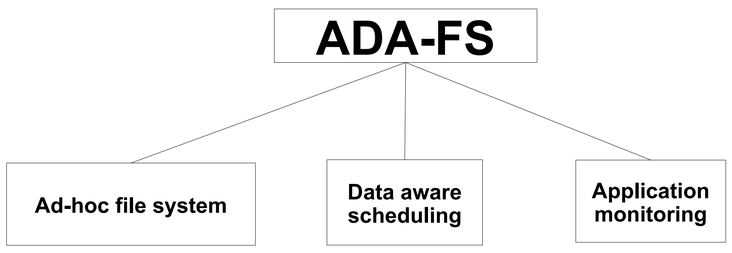
ADA-FS
Advanced Data Placement via Ad-hoc File Systems

Towards High-Performing On-Demand File Systems at Extreme Scales

Mehmet Soysal[1], Marc-André Vef[2], Sebastian Oeste[3], Achim Streit[1], André Brinkmann[2], Michael Kluge[3]

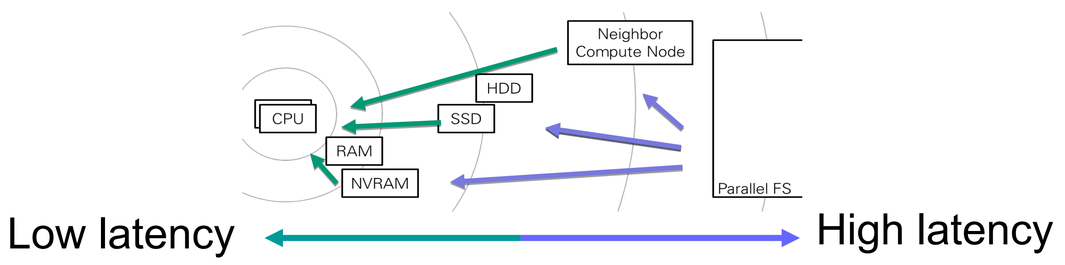
1. Project overview

- New project in the second period of
  The Priority Programme „Software for Exascale Computing“ (SPPEXA)
- Funded trough the „Deutsche Forschungsgesellschaft“ (DFG)
- Started 02/2016 till 02/2019



2. Motivation

- Today, large parallel file systems are shared between many concurrently running applications which suffer from I/O bottlenecks and unreliable IOPS on HPC systems.
- As a result, emerging storage technologies (e.g., SSD, NVRAM, HBM, …) cannot be fully utilized by applications.
- Further, cluster topologies are becoming increasingly more complex within HPC systems.



3. Ad-hoc file system

- Based on the Fuse library.
- The cluster‘s batch system deploys the file system on a number of nodes that are allocated for a job with a single namespace.
- Relaxed POSIX I/O Semantics.
  - Ignore Metadata fields, such as mtime, atime, filesize
  - Simplify file system protocols
  - Avoid locking whenever possible
- Scalable metadata approach.
  - Use Key-Value store
  - No rigid data structures (e.g. directory blocks)
  - Last writer wins
- Distribute data across disks (taking locality into account).
- The file system uses, among others, the cluster topology to decide where to place the data (i.e., which node and storage device).

4. Data aware scheduling and data management

- Interaction with the existing batch environment - no replacement of existing components
- A central I/O planner coordinates which data is staged to which nodes.
- Improving batch system node allocation with machine learning for better wall-clock prediction.
- A distributed file system is deployed on the local storage of allocated compure nodes.
- Detailed knowledge about the cluster topology and task placement improves initial data placement within the distributed file system.
- A new data management concept is introduced to handle data staging within complex environments, offering a global identifier for data regardless of its physical location.
- Pre-staging data using RDMA (NVMe).

5. Application monitoring & Resource discovery

- I/O monitoring of applications and analyzing the usage of the private file system provides hints for better data staging.
- I/O tracking of applications would also provide information for consecutive runs, since most applications show similar I/O behavior during their runs.
- Complex memory hierachies make performance a matter of locality.
- Information about node-local resources (e.g., available local storage) and topology information are mandatory for a central I/O planer.
- Bandwidth within a set of nodes vs. bandwidth to the PFS helps to choose the right caching places (e.g., neighbor nodes).

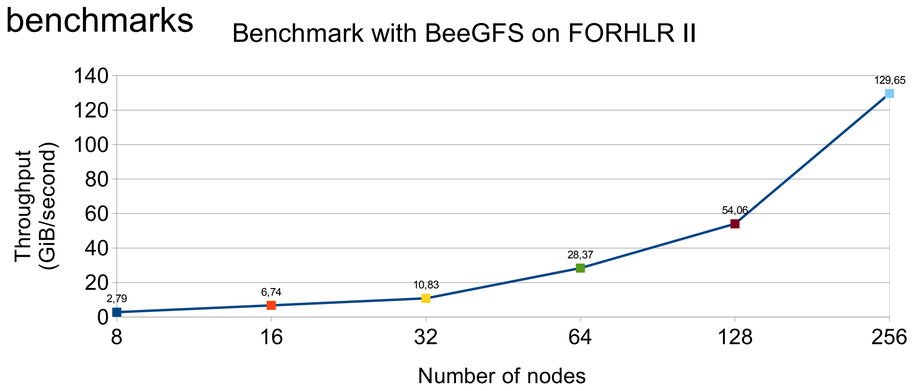
6. Status

- Ad-Hoc file system with relaxed POSIX semantics is under development.
  - Basic design choices already finished
  - Using BeeGFS as On-Demand filesystem as a prototype
- Requesting on-demand private PFS and data staging available @ForHLR2.
  - Prototype with moab‘s job-chaining functional
- Prototype tools for topology discovery under testing.
- Tools for application monitoring are in progress.
- Testing with BeeGFS until ad-hoc file is available.
- Benchmarks with BeeGFS (full POSIX) are promising.
  - Low impact on job – No impact on global PFS during job
  - Speed of SSDs are limiting factor

7. Future work

- Develop novel distributed ad-hoc filesystem
- Test application behavior with relaxed POSIX semantics
- Develop tools for ADA-FS deployment with support for other batch systems
- Test tools for topology and resource discovery on heterogeneous systems
- Track I/O behavior of different applications and create fingerprint
- Use information of I/O behavior for optimized data placement
- Evaluate impact on application with optimized data placement
- Develop prototype of a new data management concept with “workpools”
- Evaluate impact on running jobs during data pre-staging

8. Initial benchmarks



- Initial benchmarks on ForHLR II (Cluster at KIT) with the IOzone benchmark tool.
  - Using node-local storage
  - Fat Tree Topology (approximately 50 Gbit per node)
  - Compute nodes with local SSD (R/W 600/400 MB/s)
  - Measured write performance with one process per node

1) Karlsruhe Institut of Technology / Steinbuch Centre for Computing
2) Johannes Gutenberg-Universität Mainz / Zentrum für Datenverarbeitung
3) Technische Universität Dresden / Zentrum für Informationsdienste und Hochleistungsrechnen

ada-fs.github.io