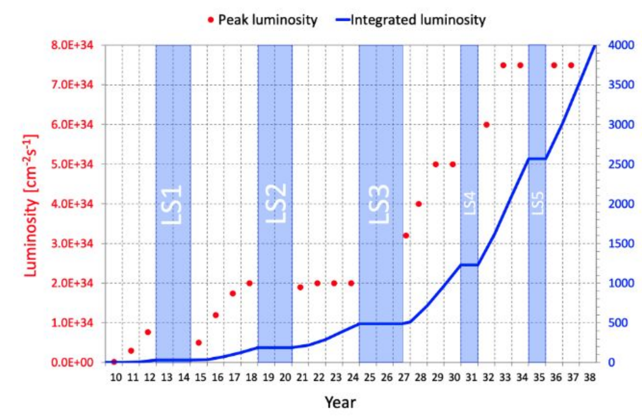
Finanziato dall'Unione europea NextGenerationEU | Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

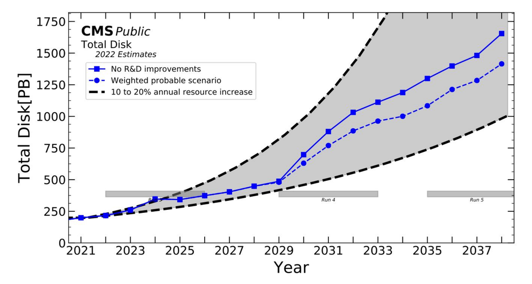
Quasi interactive analysis of High Energy Physics big data with high throughput

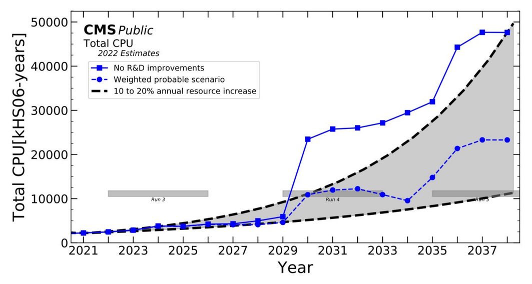
Tommaso Diotalevi[1,3], Francesco Giuseppe Gravili[1,7], Muhammad Anwar[2], Matteo Bartolini[1,5], Antimo Cagnotta[1,6], Adelina D'Onofrio[1], Paolo Mastrandrea[1], Gianluca Sabella[1,6], Federica Maria Simone[1,2], Bernardino Spisso[1], Alessandro Tarasio[4], **Tommaso Tedeschi**[1]

Motivation

The upcoming high-luminosity phase at the **CERN Large Hadron Collider (LHC)** and at future accelerator facilities will require an increasing amount of computing resources [1].







Higher rates of collision events → Higher demand for computing and storage resources

To better analyse this increasing amount of Big Data:

- Optimize the usage of CPU and storage;
- Promote the usage of better data formats;
- **Develop new analysis paradigms**!

- New software based on declarative programming and interactive workflows;
- Distributed computing on geographically separated resources.

HEP analysis performance evaluation

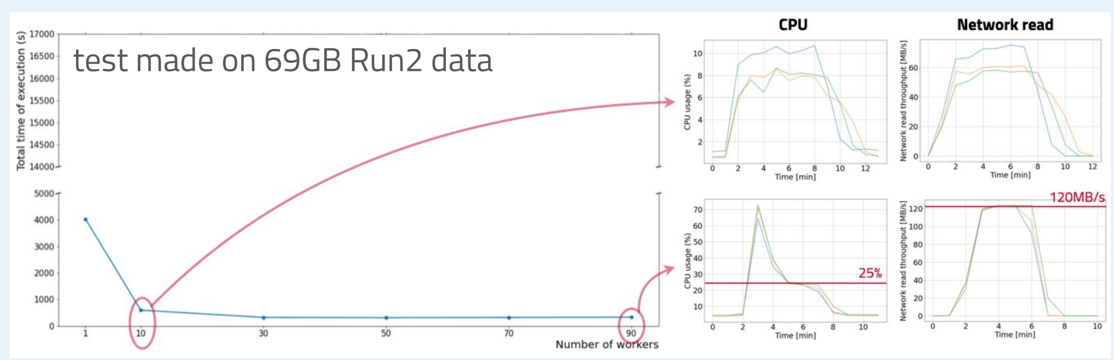
Evaluating the performance of several High Energy Physics analyses from different experiments, using an approach based on the one described in [2]

HNL Run2 search

with distributed features for DASK compatibility

Analysis made on **ROOT RDataFrame** [3] v6.27

Preliminary results



The same analysis workflow, running on an increasing number of workers shows a decrease in execution time.

- As expected, low number of workers show a CPU usage saturation;
- For a high number of workers, network access becomes the bottleneck (due to IO access, via protocols like xRootd/WebDAV).

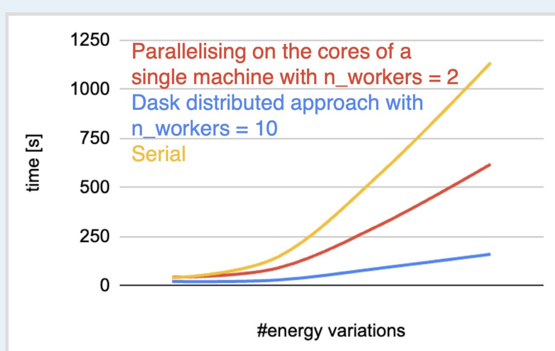
Zee simulations at Future Colliders (FCCee)

Feasibility study: Mimic systematic variations applying a gaussian smearing to e^+e^- energies many times.

Preliminary results

Events selection and histogramming: interactively with **ROOT RDataFrame** and **Jupyterlab** [4]

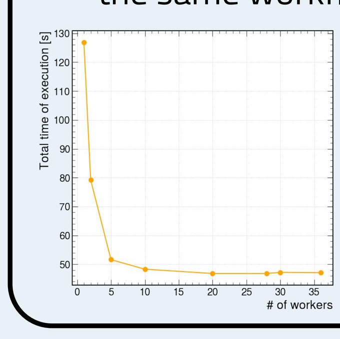
Dask [5] used as backend.



- Considering the overall execution time as metric and running the same workflow, there is a performance improvement in the distributed approach wrt the standard/serial approach;
- Moreover, it was tested that scaling resources, the performance further improves.

VLQ search

- Analysis made with **ROOT RDataFrame** v6.27;
- As performance study, we tried to increase the number of workers using the same workflow;



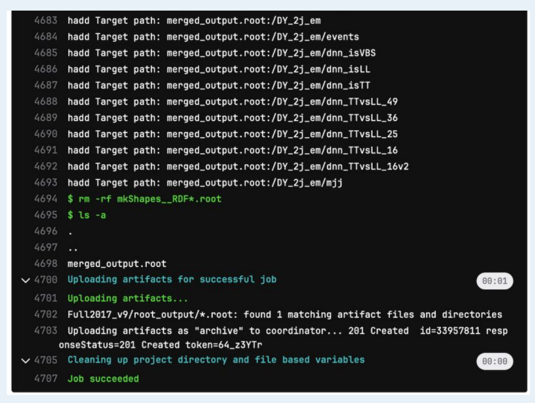
- The test has been performed on 5 MC files (~3GB).

Preliminary results

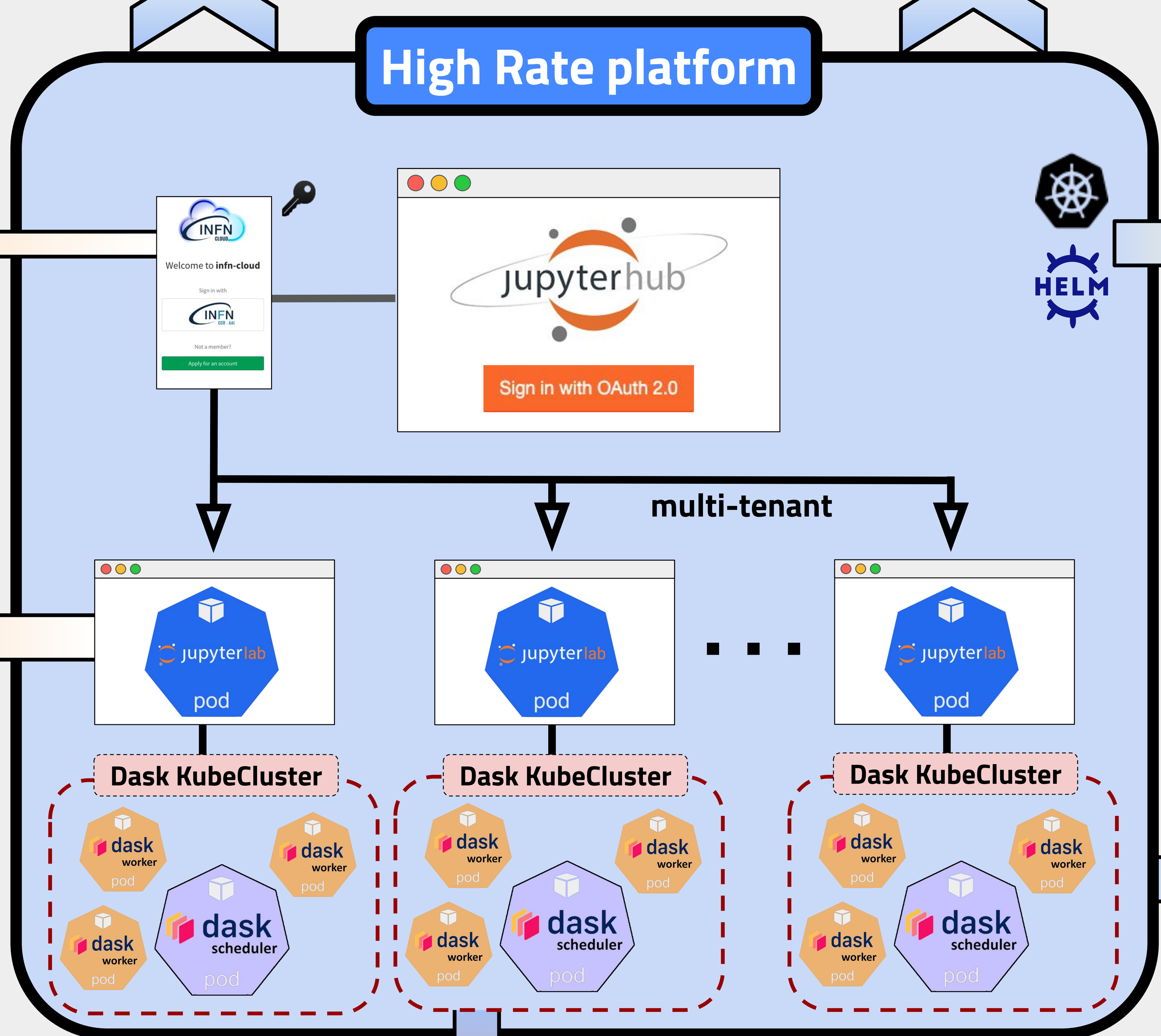
- Increasing the number of workers shows a decrease in execution time (from 3min to around 47s);
- After a certain number of workers, the execution time saturates (as shown in other applications).

CI pipeline triggering analysis execution on Analysis Facility

- Initial work setting up a CI pipeline running a full CMS (mkShapesRDF[6] framework-based) analysis on an HTCondor-based high-rate facility platform;
- Plan to start experimenting soon the use of **Dask** to improve handling and merging of the full dataset.



High Rate platform

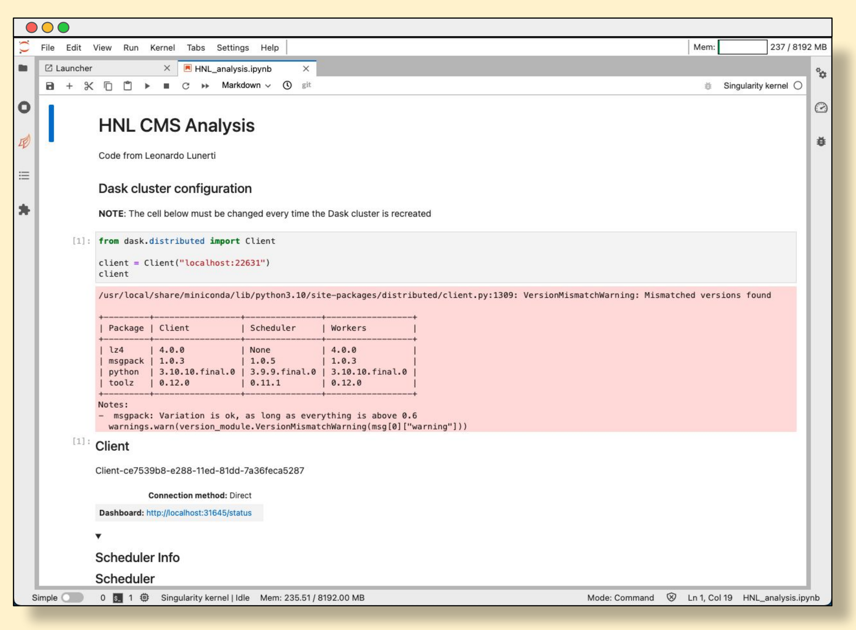


Access and security

After connecting to an entrypoint URL, the user reaches a **Jupyterhub** [7] instance that, after authentication and authorization via INDIGO-IAM [8], allocates the required resources for the user's working area.

User Interface

The user interface is based on **Jupyterlab**, customised with specific plugins for specific purposes (e.g. Dask).



The working environment is highly customizable, using tailored Docker containers. This is important when analyses require specific software (collaboration-wise).

Deployment

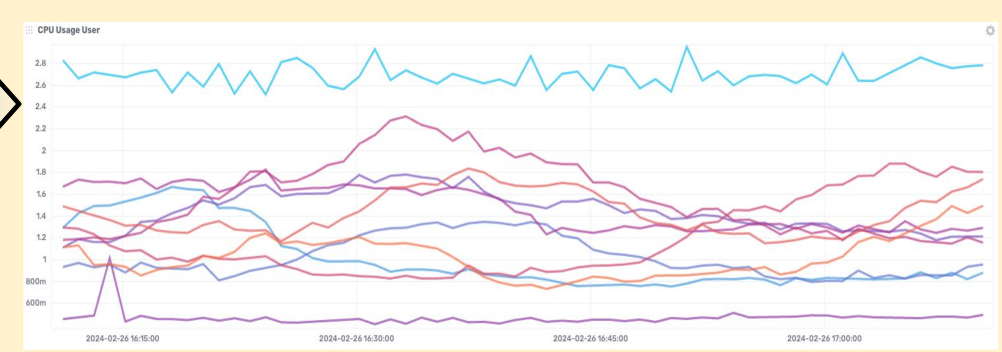
The deployment of the **Kubernetes** resources needed for the spawning of this platform, is handled via **HELM** [9] **charts** available in the GitHub organization [10].

Check the docs!

This allows a seamless, flexible, scalable and fault-tolerant deployment on the available resources, with a limited impact on the admin's work time.

Monitoring

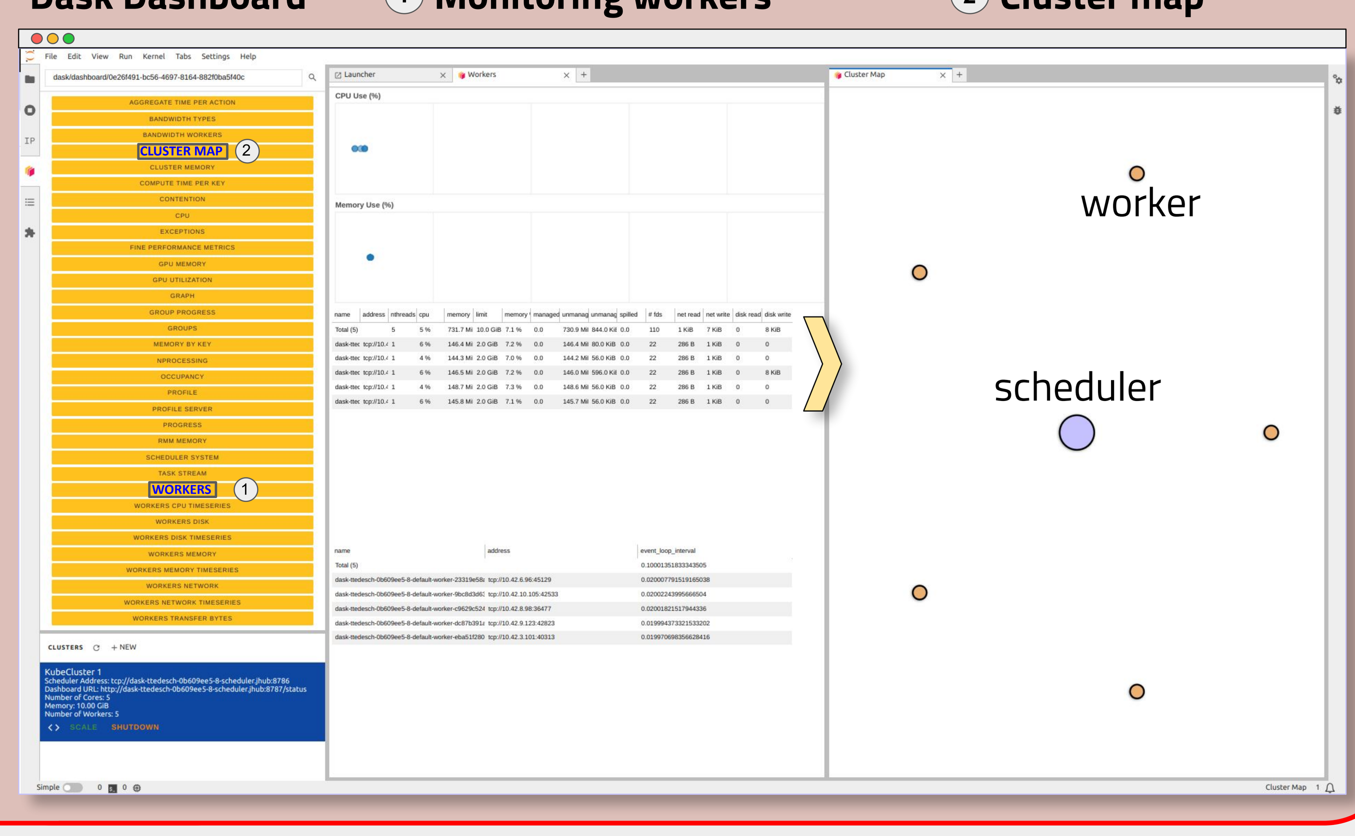
The platform is monitored using in-site metrics gathered and displayed using **InfluxDB** [13].



Distributing the workload

The **Dask Labextension** [12] plugin allows to interact with the Dask dashboard directly in the Jupyterlab session, getting access to useful monitoring panels.

Dask Dashboard ① **Monitoring workers** ② **Cluster map**



The Dask cluster is deployed on the Kubernetes (K8s) cluster using the Dask Operator [11] (a service that runs on your K8s cluster and allows to create and manage Dask clusters as K8s resources) through `dask_kubernetes.operator.KubeCluster` class, which provides a simple Python API to manage the cluster and allowing maximum flexibility for the end-user.

The deployment of such cluster can be done:

- either via the Dask Labextension (which implements a convenient GUI to create, scale and delete Dask clusters)
- or via CLI/notebook cell (this allows to better customize your cluster, choosing images, scheduler and workers resource requests, etc...).

In both cases, the user needs to instantiate a `dask.distributed` Client object to interact with the scheduler and start the computation.

References

1. https://twiki.cern.ch/twiki/bin/view/CMSPublic/CMSOfflineComputingResults
2. T. Tedeschi, V. E. Padulano, D. Spiga, D. Ciangottini, M. Tracolli, E. T. Saavedra, E. Guiraud, M. Biasotto, Prototyping a ROOT-based distributed analysis workflow for HL-LHC: The CMS use case, Computer Physics Communications, Volume 295, 2024, 108965.
3. https://root.cern/doc/master/classROOT_1_1RDataFrame.html
4. https://jupyterlab.readthedocs.io/en/latest
5. https://docs.dask.org/en/stable/
6. https://gitlab.cern.ch/cms-analysis/general/mkShapesRDF
7. https://jupyterhub.readthedocs.io/en/stable/
8. https://github.com/indigo-iam/iam
9. https://helm.sh/
10. https://github.com/ICSC-Spoke2-repo/HighRateAnalysis-WP5
11. https://kubernetes.dask.org/en/latest/operator.html
12. https://github.com/dask/dask-labextension
13. https://www.influxdata.com/

1: INFN - Istituto Nazionale di Fisica Nucleare
2: Polytechnic of Bari
3: University of Bologna
4: University of Calabria
5: University of Firenze
6: University of Naples
7: University of Salento

This work is supported by ICSC – Centro Nazionale di Ricerca in High Performance Computing, Big Data and Quantum Computing, funded by European Union – NextGenerationEU