ROOT 2022

Axel Naumann axel@cern.ch for the ROOT team

2022-03-09

2021

Support!

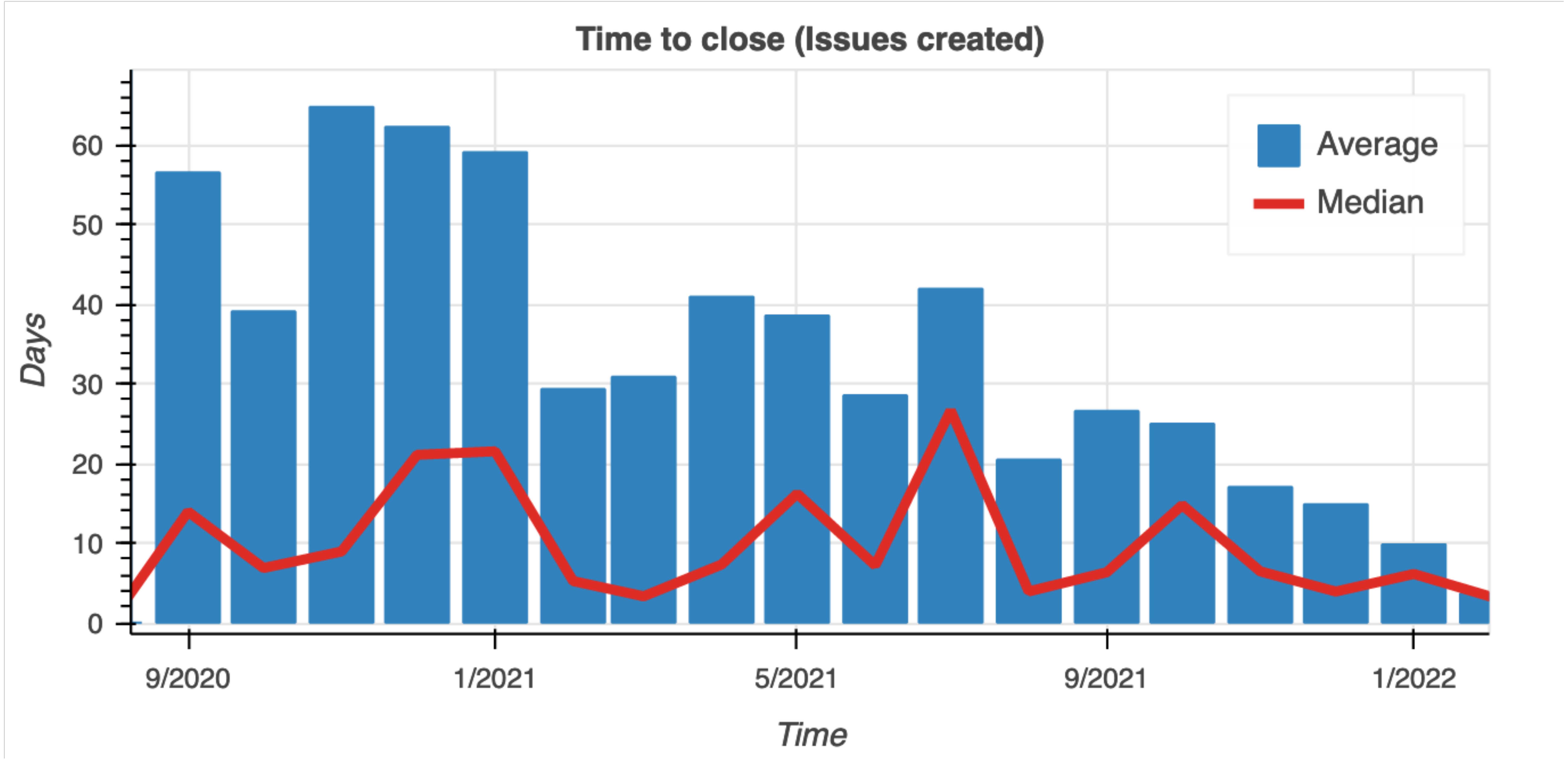
- https://root-forum.cern.ch stats of 2021:
  - 15'000 posts, after 17k in 2020 and 14k in 2019
  - 1.4k new users, after 1.3k (2020) and 1.0k (2019)
  - 1st response on average after 11h, after 19h (2020), 30h (2019) [only topics with an answer are taken into account]
- We spend a considerable time here, web forum is virtually only channel

Bugs

- Everyone loves GitHub
- 650 issues created (compared to 710 in 2020)
- 480 closed (compared to 680 in 2020)
- Current open issues: 380 in GitHub, 1071 in Jira (down from 1150 in 2020)

Bugs

- 50% of issues closed after about 10 days,
  all stats thanks to https://cauldron.io/project/5676



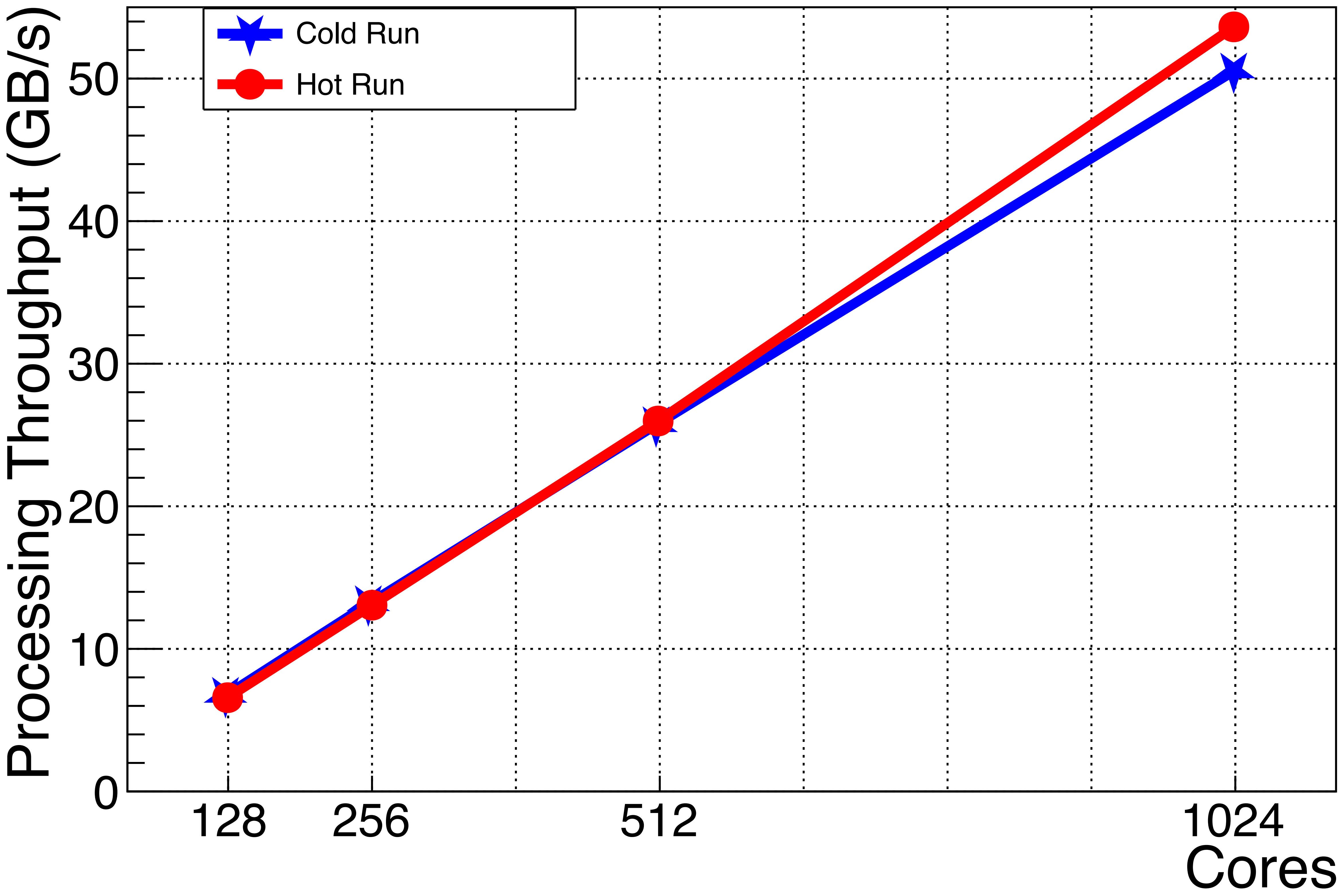
2021's Major Features

Distributed RDataFrame

- RDataFrame used by N*10% of analyses, sometimes embedded in analysis mini-frameworks: Bamboo, CROWN, Wmass,...
  - RDataFrame scales through multi-threading
- Distributed RDataFrame: scale across nodes (cluster), PROOF succession
  - Python-layer over RDataFrame: same interfaces, re-use of industry standard schedulers / cluster "adaptors": Dask (i.e. HTCondor etc), Spark, AWS Lambda

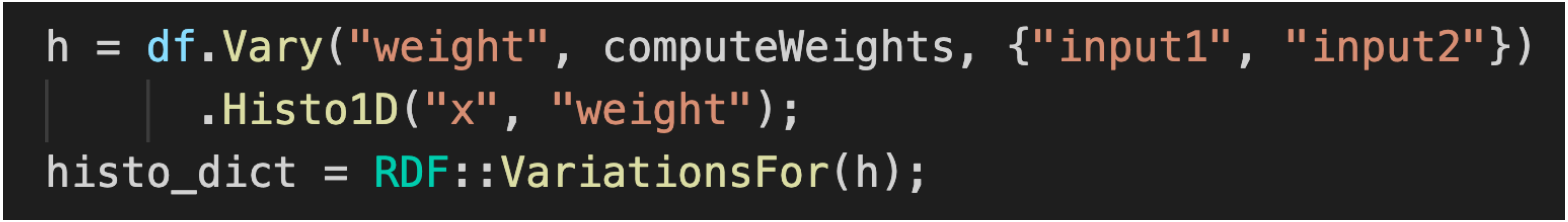
Distributed RDataFrame

- Prototype became minimal viable product in 2021
  - Feedback from physicists + first analysis groups are using it!
  - Lots of attention from the community: real demand
- Incorporating input from PROOF devs + experts, as well as cluster admins



RDataFrame::Vary()

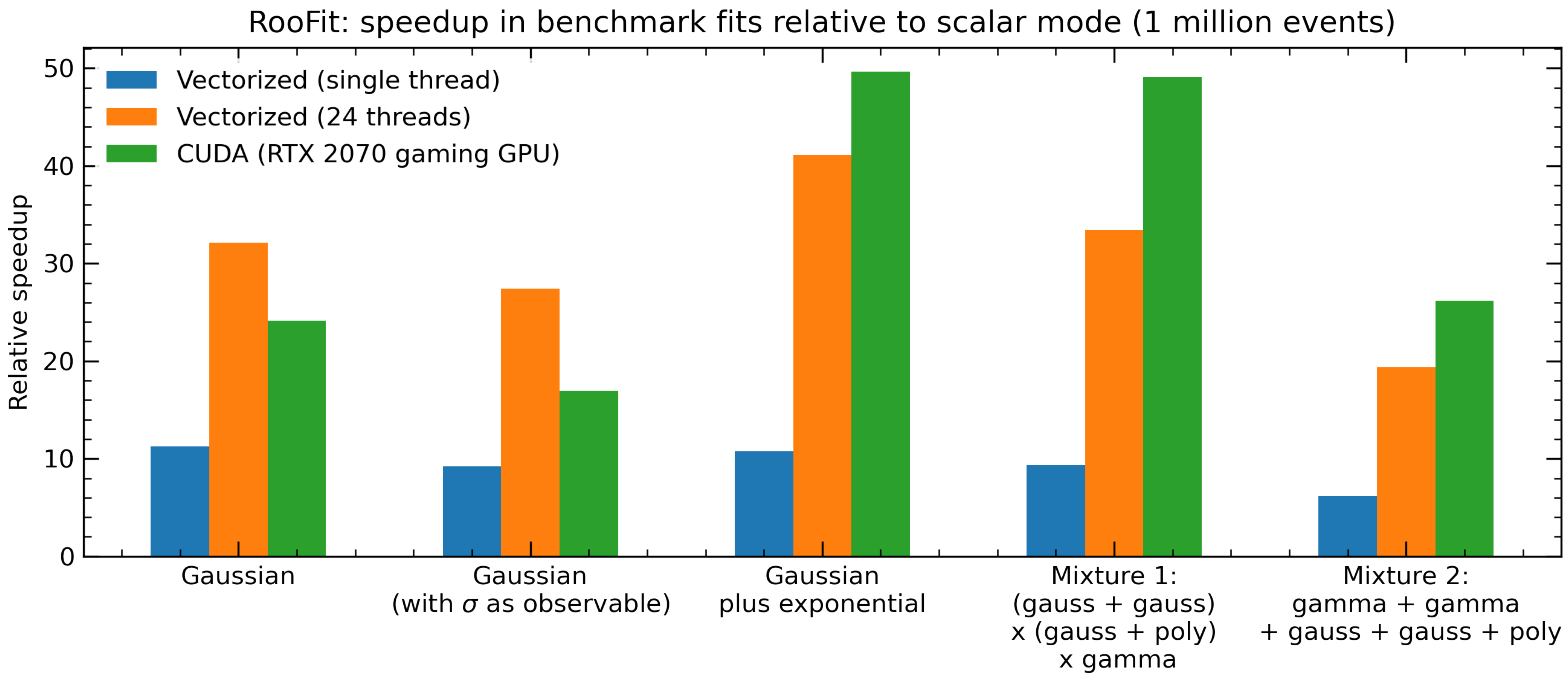
- Can be anywhere inside the whole analysis, anything: weight, input data, efficiency,...
- Creates a "parallel universe" of everything that depends on the varied value
- Evaluates everything in one single loop through data: a **game changer** behind an incredibly simple interface!



```
h = df.Vary("weight", computeWeights, {"input1", "input2"})
      .Histo1D("x", "weight");
histo_dict = RDF::VariationsFor(h);
```

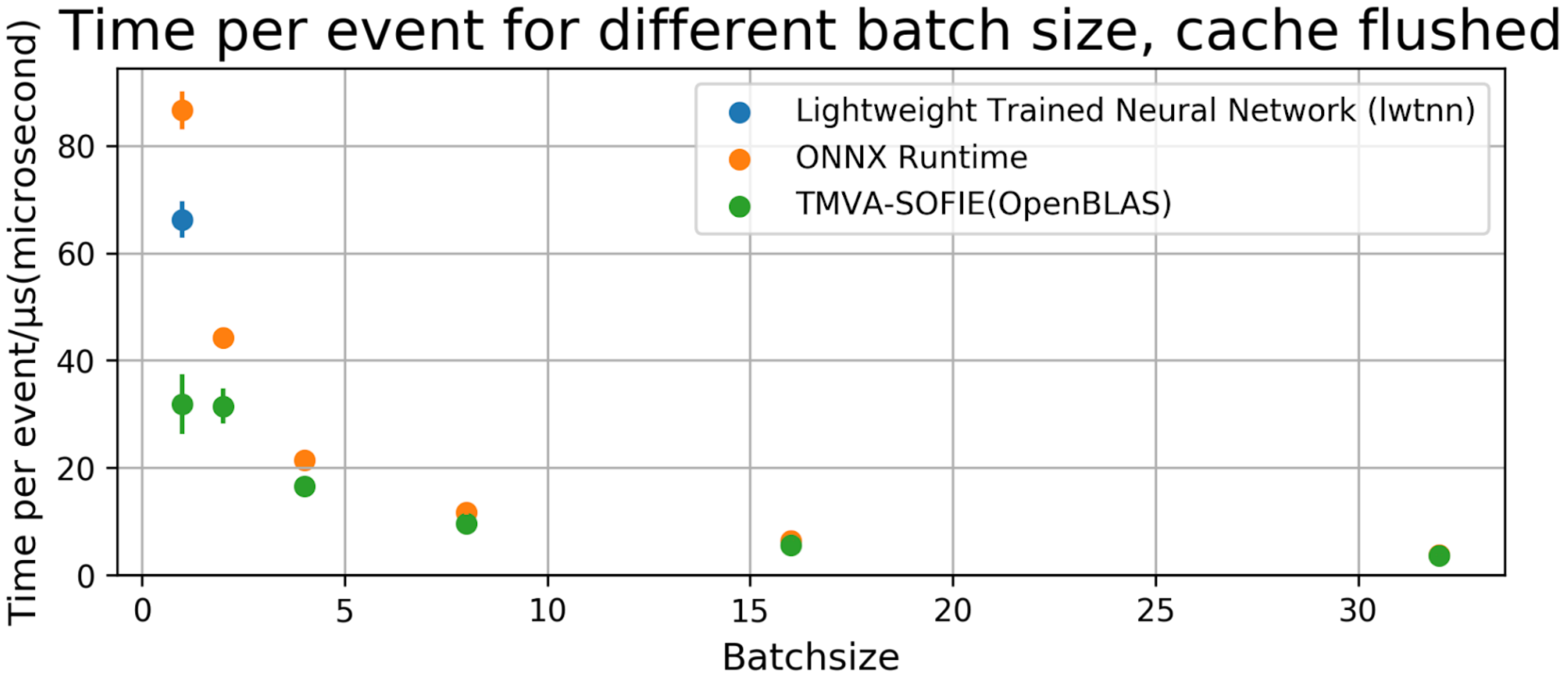
RooFit GPU + Pythonizations

- RooFit now has architecture-specific accelerator libraries for key functions
  - Optimal one loaded at runtime, given current architecture
  - Now also includes GPU version!
- Much improved Python interfaces!



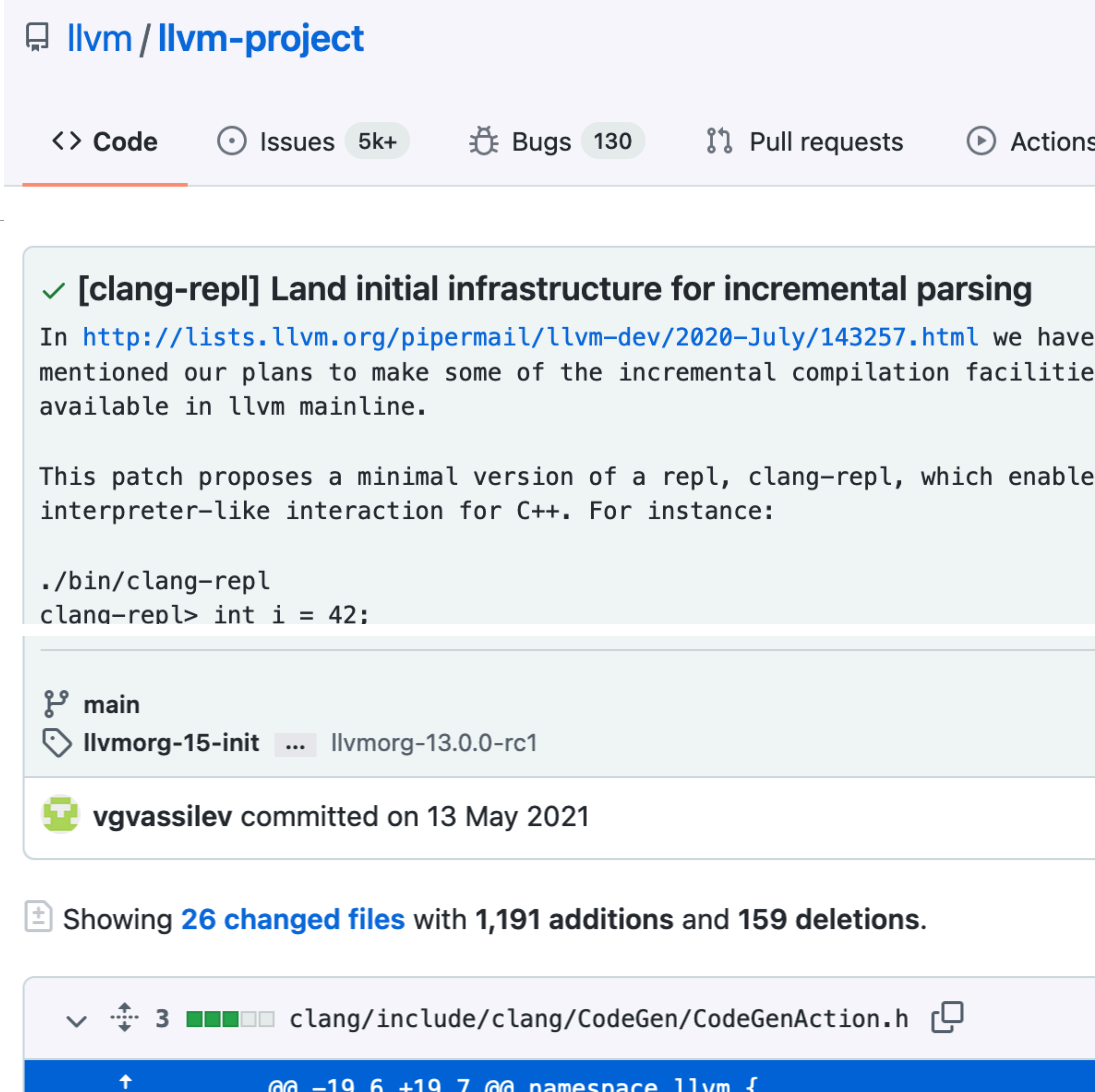
TMVA SOFIE

- ONNX is standard interchange / persistency format for trained models
- SOFIE can read those and generate C++
- Much more performant than ONNX runtime
- Incredibly lean
  (BLAS dependency)



Interpreter / Binding, Build

- clang-repl part of LLVM!



✓ [clang-repl] Land initial infrastructure for incremental parsing

```
In http://lists.llvm.org/pipermail/llvm-dev/2020-July/143257.html we have
mentioned our plans to make some of the incremental compilation facilitie
available in llvm mainline.

This patch proposes a minimal version of a repl, clang-repl, which enable
interpreter-like interaction for C++. For instance:

./bin/clang-repl
clang-repl> int i = 42;
```

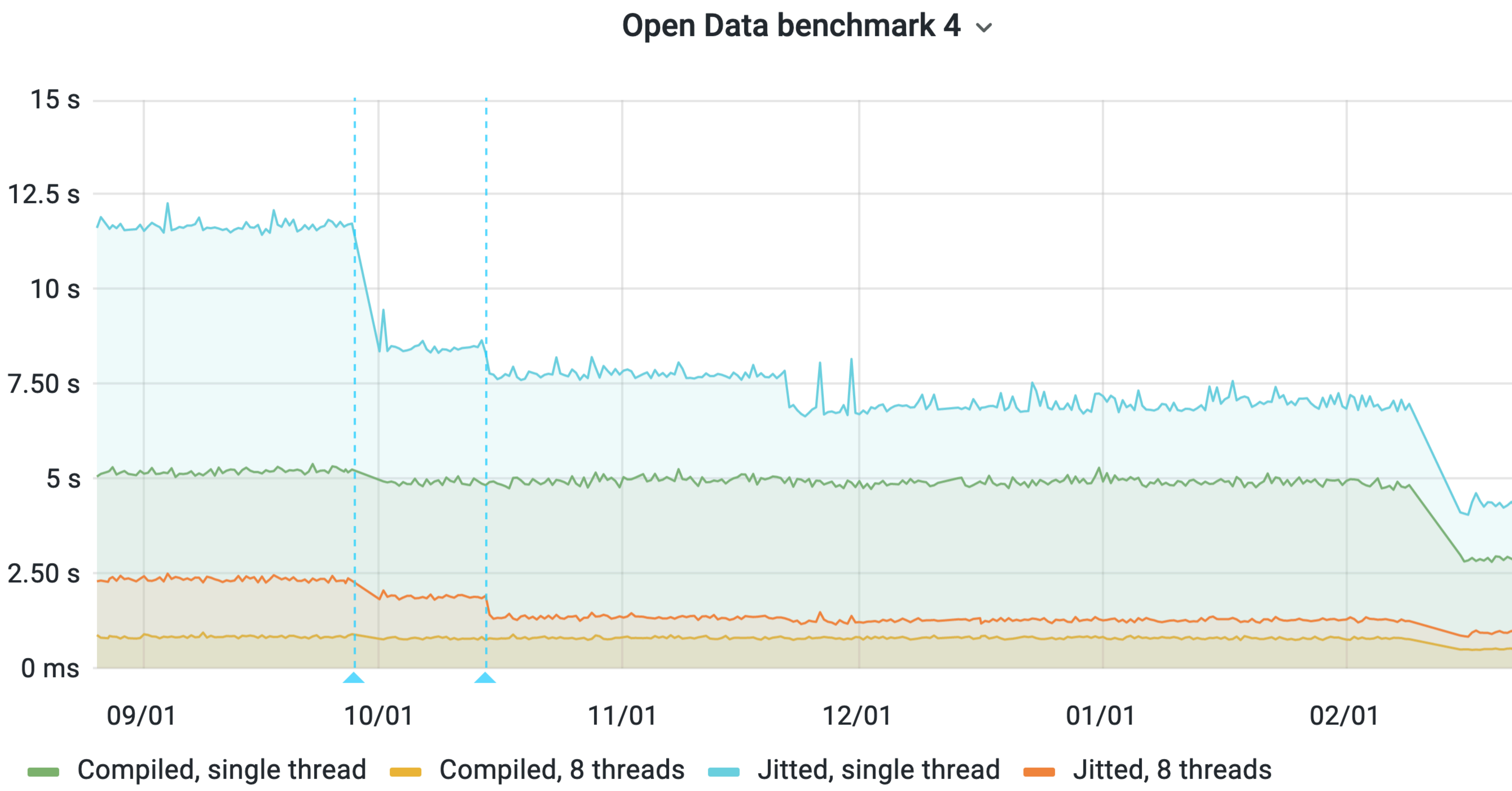
vgvassilev committed on 13 May 2021

Showing 26 changed files with 1,191 additions and 159 deletions.

```
3 clang/include/clang/CodeGen/CodeGenAction.h
@@ -19,6 +19,7 @@ namespace llvm {
```

Interpreter / Binding, Build

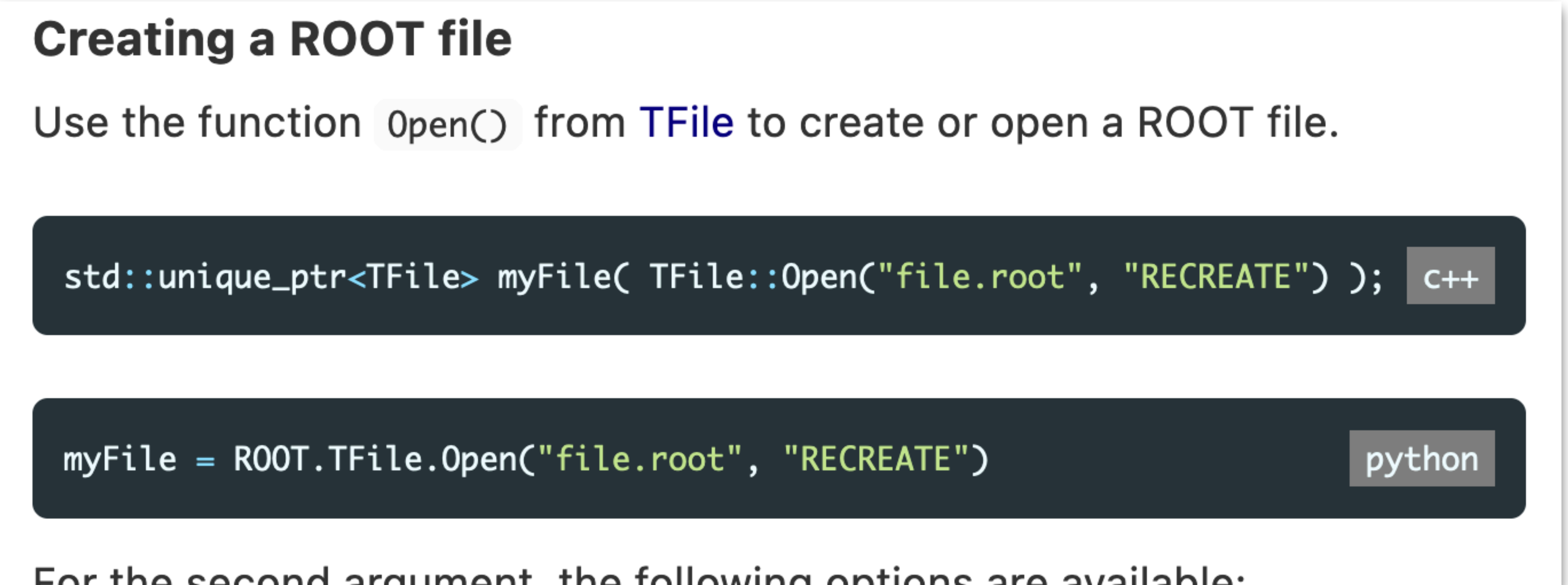
- Upgrade of cling to LLVM 9
  - ROOT now requires C++14
  - significant JIT optimization: "interpreted" code == compiled code



- ROOT has updated docker images, Conda nightlies

Documentation

- Team spent twice a week on documentation, manual
  - Complete re-write using modern ROOT, Python and C++
- Multiple blog posts, including contributed ones



Creating a ROOT file

Use the function `Open()` from TFile to create or open a ROOT file.

```c++
std::unique_ptr<TFile> myFile( TFile::Open("file.root", "RECREATE") );
```

```python
myFile = ROOT.TFile.Open("file.root", "RECREATE")
```

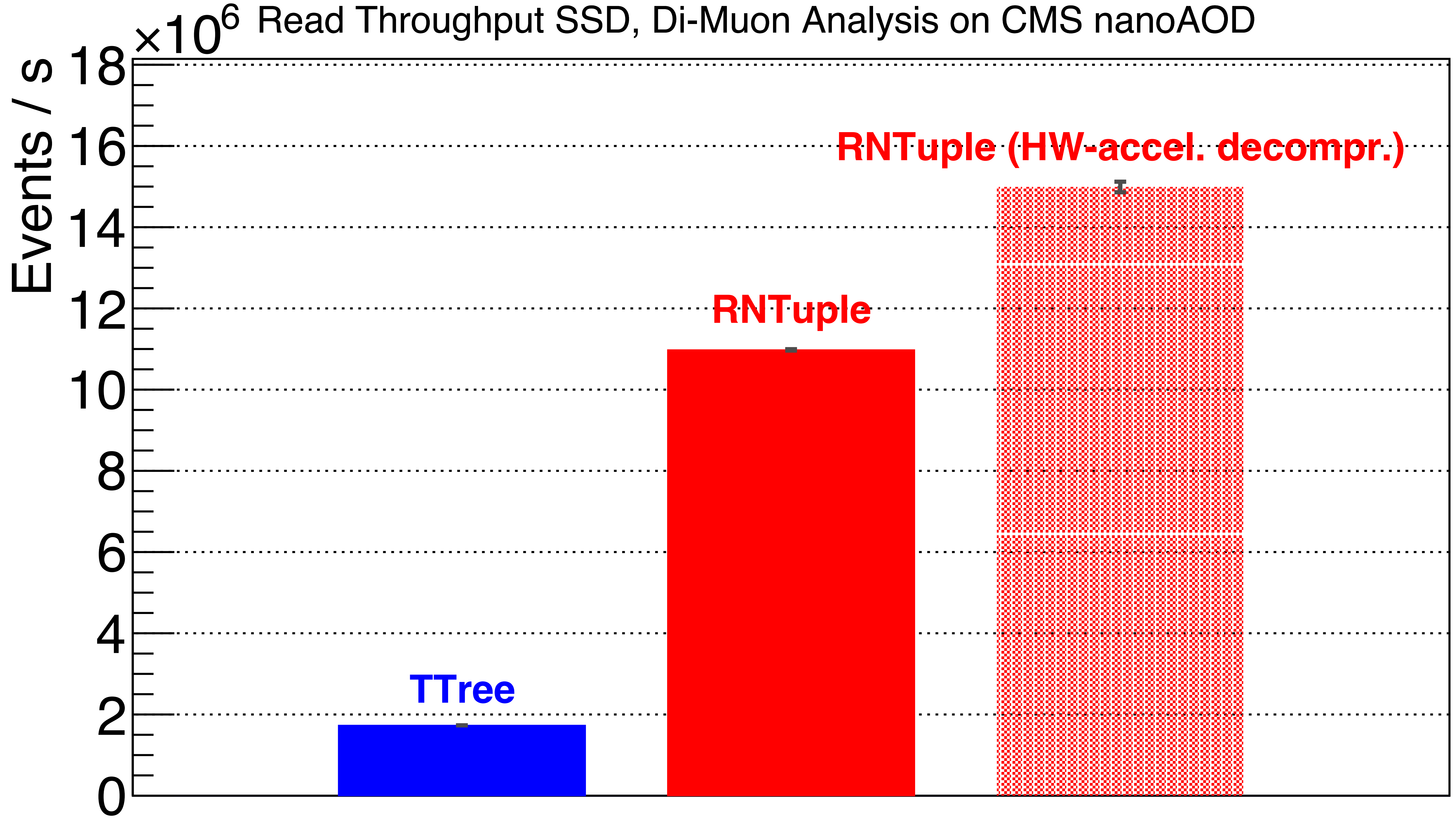
For the second argument, the following options are available:

RNTuple

- Scheduled for production for HL-LHC
- Binary layout v1 defined
- DAOS (Intel object store) backend implemented
- Requirements input from experiments
- Work towards 100% feature completeness
- Benchmarking benchmarking benchmarking

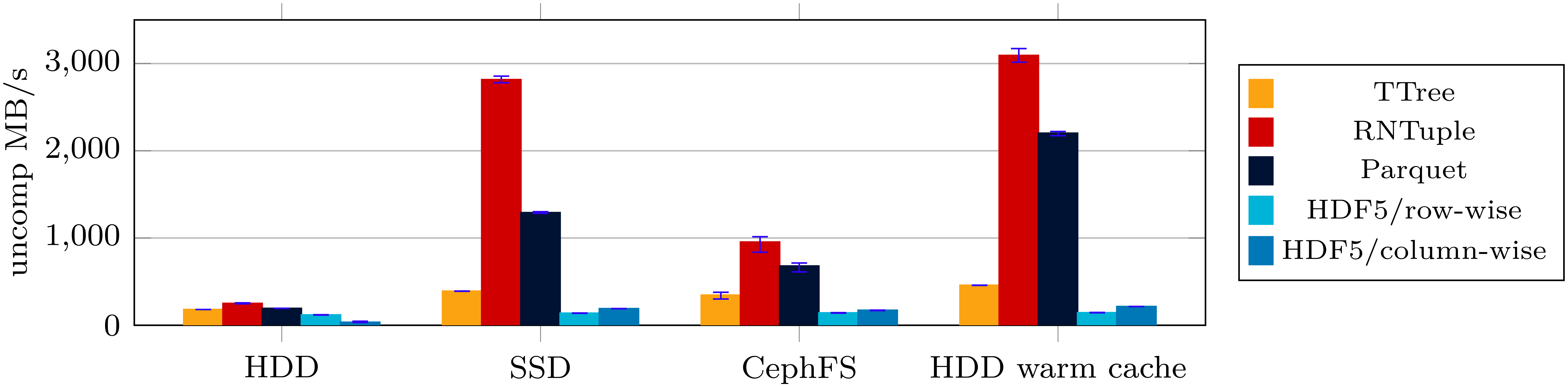
I/O Performance

RNTuple **3-5x** faster, **-10..-20%** storage = 5..10MCHF/y



RNTuple

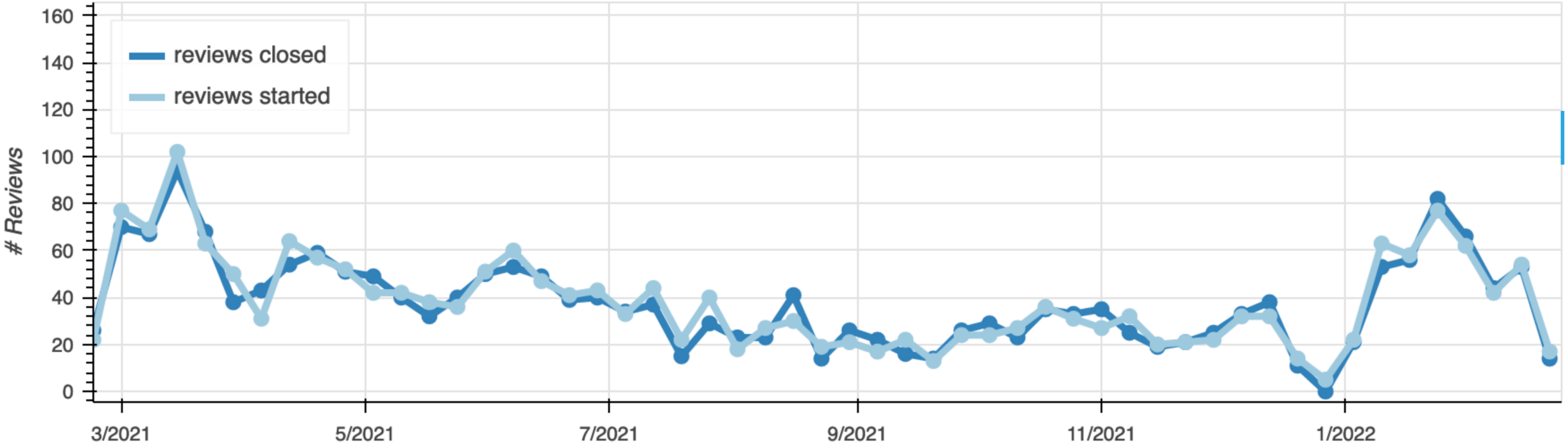
- LHCb analysis example B2HHH; 18/26 branches read; compressed files



2021 Dev Statistics

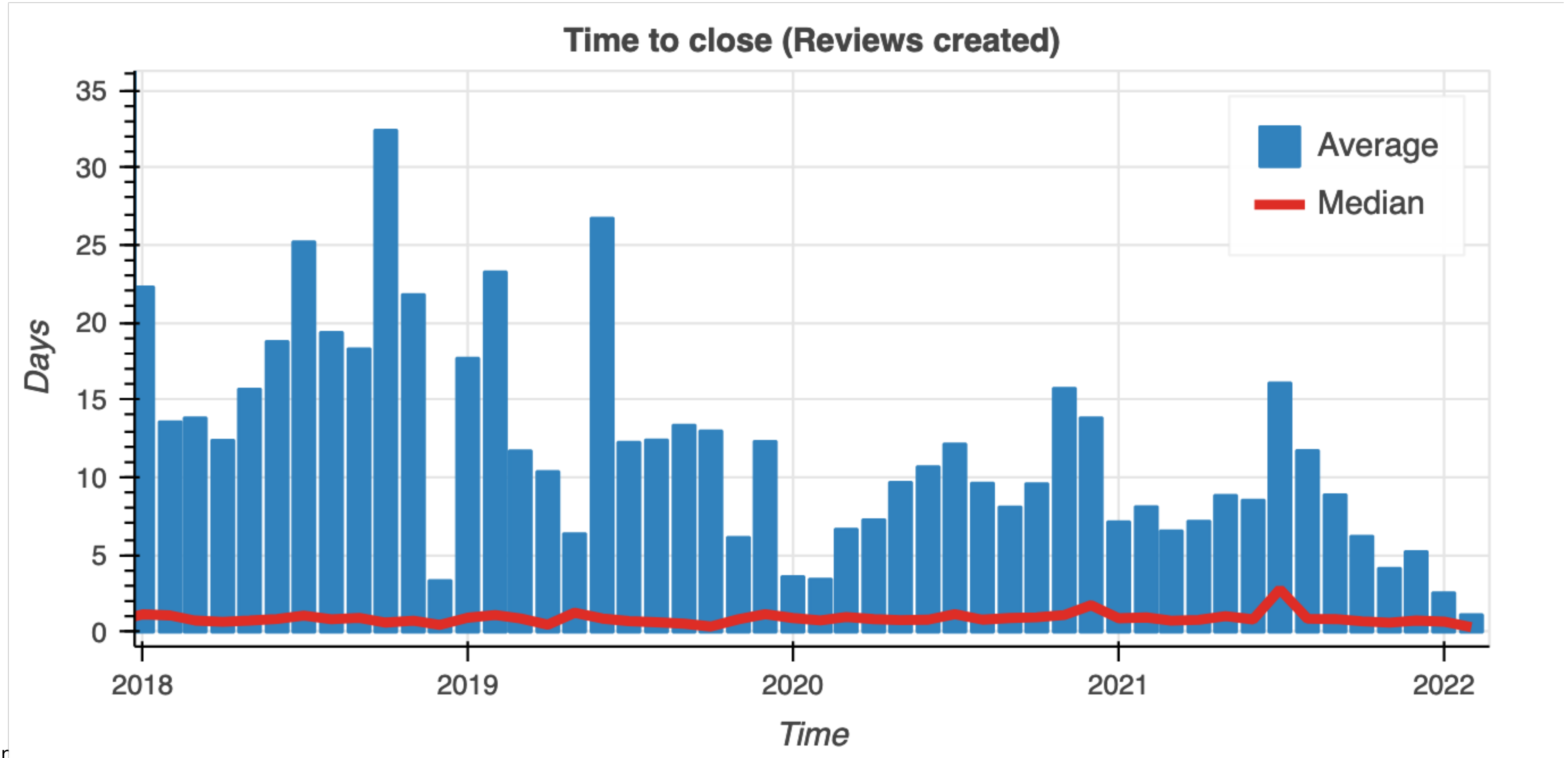
Code Change = Pull Requests

- About 2000 PRs over 2021, PRs per week:



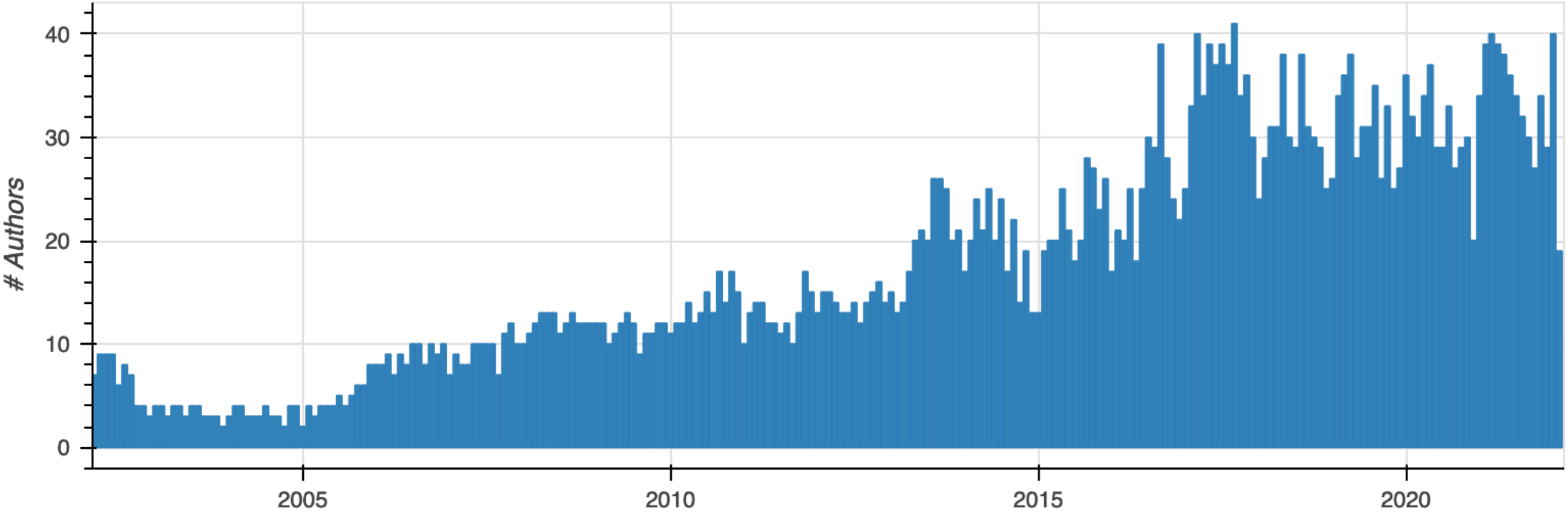
Are PRs working?

- Team invests in high PR review throughput



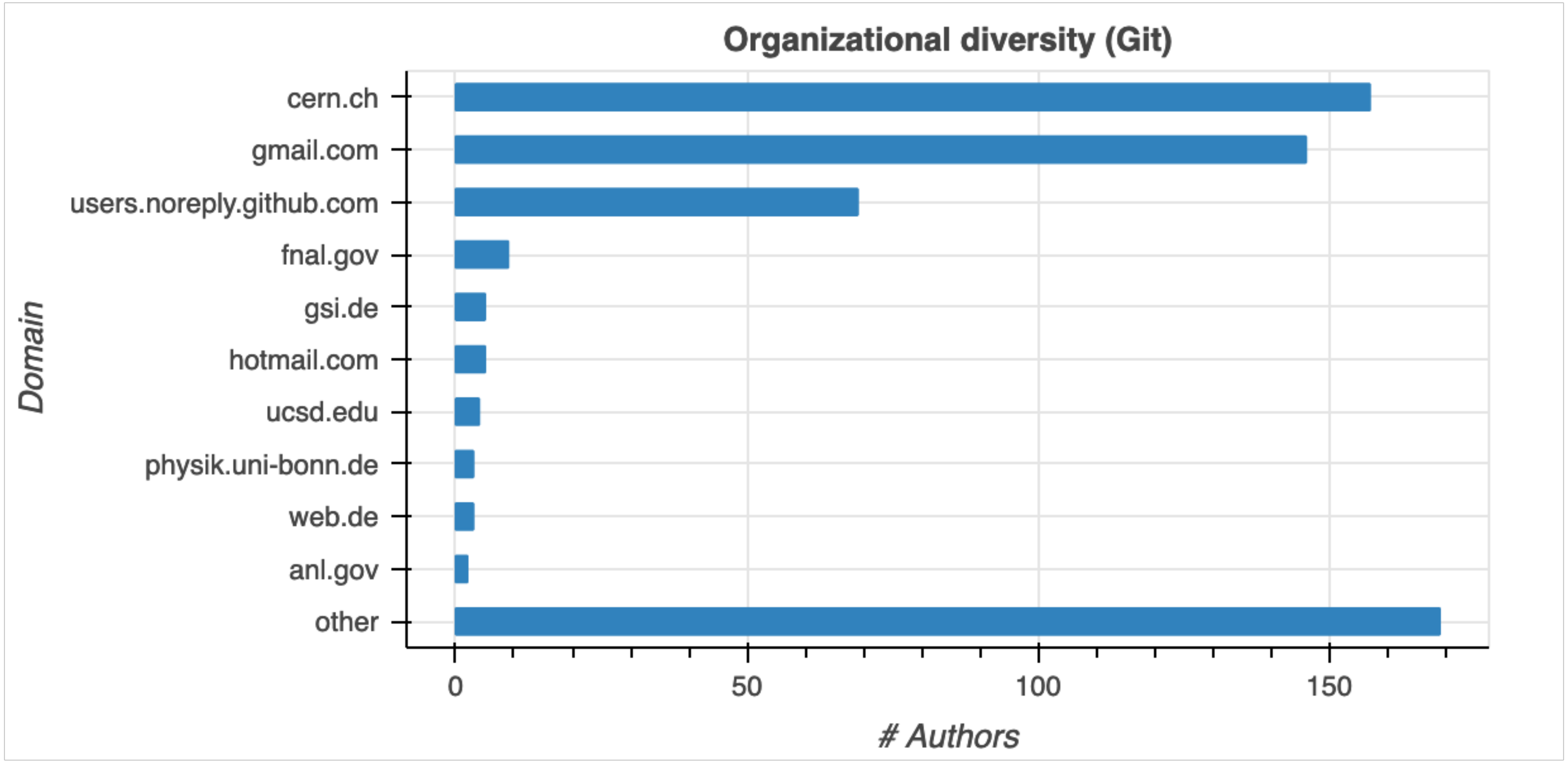
Contributors

- Consistently high number of contributors / month



Contributors

- High ratio of community contributors



Communication

Presentations, Working Groups

- Conferences: ACAT, vCHEP, EPS-HEP, LHCP, JLAB round table, PyHEP, Dask Distributed Summit, HIPS'21, CMMSE'21

- Several presentations with experiments' physics groups

- Engagement with experiments, e.g. CMS analysis tools task force, ATLAS RooFit Hackathon

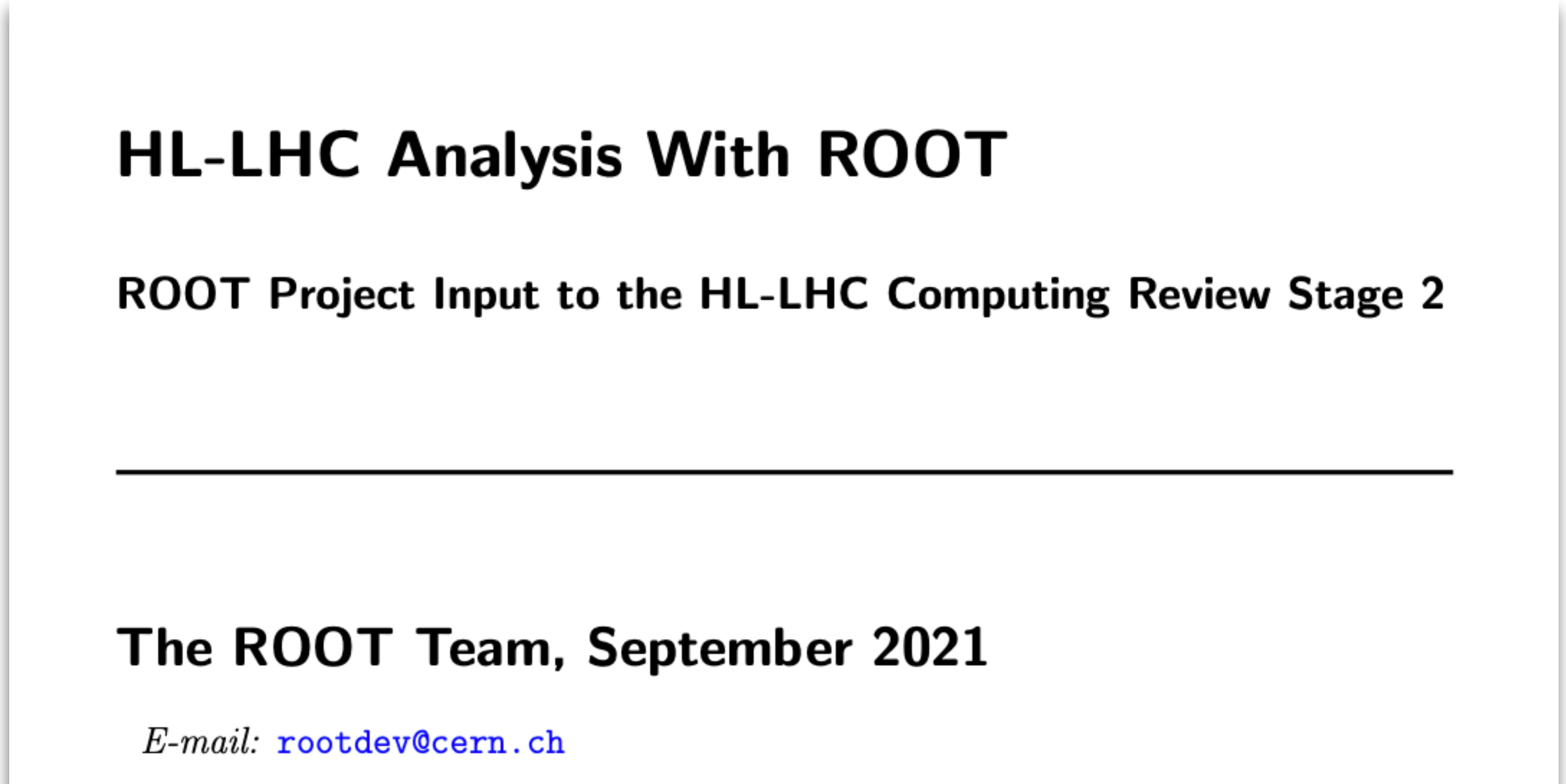
- Member of CERN's Open Science working group, CERN-IT Analysis Facility working group

Trainings

- Contributions to CMS Data Analysis school
- Software carpentries
- C++ course
- CERN Academic Training (SWAN)

LHCC Review

- Review of readiness for HL-LHC
- 60 pages of documentation of how ROOT works, what ROOT plans to do and why, risks and benefits
  - Significant load next to everything else
- One-day, hybrid event with reviewers: extremely constructive and helpful
- Report expected still in March



HL-LHC Analysis With ROOT

ROOT Project Input to the HL-LHC Computing Review Stage 2

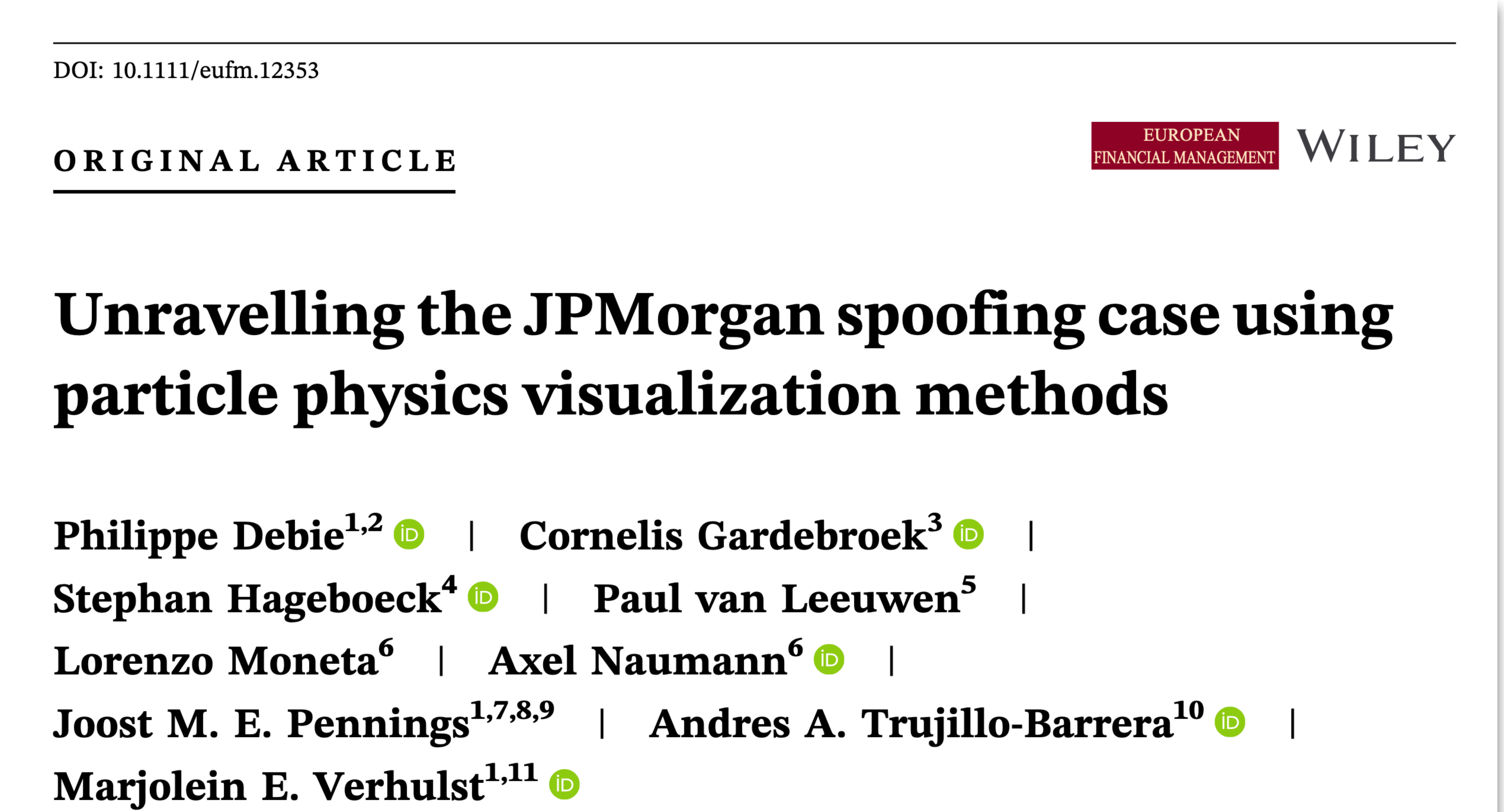
The ROOT Team, September 2021

E-mail: rootdev@cern.ch

HighLO

- Cooperation with finance research on fraud detection

- Two publications in finance journals



DOI: 10.1111/eufm.12353

ORIGINAL ARTICLE

EUROPEAN FINANCIAL MANAGEMENT WILEY

Unravelling the JPMorgan spoofing case using particle physics visualization methods

Philippe Debie[1,2] | Cornelis Gardebroek[3] |
Stephan Hageboeck[4] | Paul van Leeuwen[5] |
Lorenzo Moneta[6] | Axel Naumann[6] |
Joost M. E. Pennings[1,7,8,9] | Andres A. Trujillo-Barrera[10] |
Marjolein E. Verhulst[1,11]

2022

Workshop + training

- Virtual workshop "at" Fermilab: May 9-11
  - https://indico.fnal.gov/e/root2022
  - Register!
  - ROOT features, user feedback: shaping ROOT's development
- Planning a public training event for summer

Plan of Work

- Result of discussions within the team and with experiments
- We have many more plans, but let's keep it realistic
- Priorities and goals can shift
    - Will communicate with experiments / users if that happens
- Priorities: **super high**, medium high, fairly high

I/O

- **Address scaling issues with MT-writing to TBufferFile**
- Schema evolution improvements
- **TBufferFile > 1GB**
- Incorporation of lossy compression

RNTuple

- **Add (preliminary) support for schema evolution and I/O customization**
- **Implement backfilling**
- Disk-to-disk converter
- Add hadd support / fast merging
- Finalize support for chains and (unaligned) friends
- Add bulk I/O API
- **Finalize support for DAOS: backend improvements and data mover**

RDataFrame

- **Batch processing of RNTuple data**
- TMVA+RDF: Finalize fast inference from ONNX format for DNN+CNN, including RDF adapter
- RNTuple+RDF integration: fix RVec usage
- More pythonic usage, less C++ code in strings
- Allow default values for missing branches
- **Make nested parallelism safe (ROOT-10269)**

Distributed RDataFrame

- **Assess need for C++ version**
- **Reduce initialization time in client due to file metadata queries**
- **Systematic variations**
- JIT only once
- Better execution logs from distributed execution
- Decide on API for user code distribution

Math

- **In Minuit2, add support for external computation of Hessian**
- Make Minuit2 default minimizer for ROOT fitting and RooFit
- Improve Pythonization of Math libraries: direct Numpy interface to histograms and graphs
- **RHist: revisit class layout, RHistView (range), tutorials**

RooFit

- **Prototype usage of automatic differentiation**
- **Consolidate work on GPU support**
- **Roll out parallel gradient likelihood and parallel Hessian computation**
- Further optimize HistFactory implementation for speed
- Stabilize RooWorkspace to JSON conversion tools
- **More benchmarks with recent experiment workflows**
- Further pythonizations

TMVA

- **Consolidate SOFIE**
- Finalize RReader interface and make it default for TMVA inference
- Pythonic interfaces, e.g. direct conversion between numpy and RTensor

Interpreters

- PyROOT
  - Feasibility study on Numba understanding cppyy
  - Move cppyy to use more cling interfaces instead of ROOT meta
- Cling
  - **LLVM upgrade**
  - Address code unloading issues

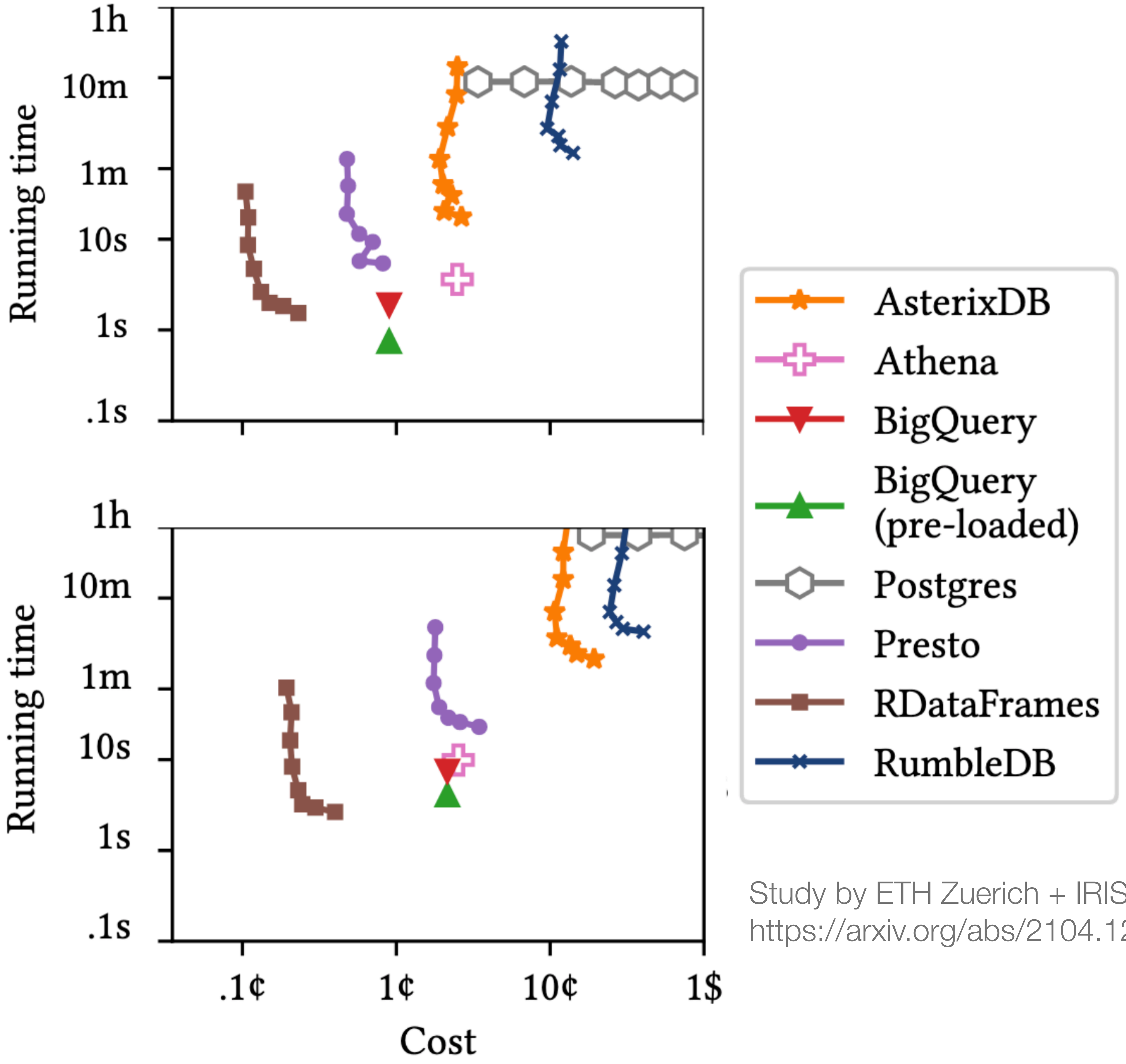
Builds, Binaries

- CI rewrite including PRs
- Windows 64 bit including roottest
- Windows cxxmodules
- Prototype CMake superbuilds
- **Add .deb package generation with CPack**

Conclusion

Why?

Independent study shows ROOT's analysis interface RDataFrame to be significantly **faster**



Study by ETH Zuerich + IRIS-HEP
https://arxiv.org/abs/2104.12615

Goal

- ROOT provides an analysis interface that's
  - Reliable
  - Sustainable
  - Supported
  - Efficient
  - Simple

THANK YOU

for your help in 2021!

Martin Vorel, CC BY-SA 4.0, via Wikimedia Commons

Thank you

- Much of what we do is initiated from, developed with, provided by the community
- The experiments do a terrific job at providing early feedback
- Thank you for your patches, bug reports, discussions, ideas, reproducers
  - The community invests a lot
  - We do our best to convert that into a better ROOT, also in 2022

Martin Vorel, CC BY-SA 4.0, via Wikimedia Commons