WHITE PAPER

sgi®

Performance and Productivity Breakthroughs with Very Large Coherent Shared Memory: The SGI® UV Architecture

January, 2012

TABLE OF CONTENTS

1. Introduction 3

2. Background 3
  - 2.1 Distributed Memory Architectures 3
  - 2.2 Data Access Hierarchy 4

3. SGI Shared Memory Architecture: Massive Shared Memory Resources 5
  - 3.1 Memory Scaling 5
  - 3.2 Advanced Features of SGI UV 5
    - 3.2.1 Flexible I/O Scaling 5
    - 3.2.2 Flexible Interconnect Scaling (Connectivity, Latency and Bandwidth) 7

4. Benefits of Global Shared Memory 7
  - 4.1 Massive In-Core Computation by Design 7
  - 4.2 Massively Memory-Mapped I/O 7
  - 4.3 Highly Efficient Message Passing 7
  - 4.4 Arbitrary Scaling of Problem Size 8
  - 4.5 Efficient Utilization of Memory Resources 8
  - 4.6 Greatly Simplified Load Balancing 8
  - 4.7 Efficient Development and Administration for Reduced TCO 9

5. Programming Models 9

6. Use Cases and New Possibilities 9
  - 6.1 Fraud Detection/Cyber Security 9
  - 6.2 BioInformatics 10
  - 6.3 Data Analytics 10
  - 6.4 Real-Time Interaction with Massive Datasets 10
  - 6.5 Memory Resident Databases such as Oracle TimesTen 11
  - 6.6 Dataflow Applications 11

7. Summary 11

1. Introduction

Computer systems that implement a large coherent shared memory (CSM) architecture while scaling to high processor counts offer a new dimension in the advancement of application performance. A large CSM architecture can produce breakthroughs in achievable performance and user productivity, while offering entirely new ways of solving a wide range of problems in science, engineering and business applications.

This paper describes the advantages of a large CSM architecture with careful consideration of the data access hierarchy that is typical in high end computing and data analysis. The speed with which a processor can access data drops by many orders of magnitude in the progression from on-chip cache to off-chip cache,to system memory, and finally to disk. A well-implemented large CSM architecture improves processor performance by maximizing the use of fast data access. Specific attributes of the SGI® UV large CSM implementation are discussed, along with real world examples of breakthrough computational results.

2. Background

2.1 Distributed Memory Architectures

Multiprocessor distributed memory systems (see Figure 1) consist of multiple "nodes" which communicate via a dedicated network of router elements and interconnect channels (links). Endpoint nodes generate and process the messages which travel across the network links, whereas router elements simply forward messages toward their intended destinations. The computational, memory and communications resources that give a system its computational capabilities reside within the individual end point nodes.

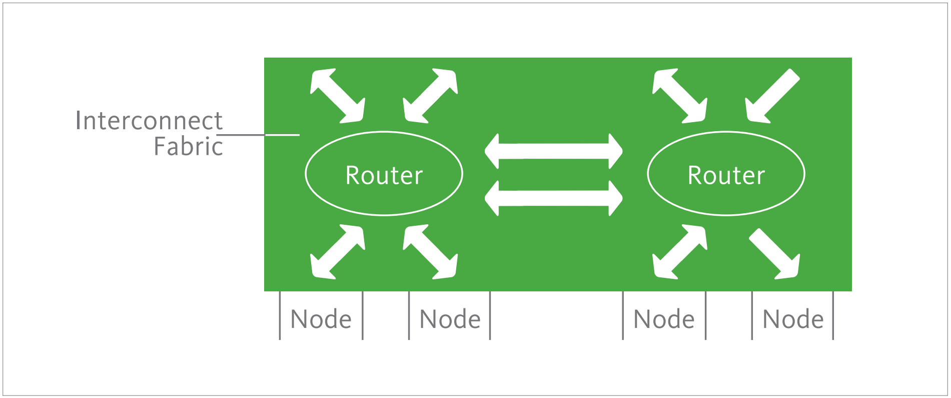

Figure 1: Typical system architecture with physically distributed memory.

In this discussion, we refer to two different system environments. The first is commodity clusters, typically based on a number of thin rackmount servers or blade servers with limited memory installed on each server. In order to access the memory on a different node, the application running on one node must communicate with the desired node where the data in the memory resides, and issue special commands to retrieve the data in the remote memory address. The memory on one node is not directly visible to the processing elements on a different node.

In the second system environment, all memory that resides on any node (or blade) is visible to any of the cores in that system. This is referred to as a coherent shared memory (CSM) system. All data within memory is available using direct access mechanisms, regardless of where the memory actually resides (local or remote). On the SGI UV system, each of the up to 2,560 cores can coherently access the maximum memory in the system, which is limited only by the address space on the Intel® Xeon® processor. The coherent memory addressable directly is not limited to that which resides on the physical board where the processor resides. Figure 2 shows these differences.

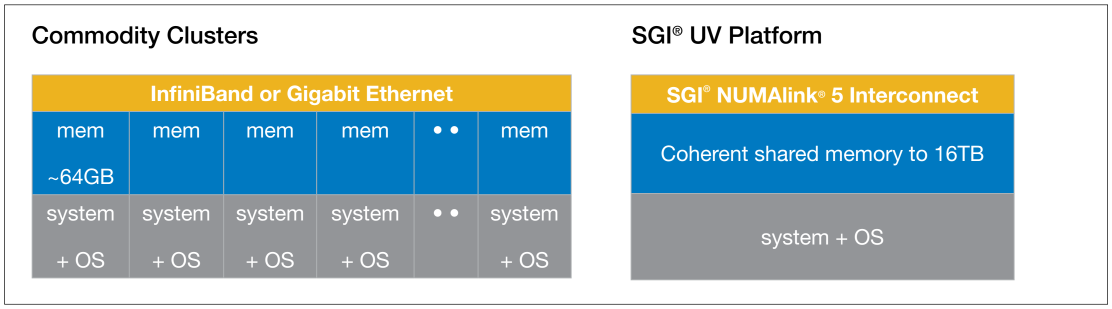

Figure 2: Distributed Memory System vs. Coherent Shared Memory System

2.2 Data Access Hierarchy

In recent years we have seen rapid improvements in microprocessor cycle times. However, the performance of data storage outside the immediate processor domain (the process registers and on chip caches) has not kept pace. In particular, main memory and I/O latencies have not improved nearly as rapidly as the speed at which the data is needed. Today, on-chip processor caches provide access to data with latencies on the order of one or two nanoseconds. These latencies tend to scale directly with clock frequency improvements and thus track those trends closely. This contrasts strongly with typical local (on-node) SDRAM memory, delivering data to the process or core with an access latency of perhaps 100 nanoseconds—roughly 100x slower. Corresponding reductions in bandwidth occur as you progress through the data hierarchy as well.

Data Hierarchy Layer	Latency	Normalized Access Times
L1 Cache	1.4 ns	1
L3 Cache	23 ns	16x
Local / Remote Memory	75 ns to 1 µsec	53x to 700x
Disk	2 ms	3.6x 10^6x

Table 1: Latency and resulting access times across the data hierarchy

These trends combine to form a tremendous gap between processor and memory performance. A generally accepted approach to mitigate the impact of this gap has been to rely upon larger processor cache subsystems in which successive levels of the cache trade off faster access times for larger capacity. Unfortunately, processor cache yields little benefit for applications that have random memory access patterns with very large memory footprints. Although cache sizes continue to grow with new generations of processors, data required by applications continues to grow as well, perhaps by even larger amounts. The performance gap is especially severe in multiprocessor servers where sharing data may require traversing multiple cache hierarchies, spanning thousands of processor clock cycles.

While much has been written about this “memory wall” (a term coined by Wulf and McKee in their paper “Hitting the Memory Wall: Implications of the Obvious” [1]), little has been written about the even more pronounced I/O wall. As shown in Table 1, the memory wall is about one to two orders-of-magnitude, the difference in cache and memory latencies. However, the latency difference between memory and disk—the I/O wall—can exceed four or more orders-of-magnitude. Huge productivity gains are achieved when an application's working data set fits entirely in main memory (as opposed to paging data in and out to disk). A familiar example of this is the common case of laptop or desktop PCs for which application performance (and therefore user productivity) is enhanced far more by installing additional memory and avoiding disk access than it is by upgrading to a faster processor.

Figure 3 summarizes the multiple order-of-magnitude differentials in latencies and bandwidth between levels within the data access hierarchy.

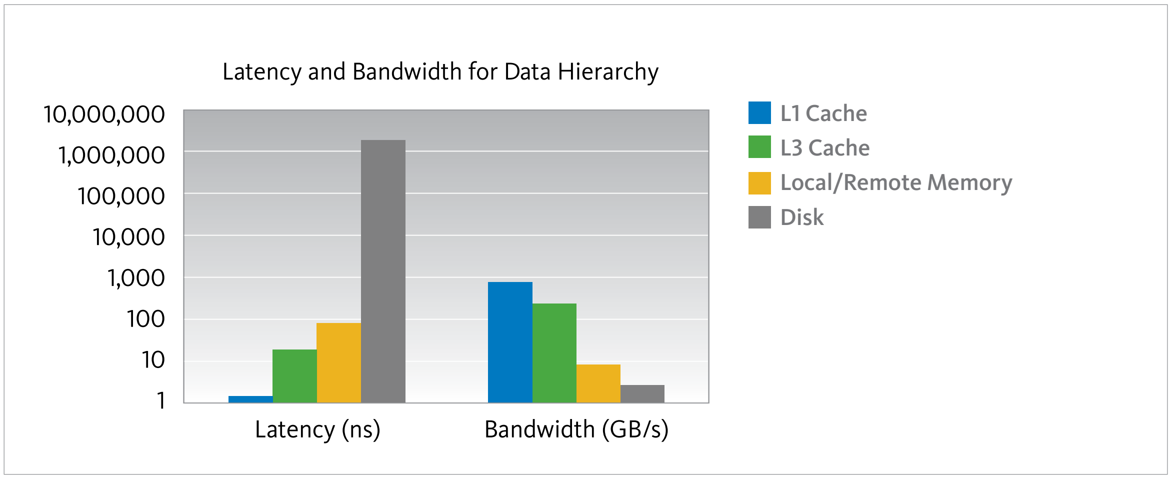

Figure 3: Bandwidth and latency worsen dramatically as data moves along the hierarchy from processor cache to disk storage.

3. SGI Shared Memory Architecture: Massive Shared Memory Resources

In recent years, the capacity of global and coherent shared memory systems which can feasibly be constructed has increased dramatically. This advance has been led by SGI with successive improvements to its system architecture with the NUMAlink® communication system. NUMAlink was introduced initially in the SGI Origin® product line and is now is in its fifth version. In addition, DRAM cost per bit has declined to the point where many terabytes of coherent shared memory is not only feasible, but more affordable.

A key to making use of CSM is a system architecture that is able to provide efficient direct processor load-store access to the entire address space. With the SGI coherent shared memory architecture, the SGI UV system couples the large physical address space of the Intel® Xeon® processor with a system interconnect capable of distributing that address space seamlessly across hundreds or even thousands of nodes and cores. This means any load instruction that a processor issues can read any address in the entire coherent shared memory space of a system (on any node) exactly as it would read data in memory on the same local node. No additional programming is needed at the application level.

In addition, the SGI system architecture as implemented on the SGI UV allows independent scaling of processors, memory capacity, memory bandwidth, interconnect bandwidth, I/O connectivity and I/O bandwidth.

3.1 Memory Scaling

Memory capacity is expanded by adding more memory DIMMs into the system. Since memory can be addressed from any processing core, by adding memory to any physical node, the entire system can use that added RAM. The Intel® Xeon® processor can address a maximum of 16TB of physical memory, with the 44 bits for physical addressing available. The SGI architecture allows for any core to access the total amount of memory in the system, rather than clusters that only allow for addressing of memory local to that server.

For one Intel® Xeon® processor series example, please consult:
http://www.intel.com/Assets/en_US/PDF/datasheet/323341.pdf

3.2 Advanced Features of SGI UV

The SGI UV platform is the culmination of years of focused effort, which has resulted in the fifth generation of shared memory systems based on the NUMAlink architecture. SGI worked closely with Intel in the development of the UV, as a tight integration was needed between the CPU manufacturer and the development of the "glue" that would enable coherent shared memory.

The blade-to-NUMAlink architecture of UV enables users to mix and match different CPU options to create a scalable system with the exact performance capabilities they require—up to a current maximum of 2,560 processor cores and up to 16TB of globally shared memory. Processing nodes in the UV design are provided by compute blades that contain two processor sockets per blade (supporting up to 10-core Intel® Xeon® processors) and 16 DIMM slots supporting a maximum of 256GB of memory per blade when 16GB DIMMs are used.

The SGI UV is available in three models:

- UV 10 – up to 40 cores and 1TB RAM
- UV 100 – up to 960 cores and 12TB RAM
- UV 1000 – up to 2,560 cores and 16TB RAM

System Category	Max. Support for Global Shared Memory
Typical x86 rackmount systems	128GB – 256GB
RISC systems (Power, SPARC)	2TB – 4TB
SGI UV	16TB to 8PB*

Table 2: Memory scaling comparisons for typical HPC platforms

* 16 Terabytes is the maximum amount of memory that SGI UV processors can directly access within the shared memory domain of a single instance of the operating system (single system image or SSI). However, the globally addressable memory (GAM) that is accessible via the GRU (Global Reference Unit) or MOE (MPI Offload Engine) mechanism in a UV system is much larger at 8 petabytes. Note that a shared memory domain in UV is a subset of the overall GAM space.

3.2.1 Flexible I/O Scaling

The architecture of SGI UV supports many I/O devices. Any processing core can access this flexible I/O as required by the application. By utilizing the standard PCIe slots available on each blade, a wide range of I/O capabilities can be accommodated. UV blades can easily connect to the SGI InfiniteStorage line of EBOD, RAID, SAN, NAS and tape storage solutions. As more blades are installed into the system, the I/O capacity and performance increase.

3.2.2 Flexible Interconnect Scaling (Connectivity, Latency and Bandwidth)

The current SGI interconnect is NUMAlink 5 and it provides a raw single link aggregate transfer rate of 15GB/s (7.5 in each direction). Commodity clusters today use links with aggregate performance ranging from 4GB/s to 12GB/s (http://en.wikipedia.org/wiki/InfiniBand). NUMAlink 5 also includes the resources and protocols necessary to efficiently propagate coherent memory traffic across large system domains. Finally, NUMAlink interconnect technology delivers the industry's fastest hardware latencies: direct memory access to remotely located memory on the order of nanoseconds versus microseconds for cluster architectures. NUMAlink interconnect technology provides the ability to independently scale both endpoint connectivity and system-wide bandwidth, allowing individual systems to be optimized for particular applications which have differing requirements of bandwidth between nodes and mixes of various endpoint resources.

4. Benefits of Coherent Shared Memory

4.1 Massive In-Core Computation by Design

A CSM architecture allows much larger and more detailed models of physical systems to be entirely memory resident. Consider, for example, modeling not only the turbine blades of an aircraft engine, but the complete engine. Lacking adequately sized coherent shared memory, one must employ one or more compromise strategies to make the computation feasible. This may include:

- Reducing mesh resolution: the number of data points used to represent the physical entity being modeled
- Breaking the problem down: managing it as a set of smaller pieces, perhaps dealing with intermediate results stored in scratch files on disk
- Reducing numerical precision: dropping to 32-bit representations
- Approximated emulation of component subsystem behaviors: often using algorithmic models and artificially constrained properties

All such strategies represent potentially significant compromises in computational accuracy, performance and productivity.

4.2 Massively Memory-Mapped I/O

For applications that are bound by random I/O accesses on large data sets, up to 10,000x performance increases can be gained by bringing the entire dataset into main memory. This becomes even more compelling when considering the price difference between memory versus disk. Current price trends for memory and disk technology predict only, at most, two orders of magnitude between these storage layers. This alone yields a price-performance advantage over disk I/O-based operations of 100x.

For example, CSM can be exploited to dramatically improve database performance. Just as the move from storing databases on magnetic tape to magnetic disk allowed much higher performance through direct random access, CSM enables similar orders-of-magnitude improvements in database performance. A large, CSM system can entirely eliminate the need to page data in from disk.

4.3 Highly Efficient Message Passing

A large CSM system is obviously well suited to shared memory applications. However, it can also deliver superior performance for distributed memory applications (for example MPI message passing) where all communications and data are explicitly defined and allocated. Moreover, when the ability to directly load and store to remote memory is correctly exploited, most software overhead is eliminated, yielding substantially reduced effective latencies. SGI UV systems utilizing the NUMAlink interconnect demonstrate superior efficiency for MPI. To go along with the new UV architecture, SGI has developed the Message Passing Interface Offload Engine (MOE), which accelerates the Mesasge Passing Toolkit (MPT). The MPT intercepts MPI calls and uses the acceleration available on the UV_HUB ASIC. This can have a significant effect on the performance of a distributed application running on an SMP system.

4.4 Arbitrary Scaling of Problem Size

Applications ported to systems with a distributed memory organization (globally shared) are often optimized by decomposing the input datasets and carefully placing the pieces in physical memory local to the processors that will most often interact with them. Consider an input dataset growing such that the partitioned pieces exceed the capacity of the individual nodes. On a clustered system, a major repartitioning, or possibly a complete change of algorithm, may be required in order to maintain a reasonable level of performance. But with a well-designed large CSM system, the latency and bandwidth penalties for fetching off-node data is comparatively minimal (whether addressing the memory directly or using message passing access). This means that simply by adding memory-only resources, one can run the same application without significant modification and still obtain the needed performance and correct computational behavior.

4.5 Efficient Utilization of Memory Resources

Coherent shared memory unifies system memory resources that are, by definition, fragmented when using a distributed memory system approach. Porting certain applications for execution on a cluster requires that some amount of the dataset be replicated, often onto each cluster node in the system. This wastes system memory capacity and effectively increases the cost per bit for memory resident data. Data replication results in multiple data copies that must be managed, increasing the burden on the system or application. A CSM system implicitly avoids these complexities.

For instance, in a distributed ray tracing application running on a commodity cluster, each node in the cluster must have the entire model replicated locally or the application will suffer unacceptable degradation in latency and bandwidth. Once the model size exceeds the size of a given node's main memory, no further scaling of problem size can be obtained on that system (at least not without heroic efforts to implement complicated data caching schemes). On a large CSM system, the effective cost-per-bit economy and the ease of scaling datasets to essentially arbitrary size yield a substantial advantage over other architectures.

4.6 Greatly Simplified Load Balancing

For many real-world parallel applications, processing resources are wasted because of load imbalances. With coherent shared memory, it's a simple matter to direct a processor that has finished one task to start on another since the data associated with all such tasks are accessed via a single common address space. In contrast, load balancing on a cluster requires data for a new task to be explicitly copied to the node in question before work can start, creating significant overhead in managing copies of data and results.

4.7 Efficient Development and Administration for Reduced TCO

When developing new codes, use of simpler programming models makes it easier to experiment with different algorithms. Shared memory programming provides a simple computational paradigm upon which to rapidly prototype and evaluate application codes. Many SGI UV users do some level of code development, benefiting from the ease of use afforded in up to 2,560 processor cores in a single, unified development and operating environment. Development can be done on a small system, down to a laptop, and then the identical binary can be moved to the large system and run at scale.

Further, managing a collection of computer resources within the context of a single instance of an operating system represents a fundamental reduction in costs associated with system administration. A recent study of total cost of ownership for high-end computing systems (Muzio and Walsh, 2003) estimates that the equivalent of one full-time person is required to administer every 128[2] nodes (in this case, "node" refers to a specific instance of an operating system). For example, a modern 256 processor cluster of 128 dual-socket, quad-processor nodes would yield a system having a peak compute power of perhaps 12 teraflops and would require one full time person just for node administration. In contrast, the UV system delivers 50% more computing power in a single node of 2,560 processors and 18+ teraflops that can be configured as a single operating system image, requiring minimal administrative resources.

5. Programming Models

A number of programming models and APIs are supported for the SGI UV. With its massively large coherent memory design, various programming environments can be supported:

- SHMEM™ – A shared memory model allows for different processes to read and write data to a large pool of memory, where other applications can do the same. OpenMP is an example of a SHMEM model.
- PGAS is the acronym for the "Partitioned Global Address Space." Languages such as Unified Parallel C (UPC) and CoArray Fortran (CAF) allow for the use of a large set of memory, which can be used by multiple threads and processes.
- Message Passing Interface (MPI) – In this programming model, the application developer has control over both data and work distribution. However, it is important to understand when an application should use the MPI API and when the communication overhead is too much compared to the data transfer. SGI offers an MPI implementation, as part of the SGI ProPack™, that has been tuned to the UV system.
- In addition, there are hybrid models, such as using MPI for between-node communication and OpenMP for communication within the node.

6. Use Cases and New Possibilities

The SGI UV, with its industry leading shared memory addressability and very high core count in a single system image, opens up a wide range of applications that can are easy to develop on smaller systems, and then easy to scale when deployed.

6.1 Fraud Detection/Cyber Security

An example of where very large shared memory systems would be of value is in the area of fraud detection and cyber security. Using historical information and real time data feeds, which are accessible by a number of applications designed to detect fraud, a large memory system will excel by keeping significant amounts of data in memory. In the cyber security field, terabytes of incoming data may be needed from many systems with historical patterns, as well to detect and act on cyber security breaches, in real time. For example, as data is ingested by the system from a variety of clients (e.g., point-of-sale terminals, kiosks, ticket machines), algorithms could be run to determine anomalies from known patterns. A number of applications could be run on the same data sitting in memory to determine in real time if fraud is evident or in process. Applications which rely on fast access to large amounts of data that are not easily partitioned lend themselves to multi-terabyte memory systems such as UV.

6.2 BioInformatics

Simulations of molecular interaction involving thousands of discrete units, each with many parameters and properties, require significant computing power to arrive at a certain state. By using a large computing platform with direct access to large numbers of processors and memory, more effective and global calculations can be performed, arriving at solutions more quickly. In addition, large databases can be brought into the massive amount of system memory, and applications can individually access the necessary data. A Europe-based pharmaceutical research institute has developed a genomics search and matching algorithm that produces results up to 1,000x faster than the commonly used BLAST application. This algorithm relies on 192GB of memory available, more than lower end two-socket servers.

The popular code NWChem is routinely run at one SGI customer site on only four processors of an SGI shared memory system, but using 80% of the 2TB of globally shared memory configured in that system. While execution may not scale efficiently across more than a modest handful of processors, big gains can be had simply by exploiting additional memory capacity. With the flexibility that SGI UV offers, applications can take advantage of large core counts, large memory addressability, or both.

6.3 Data Analytics

Many organizations are trying to cope with massive amounts of data on a wide range of topics, from determining sales patterns to administering material distribution plans. With the very large memory capacity of SGI UV, much more data can be stored for real time analytics in order to find patterns, simulate processes and make decisions. Rather than applications working on only a fraction of the data available and then segmenting the analysis and algorithms, by using the UV large memory systems much more data can be analyzed with more complete and sophisticated algorithms. With real time decision making using full data sets, new insights can be explored and better business decisions made.

6.4 Real-Time Interaction with Massive Datasets

A leading oil and gas exploration company is continuously challenged to deliver larger and larger seismic datasets to interpreting geoscientists. As datasets become larger and more precise, the geoscientists require larger memory systems in order to understand the information. In some cases geoscientists need to be able visualize and interact with more than one half of a terabyte of seismic data in real time.

Another example is real-time transaction and analytics processing. A leading casino management firm leverages an SGI shared memory system in managing their operations. They summarize a gaming day into multiple Oracle tables so users can access Oracle Discover for marketing analysis and slot analysis. The result is a mixed environment with online transaction processing and online analytical processing. Competitive systems need multiple servers to do this, but this SGI customer can do it all with the bandwidth of a single SGI shared memory server and avoid the hassle of trying to keep several systems synchronized.

6.5 Memory Resident Databases such as Oracle® TimesTen

In-memory databases, such as Oracle TimesTen, are increasing the options available to leverage SGI UV large CSM systems. In a customer benchmark, a UV system (with 1TB of memory) using Oracle TimesTen scaled to handle an in-memory database with more than 10 billion rows of data. Ingest rates were an order of magnitude faster than the performance target, and query person rates were one thousand times faster. Join person/order rates were a remarkable million times faster than the performance target. The overall search performance of the Oracle TimesTen database on the SGI UV system was over one thousand times faster than the customer target, far exceeding the capabilities of any other server on the market today.

6.6 Dataflow Applications

Certain applications are architected such that a number of processes can work on the same large data stream. An example is seen in local protein sequence alignment where millions of combinations are analyzed using the Smith-Waterman algorithm for performing this pattern matching. The high-performance algorithm is implemented using the Pervasive DataRush™ parallel dataflow platform, which developers can rapidly exploit using its extensible library of operators that seamlessly scale on commercially available multicore hardware. With the Pervasive DataRush software, almost one trillion cell updates per second were achieved on an SGI UV 1000 system with 384 cores, demonstrating unmatched performance and a world record for this type of application.

7. Summary

The advantages of a global shared memory architecture are evident both in common high performance computing workloads and in grand challenge breakthroughs. Looking at the order-of-magnitude latency and bandwidth differences along the data access pathway demonstrates the obvious benefit of keeping datasets and simulations fully resident in system memory. With the additional benefits of simplified load balancing, development and system administration, the value of this versatile architecture delivers true return on investment. SGI UV technology makes it possible to build systems with massive coherent memory that scales up to 16TB. The value of these systems has already been demonstrated across a wide range of problem types. Future achievements with very large coherent shared memory architectures are limited only by the imagination of application developers and users.

More Information

sgi.com/uv

[1] http://www.cs.virginia.edu/papers/Hitting_Memory_Wall-wulf94.pdf

[2] http://www.cug.org/5-publications/proceedings_attendee_lists/2003CD/S03_Proceedings/Pages/Authors/Muzio.pdf

Global Sales and Support: sgi.com/global