HAL
open science

A Taxonomy and Survey of Scientific Computing in the Cloud

Amelie Chi Zhou, Bingsheng He, Shadi Ibrahim

To cite this version:

Amelie Chi Zhou, Bingsheng He, Shadi Ibrahim. A Taxonomy and Survey of Scientific Computing in the Cloud. Big Data: Principles and Paradigms, Morgan Kaufmann, 2016, eScience and Big Data Workflows in Clouds: A Taxonomy and Survey, 978-0-12-805394-2. 10.1016/B978-0-12-805394-2.00018-0 . hal-01346745

HAL Id: hal-01346745

https://inria.hal.science/hal-01346745v1

Submitted on 19 Jul 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Taxonomy and Survey of Scientific Computing in the Cloud

Amelie Chi Zhou, Bingsheng He, Shadi Ibrahim

Abstract

Cloud computing has evolved as a popular computing infrastructure for many applications. With (big) data acquiring a crucial role in eScience, efforts have been made recently exploring how to efficiently develop and deploy scientific applications on the unprecedentedly scalable cloud infrastructures. We review recent efforts in developing and deploying scientific computing applications in the cloud. In particular, we introduce a taxonomy specifically designed for scientific computing in the cloud, and further review the taxonomy with four major kinds of science applications, including life sciences, physics sciences, social and humanities sciences, and climate and earth sciences. Due to the large data size in most scientific applications, the performance of I/O operations can greatly affect the overall performance of the applications. We notice that, the dynamic I/O performance of the cloud has made the resource provisioning an important and complex problem for scientific applications in the cloud. We present our efforts on improving the resource provisioning efficiency and effectiveness of scientific applications in the cloud. Finally, we present the open problems for developing the next-generation eScience applications and systems in the cloud and conclude this chapter.

1 Introduction

The development of computer science and technology widens our view to the world. As a result, the amount of data observed from the world to be stored and processed has become larger. Analysis of such large-scale data with traditional technologies is usually time-consuming and requires a large-scale system infrastructure, and therefore can hinder the development of scientific discoveries and theories. eScience offers scientists with the scope to store, interpret, analyse and distribute their data to other research groups with the state-of-the-art computing technologies. eScience plays a significant role in every aspect of scientific research, starting from the initial theory-based research through simulations, systematic testing and verification to the organized collecting, processing and interpretation of scientific data. Recently, cloud computing is becoming a popular computing infrastructure for eScience. In this chapter, we review the status of cloud-based eScience applications and present a taxonomy specifically designed for eScience in the cloud.

eScience is an important type of big data applications. eScience is also considered as the “big-data science”, which includes broad scientific research areas, from the areas that are close to our everyday life (e.g., biological study), areas concerning the planet that we live on (e.g., environmental science), to areas related to the outer space (e.g., astronomy studies to find the origins of our universe). Although the term of eScience has only been used for about a decade, the study of eScience problems started much earlier. In the early days, scientists were not able to efficiently realize the value of scientific data due to the lack of technologies and tools to capture, organize and analyse the data. Technological advances such as new computing infrastructures and software protocols have brought great opportunities for eScience projects in various fields [16, 38, 31, 51]. For example, Grid computing has greatly advanced the development of eScience. Many eScience applications are becoming data-driven. For example, with the development of modern telescope, the Large Synoptic Survey Telescope (LSST) [69] obtains 30 trillion bytes of image data of the sky every day. Currently, almost all major eScience projects are hosted in the grid or cluster environments [81]. With aggregated computational power and storage capacity, grids are able to host the vast amount of data generated by eScience applications and to efficiently conduct data analysis. This has enabled researchers to collaboratively work with other professionals around the world and to handle data enormously larger in size than before.

In the recent few years, the emergence of cloud computing has brought the development of eScience to a new stage. Cloud computing has the advantages of scalability, high capacity and easy accessibility compared to grids. Recently, many eScience projects from various research areas have been shifting from grids to cloud platforms [43, 44, 39]. It breaks the barrier of doing eScience without a self-owned

large-scale computing infrastructure. In this chapter, we review the current status of scientific applications in the cloud, and propose a taxonomy to clearly classify the related projects, according to the infrastructure, ownership, application, processing tools, storage, security, service models and collaboration aspects. We find that, for scientific applications in the cloud, resource provisioning is a major concern for the monetary cost and performance optimizations of the applications. In this chapter, we briefly introduce our initial efforts on the resource provisioning problems of scientific workflows in the cloud.

Both eScience and cloud computing are rapidly developing and becoming more mature. It is timely to examine the efforts and future work for scientific computing in the cloud. This chapter focuses especially on eScience projects in the cloud. Due to the pay-as-you-go service model of the cloud, monetary cost and performance are two important concerns for eScience applications in the cloud. In this chapter, we introduce our existing studies on cloud resource provisioning to optimize the cost and performance for eScience in the cloud. We compare the advantages and weaknesses against eScience in the grid to discuss the obstacles and opportunities of eScience services in the cloud.

The rest of this chapter is structured as follows. Section 2 introduces the background of eScience history and grid-based and cloud-based eScience. Section 3 gives the taxonomy and reviews the current status of eScience in the cloud. Section 4 introduces some of our efforts on optimizing the resource provisioning problems of scientific applications in the cloud. Lastly, we discuss the open problems for scientific computing on the cloud in Section 5 and summarize this chapter in Section 6.

2 Background

In this section, we briefly discuss some history remarks on scientific computing development, particularly for eScience. Next, we focus our review on the grid based scientific computing, and introduce cloud computing.

2.1 History Remarks

Computer infrastructures have long been adopted to host scientific data sets and computations. eScience is a new science paradigm that uses distributed computing infrastructures for computation, simulation and analysis. In addition, the scientists can make use of high speed network to access huge distributed and shared data collected by sensors or stored in database. Computing infrastructures allow scientists around the world to share knowledge and resources, and to build close scientific collaborations.

Table 1Development stages of the scientific computing.

Stage	Data Generated	Research Period	Infor. Tech.
Manual	By hand	Ad-hoc	Paper and pencil
(Semi-) Automated	With the help of machinery	Short-term	Computer assisted
Large-scale Sensing	From satellites and sensors around the world	Real-time	Cluster and grid

The term *eScience* was first introduced in 1999 and was further interpreted by more researchers since then [18]. During the development of eScience, it has gone through roughly three stages. Table 1 shows the major development stages that scientific computing has gone through. We review the history in the following dimensions.

Dimension 1: the evolution of science. We observed that technology (particularly information technology) is one of the main driving factors in pushing science forward. From the perspective of experimental methods, eScience first used *manual* measurements: meaning the measurements were taken by hand, not using machinery or electronics to fulfil the function. Then with the development of technology, machinery such as computers and metering instruments were used to help in the measurements, but with manual operations still involved. This stage is called the *semi-automated*

stage. After this stage, machinery took a greater part in the measurements and eScience evolved to the *automated* stage where machines took almost all the work with the least of human involvement. Technologies such as high performance computers, sensor networks and various experimental software make the eScience measurements evolve to the *large-scale sensing* stage [1].

Take the research in Meteorology for example, in the early stage (classified to manual stage), researchers use thermometer, barometer and hygrometer to measure the temperature, air pressure, water vapour and write down the records. In the 19th century (classified to semi-automated stage), breakthroughs occur in meteorology after observing the development of networks. The meteorological data collected in local meteorological observatories are transmitted through networks and then are gathered together by different spatial scales to study the various meteorological phenomena. Since the 20th century (classified to automated stage), with the adoption of radars, lasers, remote sensors and satellites into the meteorological research, collecting data of a large area is no longer a challenging problem and special instruments together with the automation of computers can automatically fulfil the measuring tasks. At the end of the 20th century (classified to large-scale sensing stage), large scale observation experiments are performed. The experiments relied on satellites, meteorological rockets, meteorological observatories on the ground around the world, automatic meteorological stations, airplanes, ships, buoy stations and constant level balloons. These instruments were combined to form a complete observing system to automatically measure the meteorological variables world-wide.

Dimension 2: the length of research period. eScience has gone through *ad-hoc* stage when research was done just for a specific problem or task, and not for other general purposes. Later, in the *short-term plan* stage, researchers made plans in priori for their problems about what to do in what time, so that a project of a short term could be kept on schedule. In *real-time* stage, the research is subject to real-time constraints, such as the experimental data are collected in real-time and the system needs to give out results also in real-time. This evolution on research period also require the experimental methods to be more efficient, and the support of high technology as we will discuss next.

Dimension 3: technology. eScience has gone through *paper and pencil* stage when no machinery was involved in our research and human work with paper and pencil was the only tool for science. When computers appeared, eScience was able to move to the *computer assisted* stage when computers played a great role in helping with complex calculations and solving logical problem. With the scientific problems getting more complicated and traditional computers not sufficient for the computing power required, *cluster and grid* are coming to scientists' vision and help them solving many data-intensive or compute-intensive problems within reasonable time which is not possible on traditional computers.

We summarize our findings in the three dimensions. Initially, scientists only deal with specific problems using manual methods such as doing theoretical calculation using paper and pencil at early days. As problems getting more complicated, more planning is needed for the research and semi-automated and automated methods are also required in the research during this time. Computers are used and when problem scale gets larger, and new technologies such as clusters and grids are applied for solving the problems faster. New challenges from sciences have risen. We have witnessed the recent challenges including a large-scale of scientific data (big data) and the requirement of real-time processing.

2.2 Grid-based eScience

Current major eScience projects are mostly hosted in the grid or HPC cluster environments. With aggregated computational power and storage capacity, grids have been considered an ideal candidate for scientific computing for decades. There are many labs around the world working on grid based projects, such as GridPP in UK, XSEDE in US, CNGrid in China, France Grilles in France, D-Grid in Germany and Kidney-Grid in Australia. In the following, we present some details about the grid infrastructure in UK, USA and China.

In UK, particle physicists and computer scientists have been collaboratively working on the GridPP project, aiming at providing distributed computing resources across the UK for particle physicists working on the Large Hadron Collider experiments at CERN [81]. These organizations include all of the UK universities and institutions that are working as members of this project. At the end of 2011, the project has contributed a large number of resources (29,000 CPUs and 25 Petabytes of storage) to the worldwide grid infrastructure.

The Grid Infrastructure Group (GIG) along with eleven resource provider sites in the United States have initiated an eScience grid computing project called TeraGrid. In 2007, TeraGrid provided more than 250 Teraops of computation resources, more than 30 Petabytes of data storage resources and over 100 databases of different disciplines for many researchers. The resources had grown to 2 Petaflops of computing capability and over 60 Petabytes storage in the late 2009. In 2011, after the termination of TeraGrid project, National Science Foundation (NSF) funded another high-performance project named XSEDE (Extreme Science and Engineering Discovery Environment) as a follow-up of TeraGrid.

China National Grid (CNGrid) has quickly grown to serve more than 1400 users including both research institutes and commercial companies, providing more than 380 Teraflops of computation resources and more than 2 Petabytes of shared data storage resources. Since 2009, this project has built three Petaflop-level supercomputers, in which Tianhe-1 was ranked the fastest supercomputer in the top 500 supercomputers in 2010 [10].

Besides nation-wide initiatives, volunteer computing projects have taken place to build grid platforms with public donation of computing resources. For example, SETI@home [11] is such a volunteer computing project employing the BOINC software platform to search for extra-terrestrial signals with the spare capacity on home and office computers.

The strength of grid computing has attracted many scientific applications.

- First, since governments are very concerned about the research on grid and frontier scientific research, most of the grid-based projects are funded by national funding. Resourceful funding offers good chances to boost eScience in the grid.
- Second, sharing the vast computational and storage resources from grids becomes possible.
- Third, the tools and software developed on grid can benefit more research groups besides the developers themselves. This strength can save a lot of development time for the projects developed on the grids.

While the grid is the dominant infrastructure for eScience, it faces a number of limitations. First, due to the limitation of its structure, grid is not able to provide the elasticity required by most scientific projects which are pursuing cost efficiency. Second, it is not easy to get access to grid resources for everyone because a program getting access to grid resources needs to be authorized on the project's behalf and resources would then be distributed to this project as a whole. Since grids are mostly national-wide initiatives, getting the authorization is very hard for most small-scale projects. Third, while Grid offers access to many heterogeneous resources, many applications need very specific and consistent environments.

2.3 Cloud Computing

According to the definition of the National Institute of Science and Technology (NIST), cloud computing is

"*The delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility (like the electricity grid) over a network (typically the Internet)" [61].*

Cloud computing comes into popularity in the early 2000's. Officially launched in 2006, Amazon Web Service (AWS) is the first utility computing platform that provides computation resources as services to external customers. Many other cloud service providers, including Microsoft Azure, Google Cloud

Platform and Rackspace, have come into the market since then. Open-source systems and research projects are developed to facilitate the use of cloud. For example, Eucalyptus allows deploying AWS-compatible private and hybrid cloud computing environments. The OpenNebula toolkit is designed for building private and hybrid clouds with different cloud model and flexibility from Eucalyptus.

Cloud computing bares many similarities and differences with grid computing. In the year 2008, Foster et al. [34] has compared clouds and grids mainly from a technological perspective. Compared to the grid, cloud has better scalability and elasticity.

- When developing applications on the grid infrastructure, it is not easy to scale up or down according to the change of data scale. In cloud, with the use of virtualization, clients can scale up or down as they need and pay only for the resources they used.
- Virtualization techniques also increase the computation efficiency as multiple applications can be run on the same server, increase application availability since virtualization allows quick recovery from unplanned outages with no interruption in service and improves responsiveness using automated resource provisioning, monitoring and maintenance.
- Cloud has easier accessibility compared to grid. Users can access to commercial cloud resources through log in and use the resources as they need as long as they could pay with a credit card. In this case, even small-scale scientific projects can also have the chance to use powerful clusters or supercomputers on their compute-intensive or data-intensive projects.

Scalable data analytics for big data applications in the cloud has become a hot topic. Due to the large volumes of input data, many data analytics tasks usually take a long time to finish the data processing. However, in many big data applications such as weather analytics, the input data must be analysed in a cost- and time-effective way to discover the value of the data. The easy accessibility and good scalability of the cloud make it a perfect match for serving scalable (big) data analytics applications. More and more eScience applications are beginning to shift from grid to cloud platforms [22, 76]. For example, the Berkeley Water Centre is undertaking a series of eScience projects collaborating with Microsoft [43, 44, 39]. They utilized the Windows Azure cloud to enable rapid scientific data browsing for availability and applicability and enable environmental science via data synthesis from multiple sources. Their BWC Data Server project is developing an advanced data synthesis server. Computer scientists and environmental scientists are collaborating to build new tools and approaches to benefit regional and global scale data analysis efforts [43, 44].

3 Taxonomy and Review of eScience Services in the Cloud

There have been various cloud computing techniques for eScience. We need a taxonomy to reflect the interplay between eScience and cloud computing. The taxonomy in this section gives a clear classification of cloud computing techniques used in eScience services from several perspectives, including the computation *infrastructure* for eScience applications, the *ownership* of cloud infrastructures, the eScience *application* types, the *processing tools* used for eScience applications, the *storage* model, the *security* insurance method, *service models* of the cloud and the *collaboration* goal between different research groups. Figure 1 summarizes our taxonomy. Some are mainly from eScience's perspective, and some are mainly from cloud computing's perspective.

3.1 Infrastructure

The infrastructure of cloud provides access to compute and storage resources for eScience applications in an on-demand fashion. Cloud shares some similarities with Grid while at the same time is modified to overcome the limitations of Grid. Roughly, we can classify the infrastructure into three kinds: grid, grid with virtualization (i.e., a hybrid approach), and cloud.

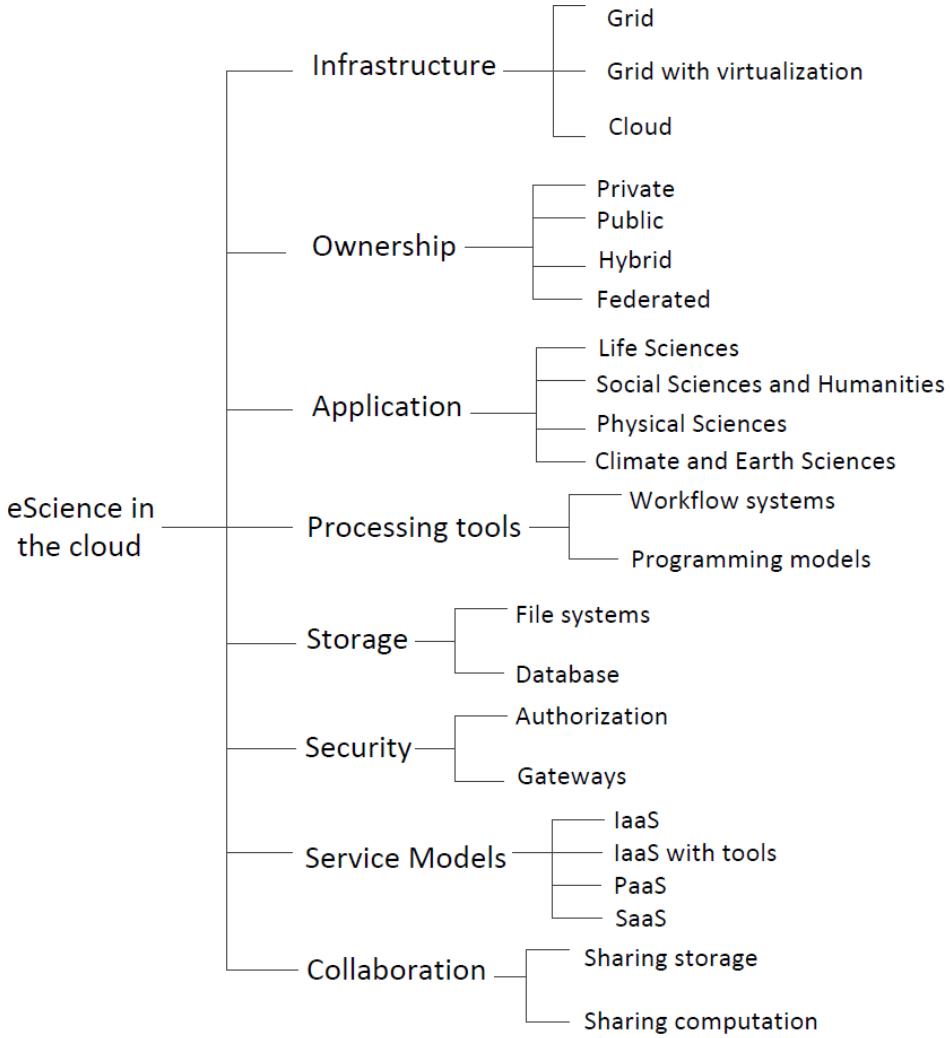

Figure 1 Taxonomy of eScience in the Cloud

One characteristic of Grid is that it assigns resources to users in the unit of organizations and each individual organization holds full control of the resources assigned to it. However, such a coarse-grained resource assignment is not efficient. There are efforts in Grid to use virtualization to address the deficiency issue. Nimbus scientific cloud is one such effort that provides a virtual workspace for dynamic and secure deployment in the Grid. Virtualization hides from users the underlying infrastructures which are usually heterogeneous hardware and software resources, and provides the users with fine-grained resource management capabilities.

As for cloud infrastructures, several national cloud initiatives have also been announced to provide on-demand resources for governmental purposes [42], such as the US Cloud Storefront [9], the UK G-Cloud [8], and the Japanese Kasumigaseki [7] cloud initiatives. Many industry players also dive in the cloud business and provide users with seemingly infinite public cloud resources. With the popularity of cloud, many eScience applications have been deployed in the general public cloud infrastructures such as Amazon EC2, Windows Azure to benefit from its high performance, scalability and easy-access [43, 44, 39, 64, 32, 57].

There have been a number of studies comparing the performance of cloud with other infrastructures. The NG-TEPHRA [57] project performed a volcanic ash dispersion simulation on both grid and cloud, using the East Cluster at Monash University and the Amazon EC2 computing resources separately. Experiments show efficient results on both platforms and the EC2 results have shown very small

differences in their standard deviation, indicating the consistent QoS of the cloud. Cloudbursting [39] implemented its satellite image processing application with three different versions: an all-cloud design on Windows Azure, a version that runs in-house on Windows HPC clusters and a hybrid cloudbursting version that utilizes both in-house and cloud resources. The hybrid version achieves the best of the previous two versions, namely the development environment of a local machine and the scalability of the cloud. Their experimental results showed that the application is benefiting from the hybrid design, both on execution time and cost.

From the existing studies, we find that the performance comparison between cloud and HPC is application dependent. Due to the scheduling and communication overhead, the applications involving large and frequent data transfer over multiple computation nodes usually perform worse on the cloud than on HPC clusters which are equipped with high bandwidth network. In contrast, the advantage of cloud is its high scalability. Users can easily and quickly scale up and down their applications as needed, without wasting too much money. Applications such as Cloudbursting [39] can benefit from this characteristic of the cloud.

3.2 Ownership

The ownership of cloud infrastructures can be classified as the following types: private, public, hybrid and federated. They have different levels of security and ownership scope.

Private clouds are infrastructures operated only by a single organization for internal use. The security level of private clouds is the highest among the four types. eScience applications which have high security requirements or possess highly sensitive data can be deployed on private clouds. OpenNebula is the first open-source software supporting private clouds deployment and is widely used by industry and research users [59].

In contrary, public clouds are more open, with their application, storage and other resources available to the public on the pay-as-you-go basis.

A federated cloud, also known as community cloud, is a logical combination of two or more clouds from either private, public or even federated clouds. In this combination, the two or more clouds often have similar goals in security, compliance and jurisdiction. Many countries have built federated clouds to support the research and education purpose of their own country. The EGI Federated Cloud Task Force [2] is a federation of academic private clouds to provide services for the scientific community. It has been used by a wide areas of eScience applications, including Gaia (a global space astrometry mission [35]) and the Catania Science Gateway Framework (CSGF) [3].

A hybrid cloud utilizes cloud resources from both private and public clouds. The benefit of hybrid clouds in an off-loading manner. While the workload is bursting and the private cloud can no longer support users' requirements, users can then request resources from the public cloud to sustain the performance and scalability.

3.3 Application

Cloud computing techniques have been applied to various eScience applications. We have surveyed a lot of eScience papers and summarized them in the following four categories based on their areas of expertise: Life sciences [51, 56], Physical sciences [28, 37], Climate and Earth sciences [32, 43] as well as Social sciences and Humanities [23, 49].

We note that those application categories can overlap with each other. There is no absolute boundary between categories. Still, different categories have their own requirements on the cloud. The first three categories, i.e., life sciences, physical sciences and climate and earth sciences, are more focusing on extending their works to large-scale datasets and thus require the cloud platform to deal with large-scale data analysis efficiently. The fourth category, i.e., social science and humanities, is more focusing on collaboration and thus requires the cloud platform to be easy for sharing.

Another observation is that, the development of eScience projects is *ad-hoc*. Some applications are developed on Amazon EC2 cloud [24], some on Windows Azure [39] while some others on both cloud platforms to verify their design [53].

However, it is not clearly explained why certain cloud platforms should be chosen over others in those projects. For example, MFA [24] is a Life Science project developed with the cloud services provided by Amazon. Its aim is to investigate whether utilizing MapReduce framework is beneficial to perform simulation tasks in the area of Systems Biology. The experiments on a 64 node Amazon MapReduce cluster and a single node implementation have shown up to 14 times performance gain, with a total cost of on-demand resources of $11. MODIS-Azure [43] is a Climate and Earth science application deployed on Windows Azure to process large scale satellite data. The system is implemented with the Azure blob storage for data repository and Azure queue services for task scheduling. However, neither of the two projects has technically explained their choice of cloud platforms. To compare the performance on different cloud platforms, a Physical science project Inversion [53] was deployed on both Amazon EC2 and Windows Azure with symmetry structures.

All these examples indicate that, it can be a challenging problem on how to choose cloud platforms for eScience applications. Due to the current *ad-hoc* implementation in specific cloud providers, the lessons learned during the implementation of one project may not be applicable to other projects or other cloud providers.

3.4 Processing Tools

From the perspective of processing tools, we have witnessed deployment of classic workflow systems in the cloud, new cloud oriented programming models such as MapReduce and DryadLINQ, and hybrid of such newly proposed tools and models.

Scientific workflows have been proposed and developed to assist scientists to track the evolution of their data and results. Many scientific applications use workflow systems to enable the composition and execution of complex analysis on distributed resources [27]. Montage is the example of a widely used workflow for making large-scale, science-grade images for astronomical research [37]. Workflow management systems (WMSes) such as Pegasus [70] and Kepler [72] are developed to manage and schedule the execution of scientific workflows. WMSes rely on tools such as Directed Acyclic Graph Manager (DAGMan) [12] and Condor [45] to manage the resource acquisition from the cloud and schedule the tasks of scientific workflows to cloud resources for execution. The application owners have to separately deploy and configure all the required software such as Pegasus and Condor on the cloud platforms to make their applications run. Such re-implementation and re-design work requires good effort from the application owners and should be avoided. Recently, the container techniques such as Docker [4] have been emerging to address this issue.

Emerging cloud oriented programming models have great promotion for the development of cloud computing. MapReduce is a framework proposed by Google in 2004 [26] for processing highly distributable problems using a large number of computers. This makes this framework especially suitable for eScience application users who may not be experts in parallel programming. We have observed the emergence of eScience applications adopting MapReduce framework for data-intensive scientific analyses [31].

Due to the large data size of many eScience applications, new data processing models, such as in-situ [73] and in-transit processing [17], have been proposed to reduce the overhead of data processing. An eScience workflow typically includes two parts, namely the simulation part which simulates the scientific applications to generate raw data and the analysis part which analyses the raw data to generate findings. The in-situ processing model co-locates the simulation and analysis parts on the same machine to eliminate data movement cost, while in the in-transit processing model, the simulation output data are staged to the analysis node directly through interconnect to avoid touching the storage system.

3.5 Storage

Data is centric to eScience applications. With the development of science, the hypothesis to data has evolved from empirical description stage, theoretical modelling stage, computational simulation stage to the fourth paradigm today, the data-intensive scientific discovery stage. Due to the vast data size, knowledge on the storage format of scientific data in the cloud is very important. Facing the massive data sets, there are two major ways for data storage: data can be stored as files in file systems or in databases.

Many distributed file systems have been proposed to provide efficient and reliable access to large-scale data using clusters of commodity hardware [36, 63]. For example, distributed and reliable file systems such as Hadoop Distributed File System (HDFS) are the primary storage system used by Hadoop applications which utilize the MapReduce model for large dataset processing. OpenStack Swift [60] is a distributed storage system for unstructured data at large scale. It currently serves the largest object storage clouds, such as Rackspace Cloud Files and IBM Sftlayer Cloud. To efficiently organize and store the massive scientific datasets, scientific data formats such as NetCDF [55] and HDF5 [68] have been widely used to achieve high I/O bandwidth from parallel file systems. The scalable and highly efficient distributed file system models together with the scientific data formats provide a promising data storage approach for data intensive eScience applications.

Databases for eScience have been emerging for a number of advantages in query processing capability, relatively mature techniques, and data integrity. HBase, a Hadoop project modelled on Bigtable, has been applied to many eScience applications such as bioinformatics domains [66]. Some array-based databases such as SciDB [19] have also been proposed to satisfy the special requirement of array-based eScience applications. SciDB is a scientific database system built from ground up and has been applied to many scientific application areas, including astronomy, earth remote sensing and environmental studies [5].

Although the data size of most eScience applications is enormous, we have observed that many of the eScience data are statically stored. For example, the SciHmm [58] project is making optimizations on time and money for the phylogenetic analysis problem. The data involved in this application are genetic data, which do not require frequent update and can be viewed as statically stored. Similarly, the bioinformatics data in the CloudBLAST [51] project and the astronomy data in the Montage Example [28], although may be updated from time to time, are seldom modified once obtained. Existing blob-based or distributed storages like Amazon S3 can be an ideal storage system for Science.

3.6 Security

Security is a big issue to eScience applications, especially for those with sensitive data. On the one hand, scientists need to make sure that the sensitive data is secured from malicious attacks. On the other hand, they also need to share data between scientific groups (possibly from different nations) working on the same project. Thus, how to find a balance point between the two aims is a challenging problem. Currently, the security level in the cloud is relatively immature compared to the Grid computing platform. One common way to make sure of security in the cloud is through logging in. Many eScience applications deployed on the cloud have designed their own way of authentication and authorization to further ensure security. Such as in [75], Group Authorization Manager is used to grant access permission based on user-defined access control policy. The emerging Open authorization (OAuth2.0) protocol is used to support authorization for users to share datasets or computing resources. In [74], the Gold security infrastructure is utilized to deal with the authentication and authorization of users to keep sensitive data secure. Data owners could specify their security preferences for the security infrastructure to control role and task based access.

Unlike in Grid computing, where the authentication and authorization mechanisms are mainly based on the public key infrastructure (PKI) protocol [13], many Cloud vendors support multiple security protocols such as OAuth2.0. eScience gateway is a commonly adopted approach to reinforce the security mechanisms.

3.7 Service Models

There are different levels of computing services offered by the cloud (i.e., IaaS, IaaS with tools, PaaS and SaaS). The IaaS model is the most basic cloud service model, where cloud providers only offer physical infrastructures such as virtual machines and raw storage to users. Amazon EC2 is such an example [32, 57, 54]. To enable the execution of scientific applications in IaaS clouds, a number of domain-specific supporting software and tools need to be installed. In order to save scientists' effort of installation, platforms providing IaaS level services but with additional tools and software, have been proposed. Nimbus [37] and Eucalyptus are examples of this kind. In the PaaS model, cloud providers provide a computing platform typically equipped with operating system, programming language execution environment and database. Users of PaaS cloud can simply develop their applications on the platform without the effort and cost of buying and managing the underlying hardware and software layers. Typical examples of this type include Windows Azure, Google's App Engine. In the SaaS model, cloud providers provide a computing platform installed with application software. Cloud providers are in charge of the software maintenance and support. One example is Altair SaaS [6] which provides high-performance computing (HPC) workload management and scheduling services for applications such as scientific simulations.

Due to the pay-as-you-go pricing feature of the cloud services, monetary cost is an important consideration of eScience in the cloud. MFA [24] reported a 14 times speedup for their metabolic flux analysis on Amazon cloud with a $11 cost, which includes the EC2 cost, EMR cost and S3 storage cost. SciHmm [58] aims to reduce monetary cost of scientists via deciding the most adequate scientific analysis method for the scientists a priori. It reported the cost for the parallel execution of SciHmm on the Amazon EC2 cloud and showed that it is acceptable for most scientists (US $47.79). Due to the large scale of data and long running jobs, eScience applications have to carefully manage the cloud resources used to optimize their monetary cost. However, this resource management problem is not trivial and requires both domain expertise and knowledge on cloud computing. A lot of on-going studies have concentrated on the monetary cost optimizations for scientific workflows [78, 77, 47, 41, 28]. In Section 4, we present our experiences on cloud resource provisioning problems to optimize the monetary cost and performance of eScience in the cloud.

3.8 Collaboration

Another important usage of cloud for eScience applications is to realize collaboration. The collaboration between the groups includes two different focuses: sharing storage and/or sharing computation. Sharing storage is the sharing mechanism of scientific data and analysis results between different research groups working on the same project. Sharing computation is to share the idle computing resources of one group to the others such that the resource utilization rate of all the collaborating groups can be highly improved. Collaboration between these groups is very important to the success of the projects. With the development of Internet and the popularity of social networks, some previous studies have leveraged cloud computing techniques and social network APIs to provide a collaboration platform for eScience researchers [67, 21].

The Life science project CloudDRN [50] moves medical research data to the cloud to enable secure collaboration and sharing of distributed data sets. It relies on authentication and authorization to ensure security. Also, many applications in Social Science and Humanities have shown increasing collaboration. The SoCC [67] project leverages social network platform for the sharing of resources in scientific communities. They provide a PaaS social cloud framework for users to share resources and support creating virtual organizations and virtual clusters for collaborating users. The SCC [21] project is also leveraging social network and cloud computing to enable data sharing between social network users.

4 Our Expeditions on Resource Provisioning for eScience in the Cloud

From our survey on existing scientific applications, we find that resource provisioning is an important problem for scientific applications in the cloud. In this section, we present our existing studies on this direction. We can easily classify the resource provisioning problems using our taxonomy. We study the problems in public IaaS clouds for scientific workflows. Workflows in physical sciences (e.g., Montage and Ligo) and biological sciences (e.g., Epigenomics) are studied. The Pegasus workflow

management system is used to run the scientific workflows in the cloud. Distributed file systems are used to store the large-scale scientific data and workflow tasks share cloud computation resources for monetary cost optimization.

4.1 Motivation

Scientists often use scientific workflows to analyse and understand scientific data. Scientific workflows involve complex computational processes, often requiring accessing a large amount of data. Montage workflow [52] is an example in astronomical study for generating sky mosaics in the scale of hundreds of GBs. CyberShake [40] is a data-intensive workflow for characterizing earthquake hazards. CyberShake workflows are composed of more than 800,000 tasks and have input data larger than 200TBs. Another example is the Epigenomics workflow [40], which is a biological application that studies the set of epigenetic modifications on human cells. All these example workflows involve managing and processing very large data sets, such as the sky image data and human genetic data.

Due to the pay-as-you-go characteristic of the cloud, many real-world scientific workflows are currently deployed and executed in IaaS clouds [14]. Although the scalability and elasticity of the cloud have brought great opportunities for the workflows, many research problems also arise. Resource provisioning is one important problem for the monetary cost and performance optimizations of scientific workflows in IaaS clouds. Since cloud providers usually offer multiple instance types with different prices and computational capabilities, we need to carefully decide the types of instances that each task of a workflow executes on to optimize the performance and monetary cost. However, making the resource provisioning decisions is non-trivial, involving the complexities from cloud, workflows, and users.

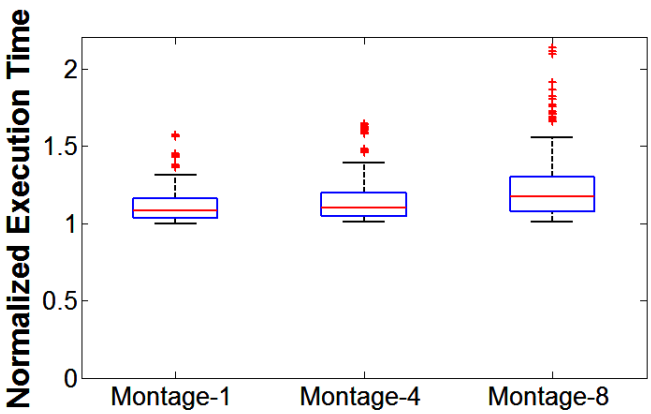

Figure 2 Execution time variances of running Montage workflows on Amazon EC2.

The resource provisioning for workflows in IaaS clouds is a complex problem, from the following three aspects.

Diverse cloud offerings. The IaaS clouds usually offer a large number of instance types. For example, Amazon EC2 provides more than 20 types of instances (only counting the latest generation) for the users [15]. Different types of instances usually have diversified capabilities and prices. For example, Amazon EC2 offers storage optimized instances to provide very high random I/O performance for I/O-intensive applications. If we consider multiple clouds, the situation is even worse since the cloud providers usually adopt different cloud offerings. For example, Amazon EC2 adopts hourly pricing scheme while Google Compute Engine charges users by minute.

The dynamics in cloud performance and prices make the problem even more complex. Most existing resource provisioning approaches for scientific workflows in IaaS clouds [47, 33, 30] assume that the execution time of each task in the workflow is static on a given VM type. However, this assumption does not hold in the cloud. The cloud environment is by design a shared infrastructure. The performance of cloud resources, such as I/O and network, is dynamic due to interferences between users [62]. We have observed remarkable dynamics in the I/O and network performances from Amazon EC2 [80]. Figure 2 shows the quantiles of the normalized execution time of the Montage

workflows in different scales running on Amazon EC2 for 100 times each. The execution time of the three workflows varies significantly. The variances are mainly from the interferences from disk and network I/O. In fact, scientific workflows may process input data of a large size. Due to the significant performance variance of scientific workflows in IaaS clouds, the deterministic notions of performance/cost constraints are not suitable, and a more rigorous notion is required.

On another hand, the cloud is an economic market and has dynamic prices [71]. Amazon EC2 offers spot instances, whose prices are determined by market demand and supply. Most existing optimization approaches for scientific workflows in IaaS clouds [47, 48] adopt static notions of performance and cost, which are not suitable for performance and cost optimizations in the dynamic cloud environment. Effective optimization techniques and more rigorous QoS notions are in need to capture the cloud dynamics.

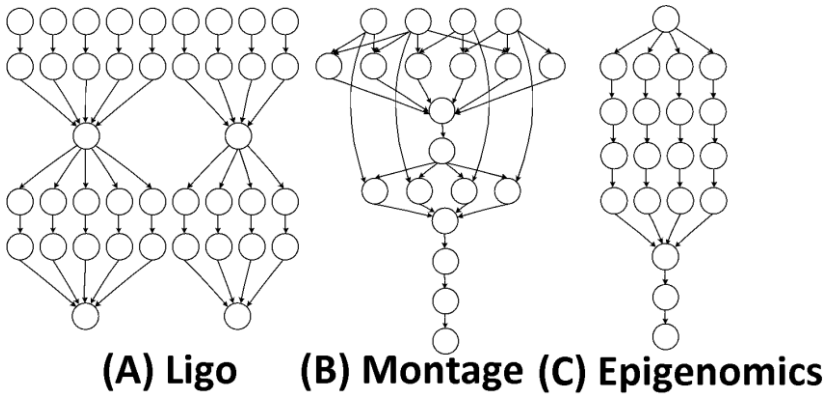

Figure 3 Workflow structure of Ligo, Montage and Epigenomics.

Complex workflow structures and characteristics. Workflows can have very different and complex structures. For example, Figure 3 shows the DAG structure of Ligo, Montage and Epigenomics workflows. We can easily observe from the figure that, the structure of Montage is the most complicated in the three workflows while Ligo and Epigenomics workflows have higher parallelism compared to Montage. Within a single workflow, the characteristics of tasks also vary. For example, in the Montage workflow, some tasks are computation-intensive (i.e., most of the task execution time is spent on CPU computations) and some are I/O-intensive (i.e., most of the task execution time is spent on I/O operations). There are also different application scenarios of workflows. For example, the workflows can be continuously submitted to the cloud and the optimizations are made for each workflow individually [48, 20]. Users can also group the workflows with similar structure but different input parameters as an ensemble, and submit QoS and optimization requirements for the entire ensemble. We need an effective system that is capable of simplifying the optimizations of different kinds of tasks and workflows. We should also consider how to make use of the different workflow structures for cost and performance optimizations.

Various user requirements. Scientists submit their workflow applications to the IaaS clouds usually with some predefined optimization objectives and QoS requirements. For example, one may desire to finish a workflow execution with a minimum monetary cost before a predefined deadline while another one may desire to execute a workflow as fast as possible with a given budget. Users may also define skyline optimization objectives, e.g., minimizing both of the monetary cost and the execution time of workflows. The users' requirements are also evolving. For example, a user may want to minimize the execution time of a workflow on a cloud $C1$ with a predefined budget. On the other scenario, she may consider running the workflow on multiple clouds besides $C1$. At this point, the optimal solution depends on the offerings of the multiple clouds and the network performance across clouds. Existing optimization algorithms are specifically designed for certain optimization problems and are usually not extensible or flexible to various evolving user requirements. Different resource provisioning schemes result in significant monetary cost and performance variations. Figure 4 shows the normalized average cost of running Montage workflow with deadline constraint using different instance configurations on Amazon EC2. We consider seven scenarios: the workflow is executed on a single instance type only (m1.small, m1.medium, m1.large and m1.xlarge), on randomly chosen

instance types, and using the instance configurations decided by Autoscaling [48] and by a optimization engine proposed in this chapter (denoted as Deco). Although the configurations m1.small and m1.medium obtain low average cost, they cannot satisfy the performance constraint of the workflow. Among the configurations satisfying the deadline constraint, Deco obtains the lowest monetary cost. The cost obtained by Deco is only 40% of the cost obtained by the most expensive configuration (i.e., m1.xlarge).

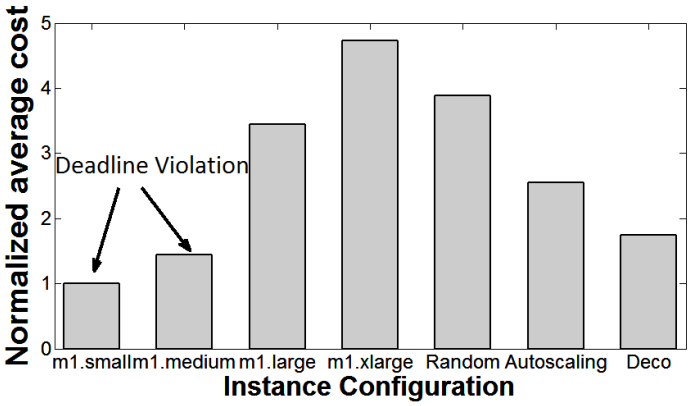

Figure 4 Average cost of running Montage workflows under different instance configurations on Amazon EC2.

4.2 Our Solution

To address the above challenges, we design a flexible and effective optimization system to simplify the optimizations of monetary cost and performance for scientific workflows in IaaS clouds. Figure 5 shows our overall design. Specifically, we propose a probabilistic scheduling system called *Dyna* [80] to minimize the cost of workflows while satisfying the probabilistic performance guarantees of individual workflows predefined by the user. We also abstract the common monetary cost and performance optimizations of workflows as transformation operations, and propose a transformation-based optimization framework named *ToF* [78] for the monetary cost and performance optimizations of workflows. Finally, we propose a declarative optimization engine named *Deco* [79], which can automatically generate resource provisioning plan for various workflow optimization problems, considering the cloud performance dynamics. We introduce the details of the three projects in the following subsections.

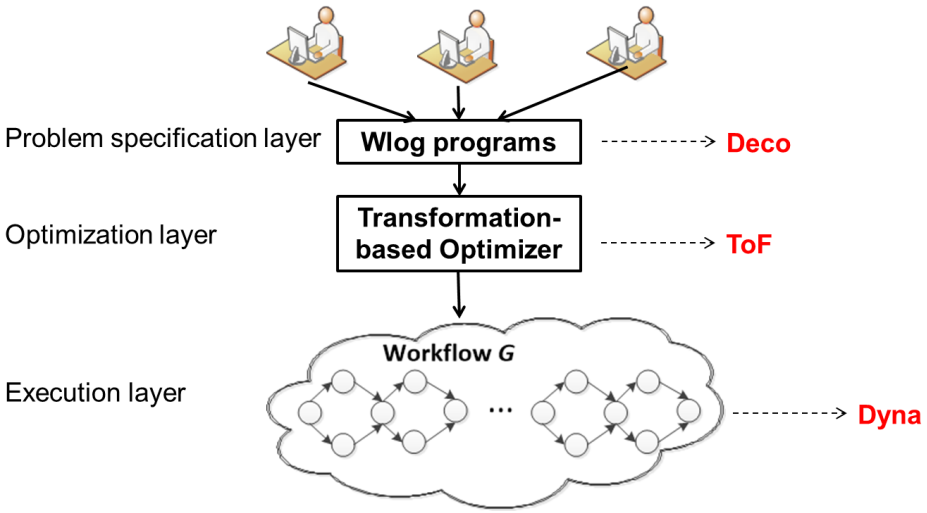

Figure 5 Overall Design.

4.2.1 Effective Monetary Cost Optimizations for Workflows in IaaS Clouds

Cloud dynamics, including the price and performance dynamics, can greatly affect the resource provisioning result of workflows in IaaS clouds. In this project, we consider a typical scenario of providing software-as-a-service for workflows in the IaaS clouds. We denote this model as workflow-

as-a-service (WaaS). We propose a dynamics-aware optimization framework called Dyna, to improve the effectiveness of monetary cost optimizations for WaaS providers. Compared with existing scheduling algorithms or systems [48], Dyna is specifically designed to capture the cloud performance and price dynamics. The main components of Dyna are illustrated in Figure 6.

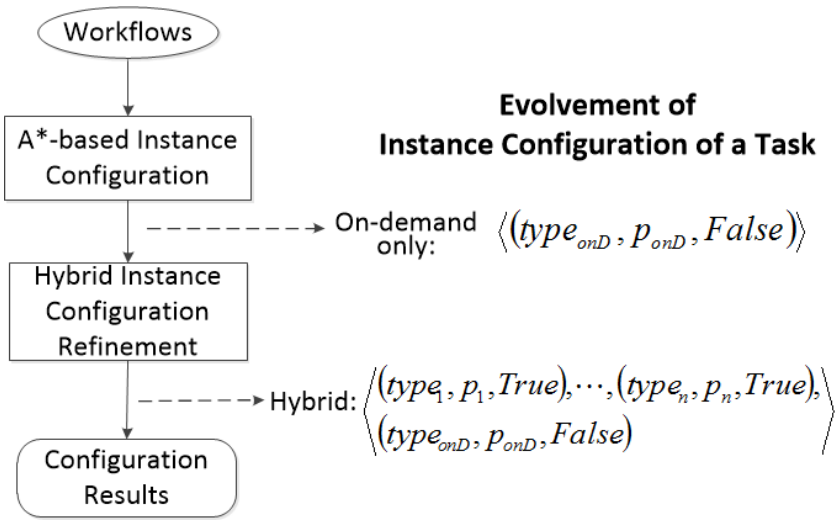

Figure 6 Overview of the Dyna system.

When a user has specified the probabilistic deadline requirement for a workflow, WaaS providers schedule the workflow by choosing the cost-effective instance types for each task in the workflow. The overall functionality of the Dyna optimizations is to determine the suitable instance configuration for each task of a workflow so that the monetary cost is minimized while the probabilistic performance requirement is satisfied. We formulate the optimization process as a search problem, and develop a two-step approach to find the solution efficiently. The instance configurations of the two steps are illustrated in Figure 6. We first adopt an *A**-based instance configuration approach to select the on-demand instance type for each task of the workflow, in order to minimize the monetary cost while satisfying the probabilistic deadline guarantee. Second, starting from the on-demand instance configuration, we adopt the hybrid instance configuration refinement to consider using hybrid of both on-demand and spot instances for executing tasks in order to further reduce cost. After the two optimization steps, the tasks of the workflow are scheduled to execute on the cloud according to their hybrid instance configuration. At runtime, we maintain a pool of spot instances and on-demand instances, organized in lists according to different instance types. Instance acquisition/release operations are performed in an auto-scaling manner. For the instances that do not have any task and are approaching multiples of full instance hours, we release them and remove them from the pool. We schedule tasks to instances in the earliest-deadline-first manner. When a task with the deadline residual of zero requests an instance and the task is not consolidated to an existing instance in the pool, we acquire a new instance from the cloud provider, and add it into the pool. In our experiment, for example, Amazon EC2 poses the capacity limitation of 200 instances. If this cap is met, we cannot acquire new instances until some instances are released.

The reason that we divide the search process into two steps is to reduce the solution space. For example, consider searching the instance configuration for a single task, where there are *n* on-demand types and *m* spot instance types. If we consider spot and on-demand instances together, the number of configurations to be searched is $\binom{n}{1} \times \binom{m}{1}$ while in our divide-and-conquer approach, the complexity is reduced to $\binom{n}{1} + \binom{m}{1}$. In each search step, we design efficient techniques to further improve the optimization effectiveness and efficiency. In the first step, we only consider on-demand instances and utilize the pruning capability of $A*$search to improve the optimization efficiency. In the second step, we consider the hybrid of spot and on-demand instances as the refinement of the instance configuration obtained from the first step. We give the following example to illustrate the feasibility of the two-step optimization.

EXAMPLE 1. Consider the instance configuration for a single task. In the A-based instance configuration step, the on-demand instance configuration found for the task is* $< (0, 0.1, False) >$*. In the refinement step, the on-demand instance configuration is refined to* $< (0,0.01, Ture), (0, 0.1, False) >$*. Assume the expected execution time of the task on type* 0 *instance is 1 hour and the spot price is lower than \$0.01 (equals to \$0.006) for 50% of the time. The expected monetary cost of executing the task under the on-demand instance configuration is \$0.1 and under the hybrid instance configuration is only \$0.053 (*$\$0.006 \times 50\% + \$0.1 \times 50\%$*). Thus, it is feasible to reduce the expected monetary cost by instance configuration refinement in the second step.*

Evaluation results. We compare Dyna with the state-of-the-art algorithm [48] (denoted as Static) on three different workflow applications shown in Figure 3 and find that Dyna saves monetary cost over Static by 15{73% when the probabilistic deadline requirement is 96%. Although the average execution time of Dyna is longer than Static, it can guarantee the probabilistic deadline requirements under all settings.

4.2.2 Transformation-based Optimizations for Workflows in IaaS Clouds

Due to the diversified cloud offerings and complex workflow structures and characteristics, resource provisioning for scientific workflows in IaaS clouds is a complicated optimization problem. To address the complexity issues, we propose a transformation-based optimization framework called ToF to simplify workflow optimizations effectively. In ToF, we abstract the common operations in the monetary cost and performance optimizations of scientific workflows as *transformations* and design a cost model to guide the selection of transformations effectively.

Table 2 Details of the six transformation operations. The formulation $V_i(t_0, t_1)$ *stands for an instance of type i and the task on this instance starts at* t_0 *while ends at* t_1*.*

Name	Category	Description	Formulation
Merge	Main	Merge multiple tasks to the same instance to fully utilize partial hours.	$\mathcal{M}(V_i(t_0, t_1), V_i(t_2, t_3)) \rightarrow V_i(t_0, t_3)$
Demote	Main	Assign a task to a cheaper instance type.	$\mathcal{D}(V_i(t_0, t_1)) \rightarrow V_j(t_2, t_3), where\ i > j$
Move	Auxiliary	Delay a task to execute later.	$\mathcal{V}(V_i(t_0, t_1)) \rightarrow V_i(t_2, t_3), where\ t_3 = t_2 + (t_1 - t_0)$
Promote	Auxiliary	Assign a task to a better instance type.	$\mathcal{P}(V_i(t_0, t_1)) \rightarrow V_j(t_2, t_3), where\ i < j$
Split	Auxiliary	Stop a running task at some checkpoint and restart it later.	$\mathcal{S}(V_i(t_0, t_1)) \rightarrow V_{i1}(t_0, t_2), V_{i2}(t_3, t_4)$
Co-scheduling	Auxiliary	Assign two or more tasks to the same instance for execution.	$\mathcal{C}(V_i(t_0, t_1), V_i(t_2, t_3)) \rightarrow V_i(\min(t_0, t_2), \max(t_1, t_3))$

We have developed six basic transformation operations, namely *Merge*, *Split*, *Promote*, *Demote*, *Move* and *Co-scheduling*. These basic transformations are simple and lightweight. Moreover, they can capture the current cloud features considered in this chapter. They are the most common operations and widely applicable to workflow structures. For example, the operations of all the comparison algorithms used in the experiments can be represented using those transformations. However, we do not claim they form a complete set. Users can extend more transformation operations into the transformation set. Adding a transformation operation requires the modifications including adding the cost model and transformation implementation on the instance DAG.

Based on their capabilities in reducing monetary cost, we categorize the transformation operations into two kinds, namely *main* schemes and *auxiliary* schemes. A main scheme can reduce the monetary cost while an auxiliary scheme simply transforms the workflows so that the transformed workflow is suitable for main schemes to reduce cost. By definition, Merge and Demote are main

schemes, and the other four operations are auxiliary schemes. Table 2 summarizes the definition and categorization for the six operations.

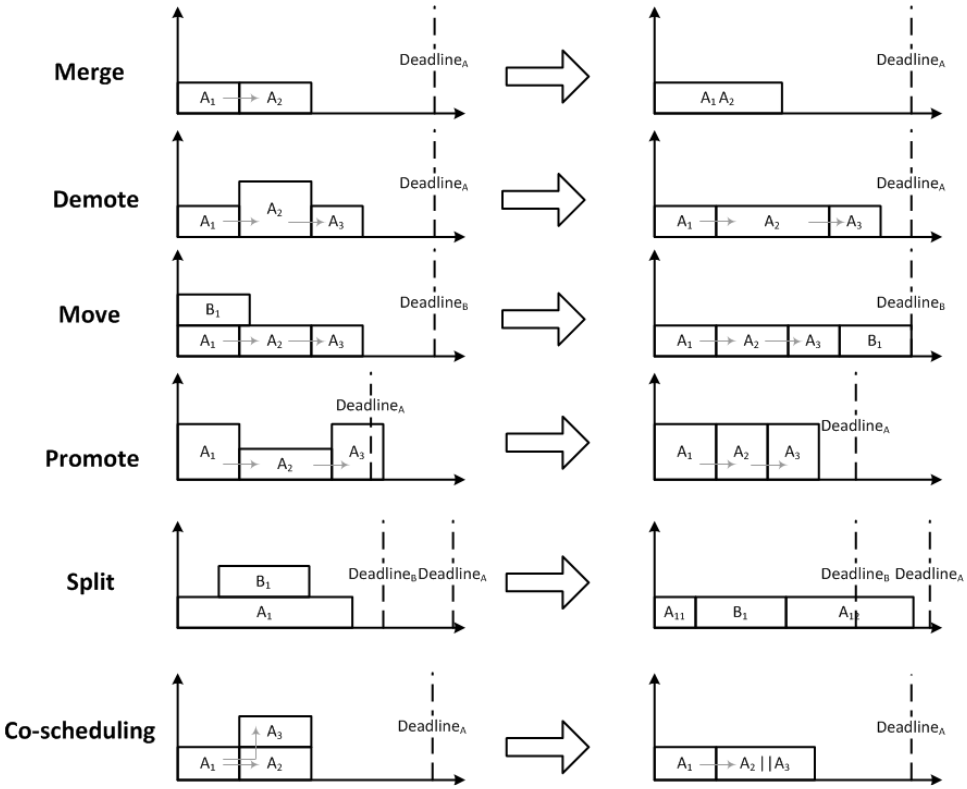

Figure 7 Use cases of the six transformation operations shown in the instance-time chart.

Some examples of transformation are illustrated in Figure 7. We illustrate the transformation operations with an instance-time chart, where the x axis represents time and y axis represents the instance. An instance-time chart is similar to Gantt chart, with the box width as the execution time and with dependencies between boxes. The height of the boxes stand for prices of instances. During the transformation, we maintain the structural dependency among tasks even after transformations.

We develop simple yet effective cost models to estimate the cost and the time changes for applying one transformation operation on the instance DAG. Since an auxiliary scheme does not directly reduce the cost, we estimate the potential cost saving of the main schemes after applying the auxiliary scheme. As for the time estimation, the changes of execution time need to be propagated to all the tasks with dependencies on the vertices affected by the transformation operation. This article estimates the worst case for the change of execution time, since worst-case analysis usually can have simplified estimation process. For details on cost model, readers can refer to the paper [80].

In one optimization iteration, we first estimate the (potential) cost reduction of each operation which satisfies the deadline constraint, using the cost models. Second, we select and perform the operation with the most cost reduction. All selected transformations form the optimization sequence.

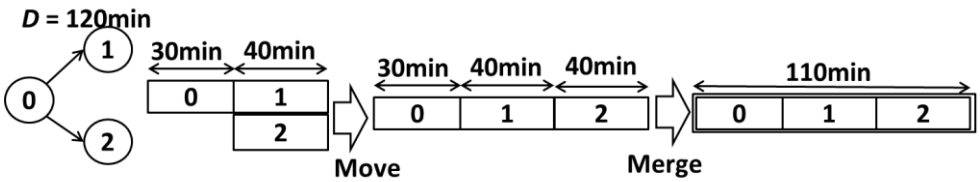

Figure 8 Example of applying transformation operations on a three node structured workflow.

Figure 8 shows an example of a simple structured workflow with three tasks. The deadline of the workflow is 120 minutes and the execution time of Tasks 0, 1 and 2 on the assigned instance types are

30, 30 and 40 minutes respectively. In the first iteration, we first check the operations in main schemes and find that no one can reduce cost. We then check the operations in auxiliary schemes and select the Move operation to perform as it can introduce the most cost reduction. In the next iteration, Merge from the main schemes is selected and performed, after which no operation can further reduce the cost of the workflow. After applying the Move and Merge operations, the charging hours of executing this workflow is reduced from three to two.

Evaluation results. We demonstrated the accuracy of our cost model estimation and compared ToF with the state-of-the-art algorithm [48] on the Montage and Ligo workflows. ToF outperforms the state-of-the-art algorithm by 30% for monetary cost optimization, and by 21% for the execution time optimization. Please refer to our previous work [78] for experimental details.

4.2.3 A Declarative Optimization Engine forWorkflows in IaaS Clouds

WMSes [29, 46, 65] are often used by scientists to execute and manage scientific workflows. Those workflow management systems often have dependent software tools such as Condor and DAGMan [12], and require specific skills to implement the specific optimization algorithms in the cloud. All those software packages are interplayed with the resource provisioning problem in the cloud. It is desirable to abstract these complexities from users and shorten the development cycle. In this chapter, we develop a declarative resource provisioning engine named Deco and integrate it into a popular WMS named Pegasus for executing scientific workflows in IaaS clouds. Figure 9 presents a system overview of Deco and its integration in the Pegasus WMS.

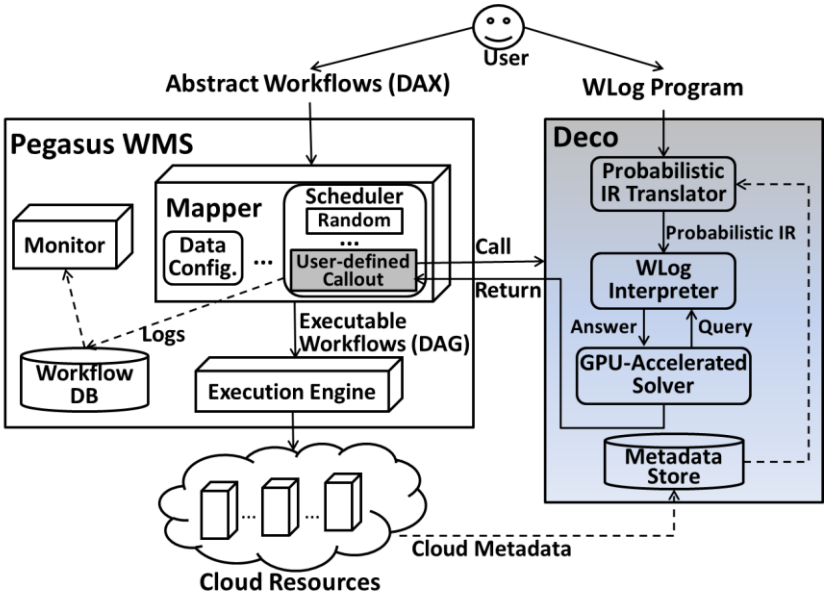

Figure 9 System Overview of Deco with integration in Pegasus.

Table 3 Workflow and cloud specific built-in functions and keywords in WLog.

Function/Keyword	Remark
goal	Optimization goal defined by the user.
cons	Problem constraint defined by the user.
var	Problem variable to be optimized.

In order to schedule the workflows in the cloud, users can alternatively choose from several traditional schedulers provided by Pegasus and our proposed Deco. For example, Pegasus provides a Random scheduler by default, which randomly selects the instance to execute for each task in the workflow. With Deco, we model the resource provisioning problem as a constrained optimization problem. Users can specify various optimization goals and constraints with WLog programs. WLog is a declarative language extended from ProLog, with special extensions for scientific workflows and the dynamic clouds. Table 3 gives several examples of such extensions and explains their functionality.

Deco allows users to use probabilistic notions to specify their optimization requirements in the dynamic clouds. We model the dynamic cloud performance with probabilistic distributions, which is transparent to users. Deco automatically translates a WLog program submitted by users to probabilistic intermediate representation (IR) and interpret it using the WLog interpreter. We traverse the solution space to find a good solution for the optimization problem. For each searched solution, we evaluate it with the probabilistic IR, which requires a lot of computation [25]. To effectively and efficiently search for a good solution in a reasonable time, we implement a GPU-accelerated parallel solver to leverage the massive parallelism of the GPU. After the optimization process, Deco returns the found resource provisioning plan (indicating the selected execution site for each task in the workflow) to Pegasus for generating the executable workflow.

Evaluation results. We use Deco to solve three different workflow optimization problems. Specifically, we formulate a workflow scheduling problem (single workflow and single cloud), a workflow ensemble optimization problem (multiple workflows and single cloud) and a workflow migration problem (multiple workflow and multiple clouds). These use cases have covered a large part of resource provisioning problems for scientific workflows. Our experimental results show that, Deco is able to obtain better optimization results than heuristic based methods in all use cases. Specifically, Deco can achieve more effective performance/cost optimizations than the state-of-the-art approaches, with the monetary cost reduction by 30-50%. The optimization overhead of Deco takes 4.3-63.17 ms per task for a workflow with 20-1000 tasks.

5 Open Problems

Previous sections have reviewed the status and the observations in building eScience applications and systems in the cloud. Despite the fruitful results along this research direction, we clearly see that there are still many open problems to be addressed in order to fully unleash the power of cloud computing for eScience. We present the open problems for developing the next-generation eScience applications and systems in the cloud. Those open problems are rooted at the interplay between eScience requirements and cloud computing features.

Data Lock-In: There is no standardization between different cloud platforms, such as different clouds use different data storage formats. For example, data stored in Amazon S3 cannot be easily used by the jobs running on the Windows Azure platform due to different APIs, data storage techniques such as encryption technique and security protocols. On the other hand, due to the fact that eScience projects usually involve a large amount of data for scientific research, such as the genome sequence data and seismographic data, data transfer cost between different cloud platforms is substantial. It requires further research on reducing the network data transfer in terms of both performance and monetary cost.

Performance Unpredictability: Some eScience applications have rather rigid performance requirements. Performance unpredictability is a critical problem for running those applications in the cloud, due to the interference among concurrent applications running in the same cloud. This problem is particularly severe for disk I/O and network traffic, especially for data-intensive eScience applications. The other factor of performance unpredictability is VM failures or unreliability. In [44], the authors issued a total of 10,032 VM unique instance start events on Windows Azure cloud and only 8,568 instances started once during their lifetimes while the others had encountered various unknown problems during their run and were restarted by the Azure infrastructure for many times.

Data Confidentiality and Auditability: Current commercial clouds are essentially open to public and are consequently exposing themselves to more attacks. For eScience applications, the data involved could be relevant to the homeland security of a country, such as the geographical data of the country, or even the security of human beings, such as the human genome data. So protecting these sensitive data from unauthorized or even malicious access is an important ongoing research topic.

Lacking of eScience Common System Infrastructure: As we discussed in the previous section, the efforts of implementing eScience projects on the cloud are quite ad-hoc. For example, the Montage workflow, an astronomy toolkit, is commonly used to discuss the pros and cons of using cloud computing for scientific applications [28, 37] and such physical science systems built in the cloud are specifically designed to better fit the cloud for scientific workflow applications. Thus, such developmental experiences may not be useful to scientific applications in other areas. In order to save the development cycle and better exploit the experiences of current systems, we need a holistic platform which enables various research fields can build their systems upon and offers opportunities for application specific optimizations.

Resource Management on Future Clouds: With the presence of new cloud service models and new technologies, the resource management problem is becoming more important and complex in the future clouds. The high-speed Internet connection makes hybrid cloud resources available and perform as if they are physically located close to the users. However, to enable such cost-efficient and low-latency services to users, we need to design a fine-grained and extensive resource management system providing different ways of measuring and allocating resources. As the cloud gets popular, cloud autonomics is on its way. As a result, automated resource management systems are also required to ease the users from tedious system configurations, monitoring and management.

6 Conclusion

Scientific computing is an emerging and promising application in the big data era. Recently, we witness many scientific applications have been developed and executed on the cloud infrastructure, due to its elasticity and scalability. In this chapter, we develop a taxonomy and conduct a review on the current status of eScience applications in the cloud. Due to the pay-as-you-go pricing feature of the cloud, we find that resource provisioning is an important problem for scientific applications in the cloud. We present our experiences on improving the effectiveness of monetary cost and performance optimizations for scientific workflows in IaaS clouds. Finally, we propose the open problems in this area and call for more support from the cloud community and more investment and efforts from other communities.

References

[1] US Naval Research Laboratory, Monterey, Ca. http://www.nrlmry.navy.mil/sec7532.htm.
[2] EGI Federated Cloud Task Force. https://www.egi.eu/infrastructure/cloud/.
[3] Catania Science Gateway. http://www.catania-science-gateways.it/.
[4] Docker - An open platform for distributed applications for developers and sysadmins. https://www.docker.com/.
[5] SciDB Use Case. http://scidb.org/use/.
[6] Altair SaaS. http://www.altairfms.com/altairfms-new/products/altaircloud.aspx.
[7] The kasumigaseki cloud concept. http://www.cloudbook.net/japancloudgov.
[8] The UK G-cloud, 2009. http://johnsuffolk.typepad.com/john-suffolk-government-cio/2009/06/government-cloud.html.
[9] The US cloud storefront, 2009. http://www.gsa.gov/portal/content/103758.
[10] Top 500 supercomputer, 2010. http://www.top500.org/lists/2010/11/.
[11] University of California, 2012. http://setiathome.berkeley.edu/.
[12] DAGman: A directed acyclic graph manager. Condor team, July 2005. http://www.cs.wisc.edu/condor/dagman/.
[13] R. Alfieri, R. Cecchini, V. Ciaschini, and F. Spataro. From gridmap-file to voms: managing authorization in a grid environment. *Future Generation Computer Systems*, 21:549-558, 2005.
[14] Amazon Case Studies. *http://aws.amazon.com/ solutions/case-studies/*. accessed on July 2014.
[15] Amazon EC2 Instance Types. *http://aws.amazon.com/ ec2/instance-types/*. accessed on July 2014.
[16] Gabriel Antoniu, Alexandru Costan, Benoit Da Mota, Bertrand Thirion, and Radu Tudoran. A-Brain: Using the cloud to understand the impact of genetic variability on the brain. *ERCIM News*, 2012(89), 2012.
[17] Janine C. Bennett, Hasan Abbasi, Peer-Timo Bremer, Ray Grout, Attila Gyulassy, Tong Jin, Scott Klasky, Hemanth Kolla, Manish Parashar, Valerio Pascucci, Philippe Pebay, David Thompson, Hongfeng Yu, Fan Zhang, and Jacqueline Chen. Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 49:1-49:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.
[18] S. Bohle. What is e-science and how should it be managed? http://www.scilogs.com/scientific and medical libraries/what-is-e-science-and-how-should-it-be-managed/.
[19] Paul G. Brown. Overview of SciDB: large scale array storage, processing and analysis. In *SIGMOD '10*, pages 963-968, 2010.
[20] R.N. Calheiros and R. Buyya. Meeting Deadlines of Scientific Workflows in Public Clouds with Tasks Replication. *IEEE Transactions on Parallel and Distributed Systems*, 25(7):1787-1796, July 2014.
[21] Kyle Chard, Kris Bubendorfer, Simon Caton, and Omer Rana. Social cloud computing: A vision for socially motivated resource sharing. *IEEE Trans. Serv. Comput.*, 5(4):551-563, January 2012.
[22] P.C. Church and A.M. Goscinski. A survey of cloud-based service computing solutions for mammalian genomics. *IEEE Trans. Serv. Comput.*, 7(4):726-740, Oct 2014.
[23] Roger Curry, Cameron Kiddle, Nayden Markatchev, Rob Simmonds, Tingxi Tan, Martin Arlitt, and Bruce Walker. Facebook meets the virtualized enterprise. In *EDOC '08*, pages 286{292, 2008.
[24] Tolga Dalman, Tim Doernemann, Ernst Juhnke, MichaelWeitzel, Matthew Smith, Wolfgang Wiechert, Katharina Noh, and Bernd Freisleben. Metabolic flux analysis in the cloud. In *ESCIENCE '10*, pages 57-64, 2010.
[25] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. Problog: A probabilistic prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, IJCAI'07, pages 2468-2473, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.
[26] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107-113, 2008.
[27] Ewa Deelman, Dennis Gannon, Matthew Shields, and Ian Taylor. Workflows and e-science: An overview of workflow system features and capabilities, 2008.
[28] Ewa Deelman, Gurmeet Singh, Miron Livny, Bruce Berriman, and John Good. The cost of doing science on the cloud: the montage example. In *SC '08*, pages 50:1-50:12, 2008.

[29] Ewa Deelman, Gurmeet Singh, Mei-Hui Su, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, G. Bruce Berriman, John Good, Anastasia Laity, Joseph C. Jacob, and Daniel S. Katz. Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems. *Sci. Program.*, 13(3):219{237, July 2005.
[30] Kefeng Deng, Junqiang Song, Kaijun Ren, and Alexandru Iosup. Exploring portfolio scheduling for long-term execution of scientific workloads in IaaS clouds. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 55:1-55:12, New York, NY, USA, 2013. ACM.
[31] Jaliya Ekanayake, Shrideep Pallickara, and Geoffrey Fox. MapReduce for data intensive scientific analyses. In *ESCIENCE '08*, pages 277-284, 2008.
[32] Constantinos Evangelinos and Chris N. Hill. Cloud computing for parallel scientific HPC applications: Feasibility of running coupled Atmosphere-Ocean climate models on amazon's EC2. In *Cloud Computing and Its Applications*, 2008.
[33] Hamid Mohammadi Fard, Radu Prodan, and Thomas Fahringer. A truthful dynamic workflow scheduling mechanism for commercial multicloud environments. *IEEE Trans. Parallel Distrib. Syst.*, 24(6):1203-1212, 2013.
[34] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *GCE08*, 2008.
[35] GAIA-Space. http://www.esa.int/Our Activities/Space Science/Gaiaoverview.
[36] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *SOSP '03*, pages 29-43, 2003.
[37] Christina Hoffa, Gaurang Mehta, Tim Freeman, Ewa Deelman, Kate Keahey, Bruce Berriman, and John Good. On the use of cloud computing for scientific workflows. In *ESCIENCE '08*, pages 640-645, 2008.
[38] Dexter H. Hu, Yinfeng Wang, and Cho-Li Wang. Betterlife 2.0: Large-scale social intelligence reasoning on cloud. In *CLOUDCOM '10*, pages 529-536, 2010.
[39] Marty Humphrey, Zach Hill, Catharine van Ingen, Keith R. Jackson, and Youngryel Ryu. Assessing the value of cloudbursting: A case study of satellite image processing on windows azure. In e*Science'11*, pages 126-133, 2011.
[40] Gideon Juve, Ann Chervenak, Ewa Deelman, Shishir Bharathi, Gaurang Mehta, and Karan Vahi. Characterizing and profiling scientific workflows. *Future Gener. Comput. Syst.*, 29(3):682-692, March 2013.
[41] Derrick Kondo, Bahman Javadi, Paul Malecot, Franck Cappello, and David P. Anderson. Cost-benefit analysis of cloud computing versus desktop grids. In *IPDPS '09*, pages 1-12, 2009.
[42] Craig A. Lee. A perspective on scientific cloud computing. In *HPDC '10*, pages 451-459, 2010.
[43] Jie Li, Marty Humphrey, Deborah A. Agarwal, Keith R. Jackson, Catharine van Ingen, and Youngryel Ryu. escience in the cloud: A MODIS satellite data re-projection and reduction pipeline in the windows azure platform. In *IPDPS'10*, pages 1-10, 2010.
[44] Jie Li, Marty Humphrey, You-Wei Cheah, Youngryel Ryu, Deborah A. Agarwal, Keith R. Jackson, and Catharine van Ingen. Fault tolerance and scaling in e-science cloud applications: Observations from the continuing development of modisazure. In *eScience'10*, pages 246-253, 2010.
[45] Michael Litzkow, Miron Livny, and Matthew Mutka. Condor - a hunter of idle workstations. In *ICDCS*, June 1988.
[46] Bertram Ludascher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific Workflow Management and the Kepler System: Research Articles. *Concurr. Comput. : Pract. Exper.*, 18(10):1039-1065, August 2006.
[47] Maciej Malawski, Gideon Juve, Ewa Deelman, and Jarek Nabrzyski. Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. In *SC '12*, pages 22:1-22:11, 2012.
[48] Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 49:1{49:12, New York, NY, USA, 2011. ACM.
[49] Nayden Markatchev, Roger Curry, Cameron Kiddle, Andrey Mirtchovski, Rob Simmonds, and Tingxi Tan. A cloud-based interactive application service. In *E-SCIENCE '09*, pages 102-109, 2009.
[50] Humphrey Marty, Steele Jacob, In Kee Kim, G. Kahn Michael, Bondy Jessica, and Ames Michael. Clouddrn: A lightweight, end-to-end system for sharing distributed research data in the cloud. In *ESCIENCE '13*, 2013.

[51] Andrfiea Matsunaga, Maurfificio Tsugawa, and Josfie Fortes. Cloudblast: Combining MapReduce and virtualization on distributed resources for bioinformatics applications. In *ESCIENCE '08*, pages 222-229, 2008.
[52] Montage Workflow. *http://montage.ipac.caltech.edu/docs/download2.html*. accessed on July 2014.
[53] J. Craig Mudge, Pinaki Chandrasekhar, Graham S. Heinson, and Stephan Thiel. Evolving inversion methods in geophysics with cloud computing – a case study of an escience collaboration. In *eScience*, pages 119-125, 2011.
[54] Ashish Nagavaram, Gagan Agrawal, Michael A. Freitas, Kelly H. Telu, Gaurang Mehta, Rajiv G. Mayani, and Ewa Deelman. A cloud-based dynamic workflow for mass spectrometry data analysis. *eScience*, pages 47-54, 2011.
[55] NetCDF. *http://www.unidata.ucar.edu/software/netcdf*.
[56] Andrew Newman, Yuan-Fang Li, and Jane Hunter. Scalable semantics - the silver lining of cloud computing. In *ESCIENCE '08*, pages 111-118, 2008.
[57] Santiago Nunez, Blair Bethwaite, Jose Brenes, Gustavo Barrantes, Jose Castro, Eduardo Malavassi, and David Abramson. Ng-tephra: A massively parallel, nimrod/g-enabled volcanic simulation in the grid and the cloud. In *ESCIENCE '10*, pages 129-136, 2010.
[58] Kary A.C.S. Ocana, Daniel de Oliveira, Jonas Dias, Eduardo Ogasawara, and Marta Mattoso. Optimizing phylogenetic analysis using scihmm cloud-based scientific workflow. *eScience* 62-69, 2011.
[59] OpenNebula. http://opennebula.org/users:users.
[60] OpenStack Swift. https://swiftstack.com/openstack-swift/architecture/.
[61] Mell Peter and Grance Timothy. The nist definition of cloud computing. *National Institute of Standards and Technology*, October 7 2009.
[62] J org Schad, Jens Dittrich, and Jorge-Arnulfo Quianfie-Ruiz. Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance. *Proc. VLDB Endow.*, 3(1-2):460{471, September 2010.
[63] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *MSST '10*, pages 1-10, 2010.
[64] Vedaprakash Subramanian, LiqiangWang, En-Jui Lee, and Po Chen. Rapid processing of synthetic seismograms using windows azure cloud. In *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*, CLOUDCOM '10, pages 193-200, 2010.
[65] Wei Tang, Jared Wilkening, Narayan Desai, Wolfgang Gerlach, Andreas Wilke, and Folker Meyer. A Scalable Data Analysis Platform for Metagenomics. In *The Proceedings of the 2013 IEEE International Conference on Big Data*, BigData, 2013.
[66] Ronald Taylor. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC bioinformatics*, 11 Suppl 12, 2010.
[67] Ashfag M. Thaufeeg, Kris Bubendorfer, and Kyle Chard. Collaborative research in a social cloud. In *ESCIENCE '11*, pages 224-231, 2011.
[68] The HDF5 Format. *http://www.hdfgroup.org/HDF5*.
[69] The Large Synoptic Survey Telescope (LSST). *http://www.lsst.org/* .
[70] Jens-S onke V ockler, Gideon Juve, Ewa Deelman, Mats Rynge, and Bruce Berriman. Experiences using cloud computing for a scientific workflow application. In *ScienceCloud '11*, 15-24, 2011.
[71] Hongyi Wang, Qingfeng Jing, Rishan Chen, Bingsheng He, Zhengping Qian, and Lidong Zhou. Distributed systems meet economics: Pricing in the cloud. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'10, Berkeley, CA, USA, 2010.
[72] Jianwu Wang and Ilkay Altintas. Early cloud experiences with the Kepler scientific workflow system. *Procedia Computer Science*, 9(0):1630-1634, 2012.
[73] Yi Wang, Gagan Agrawal, Tekin Bicer, and Wei Jiang. Smart: A MapReduce-like framework for in-situ scientific analytics. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, pages 51:1{51:12, New York, NY, USA, 2015.
[74] Paul Watson, Phillip Lord, Frank Gibson, Panayiotis Periorellis, and Georgios Pitsilis. Cloud computing for e-science with carmen. In *2nd Iberian Grid Infrastructure Conference*, pages 3-14, 2008.
[75] Wenjun Wu, Hui Zhang, ZhenAn Li, and Yaokuan Mao. Creating a cloud-based life science gateway. *eScience*, 55-61, 2011.
[76] Y. Zhao, Y. Li, I. Raicu, S. Lu, C. Lin, Y. Zhang, W. Tian, and R. Xue. A service framework for scientific workflow management in the cloud. *IEEE Trans. Serv. Comput.*, PP(99):1-1, 2014.
[77] Amelie Chi Zhou and Bingsheng He. Simplified resource provisioning for workflows in IaaS clouds. In *IEEE CloudCom*, pages 650-655, 2014.

[78] Amelie Chi Zhou and Bingsheng He. Transformation-based monetary cost optimizations for workflows in the cloud. *IEEE Transactions on Cloud Computing*, 2(1):85-98, 2014.
[79] Amelie Chi Zhou, Bingsheng He, Xuntao Cheng, and Chiew Tong Lau. A declarative optimization engine for resource provisioning of scientific workflows in IaaS clouds. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '15, pages 223-234, New York, NY, USA, 2015. ACM.
[80] Amelie Chi Zhou, Bingsheng He, and Cheng Liu. Monetary Cost Optimizations for Hosting Workflow-as-a-Service in IaaS Clouds. *IEEE Transactions on Cloud Computing*, 1, 2015.
[81] R. W. Zurek and L. J. Martin. GridPP: Development of the UK computing grid for particle physics. *Journal of Physics G: Nuclear and Particle Physics*, 32:1-20, 2006.