PROOF-Lite running experience in High Energy Physics data analysis

L Barbone\textsuperscript{a,b}, G Donvito\textsuperscript{a} and A Pompili\textsuperscript{a,b,1}

\textsuperscript{a} INFN Sezione di Bari, via Orabona 4, I-70125 Bari, Italy
\textsuperscript{b} Dipartimento Interateneo di Fisica, via Amendola 173, I-70126 Bari, Italy
E-mail: barbone@ba.infn.it, donvito@ba.infn.it, pompili@ba.infn.it

Abstract. Solid-State-Drives are currently introduced in storage facilities for High Energy Physics (HEP) experiments and their performances are measured and compared to standard magnetic disks. For this paper a typical HEP data analysis is performed and used as a test to measure computing performances. The tests exploit the features provided by PROOF-Lite which allows to distribute a huge number of events among different CPU cores, thus reducing the overall time needed to complete the analysis task. These tests are carried on few computational devices typically hosted at a current Tier-2/Tier-3 facility. The performance results are provided in terms of figures of merit and the main issue is scalability described in terms of speed up factor and processing event rate. The obtained results can be used as guideline for both the typical HEP analyst and the Tier-2/Tier-3 manager: the former in the configuration of his own analysis task while dealing with increasing data sizes, the latter in the implementation of an interactive data analysis facility for HEP experiments while facing solutions that concern both technological and economical aspects.

1. Introduction
In High Energy Physics (HEP) the frontier accelerator machines, like the Large Hadron Collider (LHC), produce events at an enormous rate. The relative experiments, like ATLAS and CMS, collect an unprecedented volume of data. Thus a physics analysis of this data requires large computational resources.

With reference to the larger collaborations such as ATLAS and CMS, data analysis is performed by a HEP analyst (hereafter called end-user) mainly in small/medium size computing facilities like Tier-2/Tier-3 clusters. Often the end-user deals with a variety of computational issues to optimize the use of the resources for his analysis and therefore some technical knowledge is needed to efficiently face them.

A typical data analysis procedure is going to be described, not pretending to be general since data handling methodologies may considerably differ in the HEP field. The end-user processes data in relation to a specific physics task obtaining a reduced data format. The data volume is further reduced by a preliminary event selection (user-skimming, reduced collections). This initial analysis step is typically developed in a specific experimental software framework (for instance CMSSW for CMS collaborators). This step is executed by means of an initial batch activity carried out on dedicated batch farms hosted at Tier-2’s (usually accessed through Grid

\textsuperscript{1} Poster presenter at the conference Computing in High Energy and Nuclear Physics 2012, New York, USA.
interfaces); typical batch activities include private simulations as well. All these activities are characterized by long execution times, little or null interactivity and rare optimization cycles.

The LHC experiments’ data sets are formatted using ROOT [1] as back-end. Moreover the end-user data collections produced in the initial batch activity are mainly ROOT files. The end-user ROOT files can be relatively large-sized and are the input for the second analysis step by which finer reduction algorithms are applied and repeated refinement cycles are required both for selection tuning and systematic studies. Therefore the execution time of a single cycle must be reasonable (within 1 hour); the corresponding level of interactivity can vary depending on the different solutions adopted at Tier-2/Tier-3 level, but typically the full interactivity and complete user control are desirable. Finally the third step of a typical analysis consists in a set of interactive tasks (histogramming, fitting, plotting etc.) with very short response times.

The second step is crucial also for the analysis time schedule because of the relevant number (often hundreds) of cycles needed. The latter is increasingly demanding being related to the growth of the integrated luminosity collected by the experiments and the corresponding growth of the overall data set size. The size of the user’s files and the related execution time of a single cycle could be maintained under control by increasing the initial event rate skimming. However, in general, it is not possible to harden the reduction algorithms beyond a certain extent. This problem can be tackled by turning to specific characteristics of ROOT. Indeed ROOT organizes the data in tree-like structures and provides a framework for tree analysis which can be easily parallelized: the main processing part of the ROOT task, where the algorithm is applied to each event-record in the tree, can be executed independently for each event since the events are independent.

The Parallel ROOT Facility - PROOF [2,3,4,5] which is an extension of the ROOT system, allows a quicker and transparent analysis of large sets of ROOT files in parallel on remote computer clusters typically hosted at Tier-2/Tier-3 facilities. PROOF allows HEP physicists to analyze much larger data sets on a shorter time scale. It makes use of the inherent parallelism in event data and implements an architecture that optimizes I/O and CPU utilization in heterogeneous clusters with distributed storage. PROOF system consists of a 3-tier architecture and its description can be found in [3,4].

PROOF-Lite [4,5] is a dedicated version of PROOF optimized for multi-core multi-user servers or workstations (even desktops and laptops). It conveniently exploits CPU resources of multi-core servers, thus allowing the ROOT files to be analyzed in enough short single cycle duration. This capability restores the wished possibility to a fully interactive approach to the analysis’s second phase, otherwise the latter approach would be compromised by the increasing growth of the users’ files. In a PROOF session the user connects to a master PROOF server on a remote cluster from his local client ROOT session. The PROOF server creates slave servers on all the cluster nodes to parallelize the task execution. PROOF-Lite, instead, implements a 2-tier architecture where the master is merged into the client and none configuration is required. It allows the client ROOT session directly hosted on the multi-core server to execute an interactive task with parallel execution and to control the parallel ROOT processes (“workers”) through a real-time feedback.

In this PROOF-Lite approach the ROOT input files are locally stored on a multi-core server. Despite of the always improving network performances and data access protocols, data locality is still an advantage with respect to data distribution over large storage systems. In fact, in these systems, each disk server must handle hundreds of concurrent accesses on a single disk partition. On the other hand, the degree of task parallelization is clearly limited by the number of available workers but this is not an effective limit for the end-user as far as a single analysis PROOF cycle is enough short (from few minutes till a few dozens of minutes).

Users’ ROOT macro may typically be I/O-demanding and consequently PROOF-Lite tasks become I/O-limited. Therefore it is worth to test the PROOF-Lite performances with Solid State
Drives (SSD) against standard Hard Disk Drives (HDD), being the former currently introduced in Tier-2/Tier-3 storage systems.

In this article PROOF-Lite performances are investigated through tests that aim to compare a multi-core multi-user server with respect to a multi-core single-user server (workstation), as well as SSDs with respect to HDDs (both SAS and SATA) with I/O-limited tasks. These technologies are currently available at Tier2/Tier-3 HEP facilities and their managers, together with users and their needs, often have to choose among them [6].

2. Description of the test setup

In this section a description of the overall setup of the PROOF-Lite performance tests is provided, covering both some hardware details and test methodology.

2.1. Hardware configuration

Two servers hosted at the Bari Tier-2 facility have been exploited for the tests:

1) A multi-core interactive server for Bari Tier-2 users denoted by MCIS in the following. The server is used by CMS physicists to analyze data interactively. The server is configured with two 6-cores CPUs with the HyperThreading technology enabled. This implies the possibility to exploit up to 24 concurrent processes at a given time. It provides 1 GB of memory for each core for a total of 24 GB of RAM memory. The storage sub-system is realized by means of a raid of 9 SATA disks, of 2 TB each, with a mechanics of 7200 rpm. All the disks are configured in a single RAID6 in order to provide good resilience to the accidental failures of disks.

2) A 8-core server, denoted by 8CMSU in the following, has been used as single-user workstation. This server is equipped with two 4-cores, 8 GB of total RAM installed and two disk raids. The first one is a RAID1 of two SAS drives; each of them is a 146 GB drive with a fast 10 Krpm mechanics and a form factor of 2.5 in. The second raid is a RAID0 of two MLC SSDs of about 256 GB each. The first raid is connected using a 3 Gbit/s SAS raid controller whereas the second one exploits a fast 6 Gbit/s SAS2 raid controller.

2.2. Input ROOT files

The overall size of the four input ROOT data files used in the tests amounts to 344 GB. They refer to an analysis aiming to reconstruct the decay mode $D^0 \rightarrow K\pi\pi\pi$, by extracting this signal from a huge combinatorial background of four tracks, in minimum bias events collected in 2010 by the CMS experiment. The ROOT files have been preliminarily produced by running a specific channel reconstruction algorithm within CMSSW_3_8_7_patch2 experiment software release. They have been written in split mode and are read, within the PROOF task, in split mode as well. The splitting is a way to optimize the file reading since the compression and reading algorithm are more efficient. The majority of the variables in the ROOT files is read during the intensive analysis task. The four input files, as well as their relative unique merge file, were locally stored on the servers during the tests. For the test involving size-limited SAS disk only one file (of 100 GB) out of the four has been used.

The ROOT version used in the tests is: 5.27/06b.

2.3. Performance figures of merit

If $T_n$ is the overall processing time of a PROOF-Lite job executed by $n$ workers, where $n = 1, 2, \ldots, 8$ in our tests, then the average processing rate is defined as the following function of $n$:

$$<R(n)> = \frac{\text{# processed events}}{T_n}$$

(1)
measured in number of events per second. The number of processed events is common to all the
tasks since the same ROOT files are used as input for the PROOF-Lite jobs.

The speed-up figure of merit is defined as the following adimensional ratio function of $n$:

$$S(n) = \frac{T_1}{T_n}$$

Thus $S_1 = 1$ by definition. The capability to linearly scale with the number of workers is formally
represented by $S_n = n$, namely $T_n = T_1/n$, independently of the value of $n$.

For each task typology and for each fixed number of workers the PROOF-Lite tasks have been
repeated several times, thus providing a sample of $T_n$ values for each single $n$ value. The sample
mean and the sample variance are calculated and, after standard error propagation, mean and
standard deviation for $<R(n)>$ and $S(n)$ are obtained.

2.4. Features under test

The main design goals for the PROOF system are: 1) transparency, 2) scalability, 3) adaptability
[2,3,4].

1) The transparency is aimed by obtaining as little difference as possible between a local
ROOT based analysis session and a local or remote parallel PROOF session, both being
interactive and giving the same results. In these tests it has been preliminary checked that
the execution time of a PROOF-Lite task with only one worker, $T_1$, is consistent with the
processing time of a ROOT job performing the same analysis macro on the same ROOT files.

2) The scalability of the basic PROOF architecture is reached once there is not any implicit
limitation on the number of cores that can be used in parallel. The performance tests presented
here are centered on the scalability of PROOF-Lite and speed-up is the appropriate figure of
merit. PROOF scalability is a feature that has been already proven [4,6] at the low values
of the number of workers that are used here. In these tests, however, a comparison among
different technological solutions (personal workstation versus multi-user interactive server, SSDs
versus SAS disks) is provided for PROOF-Lite by still looking at the speed-up, in terms of
measurements of the degree of departure from an expected linear scalability. A PROOF-Lite
task has been realized with a ROOT macro characterized by being more I/O-limited than CPU-
limited. This allows different storage technology solutions to be tested and compared. The
reduction algorithm implements a full sequential multi-step selection based on a relevant set of
variables/objects. In addition dozens of histograms are filled at each step and several hundreds
of histograms are overall written in the ROOT output file of the task. The performances of
this reference “full macro” are compared to the results for a less I/O-demanding version of the
same macro (called “short macro” in next section) obtained simply by eliminating the heavy
histogram writing part.

3) The adaptability is a feature that mostly characterizes PROOF system when configured
to operate within a single cluster or even within a virtual global cluster. It represents how much
the system is able to adapt itself to variations in the remote environment like, for instance, the
changing load on the cluster nodes. In these tests, instead, PROOF-Lite is used to address
the simpler case of multi-core workstations or multi-user servers. Nevertheless the issue of
the performance variation related to changing load has been explored. Specifically the study
of adaptability is important when dealing with the multi-user server. In general it has been
always taken care to execute PROOF-Lite tasks when the MCIS was normally crowded by users
performing average demanding interactive tasks and the CPU wasn’t saturated. However a series
of tests has been intentionally carried out in somehow stressing conditions. These conditions
were obtained by executing a concurrent four-workers PROOF-Lite task in addition to the main
task and to the normal interactive activity of the other users.
3. Presentation and discussion of the results

The first result is presented in Fig. 1, which shows the comparison between the reference macro and the “short macro” previously defined. In this test the 8CMSU server is used and the four input ROOT data files are locally stored on the SSDs. The speed-up follows approximately the linear scalability for both versions, with the scaling performance being slightly better for the “full macro”. The average processing rate for the “short macro” is at least twice that for the I/O-demanding “full macro”; this could be reasonably expected since the inclusion or exclusion of hundreds of histograms in the macro has a crucial influence on the processing rate. The “full macro” shows two important features: a) it is I/O limited, b) it scales rather linearly by using the SSDs. These characteristics makes the PROOF-Lite task based on this reference macro and executed on the SSDs of the 8CMSU server a benchmark in the following tests.

In Fig. 2 the performances of SSD and SAS drives are presented; the test has been carried out on the 8CMSU server using only one ROOT file since the SAS disks is size-limited. As expected the performance on SSDs with a PROOF-Lite having a relatively shorter processing time reproduces very well the one presented in Fig. 1 for the same I/O-demanding macro. The average processing rate for the test carried out on SAS disks grows more slowly with the number of workers till it saturates when more than 5-6 workers are exploited. This saturation effect is also visible in the speed-up which begins its departure from linear scalability already for four workers. This effect reveals that the task executed on SAS disks is completely I/O-bound. Consequently the computing power provided by the CPU of the server cannot be efficiently used. This observation has been confirmed by direct monitoring of the server’s CPU usage.

In Fig. 3 the performances of 8CMSU (red stars) and MCIS (yellow band) are presented for the execution of the reference PROOF-Lite task, namely the I/O-demanding macro run on the four ROOT input files. The yellow band represents the performance uncertainty related to the MCIS server being used chaotically by other users running their own tasks, even though performance data are collected always when the server is not overloaded. Its borders are defined as follows: black dots refer to PROOF-Lite task executed without further concurrent PROOF-Lite tasks, whereas green dots are obtained when an additional four-workers PROOF-Lite task is run on the server competing for the CPU resources with the task under measurement. The additional PROOF-Lite task is executed on a different ROOT file: the merge file of the four
ROOT files.

The MCIS server shows higher average processing rate capability than the 8CMSU server. The former, far from overload conditions, is able to provide quite linear scalability, even if speed-up is still slightly lower than that of the latter server. The fact that a PROOF-Lite task on MCIS quite linearly scales with the number of used workers, provided that the server is operating far from overload conditions, might be surprising. Indeed the input files are locally stored on SATA disks and the results of SSDs and magnetic disks shown in Fig. 2 apply. However the MCIS server is equipped with faster CPUs, wider RAM supply and a powerful RAID: its hardware

Figure 2. $<R(n)>$ (left) and $S(n)$ (right) as a function of the number of workers ($n$) when comparing the PROOF-Lite task based on the reference macro and executed on one ROOT file stored either on SSD (red stars) and on a SAS disk (green crosses) of 8CMSU. Often errors are within marker’s size, like for red stars. Dashed line on the right plot indicates linear scaling.

Figure 3. $<R(n)>$ (left) and $S(n)$ (right) as a function of the number of workers ($n$) when comparing the PROOF-Lite task based on the reference macro and executed on a set of four ROOT files stored either on SSDs of 8CMSU (red stars) and on the SATA disks of MCIS (yellow band). The yellow band represents the performance uncertainty related to the load of MCIS as discussed in the text. Often errors are within marker’s size (always for red stars). Dashed line on the right plot indicates linear scaling.
configuration prevents from a saturation effect as found for the HDD-SAS disks of 8CMSU server. Nevertheless the limitation of the MCIS performance are behind the corner: departure from linear scalability begins at \( n = 8 \) in the speed-up plot (black dots) and its evolution can be predicted on the basis of the green dots’ evolution: the departure from linearity appears as soon as \( n \geq 4 \), and increases with \( n \) (but without saturating), when a concurrent four-workers PROOF-Lite task is running as well.

In the previous test the comparison is not applied between enough homogeneous servers but between two realistic alternative solutions that can be chosen at Tier-2/Tier-3 facilities. The difference is not only technological but also economical since the cost of a MCIS-like server is about three times higher than the cost of a 8CMSU-like server. A useful estimation of the average time duration of the PROOF-Lite tasks is provided by the following values:

- for MCIS server: \( T_1 \approx 4h, 45m \) and \( T_8 \approx 38m \div 48m \);
- for 8CMSU server: \( T_1 \approx 9h, 25m \) and \( T_8 \approx 1h, 7m \).

There is a striking gain for the analysis time-schedule of the end-user. The end-user (or his analysis team) would get a fast turn around of the analysis by having exclusive use of a good workstation, SSD-equipped, like the 8CMSU server. The benefit would be even greater by exploiting a multi-user powerful server, like the MCIS server, provided it works with moderate load as far as the other users’ concurrent activity is concerned.

Tier-2/Tier-3 managers would be required to evaluate which choice could better fit the evolving interactivity needs of the end-users of their facility. They should take into account the evolving total and average number of concurrent active users and how much widespread the use of PROOF-Lite currently is and may be in the near future, among the active analysis teams hosted at their facility.

![Figure 4.](image)

**Figure 4.** \( \langle R(n) \rangle \) (left) and \( S(n) \) (right) as a function of the number of workers \( (n) \) when PROOF-Lite task is executed on a set of four ROOT files (red stars) and on one corresponding merge ROOT file (blue triangles); in both cases files are stored on the SSD of 8CMSU. Often errors are within marker’s size (always for red stars). Dashed line on the right plot indicates linear scaling.

The results presented in Fig. 4 refer to the test, performed using the 8CMSU server and its SSDs, in which the reference PROOF-Lite task executed on the usual full set of four ROOT files is compared to the same PROOF-Lite task executed on their merge ROOT file. The two solutions show a rather similar behaviour even if, when the merge file is used as input, the speed-up dependence upon the number of exploited workers is slightly below the linear scalability. This
difference is related to the fact that the reference test with one worker provides a slightly lower value of $T_1$ (entering at the numerator of $S(n)$) when using the merge file (and correspondingly a slightly higher value of $< R(1) >$). A feasible explanation could be that the reading access to the four separated files is slightly more sequential and thus slightly more efficient than accessing the same unique merge file.

4. Conclusions
The data collected by the LHC experiments are increasing in size and the typical interactive steps of a data analysis task are often becoming a challenge. Parallelizing the execution of tasks such as event selection or systematics studies will be of great help to the HEP analysts allowing them to rapidly produce valuable results. PROOF is proved to be a reliable and high performance tool for data analysis also on a single server environment. In this study it is shown and discussed how it is possible to obtain positive results by means of a small dedicated server if this is configured with SSDs. Indeed the use of Solid State Drives allows the parallelized execution of a typical PROOF-Lite analysis task without suffering by the I/O bottleneck.

Acknowledgments
We would like to thank G.Ganis for his helpful suggestions and M.Maggi for useful discussions.

References
[1] R. Brun and F. Rademakers, *Nucl. Inst. & Meth. in Phys. Res. A* 389, 1997.
[2] M. Ballintijn et al., *Nucl. Inst. & Meth. in Phys. Res. A* 559, 2006.
[3] M. Ballintijn et al., Proceedings of CHEP 2003 Conference, San Diego, California, arXiv:physics/0306110v1.
[4] G. Ganis et al., Proceedings of Science, XII Advanced Computing and Analysis Techniques in Physics Research (ACAT08), 2008, Erice, Italy.
[5] http://root.cern.ch/drupal/content/proof; a wider list of PROOF publications can be found at http://root.cern.ch/drupal/content/publications.
[6] S. Y. Panitkin et al., Proceedings of CHEP 2009 Conference, *Journal of Physics: Conference Series* 219, 052011, 2010.