# On the Use of Cloud Computing for Scientific Workflows

Christina Hoffa[1], Gaurang Mehta[2], Timothy Freeman[3], Ewa Deelman[2], Kate Keahey[3], Bruce Berriman[4], John Good[4]

*[1]Indiana University, [2]University of Southern California, [3]Argonne National Laboratory, [4]Caltech*

## Abstract

*This paper explores the use of cloud computing for scientific workflows, focusing on a widely used astronomy application-Montage. The approach is to evaluate from the point of view of a scientific workflow the tradeoffs between running in a local environment, if such is available, and running in a virtual environment via remote, wide-area network resource access. Our results show that for Montage, a workflow with short job runtimes, the virtual environment can provide good compute time performance but it can suffer from resource scheduling delays and wide-area communications.*

## 1. Introduction

Recently, cloud computing [1, 2] has been under a growing spotlight as a possible solution for providing a flexible, on-demand computing infrastructure for a number of applications. Clouds are being explored as a solution to some of the problems with Grid computing, but the differences between cloud computing and the Grid are often so diminished and obscured that they become indistinguishable from one another. The term "Grid" computing was coined in the early 1990's to liken a distributed computing infrastructure to the electrical power grid [3]. Like the electrical power grid, a computational Grid uses resources that are potentially very geographically far apart. These resources can be allotted to combinations of one or more groups of users, with the owners of the resources deciding when and to whom they should be allotted. In this manner, collaborations can integrate pools of resources to give supercomputer-class capability to their users.

Just as resources can be spread out geographically, so too can the members of a virtual organization. A virtual organization is a group of individuals or institutions who share direct access to resources such as computers, data, and software according to a set of rules [4]. The problems with dynamic sharing of resources are that an available resource may not meet the needs of a particular virtual organization or a particular member of a virtual organization. As a result, resources may also become underutilized. Conversely, a specific resource may not be available at the time it is needed, leaving the user to wait or to seek an alternative. In addition, significant overhead is incurred to ensure software compliance on different clusters. Additionally, security is always an issue in any kind of networked environment.

In the past decade, many of these issues have begun to be addressed, but several of the problems still exist when working with a Grid. For example, security infrastructures have evolved and have been increasingly implemented as the use of Grid computing has grown, but issues with dynamic scheduling and planning still occur regularly. One solution that has come about in recent years is cloud computing [5, 6]. "The Cloud" refers to the typical depiction of the Internet as a mass of thin, branching lines spreading in all directions. Cloud computing is a set of virtual servers that work together through the Internet and can be dynamically managed, monitored, and maintained. Users are expected to develop their own virtual images or use existing ones as an executable environment on the cloud. Using virtual machines (VMs) that can be configured before deployment has the potential to reduce inefficient resource allocation and excess overhead. A VM can create an environment on a resource that is configured independently from that resource, allowing multiple such environments to be deployed on the same resource at the same time. In this manner of separation, each environment is kept secure from any others. Because sharing can be much more flexible, this also can also increase resource utilization [7].

While reducing or eliminating some sources of overhead, cloud computing introduces a few others. A network must be set up, and an image must be generated or discovered before any computations can take place. In the case of a virtual cluster, several virtual nodes need to be launched within the same time frame and made aware of each other. Once all these requirements are complete, execution overheads must still be taken into account. In addition, monetary costs must be considered. A group can purchase a Grid cluster for a relatively high cost, but it has complete control and sole access to it. The cluster will be sufficient for a certain amount of time until the cluster machines either break or become obsolete and a new cluster must be purchased. In contrast, cloud allotments may be purchased for a relatively small cost and for a relatively small amount of time, but the cost is incurred every time resources are used.

One of the primary obstacles Grid users face today is that while a Grid offers access to many heterogeneous resources, a user typically needs a very specific environment that is customized to support a legacy application, remains consistent across multiple executions, or supports the idiosyncrasies of many service configurations. Resource providers clearly cannot support the diversity of all the required environments while users are often unable to use what is available. Further, guaranteeing a desired resource allocation to a Grid user is difficult today as most of the available tools support only unstructured sharing which further limits the impact of Grid computing on science. Using virtualization in conjunction with Grid computing, known as "cloud computing" or Infrastructure-as-a-Service (IaaS), has been recognized as a potential solution to these problems [8, 9].

This paper looks at the differences between running scientific workflows on the cloud and running them on the Grid, using the Montage [10] application and Pegasus-WMS, a scientific workflow software [11, 12]. Our perspective is not that of a computer scientist trying to analyze detailed performance metrics, but rather that of a domain scientist who can use a number of different computing environments to perform computations. In our scenario, four different sizes of workflows were run on individual virtual machines and on a virtual machine cluster on the Nimbus science cloud [13] as well as on a local cluster without an overlying virtual environment to explore the differences in overhead. Using this approach, the usability of cloud computing for large-scale scientific experiments is explored.

## 2. Motivation

With some of the limitations and inefficiencies of Grid computing exposed, such as heterogeneous execution environment, difficulties of acquiring resources when needed, etc., it may be surprising that large-scale computing using a virtual cloud is not more widespread. However, cloud technology is gaining in prominence as the forerunners of mainstream computing are beginning to use their marketing power. Companies such as Google, IBM, Amazon, and Yahoo! are aiming to make supercomputing power available to the masses, not just a relatively tiny number of skilled users [2]. Services like online storage, social networking sites, e-commerce, and web-based email are all examples of how cloud computing has already begun to become mainstream. In addition, users in scientific and research communities will have the opportunity to access resources as they need them and only pay for what they use instead of paying for a certain amount of time – scheduled in advance – on a certain number of resources. Large corporations as well as individuals will be able to use cloud resources for their computing and storage needs.

The potential applications of cloud computing are many: financial applications, health care services, business enterprises and many others. Unlike other technology advances such as the web and the Grid, this new model of computing is being initiated in the business sector [5, 1] rather than in the science domain. Thus, the benefits of cloud computing to the scientific community are largely unknown and many questions are being asked by domain scientists about the promise and the pitfalls of clouds [14]. Additionally, very little research has been conducted in comparison with what has been done with Grid technology.

The main goal of this work was to determine the applicability of cloud computing to large-scale scientific workflows that are usually run on a Grid. Using the Pegasus-WMS software, workflows from the Montage application were run both on a Grid and on a science cloud to test for overall differences in time and overhead costs.

## 3. Approach

In order to provide a comparison between running on the cloud and on the Grid, Montage workflows of varying sizes were run in four different environments. Combinations of individual virtual machines as well as a virtual cluster were run on a science cloud set up at the University of Chicago [13], and the results from these two environments were compared with the results of running workflows on a local machine and a local Grid cluster. Different methods were implemented to configure each environment in order to improve the comparison between them, such as limiting the number of jobs to be released onto the local Grid cluster since it used more nodes than the science cloud as well as running workflows using several virtual machines that were not set up as a virtual cluster. Much work has been done to date on detailed comparisons of various aspects of virtual machines, including differences between various virtualization approaches, performance of the I/O subsystem, etc, [15, 16, 7, 17, 18]. In this paper we aim at providing a comparison from the perspective of a scientist who needs to run the computations and has access to limited local resources and to a potentially much larger pool of virtual resources.

### 3.1 Experimental Setup

Before any workflows could be run on the Nimbus science cloud, the appropriate software needed to be installed on a virtual machine image. A single virtual machine was launched, and necessities including Globus [19] (GridFTP, and GRAM), xinetd, Condor [20], and Pegasus-WMS were installed. The VM was terminated and saved to an image so that the software would be available for each VM that was deployed using that image.

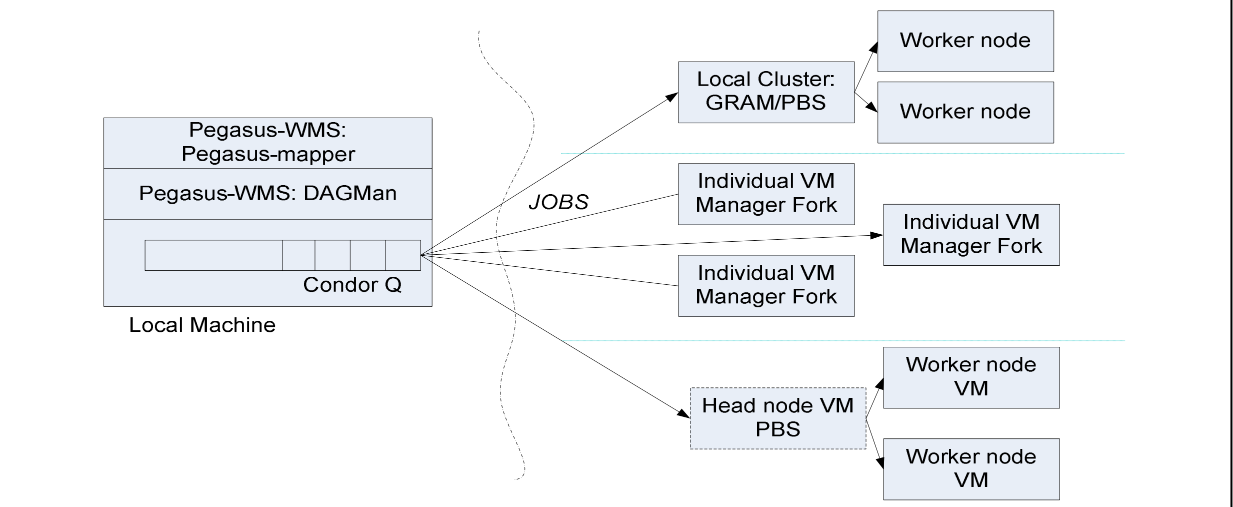

**Figure 1: Experimental Setup.**

Figure 1 shows the process of creating and running a workflow from start to finish using a local machine, a Grid cluster, a virtual cluster run on the science cloud, and a single virtual machine run on the science cloud. Although one can potentially run workflows across these four different environments, in this paper, we ran the workflow in each of them in separation. To run a workflow, a directed acyclic graph XML file (DAX) representing the Montage computation was generated using a domain-specific code. The DAX workflow representation is a high-level resource-independent description of the applicable tasks and their dependencies. Using the Pegasus-WMS software to plan out the workflow and map each job to an appropriate resource or node, a directed acyclic graph file was created. This type of file contains specific information needed by Condor to submit each job to the correct resource. After planning, a command given by Pegasus allows the workflow to be submitted to Condor. Condor's directed acyclic graph manager (DAGMan) [21] sends jobs to Condor, which uses Condor-G [22], to be run on the sites that have been selected during planning.

## 3.2 Tools

Several tools were used to set up this experiment, the first of which are the Virtual Workspace Service and the workspace cloud client [6]. The Virtual Workspace Service is installed on a Grid, allowing users to create virtual environments using the workspace cloud client. For part of this experiment, Xen virtual machines [15] were deployed using the cloud client on the University of Chicago's sixteen-node TeraPort cluster that had the Virtual Workspace Service installed to create the Nimbus science cloud [13]. The cloud client allows a user to upload virtual machine images, download, modify, delete, or save copies of preexisting images, deploy images as virtual machines, and deploy virtual clusters from virtual images.

In order to obtain the workflows that were run, the Pegasus mapper was used to map high-level workflow descriptions onto distributed resources. Pegasus stands for Planning for Execution in Grids and is used for numerous applications in many areas of science [23-25], including Montage, an astronomy application which creates science-grade image mosaics of the sky. The workflow manager depends on Condor DAGMan, which launches workflow tasks and maintains the dependencies between them. Pegasus and Condor were both installed on a running virtual machine, and then the VM was saved as an image that could be used to deploy future VMs. A convenient benefit of using the cloud is that once an image is developed, it is not necessary to reinstall software each time a new VM is launched. The configurations that are set up while a VM is running can be saved, and the image can be used to launch multiple copies of the original VM.

Parts of the Globus Toolkit [19] were used in combination with Pegasus and Condor to run the workflows, most notably GridFTP [26] and GRAM. GridFTP is a high-speed transfer service that expands the File Transfer Protocol to include features needed for Grid computing applications [27]. Globus Resource Allocation Manager, or GRAM, provides a single protocol for communicating with different job schedulers. It is used to locate, submit, monitor, and cancel jobs on remote resources while addressing security, reliability, and performance issues. This experiment used a pre-Web Services GRAM service.

## 3.3 Environments

We ran our experiments in four different environments: 1) on a local machine, 2) on multiple VMs, where each VM is treated as an independent computational site, 3) on a local cluster, and 4) on a Virtual cluster. We note that the virtual machines each have 2 CPUs thus we compared the execution not based on machines but based on CPU counts. Table 1 shows the characteristics of the machines we used in our experiments.

**Table 1: The type of machines used in the study.**

| | *Processor Type* | *Memory* | *Architecture* | *Disk space* | *Scheduler* |
|---|---|---|---|---|---|
| Local Machine | Pentium, 2.2Ghz | 1GB | 32 bit | > 100GB | fork |
| VM | AMD 248, 2.2 Ghz | 3.5GB | 64 bit | 1.0GB free local | fork |
| Local Cluster | Xeon | 2.4GB | 32 bit | > 100 GB | pbs |
| VM Cluster | AMD 248, 2.2 Ghz | 3.5GB | 64 bit | 1.5GB free shared | pbs |

The local Grid cluster used was comprised of eight compute nodes, and it had GRAM, GridFTP, and PBS software installed on it. Individual Xen virtual machines were deployed on nodes of the University of Chicago TeraPort cluster. The TeraPort cluster is composed of sixteen nodes, but each VM was only run on one node. GridFTP, GRAM, and the other required software was installed on a template image before deployment. The third environment on which the workflows were run was a virtual cluster. The virtual cluster was composed of two to five VMs: a head node and one to four worker nodes. The VMs each occupied one node of the TeraPort cluster, just as with a single VM.

## 4 Experimental Results and Analysis

We conducted the experiments using 4 different size mosaics: 0.1 deg$^2$, 0.2 deg$^2$, 0.5 deg$^2$, and 1deg$^2$, corresponding to workflows with 50, 74, 150, and 426 application jobs respectively. There are also additional jobs in the workflow, inserted by Pegasus, to perform the data movement if any is necessary, to cleanup the data as the computation progresses [28, 29], and to register the newly created data products. Another important metric in the study is the total footprint of the workflow. This footprint is 58MB, 122MB, 338MB, and 1,090MB for increasing mosaic sizes. The application executables add an additional 11MB to total footprint of each workflow. In all cases input data was available on a machine on the same local area network as the local machines. The virtual environments were connected to the local submit host via a wide area network. As a result, data had to be automatically stage-in by Pegasus-WMS. In these experiments, the Montage workflow was not optimized for performance or data footprint.

The following graphs show the execution times for the workflows of various sizes on resources with 1, 2, and 4 CPUs. The runtimes do not include the set up and teardown of the virtual environment. The local execution represents execution of the workflow on the local host. Figure 2 shows the execution of the workflow on 2 CPUs (for comparison we also include the local execution on 1 CPU). The local cluster was configured to use no more than 2 CPUs at any one time. We used 1 VM with 2 CPUs, and a virtual cluster with one headnode that runs PBS and one worker node (with 2 CPUs). We note that the 1 degree square runs conducted on individual VMs did not complete due to lack of disk space in the image. We also notice that the performance of a single processor machine outperforms that of other configurations, even though the local host is less powerful than others. Although we don't see that in these experiments, the size of the mosaics one can run on a single machine is limited by the amount of memory on that machine, so using a single machine for large-scale mosaics is not possible. Additionally, the number of simultaneous mosaics that can be executed locally is limited. As a result relying on a single machine to do many large-scale mosaics is not feasible in general. Figure 3 shows the execution of the workflow on 4 CPUs (and includes the local 1 processor execution for comparison). Again we see that local execution is the shortest. In both the 2CPU and 4 CPU case, the virtual cluster performs better than the local cluster. In the 4CPU case individual VMs perform similarly to the local cluster. In order to understand the reason for good performance of the local host versus the performance of the other platforms, we examined the constituents of the end-to-end runtime. These are composed of the Computation Time (here it also includes data staging in and out), the time it takes DAGMan to release jobs ready to run into the Condor queue, the time jobs wait in the Condor queue, and the time the jobs wait in the queue in the scheduler at the remote resource. Figure 4 shows the breakdown of the quantities for the 0.5 degree square Montage workflow running on 4 CPUs. We see that during local execution, the computational time is long—because the processor is slow. The Figure also shows that the DAGMan overhead is constant for all the runs. What is different is the overhead associated with the Condor queue. Once we are outside of the local machine, jobs need to go from the Condor queue across a network to the resource. Jobs incur the network delay as well as the delay related to the acknowledgment of the remote resource that the job has been accepted. We can see that this delay is particularly long for the virtual cluster. In our runs, the jobs also wait longer in the remote queue in the virtual cluster case. There is virtually no remote queue time seen in the virtual machine because these are managed by fork. We also analyzed the computational time of the individual VMs because it was longer than that of the virtual cluster. The reason for the difference is that the cluster uses a shared file system to access data during the workflow execution, whereas the individual VMs need to share data via GridFTP transfers. The time for these data transfers constitutes 6% of the overall computation time.

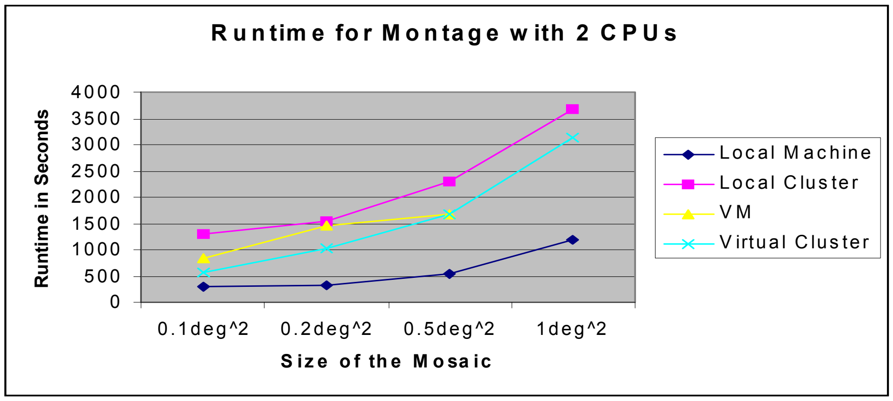

**Figure 2: Runtime of the Montage Application on 2 CPUs.**

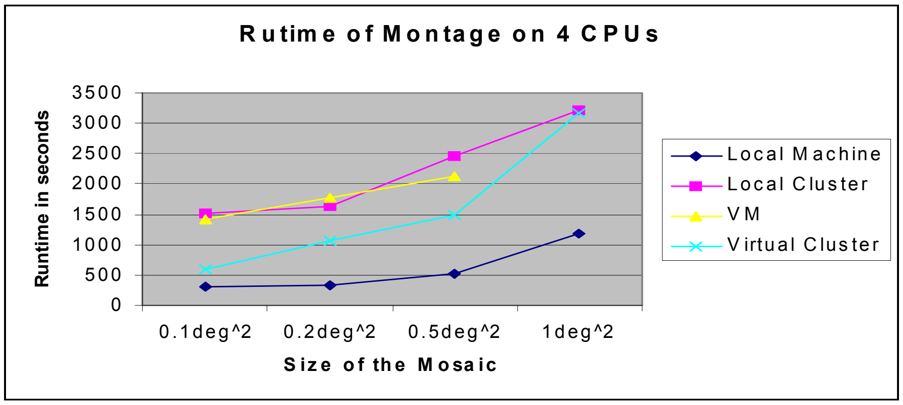

**Figure 3: Runtime of the Montage Application on 4 CPUs.**

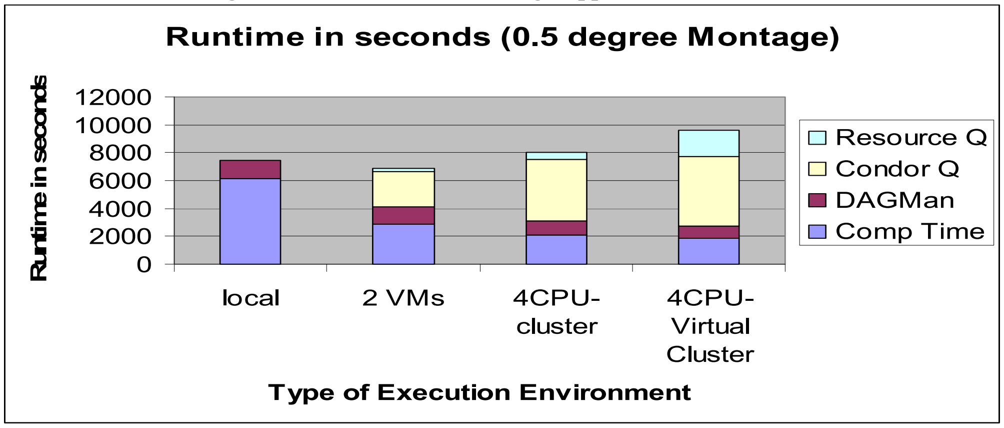

**Figure 4: Overheads in the Workflow Execution in Various Environments.**

## 5. Related Work

Virtualization technologies have taken two main approaches [30]. VMware allows multiple unmodified operating systems to run on a single host by implementing a full virtualization of the hardware and has more recently developed software for paravirtualization [18]. Denali [31], IBM rHype [32], and Xen utilize paravirtualization for decreased overhead and potentially higher performance, but paravirtualization is limited in that it can only host specially modified operating systems. A recent performance study of hypervisors showed that the VMware virtual machine monitors perform better than their Xen counterparts in a variety of tasks, but Xen is still widely used as it is open source [18].

A study for high-performance computing was done to compare Linux to Xen [33]. It was found that Xen attained slightly higher memory bandwidth but was slower with disk I/O speed. Another study using a FOAM MPI-based climate application showed a slowdown of only about 5% on 16 processors [7]. Our work differs in that we don't focus on virtualization technologies but rather on their benefits for science applications.

## 6. Discussion and Conclusions

In this paper, we aimed to quantify the performance differences an application scientist would see while running workflows on various types of resources, including both physical and virtual environments. Although we see that in some cases running in a local environment is sufficient, it is not a scalable solution. However, virtual environments can provide the necessary scalability. In multi-resource environments, we noticed large overheads of jobs waiting in the Condor and resource queues. This is particularly exacerbated in applications such as Montage where the average execution time of a job is on the order of a few seconds. However, clustering techniques [34] that cluster small jobs together can greatly reduce the scheduling overheads. In the future, we plan apply these techniques in our virtual environment evaluation studies. Finally, we also noticed that disk space management is critical in virtual environments. When a virtual image is created, the size of the disk is fixed. Having a too small initial virtual disk size can adversely affect the execution of the application.

## Acknowledgements

This work was supported in part by NSF under grant #CCF-0725332, by the Computing Research Association's Distributed Research Experiences for Undergraduates program, and by USC's Women in Science and Engineering program. This research made use of Montage, funded by NASA's ESTO, Computation Technologies Project, under Cooperative Agreement #NCC5-626 between NASA and Caltech. Montage is maintained by the NASA/IPAC Infrared Science Archive. The authors would also like to thank Shishir Bharathi for his performance analysis codes.

## References

[1] S. Lohr, "Google and IBM Join in Cloud Computing Research," *New York Times,* 2007.
[2] A. Ricadela, "Computing Heads for the Clouds," in *Business Week*, November 16, 2007.
[3] I. Foster and C. Kesselman, "The Grid: Blueprint for a New Computing Infrastructure," Morgan Kaufmann1999.
[4] I. Foster, C. Kesselman, et al., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *IJHPCAs,* 2001.
[5] "Amazon Elastic Compute Cloud,"http://aws.amazon.com/ec2/.
[6] http://workspace.globus.org/, Virtual Workspaces
[7] I. Foster, T. Freeman, et al., "Virtual Clusters for Grid Communities," *CCGRID,* 2006.
[8] I. Foster, R. Figueiredo, et al., "A case for grid computing on virtual machines," *ICDCS* 2003.
[9] K. Keahey, K. Doering, et al., "From sandbox to playground: dynamic virtual environments in the grid," 2004, pp. 34-42.
[10]G. B. Berriman, E. Deelman, et al., "Montage: A Grid Enabled Engine for Delivering Custom Science-Grade Mosaics On Demand," in *SPIE Conference 5487:* 2004.
[11]E. Deelman, G. Mehta, et al., "Pegasus: Mapping Large-Scale Workflows to Distributed Resources," in *Workflows in e-Science*, I. Taylor, E. Deelman, et al., Eds.: Springer, 2006.
[12]E. Deelman, G. Singh, et al., "Pegasus: a Framework for Mapping Complex Scientific Workflows onto Distributed Systems," *SciProg Journal,* vol. 13, pp. 219-237, 2005.
[13]"Nimbus Science Cloud," http://workspace.globus.org/clouds/nimbus.html.
[14]E. Deelman, G. Singh, et al., "The Cost of Doing Science on the Cloud: The Montage Example," in *SC'08* Austin, TX2008.
[15]P. Barham, B. Dragovic, et al., "Xen and the art of virtualization," *ACM SOSP,* pp. 164-177, 2003.
[16]B. Clark, T. Deshane, et al., "Xen and the art of repeated research," *USENIX Annual Technical Conference, FREENIX Track,* pp. 135–144, 2004.
[17]A. Norton, D. Vanderster, et al., "Evaluation of Virtual Machines for HEP Grids," *Computing in HEP,* 2006.
[18]VMWare, "A Performance Comparison of Hypervisors, "http://www.vmware.com/pdf/hypervisor_performance.pdf.
[19]"Globus,"http://www.globus.org.
[20]M. Litzkow and M. Livny, "Experience With The Condor Distributed Batch System," in *IEEE Workshop on Experimental Distributed Systems*, 1990.
[21]"DAGMAN (Directed Acyclic Graph Manager)," http://www.cs.wisc.edu/condor/dagman/, 2004.
[22]J. Frey, T. Tannenbaum, et al., "Condor-G: A Computation Management Agent for Multi-Institutional Grids," *HPDC 01*.
[23]D. A. Brown, P. R. Brady, et al., "A Case Study on the Use of Workflow Technologies for Scientific Analysis: Gravitational Wave Data Analysis," in *Workflows for e-Science*, 2006.
[24]P. Maechling, E. Deelman, et al., "SCEC CyberShake Workflows---Automating Probabilistic Seismic Hazard Analysis Calculations," in *Workflows for e-Science*, 2006.
[25] V. Nefedova, R. Jacob, et al., "Automating Climate Science: Large Ensemble Simulations on the TeraGrid with the GriPhyN Virtual Data System," in *e-Science*, 2006.
[26]W. Allcock, J. Bester, et al., "Data Management and Transfer in High-Performance Computational Grid Environments," *Parallel Computing,* 2001.
[27] M. Feller, I. Foster, et al., "GT4 GRAM: A Functionality and Performance Study," *TeraGrid Conference,* 2007.
[28] A. Ramakrishnan, G. Singh, et al., "Scheduling Data - Intensive Workflows onto Storage-Constrained Distributed Resources," in *CCGrid* 2007.
[29]G. Singh, K. Vahi, et al., "Optimizing Workflow Data Footprint " *Scientific Programming Journal,* vol. 15, 2007
[30]M. Rosenblum and T. Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends," *Computer,* 2005.
[31]A. Whitaker, M. Shaw, et al., "Denali: Lightweight virtual machines for distributed and networked applications," 2002.
[32]J. Xenidis, "rHype: IBM Research Hypervisor," *IBM Research,* 2005.
[33]L. Youseff, R. Wolski, et al., "Paravirtualization for HPC Systems," *ISPA,* vol. 4331, p. 474, 2006.
[34]G. Singh, M. H. Su, et al., "Workflow task clustering for best effort systems with Pegasus," *Proceedings of the 15th ACM Mardi Gras conference,* 2008.