Cost-Benefit Analysis of Cloud Computing versus Desktop Grids

Derrick Kondo[1], Bahman Javadi[1], Paul Malecot[1], Franck Cappello[1], David P. Anderson[2]
[1]INRIA, France, [2]UC Berkeley, USA
Contact author: derrick.kondo@inria.fr

Abstract

Cloud Computing has taken commercial computing by storm. However, adoption of cloud computing platforms and services by the scientific community is in its infancy as the performance and monetary cost-benefits for scientific applications are not perfectly clear. This is especially true for desktop grids (aka volunteer computing) applications. We compare and contrast the performance and monetary cost-benefits of clouds for desktop grid applications, ranging in computational size and storage. We address the following questions: (i) *What are the performance trade-offs in using one platform over the other?* (ii) *What are the specific resource requirements and monetary costs of creating and deploying applications on each platform?* (iii) *In light of those monetary and performance cost-benefits, how do these platforms compare?* (iv) *Can cloud computing platforms be used in combination with desktop grids to improve cost-effectiveness even further? We examine those questions using performance measurements and monetary expenses of real desktop grids and the Amazon elastic compute cloud.*

1 Introduction

Computational platforms have traditionally included clusters, and computational Grids. Recently, two cost-efficient and powerful platforms have emerged, namely cloud and volunteer computing (aka desktop grids).

Cloud Computing has taken commercial computing by storm. Cloud computing platforms provide easy access to a company's high-performance computing and storage infrastructure through web services. With cloud computing, the aim is to hide the complexity of IT infrastructure management from its users. At the same time, cloud computing platforms provide massive scalability, 99.999% reliability, high performance, and specifiable configurability. These capabilities are provided at relatively low costs compared to dedicated infrastructures.

Volunteer Computing (VC) platforms are another cost-efficient and powerful platform that use volunteered resources over the Internet. For over a decade, VC platforms have been one of the largest and most powerful distributed computing systems on the planet, offering a high return on investment for applications from a wide range of scientific domains (including computational biology, climate prediction, and high-energy physics). Since 2000, over 100 scientific publications (in the world's most prestigious scientific journals such as Science and Nature) [15, 5] have documented real scientific results achieved on this platform.

Adoption of cloud computing platforms and services by the scientific community is in its infancy as the performance and monetary cost-benefits for scientific applications are not perfectly clear. This is especially true for volunteer computing applications. In this paper, we compare and contrast the performance and monetary cost-benefits of clouds for volunteer computing applications, ranging in size and storage. We examine and answer the following questions:

- What are the performance trade-offs in using one platform over the other in terms platform construction, application deployment, compute rates, and completion times?
- What are the specific resource requirements and monetary costs of creating and deploying applications on each platform?
- Given those performance and monetary cost-benefits, how do VC platforms compare with cloud platforms?
- Can cloud computing platforms be used in combination with VC systems to improve cost-effectiveness even further?

To help answer these questions, we use server measurements and financial expenses collected from several real VC projects, with emphasis on projects that use the BOINC [1] VC middleware. With this data, we use back-of-the-envelope calculations based on current VC storage

and computation requirements, and current cloud computing and storage pricing of Amazon's Elastic Compute Cloud (EC2) [4].

2 Related Work

In [23], the authors consider the Amazon data storage service S3 for scientific data-intensive applications. They conclude that monetary costs are high as the storage service groups availability, durability, and access performance together. By contrast, data-intensive applications often do not always need all of these three features at once. In [28], the authors determine the performance of MPI applications over Amazon's EC2. They find that the performance for MPI distributed-memory parallel programs and OpenMP shared-memory parallel programs over the cloud is significantly worse than in "out-of-cloud" clusters. In [17], the author conducts a general cost-benefit analysis of clouds. However, no specific type of scientific application is considered. In [9], the authors determine the cost of running a scientific workflow over a cloud. They find that the computational costs outweighed storage costs for their Montage application. By contrast, for comparison, we consider a different type of application (namely batches of embarrassingly parallel and compute-intensive tasks) and cost-effective platform consisting of volunteered resources.

It is well-known that ISP's have always offered similar services as clouds but at much lower rates [17]. However, ISP's resources are not as scalable (according to variable workloads), configurable nor as reliable [17]. The ability to adapt to workload changes is important as server workloads can change rapidly. Configurability is important to suit project programming and application needs. Reliability is important for project scientists to receive and access results, and also to project volunteers as they prefer to receive credit for computation as soon as possible. Thus, we do not consider ISP's in our analysis.

3 Cloud versus Volunteer Computing

Both cloud and volunteer computing have similar principles, such as transparency. On both platforms, one submits tasks without needing to know the exact resource on which it will execute. For this reason, definitions of cloud computing have included VC systems [30]. However, in practice, the cloud computing infrastructures differ from volunteer computing platforms throughout the hardware and software stack. From the perspective of the user, there are two main differences, namely configurability (and thus homogeneity), and quality-of-service.

Clouds present a configurable environment in terms of the OS and software stack with the Xen virtual machine [3] forming the basis of EC2. The use of VM's in VC systems is still an active research topic [7, 16]. So while clouds can offer a homogeneous resource pool, the heterogeneity of VC hardware (e.g. general purpose CPU's, GPU's, the Cell Processor of the Sony PlayStation 3) and operating system (90% are Windows) is not transparent to VC application developers.

Clouds also provide higher quality-of-service than VC systems. Cloud resources appear dedicated, and there is no risk of preemption. Many cloud computing platforms, such as Amazon's EC2, report several "nine's" in terms of reliability. Cloud infrastructures consist of large-scale centralized compute servers with network-attached storage at several international locations. The infrastructures are accessed through services such as S3 also provide high-level web services for data management. By contrast, guarantees for data access or storage, or computation across volatile Internet resources over low-bandwidth and high-latency links is still an open and actively pursued research problem.

3.1 Apples to Apples

Given these dramatic differences between cloud and VC computing, it begs the question of how to compare these systems. **We compare the cost-benefits of cloud versus volunteer computing from the perspective of an embarrassingly parallel and compute-intensive application.**

This is a useful for the following reasons. EC2 is popular computing environment for task parallel batch jobs. This is evident by the fact that Condor is used extensively on EC2, and there are even corporations that specialize in Condor deployments over EC2 [8]. An alternative platform (that is perhaps cheaper and provides higher performance) for these tasks could be a VC system. Conversely, VC scientists may consider hosting servers or even task execution on EC2, depending on the cost-benefits.

4 Platform Performance Trade-offs

Here we describe the performance costs for an application executed over a VC system, and compare them to EC2 costs. Roughly the stages of a VC project and application are the following:

- Platform construction. One must wait and gather enough volunteers in the project.
- Application deployment. As VC systems have a client-server pull architecture, an application will be deployed only as fast as the rate of client requests.
- Execution. During execution, we must consider the effective compute rate of the platform given resources' volatility and task redundancy.

- Completion. The unavailability or slowness of volunteer resources near the end of the computation can stretch task completion times.

In the subsections below, we quantify the performance costs of each of these stages.

4.1 Execution: Cloud Equivalence

We compute the cloud equivalence of a VC system. We answer the following question: how many nodes in a VC system are required to provide the same compute power in FLOPS of a small dedicated EC2 instance? This is similar to the notion of cluster equivalence in [20]. However, in that study the equivalence was computed for an enterprise (versus Internet) desktop grid, and limited to a few hundred machines.

To compute this cloud equivalence ratio, we used the statistics for SETI@home presented in [26]. We find that the average FLOPS of SETI@home is about 514.798 TeraFLOPS. We assume a replication factor of 3 (required for result verification and time task completion), which is quite conservative as projects such as World Community Grid [29] use levels 50% lower. Thus, the effective FLOPS is about 171.599 TeraFLOPS.

Moreover, there are about 318,380 hosts that were active in the last 60 days. This means on average, each host contributes 0.539 GigaFLOPS. We ran the Whetstone benchmark by means of the BOINC client on an EC2 small instance, and the result was about about 1.528 GigaFLOPS for the single core allocated on an AMD Opteron Processor 2218 HE. Thus, the cluster equivalence is about 2.83 active volunteer hosts / 1 dedicated small EC2 instance.

4.2 Platform construction

We compute how long it takes on average for new hosts to register with a project. We used a trace of registration time of SETI@home between April 1, 20007 to January 31, 2009. We found the mean rate of registration to be about 351 volunteer hosts per day. We normalize this rate according to the cloud equivalence (2.83), giving about 124 cloud instances per day.

Figure 1 shows how much time it takes before a certain number of cloud nodes and compute power is reached. For example, we find that is takes about 7.8 days to achieve a platform equivalent to 1,000 cloud nodes (1.5 TeraFLOPS), 2.7 months for 10,000 cloud nodes (15.3 TeraFLOPS), and 2.24 years for 100,000 cloud nodes (152.8 TeraFLOPS).

Note this is a best-case scenario as the rates were determined from an extremely popular project, SETI@home. While we used the mean rate to plot Figure 1, the rate varies greatly over time. We computed the mean rate per day over week, month, and quarter intervals. While the mean rate was roughly the same, the coefficient of variation was as high as 0.83.

In fact, the rate depends on several factors, such as the level of publicity for the project. Clearly, the rate of registration can plateau for some projects. Also, the calculations did not include the limited lifetimes of some of the nodes.

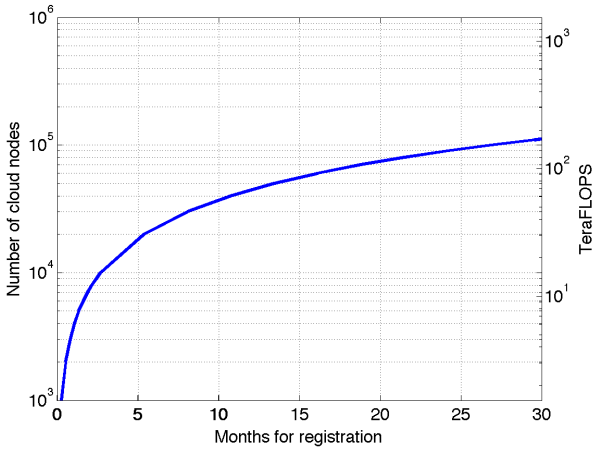


Figure 1. Time to form VC Platform

4.3 Application Deployment

Assuming a system in steady state, the time to send out all tasks in a batch can be lengthy as clients use a pull method for retrieving tasks, and clients only connect to the server periodically.

Here we summarize the work of Heien et al. [18] where the authors determined the time to deploy a batch of tasks. In particular, the authors found that:

$$L = \frac{TQ}{P} \tag{1}$$

L is the time frame during which tasks are distributed, P is the number of clients, and Q is 1.2 $\times$ the number of tasks. T is the reconnection period, which is a parameter specified by the project scientist to the client denoting the time that must expire before it reconnects to the server. By default, in the BOINC VC system, T is six hours.

Figure 2 shows the time required to assign all tasks in a batch, assuming a replication factor of 3. We consider three batch sizes of 100, 1000, and 10000 tasks (and with replication, a total of 300, 3000, and 30000 tasks). For example, deploying a batch with 100, 1000, and 10000 unique tasks over a platform with 10,000 cloud nodes (or equivalently 28300 volunteer nodes) would take 4.6, 45.8, or 458 minutes, respectively.

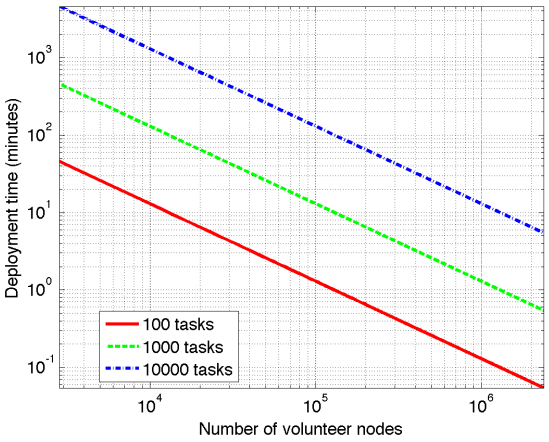


Figure 2. Time to deploy batch of tasks

Instance Type	Cost/hour (USD)
Standard Small	0.10
Standard Large	0.40
High-CPU	0.20

Table 1. Pricing for EC2 Instances

Transfer Type	Cost/GB-Month (USD)
Inbound transfer	0.10
first 10 TB	0.17
next 10-50TB	0.13
next 50-150TB	0.11
over 150 TB	0.10

Table 2. Pricing for EC2 Data Transfer

4.4 Completion times

The volatility and heterogeneity in VC systems makes timely completion of task batches challenging. BOINC has a number of mechanisms for ensuring time completion. For example, project scientists can soft deadlines for tasks. When the soft deadline of a task approaches, the local client scheduler will increase the task's priority relative to others. In addition, the server-side scheduler uses the deadline for determining timeouts, i.e., when another task instance should be sent out.

With these mechanisms, task completion is usually done at a high success rate. For example, in the World Community Grid project (a non-profit project for volunteer computing), 96.1% of tasks met their deadline out of 227,485 tasks [24].

Nevertheless, VC users should expect a stretch (defined by the amount of time spent by the job in the system and its execution time) of at least 5 according to our simulation results in [21]. This is because the task deadlines are usually high relative to the amount of actual work. The median project deadlines are around 9 days, where as the execution time per task is about 3.67 hours on a dedicated 3GHz host [6]. Recently, there has been promising results in using predictive models for achieving fast turnaround time [2, 19, 14]

By contrast, on EC2, platform construction takes a few minutes to deploy an image. This assumes that the platform is not overloaded. As resources are dedicated, application deployment is instantaneous, and task execution and completion are relatively constant and low.

5 Cloud Computing Costs

We present an overview of Amazon's cloud services and pricing [13] to be used in our calculations. Amazon has two relevant cloud computing services. First, Amazon offers the Elastic Computing Cloud service.EC2 charges each hour an instance is running, and it offers instances with different compute power and memory. The pricing for EC2 is shown in Tables 1 and 2.

Second, in conjunction with EC2, Amazon offers the Elastic Block Store (EBS) service. This provides reliable and persistent storage with high IO performance. EBS charges per GB of storage and per million IO transactions. The pricing for EBS is shown in Table 3. Amazon also offers the Simple Storage Service (S3). This service provides access through web services to persistent data stored in buckets (one-level of directories) along with meta-data (key/value pairs). S3 charges per GB of storage and HTTP requests concerning it. PersistentFS offers a POSIX-compliant file system using S3 and is arguably cheaper than EBS for mainly read-only data. However, for volunteer computing projects, the cost difference between S3/PersistentFS and EBS is not significant and does not change our conclusions. Thus we assume all storage occurs on EBS. We do not consider costs of snapshots, i.e., EBS volume backups to Amazon's S3.

Resource	Rate (USD)
Storage	0.10 / GB-Month
IO request	0.10 / million

Table 3. Pricing for EBS

Component	Project	
	SETI@home	*XtremLab*
Salaries	10K for sys admins	5K
Electricity	90 for 6 servers	15
Network	2K for 100 Mbit	covered by university
Hardware	18K for servers, 25K for air conditioner	4K
Total startup	43K	4K
Total monthly	12K	5k

Table 4. Project Costs (monthly)

6 Volunteer Computing Project Costs

We detail the costs of maintaining large and small volunteer computing projects. The main costs are due staff salaries for the installation and programming of server software, hardware maintenance, and recruitment and maintenance of volunteers. There are also high start-up costs for purchasing hardware. These costs could potentially be subsumed or lowered by deploying a volunteer computing project or service over a cloud. For example, the cost of one full-time systems programmer could be amortized over several VC project servers hosted on a cloud.

Listed below are the costs for a large project, namely SETI@home, with about 318,380 active hosts. All costs are in terms of USD.

The majority of SETI@home's costs are for salaries for system administrators and programmers, hardware, and network bandwidth. SETI pays about 10K / month for 2 system administration and programmers that spend about 50% of time on work that would be subsumed by a cloud. (Note that staffing costs tend to be variable across projects. For example, EINSTEIN@home uses student programmers which are significantly cheaper than full-time staff.)

SETI@home is unique in the sense that its network costs are not supported by the University. Instead, it pays about 1K / month for a 1 Gbps connection with Hurricane electric. SETI also pays about 1K / month in university fees and networking hardware. Its 6 servers use about 400 watts each, for a total of 2400 watts. This equates to 24*2400 = 60 KWh per day. The University rate is about 5 cents per kWh. The cost of the 6 servers is about 18K, although they were actually donated. In addition, the project managers purchased an air conditioner for 25K. Ultimately, the total start-up costs were about 43K, and total monthly costs were about 12K.

We also compute the costs for a relatively small project called XtremLab [22] with about 3,000 active hosts. The majority of costs are for a 50%-time system administrator/programmer. The total start-up cost was about 3K/month for a server with a RAID storage server. The total monthly cost was about 5K/month.

While the total costs are relatively low, they are not negligible. This begs the question of when it is more cost-efficient to host the entire platform over a cloud instead.

7 Platform Hosting on a Cloud

7.1 Number of volunteers needed before VC is more effective than cloud

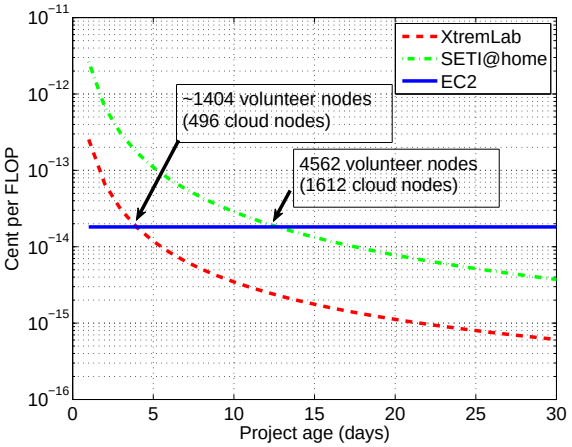


Figure 3. Cost-effectiveness of Cloud versus VC

We determine how many volunteers are required before VC becomes more cost-effective than clouds. We also determine the minimum project age need to achieve that number of volunteers. We assume volunteers register at the rate of 124 cloud nodes per day as calculated in Section 4.2. Figure 3 shows the cent per flop for an EC2 small instance, and two VC systems, namely SETI@home (large project) and XtremLab (small project). In the cost calculation, use we include both startup and monthly costs for SETI@home and XtremLab as shown in Table 4.

We find that for a relatively small project such as XtremLab, one must have at least $\sim$1404 volunteer nodes (equivalently $\sim$496 dedicated cloud nodes) and wait at least $\sim$4 days before the VC system becomes cheaper per FLOP than EC2. For a large project, one must have at least $\sim$4562 volunteer nodes (equivalently $\sim$1612 cloud nodes) and wait at least $\sim$13 days.

7.2 VC versus cloud costs over time

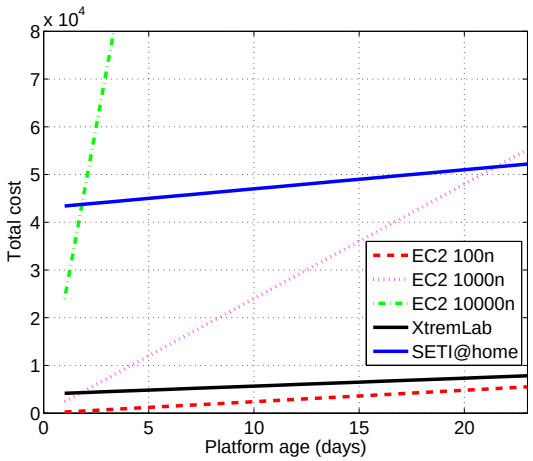


Figure 4. Total cost of Cloud versus VC over time

We also computed the total cost of EC2 versus the cost on a VC system over time (see Figure 4). We did so with EC2 platforms with 100, 1000, and 10000 nodes, and also XtremLab and SETI@home. As XtremLab and SETI@home have high start-up costs, the EC2 platforms start off cheaper with the exception of the 10000 node platform and XtremLab.

After three days, the cost of 1000 and 10000 node platforms becomes higher than that of the VC platform. The 100-node EC2 platform has lower month costs than both VC platforms. Thus, it is always cheaper. Clearly, VC systems are advantageous as the cost remains constant with number of number of nodes, though it takes time to get enough volunteers, as shown in Section 4.2.

7.3 Months of VC supportable by x months of cloud costs

We determined the cost of x months on an EC2 platform and determined how many months y on a VC platform could be supported with the same monetary amount (see Figure 5). We did so with an EC2 platform of 1000 and 10000 small instances, and with SETI@home and XtremLab monthly expenses. We find, for example, that 4 months on EC2 with 1000 nodes can support over a year of SETI@home. Of course, this comparison is only applicable assuming the cloud equivalence of the VC platform is at least as high as the EC2 platform.

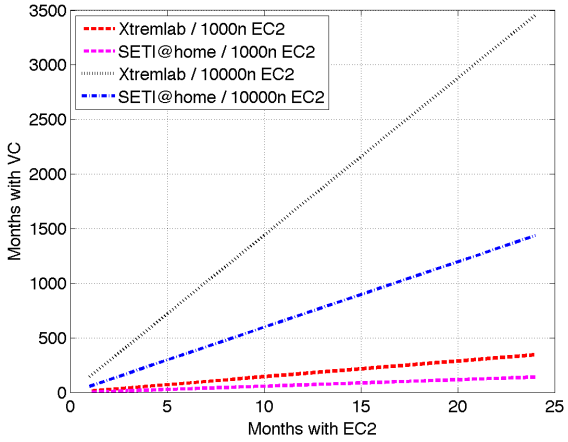


Figure 5. VC month costs versus EC2 month costs

7.4 Cloud platform sustainable by VC monthly costs

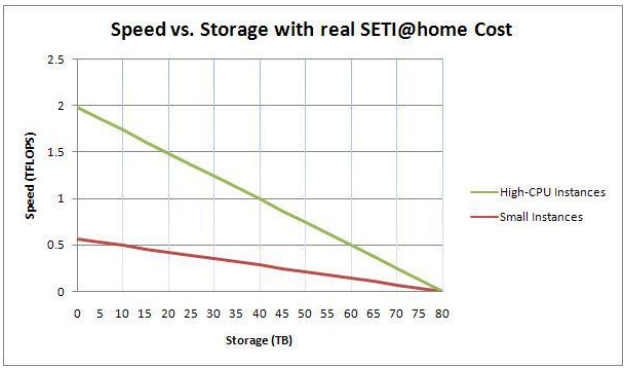
We also determined the size of a cloud platform that would be supportable based on a VC project's current costs (see Figure 6). With 12K per month, SETI could purchase a maximum of 2 TeraFLOPS sustained over a month with High CPU instances (and no storage). Alternatively, SETI could purchase 80 TeraBytes of storage (and no computing power) per month. By contrast, SETI@home provides about 514.798 TeraFLOPS of compute power and 7740 GigaBytes of storage. These levels of computation and storage are 2 orders of magnitude less than the levels currently provided by project volunteers.

Similarly, the cloud resources purchasable using the current budget of XtremLab (a project much smaller than SETI@home) are still an order of magnitude lower than what the cloud platform can provide. XtremLab provides about 2.9 TeraFLOPS of compute power and 108 GigaBytes of storage. For cloud computing to become more cost effective than volunteer computing, costs would have to decrease by at least an order of magnitude.

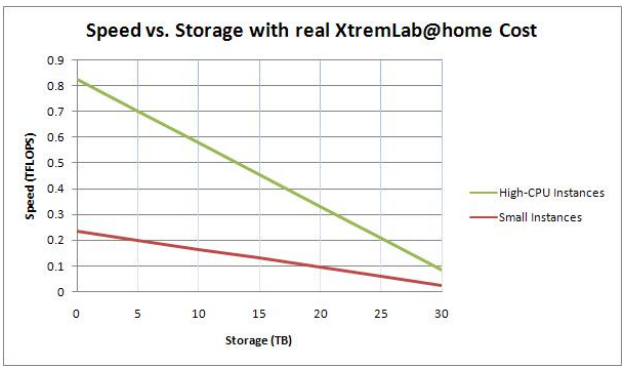
8 Server Hosting on a Cloud

Cloud computing is advantageous for variable workloads as the infrastructure can scale with rapid increases (or decreases). Moreover, costs are variable (and in total less than the fixed costs). Cloud computing is effective for small to medium sized applications.

For large projects, the costs are simply too high [25, 27] to host on cloud. For example, with the FOLDING@home project, the storage requirements are about 500 TeraBytes.



(a) SETI@home server



(b) XtremLab server

Figure 6. Cloud resources obtainable given current project costs

If stored on Amazon's S3, the 500TB of Folding@home's data would cost over 50K USD / month. Moreover, Folding@home data analysis requires data access and manipulation so the potential costs of inbound and outbound data transfers from S3 would make the estimate even higher.

In comparison to FOLDING@home, SETI@home and XtremLab have less demands in terms of the computing infrastructure. In the sections below, we determine the cost of hosting these projects on the cloud, and which server components make up the majority of costs when moving to the cloud.

We believe that server hosting on a cloud has potential for several reasons. First, server workload has significant variation. In Figure 7, we see that the number of active hosts over time participating in various BOINC projects. We find that the number of hosts varies greatly, and thus one could expect fluctuations in server host load. For example, in SETI@home, the number of hosts increased by almost an order of magnitude within a one month period. Spikes in load can be due to project publicity. In addition, there can be significant decreases in load. For example, with the Predictor@home project, the load decreased about 80% over a 1 month time period. Decreases in host load can be due to the fact that most projects (with the exception of SETI@home) have finite workloads in batches. Often, the server has no work to distribute and servers are idle.

Moreover, most volunteer computing projects are relatively small. We observe that more than half the projects have less than ten thousand hosts. If a global volunteer computing service over a cloud was offered, the project programming and maintenance costs could be lowered by amortizing over several projects and applications (in addition to amortizing the hardware maintenance costs).

8.1 Project Resource Usage

Here we characterize the resources used when hosting a BOINC project server. BOINC project servers consist of several components [1]. A server-side Scheduler is used to receive client requests for workunits (aka tasks). Upon request, the Scheduler assigns workunits to the client, which involves querying the BOINC database for workunit information. After workunit assignment by the Scheduler, the client downloads the workunits. Upon workunit completion, the client will upload the results to a (possible separate) data server. A File Upload Handler on the data server is used to receive results from clients and to store them on a file system. Uploaded results are periodically moved to a science database and file system.

The monthly resource usage of typical VC projects is shown in Table 5. An upper bound on the number of IO transactions was determined using /proc/diskstats [10]. We determine the other statistics using standard Linux tools.

In general, the SETI@home server uses about 3 TB of storage, 100Mbps of bandwidth, and has modest IO rates. The server serves about 318,380 active clients. The scheduler and download outbound data transfer rate is much greater than the inbound. The download throughput is constrained by a 100Mbit limit.

We also determine the resource usage for a smaller BOINC project called XtremLab [22]. In general, the XtremLab server uses about 65 GB of storage, 11Kbits/sec of bandwidth, and very light IO rates. The server serves about 3,000 active clients. As the XtremLab project ended in 2007, these estimates were computed post-mortem using the server logs and files.

8.2 Server Costs on Cloud

The mapping of the components of VC servers (as described in Section 8) to Amazon's cloud components (as

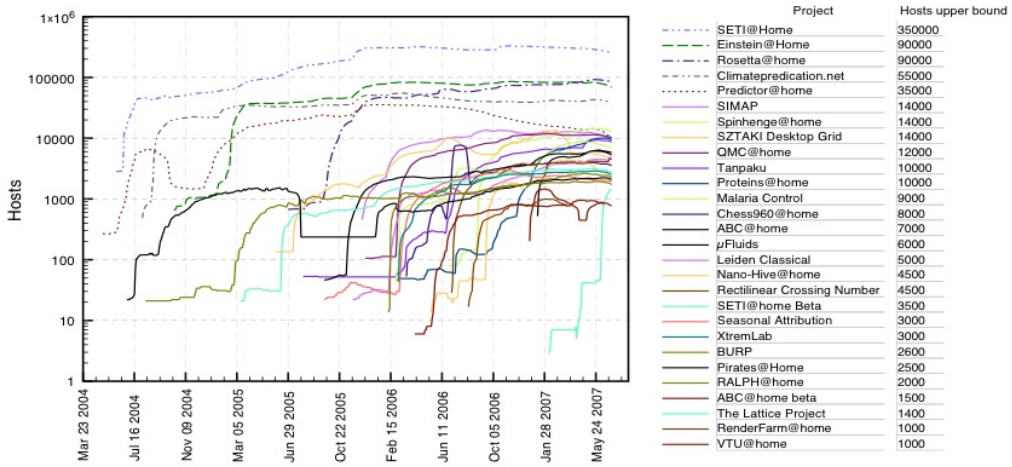


Figure 7. Number of hosts per project over time

Component	Project *SETI@home*	*XtremLab*
Upload (result) storage	200GB	negligible
Download (workunit) storage	2,500GB	.14GB
BOINC database storage	200GB	1GB
Science results/database storage	1,000GB	64GB
Scheduler throughput	6Mbits/sec outbound	negligible
Upload throughput (peak)	10Mbits/sec inbound	9.3Kbits/sec
Download throughput (peak)	92Mbits/sec outbound	1.7Kbits/sec
IO transactions	141.9 million	negligible

Table 5. Project Resource Usage

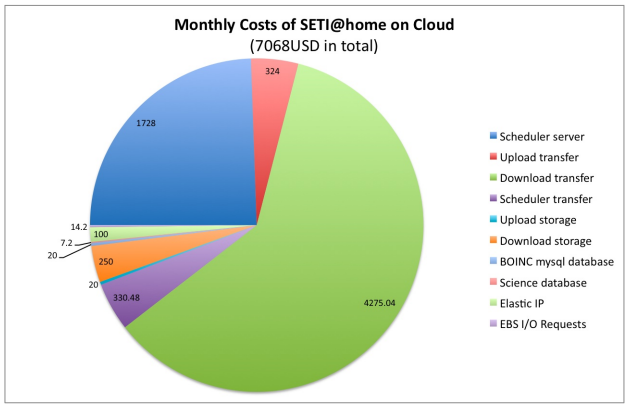
described in Section 5) is as follows. We assume the Scheduler and File Upload Handler execute over EC2. We assume the BOINC database is hosted on EBS. We assume the storage for uploads, downloads, and science results is stored on S3.

Figure 8(a)shows the costs for SETI@home on Amazon's cloud. In total, it would cost about 7K USD / month to host the SETI@home server on cloud. The majority of costs (~60%) are due to bandwidth alone. About 25% of costs are due to CPU time of the 6 instances. Nevertheless, the cloud costs are less than 60% of SETI@home's current costs. So surprisingly, clouds may be cost-effective for even a "large" project such as SETI@home. However, one has to consider that staff costs for maintenance and etc. are variable across projects, and that we assume these costs would be subsumed by the cloud.

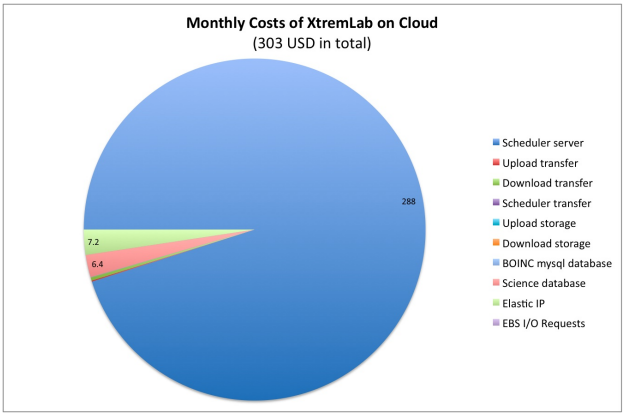
Figure 8(b)shows the costs for XtremLab. Monthly costs amount to about 300 USD / month. 95% of costs are due to the CPU time of the instance. The cloud costs are about 6% of XtremLab's standalone costs. Clearly, clouds are advantageous for smaller, less-bandwidth intensive projects.

8.3 Server resources with given yearly budget

We also determined the amount of server resources for a given budget. In Figure 9, we show the amount of cloud download bandwidth and storage for a yearly budget of 10K and 15K USD. We assume a Large instance, and show total bandwidth and storage (over all instances when multiple exist). Clearly, as the number instances increases, one has to pay more for their CPU time, and the total amount of purchasable bandwidth and storage decreases. An application scientist could use these graphs to determine whether their

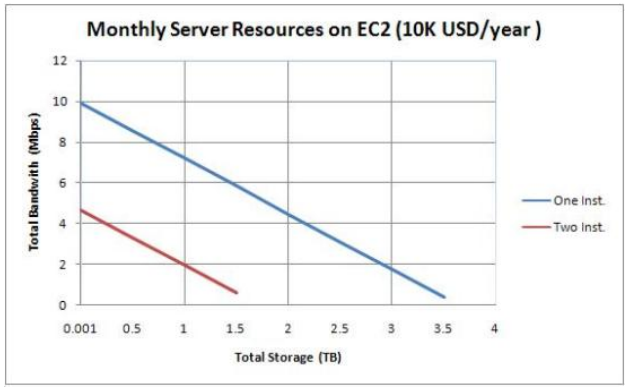


(a) SETI@home server

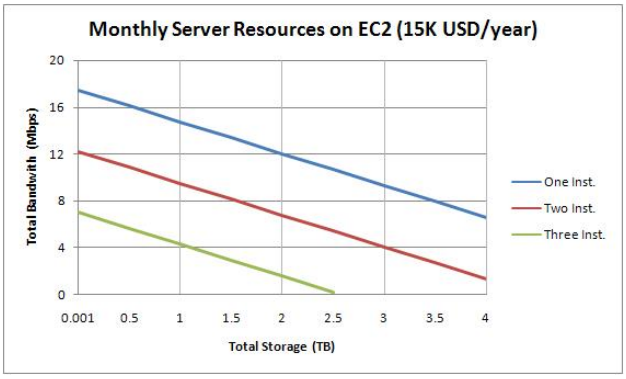


(b) XtremLab server

Figure 8. Server costs on Cloud



(a) 10k USD



(b) 15k USD

Figure 9. Cloud resources with given yearly budget

application requirements can be met using a cloud with a given annual budget. For example, at 10K USD / year, one could have a large instance with about 6Mbps in bandwidth and 1.5 TB of storage total. At 15K USD, one could have two instances with 8Mbps and about 1.6TB of storage total.

9 Conclusions

We determined the cost-benefits of cloud computing versus volunteer computing applications. We calculated VC overheads for platform construction, application deployment, compute rates, and completion times. We found that in the best-case scenario, hosts register at a rate of 124 cloud nodes per day. We found that the ratio of volunteer nodes needed to achieve the compute power of a small EC2 instance is about 2.83 active volunteer hosts to 1.

We detailed the specific costs of a large and small VC project. We find that monthly VC project costs range between 5K-12K, and startup costs range from 4K to 43K. If cloud computing systems are to replace VC platforms, pay-per-use costs would have to decrease by at least an order of magnitude.

With these performance and monetary cost-benefits in mind, we compared the two platforms. We find that at least $\sim$1404 volunteer nodes are needed before VC becomes more cost effective in terms of cents per FLOP. Nevertheless, the cost of a 1000-node cloud will exceed that of VC system after three days. We also find that 4 months on EC2 with 1000 nodes can support over a year of SETI@home. We also examined the size of a cloud platform sustainable by VC costs. With 12K per month, SETI could purchase a maximum of 2 TeraFLOPS sustained over a month with High CPU instances.

We also consider hybrid approaches where a VC server is hosted on a cloud to lower the start-up and monthly costs. The savings ranges between 40-95% depending on resource usage. In general, if bandwidth needs do not exceed 100Mbit and storage needs are less than 10TB's, hosting a server on a cloud is likely cheaper than conducting a project on one's own. Server bandwidth on cloud is particularly expensive.

We have made available online our Excel file [12] so that scientists can determine themselves their own project cost-benefits. Also, to allow users to quickly and easily deploy a BOINC server on EC2, we have created an Amazon Machine Image (AMI) with a BOINC server pre-installed and configured. Instructions for the AMI deployment have been made available online [11]. This can be used as a testing or production server.

For future work, we will consider cost models of other clouds or ISP's other than Amazon where for example network bandwidth is significantly cheaper. In this case, different components of the server can be hosted at different clouds or ISP's depending on costs. We will also investigate the reduction of server costs on EC2 using dynamic instance creation and load balancing.

10 Acknowledgements

We thank Vijay Pande and Adam Beberg (of Folding@home), Bruce Allen (of EINSTEIN@home), Milo Thurston (of climateprediction.net), and Matt Lebofsky (of SETI@Home) for useful discussions and input about project resource usage and costs. We also thank Raphael Bolze, Gilles Fedak, and our anonymous reviewers for their insightful comments that greatly improved this manuscript.

References

[1] D. Anderson. Boinc: A system for public-resource computing and storage. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, Pittsburgh, USA, 2004.

[2] Artur Andrzejak, Derrick Kondo, and David P. Anderson. Ensuring collective availability in volatile resource pools via forecasting. In *DSOM*, pages 149–161, 2008.

[3] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Timothy L. Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In *SOSP*, pages 164–177, 2003.

[4] J. Bezos. Amazon.com: Amazon EC2, Amazon Elastic Compute Cloud, Virtual Grid Computing: Amazon Web Services. `http://www.amazon.com/gp/browse.html?node=201590011`.

[5] BOINC Papers. `http://boinc.berkeley.edu/trac/wiki/BoincPapers`.

[6] Catalog of boinc projects. `http://boinc-wiki.ath.cx/index.php?title=Catalog_of_BOINC_Powered_Projects`.

[7] Attila Csaba Marosi, Peter Kacsuk, Gilles Fedak, and Oleg Lodygensky. Using virtual machines in desktop grid clients for application sandboxing. Technical Report TR-0140, Institute on Architectural Issues: Scalability, Dependability, Adaptability, CoreGRID - Network of Excellence, August 2008.

[8] Cycle Computing Inc. . `http://my.cyclecloud.com/welcome/`.

[9] E. Deelman, S. Gurmeet, M. Livny, J. Good, and B. Berriman. The Cost of Doing Science in the Cloud:

The Montage Example. In *Proc. of Supercomputing'08, Austin*, 2008.

[10] IO statistics fields. http://devresources.linux-foundation.org/dev/robustmutexes/src/fusyn.hg/Documentation/iostats.txt.

[11] How to Deploy a BOINC server on the Amazon Elastic Compute Cloud. http://boinc.berkeley.edu/trac/wiki/CloudServer.

[12] Excel file for EC2 costs. http://mescal.imag.fr/membres/derrick.kondo/cloud_calc.xlsx.

[13] Amazon EC2 pricing. http://aws.amazon.com/ec2/faqs.

[14] T. Estrada, D. Flores, M. Taufer, P. Teller, A. Kerstens, and D. Anderson. The Effectiveness of Threshold-based Scheduling Policies in BOINC Projects. In *Proceedings of the 2st IEEE International Conference on e-Science and Grid Technologies (eScience 2006)*, December 2006.

[15] Folding@home Papers. http://folding.stanford.edu/English/Papers.

[16] Arijit Ganguly, Abhishek Agrawal, P. Oscar Boykin, and Renato J. O. Figueiredo. Wow: Self-organizing wide area overlay networks of virtual workstations. *J. Grid Comput.*, 5(2):151–172, 2007.

[17] Simson Garfinkel. Commodity grid computing with amazons s3 and ec2. In *login*, 2007.

[18] Eric Martin Heien, David P. Anderson, and Kenichi Hagihara. Computing Low Latency Batches with Unreliable Workers in Volunteer Computing Environments. 2008. under submission.

[19] Eric Martin Heien, Noriyuki Fujimoto, and Kenichi Hagihara. Computing low latency batches with unreliable workers in volunteer computing environments. In *Workshop on Volunteer Computing and Desktop Grids (PCGrid)*, pages 1–8, 2008.

[20] D. Kondo, M. Taufer, C. Brooks, H. Casanova, and A. Chien. Characterizing and Evaluating Desktop Grids: An Empirical Study. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'04)*, April 2004.

[21] Derrick Kondo, David P. Anderson, and John McLeod. Performance evaluation of scheduling policies for volunteer computing. In *eScience*, pages 415–422, 2007.

[22] P. Malecot, D. Kondo, and G. Fedak. Xtremlab: A system for characterizing internet desktop grids (abstract). In *in Proceedings of the 6th IEEE Symposium on High-Performance Distributed Computing*, 2006.

[23] Mayur Palankar, Adriana Iamnitchi, Matei Ripeanu, and Simson Garfinkel. Amazon S3 for Science Grids: a Viable Solution? In *Data-Aware Distributed Computing Workshop (DADC)*, 2008.

[24] K. Reed. Personal communication, 2008.

[25] Ben Segal. Personal communication, June 2008.

[26] Boinc stats for seti. http://boincstats.com/stats/project_graph.php?pr=sah\&view=hosts.

[27] Vijay Pande. Private communication, 2004.

[28] Edward Walker. Benchmarking Amazon EC2 for high-performance scientific computing. In *USENIX LOGIN*, 2008.

[29] World community grid. http://www.worldcommunitygrid.org/.

[30] Cloud Computing. http://en.wikipedia.org/wiki/Cloud_computing.

Biographies

Derrick Kondo is a research scientist at INRIA Rhone-Alpes Grenoble. He received his Bachelor's in Computer Science at Stanford University, and his Master's and Ph.D. in Computer Science from the University of California at San Diego. He was an INRIA post-doctoral fellow at LRI (Computer Research Laboratory) at the University of Paris-Sud. He founded and serves as Chair/Co-Chair of the Workshop on Volunteer Computing and Desktop Grids (PCGrid). He is co-guest editor of a special issue of the Journal of Grid Computing on desktop grids. His research interests include the characterization, modelling, and simulation of large-scale distributed computing systems, and scheduling mechanisms and algorithms for volatile resources.

Bahman Javadi is an INRIA post-doctoral fellow in the MESCAL team at INRIA Rhone-Alpes Grenoble. He received his PhD and Master's in Computer Engineering from Amirkabir University of Technology. His research interests include cluster and grid computing, parallel and high performance computing and networks and interconnection networks.

Paul Malecot is a PH.D. student in the Grand-Large group of INRIA at the University of Paris-Sud. His research interests include characterization, modelling, and emulation of large-scale distributed and volatile computing systems.

Franck Cappello is a research scientist at INRIA, and leads the Grand-Large group at LRI (Computer Research Laboratory) in Paris-Sud University, in Orsay, France. He is currently the director of the INRIA-Futurs division focusing on distributed systems research, and co-director of the Grid 5000/Alladin project, which is the largest grid computing platform in France. His research interests include parallel programming models, parallel runtime environments and performance evaluation of clusters, clusters of multiprocessors, and large-scale distributed systems. His research group is currently working on large-scale emulation environments, fault-tolerant MPI (MPICH-V), and peer-to-peer global computing using large collections of idle computers over the Internet.

David P. Anderson is a Research Scientist at the Space Sciences Laboratory, at the University of California, Berkeley. He leads the SETI@home, BOINC, Bossa and Bolt projects. He received a BA in Mathematics from Wesleyan University, and MS and PhD degrees in Mathematics and Computer Science from the University of Wisconsin-Madison. From 1985 to 1992 he was an Assistant Professor in the UC Berkeley Computer Science Department, where he received the NSF Presidential Young Investigator and IBM Faculty Development awards. His research focused on distributed systems for handling digital audio and video in real time. In 1995 he joined David Gedye and Dan Werthimer in creating SETI@home, which he continues to direct. From 2000 to 2002, he served as CTO of United Devices. In 2002 he created the BOINC project, which develops an open-source software platform for volunteer computing. The project is funded by NSF and is based at the UC Berkeley Space Sciences Laboratory. BOINC is used by about 100 projects, including SETI@home, Einstein@home, Rosetta@home, Climateprediction.net, and the IBM World Community Grid. In 2007 Anderson launched two new software projects: Bossa (middleware for distributed thinking), and Bolt (a framework for web-based training and education in the context of volunteer computing and distributed thinking).