High Energy Physics
Center for Computational Excellence
Optimizing Data Storage for Next-Generation HEP Experiments

Peter van Gemmeren
for HEP-CCE, Thanks to all the experts

ROOT RNTuple API Review, Outcome
December 2nd 2024

HEP-CCE Review: Timeline

- As part of the last year’s ROOT RNTuple Format and Feature Assessment (6-7 November 2023) · Indico, CCE was asked to host a RNTuple API Review
- Started with Special CCE-SOP tele-conference: RNTuple API Review Kick Off (February 28, 2024) · INDICO-FNAL (Indico)
  - Including Experiment experts form ATLAS, CMS & DUNE,
    - represented in CCE, plus ROOT experts (some part of CCE)
    - Open to everyone (as typical for CCE meetings)
    - Invited additional experts
- Several meetings focusing on different parts of the RNTuple API:
  - Special CCE-SOP tele-conference: RNTuple API Review Reader/Writer (April 3, 2024) · INDICO-FNAL (Indico)
  - Special CCE-SOP tele-conference: RNTuple API Review Model/Field/Entry (May 1, 2024) · INDICO-FNAL (Indico)
  - Special CCE-SOP tele-conference: RNTuple API Review Remaining modules (June 26, 2024) · INDICO-FNAL (Indico)
- Produced mid-term report that was submitted to the ROOT team on Special CCE-SOP tele-conference: RNTuple API Review Mid-Report discussions (September 25, 2024) · INDICO-FNAL (Indico) with several suggestions for improvement and general finding that RNTuple API is sufficient for adoption by experiment’s production frameworks:
  - Hear later, ATLAS and CMS have functional prototypes allowing their software to write production collision data to RNTuple.
  - Some DUNE experts participated in the Review, but since DUNE is just starting to design their production framework, the outcome is less conclusive
  - Analysis use of RNTuple was considered out of scope for this review.

HEP-CCE Review: Timeline & Participation, continued

- Continued Review:
    - Special CCE-SOP tele-conference: RNTuple API Review Error Handling (October 16, 2024) · INDICO-FNAL (Indico)
    - Special CCE-SOP tele-conference: RNTuple API Review Wrap Up (November 20, 2024) · INDICO-FNAL (Indico)
- Reviewers, Experts
    - ATLAS: Marcin Mowak, Serhan Mete, Peter van Gemmeren
    - CMS: Chris Jones, Matti Kortelainen, Dan Riley
    - CAF: Amit Bashyal
    - DUNE: Barnali Chowdhury
    - CCE: Saba Sehrish, Philippe Canal, and several experts from Computer Science
- Parallel to the API Review, experts in CCE shared studies about RNTuple functionality and performance which CMS and ATLAS will present in dedicated sections.
- Great progress overall, Thanks to everyone.

HEP-CCE Review: Findings, functionality

1) Page Size: Need mechanism to customize page size for particular fields

2) Read Caching: For RClusterPool, based on past experience with TTree I/O, the experiments desire configurability similar to TTreeCache.

    - Tracked in root-project/root/#16325

4) Memory Writing: The experiments need an ability to tune the memory usage in RNTupleWriter

    - Adaptive page size algorithm was merged. First results for ATLAS/CMS are promising. Further knobs would still be appreciated.

5) Parallel Writer: Inability to use RNTupleParallelWriter as long as it is restricted to have only one Writer per file. (Experiments need to store several container/TTree/RNTuple in the same file)

    - Need clear guarantees about the locking around TFile.
    - New method FillNoCommit() allowing framework to control the time of TFile access.

6) Parallel Writer: For every RNTupleParallelWriter `Fill()` call, need to know what entry number that `Fill()` call corresponds to

    - New staged cluster committing should help

HEP-CCE Review: Findings, interface and documentation

7) Object Ownership: Desire to avoid passing dummy `std::shared_ptr<void>` to `RNTupleReader::GetView<void>()`

- To be addressed in the future

8) Model freezing: Having to have two separate loops to call `RNTupleModel::AddField()` (requires un-frozen model) and `RNTupleModel::GetToken()` (frozen model) feels suboptimal

- To be addressed in the future

9) Destructor: `REntry::RFieldToken` not having a default constructor is somewhat inconvenient

- To be addressed in the future

Documentation:

- Relationship between RNTupleModel’s GetToken() and freezing
- Projected fields
- Entry invalidation when RNTupleModel is extended

New:
Empty view: To create an empty view, we need to use dynamic allocation, eg. `viewMap[fieldID] = std::make_unique>(reader->GetView(fieldName, nullptr));` Potential for improvement
Configure split/unsplit encoding in an easier way

HEP-CCE Review: Outlook

- RNTuple API Review was a very useful exercise for CCE and the experiments
  - Allowed us to give early feedback on the API we expect to use for years to come
  - And see the feedback being taken seriously. Thanks to the ROOT team!
    - Thanks also to all CCE and non-CCE experts for their work.
- Hopefully beneficial for ROOT experts.
- Looking forward to continued cooperation.