News and Decisions

Danilo Piparo (CERN, EP-SFT)

4-03-2024

Some Stats

2023 PoW Completion

- Exercise the formalism on the ***2023 PoW***
- Completion status:
  - With extra items: 56.4 %
  - W/o extra items: 49 %
- **CAVEAT:** this is just a number, which does not represent the performance of the team nor the enormous work done by the ROOT team during 2023
  - E.g. see the achievements slides of the PoW talk

DONE DOING NOT DONE		Priority (1=highest)	Completion Status: 0, 0.5 or 1
Builds and Binaries	CI rewrite including PRs to use GH actions	1	0.5
	Add .deb package generation with CPack	1	0
	Prototype CMake superbuilds	2	0
	Pip install ROOT	2	0
I/O and TTree	Address scaling issues with MT-writing to TBufferFile	1	0.5
	TBufferFile > 1 GB	1	0
	Schema evo improvements	2	0.5
	Beta release of lossy compression + incorporation in RNTuple	2	0.5
	Support for STL collection of std::array	2	0
RNTuple	Bulk I/O API and RDataFrame connection	1	1
	Late schema extension	1	1
	[Prototype merged, missing support for edge cases] Support for merging and chains	2	0.5
	Unaligned friends	2	0
	[Double32_t, f16 support available, Float16_t & custom precision missing] Support of lossy compression (low-precision floats)	2	0.5
	[merged, prototype used for CHEP benchmarks] S3 Backend	3	0.5
	Implementation of meta-data API	3	0
	Prototype schema evo based on existing I/O customisation support	3	0
RooFit	Execute LHCb benchmark fits fully on the GPU (result for CHEP2023)	1	1
	Engine for C++ code generation from RooFit model (as in input for AD)	1	1
	Finalise redesigned minimiser interfaces for better support of automatic differentiation (Clad)	1	0.5
	Further consolidate JSON standard, joint target with ATLAS: publish joint ttW and ttH multilepton full Run-2 analysis	1	1
	Support for likelihood parallelisation with new test statistics and improve scheduling of gradient parallelisation	2	1
	Consolidate new test statistics classes: deduplicate common code, unify interfaces, enable vectorised/gpu + MP fits, etc.	2	0.5
	Stabilise and improve the code, and speed up the HistFactory	2	1
	Pythonise RooWorkspace factory language	3	0
	Create more ROOT benchmarks that compare RooFit also with other fitting tools to get a better overview on fitting tools in HEP	3	1
RDataFrame	Bulk processing, also with RNTuple readers	1	0.5
	Default values for missing columns	1	0
	Distributed support for RDatasetSpec	1	0
	Prototype CUDA kernels in RDF	2	1
	Varied Snapshot	2	0
	Live histograms (streaming results as they come from the mappers)	2	1
Math	Apply several improvements in Minuit2 (e.g. Fumili algorithm)	1	1
	PARTIALLYComplete Pythonizations of Histograms and Graph classes	1	0.5
	Benchmark Minuit2 against optimisers of scipy and eventually integrate some of those algorithms in ROOT	2	0.5
TMVA	Batch generator integrated with RDataFrame to train ML models	1	1
	Add support for SOFIE for inference of GNN	1	1
	Consolidate SOFIE adding support for missing ONNX operators according to user requests	1	1
	Make SOFIE inter-operable with HLS4ML	2	0
	Consolidate RBDT (fast BDT inference)	2	0
Visualisation and UI	REve - RenderCore using GPU, window manager, drop ROOT::Experimental namespace	1	0.5
	TWebCanvas - use by default as with TBrowser	1	1
	Optimise object Paint methods - avoid gPad as much as possible	2	0.5
	Support RWebWindow in JupyterLab - make it fully interactive	2	0
Interpreters	Seamlessly translate PyROOT-bound C++ code via Numba	2	0.5
	Risc-V support for Cling	1	1
	Cling: O2 for non-interactive ROOT on Linux and Mac	1	0
	Reduce dependency on precise version of SDK headers, esp. For macOS	2	0
Extra Items	5x speedup of import ROOT	1	1
	RNTuple support of std::unordered_set/map	1	1
	Comprehensive RDF scaling tests on a single multicore node and in distributed mode on many multicore nodes	1	1
	RNTuple Inspector	2	1
	RNTuple support for all ATLAS data tiers persistified so far with TTree	1	1
	RDataFrame analysis chains of RNTuple datasets	1	1
	Migrate ROOT's LLVM to LLVM 16 and reduce the number of custom patches from 86 to 56	1	1
	Deprecation of Python 2	2	1
		overall:	49.0
		overall Extra Items:	56.4

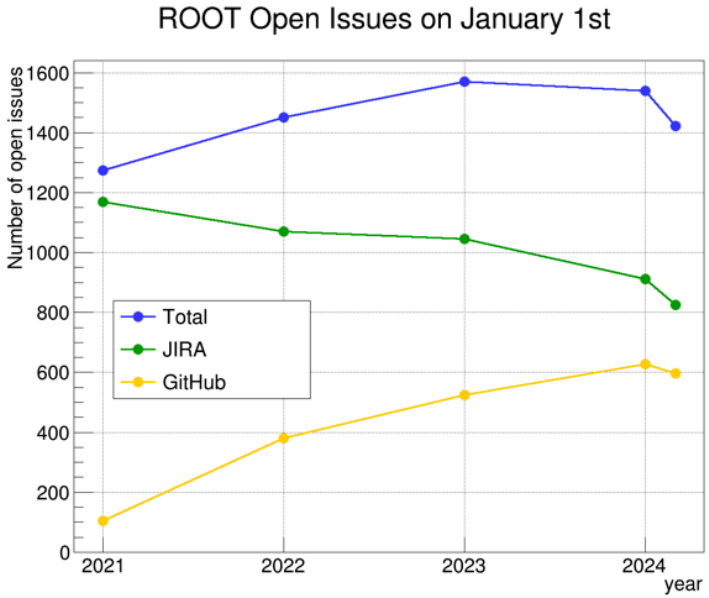
2024 PoW Completion

DONE					
PARTIALLY DONE			Completion		
NOT DONE		Priority	Status:		
		(1=highest)	0, 0.5 or 1		
Builds and Binaries	pip install ROOT for some selected platforms	1	0		
	Complete transition to GH Actions, adding GPU runners	1	0		
	Reduce number of services hosted by root.cern with a combination of CERN IT central services	1	0		
	Win: Replace Debug builds with ReleaseWithDebInfo in the CI	1	0		
	Optimise dictionary dependencies to minimise build real time	2	0		
	Win: Add support for Ninja	2	0	0	%
I/O and TTree	Support std::variant, both in TTree and RNTuple (CMS)	2	0		
	Support writing objects larger than 1GB (TBufferFile > 1 GB, ALICE)	1	0		
	Complete schema evolution improvements	2	0		
	Ensure consistency of std::int types across ROOT I/O	2	0		
	Address residual scaling issues with MT writing	2	0	0	%
RNTuple	Support for chains of datasets and merging	1	0		
	Complete implementation of datasets chains and merging	1	0.5		
	Limit testing in collaboration with CERN IT	1	0		
	Follow-up on API review by HEP-CCE	1	0.5		
	Implement unsplit ("blobbified") encoding	1	0.5		
	Support for unaligned friends	1	0		
	Further develop support for lossy compression with low-precision floats	2	0		
	Design compression dictionaries and understand implications for the specification	2	0		
	First implementation of highly-scalable parallel writing	2	0.5		
	Organise a Design Workshop to discuss intra-link events, metadata, native SoA layout for events	2	0	20	%
RooFit	Workshop with Experiments: promote features, gather input, speedup integration of RooFit in the existing sw setups	1	0		
	Numeric integrals in n-dim with CUDA	1	0		
	Evaluation of custom user functions in CUDA	1	0		
	Make the new vectorized CPU likelihood evaluation interface the default	1	1		
	Reduce JITting time for AD in RooFit	1	0		
	PyROOT: express RooStats configuration with C++-oriented Set* as kwargs	2	0		
	Integration of Fumili in RooFit	2	0	14	%
RDataFrame	Put existing bulk processing in prod	1	0		
	DistRDF: reduce memory usage on HTCondor Workers	1	0		
	DistRDF: improve user experience when integrated with notebooks and nb services like SWAN	1	0		
	Further Pythonise the interface	2	0		
	Deliver varied snapshots	2	0	0	%
Math	PyROOT: better histos ang graph interoperability with NumPy and UHI protocol	1	0		
	Histos: advance current RHist implementation to one testable by experiments	1	0		
	Improve interface to pass initial error values/cov matrix to Minuit2	1	0		
	Release a library for Lorentz vector computations on accelerators in SYCL	1	0.5		
	Deliver plan and prototype of algorithmic improvements when dealing with param constraints in ROOT's minimisers	2	0		
	PyROOT: Pythonise TF{1,2,3} and numerical algorithms interfaces (e.g. minimisers)	2	0		
	Prototype SYCL kernels to be JITted (see Interpreters objectives)	2	0		
	Histograms: Model and prototype of pipelining GPU histogram filling	2	0	6.25	%
ML/AI	Put RBatchGenerator in production	1	0		
	Consolidate RBDT	1	0		
	Support of integration of SOFIE in experiments Fast Simulation pipelines	1	0		
	Add support in SOFIE for NVidia GPUs in CUDA	1	0		
	Continue to add support for the ONNX operators requested by experiments	1	0		
	Make HLS4ML interoperable with SOFIE	2	0		
	Streamline ROOT's inference interface, making it able to use models for Python ML frameworks (e.g. Keras/TF) directly	2	0	0	%
Visualisation and UI	Automated placement/tune of plot elements, "Auto Style"	1	0		
	Add missing features of classic graphics to the web-based one	1	0		
	Automate web-based graphics test suite	1	0		
	Add residual missing TEve features to REve, e.g. digit visualisation and text elements overlay	1	0		
	Visualization of flat ntuples using predefined visual summary data structures	1	0		
	Improve REve window manager and browser, polish render engine	2	0	0	%
Interpreters	Cling: identify potential Cling codebase reductions through the reuse of parts of clang-repl	1	0		
	Cling: cppyy rebase on top of cling/clang-repl	1	0		
	Migrate PyROOT to the latest Cppyy	1	0		
	Cling: Prototype SYCL support	2	0	0	%
Doc and Edu	Re-evaluate, update, and improve course material, making it more visible and better organised on the website	1	0		
	(Re-)evaluate tuts, eliminating what's outdated, what newer features would benefit from a (better) tutorial, improve visibility	1	0.5	25	%
Extra Items: Core	Copyless reading in RNTuple - ALICE	1	1		
E.I. CMS CAT	Physics objects representations out of NanoAOD in RDataFrame	1	0		
	Bulk Processing + GPU offloading for distRDF	1	0		
	include the open source Tex Gyre Heros clone of Helvetica in root fonts	2	1		
	Friend trees and RNtuples	2	0		
	Support for joins in RNtuples	2	0		
E.I. CMS TSG	Multithreading-friendly interfaces to the histogram types	1	0		
	A library of matrix operations that can run on GPUs	1	0	25	%
			overall:	6.7	%
			overall + Extra Items:	8.8	%

- A large PoW: currently 77 items on it
- **Current person power in the ROOT team insufficient to deliver all items**
- **New arrivals foreseen during the year; final completion percentage will depend a lot on these new colleagues**
- External help, i.e. ROOT community (e.g. experiments) can make the difference, too

One Important KPI: # Open Issues

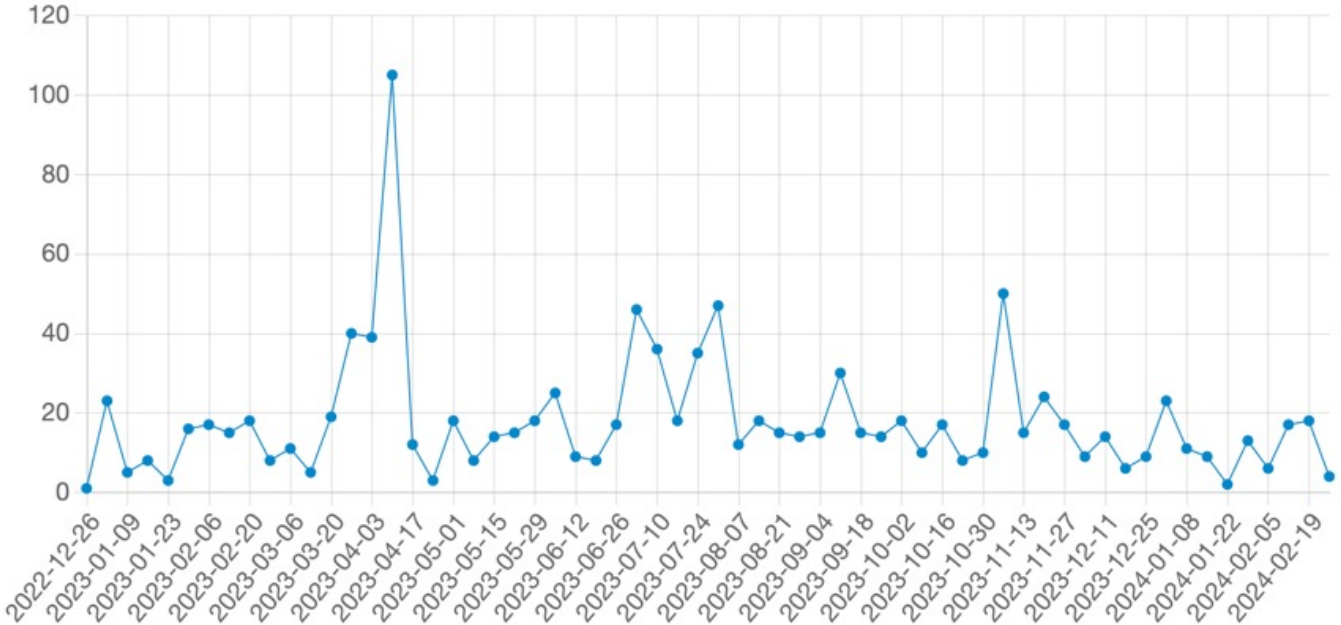
	JIRA	GitHub	Total	Notes
Dec-20	1169	104	1273	
Dec-21	1071	380	1451	
Dec-22	1045	525	1570	
Dec-23	912	627	1539	
Feb-24	826	596	1422	54 issues migrated from JIRA to GitHub



- Strong focus on reducing number of open issues
- Implicitly make backlog reduction *part of the PoW*
- Great boost given by the ROOT community and team at the 1st ROOT Hackathon: the Fixathon (14-15 February)
  - 68 issues closed thanks to that sprint, 440 during 2023

Forum: Time to Give a First answer

- 2023: 20 hours
- 2024 so far: 12 hours (no major holiday period yet)



Stats from the
[Forum admin page]

AI Assisted ROOT Forum Answers

AI Assisted Forum Answers: A Hiccup

- Very interesting idea presented by Ludovico at the 160th PPP Meeting: *A2rchi: a system to enhance LLMs via document retrieval. The use case of ROOT*
- AI able to answer Forum posts
- Same technology in production at MIT for ticketing systems like CERN Snow
- Idea at the meeting: Can A2rchi provide draft answers to forum posts? If yes, ROOT team members can review them and then 1) send them as they are 2) improve them 3) discard them
- **Objective: maintain current support level with a lower investment of human effort**
- **Hiccup: Discourse does not seem to provide an interface to post draft answers**

AI Assisted Forum Answers: A Way Forward?

A proposal to overcome the hurdle just described without generating more work for the ROOT team is:

- Set up a Mattermost private channel with ROOT Devs, A2rchi team and A2rchie itself.
- Archie writes a MM message containing the link to the post and its draft answer
- ROOT Devs (in particular the shifter) keep an eye on the channel and re-use A2rchie answers if they are good
- Periodically, the MM messages linked to threads with >1 posts are removed from the MM channel automatically (avoid unmanageable list of A2rchie messages)

Question to the ROOT Team: **Do you support the strategy described above?**

Releases

Releases: Some Facts

- CMS and LHCb can pick up for the 2025 data taking ROOT 6.32 if a release is provided in August
- A release in August would not have quite a few features which will land in the repo during the last few months of the year
- It would be useful to expose new features (and the many, many fixes) in a release in time for the Summer Students
- We need to expose to users LLVM 16, now only in master
- It is useful to update LLVM as often as possible, ideally once per year, even more ideally upstreaming as many ROOT patches as possible
- We cannot sustain two active releases per year, one has to be LTS the other one frozen when the subsequent LTS is out
- RNTuple activity by the experiments can happen with ROOT master, e.g. with the dev3 LCG slot.

Two possible scenarios

W1, W2, W3, W4 are the weeks of the month.
Release candidate (RC): trigger a LCG release for that, to replace the RC with the release

Scenario A

May Release, W4, short term support:

- LLVM16, secure web graphics + more to be decided
- **Branch 1 months before: April W4**
- **RC 2 weeks before: May W2**

September release, W1, long term support (data taking):

- LLVM18 , secure web graphics, analysis features, RF, math
- **Branch 1.5 months before: July W2**
- **RC 1 month before: August W1**

Advantages:

- Traditional role of releases

Risks and criticalities:

- process leading to the data taking release entirely during the holiday period,
- short lifetime of LLVM16 and uncertainty about 18
- September release risks not to be adopted if the schedule slips

Scenario B

May Release, W4, long term support (data taking):

- LLVM16, secure web graphics + more to be decided
- **Branch 1.5 months before: April W2**
- **RC 1 month before: April W4**

November release, W1, short term support:

- LLVM18 , secure web graphics, +
- Branch 1.5 months before: September W2
- RC 2 weeks before: October W3

Advantages:

- summer months only for development and consolidation
- second release of the year has many more features
- Fits with the LLVM cycle: currently it foresees **an even release branch in January, release in March with point releases during May: window to upstream our patches**

Risks and criticalities:

- The ones we are already exposed to now?

Proposed Scenario

W1, W2, W3, W4 are the weeks of the month.
Release candidate (RC): trigger a LCG release for that, to replace the RC with the release

Scenario A
May Release, W4, short term support:
- LLVM16, secure web graphics + more to be decided
- **Branch 1 months before: April W4**
- **RC 2 weeks before: May W2**

September release, W1, long term support (data taking):
- LLVM18 , secure web graphics, analysis features, RF, math
- **Branch 1.5 months before: July W2**
- **RC 1 month before: August W1**

Advantages:
- Traditional role of releases

Risks and criticalities:
- process leading to the data taking release entirely during the holiday period,
- short lifetime of LLVM16 and uncertainty about 18
- September release risks not to be adopted if the schedule slips

Scenario B
May Release, W4, long term support (data taking):
- LLVM16, secure web graphics + more to be decided
- **Branch 1.5 months before: April W2**
- **RC 1 month before: April W4**

November release, W1, short term support:
- LLVM18 , secure web graphics, +
- Branch 1.5 months before: September W2
- RC 2 weeks before: October W3

Advantages:
- summer months only for development and consolidation
- second release of the year has many more features
- Fits with the LLVM cycle: currently it foresees **an even release branch in January, release in March with point releases during May: window to upstream our patches**

Risks and criticalities:
- The ones we are already exposed to now?