Status report

Maysam Rahmanpour, Reza Salkhordeh, Frank Mass, André Brinkmann

Aachen– 30th September 2024

IDIUM

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

JG|U

OVERVIEW

Current Work

- Integration of Lustre HSM and Gekkofs
- Providing caching by using ad-hoc file systems
- Compatibility of HPC applications and ad-hoc file systems
- Analysis of BESIII & LOFAR workloads on MOGON II and MOGON-NHR

HPC APPLICATION ANALYSIS

- HPC I/O Analysis
- Released at : https://hpcioanalysis.zdv.uni-mainz.de/
- Capturing I/O behavior with DARSHAN
- Analyzing I/O behavior
- Ongoing applications:

BESIII

LOFAR

AD-HOC FILE SYSTEMS

- Employing unused node-local storages like SSDs and NVMs
- Distributing data and metadata across node-local storage
- Parallel read/write on multiple nodes
- Hiding slow performance of backend storage
- Reducing I/O bottlenecks

AD-HOC FILESYSTEMS LIMITATIONS

- User responsibilities with ad-hoc file systems
- Transferring data to node-local storage
- Separated namespace with back-end storage
- Risk of data inconsistency
- Node-local storage integration into the storage hierarchy is still a problem
- Requiring restart of the job when compute node is restarting

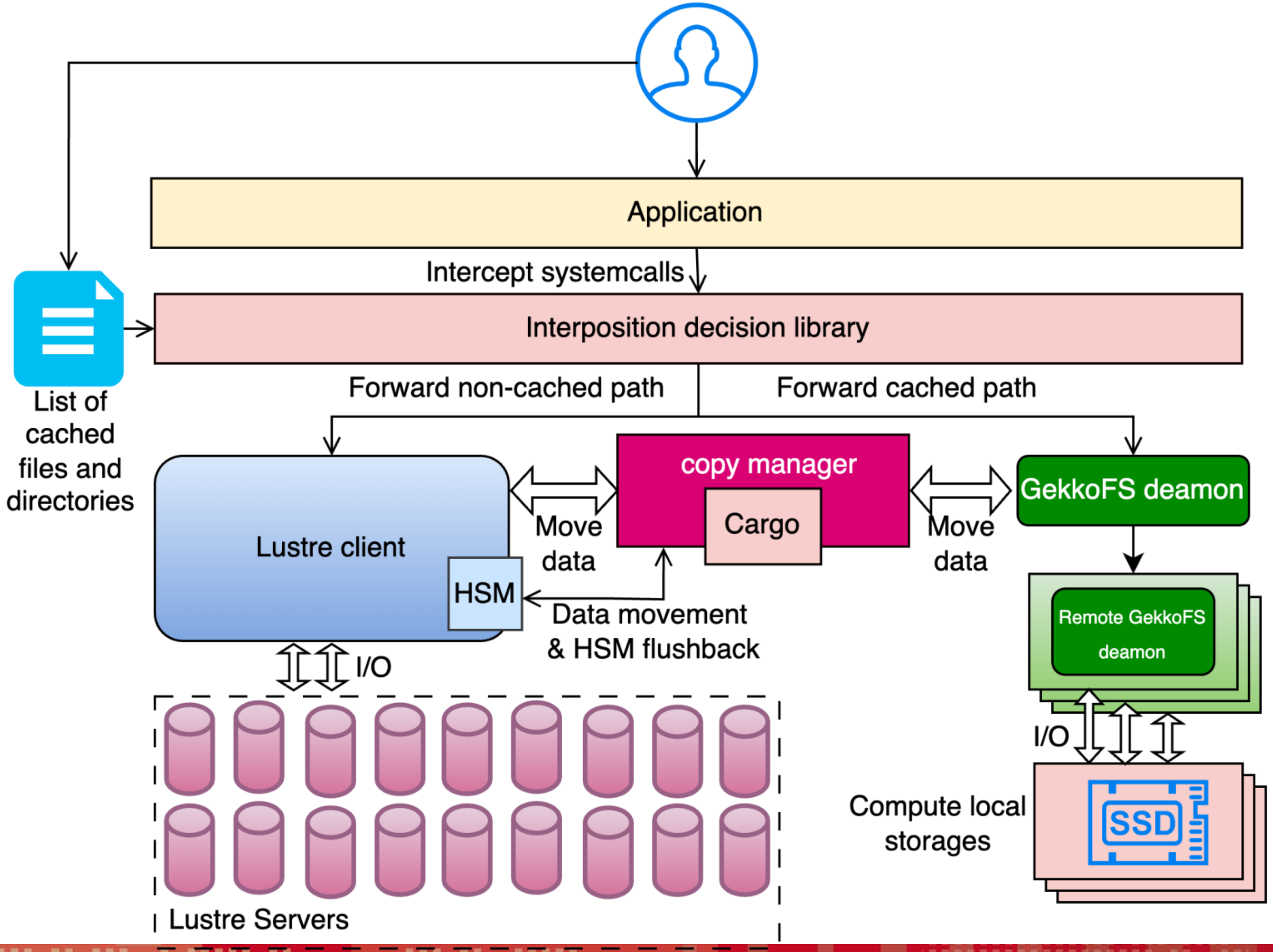
EXISTING SOLUTION - LPCC

- Offers client cache to user
- Employs HSM from Lustre to guarantee consistency
- RW mode
  - Transfers files which accessed exclusively by one node
- Read mode
  - Transfers a version of read-only data to each node
- No single namespace

LUSTRE HSM INTEGRATION WITH GEKKOFS

- Providing a single namespace
- Guarantee data consistency
- Providing a cache for HPC application with node local storage
- Integration of node local storage into the storage hierarchy
- Minimize user interaction with transparent design
- Automatic data transfer

INTEGRATION LUSTRE HSM AND GEKKOFS

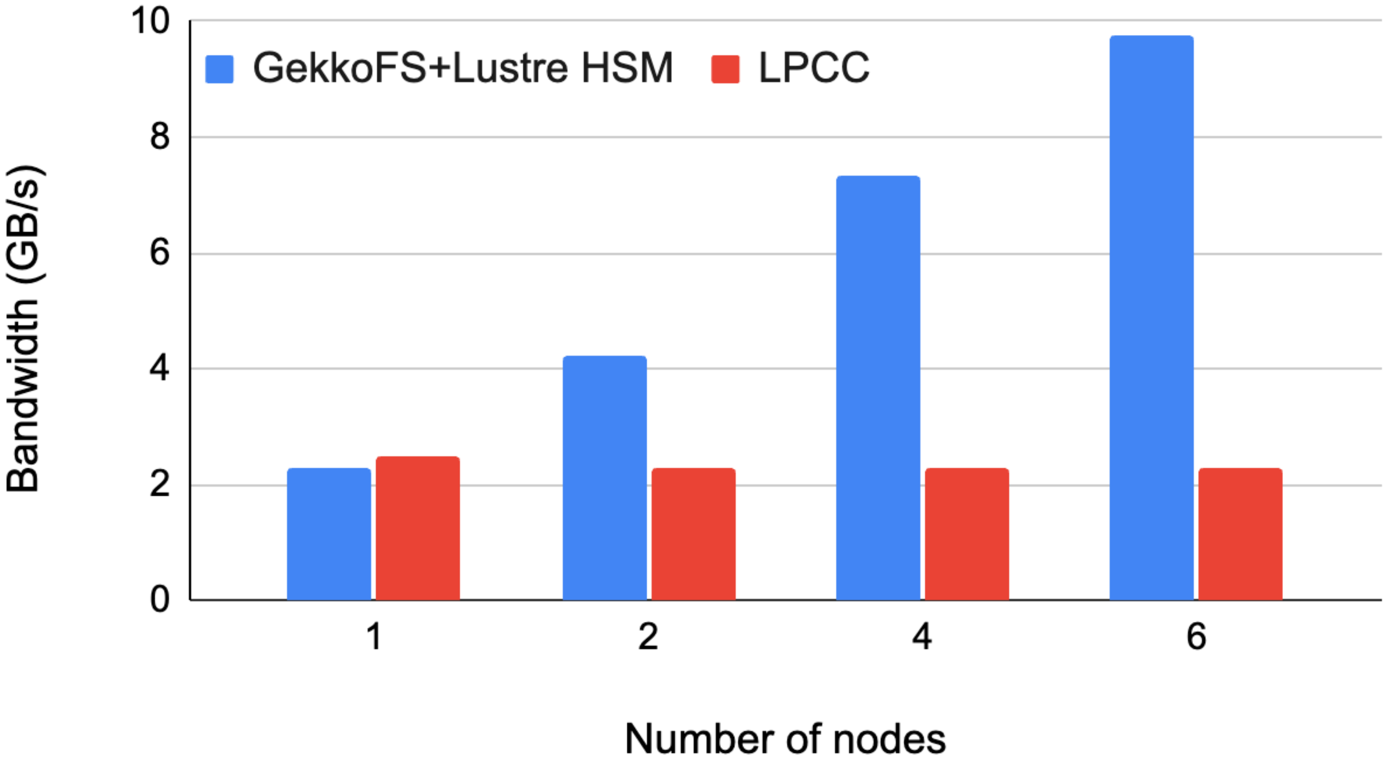


DESIGN FEATURES

- User can define files and directories path for caching
- Using interception library to dispatch requests between Lustre and Gekkofs
- Using Cargo as a copy tool
- Using Lustre HSM to offer single namespace
- HSM is storing the status of a file (existing on Lustre, moved to Gekkofs)
- HSM triggers a file flush back from Gekkofs to Lustre in the case of conflicting access
- Read-only mode

READ PERFORMANCE – HSM INTEGRATED

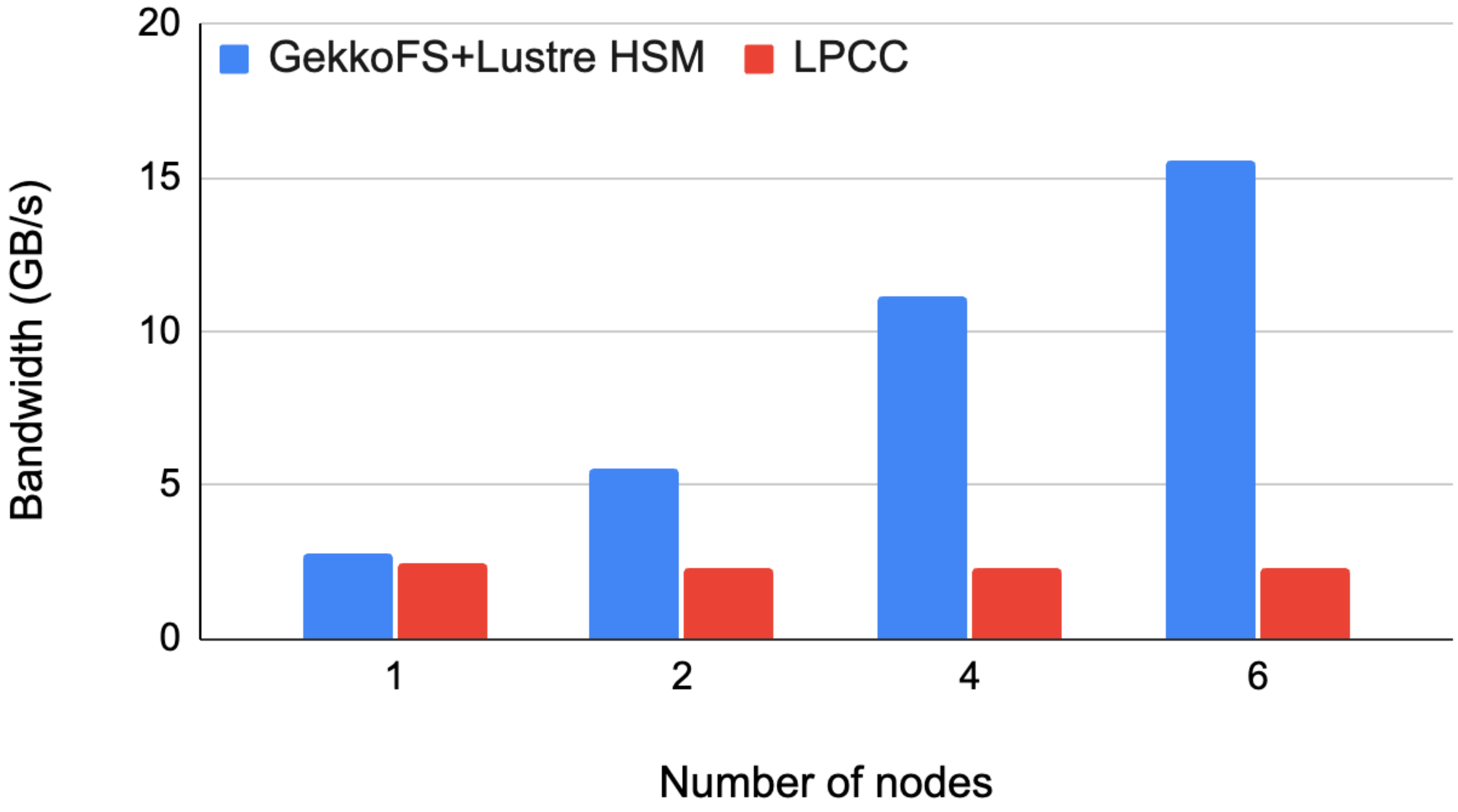
- IOR test
- Read each file 10 times
- One file per IOR
- IOR transfersize = 1 MB
- 32 processes per each node
- 4 GB per process
- Overhead: an extra read from Lustre and an extra write on NVM
- LPCC can not cache shared data between nodes



Lustre min bandwidth = 1071MiB/s
Lustre max bandwidth =3059 MiB/s
Lustre average bandwidth = 2620 MiB/s

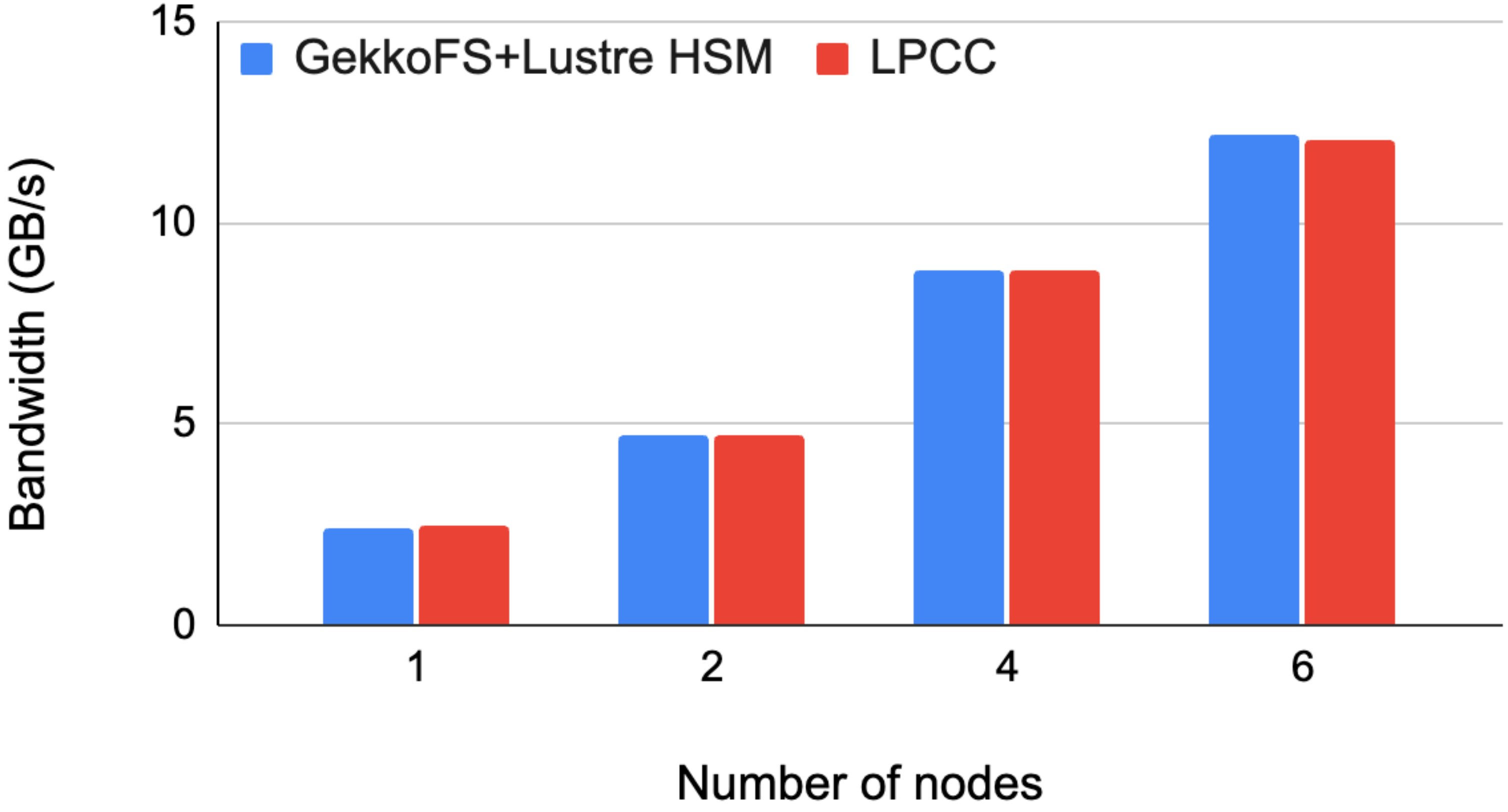
WRITE PERFORMANCE- HSM INTEGRATED

- IOR test
- Read file 10 times
- One file per IOR
- IOR transfersize = 1 MB
- 32 processes per each node
- 4 GB per process
- Creates file directly on NVM
- No additional overhead



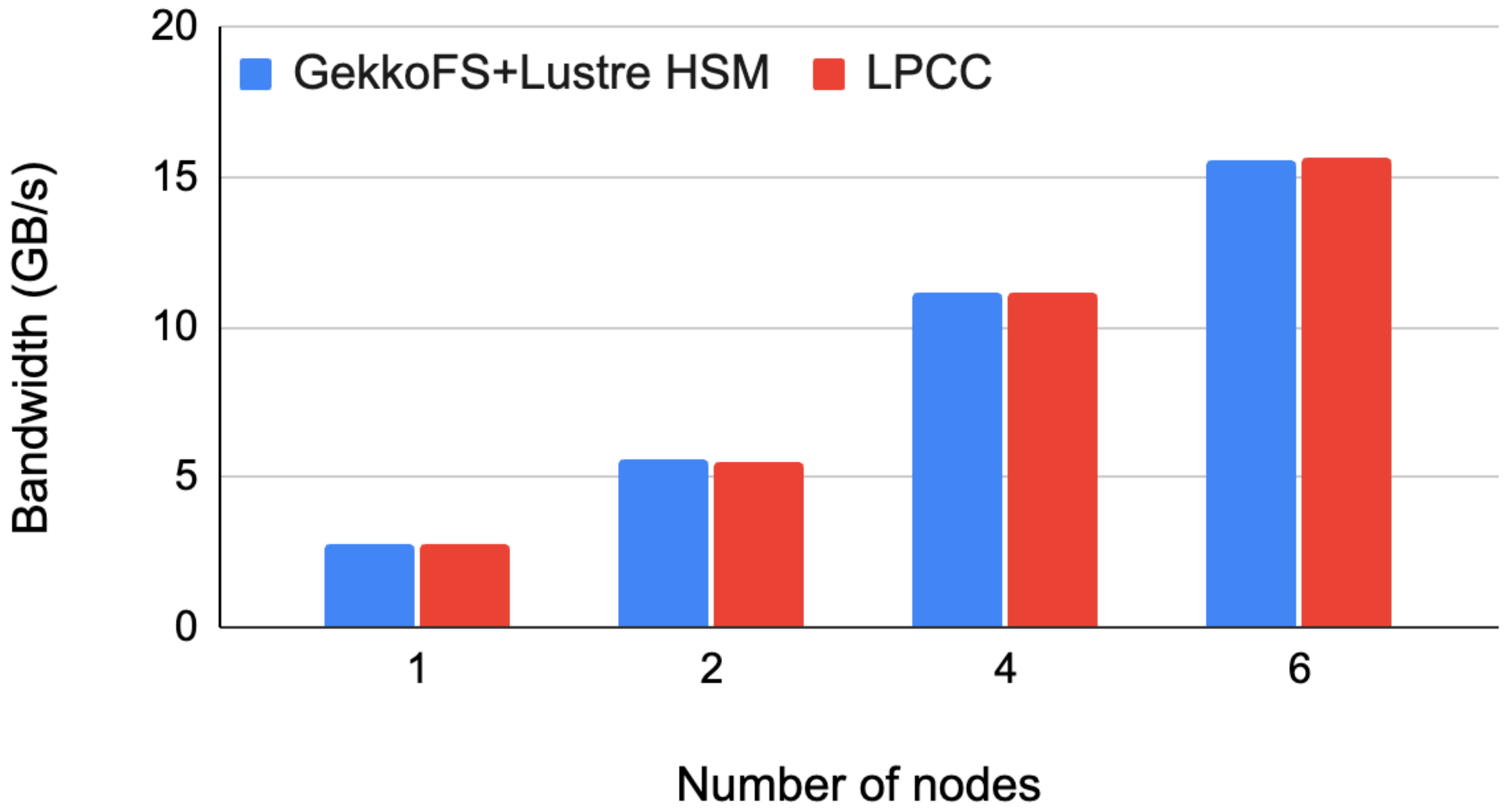
READ PERFORMANCE – HSM INTEGRATED

- IOR test
- Read file 10 times
- One file per process
- IOR transfersize = 1 MB
- 32 processes per each node
- 4 GB file per node
- Overhead: an extra read from Lustre and an extra write on NVI
- LPCC can cache unique files per node



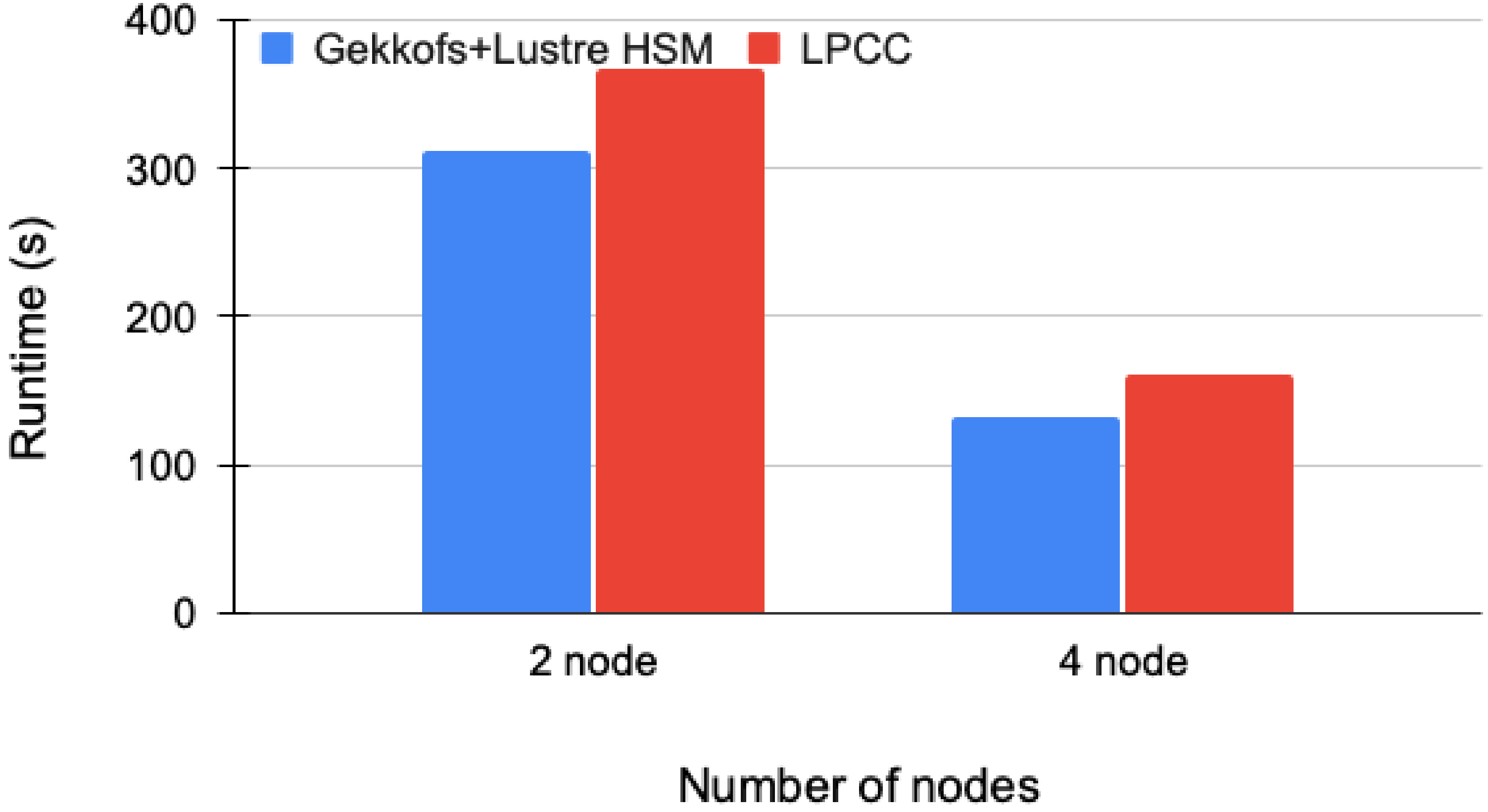
WRITE PERFORMANCE- HSM INTEGRATED

- IOR test
- Read file 10 times
- One file per process
- IOR transfersize = 1 MB
- 32 processes per each node
- 4 GB file per node
- Creates file directly on NVM
- No additional overhead



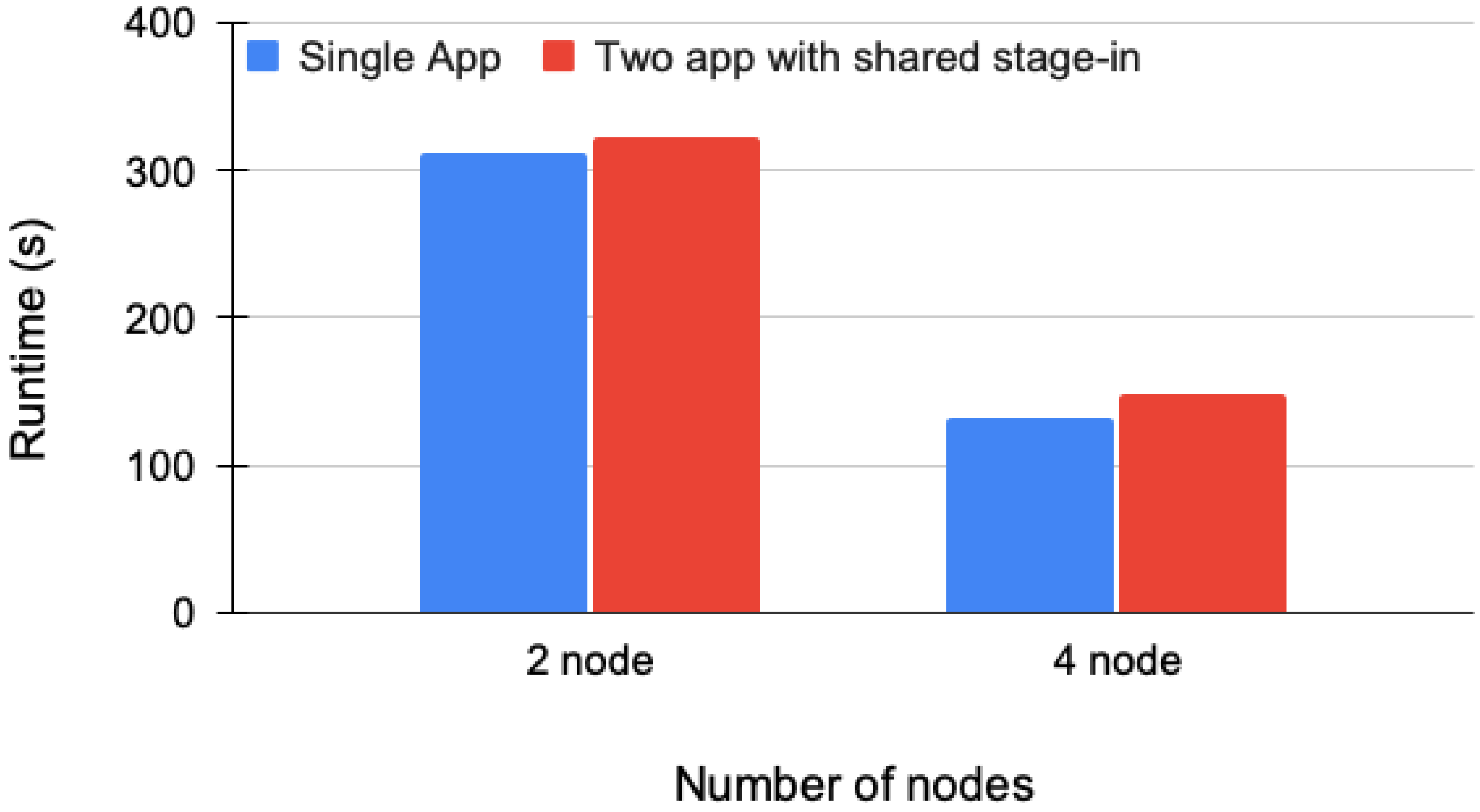
NEK5000

- With 50 steps
- 16 processes per node
- Writes 10 output file
- Each file less than 700 MB
- Stage-in time less than 1 second



NEK5000

- Running two Nek5000 applications
- Shared stage-in dataset
- Each application stage-in a copy of data to node-local storage
- Writes 10 output file
- Reporting the worst runtime between two applications
- Each application separately working on their dataset



BESIII

Done:

- Adapting Darshan to trace BESIII
- Analyzing I/O behavior in small-scale configurations

Ongoing/todo:

- Adapting GekkoFS to run inside BESIII container
- Running BESIII with larger scale configurations
- Providing I/O analysis and possible improvements
- Optimizing performance of BESIII by using GekkoFS

LOFAR

Done:

- Analysed I/O performance in wsclean stage
- Evaluating performance improvement of using local scratch device over parallel file systems

Ongoing:

- Adapting GekkoFS to run LOFAR
- Reducing LOFAR runtime by asynchronously staging data

FIDIUM 2.0?

- Intelligent data staging
- File system interference with applications and NVMe over Fabrics (NVMe-oF)

SUMMARY

- Adding node-local SSDs to storage hierarchy
- Transparent solution for using ad-hoc storage
- Analyzing BESIII and LOFAR on MOGON II & MOGON-NHR
- Compatibility of HPC Application
- Analyzing HPC traces and applications

THANK YOU

JGU

Maysam Rahmanpour	mrahmanp@uni@mainz.de
Reza Salkhordeh	rsalkhor@uni-mainz.de
André Brinkmann	brinkman@uni-mainz.de

Gitlab-Repo: https://storage.bsc.es/gitlab/hpc/gekkofs/

JG|U
JOHANNES GUTENBERG UNIVERSITÄT MAINZ

ADMIRE
malleable data solutions for HPC
ADAPTIVE MULTI-TIER INTELLIGENT DATA MANAGER FOR EXASCALE

IDIUM

MOGON
High Performance Computing

EuroHPC
Joint Undertaking

Bundesministerium
für Bildung
und Forschung