Explore the massive Volunteer Computing resources for HEP computation

Wenjing Wu[1]
IHEP
19B Yuquan Road, Beijing, 100049 China
E-mail:wuwj@ihep.ac.cn

David Cameron[2]
Department of Physics, University of Oslo
P.b. 1048 Blindern, N-0316 Oslo, Norway
E-mail: david.cameron@cern.ch

Abstract. It has been over a decade since the HEP community initially started to explore the possibility of using the massively available Volunteer Computing resource for its computation. The first project LHC@home was only trying to run a platform portable FORTRAN program for the SixTrack application in the BOINC traditional way. With the development and advancement of a few key technologies such as virtualization and the BOINC middleware which is commonly used to harness the volunteer computers, it not only became possible to run the platform heavily dependent HEP software on the heterogeneous volunteer computers, but also yielded very good performance from the utilization. With the technology advancements and the potential of harvesting a large amount of free computing resource to fill the gap between the increasing computing requirements and the flat available resources, more and more HEP experiments endeavour to integrate the Volunteer Computing resource into their Grid Computing systems based on which the workflows were designed. Resource integration and credential are the two common challenges for this endeavour. In order to address this, each experiment comes out with their own solutions, among which some are lightweight and put into production very soon while the others require heavier adaptation and implementation of the gateway services due to the complexity of their Grid Computing platforms and workflow design. Among all the efforts, the ATLAS experiment is the most successful example by harnessing several tens of millions of CPU hours from its Volunteer Computing project ATLAS@home each year. In this paper, we will retrospect the key phases of exploring Volunteer Computing in HEP, and compare and discuss the different solutions that experiments coming out to harness and integrate the Volunteer Computing resource, finally based on the production experience and successful outcomes, we envision the future challenges in order to sustain, expand and more efficiently utilize the Volunteer Computing resource. Furthermore, we envision common efforts to be put together in order to address all these current and future challenges and to achieve a full exploitation of Volunteer Computing resource for the whole HEP computing community.

International Symposium on Grids and Clouds 2018 -ISGC 2018-
16-23 March 2018

[1]Wenjing Wu

1. Introduction

The concept and practice of Volunteer Computing was started in the late 1990s with the launch and big success of the SETI@home [1] project. SETI@home not only succeed in harvesting a big amount of free computing resource from the volunteer computers, but also became a popular cultural phenomenon, with stories, news and interviews appearing on the media. In order to generalize the technology and make it beneficial to other scientific computing groups, the original SETI@home team rewrote the software BOINC(Berkeley Open Infrastructure for Network Computing) [2], making it into a generic middleware which can harness the idle CPU cycles from geographically distributed and heterogeneous computers for scientific computing.

BOINC soon became the most popular middleware for Volunteer Computing, around 50 science groups adopted it to set up their @home projects. As of the time of writing this paper, the real time computing power (measured in the recent 24 hours) exploited by BOINC is 18.123 PetaFLOPS which is contributed by 176149 volunteers with their 675019 volunteer computers. Among which, there are some extremely successful projects, such as Eintein@home [3] which yields 5.6 PetaFLOPS real time computing power after the big scientific discovery from the LIGO experiment. The major motive from the volunteers is their enthusiasm and eagerness to participate in the development of science, so the big and well known scientific research projects such as SETI (SETI@home), LIGO (Eintein@home) tend to gain much more computing power from the community than the others. This is an advantage that the HEP experiments can take, as most of them are well known and established, and also have good outreach to the public.

2. A time line for Volunteer Computing in HEP

With the popularity and potential of gaining a large amount of extra computing resource, Volunteer Computing also drew the attention of the HEP computing community. The HEP experiments can also benefit from using the volunteer community for their outreach purposes. The appliance and development of Volunteer Computing in the HEP field can break into a few phases according to the technology evolvements and scale in the past decade.

2.1. The starting point

LHC@home [4] is the first Volunteer Computing project in the HEP field, and it was started in 2004, as a gift to the 50th anniversary of CERN. The project features an application named SixTrack which simulates particles traveling through the LHC ring. As the SixTrack application is a small FORTRAN program, it was possible to port it to various platforms such as Windows, Linux and Mac OS. With this advantage, immediately the LHC@home project attracted intensive attention from the Volunteer Computing community to the HEP field and easily yielded a sustainable 20Teraflops of real time computing power after its launch.

2.2. The challenges to apply Volunteer Computing to big HEP experiments

With the successful case of LHC@home, people started to explore the possibility of running computing tasks from the HEP experiments on volunteer computers. However there are a few challenges which were commonly faced by these experiments: 1) The HEP experiment software is very platform dependent, namely it only runs on Scientific Linux, and would cost a lot of manpower to port the software to other platforms such as Windows and Mac OS. This is a huge disadvantage because over 90% of the volunteer computers are Windows machines, and only less than 5% are Linux machines; 2) All the HEP experiments already have Grid Computing platforms as their main computing infrastructure, and the workflows are also designed based on the concept of Grid Computing. Particularly, Grid Computing assumes all the computing resources are trustable and requires each work node to store the grid credential for interacting with the grid services. This is contrary to the untrusted feature of volunteer computers; 3) The HEP experiment software is also big in its release size, i.e., a single release of ATLAS software can reach over 20GB. It seems impractical to "copy" this amount of software to the volunteer computers as most of them have very limited bandwidth and hard disk resources.

Due to the above challenges and the lack of supporting technologies, the process of applying Volunteer Computing to the big HEP experiments was very slow.

2.3. The development of key technologies

The key technology required to address challenge 1 is virtualization which would convert the heterogeneous volunteer computers into the "volunteer clouds" [5] and provide the suitable operating system and software environment that all the HEP experiment computing requires. As early as 2004, people from various BOINC projects started to experiment using VirtualBox as the hypervisor to run on the volunteer computers in order to run platform dependent computing tasks. The key point is the development of a BOINC wrapper application which can control the life time and status of the virtual machines on the volunteer computers, then computing tasks can be run inside the virtual machines. In the beginning, there were different versions of BOINC VirtualBox wrappers from different projects, and in 2006, finally all of them merged into one common version named vboxwrapper. Hence it became possible to run the big HEP experiments computing tasks on any volunteer computers.

However the performance was not ideal, due to performance loss from using VirtualBox, and the cost to distribute a big image which contains the big software repository. The emergence of CernVM [6] and CVMFS helped to improve the performance and hence solved the challenge 3. CernVM provides ultralight virtual machine image for HEP computing, especially the micro CernVM image make it possible to create a working image within one hundred MB. CVMFS is a network file system which nowadays is widely used for HEP software distribution. One advantage CVMFS offers to Volunteer Computing is its on demand requesting feature, so it is not necessary to cache all the software release to the virtual machine image, and anything needed by the computing tasks can be requested on demand. In real practice, in order to balance the size of the image and the cost of downloading the software required by the computing task, it is common to run the computing task (usually an event simulation job) in the initial image, which caches a certain amount of basic software in the image, and also downloads a small amount of extra software during the job running. For the ATLAS case, the software cached in the image is around

1GB, and an extra tens of MB needs to be downloaded for each job. This usually includes condition database files.

Also the continuous improvements from the hypervisor VirtualBox helps to reduce the performance loss and bugs introduced to the volunteer computers.

The vboxwrapper keeps evolving as it is being used by more Volunteer Computing projects. It added new features such as supporting multi core jobs, console control and graphic interface from the BOINC GUI. The support of multi core jobs greatly saves the memory/hard disk/bandwidth usage on the volunteer computer with the same CPU utilization, so it attracts more volunteer computers to participate in the HEP @home projects. The console control make it easier for volunteers to report error messages to the projects, and the graphic interface is a nice feature to reward and attract volunteers with visualization from the computing tasks, i.e. real time display of the simulated events.

2.4. Apply Volunteer Computing to big HEP experiments

The development of the above key technologies paved the road for applying Volunteer Computing to the big HEP experiments. Respectively LHCb started its development in 2012, and launched the project LHCb@home to the public in 2016; ATLAS started its development in the middle of 2013 and launched the ATLAS@home [7-8] project to the public a few months afterwards; CMS also released its CMS@home [9] in 2016. Other experiments such as BelleII and BESIII also started their development of Volunteer Computing projects in 2013, but not released their projects to the public yet.

3. Different implementations in HEP experiments

3.1. ATLAS@home

ATLAS@home was the first Volunteer Computing project released to the public from the big HEP experiments. This is due to the flexibility of the ATLAS Grid Computing platform PanDA [10], which makes it easier to solve both challenges.

In order to integrate the ATLAS@home resource into PanDA, ATLAS uses the ACT (ARC Control Tower) [11] and ARC CE [12] to forward the job to the BOINC server. As shown in Figure 1, ARC CE caches the job and its data with its grid credential, then use the BOINC plugin to “submit” the job to the BOINC server. When the BOINC client which runs on the volunteer computer requests a job, both the job and input data are sent to the volunteer computer without requiring any grid credentials. The job is run through the ATLAS pilot [13] which can “recognize” the pre-downloaded data, and start the payload. Upon finishing the payload, the output data is uploaded to the BOINC server without any grid credential, and the BOINC assimilator service uploads or copies the data to the ARC CE. When ARC CE discovers the job is finished, it uploads the data to the Grid SE with the grid credential stored in ARC CE. In this diagram, no grid credential is required on either the BOINC server or clients, and ARC CE acts as the gateway which bridges the two zones (Grid Computing and Volunteer Computing).

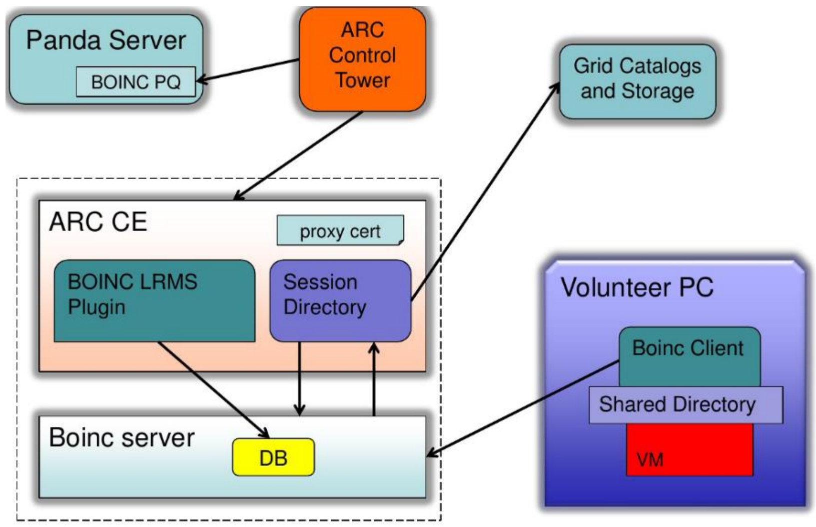

Figure 1. The architecture of ATLAS@home

3.2. CMS@home

It is more complicated for the CMS to integrate the Volunteer Computing resource into their Grid Computing system Crab3, as Crab3 uses the “push” model to send jobs to the work nodes, while BOINC uses the “pull” model to dispatch jobs to the work nodes.

In order to bridge these two different workflows, CMS developed a gateway service named DataBridge to bridge the CMS grid job submission system Crab3 and the volunteer computing resource. The DataBridge has two main functions: 1) Acts as a Crab3 plugin, which receives job descriptions from the CMS job submission and stores them in a message queue; 2) Stages input files from the SE and output files to the SE. In stage-in, Crab3 puts input data into the Ceph input bucket, then the volunteer computer gets the input data from the Ceph input bucket; In stage-out, the volunteer computer puts the output data into the Ceph output bucket, and a cron job running on the DataBridge gets the output data from the Ceph bucket and uploads it to the SE. Grid credentials are stored on the DataBridge for its interactions with other grid services.

From the volunteer computer side, the following operations happen when it requests and finishes a job: 1) Pulls the job from the message queue on the DataBridge, and gets input sandbox data from the DataBridge; 2) Downloads the input data from the Ceph input bucket; 3) Puts output sandbox data to the DataBridge; 4) Uploads the output data to the Ceph output bucket. The volunteer computers can use the BOINC authenticator to access the two Ceph buckets.

3.3. LHCb@home

The challenge for LHCb comes from the merge of the two authentication methods (GSI authentication for grid services and untrusted feature of volunteer computers). LHCb uses DIRAC [14] as its Grid Computing platform, and one can’t easily separate the payload running from all the operations requiring grid credentials because of the implementation of the DIRAC pilot.

LHCb also implemented a gateway service named WMSSecureGW to bridge the DIRAC and volunteer computers. The gateway appears to be a DIRAC client and pilot to other DIRAC services, and stores all the required credentials to interact with the DIRAC services and SE(s) to fetch jobs and stage data. On the volunteer computer, it has a fake credential (a self-signed certificate) which allows the volunteer computer to interact with the gateway for credential required operations (fetching job and staging data).The fake certificate does not serve any authentication purpose, but simply to keep the DIRAC pilot running as the way it is on the grid work nodes. This approach keep the DIRAC pilot as the way it works on the grid work nodes.

4. The current scale of HEP @home projects

In March 2016, all the @home projects at CERN merged into one project LHC@home to consolidate the BOINC services. LHC@home became a big umbrella project which hosts several applications, with each application corresponding to one HEP experiment at CERN. Figure 2 shows the current daily CPU time of good jobs of the different applications in LHC@home. At peak, the daily CPU time reaches 25,000 CPU days, and average core power of the CPU in LHC@home is about 10HS06, so the computing power of LHC@home is around 400KHS06 considering the fact that the average CPU utilization of clusters is around 60%. Take ATLAS for example, it serves as one of the biggest simulation sites, and contributes 2% to the whole ATLAS computing resources which includes Grid Computing, Cloud Computing and HPC in the past 3 years. This proves the contribution and potential that Volunteer Computing brings to the HEP field.

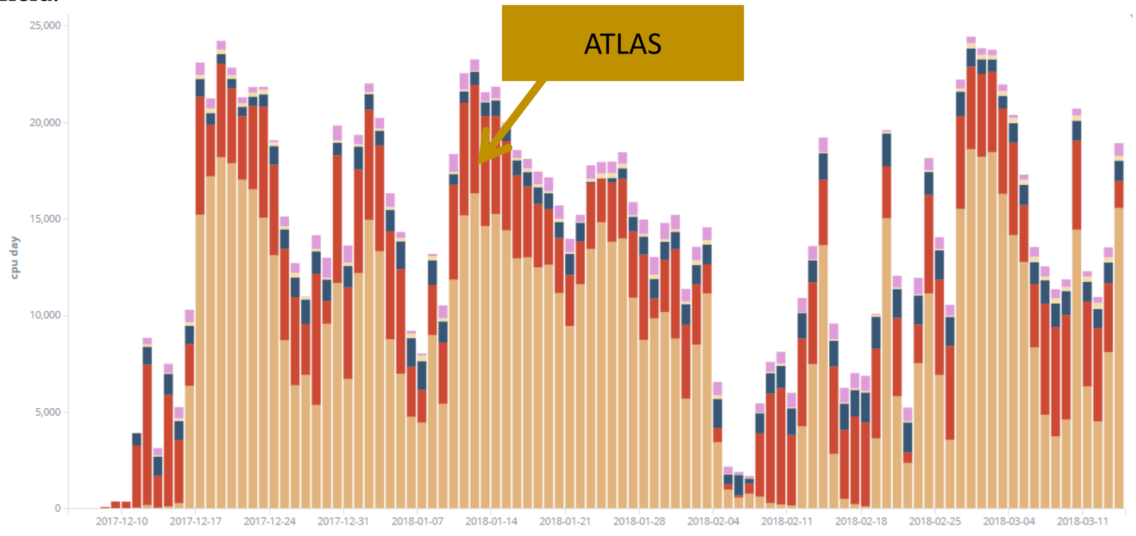

Figure 2. The daily CPU time (in CPU days) of LHC@home

5. Recent development and new practice

Table 1. Average daily CPU utilization in BEIJING_LCG2 site with backfilling in 100 days

	#cores	**Avg. walltime (days) per day**	**Avg. cputime (days) per day**	**CPU Utilization (%)**
BEIJING_Grid	468	411	307	65.60
BEIJING_BOINC	468	437	110	23.50
Total	468	848	417	89.10

A recent trend in HEP volunteer computing is using containerization instead of virtualization, which provides a lightweight solution, and ATLAS@home is leading this effort by using Singularity to replace VirtualBox to provide the required computing environment. With the lightweight solution, ATLAS@home can also be used to backfill the busy grid sites. In this use case, the ATLAS@home jobs are niced with lower priority compared to the grid jobs, so the ATLAS@home jobs would only use the CPU time when the grid jobs voluntarily releases it. This practice has been running on the BEIJING_LCG2 Tier2 site for about 6 months, and as shown in table 1, it proves that when the Tier2 site is fully loaded (with the wall time utilization rate of 87.8%), the ATLAS@home jobs can still exploit an extra 23.5% of CPU from the Tier2 site which would otherwise be wasted, and this prompts the total CPU utilization rate of the site to 89.1%.

6. Common challenges in the future

With the appliance and growing of these HEP @home projects, some common challenges keep emerging.

- ✓ Needs more flexible BOINC scheduling. The BOINC default scheduling does not provide the best efficiency in utilizing the available Volunteer Computing resource, especially for the multi core applications, they require more flexing scheduling policies, such as matching job size to the available cores on the volunteer computer.
- ✓ Discontinued BOINC development support due to fund losing. All the projects need to develop the extra BOINC features they need.
- ✓ Scalability issue. IO is obvious a bottleneck for scaling up the projects, even for simulation jobs which last a couple of CPU hours, it still requires over hundreds of MB input and output data.
- ✓ Outreach. How to attract more volunteer computers from the current BOINC user and computer pool?

Some common efforts can also be put together in working towards generic solutions for the following issues.

- ✓ The creation and distribution of virtual machine images.
- ✓ The development of common BOINC features, such as scheduling policies.
- ✓ Providing a common interface for all HEP projects to the volunteers.
- ✓ Providing graphic interface interactions to volunteers, such as event display from the simulation jobs that all HEP @home projects run.
- ✓ Using BOINC to exploit extra CPU resources from our grid sites.

7. Summary

It has been over a decade since the Volunteer Computing was firstly applied to the HEP field, however its appliance to the big HEP experiments only started 4 years ago thanks to the evolvements and advancements of some key technologies. Since then all the major HEP experiments started their @home projects which have harvested very considerable computing power. These projects keep growing as more volunteer computers join, and more challenges appear and common efforts should be put together inside the community to seek solutions.

8. Acknowledgements

This project is supported by the Chinese NSF grants "Research on fine grained Event Service for the BESIII offline software and its scheduling mechanism (No.11675201)" and "Research on BESIII offline software and scheduling mechanism on desktop grid （No.11405195）". We also thank sincerely all our volunteers for their computing contributions to the HEP field!

References

[1]. Anderson D P, Cobb J, Korpela E, et al. SETI@ home: an experiment in public-resource computing [J]. Communications of the ACM, 2002, 45(11): 56-61.

[2]. Abbott B P, Abbott R, Adhikari R, et al. Einstein@ Home search for periodic gravitational waves in early S5 LIGO data [J]. Physical review d, 2009, 80(4): 042003.

[3]. Anderson D 2004 BOINC: a system for public-resource computing and storage Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing, GRID 04 (IEEE Computer Society) pp 4-10 (BOINC)

[4]. Buncic P, Sanchez C A, Blomer J, et al. CernVM–a virtual software appliance for LHC applications[C]//Journal of Physics: Conference Series. IOP Publishing, 2010, 219(4): 042003.

[5]. Sanchez C A, Blomer J, Buncic P, et al. Volunteer clouds and citizen cyberscience for LHC physics[C]//Journal of Physics: Conference Series. IOP Publishing, 2011, 331(6): 062022. (LHC@home)

[6]. Høimyr N, Marquina M, Asp T, et al. Towards a Production Volunteer Computing Infrastructure for HEP[C]//Journal of Physics: Conference Series. IOP Publishing, 2015, 664(2): 022023. (LHC@home)

[7]. Cameron D. ATLAS@ Home: Harnessing Volunteer Computing for HEP[R]. ATL-COM-SOFT-2015-018, 2015. (ATLAS@home)

[8]. Cameron, David, et al. Volunteer Computing Experience with ATLAS@ Home. No. ATL-SOFT-SLIDE-2016-637. ATL-COM-SOFT-2016-069, 2016.

[9]. Field L, Borras H, Spiga D, et al. CMS@ home: Enabling Volunteer Computing Usage for CMS[C]//Journal of Physics: Conference Series. IOP Publishing, 2015, 664(2): 022017. (CMS@home)

[10]. Maeno T 2008 PanDA: distributed production and distributed analysis system for ATLAS Journal of Physics: Conference Series vol 119 (IOP Publishing) p 062036 (PanDA)

[11]. Filipcic A 2011 arcControlTower: the System for Atlas Production and Analysis on ARC J. Phys.: Conf. Ser. vol 331 p 072013

[12]. Ellert M, Gr_nager M, Konstantinov A et al. 2007 Future Gener. Comput. Syst. 23 219{240 ISSN 0167-739X (ARC CE)

[13]. Nilsson P, Caballero J, De K, et al. The ATLAS PanDA pilot in operation[C]//Journal of Physics: Conference Series. IOP Publishing, 2011, 331(6): 062040.(ATLAS pilot)

[14]. Tsaregorodtsev A. DIRAC distributed computing services[C]//Journal of Physics: Conference Series. IOP Publishing, 2014, 513(3): 032096. (DIRAC)