ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

Basic Algorithm Implementation on High Performance Computing Systems

Venkatesh K[1], Siva Parvathi V[2], Phani Praveen S[3], Rajesh V[4]

Department of Computer Science & Engineering, PVP Siddhartha Institute of Technology, Vijayawada, India[1, 2, 3, 4]

ABSTRACT: Now a days the need to implement complex algorithms is at acme level. The ever growing business data or social media or other sources play huge importance on their representation. For efficient implementation and fast retrieval, complex systems are required. In this paper, we enlighten the overview of High Performance Computing Systems and implementation details of simple algorithm. Later, discussed the result analysis of dual core and HPC systems.

I. INTRODUCTION

A High performance computing system is a computer with a high-level computational capacity compared to a general-purpose computer. Performance of a High performance computers is measured in floating-point operations per second (FLOPS) instead of million instructions per second (MIPS). As of 2015, there are High performance computers which can perform up to quadrillions of FLOPS. High-performance computing (HPC) is the use of parallel processing for running advanced application programs efficiently, reliably and quickly [1]. The term applies especially to systems that function above a teraflop or 10^{12} floating-point operations per second. The term HPC is occasionally used as a synonym for supercomputing, although technically a High performance computers is a system that performs at or near the currently highest operational rate for computers. Some High performance computerss work at more than a pet flop or 10^{15} floating-point operations per second. The point of having a high performance computer is so that the individual nodes can work together to solve a problem larger than any one computer can easily solve [2]. And, just like people, the nodes need to be able to talk to one another in order to work meaningfully together. Just like ordinary desktop or laptop, HPC cluster will not run without software. Two of the most popular choices in HPC are Linux and Windows. Linux currently dominates HPC installations, but this in part due to HPC's legacy in supercomputing, large scale machines, and UNIX. High-performance computing play an important role in the field of computational science, and are used for a wide range of computationally intensive tasks in various fields, including quantum mechanics, weather forecasting, climate research, oil and gas exploration, molecular modelling computing the structures and properties of chemical compounds, biological macromolecules, polymers, and crystals, and physical simulations such as simulations of the early moments of the universe, airplane and spacecraft aerodynamics, the detonation of nuclear weapons, and nuclear fusion [3]. Throughout their history, they have been essential in the field of cryptanalysis. Systems with massive numbers of processors generally take one of two paths: in one approach, hundreds or thousands of discrete computers distributed across a network devote some or all of their time to solving a common problem; each individual computer receives and completes many small tasks, reporting the results to a central server which integrates the task results from all the clients into the overall solution. In another approach, thousands of dedicated processors are placed in proximity to each other; this saves considerable time moving data around and makes it possible for the processors to work together (rather than on separate tasks), for example in mesh and hypercube architectures. While the High performance computers of the 1980s used only a few processors, in the 1990s, machines with thousands of processors began to appear both in the United States and Japan, setting new computational performance records. Fujitsu's Numerical Wind Tunnel High performance computers used 166 vector processors to gain the top spot in 1994 with a peak speed of 1.7 gigaflops (GFLOPS) per processor. The Hitachi SR2201 obtained a peak performance of

600 GFLOPS in 1996 by using 2048 processors connected via a fast three-dimensional crossbar network. The Intel Paragon could have 1000 to 4000 Intel i860 processors in various configurations, and was ranked the fastest in the world in 1993. The Paragon was a MIMD machine which connected processors via a high speed two dimensional mesh, allowing processes to execute on separate nodes, communicating via the Message Passing Interface [4].

The most common users of HPC systems are scientific researchers, engineers and academic institutions. Some government agencies, particularly the military, also rely on HPC for complex applications. High-performance systems often use custom-made components in addition to so-called commodity components [5]. As demand for processing power and speed grows, HPC will likely interest businesses of all sizes, particularly for transaction processing and data warehouses. An occasional trend might use an HPC system to satisfy an exceptional desire for advanced technology [6].

II. HIGH PERFORMANCE COMPUTERS HARDWARE AND SOFTWARE

This section describes the Hardware and Software components of High performance computers.

A. SYSTEM HARDWARE

Approaches to High performance computers architecture have taken dramatic turns since the earliest systems were introduced in the 1960s. Early High performance computers architectures pioneered by Seymour Cray relied on compact innovative designs and local parallelism to achieve superior computational peak performance [7]. However, in time the demand for increased computational power ushered in the age of massively parallel systems [8].
While the High performance computers of the 1970s used only a few processors, in the 1990s, machines with thousands of processors began to appear and by the end of the 20th century, massively parallel High performance computers with tens of thousands of "off-the-shelf" processors were the norm [9]. High performance computers of the 21st century can use over 100,000 processors (some being graphic units) connected by fast connections. Fig. 1 shows the architecture of High Performance Computers.

ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

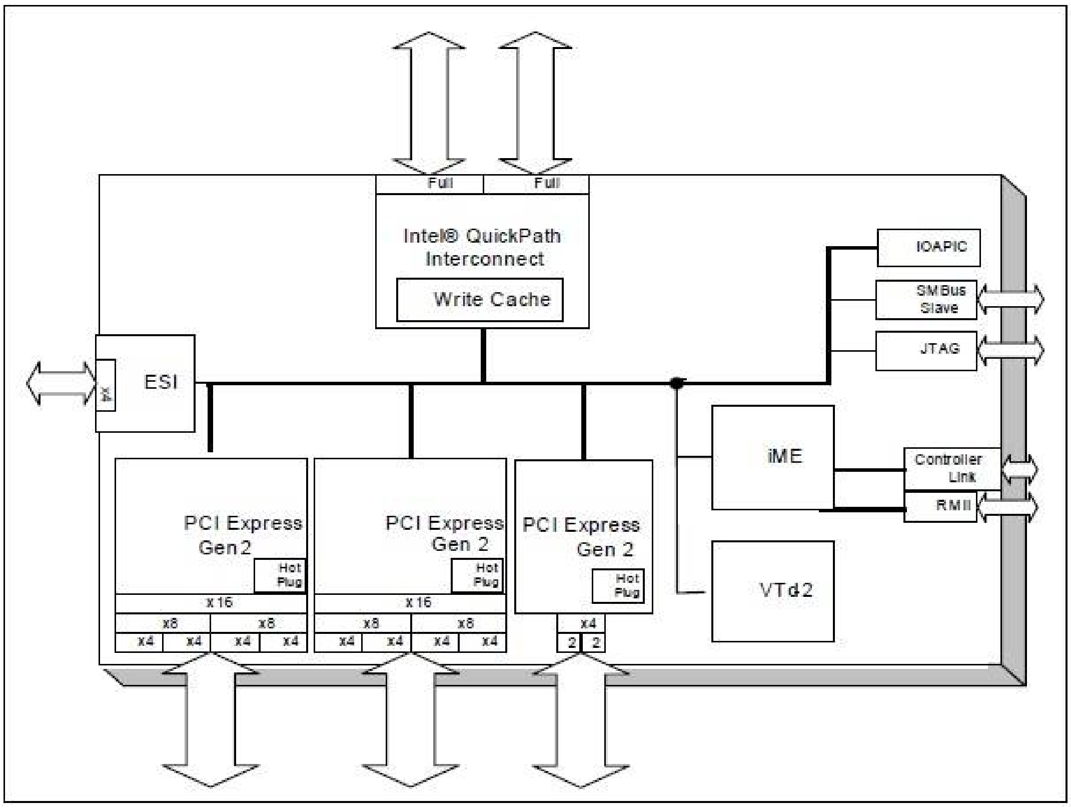

Fig. 1. Architecture of High Performance Computers.

The Connection Machine CM-5 High performance computers are a massively parallel processing computer capable of many billions of arithmetic operations per second.
Throughout the decades, the management of heat density has remained a key issue for most centralized High performance computers. The large amount of heat generated by a system may also have other effects, e.g. reducing the lifetime of other system components. There have been diverse approaches to heat management, from pumping Fluorine through the system, to a hybrid liquid-air cooling system or air cooling with normal air conditioning temperatures. A typical supercomputer consumes large amounts of electrical power, almost all of which is converted into heat, requiring cooling. For example, Tianhe-1A consumes 4.04 megawatts (MW) of electricity. The cost to power and cool the system can be significant, e.g. 4 MW at $0.10/kWh is $400 an hour or about $3.5 million per year [10].

Heat management is a major issue in complex electronic devices, and affects powerful computer systems in various ways. The thermal design power and CPU power dissipation issues in supercomputing surpass those of traditional computer cooling technologies. The supercomputing awards for green computing reflect this issue [11]. The packing of thousands of processors together inevitably generates significant amounts of heat density that need to be dealt with. The Cray 2 was liquid cooled, and used a Fluor inert "cooling waterfall" which was forced through the modules under pressure. However, the submerged liquid cooling approach was not practical for the multi-cabinet systems based on off-the-shelf processors, and in System X a special cooling system that combined air conditioning with liquid cooling was developed in conjunction with the Liebert company. The following Fig. 2 shows the cores of High performance Computers [12].

ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

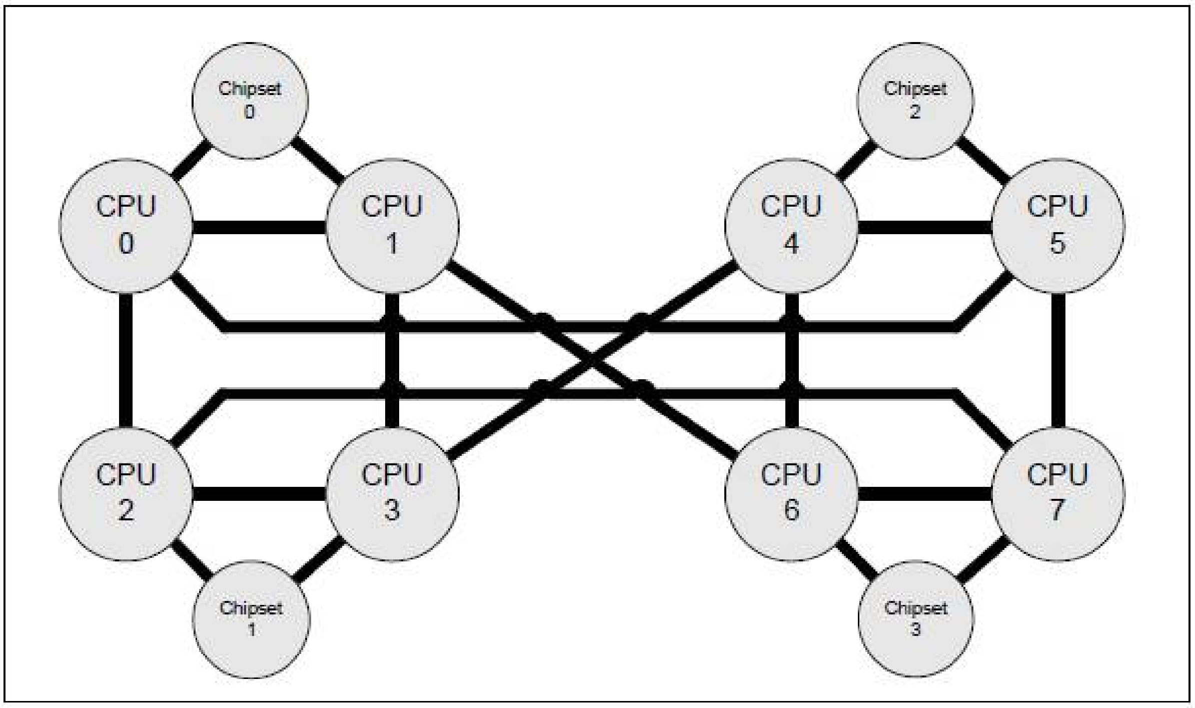

Fig. 2. Cores of High Performance Computers.

In the Blue Gene system, IBM deliberately used low power processors to deal with heat density. On the other hand, the IBM Power 775, has closely packed elements that require water cooling. The IBM Aquasar system, on the other hand uses hot water cooling to achieve energy efficiency, the water being used to heat buildings as well. The energy efficiency of computer systems is generally measured in terms of "FLOPS per watt". IBM's Roadrunner operated at 3.76 MFLOPS/W. The Blue Gene/Q reached 1,684 MFLOPS/W. Because copper wires can transfer energy into a supercomputer with much higher power densities than forced air or circulating refrigerants can remove waste heat, the ability of the cooling systems to remove waste heat is a limiting factor [13]. Many existing supercomputers have more infrastructure capacity than the actual peak demand of the machine designers generally conservatively design the power and cooling infrastructure to handle more than the theoretical peak electrical power consumed by the supercomputer. Designs for future supercomputers are power-limited the thermal design power of the supercomputer as a whole, the amount that the power and cooling infrastructure can handle, is somewhat more than the expected normal power consumption, but less than the theoretical peak power consumption of the electronic hardware [14].

B. SYSTEM SOFTWARE

High Performance Computers have undergone major transformations, based on the changes in supercomputer architecture. While early operating systems were custom tailored to each supercomputer to gain speed, the trend has been to move away from in-house operating systems to the adaptation of generic software such as Linux [15]. Since modern massively parallel supercomputers typically separate computations from other services by using multiple types of nodes, they usually run different operating systems on different nodes, using a small and efficient lightweight kernel such as CNK or CNL on compute nodes, but a larger system such as a Linux-derivative on server and I/O nodes. While in a traditional multi-user computer system job scheduling is, in effect, a tasking problem for processing and peripheral resources, in a massively parallel system, the job management system needs to manage the allocation of both computational and communication resources, as well as gracefully deal with inevitable hardware failures when tens of thousands of processors are present. Although most modern supercomputers use the Linux operating system, each

ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

manufacturer has its own specific Linux-derivative, and no industry standard exists, partly due to the fact that the differences in hardware architectures require changes to optimize the operating system to each hardware design [16].

The parallel architectures of supercomputers often dictate the use of special programming techniques to exploit their speed. Software tools for distributed processing include standard APIs such as MPI and PVM, VTL, and open source-based software solutions such as Beowulf. In the most common scenario, environments such as PVM and MPI for loosely connected clusters and OpenMP for tightly coordinated shared memory machines are used. Significant effort is required to optimize an algorithm for the interconnect characteristics of the machine it will be run on; the aim is to prevent any of the CPUs from wasting time waiting on data from other nodes. GPGPUs have hundreds of processor cores and are programmed using programming models such as CUDA or OpenCL. Moreover, it is quite difficult to debug and test parallel programs. Special techniques need to be used for testing and debugging such applications. The following Fig. 3 shows the constructs of High Performance Computers [17].

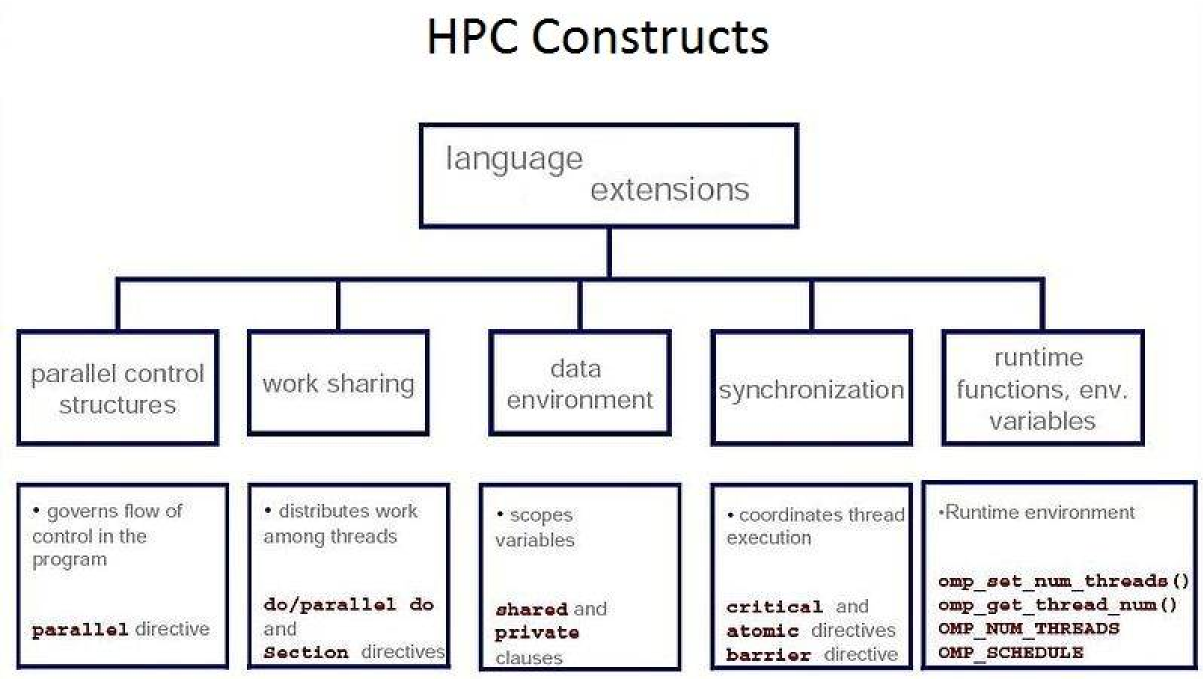
HPC Constructs
language extensions
parallel control structures
work sharing
data environment
synchronization
runtime functions, env. variables
• governs flow of control in the program
parallel directive
• distributes work among threads
do/parallel do and Section directives
• scopes variables
shared and private clauses
• coordinates thread execution
critical and atomic directives barrier directive
•Runtime environment
omp_set_num_threads()
omp_get_thread_num()
OMP_NUM_THREADS
OMP_SCHEDULE

Fig. 3. HPC Constructs.

III. SAMPLE ALGORITHM IMPLEMENTATION ON HPC SYSTEM

Two factors are creating major new challenges at the interface between numerical analysis and high performance computing (HPC). The use of more realistic but complicated mathematical models is leading to problems for which new algorithms need to be developed for efficient solution or, in many cases, solution at all [18]. Meanwhile, recent developments in computer architectures the drive to multi-core chips and the increasing use of GPUs and FPGAs as accelerators are affecting the ability of current algorithms and software to fully exploit the hardware. The long-standing lack of investment in the development of high quality mathematical software exacerbates these problems. The net effect is that scientists are unable to make the best use of computational resources and hence are unable to tackle problems of the size that the hardware potentially allows. For all these reasons, new algorithms need to be developed that employ novel mathematical, algorithmic and coding techniques [19]. The purpose of this network is to provide a focus for a new collaboration between numerical analysts, computer scientists and developers and users of software and HPC within the nodes, supported by the necessary administrative organization. The network will build a new interdisciplinary community at the numerical algorithms/HPC interface and thereby provide added value to existing funded research at the nodes in numerical algorithms and in HPC and its applications [20].

ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

The algorithm we implemented on HPC is as follows, which computes sum of elements in an array.

ALGORITHM: *sum(A)*

```
Start
1 if |A| = 1 then return A[0]
2 else return sum({A[2i] + A[2i + 1] : i ∈ [0..|A|/2)})
The work and depth for this algorithm are given by the recurrences
        $W(n) = W(n/2) + O(n)$        (1)
        $D(n) = D(n/2) + O(1)$        (2)
which have solutions $W(n) = O(n)$ and $D(n) = O(\log n)$
end
```

This algorithm can also be expressed without recursion using a while loop, but the recursive version as indicated in (1) & (2) for shadows the recursive algorithm for the scan function. the algorithm only works on sequences that have lengths equal to powers of 2. Removing this restriction is not difficult by checking if the sequence is of odd length and separately adding the last element in if it is. This algorithm can also easily be modified to compute the "sum" using any other binary associative operator in place of +. For example the use of max would return the maximum value of in sequence.

A) Efficiency of the HPC over Single Core System

The algorithm is successfully deployed on dual core system as well as HPC the following are the results. HPC has produced the better results than dual core system.

IV. CONCLUSION

In this paper, we implemented a basic algorithm on dual core as well as HPC system. The performance of dual core system is far lesser than HPC system. The implementation process is similar in both the systems. Implementations of complex approaches give better results using HPC. Multi core systems always produce efficient results. HPC platform provides a better approach to implement graphic level applications.

REFERENCES

[1] Bacon, David F., Susan L. Graham, and Oliver J. Sharp. "Compiler transformations for high-performance computing." *ACM Computing Surveys (CSUR)* 26.4 (1994): 345-420.
[2] Fan, Zhe, et al. "GPU cluster for high performance computing." *Proceedings of the 2004 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2004.
[3] Gropp, William, et al. "A high-performance, portable implementation of the MPI message passing interface standard." *Parallel computing* 22.6 (1996): 789-828.
[4] Dowd, Kevin. *High performance computing*. O'Reilly & Associates, Inc., 1993.
[5] Huang, Wei, et al. "A case for high performance computing with virtual machines." *Proceedings of the 20th annual international conference on Supercomputing*. ACM, 2006.
[6] Tang, Qinghui, Sandeep Kumar S. Gupta, and Georgios Varsamopoulos. "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach." *IEEE Transactions on Parallel and Distributed Systems* 19.11 (2008): 1458-1472.
[7] Mergen, Mark F., et al. "Virtualization for high-performance computing." *ACM SIGOPS Operating Systems Review* 40.2 (2006): 8-11.
[8] Buyya, Rajkumar. "High performance cluster computing: Architectures and systems (volume 1)." *Prentice Hall, Upper SaddleRiver, NJ, USA* 1 (1999): 999.
[9] Taubenblatt, Marc. "Optical interconnects for high performance computing."*Optical Fiber Communication Conference*. Optical Society of America, 2011.
[10] Katopis, George A. "Delta-I noise specification for a high-performance computing machine." *Proceedings of the IEEE* 73.9 (1985): 1405-1415.
[11] Plaza, Antonio J., and Chein-I. Chang, eds. *High performance computing in remote sensing*. CRC Press, 2007.
[12] Kindratenko, Volodymyr V., et al. "GPU clusters for high-performance computing." *2009 IEEE International Conference on Cluster Computing and Workshops*. IEEE, 2009.
[13] Younge, Andrew J., et al. "Analysis of virtualization technologies for high performance computing environments." *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. IEEE, 2011.
[14] Ayres, Daniel L., et al. "BEAGLE: an application programming interface and high-performance computing library for statistical phylogenetics." *Systematic biology* (2011): syr100.

ISSN(Online) : 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

[15] Armstrong, Rob, et al. "Toward a common component architecture for high-performance scientific computing." *High Performance Distributed Computing, 1999. Proceedings. The Eighth International Symposium on*. IEEE, 1999.
[16] Barker, Kevin J., et al. "On the feasibility of optical circuit switching for high performance computing systems." *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2005.
[17] Tezduyar, T., et al. "Flow simulation and high performance computing."*Computational Mechanics* 18.6 (1996): 397-412.
[18] Yoo, Richard M., et al. "Performance evaluation of Intel® transactional synchronization extensions for high-performance computing." *2013 SC-International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*. IEEE, 2013.
[19] Giunta, Giulio, et al. "A GPGPU transparent virtualization component for high performance computing clouds." *European Conference on Parallel Processing*. Springer Berlin Heidelberg, 2010.
[20] Le, Kien, et al. "Reducing electricity cost through virtual machine placement in high performance computing clouds." *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2011.