WHITE PAPER

ANSYS® on Advanced SGI® Architectures*

Olivier Schreiber,† Scott Shaw,† Jeff Beisheim**

Abstract

ANSYS Mechanical™ tackles all important Normal Mode Analyses utilizing either Shared Memory Parallelism(SMP) and Distributed Memory Parallelism (DMP). The technical approaches include block Lanczos and supernode eigensolver. Efficient execution of above two paradigms and solutions requires extreme care in allocating hardware resources. The topic is even more important given hardware variety today, ranging from single node multi-core workstations through multiple nodes clusters to single image many-core systems addressing very large memory space. The paper will explore Memory, Input/Output, Communication latency and bandwidth requirements of such solutions to establish guidelines for advanced SGI computer hardware systems.

** Presented at 2010 ANSYS Regional Conferences*
† SGI Applications Engineering
*** Senior Development Engineer, ANSYS*

TABLE OF CONTENTS

1. Introduction

Various hardware such as SGI Octane III, Altix (Itanium®), Altix XE, Altix ICE and Altix UV architectures are not Zare available using most recent technologies (Figure 1) to extend computational performance. ANSYS computational technologies such as Shared Memory Parallel (SMP), Distributed Memory Parallel (DMP) and their combination (hybrid mode) can exploit this advanced hardware for normal modes analyses. The strategies employed by ANSYS and the practical importance of such analyses are demonstrated in Ref [1]. How to best use SGI hardware is described in Ref [2].

Figure 1: New hardware technologies

1.0 SGI hardware overview

Various systems comprised in SGI product line and available through SGI Cyclone™, HPC on-demand Cloud Computing (see section 1.7) were used to run the benchmarks described in section 3.1.

1.1 SGI Octane III

Scalable deskside multi-node system with GigE or Infiniband interconnects, up to 10 nodes, 120 cores with SUSE® Linux® Enterprise Server 10 SP2, SGI ProPack™ 6SP3.

Figure 2: SGI Octane III

- Dual-socket nodes of 2.93GHz six-core Xeon® X5670, 12MB cache
- Dual-socket nodes of 2.93GHz quad-core Xeon® X5570, 8MB cache
- Dual-socket nodes of 2.53GHz quad-core Xeon® E5540, 8MB cache
- RAM: 48, 72, 96GB/node 1066, 1333MHz DDR3 ECC

1.2 SGI Altix XE 1300 cluster

Highly scalable and configurable rack-mounted multi-node system with GigE and/or Infiniband interconnects.

Figure 3: SGI Altix XE 1300 cluster

- SGI XE250 or XE270 Administrative/NFS Server node
- SGI XE340 Dual-socket compute nodes of 2.93GHz six core Xeon X5670 12MB Cache
- SGI XE340 Dual-socket compute nodes of 2.93GHz quad core Xeon X5570 8MB Cache
- SGI XE250 Dual-socket compute nodes of 3.0GHz quad core Xeon X5472 12MB Cache, 1600MHz Front Size Bus.
- 32GB 1333MHz RAM (max 96GB/node)
- SUSE Linux Enterprise Server 11 SP2, SGI ProPack 6SP3
- SGI Foundation Software 1SP5
- Infiniband ConnectX QDR PCIe Host Card Adapters
- Integrated GigE dual port Network Interface Cards

SGI Altix XE1300 cluster with dual Ethernet and Infiniband switch is illustrated in Figure 4.

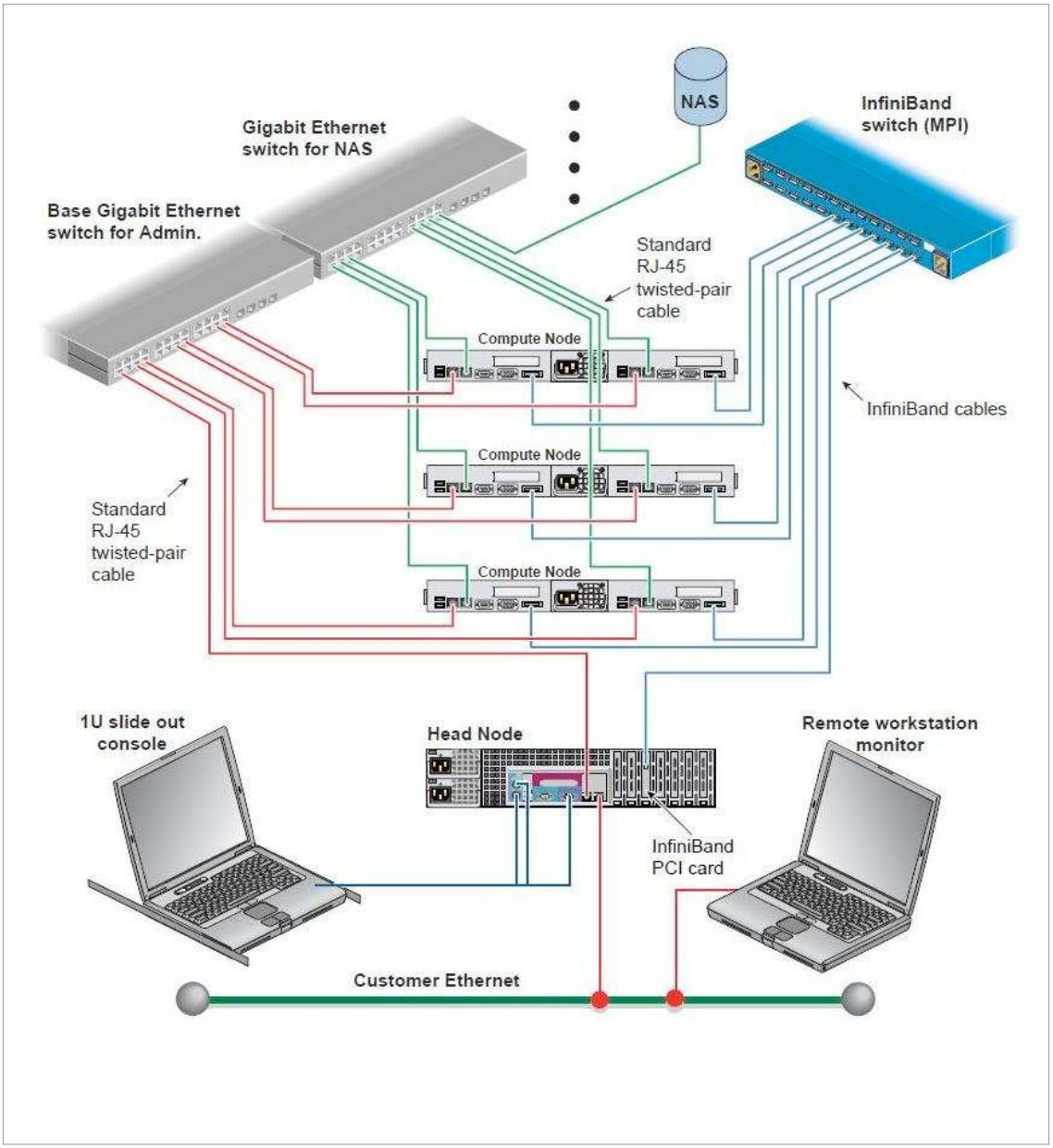

Figure 4: Dual Ethernet and Infiniband switch cluster configuration example

1.3 SGI Altix ICE cluster

Highly scalable, diskless, integrated cable-free Infiniband interconnect rack mounted multi-node system. (Figure 5).

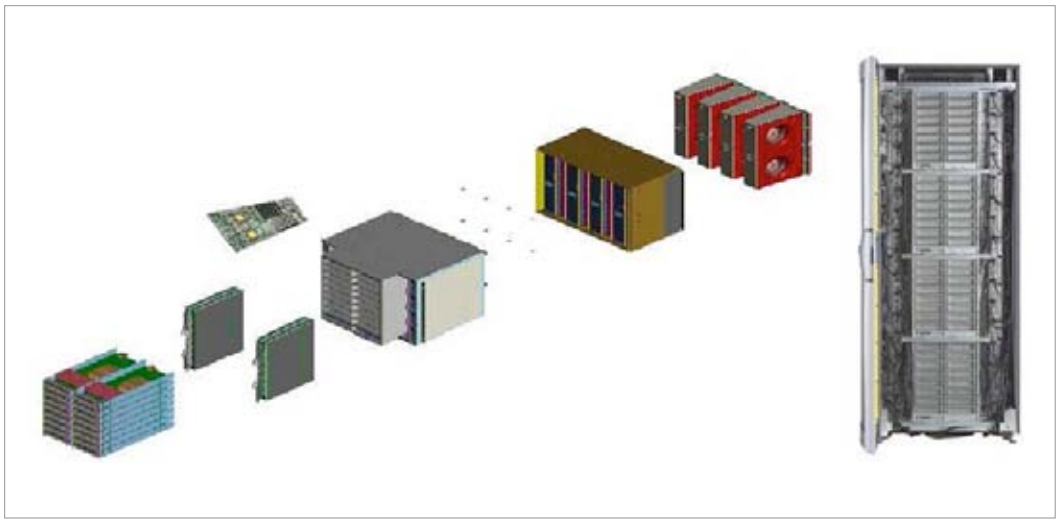

Figure 5: SGI Altix ICE cluster and IRU

- Intel Xeon 5500 2.93GHz quad-core or 5600 3.46 GHz six-core
- Two single-port ConnectX-2 IB HCA
- 12 DDR3 1066MHz or 1333MHz ECC DIMM slots per blade
- SGI Tempo management tools
- SGI ProPack™ for Linux®
- Altair® PBS Professional™ workload manager

1.4 SGI Altix 450 and Altix 4700 (SMP)

Highly scalable Shared Memory Parallel (SMP) system. Allows flexibility of sizing memory allocated to a job independent of the core allocation. In a multi-user, heterogeneous workload environment, this prevents jobs requiring a large amount of memory to be starved for cores.

Figure 6: SGI Altix 4700, 450 SMP

- SGI Altix 4700 128 1.669GHz dual core Itanium 9150M 24MB cache processors, 512GB RAM NUMAlink® 4

1.5 SGI Altix UV 10, UV 100, UV 1000 SMP

Highly scalable latest generation x86-based Shared Memory Parallel system. Affords the same flexibility as the architecture of 1.4.

Figure 7: SGI Altix UV 10, UV 100, UV 1000 SMP

- Intel® Xeon® Processor X7542 2.66 GHz (12 cores/blade)
- 192GB RAM
- NUMAlink® 5
- SGI Foundation Software, SGI ProPack 7

1.6 Altix XE, Altix ICE are Intel Cluster Ready Certified

Altix XE, Altix ICE are Intel® Cluster Ready Certified (figure 8).

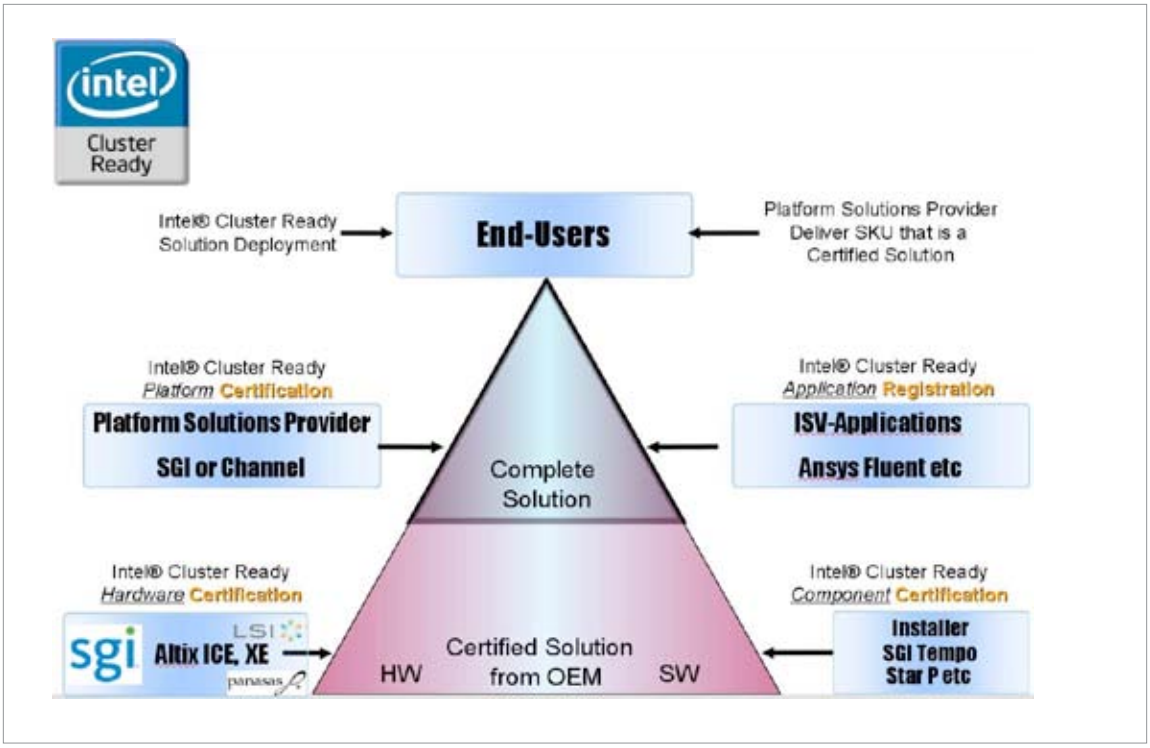

Figure 8: Intel Cluster Ready Certification

1.7 Cloud access to benchmark systems: SGI Cyclone™

SGI offers Cyclone, HPC on-demand computing resources of all SGI advanced architectures aforementioned (Figure 9). There are two service models in Cyclone: Software as a Service (SaaS) and Infrastructure as a Service (IaaS) (figure 10). With SaaS, Cyclone customers can significantly reduce time to results by accessing leading-edge open source applications and best-of-breed commercial software platforms from top Independent Software Vendors (ISV's).

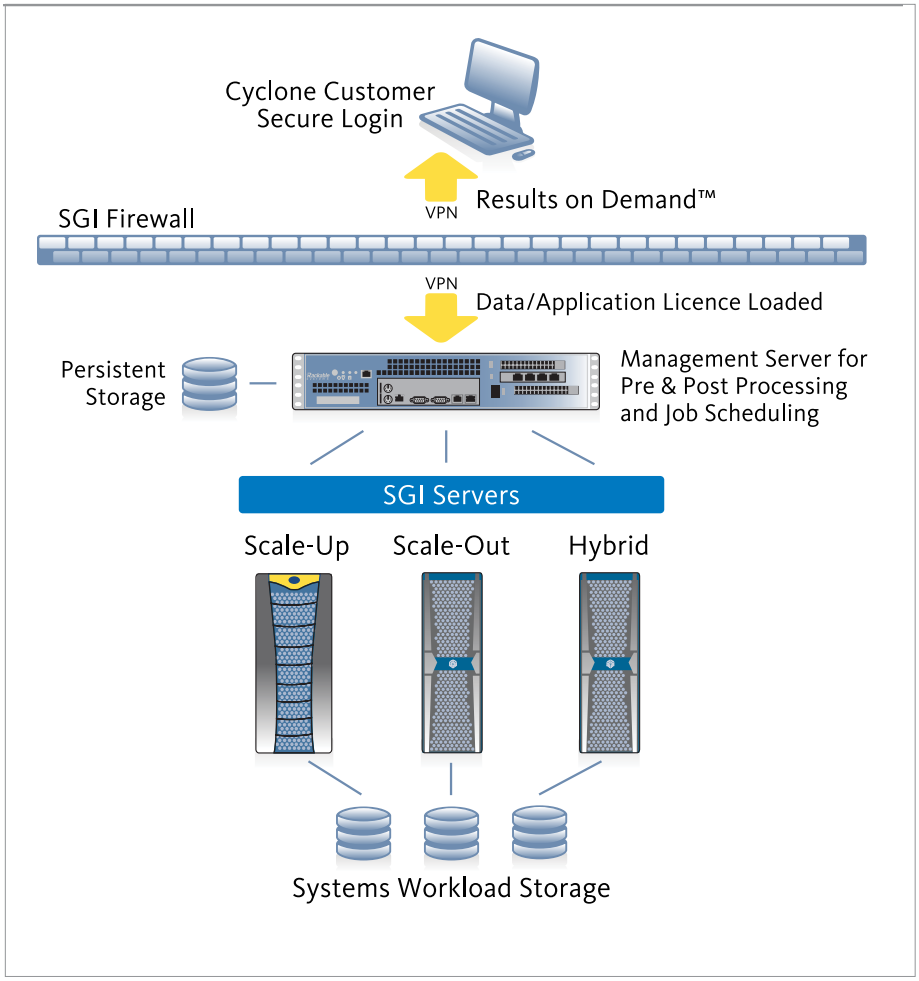

Figure 9: SGI Cyclone – HPC on-demand Cloud Computing

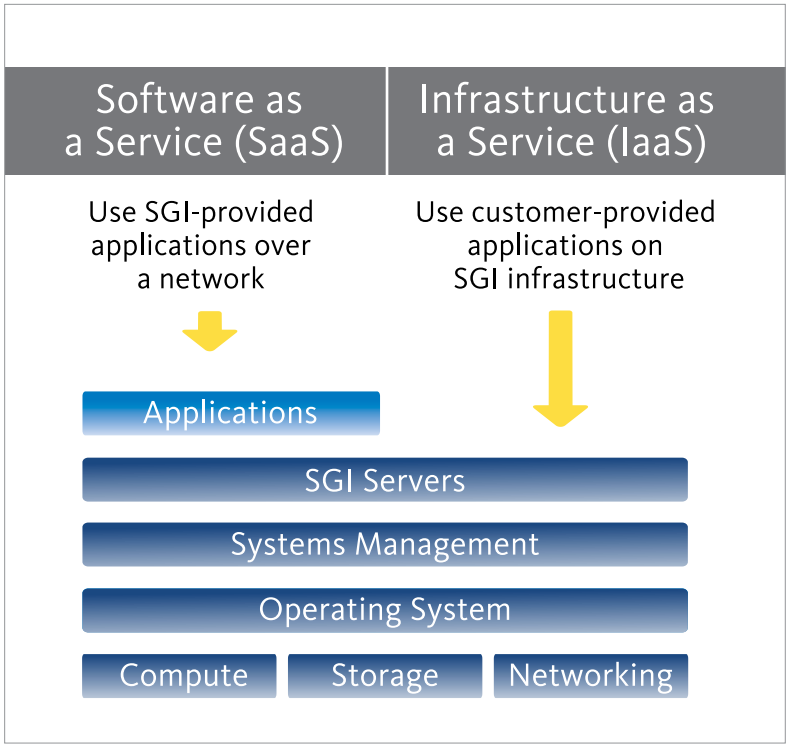

Figure 10: SGI Cyclone – Service Models

The physical elements of Cyclone feature:

- Pre-configured, pre-certified applications and tools
- High speeds Scale-Up, Scale-Out and Hybrid(GPU/Graphics) platforms
- High speed processors
- High speed networking (NUMAlink,InfiniBand)
- Non-virtualized environments
- Dedicated management node for security
- SSH or Virtual Private Network(VPN) access
- From scratch storage to long-term storage
- Data exchange service
- 24 x 7 x 365 monitoring and support
- Dedicated customer accounts

2.0 ANSYS Overview

2.1 Parallel Processing Capabilities of ANSYS

Parallelism in computer systems exists in two paradigms:

- Distributed Memory Parallelism (DMP) uses MPI Application Programming Interface (API) focused on Finite Element computations or physical domain decomposition. The resulting reduced size geometric partitions have lesser processing resource requirements, resulting in increased efficiency, but the size of the common boundary should be kept minimal to decrease inter-process communication.

- Shared Memory Parallelism (SMP) uses OpenMP™ (Open Multi-Processing) Application Programming Interface focused on numerical modules.

These two paradigms can simultaneously map themselves on two different system hardware levels:

- Inter-node or cluster parallelism (memory local to each node)–DMP only.
- Intra-node or multi-core parallelism (memory shared by all cores of each node).

The hybrid approach provides increased performance yields and additional possibilities of memory utilization by running SMP on intra-node network simultaneously with DMP on inter-node and/or intra-node network. In this study parallelism is exercised through these two paradigms. For cluster computing with multi-core processors, DMP is also run for both inter-node and intra-node.

2.2 Distributed parallel capabilities in ANSYS

A distributed parallel capability is a technique that partitions the geometry model and distributes the partitions efficiently to each processor cores. This technique partitions the geometry model and distributes the partitions among the cores on each processor sockets. The partitioning is executed either in the Finite Element model or at the Matrix Level. The computations in the geometry partitions are dependent through their common boundaries. Care must be taken to minimize the boundary sizes between partitions to decrease inter-process communication. Load balancing is just as important as minimizing the communication costs. Workload for each MPI process is balanced so that each process does roughly the same number of computations during the solution and therefore finishes at the same time. Allocation of the total number of MPI processes over the nodes may be made by filling up each node's cores designated for processing first ('rank' allocation) or by distributing them in round-robin fashion across all the nodes.

2.3 Distributed execution control

2.3.1 Submittal procedure

Submittal procedure must ensure:

- Placement of processes and threads across nodes and sockets within nodes
- Control of process memory allocation to stay within node capacity
- Use of adequate scratch files across nodes or network

Batch schedulers/resource managers dispatch jobs from a front-end login node to be executed on one or more compute nodes. To achieve the best runtime in a batch environment disk access to input and output files should be placed on the high performance filesystem closest to the compute node. The high performance filesystem could be in-memory filesystem (/dev/shm), a Direct (DAS) or Network (NAS)Attached Storage filesystem. In diskless computing environments in-memory filesystem or network attached storage are the only options. This filesystem nomenclature is illustrated in Figure 11.

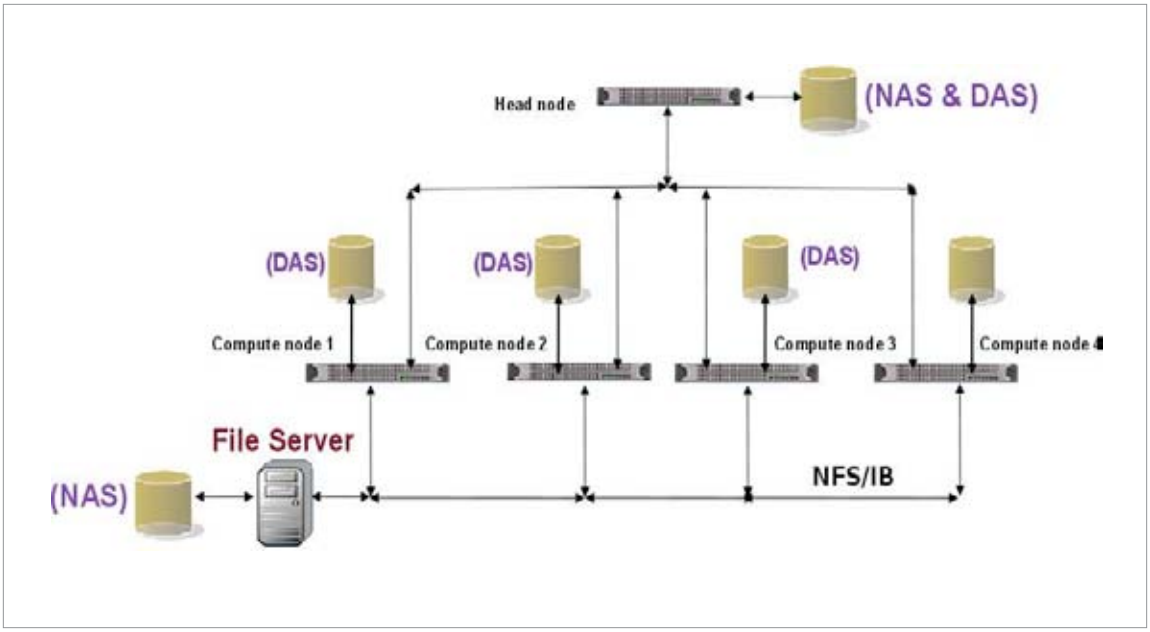

Figure 11: Example filesystems for scratch space

Following is the synoptic of a job submission script.

1. Change directory to the local scratch directory on the first compute node allocated by the batch scheduler.
2. Copy all input files over to this directory.
3. Create parallel local scratch directories on the other compute nodes allocated by the batch scheduler.
4. Launch application on the first compute node. The executable may itself carry out propagation and collection of various files between launch and the other nodes at start and end of the main analysis execution.

2.3.2 Submittal command

The following keywords were used for the ANSYS execution command:

- -dis: Enables Distributed Memory Parallelism
- -np argument: Number of distributed processes in the solution and number of OpenMP threads (SMP) during all pre/post operations that are parallelized.

2.3.3 Software Environment

- ANSYS Mechanical Release 13.0
- Compilers: Fortran: Intel Fortran Compiler 11.1 for EM64T-based applications
- MPI: HP-MPI, Intel MPI, MPT 2.01

2.3.4 Application Tuning

An I/O dominated application relies on a high performance filesystem for best performance. The I/O subsystem either being DAS or NAS needs to be configured to support fast I/O sequential transactions as illustrated in

section 2.3.1 by figure 11. In cluster computing environments with a common scratch location, such as a Network Attached Filesystem, isolating application MPI communications and NFS traffic will provide the best NFS I/O throughput for scratch files.

Another factor regarding performance is memory per core. Having more memory per core in most cases will increase performance since more memory can be allocated for the analysis and any unallocated memory will be used by the Linux kernel buffer cache. SGI's FFIO is a link-less library (which does not need to be linked to the application) bundled with SGI ProPack which implements user defined I/O buffer cache to avoid memory buffer cache thrashing when running multiple I/O intensive jobs or processes in Shared Memory Parallel systems or cluster computing environments using DAS or NAS storage subsystems. FFIO isolates user page caches so jobs or processes do not contend for Linux Kernel page cache. Hence, FFIO minimizes the number of system calls and I/O operations to and from the storage subsystem and improves performance for large and I/O intensive jobs. (Ref [2], Chapter7 Flexible File I/O).

3.0 Results

3.1 Benchmark example

The V12.1 benchmarks are 10 engineering problems:

- 1 JCG iterative solver benchmark: V12cg-1
- 2 PCG iterative solver benchmarks: V12cg-2,3
- 2 Lanczos eigensolver benchmarks: V12ln-1,2
- 5 sparse direct solver benchmarks: V12sp-1-5

Figure 12: V12.1 standard benchmarks

All can be run with SMP or DMP except for V12In-2 (only SMP). So the total number of benchmarks is 19.

3.2 Benchmark Results

3.2.1 SMP scaling of benchmarks

Figure 13 shows scaling within one 8-core X5570/2.53GHz node in a configuration similar to 1.1 or 1.2 for the standard benchmarks run in SMP (OpenMP) mode parallelism. Elapsed time decreases as shown by abscissa spanning values of 1, 2, 4, 8 cores with a minimum achieved for the maximum number of physical cores available. Beyond this value, runs with 16 virtual cores using hyper-threading mode prove not to be beneficial, as later illustrated in sub-section 3.2.3. (Hyper-Threading (HT) is a feature which can increase performance for multi-threaded or multi-process applications. It allows a user to run twice the number of OpenMP threads or MPI processes than available physical cores per node.)

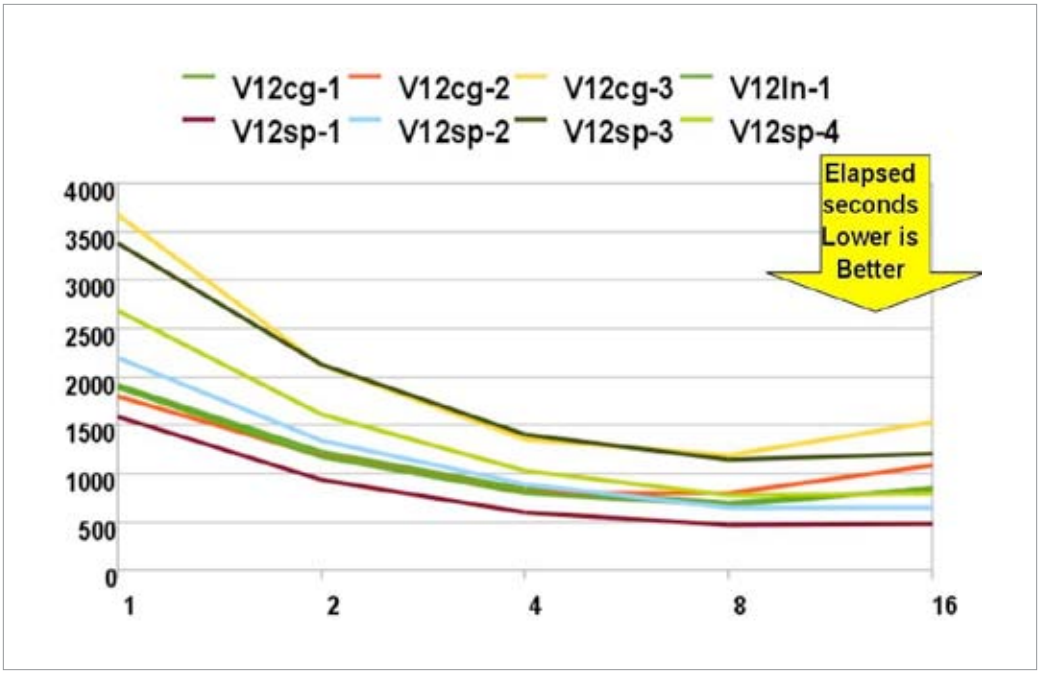

Figure 13: SMP Scaling

3.2.2 DMP scaling of benchmarks

Application parallel performance is mitigated by the performance of the network interconnect used for DMP. In figure 14 we demonstrate application scaling by fully subscribing all cores within the node and using multiple compute nodes (8 cores/node) with GbE and QDR Infiniband interconnects. The X axis of the chart is the total cores used to demonstrate application scaling. As the chart illustrates DMP scaling performance with GbE is impacted at node counts greater then two nodes, while QDR Infiniband continues to scale reducing the elapsed time of each test case. QDR Infiniband is a low latency high bandwidth interconnect and out performs GbE in every test. Further tests where conducted with GbE and Infiniband in section 3.2.7.

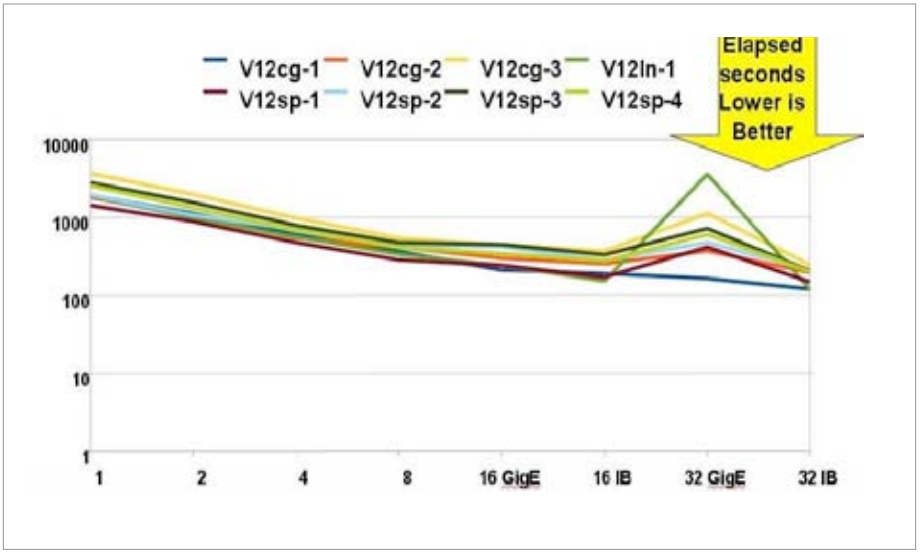

Figure 14: DMP scaling

3.2.3 Evaluating hyper-threading

Figure 15 shows hyper-threading elapsed time reduction of 2% for the only one case showing a benefit, cg-1 in DMP mode by over-subscribing 16 MPI processes on 8 cores of 1 node. This effect is shown for several frequencies in a configuration similar to that of 1.1 or 1.2.

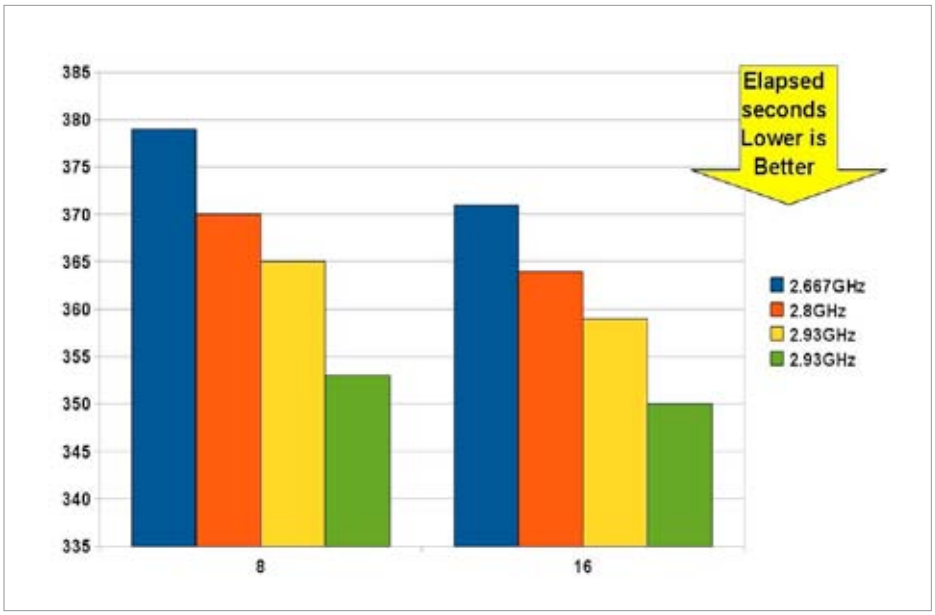

Figure 15: Comparison 8 and 16 MPI processes on 8 cores for cg-1 case only

3.2.4 Effect of core frequency

Figure 16 shows the effect of increased core operating frequencies on the SMP benchmarks for 8 OpenMP threads running on 8 cores by plotting elapsed times ratios referenced to the slowest frequency. It is clear that the percentage increases of 1.05, 1.11 and 1.16 do not translate into corresponding decreases in elapsed times– reflecting the fact that the cores are not workload saturated due to communication costs associated with shared memory parallelism.

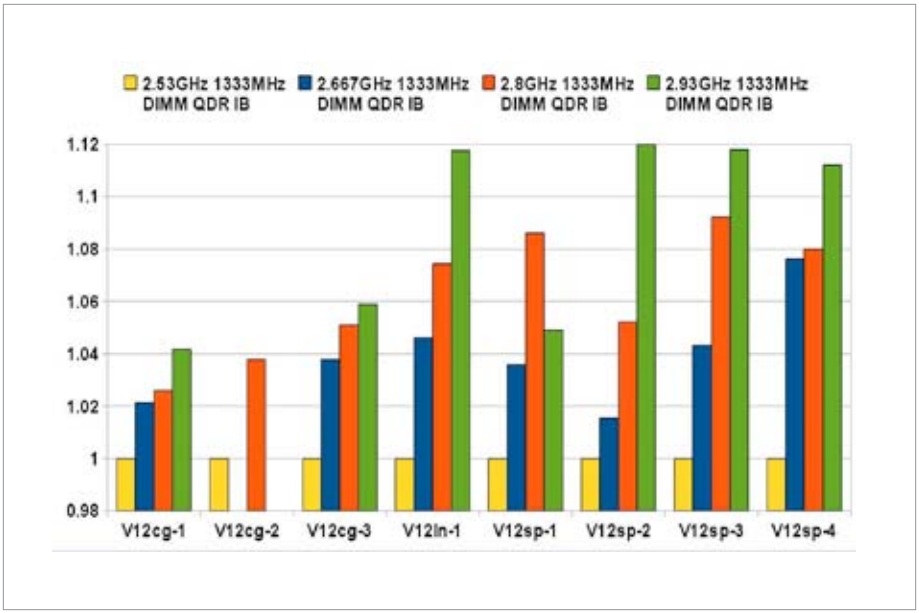

Figure 16: Effect of core frequency on SMP benchmarks

Figure 17 shows the same effect of increased core operating frequencies on the DMP benchmarks for 8 MPI processes running on 8 cores by plotting elapsed times ratios referenced to the slowest frequency. The percentage increases of 1.05, 1.11 and 1.16 translate better into corresponding decreases in elapsed times because of the better efficiency of DMP compared to SMP processing. The configuration is similar to 1.1 or 1.2.

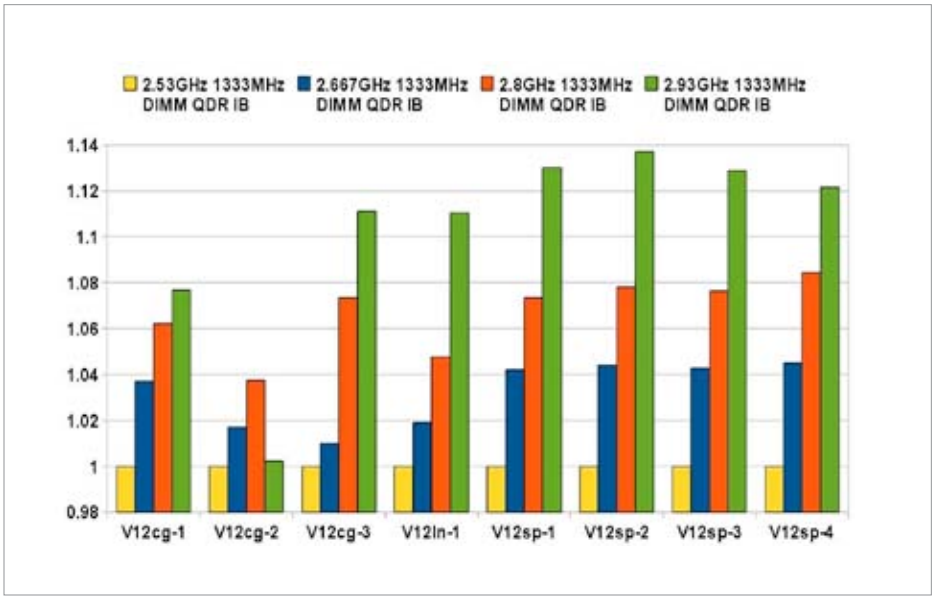

Figure 17: frequency DMP

3.2.5 Effects of Intel® Turbo Boost Technology

Turbo Boost is a feature first introduced in the Intel® Xeon® 5500 series, for increasing performance by raising the core operating frequency within controlled limits depending on thermal envelope. The mode of activation is a function of how many cores are active at a given moment as may be the case when OpenMP threads or MPI processes are idle under their running parent. For example, for a base frequency of 2.93GHz, when 1-2 cores active, their running frequencies will be throttled up to 3.3GHz, but with 3-4 cores active only to 3.2GHz. For most computations, utilizing Turbo Boost technology can result in improved runtimes, but overall benefit maybe mitigated by the presence of other performance bottlenecks than pure arithmetic processing. Figure 18 plots the ratios between elapsed times with Turbo Boost OFF versus ON runs for values of cores used of 1, 2, 4 and 8 out of 8 physical cores of 1 node using SMP mode for the standard benchmarks.

Turbo Boost improves performance for low numbers of cores used and in some case up to the ratio of the maximum frequency over nominal value which is 13%. As more cores are used, Turbo Boost is not as effective because all the cores are kept busy all the time whereas when less cores are used, Turbo Boost can accelerate the active cores more consistently.

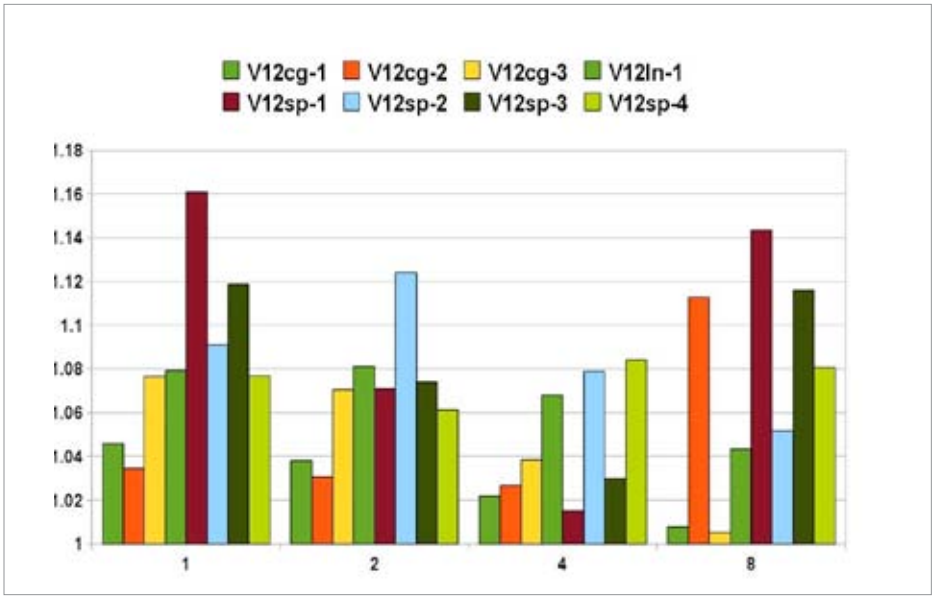

Figure 18: Effect of turbo mode on SMP benchmarks

Similarly, figure 19 plots the ratios between elapsed times with Turbo Boost OFF versus ON runs for values of cores used of 1, 2, 4 and 8 out of 8 physical cores of 1 node using DMP mode for the standard benchmarks. On average, in this case, Turbo Boost is not as effective because more cores are kept busy all the time due to the greater efficiency of DMP processing. The configuration is similar to 1.1 or 1.2.

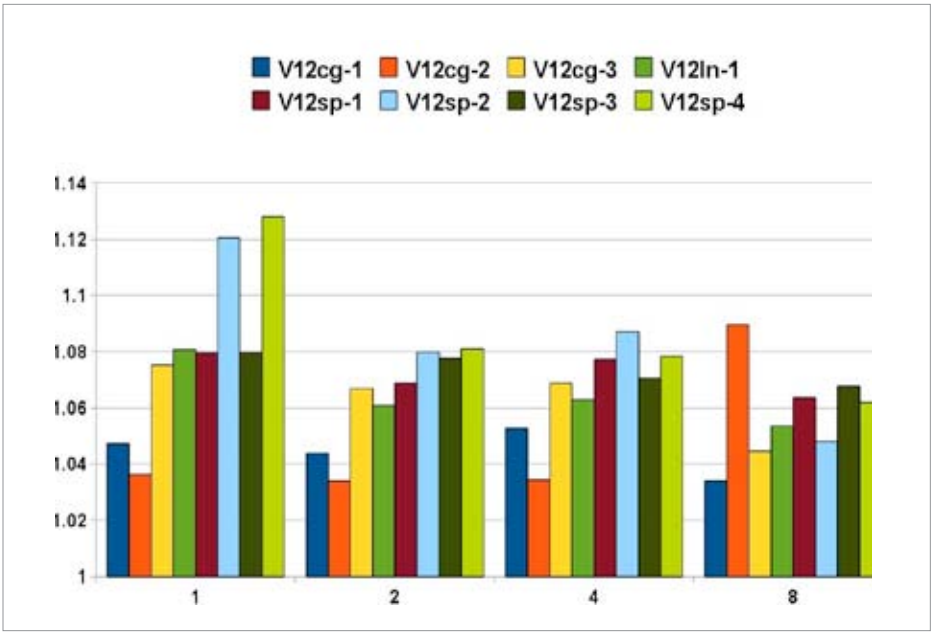

Figure 19: Effect of turbo mode on DMP benchmarks

3.2.6 Effect of DIMM speed

Figure 20 plots the ratios between elapsed of DIMM's with 1066 MHz versus 1333MHz speeds for the standard benchmarks using SMP mode and 8 threads per 8 core-nodes. The ratio is on average 5% which is substantially lower than the 25% ratio of the memory ratings.

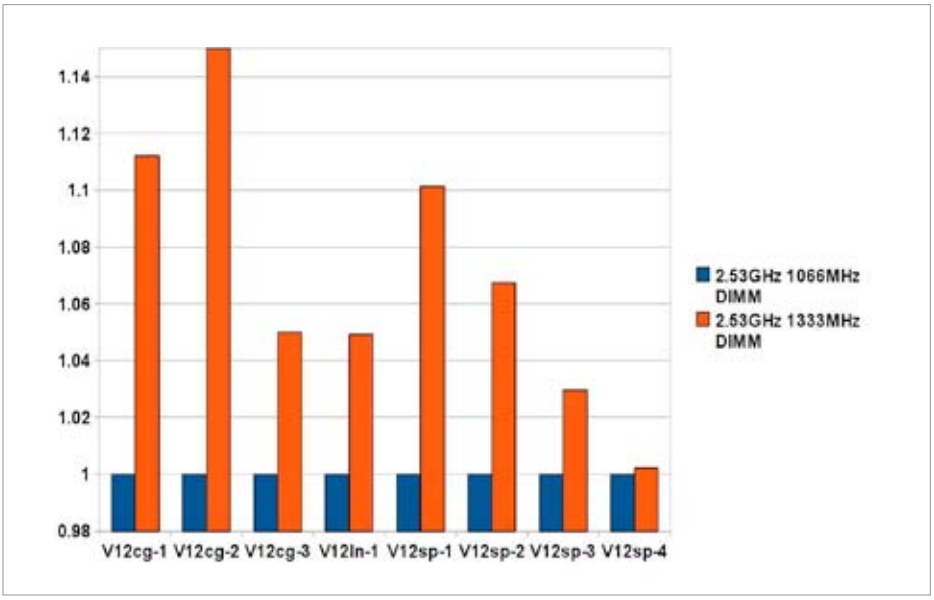

Figure 20: DIMM speed effect with 8 SMP threads

On average, DIMM speed benefits SMP mode more than DMP mode because of the dependency of OpenMP computations on memory latency and bandwidth as seen in comparison with Figure 21 which plots the ratios between elapsed of DIMM's with 1066MHz vs 1333MHz speeds for the standard benchmarks using DMP mode and 8 MPI processes per 8 core-nodes. The configuration is similar to 1.1 or 1.2.

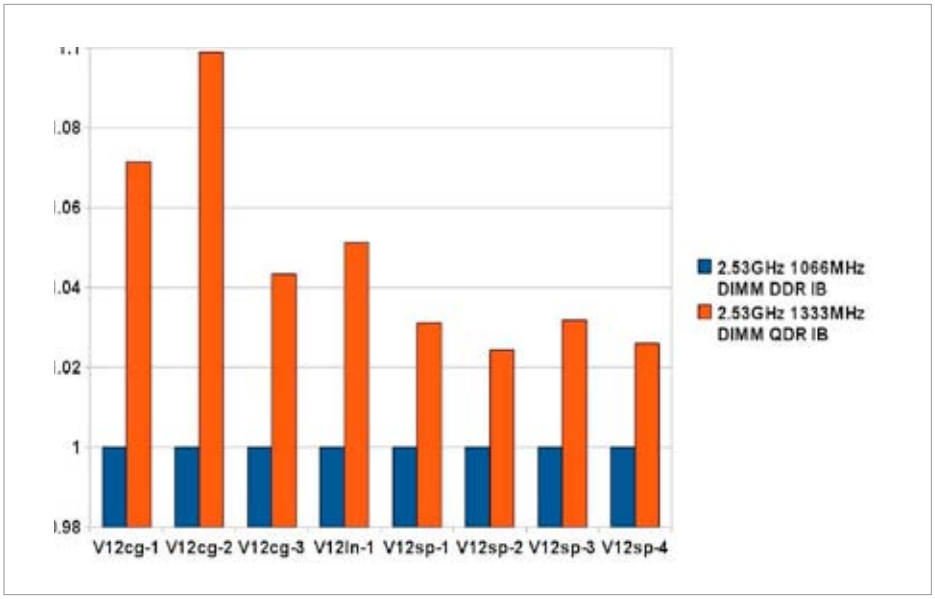

Figure 21: DIMM speed effect with 8 DMP processes

3.2.7 Effect of interconnect on DMP benchmarks

Figure 22 plots elapsed times obtained for the DMP benchmarks run with 16 MPI processes across 2 nodes connected by either GigE or QDR Infiniband interconnects. On some datasets, the performance differences can vary from 12% to 50%. The configuration is similar to 1.1 or 1.2.

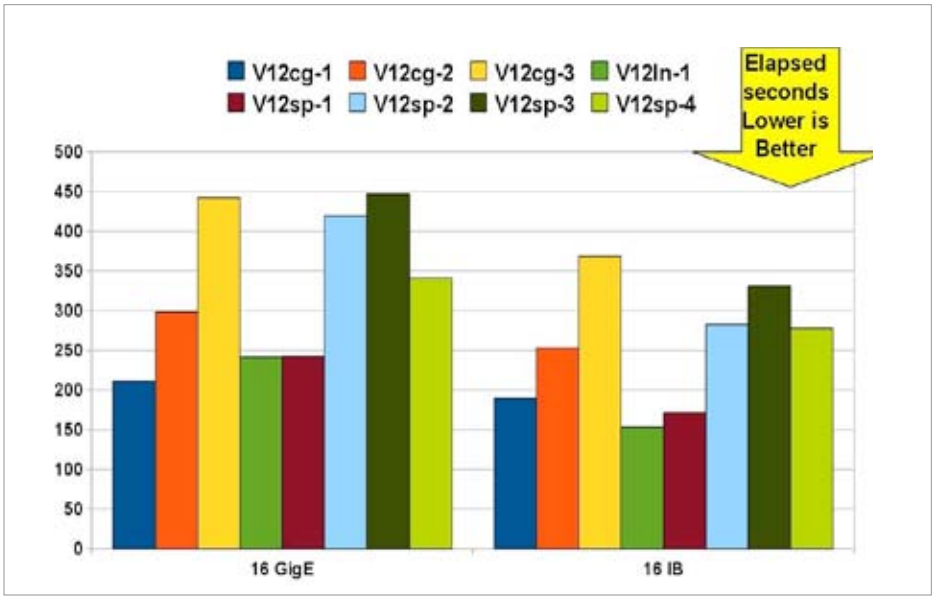

Figure 22: Effect of interconnect on DMP benchmarks

3.2.8 Effect of MPI flavor on DMP benchmarks

An MPI library capability to bind an MPI rank to a processor core is key to control performance because of the multiple node/socket/core environment. From [3], '3.1.2 Computation cost-effects of CPU affinity and core placement [...]HP-MPI currently provides CPU-affinity and core-placement capabilities to bind an MPI rank to a core in the processor from which the MPI rank is issued. Children threads, including SMP threads, can also be bound to a core in the same processor, but not to a different processor; additionally, core placement for SMP threads is by system default and cannot be explicitly controlled by users.[...]'.

In contrast, MPT, through the omplace command uniquely provides convenient placement of hybrid MPI/OpenMP processes and threads within each node. This MPI library is linklessly available through the PerfBoost facility bundled with SGI ProPack. PerfBoost provides a Platform-MPI, IntelMPI, OpenMPI, HP-MPI ABI-compatible interface to MPT MPI.

Figure 23 shows SGI MPT MPI performing consistently on par or better than Intel MPI or HP-MPI on a single 12 core node on configuration similar to 1.1 or 1.2. Geometric mean values are: MPT: 352, IntelMPI: 371, HP-MPI: 433 seconds.

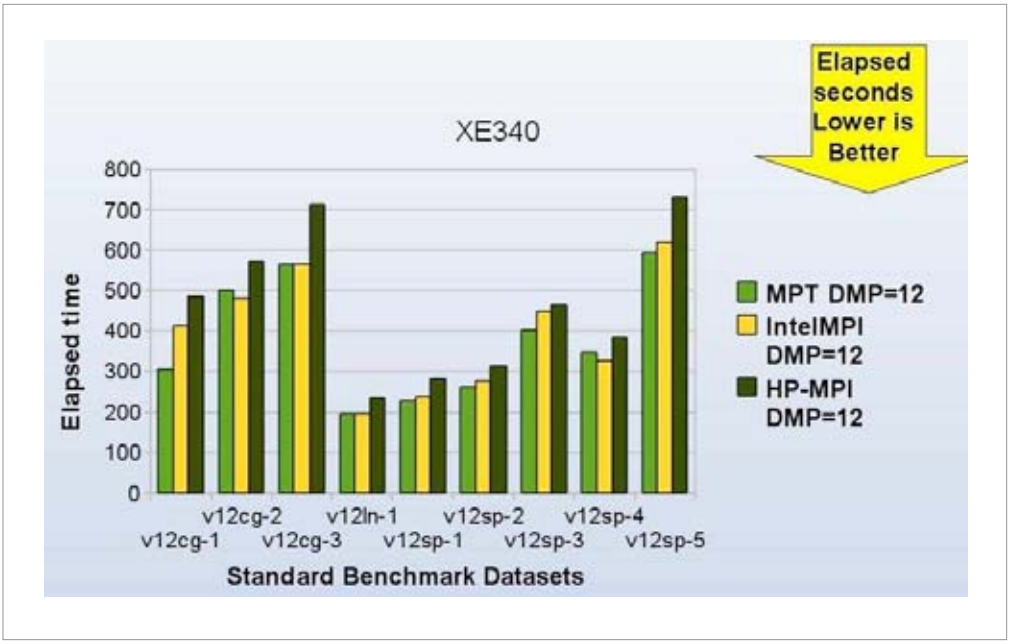

Figure 23: MPT, HP-MPI, Intel MPI performance on XE340 with 12 MPI processes

The advantage is also displayed in Figure 24 for the case of the SGI Altix UV architecture (1.5). Geometric mean values are: MPT: 375, HP-MPI: 388, IntelMPI: 457 seconds.

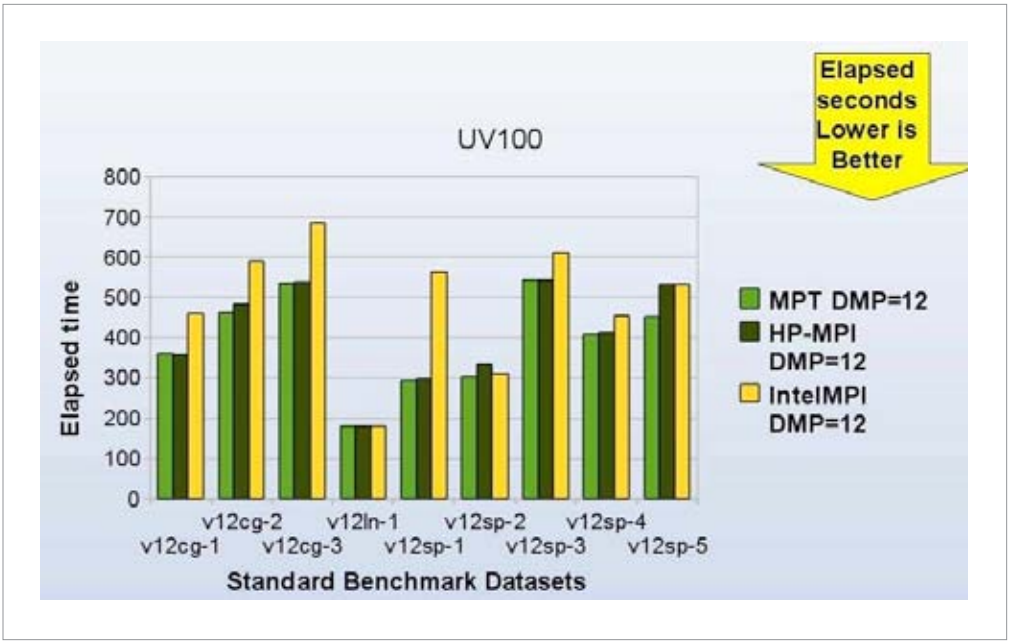

Figure 24: MPT, HP-MPI, Intel MPI performance on Altix UV 100 with 12 MPI processes

Because SGI MPT is able to take advantage of the Altix UV architecture (1.5), for 24 MPI processes, its use becomes even indispensable as shown by Figure 25. Geometric mean values are: MPT: 277, HP-MPI: 324, IntelMPI: 354 seconds.

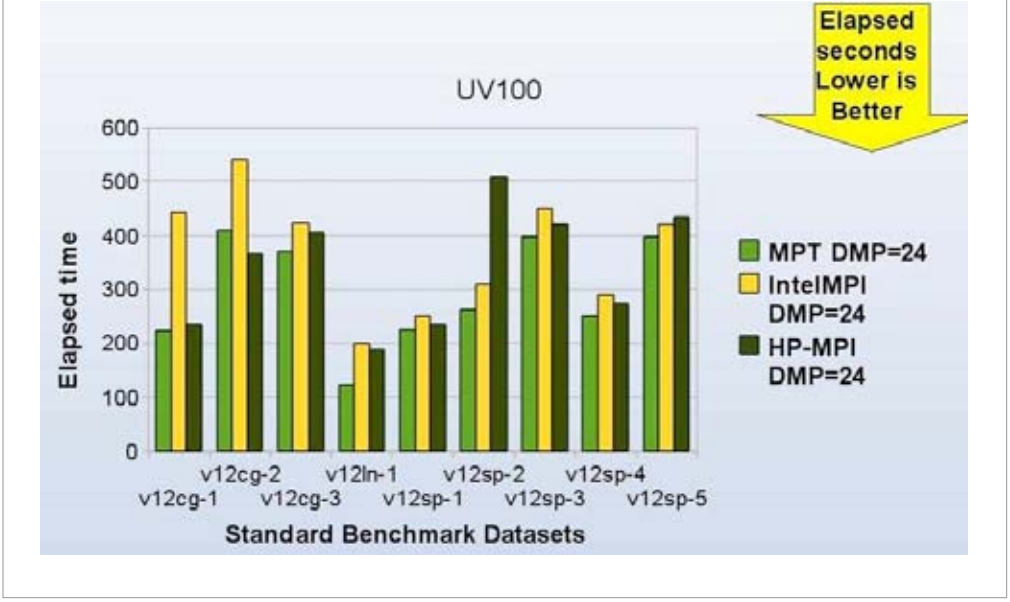

Figure 25: MPT, HP-MPI, Intel MPI performance on Altix UV 100 with 24 MPI processes

3.2.9 Trade-offs between Altix, Altix XE and Altix UV

Figure 26 show how for single core processing, Xeon-based architectures hold a certain advantage due to higher clock frequency versus Itanium-based architecture from 1.4.

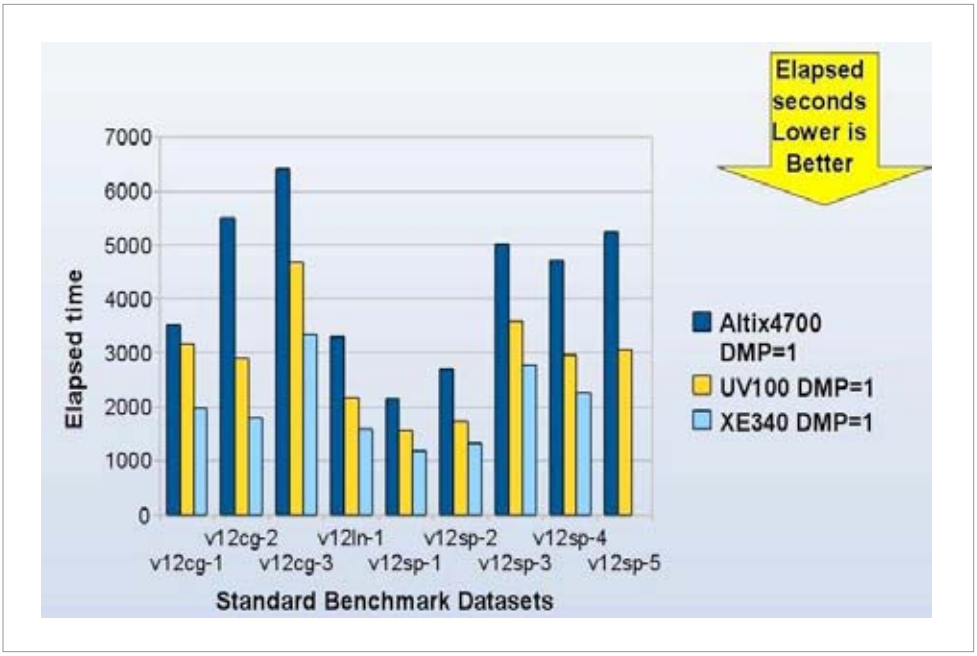

Figure 26: Altix (Itanium), Altix XE and Altix UV (Xeon) for serial processing

However, Figure 27 shows that the advantage of Altix XE over UV is variable across benchmarks when 12 MPI processes are run.

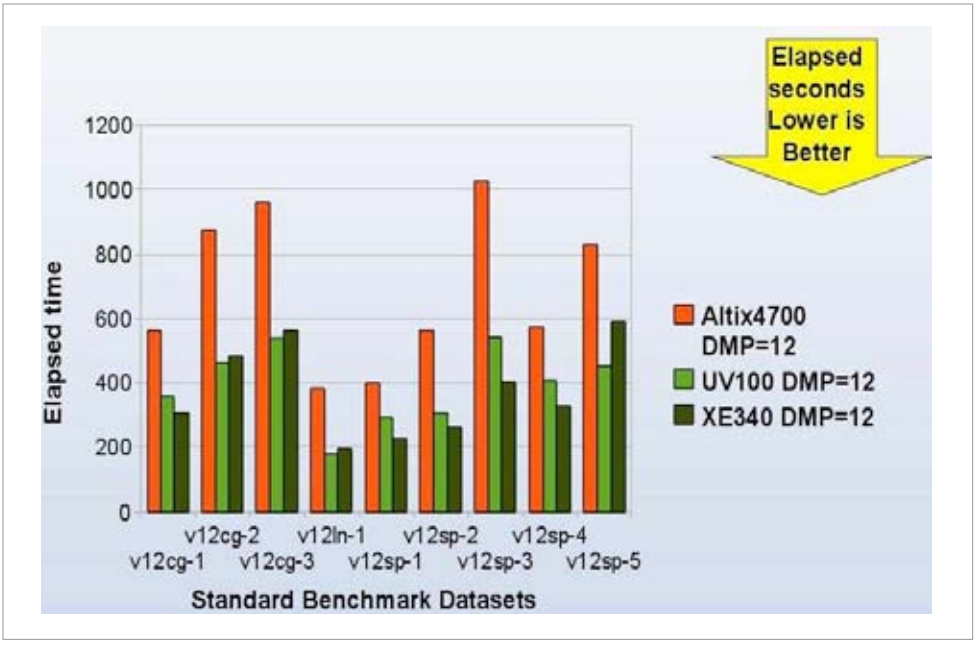

Figure 27: Altix (Itanium), Altix XE and Altix UV (Xeon) for 12 MPI processes

Figure 28 shows this advantage further mitigated with 24 MPI processes leaving the flexibility of UV architecture as detailed in 1.5 intact.

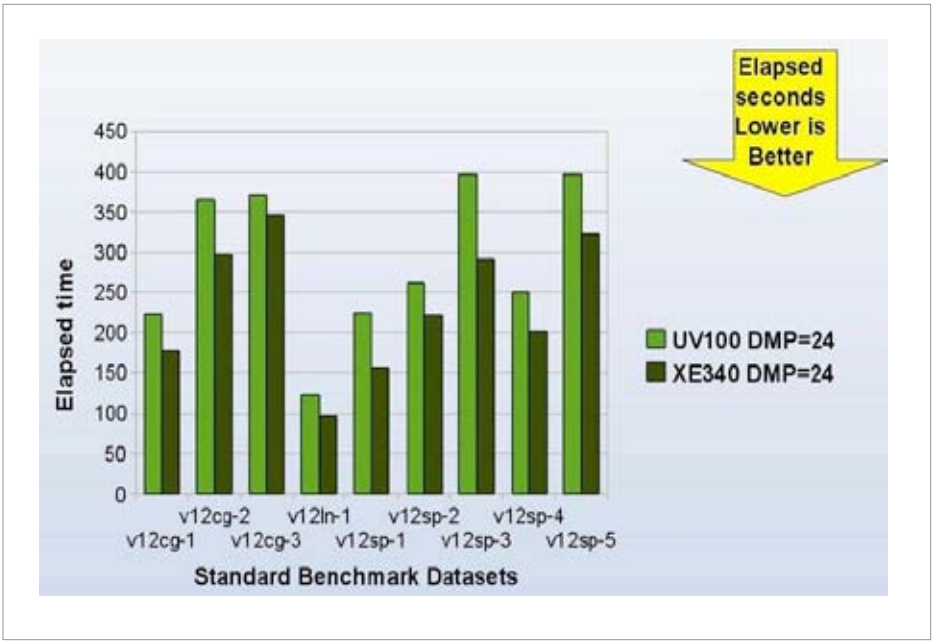

Figure 28: Altix (Itanium), Altix XE and Altix UV (Xeon) for 24 MPI processes

Conclusions

This study showed how interconnect, core frequency, Turbo Boost, hyper-threading, memory speed, filesystem effects on performance can be gauged for a given dataset, in particular one observed:

- Good scaling for SMP and DMP beyond one node/16 cores.
- Effect of frequency is partial as application not CPU limited.
- Turbo is effective generally at 10%
- Hyper threading benefits only cg-1 DMP.
- DIMM speed effect is5 %, less than nominal 25% of 1333MHz/ 1066MHz.
- Infiniband can be twice as fast as GigE interconnect.
- SGI MPT (through PerfBoost) compares favorably to other MPI's in all cases.

All these effects are definitely dependent on the dataset and solution methods used. Allocating procurement to the right mix of resources should therefore be tailored to the range of datasets envisaged. Moreover, the metric to minimize could be one of many such as turnaround time or throughput or cost–itself comprised of equipment, licenses, energy, facilities and services.

Attributions

References

[1] White Paper, “Obtaining Optimal Performance in ANSYS 11”. http://tinyurl.com/OptimalPerformanceANSYS11-pd, December 2007.

[2] SGI. SGI Developer’s Guide. Silicon Graphics International, Fremont, California, 2009.

[3] Yih-Yih Lin and Jason Wang. “Performance of the Hybrid LS-DYNA on Crash Simulation with the Multicore Architecture”. In 7th European LS-DYNA Conference, 2009.

Corporate Office
46600 Landing Parkway
Fremont, CA 94538
tel 510.933.8300
fax 408.321.0293
www.sgi.com

North America +1 800.800.7441
Latin America +55 11.5185.2860
Europe +44 118.912.7500
Asia Pacific +61 2.9448.1463