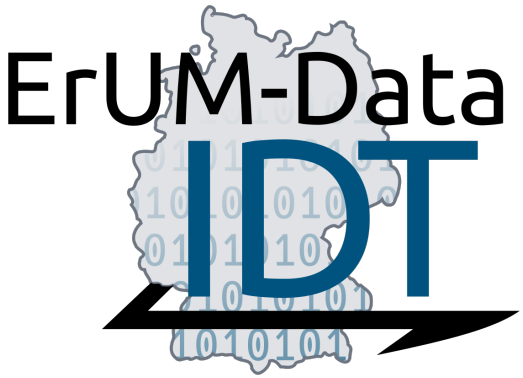
Efficient use, tests and benchmarks of a dynamic caching solution based on XRootD

Michael Böhler, Anton J. Gamel, Dirk Sammel, Markus Schumacher

14.02.2022
IDT-ErUM Collaboration Meeting

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG



Bundesministerium
für Bildung
und Forschung

Overview

Themenbereich A: Entwicklungsarbeiten zur Bereitstellung von Technologien zur Nutzung heterogener Computing-Ressourcen

- Arbeitspaket 1: Werkzeuge zur Einbindung von heterogenen Ressourcen in das wissenschaftliche Rechnen
- Arbeitspaket 2: Effiziente Nutzung von heterogenen Ressourcen
- Arbeitspaket 3: Identifikation und Steuerung von Workflows auf heterogenen Ressourcen

Themenbereich B: Anwendung und Test von virtualisierten Softwarekomponenten im Umfeld heterogener Computing-Ressourcen

- Arbeitspaket 2: Job- und Ressourcenmanagement
- Arbeitspaket 4: Kombinierte Tests

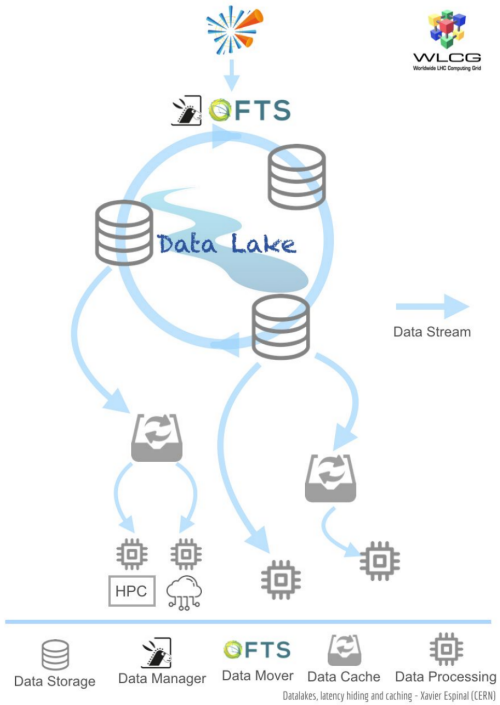
Themenbereich A: Entwicklungsarbeiten zur Bereitstellung von Technologien zur Nutzung heterogener Computing-Ressourcen

- Arbeitspaket 1: Werkzeuge zur Einbindung von heterogenen Ressourcen in das wissenschaftliche Rechnen
- Arbeitspaket 2: Effiziente Nutzung von heterogenen Ressourcen
- Arbeitspaket 3: Identifikation und Steuerung von Workflows auf heterogenen Ressourcen

- Implementation of a data caching solution for the ATLAS computing environment in Freiburg

Introduction

- ATLAS vision:
  - Centralize data storage
  - Huge storage centers with long-term storage
  - Compute sites without local long-term storage
- How do compute sites get the data?
  → implement a caching setup



- What is caching?
  → local, temporary storage of data
- Why caching?
  - Local: fast access for subsequent requests
  - Temporary: limited resources for long-term storage, automatic deletion if not accessed again

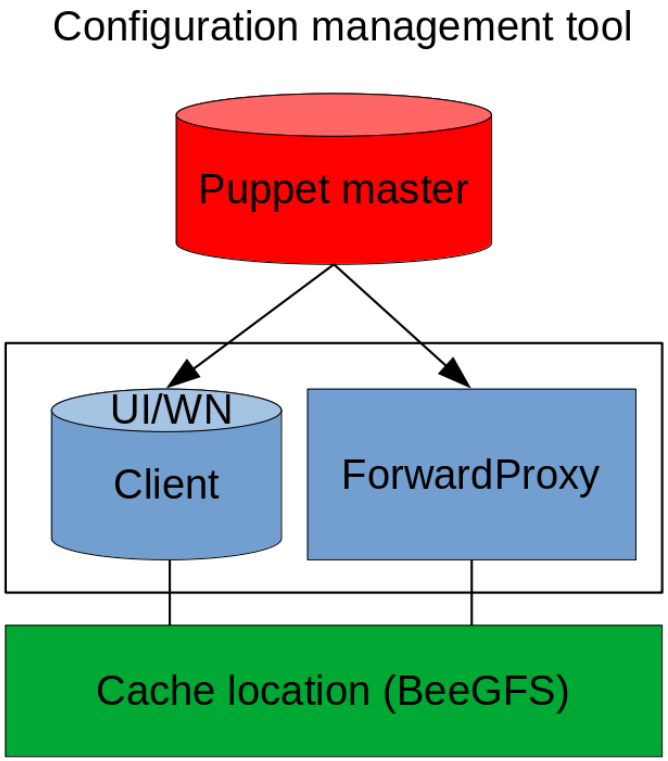
Caching in Freiburg

- Caching setup in Freiburg with XRootD
- XRootD: software framework for data access
- Starting point: sandbox implementation by group of Dr. Kilian Schwarz at Uni. Frankfurt/GSI

- Ingredients:
  - Client
    - Virtual machine on node of NEMO HPC cluster
    - Default client configuration
    - Non-evasive setup of caching via environment variables
    - Enable and configure XrdClProxyPlugin
  - Proxy server
    - Virtual machine independent from NEMO cluster
    - XRootD server daemon configured for disk caching and as "forward proxy"
  - Cache space
    - BeeGFS file system in Freiburg

Freiburg setup at local HPC cluster NEMO

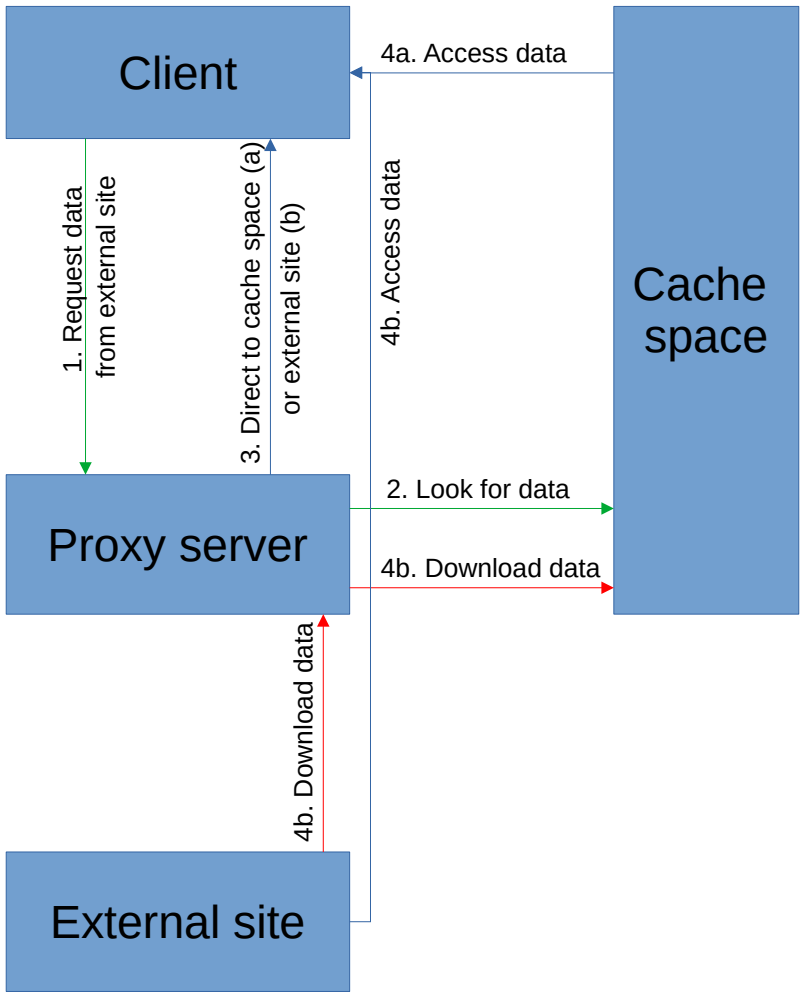
- Deployment with Puppet
- Deploys setup to 2 VMs: Client, Proxy server
- BeeGFS mounted on VMs



- Puppet: machines can be configured at any place
  $\rightarrow$ easy to deploy when ready for production

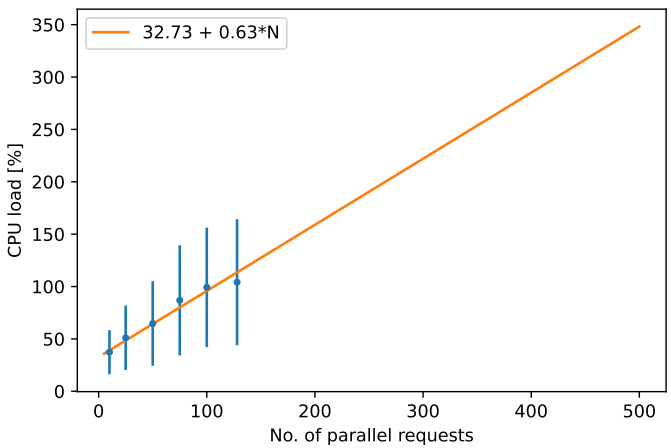
Workflow of caching setup

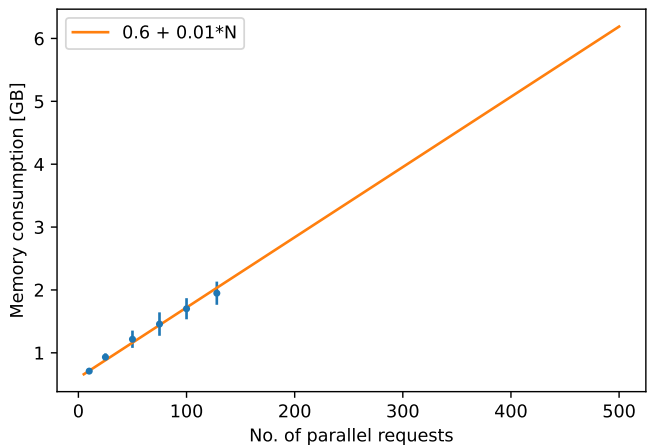
- Client:
  - Requests data from external site
  - XrdClProxyPlugin: redirects request to proxy server
- Proxy server:
  - Checks if data is already in cache
  - Yes:
    Forwards client to location in cache
  - No:
    Forwards client to external site and downloads data to cache



Proxy server stress test

- What is the resource consumption on the proxy server?
  - VM on machine with Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.60 GHz
  - 4 vcores
  - 8 GB RAM
- Real world scenario: ∼500-1000 jobs at once
- Tested up to 128 parallel requests and extrapolated demand





- Projected requirement for 1k parallel requests:
  - At least ∼7 cores
  - At least ∼11 GB RAM
- Better: scalable proxy cluster with multiple servers

- First tests show: proxy server works well on VM

Themenbereich B: Anwendung und Test von virtualisierten Softwarekomponenten im Umfeld heterogener Computing-Ressourcen

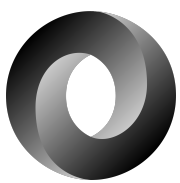
- Arbeitspaket 2: Job- und Ressourcenmanagement
- Arbeitspaket 4: Kombinierte Tests

- Benchmarking of caching setups and visualization of the results

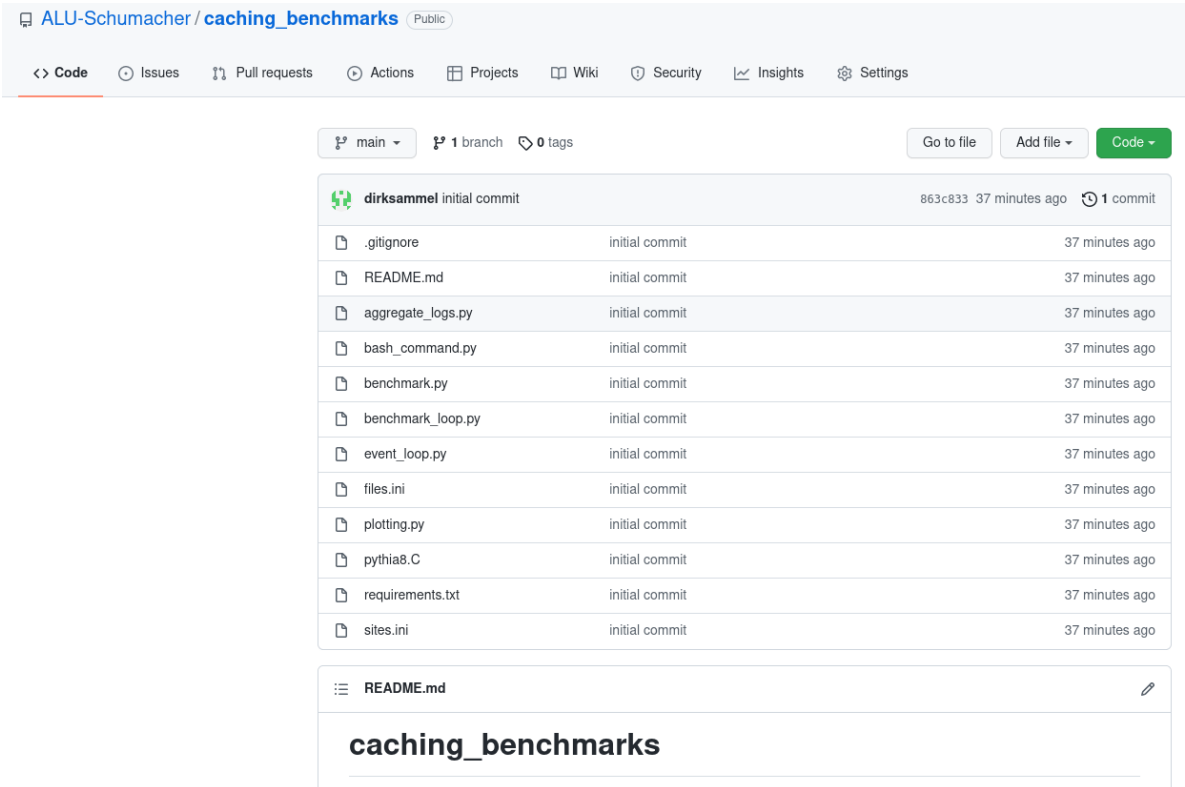
Benchmarking & visualization

- Benchmarking
  - Python script with ROOT module
  - ROOT: data analysis framework
  - 'Pseudo analysis', implements features of typical user analysis:
    - Open file
    - Retrieve data
    - Loop over data entries
    - Fill information in histograms
  - Setup options: file size, site, number of data entries
  - Output: time for completion and information about setup in JSON format
- Visualization
  - Aggregation of JSON files
  - Dataframe creation with pandas
  - Visualization with Matplotlib & seaborn






Benchmarking repository



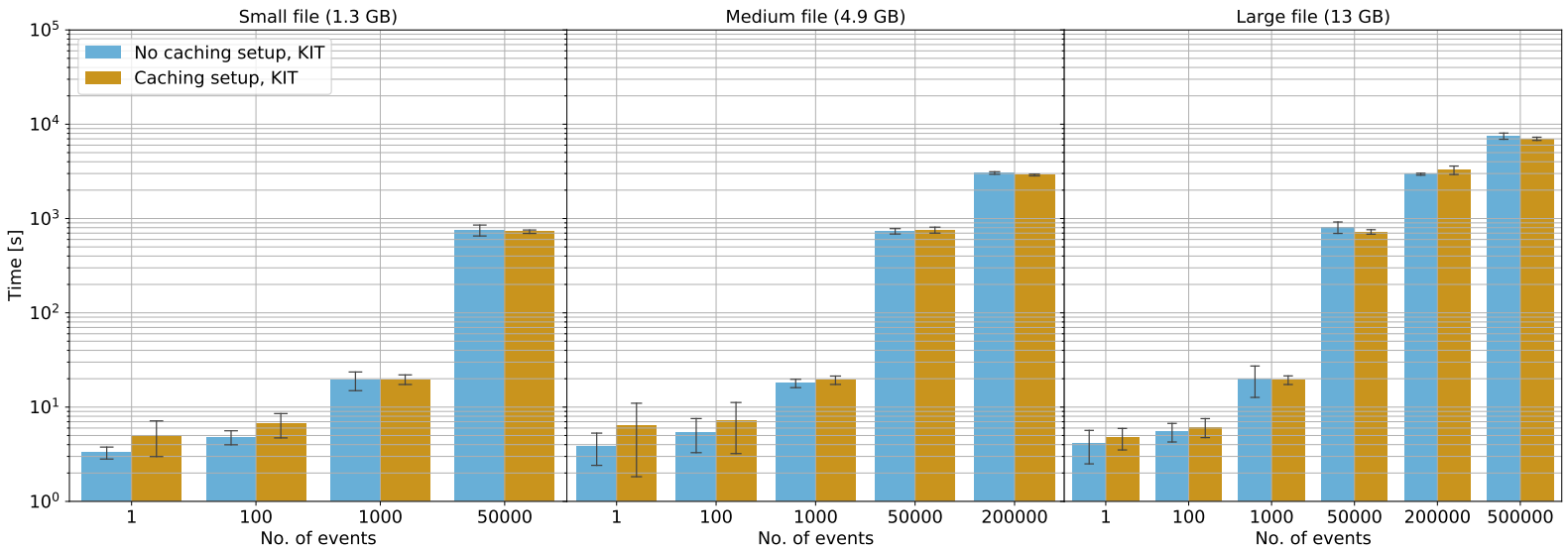
- Benchmarking- and visualization scripts are publicly available on GitHub: https://github.com/ALU-Schumacher/caching_benchmarks
- Contains detailed documentation and instructions

Example: benchmarks of the Freiburg caching setup

- Access configurations:
  - Caching without file in cache space
    → caching setup, file read and downloaded from external site
  - Caching with file in cache space
    → caching setup, file read from local BeeGFS
  - Direct external access
    → caching setup not active, file read from external site
- ATLAS sites
  - KIT (Karlsruhe)
  - TRIUMF (Vancouver, Canada)
  - DESY (Hamburg)
- File sizes and number of data entries
  - 1.3 GB, 4.9 GB, 13 GB (s, m, l)
  - 1, 100, 1000, 50k,
    200k (only m & l), 500k (only l)
- Repeat each test multiple times
  → mean value of elapsed time + standard deviation

Results - Overhead

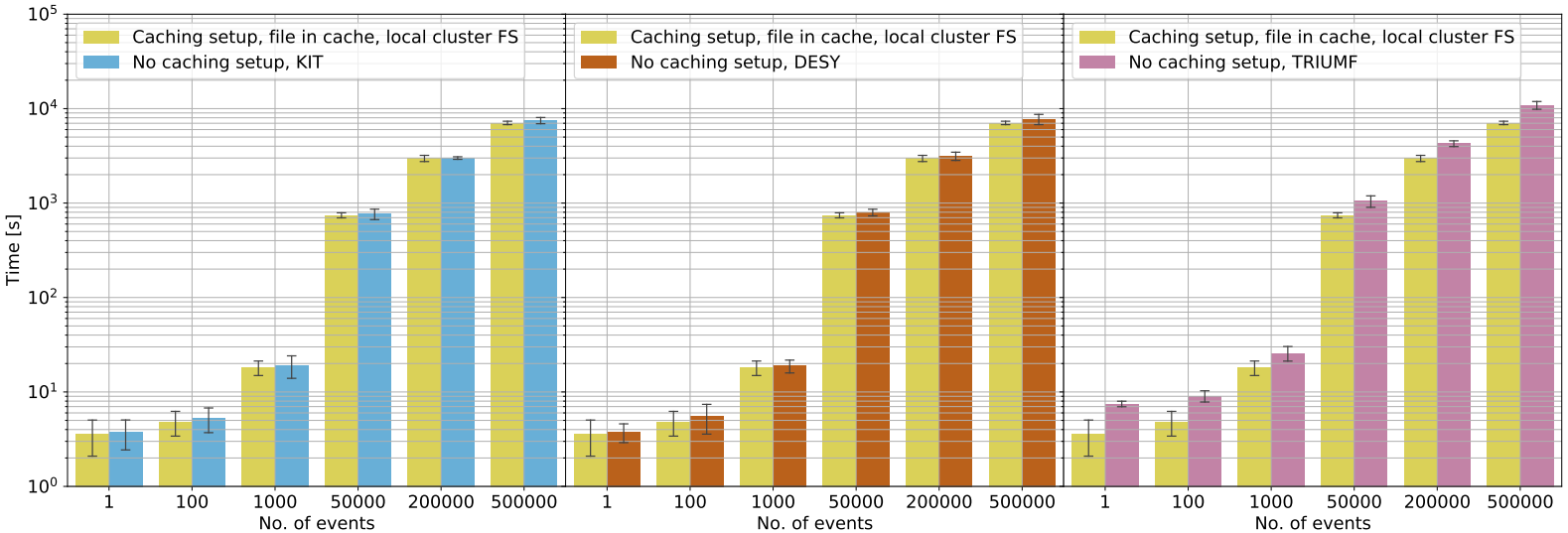
- Any overhead in operation caused by caching setup?
- Direct access to KIT vs. access to KIT with caching setup



- Both setups are comparable
  $\rightarrow$ no overhead by caching setup
- Results differ for number of data entries, but not for file sizes
  $\rightarrow$ merge file sizes in remaining figures

Results - File in cache space

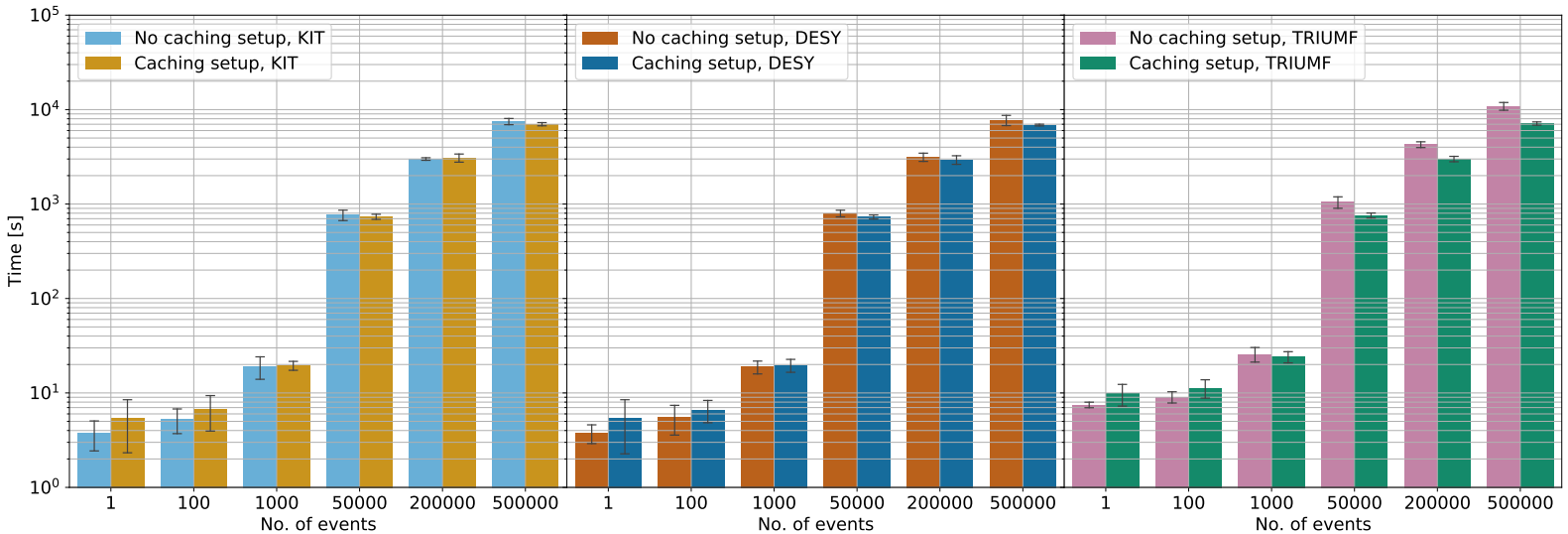
- Direct access to KIT/DESY/TRIUMF vs. file available in cache space



- Near sites (KIT/DESY): results are comparable
- Distant sites (TRIUMF): access to cached file faster
  $\rightarrow$ client profits from caching setup and already cached file

Results - Initial access

- Direct access to KIT/DESY/TRIUMF vs. access with caching setup
- File not in cache space



- Near sites (KIT/DESY): results are comparable
- Distant sites (TRIUMF):
  access with caching setup is faster for large numbers of data entries
  - Download of file via proxy server faster than benchmark runtime
  - After download: client reads file in cache $\rightarrow$ faster access
- Without caching setup: file access completely from external site

Conclusion

- Successfull implementation of caching setup for NEMO HPC cluster
  - Clients: no additional installations, setup via environment variables
  - Proxy server: setup on VM, independent from NEMO cluster
- Benchmark results
  - Caching setup causes no overhead
  - Access to already cached files is faster than access to distant external sites
  - Initial access to files on distant external sites is faster with caching setup
- Caching is a good alternative to locally managed storage!
- Benchmarking- and visualization code is published on GitHub: https://github.com/ALU-Schumacher/caching_benchmarks
- Presentations about caching setup and benchmarks (see backup)
  - Forthcoming publication of bwHPC Symposium contribution

Conclusion

- Successfull implementation of caching setup for NEMO HPC cluster
  - Clients: no additional installations, setup via environment variables
  - Proxy server: setup on VM, independent from NEMO cluster
- Benchmark results
  - Caching setup causes no overhead
  - Access to already cached files is faster than access to distant external sites
  - Initial access to files on distant external sites is faster with caching setup
- Caching is a good alternative to locally managed storage!
- Benchmarking- and visualization code is published on GitHub:
  https://github.com/ALU-Schumacher/caching_benchmarks
- Presentations about caching setup and benchmarks (see backup)
  - Forthcoming publication of bwHPC Symposium contribution

Thanks!

Presentations

- **How to bring HTC to HPC resources - A caching solution for the ATLAS Computing environment in Freiburg**
  14th Annual Meeting of the Helmholtz Alliance "Physics at the Terascale"(virtuell), November 2021
  https://indico.desy.de/event/31325/contributions/112495/
- **How to bring HTC to HPC resources - A caching solution for the ATLAS Computing environment in Freiburg**
  7th bwHPC Symposium (virtuell), November 2021
  https://indico.scc.kit.edu/event/2399/contributions/9208/
- **Results of Disk-Caching-On-the-Fly Benchmarks in Freiburg**
  IDT-ErUM Collaboration Meeting (virtuell), Mai 2021
  https://indico.physik.uni-muenchen.de/event/117/contributions/538/
- **Implementation and benchmarking of a caching solution in the ATLAS Freiburg environment**
  DPG-Frühjahrstagung 2021, Dortmund (virtuell), März 2021