Storing LHC Data in DAOS and S3 through RNTuple

26^{th} International Conference on Computing in High Energy and Nuclear Physics

Giovanna Lazzari Miotto – Federal University of Rio Grande do Sul (BR)
Javier López-Gómez – CERN (CH)

CHEP 2023, 9 May 2023

Introduction

ROOT RNTuple

HL-LHC is projected to increase ROOT data at least tenfold

- **New HW landscape:** NVMe devices, architectural heterogeneity, parallelism, distributed facilities. . .
- **RNTuple** (“TTree 2.0”) is optimized to accommodate this leap!

ROOT RNTuple

HL-LHC is projected to increase ROOT data at least tenfold

- **New HW landscape:** NVMe devices, architectural heterogeneity, parallelism, distributed facilities. . .
- **RNTuple** ("TTree 2.0") is optimized to accommodate this leap!

RNTuple goals:

- Over 2× better single-core performance
- 1 GB/s/core sustained end-to-end throughput
- 25% smaller files
- Robust interfaces and systematic use of exceptions
- . . . and more: check yesterday's session! ▸ talk: RNTuple ▸ talk: RNTuple in Athena

Why Object Stores

In a highly-parallel setting, object stores align well with our requirements:

- Extremely scalable
- Widely deployed in cloud service providers

Contexts: **HPC** (Intel DAOS) = **Cloud** (Amazon S3, Microsoft Azure, Google Cloud)

Why Object Stores

In a highly-parallel setting, object stores align well with our requirements:

- Extremely scalable
- Widely deployed in cloud service providers

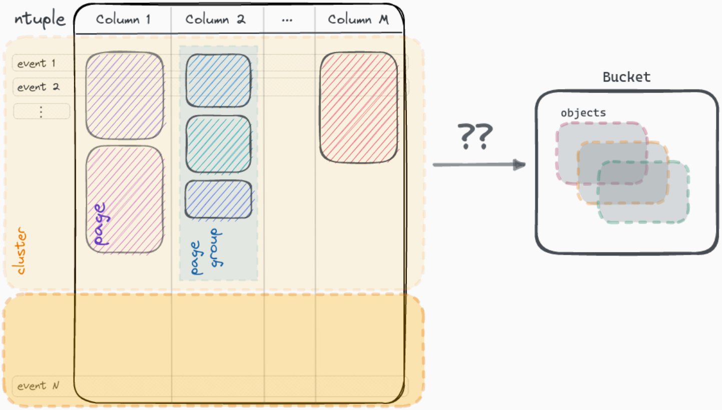
Contexts: **HPC** (Intel DAOS) = **Cloud** (Amazon S3, Microsoft Azure, Google Cloud)

But why invest in native support? **Granular parallel access**

- **Downside:** no storage of other ROOT classes: histograms, etc.

Challenge: Mapping RNTuple → Object Stores

Granularity: **page** $O(KiB)$, **page group**, **cluster**[a] $O(MiB)$



Factors to consider

- Analysis pattern
- Throughput, latency
- Cost per request
- Memory consumption

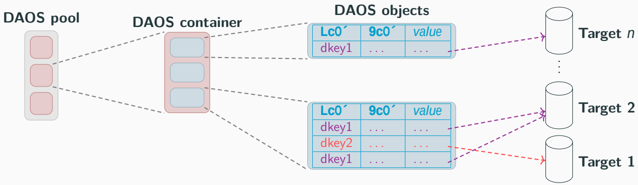
[a] For columnar access, storing clusters wholesale requires byte ranges support

Storing RNTuple data in DAOS

DAOS: Fault-tolerant object store for HPC

Foundation of the Intel exascale storage stack

- Built for NVMe and persistent memory
- Low-latency, high-bandwidth, high IOPS
- Used in 44% of the top 25 systems in IO500[1] (e.g., **ANL Aurora**)



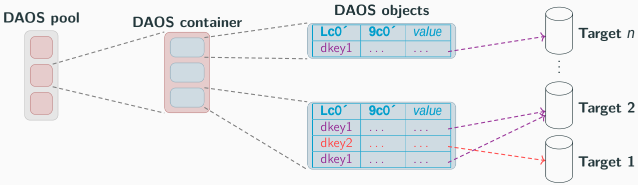
- Locality feature: *object*, *dkey* → target

[1] \§§q I¤¤_nU··un}X¤

DAOS: Fault-tolerant object store for HPC

Foundation of the Intel exascale storage stack

- Built |KSIIN²q0}_h0j§9eIIS}nh|i¦«qe0rzK0H9´¦}00zJ zL9n I¤¤h´]qnne¤h´]Hnj§9_j0}z
- Low-latency, high-bandwidth, high IOPS
- Used in 44% of the top 25 systems in IO500[1] (e.g., **ANL Aurora**)



- Locality feature: *object*, *dkey* → target

[1] \§§q I¤¤_nU··un}X¤

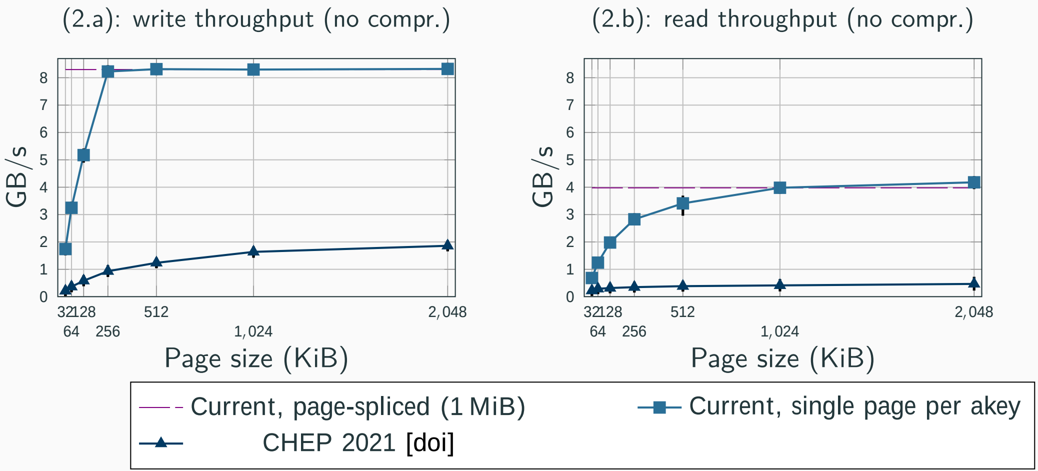
RNTuple ↔ DAOS

RNTuple ↔ DAOS

"**Page splicing**": empirically optimal with 1 MiB blobs (throughput × granularity)

DAOS backend evaluation: LHCb

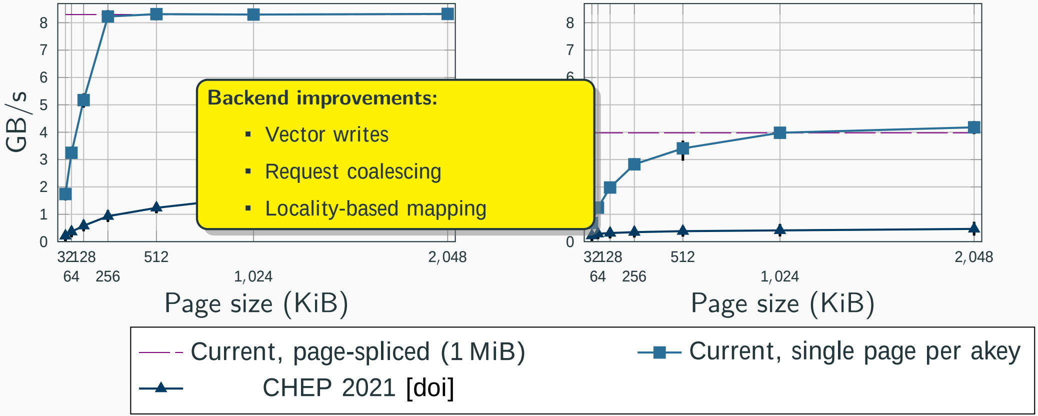
E2E analysis on single DAOS client-server pair (**HPE**'s Delphi cluster w/ **Infiniband** interconnect)



DAOS backend evaluation: LHCb

E2E analysis on single DAOS client-server pair (**HPE**'s Delphi cluster w/ **Infiniband** interconnect)

(2.a): write throughput (no compr.)   (2.b): read throughput (no compr.)

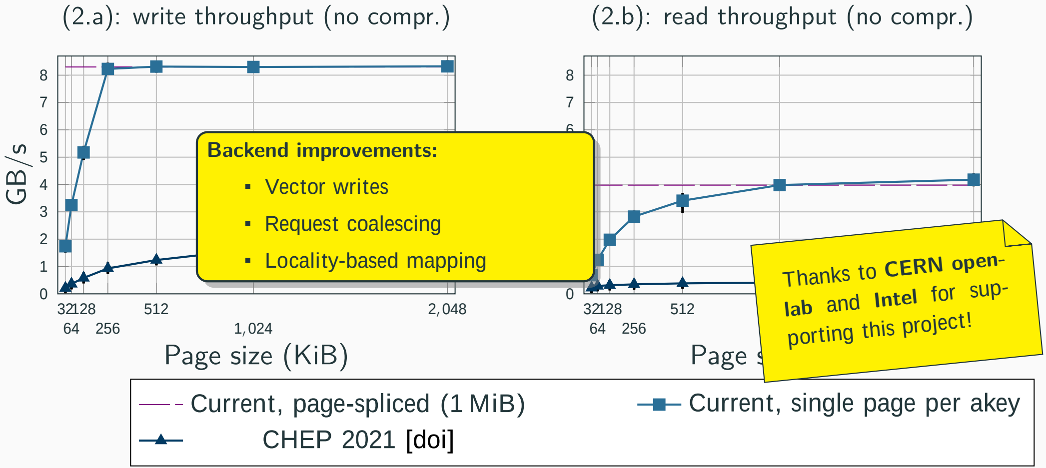


Backend improvements:

- Vector writes
- Request coalescing
- Locality-based mapping

DAOS backend evaluation: LHCb

E2E analysis on single DAOS client-server pair (**HPE**'s Delphi cluster w/ **Infiniband** interconnect)



Backend improvements:

- Vector writes
- Request coalescing
- Locality-based mapping

Thanks to **CERN openlab** and **Intel** for supporting this project!

Storing RNTuple data in S3

Amazon S3

Object storage's de facto standard

Amazon S3

Object storage's de facto standard

Main differences w.r.t. DAOS:

- Flat namespace
- Worldwide server distribution across "regions" and "edges"
- Potential interop with research computing infrastructures, e.g., WLCG
- Network latency...

AWS edge locations and regional caches. Amazon.

Object storage's de facto standard

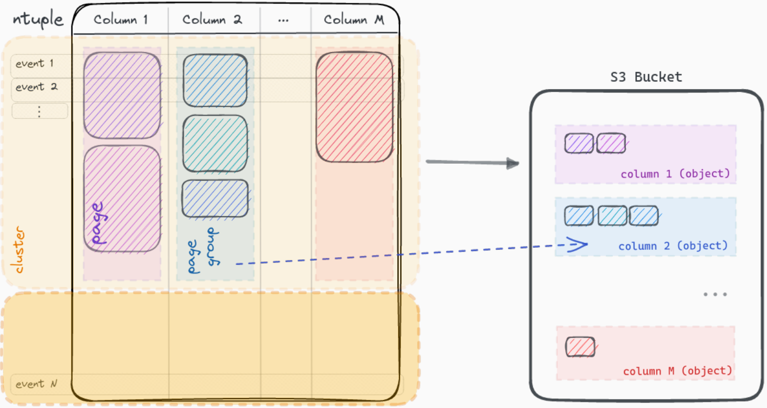
Main differences w.r.t. DAOS:

- Flat namespace
- Worldwide server distribution across "regions" and "edges"
- Potential interop with research computing infrastructures, e.g., WLCG
- Network latency...

AWS edge locations and regional caches. Amazon.

|KSIIN²qO}_hOj§9eIIS}nh|i¦«qeOrzKOH9´¦}OOzJ z ¨I¤¤9°]}OX_nj¤h´]@«HcO§z

RNTuple ↔ S3



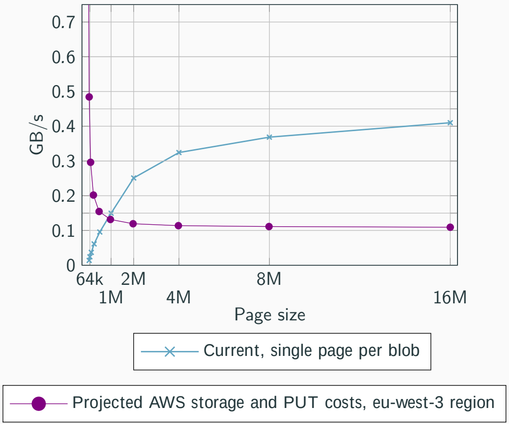
Bigger blobs (**page groups** or **clusters**) may mitigate latency

- Best with scatter-gather I/O, requires **|9jX0** HTTP requests

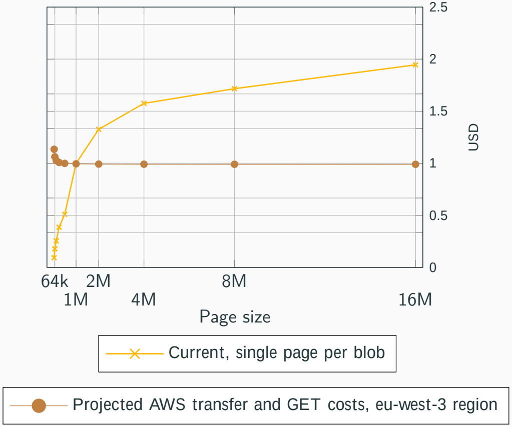
S3 backend evaluation: LHCb

E2E analysis on client-server pair (MinIO). Nodes provided by openlab and ROOT.

Plot (1.a): write costs, throughput (no compr.)



Plot (1.b): read costs, throughput (no compr.)



On data migration

RNTuple Migrator

RNTuple performance is very sensitive to its I/O parameters ▸ poster: ML + RNTuple

- Even between two object stores!
- Moving data between storage systems requires reshaping for efficient access

Implemented

Fast cloning with field selection (like **¦¦}OO**'s **GenjO¦}OO**)

- Avoids page decompression unless necessary

In development

Reshaping ntuples

- Compression algorithm, page size, cluster size, . . .

Summary

In conclusion

- RNTuple is set to be production-ready in late 2024, leveraging **HPC** and **cloud** object stores
  - **Native**, **mature DAOS backend** with 8+ GB/s writes, 4+ GB/s reads single-node
  - **Native**, **experimental S3 backend**
- Efficient data migration across storage systems, ntuple reshaping (WIP)

In conclusion

- RNTuple is set to be production-ready in late 2024, leveraging **HPC** and **cloud** object stores
  - **Native**, **mature DAOS backend** with 8+ GB/s writes, 4+ GB/s reads single-node
  - **Native**, **experimental S3 backend**
- Efficient data migration across storage systems, ntuple reshaping (WIP)

Next steps

- Optimize S3 backend based on first results
  - Investigate AWS' C++ SDK, retaining Davix as a compatibility fallback
- Expand RNTuple Migrator

Thank you!

CHEP 2023

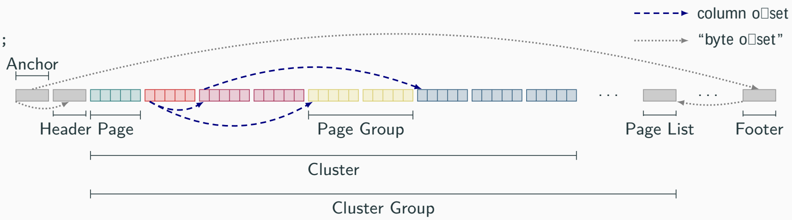
Questions?

glazzari@cern.ch

Acknowledgements: This work benefited from support by the CERN Strategic RD Programme on Technologies for Future Experiments CERN-OPEN-2018-006 and the Intel–CERN openlab collaboration. Access to the hardware for the experimental evaluation was provided by Hewlett-Packard Enterprise.

Breakdown of the RNTuple On-Disk Format

```
struct Event {
  int fId;
  vector<Particle> fPtcls;
};
struct Particle {
  float fE;
  vector<int> fIds;
};
```



Cluster

- Block of consecutive complete events
- Defaults to 50 MB compressed

Page

- Unit of (de-)compression
- Defaults to 64 KiB uncompressed
- Not necessarily aligned on event boundary

▸ Format specification