Karlsruhe Institute of Technology

Using On-Demand File Systems in HPC Environments

Mehmet Soysal, Marco Berghoff, Thorsten Zirwes, Marc-André Vef, Sebastian Oeste, André Brinkmann, Wolfgang E. Nagel, and Achim Streit

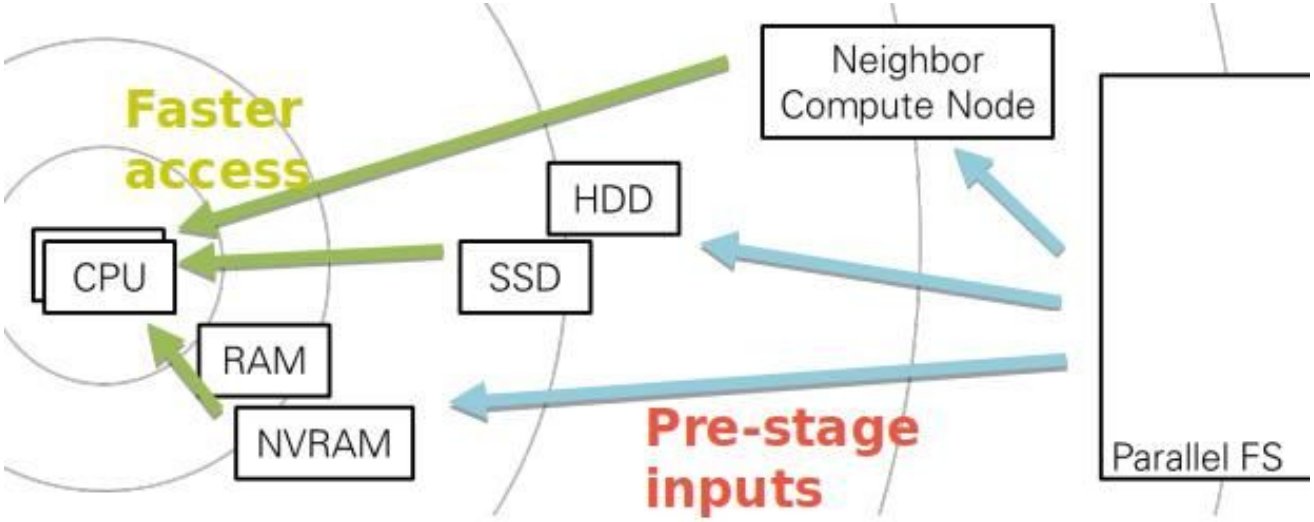
Steinbuch Centre for Computing (SCC) / Scientific Computing and Simulation (SCS)

Overview

- Motivation
- Approach
- Related Work
- Use Cases and Results
- Remarks and Observations
- Conclusion & Future work

Motivation

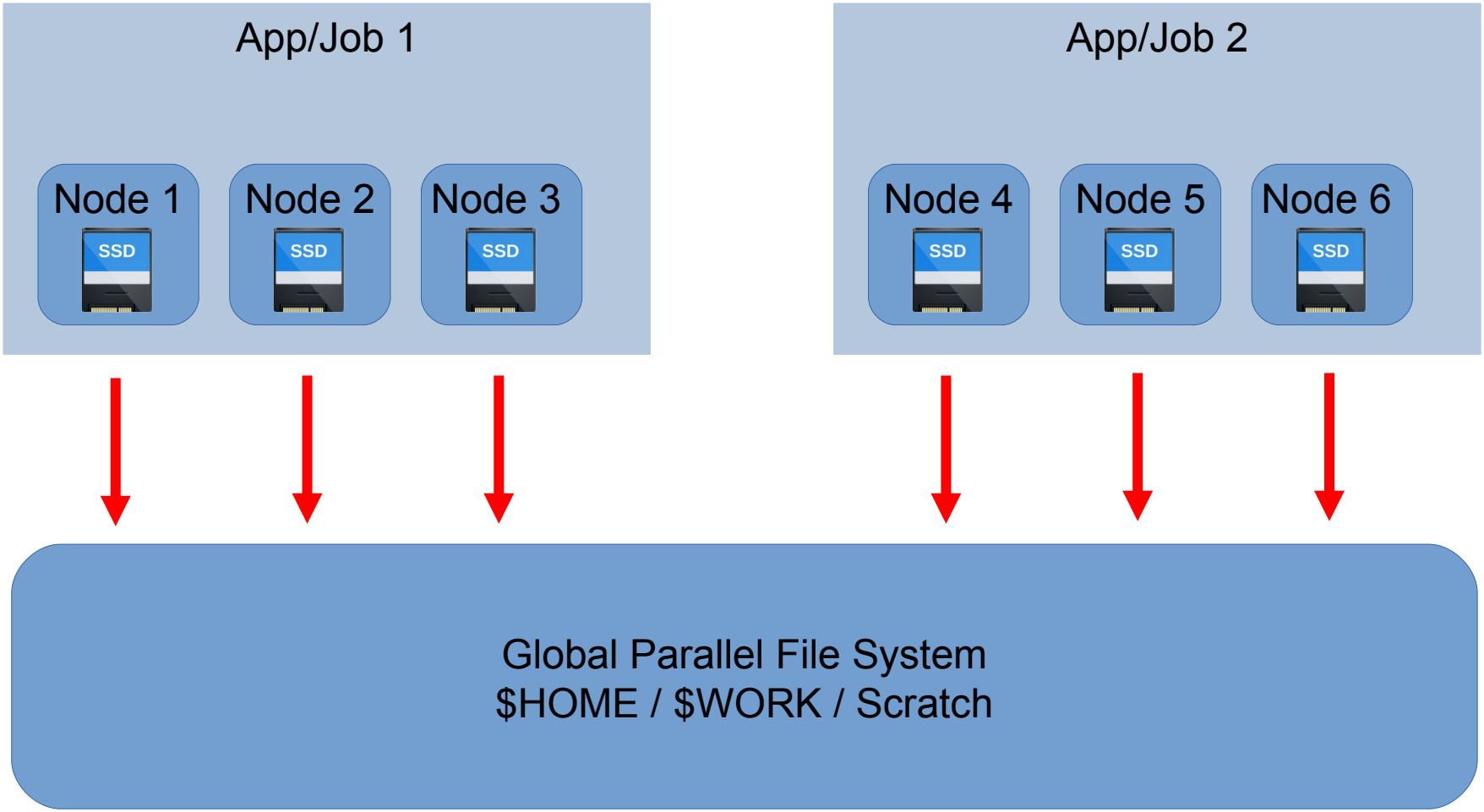
- The I/O Subsystem (parallel FS) is a bottleneck in HPC Systems
  - Bandwidth, metadata or latency
- Shared medium
- Applications interfere with each other
- New storage technologies (SSD, NVMe, NVRAM)



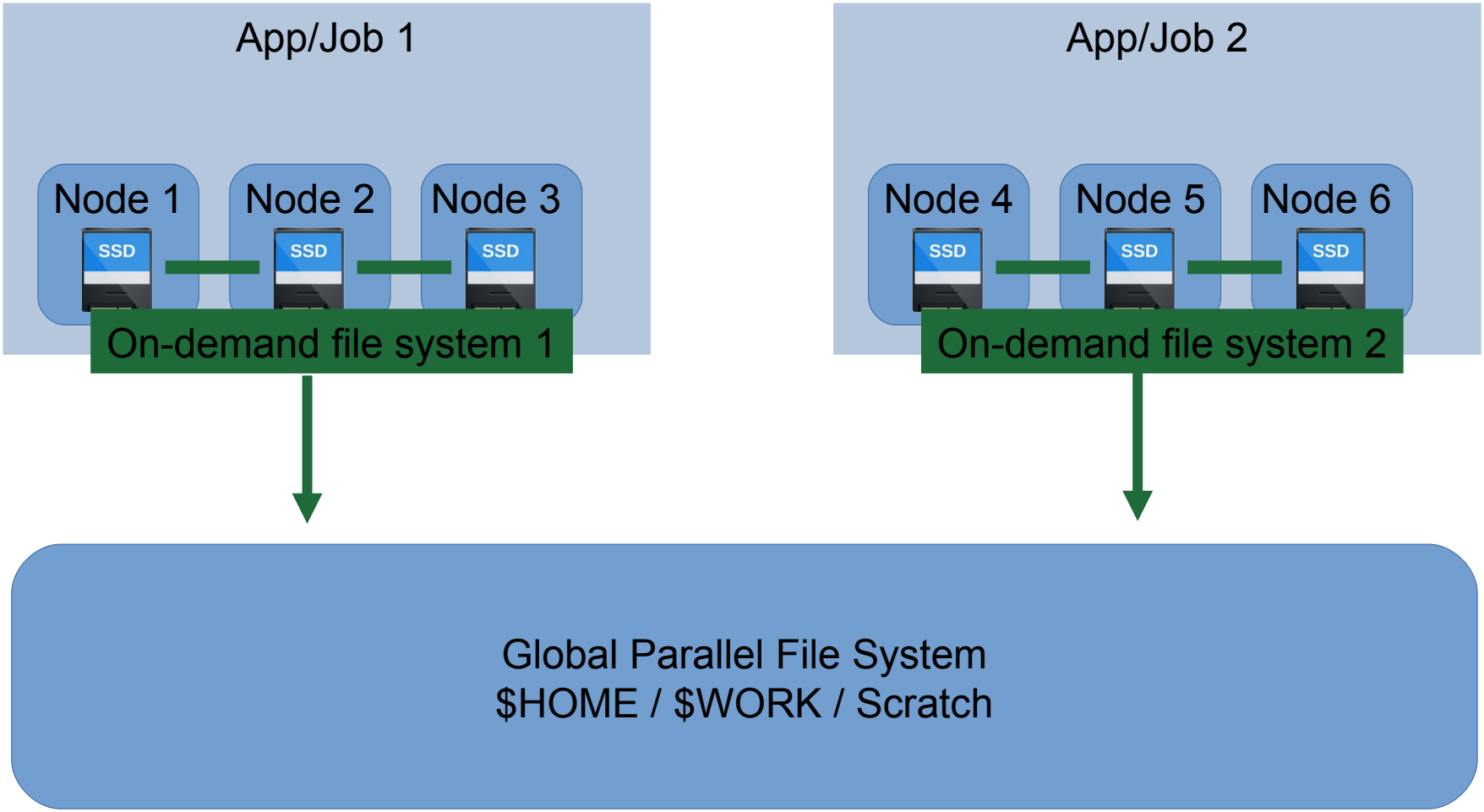
Motivation

- Proposed Solution
  - Bring data closer to compute nodes
  - On-demand file system (ODFS) node-local storage
  - Tailor private ODFS
- Advantages
  - Dedicated bandwidth / IOPS
  - Independent to global file system
  - Low latency due to SSD / NVMe / NVRAM
  - No code changes needed to application

HPC: current file system usage



HPC: usage with on-demand fs



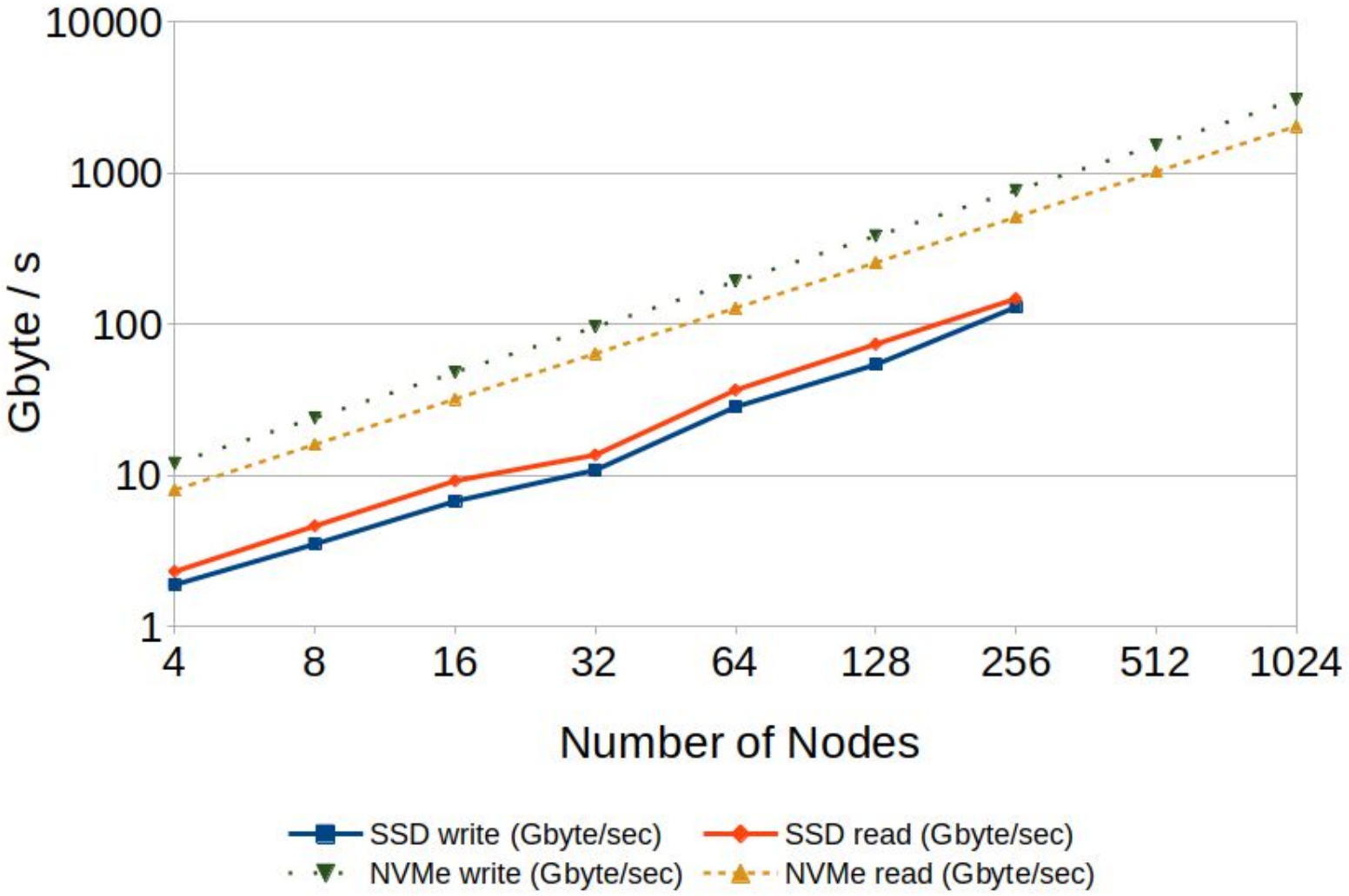
Related Work / Approaches

- File system features
  - Spectrum Scale (GPFS) - HWAC
  - Lustre – PFL / DOM / PCC
  - Beeond – Storage pools
- Hardware solutions
  - Solid state disks
  - Burst buffers
  - In bound cache
- Libaries
  - MPI-IO
  - Sionlib
  - HDF5 / NETCDF
  - ADIOS
- System reconfiguration
  - Dynamic Remote Scratch
  - Ramdisk storage accelerator
  - BeeGFS On Demand - BeeOND
  - Lustre On Demand - LOD

Testing Environment

- ForHLR II Cluster @KIT
  - 1152 Nodes / 2 X E5-2660 v3 (20 cores) / 64 GB RAM
  - 2 Island (816 / 336 Nodes) / 56 Gbit per node / CBB Fabric
  - Local SATA-SSD (480GB) per node - approx. 600 / 400 MB R/W
- Scenarios
  - Generic Benchmark
  - Two use cases from our users (240 Nodes + 1)
    - OpenFOAM
    - NAStJA
  - Concurrent data staging (23 Nodes + 1)

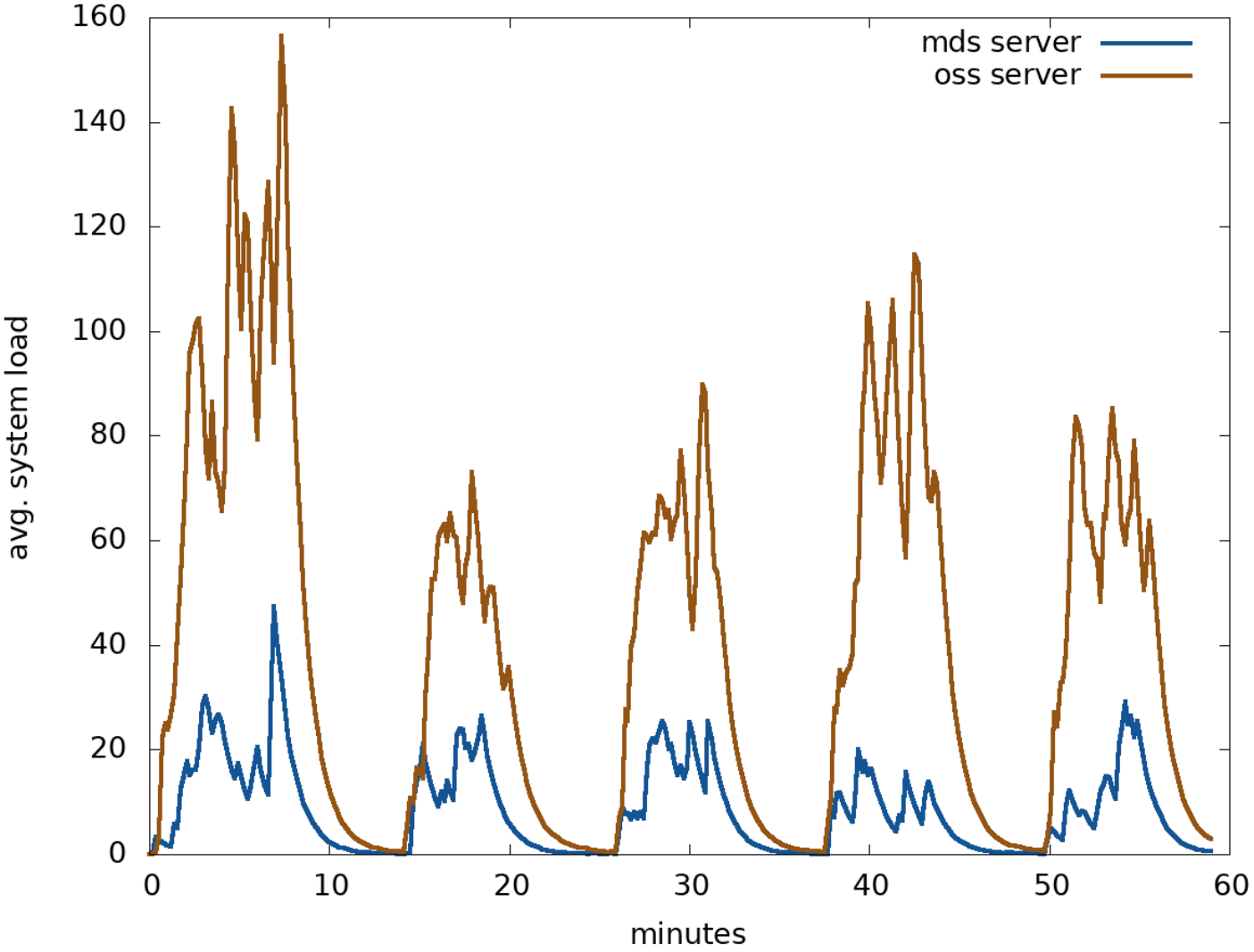
Throughput with IOZone



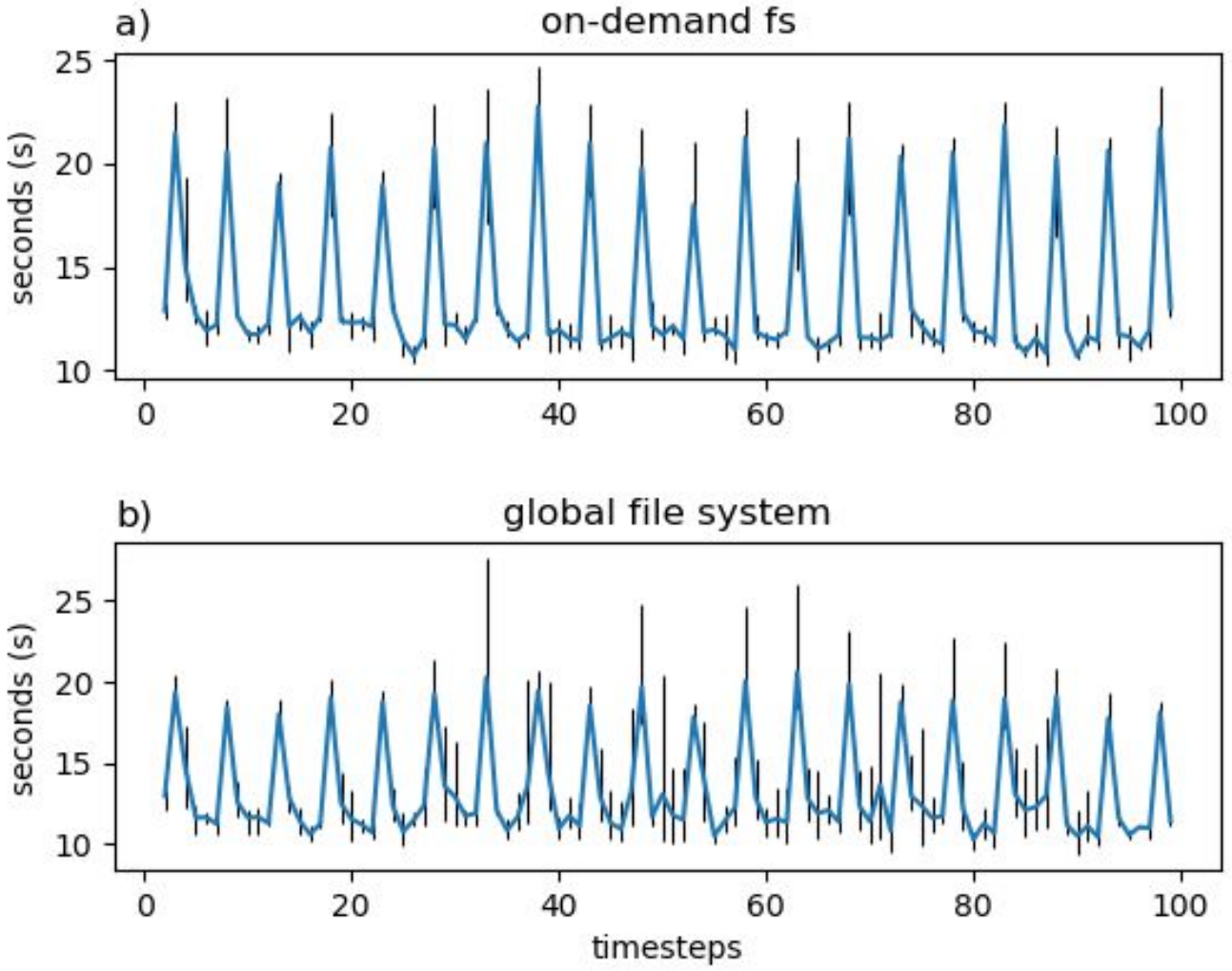
Use cases

- NAStJA
  - Use case 1: 240 nodes / 1 Block per core
    - 4800 files / 4800MB per snapshot
  - Use case 2: Data staging with 16, 19, ans 20 cores per node / 23 nodes
    - Concurrent stage out
- OpenFOAM
  - Use case 1: Laboratory burner flame
    - ~450k files / 120 GB per snapshot
  - Use case 2: Mixing methane / air
    - Generates files for use case 1 / write at high frequency results
- Use cases are user provided and actively used

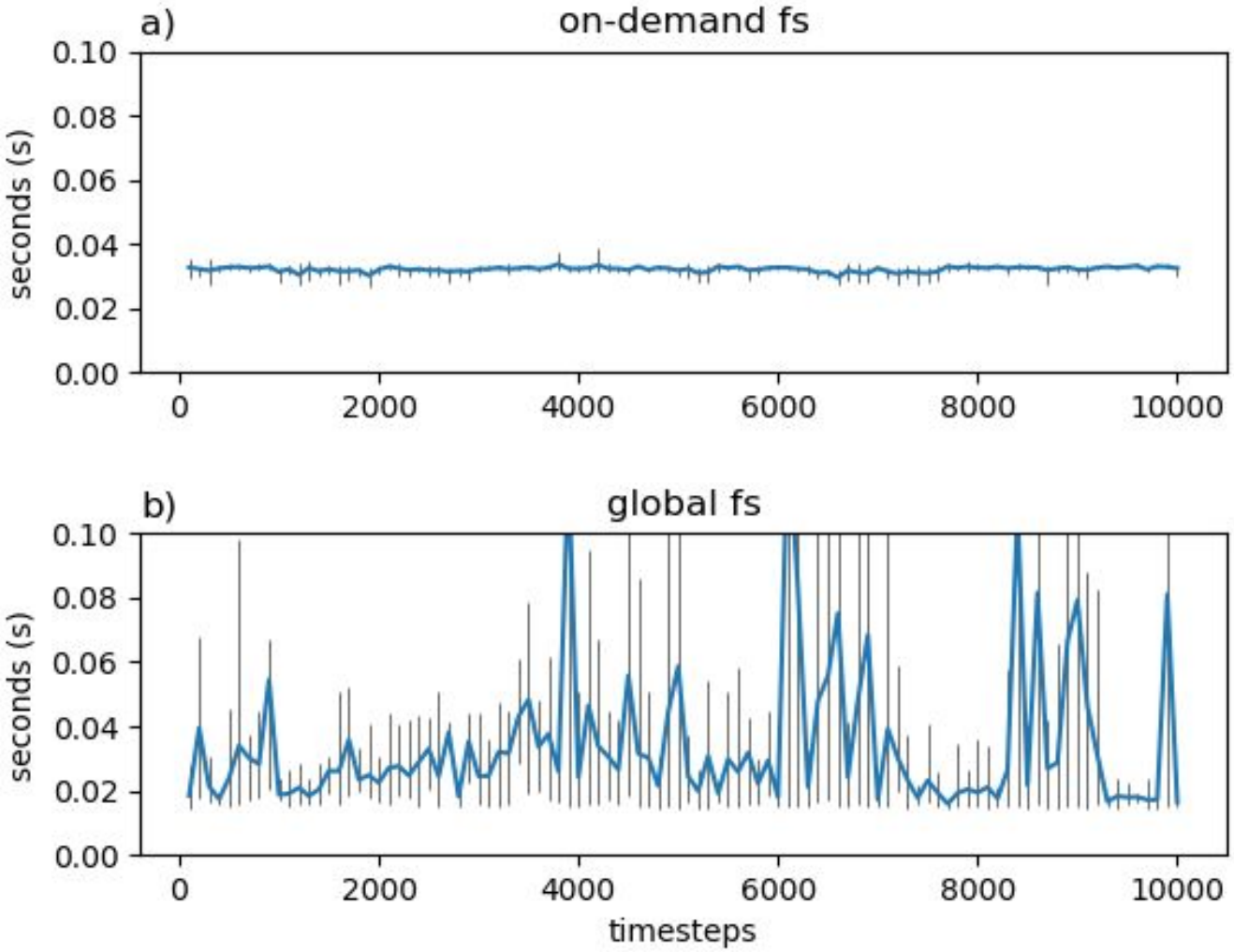
Average load on global FS (NAStJA)



OpenFOAM use case 1



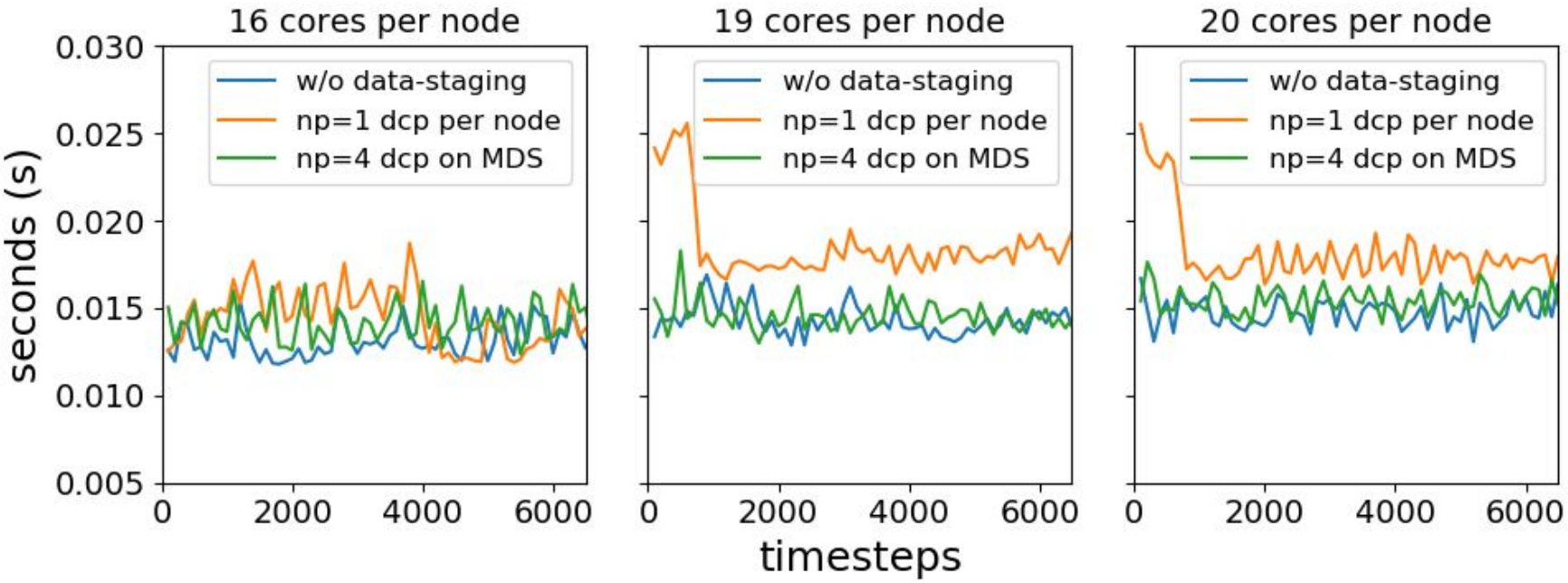
NAStJA use case 1



Concurrent data staging

- Application (NAStJA) runs on 23 nodes + 1 MDS
  - Application runs with 16, 19 and 20 tasks per node
    - Comparative run without data staging
    - Data staging with 1 process per node
    - Data staging on MDS with four processes
- Impact on application with concurrent data staging?

NAStJA use case 2 / stage out



- Using data staging on MDS has only minimal impact
- With 19 and 20 cores for application very high initial peaks
- Fast data staging – Slow data staging
  High impact – Low impact

Remarks and observations

- Loopback device
  - Speedup
  - Faster cleanup after job
- Storage targets are very small (chunk size/stripe count!)
- Solution for very problematic use cases
  - Applications I/O behavior important

Conclusion & future work

- Reduces load on global file system
- Easy to set up
- Some application might run slower
  - I/O analysis helpful
- Topology awareness
- In-situ post processing

- Add file systems(Ceph, GekkoFS) and pre-sets (small/huge files)
- Automatic data staging

Acknowledgement

- ADA-FS Project
- DFG priority programm SPPEXA “Software for exascale Computing”
- Steinbuch Centre for Computing

- Contact: Mehmet.Soysal@kit.edu

Questions?