Cloud Versus In-house Cluster: Evaluating Amazon Cluster Compute Instances for Running MPI Applications

Yan Zhai, Mingliang Liu, Jidong Zhai
Department of Computer Science and Technology, Tsinghua university
{zhaiyan920,liuml07,zhaijidong}@gmail.com

Xiaosong Ma
Department of Computer Science, North Carolina State University; CSMD, Oak Ridge National Laboratory
ma@csc.ncsu.edu

Wenguang Chen
Department of Computer Science and Technology, Tsinghua university
cwg@tsinghua.edu.cn

ABSTRACT

The emergence of cloud services brings new possibilities for constructing and using HPC platforms. However, while cloud services provide the flexibility and convenience of customized, pay-as-you-go parallel computing, multiple previous studies in the past three years have indicated that cloud-based clusters need a significant performance boost to become a competitive choice, especially for tightly coupled parallel applications.

In this work, we examine the feasibility of running HPC applications in clouds. This study distinguishes itself from existing investigations in several ways: *1*) We carry out a comprehensive examination of issues relevant to the HPC community, including performance, cost, user experience, and range of user activities. *2*) We compare an Amazon EC2-based platform built upon its newly available HPC-oriented virtual machines with typical local cluster and supercomputer options, using benchmarks and applications with scale and problem size unprecedented in previous cloud HPC studies. *3*) We perform detailed performance and scalability analysis to locate the chief limiting factors of the state-of-the-art cloud based clusters. *4*) We present a case study on the impact of per-application parallel I/O system configuration uniquely enabled by cloud services. Our results reveal that though the scalability of EC2-based virtual clusters still lags behind traditional HPC alternatives, they are rapidly gaining in overall performance and cost-effectiveness, making them feasible candidates for performing tightly coupled scientific computing. In addition, our detailed benchmarking and profiling discloses and analyzes several problems regarding the performance and performance stability on EC2.

SC'11, November 12-18, 2011, Seattle, Washington, USA
2011 ACM 978-1-4503-0771-0/11/11 ...$10.00.

1. INTRODUCTION

Cloud computing platforms have gained significant popularity in the past several years, especially among small businesses who view cloud services a flexible, powerful, convenient, and cost-effective alternative to owning and managing their own computing infrastructure. In the HPC (High Performance Computing) community, on the other hand, a similar trend of paradigm shift has not yet established. While IaaS (Infrastructure as a Service) clouds easily enable users to acquire a set of nodes and set up a virtual cluster, supercomputers (mainly located in large research laboratories, supercomputing centers, and universities) and small- to medium-sized "local" clusters continue to be the overwhelming mainstream platforms for parallel program developers and users.

There have been several studies to evaluate the promise of cloud platforms for HPC users since 2008 [10,12,14,19–21,26, 30, 34, 38]. The consensus reached appears to be that cloud-based clusters need a significant (an-order-of-magnitude or more) performance improvement to become a competitive choice for MPI-style parallel applications. However, such results were obtained before Amazon EC2, the leading IaaS provider and the most evaluated cloud platform, released *cluster compute instances (CCIs)* in July 2010 [2, 13]. The CCI platform was designed to explicitly target HPC applications with dedicated physical node allocation, powerful CPUs, and improved interconnection. With a test conducted by Lawrence Berkeley National Laboratory researchers, a 7040-core run placed Amazon CCIs at the 231st on the most recent Top500 [35] list (November 2010).

Does this change the landscape of running tightly coupled parallel applications on clouds? The answer is not obvious, as it is well known that Linpack numbers do not translate into application performance, which depends on the composition and pattern of computation, communication, and I/O within each individual parallel application.

Besides performance, there are many more factors involved in HPC users' choice of platforms. Examples include cost, performance stability, ease of management and job submission, capability of interactive job execution, instant reconfiguration of clusters, performance isolation between concurrently running jobs, and the impact of root level access in HPC related research and development activities. These

issues have hardly been assessed in existing cloud HPC studies.

Considering that current commercial and research clouds offer users only a small number of instances for contructing virtual clusters, we search for the answer to a more specific question: *with the new cloud infrastructure (such as the Amazon EC2 CCIs) specifically targeting HPC, does cloud computing become a viable alternative to owning and using small- or medium-scale clusters?* We focus on tightly-coupled parallel programs for two reasons. First, such workloads are the mainstream consumers of traditional HPC resources. Second, this type of parallel computing is more challenging for public clouds and is therefore where the value and potential of cloud computing are mostly undecided. In contrast, multiple existing efforts have demonstrated the competitiveness and cost-effectiveness of using public clouds for loosely-coupled bags of tasks [18] and workflows [9, 15, 20, 23, 37].

To answer the above question, we conduct a comprehensive examination of issues related to running tightly coupled parallel applications on commercial cloud platforms. Our qualitative discussion is accompanied by extensive performance benchmarking and detailed quantitative performance/cost analysis. This work distinguishes itself from past cloud HPC evaluation work in the following aspects:

- We perform comprehensive benchmarking to assess the newly available HPC-oriented Amazon EC2 CCI platform, with experiments to assess its computation, communication, and I/O capability. In addition, we monitor and report performance variability during our benchmarking. Compared to prior published results, our measurements reveal a more positive picture on the performance and scalability of running tightly coupled parallel programs on public clouds.
- We perform detailed performance analysis through communication profiling to locate the sources of scalability problems on EC2 and use a local InfiniBand-connected cluster with similar compute hardware as a reference system. Our investigation brings several interesting observations regarding parallel program execution on the EC2 clouds, such as that communication may induce computation load imbalance and that latency, rather than bandwidth, is the major limiting factor contributing to the inferior communication performance on EC2.
- To our knowledge, this is the first study that evaluates parallel I/O with different file system setup for MPI programs on cloud platforms. Our experiments demonstrate that the unique configurability offered by cloud systems can produce significant performance impact on I/O-intensive applications.
- Based on our performance benchmarking results and the acquisition/maintenance cost for our reference cluster, we conduct a quantitative analysis to compare the cost of relying on public clouds vs. owning and using in-house clusters.

The rest of the paper is organized as follows. Section 2 gives an overview and qualitative discussion on issues related to running tightly coupled parallel applications on clouds and local clusters. Section 3 presents performance benchmarking results obtained from Amazon EC2 CCIs and our reference cluster, as well as detailed performance and scalability analysis. Section 4 gives a quantitative cost analysis, based on the relative performance between EC2 and the reference cluster. Section 5 discusses related work and finally, Section 6 concludes the paper.

2. OVERVIEW OF CLOUD HPC ISSUES

In assessing the suitability of cloud platforms for parallel computing, one fact seldom brought to attention in past cloud evaluation is that HPC users do not have a single mode in using parallel computing resources. The same users typically go back and forth among several stages of the scientific computing cycle, including development/testing, production executions, and pre- or post-job scientific data analysis. In addition, users have different objectives and requirement when they use parallel platforms to conduct computing research. For each of these usage modes, people have different computing behavior, priorities, and constraints. Therefore, in this work we need to consider the unique challenges and opportunities offered by public clouds for common HPC activities on small- or medium-scale clusters. In the following, we give brief overview discussion on related issues, which is to be complemented by quantitative analysis in Section 3 and Section 4.

Performance: Performance is the ultimate motivation and goal driving parallel computing, and is a key parameter to consider in the cost comparison regarding clouds' competitiveness. Public clouds such as Amazon EC2 provide powerful computing hardware. It has been shown in multiple prior studies that single-node performance on the cloud in executing HPC workloads is on par with traditional clusters and virtualization brings negligible overhead [12, 26, 38, 40]. The interconnection network, on the other hand, is considered the major reason for the performance degradation and lack of scalability as observed in prior research [12, 14, 19, 38].

The new Amazon CCIs are intended for HPC and these nodes are interconnected with 10 Gigabit Ethernet, rather than Gigabit Ethernet for the traditional classes of instances. However, it is hard for EC2 to match the communication performance of clusters using InfiniBand, the overwhelmingly popular selection for HPC today. Section 3 reports our measurement results from EC2 CCIs and compares them with those from local InfiniBand connected clusters.

Performance stability is a closely related issue. While it is less important for code development and correctness testing, performance stability or predictability is crucial for long production runs, where the performance directly translates into costs to the users. High performance variability also increases the difficulty in performance benchmarking, tuning, and optimization.

Another related metric is the end-to-end turn-around time of parallel jobs. Clouds offer a unique advantage of near-interactive execution, where it generally takes no more than tens of seconds to acquire the requested instances and minutes to finish the environment configuration. Once setup, jobs can be run in an interactive manner. The elimination of queue waiting time can greatly improve user experience and in particular, the productivity of program development, testing, and debugging.

Cost and pricing: The cost of public clouds is straightforward and quite similar to how supercomputers charge users: the money (or service units) charged for a job is proportional to the product of the total run time and the

number of instances (nodes) allocated. The cost of owning and managing a local cluster, on the other hand, is quite complicated and concerns expenses for management, maintenance, space rental and energy. In Section 4, we carry out a rather simplified quantitative cost analysis using hardware cost, which under-estimates the local cluster's overall cost of ownership.

The relationship between cost and performance is more sophisticated than indicated by the pricing policy. When resource is limited, a physical cluster tends to be shared by more people within the same organization. Increased cluster utilization lowers actual cost, but produces longer queue waiting time as well as higher performance variability. Especially, bursty workloads often occur when members sharing the cluster have similar work patterns, caused by their common affiliation and deadlines (such as for software release or research paper submission). The elasticity of on-demand public clouds will likely mitigate this problem, where a much larger amount of computing resources is aggregated and the workload will be amortized over a larger and more diverse user community.

Configurability: Clouds provide individual users with the capability to customize their cluster instances, a possibility not afforded by traditional clusters. This is particularly meaningful for settings such as parallel file system configuration, where users can choose different parallel file systems, deploy a different numbers I/O servers, and use different disk configurations according to the needs of each individual application they run. Our results in Section 3 demonstrate the advantage of this capability. Also, the configurabity offered by clouds brings unique opportunities to parallel computing researchers, who would not have the access privilege to deploy their new or modified system software or tools.

3. PERFORMANCE EVALUATION AND EXPERIENCE

We conducted extensive performance benchmarking and comparison using the Amazon EC2 CCIs and local InfiniBand-connected cluster. The programs used in our evaluation include the NAS NPB benchmarks [27] (including the BTIO parallel I/O benchmark) and three real-world applications with different computation, communication, and I/O patterns. In addition, we ran micro-benchmarks such as the Intel MPI Benchmark [17] and IOR [33] to help us understand the performance behaviors observed in the macro-benchmark and applications. Finally, we report the performance variance collected in our experiments.

3.1 Platform Setup

3.1.1 Amazon EC2 CCI

In our cloud experiments, we used the HPC-oriented Amazon EC2 Cluster Compute Instances (CCIs), which became available in July 2010. Each CCI has 2 quad-core Intel Xeon X5570 "Nehalem" processors, with 23GB of memory. Each instance will be allocated to users in a dedicated manner, unlike those in most other EC2 instance classes (where two users may be allocated virtual machines sharing the same underlying physical node). The instances are interconnected with 10 Gigabit Ethernet.

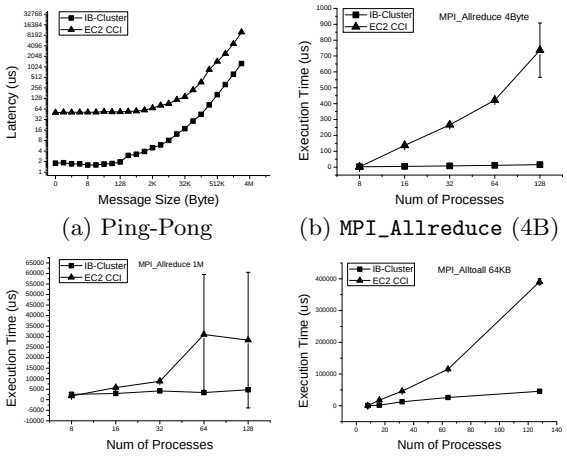
Each instance can access three forms of storage: (1) the local block storage ("ephemeral") of 1690GB capacity, where user data are not saved once the instances are released, (2) off-instance Elastic Block Store (EBS), where volumes can be attached to an EC2 instance as block storage devices, whose content persist across computation sessions, and (3) Simple Storage Service (S3), Amazon's key-value based object storage, accessed through a web services interface, which is designed more for Internet or database applications and lacks general file system interfaces needed in HPC programs. As HPC users is accustomed to viewing the shared or parallel file system available on clusters and supercomputers as a "scratch file system" not intended for long term data storage, in our application and I/O experiments we evaluate only the ephemeral disks, which provide higher performance than EBS devices. This is also due to that we have experienced several service interrupts when using EBS.

In our tests, we use the cluster instances Amazon Linux AMI 2011.02.1, the Intel compiler 11.1.072, and Intel MPI 4.0.1. The default compiler optimization level is *-O3*.

3.1.2 Local Clusters with IB Interconnection

For comparison, we also ran experiments on a newly purchased local Linux cluster. This cluster (also referred to as IB-Cluster interchangeably later) has compute hardware comparable to the EC2 CCIs, with 2 Intel Xeon X5670 6-core processors on each compute node. The nodes have 32GB memory each and are interconnected via QDR InfiniBand network. NFS is used as the shared file system. The Operating system installed is Red Hat Enterprise Linux 5.4 and the same compiler and MPI library as on the cloud platform are used. Note that although the cluster has different interconnection from the EC2, our comparison between the two is meaningful as it helps to gauge the performance/cost tradeoff for HPC users contemplating switching to using clouds - if they acquire an in-house cluster today, it will very likely have IB connection.

3.2 The Intel MPI Benchmark Results



(a) Ping-Pong (b) `MPI_Allreduce` (4B)

(c) `MPI_Allreduce` (1 MB) (d) `MPI_Alltoall` (64 KB)

Figure 1: The performance for `MPI_Allreduce` and `MPI_Alltoall` on EC2 platform.

For tightly coupled parallel applications, communication performance is crucial to their overall performance. We use The Intel MPI Benchmark(IMB) to evaluate the communication performance of Amazon EC2 CCIs and local

InfiniBand-connected cluster.

In Figure 1(a), the Ping-Pong results show significant performance gap on the message latency between Amazon EC2 CCIs and IB-Cluster,especially when the message size is small. For example, when the message size is 1 byte, the message latency is 50us in Amazon EC2 CCIs while only 1.42us in IB-Cluster. When the message size is larger, the performance gap becomes smaller, but still significant.

We show results of `MPI_Allreduce` for both small messages (4 Byte) and large messages (1 MB) in Figure 1(b) and (c) respectively. For both sizes, the IB-Cluster outperforms Amazon EC2 CCIs significantly and the gap widens rapidly when the number of processes increases. In Figure 1(d), we also show `MPI_Alltoall` results for a moderate message size (64KB),[1] which demonstrate an up to 10× performance difference. Other collective communications show similar behaviors and we omit related results due to the space limit.

From the IMB testing, we have the following observations:

1. Compared with previous EC2 instances which are connected with GbE, the 10GbE improves the bandwidth between cloud instances significantly. But for the performance of Amazon EC2 CCIs with 10GbE interconnects still fall 10 times slower than the state of art InfiniBand network.
2. The latency of Amazon EC2 CCIs for small messages are far worse, about 30 times more than InfiniBand, which indicates that applications with small messages may not be expected to have good performance in the current Amazon EC2 CCIs yet.

3.3 NPB Results

The NASA Parallel Benchmark(NPB) includes 8 benchmark programs derived from computational fluid dynamics (CFD) applications.We run both class C and class D for each of the NPB programs, both up to 128 processes. Due to the space limit, here we only include the class D results, which are more representative of application runs enabled by the larger capacity of today's parallel machines (class C results and analysis can be found in our technique report [36]).

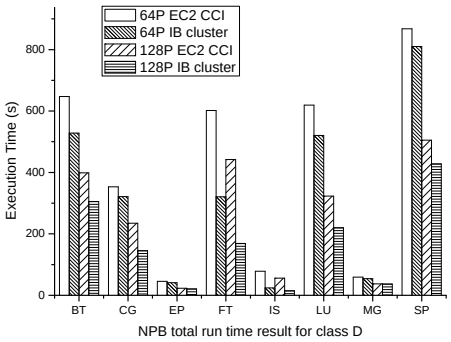


Figure 2: NPB performance (class D)

The class C results show that the cloud significantly lags behind: all benchmarks but EP run at least twice as slow on EC2 as on the local cluster. FT and LU behave the worst, with a 4-5 times slowdown, caused by intensive `MPI_Alltoall` uses in FT and a large amount of small messages in LU. This result, combined with the fact that the embarrassingly parallel EP program obtains almost equal performance on EC2 CCIs and the local cluster, suggests that the overall inferior performance on EC2 is mainly caused by its slow communication.

As expected, higher computation to communication ratios in NPB class D runs result in better relative performance for these benchmarks on EC2. Figure 2 portraits the overall execution times of the NPB class D benchmarks, comparing the performance on EC2 and the local cluster at two execution scales: 64 and 128 processes. On average, the local cluster's performance leads that of EC2 by 23.1% at 64 processes and by 32.0% at 128 processes. For several programs, such as EP and MG, SP, and CG at 64 processes, EC2's performance matches or closely follows the local cluster performance. Even LU, which showed severe problems on EC2 in class C runs, has no more than 50% slowdown compared with the local cluster. On the other hand, FT and IS fall far behind, where again slow `MPI_Alltoall` and `MPI_Alltoallv` operations take most of the time.

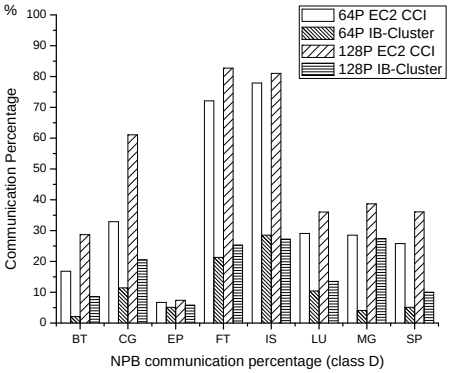


Figure 3: The percentage of communication time for NPB (class D)

To further confirm that communication is the main bottleneck on EC2, we measured the percentage of execution time spent on communication calls for each benchmark (Figure 3). We see clearly that the applications' communication intensity is correlated with the performance gap between the EC2 and local cluster executions. This percentage is taken by instrumenting the MPI library, so it does not account for the communication time hidden by computation.

In summary, CPU- and memory-intensive applications run well on cloud, while those with heavily message passing might fail to achieve good performance on the current CCI platform though it provides better interconnection than non-HPC-oriented instance classes. Our application results also support this observation.

Also, while running NPB, we find an interesting issue of load imbalance with benchmarks that heavily use non-blocking communication operations, such as BT and SP. From figure 4, we can see the curve EC2-CCI expresses regular fluctuation in computation time, which does not happen on the local IB cluster (the tail of IB cluster is due to process distribution, where the last four processes occupy a 12 core node and have less cache conflict).

We investigated the cause for the imbalance and found it was caused by TCP/IP protocol processing. Because the

[1] 64KB is also the message size used by NPB-FT in its `MPI_Alltoall` operations, for class C runs with 128 processess.

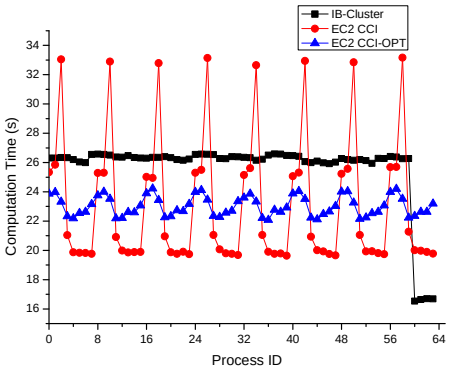


Figure 4: Load balance in SP between local cluster and EC2 platform (`OPT` means the computation time after optimizing the schedule strategy).

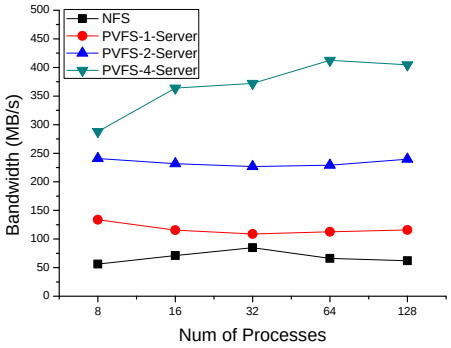
10GbE network interfaces used on EC2 CCIs do not support TCP/IP offloading, TCP/IP protocol processing has to be performed on CPU cores instead of network interfaces. The default configuration of Linux would always deliver the interrupts to the same pre-configured core, which will slow down the computation assigned to that core and result in load imbalance.

To verify this, we tried to modify the interrupt affinity file to deliver the interrupt to another core. However, the attempts always fail on EC2. This may be caused by certain protection mechanisms in either the virtual machine or the OS. We did not conduct further investigation as we found another way to do this: scheduling the processes round robin to a certain core. As seen in figure 4, the computation time is more balanced if we schedule different processes to the interrupt handling core every 5 seconds (the "EC2 CCI-OPT" curve). According to figure 4, the computation is now more balanced. Therefore, cloud developers should be more aware of the overlapping strategy, as long as Amazon does not upgrade the hardware to support communication offloading.

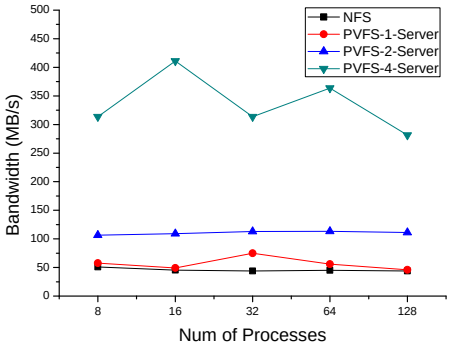
3.4 Parallel I/O Performance

HPC applications usually require a shared storage system to hold application executables, input data and output data. In practice, NFS [29] remains the dominantly popular choice for small and some medium-scale clusters, which was chosen in most previous cloud HPC evaluation work. On larger clusters and supercomputers, in contrast, there is typically a parallel file system, such as such as GPFS [32], PVFS [4], or Lustre [7], which are optimized for high parallel I/O bandwidth, high concurrency in metadata and file data accesses, and larger storage capacity. However, for clusters and supercomputers alike, users usually have to stay with the file system choice set at the equipment acquisition time and will not be able to configure the shared or parallel file system according to the needs of their specific applications.

To this end, clouds provide a unique opportunity for users to perform customized parallel I/O at an application-by-application, or even run-by-run basis. In our work, we build both the PVFS parallel file system and NFS on the EC2 CCI platform to compare their performance. For the PVFS file system, we use 1, 2, and 4 dedicated I/O servers, with one of them doubling as the metadata server. For NFS, we dedicate one instance as the file server and use the asynchronous mode. The setup and configuration turn out to be quite easy for both PVFS and NFS, each with a ~200-line script. The file system setting (such as the number and placement of PVFS servers) can be modified with a one-line script modification, and such changes can be made during a single execution session.



(a) Read Bandwidth

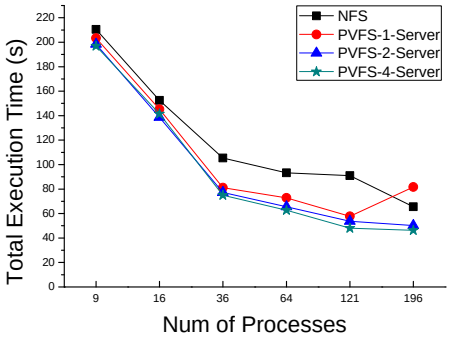


(b) Write Bandwidth

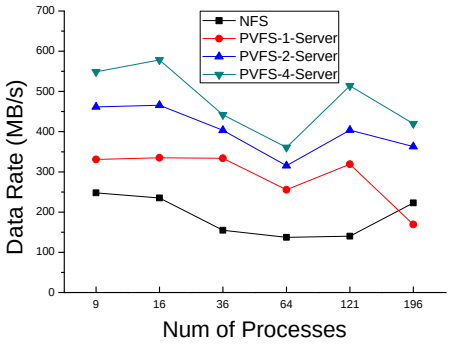
Figure 5: Read and write bandwidth for NFS server and PVFS server on EC2 CCIs platform.

We perform two classes of experiments. First, we use the IOR micro-benchmark [33] to test the aggregate read and write bandwidth of PVFS (using different number of I/O servers) and NFS. In our experiments, each of the instances run 8 parallel processes, reading and writing a 16GB single file collectively with MPI-IO. The results are shown in Figure 5. In general, PVFS provides much higher read/write bandwidths than NFS, even with the case of using only one I/O server. As the number of I/O servers increases, the PVFS aggregate bandwidth grows as expected, but saturate with a rather small number of client (compute) processes, such as 8 or 32. The unproportionally high write bandwidth of PVFS with 4 servers is likely due to the increased combined buffer capacity when more servers are used. We are currently investigating the I/O performance to explain the performance anomalies, such as the oscillation in the PVFS 4 server write performance.

Second, we analyze the performance for an I/O-intensive parallel program, BT-IO, on both file systems. The problem size we use is CLASS C and the I/O size is full SUBTYPE. With this setting, all of processes append to a single file through MPI-IO in 40 collective write operations, generating a total output volume of about 6.4 GB. Figure 6(a) shows the total execution time of BT-IO for both file systems, which is



(a) Total Execution Time



(b) Data Rate

Figure 6: Execution time and Data rate for BT-IO with PVFS server and NFS server.

consistent with the IOR results. In most cases, PVFS outperforms the NFS server and generates a meaningful overall program performance improvement. For example, with 121 processes, the total execution time is improved by 37% from NFS by using PVFS with only one server. Note that this configuration change does not bring additional cost to users. Similarly, at 196 processes, using 2 PVFS servers will lower the total execution time far enough to offset the cost of one extra instance, which does not apply when we increase the number of PVFS servers to 4. This shows that parallel file system selection and configuration can generate significant impact in optimizing I/O-intensive parallel applications, for both performance and cost.

Figure 6(b) shows the data rate recorded inside the BT-IO. In the BT-IO, the computation and I/O operations are executed in the interleaved mode, so the I/O requests can be buffered at the I/O server side. As a result, the data rate recorded in BT-IO is larger than the one tested with IOR benchmark, which is continuous I/O operations.

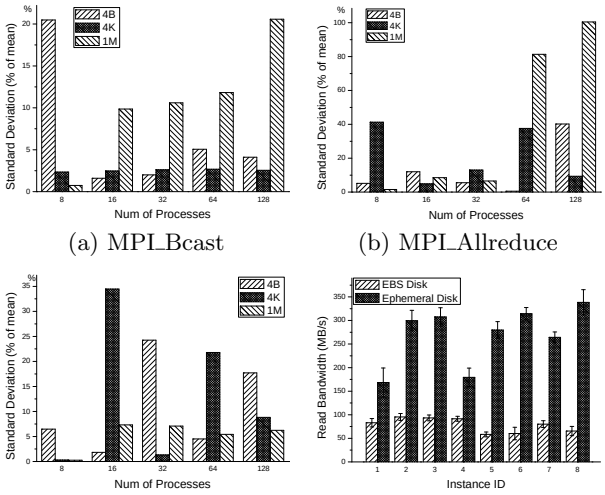
3.5 Application Results

We also performed testing on both cloud and local clusters on three real-world applications: (1) POP [22], an ocean circulation model simulation, which solves the three-dimension primitive equations for fluid motions on the sphere and forms the ocean component of the larger Community Climate System Model (CCSM), (2) Grapes (Global/Regional Assimilation and PrEdiction System) [6], a numerical weather prediction system developed by the Chinese Meteorological Administration, and (3) mpiBLAST [8, 24], a parallel implementation of the widely used NCBI BLAST (Basic Local Alignment Search Tool) [28], which searches sequence databases for similarities between a protein or DNA query and known sequences. These applications have been evaluated at a 10,000-core or larger scale on Tianhe-1A (Grapes), Jaguar (POP), and Intrepid (mpiBLAST), the No. 1, No. 2, and No. 13 supercomputers on the current Top500 list.

In our experiments, we have found that the I/O-intensive mpiBLAST scales well on EC2 CCIs, POP scales poorly (mainly because of the large amount of 1 element `MPI_Allreduce`), and Grapes, whose communication pattern is similar to POP but with fewer messages and larger message sizes, has scalability placed in between. The causes for their scalability (or lack thereof) are similar to those as discussed in the NPB result analysis. Again, due to space limitation, detailed application results and analysis are omitted here, but included in the full report [36].

3.6 Performance Variability

With Amazon EC2 CCIs, each instance is used by one user exclusively, which is different from the majority of other instance classes. For this new cloud platform, the main difference with traditional HPC clusters comes from the interconnection, shared off-instance storage, and the layer of virtualization. In this work, we monitor the computation, communication, and I/O performance for EC2 CCI platform on different times of the day, for several days. In this set of experiments, we first took measurements at fixed morning, afternoon, and evening points in two days, then randomly repeated the tests before carrying out the other experiments. The results are shown as the standard deviation over the mean values of different measurements to measure the performance variance. The larger this value is, the more unstable of the system looks.



(a) MPI_Bcast (b) MPI_Allreduce

(c) MPI_Alltoall (d) Disk Read Bandwidth

Figure 7: The performance variance on EC2 CCI platform.

Although the overall NPB performance results shown previously had quite tolerable variances (see error bars in the NPB figures), Figure 7 indicates that the communication performance still carries significant variability. For `MPI_Bcast`, the performance is not stable only for large messages. For `MPI_Allreduce`, when the number of processes is

greater than 32, the performance represents large variance for both small messages and large messages. Such large variance in communication may potentially bring higher synchronization overhead and makes operations such as parallel I/O harder to optimize.

We test the disk performance variance of read bandwidth using the `hdparm` tool under Linux. In this set of experiments, we care about variance in two dimensions: across virtual devices created and attached to different instances, and on the same device across multiple experiments (run at six time points, one hour apart from each other). The results are shown in Figure 7(d). There are 8 cluster instances, each of which mounted one EBS disk and one ephemeral disk for speed test once an hour. We found that each given instance performs rather stably across a series of tests. On the other hand, the virtual storage devices created and mounted simultaneously yield very different performance (across the x axis). In particular, the "local" ephemeral disks show surprisingly high variability. This indicates that it might be challenging to build efficient cloud parallel file systems, which typically stripe data across the virtual storage devices.

3.7 Reliability

During our intensive benchmarking on EC2 in a period of four weeks in March and April 2011, we encountered no availability issue with its instance computing resources or the network communication infrastructure. On the other hand, there was one time that the EBS volumes could not be mounted at all. In addition, for several times we encountered extremely poor EBS read/write bandwidths for hours. The ephemeral devices, in contrast, are found to be much more reliable.

4. COST ANALYSIS

Based on the performance results given in the previous section, we conduct a rough cost analysis to compare the cost of relying on cloud computing and that of owning an in-house cluster. As our IB-interconnected cluster has similar processor configuration and single-node performance with the CCIs (see EP performance in Section 3), we use its acquisition and operational expense in estimating the local clusters' cost. In our cost calculation we use the NAS NPB benchmark performance, which was presented and discussed in detail in the previous section.

As discussed in Section 2, the actual cost of local clusters is related to its utilization level. For an in-house cluster acquired as one unit and maintained for several years, the higher the actual utilization level, the lower the effective cost rate ($/hour). Therefore, we estimate *target utilization level*, the utilization level a local cluster needs to operate at to beat the cost of using EC2. More specifically, for each benchmark B_i, we compute the minimum local cluster utilization level U_{B_i} (assuming that B_i is the major workload type on the local cluster), which results in a lower cost compared to running B_i on the cloud.

Suppose the hourly rates of each instance/node on the cloud and the local cluster are R^{cloud} and R^{local}, respectively. Similarly the average execution times benchmarked for B_i in the two environments are $T_{B_i}^{cloud}$ and $T_{B_i}^{local}$. The CCIs are charged at $1.6 per hour and the ephemeral disks are free. Therefore $R^{cloud} = 1.6$. For the local cluster,

$$R_{B_i}^{local} = \frac{C^{local}}{365 \cdot 24 \cdot Y \cdot U_{B_i}}, \quad (1)$$

where C^{local} is the total cost of purchasing and managing the local cluster, assuming it is used for Y years.

Then to have

$$R^{cloud} \cdot T_{B_i}^{cloud} \geq R_{B_i}^{local} \cdot T_{B_i}^{local}, \quad (2)$$

We substitute $R_{B_i}^{local}$ with the right hand side in (1) and get

$$U_{B_i} \geq \frac{C^{local} \cdot T_{B_i}^{local}}{365 \cdot 24 \cdot Y \cdot 1.6 \cdot T_{B_i}^{cloud}} \quad (3)$$

Expense item	Amount
Dell 5670 Server (service included)	$6508/node
InfiniBand NIC	$612/node
InfiniBand switch	$6891
SAN with NFS server and RAID 5	$36753
Hosting (energy included)	$15251/rack/year

Table 1: Expense items used in calculating the cost of local cluster

To estimate C^{local}, we factor in the expense items based on the market price and hosting charges for our local IB cluster (acquired in March 2011), as listed in Table 1. Finally, we consider the case where the local cluster is in service for 4 years, so Y is set to 4.

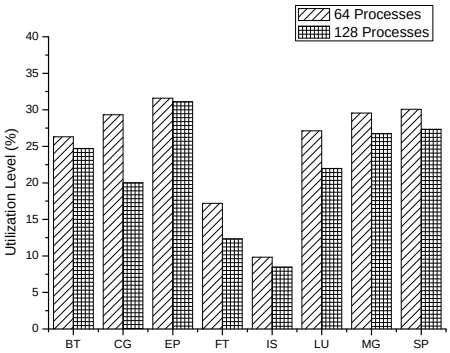


Figure 8: Target local cluster utilization levels based on NPB performance

Figure 8 gives the calculated target utilization level (U_{B_i}) for each NAS NPB class D benchmark, based on the performance results reported in Section 3.3. As the relative performance between the cloud and the local cluster is different at various job sizes, we calculate U_{B_i} using the performance numbers obtained at 64-process and 128-process runs respectively.

Due to the weaker scalability of EC2 runs, the target local cluster utilization levels derived from 128-process runs are lower, ranging from 8.5% (IS) to 31.1% (EP), with an average of 21.6%. 64-process runs generate a higher target utilization level, ranging from 9.8% (IS) and 31.6% (EP), with an average of 25.1%.

Based on our experience and observation, such utilization levels may fall between the typical actual utilization rate of research/development clusters and production- or semi-production clusters. In addition, the above calculation is biased against the cloud in several ways. First, in computing C^{cloud}, the local cluster cost, we did not include labor cost in management and maintenance, which could be significant. Second, our calculation assumes a fixed performance ratio between the cloud and local, while it is unlikely that the cloud instance hardware configuration remains fixed for 4 years. Cloud hardware upgrades are usually free to users. In contrast, a local cluster can only be upgraded with additional investment. Third, the utilization percentage we calculated is based on continuous usage (365 days, 24 hours per day), without removing weekends or holidays. Finally, a local cluster requires a substantial up-front investment for equipment acquisition in the first place, while cloud usage can be initiated and sustained with arbitrary amounts.

Considering the factors discussed above, we argue that public clouds with HPC-oriented hardware has become cost-effective to tightly coupled parallel program developers and users, compared to owning and sharing in-house clusters. At least, we see that even for the worst scaling applications (such as LU and FT), EC2 does not need an "order-of-magnitude performance improvement" to become a viable option for HPC users or organizations, as indicated by multiple previous studies. In addition, cloud-based clusters are particularly competitive cost wise, if users' major workloads consist of development, testing, or debugging with a small number of instances.

5. RELATED WORK

In the past three years there have been a wave of research efforts to examine the feasibility of using public clouds for high-performance or scientific computing. The great majority of these studies evaluated Amazon EC2, the leading public IaaS platform, but with different focuses. Quite several studies evaluated tightly coupled, MPI-style applications running on cloud platforms [10,12,14,19,26,30,34,38]. However, most of these efforts (especially earlier evaluations) were very limited in several ways. First, most existing studies only tested microbenchmarks such as NAS NPB [10, 14, 30, 38] and LINPACK [26]. Second, the experiments were of very small scale, typically testing with NPB problem size class A or B, and using fewer than 64 processes. Third, there lacks in-depth performance analysis to help locate the reasons of the unsatisfactory parallel program performance or scalability on clouds. Fourth, existing studies in this area seldom examined I/O performance. With the only exception [10], the authors reported IOR and BTIO results, but they seem to be obtained from sequential runs and used a basic NFS setting. Last but not least, previous evaluations were carried out before Amazon provided HPC-oriented cluster compute instances (CCIs).

Two more recent studies evaluated cloud performance with real-world applications. Jackson et al. measured the performance of seven applications from a parallel simulation workload composition, used in NERSC to benchmark production supercomputers, plus the HPCC benchmark suite. The authors also carried out profiling to measure the percentage of run time spent on communication. However, it is not clear how much of the cloud communication performance problem is caused by the use of EC2 *ml* instances, where physical nodes are not allocated in a dedicated manner and heterogeneity is introduced by node sharing and different node types (three different CPUs were detected by the authors). Also, there lacks scalability observation for the applications on either EC2 and the other local clusters/clouds examined. He et al. carried out a case study with a NASA climate prediction application [12], which confirms that interconnection is the chief factor contributing to unscalable cloud performance. The authors tested three public clouds but did not give detailed analysis on the difference of performance or scaling behavior on these platforms.

Up to now, most successful stories on using public clouds for high performance computing come from the execution of scientific workflows, or a large set of mutual-independent tasks [9, 15, 18, 20, 23, 37]. This is to be expected as such applications fall into the high-throughput computing category. They use the cloud more like a distributed computing platform, which is the primary use mode that public IaaS clouds such as EC2 have been designed for.

Some of the cloud HPC performance studies mentioned above conducted cost analysis (e.g., [20, 26, 30]), but were more focused on comparing the cost of different modes in setting up and running parallel applications or workflows. Deelman et al. performed detailed study on the end-to-end cost of executing the Montage astronomy workflow on EC2 [9] and again focused on evaluating workload composition and configuration. Wang et al. investigated cloud pricing schemes with traditional (commercial) cloud workloads [39], while in this work we discusses cloud pricing and cost issues unique to running tightly-coupled parallel applications. Finally, very recently Carlyle et al. reported results from an interesting comparison [3], which found that more Purdue HPC users will find using EC2 more costly than using the local "community clusters", where PIs pay lump sum for access contracts. However, such community facilities are subsidized by the university and are not available to the majority of users. As a side remark, this is currently the only publication known to us to report performance results using the Amazon EC2 CCIs since mid 2010 for HPC.

The concept of "virtual clusters" has been established prior to the existence of commercial, public clouds. For example, the use of virtual machines has been studied by researchers for high performance computing [5,11,16], which explored or designed concepts and mechanisms useful in current cloud HPC practice. These earlier virtual clusters were typically managed similarly as traditional batch systems. Recently, projects such as the Nimbus Cloud at the University of Chicago has examined the use of public cloud resources to extend the capacity of local clusters [25]. In this paper, we focus on the feasibility of HPC developers or users replacing their physical cluster (or even supercomputer) usage with pure cloud-based, pay-as-you-go computing.

Several prior efforts have investigated I/O and storage issues in virtualized or cloud environments. Some of them focused on long-term storage or inter-job or inter-task data flow. For example, Juve et al. studied the performance and cost of different storage options for scientific workflows running on Amazon EC2 [20]. Palankar et al. assessed the feasibility of using Amazon S3 for scientific grid computing [31]. Abe et al. constructed pWalrus, which provides S3-like storage access on top of the PVFS parallel file system on an open-source cloud [1]. Yu and Vetter studied parallel I/O in Xen-based HPC, but the environment used

only have virtualized compute nodes and the authors only tested at most 16 processes [41]. To our best knowledge, our work is the first to evaluate different parallel/shared file system settings for parallel I/O on public clouds.

6. CONCLUSION

In this work, we conducted a comprehensive evaluation of the recently released Amazon EC2 Cluster Compute Instances, which are intended for high-performance computing. We assessed the feasibility of replacing in-house cluster ownership with public cloud usage for running tightly coupled MPI programs, and report our evaluation results as well as experience regarding a variety of relevant issues, such as performance, scalability, performance variability, the impact of customized per-job file system configuration, system reliability, and cost.

Overall, our study reveals a picture considerably more positive for running MPI applications on public clouds, compared to previously published evaluation results. For many typical HPC workloads, the cloud performance relative to that using in-house InfiniBand-connected clusters justifies the option of going for clouds as cost-effective, unless the in-house cluster achieves a rather high utilization level, which is often not the case. This, in addition to the well known elasticity and flexibility in using cloud computing, can make "HPC in the cloud" a viable solution for many individual users and organizations.

Meanwhile, our investigation also confirms that the 10-Gigabit Ethernet interconnection (provided by the EC2 Cluster Compute Instances as improved communication infrastructure) remains the chief problem in scaling MPI programs. Particularly, the high latency poses a more severe obstacle and dramatically hurts applications come with a lot of small messages. 10-Gigabit Ethernet interrupt handling may also incur computation load imbalance when all the cores are occupied by MPI processes. In addition, although the overall performance variance observed in our benchmark and application executions appear tolerable, the communication and virtual storage devices can suffer from significant performance turbulence, which increases the challenges in performance tuning and designing efficient parallel I/O solutions.

7. ACKNOWLEDGEMENT

We want to express our thanks to our main supporter Intel inc., especially Scott McMillan, Nan Qiao, and Bob Kuhn. This work also gets support partly from Chinese "863" project 2010AA012403, 2006AA01A105, NSFC project 61073175, NSF grants 0546301 (CAREER), 0915861, 0937908, and 0958311, in addition to a joint faculty appointment between Oak Ridge National Laboratory and NC State University, as well as a senior visiting scholarship at Tsinghua University.

8. REFERENCES

[1] Y. Abe and G. Gibson. pWalrus: Towards Better Integration of Parallel File Systems into Cloud Storage. In *Workshop on Interfaces and Abstractions for Scientific Data Storage*, 2010.

[2] Amazon Inc. High Performance Computing (HPC). http://aws.amazon.com/ec2/hpc-applications/, 2011.

[3] A. G. Carlyle, S. L. Harrell, and P. M. Smith. Cost-effective hpc: The community or the cloud? In *IEEE International Conference on Cloud Computing Technology and Science*, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[4] P. Carns, W. Ligon III, R. Ross, and R. Thakur. PVFS: A parallel file system for Linux clusters. In *Proceedings of the 4th annual Linux Showcase & Conference-Volume 4*, pages 28–28. USENIX Association, 2000.

[5] J. S. Chase, D. E. Irwin, L. E. Grit, J. D. Moore, and S. E. Sprenkle. Dynamic Virtual Clusters in a Grid Site Manager. In *International Symposium on High-Performance Distributed Computing*. IEEE Computer Society, 2003.

[6] D. Chen, J. Xue, X. Yang, H. Zhang, X. Shen, J. Hu, Y. Wang, L. Ji, and J. Chen. New generation of multi-scale NWP system (GRAPES): general scientific design. *Chinese Science Bulletin*, 53(22):3433–3445, 2008.

[7] Cluster File Systems, Inc. Lustre: A scalable, high-performance file system. http://www.lustre.org/docs/whitepaper.pdf, 2002.

[8] A. Darling, L. Carey, and W. Feng. The design, implementation, and evaluation of mpiBLAST. In *Proceedings of the ClusterWorld Conference and Expo, in conjunction with the 4th International Conference on Linux Clusters: The HPC Revolution*, 2003.

[9] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good. The Cost of Doing Science on the Cloud: the Montage Example. In *Proceedings of the ACM/IEEE conference on Supercomputing*, 2008.

[10] C. Evangelinos and C. Hill. Cloud Computing for parallel Scientific HPC Applications: Feasibility of Running Coupled Atmosphere-Ocean Climate Models on Amazon's EC2. In *The 1st Workshop on Cloud Computing and its Applications (CCA)*, 2008.

[11] R. J. Figueiredo, P. A. Dinda, and J. A. B. Fortes. A Case For Grid Computing On Virtual Machines. In *International Conference on Distributed Computing Systems*, 2003.

[12] Q. He, S. Zhou, B. Kobler, D. Duffy, and T. McGlynn. Case Study for Running HPC Applications in Public Clouds. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, New York, NY, USA, 2010. ACM.

[13] N. Hemsoth. Amazon adds hpc capability to ec2. HPC in the Cloud, July 2010.

[14] Z. Hill and M. Humphrey. A Quantitative Analysis of High Performance Computing with Amazon's EC2 Infrastructure: The Death of the Local Cluster? In *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing*, 2009.

[15] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good. On the Use of Cloud Computing for Scientific Workflows. *IEEE International Conference on eScience*, pages 640–645, 2008.

[16] W. Huang, J. Liu, B. Abali, and D. K. Panda. A Case for High Performance Computing with Virtual Machines. In *Proceedings of the 20th International Conference on Supercomputing*, 2006.

[17] Intel Inc. Intel MPI Benchmarks. http://software.intel.com/en-us/articles/intel-mpi-benchmarks/.

[18] A. Iosup, S. Ostermann, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema. Performance analysis of cloud computing services for many-tasks scientific computing. *IEEE Transactions on Parallel and Distributed Systems*, 99, 2011.

[19] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. In *IEEE Second International Conference on Cloud Computing Technology and Science*, 2010.

[20] G. Juve, E. Deelman, K. Vahi, G. Mehta, B. Berriman, B. P. Berman, and P. Maechling. Data sharing options for scientific workflows on amazon ec2. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–9, 2010.

[21] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa. Science Clouds: Early Experiences in Cloud Computing for Scientific Applications. In *The 1st Workshop on Cloud Computing and its Applications (CCA)*, 2008.

[22] LANL. Parallel ocean program (pop). `http://climate.lanl.gov/Models/POP`, April 2011.

[23] J. Li, M. Humphrey, D. Agarwal, K. Jackson, C. van Ingen, and Y. Ryu. eScience in the Cloud: A MODIS Satellite Data Reprojection and Reduction Pipeline in the Windows Azure Platform. In *IEEE International Symposium on Parallel Distributed Processing*, 2010.

[24] H. Lin, P. Balaji, R. Poole, C. Sosa, X. Ma, and W. Feng. Massively parallel genomic sequence search on the Blue Gene/P architecture. Austin, TX, Nov. 2008.

[25] P. Marshall, K. Keahey, and T. Freeman. Elastic Site: Using Clouds to Elastically Extend Site Resources. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010.

[26] J. Napper and P. Bientinesi. Can Cloud Computing Reach the Top500? In *Proceedings of the combined workshops on UnConventional high performance computing workshop plus memory access workshop*, New York, NY, USA, 2009. ACM.

[27] The NAS Parallel Benchmarks. http://www.nas.nasa.gov/Resources/Software/npb.html.

[28] National Center for Biotechnology Information. NCBI BLAST. http://www.ncbi.nlm.nih.gov/BLAST/.

[29] B. Nowicki. *NFS: Network File System Protocol Specification*. Network Working Group RFC1094, 1989.

[30] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema. A performance analysis of ec2 cloud computing services for scientific computing. In *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 2010.

[31] M. R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel. Amazon S3 for Science Grids: A Viable Solution? In *Proceedings of the International Workshop on Data-Aware Distributed Computing*. ACM, 2008.

[32] F. Schmuck and R. Haskin. GPFS: a shared-disk file system for large computing clusters. In *Proceedings of the First Conference on File and Storage Technologies*, 2002.

[33] H. Shan, K. Antypas, and J. Shalf. Characterizing and predicting the I/O performance of HPC applications using a parameterized synthetic benchmark. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, page 42. IEEE Press, 2008.

[34] T. Sterling and D. Stark. A High-Performance Computing Forecast: Partly Cloudy. *Computing in Science and Engineering*, 11, 2009.

[35] Top500 supercomputer sites. http://www.top500.org/.

[36] T. University. Technique report r2011.4.10. http://www.hpctest.org.cn/resources/cloud.pdf.

[37] C. Vecchiola, S. Pandey, and R. Buyya. High-performance cloud computing: A view of scientific applications. In *International Symposium on Parallel Architectures, Algorithms, and Networks*. IEEE Computer Society, 2009.

[38] E. Walker. Benchmarking Amazon EC2 for High-Performance Scientific Computing. *Login*, 33(5), 2008.

[39] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou. Distributed Systems Meet Economics: Pricing in the Cloud. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, HotCloud'10. USENIX Association, 2010.

[40] L. Youseff, R. Wolski, B. Gorda, and C. Krintz. Evaluating the Performance Impact of Xen on MPI and Process Execution For HPC Systems. In *Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*, 2006.

[41] W. Yu and J. S. Vetter. Xen-Based HPC: A Parallel I/O Perspective. In *IEEE International Symposium on Cluster Computing and the Grid*. IEEE Computer Society, 2008.