Analysis Grand Challenge

Alexander Held (University of Wisconsin–Madison)
Oksana Shadura (University Nebraska–Lincoln)

Analysis Facilities Workshop, 18-20 June 2024

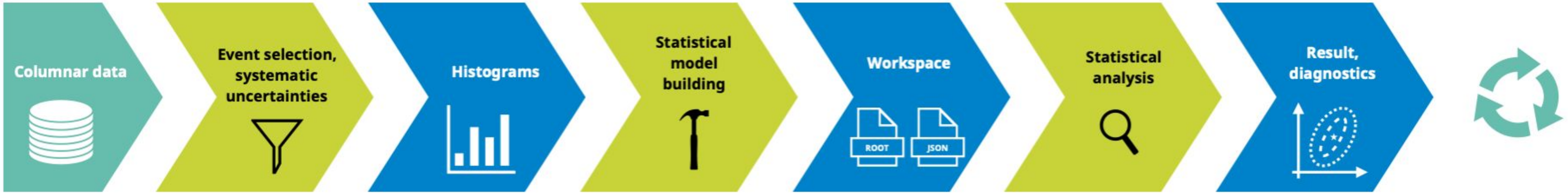
https://indico.desy.de/event/44722/

This work was supported by the U.S. National Science Foundation (NSF) cooperative agreements OAC-1836650 and PHY-2323298 (IRIS-HEP).

Analysis Grand Challenge (AGC):
execute series of increasingly realistic exercises toward HL-LHC

The AGC is about executing an analysis to test workflows designed for the HL-LHC.
This includes:

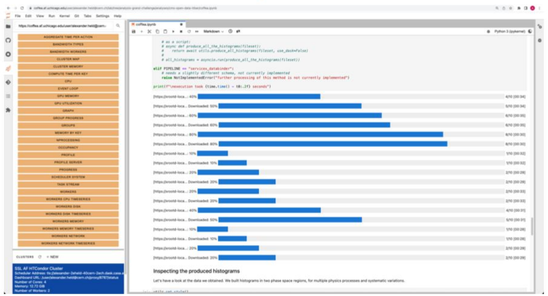
- **columnar data extraction** from large datasets,
- **data processing** (event filtering, construction of observables, evaluation of systematic uncertainties) into histograms,
- **statistical model construction and statistical inference**,
- relevant **visualizations** for these steps



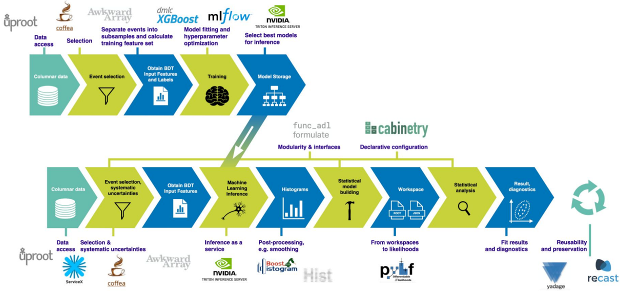
AGC - what was already done

within IRIS-HEP

The **AGC project started** properly in the **autumn of 2021**



- Physics task definition (multiple versions)
  - Capturing physics analysis requirements matching practical needs of physicists
  - Using CMS Open Data (reformatted to 2 TB of NanoAODs)
- IRIS-HEP AGC reference pipeline implementation
  - Analysis implementation based on IRIS-HEP stack of tools
  - Connecting many projects and developers
  - Cycle: iterating with experts and improving implementation



AGC - what was already done

- Developed **website as central resource**: https://agc.readthedocs.io/en/latest/
  - Work based on IRIS-HEP fellow project — AGC hosted and benefited from many great IRIS-HEP fellows

Analysis Grand Challenge Documentation

ttbar with CMS Open Data

CMS Open Data $t\bar{t}$: from data delivery to statistical inference

AGC Analysis Task Versions

Running at Analysis Facilities

$t\bar{t}$ Analysis Background

Plot $t\bar{t}$ Events

H>ZZ* with ATLAS Open Data

ATLAS Open Data $H \rightarrow ZZ^*$ with `ServiceX`, `coffea`, `cabinetry` & `pyhf`

Analysis Grand Challenge Documentation

launch binder | DOI 10.5281/zenodo.8338901

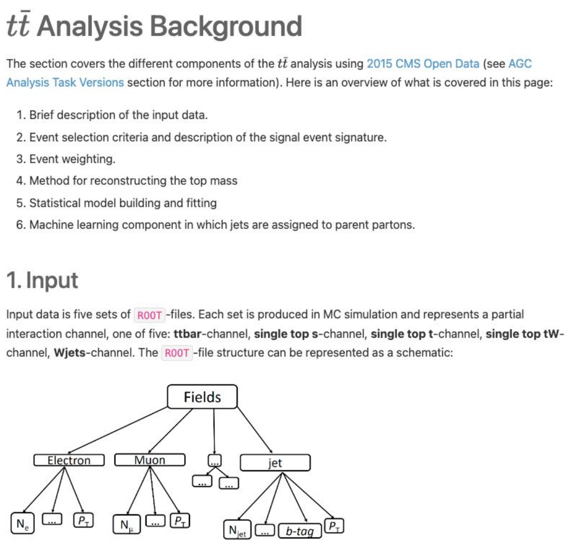
Contents

Analysis Grand Challenge Documentation

- Introduction
- AGC and IRIS-HEP
- AGC and you
- More details: what is being investigated in the AGC context
- Acknowledgements

Search

Analysis task details to allow for re-implementations



$t\bar{t}$ Analysis Background

The section covers the different components of the $t\bar{t}$ analysis using 2015 CMS Open Data (see AGC Analysis Task Versions section for more information). Here is an overview of what is covered in this page:

1. Brief description of the input data.
2. Event selection criteria and description of the signal event signature.
3. Event weighting.
4. Method for reconstructing the top mass
5. Statistical model building and fitting
6. Machine learning component in which jets are assigned to parent partons.

1. Input

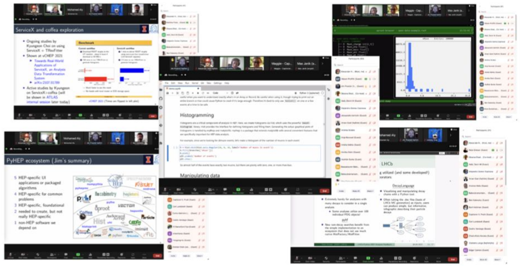
Input data is five sets of `ROOT`-files. Each set is produced in MC simulation and represents a partial interaction channel, one of five: **ttbar**-channel, **single top s**-channel, **single top t**-channel, **single top tW**-channel, **Wjets**-channel. The `ROOT`-file structure can be represented as a schematic:

AGC - what was already done

IRIS-HEP and the broader community

- Provided support & co-supervised fellows working on other implementations:
  - ROOT RDataFrame, Julia programming language
- AGC workshops (virtual) + hybrid
  - Reaching ~50-100 people (+ recordings)
- AGC demo event (+ community contributions)
- AGC demo days **(last demo day https://indico.cern.ch/event/1394151/)**
  - Short & focused technical talks and demonstrations
- Interactions with US ATLAS + CMS operations programs, HEP Software Foundation, …

AGC tools 2022 workshop



-> Established a range of events and activities to engage & disseminate

Yearly benchmarking exercises

- Provide a **stable analysis pipeline** at scale with 30 simultaneous users
- Benchmark iterative scaling to HL-LHC needs

getting ready for HL-LHC

Timeline	Fraction of HL-LHC dataset processed in 1h
2025	20% (40 TB)
2026	50% (100 TB)
2027	75 % (150 TB)
2028	100% (200 TB)

How is this challenge connected to analysis facilities R&D?

HEP Analysis Facilities

What physicists expect to see from “Analysis Facility”?

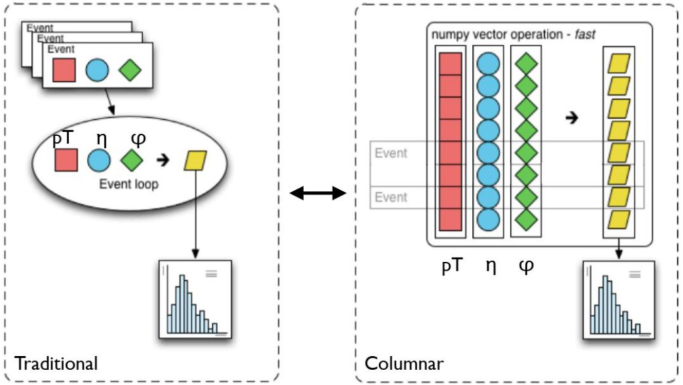
Homelab (https://domalab.com)

“Analysis facility” could be any type of resource from laptop to Tier-2

- HEP data access
- Number of cores to scale
- Recipe how to run code
- Disk space
- Favorite analysis framework already available

We need to think **now** how will look like Analysis Facility for HL-LHC and after

Building blocks: columnar analysis and support new pythonic ecosystem



Coffea Analysis Framework

ROOT RDataFrame

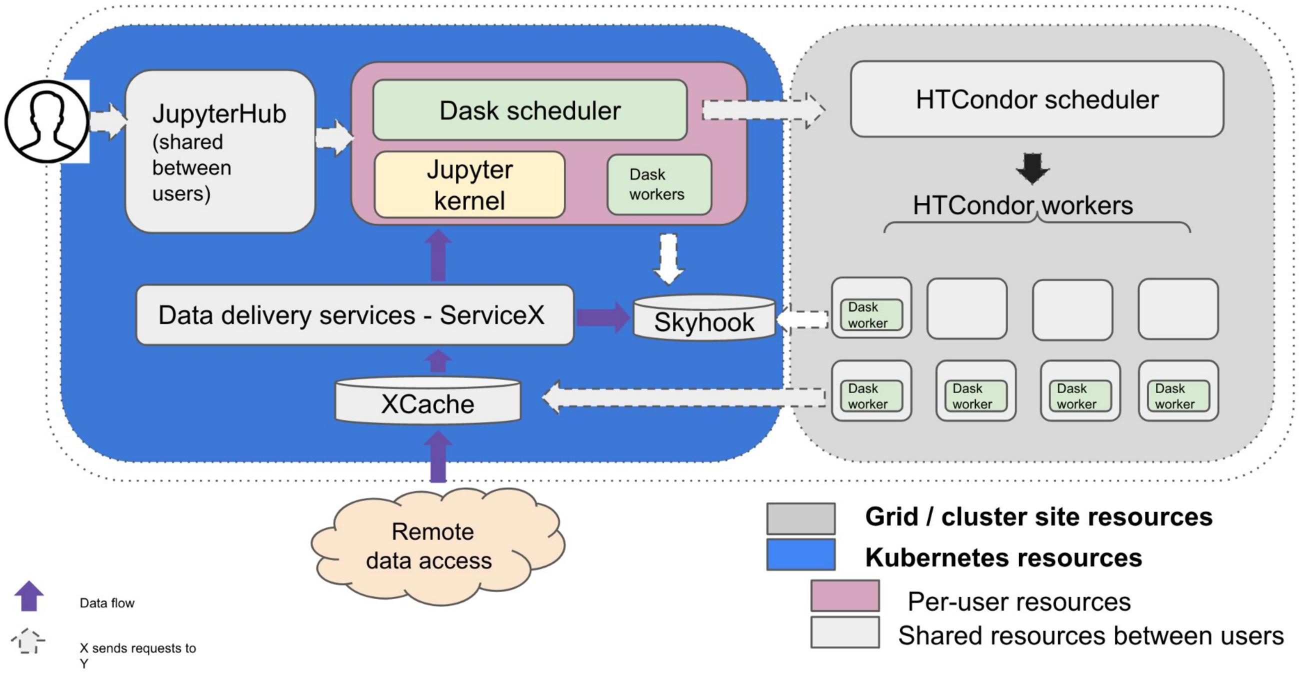
New columnar data analysis concepts

Analysis frameworks

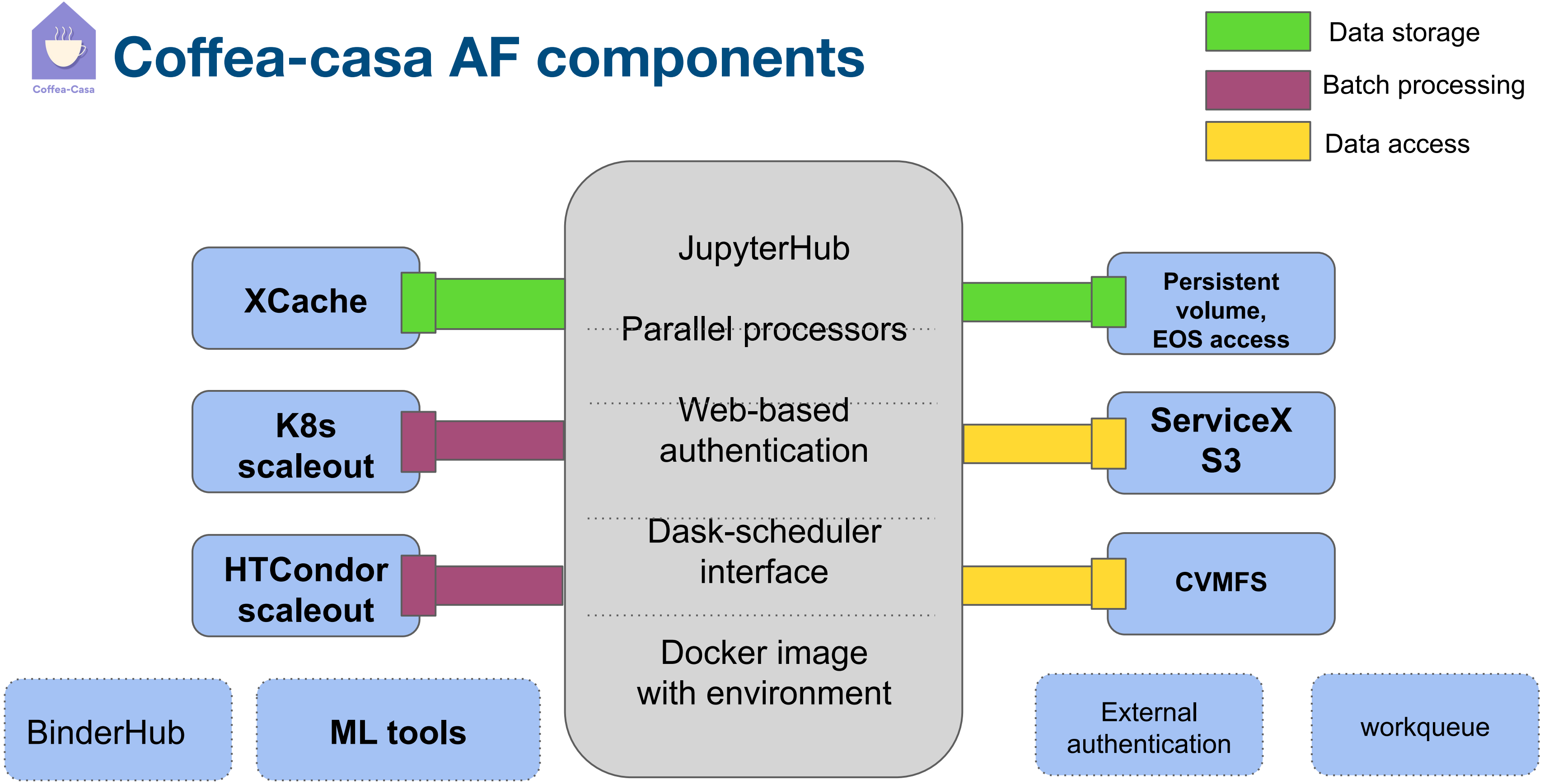
Distributed executors

Analysis Grand Challenge (AGC): preparing next generation of Analysis Facilities

Coffea-casa Analysis Facility is providing **AGC execution environment** to explore analysis workflows at scale

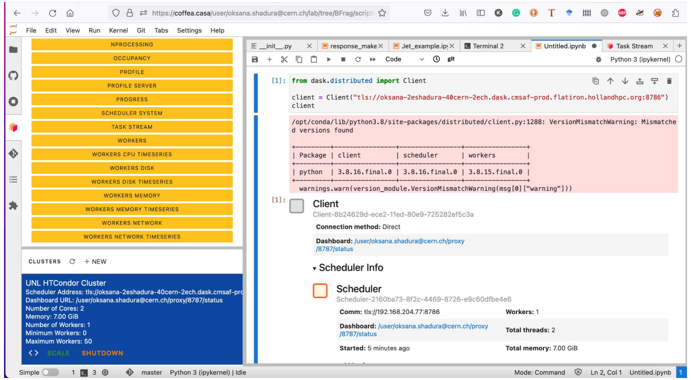


Coffea-casa AF components

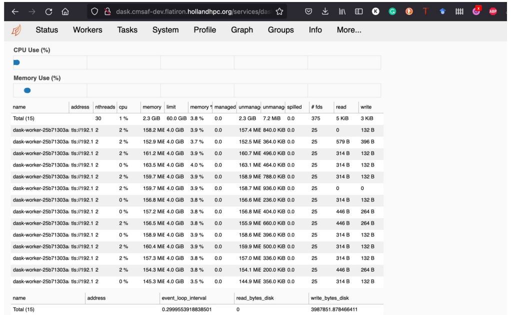


For coffea 2023 upgrade will will provide separate environments (Docker images) to be able to select on startup

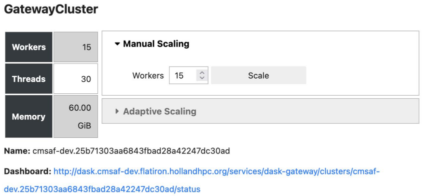
Building blocks: easy integration with scalable computing resources



- **Dask** task-management computational framework in Python (based on the manager-worker paradigm) integrates with <u>HTCondor @ UNL Tier 2 via ***"dask-jobqueue"***</u>



- Looking into **"dask-gateway"** (as backend we are testing scaling over Kubernetes = **significantly faster startup**)



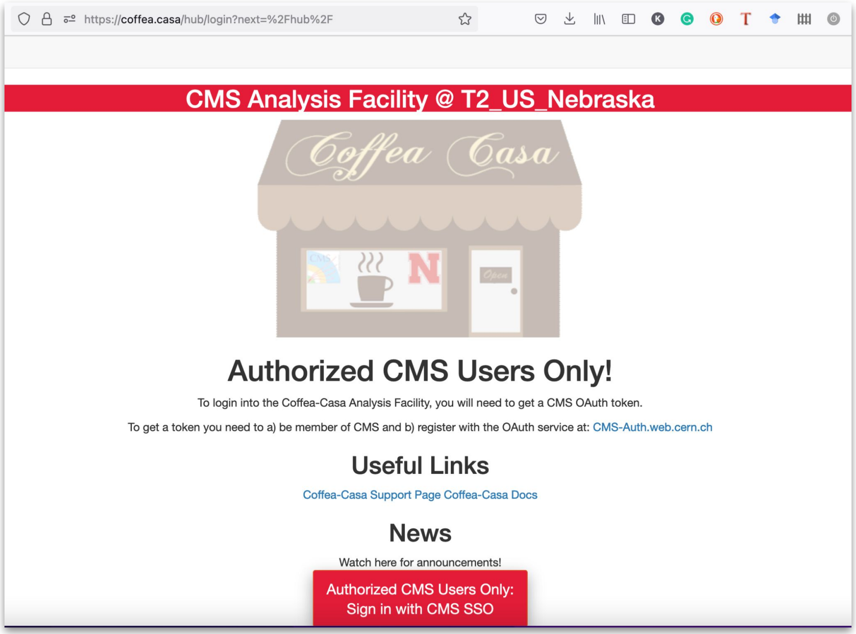
Investigating feature to be launch scheduler through Jupyter and connect directly from your laptop

(e.g. using *`oksana-2eshadura-40cern-2ech.dask.cmsaf-prod.flatiron.hollandhpc.org`* **)**

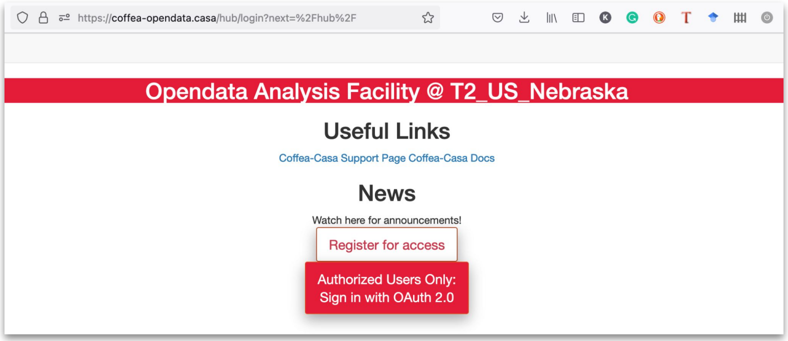
Building blocks: modern authentication (IAM/OIDC)

Authentication inside the system is independent of grid credentials

CMS Coffea-Casa Analysis Facility: https://coffea.casa



Opendata Coffea-Casa Analysis Facility: https://coffea-opendata.casa (register for access before)



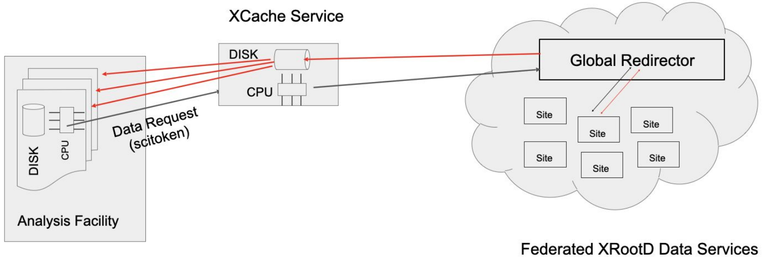
Powered by CILogon COmanage™

Powered by CMS IAM instance (and available for anyone in CMS)

An ATLAS instance at UChicago also exists: https://coffea.af.uchicago.edu/ (these slides focus on the UNL deployment)

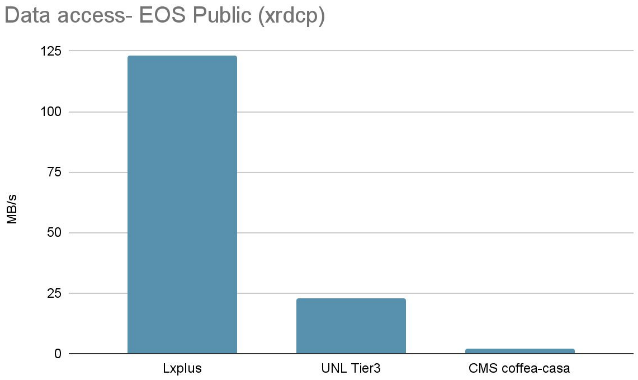
Building blocks: tokens for data access

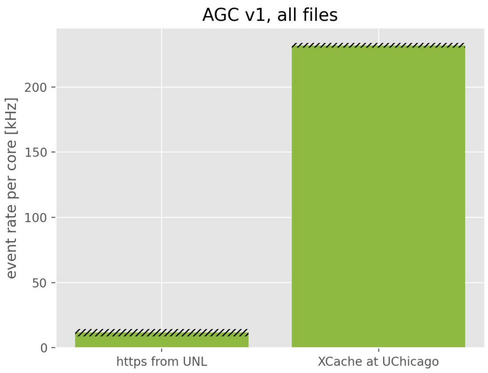
- **Enabled Token authentication (WLCG Bearer JWT pr**



JWT + XCache ServiceX

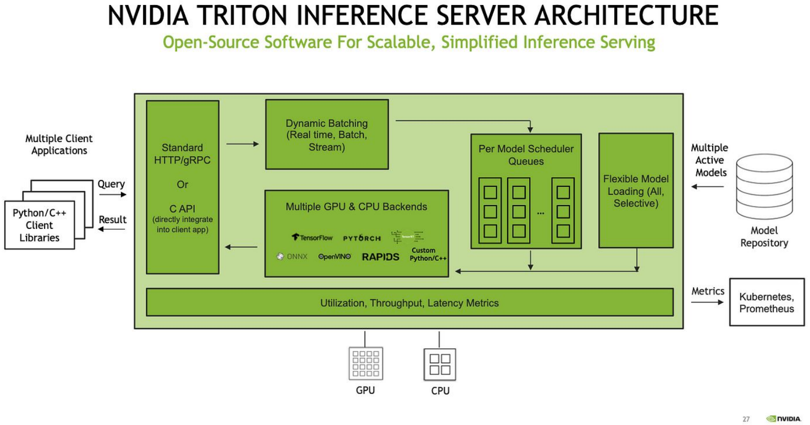
- Looking into **improving XCache setup**





Building blocks: machine learning services and tools

- **Triton is natively integrated in Coffea analysis framework** (the wrapper in coffea 2024)
- **Support for various deep-learning (DL) frameworks**
- **Simultaneous execution** - Triton can run multiple instances of a model, or multiple models, concurrently, either on multiple GPUs or on a single GPU
- **Dynamic scheduling and batching**
- **Nicely scales for multiple users of Analysis Facility**



https://developer.nvidia.com/nvidia-triton-inference-server

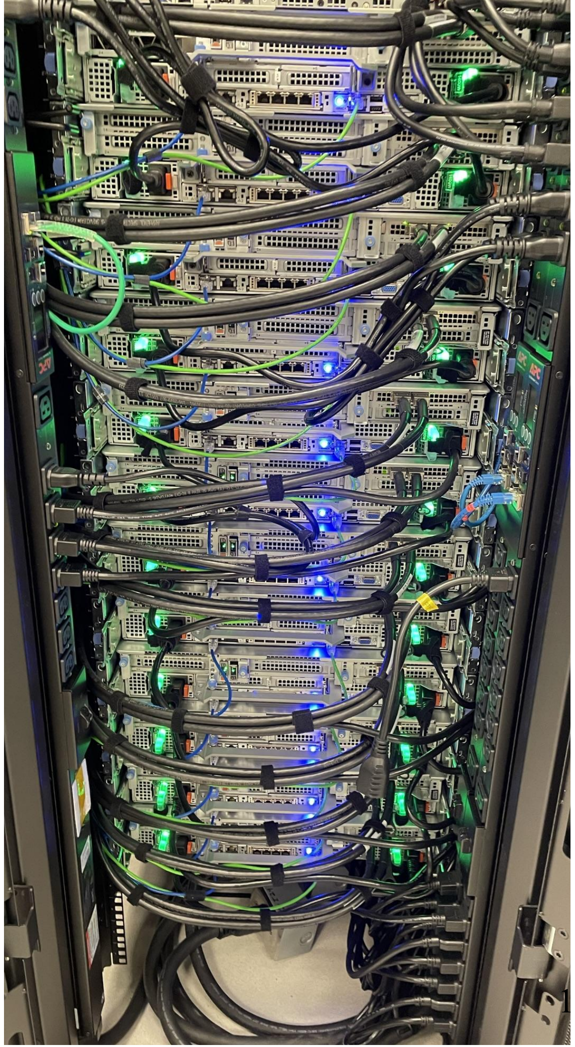
We have it available for you on CMS coffea-casa AF: https://coffea.casa

Casa Hardware – Flatiron

- 12 Dell R750 Servers, 512 GB Ram, 10 3.2 TiB NVMe Drives
  Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz (56 threads/CPU, 2 CPU per node)
- 1x V100S GPU, 2x P100 GPUs, and soon 2x L40S GPUs
- 2 x 100Gbps Networking, Calico + BGP (per node)
- Running Alma Linux 8.8 (Sapphire Caracal)
- Ceph-Rook Filesystem @ 103 TiB
- Ceph via Tier2 @ 8.7 TiB Usable
- Kubernetes (v1.27.6)
- Cert-manager, Dex, External-dns, Sealed-secrets, Traefik,CVMFS

ceph

NVIDIA

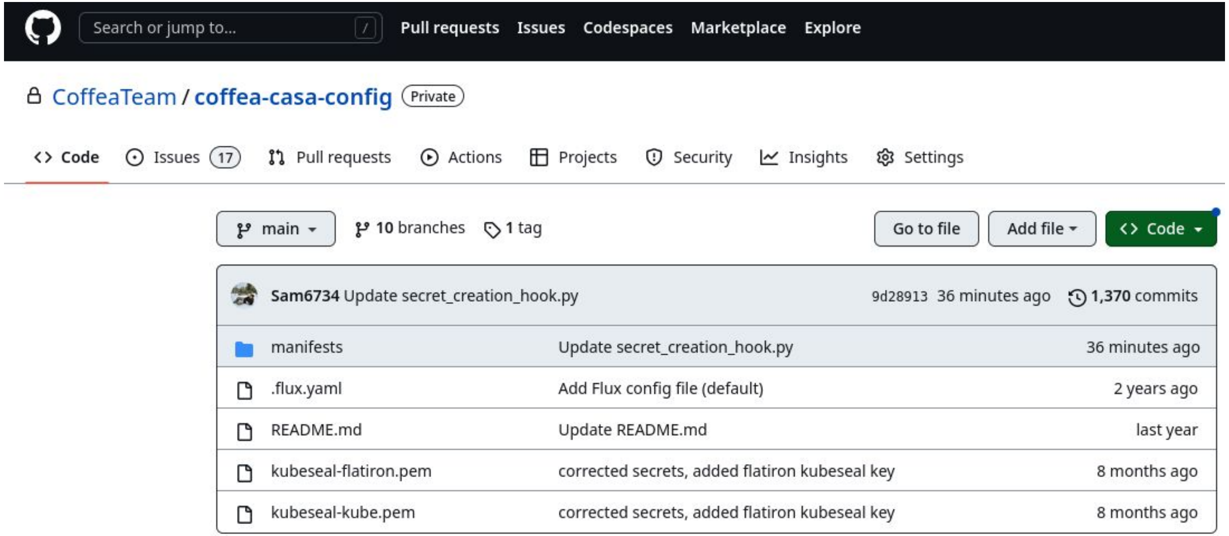


Why we prefer to use Kubernetes at UNL?

- Easy to manage (e.g. automatic management of configurations)
- Simple to integrate of new services (e.g. already available helm charts)
- Easy to scale deployment in case of the need (e.g. XCache deployment for 200 Gbps exercise)

Easy to manage: Infrastructure & Management

- Configs for casa facility are kept in git
- Changes follow GitOps techniques
- Changes are applied in-situ via a Flux agent



git

flux

kubernetes

Easy to deploy: example of Triton Inference Service

- To leverage the presence of our GPUs, **an inference service is deployed in the Kubernetes**
- Training sets are able to be stored in an **S3 bucket (object store)** deployed for it

```
I0426 14:13:43.918751 1 metrics.cc:650] Collecting metrics for GPU 0: Tesla V100S-PCIE-32GB
I0426 14:13:43.918929 1 tritonserver.cc:2214]
+----------------------------------+------------------------------------------------------------------------------------------
| Option                           | Value
+----------------------------------+------------------------------------------------------------------------------------------
| server_id                        | triton
| server_version                   | 2.25.0
| server_extensions                | classification sequence model_repository model_repository(unload_dependents) schedule_policy model
| model_repository_path[0]         | s3://rook-ceph-rgw-my-store.rook-ceph.svc:80/triton-c9adf042-ffb8-4221-bd42-e385efb1d0e2
| model_control_mode               | MODE_EXPLICIT
| startup_models_0                 | *
| strict_model_config              | 0
| rate_limit                       | OFF
| pinned_memory_pool_byte_size     | 268435456
| cuda_memory_pool_byte_size{0}    | 67108864
| response_cache_byte_size         | 0
| min_supported_compute_capability | 6.0
| strict_readiness                 | 0
| exit_timeout                     | 30
+----------------------------------+------------------------------------------------------------------------------------------
```

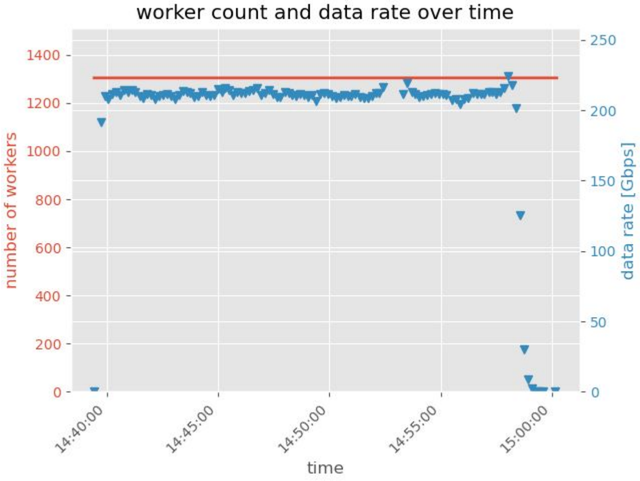
Scaling to HL-LHC: 200 Gbps setup

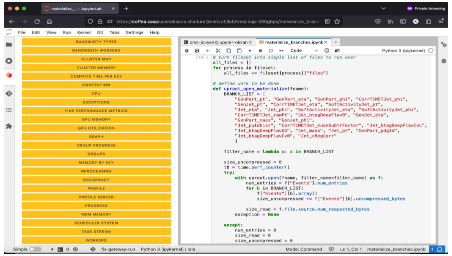
- Uproot + Coffea notebooks https://github.com/iris-hep/idap-200gbps and using CMS Run2 NanoAOD (~100TB)
    - **Read data from XCache on the Coffea-Casa facility at the Nebraska Tier-2 (running in Kubernetes).**
    - **Expand scale out into the site HTCondor and Kubernetes cluster.**
    - Dask tasks processed in TaskVine & Dask backends.
  - Compute values from the events read in; accumulate into histograms: “Direct from NanoAOD” style analysis.
- Notes on realism:
  - Real XCache setup. Token-based auth using the IAM service at CERN.
  - LZMA decompression dominates analysis time (~70%). To hit our target 25KHz-per-core processing rate, we recompressed the NANOAOD using ZSTD. About 20% larger than the original dataset, ~2.5x faster.
    - N.b.: our strong opinion is CMS needs to make this change.
  - We scale-out to HTCondor but, for these tests, pre-create the workers.

Uproot results

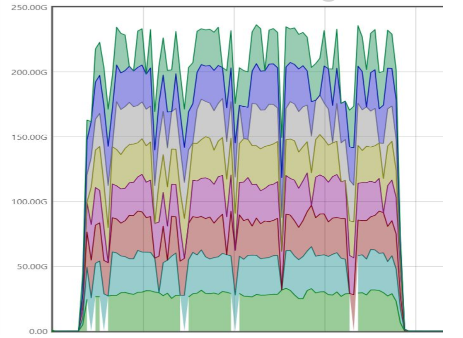
From the statistics in the notebook:

- Data read (compressed): 58.33TB
- Average data rate: 221Gbps
- Peak data rate: 240Gbps
- Total event rate : 32,256 kHz
- Processed 40,276,003,047 events total
- Per-core event rate : 27.66 kHz
- Files processed: 63,762 (17 failed)



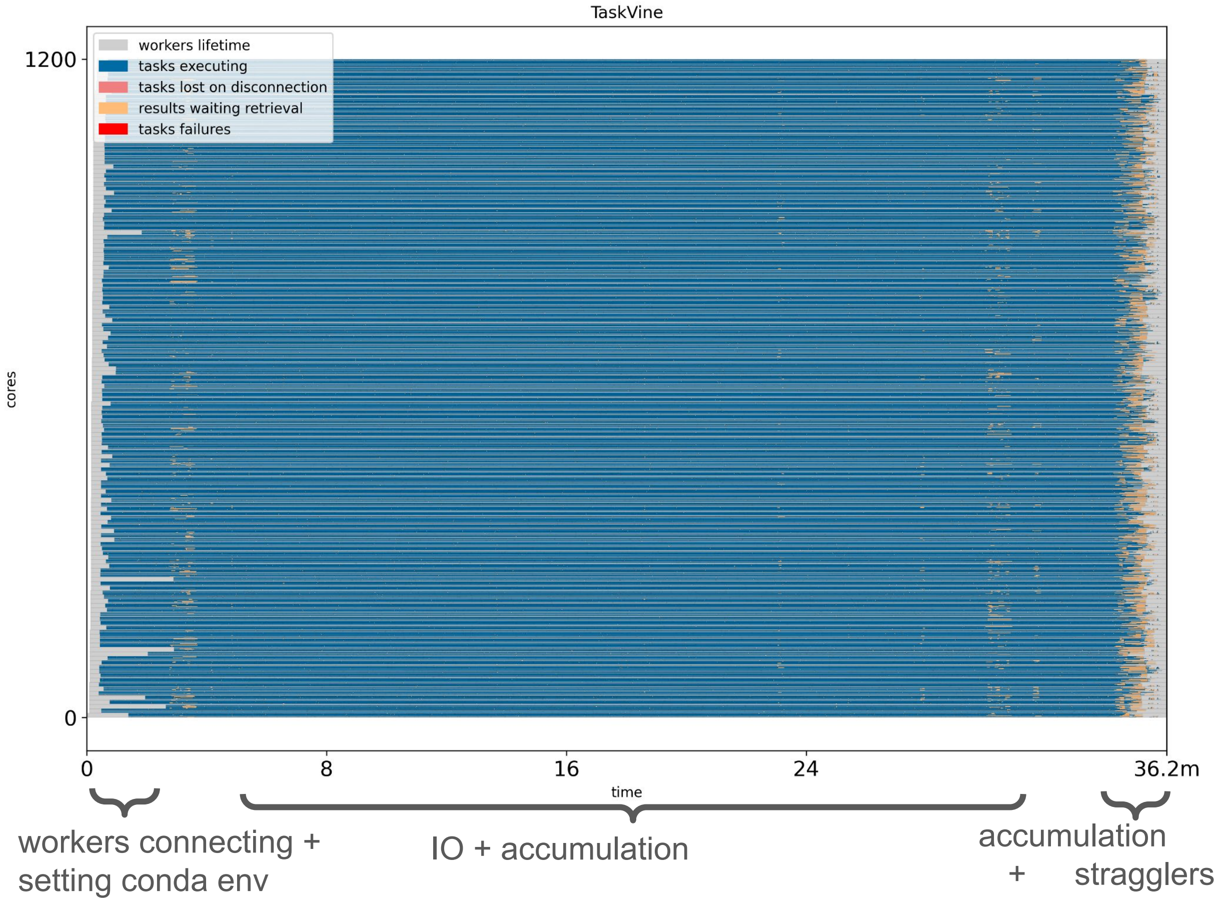


Network rates from XCache storage:

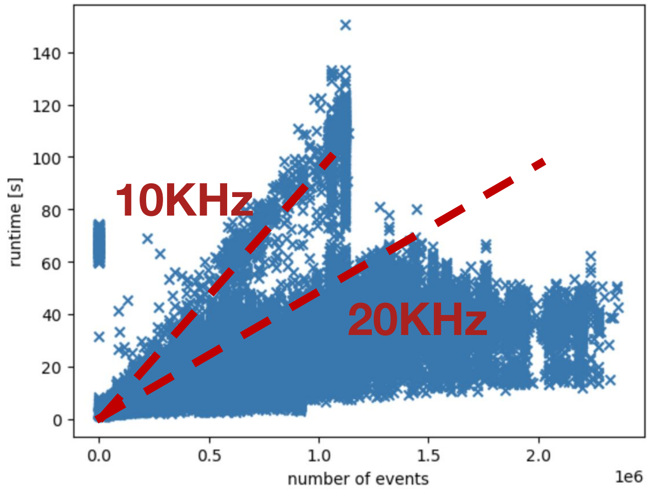


Rates from different, but representative run)

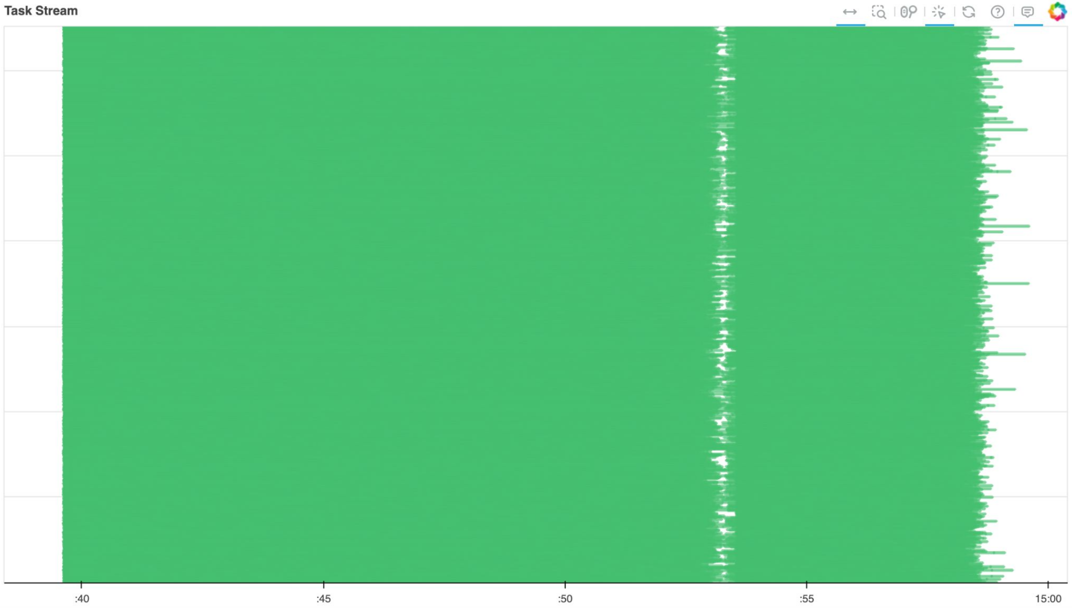
1200 cores across 150 8-core workers

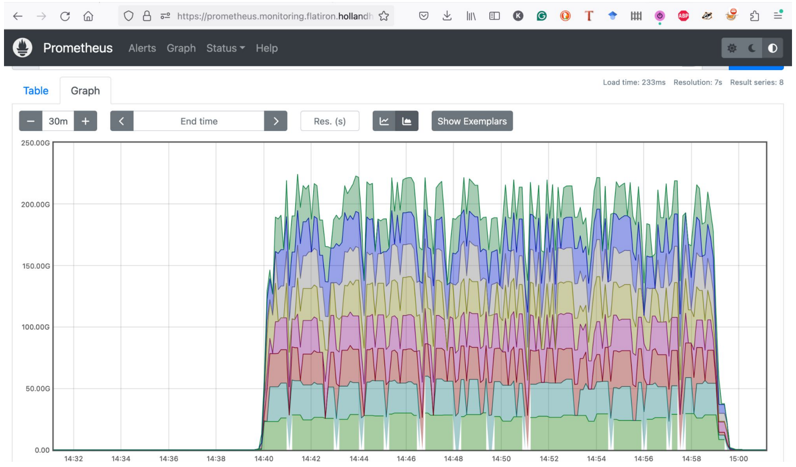


Runtime vs # Events as seen by xcache



Dask task stream and xcache stats over the same run



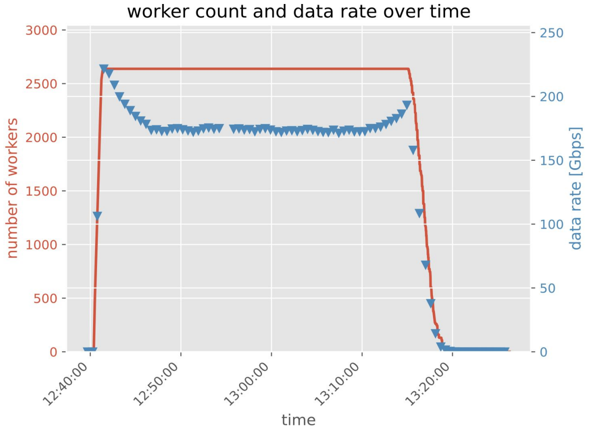


More results coming soon for upcoming CHEP 2024 conference

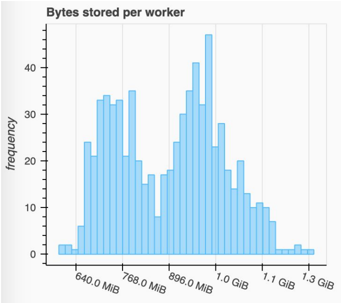
Uproot Toolset, PHYSLITE

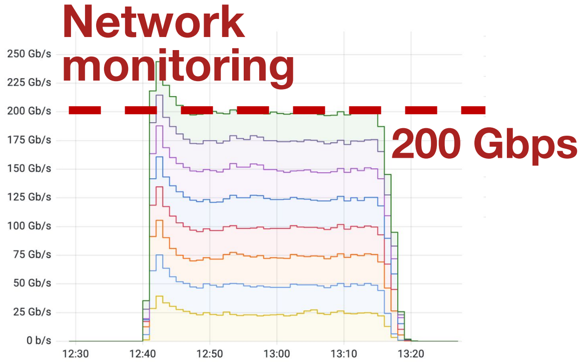
From Brian Bockelman talk "IRIS-HEP 200Gbps challenge"
HSF/WLCG workshop

- Several variants were explored; Dask vs TaskVine, dask-jobqueue vs dask-gateway.
- At UChicago, also processed ATLAS PHYSLITE files directly in Python.
  - Goal was using coffea 2024, dask-awkward, uproot; ended up using direct processing in uproot.
  - 218k files, 190TB data, 23B events, ~8kHz/core
- Highlights:
  - Scaled Dask up to around 2.5k cores
  - 200Gbps throughput sustained in network monitoring; slightly less in 'effective bytes' into Dask.
- Biggest challenge has been understanding memory usage; significant difference between "uproot only" and the full Coffea 2024.



memory profile across workers



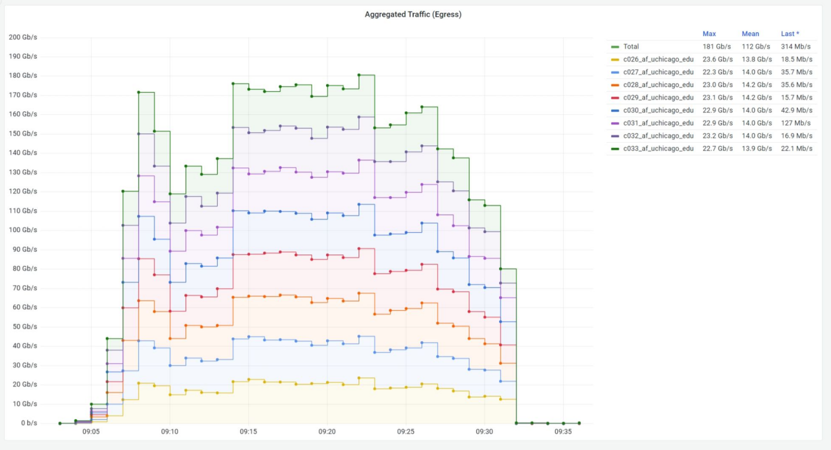


ServiceX Results

From Brian Bockelman talk [“IRIS-HEP 200Gbps challenge”](#)
HSF/WLCG workshop

Using ServiceX data extraction and delivery delivery service as part of pipeline:

- To reduce the overhead of small datasets, we ran on a subset that consisted of the bulk of the data.
- Highlight run:
  - 4 Datasets
  - 146TB total
  - 19,074,862,754 Events
  - 170Gbps
  - Limited to 1,000 pods.
  - Time: 32:28
  - Event Rate: 9,787 kHz



200 Gbps related slides summarizes a large body of work across IRIS-HEP and USCMS/USATLAS:

- Fermilab: Lindsey Gray, Nick Smith
- Morgridge: Brian Bockelman
- Notre Dame: Ben Tovar
- Princeton: Jim Pivarski, David Lange
- UChicago: Lincoln Bryant , Rob Gardner, Fengping Hu, David Jordan, Judith Stephen , Ilija Vukotic
- National Center for Supercomputing Applications: Ben Galewsky
- U. Nebraska: Sam Albin, Garhan Attebury, Carl Lundstedt, Ken Bloom, Oksana Shadura, John Thiltges, Derek Weitzel, Andrew Wightman
- UT-Austin: KyungEon Choi, Peter Onyisi
- U. Washington: Gordon Watts,
- U. Wisconsin: Alex Held, Matthew Feickert

Thank you for your attention!

If you have any questions, please feel free to get in contact directly or via analysis-grand-challenge@iris-hep.org (sign up: google group link)

Backup
AGC configuration

AGC versions

Description of versioning scheme: documentation

- The **AGC analysis task** evolves via **major versions**
  - **v0:** custom ntuple inputs -> superseded (do not use this anymore)
  - **v1:** NanoAOD inputs -> baseline to use
  - **v2:** machine learning, more systematic uncertainties -> heavier CPU & I/O requirements (almost HEAD)
  - We are developing new version with new coffea 2024 (check notebook from demo day)

AGC pipeline configuration

- **Baseline:** full AGC pipeline with distribution via **Dask** (`USE_DASK = True`)
  - Can also be ROOT version with distributed RDF
- **Advanced:** pipeline with **ServiceX** (optional)
  - USE_SERVICEX = True
  - Employ your XCache if available and compare performance
- **Advanced:** include **additional ML functionality** (optional, AGC v2)
  - Training: run `jetassignment_training` & reproduce models, more advanced: `USE_MLFLOW = TRUE`
  - Inference: `USE_TRITON = TRUE`

Options on this slide refer to the ttbar_analysis_pipeline.ipynb implementation.

AGC metrics that might be of interest

Metrics that might be of interest

- **Standard metrics** (in the many configurations outlined previously)
  - Data volume processed (per time and core)
  - Event processing rate per core
  - Scheduling efficiency
- **Data pipeline comparisons**: ratio of ServiceX+coffea and coffea (directly reading original input) runtimes
  - Assumption: input data sitting in XCache
  - Goals: no substantial slowdown of initial execution of ServiceX+coffea setup, demonstrate significant speedup in repeated runs (hitting ServiceX cache)
- **Additional points of interest**
  - Capture multi-user setups: run multiple AGC pipelines in parallel
  - Evaluate UX: how much manual intervention is needed (e.g. copying & settings proxy or tokens)