TECH GUIDE

SGI® Technology Guide For ANSYS® Fluent™ Analysts

Author

By Spencer Swift[†], Tony DeVarco[††]

Abstract

ANSYS Fluent software contains the broad physical modeling capabilities needed to model flow, turbulence, heat transfer, and reactions for industrial applications ranging from air flow over an aircraft wing to combustion in a furnace, from blood flow to semiconductor manufacturing, and from clean room design to wastewater treatment plants. Special models that give the software the ability to model in-cylinder combustion, aero acoustics, turbo machinery, and multiphase systems have served to broaden its reach. ANSYS Fluent solutions provide volume parallel capabilities to support multiple users, with complete flexibility to deploy and use the software wherever there are distributed resources.

Nevertheless, execution of 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 clusters with multiple nodes to single image many-core systems addressing very large memory space. This paper will explore MPI performance, core frequency, choice of a network topology, memory speed and use of hyper-threading of such solutions to establish guidelines for running ANSYS Fluent on advanced SGI computer hardware systems.

[†] Senior Application Analyst, SGI Applications Engineering
[††] Director of SGI Virtual Product Development Solutions

TABLE OF CONTENTS

1.0 About SGI Systems

SGI systems used to perform the benchmarks outlined in this paper include the SGI Rackable® standard depth cluster; SGI® ICE™ X integrated blade cluster and the SGI® UV™ 2000 shared memory system. They are the same servers used to solve some of the world's most difficult computing challenges.

1.1 SGI Rackable® Standard-Depth Cluster

SGI Rackable standard-depth, rackmount C2112-4GP3 2U enclosure supports four nodes and up to 4TB of memory in 64 slots (16 slots per server). It also supports up to 144 cores per 2U with support of FDR Infiniband® fourteen-core Intel® Xeon® processor E5-2600 v3 series and 2133 MHz DDR4 memory running SUSE® Linux® Enterprise Server or Red Hat® Enterprise Linux for a reduced TCO (Figure 1).

SGI Rackable Configurations used in this paper:

Benchmark System

- SGI Rackable C2112-4GP3
- Intel® Xeon® 14-core 2.6 GHz E5-2697v3
- IB FDR Interconnect
- 4.5 GB of Memory/core Memory Speed 2133 MHz
- Altair® PBS Professional Batch Scheduler v12
- SLES or RHEL with latest SGI Performance Suite™

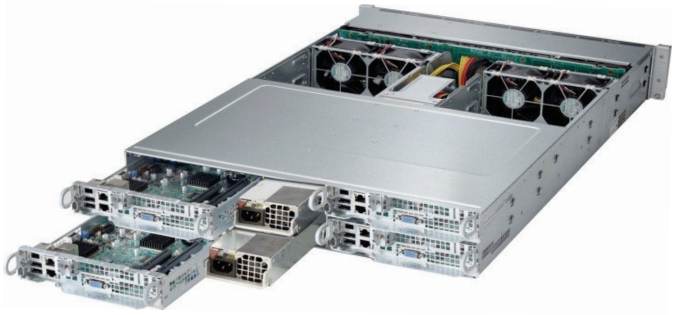


Figure 1: *Overhead View of SGI Rackable Server with the Top Cover Removed*

1.2 SGI® ICE™ X System

SGI® ICE™ X is one of the world's fastest commercial distributed memory supercomputer. This performance leadership is proven in the lab and at customer sites including the largest and fastest pure compute InfiniBand cluster in the world. The system can be configured with compute nodes comprising Intel® Xeon® processor E5-2600 v3 series exclusively or with compute nodes comprising both Intel® Xeon® processors and Intel® Xeon Phi™ coprocessors or Nvidia® compute GPU's. Running on SUSE® Linux® Enterprise Server and Red Hat® Enterprise Linux, SGI ICE X can deliver over 172 teraflops of performance per rack and scale from 36 to tens of thousands of nodes.

SGI ICE X is designed to minimize system overhead and communication bottlenecks and can be architected in a variety of topologies with choice of switch and single or dual plane FDR interconnect. The integrated bladed design offers rack-level redundant power and cooling via air, warm or cold water and is also available with storage and visualization options (Figure 2).

SGI ICE X and XA configuration used in this paper:

Benchmark System

- SGI ICE X and XA
- Intel® Xeon® 12-core 2.6 GHz E5-2690v3
- IB FDR Interconnect
- 5.3 GB of Memory/core Memory Speed 2133 MHz
- Altair® PBS Professional Batch Scheduler v12
- SLES or RHEL with latest SGI Performance Suite™

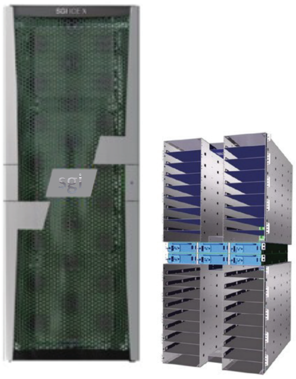


***Figure 2:** SGI ICE X Cluster with Blade Enclosure*

1.3 SGI® UV™ 2000

SGI UV 2000 server comprises up to 256 sockets (2,048 cores). Support for 64TB of global shared memory in a single system image enables efficiency of SGI UV for applications ranging from in-memory databases, to diverse sets of data and compute-intensive HPC applications all the while programming via the familiar Linux OS [2], without the need for rewriting software to include complex communication algorithms. TCO is lower due to one-system administration needs. Workflow and overall time to solution is accelerated by running Pre/Post-Processing, solvers and visualization on one system without having to move data (Figure 3).

Job memory is allocated independently from cores allocation for maximum multi-user, heterogeneous workload environment flexibility. Whereas on a cluster, problems have to be decomposed and require many nodes to be available, the SGI UV can run a large memory problem on any number of cores adapting to application license availability and with less concern about lack of memory resources killing the job.

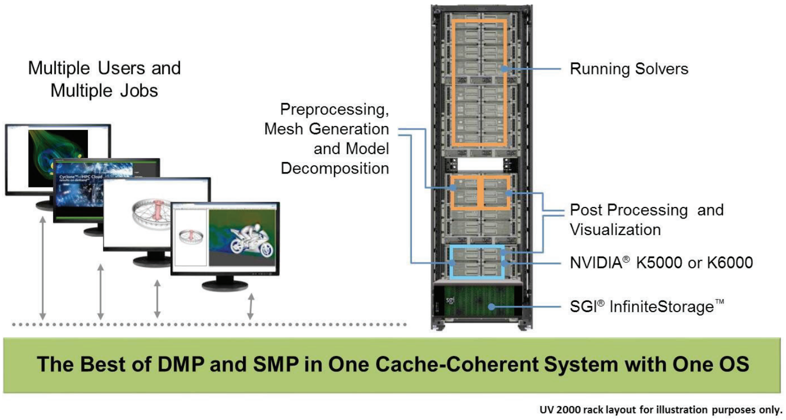

Multiple Users and Multiple Jobs
Preprocessing, Mesh Generation and Model Decomposition
Running Solvers
Post Processing and Visualization
NVIDIA® K5000 or K6000
SGI® InfiniteStorage™
The Best of DMP and SMP in One Cache-Coherent System with One OS
UV 2000 rack layout for illustration purposes only.

***Figure 3:** SGI UV Computer-aided engineering (CAE) workflow running ANSYS applications*

SGI UV configuration used in this paper:

- SGI® UV™ 2000
- Intel® Xeon® 8-core 3.3 GHz E5-4527v2
- NUMAlink® 6 Interconnect
- 8 GB of Memory/core Memory Speed 1867 MHz
- Altair® PBS Professional Batch Scheduler v12
- SLES or RHEL with latest SGI Performance Suite™

1.4 SGI Performance Tools

Utilizing the latest MPI compliant libraries and standard-distribution Linux, SGI® Performance Suite (Figure 4) fuels HPC applications to achieve breakthrough speed and scale. A feature-rich tool set optimizes application placement, enables application tuning at runtime without recompiling, and can boost performance up to 70%. Fine-grain metrics facilitate MPI analysis. Checkpoint Restart augments productivity. And hard real-time performance can be realized without special kernels on standard Linux. Coupled with world-class application expertise, SGI takes Linux to the next level. For detailed information: http://www.sgi.com/products/software/.

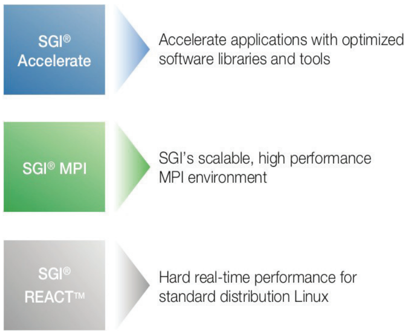

SGI® Accelerate
Accelerate applications with optimized software libraries and tools
SGI® MPI
SGI's scalable, high performance MPI environment
SGI® REACT™
Hard real-time performance for standard distribution Linux

***Figure 4:** SGI Performance Suite Component*

1.5 SGI System Management Tools

Spanning bare-metal provisioning and protection against memory failure, to 24x7 systems monitoring, task automation, and innovative power optimization, SGI® Management Suite helps maximize productivity and achieve a high return on your investment. Administrators can deploy systems and upgrades with unparalleled speed, proactively manage system health and energy consumption, and deliver consistently high service levels — enabling users to run more jobs in less time and without interruption. For detailed information: http://www.sgi.com/products/software/smc.html

1.6 Resource and Workload Scheduling

Resource and workload scheduling allows one to manage large, complex applications, dynamic and unpredictable workloads, and optimize limited computing resources. SGI offers several solutions that customers can choose from that best meet their needs.

Altair Engineering PBS Professional® is SGI's preferred workload management tool for technical computing scaling across SGI's clusters and servers. Features:

- Policy-driven workload management which improves productivity, meets service levels, and minimizes hardware and software costs
- Integrated operation with SGI Management Center for features such as workload-driven, automated dynamic provisioning
- Altair PBS Professional Power Awareness integrates job-level power management with SGI Management Center 3

Adaptive Computing Moab® HPC Suite Basic Edition

Adaptive Computing Moab® HPC Suite enables intelligent predictive scheduling for workloads on scalable systems.

- Policy-based HPC workload manager that integrates scheduling, managing, monitoring and reporting of cluster workloads
- Includes TORQUE resource manager

1.7 SGI VizServer® with NICE DCV

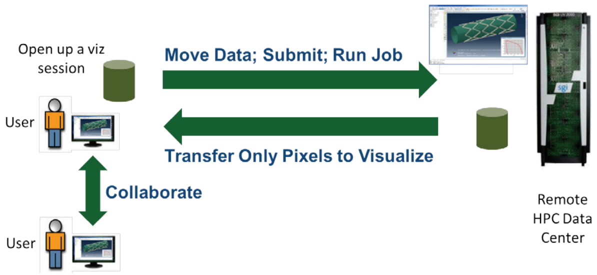


Figure 5: *SGI VizServer workflow*

SGI VizServer with NICE DCV installed on a company's servers can provide LS-PrePost remote visualization capabilities through a software-as-a-service (SaaS) built in the company's private network. The LS-PrePost software is accessed through an easy-to-use web interface, resulting in simplicity for the end user. This solution provides intuitive help and guidance to ensure that less-experienced users can maximize productivity without being hindered by complex IT processes.

SGI VizServer with NICE DCV Components:

- Engineer-friendly self-service portal: The self-service portal enables engineers to access the ANSYS post processing application and data in a web browser–based setting. It also provides security, monitoring, and management to ensure that users cannot leak company data and that IT managers can track usage. Engineers access the LS-PrePost application and data directly from their web browsers, with no need for a separate LS-PrePost software installation on their local client.
- Resource control and abstraction layer: The resource control and abstraction layer lies underneath the portal, not visible to end users. It handles job scheduling, remote visualization, resource provisioning, interactive workloads, and distributed data management without detracting from the user experience. This layer translates the user request from the browser and facilitates the delivery of resources needed to complete the visualization or HPC tasks. This layer has a scalable architecture to work on a single SGI Rackable cluster or SGI UV server, as well as a multi-site WAN implementation.
- Computational and storage resources: The SGI VizServer with NICE DCV software takes advantage of the company's existing or newly purchased SGI industry-standard resources, such as servers, HPC schedulers, memory, graphical processing units (GPUs), and visualization servers, as well as the required storage to host application binaries, models and intermediate results. These are all accessed through the web-based portal via the resource control and abstraction layer and are provisioned according to the end user's needs by the middle software.

The NICE DCV and EnginFrame software is built on common technology standards. The software adapts to network infrastructures so that an enterprise can create its own secure engineering cloud without major network upgrades. The software also secures data, removing the need to transfer it and stage it on the workstation, since both technical applications and data stay in the private cloud or data center. These solutions feature the best characteristics of cloud computing—simple, self-service, dynamic, and scalable, while still being powerful enough to provide 3D visualization as well as HPC capabilities to end users, regardless of their location.

2.0 ANSYS Fluent Overview

Computational fluid dynamics (CFD) simulation software provides the ability to predict, with confidence, the impact of fluid flows on your product — throughout design and manufacturing as well as during end use. For example, fluid flow analysis capabilities can be used to design and optimize new equipment and to troubleshoot already existing installations to give valuable insight into the product's performance. ANSYS Fluent software is a leader in the engineering simulation industry with the broad physical modeling capabilities needed to model flow, turbulence, heat transfer, and reactions for industrial applications ranging from air flow over an aircraft wing to combustion in a furnace, from bubble columns to oil platforms, from blood flow to semiconductor manufacturing, and from clean room design to wastewater treatment plants. Special models provide simulation of in-cylinder combustion, aeroacoustics, turbomachinery, and multiphase systems.

These complex simulation models generate vast quantities of data that requires the highest performance computing in order to provide a solution in a reasonable period of time. For example, the software represents the real world product as a series of fine meshes that contain a billion or more cells, each of which is described by a series of mathematical equations that must be solved multiple times as the solution converges. ANSYS Fluent using advanced parallel processing numerics efficiently utilizes numerous multi-core processors in a single machine and in various machines on a network to minimize the time to solution. Dynamic load balancing automatically detects and analyzes parallel performance and adjusts the distribution of computational cells among the processors so that a balanced load is shared by the CPUs even when complex physical models are in use. ANSYS Fluent is available on the Linux® platform from SGI.

2.1 ANSYS Fluent Application Architecture

ANSYS Fluent involves three distinct phases of use, each with its own hardware requirements.

2.1.1 Pre-Processing Phase

Pre-processing for ANSYS Fluent involves other ANSYS software applications, including ANSYS CAD Interfaces (for access to CAD geometries), ANSYS DesignModeler (for geometry creation or modification), ANSYS Meshing, and ANSYS DesignXplorer. The ANSYS Fluent user interface will also be invoked during pre-processing. All of these applications are hosted in the ANSYS Workbench environment and are used in an interactive, graphical mode.

Typically, these applications run on standalone desktop workstations or via a remote visualization tool on a server and executed on single processors. Technologies creating the computational elements needed by the solver can be developed using many processors, in parallel. Memory requirements for ANSYS Meshing, ANSYS DesignModeler and ANSYS DesignXplorer depend on the size of the model. Typical input files (also called "case files") created during the pre-processing phase will range in size from 100 MB (or less) to 2–3 GB for larger workloads. Output files (also called "data files"), as noted below, will be significantly larger. Pre-processing is graphically intensive and requires a high-end ANSYS certified graphics card.

2.1.2 Solution Phase

The solution phase involves running the ANSYS Fluent solver to solve the equations that describe the physical behavior under consideration. This phase is computationally and memory-intensive and is optimized through the use of parallel processing on a multi-core workstation, a server or a cluster of servers/blades. Appropriately sized hardware can reduce turnaround time from weeks to days or from days to hours. Proper hardware also enables larger, more detailed simulation models.

Most ANSYS Fluent simulation models for product design needs are executed on 16 to 128 computational cores depending on the resolution and size of the model. Large, high-resolution models can take advantage of a thousand or even more than 10 thousand processing cores.

Fluent Rating and Speed-up

Two measures are used to assess the performance of the solution phase in a cluster environment:
Fluent Rating and Speed-up.

Fluent Rating is a throughput measure defined as the number of benchmark jobs that can be performed within a 24-hour period:

Fluent Rating = 86,400 seconds/Number of seconds required to complete a single benchmark job.

Speed-up is a factor of improvement over a reference platform. For example, if the reference platform is a 2-node configuration, speed-up for a 32-node configuration is: Speed-up = Fluent Rating on 32 nodes/Fluent Rating on two nodes

Parallel efficiency is: Parallel Efficiency = Speed-up/(Number of nodes in the configuration/Number of nodes in the reference configuration)

Usually, one node is used in the reference configuration. Sometimes, the minimum configuration tested can be more than one node due to limitations such as insufficient memory. ANSYS Fluent is a highly scalable application. When implemented on networks that minimize barriers to scalability, it can result in excellent speed-up and efficiency.

Parallel Processing Capabilities

Parallelism in computer systems exists in two paradigms:

- Distributed Memory Parallelism (DMP) uses the MPI Application Programming Interface (API) which focuses on an explicit physical domain decomposition. The resulting reduced size geometric partitions have lesser processing resource requirements, resulting in increased efficiency. However, the size of the common boundary should be kept minimal to decrease inter-process communication.
- Shared Memory Parallelism (SMP) uses various kinds of shared threads (e.g., OpenMP, pthreads, etc.). 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 ANSYS Fluent features that use the above hybrid approach are the AMG solver, particle tracking, ray tracing and architecture-aware partitioning.

Distributed Parallel Capabilities in ANSYS Fluent

Fluent parallel simulations always begin with a Geometry Domain Decomposition. This technique partitions the geometry model and distributes the partitions among the cores on each processor socket. 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. The workload for each Message Passing Interface (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 a round-robin fashion across all the nodes. Practically all domain decompositions are transparent to the user, but he/she can choose among available domain decomposition techniques (Principal Axes, Metis, etc.).

2.1.3 Hardware and Software Nomenclature

Distributed Memory Parallelism is implemented through the problem at hand with domain decomposition. Depending on the physics involved in their respective industry, the domains could be geometry, finite elements, matrix, frequency, load cases or right hand side of an implicit method. Parallel inefficiency from communication costs is affected by the boundaries created by the partitioning. Load balancing is also important so that all MPI processes perform the same number of computations during the solution and therefore finish at the same time. Deployment of the MPI processes across the computing resources can be adapted to each architecture with 'rank' or 'round-robin' allocation.

2.1.4 Parallelism Metrics

During the solution phase, relatively little file Input/Output (I/O) is required, although some I/O is typically done to monitor solution progress. At the end of the solution phase, ANSYS Fluent will save a results file (also called a "data file"). This output file will range in size from a few hundred MB up to 10 to 20 GB on the high end. Many workloads will create multiple output files. For long-running transient simulations, it may help to auto-save intermediate data throughout the calculation. These workloads can then be optimized with a high-performance file system and storage network. The post-processing phase may involve ANSYS Fluent (which includes an integrated post-processor) or ANSYS CFD Post.

3.0 Major components of ANSYS Fluent Architecture

The six major components of ANSYS Fluent architecture are:

CORTEX
CORTEX is the front-end GUI for ANSYS Fluent. It allows end-users to interact with the application when it is run interactively.

HOST
The HOST process reads the input data, such as case and data, and communicates with Computation Task 0 to distribute the mesh information to the rest of the computation tasks. In addition, the HOST task is used to perform I/O during simulation of transient models.

Computation Tasks
The set of computation tasks is the software abstraction that implements the solution phase described earlier. Computation tasks communicate with each other using MPI. Computation Task 0 has special significance in that it interacts with the HOST process to receive/send mesh-related information. Optionally, the entire set of computation tasks can retrieve/store model state information (.pdat files) using MPI-IO directly from the I/O device, avoiding the overhead of the HOST process. A parallel file system is required to support parallel I/O performed by the computation tasks.

File System

A file system is needed to store/retrieve information during the ANSYS Fluent solution process. The file system can be local to the node where the HOST task resides or it can be a shared file system such as a Network File System (NFS) or General Parallel File System (GPFS). Both of which are accessible to the HOST task over an interconnect fabric such as Ethernet or InfiniBand.

Computation Nodes

Computation nodes are all multi-core and are used to run ANSYS Fluent computation tasks. Each core in a computation node runs one computation task.

Network (Interconnect)

This is a set of communication fabrics connecting the computation nodes that run HOST tasks, file servers and client workstations. Usually, these systems are grouped by function and each is connected by a separate network. For example, all computation nodes may be connected by a private network. However, there is always a gateway-type node, such as a head node, that belongs to two networks so that the entities on one network can route messages to entities on another network. When parallel processing is invoked for ANSYS Fluent, the software partitions the simulation model and distributes it as a set of computational tasks for a specific set of processors. Each task is responsible for computing the solution in its assigned portion of the model.

The main activity of each task is computational: carrying out an iterative process to compute the final solution in its assigned grid partition. However, because the solution near the boundary of each partition generally depends on the solution in neighboring partitions, each task must also communicate and exchange data with the tasks responsible for the neighboring grid partitions. The communication and data exchange in Fluent involves the transmission of relatively short messages between the communicating tasks. This is accomplished using an MPI software layer between nodes and (for certain models) a shared memory OpenMP approach within each node. The software layers are packaged and distributed with ANSYS Fluent and do not need to be procured separately.

4.0 SGI Benchmarks of ANSYS Fluent

4.1 Job Submittal Procedure

The following are important for Submittal Procedure:

- Placement of processes and threads across nodes and sockets within nodes
- Control of process memory allocation to stay within node capacity

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 file should be placed on the high performance shared parallel file system. The high performance file system could be an in-memory file system (/dev/shm), a Direct (DAS) or Network (NAS) Attached Storage file system. In diskless computing environments, in-memory file system or network attached storage are the only options. This file system nomenclature is illustrated in Fig. 6.

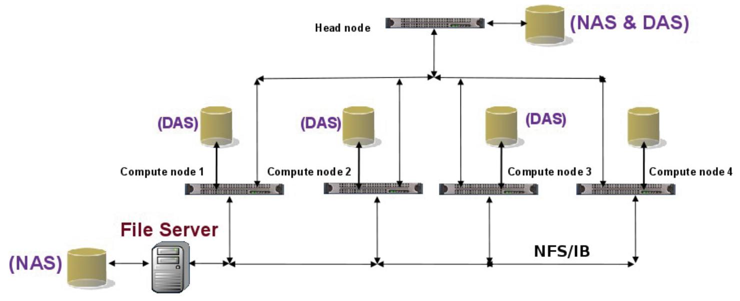


Figure 6: *Example File Systems for Scratch Space*

The following is the synopsis of a job submission script.

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

Fluent employs both serial and parallel I/O. The latter one relies on parallel MPI I/O functions of MPI. The choice of serial or parallel I/O is controlled by an extension of a stored solution file: xx.dat for a serial I/O and xx.pdat for a parallel I/O.

4.2 Submittal command

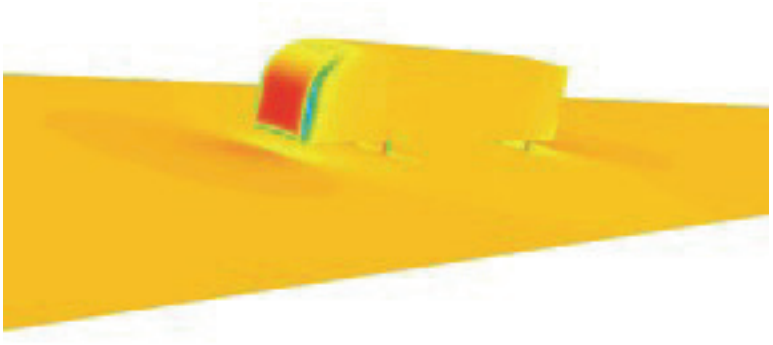
The following keywords were used for the ANSYS Fluent execution command:

```
FLUENT <precision> –<rshel> –p<network> –cnf =hosts –t<nprocs> -mpi =<MPI > -i <journal> <GUI>
```

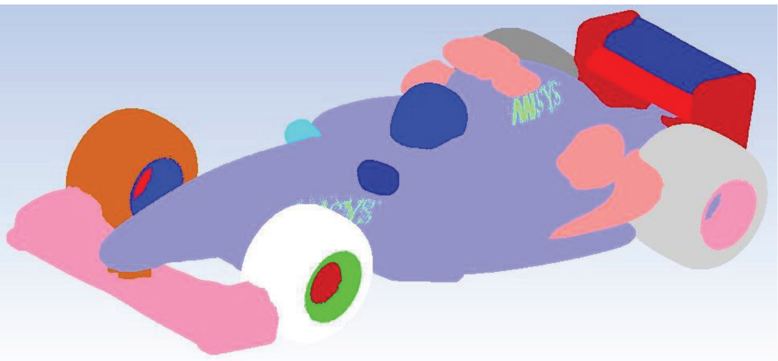
- precision: single (3d) or double (3ddp) precision solver
- rshel: remote shell (rsh or ssh)
- network: type of internode network—eth for GigE or ib for InfiniBand
- hosts: list of nodes used in a DMP job
- Nprocs: assigns a global number of MPI compute threads
- MPI: MPI implementation version (default hp)
- journal: file defining the application control flow
- -g: No GUI

5.0 Benchmark Examples

A number of performance studies were conducted based on Ansys standard benchmarks including a Truck with 111 million cells and an Open wheel Racecar with 280 million cells.



***Truck_14(111)m:** 14(111)m Cells, Turbulent Flow, Segregated Implicit Solver*



***Open_Racecar_280m:** 280m Cells, Turbulent Flow, Pressure-Based Coupled Solver*

5.1 Benchmark Results Fluent r16

As depicted in the following test results, the new generation of Intel® Xeon® processors E5-2690 v3 brings a significant performance improvement over Intel's previous family of processors. The major contributing factors were:

- Higher number of cores/socket
- Higher clock rate
- Higher memory speed

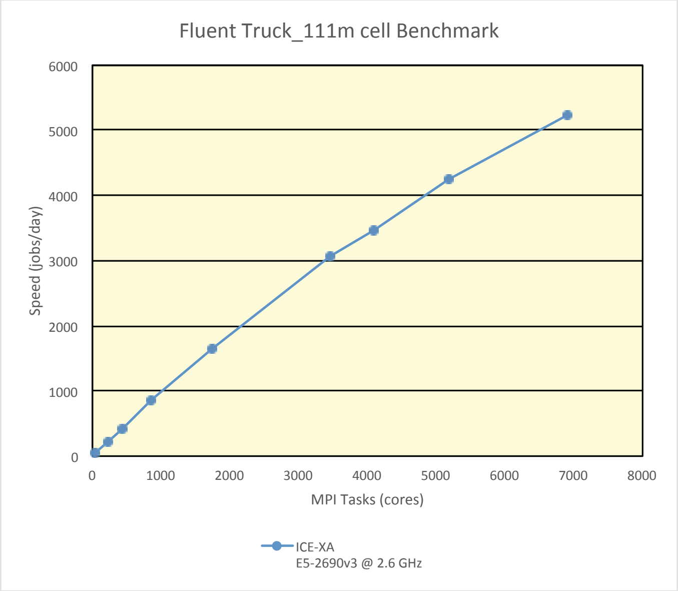


***Figure 7a:** Truck 111m*

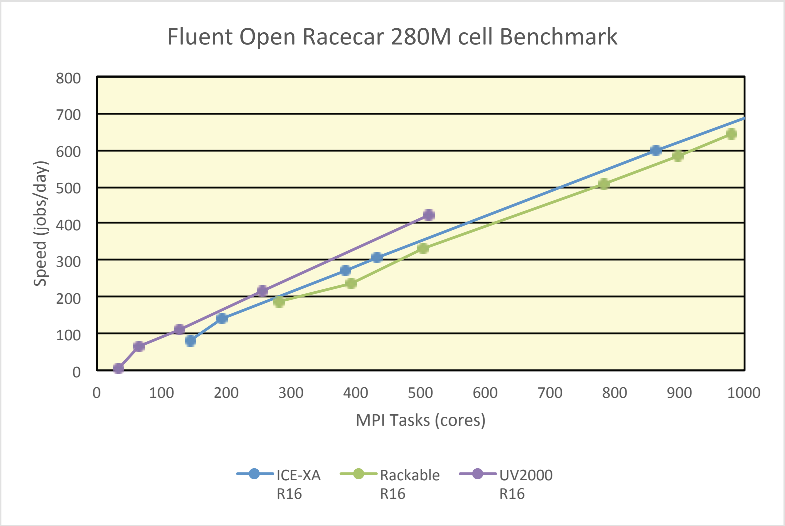


Figure 7b: *Open_racecar 280m Cells*

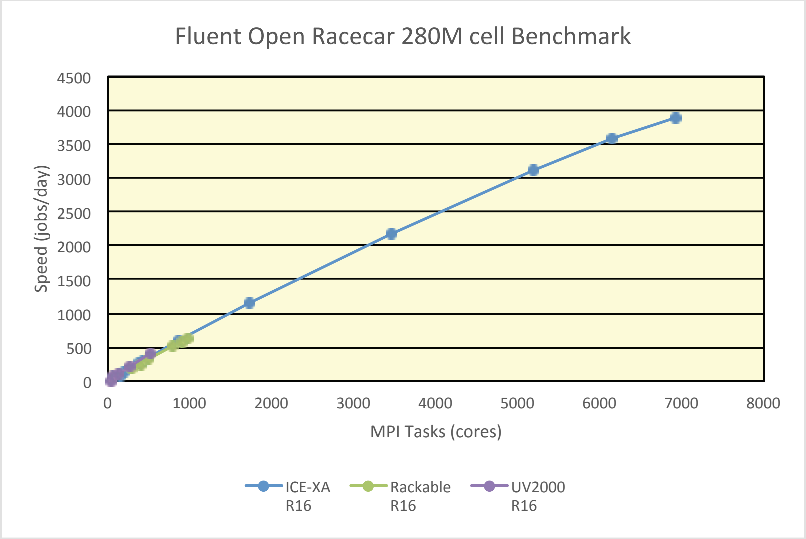


Figure 7c: *Open_racecar 280m Cells*

5.1.1 SGI ICE X Benchmark Results comparing Fluent r15 vs. Fluent r16 Performance

In Figures 8a-b we compare the performance of ANSYS Fluent r15 to the latest ANSYS Fluent r16 release running on an SGI ICE X and XA system with the latest Intel® Xeon® E5-2690 v3 processor. You will see that there have been major improvements in the parallel layer of Fluent that has lead to improved scalability. For all the Fluent r16 release highlights please visit http://www.ansys.com/Products/ ANSYS+16.0+Release+Highlights

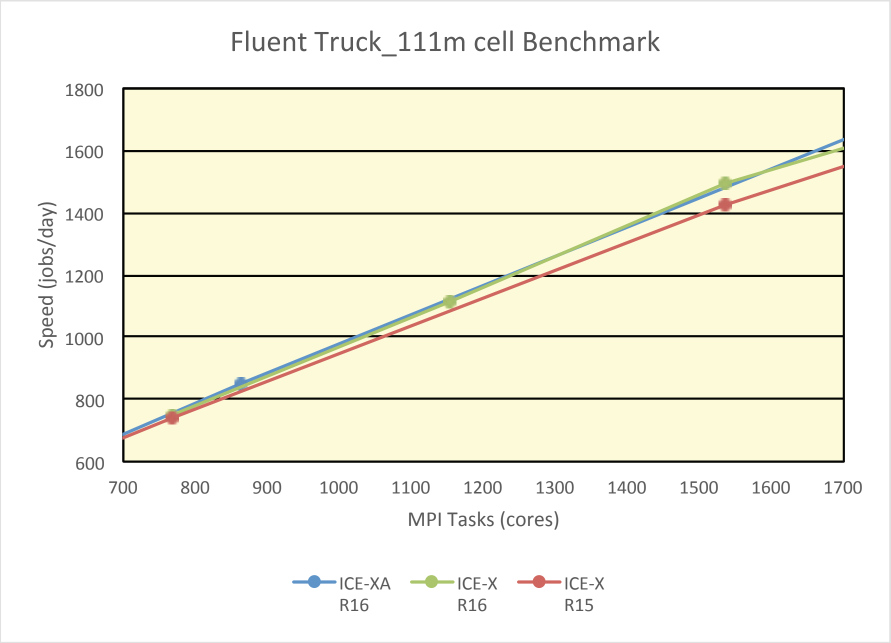


Figure 8a: *Truck 111m (detail)*

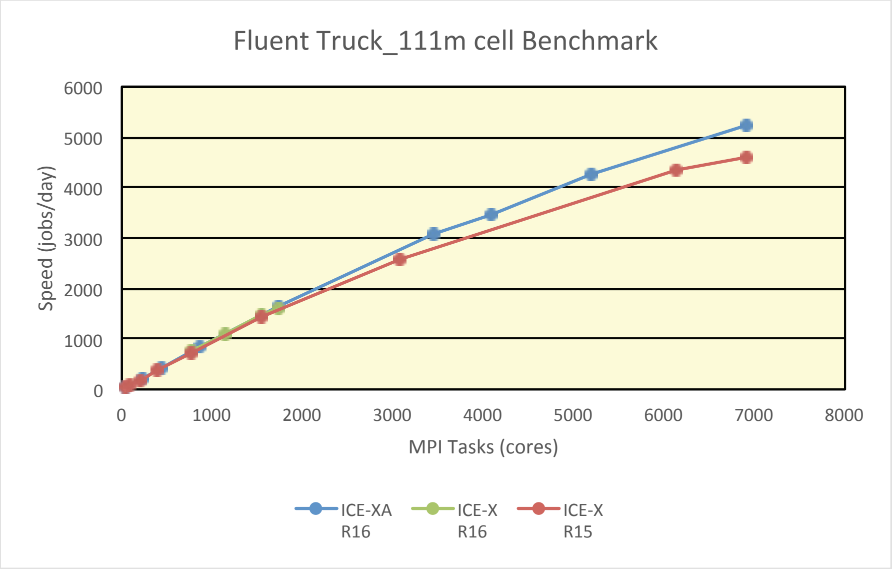


Figure 8b: *Truck 111m*

5.2 Advantages of the SGI MPI Library through SGI® PerfBoost

An MPI library capability to bind an MPI rank to a processor core is the key to control performance because of the multiple node/socket/core environments. The MPI platform 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, SGI MPI, through the 'omplace' command provides uniquely convenient placement of hybrid MPI/OpenMP processes and threads within each node. This MPI library is linklessly available through the SGI PerBoost facility found in SGI® Performance Suite. SGI PerfBoost provides a Platform MPI, Intel® MPI, Open MPI, HP-MPI ABI-compatible interface to SGI's MPI (1).

For Fluent, the major advantage of SGI PerfBoost is its ability to work on a very high number of processors. We were able to scale one of the large Fluent benchmarks (truck_111m) to 6,912 cores with Fluent 15 & 16.

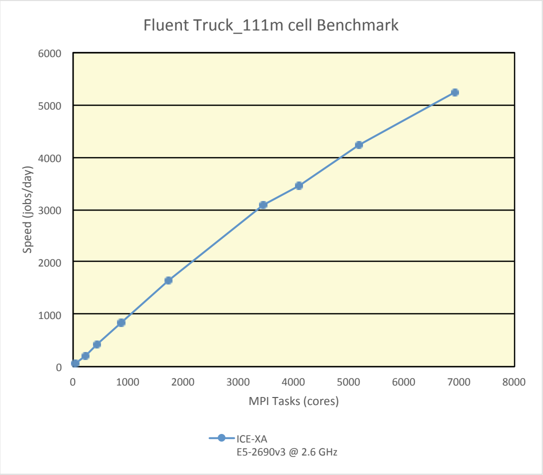


Figure 9: *Fluent Scalability Chart*

There is another compelling observation in favor of SGI PerfBoost. Using the Fluent built-in affinity setting on a large SMP system in conjunction with a batch system can lead to unexpected thread allocations since this setting doesn't have a "cpuset" feature. Cpusets constrain the CPU and memory placement of tasks to only the resources within a task's current cpuset. They form a nested hierarchy visible in a virtual file system, usually mounted at /dev/cpuset. Cpusets provide an essential mechanism for managing dynamic job placement on large systems. Without cpusets, requests are scattered across the system thus impacting the runtime performance. In contrast, SGI MPI uses this feature by default so all threads will be CPU-bound in a precise manner (the default allocation is the user defined allocation).

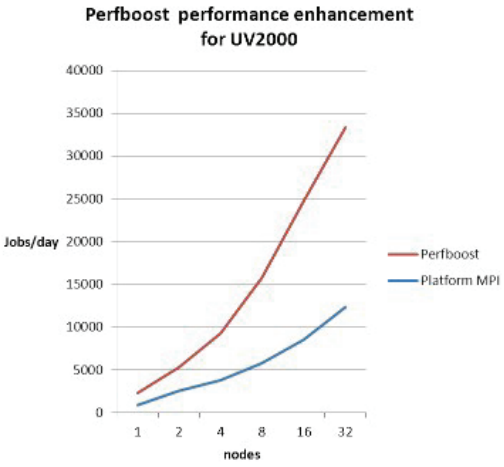


Figure 10: *SGI PerfBoost vs. Platform MPI*

5.3 Effect of core Frequency and Intel® Turbo Boost Technology

An additional data point is the added performance gained through the use of Intel® Turbo Boost. The Turbo Boost feature was first introduced in the Intel® Xeon® 5500 series. It is used to increase performance by raising the core operating frequency within controlled limits depending on the sockets' thermal envelope. The mode of activation is a function of how many cores are active at a given moment, which may be the case when OpenMP threads or MPI processes are idle under their running parent. For example, for a base frequency of 2.6 GHz, with 1-2 cores active, their running frequencies will be throttled up to 3.5 GHz, but when 3-4 cores are active, they will only be up to 3.3 GHz. For most computations, utilizing Turbo Boost technology can result in improved runtimes, but the overall benefit may be mitigated by the presence of performance bottlenecks other than pure arithmetic processing. With Turbo Boost, 80% of the simulation time runs at 3.20 GHz about a 7% improvement. Single core tasks peak at 3.50GHz, which is a 17% improvement over the base 3.00 GHz frequency.

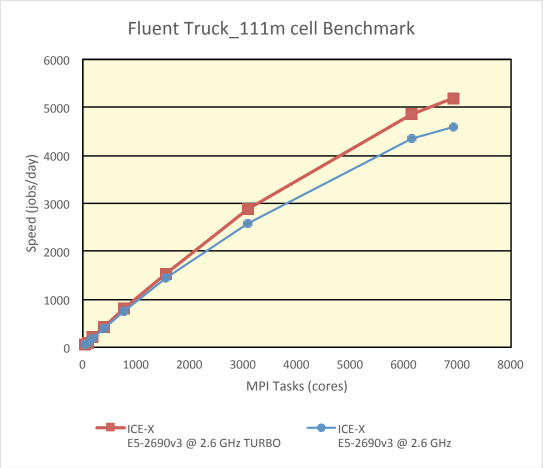


Figure 11: *Intel Turbo Boost turned on and turned off example*

5.4 Effect of Memory Speed and Memory Bandwidth

In addition to core frequency, memory speed and bandwidth can affect the overall performance of any application. In the case of Fluent, we compared performance with memory set to the default speed of 2133 MHz verse a downclocked speed of 1867 MHz. The memory was downclocked in BIOS on a subset of nodes in the Rackable Benchmark Cluster. The following memory tests were performed with the Truck_111m model. As seen in Figure 12a below, the memory speed is shown to have a measurable, but small effect of approximately 2%.

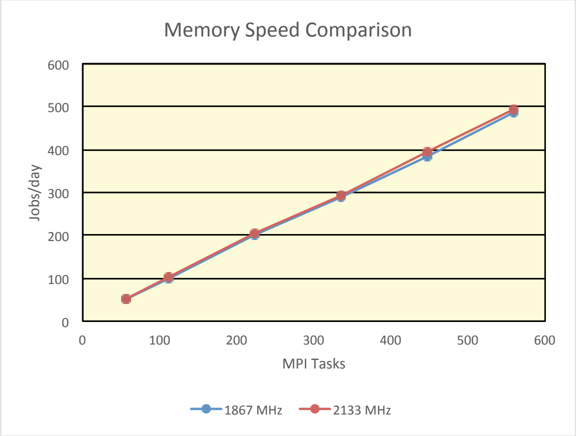


Figure 12a: *Memory Speed Test on truck_111m*

With the wide variety of CPU SKUs available from Intel®, another factor that can affect performance is memory bandwidth. The memory bandwidth to the node CPU socket is fixed. As a result, as the number of cores per socket is increased, the memory bandwidth available to each individual core is reduced. We studied this affect by running the benchmark on the ICE-X Benchmark System using all 12 cores in the CPU. The benchmark was then repeated using only 8 of the 12 available cores per CPU.

Increasing the memory bandwidth by using only 8 cores improved performance per core by 9% as seen in Figure 12b. This means that selecting a CPU SKU with fewer cores, at the same frequency, will increase the performance of each core. However, Figure 12c shows the overall performance of the node increases when choosing a CPU with more cores, rather than fewer. This is because Fluent is a highly scalable model so the performance gain of the additional MPI tasks per node outweighs the performance loss due to the decreased memory bandwidth per task. In the test, the overall gain at the node level was 28% going from 8 to 12 cores.

In addition, because the infrastructure cost of cluster scales with number of nodes, selecting a CPU SKU with higher core counts helps reduce Total Cost of Ownership. This benefit may be somewhat reduced by a higher cost of the individual CPU though. Because pricing is always subject to change, the overall benefit cannot be exactly quantified here.

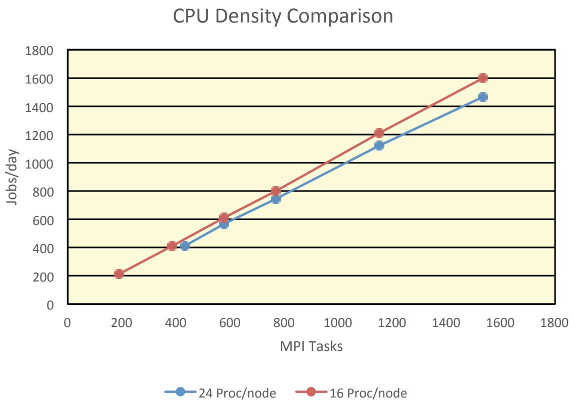


Figure 12b: *Per Core Performance on truck_111m*

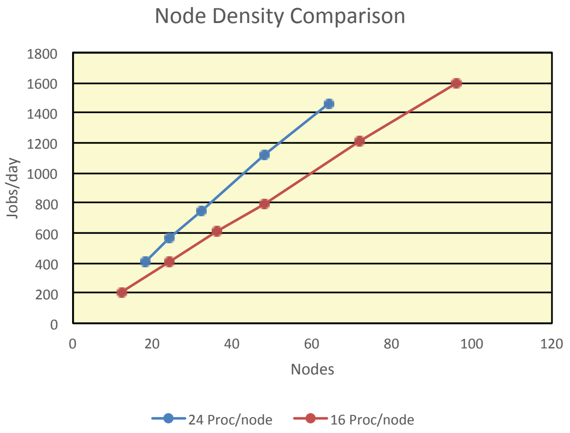


Figure 12c: *Per Node Performance on truck_111m*

5.5 Choice of Network Technology for a Cluster

Quite often during the configuration phase of architecting a cluster-based system the following question will arise; what network technology should one choose? If you are using more than two nodes it is recommended that you use InfiniBand. It is known that, generally speaking, InfiniBand FDR is faster but relatively expensive and InfiniBand QDR is cheaper than FDR but generally slower. So the choice becomes quite complex and the outcome often depends on the performance of the application of interest on a particular network type.

Therefore, we ran a few experiments running Fluent simulations with InfiniBand FDR and QDR on a SGI Rackable system with Xeon® E5-2697 v2 processors. In the range of 32 nodes (768 cores) we did not see any significant performance difference between QDR and FDR networks. This was expected because the efficiency of Fluent communication layer is defined by the network latency which is practically the same for QDR and FDR. The point-to-point Fluent messages are very small due to the nature of the linear solver algorithm. FDR can be the interconnect of choice for applications which mostly send large messages and as such depends on a fast network bandwidth or, where you have the need, scale Fluent to run on 100's of nodes. SGI expects a similar result when comparing FDR Infiniband to the newer EDR Infiniband. We will be releasing an update to this document when FDR vs. EDR benchmark data is available.

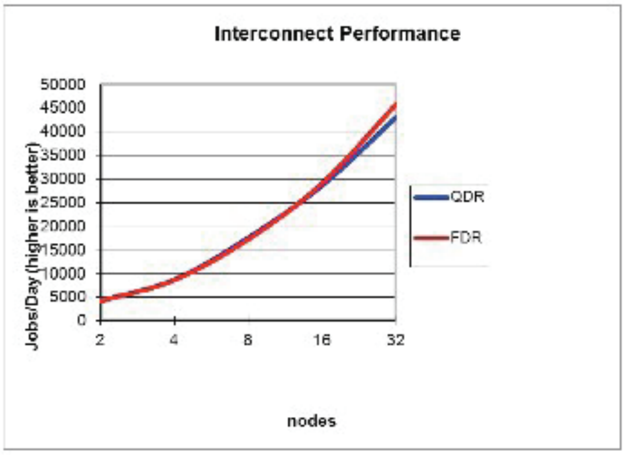


Figure 13: *InfiniBand QDR vs. FDR example*

5.6 Use of Hyper-Threading Technology

Intel® Hyper-Threading Technology (Intel® HT Technology) is an Intel® patented technique for simultaneous multithreading (SMT). In this technique some execution elements are duplicated, specifically elements that store the executional state, whereas elements that actually do the execution are not duplicated. This means that only one processor is physically present but the operating system sees two virtual processors, and shares the workload between them. Note those units such as L1 and L2 cache, and the execution engine itself are shared between the two competing threads.

Hyper-threading requires both operating system and CPU support. The Intel® Xeon® 5600 series processor reintroduced a form of Intel® HT Technology where 6 physical cores effectively scale to 12 virtual cores thus executing 12 threads. Thus, for example, in the current Intel® Xeon® 14-core 2.6 GHz E5-2697 v3 based compute node, there is a total of 56 virtual cores allowing one to execute 56 threads. In practice, an executing thread may occasionally be idle waiting for data from main memory or the completion of an I/O or system operation. A processor may stall due to a cache miss, branch misprediction, data dependency or the completion of an I/O operation. This allows another thread to execute concurrently on the same core taking advantage of such idle periods.

In general, some applications can benefit as much as 30% from hyper-threading providing a very economical way to gain extra performance. However, different software can have diverse execution profiles, thus one will expect hyper-threading to have a variable effect on different applications. For CFD software, hyper-threading benefits will not only depend on the fact that the software has different coding techniques, but also different CFD capabilities as well as the different options and features of various CFD input models. When running ANSYS Fluent, we found, depending on the size of the problem, hyper-threading might be useful on only one or two nodes. Because of the highly synchronized nature of computationally intensive HPC parallel codes, the usefulness of hyper-threading can be very limited. It should also be noted that with commercial CFD applications, a user should weigh the technical and performance benefits of hyper-threading against the possible additional licensing costs that might be incurred.

ANSYS' suggestion is to disable HyperThreads in the system BIOS on each node. This is discussed in ANSYS' Installation and Licensing Documentation. In our testing, it was found to be adequate to request PBS resources in such a way that the HyperThreads were not included in the cpuset.

6.0 Conclusion

Computational Fluid Dynamics is a time-consuming and performance-intensive activity. The SGI Rackable, ICE X and XA and UV product lines deliver the scalability and performance required of complex physical modeling that includes multi-phase flows, chemistry, combustion, and spray among other physical phenomena. This study showed how the effect on performance of Turbo Boost, specific network choice and hyper-threading can be gauged for a given dataset. In particular, one observed:

- Great parallel scaling.
- Using of SGI MPI (through PerfBoost) outperforms other MPI's for high number of threads.
- The effect of frequency and Turbo Boost are weak as this is not the only limiting factor for performance.
- InfiniBand FDR doesn't bring a significant performance boost over InfiniBand QDR on a low number of nodes but can be significantly faster on a large number of nodes.
- Hyper-threading does not help because of communication costs beyond 8 processes.
- Higher core count CPUs increase the value/performance metric.

All these effects are definitely dependent on the dataset and solution methods used. Procurement of the right mix of resources should therefore be tailored to the range of datasets envisaged. Moreover, the performance metrics discussed here could be only one of many, others of which may include turnaround time, throughput or cost-itself comprised of acquisition cost, licenses, energy, facilities and services.

SGI will support customers with any issues resulting in running ANSYS Fluent using PerfBoost. ANSYS certifies its software on specific MPI libraries. SGI MPI and SGI PerfBoost technologies are not certified by ANSYS, but assumed to be compatible. ANSYS supports customers on any issues that can be reproduced without the use of PerfBoost.

7.0 About SGI

SGI is a global leader in high performance solutions for compute, data analytics and data management that enable customers to accelerate time to discovery, innovation, and profitability. Visit sgi.com for more information

Global Sales and Support: sgi.com