Smart Caching at CMS: applying AI to XCache edge services

Daniele Spiga[1,1], *Diego Ciangottini*[1], *Mirco Tracolli*[1,2], *Tommaso Tedeschi*[1,5], *Daniele Cesini*[3], *Tommaso Boccali*[4], *Valentina Poggioni*[5], *Marco Baioletti*[5], *Valentin Y. Kuznetsov*[6]

[1]INFN Sezione di Perugia, Via Alessandro Pascoli 23c, 06123 Perugia (ITALY)
[2]Università degli studi di Firenze, Viale Morgagni, 67/a - 50134 Firenze (ITALY)
[3]INFN-CNAF, Viale Carlo Berti Pichat, 6/2, 40127 Bologna (ITALY)
[4]INFN Sezione di Pisa, L.go B. Pontecorvo 3, 56127 Pisa (ITALY)
[5]Università degli Studi di Perugia, Via Alessandro Pascoli 23c, 06123 Perugia (ITALY)
[6]Cornell University, Ithaca, (USA)

Abstract. The projected Storage and Compute needs for the HL-LHC will be a factor up to 10 above what can be achieved by the evolution of current technology within a flat budget. The WLCG community is studying possible technical solutions to evolve the current computing in order to cope with the requirements; one of the main focus is resource optimization, with the ultimate aim of improving performance and efficiency, as well as simplifying and reducing operation costs. As of today the storage consolidation based on a Data Lake model is considered a good candidate for addressing HL-LHC data access challenges. The Data Lake model under evaluation can be seen as a logical system that hosts a distributed working set of analysis data. Compute power can be "close" to the lake, but also remote and thus completely external. In this context we expect data caching to play a central role as a technical solution to reduce the impact of latency and reduce network load. A geographically distributed caching layer will be functional to many satellite computing centers that might appear and disappear dynamically. In this talk we propose a system of caches, distributed at national level, describing both deployment and results of the studies made to measure the impact on the CPU efficiency. In this contribution, we also present the early results on novel caching strategy beyond the standard XRootD approach whose results will be a baseline for an AI-based smart caching system.

1 Introduction

With the upcoming High Luminosity LHC (HL-LHC) [1] program at CERN all HEP experiments will face a new challenge, the exabyte era of computing [2]. A huge increase of storage and computing requirements are foreseen at LHC [3] and the needs are above the expected technology evolution as well as the funding. As an example, the estimated CMS [4] experiment annual disk requirements for the next decade compared to past and current

[1] Corresponding author: spiga@infn.it

ones are shown in Fig.1. In order to cope with this, a series of R&D programs have been established with the purpose of finding viable solutions for the optimization of the computing models. In this context the activity presented in this work focuses on the storage, looking for solutions not only in order to minimize the hardware usage but also to increase performances, e.g. to improve CPU efficiency by reducing I/O latencies and, of course, to introduce handles to simplify operations, which represent an important cost to the collaboration.

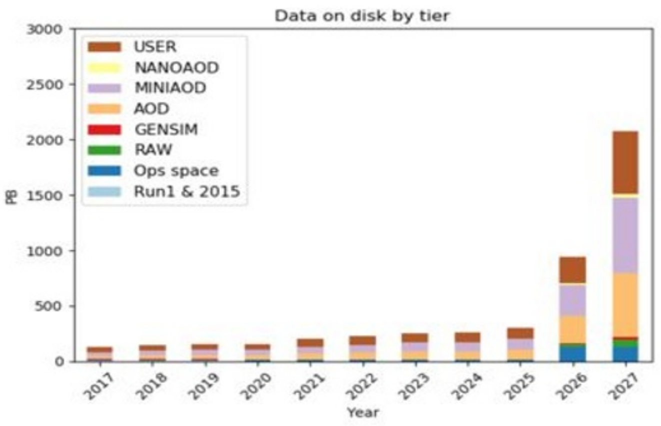

Fig. 1. Estimated CMS annual disk requirements compared to past and current ones

A primary objective in this R&D phase is to study how to optimize data usage and access for the end user analysis cases; the production data (MonteCarlo generation and data reprocessing/reconstruction) are still meant to be managed by a higher-level infrastructure.

The reference model in this work has been the one described in the "Data Access on a Data Lake straw man model" document prepared in the context of the access working group of WLCG Data Organization Management Access [5]. The document defines the Data Lake as a group of Data and Compute Centers with undefined borders by construction. The association of centers inside the Data Lake is done in terms of network latency, and, more in general, the shape of physical Data Lake topology is not meaningful and it might exist only at the Data Management layer. The document envisions a small number of Data Lakes across the world, with a very reduced number of storage endpoints with respect to the current WLCG [6]. An experiment (also referred to as VO) can be supported by several Data Lakes.

Regarding the data access, the model envisions a mix of distributed caches directly accessed from compute nodes, thus making the whole lake topology transparent. As a consequence, the caching system is a key component. The data cache is meant as a self-managed ephemeral storage layer at the edge of a site for streaming data and providing read ahead capability, which allows to reduce the impact on latency and thus optimize the CPU efficiency. Last but not least, a data cache, in a Data Lake topology, has the role to reduce the wide-area network bandwidth required to serve clients' requests.

In this contribution, we focus on data cache layer as meant in the above description. In Sec.2 we report on the deployment done to integrate an INFN federation of distributed caches within the "Anydata, Anytime, Anywhere (AAA)" federation [7] of CMS. This includes a summary of the studies made to measure the effect of data caches on CPU job efficiency. Then we introduce the ideas and strategy for a disk cache optimization (Sec.3). Finally, we show results of a simulation study done to evaluate the effect and potentiality of a custom caching logic. We conclude with the plan for the extension toward an ML-based mechanism for caching management.

2 The INFN distributed cache system

In a future Data Lake model, as described above, we have implemented a geo-distributed layer of cache servers at INFN [8]. The setup relies on XRootD [9] technology, a modular, fast, low-latency, and scalable system which can be configured to perform a variety of services:

- Data Source: XRootD server that serves data
- Redirector: XRootD server that communicates with other XRootD servers to locate a requested file
- XCache: a special configuration of XRootD that features a caching mechanism

Such setup seamlessly integrates the CMS AAA federation and allows leveraging cached data over a national high-bandwidth network to optimize the amount of disk space for user analysis workflow. As shown in Fig. 2 the testbed has been deployed at 3 Italian computing sites (namely Legnaro and Bari Tier2s, and CNAF Tier1). This is an activity done also in collaboration with the Extreme Data-Cloud (XDC) project [10] and in the future collaboration with ESCAPE EU project [11].

2.1 Studies on cache effect on CMS analysis jobs

Studies on the effect of a cache layer on the CMS analysis workflow have been performed using monitoring data from CERN MONIT project [12]. Data from all 2018 have been used and filtered by data format: the presented results refer to MINIAOD (recorded data) and MINIAODSIM (simulation). Only user jobs requiring datasets in these formats and running in one of the italian Tier2's have been analyzed and two main categories have been created based on the data access pattern. In fact CMS payloads can read data either directly from the storage at the site where they are running or remotely from another site on the grid (the AAA federation mentioned above). This can be done on purpose (chosen by the users at job submission time) or as a fallback mechanism when a job fails reading files from local

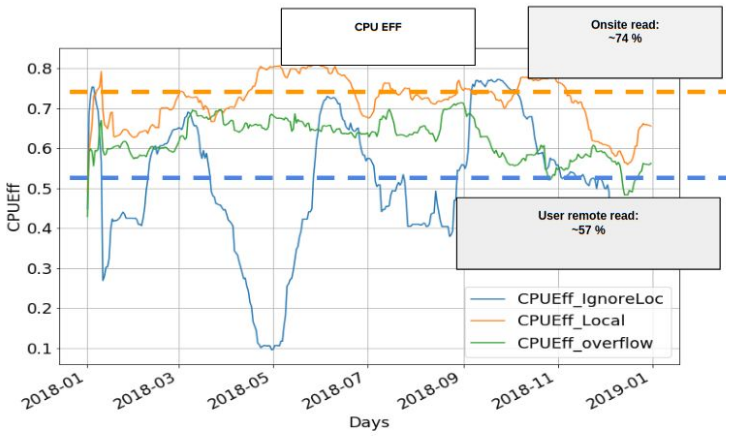

Fig. 2. Distribution of CPU efficiency over 2018. Yellow is the CPU eff. of jobs reading data from the local storage. Blue color relates to jobs reading data from remote storage (no Cache is involved).

storage. Beside the detailed studies in such cache system [8] the metric evaluated in this specific analysis is the CPU efficiency. This, again, refers to the efficiency of the analysis workflows. CPU efficiency is defined as the ratio of *sum of CPU time* and *sum of walltime*. As shown in Fig.3, remote data reading, without any caching system, costs on average about 15% of CPU time with respect to local data reading. These results look promising and thus it has been decided to experimentally measure the very same effect on the distributed cache setup at INFN and to compare the effect of the caches with respect to the CPU efficiency. First of all we defined the 5 scenarios to be compared:

- *On Site*: No Cache. Payloads run at available Italian Tier2s reading input data from the local storage element.
- *Overflow*: No Cache. Payloads run at available Italian Tier2s reading input data from local storage and, for failing jobs, fallback enforces remote data reading.
- *Ignore Locality*: No Cache. Payloads run at available Italian Tier2s while reading input data remotely, from the original site.
- *Cold Cache*: Payloads run at available Italian Tier2s and seamlessly use the regional cache cluster for input data starting from an empty cache, so testing performances of proxy while caching mode.
- *Warm Cache*: after *Cold Cache* test, the same workflow is re-run to test performances when data already in cache are requested.

To this end we did a series of dedicated tests by enforcing the usage of the cache for some of the jobs running in the INFN Tier2. As shown in the histogram in Fig.4, where the

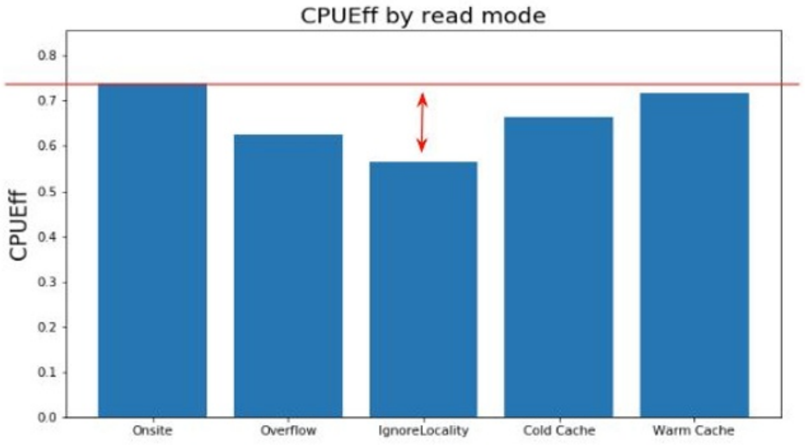

Fig. 3. Mean value of CPU eff. by identified scenarios. Red line shows how the *Warm Cache* allows us to reach almost the same value of the CPU eff. obtained running jobs that read from local storage. The arrow highlights the loss of CPU eff.

results of the CPU eff. for each category of job have been reported, the two main results obtained are that *Warm Cache* setup allows to obtain the very same CPU efficiency as in the *On Site* scenario; the *Cold Cache* setup shows how the read-ahead functionality allows for a sizeable improvement in efficiency. It is possible to conclude, as expected, that with CMS analysis workflows, a distributed setup of federated caches, as the one deployed at INFN, allows to reduce the overall WAN traffic and makes the processing job that requested the data more efficient by reducing I/O wait time for remote data.

3 Disk cache optimization: the idea

The obtained results represent the main motivations for the R&D activity discussed in this paper, which is about the exploitation of advanced solutions, possibly AI-based, to optimize the cache utilization. In other words, the aim of this study can be summarized as:

- Can the disk space utilization of the cache be reduced/optimized?
- Can the WAN traffic be reduced/optimized while keeping the optimal throughput from cache disk to clients?
- Can the routing between distributed caches be optimized ?

To answer these questions we decided to start some systematic studies on caching algorithm optimization. A key aspect of this R&D has been, since the beginning, to address these optimization studies being as much as possible experiment agnostic. For this reason the presented study does not rely on experiment specific monitoring data and as such does not depend on specific conventions like file names, but on information taken from a cache and thus related to files.

3.1 Strategy

The adopted strategy has been based on the idea of changing the baseline approach implemented by XCache middleware and to evaluate the possible improvements. The baseline XCache implementation assumes that all files requested by clients are saved on disk following the *write everything* approach. This is supposed to happen till a *high watermark* (defined as fraction of cache storage used) is reached. At that point the system starts deleting files until reaching a *low watermark*;file deletion happens on LRU (Least Recently Used) based replacement policy, i.e. files least recently requested are removed first.

Starting from this we propose to evolve towards a *suggestion based model* meaning that with respect to the *write everything* logic we can have something like *write only what it is worth to keep*. So far it's important to note that we do not propose to change the deletion logic based on LRU, that is kept *as-is*.

The adopted strategy introduces a *weighted hit rate* as a reference metric to give each hit and miss a weight based on file size. Starting from that, we want to evaluate the effect on cache throughput (and thus on the disk usage). To verify the behavior, we made a simulation considering all CMS analysis workflow data taken from CERN MONIT HDFS. In order to implement all these features, we decided to start simple using a function with the aim to define a baseline in the expected gain. In our vision, this will be the benchmark for the next implementation of a Machine Learning model for the AI smart decision service.

3.2 The weight function

In order to implement the *weighted hit rate* as defined in the previous section, we decided to give a score to all requested files and to use the score in order to build the baseline (the benchmark). We defined the weight of a given file as the following (the weight also refers as score):

$$file\_weight(t,f) = avg\_time(t,f) + \frac{size(f)}{tot\_req(t,f)^2} \tag{1}$$

where the *avg_time* is the average *time-delta* of the last *k-1* requests from the request *k*:

$$\frac{\sum_{i=1}^{k-1}(time_f(k) - time_f(i))}{k} \tag{2}$$

size is the size of the file and the *tot_req* is the total number of times a given file has been requested. The former function does not come from theoretical assumptions and modelling, but it was tested and selected among many, tells that the more the weight of a file, the less it is convenient to be kept in the disk cache.

As anticipated we finally extracted file weights using a simulation, based on historical data of CMS analysis workflows and we studied the weighted hit-rate.

4 Results

The simulation was performed considering two distinct geographical regions, namely Italy and US. For both, we compared the *write everything* approach with respect to the *write on suggestion* as described in Sec.3.1. The metric used for the comparison is the throughput from cache defined as:

- Throughput = (data served from the cache disk) / (data incoming from remote storage)

As shown in Fig.4 and Fig.5, the latter this quantity is almost doubled with the *write on suggestion* approach. The obtained result clearly confirms that the cache management

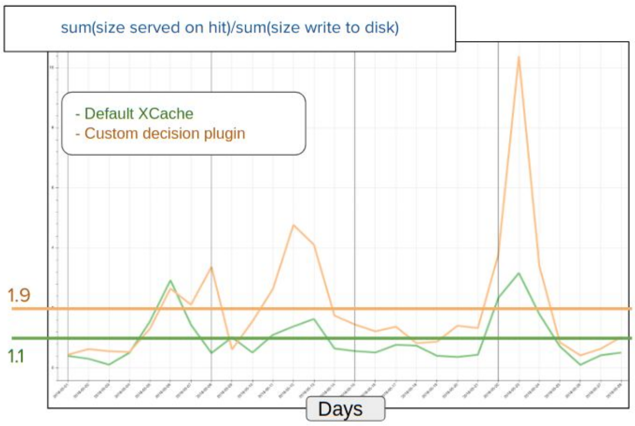

Fig. 4. The plot shows the sum(size served on hit)/sum(size write to disk) for jobs running only on Italian Tier2s. Yellow color refers to the *write on suggestion* mode. Green relates to *write all*, the default behaviour.

mechanism coming with standard XCache setup has room for improvements and optimization. The fact that two completely different regions in terms of the amount of resources, number of users (and thus the pattern of usage of the distributed computing systems) show the very same behavior is the first proof of the value of the strategy. Moreover, in order to further check the stability of the studied function, a set of control tests have been performed among which the sliding window check, calculating weights on week *i* and applying them on week *i+n.*

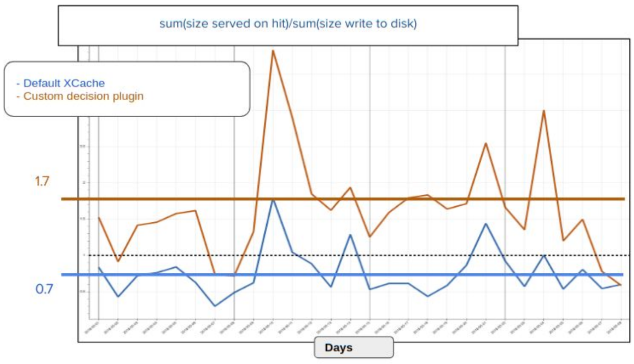

Fig. 5. The plot shows the sum(size served on hit)/sum(size write to disk) for jobs running only on US Tier2s. Brown color refers to the *write on suggestion* mode. Blue relates to *write all*, the default behaviour.

5 Summary and future directions

In this contribution we presented the infrastructural work done to implement a system of distributed cache, compliant with the current model foreseen for a WLCG Data Lake as well as the idea and early results using a novel and experiment independent approach to the data caching with XCache middleware.

All the obtained results are guiding towards further investments in the presented R&D. From the infrastructural perspective the geographical distributed system of cache does not introduce any problematic effect. All the tests and measurements show how beneficial a national layer of cache in front of the regular Tier sites of WLCG can be. Our plan is to move the testbed into full production for all the analysis workflows running at INFN sites. The plan also foresees to extend the described setup at international level exploiting the Data Lake testbed which will be provided by the ESCAPE EU Project.

The results obtained studying the possibility to improve the disk cache utilization, which in turn means to maximise the throughput of the cache system, seem really promising and convinced ourselves to move toward an AI based model and the direction we set is to move to a reinforcement learning strategy. The technical feasibility of the full integration of XCache with a smart decision service has been already demonstrated [13].

This work has been partially supported both by the XDC EU project G.A 777367, and by ESCAPE EU project, G.A. 824064.

References

1. The High-Luminosity LHC project. url: https://home.cern/topics/high-luminosity-lhc.
2. A. A. Alves Jr., et. al., "A Roadmap for HEP Software and Computing R&D for the 2020s", arXiv:1712.06982 [physics.comp-ph]
3. L. Evans and P. Bryant. "LHC Machine". In: Journal of Instrumentation 3.08 (2008), S08001

4. S. Chatrchyan et al. "The CMS Experiment at the CERN LHC". In: JINST 3 (2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004
5. D. Bersano et al. HEP Software Foundation Community White Paper Working Group -- Data Organization, Management and Access (DOMA), arXiv:1812.00761 [physics.comp-ph]
6. Worldwide LHC Computing Grid. url: http://wlcg.web.cern.ch/.
7. D. Ciangottini et.al, (2019). "Integration of the Italian cache federation within the CMS computing model." 014. 10.22323/1.351.0014.
8. K. Bloom et al. "Any Data, Any Time, Anywhere: Global Data Access for Science", arXiv:1508.01443 [physics.comp-ph]
9. https://xrootd.slac.stanford.edu
10. D. Cesini et al, (2018). "The eXtreme-DataCloud project: data management services for the next generation distributed e-infrastructures." 1-4. 10.1109/ROLCG.2018.8572025.
11. S. Campana et al, (2019) "ESCAPE prototypes a Data Infrastructure for Open Science", proceedings of this conference
12. A. Aimar, et. al., "Unified Monitoring Architecture for IT and Grid Services", Journal of Physics Conference Series 2017, 898 092033, doi: 10.1088/1742-6596/898/9/092033
13. M. Tracolli, *Using DODAS as deployment manager for smart caching of CMS data management system* (ACAT, 2019)