SECOND GENERATION LHC ANALYSIS FRAMEWORK: WORKLOAD-BASED AND USER-ORIENTED SOLUTION

S. Boychenko, C. Aguilera-Padilla, M.A. Galilee, J.C. Garnier, A. Gorzawski, K. Krol, J. Makai, M. Osinski, M.C. Poeschl, T.M. Ribeiro, A. Stanisz, M. Zerlauth, CERN, Geneva, Switzerland
M.Z. Rela, CISUC, University of Coimbra, Portugal

Abstract

Consolidation and upgrades of accelerator equipment during the first long LHC shutdown period enabled particle collisions at energy levels almost twice higher compared to the first operational phase. Consequently, the software infrastructure providing vital information for machine operation and its optimisation needs to be updated to keep up with the challenges imposed by the increasing amount of collected data and the complexity of analysis. Current tools, designed more than a decade ago, have proven their reliability by significantly outperforming initially provisioned workloads, but are unable to scale efficiently to satisfy the growing needs of operators and hardware experts. In this paper we present our progress towards the development of a new workload-driven solution for LHC transient data analysis, based on identified user requirements. An initial setup and study of modern data storage and processing engines appropriate for the accelerator data analysis was conducted. First simulations of the proposed novel partitioning and replication approach, targeting a highly efficient service for heterogeneous analysis requests, were designed and performed.

INTRODUCTION

It has been almost a year since the LHC [1] was successfully re-commissioned after the long shut-down phase and for the whole operational period the performance of the transient data storage and processing systems has been closely monitored. As expected [2] the amount of information to be persisted and analysed has been steadily growing over the observation period and in more general has increased almost tenfold since the startup of the LHC in 2006. According to our observations, the acquisition data rates will continue to increase, especially with the considerable infrastructure improvements foreseen in the context of the High Luminosity LHC project [3]. Despite the efforts for improving and optimizing current data storage infrastructures (new CERN Accelerator Logging System [4] and Post Mortem APIs [5,6]), the problems described in our previous work [2] still persist and require a solution to scale up efficiently with machine upgrades.

The study and evaluation of modern data storage and analysis frameworks led us to conclude that a plain integration of those tools into the existing infrastructure will not overcome the limitations of the current analysis process. A scalable long term solution can be provided by putting an emphasis on studying in detail user requirements and current analysis shortcomings and build a workload-driven framework based on these observations. Amongst the many factors determining the system performance, we believe that the influence of the data storage level is crucial for the success of the final solution. Hence we propose a novel Mixed Partitioning Scheme Replication (MPSR), which we believe can greatly enhance the characteristics of the data storage and analysis framework. The proposed solution aims at providing an improved query response time for heterogeneous workloads by optimizing individual replica's partitioning for determined query types. In this work, we present the study we have conducted to prove the efficiency of MPSR and the methodology which will be used to continue studying the viability of the proposed solution in CERNs operational environments.

The remaining document is organized as follows: the following section focuses on providing the proof of the Mixed Partitioning Scheme Replication model viability. The 3rd section describes the experimental setup which is being used to study the solution's performance with real data and use cases. In the section on future work we outline the goals for future research and developments. Finally in the last section the conclusions are presented.

MIXED PARTITIONING SCHEME REPLICATION STUDY

One of the key factors which defines the strategies of replicating and partitioning using the proposed MPSR technique is workload. Hence the first phase of our studies was dedicated to the inquiry of current service use cases and usage patterns. For this we have examined Post Mortem (PM) and CERN Accelerator Logging frameworks (CALS) for the periods of 2012 and 2015, corresponding to the first operational run of the LHC. We were able to identify an increase of 27% in PM analysis sessions, whereas in manual data processing (which correspond to only 0.9% of total sessions) we were able to observe an 159% increase in number of sessions. The distinction between manual and automatic analysis types is extremely important, since manual data processing is unpredictable and hard to characterize. On the other hand, for CALS we were not able to determine the exact statistics for the whole periods, since the latest API does not feature any workload monitoring module anymore. For the first two quarters of 2012 and 2015 an increment of 14% in number of requests could be observed. The manual analysis hereby corresponds to 0.62% of the total number of executed requests and increased for the indicated periods by 113%. Workload studies allowed to determine the most queried device types, the distribution of requests by device location in the LHC ring and the age of the data which is being accessed most frequently. Consolidation of collected

metrics with future use cases allowed us to conduct a qualitative analysis of data dimensions (time, device type, location, accelerator state, etc.) and prioritize partitioning strategies for the first experiments.

Based on our findings in workload analysis we have designed a model which allowed us to study the viability of homogeneous Mixed Partitioning Scheme Replication before actually implementing and integrating it into modern data storage and processing tools. Among the different studied scenarios we are going to present the one which compares the best to the real database load. The proposed mathematical model [Eq. (1)] simulates the data storage accepting multiple queries at the same time, which do not follow uniform distributions for incoming request types.

$$T = \left(\frac{s * Rg}{Mo}\right) * \left(p * \frac{c-1}{Mo} + 1\right) \quad (1)$$

$$Rg = \frac{s-g}{s} \quad (2)$$

$$Ti = \frac{c * s}{M} \quad (3)$$

The variables which were used in the above formula are described in the Table 1. The final value which we are calculating, T, is an average execution time when executing a determined query type on an optimized partitioning scheme.

Table 1: Mathematical Model Variables

Variable	Variable Meaning
M	total number of nodes in the cluster
Mo	number of nodes with optimized partitioning (for current workload)
c	concurrently executed workloads
s	data size to be processed
g	data size reduction when executing query on nodes with optimized partitioning
p	probability of the same workload type being executed concurrently

The Rg variable in the execution time formula [Eq. (2)] corresponds to the rate of data processing on nodes with optimized partitioning for a determined query type. In the presented use case, the rate of processing is not being taken into consideration, since we assume that cluster resources are being constantly occupied executing other queries (than those they are optimized for).

Finally, the goal of the simulation is to determine the workloads where the Mixed Partitioning Scheme Replication execution time is smaller in comparison to an average execution time on the same infrastructure where the partitioning is equally optimized for all workload types [Eq. (3)]. When calculating Ti we assume that the cluster uses a fair scheduler, which distributes equally the resources amongst the active jobs.

Deduced workload execution formulas neglect job staging and concurrency overhead times to simplify the calculations and assume that the main factor which is influencing execution time is the data size to be processed s (for which we were able to observe a 99% correlation between execution time and input data size when running Hadoop [7] and Spark [8] experiments).

For the purpose of the simulation we have considered that the maximum possible gain in the amount of data processing is 50% of the total number of blocks to be read from the filesystem. Simulation results suggest that we would see MPSR performing better in 19.94% of the analysed workload combinations (the gains are presented in Fig. 1).

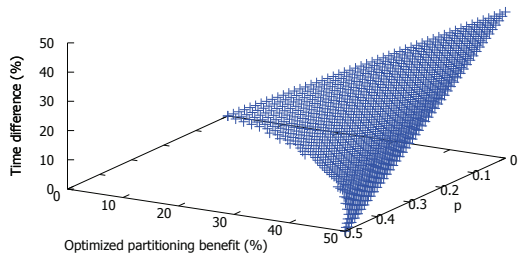

Figure 1: Normalized time gains with optimized partitioning.

For the simulations we have tried different combinations of the variables and the results suggest that p and Rg are playing a vital role in the performance of MPSR. According to the above figure we could conclude that the most significant improvements are observed when the probability of execution of concurrent workloads is low and the optimized partitioning gains are significant. The obtained results, for the simple use case, are encouraging us to investigate further the viability of the proposed solution and design more sophisticated simulations which would allow to better understand Mixed Partitioning Scheme Replication.

EXPERIMENT SETUP

In parallel with the theoretical study of the Mixed Partitioning Scheme Replication we have started setting up the infrastructure for the first benchmarking experiments. During the first phase of the tests we use a relatively small cluster (the machine configuration is presented in Table 2) with a relatively small amount of data, in order to be able to reset the environment and migrate the data fast. During the second phase we plan to scale the benchmarks to a larger cluster which closely replicates the production environment with most promising configurations derived from the initial experiments.

So far we already performed tests with Hadoop Map Reduce and Apache Spark installations. Both of the data processing engines are using HDFS for data storage and YARN for resource management. All the experiments were performed without any optimizations of the mentioned data storage and processing engines. The input data was extracted

ISBN 978-3-95450-147-2

Table 2: Cluster's Node Configuration

CPU	8 Core Intel(R) Xeon(R) E5420 2.50GHz
RAM	8 GB DDR2 667 MHz
Storage	2x1TB SATA 7200 rpm

from CALS and stored in plain text (variable name, measurement stamp and measured value).

To simulate incoming queries we have developed a highly parametrizable workload generation tool, which in addition is able to collect job related metrics. There are two workload categories supported by the tool: operational and analytical. An operational query module allows to generate most common queries observed during CALS workload analysis, which does not operate on large amounts of data. On the other hand, the analytical query module generates requests which require processing significant amounts of data.

In initial experiments we have used simple partitioning, based on measurement time and device type as main dimensions for organizing the data. Resulting directories were organized in a tree structure, divided by year, month, day and device_type metadata. One of the main problems of the proposed partitioning solution is with its load balancing. Devices which report large amounts of data will produce directories with very large amount of blocks, inducing a high variation in average query execution time in workload execution experiments.

In the Table 3 we present operational workload experiment results executed on the described infrastructure. Each test consisted of executing 1000 queries on the same configuration and was repeated three times, on different days preceded by a complete infrastructure reset. One could conclude that with an operational workload Spark was performing worse than Map Reduce for every use case. The investigation revealed that Spark jobs were severely penalized by the staging overhead when analysing directories containing only a few blocks (which are predominant in described partitioning). Additionally, the amount of intermediary data generated by jobs was not enough for Spark to take advantage of in-memory processing. We could also observe that in some cases the average data processing rate was worse when being executed on more nodes. Isolated test analysis allowed us to conclude that once again the staging overhead for jobs with small amount of data to be processed were influencing heavily the average processing rate.

Therefore in the analytical workload simulations, the results (see Table 4) show Spark outperforming Map Reduce up to seven times. In later experiments there was enough intermediary data to trigger Spark in-memory computations and take advantage of its strong points.

The observations of both tools with different workload types encourages us to continue the exploration of partitioning solutions, which we believe is the key performance factor in heterogeneous workload environments.

Table 3: Average Data Processing Rate (MBytes/sec) for an Operational Workload.

Nodes	Stats. Calculation	Filtering	Iterative
Hadoop (Map Reduce)			
1	46.41	39.69	40.69
2	73.03	65.18	55.96
3	85.53	74.33	50.66
Apache Spark			
1	37.35	36.79	23.5
2	67.63	63.71	44.83
3	77.09	71.9	46.98

Table 4: Average Data Processing Rate (MBytes/sec) for an Analytical Workload

	1 node	2 nodes	3 nodes
Hadoop (Map Reduce)	19.15	21.99	27.94
Apache Spark	56.78	103.65	159.14

FUTURE WORK

Based on the findings presented in this work we were able to identify several directions which require additional investigation to approach the phase of designing the new infrastructure. First of all we need to perform extensive exploratory analysis on the proposed data storage solution (MPSR), to study the scenarios when non-optimized partitions will be used to process some data (variable request arrival rate for example). Additionally we need to understand how the solution will perform when uncategorised queries arrive and none of the replicas is optimized for their processing. Secondly, we need to investigate different partitioning schemas which would be efficient for the described workloads and tools' combination. Finally, we need to extend the experiment scope to different distributed data storage and processing engines, as well as scale the benchmarks of the most promising solutions to a production-like environment.

CONCLUSION

The results of the first simulations have confirmed the superiority of the proposed data partitioning and replication technique relatively to conventional models for the described scenario. We have identified the most significant factors which influence the efficiency of the model and the regions where theoretically we could expect the MPSR to perform better in comparison to general-purpose partitioning and replication. We could conclude that further analysis of the proposed solution is required to determine more accurately its strengths and weaknesses before implementing the first prototype. Additionally we need to choose carefully the data storage and processing engines, since it specificities will have a great impact on the future analysis framework.

REFERENCES

[1] LHC Study Group et al., "The large hadron collider, conceptual design," CERN, Geneva, Switzerland, Rep. CERN/AC/95-05, Oct. 2012.

[2] S. Boychenko et al., "Towards a second generation data analysis framework for LHC transient data recording," in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015, paper WEPGF046, pp. 802-805.

[3] L. Rossi,"LHC upgrade plans: options and strategy," in *Proc. IPAC'11*, San Sebastian, Spain, Sep. 2011, paper TUYA02, pp. 908-912.

[4] C. Roderick et al., "The CERN accelerator measurement database: on the road to federation," in *Proc. ICALEPCS'11*, Grenoble, France, Oct. 2011, paper MOPKN009, pp. 102-105.

[5] O. Andreassen et al.,"The LHC post mortem analysis framework," in *Proc. ICALEPCS'09*, Kobe, Japan, Oct. 2009, paper TUP021, pp. 131-133.

[6] C. Aguilera-Padilla et al., "Smooth migration of CERN post mortem service to a horizontally scalable service," in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015, paper WEPGF047, pp. 806-809.

[7] Apache Hadoop, `https://hadoop.apache.org/`

[8] Apache Spark, `http://spark.apache.org/`

ISBN 978-3-95450-147-2