Analysis & friends

ROOT PoW 2021 planning meetings

Contents

- Build system & packaging
- PyROOT
- RDF
  - more Python
  - more ML
  - distributed execution
  - fixing pain points in realistic use-cases

Contents

- Build system & packaging
- PyROOT
- RDF
  - more Python
  - more ML
  - distributed execution
  - fixing pain points in realistic use-cases

If your proposed item or issue raised is not in the presentation, it was my failure in condensing the feedback. Please speak up!

Contents

- Build system & packaging
- PyROOT
- RDF
    - more Python
    - more ML
    - distributed execution
    - fixing pain points in realistic use-cases

If your proposed item or issue raised is not in the presentation, it was my failure in condensing the feedback. Please speak up!

No feedback regarding ROOT7 received

Contents

- Build system & packaging
- PyROOT
- RDF
  - more Python
  - more ML
  - distributed execution
  - fixing pain points in realistic use-cases

If your proposed item or issue raised is not in the presentation, it was my failure in condensing the feedback. Please speak up!

No feedback regarding ROOT7 received

The elephant in the room: documentation

Build system

- ❑ modularization of build system (one CMake project per ROOT module)
  - ➥ ROOT mini-libraries, e.g. miniRNTuple
  - ➥ rationalization of build options

Packaging

- appreciation for third-party packaging efforts and Docker images
- suggestion to sustain these efforts
- what’s missing? how can we help packagers, increase **sustainability**?

❏ root.{deb,rpm}

★ better cling ➔ ROOT ➔ PyROOT layering: building one on top of the other
★ matching tags/releases for ROOT and compatible cling
★ consider upgrade to llvm 11?

PyROOT

- ❑ keep pythonizing: RooFit, histos, graphics, TMVA, ...
- ❑ support Python 3.9
- ❑ support experiments in completing transition to new PyROOT
- ❑ fix remaining issues with Python/C++ cross-inheritance

- ★ long-term sustainability w.r.t. tracking upstream cppyy?

RDF: more Python

- ❑ make interfaces more Pythonic (no C++ required)
- ❑ improve RDF Python performance
  - ★ add option to generate and compile full C++ from an RDF computation graph?
    ➥ PyRDF might also take advantage of the feature

RDF: more ML

- hot topic: better integration of RDF and common ML tasks

❏ RDF as a Python generator of data batches
  ❏ separate computation graph from dataset it runs on
    ➥ PyRDF might also benefit from this feature

★ export collections as jagged arrays? (AwkwardArrays?)

RDF: distributed execution

- consensus that the feature is sorely needed

❏ keep up R&D efforts: demonstrators, caching optimization

❏ investigate distributed execution solutions

-- OR --

❏ integrate PyRDF into ROOT

★ plans for ROOT running on heterogeneous infrastructures?

RDF: fixing pain points in realistic usecases

- ❏ add syntax to support systematic variations (df.Vary)
- ❏ fix remaining performance/scaling bottlenecks
- ❏ make nested parallelism safe
  - ➥ RNTuple can spawn async decompression TBB tasks
- ❏ debug/verbose mode
- ❏ tooling for ROOT I/O performance inspection
- ❏ df.DefinePerSample
- ★ better support for datasets with some samples missing a column
- ★ allow redefinition of columns (df.Redefine)
- ★ operations on groups of columns
  - ➥ symultaneous selection of elements, systematic variations, ...