An Analysis of HPCC Results on the Cray XT4

Jeff Kuehn, ORNL; Jeff Larkin, Cray, Inc.; Nathan Wichmann, Cray, Inc.

ABSTRACT: The ORNL XT3 system has had numerous upgrades since it was built in 2005, taking it from a 56 cabinet single core XT3 system to a 124 cabinet, dual core XT3/XT4 combined system. Along the way the HPCC benchmark suite has been used to evaluate system performance. We will show and analyze the results from these benchmark runs and discuss the implications upon application writers.

KEYWORDS: CRAY XT3 XT4 HPCC BENCHMARK PERFORMANCE

1. Introduction

In the proceedings of CUG2006 we published a paper evaluating the performance of the HPCC benchmark suite on the Cray XT3, jaguar, and newly-upgraded Cray X1E, phoenix, supercomputers at the Center for Computation Sciences (CCS) at Oak Ridge National Laboratory (ORNL). Although phoenix has remained unchanged since this paper was published, jaguar has undergone significant changes. It has been upgraded from a single-core XT3 system to a combined dual-core XT3 and XT4 system. This paper revisits HPCC on jaguar and discusses the effect that the past year's system upgrades have had on overall performance.

2. Machine Descriptions

2.1. Cray XT3/XT4

The Cray XT4 supercomputer[6] is an evolutionary descendant of the Cray XT3 [1-5], Cray's third-generation MPP system. The XT4 includes an upgrade to the SeaStar network and system memory, as outlined below. The XT3/XT4 supercomputers are focused on high scalability and sustained application performance. Their principal attributes are balanced system performance, usability, scalability, and ease of upgrade.

The Cray XT4 can be constructed with as many as tens of thousands of nodes, or PEs, configured as either compute nodes or service nodes. Each node includes a 64-bit AMD Opteron processor with dedicated memory and a data routing resource. The Cray XT4 uses a simple memory model in which applications are distributed across any number of nodes, each node with its own processor and local memory, no shared memory. The XT3 used DDR400 memory parts which can deliver up to 6.4 GB/second bandwidth, more than 1 byte per flop per processing element. The XT4 uses more recent Opterons, which include a DDR2 memory controller. By upgrading to DDR2 memory, the effective memory bandwidth to each processor core improves from 6.4 GB/s for DDR-400 memory to 10.6 GB/s for DDR2-667 memory and 12.8GB/s for DDR2-800 memory. It has a very low memory latency of near 50 ns, benefiting performance algorithms that require irregular memory access patterns.

Each node communicates with the rest of the system via a high bandwidth, low-latency 3-D torus interconnect network. The processor is connected directly to the interconnect via a HyperTransport link to the Cray SeaStar communication processor and router chip. The SeaStar is designed to offload communications overhead from the Opteron processor to increase application efficiency and accelerate communications. The Opteron has a bidirectional injection bandwidth of more than 2 GB/s to and from the SeaStar chip, while each SeaStar can sustain a bandwidth of 6.5 Gbytes/s on each of the six links, one in each direction of the 3-D torus. The Cray XT4 uses the next-generation SeaStar2 interconnect, which is link-compatible with SeaStar, but nearly doubles the injection bandwidth.

The Cray XT3/XT4, like previous Cray MPP systems, uses a microkernel on its compute PEs called Catamount to minimize system overhead and in turn reduce "jitter". The general rule is that the microkernel only runs when it is working on behalf of the application. Catamount eliminates SMP overhead, paging, and spurious daemons, ensuring high, reproducible performance. As the XT3 supercomputers were upgraded from single to dual-core processors, Catamount had to be modified to support the additional core. This was done through *virtual node* mode, which gives half of the memory to each core and assigns a *master node* to handle interrupts and communications. The effects of this OS design on benchmark results are discussed later in this paper.

For the benchmarks, we compare last year's software stack of PGI 6.1 compilers, ACML 3.0 BLAS, and MPT 1.3 message passing libraries versus this year's stack of PGI 6.1, ACML 3.6, and MPT 1.5.

3. HPCC Benchmark Description

The HPC Challenge benchmark suite[9-12] tests multiple system attributes which exhibit substantial impact on the real-world performance of applications on HPC systems. The design goal for HPCC was to augment the venerable LINPACK benchmark with additional tests to support a broader analysis of HPC architectures. Additionally, since locality of reference can be a dominating factor in software performance on modern architectures, the selected tests examine how the architecture performs for high and low degrees of both spatial and temporal locality (see Figure 1). Collectively, the tests provide performance indicators for processing power, interconnect, and memory system performance. To do this, the test suite needed to use kernels that reflect real-world patterns of memory access and interprocessor data exchange. The resulting suite includes 23 tests in eight categories, and stresses not only the processors, but also the memory system and the interconnect maps each test category to the aspects of the HPC system it evaluates.

3.1. Network Parameterization

There are several different approaches to characterizing interconnect performance for HPC systems. The most thorough approach attempts to map the "performance surface" for several to all of the MPI data exchange routines as the processor count and message size are independently varied. The results are reported variously as either bandwidth or time, dependent on whether the concept of bandwidth is well defined for the operation in question. While this approach is the most thorough, performing the required tests to a high degree of accuracy is resource intensive. A simpler method which attempts to characterize the zero-byte message latency and the asymptotic bandwidth runs much faster and provides much of the same information, since more complex MPI operations can be understood in terms of these fundamentals. For instance, we can *approximate* the time for a single communication as:

time = latency + bandwidth*(message size)

Though this simplistic model fails to characterize performance inflections which indicate points at which the MPI libraries transition from one algorithm to another (e.g. Eager vs. Rendezvous), it captures enough of the essential character of the interconnect to indicate its impact on latency sensitive or bandwidth sensitive applications. HPCC takes this abbreviated approach of measuring latency and bandwidth, but recognizes that the communication pattern can play a role in determining these factors. Thus, the latency and bandwidth are measured for three different fundamental communication patterns.

3.1.1. PingPong

This is the most basic communication pattern and involves a simple send-receive exchange between two processors. While frequently you will see vendors publish results for the MPI PingPong Latency test, as it is relatively easy to measure, the MPI test measures a transfer of only 1 byte between only 2 processors in a system. Additionally, because latency and bandwidth vary for some interconnect topologies depending on the number of network hops between the sending node and the receiving node, HPCC reports not only an average PingPong time, but also the minimum and maximum PingPong times. The minimum PingPong time will be representative of the latency and bandwidth to an MPI task running on the same node for systems which run multiple MPI tasks per node or to an MPI task running on the nearest network neighbor for systems which run only a single MPI task per node.

3.1.2. Natural Ring

The Natural Ring test uses a more realistic scenario where all processors in the system communicate with their “neighbors”, at approximately the same time. However, it should be noted that “neighbors” in the context of the benchmark are defined as MPI tasks whose ranks differ by one. Thus, for larger SMP nodes, “neighbors” are likely a task on the same node, whereas for uniprocessor nodes, “neighbors” are at least one network hop away. In systems with SMP nodes, as the processor count per node increases, this benchmark becomes increasingly weighted towards reflecting the latency and bandwidth within a node. Codes parallelized with domain decomposition generally represent the decomposition in one of the dimensions as adjacent MPI ranks with the communication in this dimension mapping well to the Natural Ring communication pattern.

3.1.3. Random Ring

The Random Ring test is similar to the Natural Ring, except that the order of the ring is "randomized" so that each processors "ring neighbor" is neither its logical neighbor as ordering in the MPI communicator, nor its physical neighbor in the machine, instead it is a processor chosen at random in the system. While some applications do perform communication between random nodes, even more regular problems exhibit some behavior consistent with the Random Ring pattern, as is the case for domain decomposition in multiple dimensions. Specifically, only one of the dimensions can be represented by adjacent MPI ranks, and thus the others may be more topologically remote, mapping well to the Random Ring pattern.

Codes that send many small messages are very latency sensitive. For example the widely used ocean modeling code, Parallel Ocean Program (POP) from Los Alamos National Laboratory, and LS-DYNA engineering software from Livermore Software Technology Corp. (LSTC) are highly dependent on global summations, which require the processors to communicate frequently with small messages. The computing system’s latency is a significant factor in how well POP or LS-DYNA will run on that system, and therefore, Random Ring latency is a good predictor of real world performance for applications that place heavy demands on simultaneous communications.

3.2. Spatial vs. Temporal Locality of Reference

The design of the HPCC benchmarks acknowledges that applications vary in the type and magnitude of the data locality intrinsic to their core algorithms. Two types of locality are recognized: temporal and spatial. Temporal locality refers to the likelihood of a single memory address being referenced several times in a short period, whereas spatial locality refers to the likelihood of adjacent memory addresses being referenced in a short period of time. Since these two types of locality are orthogonal, one can locate any algorithm on a 2D plot of spatial-temporal locality diagram (STLD) (see Figure 1). 12 of the HPCC tests were selected because of their correspondence to the corners of this space, and the performance of these 12 tests provides us with a clearer picture of how the full application space will map to the architecture, than would be possible with but a single benchmark. (See Figure 1).

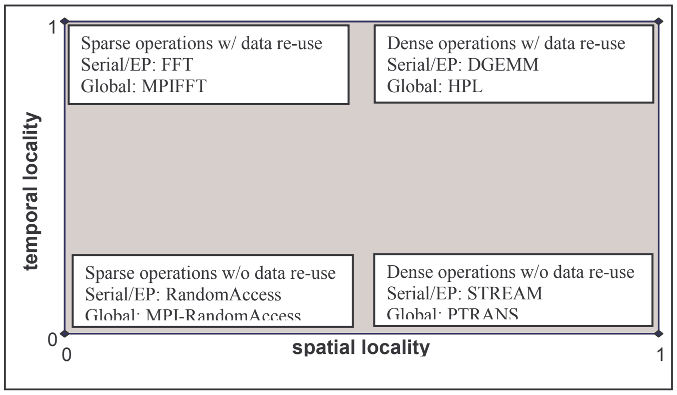


Figure 1: Spatial-Temporal Locality Diagram

3.3. Serial vs Embarrassingly Parallel vs Global

In addition to varying the spatial and temporal locality, the HPCC benchmark also recognizes three levels of parallelism. First, a serial (“SP” or “single”) test involves only one processor running the test. Second, an embarrassingly parallel (“EP” or “*”) test is run simultaneously by each of the processors, though without any communication required between the processors. And third, a globally parallel ("G") test is run in which all of the processors collaborate on a single test requiring communication. Thus, for each of the four corners of the STLD (Figure 1), we characterize the architecture's performance for each of three levels of parallelism creating a total of 12 application kernels.

3.4. High Spatial and High Temporal Locality

Dense linear algebra was chosen to represent those algorithms which exhibit both high spatial locality and high temporal locality. DGEMM measures the floating point execution rate of double precision real matrix-matrix multiplication in GigaFLOPS per second (GFLOPS) using the DGEMM BLAS library routine in both SP and EP mode. DGEMM de-emphasizes memory performance relative to CPU performance and the timed kernel of the benchmark does not use the interconnect. DGEMM is useful as a measure of the very best performance one could ever achieve on highly computational intensive codes.

The High Performance LINPACK test (HPL) is used for the global test in this quadrant. It primarily tests processor performance and is included in this suite to allow comparison with the Top 500 list, which is based on the very same test. HPL solves a randomly generated dense linear system of equations in double floating-point precision (IEEE 64-bit) arithmetic using MPI and is measured in teraflops, or trillions of calculations per second (TFLOPS). HPL has the enviably characteristics of being very computationally intensive as well as requiring relatively little local or global memory bandwidth. As a result, many systems can sustain 70-80+% of the theoretical peak floating point performance. Results for DGEMM correlate strongly with per CPU HPL results.

3.5. High Spatial and Low Temporal Locality

Algorithms dominated by data movement were chosen to represent those algorithms which exhibit high spatial locality but low temporal locality. The STREAM benchmark was chosen for the SP and EP tests in this quadrant. STREAM is a simple synthetic benchmark that measures sustainable memory bandwidth to local memory, in Gigabytes per second, and the corresponding computation rate for a simple vector kernel. STREAM provides a good approximation for high memory bandwidth codes. It is worth noting that STREAM assesses bandwidth to memory only, and does not stress interprocessor communications, or I/O communications. For the global benchmark in this quadrant a parallel matrix transposition is used. The PTRANS (parallel matrix transpose) benchmark implements a parallel matrix transpose for two-dimensional block-cyclic storage. This is an important benchmark because it exercises the communications of the computer heavily on a realistic problem where pairs of processors communicate with each other simultaneously. PTRANS is a useful test of the total communications capacity of the network, measured in gigabytes per seconds (GB/s).

Several molecular dynamic codes and most climate models must transpose large arrays to perform multi-dimensional Fast Fourier Transforms (FFTs), relying heavily on the processor's ability to exchange data quickly. Applications such as CPMD (a computational chemistry code that simulates static and dynamical properties of solids, liquids and disordered systems), FPMD, and VASP molecular dynamics simulation codes and climate spectral models, are most likely to perform well on systems that do well on the PTRANS benchmark.

3.6. Low Spatial and High Temporal Locality

Fast Fourier Transform algorithms were chosen to represent those algorithms which exhibit low spatial locality but high temporal locality. The FFT test measures the floating point rate of execution of complex one-dimensional Fast Fourier Transforms in double precision. The FFT is performed in each of the three (single, EP, global) modes. The global FFTE exercises the computation and the all-to-all communications capabilities of the computer, providing a useful test of the total communications capacity of the computers network, or interconnect. It is measured in Gigaflops per second.

3.7. Low Spatial and Low Temporal Locality

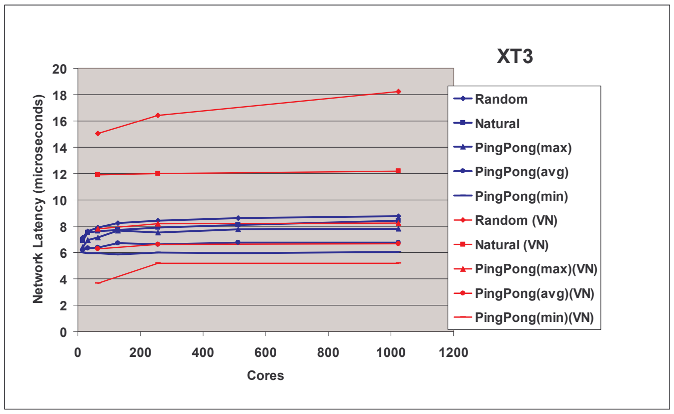
This quadrant is the newest frontier of scientific algorithm research, including graph theory and adaptive mesh refinement (AMR). Excelling in this quadrant is challenging for most modern computers since the locality constraints provide few opportunities to amortize memory latency. A graph theory kernel called RandomAccess (RA) was chosen to represent those algorithms with both low spatial and low temporal locality. The Random Access test measures the rate at which the computer can update pseudo-random locations of a large table stored in its memory, expressed in billions (giga) of updates per second, or GUPS. By testing gather-scatter functions, the Random Access kernel provides an indication of how codes (including graph theory and AMR) which access data structures in more random patterns, will perform on a given system. The RA benchmark is run in single (one table on a single node), EP (each node with its own table) and global (one large table spanning all nodes) modes.

4. Results and Discussion

4.1. Network Parameterization

The HPCC benchmark measures latency and bandwidth for a two task Ping Pong pattern, and two ring patterns, one ordered naturally as implied by the MPI task ranks and one ordered randomly. For the Ping Pong test, multiple pairs are tested and reported as a minimum, average and maximum for both the latency and bandwidth. For a system built on single processor nodes with a reasonably good mapping of MPI tasks to nodes, one would expect the latency and bandwidth for the Natural Ring to closely track those of the Ping Pong test since the minimum latency and maximum bandwidth would typically be observed between nodes which fall closer together within the network. Whereas the Random Ring test typically generates communications between MPI tasks ranks that are further apart, and hence between nodes that are further apart within the network. The latency and bandwidth results on the Cray XT3 and XT4, fail to follow this pattern (Figures 3 and 4). The Natural Ring latency and bandwidth track more closely with the Random Ring latency and bandwidth rather than the Ping Pong latency and bandwidth. Thus the task placement, the mapping of MPI task ranks to nodes on the network, behaves less like an expected good placement and more like a random placement, i.e. the placement is far from optimal. It should be noted that while the XT3/XT4 behavior is not optimal, it is typical of the majority of the results available at the HPCC results website[12].

The cause of these results is apparent from the manner in which the nodes, cabinets, and rows within the Cray XT3/XT4 system are numbered sequentially in each of these directions, presumably to simplify locating individual hardware components by physical location for maintenance.[7,8] However, to minimize the maximum cable length, the wiring within the cabinets and between cabinets down a row are cabled in a serpentine fashion – in each of these directions, the odd numbered nodes are linked in a chain and the even nodes are linked in a chain, then the chains are linked at the ends. Thus, with the exception of those nodes which are the endpoints linking the odd and even chains, nodes which are physically adjacent to each other, are topologically remote, potentially opposite each other on a torus ring. Thus, nodes allocated and mapped sequentially (according to physical position) when provisioning resources for a job, will be topologically remote from each other in the network, giving rise to the observed behavior. Furthermore, as multiple jobs are provisioned, the overall impact will be to interleave jobs among each other on the torus, causing the majority of the node to node links to be shared by more than one job. Also worth noting is that the latencies increase and the ring bandwidths decrease as the processor count increases. The latency increase is easily explained by the increasing network hop count required to span the larger job configuration. The bandwidth decrease is less intuitive, but in the case of the Random Ring test, is not difficult to imagine the ring wrapping back on itself due to the random place of tasks, leading to contention for the links. Because of the placement problems noted previously, the Natural Ring pattern is similarly impacted, though to a lesser extent. It should be noted that the benchmarks were not run on a dedicated system and the scattered placement implies that nodes belonging to other jobs are likely to be interspersed within the nodes for the benchmark jobs as an additional source of link contention.



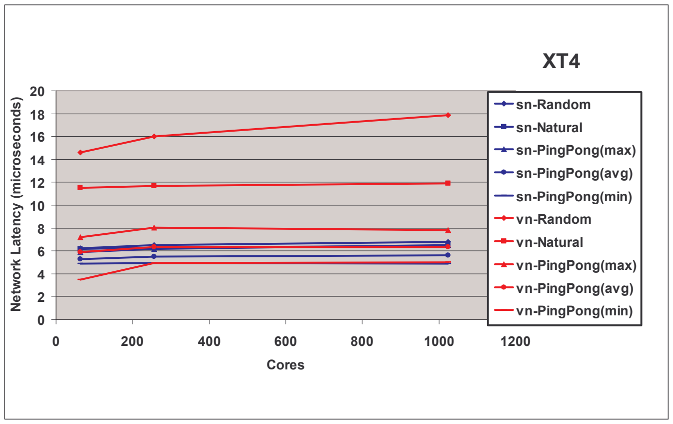
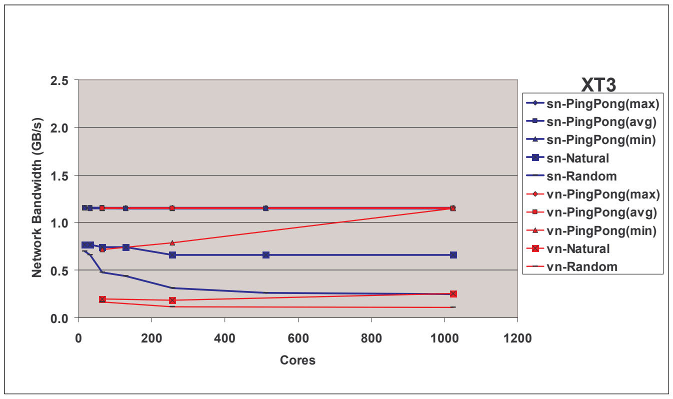


Figure 2a-b: XT3 and XT4 latencies.

When comparing the results from the XT4, one will notice that both latencies and bandwidth improved over the XT3. The improved latency may be as much due to software improvements since the original XT3 benchmarks were run as they are due to the newer interconnect. The doubled injection bandwidth of the SeaStar2 NIC is clearly evident in Figure 3b. We still observe that rank ordering negatively affecting both benchmarks.



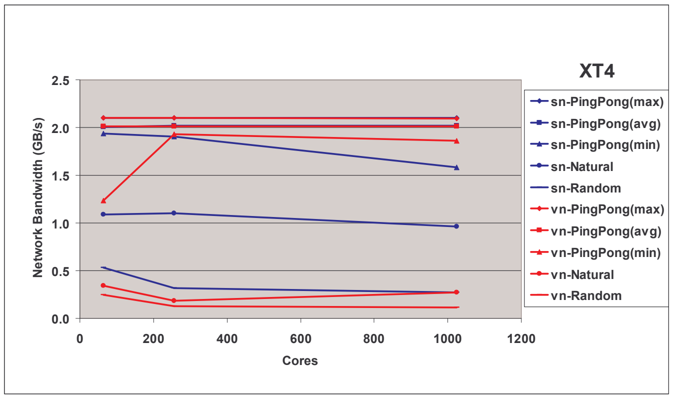
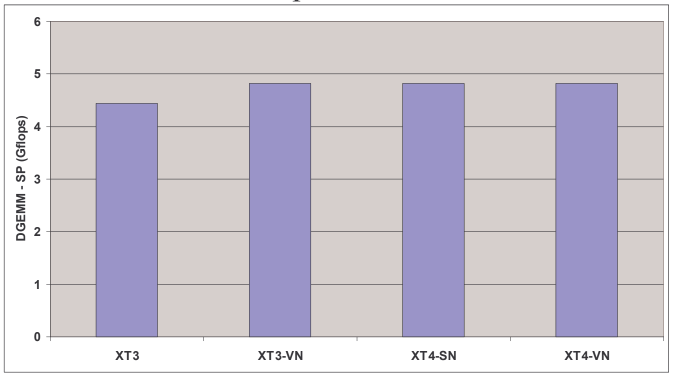


Figure 3a-b: XT3 and XT4 Bandwidths

4.2. High Spatial and High Temporal Locality

4.2.1. DGEMM

Figure 4 shows both the SP and EP performance of the DGEMM benchmark on the XT3 and XT4. The charts clearly show the higher clock frequency of the XT3 dual-core upgrade and the XT4. We observe the expected performance improvement from the minor clock speed increase.



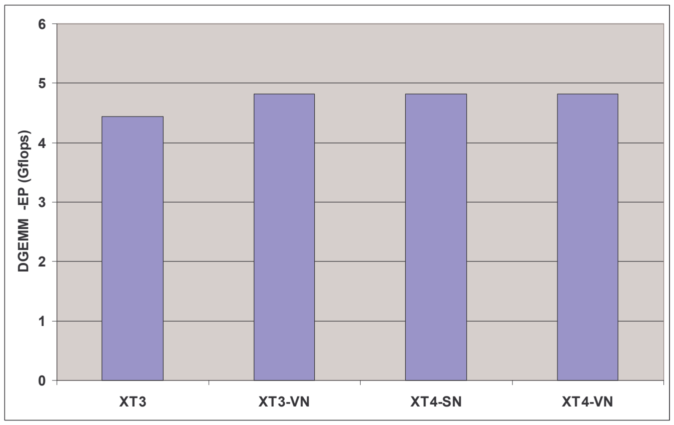
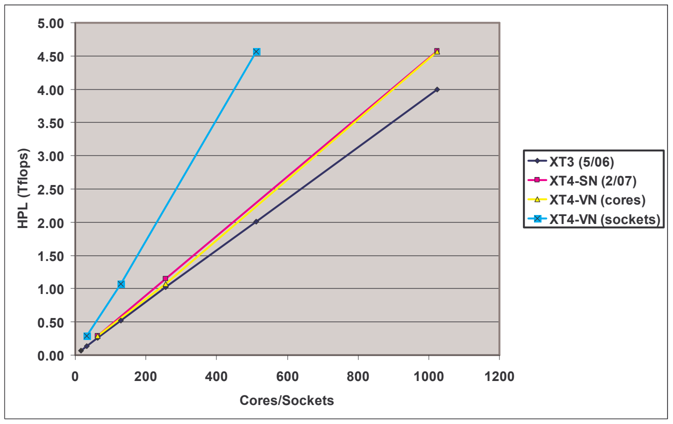


Figure 4a-b: DGEMM SP and EP performance

4.2.2. HPL

The results from the HPL benchmark are likewise exactly as expected: the slight increase in clock speed gives a slight improvement in performance. Overall performance improved both when looking at a per-core or per-socket level. Figure 5b shows HPL results when normalized to the theoretical peak performance. On each system tested we were able to achieve greater than 80% of peak. The XT4-SN benchmark received roughly half of the performance of the XT4-VN performance, which is exactly what one would hope.



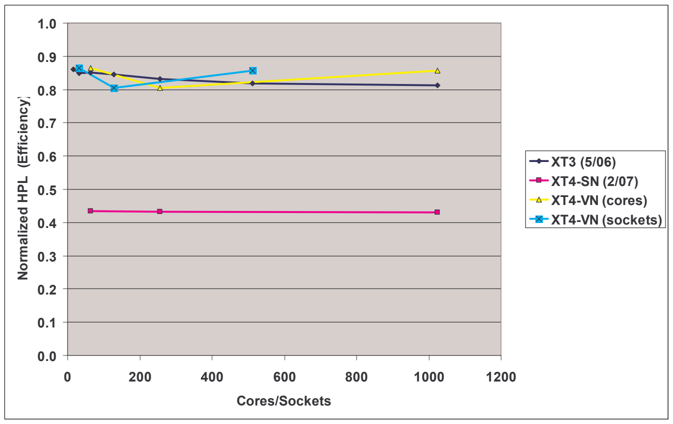
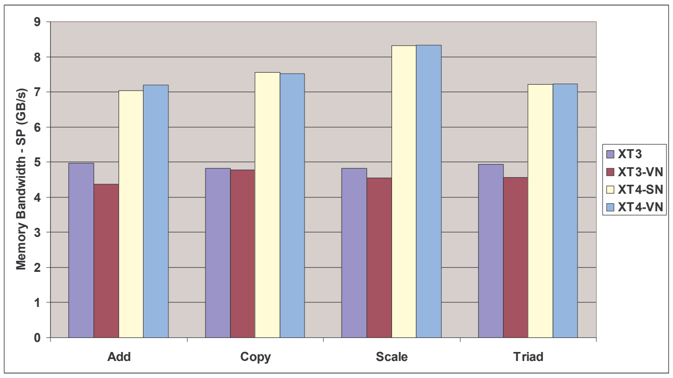


Figure 5a-b: HPL performance and Normalized HPL Performance

4.3. High Spatial and Low Temporal Locality

4.3.1. STREAM

The STREAM benchmark measures memory bandwidth. Figure 6 shows both the SP and EP versions of the STREAM benchmark. Notice that the DDR2 memory of the XT4 achieves higher bandwidth than the DDR memory of the XT3. The EP graph shows a very clear indication of a memory bottleneck when using both cores of a dual-core processor. While the AMD Opteron has two general purpose computational cores, they share a single memory subsystem. The graphs in Figure 6 clearly show that for both the XT3 and XT4, one core is capable of saturating the memory subsystem and dual core bandwidth suffers as a result.



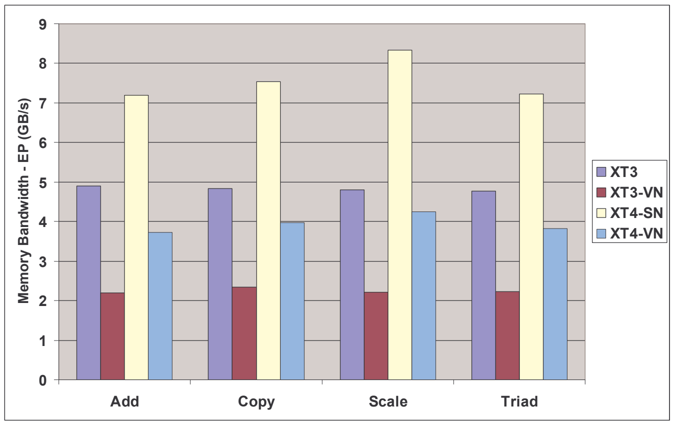
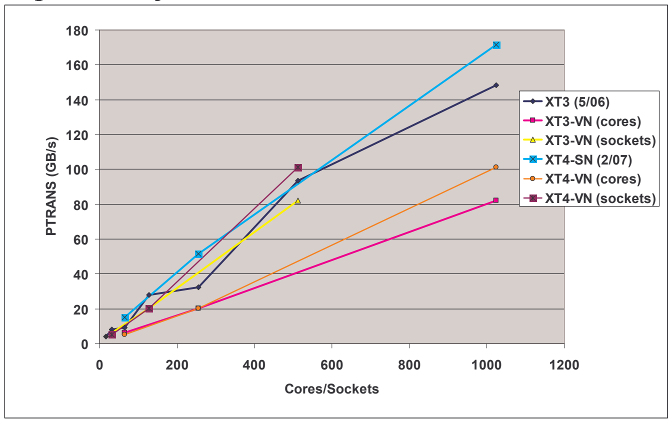


Figure 6a-b: STREAM SP and EP performance.

4.3.2. PTRANS

The PTRANS benchmark performs a parallel matrix transpose, much like those needed by applications that require large FFT operations. The single core performance is greater than the dual-core performance and the XT4 performance is slightly higher than XT3 performance due to the improved injection bandwidth.



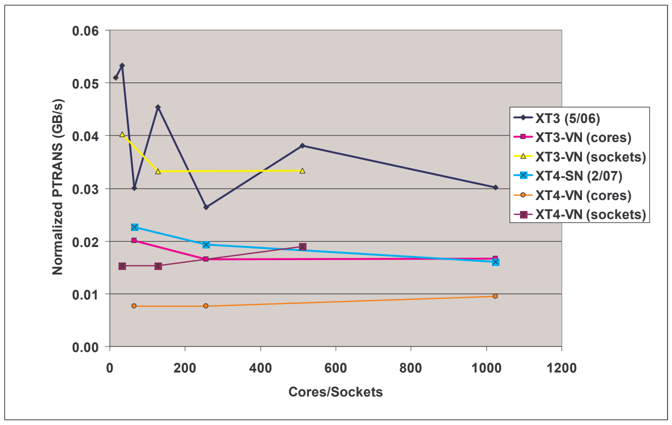
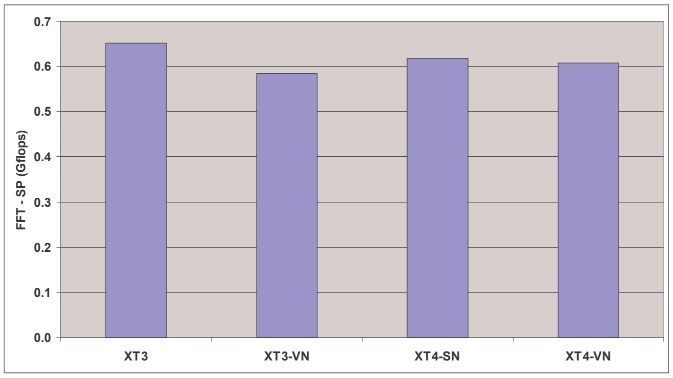


Figure 7a-b: PTRANS results and normalized results

4.4. Low Spatial and High Temporal Locality

4.4.1. FFT

Figure 8 shows the SP and EP FFT performance on the XT3 and XT4. It is important to note that the results labeled *XT3* appear to be anomalous and should be disregarded. While we cannot fully explain why these results were greater than the other results, it is most likely a difference in system software or benchmark configuration. The improved memory bandwidth clearly gives improved performance in the SP benchmark. The EP benchmark results show the memory controller bottleneck on a dual-core Opteron, but the dual-core results are still an over-all win and the XT4 dual-core results are slightly better than the XT3 dual-core results.



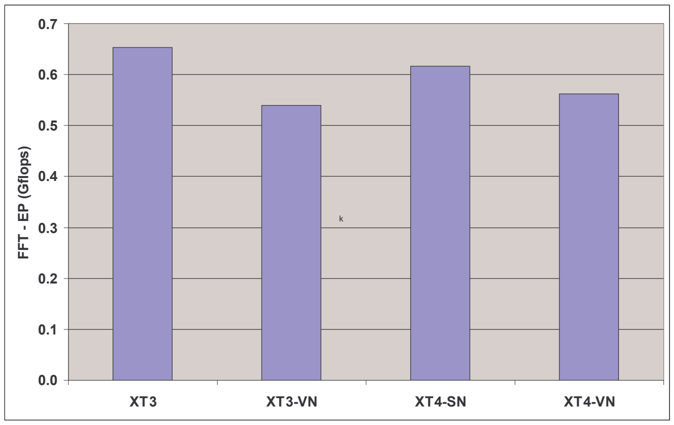
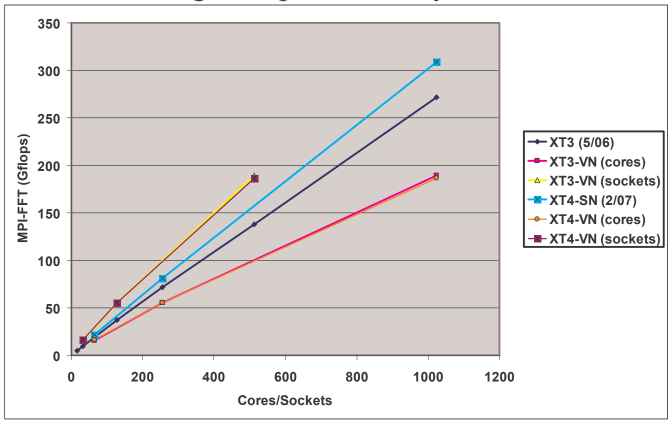


Figure 8a-b: FFT SP and EP performance

4.4.2. MPI-FFT

The MPI-FFT benchmark extends the FFT benchmarks to the entire system. The benchmark shows very favorable results that are similar, but slightly lower-performing, as the HPL benchmark.

While the single-core results are better than the dual-core results on a core-by-core comparison, when comparing on a socket-by-socket level the dual-core performance is slightly better. Thus the dual-core processor does provide a benefit in kernels with high temporal locality.



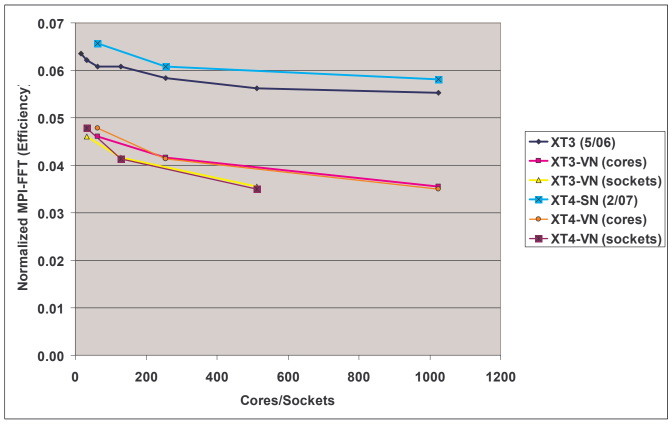
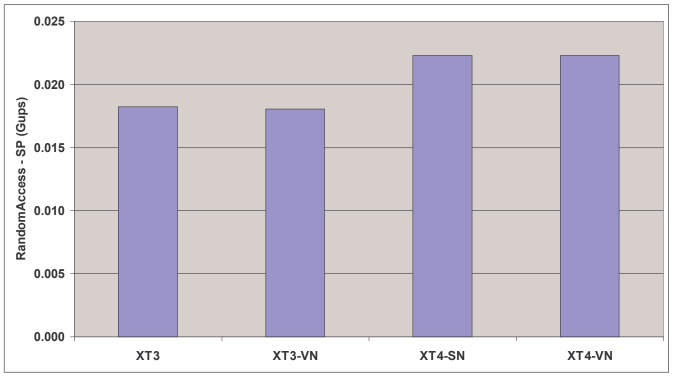


Figure 9a-b: MPI-FFT results and normalized results

4.5. Low Spatial and Low Temporal Locality

4.5.1. RandomAccess

Figure 10 shows the results of the RandomAccess benchmark for both SP and EP mode. The SP graph shows that the faster memory gives the XT4 better RandomAccess results. As with the STREAMS results, the EP results show the bottleneck of the Opteron's single memory controller.



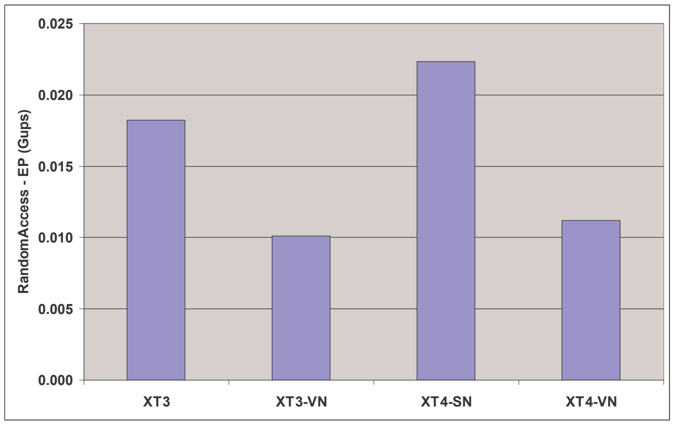
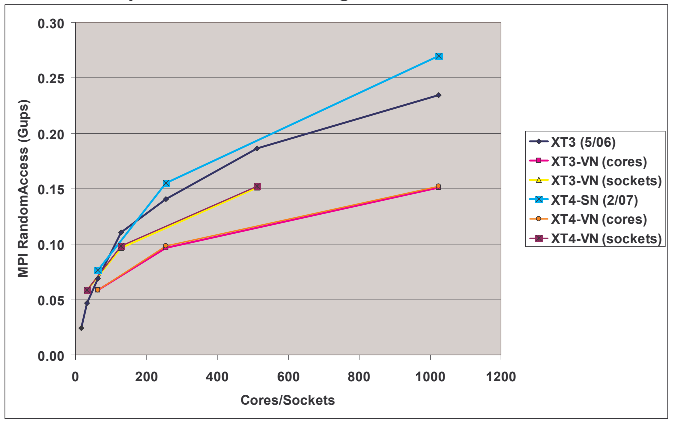


Figure 10a-b: RandomAccess SP and EP results

4.5.2. MPI-RandomAccess

The MPI-RandomAccess results are below. The graphs show a slight improvement in performance between the single-core XT3 and XT4 results, which may result from the increased injection bandwidth from the XT4 nodes or simply due to software improvements during this time. Dual-core results for XT3 and XT4 are virtually identical and universally worse than single-core results.



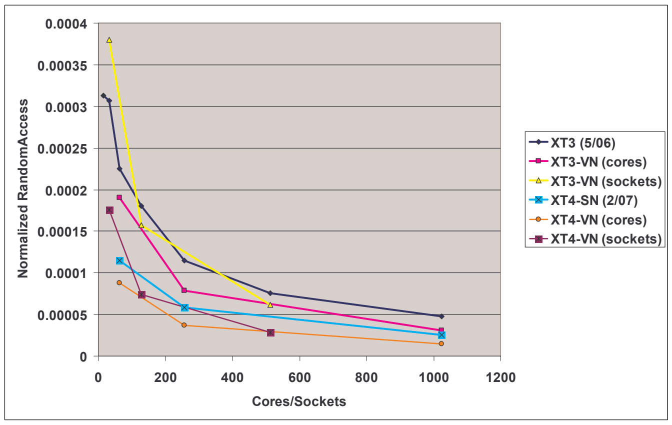


Figure 11a-b: MPI-RandomAccess results and normalized results

5. Conclusions

Several conclusions over the utility of multi-core processors arise from examining the HPCC results on the XT3 and XT4. For both EP and Global results, we note that multi-core processors deliver improved performance for those benchmarks with high temporal locality – almost a doubling over single-core results. On the other hand, those benchmarks with low temporal locality see little if any benefit from engaging a second core on a node. Though the design of the HPCC benchmark doesn't encourage direct comparison of results from differing corners of the STLD, we see that increasing or decreasing spatial locality (while holding temporal locality constant) has a very limited impact on the overall performance *per socket*, since the memory controller, channels, and DIMMS are shared. Increasing spatial locality for codes with high temporal locality, however, will have a greater effect on performance than codes with low temporal locality.

In most cases the faster memory and improved injection bandwidth gave improved results, as expected. Codes that rely heavily on memory or network bandwidth will likely see increased performance when migrating from an XT3 to an XT4 system.

6. Future Work

The Center for Computation Sciences at ORNL has a very aggressive plan to continue system upgrades and new installations over the next several years[13]. We will continue to use the HPCC benchmark suite to evaluate each of these systems and report our finding where appropriate.

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-Battelle, 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 the Innovative Computing Laboratory at the University of Tennessee for development of the HPCC Benchmarks and public results repository, Cray, Inc., and the ORNL Center for Computational Sciences (CCS) for access to the Cray XT3 and Cray XT4. The CCS is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

References

1 W. J. Camp and J. L. Tomkins, "Thor's hammer: The first version of the Red Storm MPP architecture," Proceedings of the SC 2002 Conference on High Performance Networking and Computing, Baltimore, MD, November 2002.
2 Sandia Red Storm System.
3 S. R. Alam, R. F. Barrett, M. R. Fahey, J. A. Kuehn, O. E. B. Messer, R. T. Mills, P. C. Roth, J. S. Vetter, and P. H. Worley, "An Evaluation of the ORNL Cray XT3," International Journal of High Performance Computing Applications, 2006.
4 J. S. Vetter, S. R. Alam, et al., "Early Evaluation of the Cray XT3," Proc. IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2006.
5 Cray XT3 Data Sheet, http://cray.com/downloads/Cray_XT3_Datasheet.pdf
6 Cray XT4 Data Sheet, http://cray.com/downloads/Cray_XT4_Datasheet.pdf
7 J.A. Kuehn and N.L. Wichmann, "HPCC update and analysis," Proc. Cray Users Group 2006 Annual Meeting, 2006.
8 D. Weisser, N. Nystrom et al., "Performance of applications on the Cray XT3," Proc. Cray Users Group 2006 Annual Meeting, 2006.

9 P. Luszczek, J. Dongarra, D. Koester, R. Rabenseifner, B. Lucas, J. Kepner, J. McCalpin, D. Bailey, D. Takahashi, "Introduction to the HPC Challenge Benchmark Suite," March, 2005.

10 J. Dongarra, P. Luszczek, "Introduction to the HPCChallenge Benchmark Suite," ICL Technical Report, ICL-UT-05-01, (Also appears as CS Dept. Tech Report UT-CS-05-544), 2005.

11 P. Luszczek, D. Koester, "HPC Challenge v1.x Benchmark Suite," SC|05 Tutorial-S13, Seattle, Washington, November 13, 2005.

12 High Performance Computing Challenge Benchmark Suite Website, http://icl.cs.utk.edu/hpcc/

13 Studham, R.S., Kuehn, J.A., White, J.B., Fahey, M.R., Carter, S., and Nichols, J.A., "Leadership Computing at Oak Ridge National Laboratory", Proc. Cray User Group Meeting (CUG 2005), 2005, http://www.cug.org,