Seamless transition from TTree to RNTuple analysis with RDataFrame

Marta Czurylo[1], Andrii Falko[3], Danilo Piparo[1], Enric Tejedor Saavedra[1], Enrico Guiraud[1,2], Jakob Blomer[1], Philippe Canal[4], Vincenzo Eduardo Padulano[1]

[1]CERN, Esplanade des Particules 1, 1211 Geneva 23, Switzerland

[2]Princeton University, Princeton, NJ 08544, USA

[3]Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

[4]Fermi National Accelerator Laboratory, Batavia IL 60510-5011, the United States

E-mail: marta.maja.czurylo@cern.ch

Abstract. As the High-Luminosity LHC era is approaching, the work on the next-generation ROOT I/O subsystem, embodied by the development of RNTuple, is advancing fast with demonstrated implementations of the LHC experiments' data models and clear performance improvements over TTree data format. Part of RNTuple development is to guarantee no change in the RDataFrame analysis flow despite the change in the underlying data format. In this talk, we present integration of RNTuple and RDataFrame. The engine can process RNTuple datasets on a local machine, sequentially with one core or using implicit multithreading with multiple cores. Furthermore, RNTuple processing is also introduced in the distributed RDataFrame layer. The new workflow is demonstrated using existing RDataFrame analyses on one or multiple nodes with no change in the API. One notable example is the $t\bar{t}$ Analysis Grand Challenge benchmark, which is also used as a blueprint to showcase differences in performance of (distributed) execution with the two data formats.

1 Introduction

Many High Energy Physics data analyses depend on the ROOT software [1]. The ROOT native I/O subsystem for the Large Hadron Collider (LHC) Runs 1 - 3 is TTree. However, starting with LHC Run 4, TTree is going to be replaced with the new I/O subsystem in the form of RNTuple. The technical side of RNTuple and its readiness for production have been vastly demonstrated, for example, in [2] and [3]. In this contribution, the focus is given to the user side of the data format transition.

The users interact with ROOT data through an RDataFrame based analysis. The suggested way for users to manipulate, process and analyse ROOT data is through RDataFrame. RDataFrame is the high level interface offered by ROOT since 2018 (ROOT version 6.14) [4]. Without any modifications, the same analysis code can be executed on a single core, implicitly on multiple cores of the same machine or on many nodes of multiple machines using the Python-based distributed RDataFrame package [5]. It has been demonstrated in [6] that the distributed analysis works well by running a benchmark analysis code - Analysis Grand Challenge (AGC) [7]. The AGC provides a number of realistic analysis workflows for the testing and optimising purposes. In [6], the AGC version 0.1.0 is used and the analysis is run on an HPC system at CERN while the input data are stored in the native TTree format. Since then, the RDataFrame implementation of AGC was updated to version 1.0, and most notably, the new data format, the CMS NanoAOD is used now. In this publication, we perform the distributed AGC analysis in a modified manner which is closer to what an end user of the ROOT software, the physicist analysing the ROOT data, is going to experience. In particular, firstly, we run the analysis on the CERN web-based platform SWAN [8] which allows to seamlessly run jobs on the HTCondor pools [9] using the Dask scheduler from the Python Dask library. The details of the setup are described in Section 2. Secondly, the input data are converted into the RNTuple format. The impact of the new data format on the user's analysis as well as the scaling test performance comparison results are given in Section 3. Finally, we summarize the work done in Section 4.

2 Distributed analysis environment setup

As briefly mentioned in Section 1, the distributed analysis setup that is used in this study consists of a few key elements, which are part of production services developed and maintained at CERN.

We make use of the web-based platform, SWAN, which provides the access to the CERN's storage system EOS [10]. It also exposes curated software stacks deployed via CernVM File System (CernVM-FS) [11]. This provides users with all the software needed for the majority of the use cases. The analysis application can be offloaded to the CERN HTCondor pools in a user-friendly way via the Dask plugin available in SWAN. All these components used together, paired with the analysis software provided by ROOT, constitute an interactive analysis environment that could resemble the first step towards CERN Analysis Facility.

3 Distributed Analysis Grand Challenge with TTree and RNTuple

With the described distributed analysis setup, we can now move towards the comparison between an analysis run with data stored in the TTree format compared to the same analysis reading data stored in the RNTuple format. Here, we focus on the user side of transitioning to running the analysis with the RNTuple data as well as the preliminary performance comparisons between the data analysis using either of the two data formats. The current status of the analysis can be followed on the public repository [12].

3.1 User side

First of all, let us consider the user side. In order to run the AGC analysis, the user needs to specify a number of input arguments. One of those is the input dataset which we call *REMOTE_DATA_PREFIX* as the input AGC data for the RDataFrame implementation is stored on public EOS. In Listing 1, we can see that what needs to be specified in order to run the RDataFrame analysis with either TTree or RNTuple is only the path to the input dataset. In fact the rest of the code remains unchanged.

```
REMOTE_DATA_PREFIX:
str = "root://eospublic.cern.ch//eos/root-eos/AGC/ttree/"

REMOTE_DATA_PREFIX:
str = "root://eospublic.cern.ch//eos/root-eos/AGC/rntuple/"
```

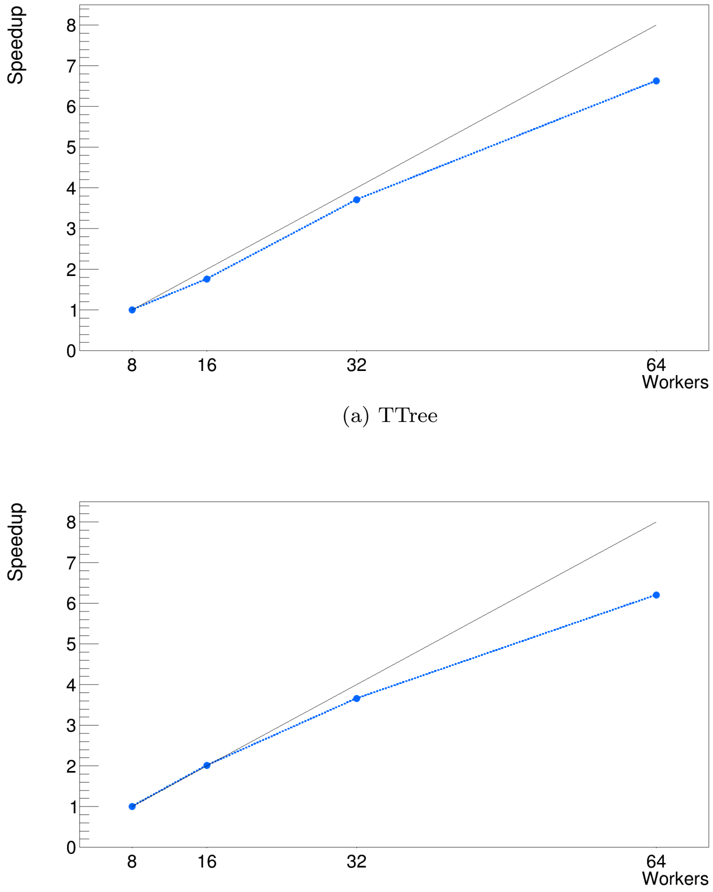
Listing 1: Setting the *REMOTE_DATA_PREFIX* variable to demonstrate the seamless transition from TTree to RNTuple for the user.

In order to compare whether the results of the AGC analysis implemented in the RDataFrame, with either TTree or RNTuple are correct, we perform histogram validation. Post-analysis, we compare 122 histograms with the equivalent histograms from other execution policies. We obtain satisfactory agreement with 120 histograms (100% of the bin-by-bin agreement), while for the remaining 2 histograms we obtain <1% disagreement because of the bin migrations.

3.2 Performance comparison - the scaling test

Finally, we want to compare the performance results between the two data formats. As the RNTuple binary format is being finalised[1] [2], we compare the speedup times of the two versions of the analysis (input data in TTree vs in RNTuple formats) rather than the absolute, end-to-end runtimes. The analyses are run on full datasets available while the number of workers, each with one core, are being gradually increased. The maximum number of workers in this study is 64.

[1] Provisionally this will happen at the end of 2024.



(a) TTree

(b) RNTuple

Figure 1: Speedup vs number of workers for the input data stored in (a) TTree and (b) RNTuple, measured while running the AGC analysis on SWAN web-based platform using the HTCondor pools scheduled via Dask. The blue line shows the measured and computed values while the black dashed line represents the perfect theoretical scaling.

Figure 1 shows the two plots of speedup vs number of workers for the two data formats, TTree at the top and RNTuple at the bottom, where speedup is relative to the measured time of running the distributed analysis with 8 workers. The blue line shows the calculated speedup based on the measured runtimes, while the black dashed line represents the ideal linear scaling that is theoretically expected. The error bars of the speedups are included but are not visible due to the scaling of the graph.

It can be observed that for both TTree and RNTuple, the scaling begins to deflect from the ideal starting at the 32 workers. For TTree the scaling for the 64 workers is at 6.6 instead of 8 while for RNTuple it is at 6.2.

The non-idealities shown in these plots can be accounted for by the analysis reading remote data from a highly congested storage system, using a shared HTCondor pool. The test setup is more geared towards interactivity and user-friendliness than pure performance. The actual performance improvements that RNTuple offers over TTree have been shown and demonstrated on many other occasions, for example, in [2], [3] and [13].

4 Conclusion

This contribution aims at showing a few aspects of the state-of-the-art data analysis with ROOT RDataFrame. The Analysis Grand Challenge is used as an example, so that the results can be verified and compared to other execution policies. We show that the distributed RDataFrame computation works well with both TTree and RNTuple, including the CERN infrastructure and resources which any user with a CERN computing account has access to. Even though the transition from TTree to RNTuple is non-trivial from the technical point of view [2], from the RDataFrame user perspective, such transition is seamless. The performance of the two data formats is also validated by measuring the speedup times of the distributed analyses.

Now that RDataFrame allows reading and processing RNTuple datasets, the focus will shift towards improving the I/O scheduling algorithms of the tool, as well as keeping up with the new features of the data format designed and developed according to the experiments requirements.

References

[1] Brun, R. and Rademakers, F. ROOT: An object oriented data analysis framework. *Nucl. Instrum. Meth. A*, 389:81–86, 1997. doi: 10.1016/S0168-9002(97)00048-X.

[2] Blomer, J., Canal, P., de Geus, F., Hahnfeld, J., Naumann, A., Lopez-Gomez, J., Lazzari Miotto, G., and Padulano, V.E. ROOT's RNTuple I/O Subsystem: The Path to Production. *EPJ Web of Conf.*, 295:06020, 2024. doi: 10.1051/epjconf/202429506020. URL `https://doi.org/10.1051/epjconf/202429506020`.

[3] Mete, A.S., Nowak, M., and Van Gemmeren, P. Persistifying the complex Event Data Model of the *ATLAS* Experiment in RNTuple, 2024. URL `https://indico.cern.ch/event/1330797/contributions/5796492/`.

[4] Piparo, D., Canal, P., Guiraud, E., Pla, X.V., Ganis, G., Amadio, G., Naumann, A., and Tejedor, E. RDataFrame: Easy Parallel ROOT Analysis at 100 Threads. *EPJ Web Conf.*, 214:06029, 2019. doi: 10.1051/epjconf/201921406029. URL `https://doi.org/10.1051/epjconf/201921406029`.

[5] Padulano, V.E., Kabadzhov, I., Saavedra, E., Guiraud, E., and Alonso, P. Leveraging State-of-the-Art Engines for Large-Scale Data Analysis in High Energy Physics. *Journal of Grid Computing*, 21, 02 2023. doi: 10.1007/s10723-023-09645-2.

[6] Padulano, V.E., Guiraud, E., Falko, A., Gazzarrini, E., Garcia Garcia, E., and Gosein, D. First implementation and results of the Analysis Grand Challenge with a fully Pythonic RDataFrame. *EPJ Web of Conf.*, 295:06011, 2024. doi: 10.1051/epjconf/202429506011. URL `https://doi.org/10.1051/epjconf/202429506011`.

[7] Held, A. and Shadura, O. The IRIS-HEP Analysis Grand Challenge. page 235, 11 2022. doi: 10.22323/1.414.0235.

[8] Piparo, D., Tejedor, E., Mato, P., Mascetti, L., Moscicki, J., and Lamanna, M. SWAN: a Service for Interactive Analysis in the Cloud. *Future Gener. Comput. Syst.*, 78:1071–1078, 2018. doi: 10.1016/j.future.2016.11.035. URL `https://cds.cern.ch/record/2158559`.

[9] Thain, D. and Tannenbaum, T. and Livny, M. Distributed computing in practice: the Condor experience. *Concurrency - Practice and Experience*, 17(2-4):323–356, 2005.

[10] Peters, A.J., Sindrilaru, E.A., and Adde, G. EOS as the present and future solution for data storage at CERN. *J. Phys.: Conf. Ser.*, 664(4), 201. doi: 10.1088/1742-6596/664/4/042042. URL `https://cds.cern.ch/record/2134573`.

[11] Blomer, J., Buncic, P., Ganis, G., Hardi, N., Meusel, R., and Popescu, R. New directions in the CernVM file system. *J. Phys.: Conf. Ser.*, 898(6):062031, 2017. doi: 10.1088/1742-6596/898/6/062031. URL `https://cds.cern.ch/record/2297056`.

[12] The ROOT Team. RDataFrame implementation of the Analysis Grand Challenge. URL `https://github.com/root-project/analysis-grand-challenge`. Accessed: 30th July 2024.

[13] Naumann, A., Canal, P., Tejedor, E., Guiraud, E., Moneta, L., Bellenot, B., Couet, O., Tadel, A.M., Tadel, M., Linev, S., Lopez Gomez, J., Rembser J., Padulano, V.E., Blomer, J., Hahnfeld, J., Gruber, B.M., and assilev, V. ROOT for the HL-LHC: data format. 2022. URL `https://arxiv.org/abs/2204.04557`.