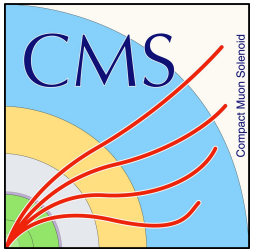


The Spanish CMS Analysis Facility at CIEMAT

A. Delgado Peris, J. Flix[1], J. M. Hernández, C. Morcillo Pérez, A. Pérez-Calero Yzquierdo[1], F. J. Rodríguez Calonge, J. León Holgado, M. Cárdenas-Montes

CIEMAT and PIC ([1])

Outline

- Motivation and context
- The Spanish Analysis Facility
- The CIEMAT analysis framework
- Performance measurements
- Support to other scientific communities
- Conclusions and outlook

Motivation and context

- The LHC Run3 & High Luminosity phase challenge: more and busier events
  - Higher pile-up, higher trigger rate, more Monte Carlo simulated events
- Analysis at HEP
  - Iterative data reduction process, often ending in manageable dataset that can fit in a laptop, which is often analyzed interactively
- From the laptop to the local facility
  - Growing dataset sizes (15-20x) demand a change of paradigm, where high-performance local facilities are used for final analysis
  - Key objectives: ease of use, performance, scalability, sustainability
- Evolution of analysis software and paradigms
  - New centrally produced data formats of reduced size (*NanoAOD*, in CMS)
  - Use of modern programming interfaces and tools (declarative, columnar)
  - Enable trivial/implicit parallelization of the code
  - Growing use of Python libraries, Jupyter notebooks, Machine Learning, etc.

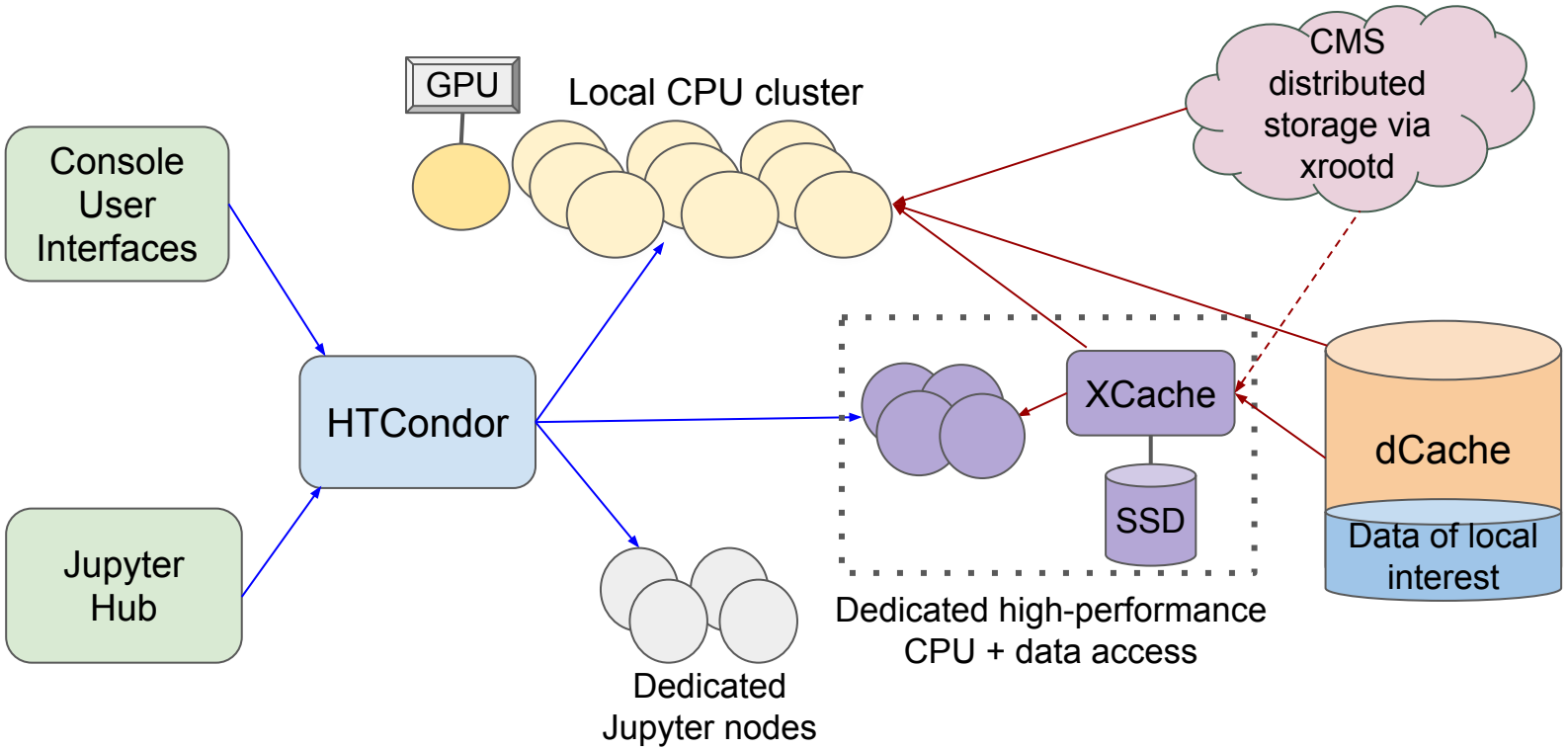
The Spanish analysis facility at CIEMAT

- Hardware
  - New high-performance SSD-equipped server for analysis
    - 180 TB SSD disk, 128 CPU cores, 2968 HS06
    - Additional server being purchased: 180 TB SSD, 172 CPU cores
  - Local batch CPU nodes
    - Few hundred CPU cores, co-located and managed with Tier-2 HTCondor pool
  - Dedicated disk area at the site's SE for pre-staging of relevant NanoAOD
    - Currently, ~1 PB
  - GPU(s) accessible by HTCondor jobs and Jupyter notebooks
    - One server with NVIDIA HGX, 4 GPUs A100
- Software
  - New NanoAOD/RDataFrame-based CIEMAT analysis framework
  - Intensive use of other technologies
    - *HTCondor* (resource mgmt.), *Rucio* (data synchronization), *xrootd and NFS* (versatile data access), *Kerberos* (authentication)

The Spanish analysis facility at CIEMAT

- Services
  - *Local analysis-only* XCache, using the SSD area
  - JupyterHub deployment as interface for CMS and other local communities
  - *Future services*
    - *NanoAOD augmentation: add information required for local analysis when not included in standard NanoAOD (efficiencies, trigger info, etc.)*
    - *Dask job submission from Jupyter to HTCondor*
- People
  - Tighter collaboration between CIEMAT computing and analysis groups
    - Support, consultancy and collaborative planning
  - Main examples: gradual improvements in the analysis framework, continuous data pre-staging, future NanoAOD augmentation service

Architecture



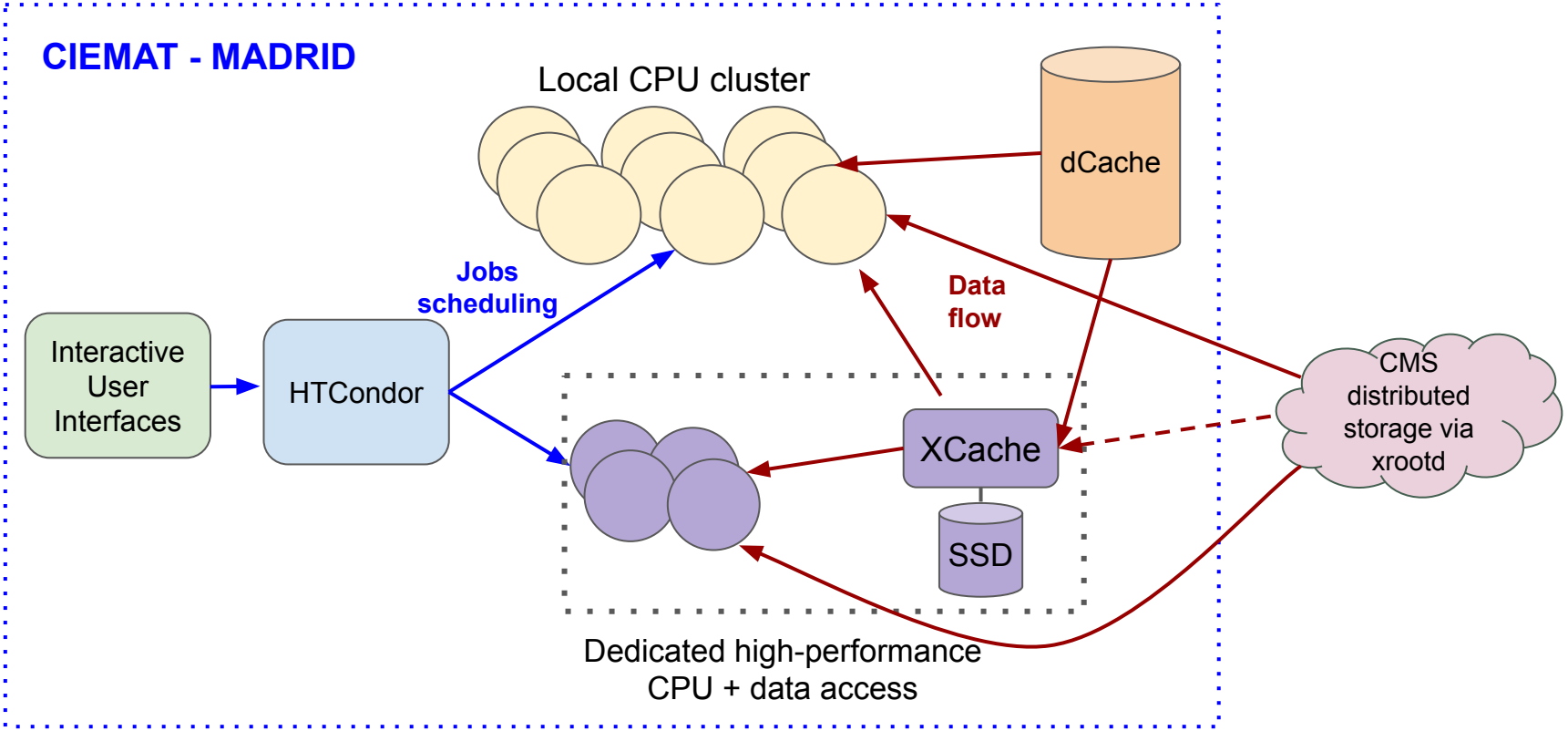
The CIEMAT analysis framework (link)

- Motivated by a single use case
  - Repeating an $HH \rightarrow bb\tau\tau$ analysis using NanoAOD
  - Grown to be general and flexible, for any CMS analysis
  - Contributions from both the CIEMAT analysis and computing groups
- Objectives: user-friendly, fast, general
- In line with general trends in CMS analysis tools
  - User code mostly in Python (although a lot of C++ code is used)
  - Designed for NanoAOD (*or flat tuples*)
  - ROOT's RDataFrame at its core
- Building on Luigi Analysis Framework (*law*) (link)
  - Tasks organization, batch and file access support, command line interface…
- Built-in parallelization: job splitting or RDF multi-threading

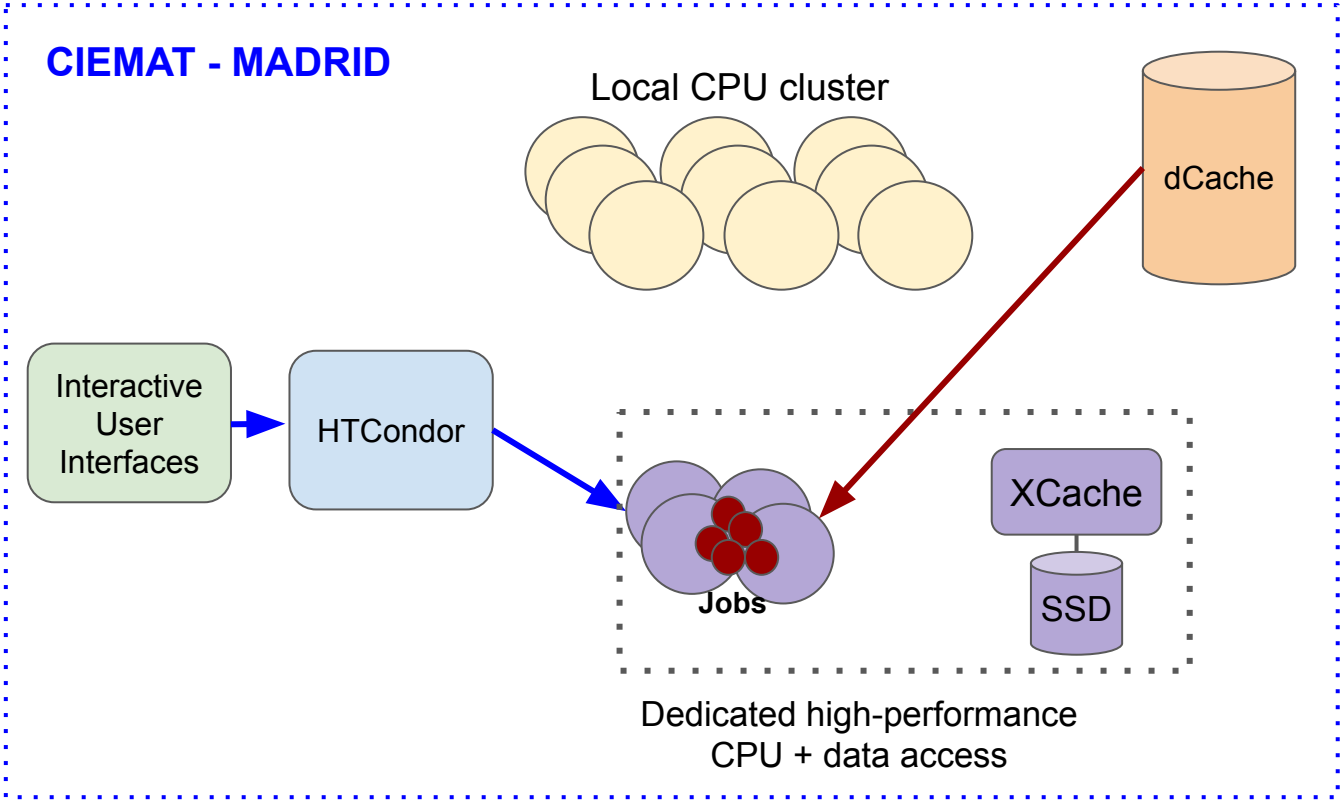
Performance measurements

- What we run
  - Processing stage (bulk of CPU time) of the `HH → bbττ` analysis
  - Limited to 3 datasets (251 files, 393 GB, 450 k events, ~20% of whole analysis), to avoid scheduling queues in our tests
- Considered several cases
  - Notation in these slides: `<Run at>_<Reading from>`
  - Cases described in the next slides
- Results normalized relative to best case
  - Workflow turn-around time measured (including job scheduling, etc.)
  - Best case (`AF_XCache`) took on average 44.6 minutes (using ~85 cores), i.e. ~1.75 K events/s/core, ~3.5 K events/s/core without scheduling and initialization

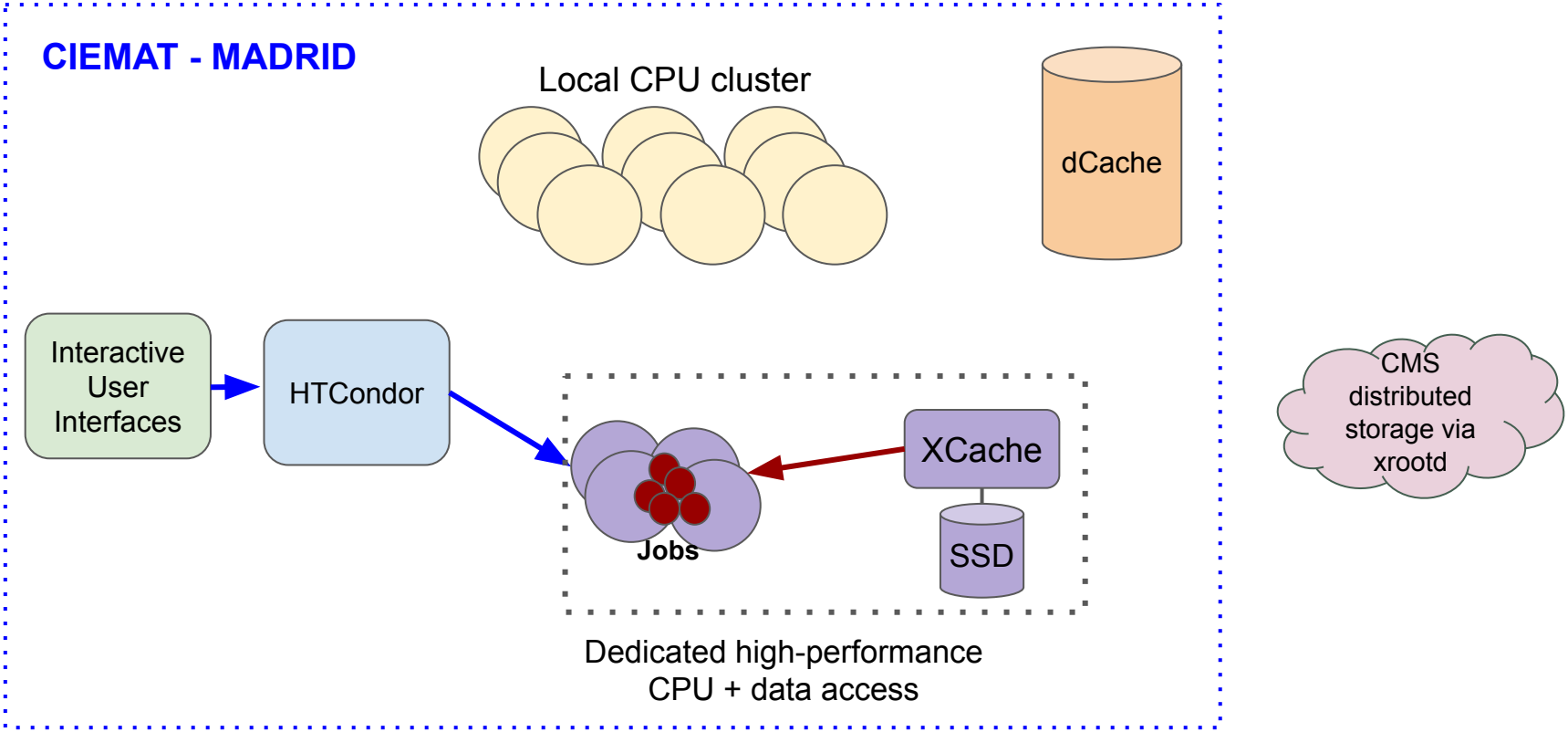
Test cases: general configuration



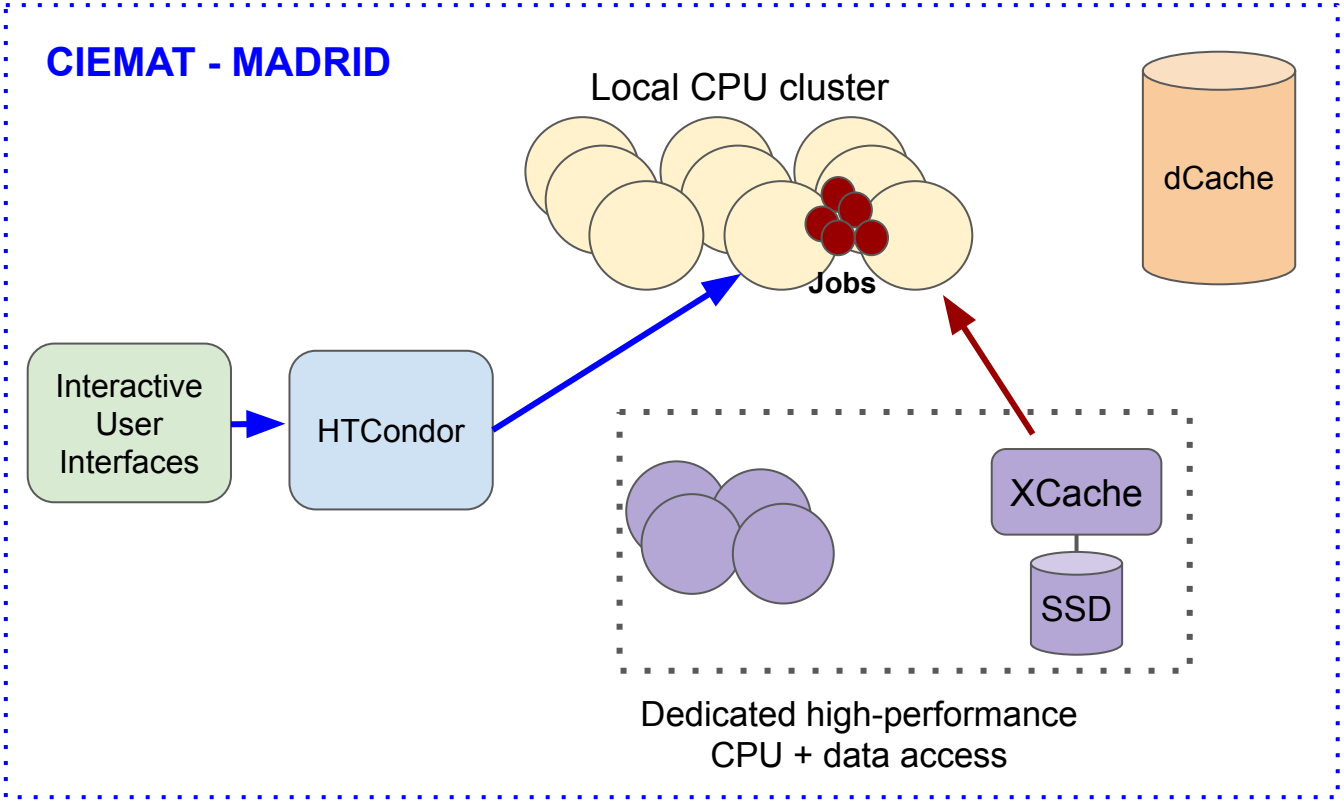
AF_SE: run at AF, read from SE



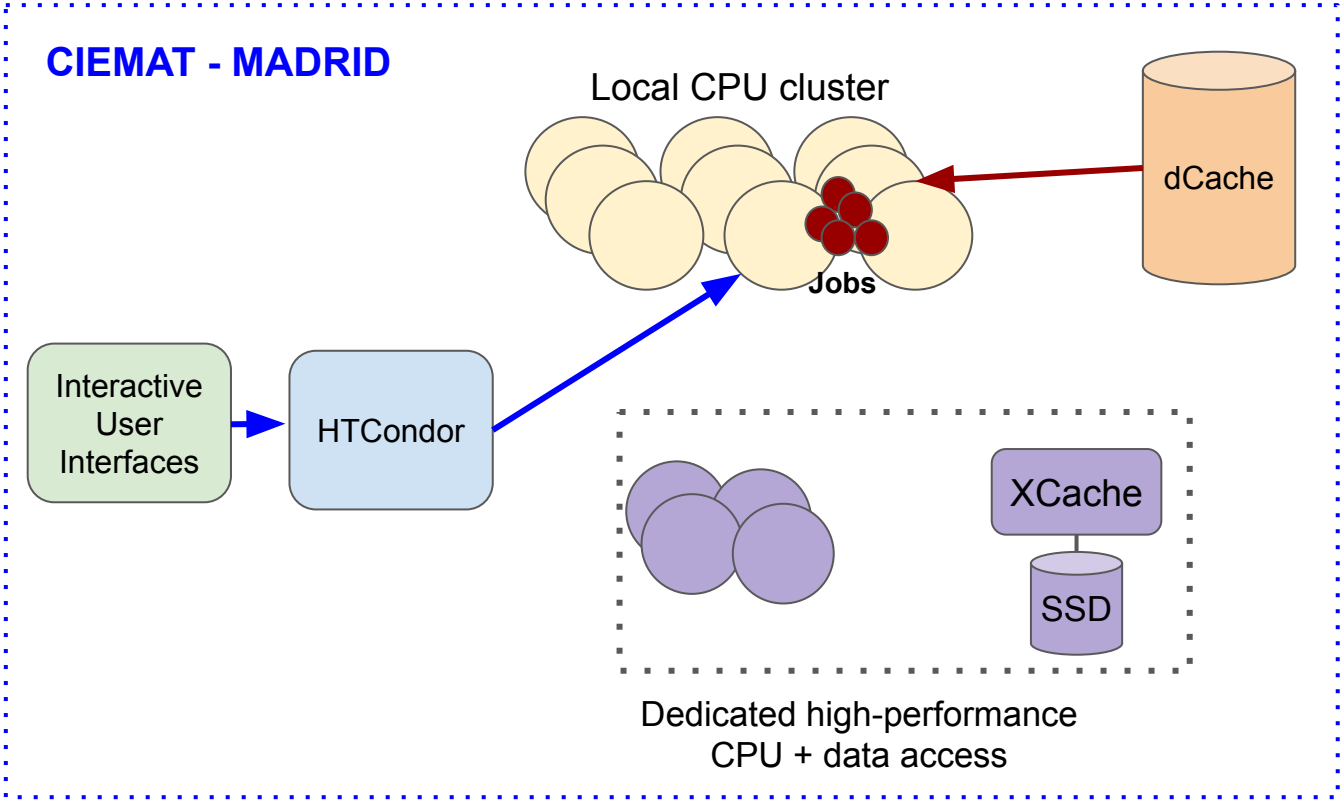
AF_XCache: run at AF, read from XCache



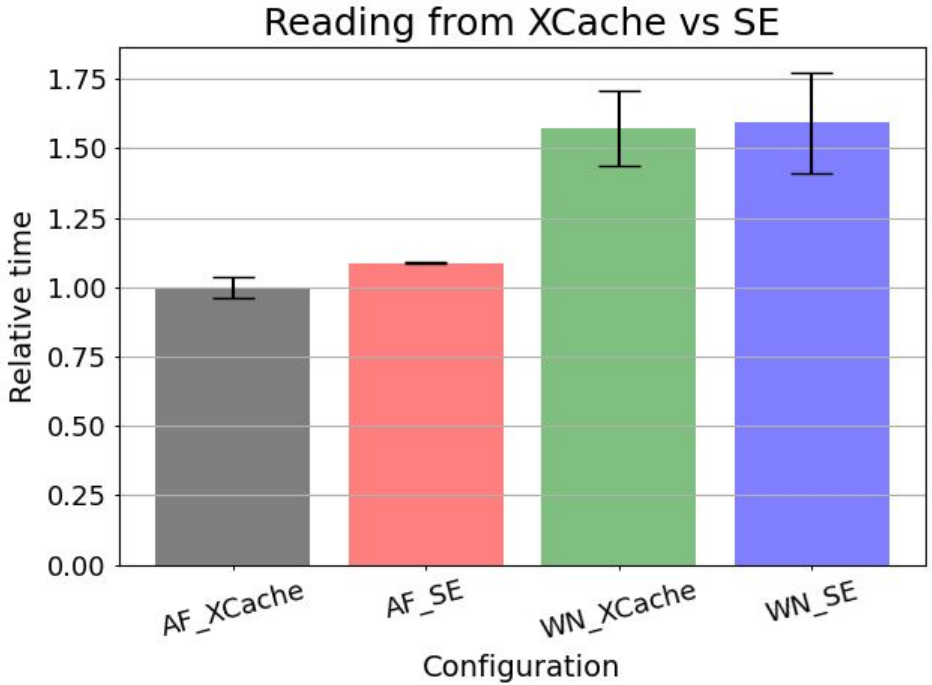
WN_XCache: run at WN, read from XCache



WN_SE: run at common WN, read from SE

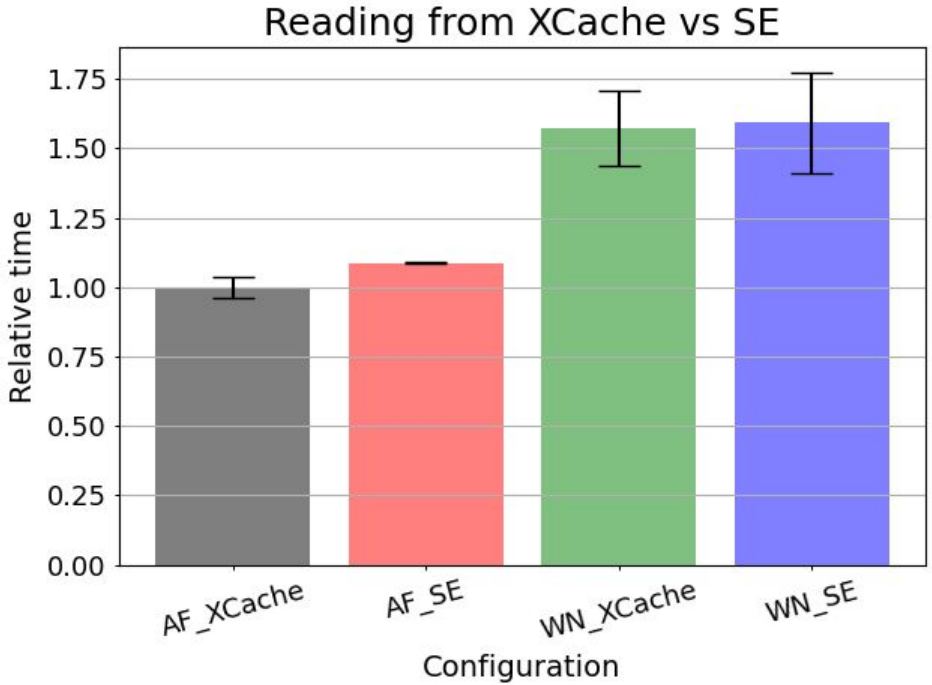


Test results: AF vs WN, XCache vs SE



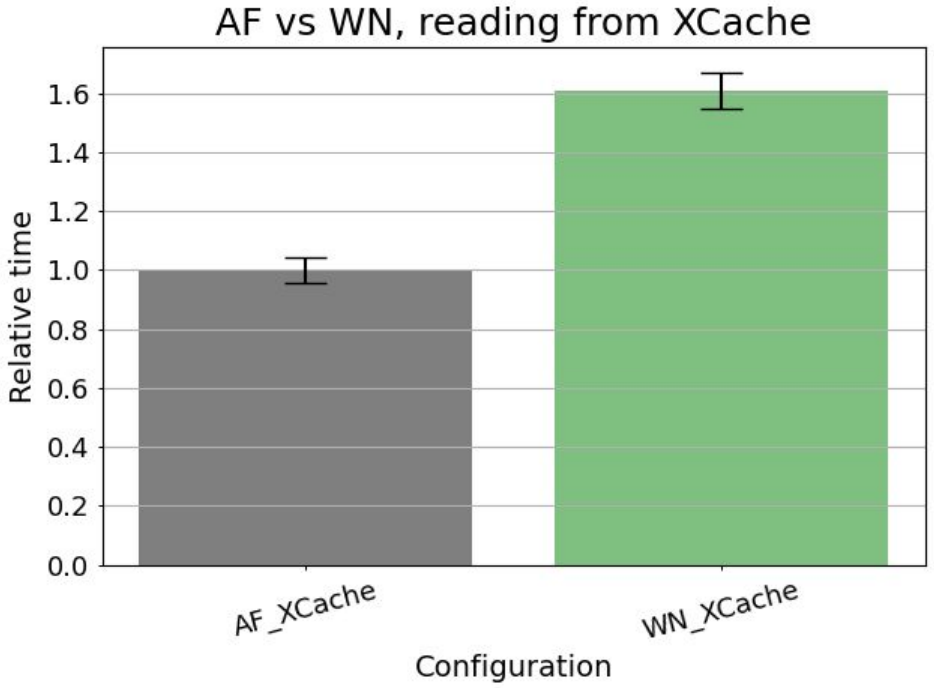
- Execution in new analysis server significantly faster than in common WN

Test results: AF vs WN, XCache vs SE



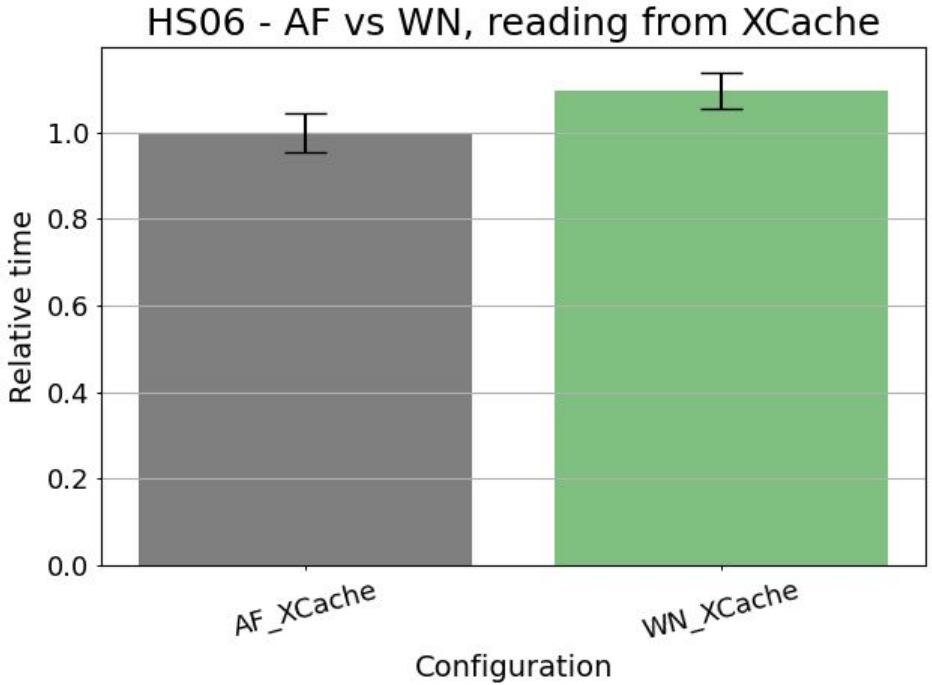
- Execution in new analysis server significantly faster than in common WN
- Reading from XCache or SE yields similar results
  - SE offers multiple disk servers, while XCache profits from SSD disks
  - Local network does not seem to be a bottleneck

Detail: Read from cache, AF vs WN



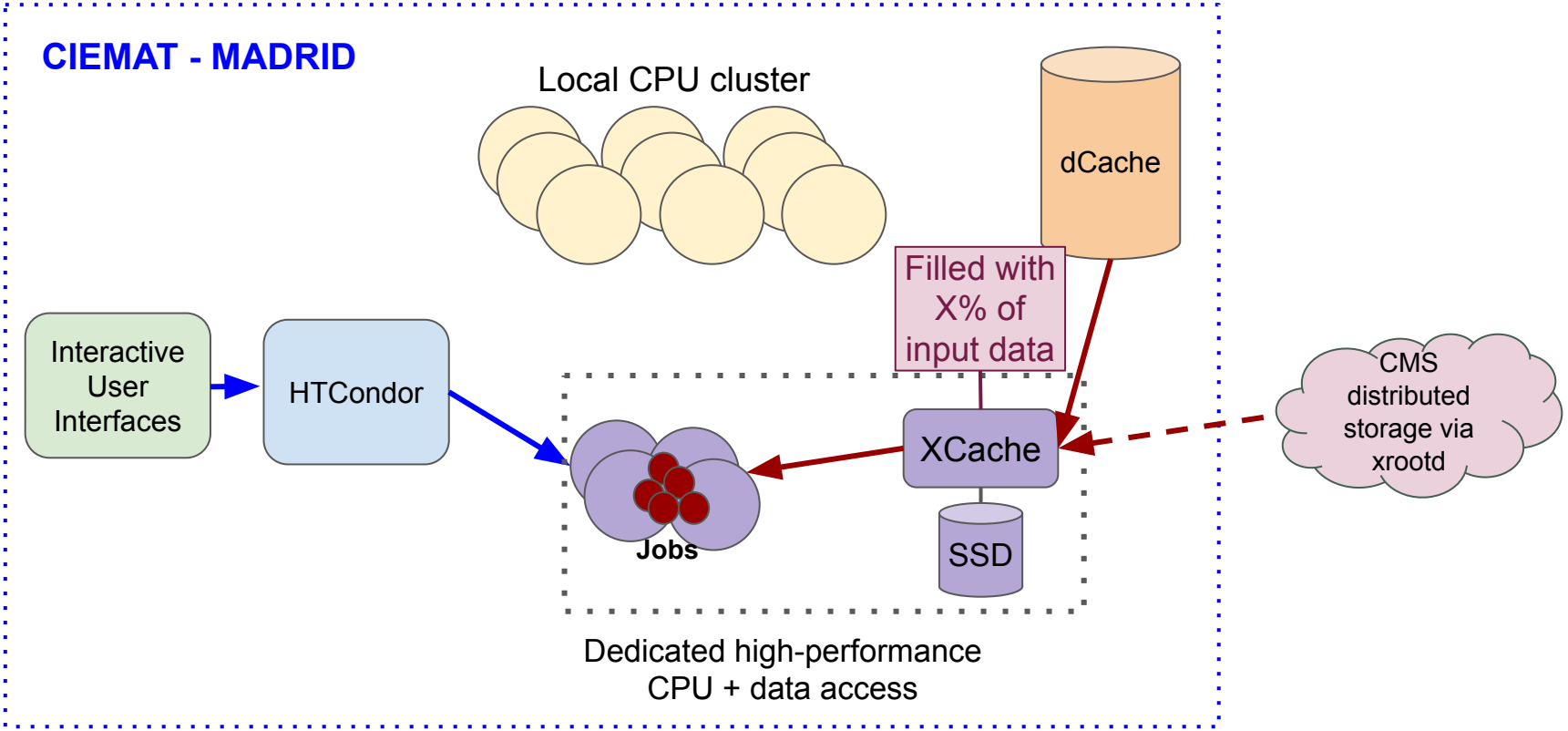
- Better performance for AF execution due to faster CPU and local access to data

HS06 normalization (Read from cache, AF vs WN)

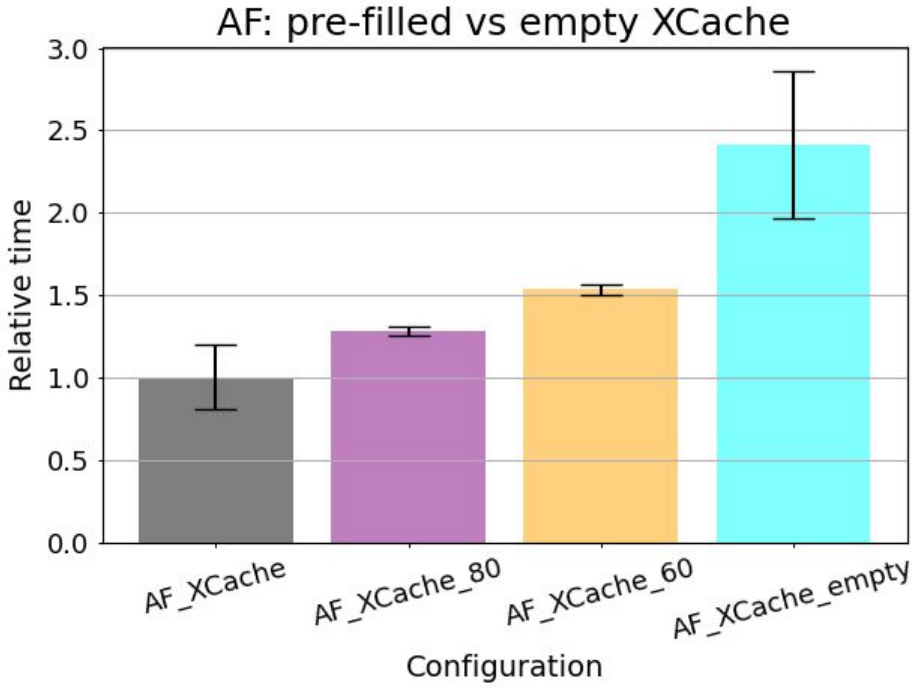


- HS06 normalization shows that XCache access through internal network induces very little penalty

AF_XCache_X: run at AF, partially filled XCache

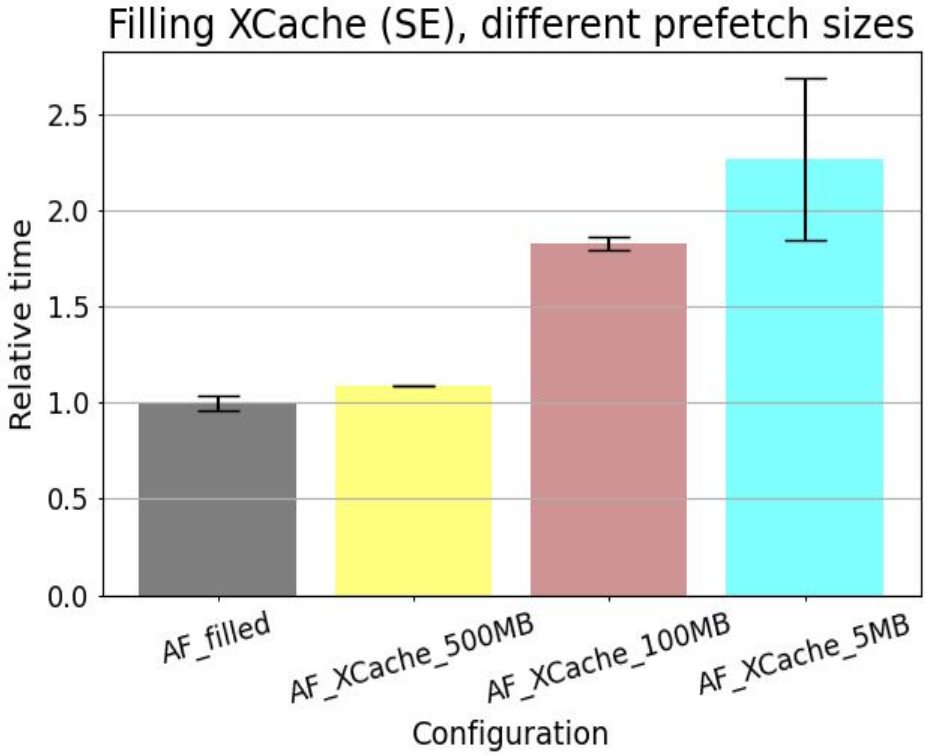


Run at AF, filled vs (partially) empty cache



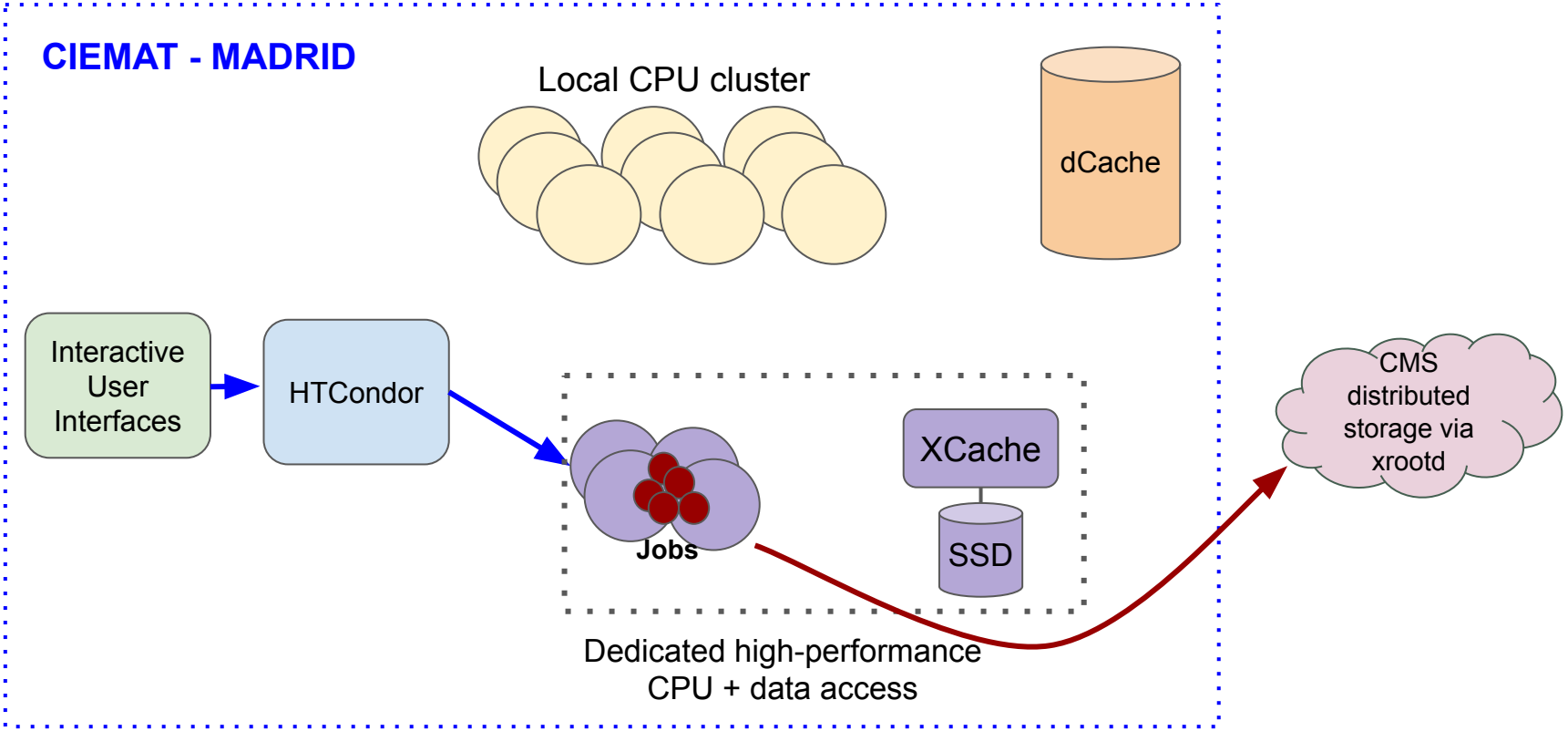
- Cases: 100%,80%, 60%, 0% of the files in the cache
  - Most files come from local SE
  - Sporadic reads from PIC
    - Could be optimized
- Significant performance loss when files retrieved from SE
  - Still under investigation
    - Multiple WN → XCache → SE interactions per file might be the problem

Filling the XCache with different *prefetch* sizes

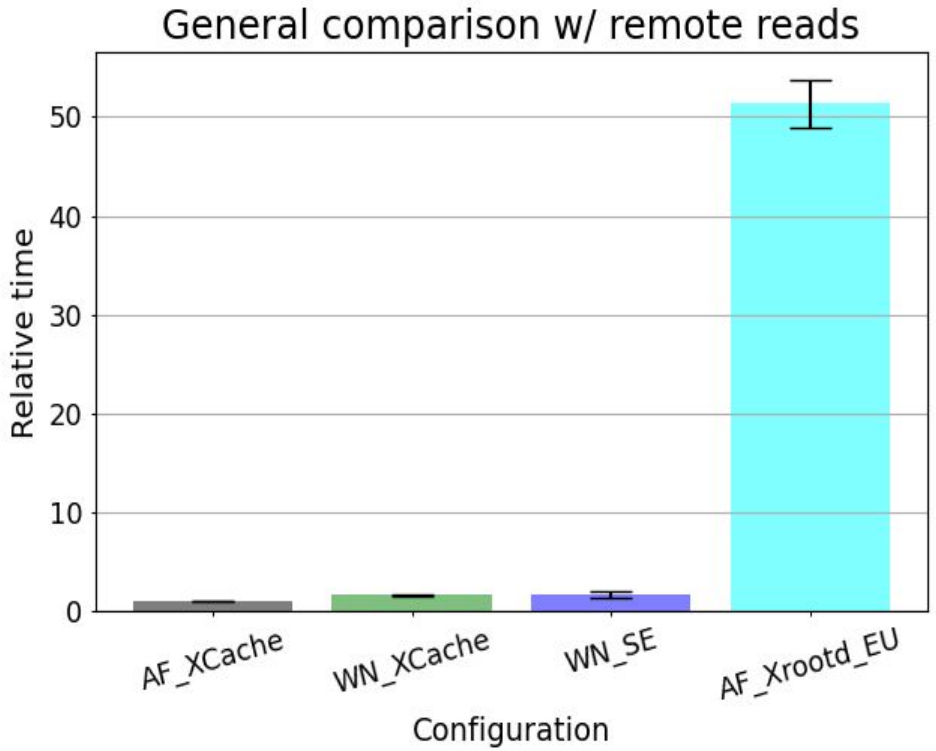


- Increasing the read-ahead (`prefetch` setting in XCache) improves performance when filling the XCache
  - Original value (default): `prefetch = 5 MB`
  - Tested with `100 MB` and `500 MB`
- Hides latency and non-optimal data access patterns

AF_Xrootd: run at AF, read remotely

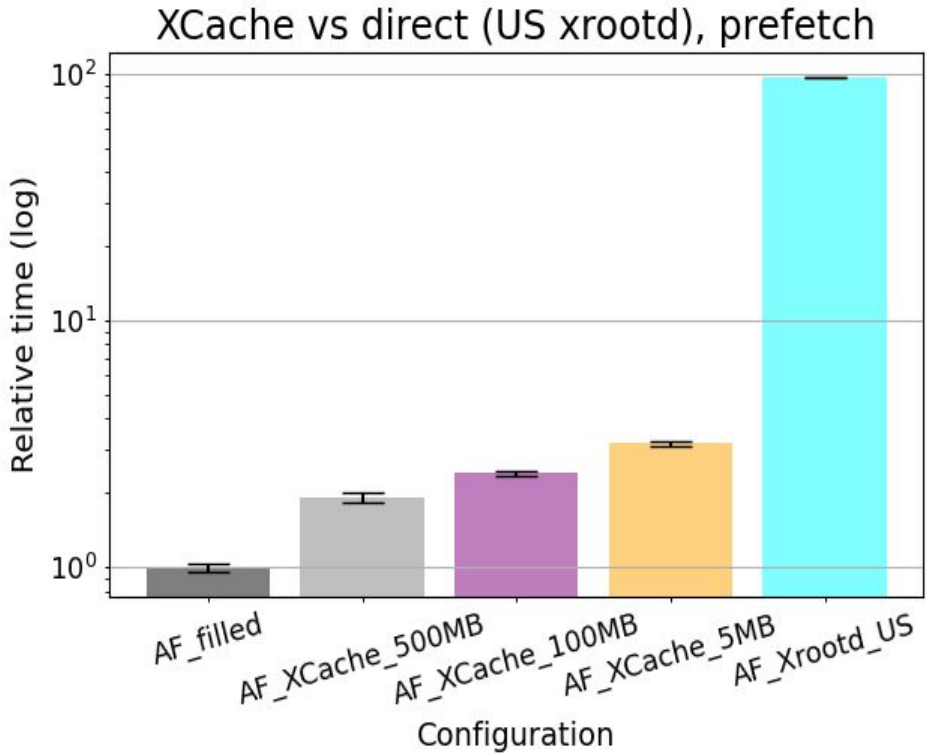


Test results: Remote access



- Remote reads significantly increases full workload execution time
  - Extremely inefficient analysis with high latency
- Results of the test under careful scrutiny
  - Many end-to-end interactions
  - Something wrong in configuration?

XCache for remote access, different *prefetch* sizes



- Direct access to remote data (with no cache) results in a x100 execution time
  - Compared to reading from local data
- Interposing the XCache achieves much better results
  - Same remote data source
  - Benefits from read-ahead
  - Even with default pre-fetch value (<1% of file size)
- Performance can be further improved by tuning the XCache `prefetch` setting

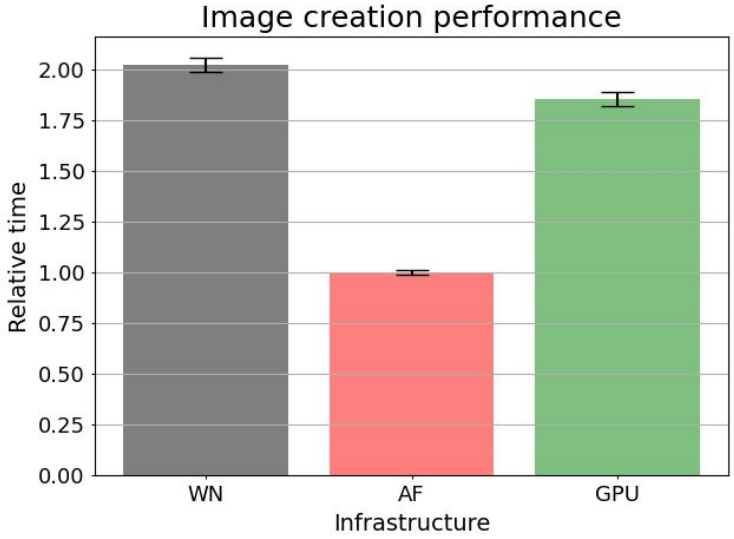
Support for other scientific communities

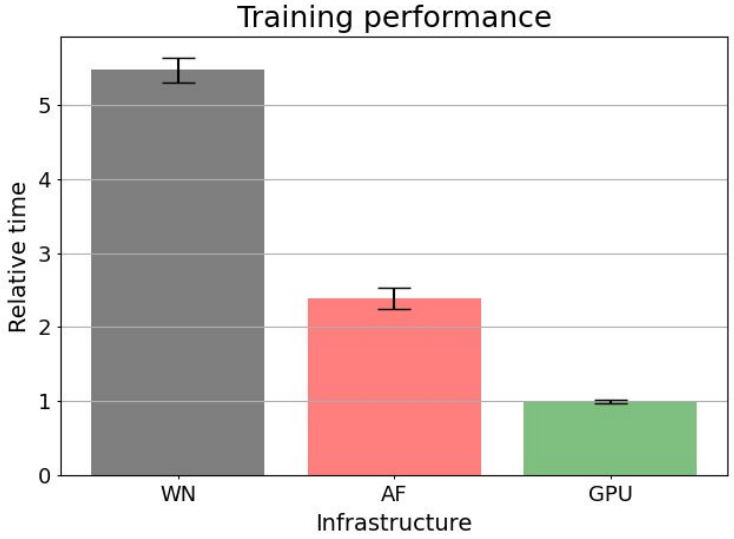
- New infrastructure is a friendly environment for non-WLCG scientific groups
  - Jupyter interface found convenient by many groups
  - NFS makes massive storage (dCache) accessible to almost everyone
  - HTCondor allows to ask for required resources (GPU, high performance SSD nodes, etc.), both from command line or through Jupyter
- First tests with several Machine Learning studies on-going/planned
  - Time series classification performance tests (next slides)
  - Classification of gravitational wave images from binary black holes merge
    - Tests on-going
  - Others: pollution forecast, dark matter detection in liquid argon, oncology studies

Machine learning performance tests

- Time series classifier
  - 40 series generated with 3 different functions, then converted to wavelet images
  - Images are classified with a neural network (½ million params)
  - Test designed to stress disk and CPU/GPU mathematical processing
- Two stages measured
  - Image creation
  - Training of the classifier neural network
- Three different infrastructure evaluated
  - **`WN`**: Jupyter notebook run on a common local cluster worker node
  - **`AF`**: Jupyter notebook run on new dedicated WN of the AF
  - **`GPU`**: Jupyter notebook run on a common WN with GPU access

Machine learning tests results





- New AF perform significantly better for image creation
- GPU is faster for neural network training
  - Important for many current lines of research (including HEP)

Conclusions and outlook

- New hardware and services for CMS analysis at CIEMAT already deployed
  - Progressively adopted for production activities
  - Performance studies show promising results for the new dedicated infrastructure applied to fast-turnaround analysis
- CMS-CIEMAT adopting recommended CMS practices for analysis
  - Reduced data format, data caching, new software tools and paradigms
- Support for other communities (non-HEP)
  - Proof of principle and some tests
  - New infrastructure expected to be very relevant for projected oncology studies
- Continuous enhancement process
  - Additional, more powerful, SSD-equipped machine already being procured
  - New NanoAOD augmentation and Dask services to be defined and implemented

Thank you!

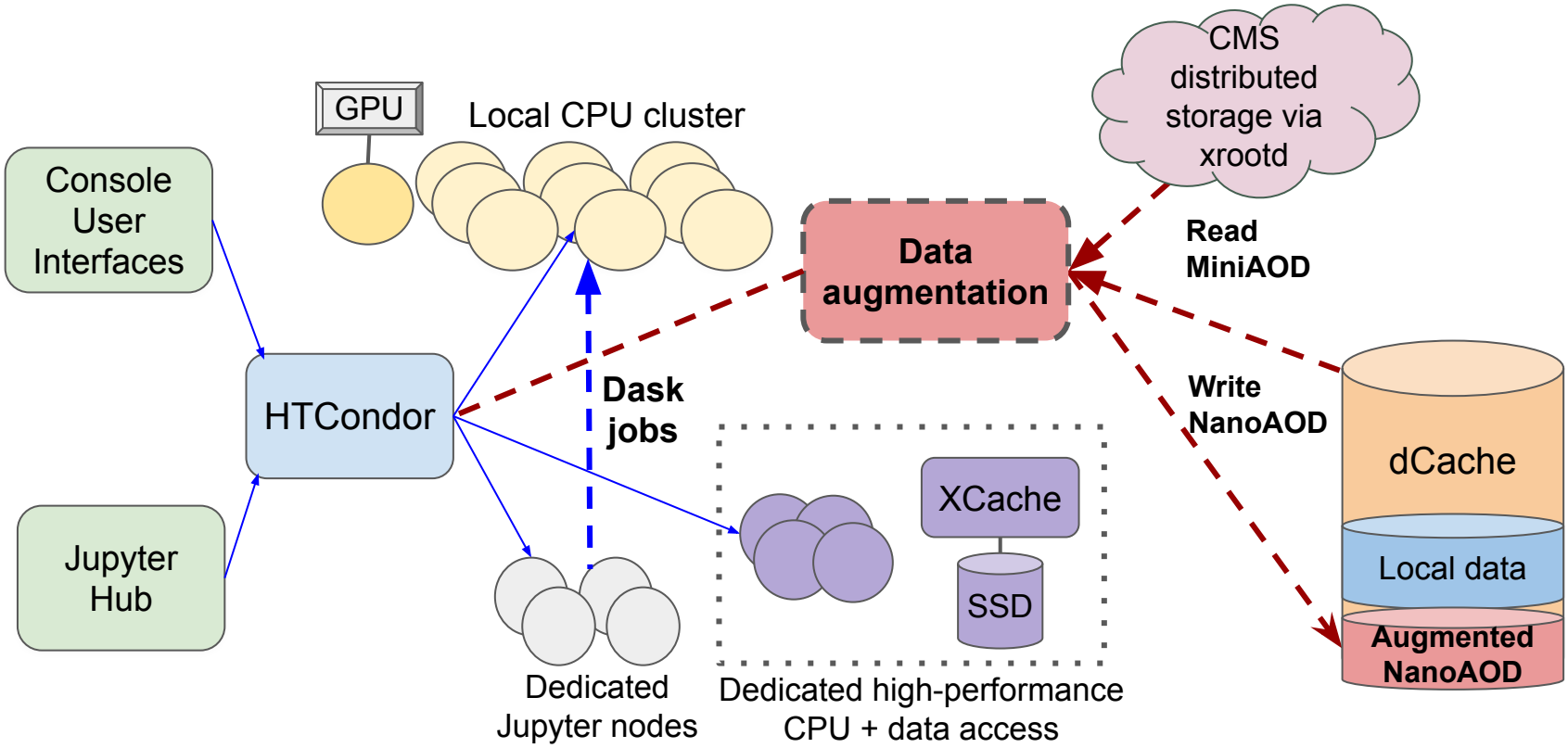
Projects PID2019-110942RB-C21 and
PID2020-113807RA-I00 funded by:

Back-up Slides

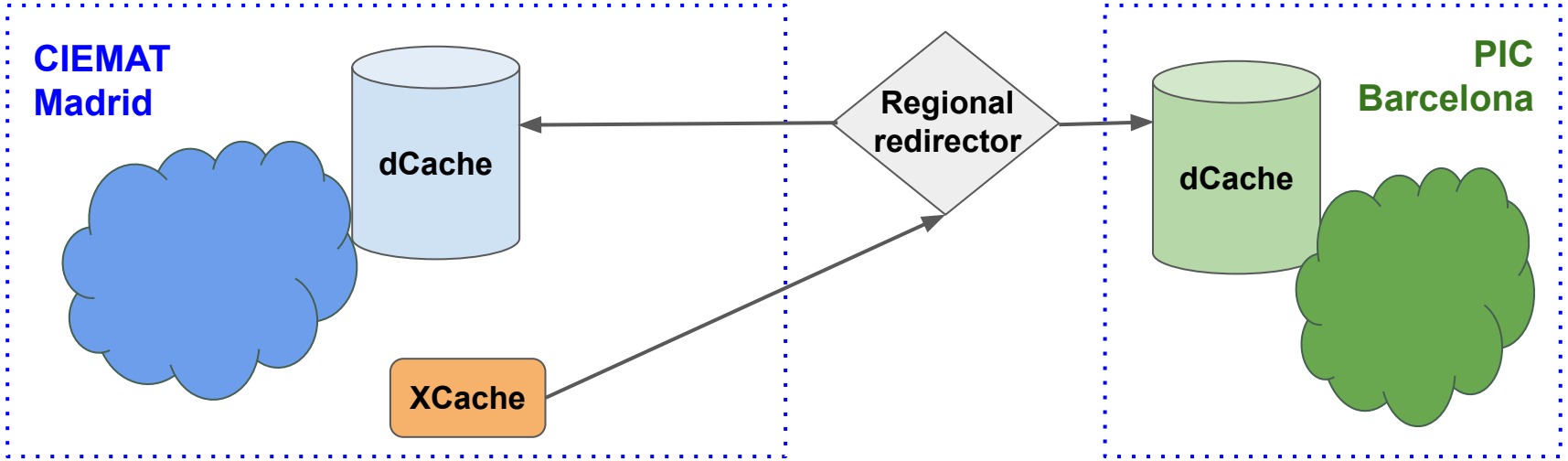
Abstract

The increasingly larger data volumes that the LHC experiments will accumulate in the coming years, especially in the High-Luminosity LHC era, call for a paradigm shift in the way experimental datasets are accessed and analyzed. The current model, based on data reduction on the Grid infrastructure, followed by interactive data analysis of manageable size samples on the physicists' individual computers, will be superseded by the adoption of Analysis Facilities. This rapidly evolving concept is converging to include dedicated hardware infrastructures and computing services optimized for the effective analysis of large HEP data samples. This contribution will describe the actual implementation of this new analysis facility model at the CIEMAT institute, in Spain, to support the local CMS experiment community. Our presentation will report on the deployment of dedicated highly-performant hardware, the operation of data staging and caching services, that ensure prompt and efficient access to CMS physics analysis datasets, and the integration and optimization of a custom analysis framework, based on ROOT's RDataFrame and CMS NanoAOD format. Finally, performance results obtained by benchmarking the deployed infrastructure and software against a full CMS reference analysis workflow will be presented.

Architecture: planned services

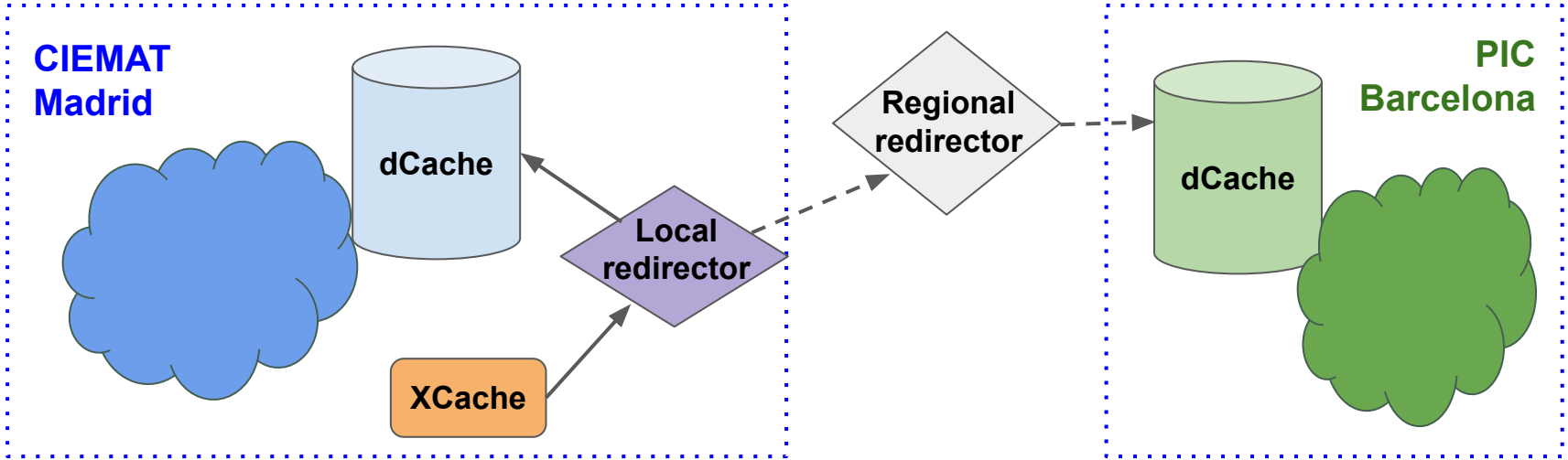


Current xrootd configuration for cache misses

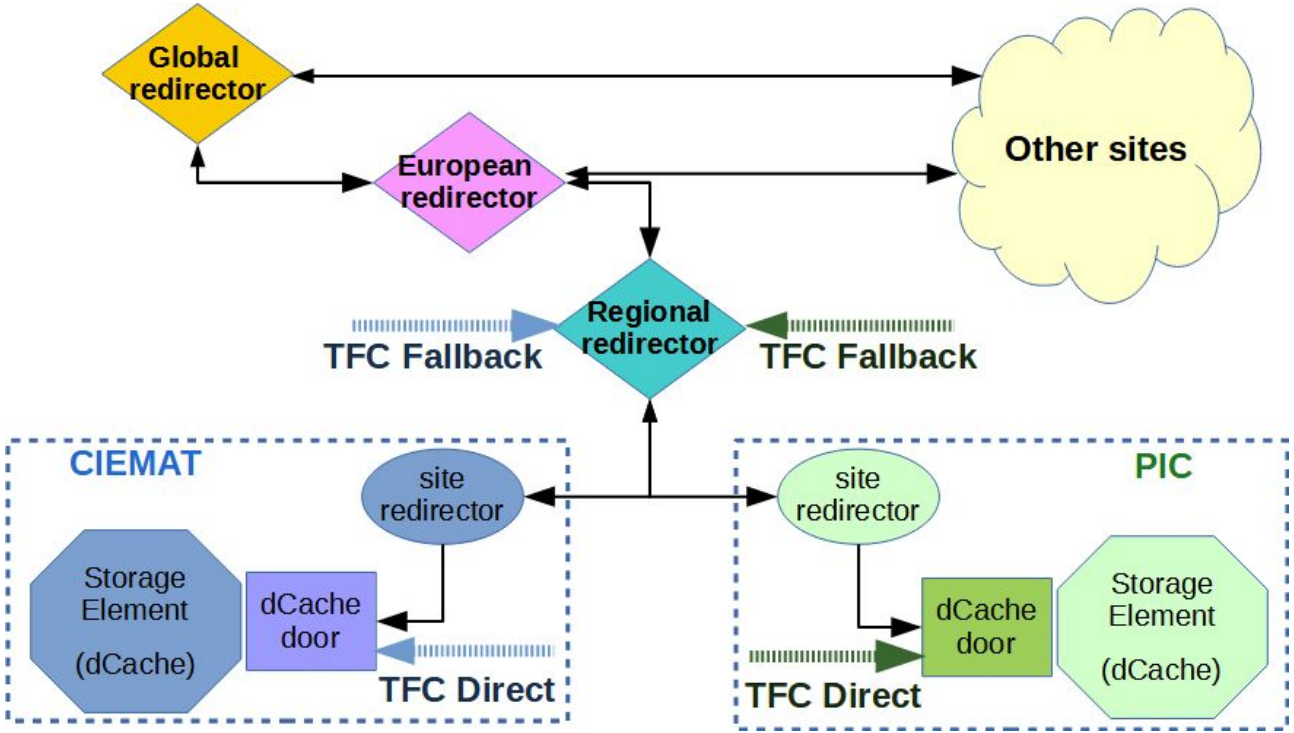


Lightweight site federation for CMS support, CMS Collaboration, C. Acosta-Silva, A.Delgado Peris, J. Flix, et al., CHEP 2019, Adelaide, Australia. Published in: EPJ Web Conf. 245 (2020), 03013

Possible optimized configuration



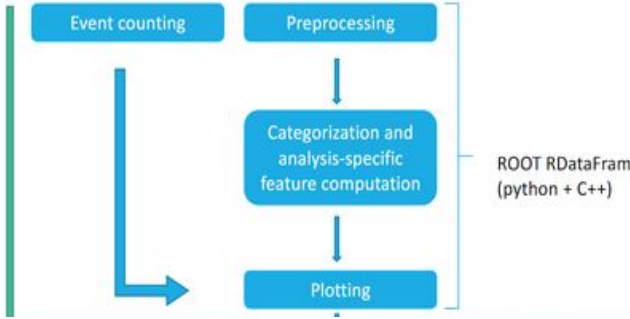
Spanish xrootd federation



The CIEMAT analysis framework: details

- Supports different analysis, uses central code (Physics Object Groups) if possible
- Run a complete analysis:
  - Event counting and weighing
  - Pre-processing: filtering, variable computation, correction factors
  - Categorization: analysis-specific filtering/selection
  - Histogram production
- Easy re-use and extension (inheritance) of existing code
- Support for local/HTCondor execution and local/xrootd input data files
- Built-in parallelization: different tasks (jobs) per file or number of events, and/or RDataFrame multi-threading
- Version control part of standard workflow
- Tested at CERN and CIEMAT
  - A few infrastructure customizations included

law



The CIEMAT analysis framework: usage

- Used by CMS-CIEMAT at…
  - HH → bbττ analysis: Run 2 verification, Run 3 new analysis
  - Tasks for the Muon POG
  - Wprime → muon+neutrino analysis
  - Charged Higgs analysis
  - W → cs (charm strange) studies
- Other institutes have already expressed interest in the framework

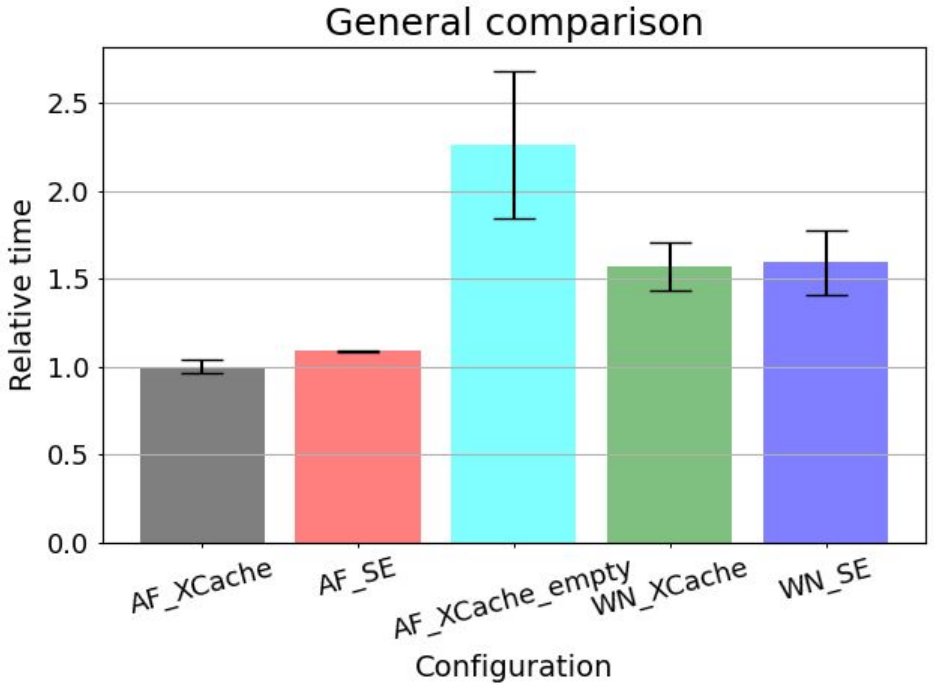
HH→bbττ analysis

- Study of two protons collision going to 2 bottom quarks and 2 tau leptons
- Many filters applied to RDataFrame
  - General (CMS) filters, such as luminosity mask, MC reweighing, MET
  - Analysis-specific filters
- Some of the filters are quite heavy in processing time
  - Especially the so-called SVFit algorithm, which calculates the Higgs mass to ττ
- Some of the filters use Machine Learning
  - Pre-trained neural networks (only inference is run during the analysis)
  - Not especially heavy processing required
- Both data and MC are consumed in the analysis (and our performance tests)

HW used for the tests

- New dedicated server
  - 2x AMD EPYC 7763 64-Core Processor, 2450 MHz
  - 1 TB RAM
  - 180 TB SSD disk
  - 2968 HS06 (144 job slots → 20.61 HS06/job)
- WNs for HH→bbττ test
  - Heterogeneous
  - Average of 14.17 HS06/job
- WN for ML tests
  - 12 CPU cores, 123.55 HS06
  - 10.30 HS06/job

Test results: General comparison



- Execution in the new dedicated analysis machine significantly faster than other configurations
  - Except when the cache must be populated
- Details follow