Using Clouds for Technical Computing

Geoffrey Fox[a] and Dennis Gannon[b]

[a] *School of Informatics and Computing, Indiana University, Bloomington IN*

[b]*Microsoft Research Redmond WA*

Abstract. We discuss the use of cloud computing in technical (scientific) applications and identify characteristics such as loosely-coupled and data-intensive that lead to good performance. We give both general principles and several examples with an emphasis on use of the Azure cloud.

Keywords. Clouds, HPC, Programming Models, Applications, Azure

In the past five years cloud computing has emerged as an alternative platform for high performance computing [1-8]. Unfortunately, there is still confusion about the cloud model and its advantages and disadvantages over tradition supercomputing based problem solving methods. In this paper we characterize the ways in which cloud computing can be used productively in scientific and technical applications. As we shall see there is a large set of application that can run on a cloud and a supercomputer equally well. There are also applications that are better suited to the cloud and there are applications where a cloud is a very poor replacement for a supercomputer. Our goal is to illustrate where cloud computing can complement the capabilities of a contemporary massively parallel supercomputer.

1. Defining the Cloud.

It would not be a huge exaggeration to say that the number of different definitions of cloud computing greatly exceeds the number of actual physical realizations of the key concepts. Consequently, if we wish to provide a characterization of what works "in the cloud", we need a grounding definition and we shall start with one that most accurately describes the commercial public clouds from Microsoft, Amazon and Google. These public clouds consist of one or more large data centers with the following architectural characteristics

1. The data center is composed of containers of racks of basic servers. The total number of servers in one data center is between 10,000 and a million. Each server has 8 or more CPU cores and a 64GB of shared memory and one or more terabyte local disk drives. GPGPUs or other accelerators are not common.

2. There is a network that allows messages to be routed between any two servers, but the bisection bandwidth of the network is very low and the network protocols implement the full TCP/IP stack at a sufficient enough level so that every server can be a full Internet host. This is an important point of distinction between a cloud data center and a supercomputer. Data center networks optimize traffic between users on the Internet and the servers in the cloud. This is very different from supercomputer networks which are optimized to minimize interprocessor latency and maximize bisection bandwidth. Application data communications on a supercomputer generally take place over specialized physical and data link layers of the network and interoperation with the Internet is usually very limited.

3. Each server in the data center is host to one or more virtual machines and the cloud runs a "fabric controller" which schedules and manages large sets of VMs across the servers. This fabric controller is the operating system for the data center. Using a VM as the basic unit of scheduling means that an application running on the data center consists of one or more complete VM instances that implement a web service. This means that each VM instance can contain its own specific software library versions and internal application such as databases and web servers. This greatly simplifies the required data center software stack, but it also means that the basic unit of scheduling involves the deployment of one or more entire operating systems. This activity is much slower than installing and starting an application on a running OS. Most large scale cloud services are intended to run 24x7, so this long start-up time is negligible. Also the fabric controller is designed to manage and deal with faults. When a VM fails the fabric controller can automatically restart it on another server. On the downside, running a "batch" application on a large number of servers can be very inefficient because of the long time it may take to deploy all the needed VMs.

4. Data in a data center is stored and distributed over many spinning disks in the cloud servers. This is a very different model than found in a large supercomputer, where data is stored in network attached storage. Local disks on the servers of supercomputers are not frequently used for data storage.

2. Cloud Offerings as Public and Private, Commercial and Academic

As stated above, there are more types of clouds than is described by this public data center model. For example, to address a technical computing market, Amazon has introduced a specialized HPC cloud that uses a network with full bisection bandwidth and supports GPGPUs. "Private clouds" are small dedicated data centers that have various combinations of the properties 1 through 4 above. FutureGrid is the NSF research testbed for cloud technologies and it operates a grid of cloud deployments running on modest sized server clusters [9].

The major commercial clouds are those from Amazon, Google (App Engine), and Microsoft (Azure). These are constructed as a basic scalable infrastructure with higher level capabilities -- commonly termed Platform as a Service (PaaS). For technical computing, important platform components include tables, queues, database, monitoring, roles (Azure), and the cloud characteristic of elasticity (automatic scaling). MapReduce [10], which is described in detail below, is another major platform service offered by these clouds. Currently the different clouds have different platforms although the Azure and Amazon platforms have many similarities. The Google Platform is targeted at scalable web applications and not as broadly used in technical computing community as Amazon or Azure, but it has been used by selected researchers on some very impressive projects.

Commercial clouds also offer Infrastructure as a Service (IaaS) with compute and object storage features and Amazon EC2 and S3 being the early entry in this field. There are four major open source (academic) cloud environments Eucalyptus, Nimbus, OpenStack and OpenNebula (Europe) which focus at the IaaS level with interfaces similar to Amazon. These can be used to build "private clouds" but the interesting platform features of commercial clouds are not fully available. Open source Hadoop and Twister offer MapReduce features similar to those on commercial cloud platforms and there are open source possibilities for platform features like queues (RabbitMQ, ActiveMQ) and distributed data management system (Apache Cassandra). However, apart from a very new Microsoft offering [11], there is no complete packaging of PaaS features available today for academic or private clouds. Thus interoperability between private and commercial clouds is currently only at IaaS level where it is possible to reconfigure images between the different virtualization choices and there is an active cloud standards activity. The major commercial virtualization products such as VMware and Hyper-V are also important for private clouds but also does not have built-in PaaS capabilities.

We expect more academic interest in PaaS as the value of platform capabilities become clearer from the ongoing application work such as that described in this paper for Azure.

3. Parallel Programming Models

Scientific Applications that are run on massively parallel supercomputers follow strict programming models that are designed to deliver optimal performance and scalability. Typically these programs are designed using a loosely synchronous or bulk synchronous model and the Message Passing Interface (MPI) communication library. In this model, each processor does a little computing and then stops to exchange data with other processors in a very carefully orchestrated manner and then the cycle repeats. Because the machine has been designed to execute MPI very efficiently, the

relative time spent in communicating data can be made small compared to the time doing actual computation. If the complexity of the communication pattern does not grow as you increase the problem size, then this computation scales very well to large numbers of processors.
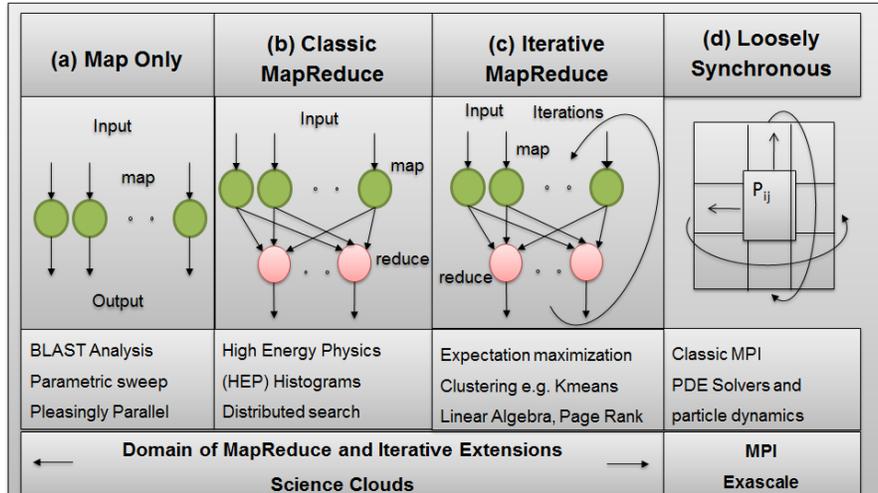

(a) Map Only	(b) Classic MapReduce	(c) Iterative MapReduce	(d) Loosely Synchronous
BLAST Analysis Parametric sweep Pleasingly Parallel	High Energy Physics (HEP) Histograms Distributed search	Expectation maximization Clustering e.g. Kmeans Linear Algebra, Page Rank	Classic MPI PDE Solvers and particle dynamics
Domain of MapReduce and Iterative Extensions Science Clouds			MPI Exascale

Figure 1: Forms of Parallelism and their application on Clouds and Supercomputers

The typical cloud data center does not have a low latency high bandwidth network needed to run communication intensive MPI programs. However, there is a subclass of traditional parallel programs called MapReduce computations that do well on clouds architectures. MapReduce computations have two parts. In the first part you "map" a computation to each member of an input data set. Each of these computations has an output. In the second part you reduce the outputs to a single output. For example, finding the minimum of some function across a large set of inputs. First apply the function to each input in parallel and then compare the results to pick the smallest. Two variants of MapReduce are important. In one case there is no real reduce step. Instead you just apply the operation to each input in parallel. This is often called an "embarrassingly parallel" or "pleasingly parallel" computation. At the other extreme there are computations that use MapReduce inside an iterative loop. A large number of linear algebra computations fit this pattern. Clustering and other machine learning computations also take this form. Figure 1 illustrates these different forms of computation. Cloud computing works well for any of these MapReduce variations. In fact a small industry has been built around doing analysis (typically called data analytics) on large data collections using MapReduce on cloud platforms.

Note that the synchronization costs of clouds lie in between those of grids and supercomputers (HPC clusters) and applications suitable for clouds include all computational tasks appropriate for grids and HPC applications without stringent synchronization. Synchronization in clouds includes both effects of commodity

networks but also overheads due to shared systems and virtualized (software) system. The latter is illustrated from Azure application reported by Thilina Gunarathne, Xiaoming Gao and Judy Qiu from Indiana University using Twister4Azure an early iterative MapReduce environment [12-14]. The Multidimensional Scaling application is dominated by two linear algebra steps denoted by alternating dark and light task groups in figure 2. As in many classical parallel applications, each step consists of many tasks that should execute for essentially identical times as they are executing identical number of elemental matrix operations. The graph shows a job with fluctuations in execution time for individual tasks, which currently limits scale at which good efficient parallel speed up can be obtained. All tasks must wait until the slowest straggler finishes. Nevertheless the observed speedups are significant and show parallel programming is satisfactory on this type of problem on up to 512 cores.
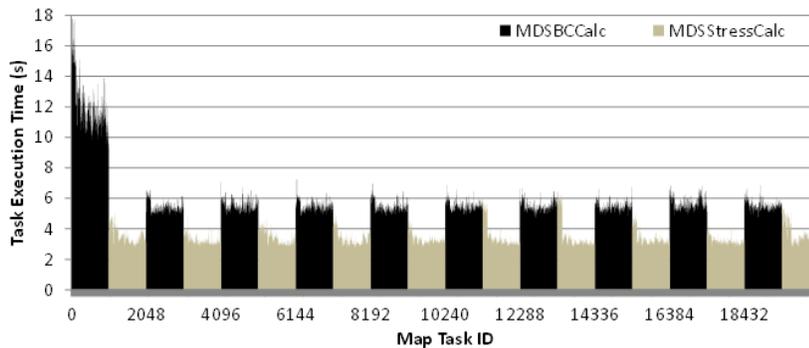

Figure 2: Execution times of individual tasks on Azure for an Iterative MapReduce Twister4Azure implementation of Multidimensional Scaling

4. Classes of Cloud Applications

Limiting the applications of the cloud to classical scientific computation would miss the main reason that the cloud exists. Large scale data center are the backbone of many ubiquitous cloud services we use every day. Internet search, mobile maps, email, photo sharing, messaging, social networks are all dependent upon large data center clouds. These applications all depend upon the massive scale and bandwidth to the Internet that the cloud provides. Note clouds are executing jobs at scales much larger than those on supercomputers but that scale does not come from tightly coupled MPI. Rather scale can come from two aspects: Scale (or parallelism) from a multitude of users and scale from execution of an intrinsically parallel application within a single job (invoked by a single user).

Clouds naturally exploit parallelism from multiple users or usages. The Internet of things will drive many applications of the cloud. It is projected that there will be 24 billion devices on the Internet by 2020. Most will be small sensors that send streams of

information into the cloud where it will be processed and integrated with other streams and turned into knowledge that will help our lives in a million small and big ways. It is not unreasonable for us to believe that we will each have our own cloud-based personal agent that monitors all of the data about our life and anticipates our needs 24x7.   The cloud will become increasing important as a controller of and resource provider for the Internet of Things. As well as today's use for smart phone and gaming console support, "smart homes" and "ubiquitous cities" build on this vision and we could expect a growth in cloud supported/controlled robotics.

Beyond the Internet of things, we have the commodity applications such as social networking, internet search and e-commerce. Finally we mention the "long tail of science" [15, 16] as an expected important usage mode of clouds. In some areas like particle physics and astronomy, i.e. "big science", there are just a few major instruments generating now petascale data driving discovery in a coordinated fashion. In other areas such as genomics and environmental science, there are many "individual" researchers with distributed collection and analysis of data whose total data and processing needs can match the size of big science. Clouds can provide scaling use for this important aspect of science.

Large scale Clouds also support parallelism within a single job with an obvious example of Internet search which has both "usage parallelism" listed above but also parallelism within each search as the summary of the web is stored on multiple disks on multiple computers and searched independently for each query. MapReduce was introduced to support successfully this type of parallelism. The examples given later will illustrate various forms of parallelism discussed above and summarized in figure 1. Today probably the "Map only" or "Pleasingly Parallel" mode is most common where a single job invokes multiple independent components. This is very similar to the parallelism from multiple users and indeed we will see examples where the different "map tasks" from a "single application" are generated outside the cloud which sees these tasks as separate usages. As well as Internet search, the second parallel category MapReduce is important in the Information Retrieval field with a good example being the well-known Latent Dirichlet Allocation algorithm for topic or "hidden/latent" feature determination. Basic data analysis can often be formulated as a simple MapReduce problem where independent event selection and processing (the maps) is followed by a reduction forming global statistics such as histograms. The final class of parallel applications explored so far is Iterative MapReduce implemented on Azure as Daytona from Microsoft or Twister4Azure from Indiana University. Here Page Rank is a well-known component of Internet search technology that corresponds to finding eigenvector of largest eigenvalue for the sparse matrix of internet links. This algorithm illustrates those that fit well Iterative MapReduce and other algorithms with parallel linear algebra at their core have been explored on Azure.

Multiple usages or splitting the Internet summary over MapReduce nodes are all forms of a generalized data parallelism. However clouds also support well the functional parallelism seen where a given application breaks into multiple components which typically supported by workflow technology [17-20]. Workflow is an old idea and was developed extensively as part of Grid research. We will see from examples below that its use can be taken over directly by clouds without conceptual change. In fact the importance of Software as a Service (SaaS) for commercial clouds illustrates that another concept Services developed for science by the grid community is a successful key feature of cloud applications. Workflow as the technology to orchestrate or control clouds will continue to be a critical building block for cloud applications.

Important access models for clouds include portals, which are often termed Science Gateways and these can be used similarly to grids. Another interesting access choice seen in some cases is that of the queue either implemented using conventional publish-subscribe technology (such as JMS) or the built in queues of the Azure and Amazon platforms. Applications can use the advanced platform features of clouds (queues, tables, blobs, SQL as a service for Azure) to build advanced capabilities more easily than on traditional (HPC) environments. Of course the pervasive "on demand" nature of cloud computing emphasizes the critical importance of task scheduling where either the built-in cloud facilities are used or alternatively there is some exploration of technologies like Condor developed for grids and clusters. Publish-Subscribe use varies from managing the launch of "worker jobs" to the control of sensors [21, 22] and robots [23, 24] from the cloud.

The nature of the use of data [25] is another interesting aspect of cloud applications that currently is still in its infancy [26] but is expected to become important as for example future large data repositories will need cloud computing facilities. A key challenge as the data deluge grows is how we avoid unnecessary data transport and if possible bring the computing to the data [27-31]. We need to understand [32, 33] the tradeoffs between traditional wide area systems like Lustre, Object stores which the heart of Amazon, Azure and OpenStack storage today and the "data-parallel" file systems popularized by HDFS, the Hadoop File System. We expect this to be a growing focus of future cloud applications.

5. The Process of Building a Cloud Application

Attempts to directly port a conventional HPC application to a cloud platform often fail in their performance goals [34-44] although an exception to this is the Amazon HPC Cloud which seems to perform very well [45]. It is not unlike early attempts to move "vectorized" HPC application to massively parallel non-shared memory message passing clusters. The challenge is to think differently and rewrite the application to

support the new computational and programming models.   In the case of clouds the following practices lead to success

1. **Build the application as a service.**   Because you are deploying one or more full virtual machines and because clouds are designed to host web services, you want your application to support multiple users or, at least, a sequence of multiple executions.   If you are not using the application, scale down the number of servers and scale up with demand. Attempting to deploy 100 VMs to run a program that executes for 10 minutes is a waste of resources because the deployment may take more than 10 minutes. To minimize start up time one needs to have services running continuously ready to process the incoming demand.

2. **Build on existing cloud deployments.**   The cloud is ideal for large MapReduce computations so use an existing MapReduce deployment such as Hadoop or a similar service. One also needs to exploit the natural elasticity for high throughput applications.

3. **Use PaaS if possible.** For PaaS clouds like Azure use the tools that are provided such as queues, web and worker roles and blob, table and SQL storage.

4. **Design for failure.**   Applications that are services that run forever will experience failures.   The cloud has mechanisms that automatically recover lost resources, but the application needs to be designed to be fault tolerant. In particular, environments like MapReduce (Hadoop, Daytona, Twister4Azure) will automatically recover many explicit failures and adopt scheduling strategies that recover performance "failures" from for example delayed tasks. One expects an increasing number of such Platform features to be offered by clouds and users will still need to program in a fashion that allows task failures but be rewarded by environments that transparently cope with these failures.

5. **Use as a Service where possible.** Capabilities such as SQLaaS (database as a service or a database appliance) provide a friendlier approach than the traditional non-cloud approach exemplified by installing MySQL on the local disk. We anticipate many prepackaged aaS capabilities such as Workflow as a Service for eScience will be developed and simplify the development of sophisticated applications. In fact "Research as a Service" is described in a recent paper [1].

6. Expect Changes. Clouds are still young and we can expect things to come and go. For example, the interesting [46-50] Dryad approach to MapReduce disappeared (replaced by Hadoop [51]) on Azure. Further we expect a growing interest in topics like GPGPU's in the cloud [52].

7. Moving Data is a challenge.   The general rule is that one should move computation to the data, but if the only computational resource available is a cloud, you are stuck if

the data is not also there. Moving a petabyte from a laboratory to the cloud over the Internet will take time.   The ideal situation is when you can gradually stream the data to the cloud over time as it is being created. But if it exists in one place the best method of moving it is to physically ship the disk drives to the data center.   This service is available from some cloud providers.

6. The Economics of Clouds

Comparing the cost of computing in the cloud to the cost of purchasing a cluster is somewhat challenging because of two factors. First, for most researchers, the Total Cost of Ownership (TCO) of a cluster they purchase is often not visible to them. Space and power are buried in the University's expense overall infrastructure bills. The systems administration for small clusters is often done by students, so it appears to be free of cost. For large, university managed supercomputer centers the situation is different and the costs are known but not widely published. However, there have been some TCO studies. Paterson, et. al. [13] report that the 3-year total cost of ownership for a cluster of servers for research varies between 8 and 15 times the purchase price of the hardware.   In a study by IMEX Research, the 3 year TCO of a 512 node HPC cluster averages about $7 million dollars.   This is based on a 2005 configurations with 1U dual core servers with 2 GB memory [14], so it is by no means current technology. However If we compare this to the cost of 1024 cores on Windows Azure running 24x7 for 3 years the total is $2.6M (with "committed use" discounts). For Linux on Amazon EC2 the cost is $2.3M. These costs do not include persistent storage or network bandwidth.   100TBytes of triply replicated on-line cloud storage for 3 years is an additional $400K. Network bandwidth and data transactions add additional costs

Based on these numbers, one may conclude a modest advantage to computing in the cloud, but that would miss a critical point. The advantage of the cloud for researchers is the ability to use, and pay for it, on demand. If a researcher has a need for 1000 cores for a week, the cost is $21K.   If the researcher needs only need a few cores between periods of heavy use the cost is a small increment.   In this case there is a clear advantage to cloud computing over purchasing a cluster that is not fully utilized.

7. Case Studies from the Microsoft Cloud Computing Engagement Program

In the following subsections we describe a number of case studies from the Microsoft Cloud Computing Engagement Program [53] which awarded time on Azure to 83 research teams.   The examples here are abstracted from reports from the top 30 groups (based on resources consumed to date).   These illustrate the models described above with services, workflow, parallelism from multiple users or repetitive internal tasks, and use of MapReduce. They also demonstrate the use of the cloud as a web-enabled

"science gateway" where the application is built as a service that can be executed by remote users. Many of the projects in our "top 30" use variations on the MapReduce theme. We have selected this collection of 12 projects because they demonstrate the variety and creativity of research community using the Azure cloud. The interesting EU partnership Venus-C [54] is part of the Azure engagement with 27 projects covering Biodiversity and Biology (2), Molecular Cell and Genomic Biology (7), Medicine(3), Chemistry(3), Physics(1), Civil Engineering and Architecture (4), Earth Science (1), Naval/Mechanical Engineering (2), Civil Protection (1), Mathematics(1) and Information Technology(2). These successful project used extensions of the Microsoft Generic Worker [55] and Barcelona COMPSs [56] systems but avoided MapReduce as it was not available at the start of the project.

7.1. University of Newcastle

Paul Watson and Jacek Cala of The University of Newcastle use an Azure based system (e-Science Central [57, 58]) to execute millions of workflows, each of which is a test of a target molecule for possible use as an anti-cancer drug. The scientists use a method known as Quantitative Structure-Activity Relationships (QSAR) to mine experimental data for patterns that relate the chemical structure of a drug to its kinase activity. The architecture of the solution is presented in **Figure** 3.
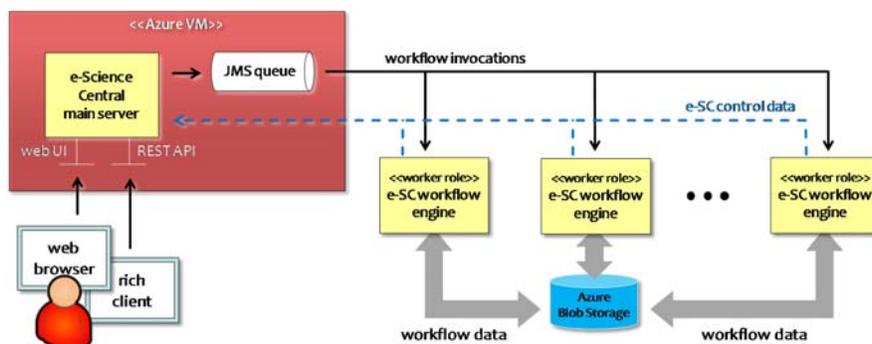

Figure 3. Architecture of the drug discovery system based on e-Science Central and Azure.

Workflows are modeled and stored within the e-Science Central main server which is the central coordination point for all workflow engines. The server dispatches work to a single JMS queue from which it is fetched by the engines. For every input data set the system issues a single top-level workflow invocation, which then results in a number of sub-workflow invocations. Altogether a single input dataset generates from 50 to 120 workflow invocations. The basic unit of work in their system — a workflow invocation — contains a number of tasks. After an engine accepts an invocation from the queue, it executes all included tasks. A task can be as simple as downloading data from blob storage or transposing a data matrix, or as complex as building a QSAR model with neural networks, which can consume over 1 CPU hour.

The Newcastle e-Science Central system also illustrates a two important characteristic of many cloud based scientific systems. This allows the tasks to be spread over 300 cores, with greater than 90% efficiency. The user input is through a web service that can allow multiple users to invoke the same instantiation of the service at the same time. This interactive model is far different from the traditional batch approach used in supercomputing facilities.
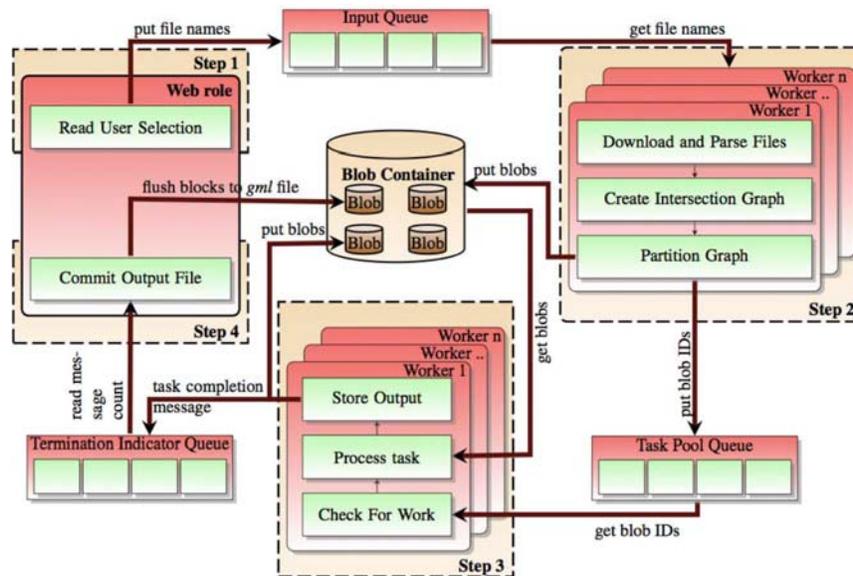
7.2. Georgia State University

Figure 4. *Crayons* GIS vector overlay processing architecture.

This basic programming model which builds the system as a web service that takes user input from the web or rich client to a server in the cloud and then allows a pool of worker servers take work that is queued by the web server is natural for the cloud. The *Crayons* project lead by Sushil K. Prasad of Georgia State University uses a similar approach to doing vector overlay computations for geographic information systems [59-61]. As shown in Figure 4, the system takes GML files and partitions them into the appropriate sub-domain tasks which are enqueued for worker servers to process. Data is stored in the cloud data storage and pulled to the workers as needed.

The Crayons project has a fixed workflow in which the tasks are distributed over a pool of workers. *Crayons* is the first distributed GIS system over cloud capable of end-to-end spatial overlay analysis. It scales well for sufficiently large data sets, achieving end-to-end speedup of over 40-fold employing 100 Azure processors.

7.3. The Microsoft Research - University of Trento Centre for Computational and Systems Biology

Angela Sanger, Michele Di Cosmo and Corrado Priami at COSBI have been investigating the behavior of p53, one of the most important transcription factors in the genome. They have developed a model that includes all known reactions between p53 and DNA, and are fitting this complex model [62] using experimental data obtained in different conditions and using different mutants of p53. COSBI has developed BetaSIM, a simulator, driven by BlenX - a stochastic, process algebra based programming language for modeling and simulating biological systems as well as other complex dynamic systems which they have ported to the cloud. AzureBetaSIM is a data-flow driven parallelization on Windows Azure of the BetaSIM simulator. The aim of AzureBetaSIM is to provide researchers with the ability to quickly run (despite the length of the job queue) a large number of concurrent simulations and, in return, quickly gather research data. More importantly, this approach enables the execution of complex flows based on the aggregation of a large number of identical or slightly different models that, because of the stochastic nature of BetaSIM, will produce different outputs. In AzureBetaSIM, the user can alter the flow dynamically based on the previous outputs and ask the researcher remotely to continue, stop or alter manually the model before a new step by providing a "sparkle" python script to control the high-level flow of execution as in Figure 5. All jobs in this use case are scripted, and the scripts can be provided by the user or generated automatically by another script.
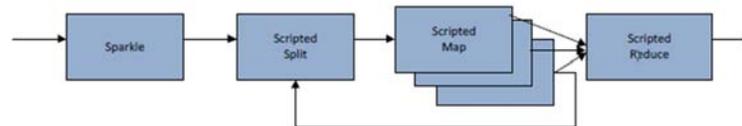

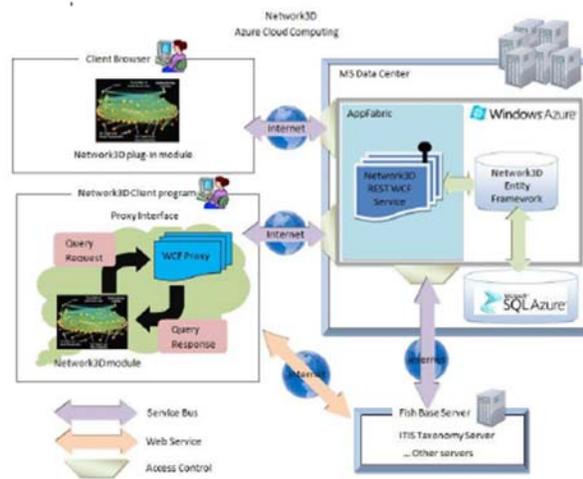**Figure 5**. Scripted dataflow execution in COSBI's AzureBetaSIM.

7.4. The University of North Carolina at Charlotte

Zhengchang Su, Srinivas Arkela and Youjie Zhou, from the Department of Bioinformatics & Genomics and Computer Science at The University of North Carolina at Charlotte are using a similar mapreduce style workflow to annotate regulatory sequences in sequenced bacterial genomes using comparative genomics-based algorithms. Regulatory sequences specify when, how much, and where the genes should be expressed in the cell through their interactions with proteins called transcription factors (TFs). They have built the system using the basic Web Role – Worker Role programming model native to Azure. Web roles are used as interfaces between the system and the users. After jobs are submitted by the users, web roles build job messages and send them through Azure's queues. Worker roles automatically

pull messages from the queues and perform real tasks.   The have also implemented this same system on top of Hadoop.

7.5. Pacific Ecoinformatics and Computational Ecology Lab (Berkeley, CA) and the Santa Fe Institute

Jennifer Dunne, Sanghyuk Yoon and Neo Martinez from the Pacific Ecoinformatics and Computational Ecology Lab (Berkeley, CA) and the Santa Fe Institute are looking at the grand challenge in the science of ecology of explaining and predicting the behavior of complex ecosystems comprised of many interacting species. The long-term scientific goal is the development of theory that accurately predicts the response of different species and whole ecosystems to physical, biological and chemical changes. The team has developed a tool called Network3D which is used to simulate the complex non-linear systems that characterize these problems. They have ported Network3D to Azure [63]. The Network3D engine uses Windows Workflow Foundation to implement long-running processes as workflows. Given requests which contain a number of manipulations, each manipulation is delivered to a worker instance to execute. The result of each manipulation is saved to SQL Azure. A web role provides the user with an interface where the user can initiate, monitor and manage their manipulations as wells as web services for other sites and visualization clients. Once the request is submitted through the web role interface, the manipulation workflow starts the task and a manipulation is assigned to an available worker to process. The Network3D visualization client communicates through web services and visualizes the ecological network and population dynamics results.

7.6. University of Washington Baker Lab

A classic example of embarrassingly parallel computation in science is the Seti@home [64].   This is based on volunteer computing where thousands of people make their pc available for an external agent to download tasks to it when it is not being heavily used.

A standard framework for these applications is BOINC from Berkeley.   In David Baker's lab at the University of Washington, they have built a protein folding application (Rosetta@home) based on BOINC.   The problem with traditional volunteer computing is that volunteers are not very reliable. On the other hand, the cloud can be considered a very large pool of capability that can easily be turned into a
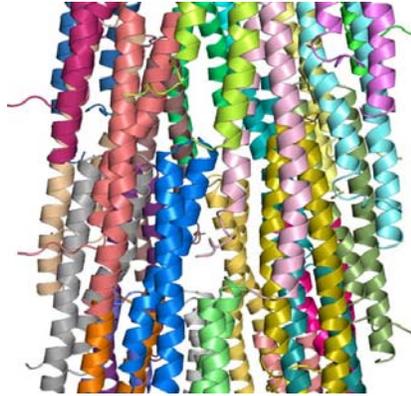
"high thoughput" BOINC service [65]. To demonstrate this they used 2000 Azure cores to run a substantial folding challenge provided by Dr. Nikolas Sgourakis, a postdoc in the lab. Specifically the challenge was to elucidate the structure of a molecular machine called the needle complex, which is involved in the transfer between cells of dangerous bacteria, such as salmonella, e-coli, and others. One of the main advantages of using Azure is that they did'nt have to handle support of the system, a very common problem with

Rosetta@home. Of course on a real volunteer system the price is free.   Consequently one has a choice.   You can get the job done quickly, or you can get it for free.

7.7. University of Nottingham

A novel use of the cloud for science was provided by Dominic Price at the University of Nottingham as part the Horizon Digital Britain project to Horizon research focuses on the role of "always on, always with you" ubiquitous computing technology. Their use of Azure was to build support for crowd-sourcing [66].   Specifically they are building a "marketplace" for crowd-sourcing activities. This takes the form of a toolkit that provides an infrastructure in which crowd-sourcing modules that support a particular activity can be developed and then different crowd-sourcing workflows constructed by combining different modules. These modules can then be shared with other users to facilitate their crowd-sourcing activities. This enables non-programmers to reuse existing modules in creating new crowd-sourcing applications. At the heart of the toolkit is the crowd-sourcing factory which is an application running in the cloud, providing the front-end interface for crowd-sourcing administrators (the user group which requests crowd-sourcing activities) and developers (the user group which develops crowd-sourcing modules). This allows the administrators and developers to log in and create crowd-sourcing modules and create an activity from selected modules. Once the workflow has been defined, the factory generates a crowd-sourcing instance, a separate application that is sandboxed from the factory and all other crowd-sourcing instances. This instance contains all of the necessary functionality for recruiting and managing crowd-sourcing participants as well as some storage for storing the results of the activity to be retrieved by the administrator once the activity ends.

7.8. Old Dominion University

Harris Wu, Kurt Maly and Mohammad Zubair of Old Dominion University are developing a web-based system (FACET) that allows users to collaboratively organize multimedia collections into an evolving faceted classification. The FACET system includes a wiki-like interface that allows users manually classify documents into their personal document hierarchies as well as the global faceted classification schema, and backend algorithms that automatically classify documents. Evaluating FACET with millions of documents and thousands of users allows them to answer research questions specific to large-scale deployment of social systems that harness and cultivate collective intelligence. For example, how to merge thousands of users' individual document hierarchies into a global schema? How to build a knowledge map of thousands of experts in different domains?

Users of the FACET system are served by multiple Web Role instances [67]. Browsing and classification are supported by queries to SQL Azure. Evaluation has shown that a single Web Role instance of medium size (2 core processors, 3.5GB RAM) can support ~200 concurrent users. The backend classification algorithms run on Worker Role instances. One of the most computing intensive backend procedures is to compute the similarities (both textual and structural) among user-created categories. By dividing the work into 2 extra-large instances, the pair-wise similarity computation of over 10,000 user categories can be computed with 24 hours.

The FACET system is a research prototype built on Joomla, a popular open-source content management system, and originally on the LAMP stack: Linux, Apache, MySQL and PHP. To deploy the FACET system on Azure, we had to move the metadata repository and session management from MySQL to SQL Azure. To continue to benefit from evolving features in the Joomla community, however, we chose to keep MySQL to support Joomla's core non-data intensive features.

7.9. Virginia Tech
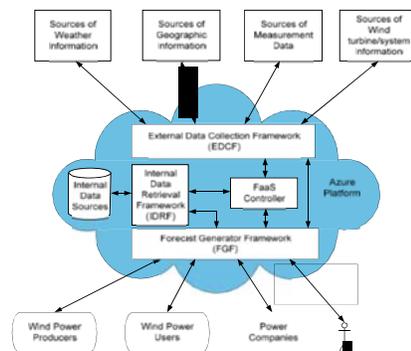
Kwa-Sur Tam of Virginia Tech have been developing a "Forecast-as-a-Service (FaaS) Framework for Renewable Energy Sources" such as wind and solar. To generate wind power forecasts at specific locations, additional data such as orography, land surface condition, wind turbine characteristics, etc., need to be obtained from multiple sources. In addition to the diversity of the types of data and the sources of data, there are different

forecasting models that have been developed using different approaches. A goal of the project is to support on-demand delivery of forecasts of different types and at different levels of detail for different prices. The FaaS framework consists of the Forecast Generation Framework (FGF), the Internal Data Retrieval Framework (IDRF), the External Data Collection Framework (EDCF) and the FaaS controller. The FGF, IDRF and EDCF all adopt service-oriented architecture (SOA) and the activities of these three frameworks are orchestrated by the FaaS controller. Since Windows Azure and the associated .NET technologies support the implementation of service- oriented architecture and its design principles, this project can focus on achieving its goals rather than dealing with the underlying support infrastructure.

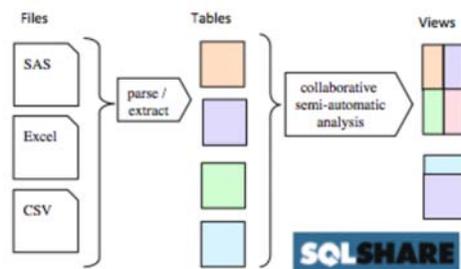*7.10. The University South Carolina and the University of Virginia*

Jon Goodall at the University of South Carolina and Marty Humphrey at the University of Virginia are creating a cloud-enabled hydrologic model and data processing workflows to examine the Savannah River Basin in the Southeastern United States [68]. Understanding hydrologic systems at the scale of large watersheds is critically important to society when faced with extreme events, such as floods and droughts, or with concern about water quality. This project will advance hydrologic science and our ability to manage water resources. Today, models that are used for engineering analysis of hydrologic systems are insufficient for addressing current water resource challenges such as the impact of land use change and climate change on water resources. The challenge being faced extends beyond modeling and includes the entire workflow from data collection to decision making. The project is being built in three stages. First they will create a cloud-enabled hydrologic model. Second, they will improve the process of hydrologic model parameterization by creating cloud-based data processing workflows. Third, in Windows Azure, they will apply the model and data processing tool to a large watershed.

They plan to use Windows Azure for data preparation, model calibration, and large-scale model execution. To date, they have focused on model calibration, whose goal is to search the space of potential model parameters for the best match against observed data. In their system the user uploads her model and then specifies the set of parameters and range of values for each parameter, effectively defining the search space. Their architecture is based on cloudbursting, which uses the Microsoft HPC support extensively. When a user submits a job, the first resources sought are the Microsoft HPC Cluster at the University of Virginia. When there is too much work in the queue, they "burst" onto Windows Azure. This architecture has shown to be very flexible, and they are able to spin up new compute nodes in Windows Azure (and shut them down) whenever they desire. Within Windows Azure, they use Blob storage to prepare the nodes – i.e., when they boot, they automatically load our generic watershed modeling code. They use the Windows Azure Virtual Private Network (VPN) support to make

the Windows Azure nodes appear to be local to their enterprise – they have found that this has greatly simplified the use of the cloud.

7.11. The University of Washington

Bill Howe, Garret Cole, Alicia Key, Nodira Khoussainova and Leilani Battle of the University of Washington have developed "SQLShare: Database-as-a-Service for Long Tail Science" [69, 70]. This project addresses a growing problem in science involving the ability of researchers to save, share and query the data results from scientific research. Spreadsheets and ASCII files remain the most popular tools for data management, especially in the long tail. But as data volumes continue to explode, cut-and-paste manipulation of spreadsheets cannot scale, and the relatively cumbersome development cycle of scripts and workflows for ad hoc, iterative data manipulation becomes the bottleneck to scientific discovery and a fundamental barrier to those without programming experience. SQLShare (http://sqlshare.escience.washington.edu) is cloud-based relational data sharing and analysis platform that allows users to upload their data and immediately query it using SQL — no schema design, no reformatting, no DBAs. These queries can be named, associated with metadata, saved as views, and shared with collaborators.

The SQLShare platform is implemented as an Azure Web Role that issues queries against a SQL Azure back end. The Azure Web Role implements a REST API and manages communication with the database, enforcing SQLShare semantics when they differ from conventional databases. In particular, the Web Role manages fault-tolerant and incremental upload of large datasets, analyzes the uploaded data to infer types and recommend example queries, manages authentication with external authentication services, operates on the system catalog, parses and formats data for interoperability with external systems, provides asynchronous semantics for all operations that may operate on large datasets, and handles all REST requests. Windows Azure was essential to the success of the project by empowering a single developer to build, deploy, and manage a production-quality web service.

7.12. Kyoto University

Daisuke Kawahara and Sadao Kurohashi of Kyoto University have been developing a search engine infrastructure, TSUBAKI [71], which is based on deep Natural Language Processing. While most of conventional search engines register only words to their

indices, TSUBAKI provides a framework that indexes synonym relations, hypernym-hyponym relations, dependency/case/ellipsis relations and so forth. These indices enable TSUBAKI to capture the semantic matching between a given query and documents more precisely and flexibly. Case/ellipsis relations have not been indexed in a large scale because the speed of these analyses is not fast enough due to the necessity of referring to a large database of predicate-argument patterns (case frames). To apply case/ellipsis analysis to millions of Web pages of TSUBAKI in a practical time, it is necessary to use 10,000 CPU cores. Because of limits on the Azure fabric controller, it was necessary to divide this into 29 hosted services of 350 CPUs each. This was the largest experiment of any of the research engagement projects.

8. Conclusions

We have discussed both general principles and some specific examples – mainly in the last section. There are many other important examples that have omitted including those on Amazon [72] and Google platforms [73, 74]. Particular technical computing cloud uses are graph and text mining [75, 76] (see also subsections 7.7, 7.8, 7.12), Seismic processing [77], Earth and Environment Science including Geographical Information systems [68, 78-80] (see also subsections 7.2, 7.5, 7.10), Particle Physics data analysis [81], Astrophysics [82] and Astronomy [83], Chemistry with NAMD [84] application, and NASA applications [45, 85, 86].

Subsections 7.1, 7.3, 7.4 and 7.6 describe Life Science applications in the cloud. This is a particularly important area as it is both rapidly growing in needed computational resources [87, 88] and very well suited to clouds [89-91]. Two large collaborations make major use of cloud technology; iPlant [92] for the NSF Plant Biology community and Kbase [93] for DoE systems biology. Genomics has important cloud applications including Blast [94-97], Crossbow [98], a novel hybrid MapReduce for secure processing [99], and MapReduce [100-105]. This field and others show the clear importance of the long tail [15] of science where an interesting approach is Datascope linking Excel to the cloud[106] (see also sections 4 and 7.11).

9. Acknowledgements

References

1. Dennis Gannon. *Research as a Service: Cloud Computing Accelerates Scientific Discovery*. [accessed 2012 December 31]; Available from: http://blogs.technet.com/b/microsoft_on_the_issues/archive/2012/12/19/research-as-a-service-cloud-computing-accelerates-scientific-discovery.aspx.
2. Gupta, A. and D. Milojicic. *Evaluation of hpc applications on cloud*. in *Open Cirrus Summit (OCS), 2011 Sixth* 2011: IEEE.
3. Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. *Cloud computing and grid computing 360-degree compared*. in *Grid Computing Environments Workshop, 2008. GCE'08* 2008: Ieee.
4. Paladugu, K. and S. Mukka, *Systematic Literature Review and Survey on High Performance Computing in Cloud*. Electrical Engineering, 2012.
5. Wlodarczyk, T.W. and C. Rong. *An Initial Survey on Integration and Application of Cloud Computing to High Performance Computing*. in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on* 2011: IEEE.
6. Barga, R., D. Gannon, and D. Reed, *The client and the cloud: Democratizing research computing*. Internet Computing, IEEE, 2011. 15(1): p. 72-75.
7. Katherine Yelick, Susan Coghlan, Brent Draney, and Richard Shane Canon, *The Magellan Report on Cloud Computing for Science*. December 2011. http://www.nersc.gov/assets/StaffPublications/2012/MagellanFinalReport.pdf.
8. Manish Parashar. *Can clouds transform science?* 2012 August 15 [accessed 2013 January 2]; International Science Grid This Week: Interview by Wolfgang Gentzsch Available from: http://www.isgtw.org/feature/can-clouds-transform-science.
9. Geoffrey C. Fox, Gregor von Laszewski, Javier Diaz, Kate Keahey, Jose Fortes, Renato Figueiredo, Shava Smallen, Warren Smith, and Andrew Grimshaw, *FutureGrid - a reconfigurable testbed for Cloud, HPC and Grid Computing*, Chapter in *On the Road to Exascale Computing: Contemporary Architectures in High Performance Computing*, Jeff Vetter, Editor. 2012, Chapman & Hall/CRC Press http://grids.ucs.indiana.edu/ptliupages/publications/sitka-chapter.pdf
10. Jeffrey Dean and Sanjay Ghemawat, *MapReduce: simplified data processing on large clusters*. Commun. ACM, 2008. 51(1): p. 107-113. DOI:http://doi.acm.org/10.1145/1327452.1327492
11. Microsoft. *Windows Azure and the Private Cloud*. [accessed 2013 January 2]; Available from: http://msdn.microsoft.com/en-us/library/jj136831.aspx.
12. Thilina Gunarathne, Bingjing Zhang, Tak-Lon Wu, and Judy Qiu, *Portable Parallel Programming on Cloud and HPC: Scientific Applications of Twister4Azure*, in *IEEE/ACM International Conference on Utility and Cloud Computing UCC 2011*. December 5-7, 2011. Melbourne Australia. http://www.cs.indiana.edu/~xqiu/scientific_applications_of_twister4azure_ucc_17_4.pdf.
13. Thilina Gunarathne, Bingjing Zhang, Tak-Lon Wu, and Judy Qiu, *Scalable Parallel Computing on Clouds Using Twister4Azure Iterative MapReduce* Future Generation Computer Systems 2012. To be published. http://grids.ucs.indiana.edu/ptliupages/publications/Scalable_Parallel_Computing_on_Clouds_Using_Twister4Azure_Iterative_MapReduce_cr_submit.pdf
14. Thilina Gunarathne, Judy Qiu, and Geoffrey Fox, *Iterative MapReduce for Azure Cloud* in *CCA11 Cloud Computing and Its Applications*. April 12-13, 2011. Chicago, ILL. http://grids.ucs.indiana.edu/ptliupages/publications/cca_v8.pdf.
15. P. Bryan Heidorn, *Shedding Light on the Dark Data in the Long Tail of Science*. Library Trends, 2008. 57(2, Fall 2008): p. 280-299. DOI:10.1353/lib.0.0036
16. Peter Murray-Rust. *Big Science and Long-tail Science*. 2008 January 29 [accessed 2011 November 3]; Available from: http://blogs.ch.cam.ac.uk/pmr/2008/01/29/big-science-and-long-tail-science/.
17. Taylor, I.J., E. Deelman, D.B. Gannon, and M. Shields, *Workflows for e-Science: Scientific Workflows for Grids*. 2006: Springer. ISBN:1846285194
18. Gannon, D. and G. Fox, *Workflow in Grid Systems, Editorial of special issue of Concurrency&Computation:Practice&Experience based on GGF10 Berlin meeting*. Concurrency and Computation: Practice & Experience, August 2006, 2006. 18(10): p. 1009-1019. DOI:http://dx.doi.org/10.1002/cpe.v18:10. http://grids.ucs.indiana.edu/ptliupages/publications/Workflow-overview.pdf
19. Deelman, E., D. Gannon, M. Shields, and I. Taylor, *Workflows and e-Science: An overview of workflow system features and capabilities*. Future Generation Computer Systems, May 2009, 2009. 25(5): p. 528-540. DOI:http://dx.doi.org/10.1016/j.future.2008.06.012

20.   Yolanda Gil, Ewa Deelman, Mark Ellisman, Thomas Fahringer, Geoffrey Fox, Dennis Gannon, Carole Goble, Miron Livny, Luc Moreau, and Jim Myers, *Examining the Challenges of Scientific Workflows.* Computer, 2007. 40(12): p. 24-32. DOI:10.1109/mc.2007.421

21.   Geoffrey Fox, Alex Ho, and Eddy Chan, *Measured Characteristics of FutureGrid Clouds for Scalable Collaborative Sensor-Centric Grid Applications*, in *International Symposium on Collaborative Technologies and Systems CTS 2011*, Waleed Smari, Editor., 2011, IEEE. Philadelphia. http://grids.ucs.indiana.edu/ptliupages/publications/cts_2011_paper_mod_6%5B1%5D.pdf.

22.   Geoffrey Fox, Alex Ho, Eddy Chan, and William Wang, *Measured Characteristics of Distributed Cloud Computing Infrastructure for Message-based Collaboration Applications*, in *International Symposium on Collaborative Technologies and Systems CTS 2009*. May 18-22, 2009, IEEE. The Westin Baltimore Washington International Airport Hotel Baltimore, Maryland, USA. pages. 465-467. http://grids.ucs.indiana.edu/ptliupages/publications/SensorClouds.pdf. DOI: 10.1109/cts.2009.5067515.

23.   *Cloud Robotics summary of work in this field*.   [accessed 2012 December 4]; Available from: http://goldberg.berkeley.edu/cloud-robotics/.

24.   James Kuffner, *ROBOTS WITH THEIR HEADS IN THE CLOUDS*, in *Discovery News*. March 1, 2011. http://news.discovery.com/tech/robots-with-their-heads-in-the-clouds-110301.html.

25.   Geoffrey Fox, Tony Hey, and Anne Trefethen, *Where does all the data come from?* , Chapter in *Data Intensive Science* Terence Critchlow and Kerstin Kleese Van Dam, Editors. 2011. http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf.

26.   Dave, A., W. Lu, J. Jackson, and R. Barga. *CloudClustering: Toward an iterative data processing pattern on the cloud*. in *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on* 2011: IEEE.

27.   Faris, J., E. Kolker, A. Szalay, L. Bradlow, E. Deelman, W. Feng, J. Qiu, D. Russell, E. Stewart, and E. Kolker, *Communication and data-intensive science in the beginning of the 21st century.* Omics: A Journal of Integrative Biology, 2011. 15(4): p. 213-215. http://www.ncbi.nlm.nih.gov.offcampus.lib.washington.edu/pubmed/21476843

28.   Gordon Bell, Tony Hey, and Alex Szalay, *COMPUTER SCIENCE: Beyond the Data Deluge.* Science, 2009. 323(5919): p. 1297-1298. DOI:10.1126/science.1170411

29.   Gray, J., *Jim Gray on eScience: A transformed scientific method*, Chapter in *The Fourth Paradigm: Data-intensive Scientific Discovery.* 2009, Microsoft Reasearch. p. xvii-xxxi.

30.   Gordon Bell, Jim Gray, and Alex Szalay, *Petascale Computational Systems: Balanced CyberInfrastructure in a Data-Centric World (Letter to NSF Cyberinfrastructure Directorate).* IEEE Computer, January, 2006. 39(1): p. 110-112. http://research.microsoft.com/en-us/um/people/gray/papers/Petascale%20computational%20systems.doc

31.   Jim Gray, Tony Hey, Stewart Tansley, and Kristin Tolle. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. 2010 [accessed 2010 October 21]; Available from: http://research.microsoft.com/en-us/collaboration/fourthparadigm/.

32.   Pan, A., J.P. Walters, V.S. Pai, D.I.D. Kang, and S.P. Crago, *Integrating High Performance File Systems in a Cloud Computing Environment.*

33.   Ghoshal, D., R.S. Canon, and L. Ramakrishnan. *I/O performance of virtualized cloud environments*. in *Proceedings of the second international workshop on Data intensive computing in the clouds* 2011: ACM.

34.   Jaliya Ekanayake and Geoffrey Fox, *High Performance Parallel Computing with Clouds and Cloud Technologies*, in *First International Conference CloudComp on Cloud Computing*. October 19 - 21, 2009. Munich, Germany. http://grids.ucs.indiana.edu/ptliupages/publications/cloudcomp_camera_ready.pdf.

35.   Constantinos Evangelinos and Chris N. Hill, *Cloud Computing for parallel Scientific HPC Applications: Feasibility of running Coupled Atmosphere-Ocean Climate Models on Amazon's EC2.*, in *CCA08: Cloud Computing and its Applications*. October, 2008. Chicago ILL USA. http://www.cca08.org/papers/Paper34-Chris-Hill.pdf.

36.   Edward Walker, *Benchmarking Amazon EC2 for High Performance Scientific Computing*, in *USENIX*. Oct 2008, 2008: Vol. vol. 33(5). http://www.usenix.org/publications/login/2008-10/openpdfs/walker.pdf.

37.   Expósito, R.R., G.L. Taboada, S. Ramos, J. Touriño, and R. Doallo, *Performance analysis of HPC applications in the cloud.* Future Generation Computer Systems, 2012.

38.   Gómez, J., E. Villar, G. Molero, and A. Cama, *Evaluation of High Performance Clusters in Private Cloud Computing Environments.* Distributed Computing and Artificial Intelligence, 2012: p. 305-312.

39. Keith R. Jackson, Lavanya Ramakrishnan, Krishna Muriki, Shane Canon, Shreyas Cholia, J. Shalf, Harvey J. Wasserman, and N.J. Wright, *Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud*, in *CloudCom*. 2010, IEEE. Indianapolis. http://www.nersc.gov/projects/reports/technical/CloudCom.pdf.

40. Roloff, E., F. Birck, M. Diener, A. Carissimi, and P.O.A. Navaux. *Evaluating High Performance Computing on the Windows Azure Platform*. in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on* 2012: IEEE.

41. Shajulin Benedict, *Performance issues and performance analysis tools for HPC cloud applications: a survey*. Computing, 2012: p. 1-20.

42. Strazdins, P.E., J. Cai, M. Atif, and J. Antony. *Scientific Application Performance on HPC, Private and Public Cloud Resources: A Case Study Using Climate, Cardiac Model Codes and the NPB Benchmark Suite*. in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International* 2012: IEEE.

43. Tudoran, R., A. Costan, G. Antoniu, and L. Bougé. *A performance evaluation of Azure and Nimbus clouds for scientific applications*. in *Proceedings of the 2nd International Workshop on Cloud Computing Platforms* 2012: ACM.

44. Gupta, A., D. Milojicic, and L.V. Kalé. *Optimizing VM Placement for HPC in the Cloud*. in *Proceedings of the 2012 workshop on Cloud services, federation, and the 8th open cirrus summit* 2012: ACM.

45. Knight, D., K. Shams, G. Chang, and T. Soderstrom. *Evaluating the efficacy of the cloud for cluster computation*. in *Aerospace Conference, 2012 IEEE*. 3-10 March 2012 2012.

46. Ekanayake, J., A.S. Balkir, T. Gunarathne, G. Fox, C. Poulain, N. Araujo, and R. Barga, *DryadLINQ for Scientific Analyses*, in *Fifth IEEE International Conference on eScience: 2009*. 2009, IEEE. Oxford. http://grids.ucs.indiana.edu/ptliupages/publications/eScience09-camera-ready-submission.pdf.

47. Jaliya Ekanayake, Thilina Gunarathne, Judy Qiu, Geoffrey Fox, Scott Beason, Jong Youl Choi, Yang Ruan, Seung-Hee Bae, and Hui Li, *Applicability of DryadLINQ to Scientific Applications*. January 30, 2010, Community Grids Laboratory, Indiana University. http://grids.ucs.indiana.edu/ptliupages/publications/DryadReport.pdf.

48. Hui Li, Ruan Yang, Yuduo Zhou, and Judy Qiu, *DRYADLINQ CTP EVALUATION: Performance of Key Features and Interfaces in DryadLINQ CTP (Revised)*. December 14, 2011. http://grids.ucs.indiana.edu/ptliupages/publications/DradLinqCTPEvaluation_revised.pdf.

49. Hui Li, Geoffrey Fox, and J. Qiu, *Performance Model for Parallel Matrix Multiplication with Dryad: Dataflow Graph Runtime*, in *2012 International Symposium on Big Data and MapReduce (BigDataMR2012)*. 01-03 November, 2012. Xiangtan, Hunan, China. http://grids.ucs.indiana.edu/ptliupages/publications/BigDataMR%232.pdf.

50. Hui Li, Yang Ruan, Yuduo Zhou, Judy Qiu, and Geoffrey Fox, *Design Patterns for Scientific Applications in DryadLINQ CTP*, in *The Second International Workshop on Data Intensive Computing in the Clouds (DataCloud-SC11) at SC11*. November 14, 2011. http://grids.ucs.indiana.edu/ptliupages/publications/Design%20Patterns%20for%20Scientific%20Applications%20in%20DryadLINQ%20CTP%20ACM%20Data307.pdf.

51. *Apache Hadoop*. [accessed 2010 November 27]; Available from: http://hadoop.apache.org/.

52. Expósito, R.R., G.L. Taboada, S. Ramos, J. Touriño, and R. Doallo, *General‐purpose computation on GPUs for high performance cloud computing*. Concurrency and Computation: Practice and Experience, 2012.

53. *Microsoft Azure Cloud Research Projects*. [accessed 2012 December 31]; Available from: http://research.microsoft.com/en-us/projects/azure/projects.aspx.

54. *VENUS-C Virtual multidisciplinary EnviroNments USing Cloud infrastructure*. June 2010 Available from: http://www.venus-c.eu/.

55. Simmhan, Y., C. van Ingen, G. Subramanian, and J. Li. *Bridging the Gap between Desktop and the Cloud for eScience Applications*. in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on* 2010: IEEE.

56. *Grid Computing and Clusters at the Barcelona Supercomputer Center*. [accessed 2013 January 1]; Available from: http://www.bsc.es/computer-sciences/grid-computing.

57. Cała, J., H. Hiden, S. Woodman, and P. Watson, *Fast Exploration of the QSAR Model Space with e‐Science Central and Windows Azure.*

58. Watson, P., H. Hiden, and S. Woodman, *e‐Science Central for CARMEN: science as a service.* Concurrency and Computation: Practice and Experience, 2010. 22(17): p. 2369-2380.

59. Agarwal, D. and S.K. Prasad. *Lessons Learnt from the Development of GIS Application on Azure Cloud Platform*. in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on* 2012: IEEE.

60. Agarwal, D. and S.K. Prasad. *Azurebench: Benchmarking the storage services of the Azure Cloud Platform*. in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International* 2012: IEEE.

61. Agarwal, D., S. Puri, X. He, S.K. Prasad, and X. Shi, *Crayons-a cloud based parallel framework for GIS overlay operations*.

62. Romanel, A., C. Priami, and L. Dematte, *The beta workbench*. 2007.

63. Martinez, N.D., P. Tonin, B. Bauer, R.C. Rael, R. Singh, S. Yoon, I. Yoon, and J.A. Dunne, *Sustaining Economic Exploitation of Complex Ecosystems in Computational Models of Coupled Human-Natural Networks*. Proceedings of The Association for the Advancement of Artificial Intelligence, 2012.

64. *SETI@Home*. [accessed 2013 January 1]; Available from: http://setiathome.berkeley.edu/.

65. Thompson, J.M., N.G. Sgourakis, G. Liu, P. Rossi, Y. Tang, J.L. Mills, T. Szyperski, G.T. Montelione, and D. Baker, *Accurate protein structure modeling using sparse NMR data and homologous structure information*. Proceedings of the National Academy of Sciences, 2012. 109(25): p. 9875-9880.

66. Price, D., C. Greenhalgh, S. Ali, J. Goulding, and R. Mortier, *A Framework for Crowd-Sourcing Personal Data*.

67. Maly, K. *Collaborative Digital Library Services in a Cloud*. in *SERVICE COMPUTATION 2010, The Second International Conferences on Advanced Service Computing* 2010.

68. Humphrey, M., N. Beekwilder, J.L. Goodall, and M.B. Ercan, *Calibration of Watershed Models using Cloud Computing*.

69. Howe, B., G. Cole, E. Souroush, P. Koutris, A. Key, N. Khoussainova, and L. Battle. *Database-as-a-service for long-tail science*. in *Scientific and Statistical Database Management* 2011: Springer.

70. Key, A., B. Howe, D. Perry, and C. Aragon. *VizDeck: self-organizing dashboards for visual analytics*. in *Proceedings of the 2012 international conference on Management of Data* 2012: ACM.

71. Keiji Shinzato, Tomohide Shibata, Daisuke Kawahara, and Sadao Kurohashi, *TSUBAKI: An Open Search Engine Infrastructure for Developing Information Access Methodology*. Journal of Information Processing, 2012. 20(1): p. 216-227.

72. Amazon AWS. *Scientific Computing Using Spot Instances*. [accessed 2013 January 1]; Available from: http://aws.amazon.com/ec2/spot-and-science/.

73. Joseph Hellerstein, M.B.S., Google,. *Keynote on Science in the Cloud at Cloud Futures Workshop 2012*. 2012 [accessed 2013 January 1]; May 7–8, 2012 at Berkeley, California, United States Available from: http://research.microsoft.com/en-US/events/cloudfutures2012/agenda.aspx.

74. Andrea Held. *Millions of Core-Hours Awarded to Science*. [accessed 2013 January 2]; Available from: http://googleresearch.blogspot.com/2012/12/millions-of-core-hours-awarded-to.html.

75. Z. Zhao, G. Wang, A. Butt, M. Khan, V. S. Anil Kumar, and M. Marathe, *SAHAD: Subgraph Analysis in Massive Networks Using Hadoop*, in *26th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. May, 2012, IEEE. Shanghai, China.

76. Simmhan, Y., L. Deng, A. Kumbhare, M. Redekopp, and V. Prasanna, *Scalable, Secure Analysis of Social Sciences Data on the Azure Platform*. http://ceng.usc.edu/~simmhan/pubs/redekopp-pargraph-2011.pdf

77. Subramanian, V., H. Ma, L. Wang, E.J. Lee, and P. Chen. *Rapid 3D Seismic Source Inversion Using Windows Azure and Amazon EC2*. in *Services (SERVICES), 2011 IEEE World Congress on* 2011: IEEE.

78. Chen, Z., N. Chen, C. Yang, and L. Di, *Cloud Computing Enabled Web Processing Service for Earth Observation Data Processing*. Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of, 2012. 5(6): p. 1637-1649. DOI:10.1109/jstars.2012.2205372

79. Humphrey, M., Z. Hill, K. Jackson, C. van Ingen, and Y. Ryu. *Assessing the value of cloudbursting: A case study of satellite image processing on windows azure*. in *E-Science (e-Science), 2011 IEEE 7th International Conference on* 2011: IEEE.

80. Li, J., M. Humphrey, D. Agarwal, K. Jackson, C. van Ingen, and Y. Ryu. *escience in the cloud: A modis satellite data reprojection and reduction pipeline in the windows azure platform*. in *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on* 2010: IEEE.

81. Marc-Elian Bégin, *AN EGEE COMPARATIVE STUDY: GRIDS AND CLOUDS Evolution or Revolution?*, in *EGEE Document Series*. May 300, 2008. https://edms.cern.ch/file/925013/3/EGEE-Grid-Cloud.pdf.

82. Keith R. Jackson, Lavanya Ramakrishnan, Karl J. Runge, and Rollin C. Thomas, *Seeking supernovae in the clouds: a performance study*, in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. 2010, ACM. Chicago, Illinois. pages.

421-429. http://datasys.cs.iit.edu/events/ScienceCloud2010/p07.pdf. DOI: 10.1145/1851476.1851538.

83.   Thakar, A. and A. Szalay. *Migrating a (large) science database to the cloud*. in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing* 2010: ACM.

84.   Wong, A.K.L. and A.M. Goscinski, *The Design and Implementation of the VMD Plugin for NAMD Simulations on the Amazon Cloud*. International Journal of Cloud Computing and Services Science (IJ-CLOSER), 2012. 1(4): p. 155-171.

85.   Mehrotra, P., J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, and R. Biswas. *Performance evaluation of Amazon EC2 for NASA HPC applications*. in *Proceedings of the 3rd workshop on Scientific Cloud Computing Date* 2012: ACM.

86.   Saini, S., S. Heistand, H. Jin, J. Chang, R. Hood, P. Mehrotra, and R. Biswas. *An Application-based Performance Evaluation of NASA's Nebula Cloud Computing Platform*. in *High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS), 2012 IEEE 14th International Conference on* 2012: IEEE.

87.   Stein, L., *The case for cloud computing in genome informatics*. Genome Biology, 2010. 11(5): p. 207. http://genomebiology.com/2010/11/5/207

88.   Elizabeth Pennisi, *Will Computers Crash Genomics?* Science, February 11, 2011, 2011. 331(6018): p. 666-668. DOI:10.1126/science.331.6018.666. http://www.sciencemag.org/content/331/6018/666.short

89.   Judy Qiu, Jaliya Ekanayake, Thilina Gunarathne, Jong Youl Choi, Seung-Hee Bae, Yang Ruan, Saliya Ekanayake, Stephen Wu, Scott Beason, Geoffrey Fox, Mina Rho, and Haixu Tang, *Data Intensive Computing for Bioinformatics*, Chapter in *Data Intensive Distributed Computing*, Tevik Kosar, Editor. 2011, IGI Publishers. http://grids.ucs.indiana.edu/ptliupages/publications/DataIntensiveComputing_BookChapter.pdf.

90.   Xiaohong Qiu, Jaliya Ekanayake, Scott Beason, Thilina Gunarathne, Geoffrey Fox, Roger Barga, and Dennis Gannon, *Cloud Technologies for Bioinformatics Applications*, in *2nd ACM Workshop on Many-Task Computing on Grids and Supercomputers (SuperComputing09)*. November 16th, 2009, ACM Press. Portland, Oregon. http://grids.ucs.indiana.edu/ptliupages/publications/MTAGSOct22-09A.pdf.

91.   David Heckerman. *Supercomputing on Demand with Windows Azure*. [accessed 2012 December 31]; Available from: http://blogs.msdn.com/b/msr_er/archive/2012/11/12/affordable-supercomputing-with-windows-azure.aspx.

92.   *iPlant Collaborative*. [accessed 2012 August 28]; Available from: http://www.iplantcollaborative.org.

93.   *KBASE: DOE Systems Biology Knowledgebase: Community-Driven Cyberinfrastructure for Sharing and Integrating Data and Analytical Tools to Accelerate Predictive Biolog*. [accessed 2012 August 28]; Available from: http://genomicscience.energy.gov/compbio/.

94.   Kolker, N., R. Higdon, W. Broomall, L. Stanberry, D. Welch, W. Lu, W. Haynes, R. Barga, and E. Kolker, *Classifing Proteins into Functional Groups based on ALL vs. ALL BLAST of 10 Million Proteins*. Omics: A Journal of Integrative Biology, 2011. In Press.

95.   Wei Lu, Jared Jackson, Jaliya Ekanayake, Roger Barga, and Nelson Araujo, *Performing Large Science Experiments on Azure: Pitfalls and Solutions*, in *CloudCom*. 2010, IEEE. Indianapolis. pages. 209-219.

96.   Wei Lu, Jared Jackson, and Roger Barga, *AzureBlast: A Case Study of Developing Science Applications on the Cloud*, in *ScienceCloud: 1st Workshop on Scientific Cloud Computing co-located with HPDC 2010 (High Performance Distributed Computing)*. June 21, 2010, ACM. Chicago, IL. http://dsl.cs.uchicago.edu/ScienceCloud2010/p06.pdf.

97.   Brock, M. and A. Goscinski. *Execution of Compute Intensive Applications on Hybrid Clouds (Case Study with mpiBLAST)*. in *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on* 2012: IEEE.

98.   Gurtowski, J., M.C. Schatz, and B. Langmead, *Genotyping in the Cloud with Crossbow*, Chapter in *Current Protocols in Bioinformatics*. 2012, John Wiley & Sons, Inc. http://dx.doi.org/10.1002/0471250953.bi1503s39.

99.   Yangyi Chen, Bo Peng, XiaoFeng Wang, and Haixu Tang, *Large-Scale Privacy-Preserving Mapping of Human Genomic Sequences on Hybrid Clouds*, in *The Network and Distributed System Security Symposium 2012 (NDSS'12*. February 5-8, 2012. Hilton San Diego Resort & Spa San Diego, CA United States. http://www.informatics.indiana.edu/xw7/papers/ndss2012.pdf.

100.   Schatz, M. *Cloud Computing and the DNA Data Race*. 2011 [accessed 2011 June 26]; Keynote
        Presentation at 3DAPAS/ECMLS workshops at HPDC June 8 2011 Available from:
        http://schatzlab.cshl.edu/presentations/2011-06-08.HDPC.3DAPAS.pdf.
101.   Langmead, B., M. Schatz, J. Lin, M. Pop, and S. Salzberg, *Searching for SNPs with cloud
        computing.* Genome Biology, 2009. 10(11): p. R134. http://genomebiology.com/2009/10/11/R134
102.   Michael C. Schatz, *CloudBurst: highly sensitive read mapping with MapReduce.* Bioinformatics,
        2009. 25(11): p. 1363-1369. DOI:10.1093/bioinformatics/btp236.
        http://bioinformatics.oxfordjournals.org/content/25/11/1363.abstract
103.   Langmead, B., M.C. Schatz, J. Lin, M. Pop, and S.L. Salzberg, *Searching for SNPs with cloud
        computing.* Genome Biol., 2009. 10: p. R134.
104.   Rohith K. Menon, Goutham P. Bhat, and M.C. Schatz, *Rapid Parallel Genome Indexing with
        MapReduce*, in *MapReduce 2011*. June 8 2011. San Jose.
105.   Titmus, M.A., J. Gurtowski, and M.C. Schatz, *Answering the demands of digital genomics.*
        Concurrency and Computation: Practice and Experience, 2012: p. n/a-n/a. DOI:10.1002/cpe.2925.
        http://dx.doi.org/10.1002/cpe.2925
106.   Roger Barga, Dennis Gannon, Nelson Araujo, Jared Jackson, Wei Lu, and Jaliya Ekanayake,
        Excel DataScope for Data Scientists, in *UK e-Science All Hands Meeting 2010*. 13-16 September,
        2010. Cardiff, Wales UK.
        http://www.allhands.org.uk/sites/default/files/2010/TuesT1BargaExcel.pdf.