ATLAS usage of the Czech national HPC center: HyperQueue, cvmfsexec, and other news

Michal Svatoš[1,*], *Jiří* Chudoba[1,**], and *Petr* Vokáč[2,***]

[1]Institute of Physics of the Czech Academy of Sciences, Na Slovance 1999/2, Prague, 18221, Czech Republic

[2]Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Břehová 7, Prague, 115 19, Czech Republic

Abstract. The distributed computing of the ATLAS experiment at the Large Hadron Collider (LHC) utilizes computing resources provided by the Czech national High Performance Computing (HPC) center, IT4Innovations. It is done through ARC-CEs deployed at the Czech Tier2 site, praguelcg2. Over the years, this system has undergone continuous evolution, marked by recent enhancements aimed at improving resource utilization efficiency. One key enhancement involves the implementation of the HyperQueue meta-scheduler. It enables a division of whole-node jobs into several smaller, albeit longer, jobs, thereby enhancing CPU efficiency. Additionally, the integration of cvmfsexec enables access to the distributed CVMFS filesystem on compute nodes without requiring any special configurations, thereby substantially simplify software distribution and broadening the range of tasks eligible for execution on HPC. Another notable change was the migration of the batch system from PBSpro to Slurm.

ATL-SOFT-PROC-2025-003
06 January 2025

1 Introduction

The distributed computing of the ATLAS experiment [1] at LHC is using computing resources of the Czech national HPC center IT4Innovations [2] through Czech Tier2 site (praguelcg2) [3]. There is significant distance between both centers (Figure 1), as praguelcg2 is located in Prague and IT4Innovations is located in Ostrava.

In 2024, ATLAS was using two HPC systems of the IT4Innovations: Karolina (jobs are being sent there since August 2021) and Barbora (used since January 2020).

Figure 1. Locations of praguelcg2 (Prague) and IT4Innovations (Ostrava)

*e-mail: Michal.Svatos@cern.ch
**e-mail: Jiri.Chudoba@cern.ch
***e-mail: Petr.Vokac@cern.ch

2 Job submission system

The system submitting ATLAS jobs to HPCs of IT4Innovation (Figure 2) works as follows: When the ARC-CE [4] receives a job from ATLAS [5], [6], it translates the job description into a script that can be run in the batch system, puts necessary files into a folder within a sshfs shared area and submits the job via a ssh connection to the HyperQueue server running on a login node. The HyperQueue server collects these job definitions and when there are enough of them, it submits jobs into the batch system. When the batch job starts, atlas-cvmfsexec [7] is mounted and then a HyperQueue worker starts and is filled with HyperQueue jobs (this is illustrated on Figure 3). In each HyperQueue job, the pilot wrapper starts, launching the pilot. The pilot contacts PanDA server through an http proxy, which is the praguelcg2 squid, to receive a payload (as there are only few open ports at each HPC). When it receives the payload, it gets the input file from the praguelcg2 storage via xroot or webdav and processes it. When the payload finishes, it sends outputs to the praguelcg2 storage, again, via xroot or webdav. When this is finished, the pilot will request another payload (if it can expect that the job can finish). When all HyperQueue jobs are finished, the HyperQueue worker finishes and atlas-cvmfsexec is unmounted.

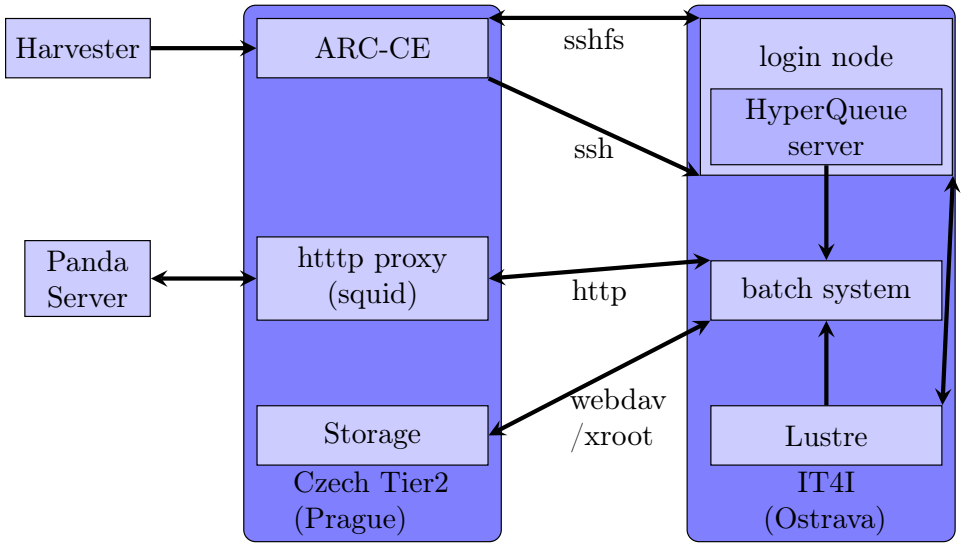

Figure 2. A scheme of the job submission system

3 HyperQueue

HyperQueue [8] is a tool designed to simplify execution of large workflows on HPC clusters. It supports PBSpro and slurm which was important when IT4Innovations transitioned from PBSpro to slurm. To allow submission to HyperQueue from the ARC-CE, only minor changes were needed in ARC-CE scripts.

3.1 User perspective

From the user perspective, there are two major points to emphasize. First, HyperQueue is a small binary with no dependencies which one can just download and run. Second, Hyper-

Queue works as a batch system under my control within a batch system outside of my control. That means, for example, if the batch system schedules only whole node jobs, one can split it (on Karolina, one 128 core job into four 32 core jobs). Or if the batch system enforces re-runnable jobs but one can set them to be not re-runnable.

Figure 3. Structure inside of a batch job submitted by HyperQueue (on Karolina)

3.2 Filling efficiency

The primary motivation for usage of HyperQueue was CPU efficiency on Karolina. On its 128-core worker nodes, ATLAS jobs were very short and therefore the CPU efficiency was not very good. Using the HyperQueue, it is possible to split one 128-cores job into four 32-cores jobs which increased the CPU efficiency by tens of percents (as shown on Figure 4). To ensure that the increase in the efficiency is not negated by partially empty worker nodes, the filling efficiency was measured. Start and end dates of jobs and workers were logged for 142 HyperQueue workers. Then their efficiency was was calculated as

$$\text{Eff} = \frac{\text{sum of all time used by jobs}}{\text{sum of all time available in workers}} \qquad (1)$$

Illustrations of how the worker nodes were filled are on Figure 5. Most of them were filled quite nicely (the left plot). But few of them were behaving oddly (center and right plot). The possible causes of late start could be lack of jobs. For the early end, it could be a killed job (if something was stuck and keep running until max time). The overall filling efficiency (even with the anomalies) was 93%.

Figure 4. CPU efficiency of ATLAS jobs without and with HyperQueue

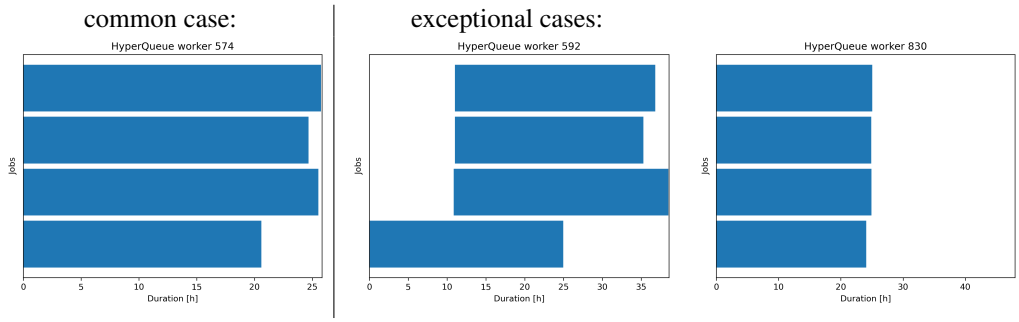

Figure 5. Duration of HyperQueue jobs within HyperQueue workers

4 atlas-cvmfsexec

ATLAS jobs require software to run. It is usually located in CVMFS [9]. But if the CVMFS is not installed, jobs can run in so-called fat containers (FC), which contain everything the job needs, or get the software through atlas-cvmfsexec [7]. This is a wrapper to install, mount, and unmount cvmfsexec [10] in the ATLAS context. The content of the CVMFS comes from a squid cache which needs to be accessible from the worker node.

Atlas-cvmfsexec is highly modular, i.e. there are parts of CVMFS (large or often used) which can be used from a local (rsynced) copy. To be specific, base system (CC7, alma9) container, fat container (FC) releases, condition ROOT files, and DBRelease can be all local. In that case, only ALRB [11] content (for every job) and release software (for non-FC releases) are obtained from remote. Here, I want to express my gratitude to Asoka De Silva for making the atlas-cvmfsexec possible and available.

Figure 6. The atlas-cvmfsexec usage. The upper plot shows number of completed jobs in 1 hour bins. The lower plot shows network traffic at the same time period.

4.1 Network usage

As shown on Figure 6, even at hundreds of concurrent atlas-cvmfsexec jobs, the network usage is rather low.

4.2 FC vs non-FC releases usage (9.2023 - 8.2024)

Atlas-cvmfsexec is used for about a year. As shown on Figure 7, during that time, 88% of jobs on Barbora and 63% of jobs on Karolina came through the atlas-cvmfsexec. This means the atlas-cvmfsexec significantly broadened the pool of available jobs and therefore improved the utilization.

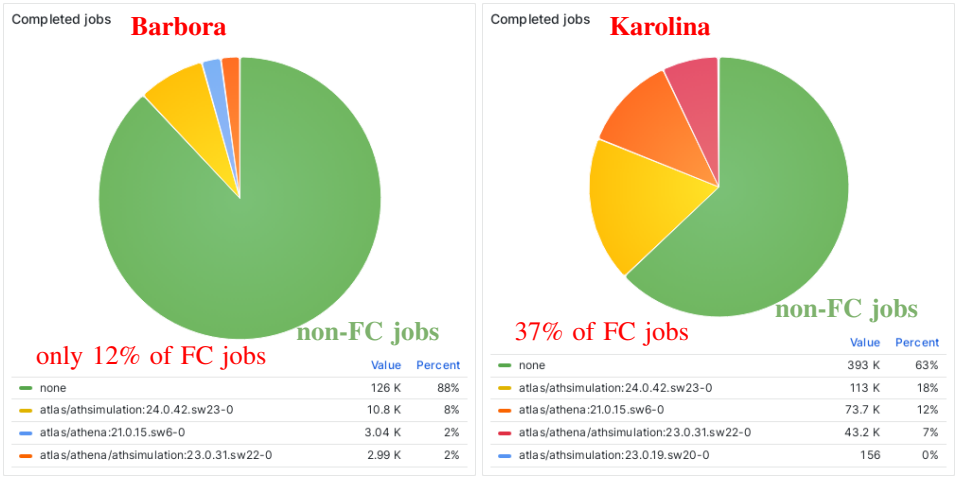

Figure 7. Usage of fat containers during the last year. "None" denotes the jobs using atlas-cvmfsexec instead.

5 Performance

In the last two years, HPCs contributed more than a third of resources available to ATLAS at praguelcg2 (left plot on Figure 8 and left plot on Figure 9).

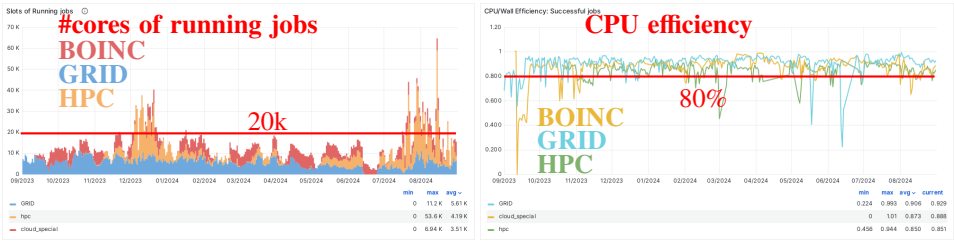

Figure 8. Performance in the last two years. The left plot shows number of cores in each of resource category. The right plot shows their CPU efficiency.

These HPCs are functioning well as shown at the failure rate plot (right plot on Figure 9) and their CPU efficiency is comparable to other resources (right plot on Figure 8).

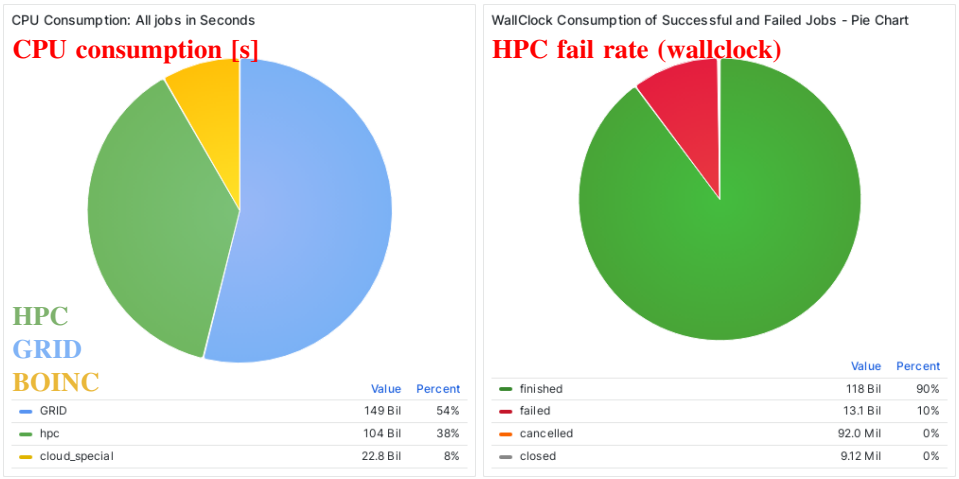

Figure 9. Performance in the last two years. The left plot shows CPU consumption (in seconds) of resources available at praguelcg2. The right plot shows failure rate of the HPC part.

6 Summary and Conclusion

The HyperQueue helped to significantly improve the CPU efficiency of ATLAS jobs running on large worker nodes at the cost of small decrease of used time caused by the worse filling efficiency. The atlas-cvmfsexec significantly broadened the pool of jobs available to the HPC and therefore helped increase their utilization. Thanks to these, the HPCs contributed significantly to amount of jobs delivered to the ATLAS Experiment.

Acknowledgement

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254). The work was also supported by the projects CERN-CZ LM2023040 and FORTE 02.01.01/00/22_008/0004632.

References

[1] ATLAS Collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, JINST **3**, S08003 (2008). 10.1088/1748-0221/3/08/S08003

[2] M. Svatoš, J. Chudoba, P. Vokáč (ATLAS), Updates on usage of the Czech national HPC center, EPJ Web Conf. **251**, 02008 (2021). 10.1051/epjconf/202125102008

[3] J. Chudoba, M. Adam, D. Adamová, T. Kouba, A. Mikula, V. Říkal, J. Švec, J. Uhlířová, P. Vokáč, M. Svatoš, A multipurpose computing center with distributed resources, J. Phys. Conf. Ser. **898**, 082034 (2017). 10.1088/1742-6596/898/8/082034

[4] D. Cameron, A. Filipčič, W. Guan, V. Tsulaia, R. Walker, T. Wenaus (ATLAS), Exploiting opportunistic resources for ATLAS with ARC CE and the Event Service, J. Phys. Conf. Ser. **898**, 052010 (2017). 10.1088/1742-6596/898/5/052010

[5] Harvester, https://github.com/HSF/harvester/, [accessed 2024-12-30]

[6] F.H. Barreiro Megino, K. De, A. Klimentov, T. Maeno, P. Nilsson, D. Oleynik, S. Padolski, S. Panitkin, T. Wenaus (ATLAS), PanDA for ATLAS distributed computing in the next decade, J. Phys. Conf. Ser. **898**, 052002 (2017). 10.1088/1742-6596/898/5/052002

[7] https://gitlab.cern.ch/atlas-tier3sw/atlas-cvmfsexec

[8] J. Beránek, A. Böhm, G. Palermo, J. Martinovič, B. Jansík, Hyperqueue: Efficient and ergonomic task graphs on hpc clusters, SoftwareX **27**, 101814 (2024). https://doi.org/10.1016/j.softx.2024.101814

[9] A. De Salvo, A. De Silva, D. Benjamin, J. Blomer, P. Buncic, A. Harutyunyan, A. Undrus, Y. Yao (ATLAS), Software installation and condition data distribution via CernVM file system in ATLAS, J. Phys. Conf. Ser. **396**, 032030 (2012). 10.1088/1742-6596/396/3/032030

[10] J. Blomer, D. Dykstra, G. Ganis, S. Mosciatti, J. Priessnitz, A fully unprivileged CernVM-FS, EPJ Web Conf. **245**, 07012 (2020). 10.1051/epjconf/202024507012

[11] https://twiki.atlas-canada.ca/bin/view/AtlasCanada/ATLASLocalRootBase2