Science in the Cloud:
Accelerating Discovery in the 21st Century

February 17, 2012

Joseph L. Hellerstein, Kai Kohlhoff, and David Konerding, Google Inc.

Abstract

Scientific discovery is in transition from a focus on data collection to an emphasis on analysis and prediction using large scale computation. These computations can be done with unused cycles in commercial Clouds if there is appropriate software support. Moving science into the Cloud will promote data sharing and collaborations that will accelerate scientific discovery in the 21st century.

1 Introduction

Recent trends in science have made computational capabilities an essential part of scientific discovery. This is often referred to as enhanced scientific discovery, or eScience. A collection of essays about the shift to eScience, The Fourth Paradigm [1], describes the evolution of experimentally driven science to collaborative, analysis-focused eScience.

eScience has been an integral part of high energy physics for several decades due to the complexity and volume of data produced by experiments. In the 1990s, eScience became central to biology because of the computational demands of sequencing the human genome. More recently, eScience has become integral to brain science for modeling neural circuits and to astronomy for simulating cosmological phenomena.

Biology provides an excellent example of how eScience contributes to scientific discovery. Much of modern biological research is about relating DNA sequences (called the genotype) to observable characteristics such as disease (called phenotypes). For example, researchers look for variations in DNA that promote cancer. The human genome has approximately three billion pairs of nucleotides, the elements that encode information in DNA. These base pairs encode common human characteristics, benign individual variations, and potential disease-causing variants. Unfortunately, it is usually the case that individual variation is much more common than disease-causing variants. So, un-

derstanding how the genome contributes to disease is much more complicated than looking at the difference between genomes. Instead, this analysis often requires detailed models of DNA-mediated chemical pathways to identify disease processes. The size of the human genome and the complexity of modeling disease processes typically requires large scale computations and often massive storage as well [2].

A common pattern in eScience is to explore many possibilities in parallel. For example, aligning a million DNA "reads" (produced by a DNA sequencer) to a reference genome can be done by aligning each read to the reference genome in parallel. The search for good models of brain activity can be done by evaluating a large number of model parameters in parallel. And, the search for supernovae can be done by analyzing different regions of the sky in parallel.

That a high degree of parallelism can advance science has been a starting point for many efforts. For example, Folding@Home[4] is a distributed computing project that enables scientists to understand the biochemical basis of several diseases. At Google, the Exacycle Project provides massive parallelism for doing science in the Cloud.

2 Harvesting Cycles for Science

Many science problems can be solved by running jobs that are structured as a large number of independently executing tasks. Such jobs are referred to as embarrassingly parallel.

The goal of the Exacycle Project is to find unused resources in the Google Cloud to run embarrassingly parallel jobs at a very large scale. This is done by creating a system that is both a simplification and a generalization of MapReduce. Exacycle simplifies MapReduce in that all Exacycle tasks are essentially mappers. This simplification enables more efficient resource management. MapReduce restricts jobs to a single cluster. Exacycle generalizes MapReduce by allowing a single job to use uses computational resources in multiple clusters. This generalization enables massive scaling for embarrassingly parallel jobs.

Google is very efficient with its use of computing resources. Even so, resource utilizations vary with time of day, day of the week, and season. For example, web users most frequently use search engines during the day, and search providers typically direct traffic to datacenters close to users to reduce latency. This leads to low utilization for clusters during (local) nighttime.

Even when resource utilizations are low, it doesn't mean that more tasks can be run in the Cloud. To understand why, note that many tasks require considerable memory or require moderate amounts of memory and CPU *in combination*. Such tasks can run in the Cloud only if there is at least one machine that satisfies all of the task's resource requirements. One way to quantify whether tasks can run is whether there are suitably sized "slots" available. For example, recent measurements of the Google Cloud indicate that there are thirteen times more slots for tasks requiring only 1 core and 4 GB of RAM than there are slots for tasks requiring 4 cores and 32 GB RAM. In general, it is much easier to find

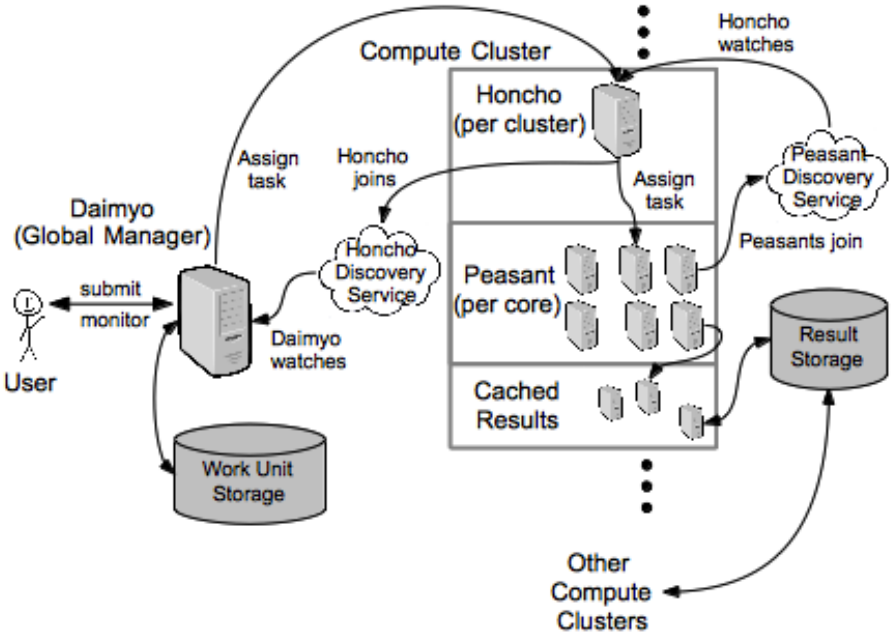

Figure 1: Architecture of the Exacycle System.

slots for tasks that require fewer resources. An Exacycle task typically consumes about 1 core and 1 GB of memory for no more than an hour.

Exacycle tasks are often preempted by higher priority work. Preempted tasks are re-run. Although task preemptions are common, Exacycle throughput is excellent because of the high degree of parallelism that Exacycle provides.

As displayed in Figure 1, Exacycle is structured into multiple layers. The top layer is the Daimyo global scheduler that assigns tasks to clusters. The second layer is the Honcho cluster scheduler that assigns tasks to machines. The third layer is the Peasant machine manager that encapsulates tasks. The bottom layer provides caching of task results. Note that both the Honcho and Peasant are stateless. This simplifies the handling of failures.

Exacycle implements the same communication interfaces between adjacent layers. Communication from an upper to a lower layer involves having the upper layer cut data into pieces that are provided to lower layers. Typically, this communication provides data to tasks within the same job. Communication from a lower layer to an upper layer involves bundling data to produce aggregations. These inter-layer interfaces have proven to be scalable and robust with minimal requirements for managing distributed state.

The primary mechanism Exacycle uses to scale is to eliminate nearly all inter-cluster networking and machine-level disk IO. An Exacycle task typically

cannot move more than 5 GB of data into or out of the machine on which the tasks executes. Exacycle reduces network usage by managing data movement on behalf of tasks. Typically, the thousands to millions of tasks in an Exacycle job share some of their input files. Exacycle uses knowledge of shared input files to co-schedule tasks in the same cluster. This strategy improves throughput by exploiting the high network bandwidths between machines within the same cluster. Further, Exacycle uses caching so that remote data are copied into a cluster only once.

When a task is assigned to a machine, there is a timeout and retry hierarchy to handle failures. The combination of timeouts and retries handles most systemic errors. Tasks have unique identifiers. The Exacycle retry logic assumes that two tasks with the same unique identifier compute the same results.

For the most part, Exacycle does not employ durable cluster-level or machine-level storage because of the engineering costs and performance penalties. Instead, Exacycle optimistically keeps nearly all state in RAM. Robustness is provided by the combination of a single authoritative store and spreading state across many machines. If an individual machine fails, the task it was executing is resubmitted to another machine. In case of failure, the cluster-level scheduler recovers state by listening to messages from a discovery service.

The Exacycle Project began two years ago. The system has been running eScience applications in production for about a year, and has had continuous, intensive use over the last six months. Recently, Google donated one billion core hours to scientific discovery through the Exacycle Visiting Faculty Grant Program[3]. To achieve this, Exacycle consumes approximately 2.7M CPU hours per day, and often much more. As of early February, visiting scientists completed 58M tasks.

Exacycle visiting faculty are addressing a variety of scientific problems that can benefit from large scale computation. The next section describes one problem in detail–discovering the operation of a trans-membrane protein that plays a critical role in many drug therapies. Below is a summary of other efforts that are underway.

- The *enzyme science* project seeks to discover how bacteria develop resistance to antibiotics, a growing problem for public health.
- The *molecular docking* project seeks to advance drug discovery by using massive computation to identify “small molecules" that bind to one or more of the huge set of proteins that catalyze reactions in cells. The potential here is to greatly accelerate the design of drugs that interfere with disease pathways.
- The *computational astronomy* project is playing an integral role in the design of the 3200 Megapixel Large Synoptic Survey Telescope. For example, the project is doing large scale simulations to determine how to correct for atmospheric distortions.
- The *molecular modeling* project is expanding the understanding of computational methods for simulating macromolecular processes. The first

application is to determine how molecules enter and leave the cell nucleus through a channel known as the nuclear pore complex.

These projects have been selected based on the opportunity to produce scientific results of major importance. One measure of impact will be publishing in top journals such as *Science* and *Nature*.

3 Simulating Molecular Dynamics

The opportunities afforded by science in the Cloud are best understood by example.

One research project undertaken by Exacycle relates to a class of molecules called G protein-coupled receptors or GPCRs. GPCRs are critical to many drug therapies. Indeed, about a third of pharmaceuticals target GCPRs. Despite this, the molecular basis of GPCR action is still not understood well.

A bit of science is needed to appreciate the computational problem that Exacycle is addressing. GPCRs are critical to trans-membrane signaling, an important part of many disease pathways. It is known that GPCRs embed in cell membranes to provide communication between extracellular signals and intracellular processes. This communication occurs when certain molecules bind to sites on GPCRs that are accessible from outside the membrane. However, scientists do not understand well the sequence of changes that then lead to communication across the cell membrane.

To gain a better understanding of GPCR activity, Exacycle is doing large scale simulations of GPCR molecular dynamics. This is a challenging undertaking because of the detail that is required to obtain scientific insight. In particular, biomolecules at body temperature undergo continuous fluctuations in the location of atoms and the molecule's three dimensional shape. Many changes occur at a time scale of femto- to nanoseconds. However, most chemical processes of interest take place at a time scale of micro- to milliseconds. The term trajectory refers to a sequence of motions of a set of atoms under study over time. Figure 2 depicts the insights possible with trajectories of different durations. Understanding GPCR actions requires simulations that generate data over milliseconds.

Exacycle simulates the trajectories of approximately 58,000 atoms, the number of atoms in a typical GPCR system, including the cell membrane and water molecules. This is done at femtosecond precision over trillions of time steps by computing trajectories using embarrassingly parallel jobs.

Trajectories are used in two ways. The first is to construct models of GPCR behavior. For example, trajectories can be used to create a Markov model whose states are defined by the pair-wise distances between atoms in the GPCR and by the kinetic proximity of atoms. Second, trajectories are analyzed for changes that are important for the activation of signaling across the cell membrane.

It takes approximately one core-day to simulate half a nanosecond of a single trajectory on a modern desktop. So, obtaining scientific insight requires mil-

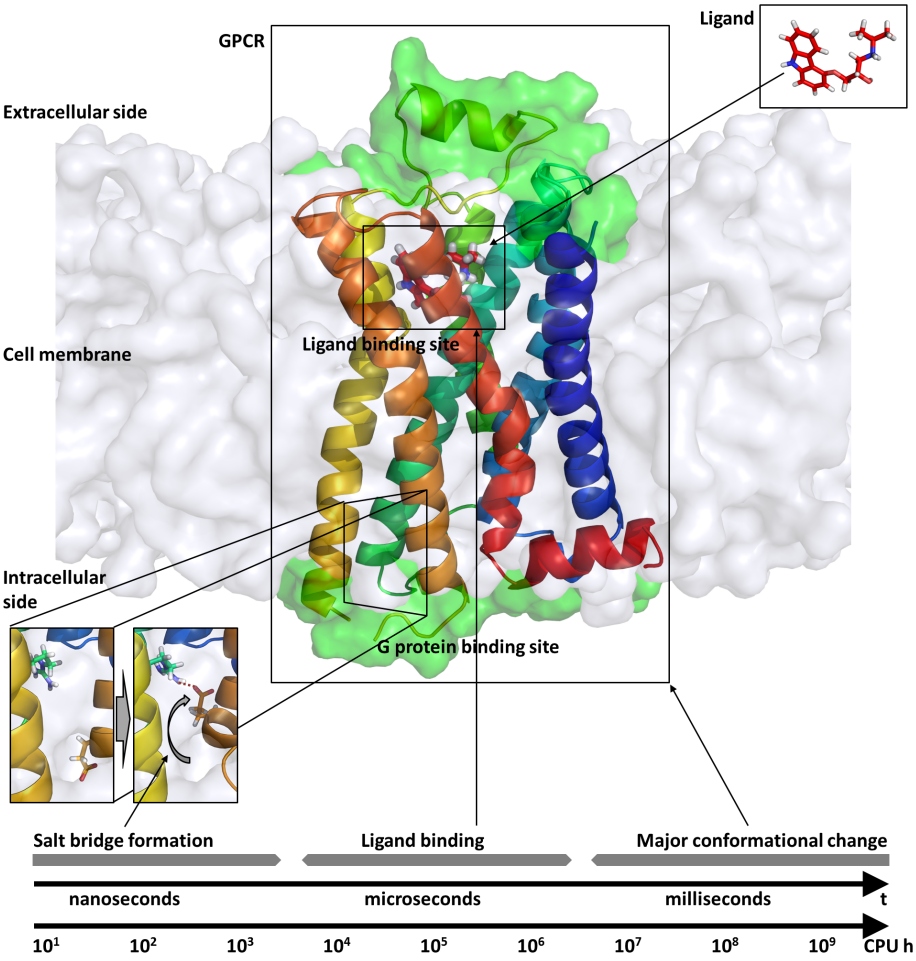

Figure 2: G protein-coupled receptors (GPCRs) are critical to the effectiveness of many drugs because of their role in communicating signals across cell membranes. The top x-axis is trajectory duration, and the figure indicates what insights are possible for trajectory durations. The bottom x-axis is the core hours needed to compute trajectory durations. Computing one millisecond of trajectory data requires millions of core days on a modern desktop computer. Exacycle can do these computations in a few days.

lions of core days to generate a millisecond of trajectory data. Clearly, massive computational resources are required.

Exacycle provides such massive computational resources to compute trajectories in parallel. However, some thought is required to use Exacycle effectively. For GPCR trajectories, the challenge is that it takes millions of core hours to compute an interesting trajectory, but an Exacycle task typically executes for no more than a core hour. So, trajectories are constructed by executing a series of task. This requires passing partially computed trajectories from one task to the next in a way that maintains high throughputs.

The approach used to compute trajectories has several parts. A driver script generates tens of thousands of tasks, and submits them to Exacycle. The script also monitors task states, and registers events such as task completions or failures. Partial trajectories computed by tasks are propagated to other tasks to ensure that Exacycle maintains high throughput. Exacycle provides mechanisms to monitor task executions and to support investigating and resolving task and system failures.

More than 150,000 trajectories with durations totaling more than 4 milliseconds have been computed. At peak, Exacycle simulates approximately 80 microseconds of trajectory data per day. This corresponds to roughly 600 TFLOP/s.

Exacycle has produced 100s of terabytes of trajectory data. The analysis of these data presents a huge challenge. One approach is to use MapReduce to calculate summary statistics of trajectories and then place the results into a relational database of trajectory tables. Much insight has been obtained by doing SQL queries against the trajectory tables. However, doing so requires the database to have the scalability of technologies such as Dremel [5] that provide interactive response times for ad hoc queries on tables with billions of rows.

4 What's Next

Amazon, Microsoft, Google, and others offer capabilities for running science applications in the Cloud. There is a good reason for this. By using the Cloud, scientists do not buy expensive, dedicated clusters. Instead, scientists pay a modest rent for on-demand access to large quantities of Cloud computing resources.

Although doing science in the Cloud has appeal, moving science into the Cloud may have hidden costs. For example, scientists may have to recode applications to take advantage of Cloud functionality and/or add new code if some features are not present in the Cloud.

Science in the Cloud offers much more than a compute infrastructure. A recent trend is that scientific contributions require the publication of large datasets. Some examples are the Allen Institute's Brain Atlas and the National Center for Biotechnology Information (NCBI) genome database. Both are repositories that are widely used by researchers to do computation-intensive analysis of data collected by others. Hosting these datasets in public Clouds is

much easier than requiring individual scientists (or even universities) to build their own data hosting systems.

There is much more to come. The use of the Cloud for computation and data storage will facilitate scientists sharing both data and computational tools. Indeed, there are already substantial efforts in this direction, such as the idea of a "Data Commons" that is promoted by Sage Bionetworks. Sharing data and code will allow scientists to more rapidly build on the results of their peers. Longer term, the big appeal of science in the Cloud is promoting planetary scale collaborations that power scientific discovery in the 21st century.

References

[1] Tony Hey, Stewart Tansley, and Kristin Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery.* Microsoft Research, Redmond, Washington, 2009.

[2] Michael Schatz, Ben Langmead, and Steven Salzberg. Cloud computing and the DNA data race. *Nature Biotechnology*, 28, 2010.

[3] Andrea Held. Exacycle visiting faculty grant program. `http://research.google.com/university/exacycleprogram.html`.

[4] Michael Shirts and Vijay Pande. Screen savers of the world unite! *Science*, 290(5498):1903, 2000.

[5] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Mat Tolton, and Theo Vassilakis. Dremel: interactive analysis of web-scale datasets. *Proceedings of the Conference on Very Large Databases*, Vol. 3, September, 2010, pp. 330-339.