PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A

rsta.royalsocietypublishing.org

Research

Cite this article: Cohen J, Filippis I, Woodbridge M, Bauer D, Chue Hong N, Jackson M, Butcher S, Colling D, Darlington J, Fuchs B, Harvey M. 2013 RAPPORT: running scientific high-performance computing applications on the cloud. Phil Trans R Soc A 371: 20120073. http://dx.doi.org/10.1098/rsta.2012.0073

One contribution of 13 to a Theme Issue 'e-Science–towards the cloud: infrastructures, applications and research'.

Subject Areas:
cloud computing, e-science

Keywords:
cloud computing, high-performance computing, data-intensive research, infrastructure as a service

Author for correspondence:
Jeremy Cohen
e-mail: jeremy.cohen@imperial.ac.uk

RAPPORT: running scientific high-performance computing applications on the cloud

Jeremy Cohen[1], Ioannis Filippis[2], Mark Woodbridge[2], Daniela Bauer[3], Neil Chue Hong[5], Mike Jackson[5], Sarah Butcher[2], David Colling[3], John Darlington[1], Brian Fuchs[1] and Matt Harvey[4]

[1]Department of Computing, [2]Bioinformatics Support Service, CISBIO, [3]Department of Physics, and [4]Imperial College HPC Service, Imperial College London, South Kensington Campus, London SW7 2AZ, UK

[5]Software Sustainability Institute, University of Edinburgh, James Clerk Maxwell Building, Mayfield Road, Edinburgh EH9 3JZ, UK

Cloud computing infrastructure is now widely used in many domains, but one area where there has been more limited adoption is research computing, in particular for running scientific high-performance computing (HPC) software. The Robust Application Porting for HPC in the Cloud (RAPPORT) project took advantage of existing links between computing researchers and application scientists in the fields of bioinformatics, high-energy physics (HEP) and digital humanities, to investigate running a set of scientific HPC applications from these domains on cloud infrastructure. In this paper, we focus on the bioinformatics and HEP domains, describing the applications and target cloud platforms. We conclude that, while there are many factors that need consideration, there is no fundamental impediment to the use of cloud infrastructure for running many types of HPC applications and, in some cases, there is potential for researchers to benefit significantly from the flexibility offered by cloud platforms.

1. Introduction

Cloud computing infrastructure platforms provide the opportunity to gain flexible, on-demand, access to remote

storage, processing power and other hardware facilities. Within an academic research environment, where significant computing resource capacity is often freely accessible at the point of use to researchers with high-performance computing (HPC) requirements, the headline benefits of cloud computing may seem of limited relevance. However, through collaborations between the London e-Science Centre (LeSC) and scientists across a range of fields at Imperial College London, it became clear that there was a general interest in how cloud computing may be of use to researchers who make extensive use of computing power as part of their work. While professionally operated HPC services provide valuable user support and access to a range of software, there may be situations where a traditional batch-style interface or simply the number of resources available when undertaking a time-critical piece of work are limiting.

The interest in cloud computing among researchers at Imperial College London led to the forming of a collaboration and the funding of the Robust Application Porting for HPC in the Cloud (RAPPORT) project [1] under the JISC/EPSRC Pilot Projects in Cloud Computing for Research call. The project collaboration included researchers from four disciplines—computing, physics, bioinformatics and digital humanities—working with researchers from the Software Sustainability Institute, University of Edinburgh and Imperial College London's HPC Service.

RAPPORT focused specifically on cloud infrastructure or infrastructure-as-a-service (IaaS) platforms rather than those categorized as platform-as-a-service (PaaS) or software-as-a-service (SaaS). IaaS platforms, at their most basic, provide computing resources on demand that users can access directly and configure as they wish. Users can scale their resource usage on demand. In a large IaaS cloud, the limiting factor is likely to be the scalability of the user's application. Some IaaS platforms provide higher-level services, for example, to simplify scaling of particular types of application to multiple resources, but these were not investigated as part of this work. IaaS was selected because it was considered to best suit the profiles and requirements of the HPC applications selected for this work. At the start of the work it was decided not to restrict investigations to one specific model of infrastructure cloud, and the project team therefore had access to cloud platforms covering private, community and public cloud models—these are described in §2.

The 6-month RAPPORT project produced two reports [2,3] that were submitted to JISC [4], and two best-practice guides, one targeted at researchers [5] and the other targeted at funders [6]. This paper, based on the content of these outputs, summarizes the work undertaken, and discusses the findings and conclusions. Section 2 describes the different cloud platforms that were available to RAPPORT with a focus on the private cloud platforms deployed as part of the project. Section 3 then describes the HEP and bioinformatics applications and outlines why there was interest in running them on cloud resources. Section 4 describes the work done to analyse the application properties and the progress made in running them on cloud resources. In the case of the bioinformatics applications, various performance tests were undertaken and these are described in §4*b*(i). Details of the best-practice guides developed by RAPPORT are provided in §5, with conclusions covered in §6.

2. Cloud infrastructure platforms

Three different models of infrastructure cloud platform—private, community and public—were considered as part of this work. These IaaS platforms provide a service interface through which registered users can provision or access processing, storage and, potentially, other infrastructure resources. The IaaS interface provides an abstraction layer above the physical hardware, hiding the underlying platform structure. Where computational grids present challenges in managing authentication in a virtual organization across resources that are owned by different physical entities and offer more traditional job submission interfaces, cloud virtual instances appear almost identical to other remote servers to which users may have access, with the added ability to configure the instances and install software as required. Resources are provisioned based on

machine images, disk images containing a complete operating system and software configuration. Users can bundle their complete application into a disk image along with the operating system or start cloud instances based on a clean operating system install and use scripting to deploy and configure their software once the resources are running.

The following subsections provide an overview of each of the three cloud models considered and detail the services made available to the researchers.

(a) Private cloud

Private cloud platforms use locally owned and hosted resources made available to a closed group of individuals, for example, a research group or centre. They are suitable in a range of environments such as where data volumes are too large for external transfer, where data security issues exist or where sufficient computing resource capacity is available locally but improved usage accounting and more flexible access for sub-groups of users is desired. Two private cloud platforms were available to RAPPORT researchers. The first was deployed at LeSC prior to the start of the project, while the second platform was deployed as part of the RAPPORT work. Private cloud was considered important because of potential issues with transferring large quantities of data to remote sites for some of the pilot applications.

The EUCALYPTUS open-source cloud platform [7,8] was the cloud software used for both private cloud deployments. It was installed as part of UBUNTU LINUX using the UBUNTU ENTERPRISE CLOUD (UEC) platform [9]. The earlier platform ran on older cluster resources that lacked hardware virtualization support and used XEN [10] as the virtualization layer. This required custom configuration, since KVM [11], which requires hardware virtualization support, is the default virtualization layer for UEC/EUCALYPTUS. A kernel was built for UBUNTU with XEN dom0 support and a deployment infrastructure was configured using kickstart to automate the installation of multiple cloud nodes. The private cloud platforms described here have constraints on the number of cloud virtual instances that can be operated because of the limited number of underlying hardware resources. However, the interface provided by EUCALYPTUS is equally applicable to much larger private cloud deployments. The specification of the initial LeSC cloud deployment, as described in [3], is shown below.

— Head node: one U server, single 8-core AMD Opteron 6136 CPU, 16 GB RAM and dual 146 GB 15 k RPM SAS disks.
Software: UBUNTU v. 10.04, EUCALYPTUS v. 1.6 Cloud Controller, Cluster Controller, Storage Controller and Walrus services. Fibre channel link to LeSC SAN for Storage Controller/Walrus storage space.
— Compute nodes: 55 servers each with dual 1.8 GHz Opteron 244 CPUs, 2 GB RAM and 80 GB SATA disks.
Software: UBUNTU v. 10.04, XEN v. 4, EUCALYPTUS v. 1.6 Node Controller software.

The updated RAPPORT cloud deployment used improved hardware and EUCALYPTUS v. 2.0. The presence of hardware virtualization on the CPUs enabled the use of KVM, simplifying the configuration and automated deployment process. The specification of the 'RAPPORT Cloud', as described in [3], is shown below.

— Head node: one server with dual-core Intel Xeon 5130 2 GHz CPUs, 6 GB RAM, 160 GB SATA hard disk.
Software: UBUNTU v. 10.10, EUCALYPTUS v. 2.0 Cloud Controller, Cluster Controller, Storage Controller and Walrus services. NFS mounted disk space from storage server.
— Compute nodes: 19 servers each with two dual-core Intel Xeon 5130 2 GHz CPUs, 6 GB RAM, 160 GB SATA hard disk.
Software: UBUNTU v. 10.10, KVM, EUCALYPTUS v. 2.0 Node Controller.
— Storage node: one storage server, quad-core CPU, 32 × 2 TB SATA hard disks.
Software: UBUNTU v. 10.10, NFS v. 4 server, disk space exported to head node.

(b) Community cloud

Community clouds extend the private cloud model to a wider community, enabling closed, but potentially large, widely distributed communities, for example, UK Research Council-funded researchers, to share cloud resources. They may take advantage of community infrastructure, such as the high-speed JANET [12] network linking UK academic institutions, in order to provide improved access to remotely located resources. An example of a community cloud resource is the UK National Grid Service (NGS) Cloud Prototype [13], which was set up as part of the NGS to provide a prototype infrastructure cloud service based on resources located at two different UK academic sites. The ability to transfer data to and from remote cloud resources via the JANET network offered the potential to undertake some data-intensive tasks that would have previously been impracticable on remote resources. The cloud prototype was available until October 2011 and some members of the project team undertaking this work acquired access to the NGS Cloud Prototype.

(c) Public cloud

For many, public clouds are likely to provide the most easily accessible option for using cloud infrastructure. A number of commercial public cloud services are available, where users register, provide payment details and can then use different hardware resources, accessed via a service interface, paying for the amount of resource capacity they use. Examples include Amazon EC2 [14], Rackspace [15] and Joyent [16]. Other services, such as those provided by VMware [17], support the development and operation of a public cloud service. If private or community cloud services are not available, signing up to use a public cloud service is a straightforward option. Users generally have access to significant capacity, a range of hardware specifications, and can scale their requirements up and down at short notice. In some cases, resource usage can be changed within minutes. While providers do not necessarily provide explicit support for HPC software, the cloud virtual instances are equivalent to standard servers, and powerful systems with fast interconnects that are suited to HPC tasks are often available. Computing resource capacity is paid for by usage and larger resources are generally more expensive. If processing time is not a critical issue, it may be cost-effective to use cheaper resources with less processing power, resulting in a longer runtime but lower overall cost.

Public cloud platforms are flexible and easily accessible but some users may have concerns about data security, data transfer speeds or the cost of accessing the required resource capacity, data transfer bandwidth and associated storage over the long term. This is particularly true in the case of HPC jobs, where data volumes may be very large, jobs may run for a significant period of time, and, in an academic environment, local HPC resources may already be available. However, public cloud platforms offer the opportunity to gain access to larger numbers of resources than may be locally available and, if there is a need to complete work within a short time period, the ability to gain easy access to large numbers of additional resources may be highly desirable. Various studies [18–20] have indicated the potential of public utility computing for running HPC applications.

For this work, Amazon EC2 was selected as the public cloud platform of choice. This was because of the previous experience that members of the project team had had with this platform and the limited time available to undertake the work.

3. Application domains

Applications from three different domains were investigated—HEP, bioinformatics and digital humanities. The chosen domains provided candidate applications that represent a range of different HPC profiles, such as high-throughput, close-coupled parallel and CPU/data-intensive. In this paper, we focus on the HEP and bioinformatics applications and the analysis carried out on these applications.

(a) High-energy physics

The HEP group is involved with the compact muon solenoid (CMS) [21] experiment at the Large Hadron Collider (LHC) [22] at CERN and investigated applications related to the analysis of CMS data. The CMS detector records the interactions of particles produced in proton–proton collisions. The data are typically presented as 'events', which contain the detector readout for one independent set of collisions. The group worked with two software processes, the analysis of CMS collision event data and the Monte Carlo generation of simulated data that is required to support the data analysis process.

HEP applications are characterized by their need to deal with extremely large amounts of data. A typical HEP computing 'cluster' will host around 0.5–1 PB of data at any given time. A typical analysis will focus on 15–50 TB of data at a time making data locality significant. Output data from analysis jobs are generally an order of magnitude smaller than the input datasets. An exception to this is Monte Carlo generation, which, while requiring only a few small configuration files as input, generates large quantities of data that simulate the CMS detector response. The output, which is comparable in size to the data produced by the actual collisions, is then stored for further analysis. Owing to the division of the data into quasi-independent events, HEP computing is highly parallelizable and, in general, makes efficient use of CPU resources, with usage approximately 80–90%. Aside from a small number of applications, HEP software in a research environment is custom-written, and CMS software [23] follows this general pattern.

The majority of HEP computing is currently undertaken on the Worldwide LHC Computing Grid (WLCG) [24], using local batch systems to supplement resources. While the ability to scale resources using on-demand cloud infrastructure could be beneficial, the additional nodes would need to be able to interface with the existing grid infrastructure. Users are typically familiar with this environment, but various wrappers also exist to hide some of the complexity from the end users. Potential gains could be made at times of high demand, typically seen on the release of a new dataset, and from the ability to gain easy access to additional systems providing the SCIENTIFIC LINUX v. 5 (SL5) operating system [25] used by CMS software.

(b) Bioinformatics

(i) Phylogenetics

Phylogenetics is a field of study in evolutionary biology that seeks to establish ancestral relationships between groups of organisms. A phylogenetic tree, representing these relationships, can be inferred by exploration of the search space of possible trees. Software tools implementing these algorithms use a mathematical model describing the evolution of characteristics of the studied organisms to generate hypotheses to which a likelihood score can be heuristically assigned.

Two widely used, open-source, computational phylogenetics tools, MRBAYES [26–28] and RAxML [29], were used for the purposes of this analysis. They differ in implementation strategy, resource usage and approaches to parallelism. Unlike many other bioinformatics applications, these tools do not typically require or generate large datasets and can easily be run on remote resources. They accept different input data formats, but these are interconvertible, so the same dataset can be used for both tools. In our case, this was a set of DNA and RNA alignments provided by the author of RAxML and previously used to generate published benchmarks for the tools in question [30].

MRBAYES is single-threaded code that can be duplicated and distributed across multiple nodes using MPI, reducing runtime approximately linearly up to a limit of the number of chains and replicates chosen for analysis. A forked version of MRBAYES that adds threading (via OpenMP) to extend parallelism is also available from the author of RAxML. MRBAYES uses the NEXUS input format.

RAxML is more actively developed than MrBayes and is designed to infer phylogenetic trees with greater efficiency than more mature tools. RAxML has historically relied on fine-grained parallelism (via Pthreads) [31] but has been enhanced to support MPI to expedite bootstrapping, the first of three phases performed during analysis. This aims to reduce the possible search space to improve performance. RAxML uses the PHYLIP input format.

(ii) Gene structure prediction

Currently, genomics and transcriptomics data are being generated at an unprecedented pace. The major bottleneck in harnessing the biological information encoded in the avalanche of next-generation sequencing data is the availability of scalable computational infrastructure and advanced algorithmic techniques. Both infrastructure and analysis methods need to keep up with the rapid advances in sequencing technology and allow for tailor-made solutions that match the needs of individual research groups. Inevitably, cloud computing has gained an increasingly prominent role for storage and analysis of sequencing data generated in an ad hoc manner [32–34].

High-throughput RNA sequencing (RNA-seq) is by far the most complex and most powerful next-generation sequencing application [35,36]. RNA-seq generates millions of short sequence fragments of cDNA (often termed reads) that can be used in a range of analyses, including structurally annotating genomes and identifying alternative splicing isoforms.

The role of the cloud in the next-generation sequencing field, the overall importance of RNA-seq analyses and the challenge for accurate spliced alignment of long high-throughput transcripts for large genomes led to the investigation of running GenomeThreader [37] in a cloud environment. GenomeThreader is a mature, closed-source software tool available in the form of binaries. On the basis of a dynamic programming algorithm, it predicts gene structures by spliced alignment of cDNA/EST and protein sequences against a reference genome. GenomeThreader is available free of charge under a license agreement for non-commercial research institutes.

GenomeThreader can be extremely memory and runtime intensive, especially for very large long transcripts and large genomes. The sizes of the reference genome, of the RNA-seq data and of the aligned reads can impose significant memory requirements, each one of a different magnitude. Spliced alignment can be performed in parallel by splitting the input data, performing separate alignments and merging the resulting output [38]. The splitting strategy should match the characteristics of the available datasets. However, since it is not possible to know, in advance, the size of the matched regions, providing an estimate of the requirements is challenging.

4. Analysis and findings

(a) High-energy physics

Two examples of software that a researcher at CMS might encounter were considered: firstly, an analysis of a specific dataset looking for J/ψ particles and, secondly, the generation of Monte Carlo data for a physics process that produces J/ψ particles. Both processes were tested locally to ensure that the applications were working correctly. Testing was then undertaken using the existing grid infrastructure and finally using cloud resources. Commercial clouds were ruled out owing to the time and cost of transferring and storing the large amount of data involved. It was decided to use local private cloud resources, which provided the advantage that it was relatively straightforward to integrate the resources with existing frameworks.

CERN provides a virtual machine image, CernVM [39], for a number of different virtualization platforms. In principle, CernVM can be used to run all standard experimental software for the LHC experiments and also provides basic grid functionality such as an interface to access grid-enabled storage elements. However, the deployment of experiment-specific software on this platform is left to the individual experiments. The CMS experiment currently does not actively encourage deployment on cloud platforms and instead leaves the decision up to

individual resource providers. This meant that the deployment of the CMS software required some manual adjustment of its configuration to support use on cloud-based virtual resources. The required changes were largely undocumented and required some 'behind the scenes' knowledge. Deployment of the CernVM image itself on the EUCALYPTUS-based private cloud platform was also not entirely straightforward. This was a result of the time taken for the relatively large disk image to be made available on the cloud resources. Their slow performance, particularly disk performance, resulted in timeouts during the virtual instance deployment. There were also driver issues and a lack of documentation owing to EUCALYPTUS not being one of CernVM's supported platforms. Chasing up individual problems on mailing lists and through personal contacts took time but the deployment was eventually completed successfully.

Once CernVM was deployed as a virtual instance on cloud resources, linking and access to the local storage element was tested and the two test applications were deployed and configured. Basic tests were performed to ensure functionality and, while no structured performance analysis was carried out, no significant decrease in performance was observed in initial tests for the examples studied. For a large-scale deployment, further integration with the existing tools, e.g. the CMS Remote Analysis Builder (CRAB), which is currently used for the majority of job submissions at CMS, would be necessary.

(b) Bioinformatics

Deployment of the phylogenetics tools, requiring transfer of the source code to the desired platform(s) and compiling to produce an executable, was relatively straightforward in comparison to the other software examined in RAPPORT. The tools use standard libraries, are easily recompiled and have no runtime dependences. A comparative analysis was carried out between several runtime platforms, both cloud and non-cloud. The aim was to analyse how the use of different platforms affected the tools rather than to analyse the relative performance or results of MRBAYES and RAxML.

It was observed that performance on two of the test cloud platforms was superior to the existing dedicated resource for a range of inputs. However, some datasets could not be run in all of the target environments owing to insufficient system memory, and others did not benefit at all owing to the overhead in provisioning resources, particularly on the LeSC cloud, which could take several minutes due mainly to network bandwidth issues and the age of the compute nodes, which resulted in poor disk performance. This demonstrates a requirement for intelligence in distributing jobs to platforms that have varying cost and performance benefits. A simple graphical user interface was developed to allow MRBAYES jobs to be submitted to the RAPPORT cloud platform, allowing users familiar with MRBAYES to run jobs on cloud resources without needing an understanding of the EUCALYPTUS cloud platform.

For the gene structure prediction application, GENOMETHREADER, attempting to run the application in a cloud environment served multiple aims. It enabled practical investigation of issues that may be caused by deploying in a cloud environment and allowed a comparative analysis between running on cloud and non-cloud resources. It also allowed investigation of the feasibility of running GENOMETHREADER in parallel in the cloud using the APACHE HADOOP MAPREDUCE software framework [40] and HADOOP Distributed File System (HDFS) [41] for distributed data access. Standard deployment of the application was straightforward, with no significant differences between cloud and non-cloud deployment. For parallel deployment, time constraints allowed only small-scale testing using 'toy' datasets. Nonetheless, it is expected that the proposed approach could accommodate RNA-seq-based annotations of large genomes and be customized to the needs of individual projects.

(i) Performance analysis—phylogenetics

MRBAYES was used to assess the performance characteristics of various runtime platforms and to determine the extent to which the cloud can provide an alternative to existing environments

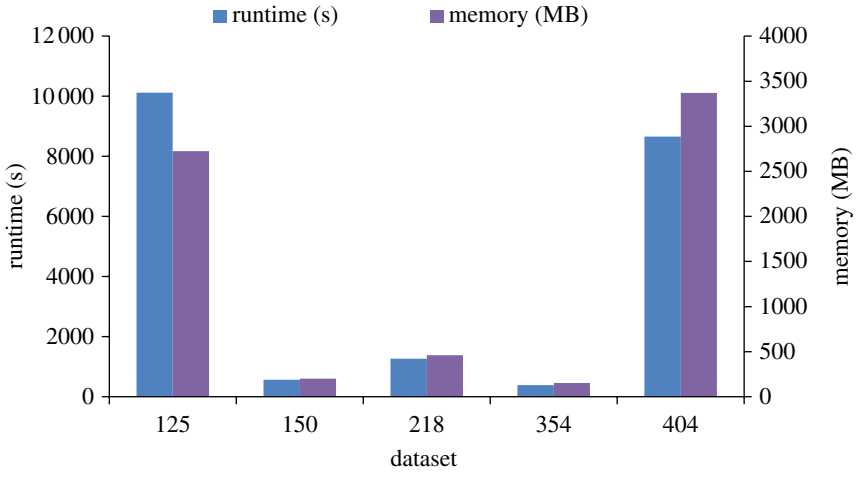

Figure 1. MrBayes resource usage for sequence alignments of varying taxa on the codon platform; from Cohen *et al.* [2]. (Online version in colour.)

used locally for phylogenetics analysis. The different platforms used for testing are now detailed, along with identifiers, shown in parentheses, used in the performance graphs: m1.large virtual instances with two CPU cores and 2 GB RAM on the original LeSC cloud (lesc) and m1.large virtual instances with two CPU cores and 2 GB RAM on the RAPPORT cloud (rapport); hardware equivalent to the resources used in the LeSC and RAPPORT clouds, but deployed as non-virtualized standalone resources (lesc-nv and rapport-nv, respectively); a standalone server used by the bioinformatics group, with 16 × 2.93 GHz CPU cores and 16 GB RAM running CentOS 5.5 x86_64 (codon); m2.4x large virtual instance on Amazon EC2 providing eight virtual cores of 3.25 EC2 compute units each, 68.4 GB RAM [42] and configured with Ubuntu 10.04 x86_64 (ec2); standard nodes on the UK National Grid Service grid (ngs).

The Intel C Compiler v. 12.0.4 was used to compile MrBayes v. 3.1.2 h and RAxML v. 7.2.8, which were dynamically linked to the OpenMP runtime libraries. Benchmarks were undertaken using a typical runtime configuration: two runs, each of four chains and 10 000 iterations. The number of iterations was chosen to minimize variance while managing overall runtime. There were not a sufficient number to achieve convergence for the datasets in question but enough iterations were undertaken to provide a much more representative indication of performance than the minimal primates.nex file frequently used in other benchmarks and in the MrBayes tutorial material [43].

A preliminary test was conducted using the 'codon' system, commonly used by phylogenetics researchers at Imperial College London, to select a suitable dataset as the basis for further analysis. The results showed that different input files have vastly different resource requirements and that a simplistic analysis of the files in terms of the number or length of aligned sequences did not indicate how long or how much memory the analysis requires (see figure 1). The 218 taxa dataset was selected for further benchmarks because of it being suitable for processing within the memory available on all test platforms.

Figure 2 shows the runtimes of a pair of MPI processes on a pair of remote, single-processor nodes or co-located on a dual-processor node, indicating the minor overhead of network communication on performance of the MrBayes application. The impact is less for virtualized hosts, possibly due to lower baseline performance in this case. The overhead resulting from virtualization is visible but is less significant on the rapport platform that has hardware with virtualization support. The figures presented are for runtime only and do not include the time taken to provision cloud nodes. This approach has been taken to enable a direct comparison between the computation time on cloud and non-cloud nodes. Since our focus is on supporting HPC applications where a significant amount of computation is involved, resource provisioning

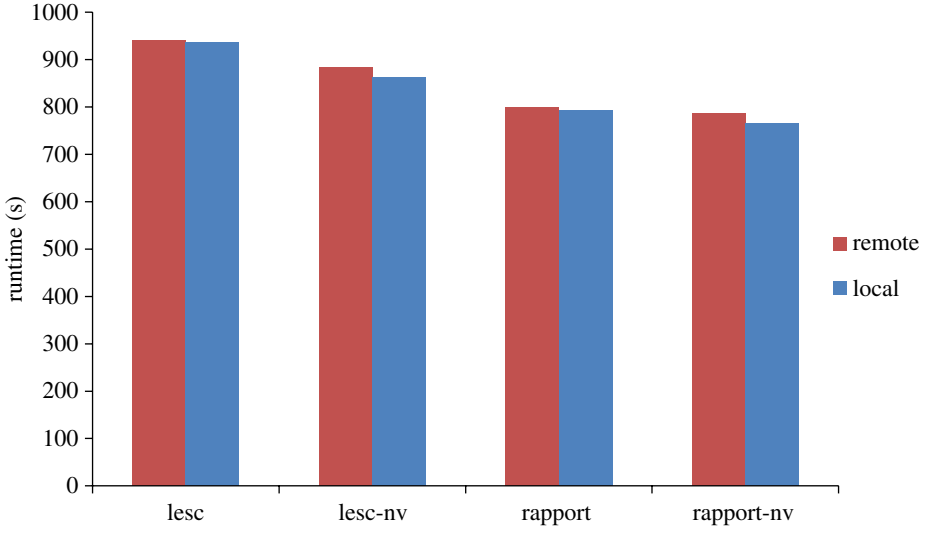

Figure 2. MrBayes dual-process performance for 218 taxa alignment on a pair of remote or co-located (local) nodes on various platforms; from Cohen *et al*. [2,3]. (Online version in colour.)

was not considered to represent a significant overhead. Limited tests of resource instantiation time for groups of eight cloud nodes of different sizes on the rapport private cloud platform, using cached images, showed times ranging from 55 to 110 s. We accept that other overheads, such as reduced data transfer rates to remote cloud platforms, can be an issue but recommend that for highly data-intensive tasks, local private cloud platforms are used.

Figure 3 shows the relative performance impact of increasing the number of threads and processes available to MrBayes. In the default configuration of two runs of four chains, each chain can be processed on a separate physical node with communication via MPI. Each chain can also be processed using multiple threads, up to the limit of the processors on a physical node. Quad-core RAPPORT cloud nodes were used for this analysis. Performance is shown to improve for every increase of nodes or threads, with the greatest benefit when moving from one to two threads or one to two processors. The optimal configuration is eight nodes each with four cores, which is limited by the runtime configuration of MrBayes (run and chains) and the processors available on single nodes.

The final MrBayes benchmark (figure 4) demonstrates the runtime of the empirically established optimal configuration for each platform. These results suggest that workloads with low memory requirements could be moved from an existing specialist, dedicated computing resource to a private cloud based on a commodity cluster. The runtime difference between the ec2 and codon platforms, despite their similar hardware specifications, may be explained by a difference in CPU cache size, but this requires further investigation.

Time constraints prevented extensive RAxML performance analysis but single-node benchmarks indicated similar performance across the available platforms. There is no benefit to multi-process parallelism in hill-climbing (standard) mode, unlike bootstrapping mode, which can exploit both fine- and coarse-grained parallelism at the cost of memory usage.

(ii) Performance analysis—gene structure prediction

Deployment of GenomeThreader on cloud resources was straightforward, requiring the binary and the datasets to be transferred to the respective machine(s) where the software could then be invoked using a command line interface. A 32-bit binary of GenomeThreader 1.4.7 was tested on the two private cloud platforms, lesc and rapport, using an m1.xlarge instance, and on the non-virtualized lesc-nv resources. Tests were also carried out on a standalone 64-bit BSS compute server (bss) and a desktop workstation with a 32-bit Linux install.

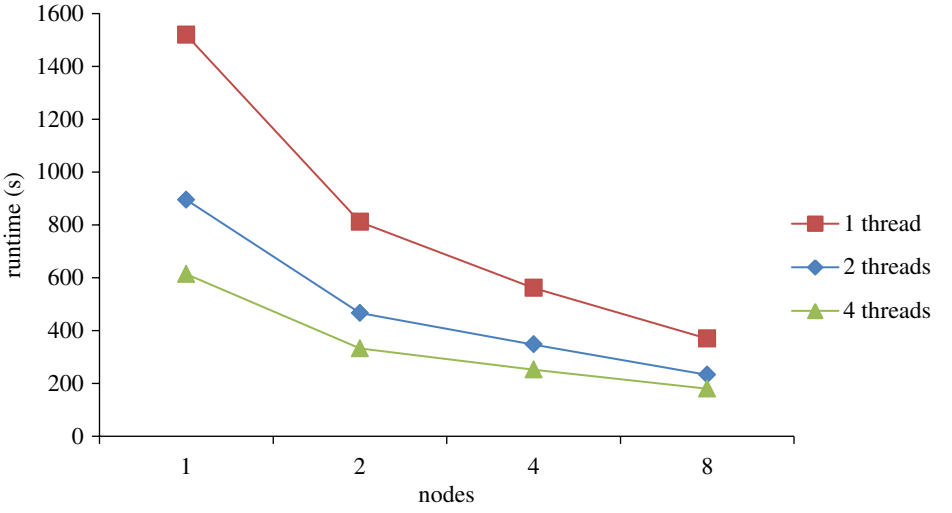

Figure 3. MrBayes multi-thread multi-process performance for 218 taxa alignment on RAPPORT cloud nodes; from Cohen *et al.* [2]. (Online version in colour.)

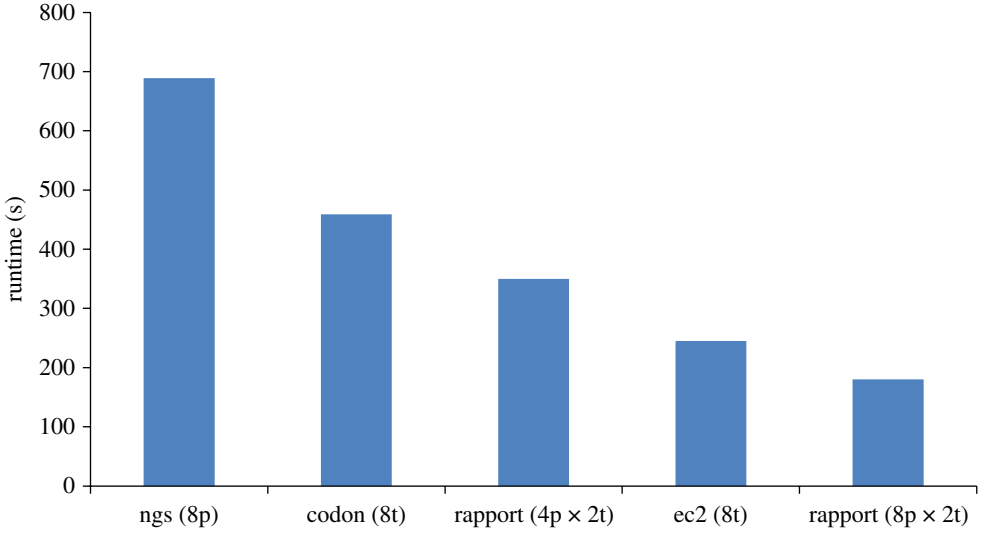

Figure 4. MrBayes multi-thread multi-process (optimal configuration) performance for 218 taxa alignment on various platforms (p, processes; t, threads); from Cohen *et al.* [2]. (Online version in colour.)

GenomeThreader was benchmarked on two transcript sets. Both were extracted from the publicly available SRA archive SRR027942, generated as part of the transcriptome analysis for the Solanum Trichome Project [44]. Chromosome 1 of the tomato genome sequence [45] has been used as reference genome. Two alignment tests were performed. A 'small' test, aligning 900 454 ESTs (162 kB file) against the repeat-masked version of chromosome 1 (88 MB), and a 'large' test, aligning 5 022 454 ESTs (997 kB) against the unmasked chromosome 1 sequence.

Table 1 shows the reported runtimes for both tests on various platforms. Testing on some hardware platforms (lesc/lesc-nv, large) was not possible because of hardware issues and time limitations, but the results show that the bss platform runs the tests with the best performance. This was expected owing to the better specification of the resource. The

Table 1. GenomeThreader runtime (s) for two datasets and various platforms.

test	bss	lesc	lesc-nv	rapport	workstation
small	51.013	117.290	87.612	90.937	94.993
large	728.357	—	—	922.821	957.371

difference in results between lesc and lesc-nv suggests a significant overhead resulting from the cloud/virtualization layers.

Parallelization of GenomeThreader runs was attempted using Apache Hadoop 0.20.2 [46]. GenomeThreader requires two types of input datasets, the genomic sequences and the RNA-seq data. It is important that both can be split to constrain memory requirements. The approach taken by HadoopBlast [47], which allows the parallel execution of Blast by distributing the blast database and software using Distributed Cache and using a custom record reader to process the set of FASTA input files, was extended by partitioning both input datasets when generating a Hadoop input file. This process was implemented as a custom Perl script that splits the datasets based on given criteria.

In the map step of the map–reduce framework, each mapper has access through HDFS to a specific designated pair of files containing the subsets of genomic sequences and reads. A subsequent reduce phase merges all spliced alignments and optionally computes the final consensus spliced alignment. Merging and computation of consensus alignment are performed using the open-source GenomeTools software [48] which, along with GenomeThreader, is distributed across nodes using Distributed Cache.

5. Best practice

As shown in the preceding sections, there are situations where cloud infrastructure offers opportunities to support users and providers of computational resources by offering a practical route for expansion of existing in-house capacity or a viable alternative to provisioning new infrastructure. While the cloud may present an ideal option for some researchers, there are still domains where traditional cluster or grid platforms may present a more suitable option. The best-practice guidance aims at supporting both potential users and funders of computational infrastructure in understanding their options and making effective, optimal decisions about infrastructure provisioning and use.

(a) Best practice for researchers

An analysis of the experiences from researchers using clouds has enabled the project to produce a guide covering best practice for using clouds in research [5]. This work is based on interviews with and information provided by the researchers working within RAPPORT in the project's three application domains, supplemented by additional material from other cloud computing pilots, and we summarize the main findings from the guide in this section.

Researchers considering the adoption of cloud computing can refer to a workflow of activities (figure 5) to help them understand how a cloud may benefit their research, how to port their application to their chosen cloud, how to understand the technical, legal, ethical, environmental and economic challenges they may face, how to consider alternative platforms and applications, and how to plan for contingencies.

Many of these activities are integral to the effective execution of any software development project. Some, however, specifically relate to challenges arising from the work described in §§3 and 4. These include the following.

— *Cloud suitability*. Researchers need to understand how the different ways an application can be run impact upon its resource demands. For example, the memory required by

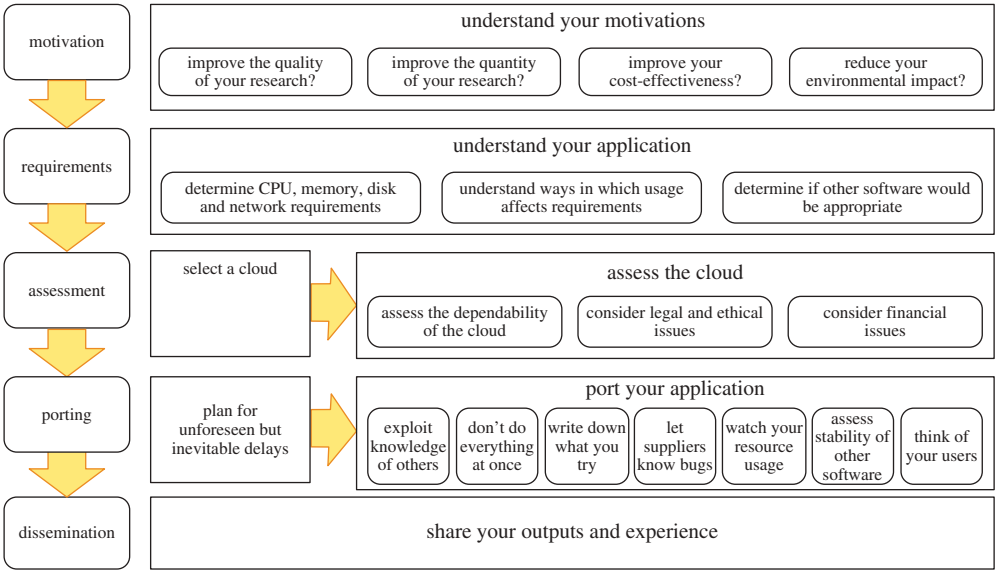

Figure 5. A best practice for researchers workflow; from Chue Hong *et al.* [5]. (Online version in colour.)

runs of GENOMETHREADER could be influenced by the way input was partitioned, and this was investigated through communication with the software's authors.

— *Legal issues*. Particularly, software licensing may restrict software transfers to third parties or license software to run only on machines with specific MAC/IP addresses. This was experienced in at least one of the application domains, where licensing issues with one candidate software package resulted in an open-source alternative being selected.

— *Financial issues*. Commercial clouds impose financial overheads that may be beyond the reach of researchers. For example, initial calculations for the cost of data transfer for a large HEP analysis task suggested that this would be prohibitive. This changed during the course of the project, but other data-related costs such as storage/download still remained an issue.

— *Managing the transition to a new type of platform*. Moving software to cloud platforms is likely to present technical challenges that are difficult to predict. Significant time can be lost in investigating and resolving them. For example, while the CernVM virtual image is provided for a number of virtualization platforms, running it under the EUCALYPTUS platform and configuring the associated CMS software both took a reasonable amount of time and effort.

— *Security issues*. While cloud security was not a focus of this work, it is a complex area presenting a challenge for researchers. In particular, researchers need to understand where data may be stored, what security guarantees, if any, are provided, and restrictions on data they hold. Where such restrictions prevent data being placed on remote resources, using a local private cloud platform may be an option. Alternatively, researchers may consider the use of hybrid public–private clouds, for which approaches to systematically analyse security requirements are starting to emerge (see e.g. [49]).

More generally, the challenges faced and tasks that RAPPORT researchers had to carry out when moving their applications to cloud resources demonstrate that there is overhead to the process and many required tasks may well be outside the traditional areas of researchers' expertise. Such tasks include developing and testing virtual disk images, resource capacity management and resource provisioning. Traditionally, these tasks would have been handled by

system administrators, and this shift may, among other consequences, result in researchers having direct responsibility for the resulting costs of their resource usage.

(b) Best practice for funders

The project also produced a guide detailing best practice for funding the use of the cloud in research [6]. A primary recommendation was to ensure that researchers demonstrate an understanding of the potential costs and challenges in using cloud computing and have suitable plans in place to address these issues. Additional recommendations cover different ways that funders can help the adoption of cloud infrastructure. Although most of the recommendations are not directly related to the findings covered in §4, two result from the assessment of practice and performance of the three application domains investigated in RAPPORT:

— providing opportunities for researchers and system administrators to exchange knowledge/experiences about working with cloud infrastructure; and
— in the case of provisioning a community cloud, being aware of the specialist requirements of research computing and how these requirements can be effectively supported via a cloud infrastructure; in general, HPC research may present fewer opportunities for resource sharing, owing to higher usage levels, and require more thought as to how virtualized resources can be made available to individual users.

It is intended that recommendations such as these help funders to deliver to researchers the resources that they need in a way that is cost-effective, and to the benefit of not just individual projects, but research communities as a whole.

6. Conclusions

The work undertaken in the RAPPORT project brought together researchers from a group of domains to investigate running a set of scientific applications on cloud resources. The work served as a means to identify the feasibility and potential benefits or issues of using infrastructure cloud resources to provide computing capacity for researchers. The experiment also sought to gain an understanding of the costs, both financial and behavioural/attitudinal, that could be incurred when moving to exploit cloud computing. The work resulted in a significant gain in technical knowledge and skills for the researchers involved and the development of best practice to guide researchers considering using the cloud and funders considering underwriting their use. Investigations suggested that there is no fundamental impediment to running certain types of HPC applications on cloud resources. The limitations many researchers face when trying to run HPC applications, such as not being able to access enough or sufficiently powerful resources or not being able to renew resources frequently enough, are all things that IaaS cloud platforms can assist with. There are many opportunities for future work, including the refining of best practice for researchers by assessing its utility in future cloud porting projects undertaken by researchers and carrying out investigations of other private cloud frameworks on a range of hardware.

We would like to thank JISC and EPSRC for funding the RAPPORT project (EP/I034246/1) under their Pilot Projects for Cloud Computing in Research programme. We would also like to thank the application providers and those who provided application-related support as part of our work to better understand the applications and the most appropriate means for their deployment onto cloud infrastructure.

References

1. JISC. 2011 Cloud Computing in Research, RAPPORT. See http://cloudresearch.jiscinvolve.org/wp/category/projects/rapport/ (accessed 19 January 2012).
2. Cohen J *et al.* 2011 RAPPORT—Porting applications to clouds. RAPPORT project report (draft, unpublished), full version submitted to JISC/EPSRC, July 2011.

3. Cohen J *et al.* 2011 RAPPORT: Robust Application Porting for HPC in the Cloud—Project Final Report. (unpublished), updated short version of [2] submitted to JISC, November 2011.
4. HEFCE. 2012 JISC: Inspiring innovation. See http://www.jisc.ac.uk/ (accessed 19 January 2012).
5. Chue Hong N, Jackson M, Cohen J. 2011 Best practice for using cloud in research. RAPPORT project technical report (draft), JISC/EPSRC Cloud Computing in Research Programme. See http://software.ac.uk/resources/guides/cloud-for-research-best-practice.
6. Chue Hong N, Jackson M, Cohen J. 2011 Best practice for funding the use of cloud in research. RAPPORT project technical report (draft), JISC/EPSRC Cloud Computing in Research Programme. See http://software.ac.uk/resources/guides/cloud-for-funders-best-practice.
7. Nurmi D, Wolski R, Grzegorczyk C, Obertelli G, Soman S, Youseff L, Zagorodnov D. 2009 The Eucalyptus open-source cloud-computing system. In *Proc. 9th IEEE Int. Symp. on Cluster Computing and the Grid* (*CCGRID 2009*), *Shanghai, China*, pp. 124–131 (doi:10.1109/CCGRID.2009.93)
8. Eucalyptus. 2011 Eucalyptus Community. See http://open.eucalyptus.com/ (accessed 4 January 2012).
9. Canonical Ltd. 2011 Cloud—Ubuntu. See http://www.ubuntu.com/cloud (accessed 4 January 2012).
10. Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A. 2003 Xen and the art of virtualization. In *Proc. 19th ACM Symp. on Operating Systems Principles (SOSP '03)*. New York, NY: ACM. (doi:10.1145/945445.945462)
11. KVM. 2012 Main Page—Kernel Based Virtual Machine. See http://www.linux-kvm.org/ (accessed 19 January 2012).
12. JANET. 2011 JANET, the UK's education and research network. See http://www.ja.net/ (accessed 4 January 2012).
13. NGS. 2011 Accessing the NGS cloud service. See http://www.ngs.ac.uk/accessing-the-ngs-cloud-service (accessed 4 January 2012).
14. Amazon Web Services LLC. 2011 Amazon Elastic Compute Cloud (Amazon EC2). See http://aws.amazon.com/ec2 (accessed 4 January 2012).
15. Rackspace Limited. 2012 Rackspace Cloud UK. See http://www.rackspace.co.uk/cloud-hosting/ (accessed 27 June 2012).
16. Joyent, Inc. 2012 Joyent Cloud. See http://www.joyentcloud.com/ (accessed 27 June 2012).
17. VMware, Inc. 2012 Public Cloud Computing Services with VMware Virtualization. See http://www.vmware.com/uk/cloud-computing/public-cloud/index.html (accessed 27 June 2012).
18. Huang W, Liu J, Abali B, Panda DK. 2006 A case for high performance computing with virtual machines. In *Proc. 20th Int. Conf. on Supercomputing* (*ICS '06*). New York, NY: ACM. (doi:10.1145/1183401.1183421)
19. Nagarajan AB, Mueller F, Engelmann C, Scott SL. 2007 Proactive fault tolerance for HPC with Xen virtualization. In *Proc. 21st Int. Conf. on Supercomputing* (*ICS '07*). New York, NY: ACM. (doi:10.1145/1274971.1274978)
20. Pellicer S, Chen G, Chan K, Pan Y. 2008 Distributed sequence alignment applications for the public computing architecture. *IEEE Trans. NanoBiosci.* **7**, 35–43. (doi:10.1109/TNB.2008.2000148)
21. CERN. 2012 CMS Experiment. See http://cms.web.cern.ch/ (accessed 4 January 2012).
22. CERN. 2008 The Large Hadron Collider. See http://press.web.cern.ch/public/en/LHC/LHC-en.html (accessed 4 January 2012).
23. CMSPublic Web. 2011 CMSSW Application Framework. See https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookCMSSWFramework (accessed 4 January 2012).
24. CERN. 2012 Worldwide LHC Computing Grid. See http://lcg.web.cern.ch/lcg/ (accessed 4 January 2012).
25. Scientific Linux. 2011 See http://www.scientificlinux.org/ (accessed 4 January 2012).
26. Huelsenbeck JP, Ronquist F. 2001 MRBAYES: Bayesian inference of phylogenetic trees. *Bioinformatics* **17**, 754–755. (doi:10.1093/bioinformatics/17.8.754)
27. Ronquist F, Huelsenbeck JP. 2003 MRBAYES 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics* **19**, 1572–1574. (doi:10.1093/bioinformatics/btg180)
28. Altekar G, Dwarkadas S, Huelsenbeck JP, Ronquist F. 2004 Parallel Metropolis-coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. *Bioinformatics* **20**, 407–415. (doi:10.1093/bioinformatics/btg427)

29. Stamatakis, A. 2006 RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics* **22**, 2688–2690. (doi:10.1093/bioinformatics/btl446)
30. Pfeiffer W, Stamatakis A. 2010 Hybrid MPI/Pthreads parallelization of the RAxML phylogenetics code. In *Proc. 2010 IEEE Int. Symp. on Parallel and Distributed Processing, Workshops and PhD Forum* (*IPDPSW*), *Atlanta, GA*, pp. 1–8. (doi:10.1109/IPDPSW.2010.5470900)
31. Ott M, Zola J, Aluru S, Stamatakis A. 2007 Large-scale maximum likelihood-based phylogenetic analysis on the IBM BlueGene/L. In *Proc. 2007 ACM/IEEE Conf. on Supercomputing* (*SC '07*). New York, NY: ACM.
32. 2010 Gathering clouds and a sequencing storm: why cloud computing could broaden community access to next-generation sequencing. *Nat. Biotechnol.* **28**, 1. (doi:10.1038/nbt0110-1)
33. Afgan E, Baker D, Coraor N, Goto H, Paul IM, Makova KD, Nekrutenko A, Taylor D. 2011 Harnessing cloud computing with Galaxy Cloud. *Nat. Biotechnol.* **29**, 972–974. (doi:10.1038/nbt.2028)
34. Baker M. 2010 Next-generation sequencing: adjusting to data overload. *Nat. Methods* **7**, 495–499. (doi:10.1038/nmeth0710-495)
35. Morozova O, Hirst M, Marra MA. 2009 Applications of new sequencing technologies for transcriptome analysis. *Annu. Rev. Genom. Hum. Genet.* **10**, 135–151. (doi:10.1146/annurev-genom-082908-145957)
36. Wang Z, Gerstein M, Snyder M. 2009 RNA-Seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.* **10**, 57–63. (doi:10.1038/nrg2484)
37. Gremme G, Brendel V, Sparks ME, Kurtz S. 2005 Engineering a software tool for gene structure prediction in higher organisms. *Inf. Softw. Technol.* **47**, 965–978. (doi:10.1016/j.infsof.2005.09.005)
38. Gremme G. 2011 GenomeThreader frequently asked questions (FAQ). See http://www.genomethreader.org/faq.html (accessed 19 January 2012).
39. CERN. 2012 CernVM. See http://cernvm.cern.ch/portal/ (accessed 4 January 2012).
40. The Apache Software Foundation. 2007 Hadoop® MapReduce. See http://hadoop.apache.org/mapreduce/ (accessed 18 January 2012).
41. The Apache Software Foundation. 2007 Welcome to Hadoop® Distributed File System! See http://hadoop.apache.org/hdfs/ (accessed 18 January 2012).
42. Amazon Web Services LLC. 2012 Amazon EC2 Instance Types. See http://aws.amazon.com/ec2/instance-types/ (accessed 18 January 2012).
43. MrBayes MbWiki. 2007 Tutorial 3.2. See http://mrbayes.sourceforge.net/wiki/index.php/Tutorial_3.2 (accessed 19 January 2012).
44. Michigan State University. 2012 The Solanum Trichome Project. See http://www.trichome.msu.edu/ (accessed 19 January 2012).
45. The Tomato Genome Consortium. 2012 The tomato genome sequence provides insights into fleshy fruit evolution. *Nature* **485**, 635–641. (doi:10.1038/nature11119)
46. The Apache Software Foundation. 2011 Welcome to Apache™ Hadoop®. See http://hadoop.apache.org/ (accessed 27 June 2012).
47. SALSAHPC. 2010 Big Data for Science: Apache Hadoop Lab 2. See http://salsahpc.indiana.edu/tutorial/hadoopblast.html (accessed 19 January 2012).
48. Gremme G. 2011 GenomeTools. See http://genometools.org/index.html (accessed 19 January 2012).
49. Watson P. 2011 A multi-level security model for partitioning workflows over federated clouds. In *Proc. 3rd IEEE Int. Conf. on Cloud Computing Technology and Science* (*CloudCom 2011*), *Athens, Greece*, pp. 180–188.