Defining Future Platform Requirements for e-Science Clouds

Lavanya Ramakrishnan *
lramakrishnan@lbl.gov

Keith R. Jackson *
krjackson@lbl.gov

Shane Canon +
scanon@lbl.gov

Shreyas Cholia +
scholia@lbl.gov

John Shalf +
jshalf@lbl.gov

*Advanced Computing for Science
Lawrence Berkeley National Lab
Berkeley, CA

+ NERSC
Lawrence Berkeley National Lab
Berkeley, CA

ABSTRACT

Cloud computing has evolved in the commercial space to support highly asynchronous web 2.0 applications. Scientific computing has traditionally been supported by centralized federally funded supercomputing centers and grid resources with a focus on bulk-synchronous compute and data-intensive applications. The scientific computing community has shown increasing interest in exploring cloud computing to serve e-Science applications, with the idea of taking advantage of some of its features such as customizable environments and on-demand resources. Magellan, a recently funded cloud computing project is investigating how cloud computing can serve the needs of mid-range computing and future data-intensive scientific workloads. This paper summarizes the application requirements and business model needed to support the requirements of both existing and emerging science applications, as learned from the early experiences on Magellan and commercial cloud environments. We provide an overview of the capabilities of leading cloud offerings and identify the existent gaps and challenges. Finally, we discuss how the existing cloud software stack may be evolved to better meet e-Science needs, along with the implications for resource providers and middleware developers.

Categories and Subject Descriptors

C.2.4 [**Computer Systems Organization**]: Computer Communication Networks—*Distributed Systems*; J.2 [**Computer Applications**]: Physical Sciences and Engineering

General Terms

Design, Performance

Keywords

cloud computing, data parallel computing, mapreduce, scientific computing

SoCC'10, June 10–11, 2010, Indianapolis, Indiana, USA.

1. INTRODUCTION

Cloud computing provides a new resource model where multiple virtual servers hosted in data centers are used by individuals or groups, usually through a *pay-as-you-go* model. Cloud computing provides an illusion of infinite computing resources available on demand, i.e. in current cloud systems resources are accessible to the user almost instantly, with startup time of the instance imposing the only delays.

Cloud computing platforms are primarily used to serve the needs of web 2.0 applications, whereas their use in scientific communities is still being evaluated [9]. Different e-Science groups including those in bioinformatics [13], astronomy [5] and high energy physics have experimented with Amazon's infrastructure-as-a-service model and have also investigated the use of Hadoop for programming loosely coupled applications. Early results indicate some performance degradation in comparison to conventional batch-scheduled clusters. But this also promises to be an avenue to address new categories of scientific applications including data intensive science applications, on-demand/surge computing, and applications that require customized software environments. This new resource model will have a substantial impact on the business model for future High Performance Computing (HPC) centers in terms of how they provide services to the scientific community and the evolution of the software infrastructure necessary to manage those resources.

Magellan is a recently funded project, through DOE ASCR, to investigate how the cloud computing business model can be used to serve the needs of midrange computing and future data-intensive computing workloads for the Office of Science that are not served through DOE data center facilities today. The distributed testbed infrastructure has been deployed at Argonne Leadership Computing Facility (ALCF) and the National Energy Research Scientific Computing Facility (NERSC). At NERSC, the testbed will consist of 1,440 Intel Nehalem quad-core processors (5,760 cores total).

There has been intense discussion on the characteristics of clouds, along with their benefits and comparisons with grid environments [1, 7]. However, comparatively little attention has been devoted to defining the workload requirements, business model, user management, scheduling, application tools and security in the context of e-Science applications. The latter topics are central to defining the role of cloud computing in existing supercomputing centers and DOE's investment strategy for future computing infrastructure.

Cloud computing encompasses a wide scope of technologies and offerings. In this paper, we specifically address the solutions that are pertinent to e-Science applications - use of virtualized environments and software tools that are useful in these environments.

Section 2 defines a taxonomy of key scientific workloads that might be served well through cloud environments based on their application characteristics and associated business model. Section 3 describes examples of e-Science workloads that could benefit from existing cloud technologies. Section 4 identifies gaps in the current cloud computing offerings for target workloads. Section 5 revisits the cloud service model and outlines the opportunities of an improved software stack that better meets the needs of e-Science applications.

2. E-SCIENCE ENVIRONMENTS

Scientific explorations consist of a broad spectrum of application codes that run from user desktops to supercomputing centers, in areas such as nuclear physics, bioinformatics, environmental sciences, etc. These applications have a varied set of requirements and often have a need for unlimited compute cycles and data storage.

2.1 Application Classes

In this section, we provide a high-level classification of workloads in the scientific space based on their resource requirements and delve into the details of why cloud computing is attractive to these application spaces.

Bulk-Synchronous Large-Scale Computations. These are complex scientific codes generally running on large-scale supercomputing centers across the nation. These are MPI codes using a large number of processors (often in the order of thousands) and may have long running jobs. These jobs are serviced at supercomputing centers through batch queue systems. Users wait in a managed queue to access the resources requested, and their jobs are run when the required resources are available and no other jobs are ahead of them in the priority list. Most supercomputing centers provide archival storage and parallel file system access for the storage and I/O needs of these applications.

Bulk-Synchronous mid-range. These applications run at a smaller scale than the above jobs. There are a number of codes that need tens to hundreds of processors. Some of these applications run at supercomputing centers and backfill the queues. More commonly, users rely on small compute clusters that are managed by the scientific groups themselves to satisfy these needs.

Asynchronous massively independent. Some scientific explorations are performed on the desktop or local clusters and have asynchronous, massively independent computations. Even in the case of large-scale science problems, a number of the data pre- and post-processing steps, such as visualization, are often performed on the scientist's desktop. However the increased scale of digital data due to low cost sensors and other technology [10] has resulted in the need for these applications to scale to environments such as cloud environments. The requirements of such applications are similar to those of the internet applications that currently dominate the cloud computing space, but with far greater data storage and throughput requirements.

The Integrated Microbial Genomes (IMG) system hosted at the DOE Joint Genome Institute (JGI) fits this category. It supports analysis of microbial community metagenomes in the integrated context of all public reference isolate microbial genomes. The content maintenance cycle for data involves running BLAST for identifying pair-wise gene similarities between new metagenome and reference genomes where the reference genome baseline is updated with new (approximately 500) genomes every 4 months. This processing takes about 3 weeks on a Linux cluster with 256 cores. The size of the databases is growing, it is important that the processing can still be accomplished in a timely manner. The primary computation in the IMG pipeline is BLAST, a data parallel application that does not require communication between tasks and thus has similarities with traditional cloud applications.

2.2 Usage model

A central component of cloud computing is the underlying usage or business model. Currently, midrange computing infrastructure consists of a large number of departmental or PI (Principle Investigator)-owned clusters that are distributed across the DOE – some placed in machine rooms and many housed in closets. The virtualization technology that enables the cloud computing business model to succeed for Web 2.0 applications, could be used to create virtual "private clusters" within a shared resource that look, for all practical purposes, to be identical to privately managed PI-owned clusters. The premise is that carving up machines from large-scale data centers enables substantial cost-savings due to economies of scale and improved energy efficiency compared to running a number of smaller clusters, while retaining all of the benefits of exclusive control of the software configuration and availability that PIs desire.

The model thus facilitates the outsourcing of resource needs to external providers on a *pay-as-you-go model* instead of maintaining local infrastructure. A number of major vendors including IBM and HP, and more recently Amazon and Microsoft have embraced this service model of operating large clusters on behalf of external clients. In the context of the application classes discussed above we describe relevant usage models.

Private cluster. Some scientific users prefer to run their own private clusters for a number of reasons. They often don't need the concurrency levels achievable at supercomputing centers and need guaranteed access to resources for specific periods of time. For these needs to be satisfied in cloud environments, we need to be able to provide guaranteed access to the cluster when needed. This matches the level of service that motivates them to operate their own private cluster.

Personalized environment. A number of scientific applications have strong OS version dependencies and need environments that are consistent with local cluster or desktop environments. In these cases while users might not care where the resources are located, they desire the flexibility associated with custom software environments. The instant-availability of the resource is not as critical to this class of users as the strict control of the entire software environment (down to the sub-revisions of the OS kernel and libraries), and the throughput of the solution.

Science Gateways Users of well-defined computational workflows often prefer to have simple web-based interfaces to their application workflow and data archives. Web interfaces enable easier access to resources by non-experts, and enable wider availability of scientific data for communities of users in a common application area (e.g. Virtual Organizations). In this case the underlying infrastructure is decoupled from the user interface - scientists interface directly with their workflows through the web. Applications become available to users using the Software-as-a-Service model.

2.3 Application Requirements

Each of the current computational modes available to e-Science applications has certain disadvantages that cloud computing promises to address. We provide a brief overview here of the needs of the applications.

Scalable Computational Capacity The scientific exploration process often requires a large number of runs with different parameters and configurations. Supercomputing centers are shared resources that often have long queue wait times and policies on how many jobs a user might have in the queue. In addition, users often have bursts in their resource needs (e.g. close to deadlines), that might be unpredictable. Such special access needs often require out-of-band discussions with resource providers and a number of such requests are not accommodated due to over-subscription of the resources. Cloud computing promises the ability to get access to unlimited resources (albeit at a cost) which is very attractive to scientific users who have periodic surges in their resource needs that can't be satisfied through local or high performance computing centers.

Scalable Computational Performance. Many distributed memory scientific applications depend heavily on message passing libraries (MPI or GasNET) for inter-processor communication. In particular, bulk synchronous applications are very sensitive to messaging latency and bandwidth. In order to reduce latency and improve bandwidth, MPI-oriented HPC systems typically use high-performance fabrics like Infiniband and use messaging technology that bypasses the operating system to give direct access to the hardware. Collective constructs in MPI applications, such as barriers and reductions, are highly sensitive to subtle load-imbalances. Therefore reducing random sources of noise in the operating system (OS jitter) is critical for maintaining scalable performance.

Consistent Software Environments. HPC systems and, often, local cluster resources are shared environments where users are affected by any software upgrades that might occur at the sites. Compiler and library upgrades can cause many unproductive hours for the scientist. Currently, users either have to spend hours upgrading their software environments or must explicitly run only on resources where the environment is compatible with their needs. Thus user groups are often unable to use available idle resources during periods of surge due to software incompatibility.

Software distributions. A number of large-scale multi-institution collaborations have common shared software codes. The source code for these applications are distributed today and each group then individually installs them on local resources. Due to the variability in the software supported at each site, configuring and reproducing the exact execution state requires hours to days of work and coordination. Some groups are investigating the use of virtual machine images for distribution of all required software [3]. This would enable sites to boot the virtual machines at different sites with minimal or no work involved with software management.

Programming model. Scientific computing facilities mostly serve the needs of high-end compute intensive scientific codes. But as digital data becomes more readily available, there is an increasing need for scaling data intensive science that has traditionally run on desktop machines. Cloud computing works well with the MapReduce programming model that promises to be useful for some applications with a large amount of parallel data processing. For example, the JGI IMG pipeline mentioned earlier can benefit from a MapReduce framework where a number of sequence comparisons are processed in parallel.

Data Storage and Data Management HPC systems typically provide high-performance parallel file systems that enable parallel coordinated writes to a shared filesystem with high bandwidth and capacity. The file system also needs to allow for access patterns where multiple clients concurrently write to the same file, which is not supported by NFS based solutions. Parallel file systems like GPFS, Lustre and Panasas are usually employed to meet these needs and are often layered on top of high-end storage systems or use specially designed hardware. Even with these capability focused file systems and storage systems, data storage and movement remain one of the most challenging aspects of high-performance computing. In addition to the high-performance file systems, HPC centers typically provide an archival storage system to archive critical output and results. These systems are geared towards storing many petabytes of data in a reliable fashion. Many of these systems still use tape for storage. This is both for its cost effectiveness per byte and reliability. Cloud computing models will need similar data storage and management options for scientific applications to effectively use these environments.

3. CURRENT CLOUD SYSTEMS

We have worked with a number of applications and experimented with different cloud providers and technologies available. We have run a number of high performance benchmark [8] and application codes on Amazon web services to understand the performance implications of virtual machines. In addition, we have used Hadoop to manage the BLAST computations in the JGI IMG pipeline and compared its performance on different platforms including traditional HPC platforms, Amazon EC2 and the Yahoo M45 clusters [2]. In these sections we detail our experiences with using these technologies for these application studies.

3.1 Hadoop

The Apache Hadoop project is an open-source software that provides capabilities to harness commodity clusters for distributed processing of large data sets through the MapReduce [4] model.

The Hadoop streaming model allows one to create map and reduce jobs with any executable or script as the mapper and/or the reducer. This is the most suitable model for scientific applications that have years of code in place capturing complex scientific processes. The Hadoop framework does, however, make assumptions about the data model (e.g., single line inputs per process) that are not valid for scientific applications. This requires re-engineering of the application data used with Hadoop jobs.

The Hadoop File System (HDFS) is the primary storage model used in Hadoop. HDFS is modeled after the Google File system and has several features that are specifically suited to Hadoop / MapReduce. Those features include exposing data locality and data replication. Data locality is a key aspect of how Hadoop achieves good scaling and performance and Hadoop attempts to locate computation close to the data. This is especially true in the Map phase which is often the most I/O intensive phase. Data is replicated for fault tolerance and to provide more opportunities to execute computation near data. The data is transparently replicated by the file system.

3.2 Amazon Web Services

Amazon Web Services is a very popular cloud computing platform today. Amazon provides a number of different instance types in terms of its computational power for different pricing. We have run a number of benchmarks on the platform and the high performance MPI applications tend to experience a performance hit. Additionally, Amazon also provides higher-level services such as Elastic Map reduce. However our experience with elastic map reduce revealed that applications that didn't fit the traditional Hadoop data model could not use the existing API.

The primary methods for data storage in Amazon EC2 are S3 and Elastic Block Storage (EBS). S3 is a highly scalable key based storage system that transparently handles fault tolerance and data integrity. EBS provides a virtual storage device that can be associated with an Elastic Computing instance. S3 charges for space used per month, the volume of data transferred and the number of metadata operations (in 1000 allotments). EBS charges for data stored per month.

Scientific experimentation often results in changes to code and configuration, which may involve recreation of the virtual machine image. To avoid that problem and to additionally serve as the global file system needed for MPI jobs, we used EBS on a single node containing our binary and input data. This was then mounted through NFS on the rest of the nodes.

3.3 Yahoo M45

The Yahoo M45 cluster is a shared Hadoop platform-as-a-service cloud environment. The cluster resources are shared amongst users using a fair-share scheduler. The goal of this cluster is to provide a Hadoop MapReduce platform for data parallel applications in the scientific research space. At the time of writing, the cluster is comprised of 400 dual quad-core Intel Xeon E5320 1.86GHz nodes with 6GB of memory per node. Each node is configured to run 2 map tasks and 1 reduce task.

We did a performance analysis for the IMG metagenomics computations using the BLAST application in this framework and found that it was comparable to that of other traditional HPC platforms and Amazon EC2. Memory appears to be the bounding factor for BLAST, since the genome database must be loaded in memory. BLAST itself scales linearly with the size of the database but we hit a performance cliff when the database size exceeds the available memory size. Additionally, since our tests were run on a shared Hadoop cluster, it was difficult to get consistent overall performance numbers. The load on the system from other users impacted our total time-to-solution.

The Yahoo! M45 system allowed us to identify important bottlenecks and limitations in this problem space. In order to make the M45 environment more suited to a cloud-implementation of BLAST or similar memory intensive scientific applications we would benefit from a) higher memory limits in the software configuration, b) nodes with a large amount of available physical memory (similar to the extra-large instances on Amazon EC2 which have 15GB) c) ability to perform reservations for on-demand computing giving us more reliable throughput times for total time-to-solution.

4. GAP ANALYSIS

In the previous section, we detailed current cloud offerings. Here we examine some of the gaps and challenges in using existing cloud offerings directly for scientific computing.

4.1 Resource Provider Policies

Clouds promise an unlimited supply of resources on-demand. While this was true in early days of cloud computing where demand for resources was still ramping up, more recently users have noticed that their requests have not been satisfied on providers such as Amazon EC2 due to insufficient capacity. This situation is similar to current day supercomputing and grid resources that are often over-subscribed and have long wait queues. Thus for the end-scientist cloud computing as an unlimited supply of cycles tends to be less promising. There is a need for differentiated levels of service similar to Amazon's current offerings but with advanced resource request interfaces with specific QoS guarantees to avoid users needing to periodically query to see if resources have become available.

Portability. The vision of grid computing has been to enable users to use resources from a diverse set of sites by leveraging a common set of job submission and data management interfaces across all sites. However experiences revealed that there were challenges due to different software stacks and software compatibility issues. Virtualization facilitates software portability. Open source tools such as Eucalyptus [11] enable transition from local sites to Amazon EC2, but cloud interfaces in general are diverse and specific to each site making it hard to easily use multiple sites. In addition, cost of data movement to and especially from the cloud tends to be very expensive, discouraging portability. The data costs are also an issue for applications where scientists would like to perform post-processing on their end-desktop or local clusters, or would like to share their output data with other colleagues.

Cost. Today's cloud providers have a pay-as-you-go model for cloud resources. Cloud computing essentially enables anyone with a credit card to get access to resources. However infrastructure needs for scientific computing are either addressed through large upfront grants for equipment and/or through peer-reviewed allocations on supercomputing resources. Credit card transactions for resources don't fit into the current budget model at research institutions. In addition, PIs often have to distribute or carve out some percentage of their entire allocation to different users which cannot be accomplished in today's cloud computing scenarios. In addition, providers such as Amazon EC2 provide a plethora of options (e.g., spot instances, reserved instances, etc). under different pricing models. The diversity and unlimited scope of the scientific processes necessitates a runtime cost-benefit evaluation with respect to these offerings. Thus, we need to revisit institutional policies and software frameworks that can capture these policies to spearhead cloud computing adoption for e-Science.

4.2 Application Performance

Scientific applications have fairly large memory, compute, data and network needs. Our experiences with running Blast on the Yahoo! M45 cluster pushed us against some of these limits. For example, Blast performance scales linearly with the size of data, and is bound by the available physical memory - once the search database can no longer fit into physical memory we notice a sharp performance drop-off.

The traditional synchronous applications such as MPI perform poorly on virtual machines and have a huge performance overhead. Application codes with minimal or no synchronization, modest I/O requirements, with large messages or very little communication tend to perform well in cloud virtual machines. Traditional cloud computing platforms are designed for applications where there is little or no communication between individual nodes and where applications are less affected by failures of individual nodes. This assumption breaks down for large-scale synchronous applications.

In general cloud providers need to build systems that can meet the more intensive requirements of scientific computing. The standard commercial offerings might not be completely suitable for scientific needs out of the box, and at the very least, seem to require a fair amount of initial setup to meet the needs of science applications.

Cloud providers such as Amazon EC2 provide simple web service APIs for access and setup of resources. However running on Amazon EC2 often requires creating customized images, determining how resources are managed, implementing fault tolerance, etc. This requires a fair amount of system administrator experience. Similarly Hadoop applications require a fair amount of development experience. Thus there is a need for developing application tools that exist above these current offerings that account for the needs of the science and enable easier access to cloud resources.

Additionally in the cloud model, time-to-solution is a more critical metric than individual node performance. Scalability is achieved by throwing more nodes at the problem. However, in order for this approach to be successful for e-Science, it is important to understand the setup and configuration costs associated with porting a scientific application to the cloud - this must be factored into the overall time-to-solution metrics in resource selection and management decisions.

4.3 Data management

As discussed earlier, scientific applications have a number of data and storage needs. Synchronized applications need access to parallel file systems. There is also a need for long-term data storage. None of the current cloud storage methods resemble the high-performance parallel file systems that HPC applications typical need. Hadoop is optimized such that applications can benefit from data locality in the underlying HDFS system. This requires that Hadoop applications are written to be able to store and retrieve data from HDFS. Thus while applications can leverage the features of Hadoop for task farming and coordination of tasks, rewriting legacy scientific applications for use in cloud environments is often infeasible and impractical. File system modules have been written for Linux that allow access to HDFS through the standard VFS layer. The FUSE interface takes VFS request from the kernel and executes them in user space and this results in a performance overhead.

5. REVISITING THE CLOUD MODEL

The goal of the Magellan project is to evaluate possible solutions for cloud computing for science. In this section, we revisit the challenges and opportunities facing resource providers and middleware developers in providing next-generation services.

Resource Provider Model. Resource providers serving the e-Science community need to support a diverse set of models to accommodate different user needs. For example, synchronous applications often need access to non-virtualized resources for the best performance. Similarly as sites support more data intensive science, there is a need for frameworks such as Hadoop. Magellan will support provisioning through batch queue (MOAB) systems, cloud solutions such as Hadoop, and Eucalyptus, as well as access to Hadoop over batch queue systems using Hadoop On Demand(HOD).

Virtual machines do not have any persistent state or shared file systems, requiring sites to develop solutions that meet the storage needs of the end user. Resource provider sites need a) shared file systems such as GPFS or NFS accessible across all nodes allocated to the end user, that can be used to store persistent information across virtual machines and b) archival storage for long term storage of data.

Currently in supercomputing centers, the sites manage the operating system and middleware that is needed across multiple groups. Users compile and install their applications on specific systems (often with help from site personnel). As we move to the cloud computing model, sites must provide tooling and support for managing a diverse set of kernels and operating systems that required by specific groups. The clear separation of responsibilities for software upgrades and operating system patches no longer exists, and sites will need mechanisms to bridge the gap between supporting user-supported images and site security policies.

A cloud system where users have complete control of their images, has certain implications on site security policies. Additional checks and mechanisms are needed to protect critical infrastructure services (e.g. DNS, file servers) on the sites, ensure isolation between different virtual machines, and defend against malicious outgoing and incoming traffic.

We need monitoring and dynamic allocation policies that can load-balance between clusters of different types and provide guaranteed Quality of Service to users of all resource types. In addition to the performance needs of the application, there is a need for tools to manage fault tolerance and reliability of the virtual machines. Virtual machine migration [12] has been proposed as a way to provide higher quality of service especially during planned upgrades and maintenance cycles.

Software Stack. Software stacks supporting distributed scientific applications has largely evolved in the context of grid computing. The software stack can be categorized into these primary layers a) End-user portal interfaces or science gateways b) Coordination services layer that coordinates underlying resources for efficient and reliable execution c) Resource management services that interface directly with the underlying resources

Cloud computing addresses the portability of the software stack, a known issue with current day grid systems. Cloud computing for e-Science needs a similar set of software tools to harness and coordinate underlying resources. Cloud computing technologies provide additional features that greatly simplify some of the known challenges with existing software stacks, and also provides additional challenges of resource coordination. Here we present a vision for a cloud computing software stack for e-Science that mitigates some of the known problems and builds on existing commercial products.

Batch queue systems such as Moab, PBS, etc provide customizable policy points and algorithms to control scheduling of requests and resources. Similarly, there is a need for policy points in the cloud infrastructure and the ability to store and service requests when resources cannot be allocated immediately. On top of the basic resource management services, there are a number of coordination services available in cyber-infrastructure environments today. These include grid services for job submission and data transfer, application services that coordinate the application specific setup and execution, data replication services, meta-scheduling services etc. In cloud computing we need a number of similar tools and services. These tools must manage the underlying resource procurement, and apply any runtime customization needed in the virtual machine (e.g. bringing up specific services within the instance). We need monitoring services and the ability to dynamically grow and shrink resource holdings without impacting the application. Additionally, we need tools to manage job and data coordination, execution and monitoring.

Moving higher up in the software stack, a class of users will expect to interface with their underlying applications through web-based science gateways, so that they are removed from the infrastructure level details. Applications must then be accessed through a middleware layer that can expose traditional science applications as web services.

In order to make scientific computing more accessible, and to allow for a faster overall time-to-solution, it is important to have simple and robust interfaces that are seamlessly integrated with the applications and cloud infrastructure. Given the nature of the cloud, and its close coupling with web technologies, these interfaces must leverage existing http based protocols. REST [6] (Representational State Transfer) is gaining wide adoption as the underlying mechanism for building these interfaces. REST allows one to expose resources by using the existing http protocol layer. It creates a very simple and powerful means to access underlying resources using a combination of URIs and http verbs.

Most cloud providers including Amazon, Microsoft, Google, etc., are striving to make their interfaces RESTful. Grid technologies, which are similar in spirit to the cloud have been saddled by more heavyweight protocols and technologies like WSDL and SOAP, limiting overall usability. Using simple REST based APIs makes it very easy to add powerful web 2.0 functionality to science applications. Looking forward, this will allow users to *mash-up* disparate data sources across the cloud because everything is essentially just a URI.

6. CONCLUSIONS

Cloud computing promises to be an alternative approach for midrange and data intensive e-Science applications. Scalable parallel performance is achievable on these platforms for application codes with modest I/O requirements and minimal or no synchronization and communication. However, existing cloud offerings in the commercial space do not completely meet the needs of these applications. Currently, running a particular application on the cloud requires substantial understanding and assembly of the cloud computing technologies. This paper summarizes these gaps and revisits the service model both in the context of providers as well as the software stack. There is a need for differentiated service levels, hardware platforms tailored to scientific applications, and higher level software tools that can manage the complexities of the underlying technology fabric.

7. ACKNOWLEDGMENTS

This work was funded in part by the Advanced Scientific Computing Research (ASCR) in the DOE Office of Science under contract number DE-C02-05CH11231. The authors would like Yahoo! and CITRIS, UC Berkeley for access to Yahoo! M45 cluster and Greg Bell and CITRIS for Amazon EC2 access. The authors would also like to thank the Magellan team at NERSC for discussions on cloud computing, Victor Markowitz for detailed discussions on the IMG pipeline.

8. REFERENCES

[1] M. Armbrust and et al. Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

[2] S. Canon, S. Cholia, J. Shalf, K. Jackson, L. Ramakrishnan, and V. Markowitz. A performance comparison of massively parallel sequence matching computations on cloud computing platforms and hpc clusters using hadoop. 2009.

[3] Cern Virtual Machines. `http://rbuilder.cern.ch/project/cernvm/releases`.

[4] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. pages 137–150.

[5] 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 SC'08*, Austin, TX, 2008. IEEE.

[6] R. T. Fielding and R. N. Taylor. Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[7] I. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud Computing and Grid Computing 360-Degree Compared. *Grid Computing Environments Workshop, 2008. GCE '08*, pages 1–10, Nov. 2008.

[8] J. S. Katie Antypas and H. Wasserman. Nersc-6 workload analysis and benchmark selection process. Technical Report LBNL-1014, Berkeley, CA, 2008.

[9] K. Keahey and T. Freeman. Science Clouds: Early Experiences in Cloud Computing for Scientific Applications. In *Cloud Computing and its Applications (CCA)*, 2008.

[10] J. Li, D. Agarwal, M. Humphrey, C. van Ingen, K. Jackson, and Y. Ryu. eScience in the Cloud: A MODIS Satellite Data Reprojection and Reduction Pipeline in the Windows Azure Platform.

[11] D. Nurmi and et al. Eucalyptus:A Technical Report on an Elastic Utility Computing Archietcture Linking Your Programs to Useful Systems. Technical Report 2008-10, University of California, Santa Barbara, California, August 2008.

[12] K. K. Ramakrishnan, P. Shenoy, and J. Van der Merwe. Live data center migration across wans: a robust cooperative context aware approach. In *INM '07: Proceedings of the 2007 SIGCOMM workshop on Internet network management*, pages 262–267, New York, NY, USA, 2007. ACM.

[13] M. C. Schatz. CloudBurst: highly sensitive read mapping with MapReduce. *Bioinformatics*, pages 1363–1369, June 2009.