Bringing ATLAS production to HPC resources - A use case with the Hydra supercomputer of the Max Planck Society

2015 J. Phys.: Conf. Ser. 664 092019

(http://iopscience.iop.org/1742-6596/664/9/092019)

Journal of Physics: Conference Series **664** (2015) 092019 doi:10.1088/1742-6596/664/9/092019

Bringing ATLAS production to HPC resources - A use case with the Hydra supercomputer of the Max Planck Society

J A Kennedy[2], S Kluth[1], L Mazzaferro[1] and Rodney Walker[3] on behalf of the ATLAS Collaboration

[1]Max Planck Institut für Physik, Föhringer Ring 6, 80805 München
[2] Rechenzentrum Garching, Giessenbachstraße 2, 85748 Garching
[3] Ludwig Maximilians Universität München

Abstract. The possible usage of HPC resources by ATLAS is now becoming viable due to the changing nature of these systems and it is also very attractive due to the need for increasing amounts of simulated data.

In recent years the architecture of HPC systems has evolved, moving away from specialized monolithic systems, to a more generic linux type platform. This change means that the deployment of non HPC specific codes has become much easier. The timing of this evolution perfectly suits the needs of ATLAS and opens a new window of opportunity.

The ATLAS experiment at CERN will begin a period of high luminosity data taking in 2015. This high luminosity phase will be accompanied by a need for increasing amounts of simulated data which is expected to exceed the capabilities of the current Grid infrastructure.

ATLAS aims to address this need by opportunistically accessing resources such as cloud and HPC systems. This paper presents the results of a pilot project undertaken by ATLAS and the MPP/RZG to provide access to the HYDRA supercomputer facility. Hydra is the supercomputer of the Max Planck Society, it is a linux based supercomputer with over 80000 cores and 4000 physical nodes located at the RZG near Munich.

This paper describes the work undertaken to integrate Hydra into the ATLAS production system by using the Nordugrid ARC-CE and other standard Grid components. The customization of these components and the strategies for HPC usage are discussed as well as possibilities for future directions.

1. Introduction

After a 2 year technical stop, due to system upgrades, the Large Hadron Collider (LHC) is planned to start a new operational season in spring 2015.

During this second phase, called $Run2$, **LHC** is expected to achieve an energy in the center of mass of $14TeV$, a luminosity of $10^{34}cm^{-1}s^{-2}$ and a bunch crossing every $25ns$. The pile up, for the two major experiments ATLAS and CMS, is also expected constantly growing during the data taking period $2015 - 2018$.

This scenario will lead to an increase of the computing resource needed by the experiments.[1]

During the first **LHC** run period, $Run1$ - March 2010 - December 2012, the ATLAS experiment continually used more CPU resources than made available via the pledges from partner centers. This can be seen by looking at figures 1 and 2, and similar behavior was observed for other LHC experiments.

To be able to satisfy the over-pledged computing requests, ATLAS took benefit from other non-dedicated resources such as HPC system, domestic and commercial Cloud providers. An opportunistic approach based on the temporary not-used resources has been mainly applied. This trend is expected to continue also in *Run2*.

This document describes the technical aspects and the strategies adopted for integrating the MPG HYDRA Supercomputer inside the ATLAS Grid infrastructure, using standard Grid tools provided by Nordugrid community.

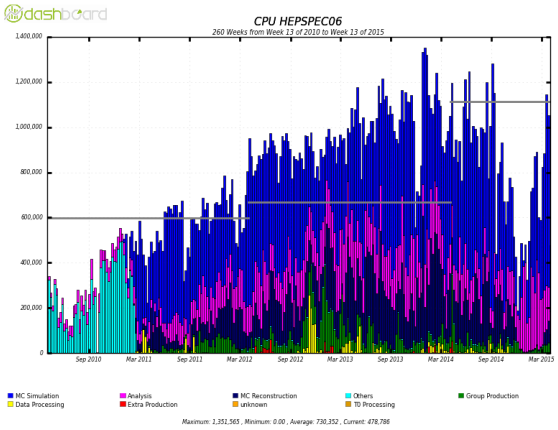

Figure 1. This plots shows the *cpu hepspec06* used by the different types of ATLAS jobs, in the *Run*1 period from April 2010 to March 2015. The grey line represents the resource pledged to ATLAS Collaboration for the same period.

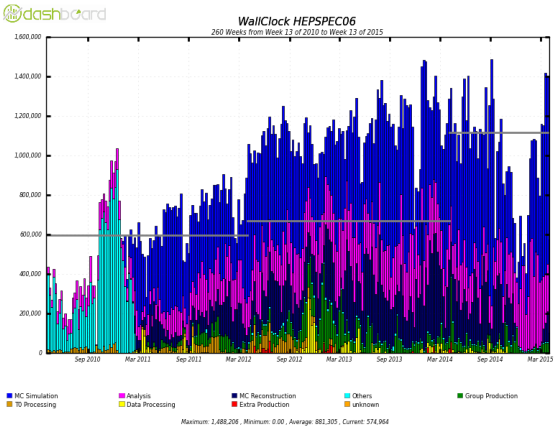

Figure 2. This plots shows the *wallclock time* consumed by the different types of ATLAS jobs, in the *Run*1 period from April 2010 to March 2015. The grey line represents the resource pledged to ATLAS Collaboration for the same period.

2. The RZG HYDRA Supercomputer

The *iDataPlex HPC system HYDRA* consists of about 3500 nodes equipped with Intel Ivy Bridge processors (20 cores per node at $2.8GHz$), and 610 nodes with Sandy Bridge-EP processors (16 cores per node at $2.6GHz$). Finally, 350 nodes of Ivy Bridge type are equipped with accelerators cards (338 nodes with 2 NVIDIA K20X GPGPUs each, 12 nodes with 2 Intel Xeon Phi cards each). The majority of the nodes have $64GB$ of main memory, while 200 Ivy Bridge plus 20 Sandy Bridge have 128 GB. The higher memory allows these machines to run jobs with extremely high memory requirement. The jobs are managed by the *IBM LoadLeveler V5.1* batch system which supports the Linux system [3].

The nodes are organized in 5 domains: three with 628 nodes, one with 1800 and another one with the ones equipped with the accelerators cards. Inside each domain networking is based on InfiniBand *FDR14* technology, which ensure fast communications. The connection between the different domains is much weaker and for this reason the batch jobs are confined to run into one domain. In common with many other HPC systems, **HYDRA** nodes are not equipped with local disks, the storage area based on **GPFS** shared filesystem: 26 I/O nodes provide more than 5 PetaBytes of disk storage. This space is used for storing data, hosting jobs binaries and saving the results.

These kind of systems are designed to run tasks being CPU intensive, highly parallelized and with low I/O. These requirements contrast somewhat with sevaral types of **ATLAS** job, for

instance Data Analysis, which are I/O intensive. Following this, the type of ATLAS jobs which run on HYDRA is limited to those which best suit the system, see section 4.

The system can reach a peak performance of about $1.7 PetaFlop/s$, while the accelerator part alone up to $1 PetaFlop/s$.

It is worth to note that the **HYDRA** system can be accessed only from within the Max Plank Gesellschaft network, **MPG**: in this contest the inbound and outbound connectivity from the nodes with the World Wide Web is completely avoided included with the **ATLAS** Grid. The users interact with the system connecting to several login nodes and using standard communication protocols (*ssh*). There are no grid services provided by default: the data management as the job submission is made directly by the users.

The operative system installed on the nodes is SLES 11 sp3 for $x86\_64$ platform [2]: this is the enterprise version of SUSE Linux OS. As a consequence the ATLAS software, based on Scientific Linux 6 (**SL6**), can be easily integrated.

Due to the high resource requirements needed by classic HPC jobs, the **HYDRA** system unlikely reach a slot coverage of 100%. The unused slots, although temporary in nature, can be filled opportunistically by lighter, short running jobs. With more than 83000 cores and even with a mean slots coverage of 98%, in principle **ATLAS** could take the opportunity of using more than 1500 slots.

In the next section it will be shown how it has been possible to integrate the **HYDRA** system with the **ATLAS** Grid using the Nordugrid ARC computing element. The solutions presented can be easily generalized to other HPC systems and experiment using the grid standard protocols.

3. The integration between HYDRA and the Grid

As already introduced in the previous sections, **HYDRA** is neither accessible from outside the **MPG** network nor has any kind of *Grid* interface. As a consequence, a full integration with the ATLAS Grid needs:

- an interface to the **ATLAS** Workload management system, *PanDA*[4], for jobs scheduling;
- a service which takes care of the staging in and out the data to and from the **ATLAS** *Grid*;
- an interface to interact with the **HYDRA** *LRMS*, the *IBM LoadLeveler*.
- a local web proxy (SQUID) which allows ATLAS jobs to retrieve detector conditions data etc.

3.1. The ARC Computing Element

A solution satisfying the requirements, previously exposed, is the *ARC Computing Element*, **ARC-CE**, developed by *Nordugrid*[5].

Throught its interface (fig. 3):

- it provides a standard grid interface to the *LRMS*, for jobs submission, management and information retrieval about computing resources. Only authenticated users, via *X.509 certificates*, can access. Some adjustment to the code has been made here to enable a complete and optimized integration with **HYDRA** *LRMS*;
- it is able to stage in the input files, caching them to avoid resources wasting, and to stage out the results to a remote storage service (**DDM** in ATLAS context);
- it monitors the status of the jobs and makes this information available to the *Grid*.

The **ARC-CE** has been installed in a dedicated virtual machine with *Scientific Linux* operative system, a very light hardware configuration (2 cores, 8GB of ram) and inbound and outbount internet connectivity.

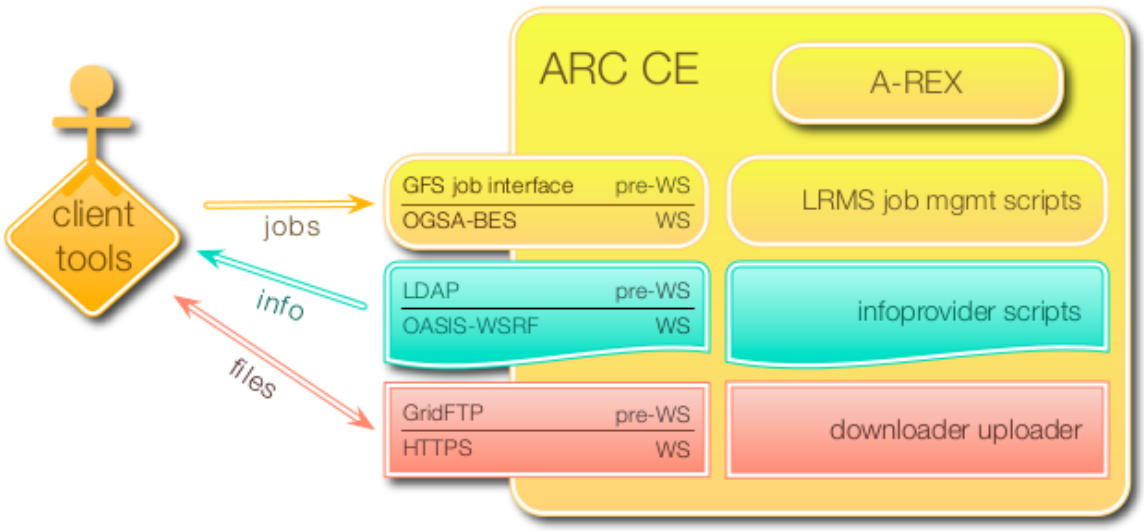

Figure 3. The ARC-CE interfaces and components [6]

The **ATLAS** software is provided via the CernVM filesystem [7], `cvmfs`, which is a read-only `POSIX` file-system designed to deliver scientific software in VM and physical machines. It is built on top of `fuse` and uses the standard `HTTP` protocol for file distribution.

Some customization has been applied to realize the full integration between **HYDRA** and ATLAS

- a new unix user has been created inside the **ARC-CE** and mapped with the unix user granted to run jobs inside **HYDRA**;
- the **HYDRA** `GPFS` file system was exported to the **ARC-CE** and the area owned by the Unix user was mounted in read and write mode. This area, shared between the compute nodes and the **ARC-CE**, contains several directories, figure 4.
  The *cache*, *session* and *tmp* directories are used for data staging, caching and monitoring.
  Binary and lib area shares the software required from ATLAS jobs but neither present inside the compute nodes nor provided by `cvmfs`.
  The *runtime* directory is filled with configuration files called by the jobs when they land inside the compute nodes. This files set up the enviroment inside each node, included required ATLAS software, to run the jobs.
  Finally the `software` area is used to share the ATLAS software coming from `cvmfs`.
- the ATLAS software which is needed by the jobs, is manually synced, via `rsync`, inside the shared *software* directory. This is needed since the HYDRA nodes are diskless (no local cvmfs) and due to the limit of the `cvmfs` technology which prevent to mount it on top of `GPFS` shared file system. As a consequence, **HYDRA** can run only predefined tasks accordingly with the software present in the *software* directory.

3.2. Workflow

Following the schema on fig. 4, the entire work-flow can be summarized like this:

(i) The ATLAS jobs ready to be processed are sent via the *arcCotrolTower*[8] to the **ARC-CE**. This service take care of all the communications with the **PanDA**;

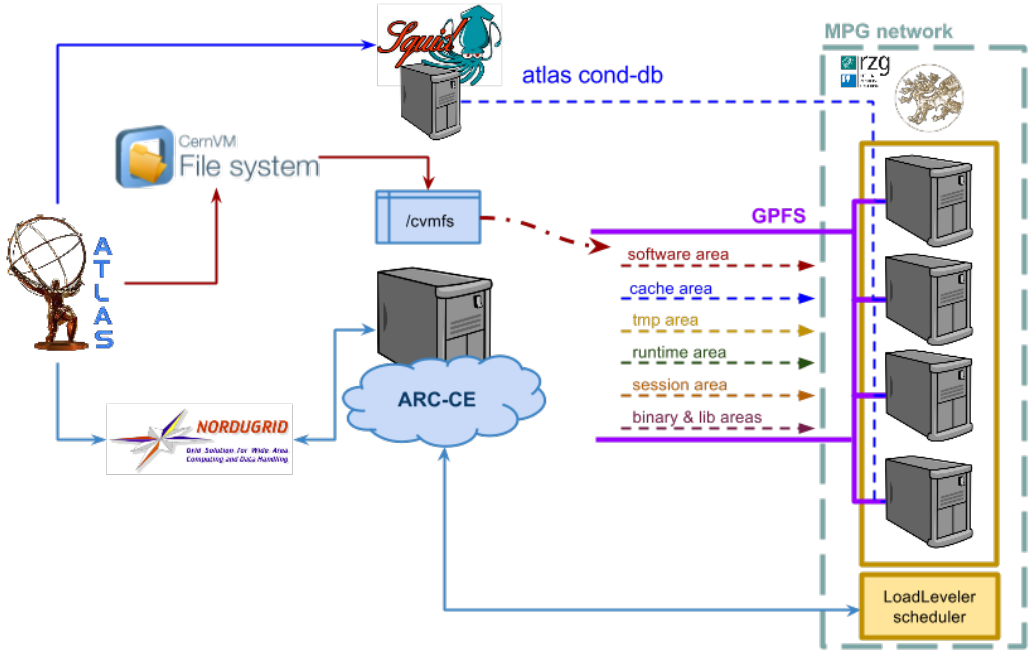

Figure 4. The HYDRA-ATLAS grid integration schema

(ii) The **ARC-CE**, using the information contained inside the job description file, downloads the input data into the shared *cache area* and submit the jobs to the *LoadLeveler*;

(iii) Once the job has been submitted the **ARC-CE**, interacting with the *LoadLeveler*, monitors the jobs status providing at the same time the information to **PanDA** via the *arcControlTower*

(iv) the jobs running inside **HYDRA** will find the required and pre-staged software and data directly inside the shared areas (*software*, *cache* and *binary and lib*).

(v) when one job is successfully finished the **ARC-CE** transfers the results directly to ATLAS.

It is worth to note that some customizations to the **ARC-CE** core code have been applied for a better integration with the **HYDRA** Supercomputer. These involve especially the interface with the *LoadLever* where, in some places, a rewrite of the code was necessary. Another area where some development was needed was the so called *cache cleaning* script, the **ARC-CE** script for automatic cache cleaning conflicted with the **HYDRA** `GPFS` file-system and was not able to manage the user quota. The script was modified to solve this issue. All these changes have been communicated to the Nordugrid developers.

4. Results

In this sections, statistics and results about the **HYDRA** usage for the ATLAS jobs will be presented. All the information has been retrieved using the ATLAS Dashboard[9], considering a period of time which spans from 01 July 2014 to 01 March 2015.

In the first two figures, figure 5 and 6, **HYDRA**'s contribution is compared to the MPPMU ATLAS Tier2, which is also located at **RZG**. The parameters used are the CPU consumption for *good jobs* (jobs successfully finished), figure 5, and the total number of completed jobs, figure 6. In the first case the **HYDRA** fraction of *CPU* time is about 24%, while considering the

total amount of jobs completed it is about 17%. This difference is due to the characteristics of the jobs which can run inside **HYDRA**, and in general in modern Supercomputers: highly parallelized, low I/O and CPU intensive.

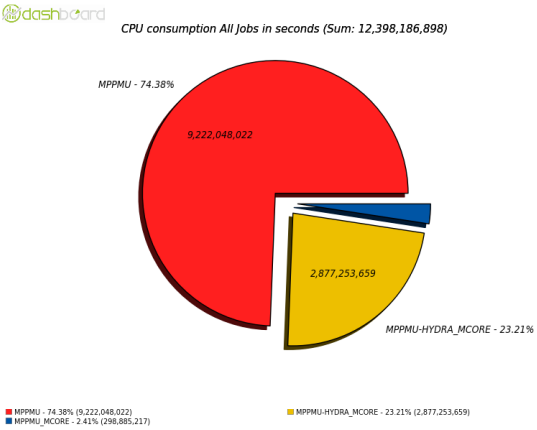

Figure 5. CPU consuption for successful jobs in seconds. HYDRA's contribution is about 24%

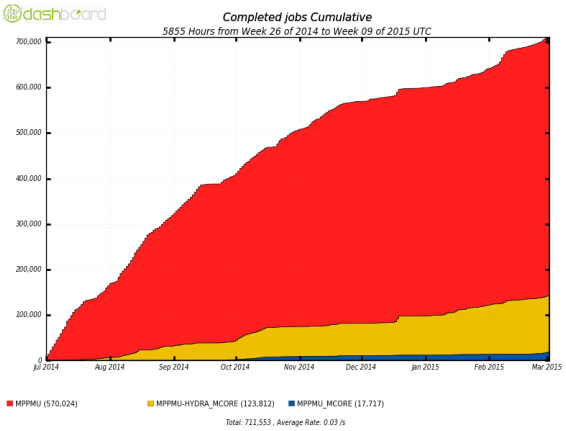

Figure 6. The total number of completed jobs processed at the MPPMU T2 and HYDRA. HYDRA's contribution is about 17%

In the next two figures, fig. 7 and 8, the *job efficiency*, defined as the ratio between the CPU time and the full walltime to run the job, is presented. Low values can be an indicator of high number of I/O processes, generally not suitable for HPC systems. In figure 7 it is also possible to note the effect of organizing jobs in a campaign for **HYDRA**, something that leads to intermittent usage, while for the MPPMU T2 a continuous flow can be seen. This is due to the fact that the most suitable jobs for **HYDRA** are *Monte Carlo simulations*. Monte Carlo jobs are ideal for HPC systems since they support parallelization; have low I/O and are CPU intensive. However they are only one part of the ATLAS production chain and this has led to a campaign like deployment with periods of high and low usage. Moreover only the most recent ATLAS code releases have the optimized code for running in such HPC systems. The results to date are very encouraging, however areas where improvements can be made are under consideration. Figure 7 shows that improvements in resource usage can be made by ensuring that a continuous flow of jobs is available, however this is also influenced by the availablility of free slots on the HPC systems. Additionally, as the usage of HPC systems progresses, the efficiency of the HPC multi-core jobs as shown in figure 8 should also improve.

Moreover, if we wish to understand the efficiency of the resource usage, it is worth comparing the usage of the Hydra system with the MPPMU tier2 w.r.t successful and failed jobs. Figures 9 and 10 show the successful, failed and deleted jobs which ran on **HYDRA** and the MPPMU Tier2. The percentage of failed jobs on **HYDRA** is slightly higher, approximately 4%, than on the MPPMU Tier2. This is an encouraging result, especially considering this also contains failed jobs from the initial test phases. The result displaying both the suitability of ATLAS simulation codes for HPC systems and the reliability of this solution for HPC utilization by ATLAS.

Finally figure 11 shows the type of jobs which ran on **HYDRA**. While this confirms the bias towards the more suitable MC simulations and it also shows some fraction of reconstructions can be performed inside the HPC systems. Monte-Carlo simulation jobs require around 15% of the compute resources of the **ATLAS** Grid. Moving these jobs onto HPC systems would allow ATLAS to make use of oportunistic resources and free up the generic Grid resources for analysis

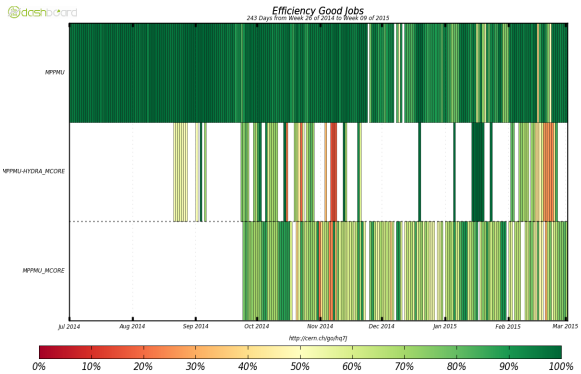

Figure 7. Efficiency of ATLAS good jobs, in time period (July 2014 - Mar 2015). Top=MPPMU, Middle=Hydra-Multicore, Bottom=MPPMU-Multicore.

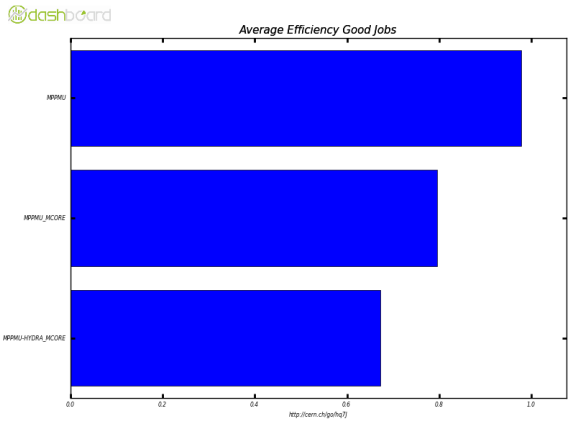

Figure 8. The average efficiency of ATLAS good jobs. Top=MPPMU, Middle=MPPMU-Multicore, Bottom=Hydra-Multicore.

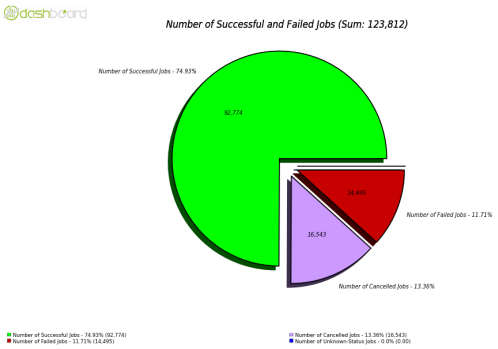

Figure 9. HYDRA: Percentage of successful(75%), failed(12%) and deleted(13%) jobs. The deleted jobs were canceled directly by ATLAS.

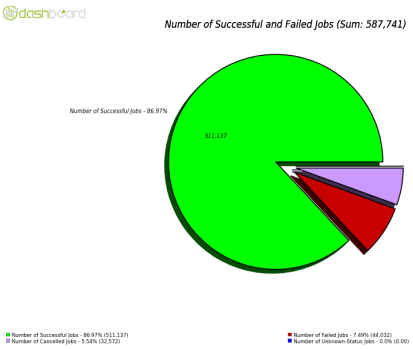

Figure 10. MPPMU Tier2: Percentage of successful(87%), failed(8%) and deleted(5%) jobs. The deleted jobs were canceled directly by ATLAS.

and other job types jobs.

5. Conclusions

The **HYDRA** Supercomputer is running ATLAS jobs, mainly *Monte Carlo simulation*, since more than one year. It is fully integrated into the ATLAS Grid infrastructure and as shown in section 4, its contribution, compared with MPPMU Tier2, has already reached a significant level.

This type of oportuinistic HPC resource usage presents benefits for both ATLAS and the institutes hosting the HPC systems. ATLAS can gain additional resources, allowing traditional Grid resources to be applied to data analysis tasks etc, and the HPC systems themselves can increase the occupancy by making use of these less resource hungry, short running jobs.

One of the major issue met during the integration was related to the usage of the **HYDRA** `GPFS` file-system: this was addressed by both the ATLAS community and Hydra Administrators,

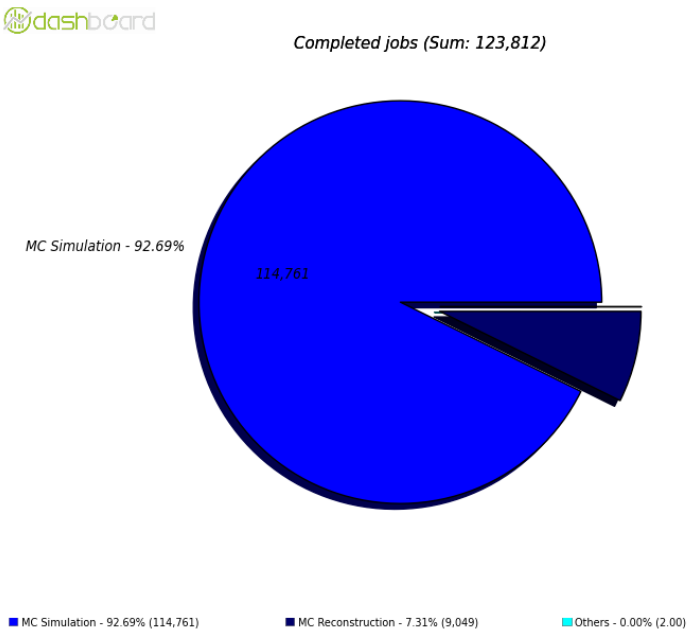

Figure 11. Breakdown of jobs types ran on HYDRA during the period April 2014 - March 2015

mainly to improve file access patterns (reducing the need to access many small files).

The major areas idetified for improvements are:

- MonteCarlo multi-core code: a further optimization in the code is needed to increase the job efficiency.
- ATLAS software distribution: moving from a manual synchronization of the ATLAS software in the shared area to an automatic deployment can increase the reliability of the entire system.
- Defining strategies for a better usage of the HPC opportunistic resources via *backfill* and *preemptive* job deletion.

References

[1] I. Bird et al, *Update of the Computing Models of the WLCG and the LHC Experiments*, CERN-LHCC-2014-014 / LCG-TDR-002 15/04/2014
[2] *SLES Home Page*, `https://www.suse.com/products/server/`
[3] IBM RedBooks, *Workload Management with LoadLeveler*, `http://www.redbooks.ibm.com/redbooks/pdfs/sg246038.pdf`
[4] Maeno T. et al., *Overview of ATLAS PanDA Workload Management*, J. Phys. Conf. Ser. 331
[5] The NorduGrid Collaboration. Web site. URL `http://www.nordugrid.org`.
[6] F. Paganelli et al, *ARC Computing Element, System Administrator Guide*, NORDUGRID-MANUAL-20 3/9/2014
[7] J. Blomer et al., *The CernVM File System, Technical Report*, CERN
[8] Andrej Filipcic for the ATLAS collaboration, *arcControlTower: the System for Atlas Production and Analysis on ARC*, J. Phys. Conf. Ser. 331 (2011) 072013
[9] ATLAS dashboard, `http://dashboard.cern.ch/atlas/`