ATLAS Tape Archival Metadata

NPPS Group Meeting, April 8th, 2026

Xin Zhao (BNL), Maria Grigoryeva (SAPHIR), Alexei Klimentov (BNL)

Outline

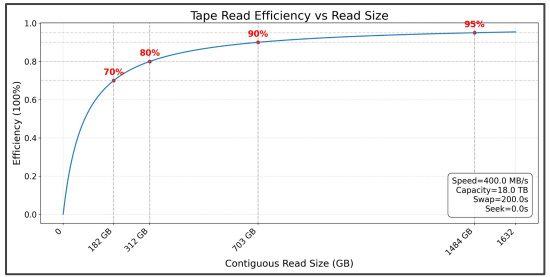
- Introduction – why do we need archival metadata
- Analysis of archival metadata based on ATLAS Run3 recall history
- Preliminary results
- Open questions
- Current status

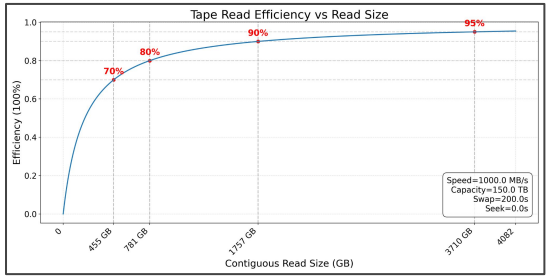
File grouping for efficient tape usage

- Efficient tape bandwidth utilization is crucial for the success of Data Carousel esp. for HL-LHC.
- One key strategy to improve tape efficiency is to co-locate files that are likely to be recalled together

Why ?

- Tape is bad for random access but good at streaming read (~400MB/s for today's tape drive)
- Two main factors affecting tape read efficiency : seek and swap
  - seek → drive head moves between target files
  - swap → mount/dismount tape cartridges
- File grouping addresses both of these factors





Good recall efficiency per tape mount – time spent on reading : should dominate the total time

$$E_{\text{recall}} = \frac{T_{\text{read}}}{T_{\text{read}} + T_{\text{seek}} + T_{\text{swap}}}$$

Archival Metadata

How do we know which files should be co-located ? ⇒ Archival Metadata

- To encode grouping hints on a per file basis
- A hierarchical structure in json format,
  - proposed by storage providers (CTA and dCache teams)

It's the experiments responsibility to provide archival metadata when writing files to tape, as guidance for sites to group files on tape.

But firstly, experiments themselves need to understand how they recall data from tape!

```
archive_metadata = {
    "scheduling_hints": {
        "archive_priority": "100"           # highest priority
    },
    "collocation_hints": {
        "0": "data23_13p6TeV",                                  # project
        "1": "RAW",                                             # datatype
        "2": "physics_Main",                                    # stream_name
        "3": "data23_13p6TeV.00452799.physics_Main.daq.RAW",    # dataset
    },
    "additional_hints": {
        "activity": "T0 Tape",        # Tier-0/DAQ
        "3": {                        # dataset level
           "length": "19123",             # total number of files at specified level
           "size": "80020799318456"       # total size of files at specified level
        }
    },
    "file_metadata": {                  # file content metadata
        "size":"193734404",
        "adler32":"379ebf71",
        "md5":"952c4c0dabc622a94f09b053d71d0dfb"
    }
}
```

Archival Metadata derived from ATLAS recall history

- Data sample
  - ATLAS recall history during Run3
    - collected from ATLAS ElasticSearch
    - totally ~200k datasets since 2021-12 (gap in early 2025 due to transition of Data Carousel machinery to ProdSys to PanDA
  - The timestamp used is when Data Carousel initially created the Rucio rule to recall a dataset, reflecting the real picture of what datasets ATLAS intended to recall together. This is better than timestamps obtained from downstream layers like FTS and site storage systems.
- Each dataset name is splitted into individual metadata fields (or so called “feature” in data mining term), following the ATLAS dataset nomenclature. e.g.

“data15_13TeV:data15_13TeV.00270949.physics_Main.daq.RAW” →

scope = data15_13TeV
project = data15_13TeV
runNumber = 00270949
streamName = physics_Main
productionStep = daq
dataType = RAW

- A daily time window used – datasets requested on the same day are considered as “recalled together”

Analysis of tape recall history – Entropy (1/2)

- After exploring various analysis methods and techniques, e.g. Apriori algorithm for association rule mining and contrastive learning, we find that calculating entropy is an effective way to quantify the (un)certainty in how datasets are grouped by each feature.
- formula
  - For example, to study how datasets are spread around the “streamName” feature on a day, its daily entropy is calculated as :

$$\text{Entropy} = -\sum p_i \log(p_i)$$

where Pi is the fraction of datasets for streamName i

$$p_i = \frac{\text{number of datasets with streamName i}}{\text{total datasets that day}}$$

If entropy is 0, meaning all datasets share one single streamName on that day, very focused;

If entropy is high, datasets are spread across multiple feature values, more diverse and “unpredictable”

Analysis of tape recall history – Entropy (2/2)

For a given data type, for each feature:

- we count the total number of unique values of this feature in the recall history
- for each day, compute raw entropy and normalized entropy (by its max value)
- take daily average (mean+/- std)

Raw entropy indicates hierarchical levels (affected by cardinality);

Normalized entropy indicates grouping strength among the features within the same level

Table below shows the results for data AOD, which suggests the following archival metadata hierarchy (from high to low levels):

a. dataType (AOD)
b. streamName (e.g. physics_Main)
c. project (e.g. data16_13TeV)
d. amiTag (e.g. r9264_p3083)
e. dataset

Summary Statistics for Entropy of Target Features

Feature	Unique Values	Entropy	Normalized Entropy
project	25	0.69±0.70	0.15±0.15
runNumber	3435	4.02±2.34	0.34±0.20
StreamName	24	0.33±0.51	0.07±0.11
amiTag	390	1.34±1.18	0.16±0.14
tid	8826	2.88±2.45	0.22±0.19

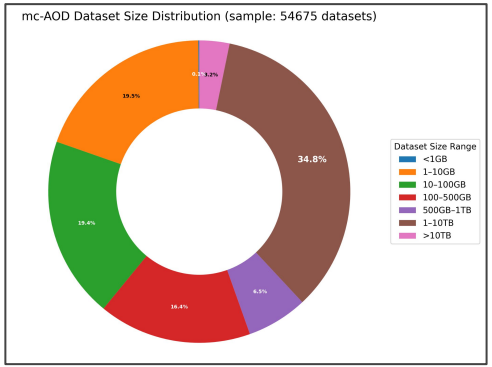
Preliminary Results – Grouping Hierarchy by Data Type

AOD/data	AOD/mc	RAW	HITS	EVNT*
- **dataType** (AOD) - **streamName** (e.g. physics_Main) - **project** (e.g.data16_13TeV) - **amiTag** (e.g. -r9264_p3083) - **dataset**	- **dataType** (AOD) - **project** (e.g. mc16_13TeV) - **amiTag** (e.g. e4944_e5984_s3126_r10724_r10726) - **dataset**	- **dataType** (RAW) - **streamName** (e.g. physics_Main) - **project** (e.g. data16_13TeV) - **dataset**	- **dataType** (HITS) - **project** (e.g. mc16_13TeV) - **amiTag** (e.g. e7307_e5984_s3126) - **dataset**	- **dataType** (EVNT) - **project** (e.g. mc16_13TeV) - **amiTag** (e.g. e7142_e5984) - **dataset**

(may not be tape-resident in the future)*

Open question – grouping size ? (1/3)

- Grouping size is an important metadata
  - defines the boundary of groups
  - sites use it to decide how many tapes a group of files should be split into, to balance between per-tape-mount efficiency and overall throughput
- Size info well known at dataset level
- For above-dataset levels, group boundary is hard to define
  - e.g. “group all physics_main streams from all years together” – too broad to be practical (not actionable by sites)
- Why do we need to worry about above-dataset grouping levels ? – ATLAS has a lot of small datasets (more on backup slide)
  - define “small” ? – refer to the efficiency curve (slide 2)



Open question – grouping size ? (2/3)

- Option 1
  - Experiment creates “artificial supercontainers” for tape
    - Following the hierarchy, especially the level right above the dataset, create “artificial supercontainers” that groups certain small datasets together
  - Cons:
    - Experiment (users) needs to respect these artificial supercontainers when recalling them back
      - less of a concern for big campaigns than for user analysis jobs
- Option 2
  - Volume wise, small datasets take a small portion (~5%?) of the whole ATLAS data sample on tape, maybe simply keep them on disk
  - can still put them on tape, but for archival purpose only so little impact on recall efficiency
  - Cons
    - disk space
    - cold data on disk

Open question – grouping size ? (3/3)

- Option 3 – do nothing ?
  - volume wise, small datasets are “small” → less impact on the overall recall efficiency
    - take the mc23_13p6TeV AOD datasets as an example (tables below)
    - overall recall efficiency (assume the whole sample is recalled at once) is dominated by the fewer big datasets (table to the left)
  - Cons
    - Not future-proofing, small datasets overhead becomes more visible as tape speed increases (table to the right)
    - not deterministic : sites want experiment to “target good recall efficiency per mount” (WLCG OTF #8)

Assumptions:
- Read speed: 400MB/s
- Swap time: 200s per tape mount

size range [GB]	total size [TB]	# dst/grp	<size> [GB]	total read [Ms]	total mount [Ms]	read eff [%]
0 - 100	392	25,431	15	0.98	5.09	16
100 - 500	550	2,357	233	1.38	0.47	74
500 - 1000	520	674	772	1.30	0.13	91
> 1000	33,881	2,943	11,512	84.70	0.59	99
all	35,343	31,405		88	6.3	93

Assumptions:
- Read speed: 1000MB/s
- Swap time: 200s per tape mount

size range [GB]	total size [TB]	# dst/grp	<size> [GB]	total read [Ms]	total mount [Ms]	read eff [%]
0 - 100	392	25,431	15	0.39	5.09	7
100 - 500	550	2,357	233	0.55	0.47	54
500 - 1000	520	674	772	0.52	0.13	79
> 1000	33,881	2,943	11,512	33.88	0.59	98
all	35,343	31,405		35	6.3	85

For (mc23_13p6TeV.*.merge.AOD.*) datasets sample (courtesy of Luc Goossens). Calculation assumes each dataset on its own tape cartridge.

Other open questions

- Besides dataset nomenclature, are there other meaningful metadata levels ?
  - one example is “request ID” from ProdSys
    - it has the same size boundary issue as the other above-dataset levels (e.g. amiTag)
- “Shadow of experiment (disk file) lifetime model”
  - experiments shuffle data between disk and tape constantly. Campaigns can get data from both disk and tape.
  - recall history may not show the complete workflow intent. But it’s still the closest truth we can get today.

Current status

- Preliminary archival metadata has been coded into Rucio FTS transfer requests, being sent to all tape sites when writing data to tape.
  - Recently dCache added support for archival metadata (release 11.2.1)
- Sites have started to use archival metadata
  - KIT Tier-1 to switch from legacy metadata communication channel to using the archival metadata
  - CERN CTA is carrying out their own studies on tape grouping for ATLAS based on the archival metadata
    - tape simulator
- ATLAS
  - continue to evaluate different options for the open questions
  - it will help a lot if ATLAS has an (updated) estimate on tape thoughput for HL-LHC

Backup Slides

Size of ATLAS datasets

- Statistics of dataset size distribution for various data types, collected from the recall history sample – a lot of small datasets