Deploying Bioinformatics Workflows on Clouds with Galaxy and Globus Provision

Bo Liu
Computation Institute, University of Chicago
Chicago, IL, USA
Argonne National Laboratory
Argonne, IL, USA
boliu@uchicago.edu

Borja Sotomayor
Department of Computer Science,
University of Chicago
Chicago, IL, USA
borja@cs.uchicago.edu

Ravi Madduri
Computation Institute, University of Chicago
Chicago, IL, USA
Argonne National Laboratory
Argonne, IL, USA
madduri@mcs.anl.gov

Kyle Chard
Computation Institute, University of Chicago
Chicago, IL, USA
Argonne National Laboratory
Argonne, IL, USA
kyle@ci.uchicago.edu

Ian Foster
Computation Institute, University of Chicago
Chicago, IL, USA
Argonne National Laboratory
Argonne, IL, USA
foster@mcs.anl.gov

***Abstract*—Cloud computing is attracting increasing attention as a means of providing users with fast provisioning of computational and storage resources, elastic scaling, and pay-as-you-go pricing. The integration of scientific workflows and Cloud computing has the potential to significantly improve resource utilization, processing speed, and user experience. This paper proposes a novel approach for deploying bioinformatics workflows in Cloud environments using Galaxy, a platform for scientific workflows, and Globus Provision, a tool for deploying distributed computing clusters on Amazon EC2. Collectively this combination of tools provides an easy to use, high performance and scalable workflow environment that addresses the needs of data-intensive applications through dynamic cluster configuration, automatic user-defined node provisioning, high speed data transfer, and automated deployment and configuration of domain-specific software. To demonstrate how this approach can be used in practice we present a domain-specific workflow use case and performance evaluation.**

Keywords-Scientific workflow; Cloud computing; Galaxy; Globus provision

I. INTRODUCTION

Scientific workflows represent an important paradigm for facilitating computational research. In a scientific workflow, developers create applications by composing multiple executable tasks (isolated "units" of computation or data manipulation) that are executed in a specified order. Traditionally, scientific workflows are executed locally, or on High Performance Computing (HPC) clusters, Grids [1, 2] and service-oriented architectures [3]. However, the task of designing, creating and deploying the underlying workflow infrastructure to support large-scale computation is non-trivial. Moreover, administrators face difficulties configuring and installing tool dependencies for the given domain, reserving appropriate resources to cater for fluctuating resource requirements, and designing scalable solutions that can handle the increasing size of scientific data.

Cloud computing offers many features that are attractive to scientific workflow developers [4-7]. For example, Infrastructure-as-a-Server (IaaS) Cloud platforms, such as Amazon EC2 [8], can provision the necessary resources to support a workflow system quickly (within a few minutes), provide as much computation and storage capacity as needed in a pay-as-you-go manner, be configured with appropriate software and applications, and elastically scale capacity as service demands change. However there remains a significant usability gap between the low-level interfaces provided by Cloud resources and the various mechanisms required by developers and users of elastic scientific workflows. For example, scientific workflows require the deployment and configuration of multiple components simultaneously (e.g., distributed file systems, virtual clusters, and a workflow portal); workflows are typically composed of many diverse executables, each with dependencies for specific packages or software that can be time consuming to install and configure; the resources used by scientific workflows may vary during run time, thus demanding an elastic reconfiguration method to alter computational clusters dynamically; and data sizes in many scientific domains require high-performance and reliable data transfer to move data between storage repositories and processing resources.

We present here an approach to realizing a high performance and scalable scientific workflow environment using Cloud resources. The approach is based on Galaxy [9], a web-based platform for scientific workflow implementation, execution and sharing. To meet the need for large-scale data transfer we have extended Galaxy to use Globus Online [10] -- a high performance, secure, and reliable data transfer service. This service addresses the challenges in moving or synchronizing large quantities of data across administrative domains. While Galaxy already provides tools for uploading and downloading files, the speed and reliability of these tools is not sufficient when transferring large datasets, as are commonly seen in biomedical research. As we demonstrate below, Globus tools can achieve performance improvements up to an order of magnitude.

Our approach builds on a tool called Globus Provision (GP) to automatically deploy and configure user-defined workflow environments. This technology supports automated deployment of all prerequisite tools and software packages required for Galaxy along with additional domain specific tools, thereby increasing GP's value in a high performance or domain-specific Cloud deployment. By exploiting the GP model, the deployed workflow environment can be modified to respond to workload changes by elastically adding or removing nodes from the cluster and changing instance "sizes" to balance cost and performance. In addition, software dependencies can be reconfigured at run-time to rapidly add or remove software from cluster nodes.

To demonstrate the flexibility of our approach we have extended this framework to meet the requirements of a specific domain, in this case cardiovascular research, by adding a set of domain-specific R based tools [11] to the deployment. Based on this toolset, we present a workflow use case in cardiovascular research, and evaluate its performance with respect to time and cost.

The remainder of this paper is structured as follows, Section 2 briefly introduces Galaxy; Section 3 provides an overview of our approach to deploy and configure Galaxy instances and elastically scale resources; Section 4 presents integration of Globus Transfer and CRData as tools in Galaxy. Section 5 presents a bioinformatics workflow use case and performance evaluation. Finally we review related work and present conclusions in Sections 6 and 7.

II. GALAXY

Galaxy [9] is a scientific workflow management system developed by Pennsylvania State University and Emory University. Galaxy provides an open, web-based platform that is widely used by biomedical scientists for data-intensive computational analyses and data integration. The main features of Galaxy are as follows.

1) Web-based platform for computational analyses

Galaxy provides a simple Web interface to a set of biomedical tools, enabling researchers to conduct their own custom analysis and manipulation without software installation or programming. Users can import datasets into their workspaces from established data warehouses and/or upload their own datasets. Interfaces to computational tools are automatically generated from abstract descriptions to ensure a consistent look and feel [9]. With Galaxy's workflow editor, various tools can be configured and composed to complete an analysis. Galaxy automatically records history and provenance information for each tool executed via a workflow.

2) Workflow publishing and data sharing

Galaxy supports reproducibility by capturing sufficient information about every step in a computational analysis, so that the analysis can be repeated in the future [9]. It tracks, in particular, all input, intermediate, and final datasets, as well as the parameters and the execution order of each step of the analysis [12]. Galaxy's sharing model, public repositories, and display framework provide users with the means to share datasets, histories, and workflows via web links, either publicly or privately. Galaxy users can annotate a history or workflow in the analysis workspace and then share them in a *Page* conveniently. A Galaxy *Page* is a mix of text, graphs and embedded Galaxy items from analyses (including datasets, histories and workflows), that allows a reader to easily view, reproduce, or extend the analyses [9]. Galaxy supports the whole lifecycle of research data, from creation, annotation, to publication and reuse.

3) Extensibility

Users can deploy their own Galaxy servers and customize it to meet particular requirements. Galaxy's flexible model makes the extension and integration of tools and data trivial. A tool can be any piece of software for which a command line invocation can be constructed. To add a new tool to Galaxy, a developer writes a configuration file that describes how to run the tool, including detailed specification of input and output parameters. This specification allows the Galaxy framework to work with the tool abstractly, for example, by automatically generating web interfaces [9].

III. GLOBUS PROVISION FOR GALAXY

Setting up a production instance of Galaxy is a non-trivial task that involves a number of manual installation and configuration steps for both the platform and any dependent software packages—steps that can be both error-prone and time consuming. These steps require that end-user researchers either become IT experts or rely upon the potentially sparse IT resources provided by their institutions. Either approach tends to result in sub-optimal use of researcher time and expertise.

Moreover, resource demands frequently fluctuate between and during execution of scientific workflows. It is often inefficient, in terms of resource usage and cost, to pre-provision infrastructure for peak usage and it is difficult to elastically scale (in near real-time) to the demands of a workflow.

To address these problems, we have designed a Globus Provision-based approach to automate the process of deploying and scaling Galaxy on Amazon EC2. This section first introduces Globus Provision, and then presents the

methods by which it is used to deploy and elastically scale Galaxy deployments on Amazon EC2.

A. *Introduction of Globus Provision*

Globus Provision [13] is a tool for automatically deploying a highly configurable and scalable distributed computing system that includes remote data access, job submission, and security. The system can be deployed with any subset of the tools it supports such as GridFTP [14] for high performance transfer, MyProxy [15] for user-based access management, and Condor [16] for job submission. As part of this configuration, GP also generates user accounts and certificates to support secure access, sets up a Network File System (NFS) and Network Information System (NIS) to provide a robust shared file system across nodes, and dynamically adds and removes software, hosts and user accounts.

Globus Provision relies on Chef [17] to configure hosts for a given *topology*. The topology is the specification of what will be deployed (e.g. a GridFTP server, a specific set of users, and a Condor cluster.). In Chef, the actions required to set up a specific piece of software are defined in a Ruby script called a *recipe*. Similar recipes are grouped into a *cookbook* which includes associated configuration templates and default values. GP defines several Chef Cookbooks to handle basic host setup and configuration of each node.

Figure 1 describes the main steps for using Globus Provision.

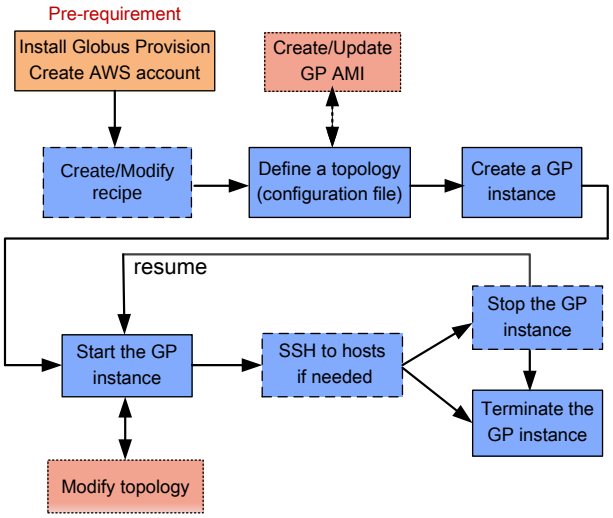


Figure 1. The main steps for using Globus Provision. (The blocks with solid lines are necessary steps, while the ones with dashed lines are optional steps (executed if needed))

1. Pre-requirement: Before starting with GP, an Amazon AWS account is needed. More specifically, the user should have a secret access key and SSH (Secure Shell) keypair for his EC2 account.
2. Create/Modify recipe: Users can modify existing recipes or add new recipes to install and configure specific software and packages, run commands and conduct other operations that should be performed on each host.
3. Define a topology: The topology file defines the user's requirements of the system. For example, whether it should have a shared file system, what users should be created, what software should be installed on each machine, etc.
4. Create/Start a GP instance: Based on the topology file, Globus Provision will create and start one or more instances on EC2 which together we refer to as a GP instance.
5. SSH to hosts if needed: When the GP instance is running, users can connect to any of its hosts via SSH.
6. Stop/Terminate the GP instance: The GP instance can be stopped while not in use (to avoid paying for idle resources), and resumed at a later time. Terminated instances cannot be resumed. All the hosts are shut down and all their resources are released after termination.
7. Modify topology: Once an instance is running, it is possible to modify its topology, e.g., by adding and removing hosts/users/domains, and adding software requirements to hosts. Users can edit the instance's topology and tell GP to modify the running instance to match the updated topology.
8. Create/Update GP AMI: Although GP already provides a public Amazon Machine Image (AMI), users can also create their own AMI (e.g., to use an AMI that is preloaded with required software packages such as specific bioinformatics tools) to speed up deployment.

B. *Globus Provision for Galaxy*

Globus Provision provides a generic architecture for automatically configuring distributed Cloud-based infrastructure and includes many valuable features for a Galaxy deployment (e.g., Condor clusters for distributed execution). For this reason we have chosen to extend GP to support configuration and deployment of a Galaxy instance with integrated Globus Transfer capabilities and user-defined domain-specific tools. In doing so, we have created an extensible framework that supports the deployment of custom Galaxy tools such as the CRData tools which are commonly used in cardiovascular research (Section 4.2). The combination of default tools and Globus Transfer tools included in the default Galaxy package simplifies the ability for users in different domains to create a user-specific Galaxy instance suitable for supporting data-intensive applications.

Figure 2 shows the architecture of our deployed system. There are three important nodes: 1) The Galaxy node provides Galaxy applications and user interface; 2) the GridFTP node operates as a Globus endpoint and allows Globus Transfer between the Galaxy system and other Globus endpoints; 3) the NFS node supplies a shared file system for all the other nodes in the system. In addition, when a Condor scheduler is configured, the Galaxy node also operates as a Condor head node that manages a set of Condor worker nodes in a dynamic Condor pool. In this model Galaxy jobs are transparently assigned to Condor worker nodes for parallel execution.

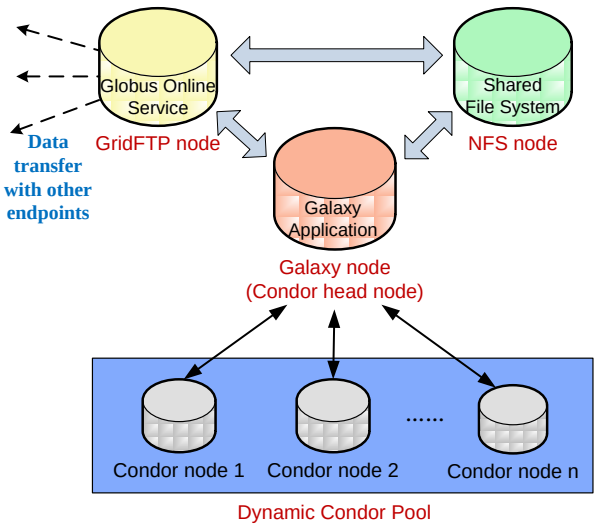


Figure 2. System architecture

The basis of the Galaxy GP model is a collection of new recipes for deploying Galaxy, its components, and the Globus Transfer additions.

One recipe ("galaxy-globus-common.rb") is responsible for installing the common requirements for Galaxy. (More specifically, for a forked version of Galaxy that includes the integrated Globus Transfer capabilities.) The recipe creates a galaxy user, downloads Globus Transfer tools as well as Galaxy from bitbucket.org, and copies default configuration files and set-up scripts for Galaxy. If Galaxy is installed on a domain with NFS/NIS, this recipe is run on the NFS/NIS server ("*simple-server*"), if not it is also run on the Galaxy server. The other core recipe ("galaxy-globus.rb"), which is run on the Galaxy server ("*simple-galaxy-condor")*, installs the Globus fork of Galaxy and Globus Transfer API, sets up the Galaxy database, executes set-up scripts and restarts Galaxy. Both of these recipes have been added to the "run_list" of the appropriate hosts when deploying Galaxy using GP.

The recipes created are open source and users can modify and create their own recipes that will deploy customized Galaxy instances.

C. *Dynamic Topology Reconfiguration*

One unique aspect of Globus Provision is its ability to dynamically alter, during runtime, the Cloud infrastructure. The Globus Provision Galaxy extensions take advantage of this functionality to balance performance and cost such that users pay only for the resources they use, while also being able to scale up to meet resource requirements. By altering the topology file and re-executing GP, users are able to add and remove instances from the Galaxy Condor pool within minutes. Unlike similar tools (e.g., CloudMan [18]), GP also supports the ability to change the Amazon EC2 instance type used in the deployment; thus, the user can easily customize a deployment to meet real-time requirements. For example, if workflow usage is low, micro or small instances can be used, while if memory requirements of a workflow increase, the running instances can be upgraded to large or extra-large instances. In addition, when the workflow platform is not being used, it can be suspended and restarted when required, thereby reducing the overhead of running an unused or sparsely used platform.

D. *Using Globus Provision to deploy Galaxy*

In order to deploy a Galaxy instance with Globus Transfer tools using GP, the user must create a topology file (galaxy.conf) describing the configuration (see Figure 3). The topology file defines the user's requirements with respect to four aspects: general, domain-simple, ec2, and globusonline. Briefly the options shown in Figure 3 are:

```
[general]
  domains: simple
[domain-simple]
  users: user1 user2
  gridftp: yes
  condor: yes
  cluster-nodes: 2
  galaxy: yes
  go-endpoint: cvrg#galaxy
[ec2]
  keypair: gp-key
  keyfile: ~/.ec2/gp-key.pem
  ami: ami-b12ee0d8
  instance-type: t1.micro
[globusonline]
  ssh-key: ~/.ssh/id_rsa
```

Figure 3. The Globus Provision topology file "galaxy.conf".

- "domain" specifies a single domain called *simple*. A topology can be divided into multiple domains, each with its own set of users, Globus services, etc.
- "users" defines the list of user names that will be added to the list of users on the Galaxy cluster.
- "cluster-nodes" specifies the initial number of worker nodes to be deployed.
- "go-endpoint" defines the name of the endpoint that will be created for this cluster. The created endpoint will be shown in the Globus Online Transfer interface for data transfer to and from the Galaxy instance.
- "keypair" and "keyfile" are the user's EC2 SSH keypair.
- "ami" is the base Amazon AMI that GP will use to create each host in the domain. Although any recent or custom AMI can be used, GP provides an AMI that has most of the necessary software pre-installed in it which considerably decreases the time taken to deploy an instance.
- "instance-type" specifies the EC2 instance types used. We have found that t1.micro is suitable for testing, c1.medium is good for demos, m1.large is used for high performance instances.
- "ssh-key" is the user's key which can be used to access Globus Transfer for high performance file transfer.

The parameters "gridftp", "condor" and "galaxy" define the required standard GP packages that will be set up in the instance. Through the topology file, the user's requirements

are translated to GP, including which servers need to be deployed, how many cluster nodes will be created, etc. When the GP instance is started, Galaxy with Globus Transfer can be accessed via the URL of the "simple-galaxy-condor" host. A Globus endpoint, with the name specified in the topology file, is also created and is accessible through either Galaxy or Globus Online.

IV. GALAXY TOOL INTEGRATION

In this section we present the development and integration of two different tools, Globus Transfer and CRData, into the Galaxy framework. The Globus Transfer tools are designed to optimize data transfer in Galaxy, while the CRData tools represent a custom tool deployment that provides a set of statistical tools for use by biomedical researchers. These two toolsets highlight the flexibility of the GP-based approach for deploying customized Galaxy environments.

A. Globus Transfer tools

Although Galaxy provides tools for uploading files via FTP and HTTP, the tools are often unreliable and inefficient when transferring large amounts of data, as is often the case in next generation sequencing. Moreover, files larger than 2GB cannot be uploaded to Galaxy directly from a user's computer. Consequently, we have extended Galaxy by adding Globus Transfer tools to provide a high performance, secure and reliable way of transferring large quantities of data in and out of Galaxy. The Globus Transfer integration in Galaxy is shown in Figure 4.

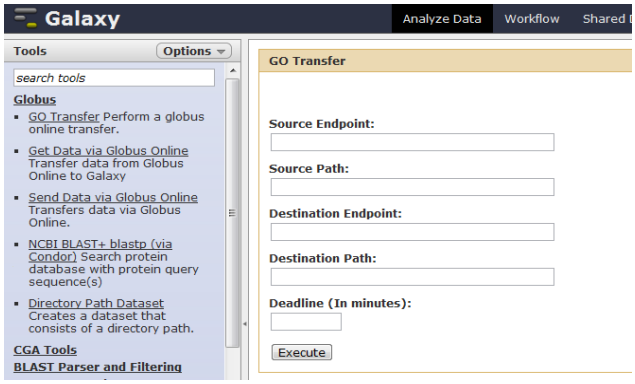


Figure 4. Globus Transfer toolset in Galaxy (this figure shows "GO Transfer" tool).

Globus Online Transfer (Globus Transfer) [19] is a hosted service that provides powerful Grid transfer capabilities to automate the task of moving files between sites, or "endpoints". Globus Transfer addresses the challenges in moving or synchronizing large quantities of data from different administrative domains in a secure, reliable, efficient and high performance way, without installing any software [10, 20]. Globus Transfer is responsible for transferring files, monitoring the transfer, retrying failures, auto-tuning performance and recovering from faults automatically, reporting status, and notifying users the completion of jobs via Email [19].

Globus Transfer moves files between Globus Endpoints (GridFTP [14] servers) using the GridFTP protocol. It also supports third party transfers, in which the selected endpoints are not collocated with the requesting user. Many resource providers include pre-configured Globus Endpoints for their user communities, while other data providers expose public Globus Endpoints as a way of easily sharing data with users. Users can also configure their own endpoints by installing Globus Connect [21] on their resources (such as a campus server, desktop computer or laptop). This allows users to access the machine as a Globus Endpoint, and transfer data to other endpoints.

By developing Globus Transfer tools for Galaxy we have incorporated the main benefits of Globus Online in Galaxy as well as providing access to the large network of existing Globus-enabled data sources. Through this integration Galaxy users are able to transfer files between existing data sources, their own resources and the Galaxy server securely, efficiently and quickly.

The Globus Transfer toolset includes three tools: 1) third party transfers between any Globus endpoints ("GO Transfer"), 2) upload to Galaxy from any Globus endpoint ("Get Data via Globus Online") and 3) download from Galaxy to any Globus endpoint ("Send Data via Globus Online"). Each of these tools has been added as a native Galaxy tool with an associated user interface to specify properties of the transfer.

For example, using the "GO Transfer" tool (see Figure 4), a file stored at a "Source endpoint" can be transferred to a "Destination endpoint" and the file is manifested as a Galaxy dataset in the history panel available for further analysis. If a "Deadline" for transfer completion is specified, the job will be terminated if it is not completed within the specified time period and Galaxy will indicate an error in its history panel. During execution, Galaxy invokes the Globus Transfer REST API to create and monitor the transfer, this information is used to update the status of the job in the Galaxy history panel. The user also receives a notification via Email when the transfer job is finished. Similarly, using the "Get data via Globus Online" tool, the "Destination endpoint" is the Galaxy server itself, and using the "Send data via Globus Online" tool, the "Source endpoint" is the Galaxy server.

Before using the Globus Transfer Galaxy tools, users need to create a Globus Online account (https://www.globusonline.org) and register an account in Galaxy with the same username. The user must also configure an X.509 certificate on the Galaxy server and in Globus Online that allows the Galaxy server to submit transfer requests on behalf of the user while guaranteeing security of the transfer. This is done by copying the generated X.509 certificate from the Galaxy Server and adding it to the user's profile through the Globus Online web site. Globus Online manages, on behalf of users, the security credentials required to authenticate against different endpoints, when submitting a transfer Globus Transfer will utilize the appropriate credential to "activate" the selected endpoint. Finally the Galaxy server must also include a registered Globus Endpoint, this is configured in the Globus

Provision for Galaxy recipes as described in the previous section.

B. CRData tools

CRData.org is a web-based computational tool designed to execute BioConductor scripts, written in R, on cloud resources. The platform gives users an intuitive interface for running R and Bioconductor, and is commonly used in many research domains. Although Galaxy offers many analysis tools for Next Generation Sequencing, it lacks many tools required for analysis in other domains. Consequently, we have developed the CRData toolset to complement the functionality of executing R scripts in Galaxy for the CardioVascular Research Grid (CVRG), which aims to create an infrastructure for sharing cardiovascular data and data analysis tools [22].

The CRData toolset consists of 35 tools with various functions. For example, the "heatmap_plot_demo.R" tool performs hierarchical clustering by genes or samples, and then plots a heatmap. The "sequenceDifferentialExperssion.R" tool (see Figure 5) performs a two-sample test for RNA-sequence differential expression. The "affyClassify.R" tool conducts statistical classification of affymetrix CEL Files into groups. And the "sequenceCountsPerTranscript.R" tool summarizes the number of reads (presented in one or more BAM files) aligning to different genomic features retrieved from the UCSC genome browser (http://genome.ucsc.edu/).

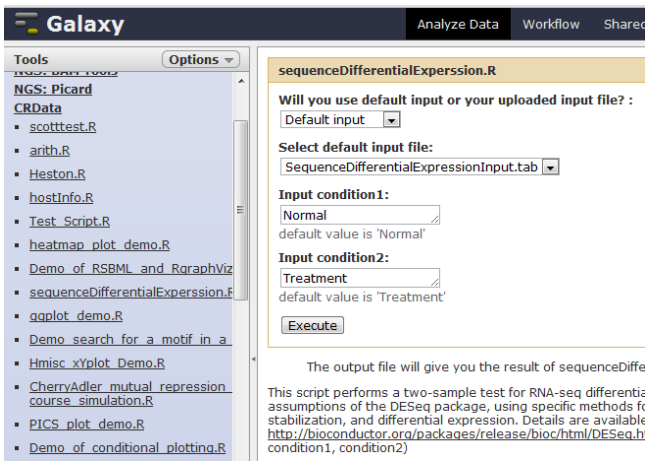


Figure 5. CRData tools in Galaxy (CRData includes 35 tools, this figure shows "sequenceDifferentialExperssion.R" tool).

Each CRData tool corresponds to an R script from CRData.org. The execution of a CRData tool invokes the corresponding R script, transfers input parameters and files to the script, and returns output files and figures after running R. The output is also shown in Galaxy's history panel for subsequent analysis or download.

Typically the process of adding new tools to a Galaxy instance is difficult as users must install all required software on the cluster and add tool definitions to Galaxy. However using our GP-based framework we can quickly and easily add the CRData toolset to the running instance. To demonstrate this flexibility we have created a new GP recipe, called "galaxy-globus-crdata.rb", which deploys and configures the CRData toolset on the cluster and adds the tool definitions to Galaxy. This recipe downloads and installs the necessary software (R, LibSBML, LibXML, GraphViz, cURL, etc.) and R packages on the "simple-galaxy-condor" host. The recipe is then transferred to the Globus Provision AMI when starting or updating a GP instance, and finally the generated Galaxy instance includes the CRData toolset.

The Globus Transfer tools and CRData tools can be accessed from the CVRG Galaxy portal [23].

V. EVALUATION

We now describe a real-world use case and the results of a performance evaluation of our extensions.

A. Use Case

After creating an AWS account and Globus Online account, a Galaxy instance is created on EC2 based on the properties specified in the topology file (galaxy.conf) described in Figure 3. The following commands are used to create and start a Galaxy instance with Globus Transfer and CRData tools.

$ gp-instance-create -c galaxy.conf
Created new instance: gpi-02156188

$ gp-instance-start gpi-02156188
Starting instance gpi-02156188... done!

The output indicates the GP instance id ("gpi-02156188") which allows monitoring of the status and URL of each host through the GP command "*gp-instance-describe*". When the GP instance is running, users (defined in the topology file) can connect to any of its constituent hosts.

In order to use Globus Transfer in Galaxy, the user must create an account in Globus Online and add an X.509 certificate to their profile in order to let Galaxy perform Globus Transfer jobs on their behalf. GP provisions the EC2 cluster with each user's Globus Online credentials so that each user can transfer data between Galaxy and any Globus-enabled endpoint. Each user in the cluster is given a user certificate signed by GP. The X.509 and GP certificates guarantee the security of data transfer when using Globus Transfer .

We consider a bioinformatics workflow as an example of using the elastic Galaxy environment. A summary of the example workflow is shown in Figure 6. In this example we configure the GP instance to run a CRData tool on a small cluster and then expand the cluster dynamically to run the same workflow on a larger dataset.

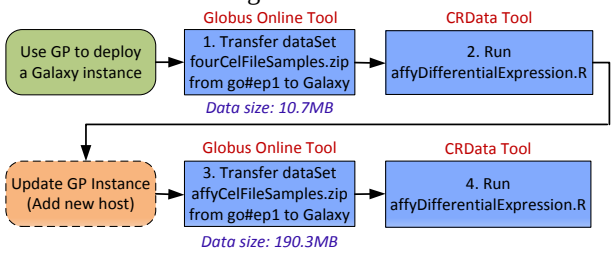


Figure 6. Bioinformatics workflow use case.

First, using the 'Get Data via Globus Online' tool in Galaxy, the dataset fourCelFileSamples.zip (10.7MB) is transferred from a Globus endpoint to the Galaxy server. The parameters "Endpoint" and "Path" are set as follows:

- Endpoint: galaxy#CVRG-Galaxy (the name of the remote endpoint)
- Path: /home/boliu/fourCelFileSamples.zip (the location of the file at this endpoint)

After execution, the uploaded dataset is shown in the History panel, it can also be downloaded by clicking the "Save" button.

After uploading the data, the user can run the appropriate statistical tool by selecting the 'CRData' tool and 'affyDifferentialExpression.R', and setting the parameters as shown in Figure 7. The tool runs the affyDifferentialExpression.R script which conducts two-group differential expression on Affymetrix CEL files. This script takes the dataset "fourCelFileSamples.zip" (uploaded in step 1) as input, and creates a "top table" of probe sets that are differentially expressed between CEL files that have been assigned to one of two groups.

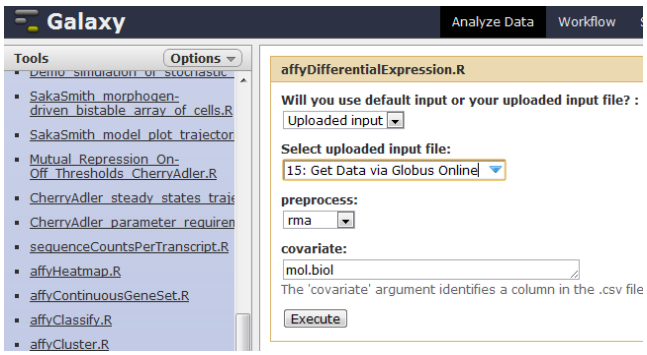


Figure 7. CRData tool "affyDifferentialExpression.R" (Step 3).

After execution, the output results are shown in the History panel, including both text output (see Figure 8) and figure output (see Figure 9 (a)).

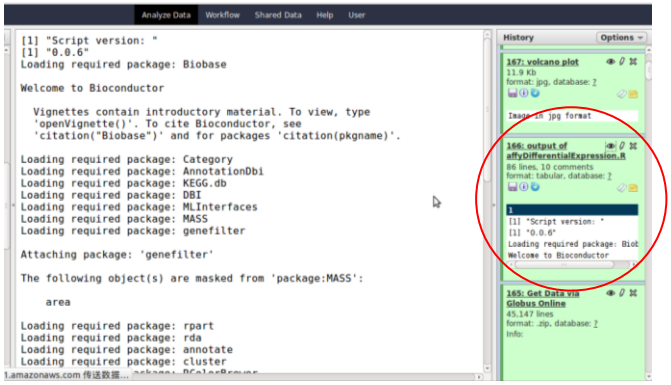
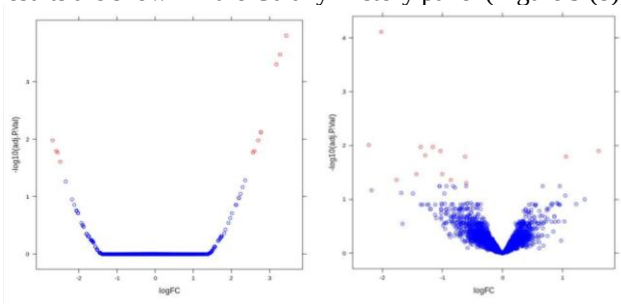


Figure 8. Text output of "affyDifferentialExpression.R" (Step 3).

The input dataset used for this example is only 10.7MB, which can be processed easily on a small EC2 instance. In the second stage of the example the user would like to process a larger dataset "affyCelFileSamples.zip" (190.3MB). However, this takes considerable time to process when using small EC2 instances. In order to speed up the workflow, a user can update the GP instance by adding a new EC2 host. This is done by creating a new GP topology file, and requesting a new host with the instance type "c1.medium".

$ gp-instance-update -t newtopology.json gpi-02156188

The user can then follow the same process as outlined above by transferring the dataset "affyCelFileSamples.zip" from galaxy#CVRG-Galaxy endpoint to the Galaxy server and then running the affyDifferentialExpression.R' tool to analyze the dataset "affyCelFileSamples.zip". The output results are shown in the Galaxy History panel (Figure 9 (b)).



(a) (b)

Figure 9. Figure output of "affyDifferentialExpression.R" in Step 3 (a) and Step 4 (b).

While the operation of updating the GP instance is optional, it does however decrease the execution time of Steps 3 and 4 from 10.7 minutes using a small instance to 6.9 minutes after adding a new medium instance. Presumably similar improvements could be obtained using larger instances. Moreover, the same approach can be applied for concurrent execution when multiple users submit tasks for execution at the same time.

This use case shows the ease by which users can deploy and scale their workflow environment to meet the needs of complex analyses or large-scale datasets. The GP-based approach can dynamically adjust the number of nodes and instance types at runtime, which can increase the performance of scientific workflows and potentially lower the cost of execution.

B. Performance evaluation

Figure 10 compares the deployment time, execution time and cost of Steps 3 and 4 on different EC2 instance types. We see that significant performance improvements can be obtained when using larger instances. For example, execution time decreases to 5.4 minutes on a large instance and to 4.6 minutes on an extra-large instance. However, performance improvements are disproportionate with cost, which almost doubles for each increase in instance size. The cost for executing Steps 3 and 4 on small and extra-large instances increases from 0.007 to 0.024 dollars. This figure also compares the deployment time using GP to set up a Galaxy instance with Globus Transfer tools and a set of bioinformatics tools. On a small EC2 instance, GP takes 8.8 minutes, the deployment time is reduced to 7.2 minutes on a medium instance and to 4.9 minutes on an extra-large instance.

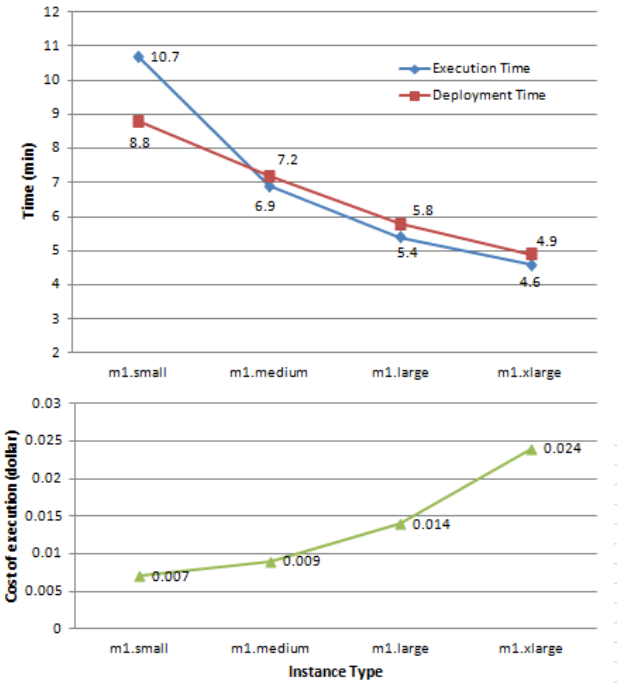


Figure 10. Comparison of execution time, deployment time and cost.

We compared the performance obtained using Globus Transfer with that achieved when using FTP and HTTP in Galaxy. Figure 11 shows the average transfer rate (in Mbits/sec) obtained when moving data from a laptop to the Galaxy server (running on a c1.medium instance) using different methods and file sizes. The transfer rate of the Globus Transfer method varies with file size, from 1.8 to 37 Mbits/sec, while the transfer rate of FTP varies from 0.2 to 5.9 Mbits/sec and HTTP is only able to achieve a transfer rate of less than 0.03 Mbits/sec (up to the maximum 2GB file size). We see that Globus Transfer outperforms FTP and HTTP significantly for all file sizes considered. In addition, Globus Transfer offers significant additional benefits in terms of security and reliability, which are crucial in many scientific domains.

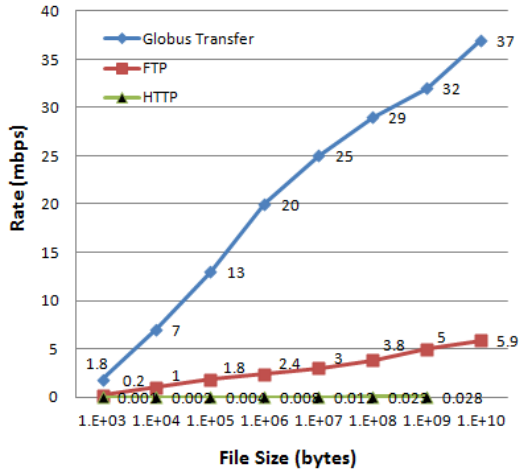


Figure 11. Comparison of average transfer performance

VI. Related Work

Many Cloud provisioning tools have been developed that can be used to accelerate the deployment of scientific workflow platforms. For example, the Chef system [17] that we build on in this work makes it easy to deploy servers and scale applications throughout the entire infrastructure; Puppet [24] gives system managers the operational agility and insight to manage dynamic infrastructure; Eucalyptus [25], Nimbus [26], and OpenStack [27] provide a software platform for the implementation of private cloud computing, and OpenNebula [28] offers complete management of virtualized data centers to enable on-premise IaaS Clouds. However, few researchers have used these tools to automate deployment and scaling of scientific workflows.

Other researchers have investigated the use of scientific workflow systems in Cloud environments. Dong et al. [29] propose a cost-effective strategy for intermediate data storage in scientific cloud workflow systems. Wu et al. [30] present a market-oriented hierarchical scheduling strategy in cloud workflow systems. Simmhan et al. [5] build the Trident scientific workflow workbench for data management in the Cloud. Zhang et al. [31] propose CloudWF, a scalable and lightweight computational workflow system for Clouds on top of Hadoop.

The most similar work to our approach is CloudMan [18], a cloud resource management system for individual researchers to compose and control an arbitrarily sized compute cluster on Amazon EC2. CloudMan provides a Web interface to automate the deployment of a Galaxy instance on EC2, and configure the number of nodes at run-time. We use Globus Provision over CloudMan for the following reasons:

1) Globus Provision provides more flexibility in defining user-specific node configuration, and adding recipes for installing additional software.

2) At run-time, CloudMan can only add or reduce the number of nodes, whereas Globus Provision can modify the whole configuration including adding and removing hosts and users, changing instance types, etc.

3) Globus Provision makes it convenient to extend Galaxy with arbitrary tools (e.g., Globus Transfer and CRData), which in our example satisfies the requirement for high performance and reliable large-scale data transfer and support for execution of R scripts in Galaxy.

VII. Conclusions and Future Work

The emergence of professionally operated Cloud computing infrastructures represents a considerable opportunity for scientific workflows, enabling easier system deployment, on-demand resource allocation, and elastic scaling. However, considerable challenges must be overcome before these infrastructures can be used effectively. These challenges include large-scale data partitioning and distribution, complex task scheduling and optimization.

In this paper, we proposed an automatic and elastic method for deploying a scientific workflow system, Galaxy, in a Cloud environment. Using Globus Provision, we demonstrated the configuration and deployment of Galaxy

on Amazon EC2 with the following features: on-demand provisioning, “pay as you go” style resource consumption, user-defined recipe configuration, and automatic instance deployment. We also extended Galaxy by adding Globus Transfer support for large scale data transfer as is commonly required in increasingly data-intensive research domains. Finally, we defined a model for users to easily and quickly add additional domain-specific tools to create a customized Galaxy system and demonstrated the value of this feature by adding the CRData toolset for our cardiovascular research scenario.

We plan to add features to our GP-based architecture, such as more fine-grained user-specific configuration, and to integrate more bioinformatics analysis tools within the Galaxy toolbox for gene and sequence data.

ACKNOWLEDGMENT

We appreciate the help of CVRG community and Globus Online team. We also appreciate the Galaxy Team for their support and maintenance of Galaxy. This work is supported by the NIH through the NHLBI grant, The Cardiovascular Research Grid, under contract number R24HL085343; US Department of Energy, under contract number DE-AC02-06CH11357; and US National Science Foundation, under contract OCI-534113.

REFERENCES

[1] Y. Zhao, I. Raicu, and I. T. Foster:, "Scientific workflow systems for 21st century, new bottle or new wine?," in 2008 IEEE Congress on Services, Honolulu, Hawaii, USA, 2008, pp. 467-471.

[2] I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, Eds., Workflows for e-Science: Scientific Workflows for Grids. Springer, New York, Secaucus, NJ, USA., 2007.

[3] W. Tan, R. Madduri, A. Nenadic, S. Soiland-Reyes, D. Sulakhe, I. Foster, and C. A. Goble, "CaGrid Workflow Toolkit: a Taverna based workflow tool for cancer grid," BMC Bioinformatics, vol. 11, p. 542, 2010.

[4] G. Juve, E. Deelman, K. Vahi, G. Mehta, B. Berriman, B. P. Berman, and P. Maechling, "Scientific workflow applications on Amazon EC2," in Workshop on Cloud-based Services and Applications in conjunction with 5th IEEE International Conference on e-Science (e-Science 2009), Oxford UK, 2009, pp. 59-66.

[5] Y. Simmhan, R. Barga, C. van Ingen, E. Lazowska, and A. Szalay, "Building the trident scientific workflow workbench for data management in the cloud," in Third International Conference on Advanced Engineering Computing and Applications in Sciences, 2009, pp. 41-50.

[6] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good, "On the use of cloud computing for scientific workflows," in IEEE International Conference on eScience, 2008, pp. 640-645.

[7] T. Dornemann, E. Juhnke, and B. Freisleben, "On-demand resource provisioning for BPEL workflows using Amazon's elastic compute cloud," in IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009, pp. 140-147.

[8] Amazon EC2. Available: http://aws.amazon.com/ec2/

[9] J. Goecks, A. Nekrutenko, and J. Taylor, "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences," Genome Biol, vol. 11, p. R86, 2010.

[10] B. Allen, J. Bresnahan, L. Childers, I. Foster, G. Kandaswamy, R. Kettimuthu, J. Kordas, M. Link, S. Martin, K. Pickett, and S. Tuecke, "Software as a service for data scientists," Commun. ACM, vol. 55, pp. 81-88, 2012.

[11] R project. Available: http://www.r-project.org/

[12] Galaxy. Available: http://en.wikipedia.org/wiki/Galaxy_(computational_biology)

[13] Globus Provision. Available: http://globus.org/provision/

[14] J. Bresnahan, M. Link, G. Khanna, Z. Imani, R. Kettimuthu, and I. Foster, "Globus GridFTP: what's new in 2007," presented at the Proceedings of the first international conference on Networks for grid applications, Lyon, France, 2007.

[15] J. Basney, M. Humphrey, and V. Welch, "The MyProxy online credential repository," Software: Practice and Experience, vol. 35, pp. 801-816, 2005.

[16] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: the Condor experience: Research Articles," Concurr. Comput. : Pract. Exper., vol. 17, pp. 323-356, 2005.

[17] Chef. Available: http://www.opscode.com/chef/

[18] E. Afgan, D. Baker, N. Coraor, B. Chapman, A. Nekrutenko, and J. Taylor, "Galaxy CloudMan: delivering cloud compute clusters," BMC Bioinformatics, vol. 11 Suppl 12, p. S4, 2010.

[19] Globus Online. Available: https://www.globusonline.org

[20] I. Foster, "Globus Online: Accelerating and Democratizing Science through Cloud-Based Services," Internet Computing, IEEE, vol. 15, pp. 70-73, 2011.

[21] Globus Connect. Available: https://www.globusonline.org/globus_connect/

[22] R. L. Winslow, J. Saltz, and I. Foster, "The CardioVascular Research Grid (CVRG) Project," AMIA Summit on Translational Bioinformatics: San Francisco, CA, 2011.

[23] CVRG Galaxy portal. Available: https://portal.cvrgrid.org/

[24] Puppet. Available: http://puppetlabs.com/

[25] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus Open-Source Cloud-Computing System," in Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on, 2009, pp. 124-131.

[26] P. Marshall, H. M. Tufo, K. Keahey, D. L. Bissoniere, and M. Woitaszek, "Architecting a Large-scale Elastic Environment - Recontextualization and Adaptive Cloud Services for Scientific Computing," in ICSOFT, 2012, pp. 409-418.

[27] OpenStack: The open source, open standards cloud. Available: http://openstack.org/

[28] P. Mvelase, N. Dlodlo, I. Makitla, G. Sibiya, and M. Adigun, "An Architecture Based on SOA and Virtual Enterprise Principles: OpenNebula for Cloud Deployment," Proceedings of the International Conference on Information Management & Evaluation, pp. 214-222, 2012.

[29] Y. Dong, Y. Yun, L. Xiao, and C. Jinjun, "A cost-effective strategy for intermediate data storage in scientific cloud workflow systems," in Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on, 2010, pp. 1-12.

[30] Z. Wu, X. Liu, Z. Ni, D. Yuan, and Y. Yang. (2011, 1.10.2012). A Market-Oriented Hierarchical Scheduling Strategy in Cloud Workflow Systems. Available: http://www.springerlink.com/content/t7t8823l8g88l827/fulltext.pdf

[31] C. Zhang and H. De Sterck, "CloudWF: A Computational Workflow System for Clouds Based on Hadoop." vol. 5931, M. Jaatun, et al., Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 393-404.