dCache.org

Improving Performance of Tape Restore Request Scheduling in the Storage System dCache

Lea Morschel (DESY)
for the dCache Workshop

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

About dCache

dCache.org

- **Storage solution for scientific data**
- Joint effort between DESY, FNAL and NDGF
- Developed for HERA and Tevatron, used by LHC and others:
  - → WLCG, Belle II, LOFAR, CTA, IceCUBE, EU-XFEL, Petra3, DUNE, and many more

Data Management & Workflow Control

dCache.org

High Speed
Data Ingest

Batch processing

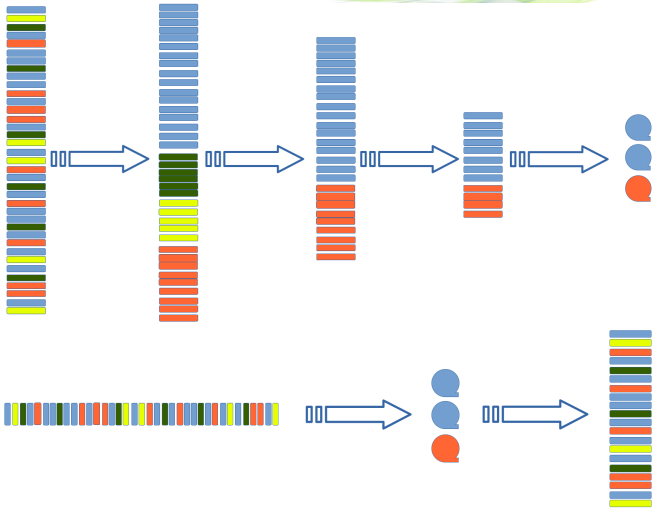
dCache

Interactive analysis

tape

dCache and Tape: Status Quo

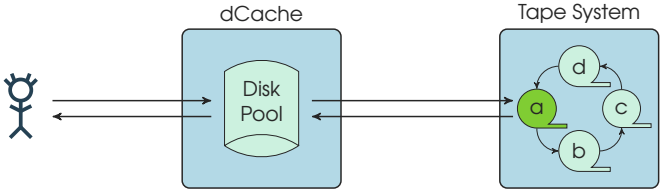
- **Flush to tape**:
  - Group requests by storage class
  - Flush each storage class to set of dedicated tapes
- **Recall from tape**:
  - Assumption: infrequent, likely same storage class $\longrightarrow$ Recall according to FIFO



ATLAS Data Carousel

- **Problem**:
  - Large discrepancy between future affordable and required disk storage resources for LHC experiments.
    → ATLAS: 700 PB vs 4.2 EB
    → Need to reduce storage costs!

- **Approach**: Use cheaper magnetic tape storage more actively, improve caching efficiency
  - Recall relevant data from tape, keep on disk for a while, remove, repeat

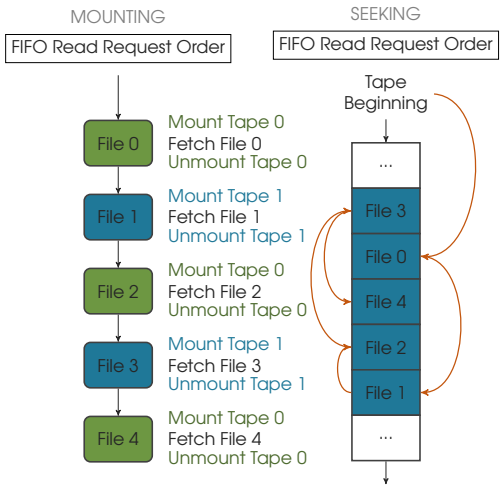


Magnetic Tape Storage Characteristics

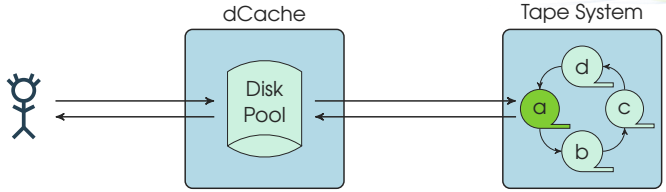
- **Tape Storage**:
  - Tapes: Offline, fetch + mount in a drive for access
  - On-tape seeks and rewinds lead to high access latency and decreased longevity

⟶ High I/O latency (mounts, seeks, rewinds), high transfer speeds

→ Optimal access: Reading large amounts of data sequentially



Where to Optimize Tape Recalls?



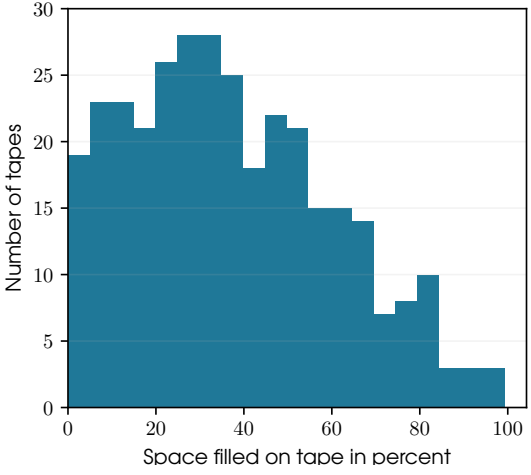
- **Main problem of active tape usage:**
  - Chaotic access to tape-resident data is highly inefficient!
- **Why not leave the optimization to the tape system?**
  - "Stupid" tape systems' optimization is limited (/to a single session)
  - dCache needs to reserve space for files requested from tape

→ The TS knowing a small subset of requests: **suboptimal reordering!**

⟶ CLUSTER REQUESTS BY TAPE in dCache!

Case Study KIT, Reprocessing Campaign Jan. 2020

- Received
  - Mapping of files to tapes
  - Dataset association
  - Planned recall sequence
  - Request queue length
  - Hardware setup
- Overall:
  - 1.1 PB
  - **495,049 files**
  - **35 datasets**

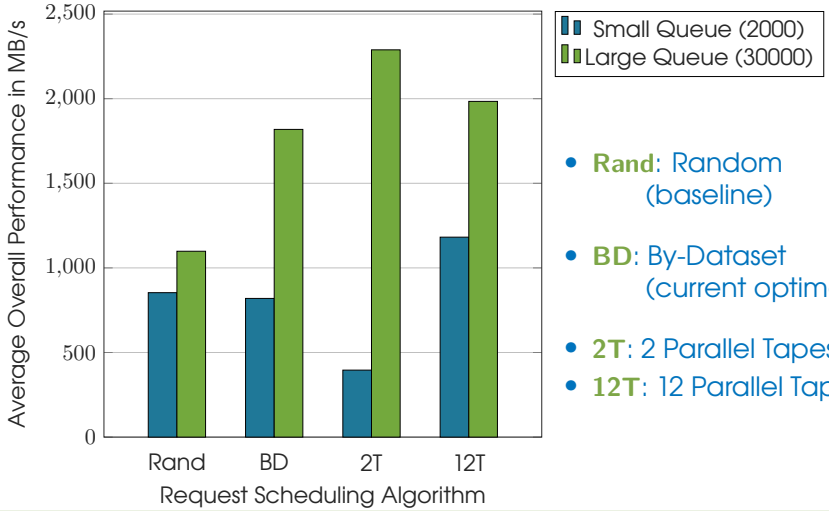


Goal and Model

- **Goal**:
  - Evaluate **READ performance** and **remounts**
  - Understand influence of **queue size** and **tape selection sequence**
- **Model**:
  - Components: Tape silo, tape drives, request queue, requestor
  - **Seeking** performance **approximated** by percentage of tape **volume and number of files recalled**
- **Setup** according to KIT Tier 1:
  - Hardware: **12 tape drives** T10KD (max. 327 MB/s per drive)
  - Tape system request queue sizes: **2k, 30k**
  - Data distribution on tapes like in Jan. 2020 data carousel exercise: $\approx$ **500K files,** 1.1 PB**, on 332 tapes**

Simulation Results

Evaluating Performance



- **Rand**: Random (baseline)
- **BD**: By-Dataset (current optimal)
- **2T**: 2 Parallel Tapes
- **12T**: 12 Parallel Tapes

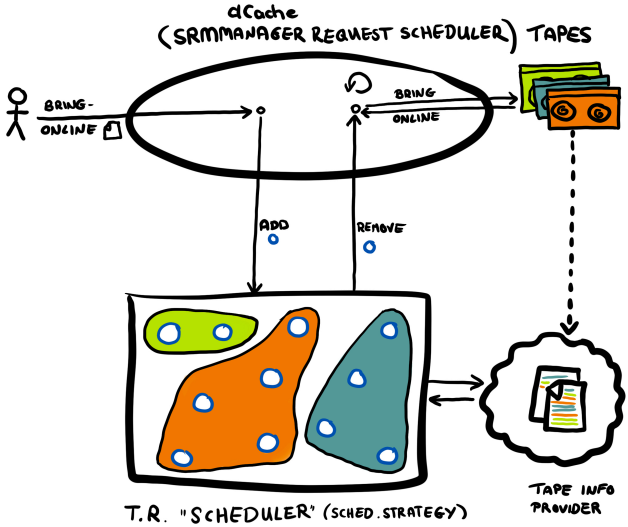
Proof of Concept Implementation

Bring-Online Scheduling in dCache

- Clustering requests by tape **most useful for large bulk recalls**
  → **SRM** is the current de facto standard for bulk tape recalls in dCache
- The **tape location information is unknown to dCache**
  → Create a way for **admins** to **provide them** (currently files)
- Behavior needs to be **dependent on site and tape system setup**
  → Configuration options:
  - `Number of tape drives`
  - `Min. recall percentage before selecting tape`
  - `Min. request count before selecting tape`
  - `Min. time since last request arrival for tape`
  - `Max. time in queue before tape selection`

Proof of Concept Implementation

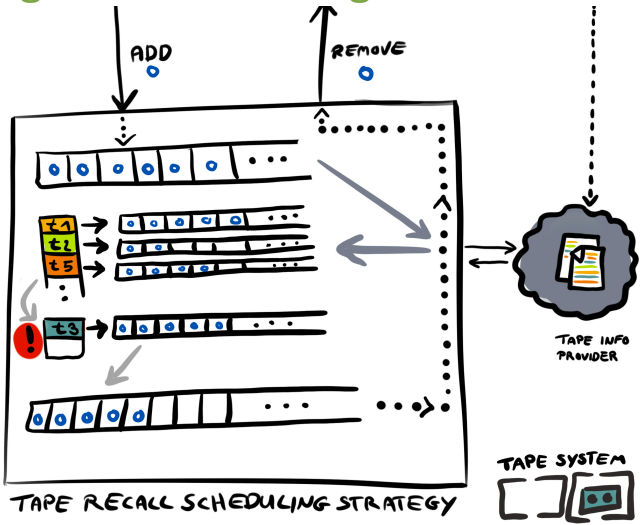
Bring-Online Scheduling in dCache



- Exclusively for `bring-online` requests
- First **requests** are associated with and **clustered by tape** info
- Then **tapes** are **selected and passed on** based on configured parameters

Proof of Concept Implementation

Bring-Online Scheduling in dCache



- Tape selection:
  - Tapes with **expired requests**? Choose oldest
  - Else: Filter tapes with **enough time since last req. arrival**
  - Check **recall volume**, else **job count sufficient**

  → Several tapes can be active, but none have to be!

Configuration

- The scheduler resides in the **SrmManager** and can be configured via config file:

```
[...Domain/srmmanager]
# ...
srmmanager.plugins.enable-bring-online-clustering = true

srmmanager.boclustering.max-active-tapes = 1
srmmanager.boclustering.min-tape-recall-percentage = 60
srmmanager.boclustering.min-request-count-for-tape = 1000

srmmanager.boclustering.min-time-since-last-tape-request = 1
srmmanager.boclustering.min-time-since-last-tape-request.unit = MINUTES

srmmanager.boclustering.max-time-in-queue = 15
srmmanager.boclustering.max-time-in-queue.unit = MINUTES
```

Tape Location Information Files

- The scheduler needs two tape-info files at **/etc/dcache/tapeinfo**
  1. **tapes.txt** – a line for each tape to be recalled from – with line format `<tapename>,<capacity in KB>,<used space in KB>`

```
1 tape1,6000000000,6000000000
2 tape2,8500000000,3000000000
3 tape3,8500000000,1111111111
4 tape4,4500000000,10000000
```

Tape Location Information Files

- The scheduler needs two tape-info files at **/etc/dcache/tapeinfo**
  1. **tapes.txt** – a line for each tape to be recalled from –
     with line format `<tapename>,<capacity in KB>,<used space in KB>`
  2. **tapefiles.txt** – a line for each file to be recalled –
     with line format `<filename>,<filesize in KB>,<tapename>`

```
1 /tape/file-0.log,100000,tape1
2 /tape/file-1.log,5000,tape1
3 /tape/file-2.log,3300,tape1
4 /tape/file-3.log,11000,tape2
5 /tape/file-4.log,10000000,tape3
6 /tape/file-5.log,2001,tape3
7 /tape/file-5.log,100,tape4
```

Conclusions

- Simulation:
  - **Significant improvements (performance, mounts) possible** by clustering per tape & activating as many tapes as there are drives before sending to tape system
  - **Tape system queue size major general influence**. Optimally fit at least 80 % of requests per tape/drive
- Proof of concept scheduler implementation:
  - **Behaviour configurable** to fit different needs
  - **First successful tests** using DESY tapes

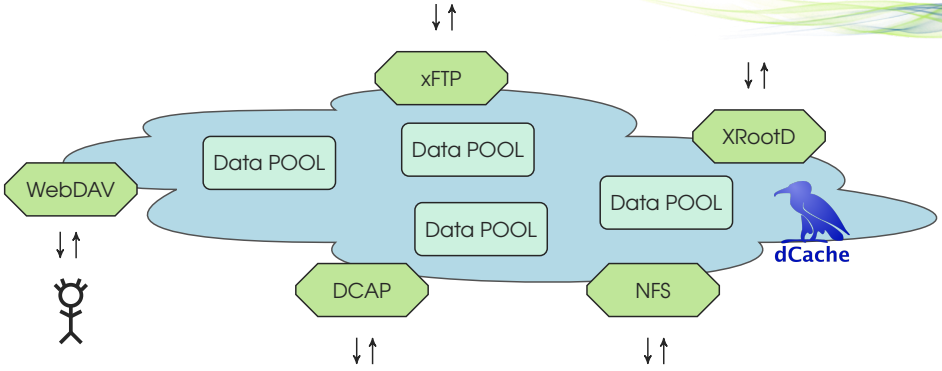
Outlook

- Deploying and testing the new component **at KIT** in the near future in order to **evaluate its impact in a realistic environment** (ATLAS data carousel)
- **Move scheduler** to a more central component within dCache to be **usable by every protocol**

Thank you for listening.

Questions?

About dCache

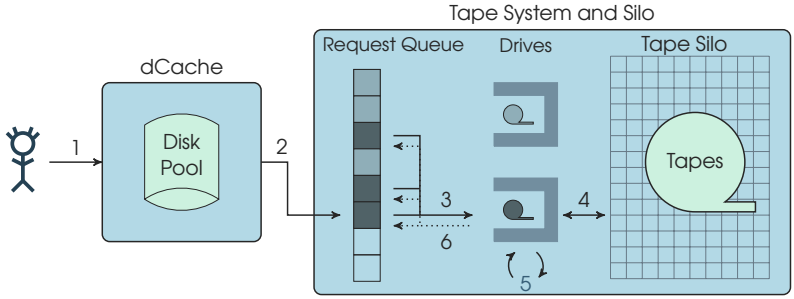


- **Distributed** and **hardware-agnostic** under a **single virtual file system tree**
- Supports **standard** and **HEP specific access protocols** and **authentication** mechanisms

Magnetic Tape Storage Characteristics

- **Tape System** – Robotically operated tape storage:
  - Hardware: Tape library, robots for fetching tapes, drives for access
  - Software: Request queue, scheduling decisions

⟶ **High level API for requesting files, variable internals**



Tape Format Characteristics

Tape Drive	Developer	Release Date	Capacity Native (Compressed)	Max. Speed Native (Compressed)
T10000C	Oracle	2011	5.0 TB	240 MB/s (360 MB/s)
T10000D	Oracle	2013	8.5 TB	252 MB/s (800 MB/s)
LTO-6	LTO Consortium	2012	2.5 TB (6.25 TB)	160 MB/s (400 MB/s)
LTO-7	LTO Consortium	2015	6.0 TB (15 TB)	300 MB/s (750 MB/s)
LTO-8	LTO Consortium	2017	12 TB (30 TB)	360 MB/s (900 MB/s)
TS1155	IBM	2017	15 TB	360 MB/s (800 MB/s)
TS1160	IBM	2018	20 TB (60 TB)	400 MB/s (900 MB/s)

Motivation

Future ATLAS Data Challenges

- **Resources**:
  > 350,000 Cores,
  > 270 PB storage
  at 150 WLCG sites
- Over 250k - 300K jobs routinely run in parallel
- Complex workflow and data management software
- **Upgrade of LHC for Run 4**: Large discrepancy between needed and available resources!

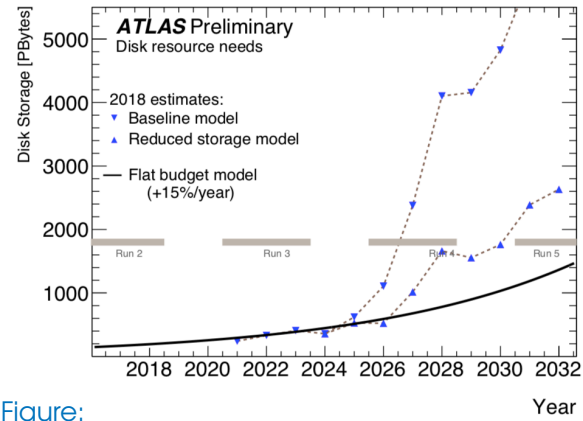
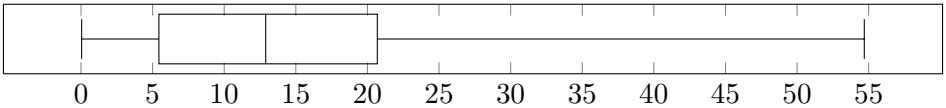


Figure:

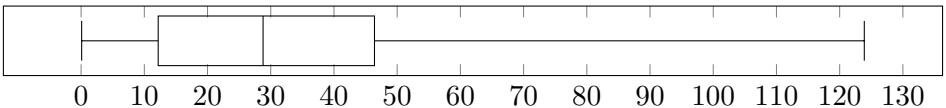
https://twiki.cern.ch/twiki/pub/AtlasPublic/ComputingandSoftwarePublicResults/diskHLLHC_18.png

ATLAS Data Carousel Exercise January 2020

- Overall: 18 PB, 8,1 Million files, 595 datasets
- Average dataset file count: 13,600
- Average dataset size: 30.7 TB
- Average file size: 2.3 GB



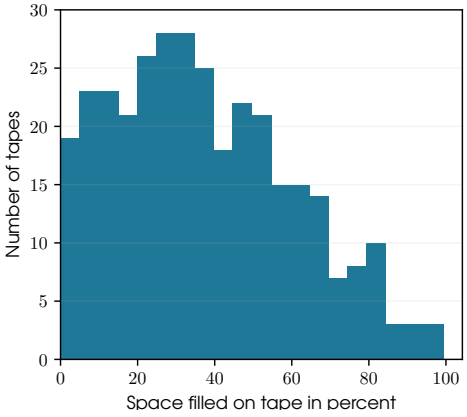
File Count in Dataset / 1000

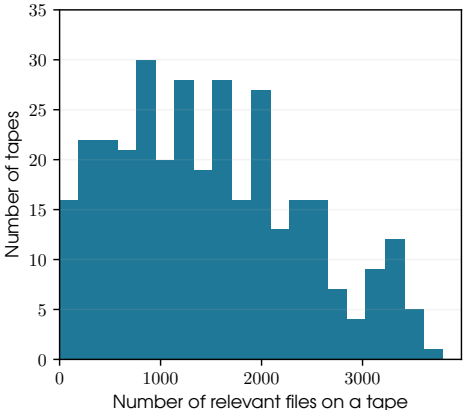


Dataset Size in TB

Case Study KIT, Reprocessing Campaign Jan. 2020

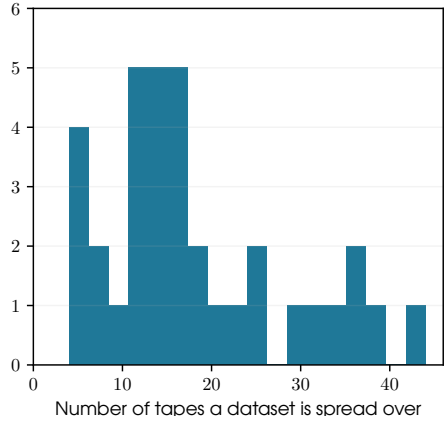
- Overall: 1.1 PB | **495,049 files** | **35 datasets**

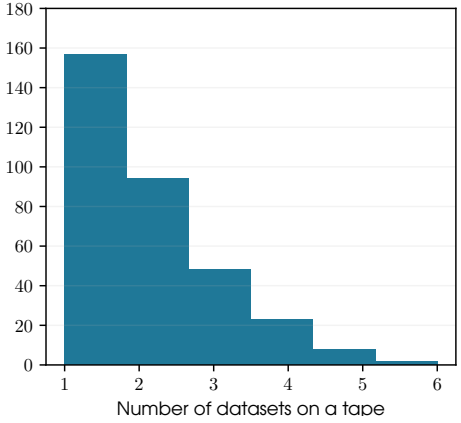




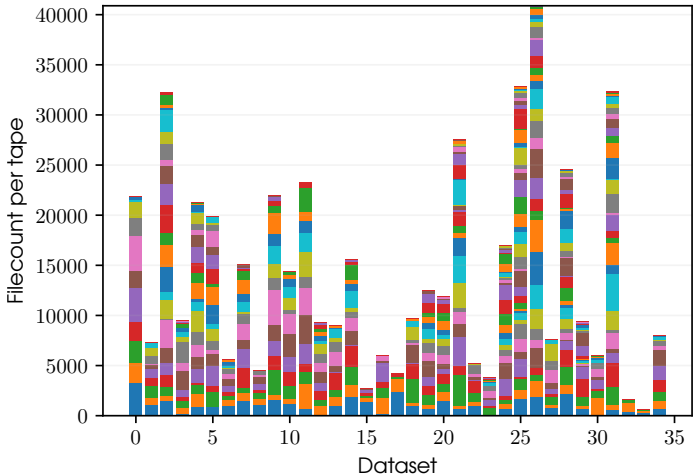
KIT Data – ATLAS DC 2020

- Dataset spread:





KIT Data – ATLAS DC 2020



Summarized Simulation Results

Queue Size	Sim Mode	Avg Overall Throughput MB/s	Avg Time in Queue in min	Mounts (Remounts)	Avg Capacity Recalled per Mount in %	Avg Mount Duration in min
2000	Random	855	88	6585 (6256)	0.3	11
2000	By Dataset	836	78	1221 (1144)	1.8	45
2000	By 2 Tapes	393	178	27 (0)	38.0	611
2000	**By 12 Tapes**	**1104**	**76**	**269 (198)**	**8.5**	153
30000	Random	1090	783	1004 (674)	2.4	40
30000	By Dataset	1779	536	192 (59)	22.0	225
30000	**By 2 Tapes**	**2263**	**509**	**146 (2)**	**33.0**	280
30000	By 12 Tapes	1958	518	168 (33)	26.0	244