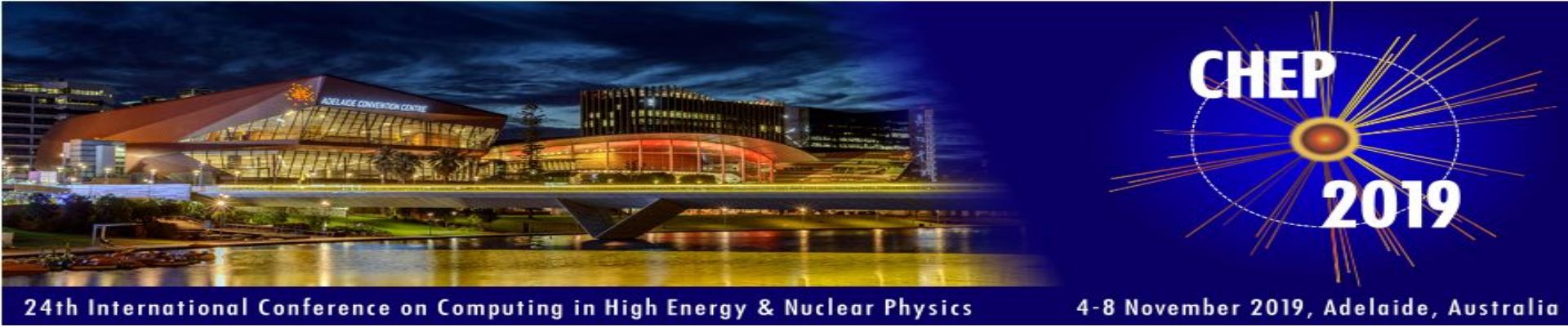


Smart caching at CMS: Applying AI to XCache edge services

Daniele Spiga - INFN - (spiga@infn.it)
On behalf of CMS Collaboration

Valentin Kuznetsov[1], Tommaso Boccali[2], Daniele Cesini[3], Marco Baioletti[4], Valentina Poggioni[4], Diego Ciangottini[5], Mirco Tracolli[5]

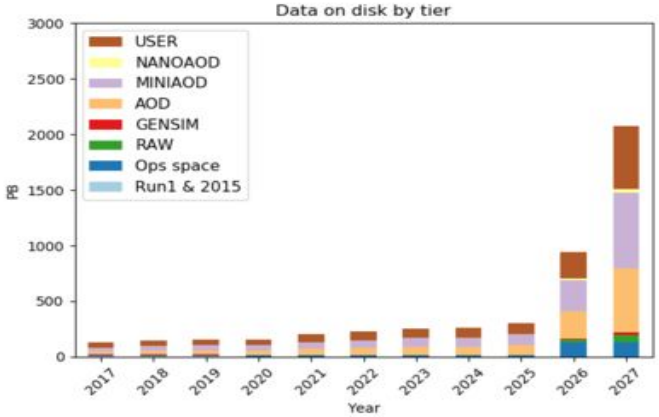
1. Cornell University (US), 2. INFN - Sezione Pisa (IT), 3. INFN CNAF - Bologna (IT), 4. Università degli studi di Perugia (IT), 5. INFN - Sezione Perugia (IT)

Outline

- Introduction to the Global context
- The testbed @ INFN
  - Setup and related studies
- Idea for cache disk usage optimization
  - Studies on simulation
  - Results and roadmap toward a AI based engine
- Summary and future plans

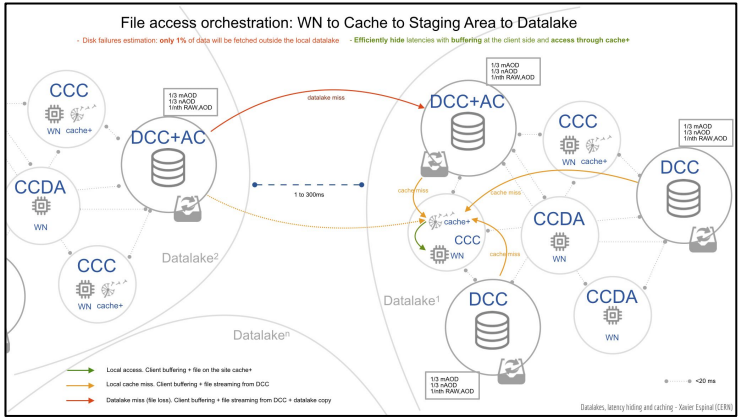
Global Context: Data Lake @ HL-LHC

- HL-LHC needs are above the expected technology evolution (and funding)
- Optimization is needed to reduce the hardware usage and operational costs
  - No a one size fits all solution



Disk requirements estimated to be required annually by CMS

From Data Lake Straw Proposal @ DOMA



Terminology:

AC - Archive center Defined as Tape or tape-equivalent-QoS enabled center able to archive custodial data.

DCC - Data and computer center providing disk-equivalent QoS storage

CCC - Compute center with cache

CCDA - Compute center without cache: relies on accessing all data via the network from either a CCC or a DCC

A small number of Data Lakes across the world

- Reduced number of storage endpoints wrt the current WLCG model

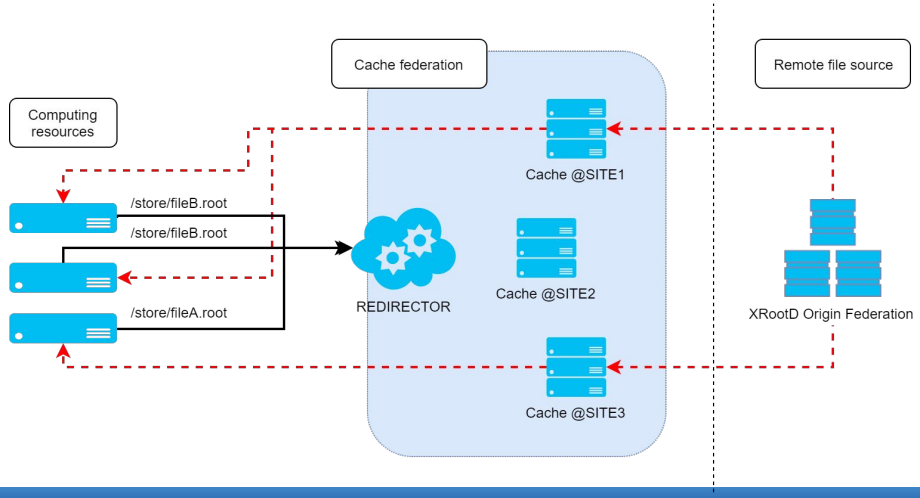
Envision a mix of distributed caches directly accessed from compute nodes

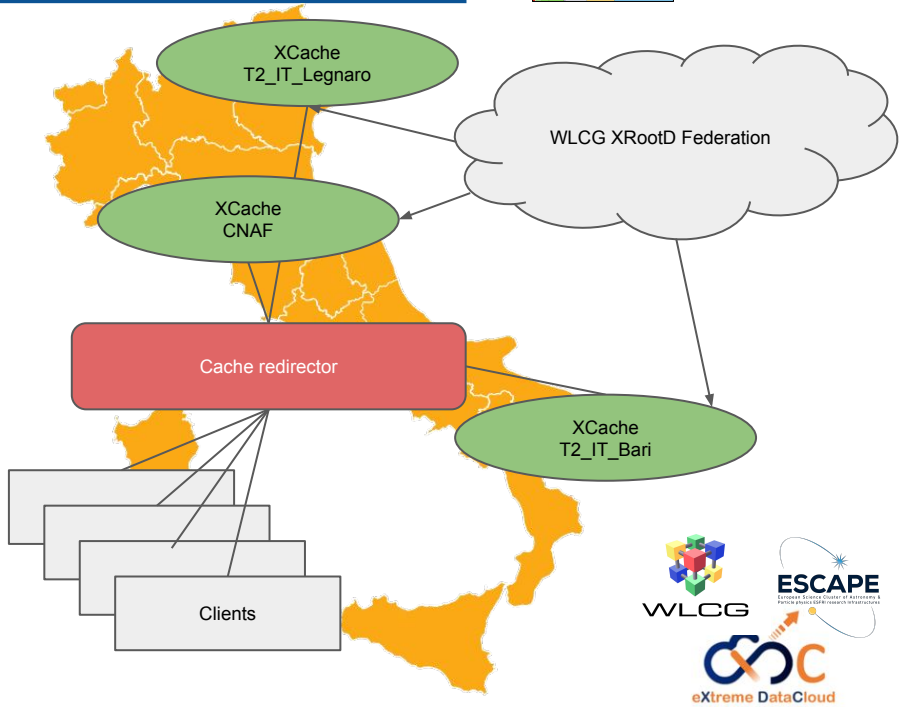
Testbed architecture

Rely on **XRootD** technology: **modular, fast, low-latency, and scalable**

Can be configured to perform a variety of services.

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





- CNAF XCache redirector federating 3 XCache servers
  - CNAF server (5TB spinning)
  - Bari server (10TB gpfs)
  - Legnaro server (22TB spinning)

Cache effect on CPU efficiency @ CMS

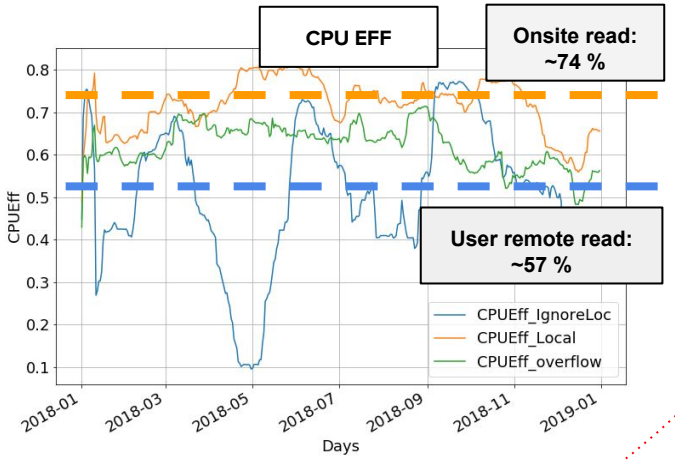
- We studied monitoring data of the whole **2018 CMS analysis workflows**
  - Remote data read costs on average **about 15% of CPU time w.r.t. Onsite data reading**

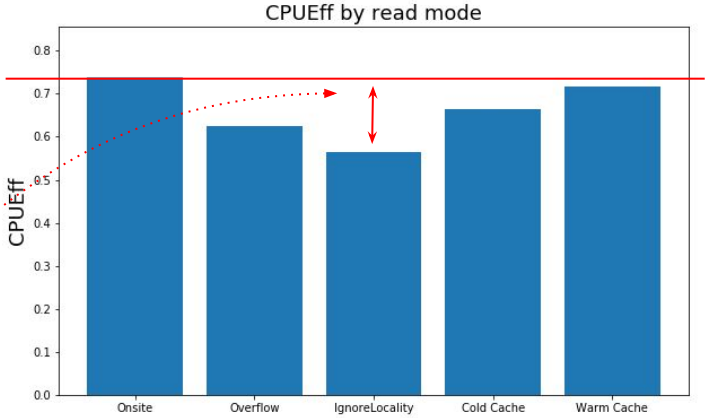
CPUEff = sum(cpu time) / sum (job time)

From data@CERN MONIT hdfs

Measured on testbed

@Italian Tier2's



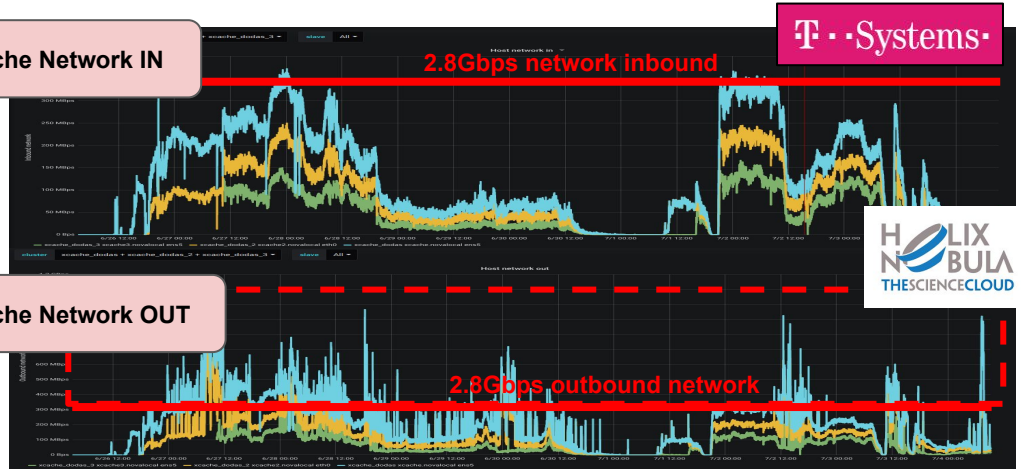


Caches allow to reduces the overall WAN traffic and, makes the processing job that requested the data **more efficient** by reducing I/O wait time for remote data.

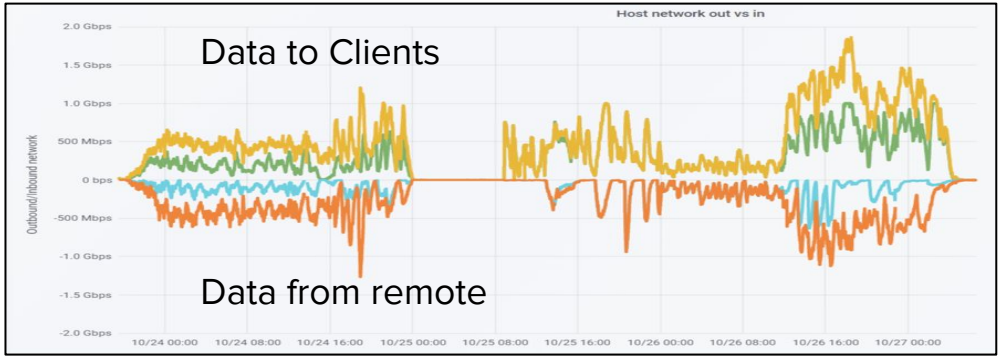
Compute & Cache: our experiences

Cache Network IN

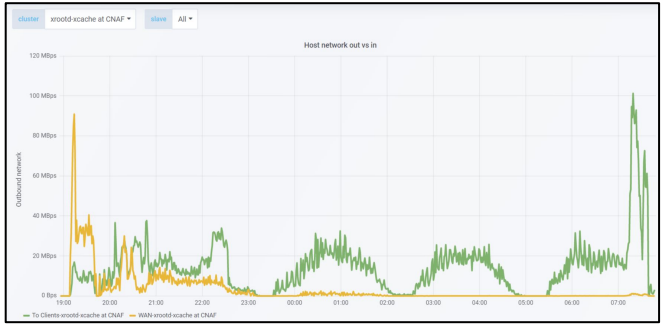
Cache Network OUT

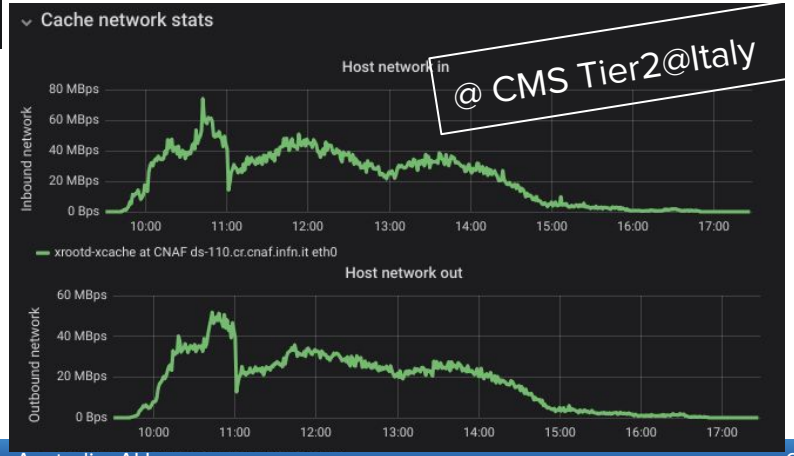


See: The DODAS experience on EGI Federated Cloud



See: Extension of the INFN Tier-1 on a HPC system





Raising the bar

- Can we exploit further and advanced solutions to optimize the cache utilization
  - Can we **reduce/optimize the disk space** for the cache?
  - Can we reduce **WAN traffic** while keeping the optimal **throughput from cache disk?**
  - Can we **optimize the routing** between distributed caches?
    - As longer term strategy

- R&D activity started on cache algorithm optimization
- Trying to be <u>experiment agnostic</u> as much as possible

XCache I/O

Maximize this component

Minimize this component

Strategy

- Based on current XCache implementation all file are saved on disk ➔ "write everything"
    - when "high watermark" is reached start deleting files until reaching "low watermark"
    - files are deleted on LRU base ➔ "files least recently requested are removed first"
- We want evolve toward a "suggestion based model"➔ "write only what it is worth to keep"
    a. And to use the **"weighted hit rate"** as reference metric
        - to give each hit/miss a weight based on file size

- We want to evaluate the effect on cache throughput (and thus disk usage) on a simulation, considering all CMS analysis workflow data @ CERN MONIT hdfs
    a. Start using a simple function to define a baseline in the expected gain: the benchmark
    b. Implementing a Machine Learning model for the AI smart decision service

Current approach: defining a baseline

We decided to give a score to all requested files, and use the score to build the baseline (the benchmark)

<u>We defined weight (the score) of a file as:</u>

File size

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

Numbers of file requests

Average *time-delta* of the last k requests

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

Extract weights in a simulation, based on historical workflow data @ CMS and study the weighted hit-rate

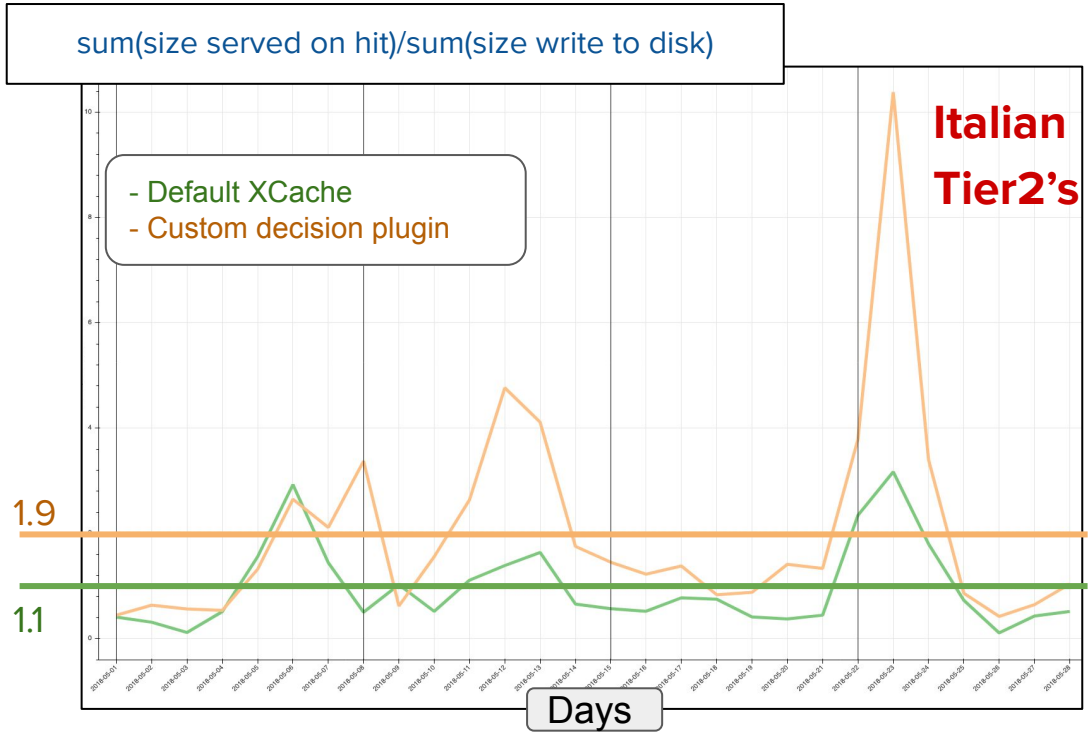
Results

Comparing the **write everything** mode wrt the **write on suggestion**

The **Throughput from cache** measured as the ratio between

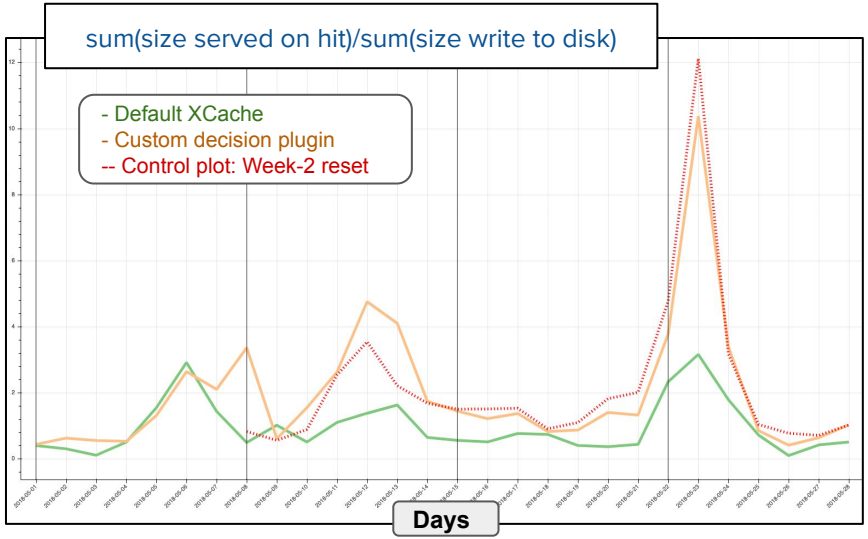
- the data served from the cache disk
- the data incoming from remote storage

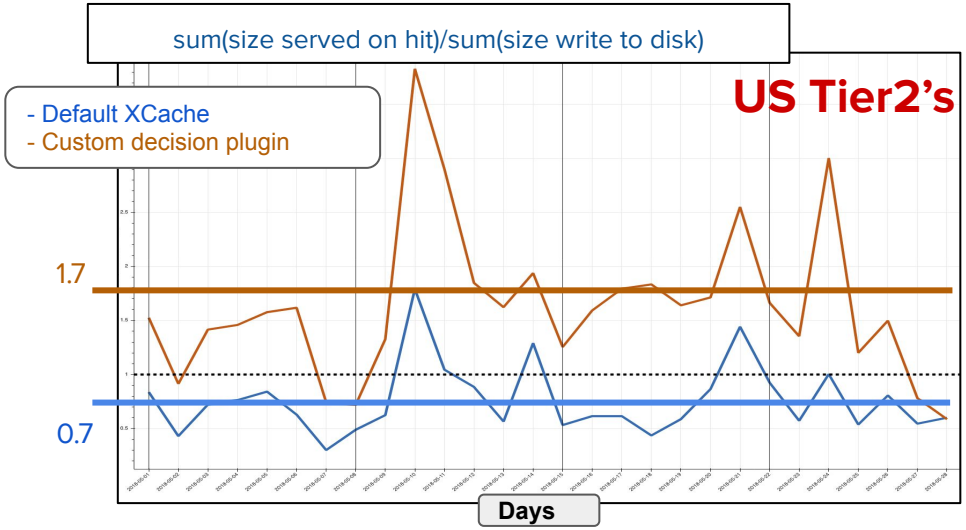
Is <u>almost doubled</u> with the write on suggestion approach



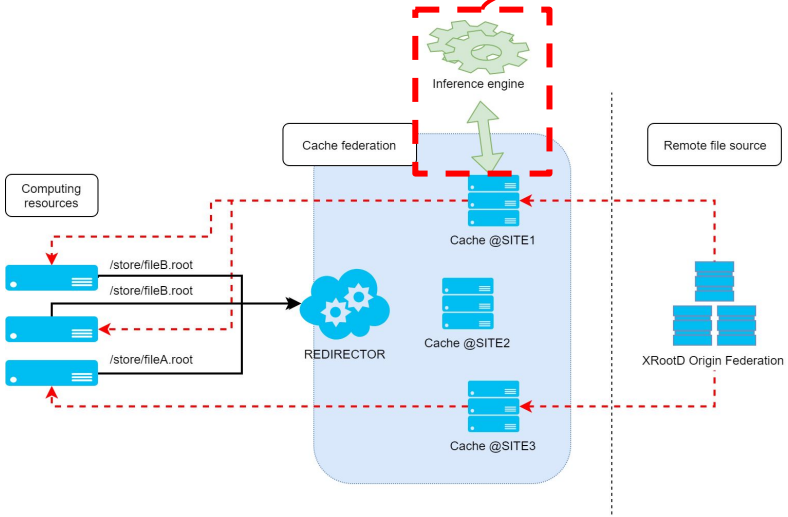
Checking the stability

- Sliding window check is OK
  - Calculate weights on week `i` and apply them on week `i+n`
- Same behavior on a different region
  - Simulation results on US-CMS region





How to integrate with XCache middleware




```
smartDecisionPlugin::Decision::Decide /store/mc/HC/GenericTTbar/GEN-SIM-RECO/CMSSW_5_3_1_START53_V5-v1/0011/BA1FAB4B-11AE-E111-957B-003048D3C932.root
smartDecisionPlugin::Decision::Decide number of rules 1
  trying .
* About to connect() to 193.204.89.71 port 4242 (#0)
*   Trying 193.204.89.71...
* Connected to 193.204.89.71 (193.204.89.71) port 4242 (#0)
> GET /resolve?lfn=/store/mc/HC/GenericTTbar/GEN-SIM-RECO/CMSSW_5_3_1_START53_V5-v1/0011/BA1FAB4B-11AE-E111-957B-003048D3C932.root HTTP/1.0
Host: 193.204.89.71:4242
Accept: application/json
Content-Type: application/json
charsets: utf-8

* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Content-Type: application/json
< Content-Length: 140
< Server: Werkzeug/0.16.0 Python/3.6.8
< Date: Mon, 28 Oct 2019 10:46:44 GMT
<
* Closing connection 0
{"filename":"/store/mc/HC/GenericTTbar/GEN-SIM-RECO/CMSSW_5_3_1_START53_V5-v1/0011/BA1FAB4B-11AE-E111-957B-003048D3C932.root","store":true}

  match. returning true
```

The smart decision plugin tells the cache to store the file

- XCache natively supports plugin based mechanism
  - We implemented a first custom plugin to contact a external inference engine
- A end-to-end test of a smart cache implementation has been performed at INFN XCache testbed

Toward a AI based solution

We are now convinced it is worth to move toward AI based model

- Starting with supervised learning approach:
  - Build a ground-truth: the best cache composition using Genetic Algorithms for a sub optimal solution
  - Deep analysis of the input features
    - Focussing primarily on cache data (experiment agnostic approach)
      - Study (clustering) & extend with domain specific inputs (dataset info, user type…)
  - Use the previously described function as baseline
- Move to a reinforcement learning strategy

Summary and future plans

After 1+ year of integrations, testing, operations and analysis of monitoring data, we achieved experience on XCaches & CMS analysis workflow

We presented an idea and early results of a novel and experiment independent approach to the data caching with XCache middleware

- Results motivate to invest further effort

We will extend the same approach and strategy to a geographically distributed setup of XCaches

Keep investing on synergies with XRootD developers as well as related activities

1. Analysis and modeling of data access patterns in ATLAS and CMS
2. Moving the California distributed CMS xcache from bare metal into containers using Kubernetes