Implementation of a PROOF Farm for ATLAS Tier 3 Analyses

C. Osuna\textsuperscript{1}, E. Accion\textsuperscript{2,3}, G. Bernabeu\textsuperscript{2,3}, A. Bria\textsuperscript{2,4}, G. Merino\textsuperscript{2,3}

\textsuperscript{1} Institut de Física d’Altes Energies, IFAE, Edifici Cn, Universitat Autònoma de Barcelona, ES-08193 Bellaterra (Barcelona), Spain
\textsuperscript{2} Port d’Informació Científica (PIC), Universitat Autònoma de Barcelona, Edifici D, ES-08193 Bellaterra (Barcelona), Spain
\textsuperscript{3} Also at Centro de Investigaciones Energéticas Medioambientales y Tecnológicas (CIEMAT), Madrid, Spain
\textsuperscript{4} Also at Institut de Física d’Altes Energies, IFAE, Edifici Cn, Universitat Autònoma de Barcelona, ES-08193 Bellaterra (Barcelona), Spain

E-mail: osuna@ifae.es

\textbf{Abstract.} The Parallel ROOT facility (PROOF) \cite{3} is a distributed analysis system that enables to parallelize analysis jobs on a cluster of computers. The PROOF system dynamically identifies free resources on the worker nodes of the farm and it simplifies the job preparation for the user.

The experience with a PROOF farm on an ATLAS Tier 3, integrated with a Xrootd distributed storage system is presented in this document.

The performance of the parallel facility is evaluated for several ATLAS analyses with different requirements on I/O and CPU processing and on different environments. In particular the correlation of the parallel processing performance with the storage system is discussed for different scenarios, like a dCache storage on a distributed set of disk servers or a Xrootd system distributed among the local disks of the PROOF workers nodes.

1. Introduction

The Atlas Experiment \cite{1} at the Large Hadron Collider (LHC) is taking data since November 2009. Since then it has been increasing the integrated luminosity collected by the detector reaching about 45 pb\textsuperscript{-1} of proton-proton collision data at the end of the run in 2010. For the 2011 run the ATLAS detector is expected to reach few fb\textsuperscript{-1} of proton-proton data. The continuously increasing number of events recorded that users need to analyze represents a very demanding challenge in terms of both computing resources and applications. In order to reduce the computing time spent in the last steps of the analysis chain performed in the local resources of the institute it is important to parallelize the applications among the cpu nodes of the facility.

The Atlas Computing model \cite{2} is based on the grid, that is a hierarchical organization of computing facilities in different levels with computing resources world wide distributed. At the top level, the Tier 0 at CERN receives the Raw data of the detector and performs an initial event reconstruction and data calibration. There are 10 Tier 1 sites world wide distributed that are in charge of further reconstruction jobs that produce datasets with a reconstructed event format, Event Summary Data (ESD). The Analysis Object Data (AOD), produced as an outcome of further reconstruction jobs on ESD, contains physics objects and event format...
suitable to perform physics analyses. Users or physics groups can run the analysis on AODs on the grid making use of the Tier 2 facilities for this purpose. The result of this first step of the analysis are datasets in the format of Derived Physics Data (DPD) which can finally be retrieved from the grid and placed on local resources to perform the last step of the analysis that typically yields plots and physics results. This last step is usually carried out on local computing resources of the institute, the Tier 3 facilities, where the latency to get the job finished and the job preparation are considerably reduced compared to other levels of the grid hierarchy.

Even after the successive filters applied on previous steps of the analysis performed on the grid, a typical Tier 3 analysis job has to process millions of events. In order to reduce the computing time to get the physics results, a parallel facility that is able to split the full statistics among all the cpu cores available in the farm of the local resources is an interesting solution to explore. A Tier 3 facility typically consists of a cluster computing system and therefore, the parallel computing system must be able to work on a scalable cluster system (discarding other kinds of shared-memory multiprocessor system).

The Parallel ROOT Facility (PROOF [3]) allows parallel computing among nodes in a cluster system which do not share memory. It exploits parallelism at event level inherent at high energy physics data. PROOF is part of the ROOT [4] framework, and it is tightly coupled with Xrootd [5] protocol and distributed storage system.

The PROOF facility at IFAE/PIC consists of a cluster of nodes installed with a PROOF service to handle the parallelization of the event analysis and an additional Xrootd service to handle the data distributed among the local disks of the different PROOF workers nodes. This configuration favors access to local data instead of remote access to disk servers. The layout and configuration of the farm and services is presented in Section 2. Performance results are shown in Section 3. And finally monitoring tools developed at IFAE/PIC to display PROOF monitoring information are presented in Section 4.

2. PROOF and Xrootd configuration

The farm is composed by 7 blade servers with 2 processors Six Core Intel Xeon X56500, 24 GB of RAM, and 2 disks SATA 5400 rpm of 500 GB each one. Every server provides cpu resources equivalent to 166.42 HS06. One server is dedicated to act as the PROOF master and Xrootd redirector, while the other 6 servers are PROOF workers and Xrootd data servers.

A layout of the PROOF farm is shown in Fig. 1.

The PROOF master work as a job scheduler: it parallelizes data by distributing dynamically (slower workers get smaller work packets than faster ones) the whole set of events among the different workers. If the input files are located in the local disk of the worker nodes, the master tries to send work packets as close as possible to the node where data is.

The parallel application can read input data from an external disk, as illustrated in Fig. 1, managed by a dCache storage system [6] or from data files located in the local disks of the PROOF workers. For this purpose, the local disks of the PROOF workers are configured as a distributed storage system (through Xrootd) that acts as a cache of the dcache storage, keeping replicas of the files used as input data in the parallel facility. To ease the administration of files in the local cache and to automatically allocate replicas of new requested files, the Xrootd [5] service was installed in the same PROOF nodes: the PROOF master is as well a Xrootd redirector while PROOF workers were installed with a Xrootd data server service.

The Xrootd storage system is designed to provide POSIX-like access to files and their enclosing directory namespace. Being developed for the ROOT analysis framework, it provides a native protocol optimized for low latency access to ROOT files (input file format of the parallel facility) of the storage cluster system.

Figure 1 illustrates how the Xrootd storage system (integrated in the PROOF farm) works to automatically allocate files in the storage system of the cluster. When the client sends a
Figure 1. Schema of the parallel facility configuration. Clients can open PROOF sessions on the master node which allocates slave processes on every worker node (with a limit of one process per core). Work packets are typically scheduled by PROOF into a node that has a replica of the requested input file. A Xrootd service is in charge of automatically look up remote resident files on a external dcache server and distribute replicas among the worker nodes. Finally a MySQL service in the PROOF master records events happening in the facility for monitoring purposes. This database is later queried by a web server to format and display the monitoring information.

PROOF job requesting a set of data files to process:

- Xrootd checks if data is present on at least one PROOF worker node.
- If not, Xrootd looks for the files in the external disk server (dcache) and it replicates them to the local disks of the worker nodes taking into account disk occupancy, data server load, etc.
- The PROOF master identifies which worker allocates each file and send packets of events to be processed to the corresponding node, asking to read data files from local disks. A maximum of 12 processes (1 process per core) per worker node are allocated.
- From the first run that triggers the replica to local disks by Xrootd on, access to data is local and files are usually found on the disk cache of the worker (unless they are too old, and the Xrootd purge service has been activated to free up some space to allocate new requests).
- Purge of old files when there is no left space in the local disk is automatically managed by Xrootd.

The integration of the Xrootd service within the PROOF system provides a much faster (see Section 3) and robust (for it does not rely on external services like disk servers, network, etc) access to data. In addition, the whole system is completely transparent to the user. The user
application does not need to know where to find the input files; it is instead the Xrootd service the responsible to find the user data and to trigger the replication to the local disk storage system if required.

3. Performance

Different test measurements have been performed on the PROOF farm to compute event rates that can be achieved with the infrastructure and layout described in Section 2 and to compare the different systems to access input data of the parallel facility.

The reference analysis used for timing measurements reads ROOT files with a flat TTree [4] structure containing 1800 integer branches plus 4651 float branches, out of which only 104 float branches are read by the application. The analysis code performs 10K hyperbolic tangent instructions per event and reads an average of 423,6 bytes per event.

Results comparing access modes to input files from the external dcache server and from the local disks of the PROOF workers are illustrated in Fig. 2.

The best performance (orange circles), according to event rate observable, is achieved when accessing files from local disks and data is already buffered in the disk cache on memory of the nodes (allocated from a previous run). The event rate scales with the number of workers almost linearly up to about 60-70 workers. This is the case that happens when most of the applications are sharing input data and are continuously reading the same data set in consecutive runs. Unless the application itself is very demanding on memory resources or the amount of input data is so large that it can not be cached into the RAM of the node, input data is usually buffered during the first run of the application.

If the data are not present in the disk cache of the nodes, the applications in the workers have to access the local SATA disk to read the input data. The event rate, as illustrated in Fig. 2 (green squares), deteriorates compared to the previous situation, and saturates much earlier. The reason is that every worker node can allocate up to 12 PROOF processes scheduled by the PROOF master, while the node has only 2 hard drives. All these processes running on the same node request data blocks to the same couple of hard drives in a non-collaborative and non-sequential manner for events scheduled to different PROOF processes are typically not

Figure 2. Event rate (and bytes per second on the right vertical axis) achieved by different data access modes as a function of the number of workers defined in the parallel application.
contiguous.

Finally, the purple triangles show the performance when reading data directly through the dcap [6] protocol from the external dcache server. In this case the large number of disks available in the pools of this external server does surpass the limitation on number of workers per hard drive that saturates the event rate at large number of PROOF workers. The smaller performance compared to the local access (using the cache resident on memory) is due to remote access of data (network, dcache storage system latencies, etc.) penalties.

4. PROOF Monitoring

Monitoring of PROOF jobs is of interest mainly for two purposes:

- To understand the performance of individual PROOF applications, discover bottlenecks like I/O and extract invaluable performance information that allow us to optimize the application and highlight problematic PROOF applications that could potentially create instabilities in the service due to vast consumption of resources (like user applications that request too much memory allocation or save very large log files on disk).
- To provide accounting information, essential to understand the level of usage of the resources in the parallel facility with time.

PROOF comes with a simple solution to record any event that happens in the parallel facility on a MySQL database [7]. For simplicity, the database is hosted on the PROOF master. An additional tool is needed to aggregate the information recorded in the database and display it in the form of plots and tables for easy visualization. At IFAE/PIC, we developed a software package (ProofMon) based on php and matplotlib that shows, in different views, a set of plots for monitoring and accounting. Although the default time window for the monitoring view is 24 hours and 3 months for the accounting view, both of them offer a custom query for every plot. ProofMon performs a single query to the MySQL database and generates data processed later by matplotlib to produce a set of plots corresponding to a given view. ProofMon is designed in a modular way that makes easy to integrate new views and plots in the case PROOF adds new fields in the monitoring database in future releases, like memory usage of jobs.

Figures 3 to 6 illustrate some examples of the plots displayed by the monitoring and accounting view of the ProofMon package.

5. Conclusions

The IFAE/PIC PROOF facility is successfully running 7 servers, each with 166.42 HS06 of cpu resources. Improvements in the performance and robustness were the main reasons to install and configure and additional Xrootd service to use the local disks of the PROOF workers as a cluster storage system, fully managed by Xrootd and transparent to users.

This solution has been proved to provide the best event rate for most of the applications, where data can be buffered in the memory of the node. The PROOF monitoring functionality has been very useful to check the health and performance of user applications and follow occupancy of resources. The ProofMon package developed at IFAE/PIC provides a quick and fast access via web server to main monitoring and accounting observables.

Acknowledgements

This work was supported in part by grants CPAN09-TS15 from the Centro Nacional de Física de Partículas, Astropartículas y Nuclear, and FPA2009-07496 and FPA2007-66152-C02-00 from the Ministerio de Educación y Ciencia, Spain. The Port d’Informació Científica (PIC) is maintained through a collaboration between the Generalitat de Catalunya, CIEMAT, IFAE and the Universitat Autònoma de Barcelona.
Figure 3. PROOF sessions per user versus time. Width of rectangles represents the walltime of the PROOF job. Colours are random for better visualization.

Figure 4. Global event rate (average in time and among all the workers) for each PROOF job, categorized by user versus time.

Figure 5. Instantaneous parallel facility occupancy, defined as the ratio of the number of workers processing data to the total number of workers available in the system. Values above 100% show and occupancy of more than one process per core due to simultaneous running sessions from different users.

Figure 6. Occupancy of PROOF farm per week, defined as the ratio of cpu consumption to the effective total cpu available. Considered an interactive service, the effective total cpu is computed as the cpu capacity provided by the farm in a week taking into account 8 running hours per day and 5 days per week.

References
[1] Armstrong W W et al. (ATLAS collaboration) 1994 Atlas: Technical proposal for a general-purpose p p experiment at the Large Hadron Collider at CERN CERN-LHCC-94-43.
[2] Adams D, Barberis D, Bee C, Hawkings R, Jarp S, Jones1 R, Malon D, Foggilli L, Poulard G, Quarrie D and Wenaus T 2005 The atlas computing model Tech. rep. CERN CERN-LHCC-2004-037/G-085
[3] M. Ballintijn, et al, Nucl.Instrum.Meth. A559, 13 (2006)
[4] Brun R, Rademakers F, Canal P and Goto M 2003 (Preprint cs/0306078)
[5] Dorigo A, Elmer P, Furano F and Hanushevsky A 2005 TELE-INFO05: Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics pp 16 ISBN 960-8457-11-4
[6] M. de Riese, et al, The dCache Book. URL http://www.dcache.org/manuals/Book/ (as at December 2010)
[7] The PROOF monitoring system. URL http://root.cern.ch/drupal/content/enabling-query-monitoring (as at December 2010)