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

ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Leveraging distributed resources through high throughput analysis platforms for enhancing HEP data analyses

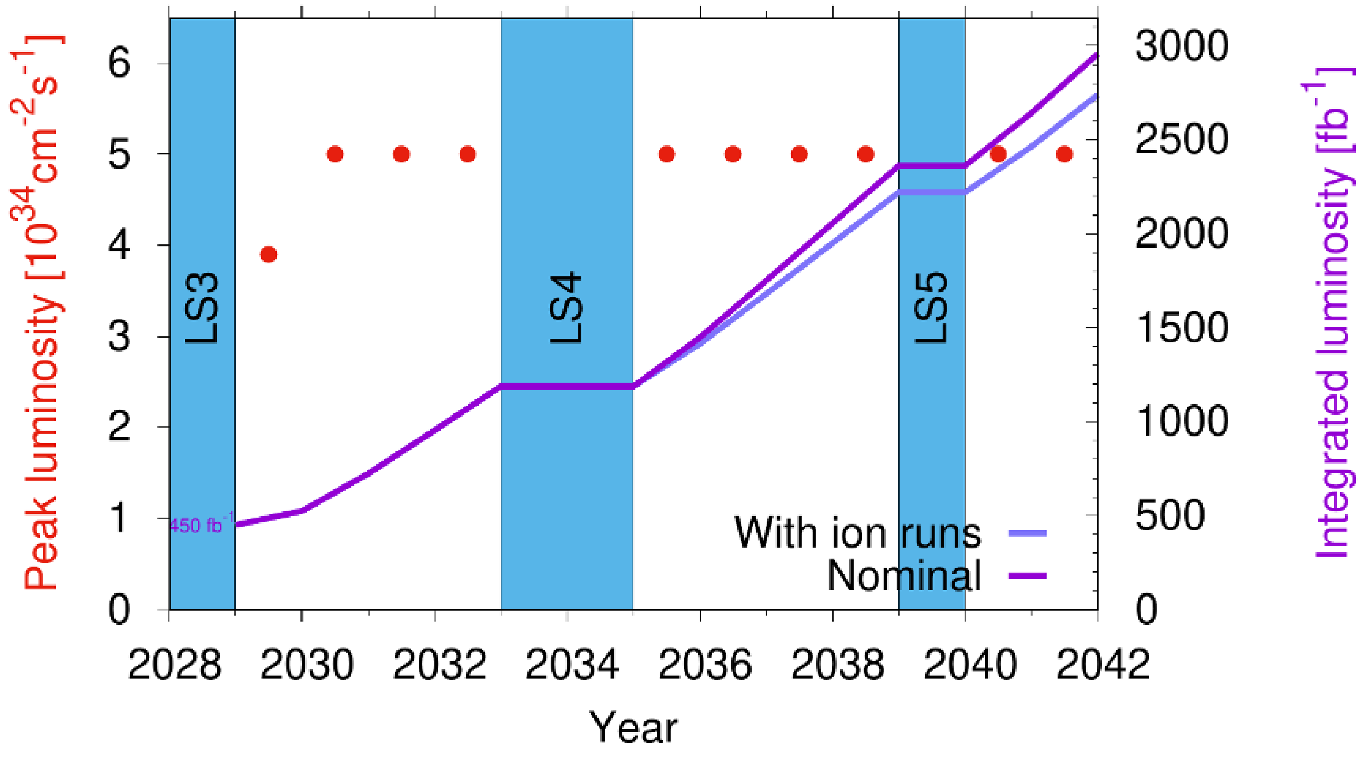
Adelina D'Onofrio[1], Tommaso Diotalevi[1,3], Francesco Giuseppe Gravili[1,5], Salvatore Loffredo[1,2], Elvira Rossi[1,2], Federica Maria Simone[1,4], Bernardino Spisso[1]
on behalf of the ATLAS and CMS Collaborations

1 INFN, 2 University Federico II, 3 University of Bologna, 4 Polytechnic Bari, 5 Università del Salento

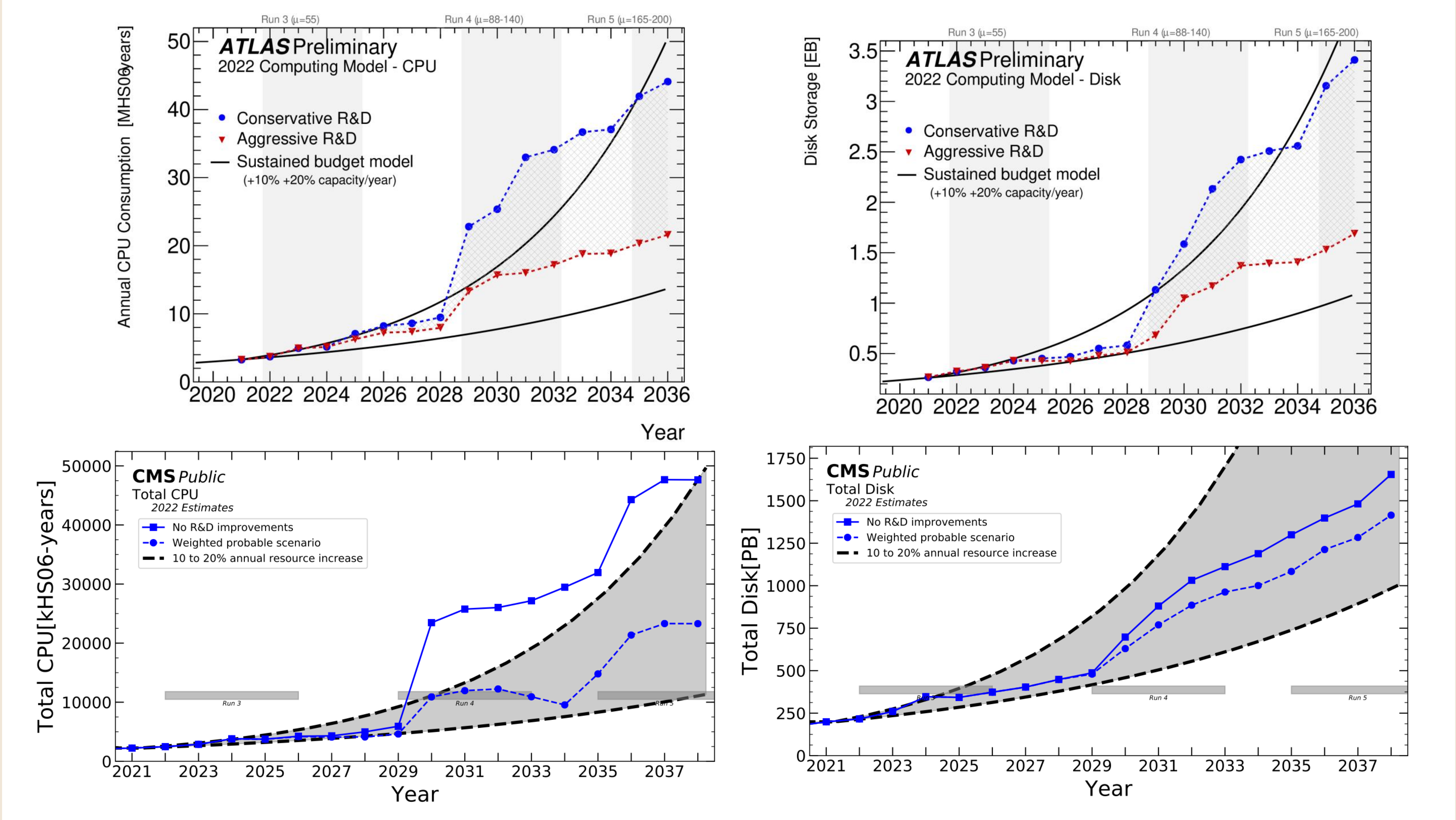
~~CHEP2024, 19-25 Oct 2024, Krakow~~ → ROOT PPP

Motivations

- **Challenges of LHC, and HL-LHC** are pushing to **re-think the HEP computing models**
  - Impact on several aspects, from software to the computing infrastructure



Similar trends for ATLAS and CMS HL-LHC projections



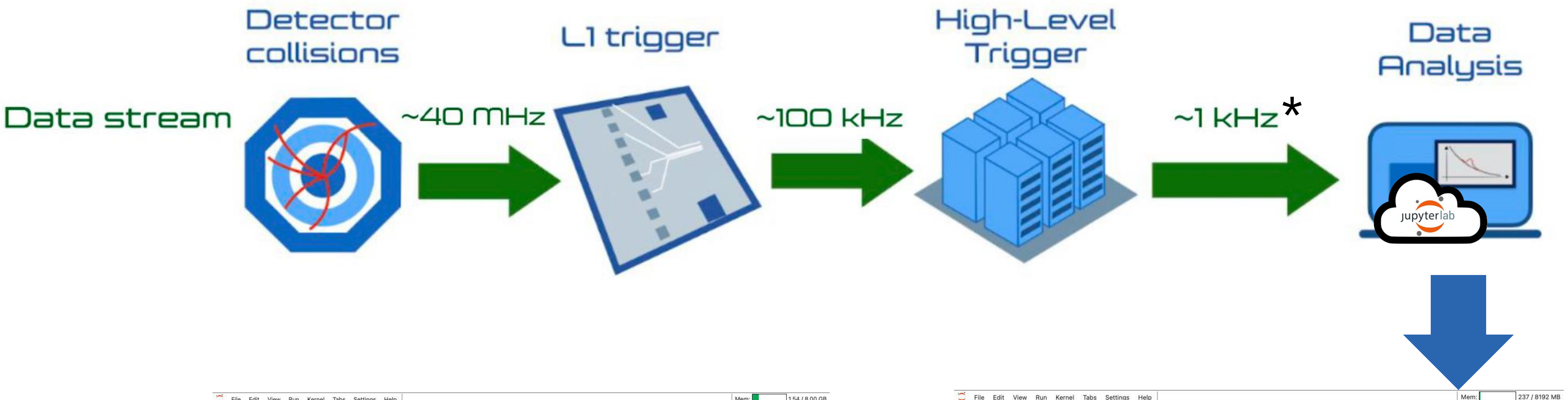
Higher rates of collision events → **Higher demand for computing and storage resources**

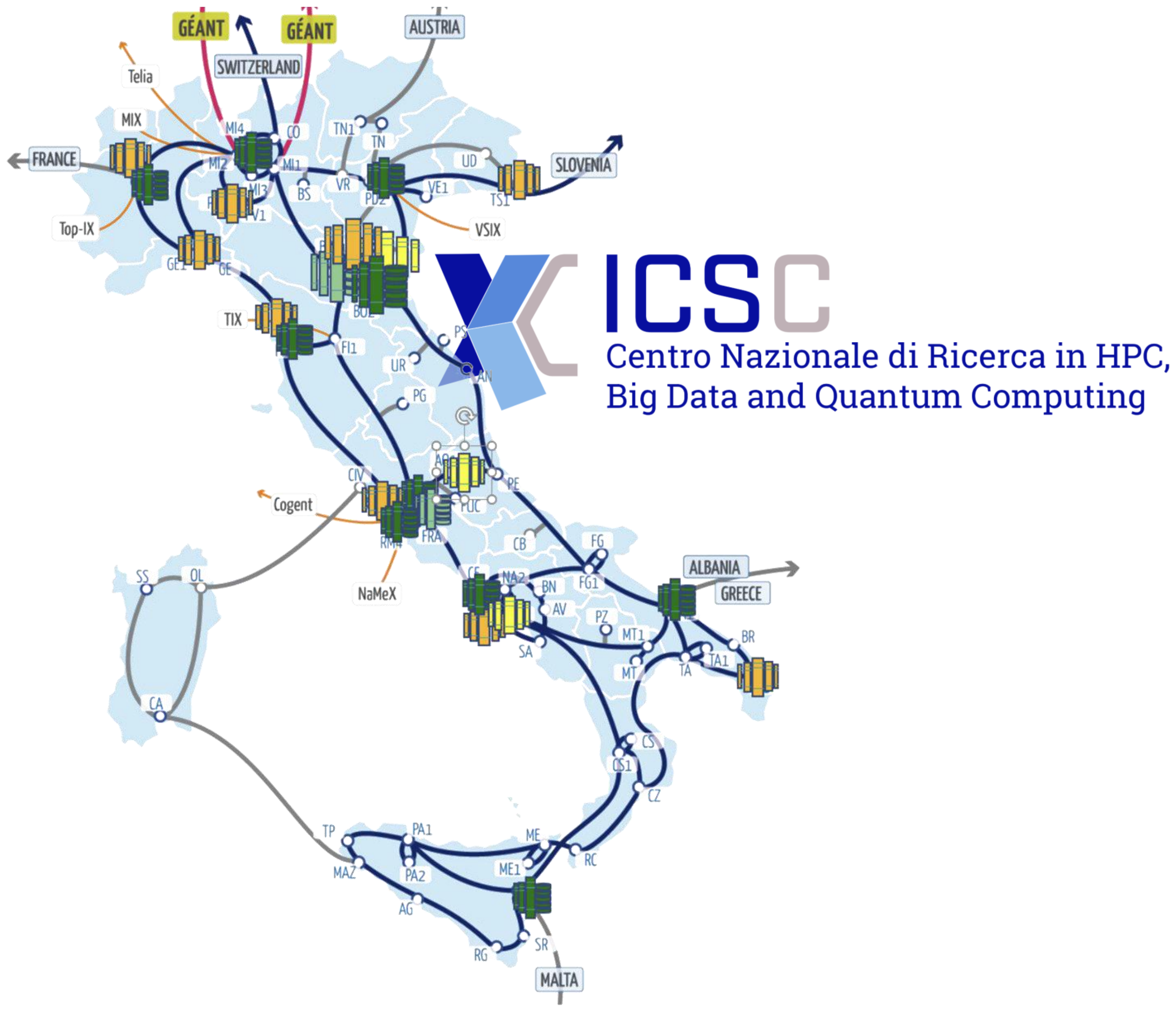
Need to:

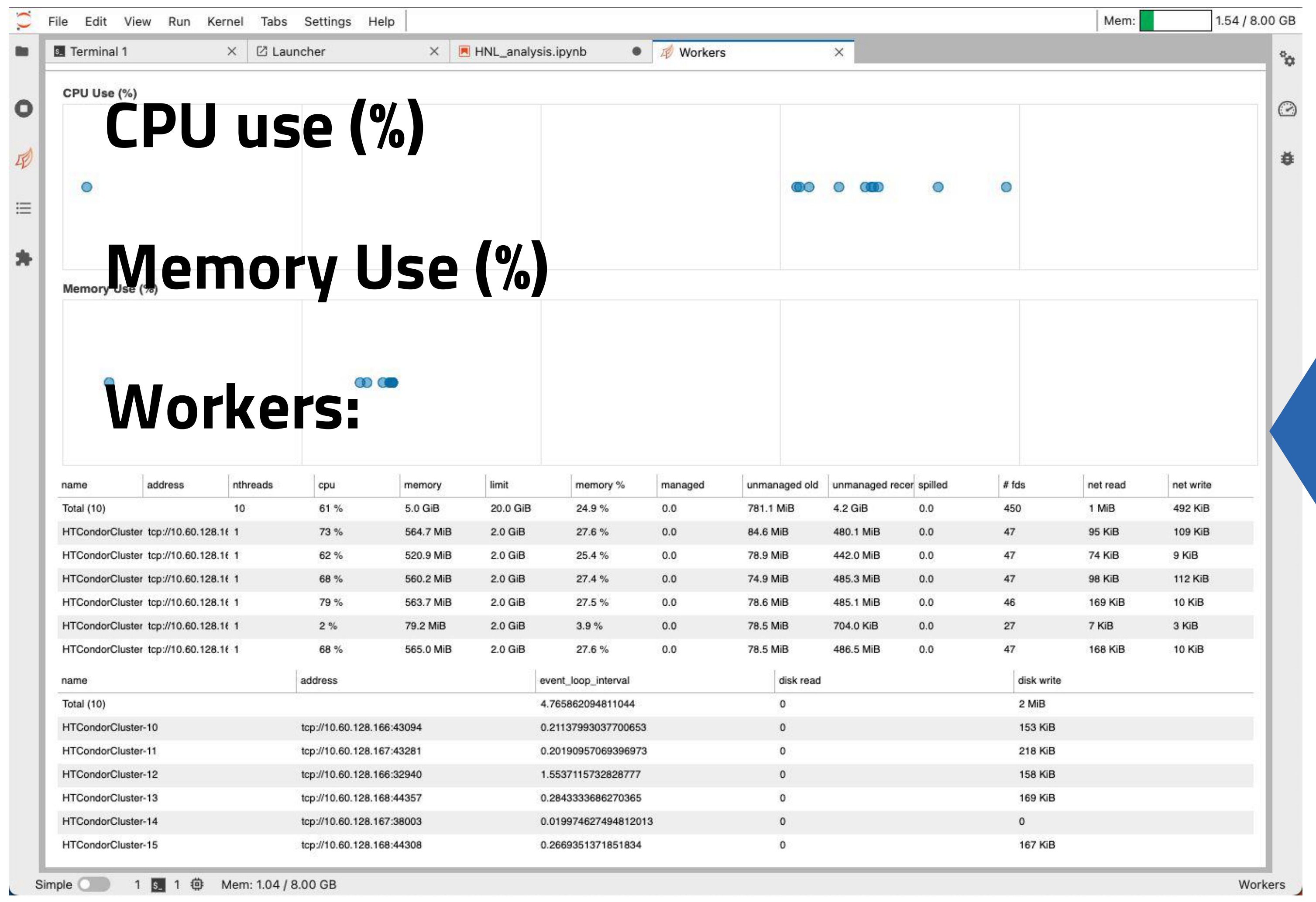
- Optimize the usage of CPU and storage
- Promote the usage of better data formats
- **Develop new analysis paradigms**!
- New software based on declarative programming and interactive workflows
- Distribute on geographically separated resources

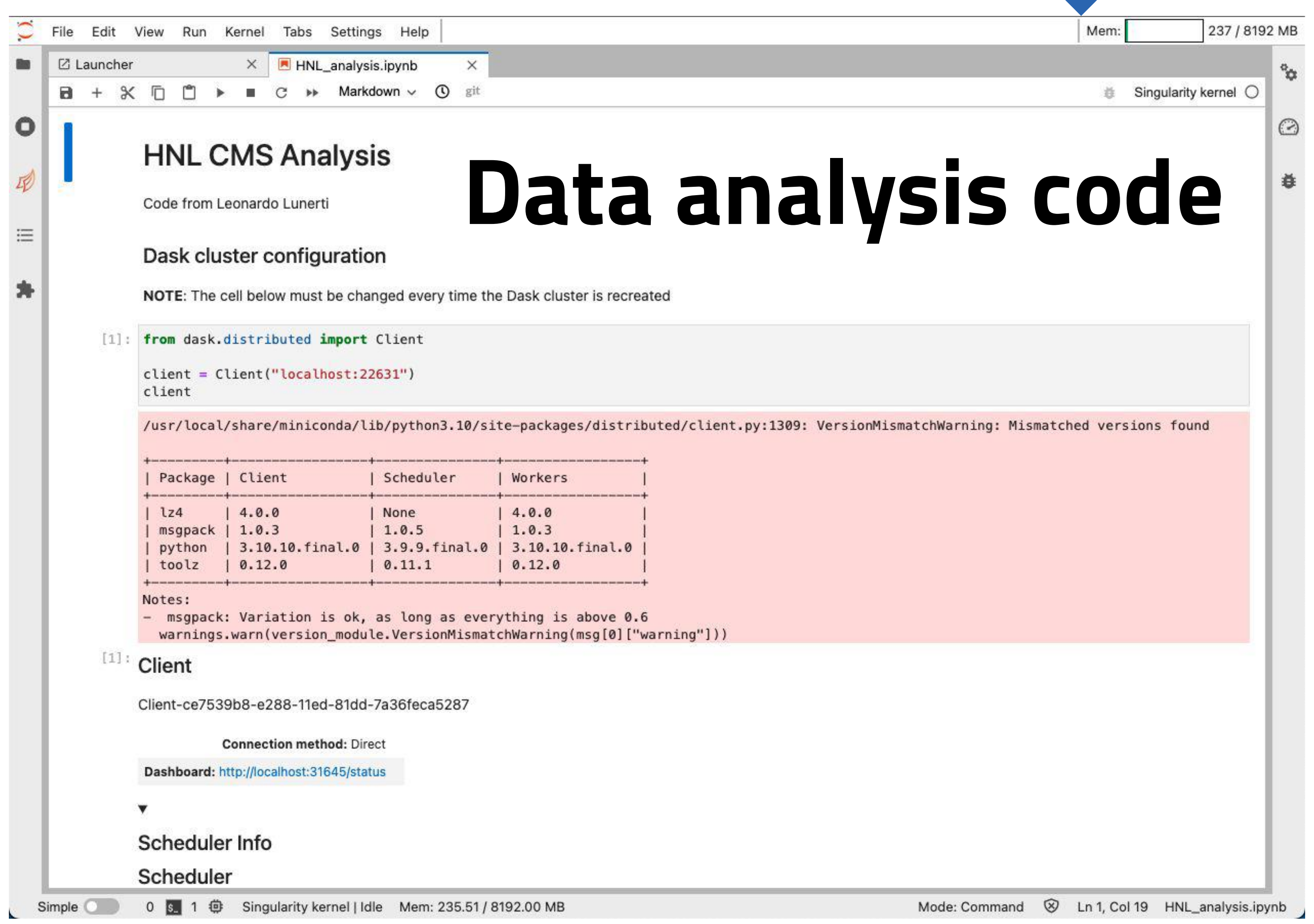
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

HEP data analysis with ICSC







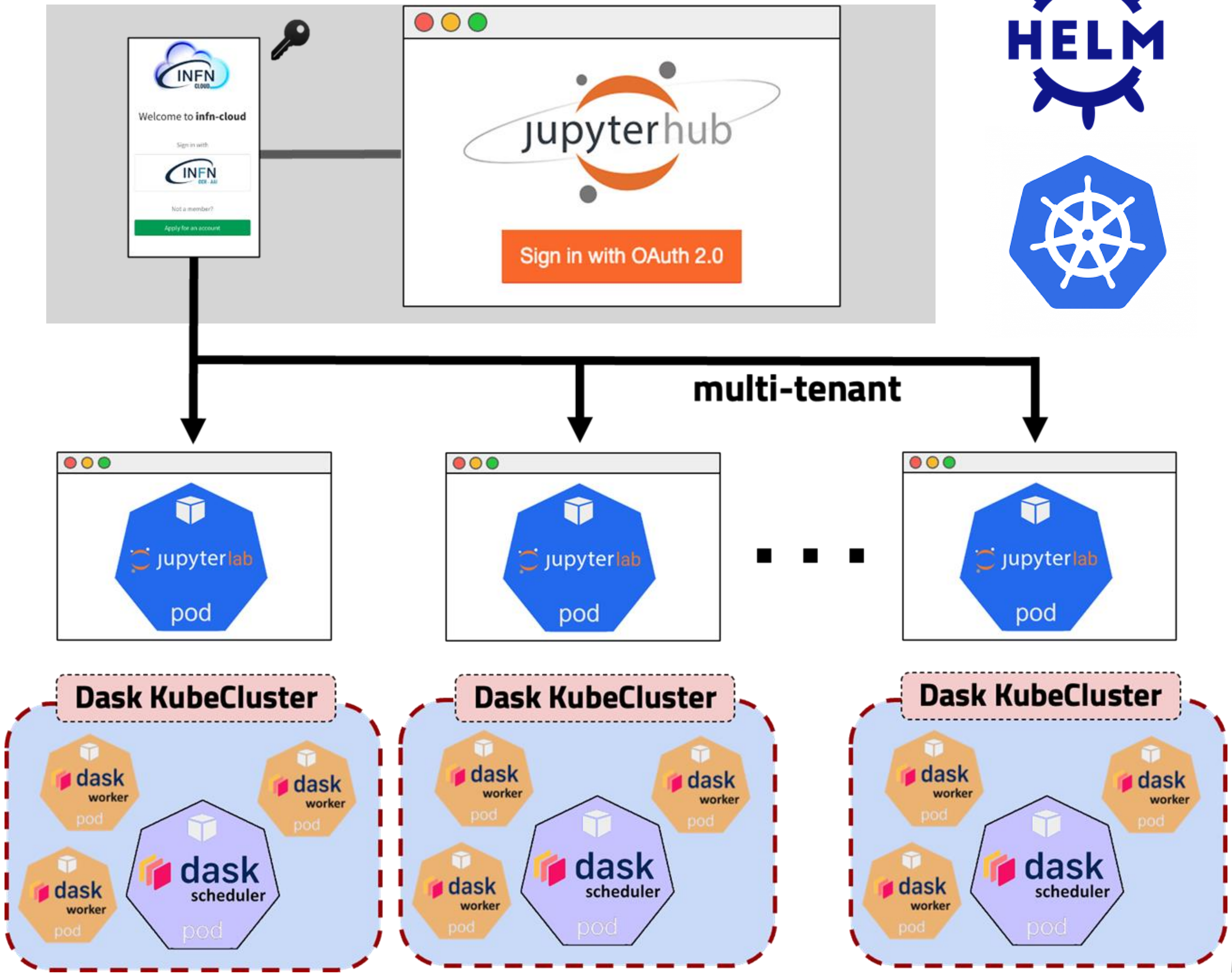


*trigger rates for previous Runs, now factor 3 ÷ 5 higher, will further scale in HL-LHC

High throughput data analysis platform

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

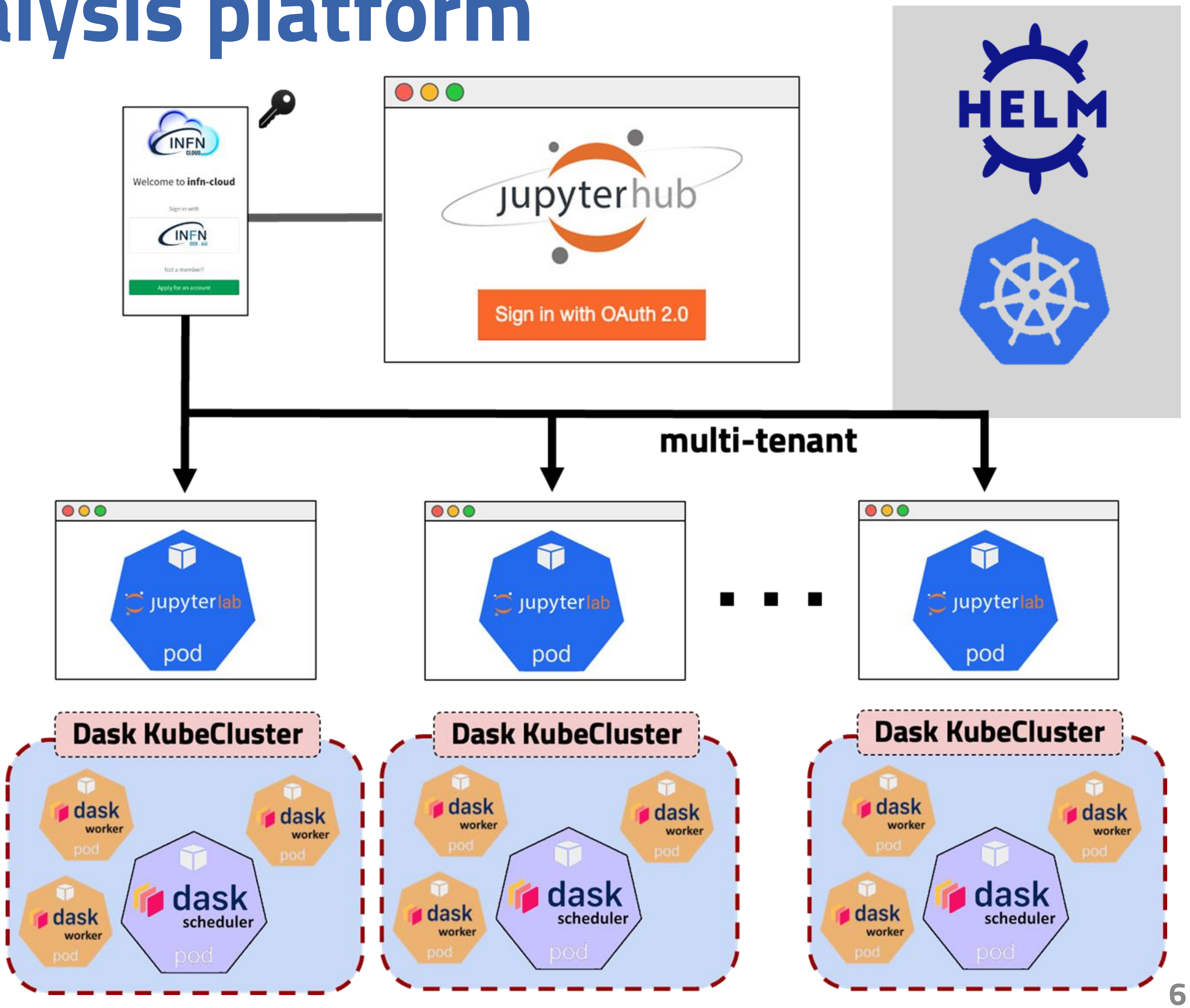
- The jupyterhub is deployed on a Kubernetes (k8s) cluster with **128 vCPUs and 258 GB**, divided into 8 nodes configured via RKE2



High throughput data analysis platform

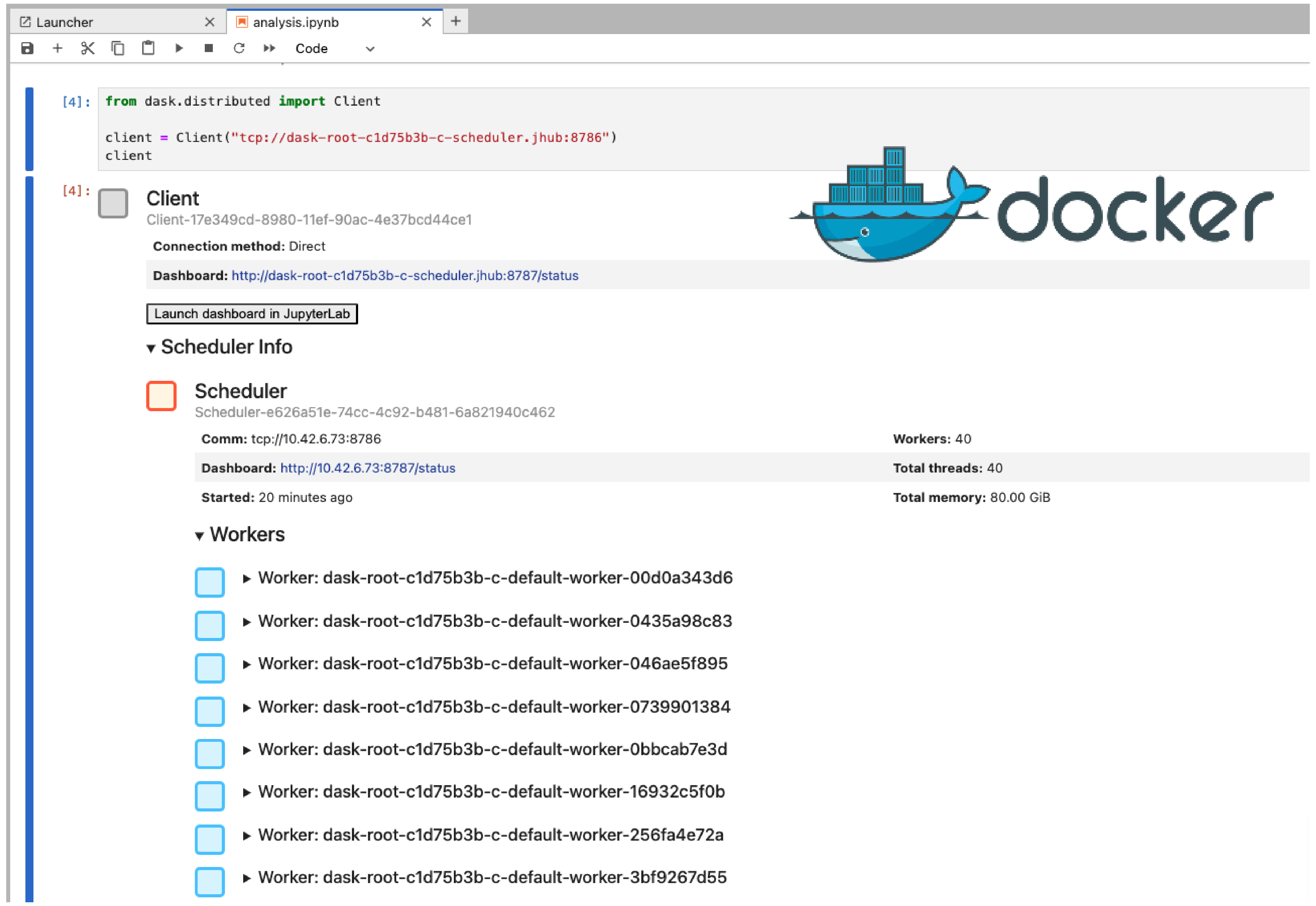
- The deployment of the Kubernetes resources is handled via HELM charts in the official Spoke2 Jhub HELM repo

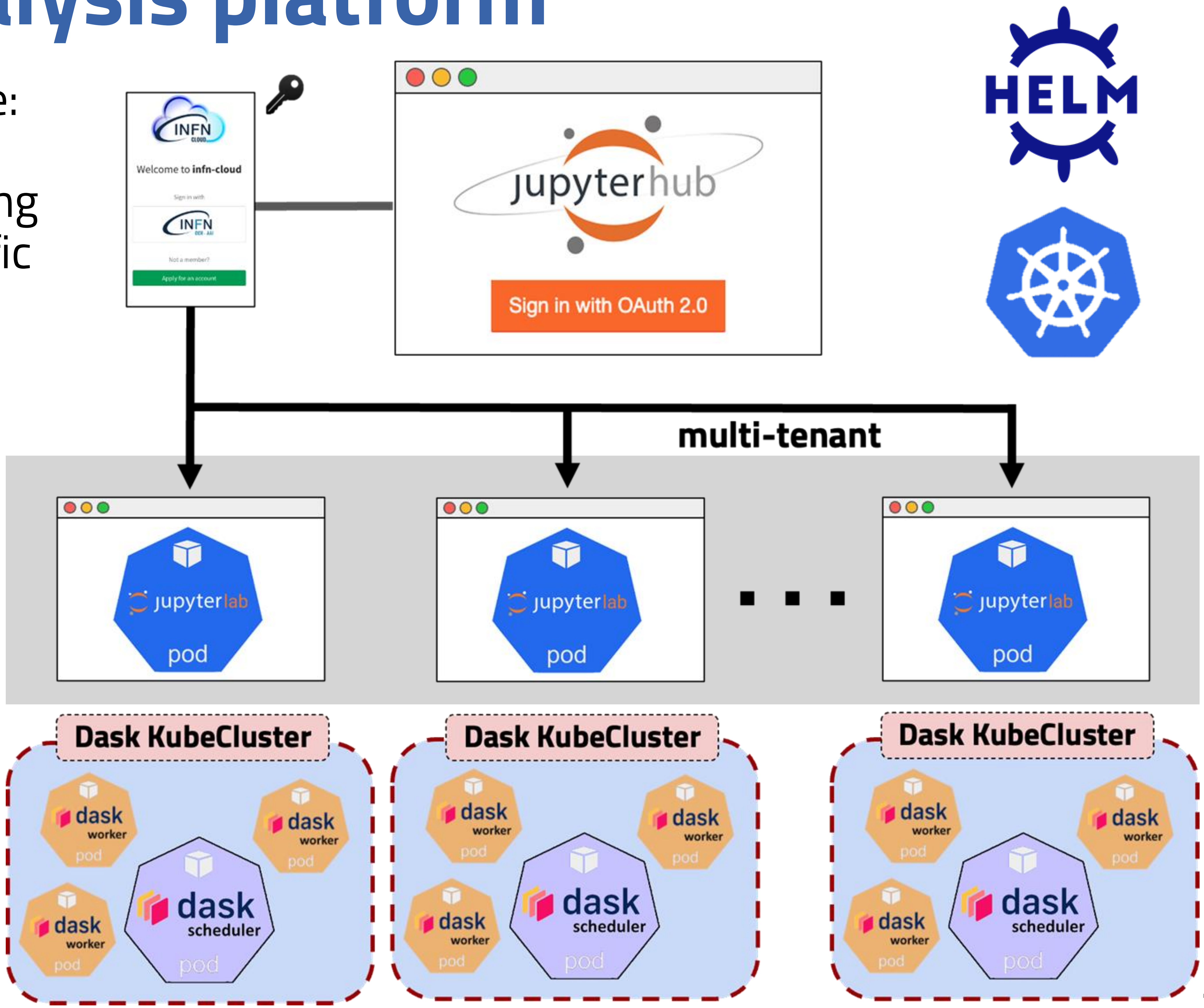
- This allows for a scalable and fault-tolerant deployment of the available resources



High throughput data analysis platform

- Jupyterlab interface is flexible and customizable:
  - Includes specific plugins (e.g. Dask)
  - Working environment highly customizable using Docker containers allowing for experiment specific software





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

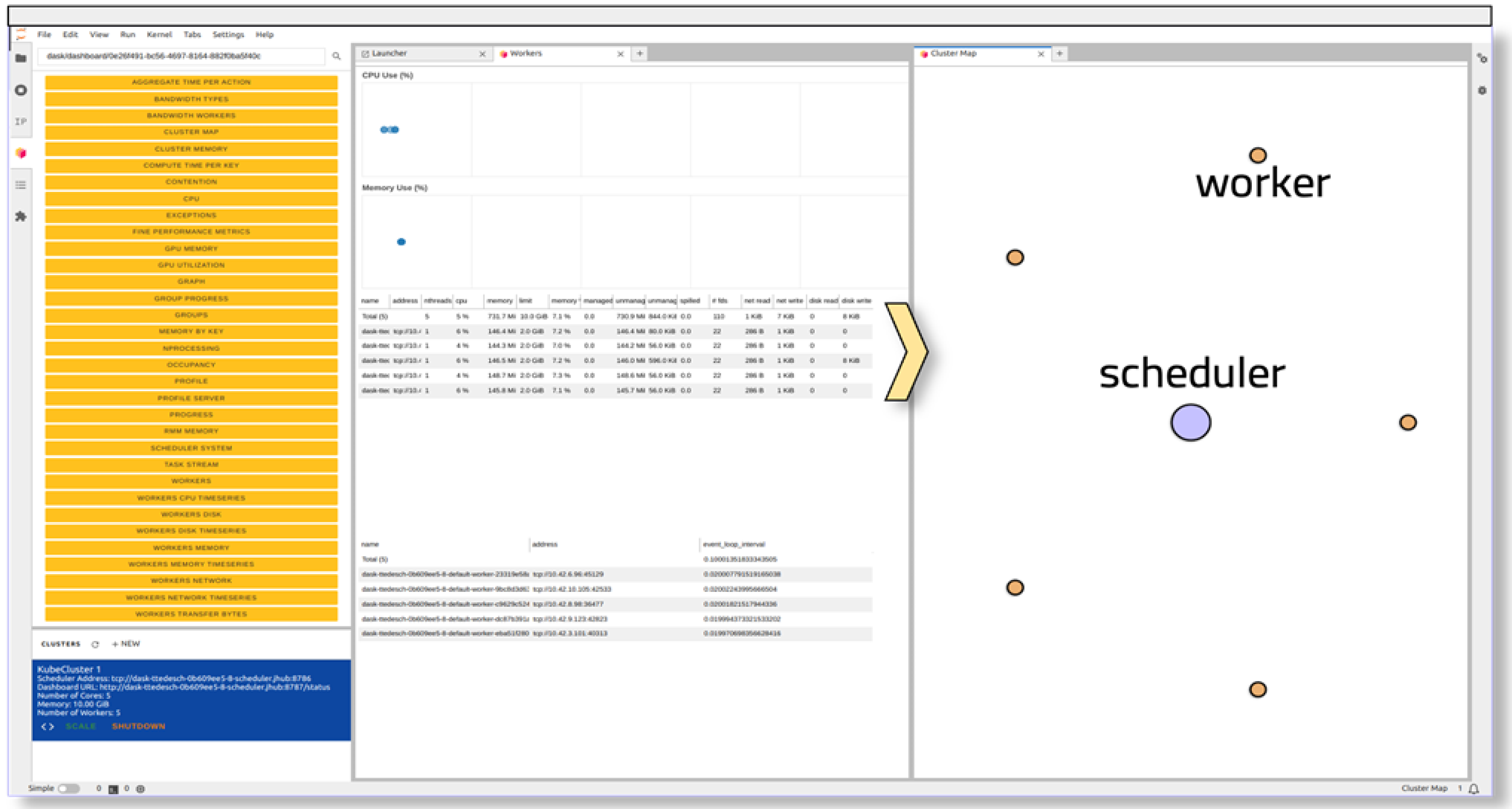
High throughput data analysis platform

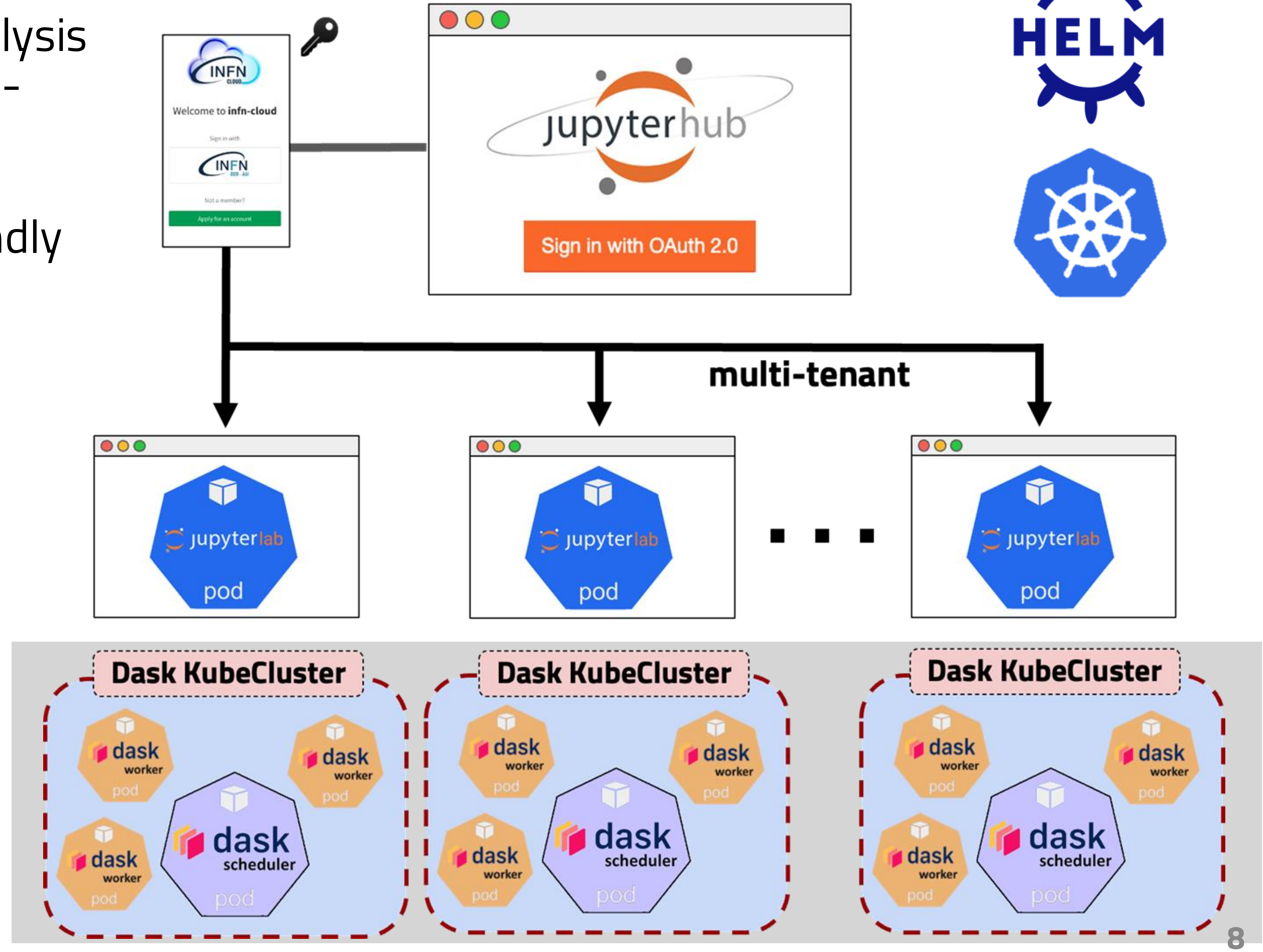
- Ideal environment for testing interactive analysis and validating new frameworks, e.g. the multi-threading features of ROOT RDataFrame

- The Dask Labextension provides a user-friendly monitoring dashboard

- More in the official docs!

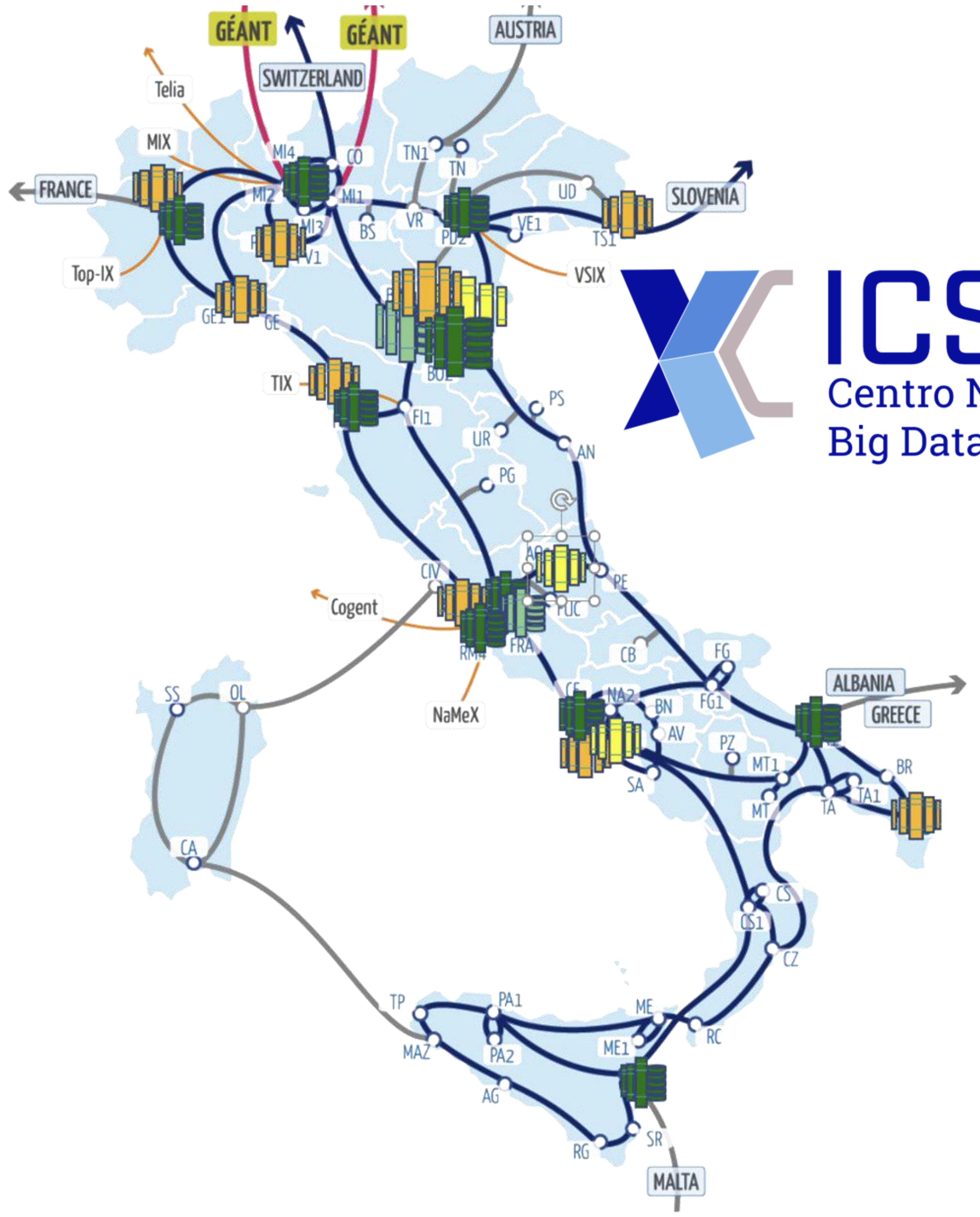
Dask Dashboard 1 **Monitoring workers** 2 **Cluster map**





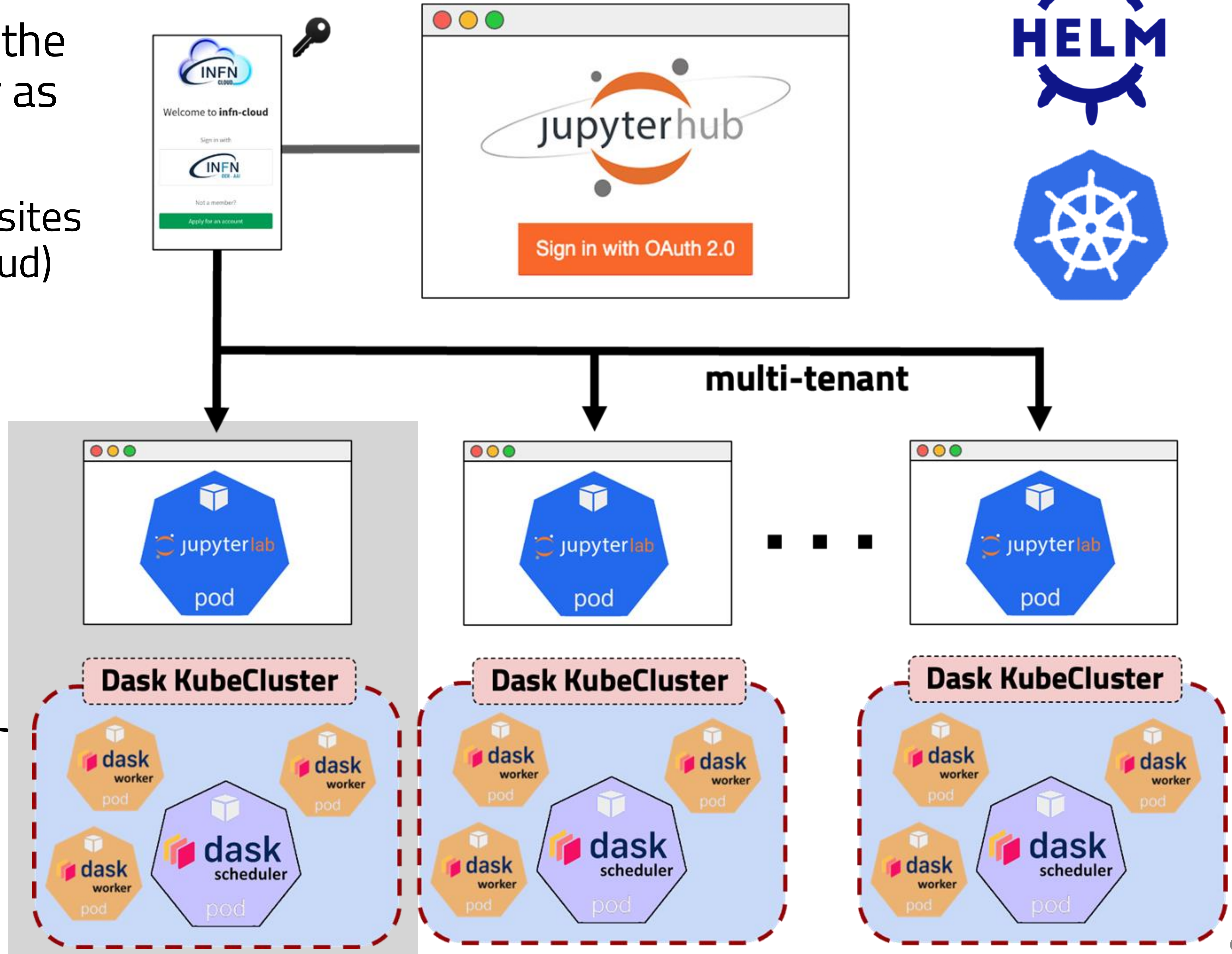
High throughput data analysis platform

- Offloading strategy: resources used to offload the computation are hosted in the same k8s cluster as the jupyter interface, via DASK KubeCluster
- **Under development:** spawning on multiple remote sites allowing for heterogeneous resources (HTC/HPC/Cloud) (see more in backup)



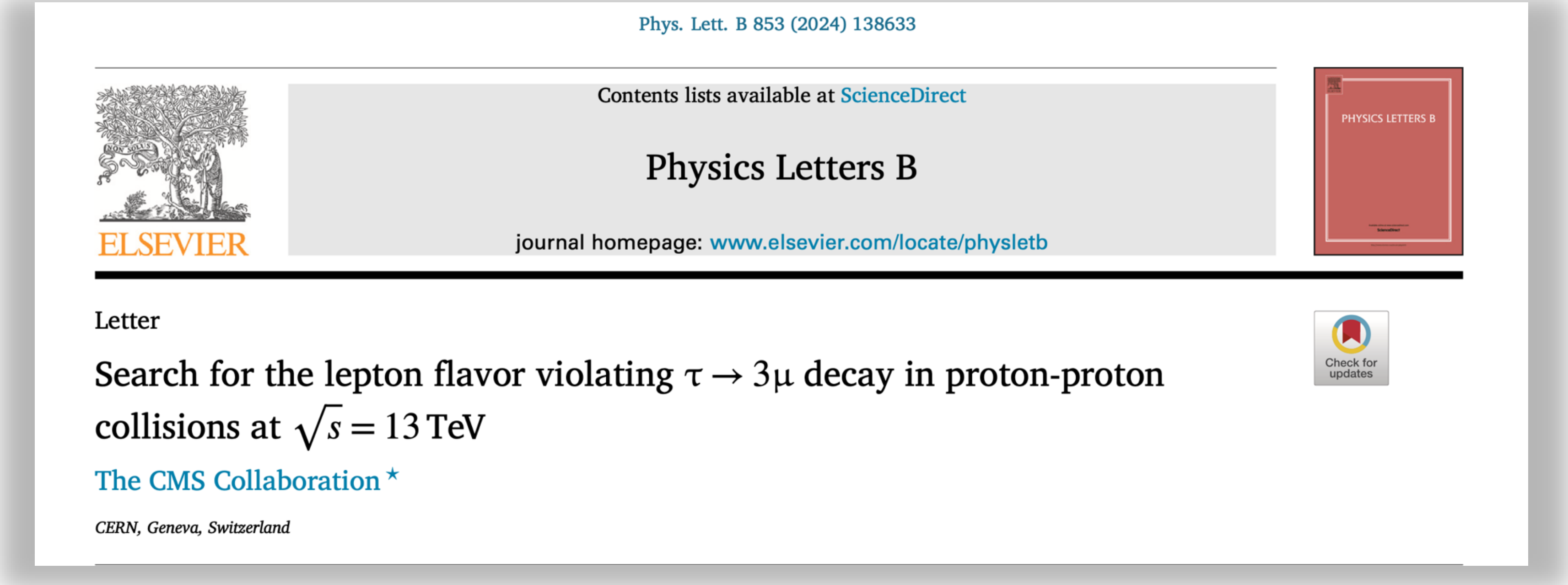
ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Offloading to..



Benchmark interactive analyses

CMS use-case


Phys. Lett. B 853 (2024) 138633

Contents lists available at ScienceDirect

Physics Letters B

journal homepage: www.elsevier.com/locate/physletb

ELSEVIER

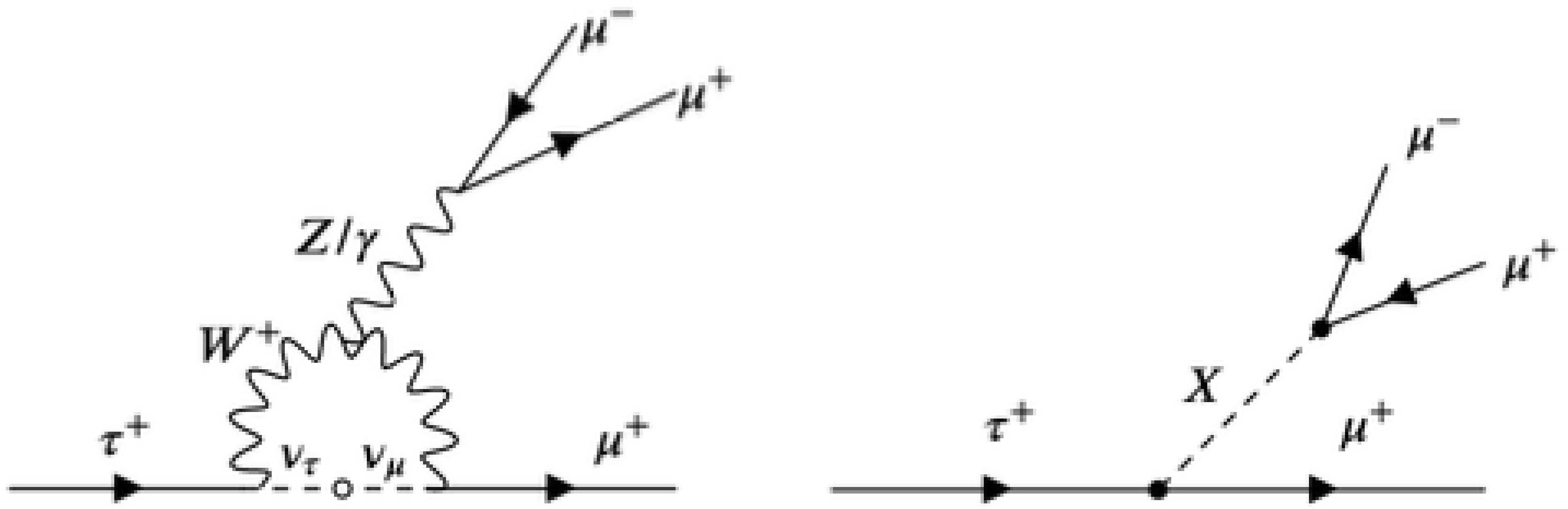
Letter

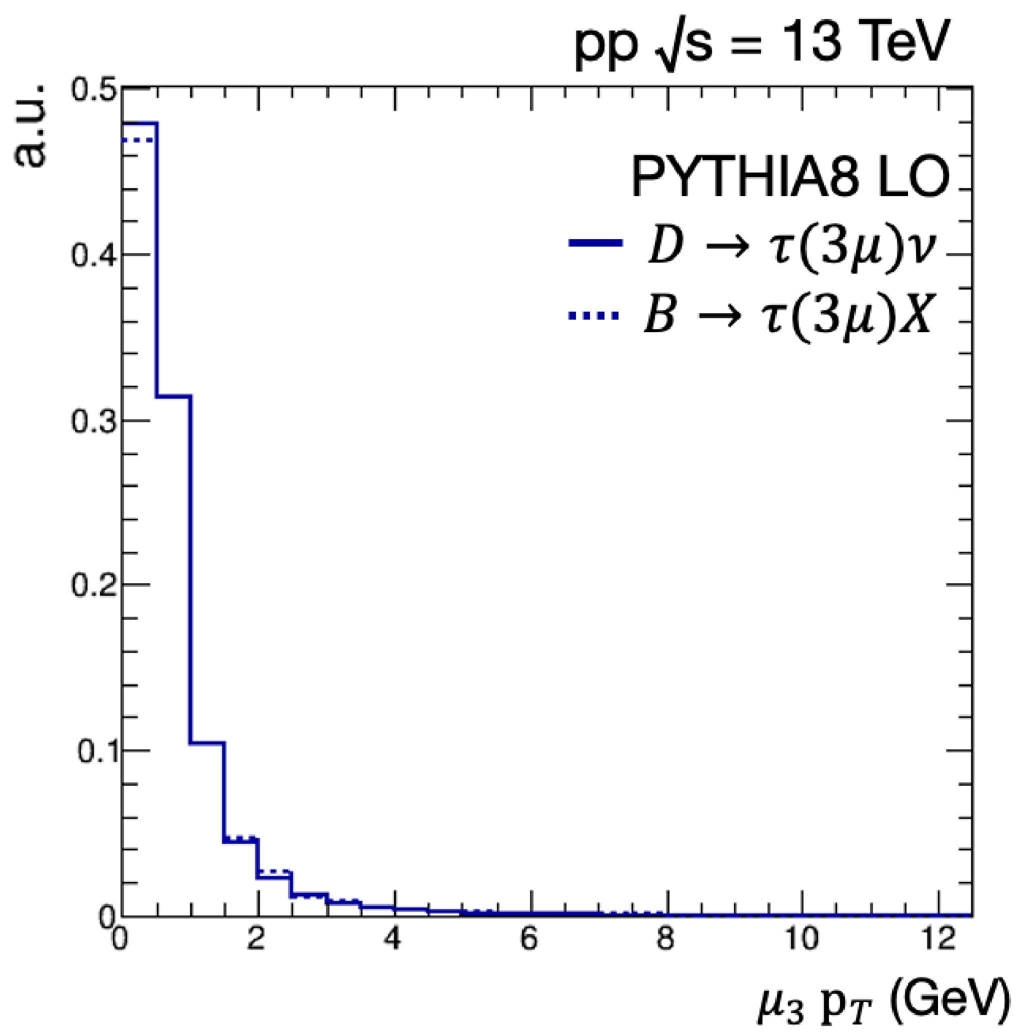
Search for the lepton flavor violating $\tau \to 3\mu$ decay in proton-proton collisions at $\sqrt{s} = 13$ TeV

The CMS Collaboration *

CERN, Geneva, Switzerland

Lepton Flavor Violation in the charged sector: $\tau \to 3\mu$





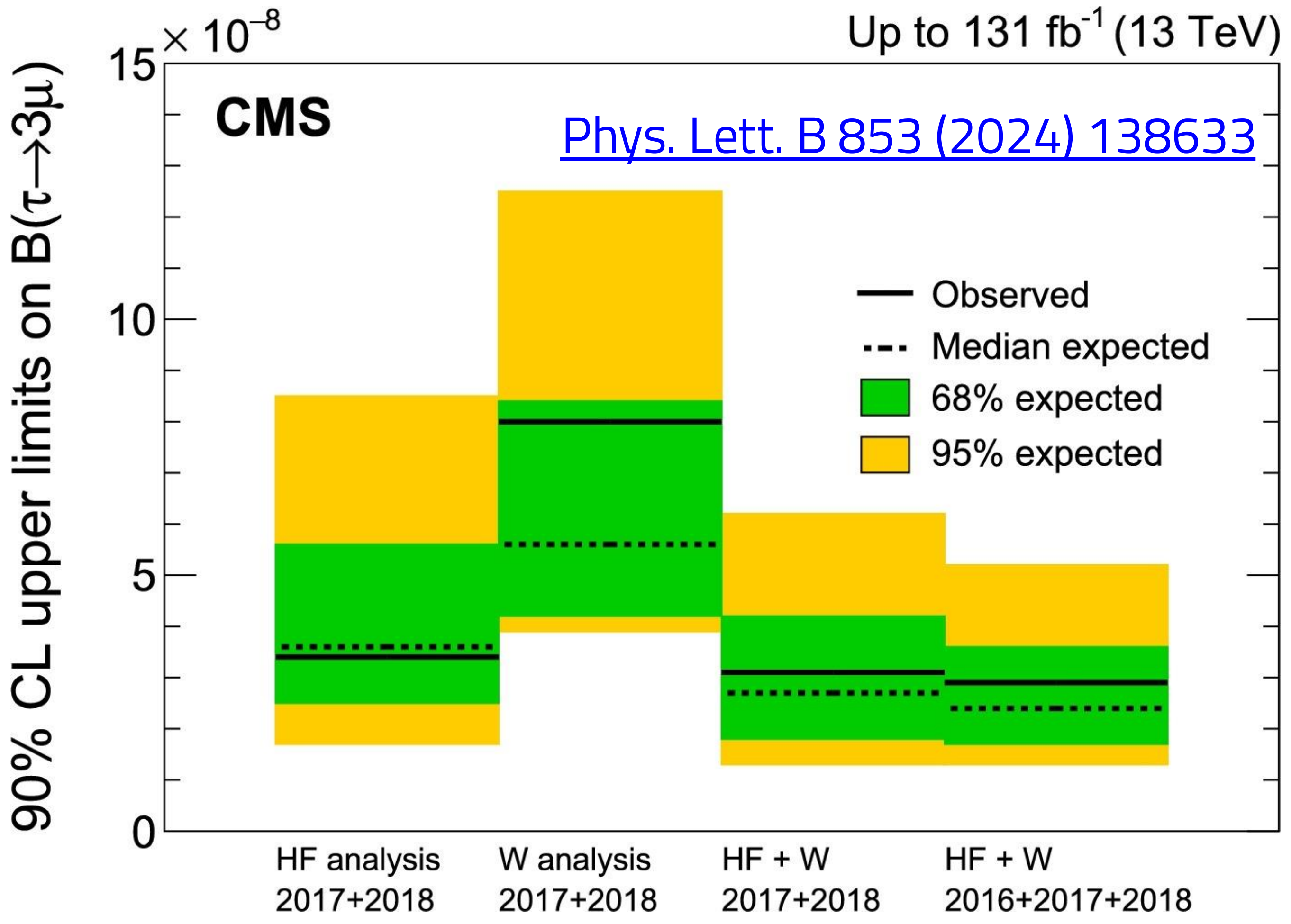
Search for $\boldsymbol{\tau \to 3\mu}$ decays, which have very small SM branching fractions $\mathrm{BR_{SM}} \sim \mathcal{O}(10^{-55})$, while being predicted with sizable BR in several BSM scenarios $\mathrm{BR_{BSM}} \sim \mathcal{O}(10^{-10} \div 10^{-8})$

- $\boldsymbol{\tau}$ leptons produced in D and B meson decays provide large statistics at LHC experiments, but are only accessible with **low-p_T muon triggers**

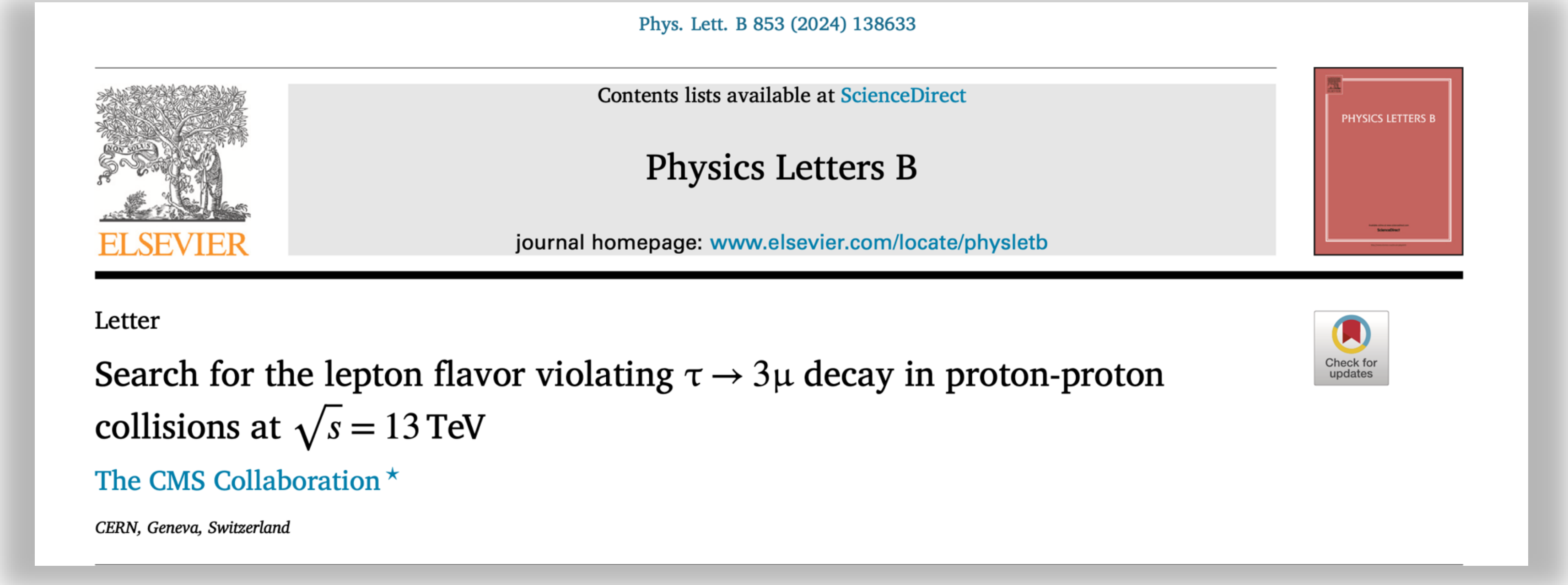
- Analysis of Run 2 data recently published, **stat. limited**

→ benefitting from inclusive low-p_T muon L1 trigger in **Run 3**

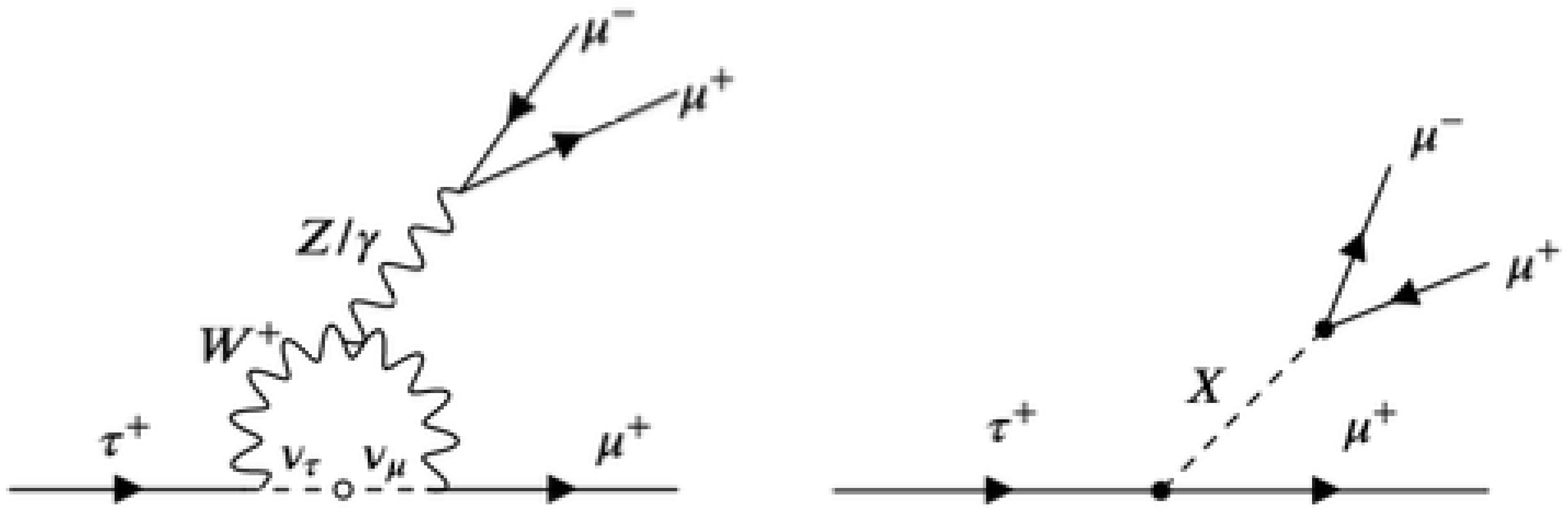
→ technical challenge: **new datasets are × 2 ÷ 3 times heavier**



CMS use-case


Phys. Lett. B 853 (2024) 138633
Contents lists available at ScienceDirect
Physics Letters B
journal homepage: www.elsevier.com/locate/physletb
Letter
Search for the lepton flavor violating $\tau \to 3\mu$ decay in proton-proton collisions at $\sqrt{s} = 13\,\text{TeV}$
The CMS Collaboration *
CERN, Geneva, Switzerland

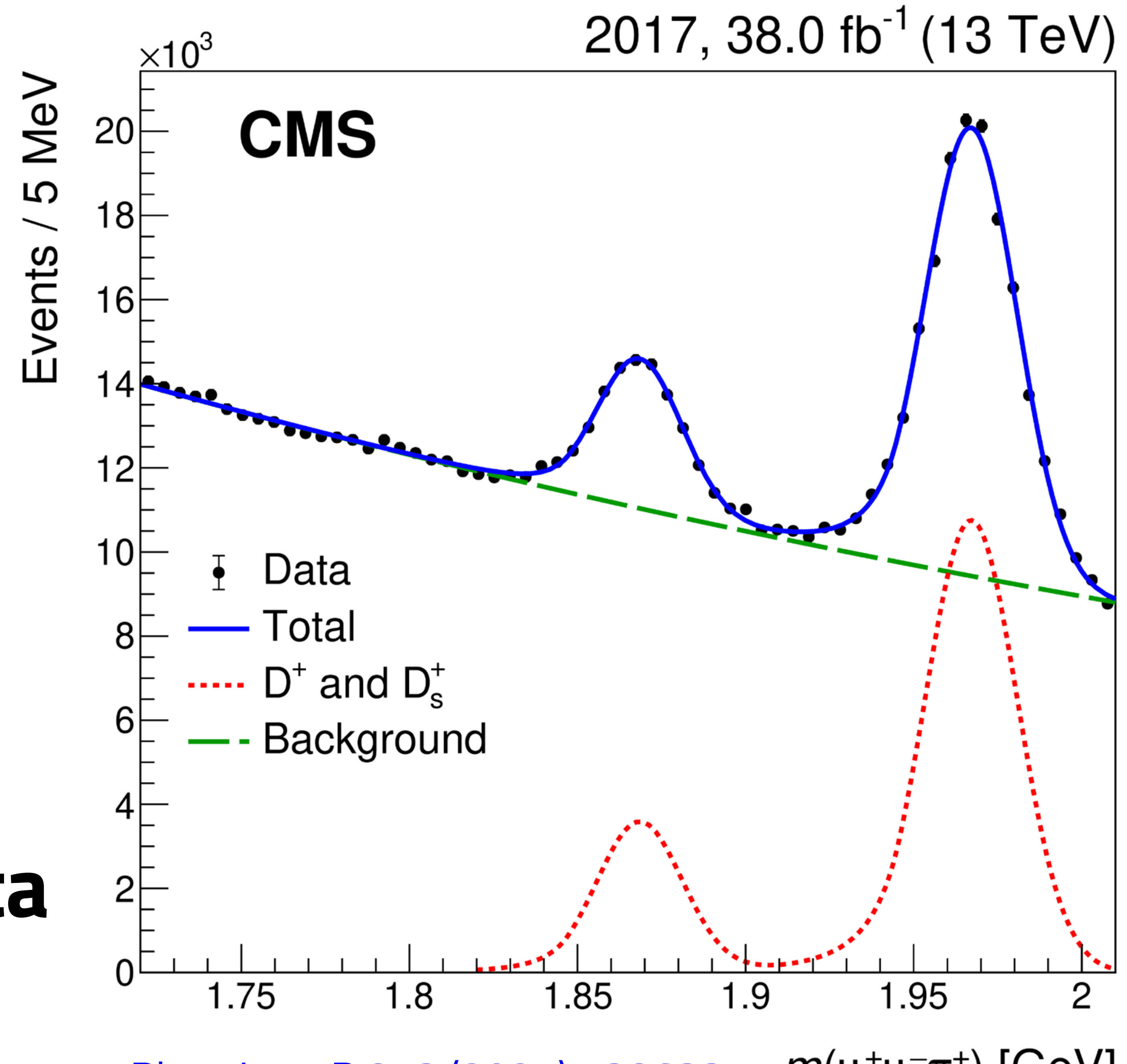
Lepton Flavor Violation in the charged sector: $\tau \to 3\mu$



Search for $\boldsymbol{\tau \to 3\mu}$ decays, which have very small SM branching fractions $\mathrm{BR}_{\mathrm{SM}} \sim \mathcal{O}(10^{-55})$, while being predicted with sizable BR in several BSM scenarios $\mathrm{BR}_{\mathrm{BSM}} \sim \mathcal{O}(10^{-10} \div 10^{-8})$

- $\boldsymbol{\tau}$ leptons produced in D and B meson decays provide large statistics at LHC experiments, but are only accessible with **low-p_T muon triggers**
- The normalisation channel used as a benchmark: $D_s^+ \to \phi(\mu\mu)\pi^+$

→ **cut-based analysis + mass fit for measuring the D_s^+ yield in data**



Phys. Lett. B 853 (2024) 138633

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis workflow

CMS Dataset

- Data collected by low p_T dimuon triggers
- MiniAOD data tier **centrally produced**

ROOT ntuples

- **Skimmed** data, events with 2μ+1track final state
- Saving only physics objects of interest
- **Plain data format**, ~ 5 GB / fb-1, stored on eos

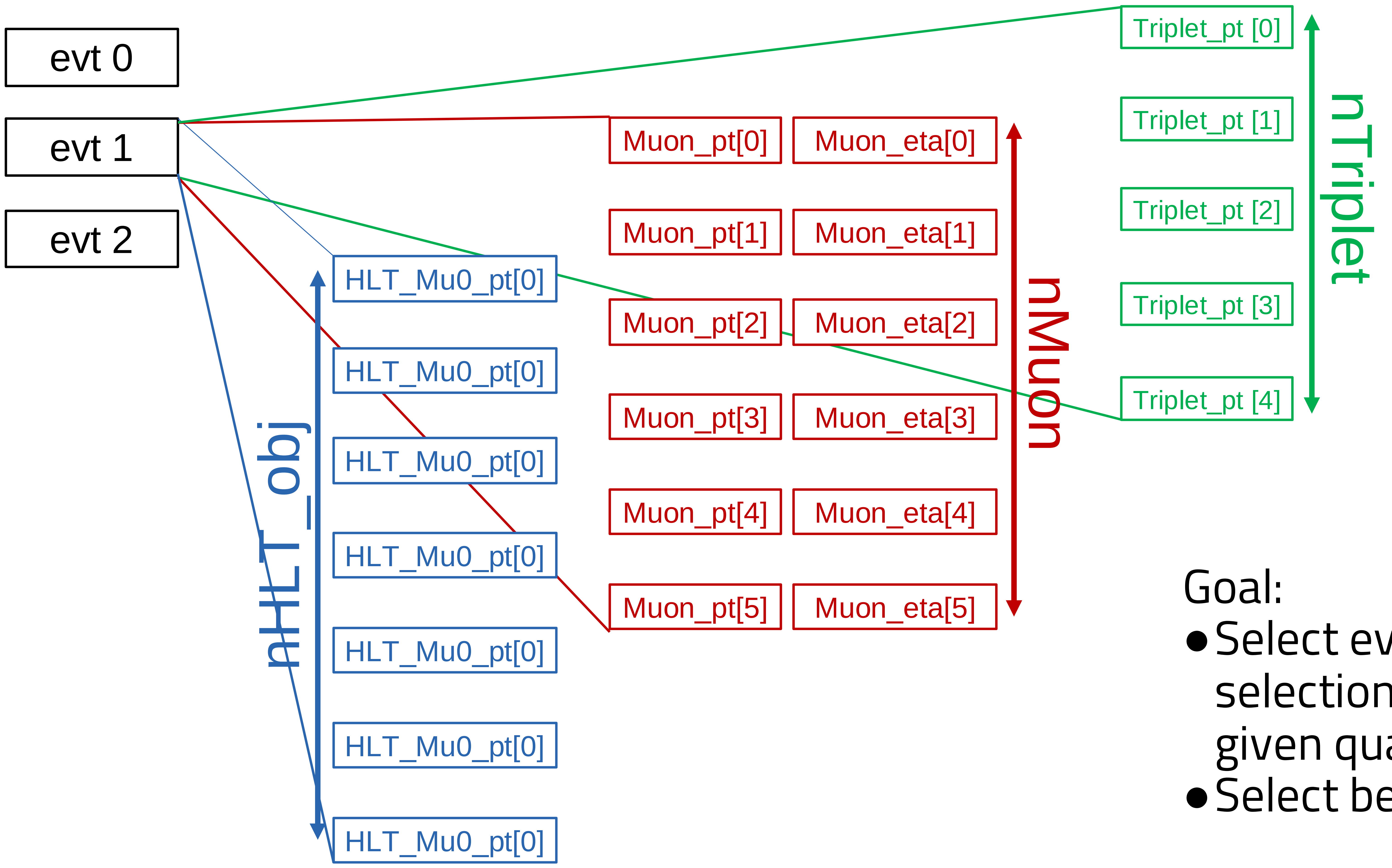
Analysis

- Define **high-level variables**
- Apply **scale factors** and corrections
- Apply **selections**, select best D_s candidate per event
- **Fit** the 2μ+1track invariant mass

- **Legacy: approach** Loop-based analysis implemented using ROOT TTree:MakeClass
  - split computation in batches of input files, run separately as HTCondor jobs, gather the output rootfiles

- **New**: Ntuples read as RDataFrame, almost all operations “lazy” → no loop triggered till the end
  - going distributed using ROOT RDataFrame distributed features, with Dask backend.

Ntuples are nanoAOD-like



Goal:
- Select events with triplets passing selections (e.g. containing muons with a given quality)
- Select best triplet per event in case >1 pass

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

Basic imports

[1]:
```python
import sys, os, time
start = time.time()
import json
import ROOT
```

```
Welcome to JupyROOT 6.30/02
```

Dask scheduler

[2]:
```python
from dask.distributed import Client, performance_report
```

[3]:
```python
Local = False

if Local:
    from dask.distributed import LocalCluster
    cluster = LocalCluster()
    client = Client(cluster.scheduler.address)
```

Now start new Dask cluster, scale the number of workers

[5]:
```python
# fill this cell with "<>" button of Dask labextension
```

[4]:
```python
from dask.distributed import Client

client = Client("tcp://dask-simonef-80b8d6c6-2-scheduler.jhub:8786")
client
```

[4]:

Client

Client-32e794b5-908b-11ef-82a4-3ee1209b75b0

Connection method: Direct

Dashboard: http://dask-simonef-80b8d6c6-2-scheduler.jhub:8787/status

Launch dashboard in JupyterLab

▸ Scheduler Info

- Define a Dask Client

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

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths

X509 proxy configuration

The `/tmp/x509up_u` file should be generated prior running the notebook using `voms-proxy-init -cert ../cert/usercert.pem -key ../cert/userkey.pem`

```
[9]: from distributed.diagnostics.plugin import UploadFile
     client.register_worker_plugin(UploadFile("/tmp/x509up_u0"))
```

```
/tmp/ipykernel_676/2847743139.py:2: DeprecationWarning: `Client.register_worker_plugin` has
been deprecated; please use `Client.register_plugin` instead
  client.register_worker_plugin(UploadFile("/tmp/x509up_u0"))
```

```
[9]: {'tcp://10.42.0.75:43259': {'status': 'OK'},
      'tcp://10.42.0.76:42581': {'status': 'OK'},
      'tcp://10.42.1.93:32875': {'status': 'OK'},
      'tcp://10.42.1.94:42729': {'status': 'OK'},
      'tcp://10.42.2.68:41105': {'status': 'OK'},
      'tcp://10.42.2.69:41157': {'status': 'OK'},
      'tcp://10.42.3.92:36155': {'status': 'OK'},
      'tcp://10.42.3.93:34933': {'status': 'OK'},
      'tcp://10.42.3.94:43541': {'status': 'OK'},
      'tcp://10.42.4.76:42331': {'status': 'OK'},
      'tcp://10.42.4.77:38303': {'status': 'OK'},
      'tcp://10.42.4.78:43961': {'status': 'OK'},
      'tcp://10.42.4.79:45559': {'status': 'OK'},
      'tcp://10.42.6.89:34275': {'status': 'OK'},
```

```
[10]: def set_proxy(dask_worker):
          import os
          import shutil
          working_dir = dask_worker.local_directory
          proxy_name = 'x509up_u0'
          os.environ['X509_USER_PROXY'] = working_dir + '/' + proxy_name
          os.environ['X509_CERT_DIR']="/cvmfs/grid.cern.ch/etc/grid-security/certificates/"
          return os.environ.get("X509_USER_PROXY"), os.environ.get("X509_CERT_DIR")
```

```
[11]: client.run(set_proxy)
```

```
[11]: {'tcp://10.42.0.75:43259': ('/tmp/dask-scratch-space/worker-ee12y_1w/x509up_u0',
        '/cvmfs/grid.cern.ch/etc/grid-security/certificates/'),
       'tcp://10.42.0.76:42581': ('/tmp/dask-scratch-space/worker-lp8yzbsh/x509up_u0',
        '/cvmfs/grid.cern.ch/etc/grid-security/certificates/'),
```

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

Declare custom C++ functions

```
[8]: text_file = open("Utilities.h", "r")
data = text_file.read()


def my_initialization_function():
    ROOT.gInterpreter.Declare('{}'.format(data))

ROOT.RDF.Experimental.Distributed.initialize(my_initialization_function)
```

```
numWorkers= len(client.scheduler_info()['workers'])
#npartitions = 2 * numWorkers
npartitions = 2 * nfiles

print("Number of workers is: {}".format(numWorkers))
print("Number of total partitions is: {}".format(npartitions))

df = ROOT.RDF.Experimental.Distributed.Dask.RDataFrame(treename, chain, daskclient=client)
```

'root://eosuser.cern.ch//eos/mypath/*.root'

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths
- Declare useful C++ functions and define Distributed.Dask.RDataFrame

$D_s^+ \rightarrow \phi(\mu\mu)\pi^+$ analysis code

```
# Selections on triplets
# 2 -> 2mu+track candidate mass in (1.62-2.02)GeV
# 3 -> at least 2 track associated with PV
# 4 -> Significance of BS-SV distance in the transverse plane > 2
triplet_selection = "Triplet2_Mass>1.62 && Triplet2_Mass<2.02 && \
                     RefittedPV2_NTracks > 1 && \
                     FlightDistBS_SV_Significance > 2 "

# Events with at least one good candidate
df = df.Define("triplet_mask1", triplet_selection).Filter("ROOT::VecOps::Sum(triplet_mask1) >0")
```

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths
- Declare useful C++ functions and define Distributed.Dask.RDataFrame

Some steps of the analysis:

- Apply selections on branches with size nTriplet

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths
- Declare useful C++ functions and define Distributed.Dask.RDataFrame

Some steps of the analysis:

- Apply selections on branches with size nTriplet
- Match other branches (e.g. Muon_*) with Triplet_* and apply selections

```
# Selections on triplets
# 2 -> 2mu+track candidate mass in (1.62-2.02)GeV
# 3 -> at least 2 track associated with PV
# 4 -> Significance of BS-SV distance in the transverse plane > 2
triplet_selection = "Triplet2_Mass>1.62 && Triplet2_Mass<2.02 && \
                     RefittedPV2_NTracks > 1 && \
                     FlightDistBS_SV_Significance > 2 "

# Events with at least one good candidate
df = df.Define("triplet_mask1", triplet_selection).Filter("ROOT::VecOps::Sum(triplet_mask1) >0")
```

```
# Find index in "Muon_" and "Track_" branches
df = df.Define("Mu01_index", "match(MuonPt, Mu01_Pt)")
df = df.Define("Mu02_index", "match(MuonPt, Mu02_Pt)")
df = df.Define("Tr_index",   "match(MuonPt, Tr_Pt)")

# 7 -> Apply Muon ID Global and Particle Flow
df = df.Define("Mu01_ID", "muon_id(Mu01_index, Muon_isGlobal && Muon_isPF)")
df = df.Define("Mu02_ID", "muon_id(Mu02_index, Muon_isGlobal && Muon_isPF)")

# 8 -> IP(track, BS) z direction < 20 cm and xy direction < 0.3 cm
df = df.Define("Tr_IPcut", "muon_id(Tr_index, (Track_dz<20 && Track_dxy<0.3) )")
df = df.Define("triplet_mask4", "Mu01_ID && Mu02_ID && Tr_IPcut").Filter("ROOT::VecOps::Sum(triplet_mask4)>0")
```

```
RVec<int> match(ROOT::VecOps::RVec<double> branch1, ROOT::VecOps::RVec<double> branch2){
//returns vector of indeces such that branch2[index]=branch1
    RVec<int> index;
    for(unsigned i = 0; i<branch1.size(); i++){
      auto idx = std::find(branch2.begin(), branch2.end(), branch1.at(i));
      if( idx != branch2.end()) index.push_back(std::distance(branch2.begin(), idx));
      else index.push_back(-99);
    }
    return index;
}
```

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

```
# Selections on triplets
# 2 -> 2mu+track candidate mass in (1.62-2.02)GeV
# 3 -> at least 2 track associated with PV
# 4 -> Significance of BS-SV distance in the transverse plane > 2
triplet_selection = "Triplet2_Mass>1.62 && Triplet2_Mass<2.02 && \
                     RefittedPV2_NTracks > 1 && \
                     FlightDistBS_SV_Significance > 2 "

# Events with at least one good candidate
df = df.Define("triplet_mask1", triplet_selection).Filter("ROOT::VecOps::Sum(triplet_mask1) >0")
```

```
# Find index in "Muon_" and "Track_" branches
df = df.Define("Mu01_index", "match(MuonPt, Mu01_Pt)")
df = df.Define("Mu02_index", "match(MuonPt, Mu02_Pt)")
df = df.Define("Tr_index",   "match(MuonPt, Tr_Pt)")

# 7 -> Apply Muon ID Global and Particle Flow
df = df.Define("Mu01_ID", "muon_id(Mu01_index, Muon_isGlobal && Muon_isPF)")
df = df.Define("Mu02_ID", "muon_id(Mu02_index, Muon_isGlobal && Muon_isPF)")

# 8 -> IP(track, BS) z direction < 20 cm and xy direction < 0.3 cm
df = df.Define("Tr_IPcut", "muon_id(Tr_index, (Track_dz<20 && Track_dxy<0.3) )")
df = df.Define("triplet_mask4", "Mu01_ID && Mu02_ID && Tr_IPcut").Filter("ROOT::VecOps::Sum(triplet_mask4)>0")
```

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths
- Declare useful C++ functions and define Distributed.Dask.RDataFrame

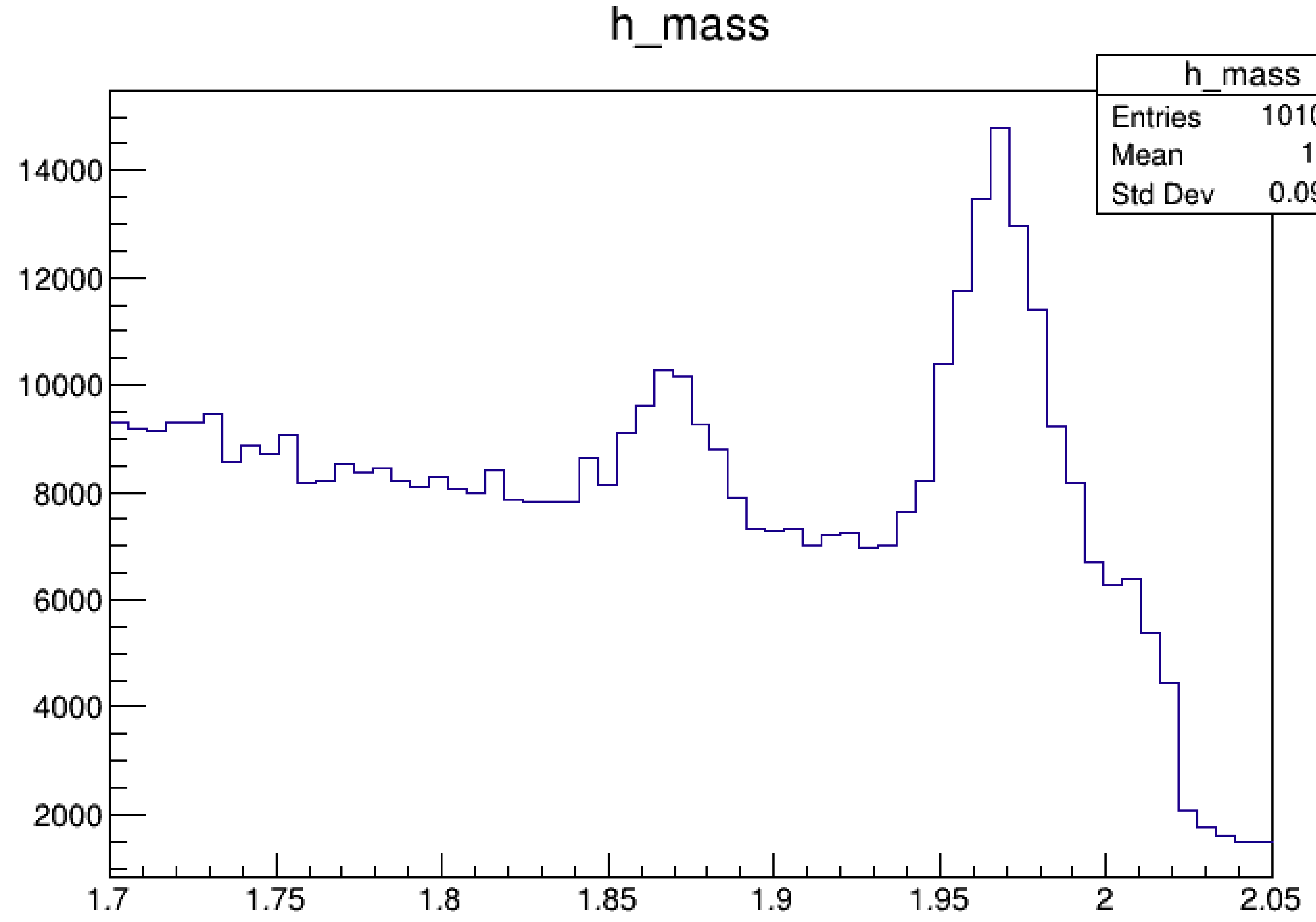
Some steps of the analysis:

- Apply selections on branches with size nTriplet
- Match other branches (e.g. Muon_*) with Triplet_* and apply selections
- Keep best mu mu track candidate

```
# Keep best candidate based on vertex chi2
df = df.Define("BestTriplet_index", "bestcandidate(TripletVtx2_Chi2)")
df = df.Define("BestTriplet_mass", "flattening(Triplet2_Mass, BestTriplet_index)")
```

$D_s^+ \to \phi(\mu\mu)\pi^+$ analysis code

```
# Create a histogram from `x` and draw it
with performance_report(filename="my_report.html"):
  h = df.Histo1D(("h_mass", "h_mass", 62, 1.70, 2.05), "BestTriplet_mass")
  c =  ROOT.TCanvas()
  h.Draw("hist")
  c.Draw()
```



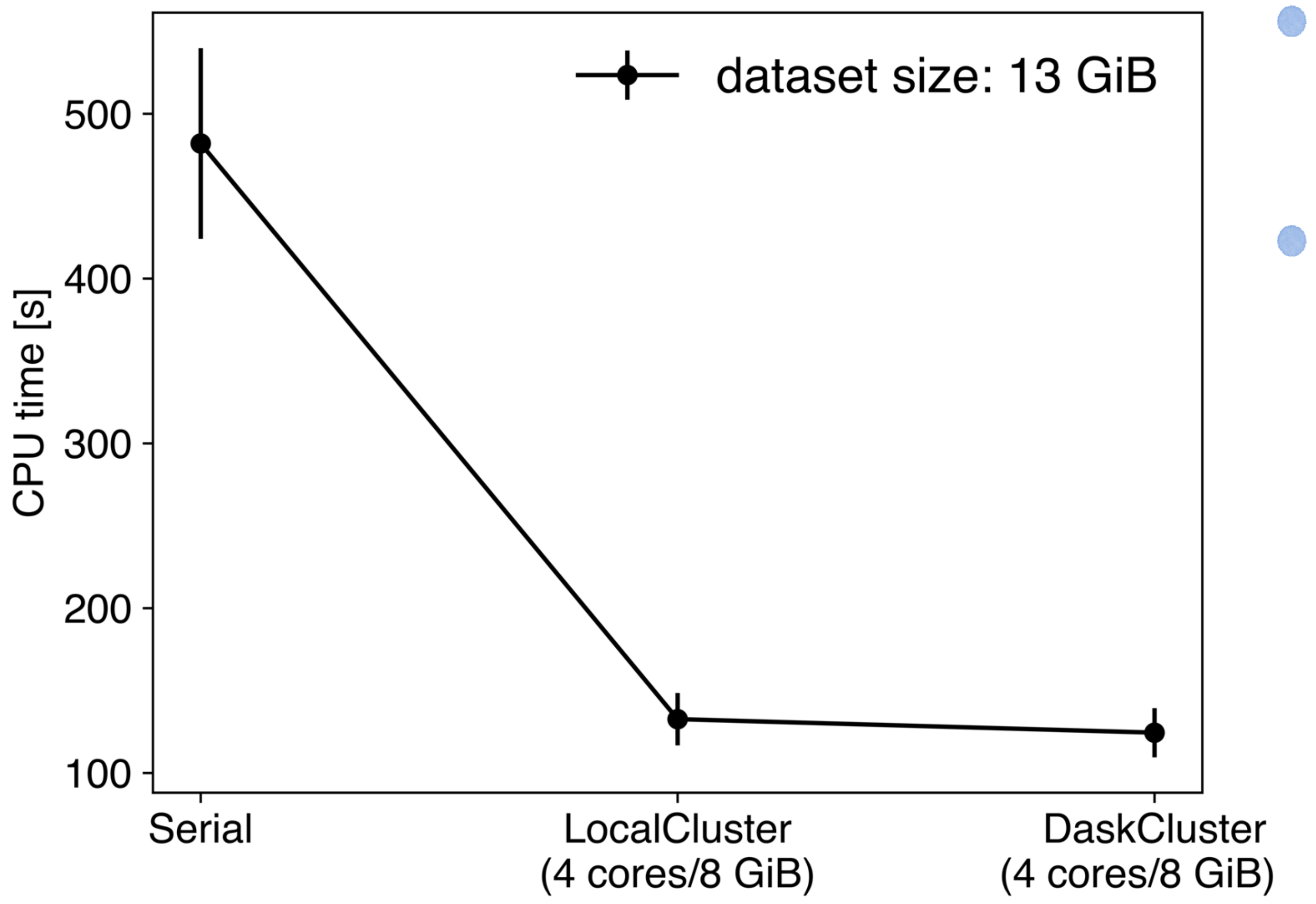

```
# Save output for further processing: snapshot saves on workers!
df_out = df.Snapshot("ntuple", "out.root", ["BestTriplet_mass"])

np_out = df.AsNumpy(columns=["BestTriplet_mass"]) #workers stay "in-memory" forever
```

- Define a Dask Client
- Load X509 user proxy to Dask workers and set env paths
- Declare useful C++ functions and define Distributed.Dask.RDataFrame

Some steps of the analysis:
- Apply selections on branches with size nTriplet
- Match other branches (e.g. Muon_*) with Triplet_* and apply selections
- Keep best mu mu track candidate

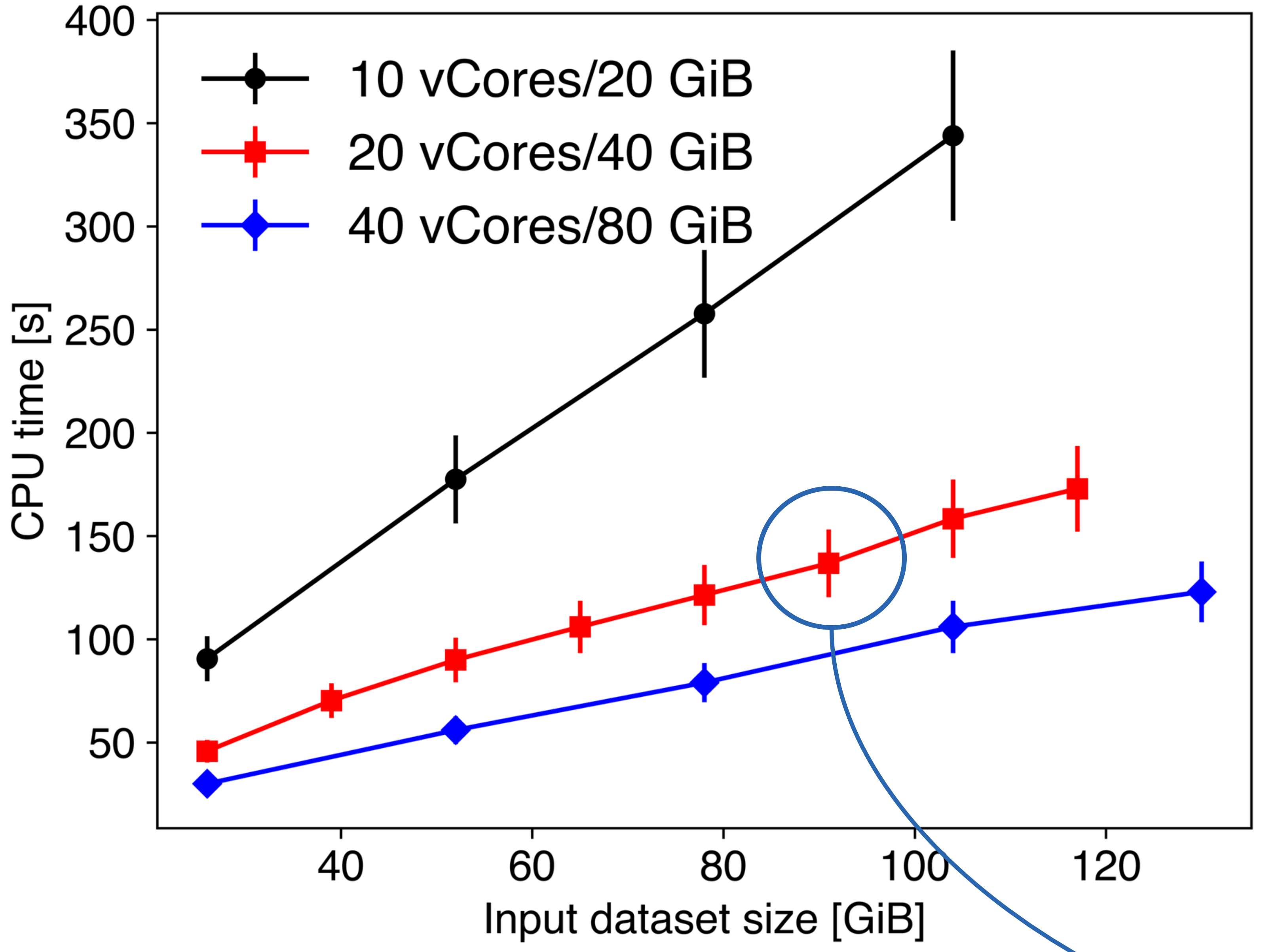
Output/results:
- Drawing or counting out of the final df triggers the computation → smooth
- Snapshoting the final df for further analysis: many "out.root" files are saved in the workers and need to be copied back
- Tried "AsNumpy" as an alternative → workers don't finish computation

Preliminary results



- Significant improvement in execution time *wrt* the standard/serial approach
- The facility allows for dynamically scaling the resources, here testing the performance at fixed #cores and memory, varying the dataset size

- Stress test at high CPU and memory occupancy
- Stable performance, linearly scaling with the input dataset size
- Dataset size ~ 100 GiB is representative of ~15 /fb of Run3 data for this specific analysis



Feedback from a user point of view:

- Implementing the analysis in RDataFrame was easy, looking at tutorials and forum
- Interfacing RDataFrame and Dask → only few lines of code, no debugging needed
- It would be nice to have a “distributed” version of Snapshot for harvesting the outputs from the workers (or maybe I missed something here? ☺)

- This AF is under testing, we presently have ~10 beta users from CMS, ATLAS, FCC (mostly from CMS tough)
- Will reach a larger audience after expanding the current pool of resources

Conclusions & Next Steps

- HL-LHC poses significant challenges to HEP experiments in terms of storage and computing resources
- An interactive high throughput platform has been developed in the framework of the "HPC, Big Data e Quantum Computing Research Centre" Italian National Center (ICSC)
  - offers users a modern interactive web interface based on JupyterLab
  - experiment-agnostic resources
  - based on a parallel and geographically distributed back-end

- Interactive analyses feasibility studies on INFN cloud succeeded
  - Performance evaluated using the high-rate platform
  - HEP analysis use-case explored from the CMS and ATLAS Collaborations

Medium-long term goals: Expand the current pool of resources by a factor of 5 in the upcoming months, to perform scale testing of the analysis workflows.

*This work is **(partially)** supported by ICSC – Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing, funded by European Union – NextGenerationEU*

Thank you!

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

Back-up

High throughput data analysis platform

- Offloading strategy: resources used to offload the computation are hosted in the same k8s cluster as the jupyter interface, via DASK KubeCluster

- **Under development:** schedule worker processes spawning on multiple remote sites dynamically and transparently → Implementation on heterogeneous resources (HTC/HPC/Cloud)

InterLink provides execution of a Kubernetes pod on almost any remote resource. Resources visible to the user thanks to an HTCondor overlay