COMPASS PRODUCTION SYSTEM: FRONTERA EXPERIENCE

A.Sh. Petrosyan[1,2]

[1] *Joint Institute for Nuclear Research, 6 Joliot-Curie st., 141980, Dubna, Russia*

[2] *Plekhanov Russian University of Economics, 36 Stremyanny per., 117997, Moscow, Russia*

E-mail: artem.petrosyan@jinr.ru

Since 2019, the COMPASS experiment has been running on the Frontera high-performance computer. This is a large machine (number 5 in the ranking of the most powerful supercomputers in 2019). Details of software setup and approaches to organizing data processing on this machine are presented in the article.

Keywords: COMPASS, PanDA, distributed computing, workflow management system, grid, high performance computing

Artem Petrosyan

1. Introduction

The COMPASS experiment [1] regularly receives the ability to process data on high-performance systems: in 2016-2018, the experiment worked on Blue Waters, from 2019 to the present, Frontera at the Texas supercomputer center has been used for data processing [2]. Frontera is a big machine, number 5 in the ranking of the most powerful supercomputers in the world in 2019 [3].

Frontera has two computing subsystems: a primary computing system focused on double-precision performance and a secondary subsystem focused on single-precision streaming memory computing. Frontera also has several storage systems, interfaces to the cloud and archive systems, and a set of nodes for hosting virtual servers. The primary computing system is Dell EMC on Intel Xeon Platinum 8280 (“Cascade Lake”) processors, 56 CPUs per node, 192GB of RAM per node, linked by the Mellanox Infiniband HDR and HDR-100 interconnect. The system consists of 8,008 compute nodes. Slurm [4] is a resource management system. Frontera’s peak performance amounts to 23.5 PFLOPS.

Data processing on high-performance computing systems (High Performance Computer, HPC) has several major differences compared to working on ordinary clusters, which are traditionally used in high-energy physics: these are conceptually different systems, but the relative similarity of the organization of computational nodes and system software makes it possible to run the applied software of experiments on supercomputers without changing the source code. The differences are architecture-driven and must be taken into account when organizing processing. They can be presented as the following list:

- each supercomputer is a machine built in a single copy and has unique characteristics inherent only to it;
- two-factor authentication is a common type of authentication on an HPC, which reduces the availability of a resource for automatic data processing systems – it is not always possible to access a computing resource from the outside, as is usually done when working with the nodes of a grid environment – it is necessary to place services, responsible for communication with the workload management system, on login or service nodes on an HPC;
- a shared file system requires delicate I/O management so as not to overload the file system and interfere with the work of other users;
- cluster management system: each HPC has its installation of one of the versions of a specially adapted scheduler and resource manager, for example, PBS, PBS Pro or Slurm;
- specialized system software;
- inability to use grid middleware such as CVMFS clients, VOMS, data management tools;
- lack of access to the Internet at compute nodes;
- user and project policy usually requires strict use of a computing resource for a limited period of time;
- usually the time for storing data on local storage is limited and not available throughout the project.

From the above, it follows that to organize data processing on such kinds of computing systems, it is necessary to carry out a deep modernization of all components of the computing infrastructure of the experiment.

2. Software setup

The PanDA workload management system manages the full cycle of data reconstruction jobs on Frontera, including both processing and merging intermediate results into the final ones [fig. 1].

Unlike the infrastructure implemented for Blue Waters, Frontera uses a new task management service named Harvester [5, 6, 7].

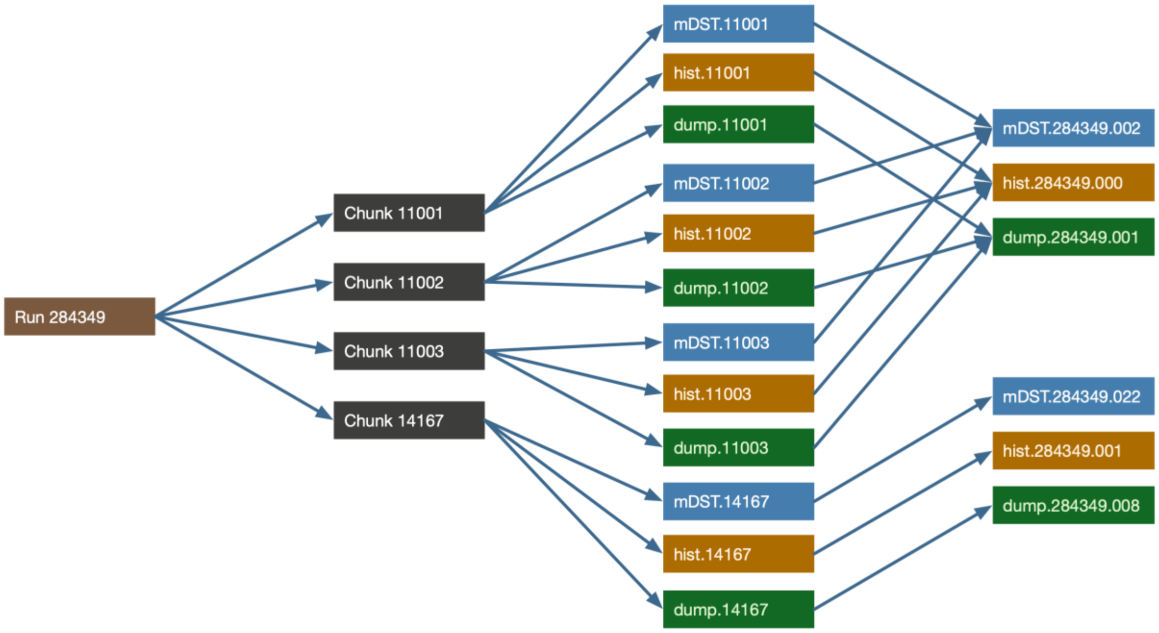


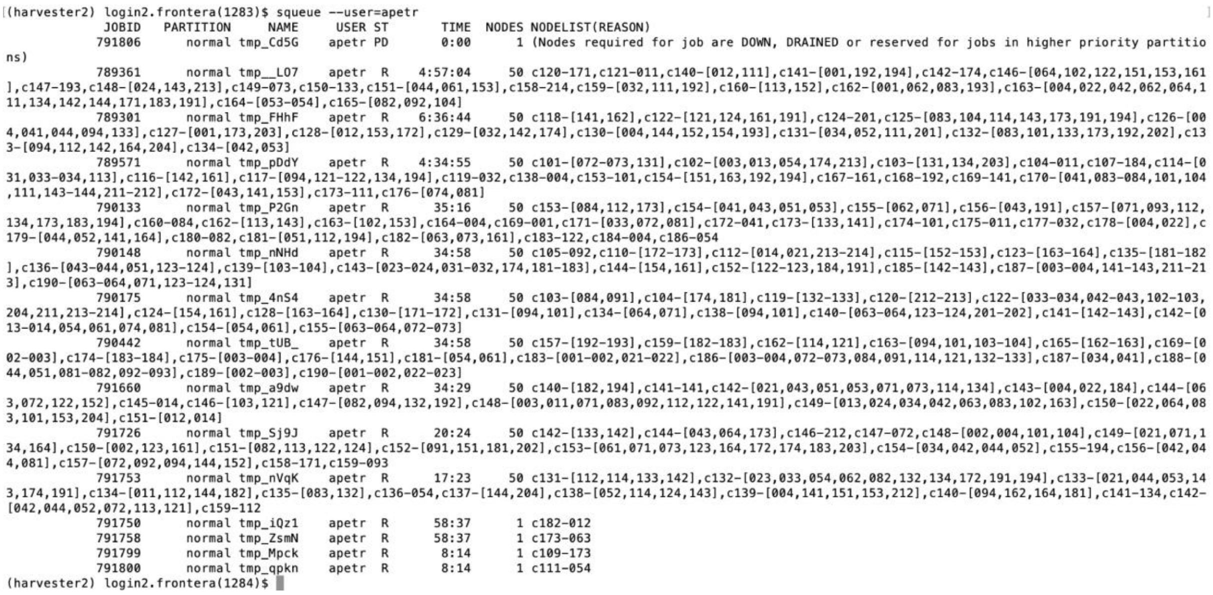
Figure 1. COMPASS reconstruction workflow

Harvester is a component of the PanDA ecosystem responsible for providing communication between WMS and pilots working on compute nodes. It can be located both on a dedicated node in the case of working with computing resources on the grid, and on a service node of high-performance computing systems. Harvester is a multithreaded service that uses an internal database to store all the necessary information. It can manage not one submission on a high-performance system, but a set, including various sizes and execution times. On supercomputers, using Harvester, it is possible to organize the processing of a multitude of large jobs. With the advent of this service, the issue of organizing processing on HPC systems has been completely resolved: for example, the ATLAS collaboration has fully integrated all available high-performance systems from the USA and Europe [8].

Several modules were added to the code of Harvester to control the processing of COMPASS data. Everything needed for the COMPASS job code, such as local MySQL database execution, payload management, errors handling and stage-out, was transferred from the Multi-Job Pilot and added as Harvester plug-ins. Since Frontera is one of the most powerful computing systems in the world, much attention is paid to optimizing I/O operations – when performing a large number of jobs there is a risk of overloading the metadata server of the shared file system. For this reason, all the software needed to complete the job is moved to a temporary folder on the compute node. The database server used by the tasks, which stores the calibration information, is also copied there. All data transfer operations to and from the node are monitored to avoid overloading the file system.

3. Submission tuning

The software infrastructure on Frontera includes 4 Harvester services, each of which is configured to perform 5 supercomputer tasks with 50 compute nodes each. In terms of grid, this is 56,000 individual jobs [fig. 2]. To handle such load, it was necessary to deploy an additional PanDA server, reserved only for working with Frontera. It was also required to increase the capacity of the database server storing information about jobs. The size of the jobs is due to the fact that on Frontera larger jobs are given priority over smaller jobs. The size of the jobs is selected in such a way that, on the one hand, it is fast enough to get access to resources, and on the other hand, so as not to overload the IT infrastructure of the experiment that controls data processing.



```
(harvester2) login2.frontera(1283)$ squeue --user=apetr
             JOBID   PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            791806      normal tmp_Cd5G    apetr PD       0:00      1 (Nodes required for job are DOWN, DRAINED or reserved for jobs in higher priority partitio
ns)
            789361      normal tmp__L07    apetr  R    4:57:04     50 c120-171,c121-011,c140-[012,111],c141-[001,192,194],c142-174,c146-[064,102,122,151,153,161
],c147-193,c148-[024,143,213],c149-073,c150-133,c151-[044,061,153],c158-214,c159-[032,111,192],c160-[113,152],c162-[001,062,083,193],c163-[004,022,042,062,064,1
11,134,142,144,171,183,191],c164-[053-054],c165-[082,092,104]
            789301      normal tmp_FHhF    apetr  R    6:36:44     50 c118-[141,162],c122-[121,124,161,191],c124-201,c125-[083,104,114,143,173,191,194],c126-[00
4,041,044,094,133],c127-[001,173,203],c128-[012,153,172],c129-[032,142,174],c130-[004,144,152,154,193],c131-[034,052,111,201],c132-[083,101,133,173,192,202],c13
3-[094,112,142,164,204],c134-[042,053]
            789571      normal tmp_pDdY    apetr  R    4:34:55     50 c101-[072-073,131],c102-[003,013,054,174,213],c103-[131,134,203],c104-011,c107-184,c114-[0
31,033-034,113],c116-[142,161],c117-[094,121-122,134,194],c119-032,c138-004,c153-101,c154-[151,163,192,194],c167-161,c168-192,c169-141,c170-[041,083-084,101,104
,111,143-144,211-212],c172-[043,141,153],c173-111,c176-[074,081]
            790133      normal tmp_P2Gn    apetr  R      35:16     50 c153-[084,112,173],c154-[041,043,051,053],c155-[062,071],c156-[043,191],c157-[071,093,112,
134,173,183,194],c160-084,c162-[113,143],c163-[102,153],c164-004,c169-001,c171-[033,072,081],c172-041,c173-[133,141],c174-101,c175-011,c177-032,c178-[004,022],c
179-[044,052,141,164],c180-082,c181-[051,112,194],c182-[063,073,161],c183-122,c184-004,c186-054
            790148      normal tmp_nNHd    apetr  R      34:58     50 c105-092,c110-[172-173],c112-[014,021,213-214],c115-[152-153],c123-[163-164],c135-[181-182
],c136-[043-044,051,123-124],c139-[103-104],c143-[023-024,031-032,174,181-183],c144-[154,161],c152-[122-123,184,191],c185-[142-143],c187-[003-004,141-143,211-21
3],c190-[063-064,071,123-124,131]
            790175      normal tmp_4nS4    apetr  R      34:58     50 c103-[084,091],c104-[174,181],c119-[132-133],c120-[212-213],c122-[033-034,042-043,102-103,
204,211,213-214],c124-[154,161],c128-[163-164],c130-[171-172],c131-[094,101],c134-[064,071],c138-[094,101],c140-[063-064,123-124,201-202],c141-[142-143],c142-[0
13-014,054,061,074,081],c154-[054,061],c155-[063-064,072-073]
            790442      normal tmp_tUB_    apetr  R      34:58     50 c157-[192-193],c159-[182-183],c162-[114,121],c163-[094,101,103-104],c165-[162-163],c169-[0
02-003],c174-[183-184],c175-[003-004],c176-[144,151],c181-[054,061],c183-[001-002,021-022],c186-[003-004,072-073,084,091,114,121,132-133],c187-[034,041],c188-[0
44,051,081-082,092-093],c189-[002-003],c190-[001-002,022-023]
            791660      normal tmp_a9dw    apetr  R      34:29     50 c140-[182,194],c141-141,c142-[021,043,051,053,071,073,114,134],c143-[004,022,184],c144-[06
3,072,122,152],c145-014,c146-[103,121],c147-[082,094,132,192],c148-[003,011,071,083,092,112,122,141,191],c149-[013,024,034,042,063,083,102,163],c150-[022,064,08
3,101,153,204],c151-[012,014]
            791726      normal tmp_Sj9J    apetr  R      20:24     50 c142-[133,142],c144-[043,064,173],c146-212,c147-072,c148-[002,004,101,104],c149-[021,071,1
34,164],c150-[002,123,161],c151-[082,113,122,124],c152-[091,151,181,202],c153-[061,071,073,123,164,172,174,183,203],c154-[034,042,044,052],c155-194,c156-[042,04
4,081],c157-[072,092,094,144,152],c158-171,c159-093
            791753      normal tmp_nVqK    apetr  R      17:23     50 c131-[112,114,133,142],c132-[023,033,054,062,082,132,134,172,191,194],c133-[021,044,053,14
3,174,191],c134-[011,112,144,182],c135-[083,132],c136-054,c137-[144,204],c138-[052,114,124,143],c139-[004,141,151,153,212],c140-[094,162,164,181],c141-134,c142-
[042,044,052,072,113,121],c159-112
            791750      normal tmp_iQz1    apetr  R      58:37      1 c182-012
            791758      normal tmp_ZsmN    apetr  R      58:37      1 c173-063
            791799      normal tmp_Mpck    apetr  R       8:14      1 c109-173
            791800      normal tmp_qpkn    apetr  R       8:14      1 c111-054
(harvester2) login2.frontera(1284)$ 
```

Figure 2. COMPASS jobs of Frontera

Since the entire chain of data reconstruction is performed on Frontera, including reconstruction jobs, as well as merging jobs of each result type: mDST, histograms, and files of selected events, Harvester was tuned for optimal performance. The fact is that reconstruction jobs are usually completed within 7-8 hours, while merging jobs are completed within an hour. To organize such work, two queues were registered in PanDA: for reconstruction jobs and for merging jobs. In Harvester, each queue has its own settings: the queue for reconstruction jobs requests 50 nodes for 10 hours, and for merging 1 node for 2 hours. Exceeding the requested time is done to prepare the input data, to have time to stage-out the output and ensure the reliable execution of jobs in the event of possible slowdowns in the work of the shared file system.

4. PanDA server tuning

Usually, when working on large computing systems, priority is given to larger jobs. In the case of Frontera, jobs of 50 nodes are considered as small and can spend up to a week waiting in the queue before reaching compute nodes. Such behavior is inconsistent with the concept of high-throughput computing, in which all components of the distributed computing infrastructure, such as PanDA, were developed. In the grid, strong connectivity with advanced monitoring tools is practiced: each job reports its status every 30 minutes. This is done to minimize the wait time of computing resources and ensure a stable number of jobs performed. When waiting in a queue for a week, such high connectivity becomes not only unnecessary, it becomes a generator of excess load on all components of the computing infrastructure.

The PanDA server is restarted once a day to reset the logger handlers and download information about the certificate revocation list. However, even the fastest server reboot under high load results in the loss of monitoring packets directed by the Harvester services to the PanDA server. The omission of the message about the status of the job leads to the recognition of the jobs as lost: the PanDA server sends a message to Harvester to remove such a job from the queue. In a grid environment, each job is processed individually, and the loss of several jobs does not lead to noticeable consequences. In the case of high-performance systems, deleting one job results in the deletion of the entire submission of 2,800 jobs, which is completely unacceptable.

To eliminate this behavior, the PanDA server was reconfigured. The logging level was set to CRITICAL, the fetch-crl service was disabled, and server reboot was disabled. A comparative profile of I/O operations before and after the reconfiguration is shown in Figure 3. Reconfiguring the PanDA server resolved the missed packets issue. Now the PanDA server can work for several months without restarting.

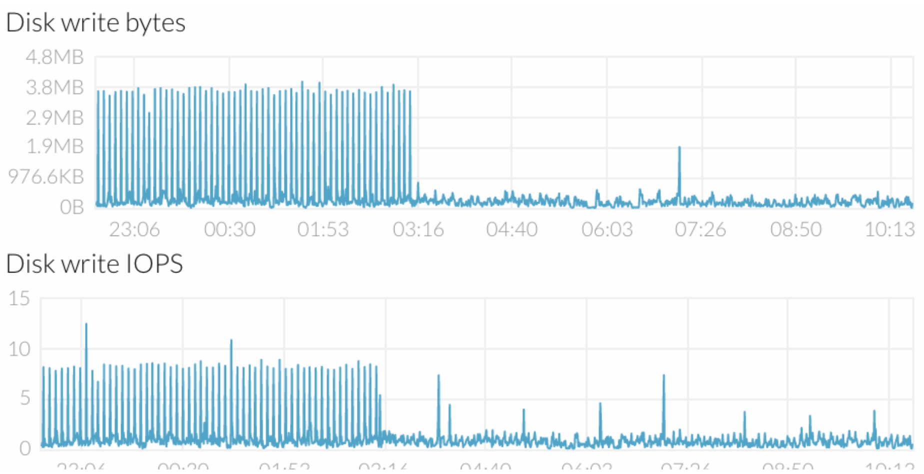


Figure 3. I/O operations profile on the PanDA server

5. Summary

Thus, the use of high-performance systems places increased demands on all components of the data processing infrastructure. This is due to the abrupt load profile characteristic of computing on high-performance systems: in the case of working on a grid farm, jobs are delivered to computational nodes and launched one at a time, while in the case of working with a supercomputer, several thousand jobs are grouped into one submission and, after waiting in the queue, start almost simultaneously. In addition, submissions for a larger number of nodes on such systems receive a higher priority in the queue, so it is necessary to keep as many large submissions in the queue as possible. A new implementation of the pilot application was developed to work on Frontera. A diagram of the data processing organization on the Frontera HPC is shown in Figure 4.

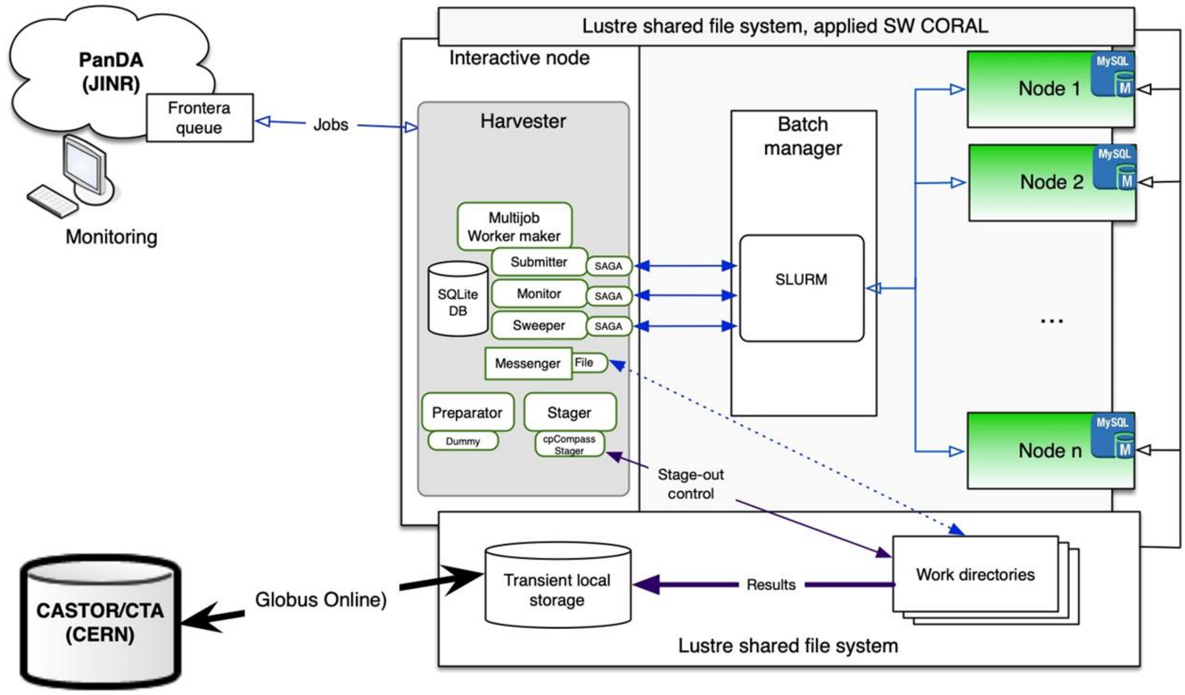


Figure 4. Data processing on Frontera

Submissions for 50 computational nodes, each of which consists of 2,800 individual jobs, were used for processing. The supercomputer was utilized to carry out the entire chain of data reconstruction tasks processing, including the aggregation of results. The management of tasks and jobs on this resource was fully integrated into the production system of the COMPASS experiment and was performed in the same way as data processing management on grid clusters [9, 10].

6. Acknowledgements

The study was supported by the Russian Science Foundation grant (project No. 19-71-30008).

References

[1] P. Abbon et al., The COMPASS experiment at CERN, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Vol. 577, pp. 455-518, 2007

[2] Texas Advanced Computing Centre, the University of Texas at Austin, available at https://www.tacc.utexas.edu/ (accessed 11.08.2021)

[3] The top 500 list, available at https://www.top500.org/lists/top500/2019/06/ (accessed 9.09.2021)

[4] Slurm workload manager, available at https://slurm.schedmd.com/ (accessed 9.09.2021)

[5] A. Petrosyan, COMPASS Production System: Processing on HPC, CEUR Workshop Proceedings, Vol. 2267, pp. 139-144, 2018

[6] Harvester, available at https://github.com/HSF/harvester/wiki (accessed 10.09.2021)

[7] F.H. Barreiro Megino et al., PanDA for ATLAS distributed computing in the next decade, Journal of Physics Conference Series, Vol. 898, 2017

[8] T. Maeno, Harvester: an edge service harvesting heterogeneous resources for ATLAS, EPJ Web of Conferences, Vol. 214, 2019

[9] A. Petrosyan, COMPASS Production System Overview, EPJ Web Conf., Vol. 214, 2019

[10] A. Petrosyan, D. Malevanniy, Distributed data processing of the COMPASS experiment, CEUR Workshop Proceedings, Vol. 2507, 2019