# Simulation of Hierarchical Storage Systems for TCO and QoS

Jakob Luettgau and Julian Kunkel

Deutsches Klimarechenzentrum GmbH,
Bundesstraße 45a, D-20146 Hamburg
{luettgau,kunkel}@dkrz.de
http://www.dkrz.de/

**Abstract.** Due to the variety of storage technologies deep storage hierarchies turn out to be the most feasible choice to meet performance and cost requirements when handling vast amounts of data. Long-term archives employed by scientific users are mainly reliant on tape storage, as it remains the most cost-efficient option. Archival systems are often loosely integrated into the HPC storage infrastructure. In expectation of exascale systems and in situ analysis also burst buffers will require integration with the archive. Exploring new strategies and developing open software for tape systems is a hurdle due to the lack of affordable storage silos and availability outside of large organizations and due to increased wariness requirements when dealing with ultra-durable data. Lessening these problems by providing virtual storage silos should enable community-driven innovation and enable site operators to add features where they see fit while being able to verify strategies before deploying on production systems. Different models for the individual components in tape systems are developed. The models are then implemented in a prototype simulation using discrete event simulation. The work shows that the simulations can be used to approximate the behavior of tape systems deployed in the real world and to conduct experiments without requiring a physical tape system.

**Keywords:** Modeling, Simulation, Tape, Long-term archive, Hierarchical Storage Systems, Performance, Total Cost of Ownership

## 1 Introduction

With the increasing demand for long-term storage, automated tape libraries will likely remain an integral part of the storage hierarchy for many years to come. Tape as a storage medium has many attractive properties. It is fairly robust and provides high data densities, but the most important factor is that tape is very affordable in comparison to other storage technologies. Standardization efforts such as LTO make tape attractive and future proof, thus protecting investments. Despite tapes long history, the technology is still competitive[3][4], but incentives to turn technological improvements in capacity and performance

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-67630-2_12

In an effort to speed up innovation and to enable also newcomers and experts and without access to large scale tape systems to contribute, the objectives of this work were to develop a simulator, tools, and primarily appropriate models required to reproduce the dynamics of hierarchical storage systems and tape libraries. Modeling a complete tape system is a complex task, because many different components are involved. It was possible to identify a number of key components that are essential to any tape system. It was further possible to provide comprehensive models to describe the dynamics of many of these key components. In particular, models for hardware and software components were proposed and isolated in such a way that turning to more accurate models is possible.

## 2 Related Work

Efforts to improve tape storage systems often focus on advancing the technology that is used to read and write tape. This is mostly in the domain of vendors and not much of the research conducted is published to protect a business advantage. More openly discussed are strategies for data placement on tape[1][9][8] and the magnetic representation[2]. Such strategies maybe exploited by higher level algorithms, but tape drives and hardware generally do not expose fine-grained control to the users. Another form of placement which was researched but has not yet found its way into many production system is RAIT [4] or TapeRAID and combinations of RAID and tape[6]. Pure tape systems cease in relevance and hybrid and hierarchical storage systems promise to provide cost-efficient solutions with the best properties of multiple technologies. Dee et al.[2] stress the opportunities of automation, which enabled scalable solutions that seamlessly integrate into existing the storage hierarchy. Koltsidas et al.[5] focus especially on the integration of disk and tape. Zhang et al.[10] explore different object placement strategies within tape libraries to optimize tape switch, data seek and transfer times using a simulation. More recently, Mäsker et al.[7] use workload traces of the European Centre for Medium-Range Weather Forecasts (ECMWF) to simulate tape libraries, albeit not integrated into the hierarchical storage system of the data center.

## 3 Simulation Overview

To simulate tape libraries within hierarchical storage systems a huge variety of subsystems and software components can be modeled and implemented for simulation. Figure 1 provides an overview of some components that are of interest because they offer opportunities for optimization. In particular the simulator is designed to consider the following hardware components:

- Multiple clients or groups of clients that act together
- I/O Servers and server local disk/flash caches
- A global online based cache as is used with, e.g, HPSS

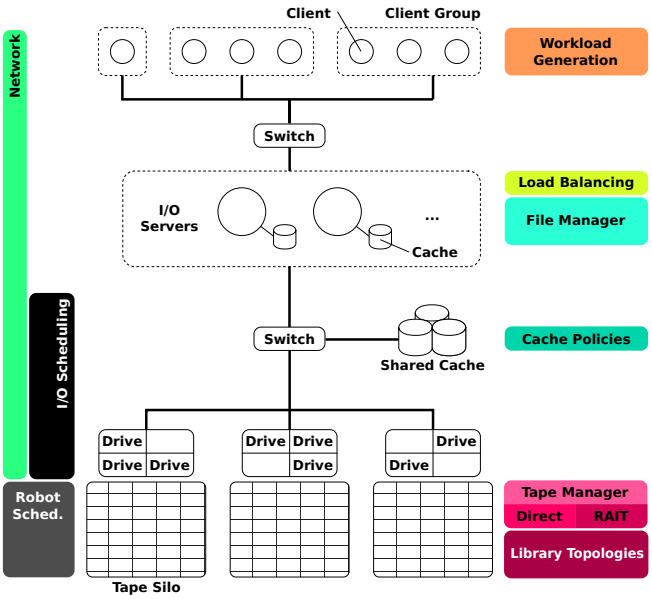



Fig. 1: An overview of different subsystems and software components that are relevant when simulating hierarchical storage systems and tape libraries.

- Multiple libraries and the position of tapes and robots within the library
- Multiple drives and tape generations (e.g., LTO)

More aspects about the network and library modeling are covered in Section 4 From the software side (see Section 5) the simulation has to implement the following components to drive the hardware:

- Various book-keeping and resource management components to keep track of files, tapes, free library slots and robots.
- Different I/O and network scheduling algorithms to grant resource allocation
- Cache displacement strategies for files in I/O node local or global caches
- (Potentially) load balancing mechanisms on different levels

Not all of these systems are implemented at this time. For example RAIT and easily exchangeable load balancing and caching policies are not supported. Besides functional components the simulator has to provide facilities to collect data that is relevant to compare different configurations of virtual systems. Different helpers to sanitize workload traces as well as R scripts to generate plots from the virtual monitoring data are provided.

## 4 Hardware Models

The problem with computer systems is the complexity that unfolds because of the large number of possible combinations for hardware and software. Modeling

hardware is particular cumbersome because in the real world the system performance emerges as a result of the laws of physics, but for a virtual model the dynamics have to be understood and abstracted. For standardized components it is often relatively easy to find a model that is adequately applicable for the whole class of components. Composite components, such as the library topologies turn out to be harder to generalize in a simple way than expected. By mixing mostly 2D and graph-based topology approaches, good approximations of the library dynamics could be achieved. Another problem occurs with proprietary designs for which detailed information is hard to find. The same is true for benchmarks and a comprehensive catalog of performance parameters. The network is an integral part of hierarchical storage systems and can be used to model and simulate even low-level components (e.g., chip level) and communication.

### 4.1 Network Topology & Data Transfers

The network topology is represented using directed graphs. The approach is straight forward so that network devices such as compute nodes, I/O nodes and switches are represented by the vertices and edges are used for the individual links used to connect them. Each link may specify a latency and bandwidth. The network topology then is keeping track of available capacities. As data is moved between components, it is possible to allocate and release network allocations. This schema was used to reduce the number of events in comparison to a packet based network simulation. But the approach does not scale for large network topologies where determining the max-flow can become prohibitively expensive.

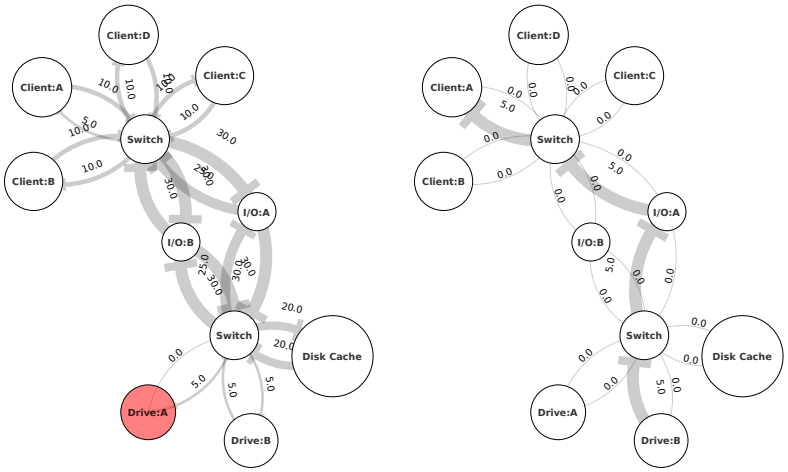



Fig. 2: Available capacities (left) and a flow from `Drive:B` to `Client:A` (right).

### 4.2 Library Topology

To model the library hardware we start with a coarse grained graph-based topology that connects individual components and combine it with detailed models where the graph-based model appears insufficient.

**Graph-based Topology Model:** A coarse grained structure (e.g., the way multiple library units are connected to form library complexes including Pass-Through-Ports and elevators) is modeled using a graph that describe the paths a tape/a robot can travel. A vertex in the graph consequently is used for components and edges can be used to store distance or travel times from component to component. For each vertex or edge also callbacks can be registered, to allow individual components to use sophisticated models underneath (e.g., to account for their current state). In principle, the approach is very flexible and fairly accurate depending on the level of detail. For highly detailed models, the approach can be tedious to configure and will be more expensive to compute. When choosing low levels of detail, errors may accumulate rapidly. Figure 3 illustrates the concept, and in this case mixes distances and times; this is just one of many ways to interpret edges and nodes in a graph based topology. By using graphs modeling becomes intuitive and e.g., the task of serving a tape that sits in `Shelf-2` to `Drive-1` becomes the problem of finding the shortest path between the two. As we want to calculate the time penalty for the next event, for two vertexes $v_i$ and $v_j$ and edges $e_{v_i,v_j}$ the time $T_G$ to get from $v_i$ to $v_j$ calculates in principle as follows. $v_{robot}$ is used to denote the maximum robot velocity:

$$\text{get\_time}(e_{v_i,v_j} \text{ or } v) := \begin{cases} t & \text{if } e_{v_i,v_j} \text{ or } v \text{ have time } t \text{ set} \\ \frac{\text{get\_distance}(v_i,v_j)}{v_{robot}} & \text{if } e \text{ but no time is set} \\ 0 & \text{otherwise} \end{cases}$$

$$T_G(v_a, v_b) = \sum_{v \,\in\, \text{shortest\_path}(v_a,v_b)} \text{get\_time}(v) + \text{get\_time}(e_{v,v+1})$$

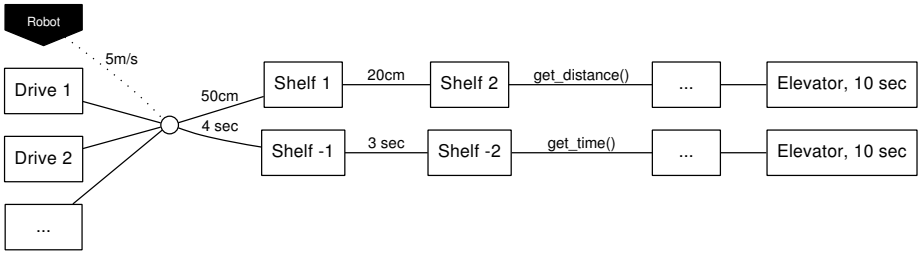



Fig. 3: The graph based topology for coarse grained relationships including library complexes, Pass-Through-Ports and Elevators connecting multiple rails.

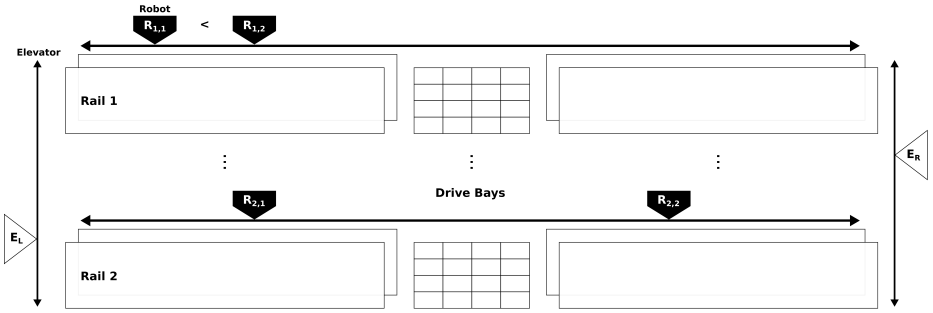



Fig. 4: The StorageTek SL8500 library in a two-dimensional model.

**2D Topology Model** Sometimes projecting complete robot libraries into a two-dimensional representation yields very good approximations. For the SL8500, this seems to be an efficient approach (see Figure 4). The reasoning is that many significant movements that can be performed by the robots or the library are at most two-dimensional anyways. Finding a path within a 2D model then becomes calculating the *Euclidean-distance* between a number of points and a check if the robot is crossing a forbidden area or an obstacle, in which case additional measures have to be taken. Movements are usually decomposed of multiple linear movements, thus care must be taken when calculating distances and travel times. Logical components are resolved to coordinates by providing a mapping function for, e.g., slots and drives. Mounting a tape placed in `Slot-6,9` to `Drive 2` requires visiting multiple coordinates. May $T_{2D}(path)$ be the time it takes to traverse a $path \in \{(p_1, ..., p_n) \mid p_i \in (x, y); x, y \in \mathbb{R}\}$. Assuming different robot velocities $v_x$ and $v_y$ for each axis, the total travel time may be defined by the sum of the time traversing between two points $T_{2D}(p_i, p_j)$ and possibly occurring work and wait times $T_{wait/work}$:

$$T_{2D}(p_j, p_i) = max\left(\frac{|p_{ix} - p_{jx}|}{v_x}, \frac{|p_{iy} - p_{jy}|}{v_y}\right)$$

$$T_{2D}(path) = \sum_{p_i, p_j}^{path} T_{2D}(p_i, p_j) + T_{wait/work}$$

An easing function $e(|p_{id} - p_{jd}|, v_{max})$ can to be applied before taking the maximum, should gradual robot acceleration be taken into account. The exact times also depend on other robots, which reinforce the need for a component that guards the behavior of robots as was discussed for graph-based topologies. In general, we assume hybrid approaches that mix graph-based and 2D models to achieve good approximations at reasonable effort.

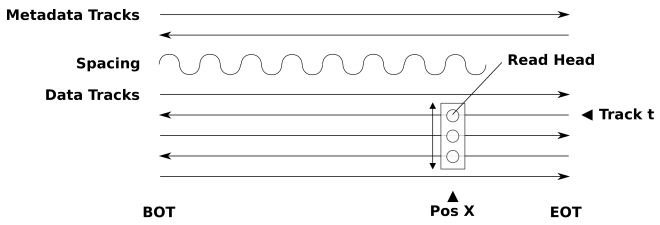



Fig. 5: Serpentine Tape Model

### 4.3 Tape-Seek- and Drive-Busy-Time Models

Commonly there are three layouts for writing data on tape (linear, linear-serpentine and helical-scan). We will consider only linear-serpentine tape as it appears to be the most relevant on modern systems. Figure 5 takes the perspective of the tape drive and illustrates how data is actually read or written on tape. An array of read/write heads can be positioned relative to the tape in two dimensions and imprint or read a magnetic signature as the tape passes underneath. When spooling to a specific position it is possible to move the tape quite fast, when reading or writing lower speeds yield the best results. By determining the following characteristics, it should become possible to approximate the time it takes to serve a request and how long a drive remains busy:

- $pos := (x, t)$: a tuple describing horizontal $x$ and vertical (track) $t$ displacements relative to the tape. $pos_{BOT}$, $pos_{EOT}$ are used to reference the Begin-of-tape and End-of-tape. A single reel cartridge is mounted and unmounted with $pos_{BOT}$.
- $T_{mount}$ and $T_{unmount}$: the time it takes to mount and unmount a tape
- $v_{spool}$, $v_{head}$: the speeds to reposition the tape and read-heads
- $v_{read}$, $v_{write}$: the speed in, e.g., $bytes/second$ to read and write

The time to transition from a current position $pos_i$ to target position $pos_j$ is calculated as follows:

$$T_{seek}(pos_j, pos_i) = max\left(\frac{|pos_{ix} - pos_{jx}|}{v_{spool}}, \frac{|pos_{it} - pos_{jt}|}{v_{head}}\right)$$

The time $T_{read/write}$ to read or write from tape is calculated as follows:

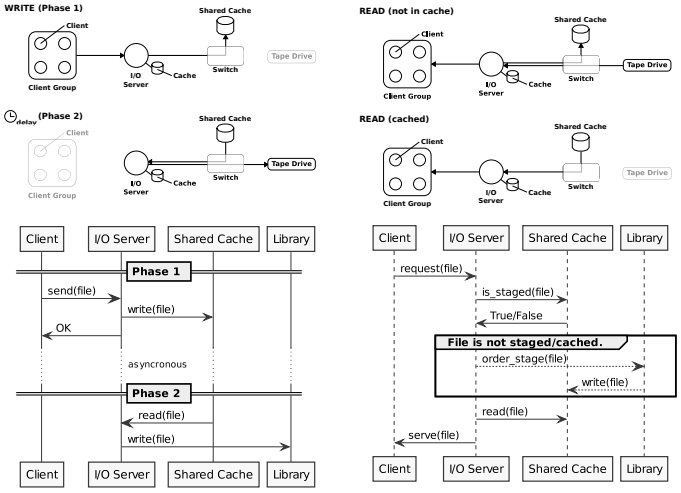
$$T_{read/write}(bytes) = \frac{bytes}{v_{read/write}}$$

The time a tape drive remains busy $T_{busy}$ would account for possibly multiple seek and reading phases, before it ejects the tape and becomes available again.

$$T_{busy} = T_{mount} + \left(\sum_{pos_i, pos_{i+1}}^{BOT,\ldots,BOT} T_{seek}(pos_i, pos_j) + T_{read/write}(bytes_i)\right) + T_{unmount}$$

## 5 Software Models

To stress the virtual tape library, a request object can be instantiated and submitted to the simulation. In addition to explicit submission, it possible to register event providers to the simulation which are polled for future events until they indicate they have been drained. A workload provider also could create requests on the fly according to a script, a probability distribution or a trace file. For a proof of concept it often seemed sufficient to turn to "naive" implementations.




(a) Life-cycle Write (b) Life-cycle Read

Fig. 6: Handling of read and write like requests for the HSM tape system.

**Request Processing** A request is modeled by a process that waits for allocations to use a particular resource. Figure 6 and Figure 7 illustrate the processing of a request. Writes occur in two-phases with the clients only waiting for the first phase. Tape I/O can be performed asynchronously for writes. Reads are handled based on their presence in the global disk cache.

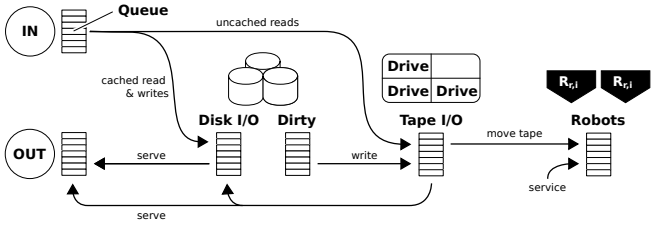



Fig. 7: Different resources and queues required to govern request handling.

# 6 Evaluation

This section outlines how the models are verified (see Section 6.1) and how simulation with alternative configurations can be used to minimize TCO. Two scenarios are discussed to show the potential.

## 6.1 Workload Trace Replay for Verification and Optimization

**Workload Description:** For verification and fine-tuning of the simulation HPSS request logs and monitoring records were used. Not all activities that occur in an actual tape system are included in these traces. As a result service workloads, and temporarily disabled drives are not accounted for. In addition, components in this simulation do not fail for the lack of reliable failure rates. The trace includes a week of scheduled downtime, which is interesting to compare the recovery of the virtual and the actual system. Figure 8 shows the distribution of file sizes and also the ratio of reads/writes that hit the system. Figure 9 shows the distribution of reads and writes over time. The following table provides a total count of requests as well as the number of involved files and clients:

| | | | |
|---|---|---|---|
| `Timeframe:` | `35 days` | `Requests:` | `213105` |
| `Files:` | `115856` | `Writes:` | `85961` |
| `Clients:` | `562` | `Reads:` | `127144` |

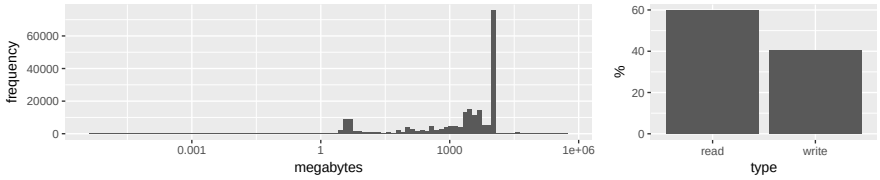



Fig. 8: Frequency of different request sizes. With a large peak for requests with a size of 10 GB, which is a result of the data centers pricing schema.

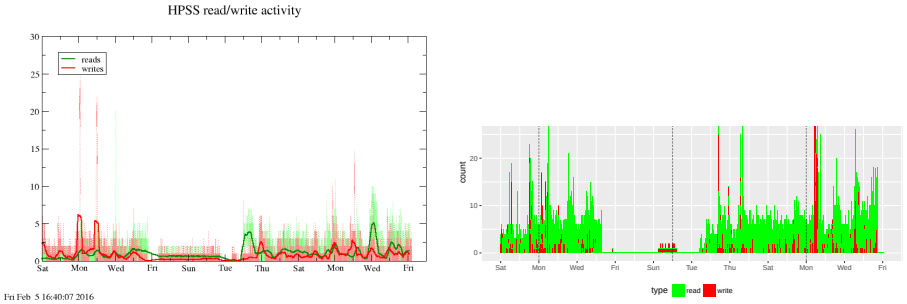
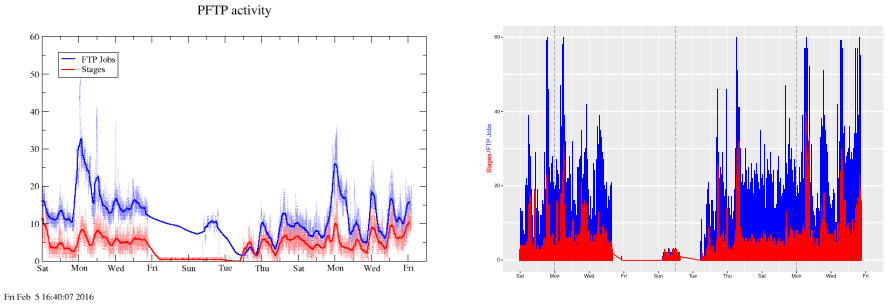



Fig. 9: Observed request types over time for a period of 3 weeks.

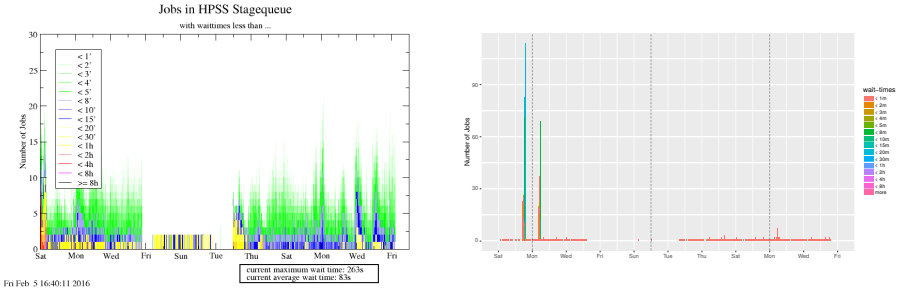
**Staging Behavior and Wait-Times:** To verify that the simulation can reproduce the run-time behavior from a virtual tape system two important factors are: 1) the number of busy drives and 2) the wait-times for requests over time. The number of busy drives is directly related to the number of stages, which is a metric provided by the HPSS monitoring. Figure 10 lists two plots each showing the number of FTP requests and the number of stages occurring as observed by (a) the HPSS monitoring and (b) the simulation. Figure 11 similarly plots the wait-times for queued requests, again (a) HPSS monitoring left and (b) the simulation result on the left. While there are differences in the perceived workloads, the peak times and magnitudes match in principle. The verification at this point also reveals that some components require further fine-tuning to more accurately approximate the original system. Yet, in comparison to conventional methods when procuring systems, the simulation allows taking site specific run-time behavior into account.




(a) FTP jobs and stage counts as reported by the monitoring.

(b) FTP Jobs as observed in the trace file and stages as occurred in the simulation.

Fig. 10: FTP activity for a period of three weeks. Validation of stage-counts by comparing the DKRZ monitoring (a) to the results of the simulation (b).




(a) Wait-time as observed in monitoring.

(b) Wait-time in simulation.

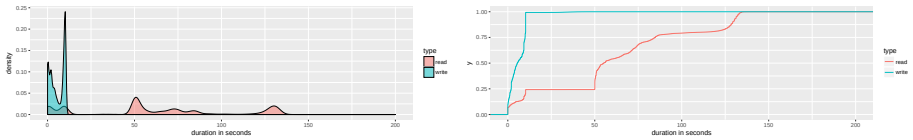
Fig. 11: The number of requests waiting in queue based on their wait-time.

### 6.2 TCO and QoS Optimization under Varying Drive Configuration

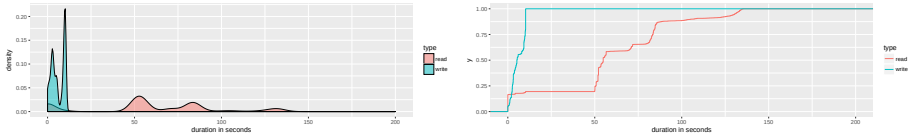
Among the most expensive components also being procured on a regular basis are tape drives. The reason for this is, that every 2-3 years a new LTO generation is released, usually doubling the capacity of tape cartridges in combination with modest improvements on read and write speeds. Depending on the use-case for a long-term archive in a data center different objectives may have priority, but common strategic decisions may relate to the following concerns:

- What is the lowest number of drives to meet a certain quality of service, e.g., serve every request to cold data within 10 minutes.
- What is the optimal placement for drives and tapes when considering library complexes, partitions and multiple service level agreements?
- When and how to switch to a new drive generation?
  - Under an expected change in workloads (e.g., additional users) when will the available resource fail to meet the requirements?
  - Tape media and drives become cheaper over time. What would be a good gradual transition strategy? Is it cheaper?
  - Is it cheaper to run on a library with many slots and use more older drives? How would using RAIT effect this?

Figure 12 shows two configurations (a) fewer drives and (b) more drives. For each configuration the distribution of request latency is plotted. A second graph plots the same data using an empirical distribution functions to allow to easily determine, e.g., quality of service guarantees that can be made for a certain percentage of requests. In many cases these curves stabilize after handling a few thousand requests allowing to spin up only short simulations, and use similar or more advanced criteria for an optimization problem.




(a) A example configuration with fewer drives.




(b) A example configuration with more drives.

Fig. 12: Example comparison of request latency for different runs.

**Power consumption:** Besides performance characteristics we can collect load statistics for individual components. One use for these statistics is the estimation of power consumption. Figure 13 shows a plot of the drive utilization over time. It is reasonable to assume that a busy drive consumes more energy than an idle drive. Thus, again only considering changes to the drive configuration, even without concrete power consumption figures for a particular drive, it is possible to compare systems. For a more accurate estimate of power consumption, we can measure the power consumption for idle and busy phases of a tape drive and then aggregate busy and idle times to derive the estimate for a given configuration. This also allows to evaluate the switch to different power tariffs, e.g. a day/night tariff.

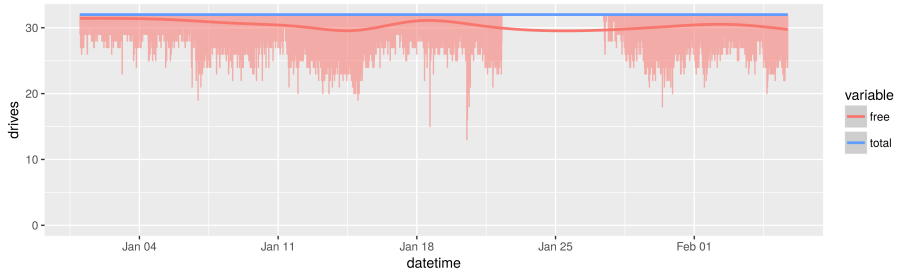



Fig. 13: Drive utilization as well as the total number of available drives over time. The trace in this case did not provide any data about enabled/disabled drives.

## 7 Summary

Equipped with comprehensive models and a simulator to approximate tape archives within hierarchical storage systems, it is possible to improve modern tape libraries without requiring a physical tape library for testing. Workflows to experiment and asses the performance directly influenced the design of the simulator, consequently the next step is to put it to use in experiments. Also, gradually turning the simulator into an open source tape library management solution for production systems could be an option. From a research perspective the interesting part of a simulation is to apply it to practical problems to learn and generate new insight. The next step is to carefully construct experiments with the current system and iteratively improve the tools required to conduct more experiments. In particular, this might include parameterized Monte-Carlo methods to optimize for budgets or quality of service. The foundation to perform these kinds of experiments is provided with this work. In addition, it would be useful to have a comprehensive database for benchmarks that collect the characteristics of tape drives, libraries and other devices. The database should also include component prices, though utilizing online price comparison APIs to fetch prices on demand may also be an option when provided with ways to apply a correction factor for discounts.

## Acknowledgments

This work is part of the ESiWACE project which received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675191.

## References


1. Dashti, A., Shahabi, C.: Data placement techniques for serpentine tapes. Proceedings of the 33rd Hawaii International Conference on System Sciences pp. 1–10 (2000), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=927005
2. Dee, R.H.: Magnetic tape for data storage: An enduring technology. Proceedings of the IEEE 96(11), 1775–1785 (2008)
3. Fontana, R.E., Decad, G.M., Hetzler, S.R.: The impact of areal density and millions of square inches (MSI) of produced memory on petabyte shipments of TAPE, NAND flash, and HDD storage class memories. In: 2013 IEEE 29th Symposium on Mass Storage Systems and Technologies (MSST). pp. 1–8. IEEE (2013), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6558421
4. Hughes, J., Fisher, D., Dehart, K., Wilbanks, B., Alt, J.: HPSS RAIT Architecture. White paper of the HPSS collaboration, www. hpss-collaboration. org/documents/HPSS RAIT Architecture. pdf (2009), http://www.hpss-collaboration.org/documents/HPSS_RAIT_Architecture.pdf
5. Koltsidas, I., Sarafijanovic, S., Petermann, M., Haustein, N., Seipp, H.: Seamlessly Integrating Disk and Tape in a Multi-tiered Distributed File System pp. 1328–1339 (2015)
6. Lingfang Zeng, D.F.: Hybrid RAID-Tape-Library Storage System for Backup. Second International Conference on Embedded Software and Systems (ICESS'05) pp. 31–36 (2005), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1609854
7. Mäsker, M., Nagel, L., Süß, T., Brinkmann, A., Sorth, L.: Simulation and Performance Analysis of the ECMWF Tape Library System. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 22:1–22:12. SC '16, IEEE Press, Piscataway, NJ, USA (2016), http://dl.acm.org/citation.cfm?id=3014904.3014934
8. Pantazi, A., Furrer, S., Rothuizen, H.E., Cherubini, G., Jelitto, J., Lantz, M.A.: Nanoscale track-following for tape storage pp. 2837–2843 (2015)
9. Pease, D., Amir, A., Villa Real, L., Biskeborn, B., Richmond, M., Abek, A.: The linear tape file system. 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies, MSST2010 4 (2010)
10. Zhang, X., He, D., Du, D., Lu, Y.: Object Placement in Parallel Tape Storage Systems. Proceedings of the 2006 International Conference on Parallel Processing (ICPP'06) pp. 0–7 (2006), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1690610