The Application of Cloud Computing to Scientific Workflows: A Study of Cost and Performance

G. Bruce Berriman, Ewa Deelman, Gideon Juve, Mats Rynge and Jens-S. Vöckler

G. Bruce Berriman
Infrared Processing and Analysis Center
Caltech, Pasadena CA, USA
gbb@ipac.caltech.edu

Ewa Deelman
University of Southern California
Marina del Rey CA, USA
deelman@isi.edu

Gideon Juve
University of Southern California
Marina del Rey CA, USA
gideon@isi.edu

Mats Rynge
University of Southern California
Marina del Rey CA, USA
rynge@isi.edu

Jens-S. Vöckler
University of Southern California
Marina del Rey CA, USA
voeckler@isi.edu

The current model of transferring data from data centers to desktops for analysis will soon be rendered impractical by the accelerating growth in the volume of science data sets. Processing will instead often take place on high performance servers co-located with data. Evaluations of how new technologies such as cloud computing would support such a new distributed computing model are urgently needed. Cloud computing is a new way of purchasing computing and storage resources on-demand through virtualization technologies. We report here the results of investigations of the applicability of commercial cloud computing to scientific computing, with an emphasis on astronomy, including investigations of what types of applications can be run cheaply and efficiently on the cloud, and an example of an application well suited to the cloud: processing a large data set to create a new science product.

1. Introduction

By 2020, new astronomical observatories anticipate delivering combined data volumes of over 100 PB, a hundred-fold increase over currently available data volumes [1]. Such volumes mandate the development of a new computing model that will replace the current practice of mining data from electronic archives and data centers and transferring them to desktops for integration. Archives of the future must instead offer processing and analysis of massive volumes of data on distributed high-performance technologies and platforms, such as grids and the cloud. The astronomical community is collaborating with computer scientists in investigating how emerging technologies can support the

next generation of what has come to be called *data-driven astronomical computing* [2]. These technologies include processing technologies such as graphical processing units (GPU's), frameworks such as MapReduce and Hadoop, and platforms such as grids and clouds. Among the questions that require investigation are: what kinds of applications run efficiently and cheaply on what platforms? Are the technologies able to support 24/7 operational data centers? What are the overheads and hidden costs in using these technologies? Where are the trade-offs between efficiency and cost? What demands do they place on applications? Is special knowledge needed on the part of end users and systems engineer to exploit them to the fullest?

A number of groups are adopting rigorous approaches to studying how applications perform on these new technologies. One group ([3]) is investigating the applicability of GPU's in astronomy by studying performance improvements for many types of applications, including I/O and compute intensive applications. They are finding that what they call "arithmetically intensive" applications run most effectively on GPU's, and they cite examples such as radio-telescope signal correlation and machine learning that run 100 times faster than on CPU-based platforms. Another group ([4]) has shown how MapReduce and Hadoop ([5]) can support parallel processing the images released by the Sloan Digital Sky Survey (SDSS)[1].

This paper describes investigations of the applicability of cloud computing to scientific workflow applications, with emphasis on astronomy. Cloud computing in this context describes a new way of provisioning and purchasing computing and storage resources on-demand targeted primarily at business users. The Amazon Elastic Compute Cloud (EC2) (hereafter, AmEC2) is perhaps the best known commercial cloud provider, but academic clouds such as Magellan and FutureGrid are under development for use by the science community and will be free of charge to end users. Workflow applications are data-driven, often parallel, applications that use files to communicate data between tasks. They are already common in astronomy, and will assume greater importance as research in the field becomes yet more data driven. Pipelines used to create scientific data sets from raw and calibration data obtained from a satellite or ground-based sensors are the best known examples of workflow applications. The architecture of the cloud is well suited to this type of application, whereas tightly coupled applications, where tasks communicate directly via an internal high-performance network, are most likely better suited to processing on computational grids ([6]). The paper summarizes the findings of a series of investigations conducted by astronomers at the Infrared Processing and Analysis Center (IPAC) and computer scientists at the USC Information Sciences Institute (ISI) over the past five years.

The paper covers the following topics:

- Are commercial cloud platforms user-friendly? What kind of tools will allow users to provision resources and run their jobs?
- Does a commercial cloud offer performance advantages over a high-performance cluster in running workflow applications?
- What are the costs of running workflows on commercial clouds?
- Do academic cloud platforms offer any performance advantages over commercial clouds?

[1] `http://wise.sdss.org/`

2. Running Applications in the Cloud Environment

Astronomers generally take advantage of a cloud environment to provide the infrastructure to build and run parallel applications; that is, they use it as what has come to called "Infrastructure as a Service." As a rule, cloud providers make available to end-users root access to instances of virtual machines (VM) running the operating system of choice, but offer no system administration support beyond ensuring that the VM instances function. Configuration of these instances, installation and testing of applications, deployment of tools for managing and monitoring their performance, and general systems administration are the responsibility of the end user. Two publications ([7] and [8]) detail the impact of this business model on end-users of commercial and academic clouds. Astronomers generally lack the training to perform system administration and job management tasks themselves, so there is a clear need for tools that will simplify these processes on their behalf. A number of such tools are under development, and the investigations reported here used two of them, developed by the authors: Wrangler [9] and the Pegasus Workflow Management System ([10]).

Wrangler is a service that automates the deployment of complex, distributed applications on infrastructure clouds. Wrangler users describe their deployments using a simple XML format, which specifies the type and quantity of VMs to provision, the dependencies between the VMs, and the configuration settings to apply to each VM. Wrangler then provisions and configures the VMs according to their dependencies, and monitors them until they are no longer needed.

Pegasus has been developed over several years. From the outset, it was intended as a system for use by end-users who needed to run parallel applications on high-performance platforms but who did not have a working knowledge of the compute environment. Briefly, Pegasus requires only that the end-user supply an abstract description of the workflow, which consists simply of a Directed Acyclic Graph (DAG) that represents the processing flow and the dependencies between tasks, and then takes on the responsibility of managing and submitting jobs to the execution sites. The system consists of three components:

- Mapper (Pegasus Mapper): Generates an executable workflow based on an abstract workflow provided by the user or workflow composition system. It finds the appropriate software, data, and computational resources required for workflow execution. The Mapper can also restructure the workflow to optimize performance and adds transformations for data management and provenance information generation.
- Execution Engine (DAGMan): Executes the tasks defined by the workflow in order of their dependencies. DAGMan relies on the resources (compute, storage and network) defined in the executable workflow to perform the necessary actions.
- Task manager (Condor Schedd): manages individual workflow tasks, supervising their execution on local and remote resources.

Pegasus offers two major benefits in performing the studies itemized in the introduction. One is that it allows applications to run as is on multiple platforms, under the assumption that they are written for portability, with no special coding needed to support different compute platforms. The other is that Pegasus manages data on behalf of the user: infers the data transfers, registers data into catalogs, and captures

performance information while maintaining a common user interface for workflow submission. Porting applications to run on different environments, along with installation of dependent toolkits or libraries, is the end user's responsibility. Both [7] and [8] point out that this activity can incur considerable business costs and must be taken into account when deciding whether to use a cloud platform. Such costs are excluded from the results presented here, which took advantage of applications designed for portability across multiple platforms.

3. Applicability of a Commercial Cloud To Scientific Computing: Performance and Cost

Cloud platforms are built on the same types of off-the-shelf commodity hardware that is used in data centers. Providers generally charge for all operations, including processing, transfer of input data into the cloud and transfer of data out of the cloud, storage of data, disk operations and storage of VM images and applications. Consequently, the costs of running applications will vary widely according to how they use resources. Our goal was to understand which types of workflow applications run most efficiently and economically on a commercial cloud. In detail, the goals of the study were:

- Understand the performance of three workflow applications with different I/O, memory and CPU usage on a commercial cloud.
- Compare the performance of the cloud with that of a high performance cluster (HPC) equipped with high-performance networks and parallel file systems, and
- Analyze the costs associated with running workflows on a commercial cloud.

Full technical experimental details are given in [6] and [11]. Here, we summarize the important results and the experimental details needed to properly interpret them.

(*a*) *The Workflow Applications and Their Resource Usage*

We chose three workflow applications because their usage of computational resources is very different. Montage[2] aggregates into mosaics astronomical images in the Flexible Image Transport System (FITS) format, the international image format standards used in astronomy. Broadband[3] generates and compares seismograms for several sources (earthquake scenarios) and sites (geographic locations). Epigenome[4] maps short DNA segments collected using high-throughput gene sequencing machines to a previously constructed reference genome. We configured a single workflow for each application throughout the study. Table 1 summarizes the resource usage of each, graded as high, medium, or low and Table 5, discussed later, includes the input and output data sizes. Montage generated an 8-degree mosaic of the Galactic nebula M16 composed of images from the Two Micron All Sky Survey (2MASS)[5]; the workflow is considered I/O-bound because it spends more than 95% of its time waiting for I/O operations. Broadband

[2] `http://montage.ipac.caltech.edu`
[3] `http://scec.usc.edu/research/cme/`
[4] `http://epigenome.usc.edu/`
[5] `http://www.ipac.caltech.edu/2mass/`

Application	I/O	Memory	CPU
Montage	High	Low	Low
Broadband	Medium	High	Medium
Epigenome	Low	Medium	High

Table 1. Comparison of Workflow Resource Usage By Application.

used 4 sources of earthquakes measured at 5 sites to generate a workflow that is memory-limited because more than 75% of its runtime is consumed by tasks requiring more than 1 GB of physical memory. The Epigenome workflow is CPU-bound because it spends 99% of its runtime in the CPU and only 1% on I/O and other activities.

(*b*) *Experimental Set-Up and Execution Environment*

We ran experiments on AmEC2[6] and the National Center for Supercomputer Applications (NCSA) Abe High Performance Cluster (HPC)[7]. AmEC2 is the most popular, feature-rich, and stable commercial cloud, and Abe, decommissioned since these experiments, is typical of High Performance Computing (HPC) systems as it is equipped with high-speed networks and parallel file systems to provide high-performance I/O. To have an unbiased comparison of the performance of workflows on AmEC2 and Abe, all the experiments presented here were performed on single nodes, using the local disk on both EC2 and Abe, and the parallel file system on Abe.

A submit host operating outside the cloud, at ISI, was used to host the workflow-management system and to coordinate all workflow jobs, and on AmEC2 all software was installed on two VM-images, one for 32-bit instances and one for 64-bit instances. These images were all stored on AmEC2's object-based storage system, called S3. Column 1 of Table 2 lists five AmEC2 compute resources ("types") chosen to reflect the range of resources offered. We will refer to these instances by their AmEC2 name throughout the paper. Input data were stored on Elastic Block Store (EBS) volumes. EBS is a Storage Area Network-like, replicated, block-based storage service that supports volumes between1 GB and 1 TB.

The two Abe nodes, shown in Table 3, use the same resource type, a 64-bit Xeon machine, but differ only in their I/O devices: abe.local uses a local disk for I/O, while abe.lustre uses a Lustre parallel-file system. Both instances use a 10-Gbps InfiniBand network. The computational capacity of abe.lustre is roughly equivalent to that of c1.xlarge, and the comparative performance on these instances gives a rough estimate of the virtualization overhead on AmEC2. All application executables and input files were stored in the Lustre file system. For the abe.local experiments, the input data were copied to a local disk before running the workflow, and all intermediate and output data were written to the same local disk. For abe.lustre, all intermediate and output data were written to the Lustre file system. On Abe, Globus[8] and Corral [14] were used to deploy Condor glide-ins that started Condor daemons on the Abe worker nodes, which in turn contacted the submit host and were used to execute workflow tasks. Glide-ins are a scheduling technique where where Condor workers are submitted as user jobs via grid

[6] http://aws.amazon.com/ec2/

[7] http://www.ncsa.illinois.edu/UserInfo/Resources/Hardware/Intel64Cluster/

[8] http://www.globus.org/

Type	Arch.	CPU	Cores	Memory	Network	Storage	Price
m1.small	32-bit	2.0-2.6 GHz Opteron	1/2	1.7 GB	1-Gbps Eth.	Local	$0.10/h
m1.large	64-bit	2.0-2.6 GHz Opteron	2	7.5 GB	1-Gbps Eth.	Local	$0.40/h
m1.xlarge	64-bit	2.0-2.6 GHz Opteron	4	15.0 GB	1-Gbps Eth.	Local	$0.80/h
c1.medium	32-bit	2.33-2.66 GHz Xeon	2	1.7 GB	1-Gbps Eth.	Local	$0.20/h
c1.xlarge	64-bit	2.0-2.66 GHz Xeon	8	7.5 GB	1-Gbps Eth.	Local	$0.80/h

Table 2. Summary of Processing Resources on Amazon EC2.

Type	Arch.	CPU	Cores	Memory	Network	Storage
abe.local	64-bit	2.33 GHz Xeon	8	8 GB	10-Gbps InfiniBand	Local
abe.lustre	64-bit	2.33 GHz Xeon	8	8 GB	10-Gbps InfiniBand	Lustre

Table 3. Summary of Processing Resources on the Abe High Performance Cluster

protocols to a remote cluster. The glide-ins contact a Condor central manager controlled by the user where they can be used to execute the user's jobs on the remote resources. They improve the performance of workflow applications by reducing some of the wide-area system overheads.

(*c*) *Performance Comparison Between Amazon EC2 and Abe.*

Figure 1 compares the runtimes of the Montage, Broadband and Epigenome workflows on all the Amazon EC2 and Abe platforms listed in Table 2 and Table 3. Runtimes in this context refer to the total amount of wall clock time in seconds from the moment the first workflow task is submitted until the last task completes. They exclude the times for starting the VMs (typically, 70-90 s), data transfer, and latency in submitting jobs on Abe.

Montage (I/O bound). The best performance was achieved on the m1.xlarge resource. It has double the memory of the other machine types, and the extra memory is used by the Linux kernel for the file system buffer cache to reduce the amount of time the application spends waiting for I/O. Reasonably good performance was achieved on all instances except m1.small, which is much less powerful than the other AmEC2 resource

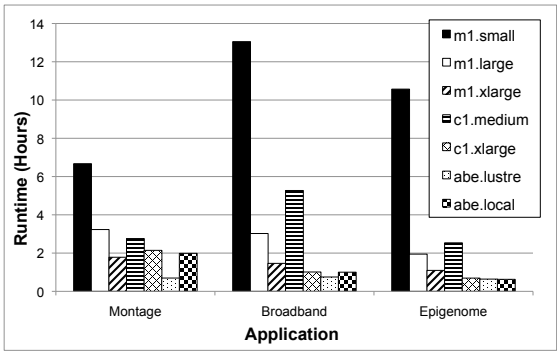

Figure 1. The runtimes in hours for the Montage, Broadband and Epigenome workflows on the Amazon EC2 cloud and on Abe. The legend identifies the processor instances listed in Table 2 and Table 3.

types. The c1.xlarge type is nearly equivalent to abe.local and delivered nearly equivalent performance (within 8%), which indicates that the virtualization overhead does not seriously degrade performance. The most important result of Figure 1 is a demonstration of the performance advantage of high-performance parallel file systems for an I/O-bound application. While the AmEC2 instances are not prohibitively slow, the processing times on abe.lustre are nevertheless nearly three times faster than the fastest AmEC2 machines. Since the completion of this study, AmEC2 has begun to offer high-performance options, and repeating this experiment with them would be valuable.

Broadband (Memory bound). For a memory bound application such as Broadband, the processing advantage of the parallel file system disappears: abe.lustre offers only slightly better performance than abe.local. Abe.local's performance is only 1% better than c1.xlarge, so virtualization overhead is essentially negligible. For a memory-intensive application like Broadband, AmEC2 can achieve nearly the same performance as Abe as long as there is more than 1 GB of memory per core. If there is less, some cores must sit idle to prevent the system from running out of memory or swapping. Broadband performs the worst on m1.small and c1.medium, the machines with the smallest memories (1.7 GB). This is because m1.small has only half a core, and only one of the cores can be used on c1.medium because of memory limitations.

Epigenome (CPU bound). As with Broadband, the parallel file system in Abe provides no processing advantage: processing times on abe.lustre were only 2% faster than on abe.local. Epigenome's performance suggests that virtualization overhead may be more significant for a CPU-bound application: the processing time for c1.xlarge was some 10% larger than for abe.local. As might be expected, the best performance for Epigenome was obtained with those machines having the most cores.

(*d*) *Cost-analysis of Running Workflow Applications on Amazon EC2*

AmEC2 itemizes charges for hourly use of all of its resources: compute resources (including running the VM), data storage (including the cost of VM images), and data transfer in and out of the cloud.

Resource Cost. AmEC2 generally charges higher rates as the processor speed, number of cores and size of memory increase, as shown by the last column in Table 2. Figure 2 shows the resource cost for the workflows whose performances were given in Figure 1. The Figure clearly shows the trade-off between performance and cost for Montage. The most powerful processor, c1.xlarge, offers a 3-threefold performance advantage over the least powerful, m1.small, but at 5 times the cost. The most cost-effective solution is c1.medium, which offers performance of only 20% less than m1.xlarge but at 5-times lower cost.

For Broadband, the picture is quite different. Processing costs do not vary widely with machine, so there is no reason to choose other than the most powerful machines. Similar results apply to Epigenome: the machine offering the best performance, c1.xlarge, is the second cheapest machine.

Storage Cost. Storage cost consists of the cost to store VM images in S3, and the cost of storing input data in EBS. Both S3 and EBS have fixed monthly charges for the storage of data, and charges for accessing the data; these vary according to the application. The rates for fixed charges are $0.15 per GB-month for S3, and $0.10 per GB-month for EBS. The variable charges are $0.01 per 1,000 PUT operations and $0.01 per 10,000 GET operations for S3, and $0.10 per million I/O operations for EBS. The 32-bit image used for the experiments in this paper was 773 MB, compressed, and the 64-bit image

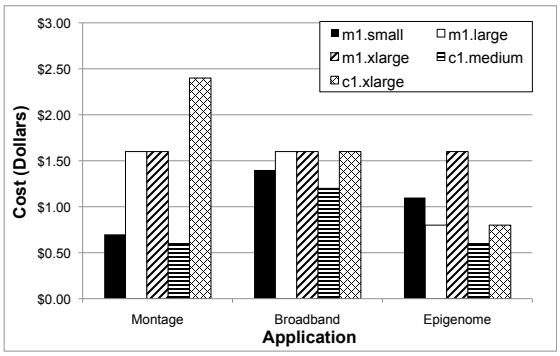

Figure 2. The processing costs for the Montage, Broadband and Epigenome workflows for the Amazon EC2 processors. The legend identifies the processor instances listed in Table 2 and Table 3.

Application	Input Volume	Monthly Cost
Montage	4.3 GB	$0.66
Broadband	4.1 GB	$0.66
Epigenome	1.8 GB	$0.26

Table 4. Monthly Storage Cost for Three Workflows

was 729 MB, compressed, for a total fixed cost of $0.22 per month. In addition, there was 4616 GET operations and 2560 PUT operations for a total variable cost of approximately $0.03. The fixed monthly cost of storing input data for the three applications is shown in Table 4. In addition, there were 3.18 million I/O operations for a total variable cost of $0.30.

Transfer Cost. In addition to resource and storage charges, AmEC2 charged $0.10 per GB for transfer into the cloud, and $0.17 per GB for transfer out of the cloud. Tables 5 and Table 6 show the transfer sizes and costs for the three workflows. In Table 5, input is the amount of input data to the workflow, output is the amount of output data, and logs refers to the amount of logging data that is recorded for workflow tasks and transferred back to the submit host. The cost of the protocol used by Condor to communicate between the submit host and the workers is not included, but it is estimated to be much less than $0.01 per workflow.

Table 6 summarizes the input and output sizes and costs. While data transfer costs for Epigenome and Broadband are small, for Montage they are larger than the processing and storage costs using the most cost-effective resource type. Given that scientists will almost certainly need to transfer products out of the cloud, transfer costs may prove prohibitively expensive for high-volume products. [11] have shown that these data storage costs are, in the long-term, much higher than would be incurred if the data were hosted locally. They cite the example of hosting the 12-TB volume of the 2MASS survey, which would cost $12,000 per year if stored on S3, the same cost as the outright purchase of a disk farm, inclusive of hardware purchase, support and facility and energy costs.

Application	Input (MB)	Output (MB)	Logs (MB)
Montage	4,291	7,970	40.0
Broadband	4,109	159	5.5
Epigenome	1,843	299	3.3

Table 5. Data transfer sizes per workflow on Amazon EC2

Application	Input	Output	Logs	Total Cost
Montage	$0.42	$1.32	$<0.01	$1.75
Broadband	$0.40	$0.03	$<0.01	$0.43
Epigenome	$0.18	$0.05	$<0.01	$0.23

Table 6. The costs of transferring data into and out of the Amazon EC2 cloud

(*e*) *Cost and Performance of Data Sharing*

The investigations described above used the Amazon EBS storage system. The performance of the different workflows do, however, depend on the architectures of the storage system used, and on the way in which the workflow application itself uses and stores files, both of which of course govern how efficiently data are communicated between workflow tasks. Traditional grids and clusters use network or parallel file system, and the challenge in the cloud is how to reproduce the performance of these systems or replace them with storage and network systems with equivalent performance. In addition to Amazon S3, which the vendor maintains, common file systems such as Network File System (NFS), GlusterFS, and the Parallel Virtual File System (PVFS), can be deployed on AmEC2 as part of a virtual cluster, with configuration tools such as Wrangler, which allows clients to coordinate launches of large virtual clusters.

We have investigated the cost and performance of the three workflows running with the storage systems listed in Table 7. The left hand panels in Figure 3 through Figure 5 show the three workflows performed with these file systems as the number of worker nodes increased from 1 to 8. The choice of storage system has a significant impact on workflow runtime. Figure 3 shows that for Montage, the variation in performance can be more than a factor of three for a given number of nodes. Amazon S3 performs poorly because of the relatively large overhead of fetching the many small files that make up its workflow. PVFS likely performs poorly because the small file optimization that is

File System	Brief Description
Amazon S3	Distributed, Object Based Storage System
Network File System (NFS)	Centralized node acts as file server for a group of servers
Gluster FS	Non-uniform file access (NUFA): write to new files always on local disk
Gluster FS	Distribute: Files distributed among nodes
Parallel Virtual File System (PVFS)	Intended for Linux Clusters

Table 7. File systems investigated on Amazon EC2. See [10] for descriptions and references

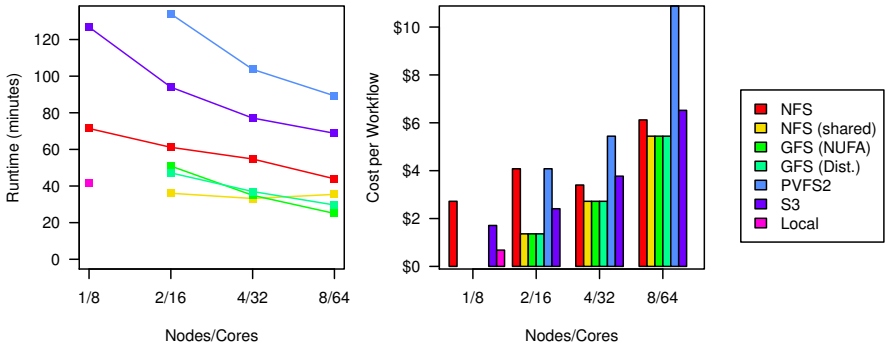

Figure 3. Variation with the number of cores of the runtime and data sharing costs for the Montage workflow for the data storage options identified in Table 7

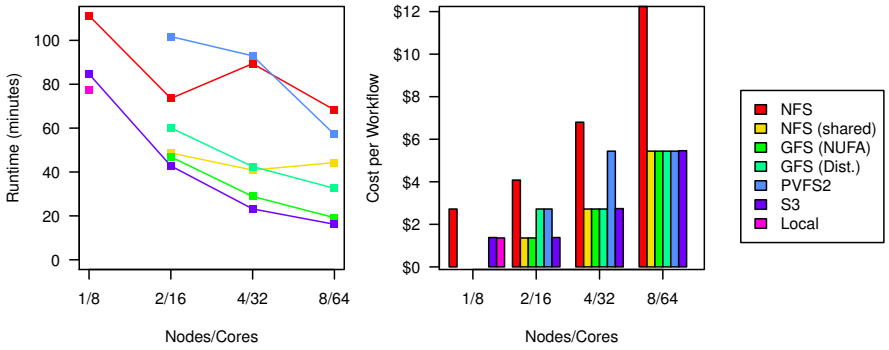

Figure 4. Variation with the number of cores of the runtime and data sharing costs for the Broadband workflow for the data storage options identified in Table 7

part of the current release had not been incorporated at the time of the experiment. The GlusterFS deployments handle this type of workflow efficiently.

By contrast, Epigenome shows much less variation than Montage because it is strongly CPU-bound. Broadband generates a large number of small files, and this is why PVFS most likely performs poorly. S3 performs relatively well because the workflow reuses many files, and this improves the effectiveness of the S3 client cache. In general, GlusterFS delivered good performance for all the applications tested and seemed to perform well with both a large number of small files, and a large number of clients. S3 produced good performance for one application, possibly due to the use of caching in our implementation of the S3 client. NFS performed surprisingly well in cases where there were either few clients, or when the I/O requirements of the application were low. Both PVFS and S3 performed poorly on workflows with a large number of small files, although the version of PVFS we used did not contain optimizations for small files that were included in subsequent releases.

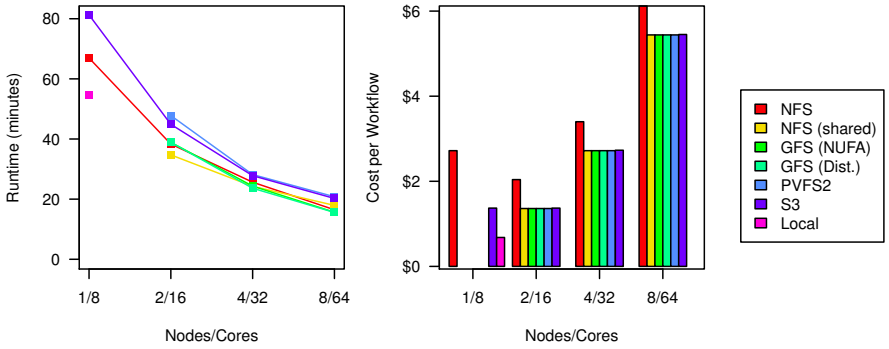

Figure 5. Variation with the number of cores of the runtime and data sharing costs for the Epiegenome workflow for the data storage options identified in Table 7

The differences in performance are reflected in the costs of running the workflows, shown in the right hand panels of Figure 3 through Figure 5. In general the storage systems that produced the best workflow runtimes resulted in the lowest cost. NFS was at a disadvantage compared to the other systems when it used an extra, dedicated node to host the file system; overloading a compute node to run the NFS server did not significantly reduce the cost. Similarly, S3 is at a disadvantage, especially for workflows with many files, because Amazon charges a fee per S3 transaction. For two of the applications (Montage, I/O-intensive; Epigenome, CPU-intensive) the lowest cost was achieved with GlusterFS, and for the other application, Broadband (Memory-intensive) the lowest cost was achieved with S3.

(*f*) *Summary of Investigations on Amazon EC2*

- Virtualization overhead on AmEC2 is generally small, but most evident for CPU-bound applications.
- The resources offered by AmEC2 are generally less powerful than those available in high-performance clusters and generally do not offer the same performance. This is particularly the case for I/O-bound applications, whose performance benefits greatly from the availability of parallel file systems. This advantage essentially disappears for CPU and memory bound applications.
- End-users should understand the resource usage of their applications and undertake a cost benefit study of cloud resources to establish a usage strategy. While the costs will change with time, this paper shows that the study must account for itemized charges for resource usage, data transfer and storage. The case of Montage, an I/O-bound application, shows why: the most expensive resources are not necessarily the most cost-effective, and data transfer costs can exceed the processing costs.
- AmEC2 offers no cost benefits over locally hosted storage, and is generally more expensive, but eliminates local maintenance and energy costs, and offers high-quality storage products.
- Performance and cost may depend strongly on the disk storage system used.

- A comparative study of the cost and performance of other commercial cloud providers will be valuable in selecting cloud providers for science applications. Such a study is, however, a major undertaking and outside the scope of this paper.

4. Running Scientific Applications on Academic Clouds

(*a*) *Development of Academic Clouds*

Clouds are under development in academia to evaluate technologies and support research in the area of on-demand computing. One example is Magellan, deployed at the U.S. Department of Energy's (DOE) National Energy Research Scientific Computing Center (NERSC) computing center with Eucalyptus technologies[9], which are aimed at creating private clouds. Another example of an academic cloud is the FutureGrid testbed[10], designed to investigate computer science challenges related to the cloud computing systems such as authentication and authorization, interface design, as well as the optimization of grid- and cloud-enabled scientific applications ([12]). Because AmEC2 can be prohibitively expensive for long-term processing and storage needs, we have made preliminary investigations of the applicability of academic clouds in astronomy, to determine in the first instance how their performance compares with those of commercial clouds.

(*b*) *Experiments on Academic Clouds*

The scientific goal for our experiments was to calculate an atlas of periodograms for the time-series data sets released by the Kepler mission[11], which uses high-precision photometry to search for exoplanets transiting stars in a 105 square degree area in Cygnus. The project has already released nearly 400,000 time-series data sets, and this number will grow considerably by the end of the mission in 2014. Periodograms identify the significance of periodic signals present in a time-series data set, such as arise from transiting planets and from stellar variability. They are, however, computationally expensive, but easy to parallelize because the processing of each frequency is performed independently of all other frequencies. Our investigations used the periodogram service at the NASA Exoplanet Archive ([12]). It is written in C for performance, and supports three algorithms that find periodicities according to their shape and according to their underlying data sampling rates. It is a strongly CPU-bound application, as it spends 90% of the runtime processing data, and the data sets are small, so the transfer and storage costs are not excessive ([12]).

Our initial experiments used subsets of the publicly released Kepler datasets. We executed two sets of relatively small processing runs on the Amazon cloud, and a larger run on the TeraGrid, a large-scale US Cyberinfrastructure. We measured and compared the total execution time of the workflows on these resources, their input/output needs and quantified the costs.

The cloud resources were configured as a Condor pool using the Wrangler provisioning and configuration tool [13]. Wrangler, as mentioned above, allows the user to specify the number and type of resources to provision from a cloud provider

[9] `http://open.eucalyptus.com/`

[10] `https://portal.futuregrid.org/about`

[11] `http://kepler.nasa.gov/`

Resources	Run 1 (AmEC2)	Run 2 (AmEC2)	Run 3 (TeraGrid)
Runtimes			
Tasks	631,992	631,992	631,992
Mean Task Runtime	7.44 sec	6.34 sec	285 sec
Jobs	25,401	25,401	25,401
Mean Job Runtime	3.08 min	2.62 min	118 min
Total CPU Time	1,304	1,113	50,019
Total Wall Time	16.5 hr	26.8 hr	448 hr
Inputs			
Input Files	210,664	210,664	210,664
Mean Input Size	0.084 MB	0.084 MB	0.084 MB
Total Input Size	17.3 GB	17.3 GB	17.3 GB
Outputs			
Output Files	1,263,984	1,263,984	1,263,984
Mean Output Size	0.171 MB	0.124 MB	5.019 MB
Total Output Size	105.3 GB	76.52 GB	3097.87 GB
Cost			
Compute Cost	$179.52	$291.58	$4,874.24 (estim.)
Output Cost	$15.80	$11.48	$464.68 (estim.)
Total Cost	$195.32	$303.06	$5,338.92 (estim.)

Table 8. Performance and Costs associated with the execution of periodograms of the Kepler data sets on Amazon and the NSF TeraGrid.

and to specify what services (file systems, job schedulers, etc) should be automatically deployed on these resources.

Table 8 shows the results of processing 210,000 Kepler time series data sets on Amazon using the 16 nodes of the c1.xlarge instance (Runs 1 and 2) and of processing the same data sets on the NSF TeraGrid using 128 cores (Run 3). Runs 1 and 2 used two computationally similar algorithms, while Run 3 used an algorithm that was considerably more computationally intensive than those used in Runs 1 and 2. The nodes on the TeraGrid and Amazon were comparable in terms of CPU type, speed, and memory. The result shows that for relatively small computations, commercial clouds provide good performance at a reasonable cost. However, when computations grow larger, the costs of computing become significant. We estimated that a 448hr run of the Kepler analysis application on AmEC2 would cost over $5,000.

We have also compared the performance of academic and commercial clouds when executing the Kepler workflow. In particular we used the FutureGrid and Magellan academic clouds.

The FutureGrid testbed includes a geographically distributed set of heterogeneous computing systems, a data management system, and a dedicated network. It supports virtual machine-based environments, as well as native operating systems for experiments aimed at minimizing overhead and maximizing performance. Project participants integrate existing open-source software packages to create an easy-to-use software environment that supports the instantiation, execution and recording of grid and cloud computing experiments.

Resource	CPUs	Eucalyptus	Nimbus
IU *india*	1,024 × 2.9GHz Xeon	400	-
UofC *hotel*	512 × 2.9GHz Xeon	-	336
UCSD *sierra*	672 × 2.5GHz Xeon	144	160
UFl *foxtrot*	256 × 2.3GHz Xeon	-	248
Total	3,136	544	744

Table 9. FutureGrid Available Nimbus- and Eucalyptus Cores in November 2010.

Site	CPU	RAM	Walltime	Cum. Dur.	Speed-Up
Magellan	8 x 2.6 GHz	19 GB	5.2 h	226.6 h	43.6
Amazon	8 x 2.3 GHz	7 GB	7.2 h	295.8 h	41.1
FutureGrid	8 x 2.5 GHz	29 GB	5.7 h	248.0 h	43.5

Table 10. Performance of Periodograms on Three Different Clouds

Table 9 shows the locations and available resources of five clusters at four FutureGrid sites across the US in November of 2010. We used the Eucalyptus and Nimbus technologies to manage and configure resources, and to constrain our resource usage to roughly a quarter of the available resources in order to leave resources available for other users.

As before, we used Pegasus to manage the workflow and Wrangler to manage the cloud resources. We provisioned 48 cores each on Amazon EC2, FutureGrid, and Magellan, and used the resources to compute periodograms for 33,000 Kepler data sets. These periodograms executed the Plavchan algorithm ([12]), the most computationally intensive algorithm implemented by the periodogram code. Table 10 shows the characteristics of the various cloud deployments and the results of the computations. The walltime measure as the end-to-end workflow execution, while the cumulative duration is the sum of the execution times of all the tasks in the workflow.

We can see that the performance on the three clouds is comparable, achieving a speedup of approximately 43 on 48 cores. The cost on running this workflow on Amazon is approximately $31, with $2 in data transfer costs.

The results of these early experiments are highly encouraging. In particular, academic clouds may provide an alternative to commercial clouds for large-scale processing.

5. Conclusions

The experiments summarized here indicate how cloud computing may play an important role in data-intensive astronomy, and presumably in other fields as well. Under AmEC2's current cost structure, long-term storage of data is prohibitively expensive. Nevertheless, the cloud is clearly a powerful and cost-effective tool for CPU and memory-bound applications especially if one-time, bulk processing is warranted and especially if data volumes involved are modest. The commodity AmEC2 hardware evaluated here cannot match the performance of a high-performance clusters for I/O-bound applications, but as AmEC2 offers more high-performance options, their cost and performance should be investigated. A thorough cost-benefit analysis, of the kind described here, should always be carried out in deciding whether to use a commercial cloud for running workflow

applications, and end users should perform this analysis every time price changes are announced. While academic clouds cannot yet offer the range of services offered by AmEC2, their performance on the one product generated so far is comparable to that of AmEC2, and when these clouds are fully developed, may offer an excellent alternative to commercial clouds.

Acknowledgment

G. B. Berriman is supported by the NASA Exoplanet Science Institute at the Infrared Processing and Analysis Center, operated by the California Institute of Technology in coordination with the Jet Propulsion Laboratory (JPL). Montage was funded by the National Aeronautics and Space Administration's Earth Science Technology Office, Computation Technologies Project, under Cooperative Agreement Number NCC5-626 between NASA and the California Institute of Technology. Montage is maintained by the NASA/IPAC Infrared Science Archive. This work was supported in part by the National Science Foundation under grants # 0910812 (FutureGrid) and # OCI-0943725 (CorralWMS)

References

[1] Hanisch, R. J. 2011. "Data Discovery and Access for the Next Decade." Keynote address at the "Building on New Worlds, New Horizons: New Science from Sub-millimeter to Meter Wavelengths" Conference, Santa Fe, New Mexico, March 7-10, 2011. (National Radio Astronomy Observatory). https://science.nrao.edu/newscience.

[2] Berriman, G. B., and Groom, S. L. 2011. "How Will Astronomy Archives Survive the Data Tsunami?" Association for Computing Machinery Queue, 9, 10. (http://queue.acm.org/issuedetail.cfm?issue=2039359). http://dx.doi.org/10.1145/2039359.2047483

[3] Barsdell, B. R., Barnes, D. G., and Fluke, C. J. 2010. "Analysing Astronomy Algorithms for GPUs and Beyond." MNRAS, 408, 1936. http://dx.doi.org10.1111/j.1365-2966.2010.17257.x.

[4] Wiley, K., Connolly, A., Krughoff, S., Gardner, J., Balazinska, M., Howe, B., Kwon, Y, and Bu, Yingyi. 201. "Astronomical Image Processing With Hadoop." Astronomical Data Analysis and Software Systems XX, ASP Conference Series (Evans, I. N., Accomazzi, A., Mink, D. J., and Rots, A. H., eds), Vol 442, 93

[5] White, T. 2009. "Hadoop: The Definitive Guide." (Sebabstopol, CA: O'Reilly), 1st ed.

[6] Juve, G., Deelman, E., Vahi, K., Mehta, G., Berriman, B., Berman, B. P. and Maechling, P. 2009. "Scientific Workflow Applications on Amazon EC2." Proceedings of the 5th International Conference on E-science Workshops (e-Science 09), 56 (IEEE). http://dx.doi.org/10.1109/ESCIW.2009.5408002.

[7] Canon, R. S., Ramakrishnan, L., Sakrejda, I., Declerck, T., Jackson, K., Wright, N., Shalf, J., and Muriki, K. 2011. "Debunking some Common Misconceptions of Science in the Cloud." Paper presented at "ScienceCloud2011: 2nd Workshop on Scientific Cloud Computing." San Jose, California, June 2011.

[8] United States Department of Energy Advanced Scientific Computing Research (ASCR) Program. 2011. "Magellan Final Report." `http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Magellan_Final_Report.pdf`

[9] Juve, G., Deelman, E. 2011. "Automating Application Deployment in Infrastructure Clouds." Paper presented at *3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2011)*.

[10] Deelman, E., Singh, G., Su, M.-H., Blythe, J., and Gil, Y. 2005 "Pegasus: A framework for mapping complex scientific workflows onto distributed systems." Sci. Program, 13, 219.

[11] Juve, G., Deelman, E., Vahi, K., Mehta, G., Berriman, B., Berman, B. P., and Maechling, P. 2010. "Data Sharing Options for Scientific Workflows on Amazon EC2." Proceedings of the 2010 ACM/IEEE

International Conference for High Performance Computing, Networking, Storage and Analysis (SC'10), 1. http://dx.doi.org/10.1109/ESCIW.2009.5408002.

[12] Berriman, G. B. and Juve, G. and Deelman, E. and Regelson, M. and Plavchan, P. 2010. "The Application of Cloud Computing to Astronomy: A Study of Cost and Performance." Proceedings of the 6th IEEE International Conference on e-Science Workshops, 1. http://dx.doi.org/10.1109/eScienceW.2010.10.

[13] Laszewski, G. von, Fox, G. C., Wang, F., Younge, A. et al. 2010. "Design of the futuregrid experiment management framework." The International Conference for High Performance Computing, Networking, Storage and Analysis (SC10), 1. (New Orleans, LA.) http://dx.doi.org/11/2010 2010. IEEE.

[14] Juve, G., Deelman, E., Vahi, K., Mehta, G. 2010. "Experiences with Resource Provisioning for Scientific Workflows Using Corral, " Scientific Programming, 18:2, pp. 77-92, April 2010.