Analysis Grand Challenge benchmarking tests on selected sites

David Koch[1]

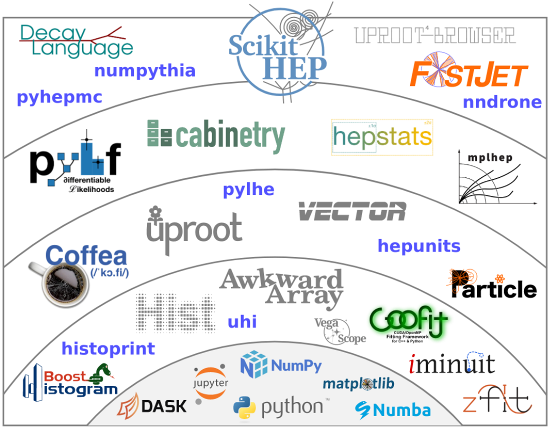
Thomas Kuhr[1], Günter Duckeck[1], Dennis Noll[2], Benjamin Fischer[2]

[1] Ludwig-Maximilians Universität München [2] RWTH Aachen

CHEP 2023

Analysis Grand Challenge

- developed by the IRIS-HEP team
- effort to demonstrate feature-completeness and scalability of scikit-HEP tools
- main framework of the analysis: coffea, offers a high level interface for columnar analysis
- github, readthedocs



Scikit-HEP: Python ecosystem for HEP analyses

Analysis Grand Challenge

ttbar-Analysis includes

- 1-lepton event selection
- top reconstruction
- cross-section measurement
- on-the-fly evaluation of systematic uncertainties

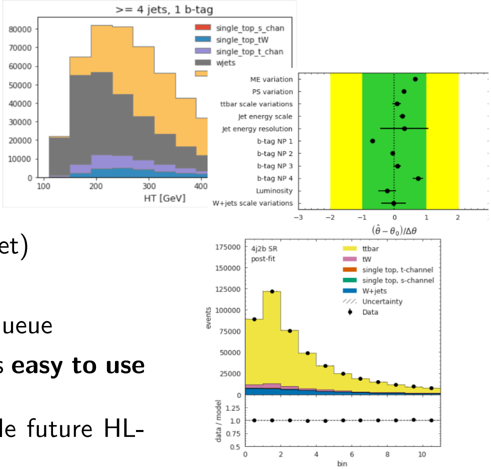
Analysis Grand Challenge

ttbar-Analysis includes

- 1-lepton event selection
- top reconstruction
- cross-section measurement
- on-the-fly evaluation of systematic uncertainties
- total of 3.44 TB of CMS open data
- only ∼138 GB are actually read (4% of the total dataset)
- 948 mio events and 10 variables
- work is distributed across many workers with dask-jobqueue

... all this sits in a single Jupyter notebook ⇒ analysis code is **easy to use** should also be **scalable** and **fast**

The AGC analysis is meant as a showcase of how a possible future HL-LHC analysis could look like



AGC at CHEP

Talks at CHEP directly related to the AGC:

- Elliott Kauffman, Machine Learning for Columnar High Energy Physics Analysis, Monday 2pm
- Andrea Scabià, I/O performance studies of analysis workloads on production and dedicated resources at CERN, Monday 3pm
- Oksana Shadura, Coffea-Casa: Building composable analysis facilities for the HL-LHC, Tuesday 10am
- Alexander Held, Physics analysis for the HL-LHC: concepts and pipelines in practice with the Analysis Grand Challenge, Tuesday 5pm
- Vincenzo Padulano, First implementation and results of the Analysis Grand Challenge with a fully Pythonic RDataFrame, Tuesday 5:15pm

Analysis Grand Challenge Benchmarks

Benchmarks performed on three different sites:

LMU institute cluster at LMU Munich consisting of one very powerful node and desktop computers
job-scheduler: SLURM
reading of the data via xrootd from LRZ

Analysis Grand Challenge Benchmarks

Benchmarks performed on three different sites:

LMU institute cluster at LMU Munich consisting of one very powerful node and desktop computers
job-scheduler: SLURM
reading of the data via xrootd from LRZ

LRZ WLCG Tier-2 site in Munich
job-scheduler: SLURM
data is stored on regular Grid storage (HDD) as well as on a XCache server (SSD)

Analysis Grand Challenge Benchmarks

Benchmarks performed on three different sites:

LMU institute cluster at LMU Munich consisting of one very powerful node and desktop computers
job-scheduler: SLURM
reading of the data via xrootd from LRZ

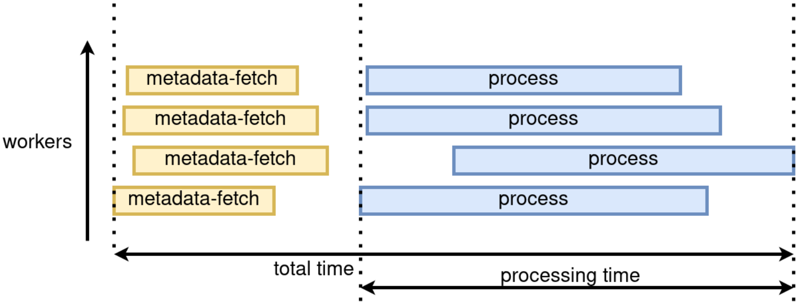
LRZ WLCG Tier-2 site in Munich
job-scheduler: SLURM
data is stored on regular Grid storage (HDD) as well as on a XCache server (SSD)

Vispa analysis facility operated by RWTH Aachen; provides a web-based terminal, code editor and jupyter hub: `https://vispa.physik.rwth-aachen.de`
job-scheduler: HTCondor
data is stored locally on SSDs and read directly; Vispa also has a very dedicated caching-system (arXiv) that I did not test with AGC yet

Measurements

Runtime

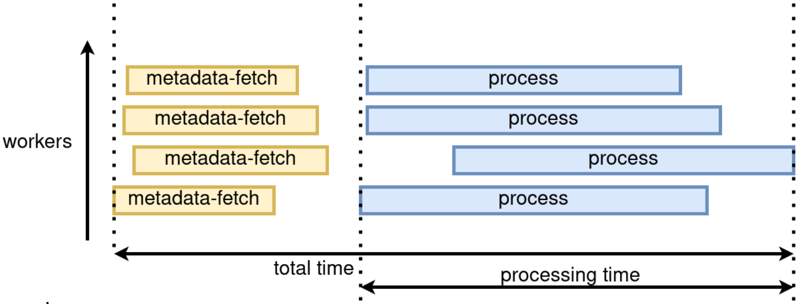
Only the part of the analysis which is run distributed is used for the benchmark → fetching some metadata and reading and processing the data



Measurements

Runtime

Only the part of the analysis which is run distributed is used for the benchmark → fetching some metadata and reading and processing the data

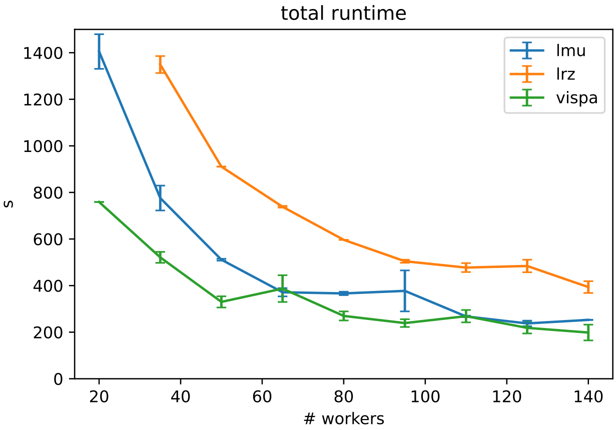


I can measure directly

- the total runtime
- the total processing time
- the sum of all process times across the workers (the sum of all blue rectangles here) via coffea's tooling

Measurements

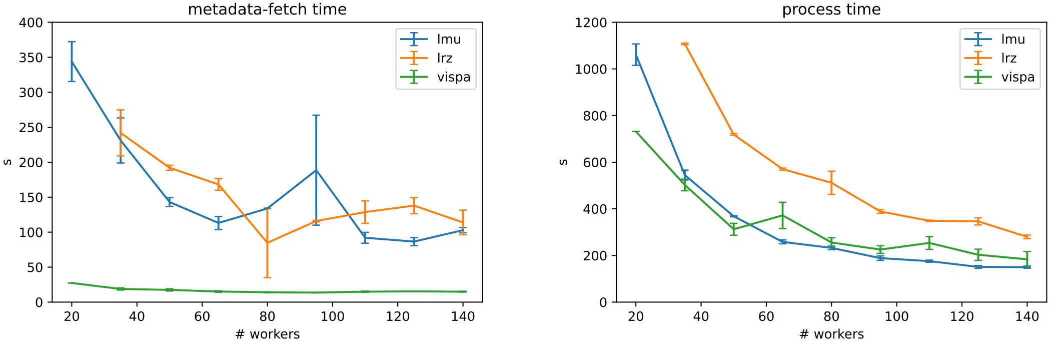
Runtime



total runtime

Measurements

Runtime



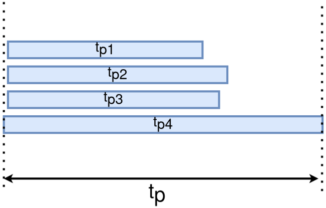
metadata-fetch time (= total - process time, also contains waiting and communication between dask workers) and process time

Measurements

Runtime

measure for the amount of overhead relative to the runtime

$$\frac{t_p}{\sum t_{p_i}/n}$$



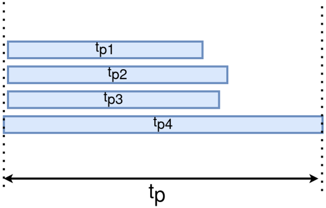
$\sum t_{p_i}/n =: \bar{t}$ is the average process time per worker – *pure computing* time without overhead

Measurements

Runtime

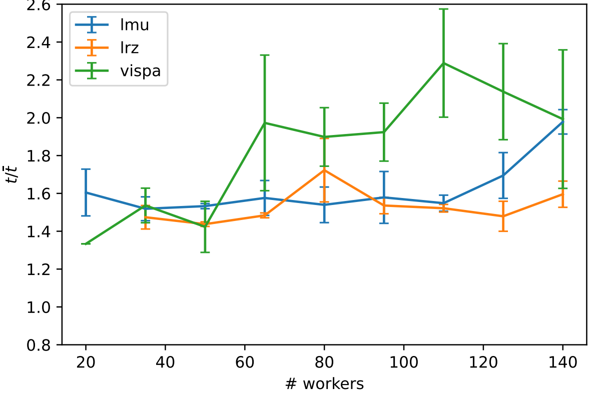
measure for the amount of overhead relative to the runtime

$$\frac{t_p}{\sum t_{p_i}/n}$$



$\sum t_{p_i}/n =: \bar{t}$ is the average process time per worker – *pure computing* time without overhead

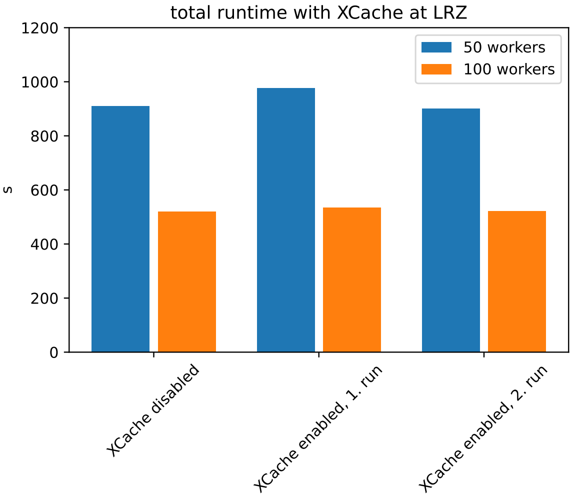
ratio between process time and average process time per worker



overhead

Measurements

XCache



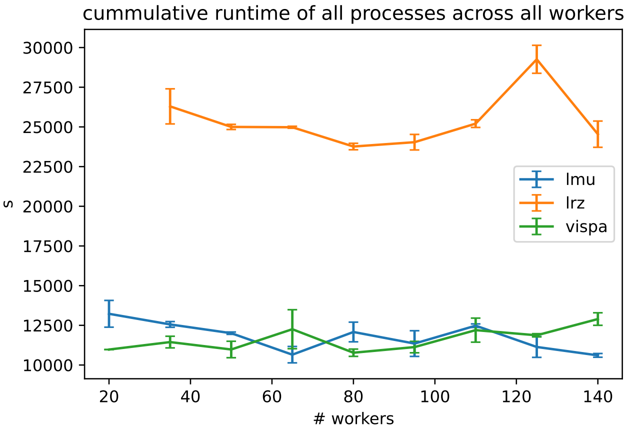
runtimes at LRZ with and without XCache enabled: makes no significant difference ⇒ with this setup, the analysis is hardly I/O limited

Questions?

Backup

Measurements

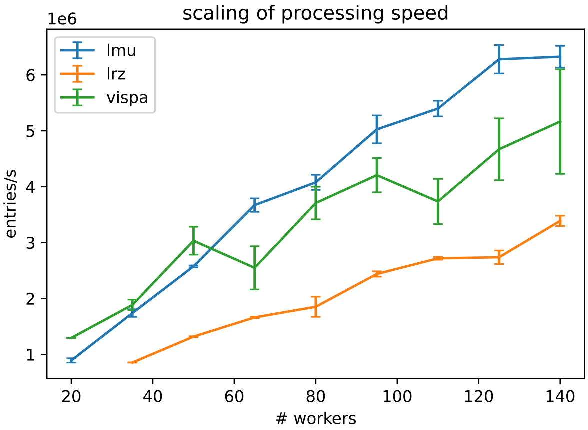
Runtime



summed process time across all workers

Measurements

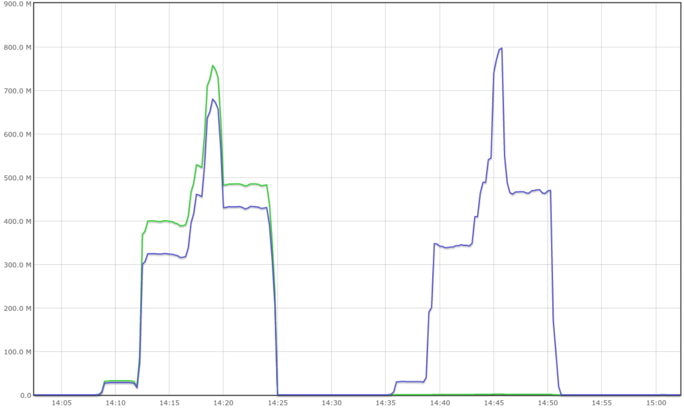
Runtime



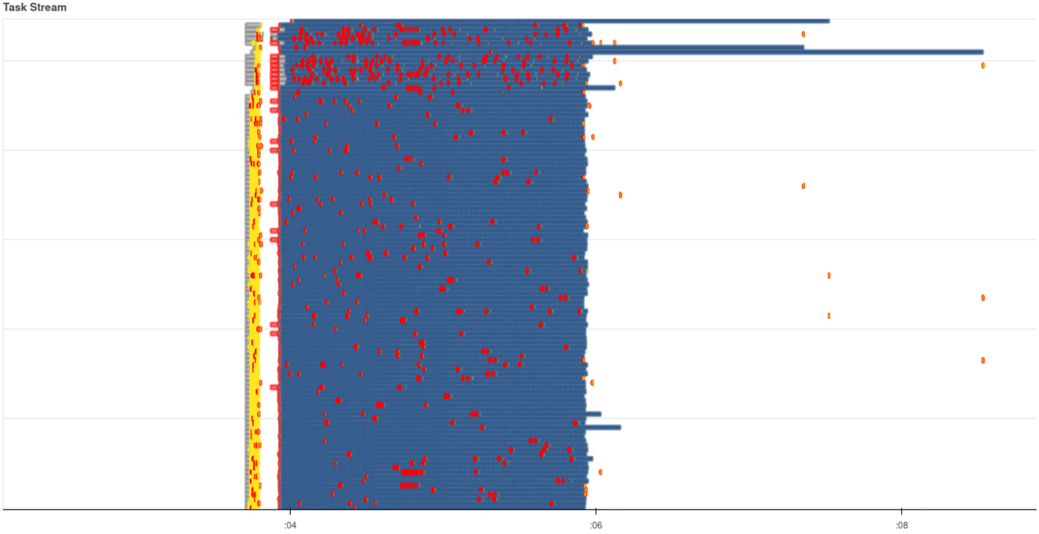
Scaling of process time

Measurements

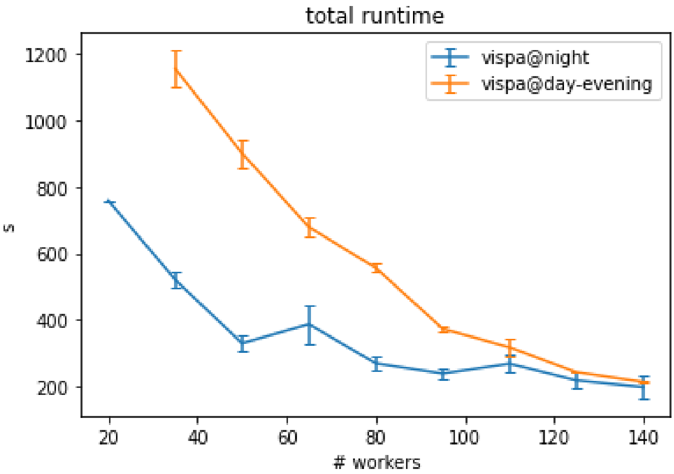
XCache



XCache load during two consecutive runs with 50 workers



example Dask dashboard with long tail



total runtimes during the day and night @Vispa