Evaluating Google Compute Engine with PROOF

Gerardo Ganis
CERN, PH Departement, SFT group, CH 1211 Geneva 23, Switzerland
E-mail: gerardo.ganis@cern.ch

Sergey Panitkin
Brookhaven National Laboratory, Upton, NY, USA
E-mail: panitkin@bnl.gov

Abstract. The advent of private and commercial cloud platforms has opened the question of evaluating the cost-effectiveness of such solution for computing in High Energy Physics . Google Compute Engine (GCE) is a IaaS product launched by Google as an experimental platform during 2012 and now open to the public market. In this contribution we present the results of a set of CPU-intensive and I/O-intensive tests we have run with PROOF on a GCE resources made available by Google for test purposes. We have run tests on large scale PROOF clusters (up to 1000 workers) to study the overall scalability of coordinated multi-process jobs. We have studied and compared the performance of ephemeral and persistent storage with PROOF-Lite on the single machines and of standard PROOF on the whole cluster. We will discuss our results in perspective, in particular with respect to the typical analysis needs of an LHC experiment.

1. Introduction

The improvements in the virtualization technology occurred in the last decades have created the conditions to have large computing farms based on cloud middleware, either commercial or private, from which any interested user can get the amount of resources required at a given moment.

Inevitably, this paradigm shift affects also High Energy Physics (HEP). The needs of the HEP community are not any longer the largest ones. Commercial companies such as Google, Amazon or Facebook have passed the number of machines that large HEP computing infrastructures - like the one at CERN, for example - have to manage. Their solutions may constitute a good starting point for the next generation of large HEP computing infrastructures. Consideration of this sort are at the base of the Agile Infrastructure project at CERN [1].

The way cloud-based solutions can be used efficiently for HEP has been investigated at least since the introduction of Amazon S3 interface in 2006. Already at CHEP 2009 there has been a plenary talk devoted to a real concrete implementation of a Amazon EC2 based solution for Monte Carlo production for the Belle experiment [2], and interest in the field has regularly increased since then[1].

In this paper we present the results of some tests we have been doing using Google Compute Engine (GCE) [3], the newest born of the commercial clouds.

[1] See http://www.google.com/trends/explore#q=Cloud+Computing,Grid+Computing,Virtualization

2. Google Compute Engine setup

GCE, Google's IaaS product, was announced in June 2012 and went public in January 2013. Based on KVM, it runs exclusively Linux images provide by Google [2]. Currently available are the latest Debian (6 and 7) and CentOS 6.2. Resources are provided in five geographical zones spanning across Europe (europe-west1-a, europe-west1-b) and US (us-central1-a,us-central1-b, us-central2-a) with pricing scheme comparable to Amazon EC2.

The type of machines that one can get vary in terms of virtual cores (1,2,4 or 8), RAM and *ephemeral* (also called *scratch*) disk. As shown in Table 1, three different profiles exists, *standard*, *highcpu* and *highmem*, each with or withour ephemeral disk [3]. Each machine has a 10 GB persistent disk where the OS and home directories are installed. The *highcpu* profiles target applications requiring a large amount of CPU and relatively not much memory; however, in absolute terms the CPU power should be the same of the equivalent *standard* profile. The *highmem* profiles machines with about three times RAM with respect to *standard*.

Profile	vCPU	GCEU/vCPU	RAM/vCPU
standard	1,2,4,8	2.75	1.875 GB
highcpu	1,2,4,8	2.75	0.9 GB
highram	1,2,4,8	2.75	6.5 GB

Table 1. Machine profiles at GCE [4].

Additional persistent disks can be obtained and can be mounted in R/O on all the machines concurrently. This allows, for example, to make data visible to all machines from a single mount point. All disks are provided as NAS.

GCE provides support for snapshots of the base disk of an instance. This allows, for example, to clone instances once contextualized. Contextualization wise, a metadata server is available to pass user data to the image; this functionality however was not used in these tests.

GCE has two main interfaces. A web portal and a command line interface via a dedicated tool (called `gcutil`) available for download from the GCE pages. Both tools allow full control on all the aspects of a project, including visualization of quotas and monitoring of resource utilization.

2.1. VMs used for this test

For this test we contextualized a virtual machine using the newest Debian 7 image available. The contextualization consisted of installing the required software (ROOT, XRootD) and the relevant configuration files to run a PROOF cluster. Once the initial machine - used as master - was ready, we have deployed the cluster by making a snapshot of the initial machine and cloning the base disks for the machines used as workers from the snapshot.

All these operations - creating snapshots, cloning disks, creating new instances, etc - have been done using `gcutil` and turned out to be quite fast, independently of the number of machines (at least up to the number of machines we used, about 60). For example, starting a new machine required typically less than a minute.

Although the *manual* setup worked well, we believe that, for a production service, the usage of the contextualization tools provided via the metadata server would be required. In this respect, the future support for private images may also facilitate setting up a facility for a specific experiment.

[2] Support for private images may be provided in the future.
[3] As can be seen from Ref. [4], starting from 3 December 2013 the offer has slightly changed; Table 1 reflects the situation at the time of the tests.

3. PROOF and the benchmark framework

3.1. PROOF

PROOF is a processing framework to run ROOT sessions in parallel in a coordinated way. The architecture (see Fig. 1) is a multi-tier one with the master in interactive control of the workers. Work distribution follows a pull model which allows to adapt to effective performances of the available resources for their optimal exploitation. In this respect PROOF is intrinsically *cloud-aware* although it has been designed well before the advent of clouds[4].

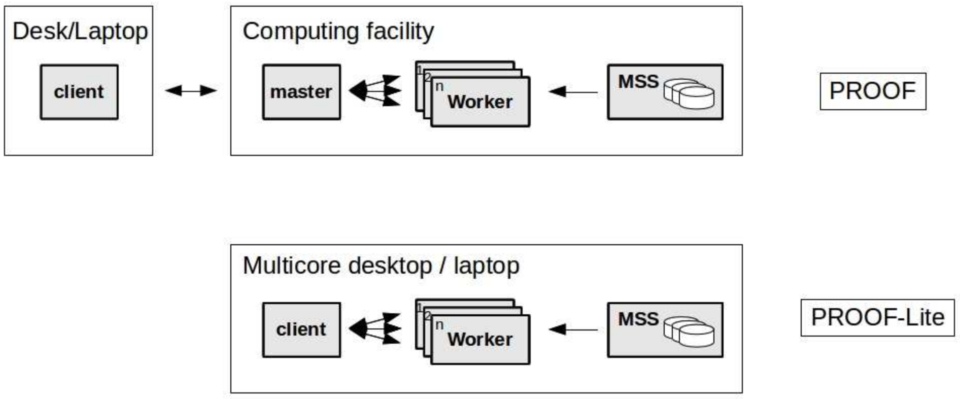

Figure 1. PROOF and PROOF-Lite architectures

Figure 1 shows also the 2-tier architecture featured by PROOF-Lite, the implementation of PROOF paradigm targetting multi-core machines and used in this analysis to benchmark the single machine.

3.2. TProofBench

To measure the performance of the facility we used the PROOF benchmark framework `TProofBench` [8]. The framework allows to measure the scalability of the system for different types of processing. By default it provides two benchmarks: a CPU-intensive one, where the unit *cycle* is the generation of a certain number of random numbers and the filling a few histograms; and an I/O intensive one, where the *cycle* is reading an event from a ROOT test TTree[5] and doing some filtering.

For this run TProofBench has been augmented with a new measurement, the maximum processing rate during the PROOF query[6]. This allows to disentangle the pure parallel component of PROOF from the effects of the serial parts, as we will see this in more detail when commenting the results.

3.3. ROOT version

For these test we used a release candidate of version 5.34/11 of ROOT. The differences with the final 5.34/11 are marginal and not related to the components relevant for this analysis.

[4] See also the discussion in [7].
[5] For the default I/O benchmark, TProofBench uses a tree of the `Event` class defined by `test/Event.h` and `test/Event.cxx` in any recent ROOT distribution.
[6] The feature is available from ROOT version 5.34/11 on

4. PROOF-Lite tests

We started with PROOF-Lite tests to measure the single machine. For the test we used two VMs as described in Table 2.

Machine name	vCPU	GCEU	RAM	Ephemeral Storage
n1-standard-8-d	8	22	30 GB	1.7 TB
n1-highcpu-8-d	8	22	7.2 GB	1.7 TB

Table 2. VM used for the PROOF-Lite test

According to the available online documentation [4], in GCE a virtual CPU is implemented as a single hyperthread on a 2.6GHz Intel Sandy Bridge Xeon processor. This means that the machine type we used sees four physical cores.

4.1. CPU intensive tests

Figure 2 shows the scaling result obtained from the CPU intensive benchmark on a *n1-standard-8-d* machine.

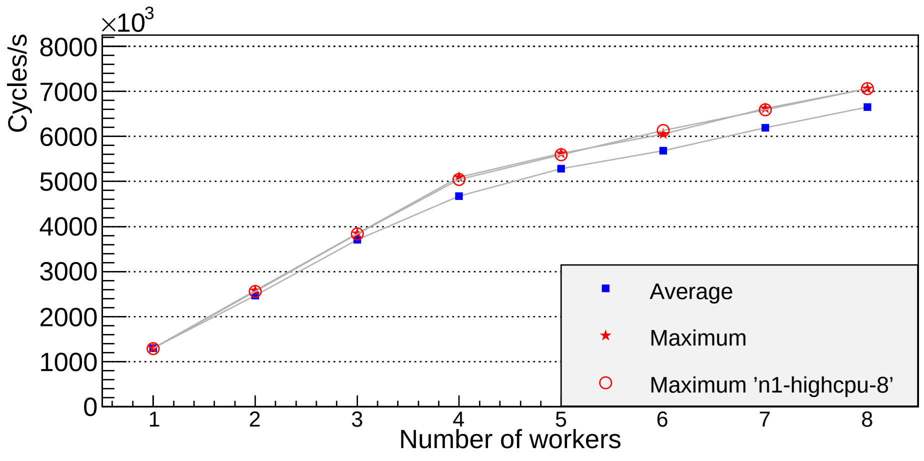

Figure 2. CPU performance scaling on a *n1-standard-8-d* machine.

The maximum rate shows the typical behaviour we expect from hyperthreading. The aggregate rate increases linearly but the slope, i.e. the contribution added by each core, changes when the hyperthreaded cores enter the game, that is for five and more workers. The slope is a measure of the processing power of the single core. Table 3 gives the values measured for these machine for *real* and *hyperthreaded* cores. For comparison we show in Table 3 the results of similar measurements done on physical machines with recent CPUs.

In Figure 2 the scaling result for the average rate is also shown. The average rate includes the worker initialization and the collection, merging and end-of-query phases. All these phases depend significantly of the type of analysis and output composition and a meaningful measurement should be done on the case of interest. These phases have important serial components impacting the overall parallel speed-up. The qualitative effect of this serial component is to lower the overall rate and to introduce a deviation from linearity.

Finally, the result of the same test repeated on a *n1-highcpu-8-d* machine is also shown in Figure 2. As expected from the amount of GCEU available on the machines, the two types of machines are indistinguishable for these purposes.

CPU type	Real *MCycles/s/core*	Hyperthreaded *MCycles/s/core*	Remarks
n1-standard-8-d	1.26	0.495	Sandy Bridge Xeon
Xeon(R) X7460	0.93	-	6x4 real cores
i7-3632QM	1.00	0.490	8 hyperthreadead cores

Table 3. Single-core contribution to the aggregate rate measured with the PROOF-Lite test. For hyperthreaded machines separate values are shown, for the real and hyperthreaded regimes, respectively. For comparison the results of similar measurements done on recent real CPUs are also shown.

4.2. I/O intensive tests
Figure 3 shows the scaling result obtained from the I/O intensive benchmark on the *n1-standard-8-d* single machine.

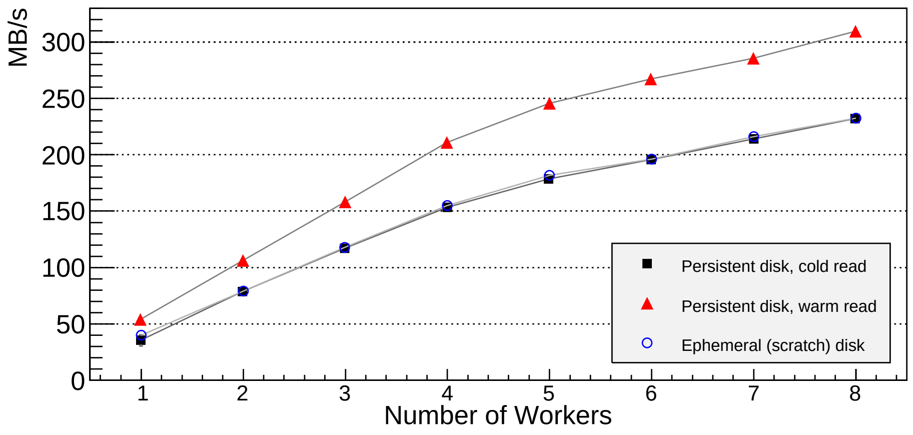

Figure 3. I/O performance scaling on a *n1-standard-8-d* machines

The purpose here was to measure both the persistent and the ephemeral disk performances in *cold* read mode. Cold read mode consists in resetting, before the run, the system cache for all files in the dataset, so that the data are always effectively read out from disks[7].

In Figure 3 we see that the ephemeral and persistent disks give the same results, which is consistent with the fact that they have similar implementations. The scalability curve shape is consistent with the presence of a saturation term, typical of cases where I/O is the factor limiting linear scalability. However, we observe that full saturation is not yet reached for 8 workers, which indicates that the machines benefit from performant network connection to the NAS.

Figure 3 also shows the result for *warm* reads, where the file system cache is not reset between runs. In this case the required buffers are found in RAM and the rates are basically limited by the available CPU for input data decompression.

[7] Technically this is done by calling `posix_fadvise(fd, 0, 0, POSIX_FADV_DONTNEED)`, where `fd` is the descriptor of the file open in read-only mode.

5. Full PROOF tests

5.1. Cluster setup

For the standard PROOF test we used 60+1 *n1-standard-8-d* VMs as described in Table 2. The machine used for the PROOF-Lite test was configured to start a *xproofd* daemon [8]; a snapshot of the hard drive was used as base to start the 60 worker machines.

5.2. CPU intensive tests

Figure 4 shows the scaling result obtained from the CPU intensive benchmark on the PROOF cluster.

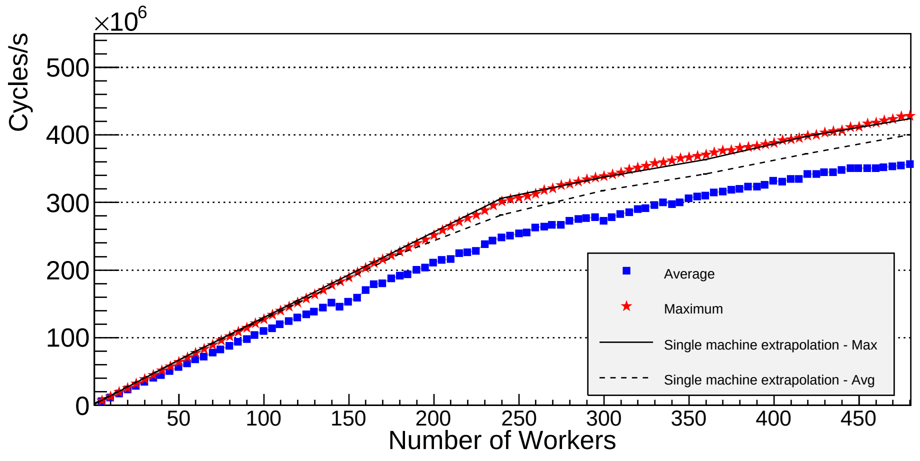

Figure 4. CPU performance scaling on the 480 core PROOF cluster. The full and dashed lines show the extrapolation of the single machine measurement under the assumption of ideal scalability of all PROOF components.

The measured maximum rate is in very good agreement with the scale up of the result obtained on the single *n1-standard-8-d* machine. This indicates that work distribution in PROOF scales very well at least up to 480 workers. For the average rate we observe instead a deviation from the expected behaviour scaling up the single machine result, indicating that in a full PROOF setup,the serial components (mostly initialization and termination) have a larger weight than in PROOF-Lite [9]; the quantification of the effect depends on the type of query and needs to be estimated case by case.

5.3. I/O intensive tests

The I/O intensive test with PROOF gives the result shown in Figure 5. The cold reads here show a clear saturation pattern, with a maximum aggregated rate of 6 GB/s obtained with 4 workers per machine, corresponding to a rate delivered by each worker of about 100 MB/s. While being remarkable, this is less than what we obtained on the single machine; the shape of the curve is also different between the two cases. We believe that the difference is due to network topology of the overall system.

From Figure 5 it seems that PROOF clusters with four workers per node are the optimal solution for PROOF clusters on these facilities.

[8] The daemon `xproofd` is the daemon starting the PROOF remote processes; it is derived from `XRootD`.
[9] The observed behaviour of the average rate is in agreement with the results of similar measurements on the Frakfurt Cloud (A.Manafov, GSI Darmstadt, private communication).

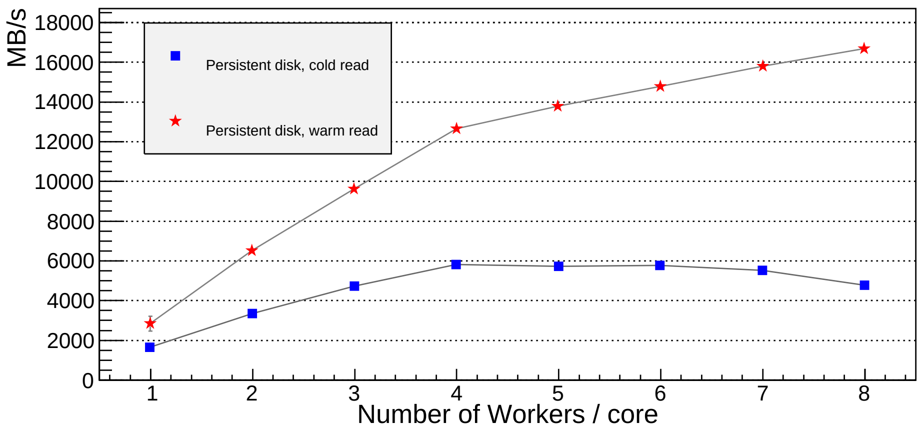

Figure 5. I/O performance per worker-in-node scaling on the 480 core PROOF cluster

6. Related work

As reported elsewhere in [9] ATLAS has run a trial project from August 2012 to April 2013 for a total of about 5 M core-hours allocated. They mostly used the facility for large scale Monte Carlo production was run on GCE for about two months using about 500 CPU (4000 cores) for a total of 214 M events generated [9].

7. Summary and outlook

In this paper we presented the results of running PROOF standard benchmarks on GCE. Overall the experience has been quite positive, the system was very stable and we could run smoothly the large number of queries `TProofBench` needs to run.

In terms of performance we obtained very good numbers both for CPU and disk I/O, with aggregate rates of 6 GB/s, corresponding to a per-node rate of about 100 MB/s.

Overall we think that GCE may represent a viable solution to cope with spikes in demand for computational resources for HEP data analysis.

References

[1] T Bell at al., *Review of CERN Data Centre Infrastructure*, 2012 J. Phys.: Conf. Ser. 396 042002.
[2] M Sevior, T Fifield and N Katayama, *Belle Monte-Carlo production on the Amazon EC2 cloud*, 2010 J. Phys.: Conf. Ser. 219 012003.
[3] https://cloud.google.com/products/compute-engine. See also https://developers.google.com/compute/ for detailed instructions.
[4] See https://developers.google.com/compute/docs/machine-types, also for the definition of the Google Compute Engine Unit (GCEU).
[5] http://xrootd.org
[6] http://root.cern.ch/drupal/content/proof
[7] D Berzano et al., *PROOF as a Service on the Cloud: a Virtual Analysis Facility based on the CernVM ecosystem*, this conference.
[8] S Ryu, G Ganis, *The PROOF benchmark suite measuring PROOF performance*, 2012 J. Phys.: Conf. Ser. 368 012020
[9] R Sobie et al., *ATLAS Cloud Computing R&D*, this conference