TECHNISCHE UNIVERSITÄT DRESDEN

Center for Information Services and High Performance Computing (ZIH)

Advanced Data Placement via Ad-hoc File Systems at Extreme Scales (ADA-FS)

Andreas Knüpfer, Wolfgang E. Nagel, André Brinkmann, Achim Streit, Michael Kluge, Sebastian Oeste, Marc-André Vef, Mehmet Soysal

SPPEXA Annual Meeting 2016 Garching, 2016-01-27

ADA-FS: Advanced Data Placement via Ad-hoc File Systems at Extreme Scales

- New project in the second funding period of SPPEXA
- Addressing SPPEXA topics:
  - system software and runtime libraries
  - data management

- Technische Universität Dresden: PI Wolfgang E. Nagel, Andreas Knüpfer, Michael Kluge, Sebastian Oeste
- Johannes Gutenberg University Mainz: PI André Brinkmann, Marc-André Vef
- Karlsruhe Institute of Technology: PI Achim Streit, Mehmet Soysal

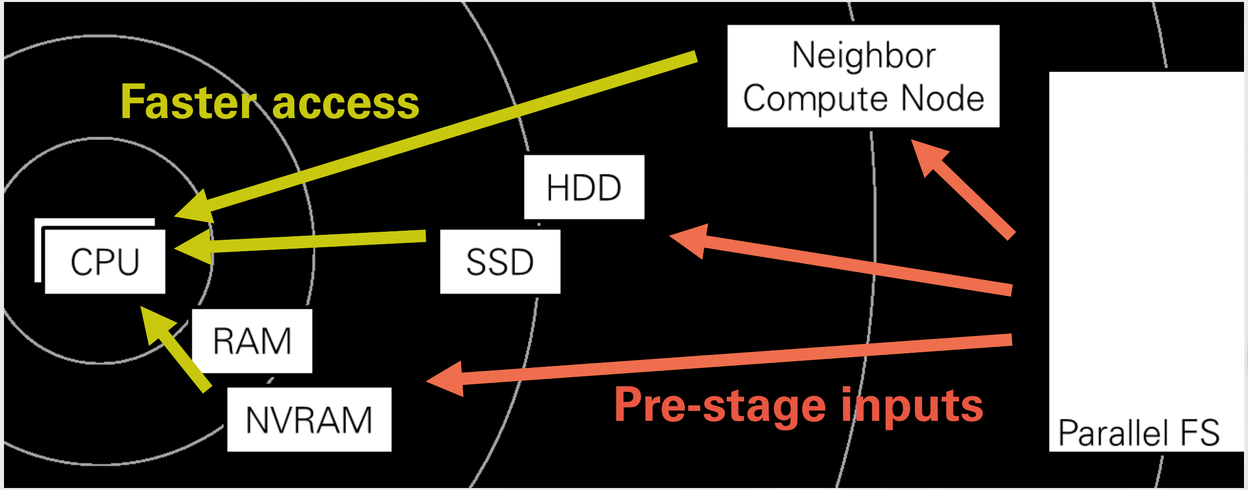
Project Rationale

I/O Challenges at Exascale

- I/O subsystem is the slowest one in a HPC machine (bytes per flop, latency)
- Shared medium: no reliable bandwidth, no good transfer time predictions
- Upcoming architectures with “fat nodes” and intermediate local storages

Goal: optimize I/O

- Using additional storages
- Transparent solution for parallel applications
- Pre-stage inputs early, cache outputs



Background: Upcoming HPC Architectures

- Expected Exascale architectures and announced 100 PF machines:
  - Orders of magnitude more processing units / compute power per node
  - Local intermediate storages, must be used for decent I/O performance
  - More complicated and machine-specific storage hierarchy
- Bandwidth perspective:

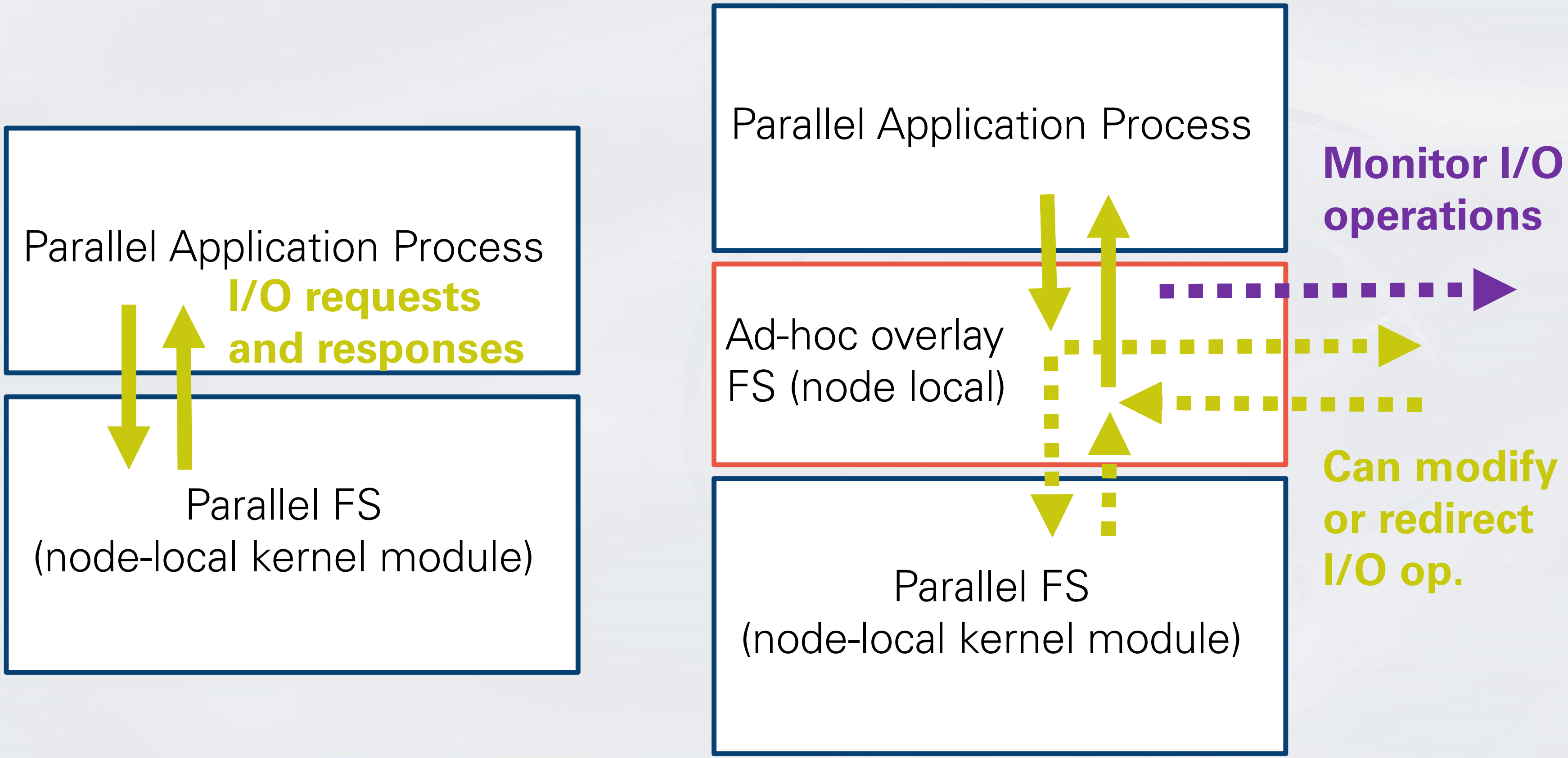
	# compute nodes N	Global I/O bandwidth S	Caching bandwidth C	Break-even point N*= S/C
SuperMUC / LRZ Phase 1	9400	200 GB/s	(0.5 GB/s)	400
SuperMUC / LRZ Phase 2	9421+3072	250 GB/s	(0.5 GB/s)	500
Titan / ORNL	18688	240 GB/s	(0.5 GB/s)	480
Summit / ORNL (anounced)	3400	1 TB/s	1.6 GB/s (assumed)	625

Proposed Solution

- Ad-hoc overlay file system
  - Separate overlay file system per application run
  - Instantiated on the scheduled compute nodes
  - From before the parallel application starts until after it finishes
- Central I/O planner
  - Global Planning of I/O including stage-in/-out of data, for all par. jobs
  - Optimization of data placement in the ad-hoc file system (resp. nodes)
  - Integration with systems batch scheduler
- Application monitoring, resource discovery
  - I/O behavior, machine-specific storage types, sizes, speeds, …

Overlay FS

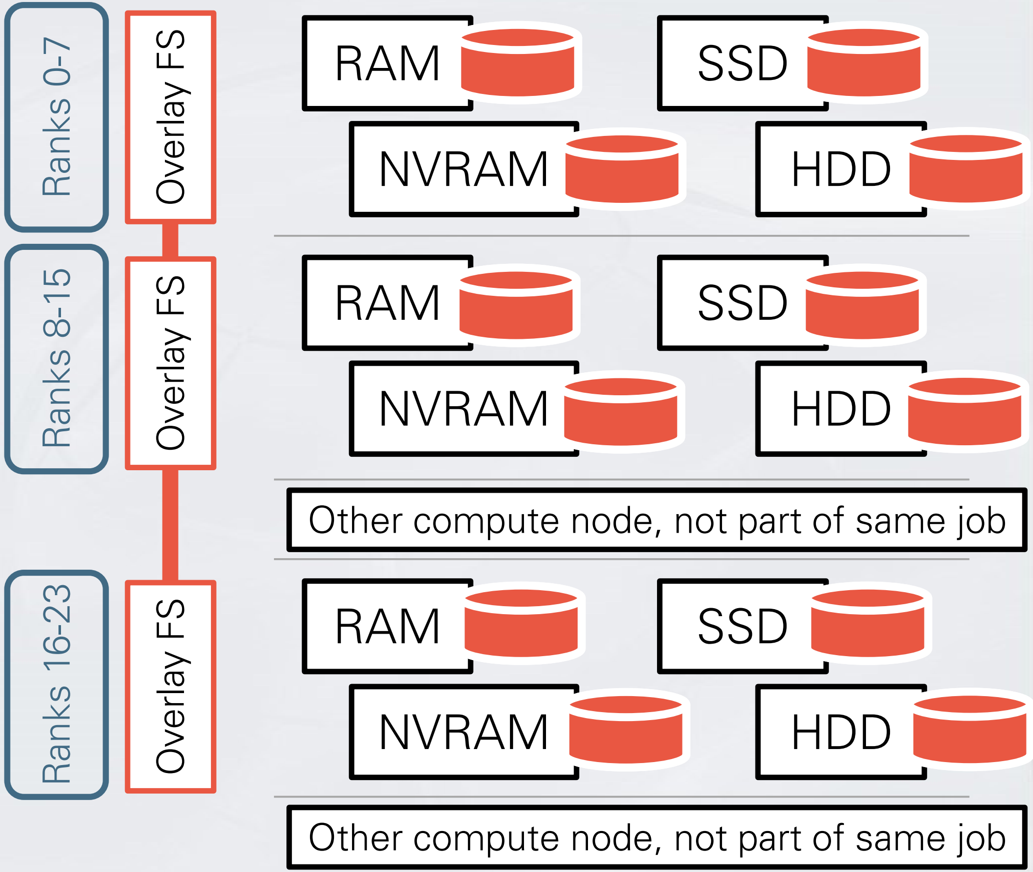
- Ad-hoc overlay file system:



Overlay FS

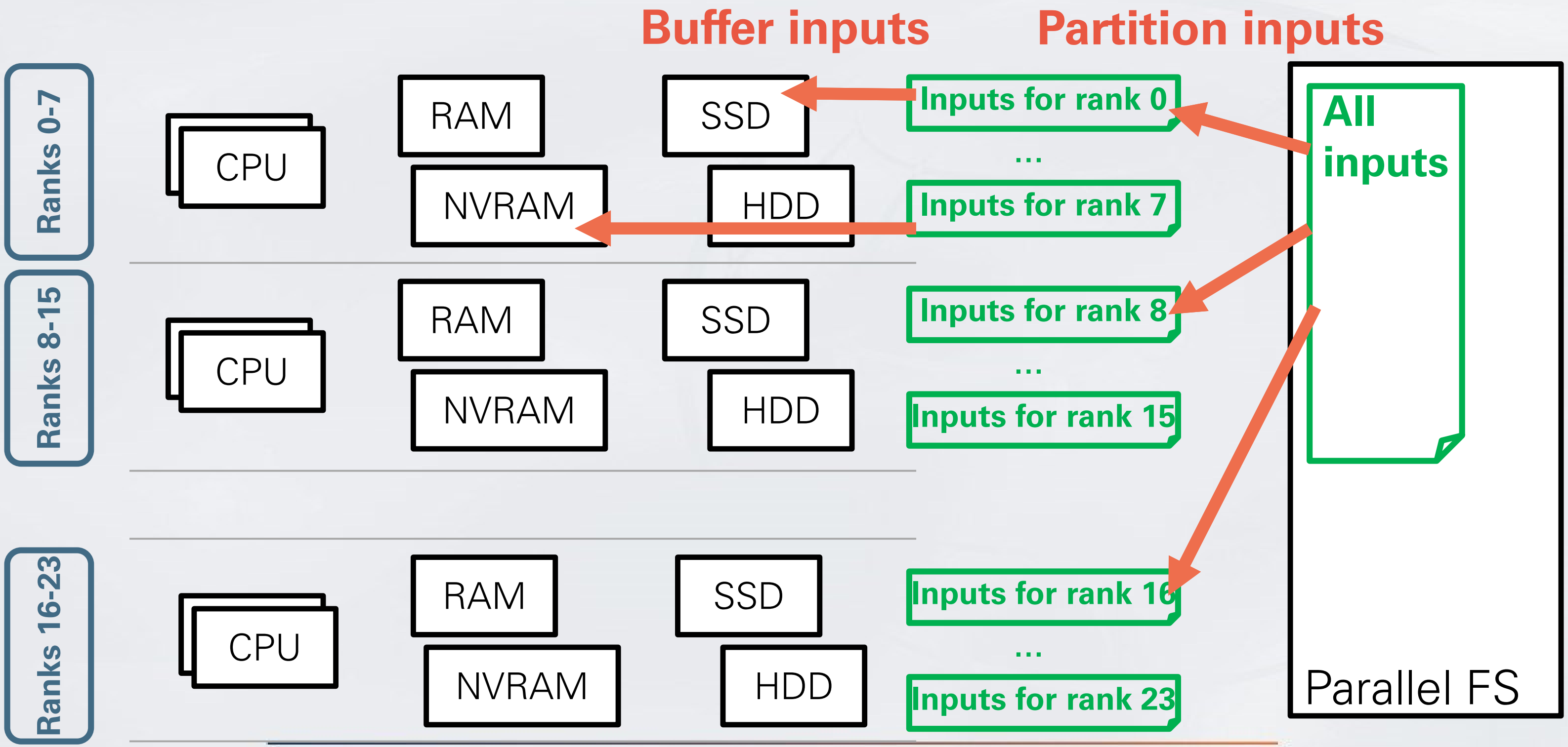
- Ad-hoc overlay file system:
  - Separate overlay file system per application run
  - Instantiated on scheduled compute nodes

- Allocate buffers in local storages



Overlay FS

- Ad-hoc file system present from before the application starts until after it finishes
  - Allow buffering beforehand/afterwards



I/O Planner

- Individual parallel jobs cannot optimize I/O performance
  - Global parallel file systems are shared resources
  - No bandwidth guaranties, no reliably I/O time estimations
- Central I/O planner schedules I/O operations
  - Assume all parallel jobs are under control of ADA-FS
  - Assume coarse-grained I/O behavior known: I/O phases and I/O gaps
  - Allow I/O phases of running jobs with priority
  - When stage-in inputs for future jobs? To which nodes?
  - When stage-out outputs from past jobs?
  - Integrate with job scheduling

Application Monitoring and Resource Discovery

- Application monitoring
  - Monitor parallel applications, record I/O behavior
  - Generalize I/O behavior for types of applications
  - Predict I/O phases and I/O gaps → Input for I/O planner
  - Predict input partitioning → Input for Overlay FS buffering
- Resource discovery and monitoring
  - Discover machine-specific storage types, sizes, reliable local speed, …
  - Monitor resource allocations by parallel applications,
determine what is left for local buffers → Input for Overlay FS deployment
- Approach: start with explicit specifications, research automated solutions

Challenges and Benefits

Restrictions

1. Each file/object is only accessed by a single application, yet from many nodes at a time
2. No **`'ls -a'`** type operations in the overlay FS
3. No "communication via files" type operations

Restricted POSIX FS semantics as additional research topic

Practical benefits

- Applications will be required to use additional burst buffers for I/O
- Separation of concerns: decouple application logic from storage hierarchy
- Enable bandwidth guarantees and reliable timing predictions for I/O operations, when combined with central I/O planning

Consortium and Previous Work

- Wolfgang E. Nagel (Speaker), TU Dresden
  - Parallel performance monitoring and I/O monitoring
  - Flexible storage system design for local HPC infrastructure
- André Brinkmann, Johannes Gutenberg-Universität Mainz
  - Development of metadata server-free parallel file system
  - Investigation of file system access traces to verify key assumptions 1 and 2 (both together with BSC)
- Achim Streit, Karlsruher Institut für Technologie
  - Job scheduling and resource management for HPC and distributed systems: planning, self-tuning, brokerage
  - Large-scale data management

Work Plan

WP 1: Ad-hoc File System (led by JGU)

- Design ad-hoc overlay file system
- Coordination between nodes
- Deployment and synchronization with underlying global parallel file system

WP 2: Planning (led by KIT)

- I/O estimation and scheduling
- Optimization of data placement
- Integration with batch job scheduler

WP 3: Discovery and Monitoring (led by TUD)

- Resource and Topology Discovery Dynamic Resource Usage Tracking
- Monitoring of ad-hoc FS behavior and of application I/O

WP4: Integration and Demonstration (led by TUD)

Summary

Project goals

- Improve I/O performance
- Adopt upcoming architectural features
- Transparent to application codes

Steps

- Design overlay file system
- Create central I/O planner
- Discovery, monitoring, learning I/O behavior
- Integration
- Demonstration at scale