A Pilot Analysis Facility at CERN, Architecture, Implementation and First Evaluation

A. Sciabà, B. Jones, D. Castro, E. Tejedor, E. García, G. Guerrieri, M. Schulz

Kraków (Poland) – October 23, 2024

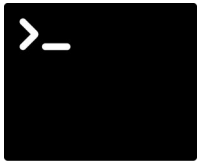
Context and motivation

- A working group was established at CERN to:
  - Design and implement a pilot of an Analysis Facility
  - Obtain feedback from early testers
  - Validate real demand before investing more effort
- Pilot designed to augment the current offering with:
  - Interactive computing on big datasets, with analysis built on frameworks like RDataFrame and coffea
  - Possibility to scale out
- Perception of demand from:
  - Forums: HSF Analysis Facilities WG, WLCG Pre-CHEP Workshop 2023
  - Users: first request for coffea + LxBatch in 2021
- Several other AF initiatives have been setup within the LHC community

CERN's Analysis Facility

- CERN provides infrastructure and services for analysis since a long time

LxPlus



LxBatch

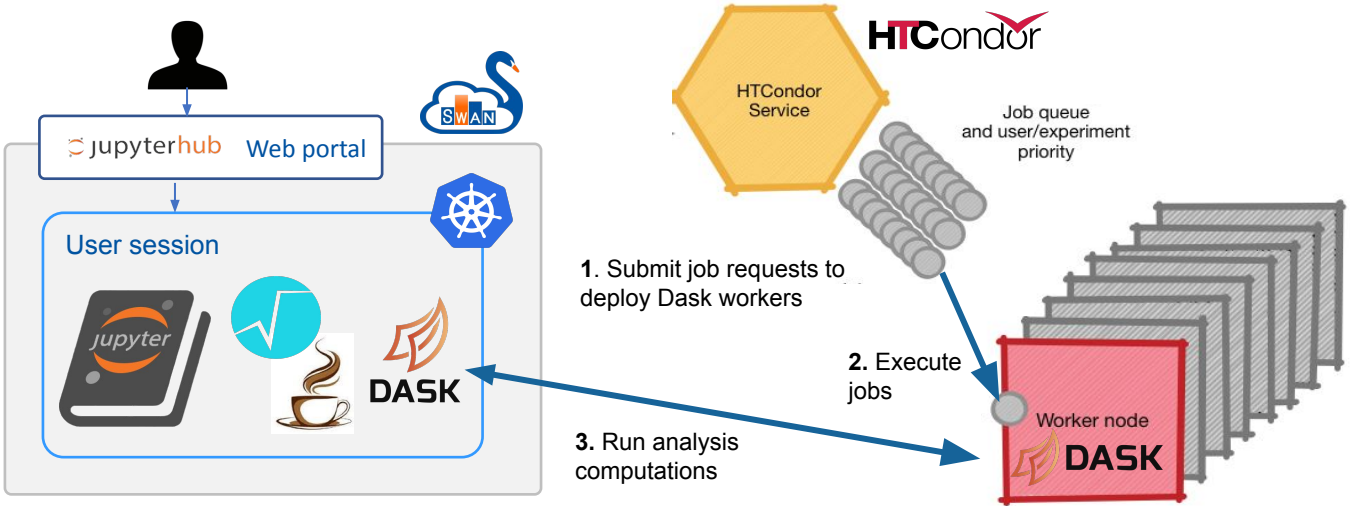
HTCondor

reana

Kubeflow

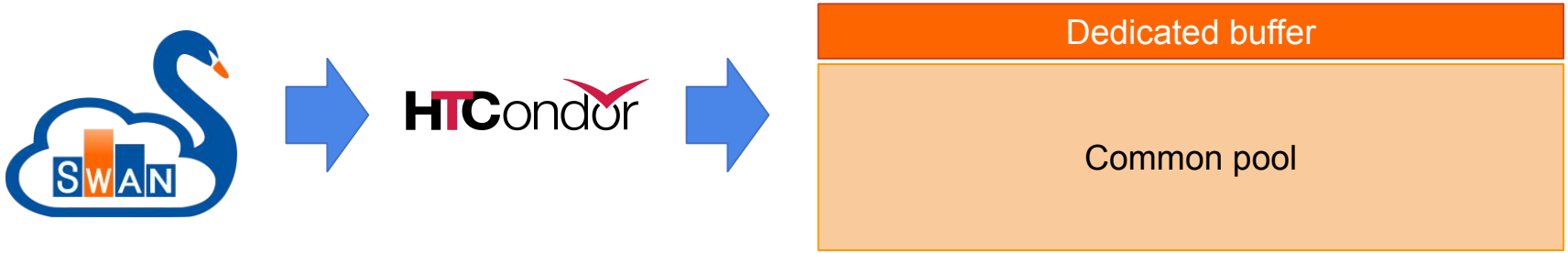
The pilot

- Focus on **scale out of interactive analysis**
  - On **already existing** CERN Batch system resources
  - Via RDataFrame / coffea + Dask



Resource management for pilot's interactive jobs

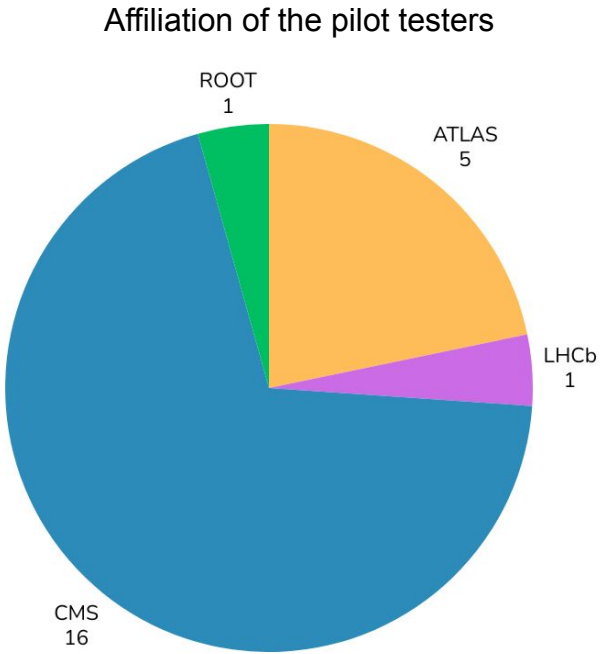
- Pilot's interactive jobs benefit from dedicated resources in the CERN HTCondor pool
  - Dedicated buffer of ~2k cores for quick allocation of resources (needed to feel interactive!)
  - Plus extra resources from common pool, start time **subject to experiment quotas**



Testing of the pilot

- A total of 23 people volunteered to test the pilot
- We have asked for their feedback via:
  - Dedicated meetings
  - A survey (6 answers so far): https://forms.gle/qSR7C8r7cA7bcXAVA

- If anyone is interested in testing the pilot, please let us know!
  - Accessible via SWAN at https://swan.cern.ch
  - Documentation on how to use it is available here



Feedback: Software provisioning

- **Pre-configured stacks** are appreciated (experiment stacks, LCG releases), but more **flexibility** is also requested
  - Custom software environments (e.g. conda)
  - Custom container images
- As an alternative, the pilot allows to install **additional packages on CERNBox/EOS**, to be used on top of an LCG release
  - Some errors accessing the mount on the batch side have been reported
  - Approach seen as a stop gap solution
- Flexibility is good, but the Analysis Facility must have a solid software foundation to ensure everything works (e.g. supported Dask version)

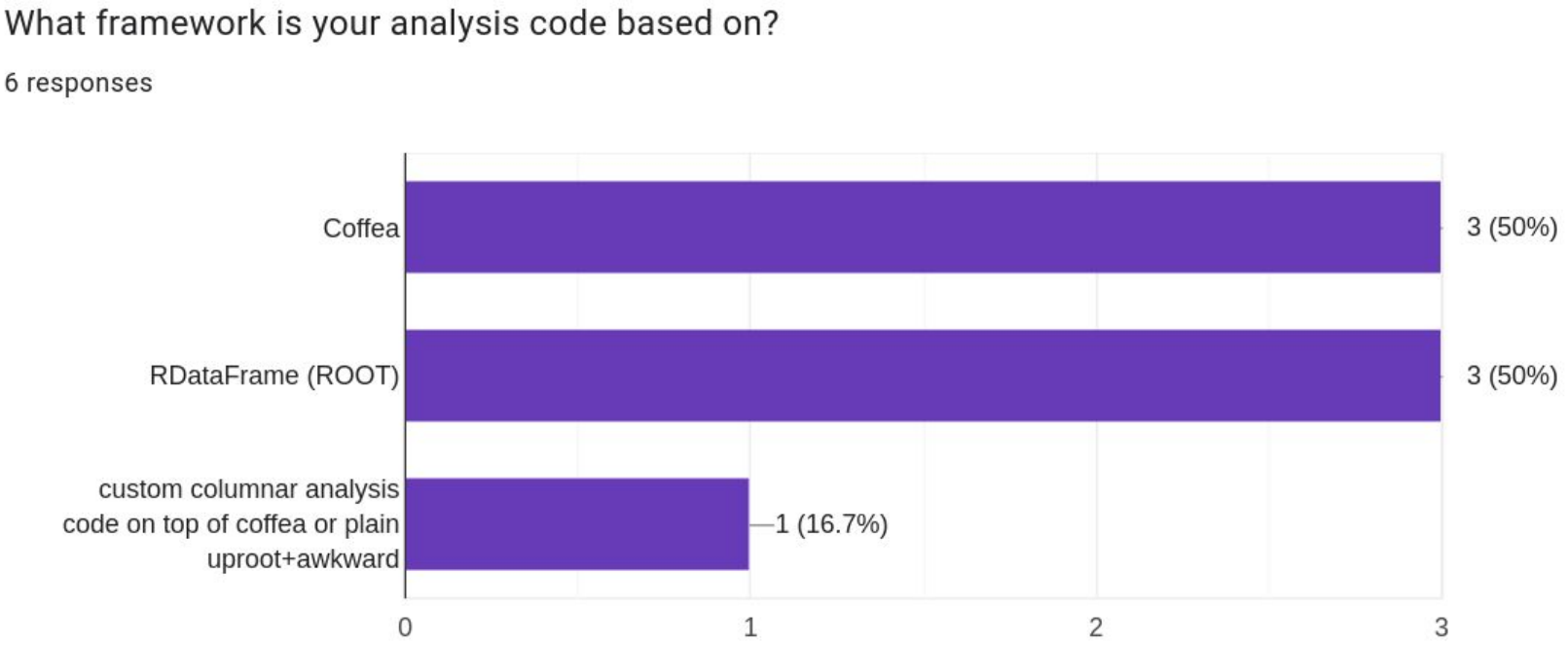
Feedback: Data access

- Both access to data on EOS and to **external datasets** (i.e. not at CERN) has been reported
  - External access usually done via remote URLs (e.g. XRootD), sometimes via Rucio copy
  - Added support for X.509 proxy generation (tokens could follow)
- Suggests that setting up a **cache** would be beneficial
  - Work in progress with storage group at CERN IT to have an XCache PoC

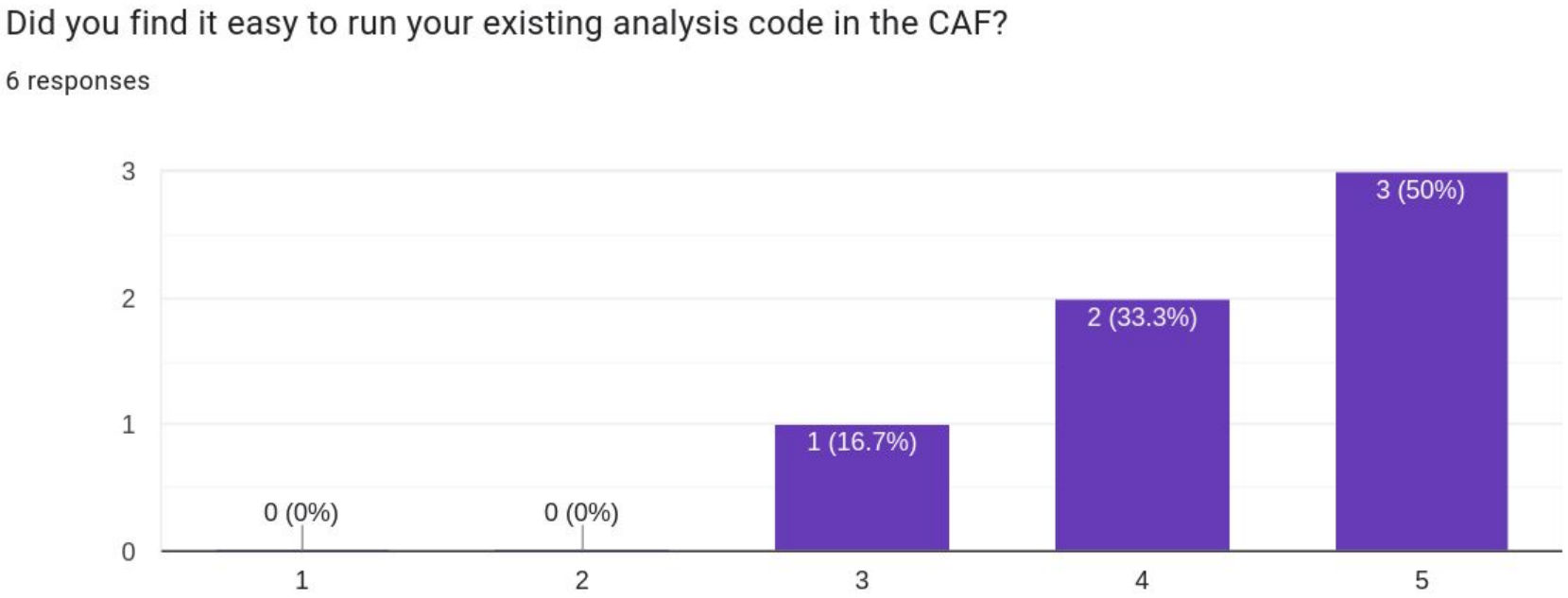
Feedback: User Interface

- The pilot offers the JupyterLab user interface, meaning:
  - Notebooks, terminal
  - Dask extension to **graphically reserve resources** (i.e. create a Dask cluster) and dashboard to **monitor** the progress of the distributed application
- Mostly satisfied with the interface, but reported some issues:
  - Creating / destroying a cluster can be slow (minutes) if many (a few 100s) Dask workers are requested, since their jobs are submitted / removed sequentially
  - The Dask monitoring dashboard can be sometimes laggy and freezing
- **Programmatic reservation of resources** and execution (from the terminal) is appreciated

Feedback: Running the analysis



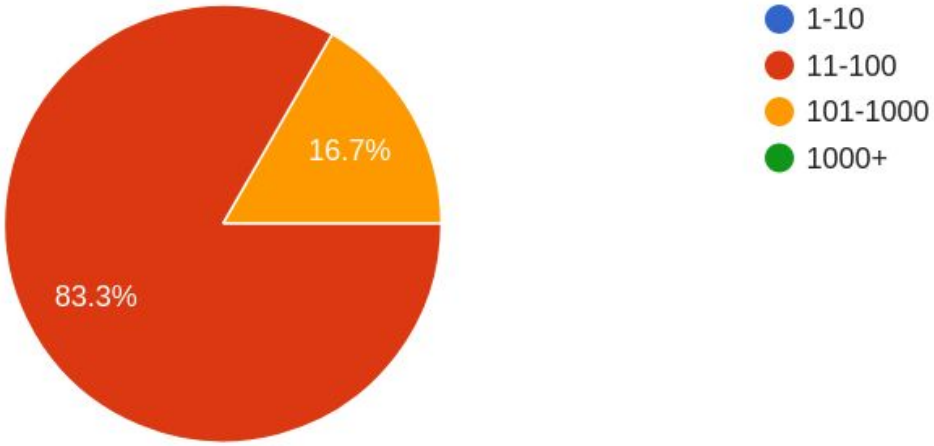
Feedback: Running the analysis



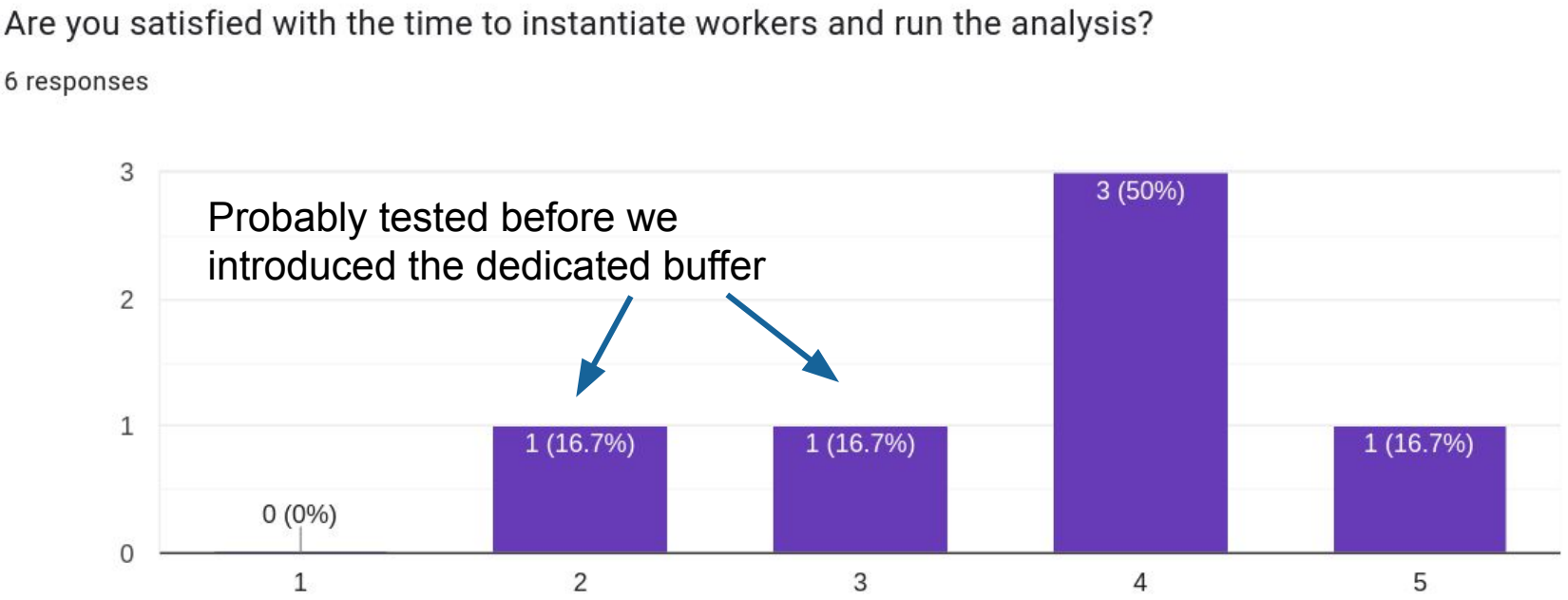
Feedback: Running the analysis

How many Dask workers do you usually request to run your code?

6 responses



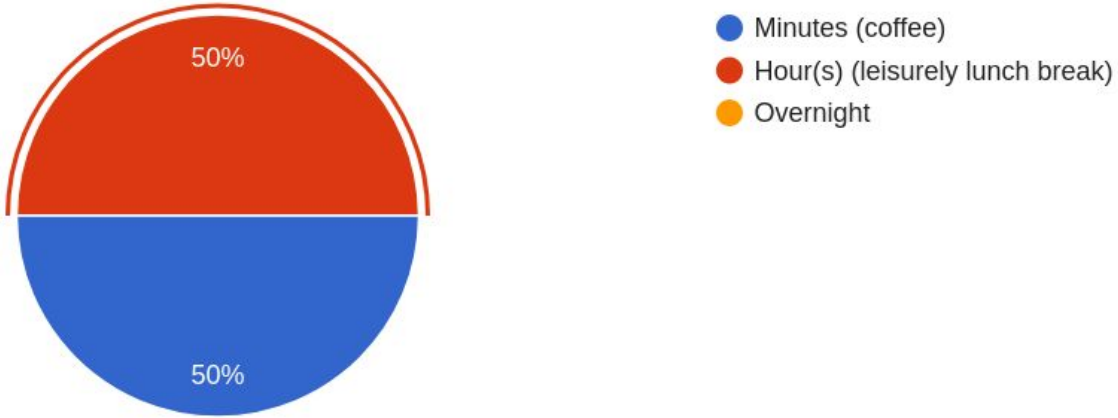
Feedback: Running the analysis



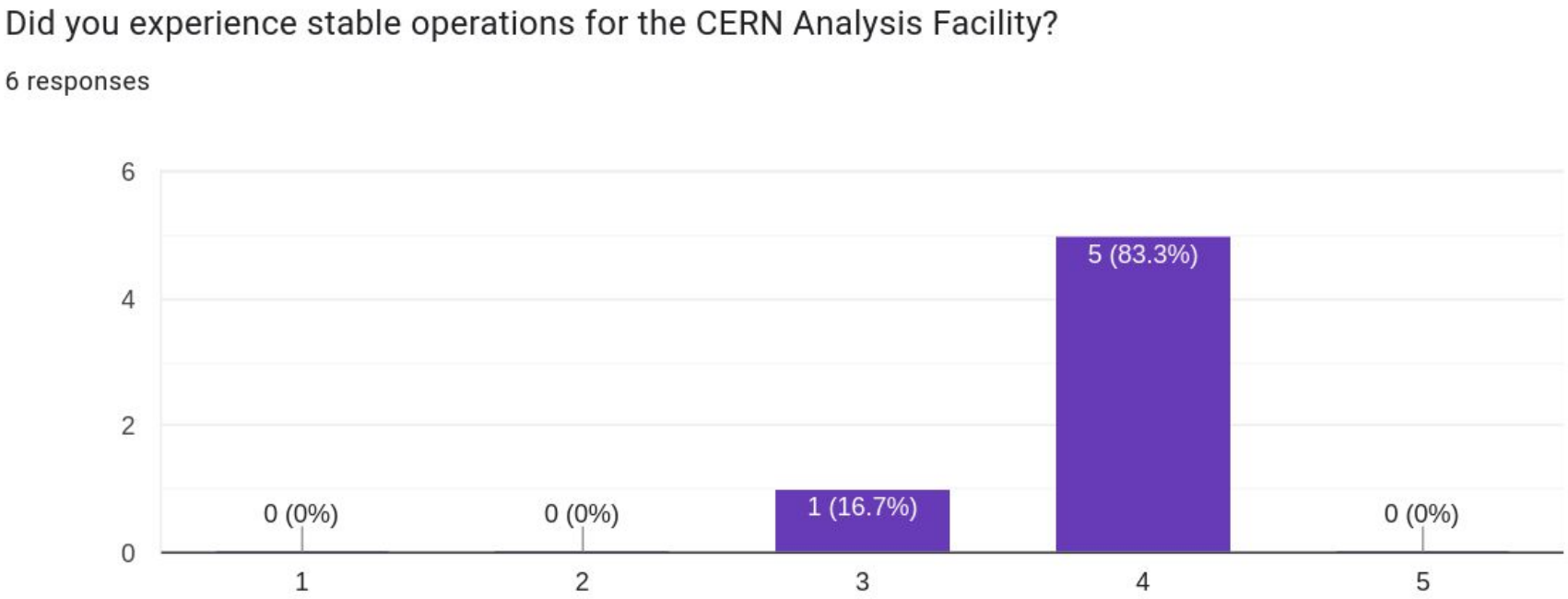
Feedback: Running the analysis

When running an analysis, what is typically the execution time?

6 responses



Feedback: Stability



Feedback: What do you use the AF for?

- Run my analysis at scale using a batch system - interactively postprocess my outputs to make plots/statistical analysis

- I access my input data from on EOS
- I define a set of selections in **RDataFrame** and run over a **subset of the data interactively** (~minutes)
- If successful I run distributed over the full dataset (also ~minutes, though with **many more cores**)
- Further analysis (fitting, plotting, etc.) is produced interactively in a broader workflow (e.g., via Snakemake) on my local machine/institute resources

- We test locally the analysis code on a subset of the files to check for code bugs and algos: running locally on the CPUs of the node. The code that we use for interactive analysis is the **same as the batch one**. - We then submit with dask the processing of dataset, usually read from xrootd. We use 200-300 workers to read few 10 B events. All the analysis is redone on the fly, from reduced nanoAOD CMS datasets - The processing usually takes few hours --> we disconnect from Swan, keeping the dask scheduler active there (much easier than on lxplus) - optionally we run the same analysis code but we save an intermediate step on EOS to reduce the full data readout (especially useful if the dataset is read from outside CERN) - Once we get outputs, we use the jupyter notebooks for plotting, and analysis of the results

Feedback: Miscellaneous

The CAF pilot goes in the right direction for our analysis needs, based on the use of the dask scheduler. A crucial component that is missing in our opinion is the possibility to perform ML model inference from dedicated GPU resources, allocated in parallel to the CAF on the GPU on condor

- Improving the scheduler condor submission to make it smoother (submit all jobs at once)
- Provide an interface to a service with GPUs to perform ML inference from a pool of resources
- Make the configuration of the cluster node (N cpus, GB ram) more user friendly. It is crucial for us to increase the amount of RAm for some workflows: it is cumbersome to modify an hidden config file on EOS and then having to restart the full swan session to restart the dask scheduler correctly

On behalf of the ROOT team we are happy with how things have been developing. Once we spot issues, we communicate those and we get replies very quickly so that testing of the distributed RDF can be run smoothly.

Summary

- Overall positive feedback on the CERN Analysis Facility pilot, with some suggestions on how to improve
- Adoption of the pilot tied to success of distributed analysis models (RDataFrame, coffea)
- What next?
  - We would like to get input from more testers
  - Analyse results internally at CERN IT and decide next steps (e.g. effort to be invested)
  - Wait for results of the WLCG survey on AF submitted by LHCC to the experiments

Backup slides

Why scale out to batch

- Analysis typically 10-20% of batch usage
  - This is providing an additional entry point
- Optimize usage of batch resource
  - A well stacked batch farm with good job mix can get to 80% cpu utilization
  - One potential barrier to getting more out of the resources is pilots can run any workload
  - Known analysis jobs potential to stack nicely with other workloads to drive up utilization
  - No additional cost
- No fundamental reason same model couldn’t scale out to other platforms
  - In the current system only the use of /cvmfs and potentially eos for lib overrides in .local

Foreseen (potential) Improvements

- Kerberos authentication from SWAN console
  - Kerberos currently used both for schedd auth and to capture token for use by job
  - HTCondor supports JWT, easy to envisage SSO based JWT assignation & use for job token
- Interactivity relies on batch queues / priorities
  - During Pilot we want to test using overcommitted “static” slots on HTCondor
  - Separate Negotiator, and limits per user -> small buffer for interactivity then normal queues
- SWAN session needs to be live
  - Would like to support “shut the laptop and come back later”
  - Some challenges with how notebooks work, but feasible to keep kernel running, and then have mechanism to replay notebook to known state
- File Transfer library overlays
- XCache for remote data sets