A Survey of Cloud-Based Service Computing Solutions for Mammalian Genomics

Philip C. Church, *Student Member, IEEE* and Andrzej M. Goscinski, *Member, IEEE*

Abstract—Cloud-based service computing has started to change the way how research in science, in particular biology, medicine, and engineering, is being carried out. Researchers in the area of mammalian genomics have taken advantage of cloud computing technology to cost-effectively process large amounts of data and speed up discovery. Mammalian genomics is limited by the cost and complexity of analysis, which require large amounts of computational resources to analyse huge amount of data and biology specialists to interpret results. On the other hand the application of this technology requires computing knowledge, in particular programming and operations management skills to develop high performance computing (HPC) applications and deploy them on HPC clouds. We carried out a survey of cloud-based service computing solutions, as the most recent and promising instantiations of distributed computing systems, in the context their use in research of mammalian genomic analysis. We describe our most recent research and development effort which focuses on building Software as a Service (SaaS) clouds to simplify the use of HPC clouds for carrying out mammalian genomic analysis.

Index Terms—Computer systems organization: general-emerging technologies, general literature: introductory and survey, software/software engineering: distributed programming

1 Introduction

Historically the price of genomic data collection has been the limiting factor in understanding and applying genomics to the mammalian species. The introduction of high-throughput data collection platforms has changed this, reducing the cost of collecting genomic data (human sequencing nearing the $1,000 mark) and enabling a range of applications, from providing disease risk assessments to identifying desirable genetic traits in livestock. However, due to the complexities of genomic analysis, putting these genomic applications into the public sector is difficult. Analysis of mammalian genomic data requires high performance computing (HPC) systems, large amounts of storage and specialized algorithms. As research is carried out by biology specialists/geneticists, these personnel must be trained in the use of HPC which adds to difficulty of analysis.

By studying the approaches currently used to analyse genomic data, we show that the limiting factor in understanding mammalian genetics no longer lies in the cost of data collection but in the manpower and computing resources required to analyse and interpret genomic data. Currently, clusters and grids offer users the required huge capacity storage, powerful hardware and advanced middleware needed to run these sophisticated discipline oriented applications. However, since these distributed systems have a high initial and maintenance costs, they are only affordable by rich institutions (research centres, universities, and industry labs). These complex computer systems require supporting staff to maintain, administrate and deploy software (and software updates), contributing to the ongoing cost of running a cluster. Furthermore, these labour-intensively managed resources are shared by many users, which usually leads to long waiting times for execution of software. Genomic applications such as personalized genomics, which require quick analysis, are unachievable when faced with shared resources with long job queues.

In response, the following first question was asked: *how can the turn-around time of genomic analysis be reduced cost-effectively?* We propose the use of on-demand computing, provided through clouds, to pay for computing resources cost by usage and to reduce the turn-around time of application execution. However, from the viewpoint of biology specialists, studies have identified performance and usability issues when using the cloud for HPC. Virtualization and low network speeds are issues brought up in (virtual machine based) cloud performance studies. Usability studies indicate that biology specialists lack the skills to setup, maintain and administrate HPC clouds. These issues led to the second question: *how can clouds be used by biology (non-computing) specialists to analyse and manipulate genomic data under a self-service model?* One solution, of critical importance, is hiding the complexities of cloud infrastructure-level services (e.g. virtual machine development, configuration and management services) through the use of software-as-a-service (SaaS) clouds. We show that deployment of a cloud-based genomic-specific SaaS framework which simplifies and/or automates common computer system administrative tasks could efficiently get non-computing specialists involved in running computationally demanding applications providing genomic analysis.

This survey is organized as follows. Section 2 describes the methodology and structure of this survey. Section 3 presents an overview of the techniques used to carry out mammalian

• *The authors are with the School of Information Technology, Deakin University, Locked Bag 20000, Geelong VIC 3220, Australia. E-mail: {pcc, ang}@deakin.edu.au.*

Manuscript received 2 Dec. 2013; revised 8 June 2014; accepted 15 Aug. 2014. Date of publication 11 Sept. 2014; date of current version 17 Dec. 2014. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TSC.2014.2353645

genomics (microarray and sequencing). Section 4 presents a survey of distributed systems (clusters, grids and clouds) and their applications in genomic analysis. Section 5 presents commonly used (HPC) cloud solutions. Section 6 examines usability and performance problems that occur when running HPC applications on clouds. Section 7 describes bioinformatics software services currently available on the cloud. Section 8 describes our current work, which aims to address usability issues such as virtual machine image indexing and auto-provisioning template discovery, HPC middleware, applications setup, and complex service deployment. The conclusion of the survey is presented in Section 9.

2 Survey Methodology and Structure

This paper is not meant to be an extensive survey; instead it has been carried out with the goal of providing an overview of the current state of mammalian genomics research on the cloud. To this end, database searches were carried out (covering up to November 2013) within the area of applied cloud and service computing bioinformatics solutions. Papers were identified which were written in English and cited more than 50 times. From these papers we surveyed the most representative cloud providers for hosting bioinformatics applications, and cloud based bioinformatics tools.

This survey is structured to benefit a reader looking to understand the growing role of service and cloud computing in mammalian genomic analysis. This paper introduces the computer-assisted methods for carrying out genomic analyses, the role of distributed and cloud computers to support these requirements, the problems biology specialists face in using these machines and the current push to software services. Lastly, we demonstrate how to take advantage of our survey to build SaaS clouds which support mammalian genomics research. By highlighting the processes undertaken by biology specialists when collecting and analysing mammalian genomic data, developers can build mammalian genomic analysis tools targeted to biology specialists. This survey also benefits biology specialists, by introducing cloud and service based systems. These systems can improve the turn-around time of genomic analysis in a cost-effective manner and make data processing easier through the use of brokers, discipline oriented user interfaces, and execution environments.

3 Genomic Analysis Methods and Software

The first step in carrying out any genomic study (discovery or analysis) is collecting data. The principles of genomic data collection have remained unchanged since the 1980's. Based on an understanding of complementary nature of nucleotide bases, cheap data collection has been made possible through miniaturization and automation of biological procedures, replacing human labour with laboratory robots. Microarrays and sequencers are the most common platforms for collecting genomic data; each has their own advantages of and disadvantages.

Once data is collected and stored, it must be analysed. The procedures carried out on genomic data are complex and therefore require the use of computers. Collected data must undergo a multi-step analysis process to extract information that can be put to use. The procedures carried out depend on the data collection platform.

3.1 Microarray Analysis

Microarrays used for gene profiling are able to examine the presence (expression levels) of thousands of pre-specified genes (probes). These platforms are cheap to construct and process data collected in hours; thus well suited to collect data from large populations or over long periods of time. For these reasons, microarrays has found use in a number of areas, including breeding of cattle [1], [2], risk estimates for diseases [3], as well as ancestry analyses [4]. Storing this data is a trivial manner as each array is small (on average 25 MB); however as experiments often consist of many arrays, some form of indexing system needs to be utilized.

The raw data of microarray experiments is stored as high resolution images. In order to analyse this data these images must first be normalized (corrected for visual errors and human variation) [5]. Various techniques exist to normalize data, including: standardizing intensity, construction of artificial reference arrays and quantile normalization [6], [7]. The choice of pre-processing algorithms and techniques has a significant impact on results.

Once the pre-processing step is completed, analysis can be carried out. Clustering is the most common method for extracting genes [8]. This technique is useful when trying to identify a set of genes which are responsible for a disease or trait. Gene Set Enrichment Analysis (GSEA) is another method that can be applied to normalized gene expression data [9]. This method can determine if a pre-defined set of genes is statistically relevant to an array of gene expression data. GSEA is often used in conjunction with clustering, where clustering is used to find genes related to a trait/disease. If GSEA finds evidence of the gene set, there is a high likelihood of the trait/disease.

3.2 Sequence Analysis

Sequence data is a representation of the complete DNA sequence in which genes are derived from. This data is collected using machines called sequencers. Like microarrays, sequences data can be used to measure gene expression (RNAseq) and identify gene variants (SNPs), however with no need to specify probes beforehand. DNA is stored as a text based format called FASTA [10], in which each base represented by one of four letters (A, T, C or G). The size of this data depends on the complexity of the organism; flu viruses are small ($\sim$5 MB) while human genomes are very large ($\sim$3.2 GB). Likewise the time taken to collect the sequence data is determined by the organism size. Modern sequencers, such as the Illumina HiSeq [11], can sequence a human genome in a week.

Current sequencing technology is based on shotgun sequencing that breaks up DNA into small fragments that have to be put together using computers. The software used to reassemble the scanned DNA fragments are called sequence assemblers. The algorithms used in these applications are often heavily memory intensive and based on pairwise comparisons that do not scale when faced with many genomes [12]. Only once a sequence has been aligned can analysis be carried out. Common analysis techniques include gene extraction, function prediction and function alignment. Most of these applications are distributed, utilizing super computing platforms.

Sequencing is currently used in a large gene discovery projects such as the 1,000 Genomes Project [13] (aimed to build a refined human genome map that can be used to find associations between gene and disease) and the Million Veterans Project [14] (that seeks to identify genes that are linked to military-related illnesses, such as post-traumatic stress disorder).

3.3 Mammalian Genomic Study Requirements

Once biological data has been obtained through genomics data collection platforms, data processing is required. There are many approaches to processing biological data depending on the type of data and biological question that the researcher wants to answer. Microarrays must be statistically analysed and turned into or compared to known profiles. For sequence data, initial analysis requires fragments of sequence data (generated by shotgun method) be re-constructed before genes and their functions are extracted.

Software applications exist to analyse genomic data collected from microarrays. Microarray applications for pre-processing are mainly command-line based. Most of the commonly used algorithms are only available through the bio-conductor library which requires knowledge of the R scripting language to utilize. Exposing these pre-processing algorithms through web interfaces could reduce the programming requirements of microarray normalization. Microarray analysis applications are better developed; clustering and GSEA algorithms can be accessed through graphical interfaces. There is room to extend current microarray methods to take advantage of distributed computing.

The algorithms applied to assemble and analyse genomic sequences are more complex and thus many software applications are written for distributed computers. The advantage is that these algorithms take less time to run assuming supercomputing resources are available. Researchers that wish to analyse sequence data must be trained in command line operation (Linux operating systems) and use of supercomputing resources. If genomics moves into the public sector, computational and training requirements will be multiplied. There is a need to expose these software packages in a manner that takes advantage of distributed systems. In the following section we investigate the types of distributed systems currently used to carry out genomic analysis. The goal is to identify the best way of providing computational resources for big gene discovery studies and genomic applications.

4 Distributed Computing

Genomic data must be analysed to obtain functional information. While microarray analysis (see Section 3.1) can be performed via off-the-shelf machines, performing sequence assembly and analysis (see Section 3.2) on these machines are computationally infeasible. There are three methods for improving performance of computers: make them faster (through improvement of CPU's), make them smarter (through development of better algorithms) or make them work together (through work distribution).

It is this third approach that has been applied to genomic sequence analysis software, as described in Section 2.3. These software applications make use of distributed computers to provide the computational resources necessary to analyse genomic data quickly. Clusters and grids are the distributed systems most often applied to genomics. However, these systems were built as high performance solutions with computing specialists as target users, and were not easy to use for biology specialists. For example, software is command line driven and no longer utilizes the graphical interfaces that simplify execution. Moreover, these systems require the support of administrative staff to deploy software and resolve user's issues.

However, as data gets cheaper and larger amounts of data are generated, requirements of distributed and parallel systems have started to change. Scalability, storage and cost are key requirements for analysing big (genomic) data. VM-based clouds provide these features at expense of performance. However, unlike managed clusters and grids, genomic analysis support provided through the self-service portal of most cloud solutions is limited from the viewpoint of biology specialists. From the cloud provider's viewpoint, the users must administrate their own computing resources, deploying and setting up genomic-specific HPC middleware. Lack of support puts more demand on the researcher; for clouds to support mammalian genomics, usability must be comparable to managed clusters and grids. In this section we describe clusters, grids and clouds, highlighting the advantages and disadvantages of each platform.

4.1 Clusters

A cluster consists of many computers connected to each other usually using a local high speed network. Compared to traditional single machine supercomputers, clusters could be more affordable in terms of scalability and fault tolerance support. By pooling resources and splitting tasks between machines the time taken to process data is reduced. Currently, the preferred approach to construct clusters is by using middleware.

Message passing interface (MPI) [15] is a middleware that is implemented as a code library that allows efficient parallel programs to be written for distributed memory systems. MPI is implemented in a number of languages including ANSI-C, FORTRAN, python and java. Hadoop (also referred to as map-reduce) is another popular cluster middleware. It consists of two steps, "map" in which the input is divided into smaller sub-problems, and distributes to worker nodes and "reduce" in which the answers to all sub-problems are collected and combines to form an answer. Both MPI and Hadoop cluster solutions are utilized when carrying out mammalian genomic analysis.

To construct a cluster, middleware is applied to all computers making up the cluster. One of these computers is defined as a head node, which serves as the access point by users. The head node is responsible for a range of tasks including network management, job submission and control, and scheduling. Users accessing the cluster first deploy their application on the head node. A script must be written consisting of the number of resources required and the application commands to be carried out. This script is submitted to the scheduler which checks the amount of available resources, if the job can be fulfilled it is sent to compute nodes, if not the job must wait in queue until enough resources are obtained.

While clusters bring considerable power to genomic analysis, there are a number of issues that need to be taken into account. First clusters can be expensive to set up; the initial cost of a cluster can range from between $100,000 for a 30 node cluster to $1,000,000 for a petascale system [16]. These machines require large amounts of power to run nodes in addition to any cooling systems. Further adding to the cost is the need for maintenance and administrative staff. Supporting staff spend time administering the system and supporting users, this includes: deploying and updating software, resolving account, security, and job deployment issues. Because these systems are expensive they are often shared, this means that researchers carrying out large studies must wait in queues until enough resources are made available. If more resources are required than what is available, the cluster must be upgraded, by adding more nodes, this adds to the cost of operating a cluster.

4.2 Grids

Like a cluster, a grid [17] consists of many different computers and hardware resources connected to a network; however a grid can share resources over private and public network, or even the internet. Grids allow for high performance computing to be done using resources that are separated by large distances. The advantage of a grid is that local resource consumption demands can be fulfilled via external resources, allowing for users to potentially avoid queues and for larger studies to be carried out.

Grid architecture is complex; issues such as resource management and scheduling of geographically distributed resources owned by different organizations with different usage policies (e.g., access rights), cost models and varying load and availability patterns ensures that grid development is a challenging and complex task [18]. Accessing a grid resource requires reservation that could affect the total execution time dramatically. In order to simplify grid development, toolkits containing grid software libraries and middleware are used. Commonly used toolkits include; Globus [19] and GRIDbus [20].

Globus is a software infrastructure that allows heterogeneous computing resources to be accessed as a single machine. It consists of a number of modules that define interfaces to low-level services, such as communications, security, data access and meta-computing directory.

GRIDbus is cluster and grid middleware technology for development of service-orientated computing. It provides a number of tools to support eResearch[1] applications. These include: visual grid application tools for creation of distribution applications, a cluster and grid scheduler, a web service directory, a grid accounting service, Gridscape for creation of dynamic and interactive test-bed portals, G-monitor a portal for web-based management of grid applications execution and GridSim a toolkit for performance evaluation.

1. eResearch refers to the use of information technology to support existing and new forms of research. The main features of eResearch applications include support for collaboration, use of grid computing technologies and being data intensive.

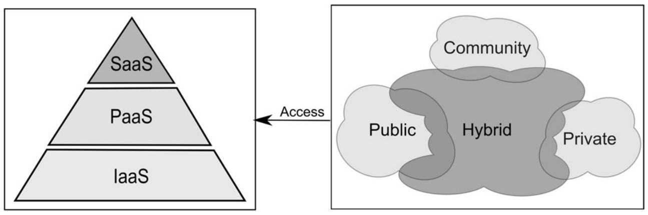


Fig. 1. NIST Definition of cloud computing.

Grids are often formed using cluster resources and therefore are just as expensive as clusters. Issues such as the initial purchase cost, maintenance and power consumption are shared between these forms of distributed systems. However, compared to clusters, grids can access computers across the world. This allows more flexibility in available resources, reducing (but not eliminating) the time spent waiting in queues. In return, grids must manage access to resources belonging to different users each with different access needs. Like in clusters, these grid resources must be administered by supporting staff. User support is more difficult, compared to clusters, as issues may occur across multiple organizational boundaries.

An example of the grid applied to bio-medical research is caBIG, a cancer biomedical informatics grid [21]. CaBIG provides tools and services to further cancer research; middleware and runtime environment are provided using the Globus toolkit. The caBIG consists of two parts: caGRID [22] that provides infrastructure and generic services and caCORE [23] that provides building blocks to create caGRID compatible software services. The benefits of caBIG are simplified access to resources through software services, provisioning of mechanisms to support development of software services and access enablement to a wide range of resources. The disadvantages are: the cost of constructing a grid and the complexity of administration and user support.

4.3 Clouds

Cloud computing [24] is a form of computing that aims to provide access to computational resources in an on demand and in a transparent way similar to that of the power grid. Clouds are unique in that they leverage virtualization technology to provide computational resources to users. Through virtualization, a cloud can better utilize resources, as multiple virtual machines can be deployed on the same computing resource without the user knowing. However, users have to wait for their virtual machines to be deployed. The NIST definition of cloud computing [24] is composed of three service models, and four deployment models (see Fig. 1).

The three service models used in clouds are infrastructure as a service (IaaS), platform as a service (PaaS) and SaaS.

- IaaS delivers compute, storage and network resources as a service. The fundamental resources provided by this layer are used by cloud platforms and software. Virtual machine users can be granted access to cloud infrastructure resources through the cloud portal or virtual machine administrator. Examples of IaaS clouds include Amazon EC2 [25] and OpenStack [26].

- PaaS facilitates deploying and running application by hiding infrastructure implementation details. Most cloud platforms consist of a high-level programming language and a well-defined Application Programming Interface. Users of a PaaS cloud are developers implementing, deploying and configuring cloud applications. Examples of PaaS clouds include Google App Engine [27] and Microsoft Azure [28].
- SaaS exposes applications designed to run on a cloud as services. It eliminates the need to install or run applications on the customer's computer and is often cheaper than buying a full software license. The cost of providing SaaS services could be increased by a need to deliver both data to the SaaS cloud service provider and results back to the user. Examples include specialized scientific software (Thermo Fisher's Laboratory Information Management System [29]), accounting software (FinancialForce@SalesForce [30]) and word processing software (Google Docs).

Who uses these clouds is defined by the deployment model. There are four models: public clouds, private clouds, community clouds and hybrid clouds. Public clouds can be accessed by anyone, allowing users to rent resources such as computational time or storage as necessary. Private clouds are used exclusively by organizations; the organizational natures of these clouds allow for a customer-specific service level agreement (SLA) to be made to ensure customer-specific minimum service quality standards. Community clouds are used by a group of users that have shared concerns, e.g., a shared mission statement that has specific security and policy requirements. The last model, hybrid clouds, combines cloud resources from two or more deployment models to accomplish a user's goal.

When using clouds as a form of distributed computing additional steps must be carried out in terms of the service model in use. At the IaaS level, this involves construction of a virtual cluster as well as compilation and deployment of distributed software. These tasks were previously the job of system administrators and are beyond the scope of most researchers. PaaS are aimed at developers, HPC is provided to users through a development environment and the automation of resource deployment. The problem of this approach is the user has limited choices of development tools and programming languages, thereby limiting the scientific applications that can be deployed. At the SaaS level the user is able to access HPC applications through graphical interfaces. These applications allow users to interact with the cloud without skills in network administration or software development; however the user is reliant on what cloud services have been made available. In specialist research areas such as gene expression profiling and drug discovery, such software services would have expensive licenses or not be readily available.

4.4 Summary

Clusters and grids are the distributed systems that are prominently used to analyse genomic data. These systems allow for high levels of performance but are expensive to purchase and maintain. Each of these distributed platforms abstract complexity from the user to simplify execution of software. Clusters make use of a single point of access, a head node. Grids extend this concept through the development of software services. These software services expose applications through web interfaces thereby simplifying access to the grid.

On the other hand, clouds are systems that trade performance for scalability and cost. Through clouds, users are able to access large amounts of storage and computational resources. Thus, instead of purchasing and constantly upgrading computer resources, users can access resources on demand for a fee. Clouds can also provide storage, which is charged in terms of GBs per month. For this reason, in cases where computational and storage requirements fluctuate, clouds are cheaper to utilize. The main downside of clouds is that they are complex systems which are difficult to use (through the same could be said of clusters and grids). When using IaaS clouds, the user must become the system administrator, allocating resources, deploying applications and setting up virtualized resources for HPC. However, clusters and grids rely on support staff to carry out these administrative tasks on behalf of the user. While this makes things easier for the users, the cost of employing these supporting staff contributes to the costs of cluster or grid computing.

HPC clouds could be a viable platform for carrying out genomic analysis, as they provide a mixture of scalability, performance, storage and cost. However, these HPC clouds are difficult for users who do not have administration and programming skills to set-up and access. This is problematic for clouds as genomic analysis is often carried out by non-computing/cloud specialists. For this reason, SaaS that is simpler to access, has been predicted as a key growth area in clouds [31]. Additionally there are number of cloud specific problems such as: virtualization, security, legal issues such as data and software ownership, and the lack of cloud standards (which make changing cloud providers difficult).

In order for clouds to be used by genomic researchers, abstraction must be provided similar to that of clusters and grids. In clouds, the necessity of this abstraction is seen through the push from IaaS to SaaS. Cloud solutions provide infrastructure, platform and software as services. Through a survey of common cloud solutions we identify the features provided by each cloud service level, and the methodologies to access these services.

5 Cloud Solutions

While cloud solutions fall into the three services levels as defined by the NIST definition (see Section 4.3), implementation depends on the provider. Only by understanding the cloud services made available by different providers, can clouds be utilized by genomic researchers. Due to insufficient cloud standards, this section surveys some representative cloud solutions using provider-specific terms when necessary. Focus is placed on the concepts used by the cloud providers.

5.1 Infrastructure as a Service Solutions

Amazon's EC2 [25] is the most well-known public infrastructure solution. Users of this cloud can rent resources on which to run their own software application. To run software, users create or modify virtual machine images called AMI (Amazon Machine Images). Once an AMI has been selected, users choose from a range of different hardware configurations (such as standard, high memory, high CPU and cluster compute). To run their configured AMI, virtualization based on Xen hypervisor technology [32] is used. The virtualization middleware provides a layer between the native OS and hardware, allowing multiple AMI to run on a single server. Once an AMI is started it is possible to access the cloud virtual machine through a terminal or in the case of Window based images, a remote desktop application. From this point onwards the operation of the cloud computer is exactly like a local desktop computer.

Based on the Amazon model a number of open source cloud solutions have been developed including Eucalyptus [33] and Openstack [26]. Users of those cloud solutions could self-manage and access their infrastructure resources like they do with Amazon EC2.

5.2 Platform as a Service Solutions

PaaS clouds are aimed at developers and thus provide functions to develop and host applications on IaaS cloud resources. Common features provided by these solutions include: development environments, programming language support and data storage. Microsoft's Azure [34] and Google's App Engine [27] are two popular public PaaS clouds.

Microsoft Azure is a platform for running and developing Windows compatible software and data. The architecture of Windows Azure has three parts, the compute service which runs applications, the storage service which stores data and the fabric which manages/monitors computers, switches and load balancers.

Google provides a PaaS based cloud called App Engine. This cloud allows users to develop and scale applications built using Python or Java. The App Engine cloud is a complex framework consisting of many services including; DataStore, a non-relational database which provides scalable data storage, Google Accounts which provides a secure login system and the Admin Console which measures application traffic (requests over time), quotas, and errors.

5.3 Software as a Service Solutions

SaaS clouds host and deliver a multitude of software services to users. SaaS solutions are often built around the concept of users accessing a repository of services. Users are given the ability to develop services through PaaS which can in turn be accessed through the service repository. Services accessed through the repository make use of IaaS to scale on demand. In this way, users can access software on the cloud without a programming or administrative background.

An example of a public SaaS cloud is SalesForce [35]; through this cloud users can access a repository of business orientated software services (the AppExchange Marketplace). To help users develop services, Salesforce provides a platform for development called force.com. This integrated PaaS cloud solution supports point and click development and a number of programming languages, including Python and Java; developed services are hosted on salesforce.com's infrastructure. Salesforce provides additional business orientated services, such as tools for keeping track of sales and customer service. Other SaaS clouds include CloudSwitch [36] (a private SaaS cloud which can deploy and run multitiered applications) and Oracle CRM [37] (a SaaS cloud that provides solutions for business and a range of industries but does not allow for public development). All of these providers focus on SaaS solutions in the area of business.

To the best of our knowledge there is no SaaS cloud solution aimed at providing applications for genomics or scientific research. The closet available SaaS solution for computational biology is Galaxy, a platform for workflow execution [38]. It provides repositories of software that can be accessed as a service, but it is not a cloud. Few applications in Galaxy take advantage of IaaS cloud scaling; instead most are hosted on servers. In terms of cloud enabled software services, providers must build these applications from the ground up resulting in a wide range of isolated services that are difficult to discover and access. By combining aspects of IaaS and PaaS clouds, a solution can be devised to support services for scientific research.

5.4 Summary

Clouds are diverse, each using different architectures and providing different services. IaaS providers allow users access to their hardware through the creation of virtual machines. The Amazon EC2 cloud is the good example of infrastructure provided as a service. Microsoft and Google provide PaaS clouds requiring users to develop their application software for these vender-specific PaaS clouds. These clouds are focused on developers, limiting the audience of these resources. Currently, there is a shift in users and companies moving towards proving cloud resources through SaaS. The resulting SaaS clouds are easier to use, and thus more profitable.

Current SaaS clouds are mainly business orientated; we are unaware of any solution dedicated to supporting scientific services. To provide gene discovery and analysis as a service, there is a clear need to provide a SaaS cloud solution. The next version of AGAVE promises to include public and private clouds to give users faster turnaround times on their experiments [39]. Such a SaaS cloud will also need to fulfil the software requirements described in section 2.3 and support HPC. Few clouds provide support for HPC, the ones that do operate at an IaaS level, for example Amazon EC2. In the next section we investigate the problems in utilizing HPC of the cloud, focusing on usability and performance.

6 HPC Cloud Current Problems

There are many advantages in using cloud computing for scientific research, in particular for execution of HPC applications. For bioinformatics, running sequence alignment on the cloud (on a once per experiment basis) represents significant savings. Despite the increased range of cloud compatible bioinformatics software (see Section 7), adoption of on demand computing has been slow. Reasons include poor usability caused by the difficulties in utilizing the cloud for

HPC and poor performance caused by the variability of the cloud due to virtualization and network infrastructure [40].

6.1 Usability

From the viewpoint of biology specialists, the difficulty of using clouds for HPC is a major contributor to the slow adoption of cloud technologies in genomics research. To utilize the cloud for HPC, a user must be aware of the procedures to setup and administrate a cluster (see Section 4.3). As IaaS clouds resources are provided at the server level, they must be modified to support distributed applications; middleware installed on each virtual server and distributed software deployed. Depending on the research problem, users must also be aware of cloud security issues stemming from network and virtualization vulnerabilities. These administrative tasks are time consuming and out of the scope of most researchers.

The issues that occur when applying cloud computing to support research are further shown in the Magellan report on cloud computing for science [41]. Released by the US Department of Energy, this report found that cloud introduced new complexities for users. First, users were not prepared for the amount of flexibility given over their virtual machines. Many of the problems faced by cloud users were centred on Linux administrative tasks. Many HPC users had little to no system administration experience and required help deploying software. The difficulty in using non-trivial security credentials was another issue where users required help generating the cryptographic tokens to access their resources. Lastly, challenges were encountered when managing data and ensuring workflows could take advantage of the scalability of the cloud. These issues were overcome with the help of support staff; however, as cases must be handled individually, this was not cost-effective.

These usability issues are also present in clusters and grids. These systems must be supported by staff that can carry out Linux administrative tasks (see Sections 4.1 and 4.2). Attempts have been made to simplify the usage of distributed systems. Work by Goscinski and Hobbs [42] proposes the use of services to provide ease of programming and use, reliability, availability through proper reaction to unpredictable changes and transparency. These concepts are currently incorporated into SaaS clouds, but have not been applied to HPC clouds. The development of SaaS HPC clouds can simplify the deployment procedures undertaken by non-computing researchers.

When using HPC clouds in an enterprise setting, different usability requirements must be met. According to [43] enterprise is focused on three key issues: security, interoperability and quality of service (QoS). Security is concerned with the protection and ownership of data due to the use of shared resources. Interoperability is the development of mechanism that allows for applications and data to be moved between different cloud providers. Finally, QoS regards establishing a service level agreement between the organization and cloud provider such that the contracted cloud services can be considered usable by the organization in terms of non-functional requirements such as availability, performance, security, etc. These issues can be applied to research, in particular the concept of interoperability. By providing compatibility between clouds resources, the cost and turnaround time of analysis can be improved.

Utilizing IaaS clouds to run applications can be a time consuming and expensive task. To access IaaS clouds users must manage cloud security credentials, and select and allocate resources. To use IaaS cloud resources, users must deploy applications, compress and transfer data. Cloud computing alleviates the cost and scalability issues of procuring required IT resources. However, the cost and time overheads in learning how to prepare a HPC cloud and then the applications to run it remains a problem. Although exposing the deployed cloud application as a service can simplify future access to the cloud, however this requires that a higher layer abstraction be created and graphical interfaces be designed to provide access to the deployed application, and applications must be stored and hosted for future use.

6.2 Performance

A number of studies have investigated the performance of cloud computers. These studies give varying results, some for and some against the use of clouds for HPC. A study by Napper and Bientinesi [44] runs the LINPACK benchmark (calculating floating point operations per second) on Amazon Extra-Large instances (in both the Standard and High-CPU categories). Results indicate that these Amazon instances are not yet mature enough for HPC computations. Suggestions are made to offer better interconnects or nodes provisioned with more physical memory. Amazon's Cluster Compute instances address both of these issues, with 10 GB interconnects and 28 GB of ram per instance. Amazon constructed a cluster of 880 cluster compute instances for a total of 7,040 cores. LINPACK was run on this virtual cluster; results ranking the Amazon EC2 cluster 231 on the TOP500 super computer list [45]. These benchmark results show the potential of cloud computing for carrying out HPC.

Other studies carried out bring up the price of virtualization. A study reported in [46] investigated the network interconnect of EC2. An application called CPUtest was used to measure processor sharing, Round-trip Delay Time, Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) throughput and packet loss. Observed results show abnormally large packet delay variations between cloud instances. Unstable TCP/UDP throughput was also seen, caused by end host virtualization. The effects of virtualization on HPC clouds were further studied by work performed at Indiana University. This study measured the virtualization overhead of Xen and Eucalyptus through three practical applications (matrix multiplication, k-means clustering and the concurrent wave equation solver). Results showed a moderate-to-high virtualization overhead when running MPI applications [47].

More recently, work reported in [48] investigated the performance of Amazon EC2 cluster compute instances, focusing on communication and scalability. Three benchmarks were carried out in this study: the Intel MPI Benchmarks to test point-to-point data transfer, the NAS Parallel Benchmarks (NPB) to test the impact of para-virtualized networks on the scalability of representative parallel codes, and the NPB Multi-Zone (NPB-MZ) suite in order to determine whether a hybrid parallel programming model could overcome the current Amazon EC2 network limitations. Analysis of benchmark results found the scalability of HPC applications on public cloud infrastructures relies heavily on communication (both on the network fabric and support in the virtualization layer). The Amazon Cluster Computer instances were found to provide more computational power and slightly better point-to-point communication performance. However, as these Cluster Compute instances do not provide proper I/O virtualization support, the use of standard EC2 instances scale better and are generally more cost effective.

6.3 Summary

From the viewpoint of biology specialists, usability is one of the major issues in utilizing clouds for HPC. Most researchers currently using HPC are familiar with administered resources. In these systems, setup and deployment are handled by system administrators. In the cloud model, support staff could be minimized such that users have to play a major role in administering their own cloud resources. As described in the Magellan report, this leads to a issues in the areas of deployment, security and resource management. Supporting staff are required to handle these issues; however this is not cost effective. While clouds can improve the use of resources, this need for system administrators increases the cost of utilizing clouds for research. Enterprise focuses on usability at an organizational level through three key requirements, security, interoperability and QoS. While all of these concepts can be applied to research, interoperability should be a key requirement. However lack of cloud standards means that interoperability is not present in most clouds.

The main performance criticisms are supposedly addressed by Amazon's HPC cluster compute instances. These instances have 10 Gb Ethernet interconnect and more physical memory. Performance results for these new cloud instances range from supportive to dismissive. One of the best cases made for HPC computing clouds is Amazon EC2's ranking on the TOP500 super computer list [45]. On the other hand, results from Expósito et al. [48] indicate that the cloud instances making up this virtual cluster still suffer from communication issues.

In conclusion, usability is an issue for users accessing clouds. Even with experience in HPC clusters, users had trouble with administrative tasks and required the help of support staff. These usability issues could be addressed through the creation of a HPC enabled SaaS cloud. In terms of performance, options are split, both for and against the cloud for HPC. This indicates that further performance testing is required focusing on how the Compute Cluster instances operate in a practical setting. If performance issues could be clarified and addressed, Amazon EC2 and similar clouds (Eucalyptus, OpenStack) could provide resources required for HPC SaaS clouds.

7 Bioinformatics Cloud Software Services

An analysis of the current state of projects and development of computing based frameworks and tools to support scientists leads to two major areas: (i) e-Research tools based on Web API and cluster and (ii) HPC research clouds, supporting applications exposed as services (SaaS). This survey investigates how popular cloud software services have been developed, operate and are exposed to users. We examine three solutions; the Galaxy framework which provides features to simplify service development (such as XML-based web form generation) and incorporates the cloud through plugins, the Tuxedo Suite which are isolated services, built from the ground up to utilize the scalability of cloud resources and CloudBioLinux, a virtual machine template containing bioinformatics tools.

7.1 Galaxy Plugins

Galaxy [38] is a data integration, data analysis, and data publishing platform as well as a scientific workflow engine for computational biology. Galaxy users can share and publish analysis software as services, exposed through web forms defined using a XML like language. Galaxy users without programming experience can access published applications, specifying parameters through a web form. While a majority of developed services are hosted and exposed through Galaxy, some services incorporate cloud resources.

A recent paper by Wong and Goscinski [49] presents a Galaxy plugin allowing users to create and terminate HPC clusters on Amazon EC2. This plugin relies on virtual machine images (AMI) with bioinformatics application already installed. Using the cluster creation tool, users select an AMI along with the amount of resources they require. The default AMI generates a MPI based cluster which contains a sequence aligner (mpiBLAST [50]) and simulation tool (NAMD [51]). Once the virtual machine image is deployed, applications are accessed via Galaxy interfaces.

Recent work from the University of Chicago [52] deploys a bioinformatics workflow across local and Amazon EC2 resources. Combining the features of Galaxy and Globus allows for a robust research cloud that supports automated GUI generation, software sharing and workflow deployment. During workflow deployment, data was transferred through a web interface and resources selected manually through creation of a topology file. Using this plugin, an R script to carry out microarray analysis was deployed to a small Amazon EC2 instance.

7.2 Tuxedo Suite

The Tuxedo suite brings together commonly used bioinformatics tools for analysing genomic data generated from high throughput sequencing machines. Some of these tools provide support for HPC platforms in order to reduce the time taken to process genomic data. Two HPC enabled tools in this suite, Crossbow and Myrna, have been made available as cloud services.

Crossbow [53] is a scalable application for genome sequencing aimed at performing alignment between the small fragments produced by current generation sequencing machines. Crossbow has been integrated into Amazon EC2 using a map-reduce strategy [54]. The operation of Crossbow is shown in Fig. 2, where a virtual Hadoop cluster is first created on the Amazon EC2 cloud. Sequence fragments are then uploaded from the user's desktop to Amazon storage (S3). Next crossbow code is uploaded to the

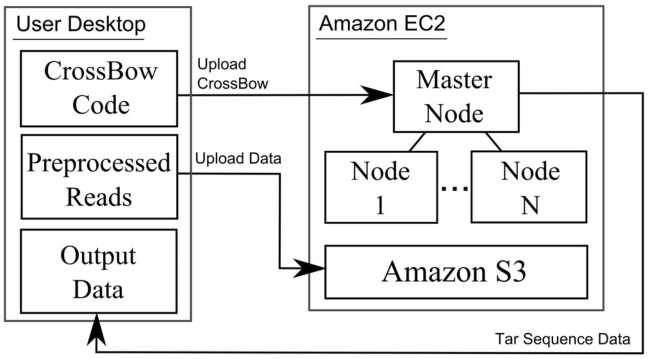


Fig. 2. Crossbow cloud workflow.

master/head node, compiled and run. Output data is compressed and sent back to the user.

Myrna [55] is a cloud-scale software for analysing RNA-seq data. It aligns sequence data, normalizes and carries out statistical modelling in a single computational pipeline. Like crossbow, Myrna can be accessed as a service through a web interface provided on the official site (see Fig. 3). Through this interface users: provide their Amazon EC2 keys, set the number of machines to run Myrna, and set Myrna specific options which can improve the accuracy of generated results.

Execution of Myrna as a service first creates a virtual cluster on Amazon EC2 using the Hadoop middleware before deploying and executing Myrna. Each step in the Myrna pipeline is then mapped and reduced. A map stage takes a stream of input data, analyses and returns results in the form of a stream. A sort and shuffle phase is carried out that bins and sorts data according to data similarity. Lastly, the reduce stage performs computation on data which is similar.

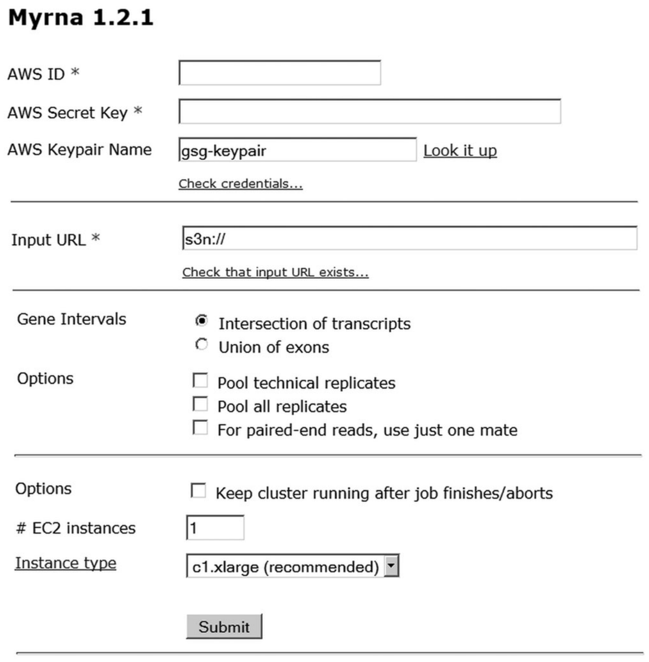
7.3 Cloud BioLinux

Cloud BioLinux is a virtual machine (VM) configured for high-performance bioinformatics using cloud platforms. At time of writing [56], over 135 bioinformatics tools have been deployed and configured on the virtual machine. Users access and execute installed applications through unique graphical interfaces or command line, and documentation of each installed application is made available through a centralized website.

Cloud BioLinux is compatible with a range of IaaS clouds including Amazon EC2 [25], OpenStack [26] and Eucalyptus [33]. Currently Cloud BioLinux does not provide HPC support; instead each user runs distinct VMs that encapsulate an operating system, analysis software and data. With this approach, users receive distinct and pre-defined computational resources, since each user initializes their own Cloud BioLinux server.

7.4 Summary

Bioinformatics cloud software services allow users to take advantage of scalability and clouds without understanding cloud architecture. Through graphical interfaces, users can access HPC clouds without needing to carry out the time consuming task of setting up middleware to create a virtual cluster in the cloud. However, this is only possible due to researchers with skills in genomics, HPC computing, programming and cloud computing, who have taken the time to develop and expose applications as services. The combination of skills required to develop bioinformatics cloud software services, has contributed to the limited number of tools targeted to the cloud.


Myrna 1.2.1

AWS ID *	
AWS Secret Key *	
AWS Keypair Name	gsg-keypair Look it up
	Check credentials...
Input URL *	s3n://
	Check that input URL exists...
Gene Intervals	(●) Intersection of transcripts
	( ) Union of exons
Options	☐ Pool technical replicates
	☐ Pool all replicates
	☐ For paired-end reads, use just one mate
Options	☐ Keep cluster running after job finishes/aborts
# EC2 instances	1
Instance type	c1.xlarge (recommended)

Fig. 3. Myrna service interface.

Popular services fall under two categories, individual isolated services (such as the Tuxedo Suite and those contained in Cloud BioLinux), and platforms (such as Galaxy). As seen in the Tuxedo Suite, behind the web interface presented to the user is the process of deploying and running the software package. Frameworks such as Galaxy can simplify the development, deployment and provision of discovery of cloud software services by automating tasks such as interface development.

8 SaaS Clouds Supporting Research in Mammalian Genomics

To facilitate genomic analysis of the mammalian species, genomic data collection platforms, data analysis software and computing resources are required. Despite improvements to data collection platforms, analysis is still very time consuming and costly. The survey has defined two areas in which improvements could reduce the cost and time of genomic analysis: development of genomic software and development of a SaaS cloud framework.

Genomic software should be developed that reduces the cost of genomic analysis. Developed software could fill the holes identified through the survey of analysis methodologies and software (see Section 3). The goal would be to improve the algorithms used to analyse genomic data and reduce the need for specialized staff. Software could be developed that utilize novel approaches to analyse data, for example microarray normalization algorithms could be written to take advantage of distributed computing platforms. The need for multidisciplinary staff/training could be reduced through the development of software services that are accessed through graphical interfaces. Through

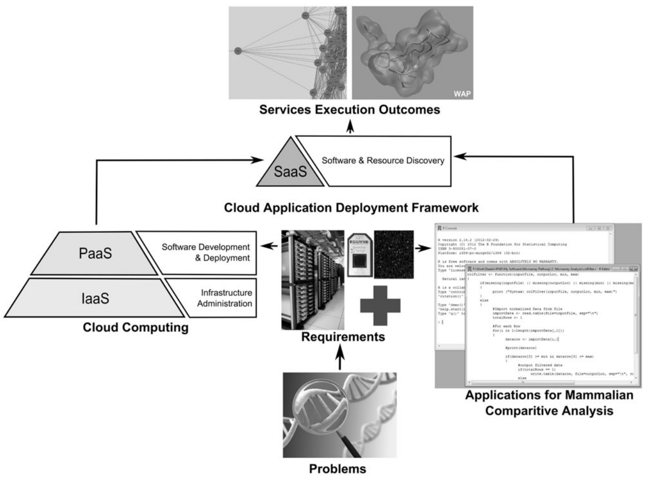

Services Execution Outcomes
SaaS
Software & Resource Discovery
Cloud Application Deployment Framework
PaaS
Software Development & Deployment
IaaS
Infrastructure Administration
Cloud Computing
Requirements
Applications for Mammalian Comparitive Analysis
Problems

Fig. 4. Personal genomics workflow.

these services, biologists will not be required to learn command line or programming languages, thus simplifying the analysis procedure.

Cloud computing could be used as a potential source of cheap resources; however when using clouds for HPC, users encounter usability and performance issues (as described in Section 5). A SaaS cloud framework could provide an answer; by providing HPC, Storage and Security, the devised cloud enabling researchers to take the role of cloud developer. Such a framework must also meet the requirements of researchers carrying out genomic analysis (usability, interoperability and performance).

Implementation of a SaaS cloud framework will reduce the cost of genomic analysis through a combination of software services and cheap computing resources. By utilizing the derived SaaS cloud in conjunction with genomic software, a combination of performance and usability can be provided. Resources are provided through the IaaS layer, while genomic software is exposed as cloud services though the SaaS layer. A summary of the proposed approach is seen in Fig. 4, where genomic analysis algorithms are deployed on the devised cloud framework as services.

The following subsections report on the construction of SaaS clouds support scientific research, in particularly mammalian genomics. Through the development and implementation of a framework built upon an IaaS cloud, researchers can deploy SaaS that take advantage of cloud resources. Three SaaS cloud solutions are described below which allow users to deploy software as services and (through these services) access on-demand resources.

8.1 HPCynergy/West-lin

HPCynergy [57] is a cloud that provides users with an HPC oriented environment where computing resources (from data storage to complete cluster environments) are accessible as on-demand, easy-to-use services. Compared to other cloud solutions, HPCynergy is focused on supporting scientific research by providing clusters and (scientific) workflows as services. HPCynergy supports the publication and selection of infrastructure and software resources. These resources are made available through a dynamic broker that can keep track of service availability.

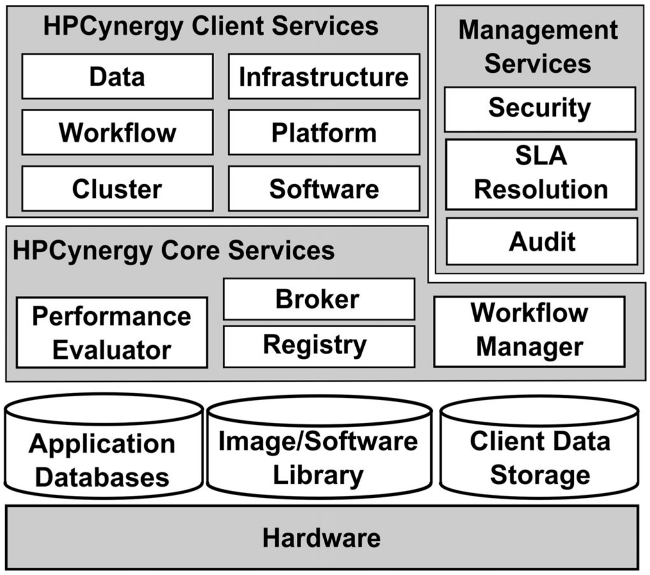
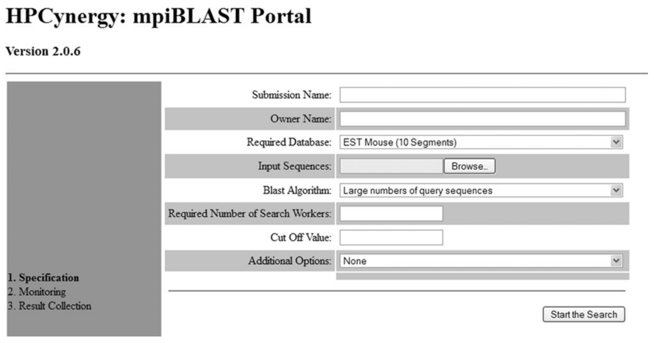


Fig. 5. Relationship of services provided by HPCynergy.

As seen in Fig. 5, HPCynergy is designed as a collection of services (core, management and client). Core Services consist of a dynamic broker, a registry for service discovery, a workflow manager and a service evaluator for service execution. On top of the Core Services are the HPCynergy Management Services that consists of security, SLA resolution and Audit. The security service provides users with congeniality, integrity and authentication. The SLA resolution service settles disputes and carries out any penalty/compensation agreements outlined in Service level agreements. The Audit service monitors all interactions with all services and ensures that clients pay for what they use. The HPCynergy Client Services allowing users to access a range of resources (including clusters, software and workflows) through web interfaces.

An mpiBLAST service was made available through the HPCynergy PaaS cloud [58]. This service required that compute resources were published to the dynamic broker service. Using the Dynamic Broker Interface, cluster requirements were published as attribute values, such as IP address/DNS, processor speed, virtualization and middleware (e.g., MPI), the number of nodes, required runtime libraries, even required search databases. For this study, resources were provided by a research cluster called West-lin. This cluster is comprised of both physical and virtualized nodes. In terms of hardware, the cluster has been constructed using 20 servers, each with two quad-core Xeon processors. Of these 20 servers, 10 have VMware vSphere [59] installed thus allowing them to run virtual machines instead of a single operating system. All nodes (virtual and physical) run the CentOS operating system and have Sun Grid Engine and Ganglia installed.

Like IaaS clouds, West-lin supports virtualization. However usage differs when compared to the Amazon-like IaaS clouds. Instead of allowing users to deploy virtual machines to nodes, the focus is put on using virtualization to gain more efficiency from the cluster. Virtual machines can be migrated/rearranged so that busy virtual machines can run on dedicated whole physical systems while idle virtual machines can be consolidated to single physical nodes.



HPCynergy: mpiBLAST Portal

Version 2.0.6

Submission Name:	
Owner Name:	
Required Database:	EST Mouse (10 Segments)
Input Sequences:	Browse...
Blast Algorithm:	Large numbers of query sequences
Required Number of Search Workers:	
Cut Off Value:	
Additional Options:	None

Fig. 6. HPCynergy service interface (mpiBLAST).

There is no focus on clients being able to create and maintain their own virtual machines.

The dynamic broker interacted with the Ganglia monitoring software installed on each node to monitor the states of available compute resources. As resources were provided through West-lin, both virtualized and non-virtualized resources could be specified. MpiBLAST was also published to HPCynergy as a software service, along with a mouse genome database. These resources and mpiBLAST application were made available through a web interface (see Fig. 6). Invocation of mpiBLAST through the HPCynergy service interface requires the specification of a number of processes, an input file containing required search sequences, and the name of a pre-formatted (gene) database to perform the search on. MpiBLAST makes use of the number of process to fragment the selected database. Through this interface, 3,200 unknown sequences were identified using 10 cluster nodes. Performance results indicated a linear execution, ideal to take advantage of scalable cloud resources.

8.2 HPCaaS

HPCaaS [49] is a software deployment tool for the Galaxy workflow engine that aims to simplify access to Amazon EC2 cloud resources. HPCaaS makes use of Galaxy (as an application broker) and an IaaS cloud (which provides, among other capabilities, storage of virtual machine images). To facilitate communication between Galaxy and IaaS clouds, a middleware layer is also defined based on the EC2 command line tools [60]. The dependencies of this middleware layer are Perl and CURL, which are normally included in many Linux distributions. The middleware

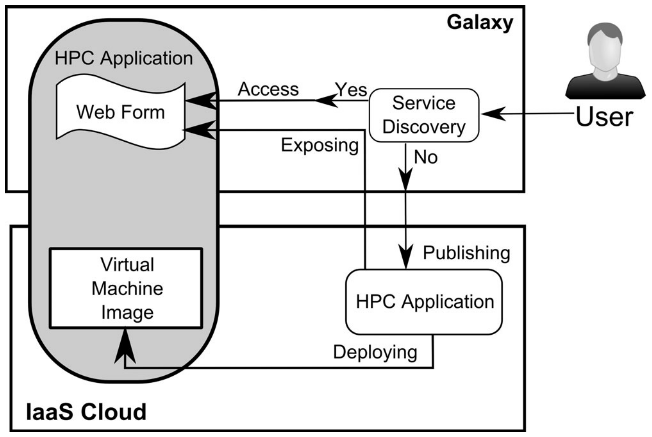
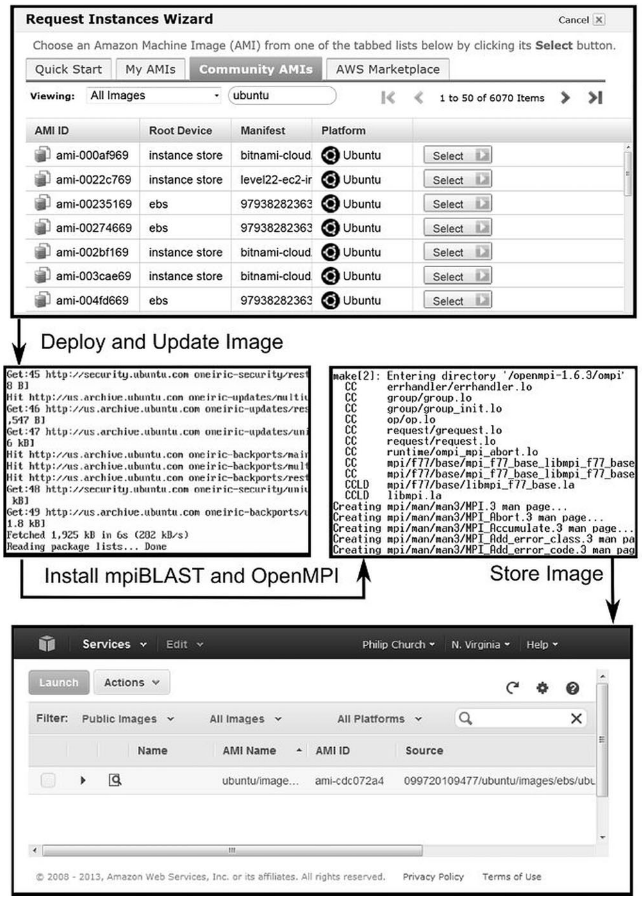


Fig. 7. Relationship of services provided by HPCaaS.



Request Instances Wizard

Choose an Amazon Machine Image (AMI) from one of the tabbed lists below by clicking its **Select** button.

AMI ID	Root Device	Manifest	Platform	
ami-000af969	instance store	bitnami-cloud	Ubuntu	Select
ami-0022c769	instance store	level22-ec2-ir	Ubuntu	Select
ami-00235169	ebs	97938282363	Ubuntu	Select
ami-00274669	ebs	97938282363	Ubuntu	Select
ami-002bf169	instance store	bitnami-cloud	Ubuntu	Select
ami-003cae69	instance store	bitnami-cloud	Ubuntu	Select
ami-004fd669	ebs	97938282363	Ubuntu	Select

```
Get:45 http://security.ubuntu.com oneiric-security/rest
8 B]
Hit http://us.archive.ubuntu.com oneiric-updates/multiu
Get:46 http://us.archive.ubuntu.com oneiric-updates/res
,547 B]
Get:47 http://us.archive.ubuntu.com oneiric-updates/uni
6 kB]
Hit http://us.archive.ubuntu.com oneiric-backports/mair
Hit http://us.archive.ubuntu.com oneiric-backports/mult
Hit http://us.archive.ubuntu.com oneiric-backports/rest
Get:48 http://security.ubuntu.com oneiric-security/uniu
 kB]
Get:49 http://us.archive.ubuntu.com oneiric-backports/u
1.8 kB]
Fetched 1,925 kB in 6s (282 kB/s)
Reading package lists... Done
```

```
make[2]: Entering directory '/openmpi-1.6.3/ompi'
  CC      errhandler/errhandler.lo
  CC      group/group.lo
  CC      group/group_init.lo
  CC      op/op.lo
  CC      request/grequest.lo
  CC      request/request.lo
  CC      runtime/ompi_mpi_abort.lo
  CC      mpi/f77/base/mpi_f77_base_libmpi_f77_base
  CC      mpi/f77/base/mpi_f77_base_libmpi_f77_base
  CCLD    mpi/f77/base/libmpi_f77_base.la
  CCLD    libmpi.la
Creating mpi/man/man3/MPI.3 man page...
Creating mpi/man/man3/MPI_Abort.3 man page...
Creating mpi/man/man3/MPI_Accumulate.3 man page...
Creating mpi/man/man3/MPI_Add_error_class.3 man pa
Creating mpi/man/man3/MPI_Add_error_code.3 man pag
```

Name	AMI Name	AMI ID	Source
	ubuntu/image...	ami-cdc072a4	099720109477/ubuntu/images/ebs/ubu

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Fig. 8. HPCaaS AMI deployment.

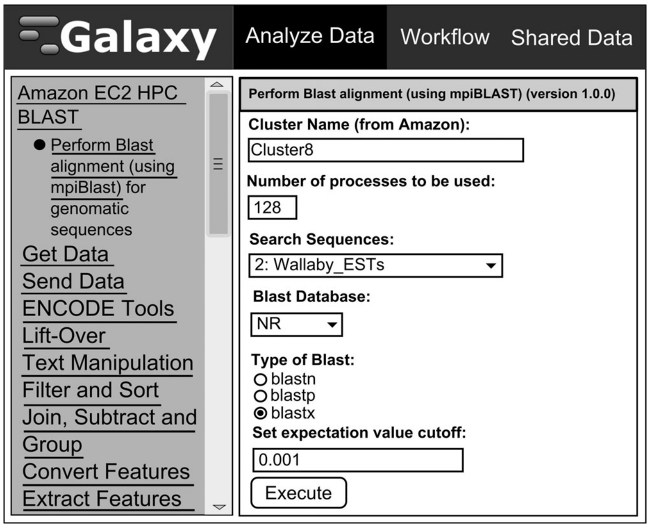
layer supports HPC cluster creation, termination, HPC-application job submission, and retrieval of results of HPC-application execution [61].

The operations undertaken by the user of the HPCaaS tool are shown in Fig. 7. Users access services through Galaxy in two ways. If users find their required service they can access the HPC application through a specified web form. Otherwise they must deploy a service on an IaaS cloud; this involves the deployment of the required HPC application on a virtual machine image, and creation of a web form exposing the virtual machine as a service.

Using HPCaaS a case study was carried out to demonstrate how a service could be deployed and executed on Amazon EC2 resources. A sequence aligner (mpiBLAST) was chosen as the service to be deployed and exposed using HPCaaS. Deployment began with the creation of the virtual machine image (see Fig. 8).

HPCaaS does not provide any mechanisms to support development of virtual machine images, therefore Amazon EC2 development mechanisms were utilized. In order to setup the cloud image for this study, an Ubuntu server AMI was first selected from the Amazon EC2 web interface and launched. The Ubuntu image is immediately updated to ensure there are no security vulnerabilities. Software is then deployed to the virtual machine (in this case mpiBLAST and OpenMPI). The Amazon cloud image is then stored in its modified form for future use.

The deployment process also required the development of an interface that exposed the virtual machine



Perform Blast alignment (using mpiBLAST) (version 1.0.0)

Cluster Name (from Amazon): Cluster8

Number of processes to be used: 128

Search Sequences: 2: Wallaby_ESTs

Blast Database: NR

Type of Blast: blastn blastp [✓ blastx]

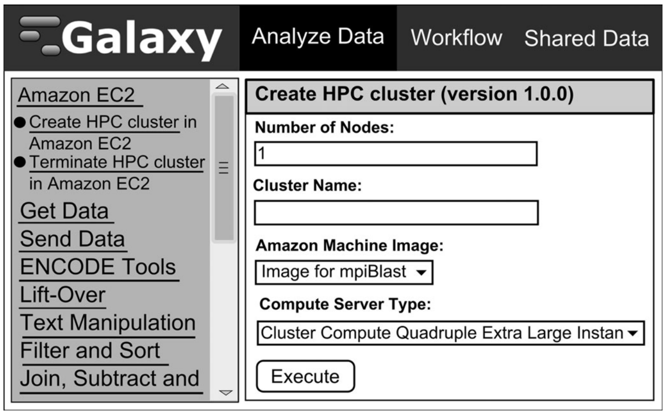
Set expectation value cutoff: 0.001

Fig. 9. HPCaaS mpiBLAST service interface.

image. These web forms are specified using XML file defining tags recognized by Galaxy. The resulting interface (see Fig. 9) consisted of six input controls, described as follows;

- *Cluster Name*. The name given to the cloud cluster deployed using HPCaaS.
- *Number of processes*. The number of instances of mpiBLAST that should be run as well as the number of fragments that the mpiBLAST database is divided into.
- *Search Sequences*. The file containing the sequence fragments in which to align.
- *The database type*. The database in which to BLAST.
- *The type of blast*. The version of BLAST to run. This depends on the type of data being analysed.
- *The expectation cutoff*. A text control which allows a user to choose a value in which to filter results.

The HPCaaS tool was then modified to enable access to mpiBLAST as a service. The generated virtual machine image was published to Galaxy, added to the list of available cloud images that could be deployed using HPCaaS (see Fig. 10). A new directory was created to store the mpiBLAST interface in the same manner as other Galaxy tools. Once the service was published, users could access the new



Create HPC cluster (version 1.0.0)

Number of Nodes: 1

Cluster Name:

Amazon Machine Image: Image for mpiBlast

Compute Server Type: Cluster Compute Quadruple Extra Large Instan

Fig. 10. HPCaaS galaxy deployment interface.

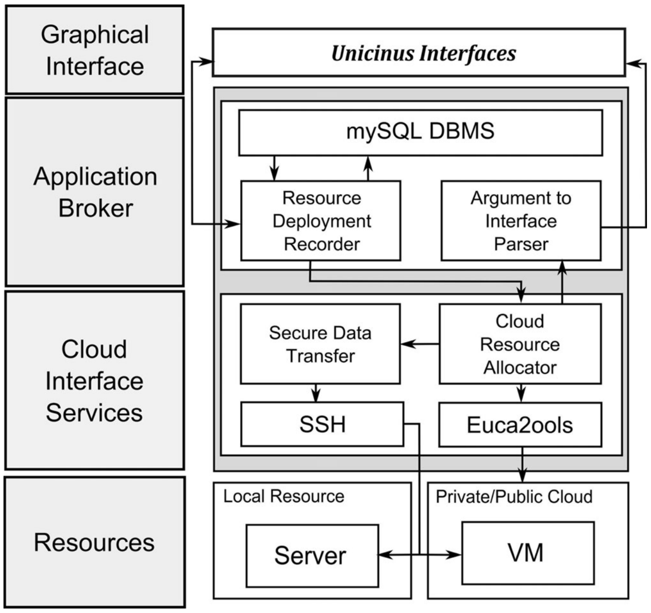


Fig. 11. Uncinus system overview.

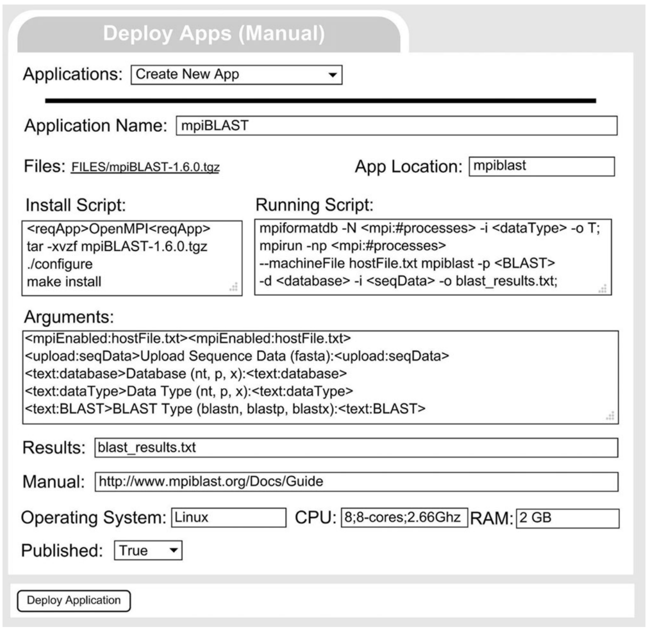
image, specify the required number and type of cloud resources. An HPC cluster consisting of eight Extra Large Cluster Compute instances was deployed. Each instance, a copy of the virtual machine image (mpiBLAST and OpenMPI already installed). Once the HPC cluster was made available to the user, mpiBLAST was executed through the web interface. Using this service a wallaby expressed sequence tag (EST) data-set was aligned against the NR (all organisms) database.

8.3 Uncinus

Uncinus is a web based environment that supports a modular approach to deploying applications on IaaS clouds. Services are provided to facilitate automated application deployment and job submission. Through an application broker, users can publish applications in the form of attributes that can be exposed as a service. Users accessing the application broker can also publish public cloud, private cloud and non-cloud resources through a single platform, allowing users to take advantage of a wider range of resources.

Users accessing Uncinus for the first time must create a user account. Through this account creation process, a user provides a unique username, password and cloud provider account details. For security reasons, cloud provider account details must be provided in the form of access and secret keys. Once logged in, users can access the components that make up Uncinus (mySQL DBMS, Application Broker, Cloud Interface Services and Euca2ools) through graphical interfaces, as shown in Fig. 11. Components are split into two layers; the Application Broker and the Cloud Interface Services.

The Application Broker allows an application service provider to publish Amazon Machine Images (AMIs) or Applications, where the AMI/App Deployment Recorder stores this data in a mySQL database on behalf of the application service provider. The Argument to Interface Parser service translates the input arguments recorded by the broker into equivalent web controls allowing for dynamic



Applications: Create New App

Application Name: mpiBLAST

Files: FILES/mpiBLAST-1.6.0.tgz App Location: mpiblast

Install Script:

```
<reqApp>OpenMPI<reqApp>
tar -xvzf mpiBLAST-1.6.0.tgz
./configure
make install
```

Running Script:

```
mpiformatdb -N <mpi:#processes> -i <dataType> -o T;
mpirun -np <mpi:#processes>
--machineFile hostFile.txt mpiblast -p <BLAST>
-d <database> -i <seqData> -o blast_results.txt;
```

Arguments:

```
<mpiEnabled:hostFile.txt><mpiEnabled:hostFile.txt>
<upload:seqData>Upload Sequence Data (fasta):<upload:seqData>
<text:database>Database (nt, p, x):<text:database>
<text:dataType>Data Type (nt, p, x):<text:dataType>
<text:BLAST>BLAST Type (blastn, blastp, blastx):<text:BLAST>
```

Results: blast_results.txt

Manual: http://www.mpiblast.org/Docs/Guide

Operating System: Linux CPU: 8;8-cores;2.66Ghz RAM: 2 GB

Published: True

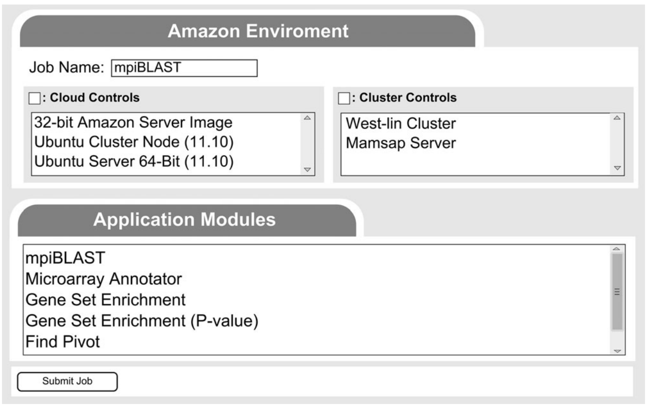
Fig. 12. Uncinus mpiBLAST Publication.

interface generation. Cloud Interface Services communicate with Amazon EC2 to provide cloud resources to the user. When a user starts a cloud job, the Cloud Resource Allocator module creates the necessary private keys and security groups before requesting resources from Amazon EC2. Once the Cloud Resource Allocator can successfully access the virtual machine, the Secure Data Transfer module is used to deploy application to the virtual machine instance.

Unlike other cloud solutions (see Sections 8.1 and 8.2), Uncinus approaches publication at the application level, allowing users with limited cloud experience to develop software services. To demonstrate this, a case study was carried out with the goal to expose and deploy mpiBLAST as a service. Fig. 12 presents the attributes required to deploy mpiBLAST as a service.

Publication of this mpiBLAST service began by assigning the service a name (Application Name). Next a compressed file containing the mpiBLAST source code was uploaded to the broker (Files). An installation script (Install Script) was provided that specified an openMPI service be loaded as a prerequisite before compiling mpiBLAST. The service's interface (Arguments) is specified by defining the type of data required. For this service the specification includes a control to upload sequence data and three text controls that determine the type of database and BLAST to be run. The commands used to execute mpiBLAST was specified (Running script) in the form as a shell script; this script formats the selected database and executes mpiBLAST over the selected number of processes, and the output of the mpiBLAST service is a file called blast_results.txt. The blast_results.txt is published through the deployment interface as the output location (Result). Lastly, requirements for running mpiBLAST are provided; Linux (Operating System), a cluster with eight nodes each with eight core running at 2.66 GHz (CPU) and 2 GB of RAM per node (RAM).

The published mpiBLAST service was deployed using the interface shown in Fig. 13. Users selecting from available computational resources (cloud and non-cloud) and a list of



Job Name: mpiBLAST

☐: Cloud Controls
32-bit Amazon Server Image
Ubuntu Cluster Node (11.10)
Ubuntu Server 64-Bit (11.10)

☐: Cluster Controls
West-lin Cluster
Mamsap Server

mpiBLAST
Microarray Annotator
Gene Set Enrichment
Gene Set Enrichment (P-value)
Find Pivot

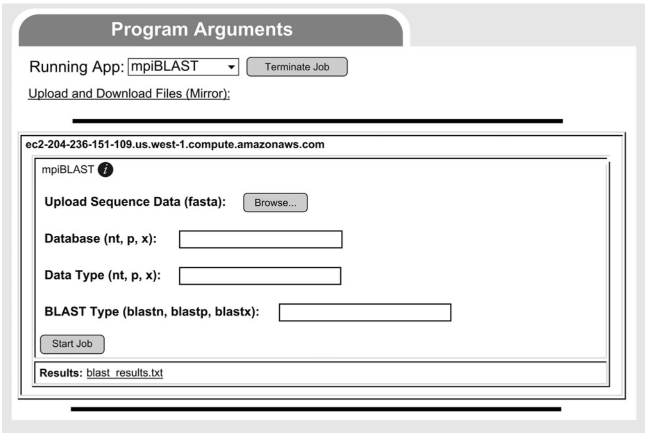
Fig. 13. Uncinus service development interface.

published services. To duplicate the HPCaaS case study, we select the mpiBLAST service and eight Cluster Compute Instances (running Ubuntu 11.10). When this job is submitted, Uncinus carries out automated resource selection to identify how to deploy services on the selected resources. This resource selection process calculates a distance metric based on the service requirements and resource specifications. For this case study, openMPI and mpiBLAST are deployed on Cluster Computer Instances.

Once the mpiBLAST service was deployed, a web interface is automatically generated using the published attributes. The mpiBLAST service interface presented to the user is shown in Fig. 14. Through this interface, users can: specify mpiBLAST argument, upload sequence data, execute mpiBLAST over the eight cluster compute instances, download results, and terminate the job (freeing all cloud resources). Also provided through this interface is a tool for transferring files to and from the cloud; uploaded files are mirrored across each node in the cluster.

8.4 Summary

This section presented three SaaS cloud solutions aimed at simplifying the deployment and usage of cloud resources executing HPC applications exposed as services. Common to all three solutions is the use a broker to support discovery and publication of services. However, these SaaS clouds differ in the features provided to support the development of services.



Running App: mpiBLAST

Upload and Download Files (Mirror):

ec2-204-236-151-109.us.west-1.compute.amazonaws.com

mpiBLAST

Upload Sequence Data (fasta):

Database (nt, p, x):

Data Type (nt, p, x):

BLAST Type (blastn, blastp, blastx):

Results: blast_results.txt

Fig. 14. Uncinus mpiBLAST service interface.

- HPCynergy allows users to publish computational resources (that have been assigned an IP address) that can be used to run software services. Resource selection is supported; a user specifies the resources required for the service they wish to run. This selection method is made more accurate through real time monitoring of resources; however this requires that each published resource runs Ganglia, thus adding to the complexity of utilizing HPCynergy. Unfortunately, there is no simple method for exposing software a service; addition of services requires the source code of the HPCynergy cloud to be modified.
- HPCaaS allows users to publish cloud resources (in the form of VM images) allowing for on-demand access to cloud resources. However, as this approach requires the development of VM images, it is only compatible with a single cloud provider at a time. As seen in the case study, there is no support for the development of VM's; instead they have to be developed through the cloud provider's interface. Furthermore, publication of these VM requires the modification of HPCaaS interface script. HPCaaS (via Galaxy) supports interface generation through the use of a XML based language, allowing users with minimal computing background to define web controls.
- Uncinus combines aspects of HPCynergy and HPCaaS, allowing users to publish cloud resources (in the form of VM's), non-cloud resources, and applications. Like HPCynergy, Uncinus publishes resources and applications as attributes and supports resource selection. Like HPCaaS; Uncinus allows user to build the interfaces that expose software as services using a XML based language. However, publication is carried out using web interfaces and doesn't require the modification of the Uncinus source code.

Through deployment of mpiBLAST on each SaaS cloud solution, we demonstrate how the development of cloud services can be simplified. Users with no cloud computing or HPC background, able to access distributed algorithms through easy to use web interfaces.

9 Conclusion

While originally developed for business, cloud and service computing can be used for research. Cloud resources can enhance local resources, reducing the turn-around time of analysis and/or allowing for bigger problems to be solved. However clouds do not fit all applications; depending on the research being carried out, HPC hardware may be required. Services built on top of HPC cloud solutions could offer researchers' access to HPC on demand without the need to understand and carry out complex deployment tasks. By combining service and cloud computing technologies, we devised solutions to problems in the area of mammalian genomics.

The outcome of the presented survey lies in the survey of three SaaS cloud solutions that draw upon concepts from SaaS clouds and bioinformatics cloud software, to solve known cloud usability issues. Through this successful integration of cloud and service computing, the analysis, interpretation and computation of genomic mammalian data was made easier, even to be carried out by non-computing discipline specialists, and cheaper.

Acknowledgments

The authors would like to express their gratitude to the reviewers for their insightful and constructive comments.

References

[1] A. Eggen, "The development and application of genomic selection as a new breeding paradigm," *Animal Frontiers*, vol. 2, pp. 10–15, Jan. 2012.

[2] D. Garrick, "The nature, scope and impact of genomic prediction in beef cattle in the United States," *Genetics Selection Evolution*, vol. 43, p. 17, 2011.

[3] M. West, G. S. Ginsburg, A. T. Huang, and J. R. Nevins, "Embracing the complexity of genomic data for personalized medicine," *Genome Res.*, vol. 16, pp. 559–566, May 2006.

[4] 23andMe. (2013). *23andMe - Genetic Testing for Health, Disease & Ancestry; DNA Test* [Online]. Available: https://www.23andme.com/

[5] D. Edwards, "Statistical analysis of gene expression microarray data," *Biometrics*, vol. 60, pp. 287–289, 2004.

[6] G. K. Smyth and T. Speed, "Normalization of cDNA microarray data," *Methods*, vol. 31, pp. 265–273, 2003.

[7] R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis, U. Scherf, and T. P. Speed, "Exploration, normalization, and summaries of high density oligonucleotide array probe level data," *Biostat*, vol. 4, pp. 249–264, Apr. 2003.

[8] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Nat. Acad. Sci. United States Am.*, vol. 95, pp. 14863–14868, 1998.

[9] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov, "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles," *Proc. Natl. Acad. Sci. United States Am.*, vol. 102, pp. 15545–15550, Oct. 2005.

[10] NCBI (2013). *FASTA format*. [Online]. Available: http://www.ncbi.nlm.nih.gov/BLAST/blastcgihelp.shtml

[11] GenomeWeb (2010). *Move Over, GA: Illumina's HiSeq 2000 Increases Output to 200 Gb Per Run* [Online]. Available: http://www.genomeweb.com/sequencing/move-over-ga-illuminas-hiseq2000-increases-output-200-gb-run

[12] P. Church, A. Goscinski, K. Holt, M. Inouye, A. Ghoting, K. Makarychev, and M. Reumann, "Design of Multiple Sequence Alignment Algorithms on Parallel, Distributed Memory Supercomputers," *33rd Annu. IEEE Eng. Med. Biol. Soc.*, Boston, MA, USA, 2011.

[13] The 1000 Genomes Project Consortium, "A map of human genome variation from population-scale sequencing," *Nature*, vol. 467, pp. 1061–1073, 2010.

[14] U.S. Department of Veterans Affairs (2013, Mar.). *Million Veterns Project*. [Online]. Available: http://www.research.va.gov/mvp/

[15] J. J. D. David and W. Walker, "MPI: A standard message passing interface," *Supercomputer*, vol. 12, pp. 56–68, 1996.

[16] W. Lang, J. M. Patel, and S. Shankar, "Wimpy node clusters: What about non-wimpy workloads?" presented at the Sixth Int. Workshop Data Manag. New Hardware, Indianapolis, IN, USA, 2010.

[17] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets," *J. Netw. Comput. Appl.*, vol. 23, pp. 187–200, 2000.

[18] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *Int. J. High Perform. Comput. Appl.*, vol. 15, pp. 200–222, Aug. 2001.

[19] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *Int. J. High Perform. Comput. Appl.*, vol. 11, pp. 115–128, Jun. 1997.

[20] R. V. Buyya and S. Venugopal, "The Gridbus toolkit for service oriented grid and utility computing: An overview and status report," in *Proc. Grid Econ. Business Models*, 2004, pp. 19–66.

[21] D. Fenstermacher, C. Street, T. McSherry, V. Nayak, C. Overby, and M. Feldman, "The cancer biomedical informatics grid (caBIG^{TM})," in *Proc. IEEE 27th Annu. Int. Conf.Eng. Med. Biol. Soc.*, 2005, pp. 743–746.

[22] J. Saltz, S. Oster, S. Hastings, S. Langella, T. Kurc, W. Sanchez, M. Kher, A. Manisundaram, K. Shanbhag, and P. Covitz, “caGrid: Design and implementation of the core architecture of the cancer biomedical informatics grid,” *Bioinformatics*, vol. 22, pp. 1910–1916, Aug. 2006.
[23] J. Phillips, R. Chilukuri, G. Fragoso, D. Warzel, and P. Covitz, “The caCORE software development kit: Streamlining construction of interoperable biomedical information services,” *BMC Med. Inf. Decision Making*, vol. 6, p. 2, 2006.
[24] P. Mell and T. Grance, “The NIST definition of cloud computing,” National Institute of Standards and Technology, Gaithersburg, MD, United States, 2009.
[25] Amazon, “Amazon Elastic compute cloud: Getting started guide,” *Amazon*, 2010.
[26] OpenStack Project. (2012). *OpenStack - Open source software for building private and public clouds* [Online]. Available: http://www.openstack.org/
[27] K. Gibbs. (2008, Apr.). *Google App Engine Campfire One Transcript* [Online]. Available: http://code.google.com/appengine/articles/cf1-text.html
[28] T. Redkar, “SQL Azure,” in *Windows Azure Platform*, ed Berkely, CA, USA: Apress, 2009, pp. 505–584.
[29] R. Mullin, “LIMS in the cloud,” *C&EN*, vol. 88, p. 4, 2010.
[30] FinancialForce, “Closing the books on traditional accounting,” FinancialForce, Ed., ed. San Francisco, CA: FinancialForce.com, inc, 2008.
[31] B. Claybrook, M. Eisenberg, R. Jennings, D. Linthicum, T. Nolle, and D. Sullivan (2013). *The cloud computing market forecast for 2013: Experts predict what to expect* [Online]. Available: http://searchcloudcomputing.techtarget.com/feature/The-cloud-computing-market-forecast-for-2013-Experts-predict-what-to-expect
[32] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” in *Proc. 19th ACM Symp. Operat. Syst. Principles*, Bolton Landing, NY, USA, 2003.
[33] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, “The eucalyptus open-source cloud-computing system,” in *Proc. 9th IEEE/ACM Int. Symp. Cluster Comput. Grid*, 2009, pp. 124–131.
[34] D. Chappel, “Introducing windows azure,” *David Chapple and Associates*, Dec. 2009.
[35] SalesForce. (2013). *Salesforce CRM (Customer Relationship Management) Softw. Business* [Online]. Available: http://www.salesforcefoundation.org/products
[36] I. CloudSwitch. (2013). *Enterprise Cloud Computing | Cloud Switch* [Online]. Available: http://www.cloudswitch.com/
[37] Oracle. (2013). *Oracle CRM On Demand | Oracle* [Online]. Available: http://www.oracle.com/us/products/applications/crmondemand/index.html
[38] J. Goecks, A. Nekrutenko, J. Taylor, and T. G. Team, “Galaxy: S comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences,” *Genome Biol.*, vol. 11, p. R86, 2010.
[39] AGAVE. (2012, Dec.). *AGAVE* [Online]. Available: http://sourceforge.net/projects/agaveapi
[40] H.-L. Truong and S. Dustdar, “Cloud computing for small research groups in computational science and engineering: current status and outlook,” *Computing*, vol. 91, pp. 75–91, 2011.
[41] K. Yelick, S. Coghlan, B. Draney, and R. S. Canon, “The Magellan Report on Cloud Computing for Science,” U.S. Department of Energy, Office of Sci., Office Adv. Scientific Comput. Res., Dec. 2011.
[42] A. Goscinski and M. Hobbs, “Experiences Gained from Building a Services-Based Distributed Operating System,” in *Proc. 10th Int. Conf. Algorithms Architect. Parallel Process.,* 2010, vol. 6082, pp. 225–234.
[43] B. Rimal, A. Jukan, D. Katsaros, and Y. Goeleven, “Architectural requirements for cloud computing systems: An enterprise cloud approach,” *J. Grid Comput.*, vol. 9, pp. 3–26, 2011.
[44] J. Napper and P. Bientinesi, “Can cloud computing reach the top500?,” presented at the Combined Workshop UnConventional High Perform. Comput. Workshop Plus Memory Access Workshop, Ischia, Italy, 2009.
[45] Top500. (2011, Jun.). *Amazon EC2 Cluster instances - TOP500* [Online]. Available: http://www.top500.org/system/177457
[46] W. Guohui and T. S. E. Ng, “The impact of virtualization on network performance of amazon EC2 data center,” in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[47] D. R. Avresky, M. Diaz, A. Bode, B. Ciciani, E. Dekel, J. Ekanayake, and G. Fox, “High performance parallel computing with clouds and cloud technologies,” in *Proc. Int. Conf. Cloud Comput.*, O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, M. Gerla, H. Kobayashi, S. Palazzo, S. Sahni, X. Shen, M. Stan, J. Xiaohua, A. Zomaya, and G. Coulson, Eds., ed: 2010, vol. 34, pp. 20–38.
[48] R. R. Expósito, G. L. Taboada, S. Ramos, J. Touriño, and R. Doallo, “Performance analysis of HPC applications in the cloud,” *Future Generation Comput. Syst.*, vol. 29, pp. 218–229, 2013.
[49] A. K. L. Wong and A. M. Goscinski, “A unified framework for the deployment, exposure and access of HPC applications as services in clouds,” *Future Generation Comput. Syst.*, vol. 29, pp. 1333–1344, 2013.
[50] A. Darling, L. Carey, and W. Feng, “The design, implementation, and evaluation of mpiBLAST,” *the 4th International Conference on Linux Clusters: The HPC Revolution 2003 in conjunction with the ClusterWorld Conference & Expo*, San Jose, CA., Jun. 2003.
[51] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kalé, and K. Schulten, “Scalable molecular dynamics with NAMD,” *J. Comput. Chemistry*, vol. 26, pp. 1781–1802, 2005.
[52] B. Liu, B. Sotomayor, R. Madduri, K. Chard, and I. Foster, “Deploying bioinformatics workflows on clouds with galaxy and globus provision,” presented at the 3rd Int. Workshop Data Intensive Comput. Clouds, Salt Lake City, UT, USA, 2012.
[53] B. Langmead, M. Schatz, J. Lin, M. Pop, and S. Salzberg, “Searching for SNPs with cloud computing,” *Genome Biol.*, vol. 10, p. R134, 2009.
[54] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” presented at the 6th Conf. Symp. Opear. Syst. Design Implementation, vol. 6, San Francisco, CA, USA, 2004.
[55] B. Langmead, K. Hansen, and J. Leek, “Cloud-scale RNA-sequencing differential expression analysis with Myrna,” *Genome Biol.*, vol. 11, p. R83, 2010.
[56] K. Krampis, T. Booth, B. Chapman, B. Tiwari, M. Bicak, D. Field, and K. Nelson, “Cloud BioLinux: pre-configured and on-demand bioinformatics computing for the genomics community,” *BMC Bioinformat.*, vol. 13, p. 42, 2012.
[57] A. Goscinski, M. Brock, and P. Church, *High Performance Computing Clouds*. New York, NY, USA: Taylor & Francis, Jun. 2011.
[58] M. Brock and A. Goscinski, “Execution of compute intensive applications on hybrid clouds,” presented at the 1st Int. Workshop Hybrid/Cloud Comput. Infrastructures e-Sci. Appl., Palermo, Italy, 2012.
[59] VMware. (2010). *Vmware vsphere* [Online]. Available: http://www.vmware.com/products/vsphere/mid-size-and-enterprise-business/features.html
[60] T. Kay. (2012). *Simple Command-Line Access to Amazon EC2.* [Online]. Available: http://aws.amazon.com/developertools/739
[61] A. Wong and A. Goscinski, “A VMD plugin for NAMD simulations on Amazon EC2,” *Procedia Comput. Sci.*, vol. 9, pp. 136–145, 2012.

Philip C. Church received the PhD degree from Deakin University, Geelong, VIC, Australia, under the supervision of Dr. Andrzej Goscinski and Dr Christophe Lefevre. He is currently carrying out research in the area of cloud and service computing supporting bioinformatics. He is a student member of the IEEE.

Andrzej M. Goscinski is currently a chair professor of computing at Deakin University, Geelong, VIC, Australia. He is recognized as one of the leading researchers in clouds and cloud computing, parallel processing, virtualization, security, autonomic and service computing. The results of his research have been published in high-quality journals and conference proceedings. He is a member of the IEEE.