and optimized for SGI Altix.[1] AMBER (Assisted Model Building with Energy Refinement) is a widely used software package for atomistic modeling of biological systems using molecular dynamics (MD) simulations, developed at University of California, San Francisco. Parallelization of MD codes is of wide interest to biological community, because with the current computational resources MD modeling falls short of simulating biologically relevant time scale by several orders of magnitude. The ratio of desired and simulated time scale is somewhere between 10^5– 10^6. Today's biological systems of interest consist of millions of atoms, which will require substantially more computing power of hundreds to thousands of processors for extended periods of time.

Pratul K. Agarwal, in collaboration with other ORNL researchers, is working on understanding protein structure, folding, dynamics and function through atomistic modeling using AMBER. Overall performance of AMBER on Altix

[1] http://www.sgi.com/industries/sciences/chembio/comp_chem.html

machine is excellent. The most important aspects are listed below:

- SGI's Altix shows a considerable decrease in AMBER time-to-solution as compared to IBM's POWER4 based machine (see Figure 31). This has wide implications for the long time scale simulations that can be carried out. Further, multiple MD runs are usually required to provide adequate sampling for most biological problems. Typically there are anywhere between 200-1,000 such runs, which use 8-16 processors per run. Use of the SGI Altix as compared to IBM machine cuts the project run time in half.
- Typically molecular modeling software show poor speedup with increasing number of processors. Figure 32 shows that speedup continues to improve for higher number of processors on the Altix as number of atoms in the simulation grows. This is a crucial point for the biologically relevant systems, which have thousands of atoms (typically 100,000-1,000,000).

CHARMM, GROMACS and NAMD are other popular parallel MD programs, which have already been ported and optimized for performance on SGI Altix. In addition, popular molecular docking program AutoDock, is also available on SGI Altix.

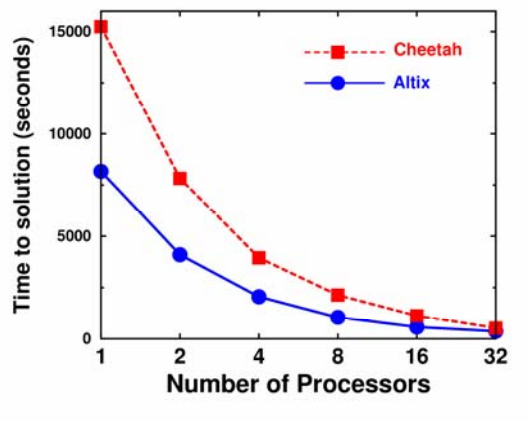

Figure 31: Amber solution time.

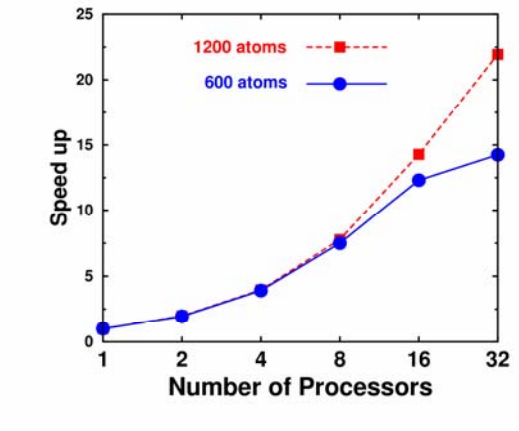

Figure 32: Amber speedup.

7 Visualization

The SGI Altix with 2 TB of shared memory and 256 Itanium-2 CPUs provides a very convenient platform to analyze data and render images for several hundreds of gigabytes to nearly a terabyte of output data from the Center for Computational Sciences (CCS) computers. A typical user would use one of the production software packages for 64 bit processors such as AVS and Ensight or one of the ASCI visualization software such as Visit or Paraview. A 32 bit processor can only address 4 gigabytes of data so it is quit limited for hundreds of gigabytes unless one has a large cluster. Such a cluster introduces additional difficulties in software efficiency and ease of use. In CCS, other than the SGI, the IBM p690 node with 128 mbytes and 32 CPUs is the second choice available for SIMD type of visualization computing with commercial software but it can't handle the hundreds of GBytes without significant preprocessing or filtering of the original data. The ease of use (i.e. I/O, job submission and interactivity) is a winner for the scientists and the analysts. A drawback to efficient usage is the network bandwidth to and from the large storage systems such as HPSS and the visualization cluster.

The main drawback of the current graphics capabilities of the SGI Altix is the lack of scalable hardware from SGI. SGI has recently revised its scalable graphics strategy and has just begun shipping and demonstrating cubes with AGP-8 support and ATI graphics processors. This is a significant upgrade from the AGP-1 graphics that was available on the SGI when it was installed and delivered at ORNL. The Altix provides the ultimate large memory platform, which is enormously easy to use for analysis and rendering. Although fast hardware graphics processors were not available, the large memory of the Altix was still able to be used to generate large and high resolution images.

7.1 Ensight

The visualization software Ensight is used extensively on the SGI Altix for large datasets and remote rendering.

The Terascale Supernova Initiative (TSI) and the ASPECT teams have analyzed and visualized large time-dependent dataset of hundreds of gigabytes (the largest so far consists of 480^3 grid points) per time step. This is the largest dataset that the SciDAC TSI team has analyzed so far. The TSI team has also experimented with geometry construction using the SGI Altix with the parallel rendering done remotely on the visualization computers at North Carolina State through Internet 2. This is essential for collaborations.

7.2 Parallel POVRAY

Parallel rendering of large and high-resolution images is a key step in producing images for the 35 MPixels resolution of ORNL's visualization facility EVEREST. A parallel ray tracing technique using POVRAY was used recently to render images from simulations of nano-scale materials on the Altix.

8 Parallel IO

8.1 ROMIO Parallel Writer for Large-Scale Visualization with PVTK

To support the needs in scientific visualization, we investigated the ROMIO-based parallel writer for large-scale visualization using the Parallel Visualization Tool Kit (PVTK) [7]. PVTK is a parallel implementation of the Visualization Tool Kit (VTK) [10], which is an object oriented tool kit for 3D graphics and visualization. PVTK provides parallel implementations for most of the available VTK algorithms, and is widely used for visualizing large datasets. Sophisticated scientific applications built within the PVTK framework generate very high resolution geometries containing billions of polygons. The generation of these geometries is both compute- and data-intensive. Multiple scientists then often view the resulting geometries. In addition, as more specialized hardware for applications are emerging, it is highly possible that the geometries are visualized at a location different from where it is originally computed. These reasons demand the need for storing geometries. Though PVTK provides scalable parallel visualization algorithms, it does not support high-performance parallel I/O in the current framework. Instead, PVTK serializes the I/O tasks through a single process, the master process. As a result, parallel visualization becomes bounded by I/O and large amounts of data transfer between processes. Efficiency of data flow between the application and storage depends on the efficiency of the I/O interface. Often serializing I/O through a single process has its own limitations in terms of memory requirements of the master process. Hence, it is highly desirable to have a parallel I/O capability within PVTK.

In this effort, we investigated our solution towards facilitating parallel I/O in PVTK. Our solution is based on MPI-IO, the I/O part of the MPI-2 standard [4]. More specifically, it utilizes ROMIO [12], a portable MPI-IO implementation that works on many different machines and file systems.

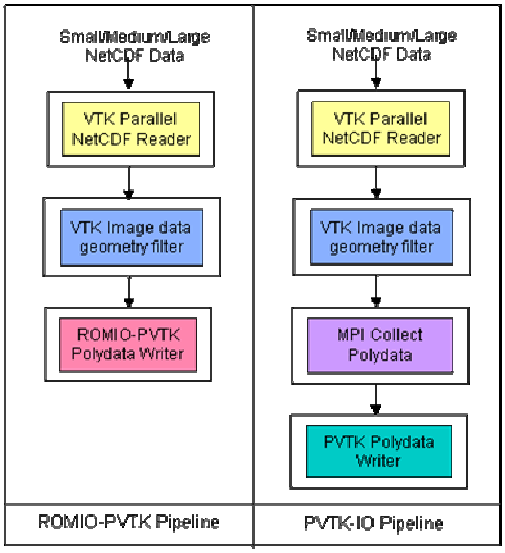

Figure 31: PVTK test pipelines.

The scalability analysis tests were performed on the SGI Altix 3700. Altix has roughly 6.2 TB of storage space mounted with SGI's XFS file system. Message Passing Toolkit (MPT version 1.6.1.) is the MPI library provided by SGI for Altix. We have used ROMIO version 1.2.4 that adjoins this distribution.

Two PVTK pipelines shown in Figure 31 were used for the testing purposes. Pipeline 1 (left), featuring the ROMIO-PVTK polydata writer has three VTK objects, our parallel NetCDF reader, an image data geometry filter and a ROMIO-PVTK polydata writer. Pipeline 2 (right) consists of four VTK objects, a parallel NetCDF reader, an image data geometry filter, an MPI polydata collector and an XML polydata writer. Both pipelines are functionally similar except that the ROMIO-PVTK polydata writer of Pipeline 1 replaces the MPI polydata collector and XML polydata writer of Pipeline 2. Three NetCDF source files of sizes 17MB (132x132x132), 137MB (262x262x262) and 274MB (330x330x330) were used for the testing purposes. When these source files were provided as input to the test pipelines they produced 64MB, 480MB and 960MB of polydata, respectively.

The pipelines were run on SGI Altix machine on 1, 2, 4, 8, 16, 32, 64, and 128 processors with the above mentioned NetCDF files as input. The outputs of the above pipelines are shown graphically in the following figures. These graphs depict the excellent scaling characteristics of ROMIO-PVTK writer. These figures show that as the number of processors increase the write time for ROMIO-PVTK PolyData Writer improves substantially, whereas the PVTK PolyData Writer writing time remains constant, thus suggesting excellent scalability across increasing number of processors. Across these figures, we compare ROMIO-

PVTK PolyData Writer with PVTK PolyData Writer for various sized outputs on 64 processors. It is seen that there is a drastic decrease in disk write time with the increase in the amount of data written while using ROMIO-PVTK PolyData Writer as compared to its peer PVTK PolyData Writer. Further, it can also be observed that the ROMIO-PVTK scales better than its counterpart.

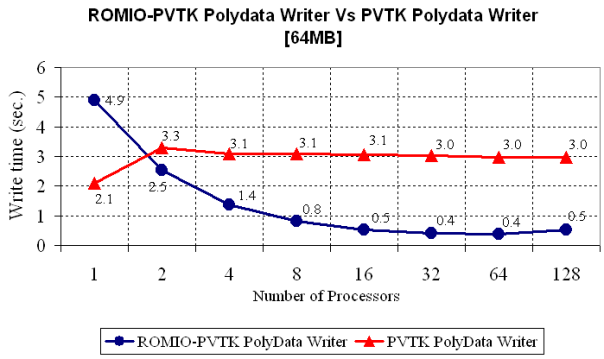

Figure 33: Performance of ROMIO-PVTK Polydata Writer and PVTK Polydata Writer for 64MB filesize.

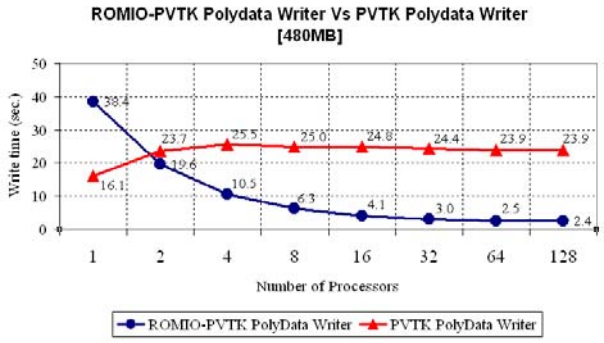

Figure 34: Performance of ROMIO-PVTK Polydata Writer and PVTK Polydata Writer for 480MB filesize.

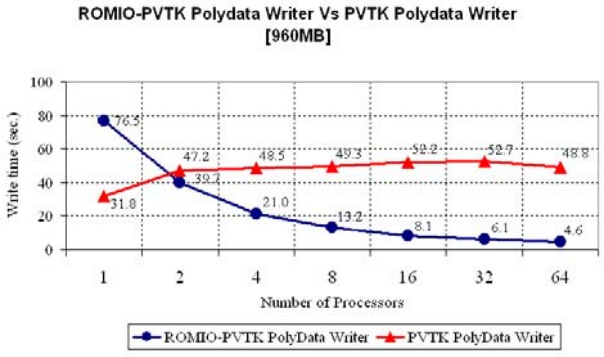

Figure 35: Performance of ROMIO-PVTK Polydata Writer and PVTK Polydata Writer for 960MB filesize.

9 Conclusions

In summary, the Altix provides many advantages over other non-vector machines and it is competitive with the Cray X1 on a few applications. Further, it has good scaling characteristics and careful consideration should be given before choosing to run IO intensive applications on the Altix architecture. We are continuing our evaluations, porting, optimizing, and analyzing additional application codes and looking in detail at open issues such as hybrid MPI/OpenMP performance and alternative parallel programming paradigms such as SHMEM and UPC. The system software on the Altix is also continuing to mature, with a recent move to a new Fortran compiler with somewhat different performance characteristics. Benchmarks will be rerun periodically to capture the performance evolution.

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 nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

Contributors

J.S. Vetter (Editor)	ORNL
G. Fann	ORNL
V. Meunier	ORNL
R. Toedte	ORNL
P. Chandramohan	ORNL
G. Kora	ORNL
R. Scott Studham	ORNL
N. F. Samatova	ORNL
D. Bauer	ORNL
P. Worley	ORNL
T. Dunigan	ORNL
P.K. Agarwal	ORNL
B. Sumpter	ORNL
F. Jaeger	ORNL
R. B. Ross	ANL
R. Latham	ANL
R. Thakur	ANL
M. Guest	CCLRC Daresbury Laboratory
J. Ahrens	LANL
R. Bleck	LANL
A. Shoshani	LBL
L. Oliker	LBL
D. Xu	University of Missouri-Columbia

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] M. Ashworth, I.J. Bush *et al.*, "Performance Comparison of Various Capability Benchmarks on the IBM p690+ and the SGI Altix 3700," Computational Science and Engineering Department, CCLRC Daresbury Laboratory, Daresbury, Warrington, Cheshire, UK, Report 2004

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

[4] W. Gropp, E. Lusk, and R. Thakur, *Using MPI-2: Advanced Features of the Message Passing Interface*. Cambridge, MA: MIT Press, 1999.

[5] 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

[6] F. Hoffman, "64-bit Computing with SGI's Altix," in *Linux Magazine*, vol. May, 2004, pp. 2-5

[7] J.Ahrens, C.Law *et al.*, "A Parallel Approach for Efficiently Visualizing Extremely Large, Time-Varying Datasets," Los Alamos National Laboratory, Los Alamos, New Mexico, Technical Report LAUR-00-1620, 2000

[8] M.E. Maltrud, R.D. Smith *et al.*, "Global eddy resolving ocean simulations driven by 1985-1994 atmospheric winds," *J. Geophys. Res.*, 103:30825--53, 1998.

[9] L. Oliker, A. Canning *et al.*, "Scientific Computations on Modern Parallel Vector Systems," Proc. SC 2004: High Performance Computing, Networking, and Storage Conference (*submitted*), 2004.

[10] W.J. Schroeder, K.M. Martin, and W.E. Lorensen, *The Visualization Toolkit An Object Oriented Approach to 3D Graphics*: Prentic Hall, 1996.

[11] R.D. Smith, M.E. Maltrud *et al.*, "Numerical simulation of the North Atlantic ocean at 1/10 degree," *J. Phys. Oceanogr.*, 30:1532-61, 2000.

[12] R. Thakur, R. Ross *et al.*, "Users Guide for ROMIO: A High Performance, Portable MPI-IO Implementation," Mathematics and Computer Science Division, Argonne National Laboratory, Technical Memorandum 234, 2002

[13] 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.