OAK RIDGE National Laboratory

Summit Burst Buffer Libraries

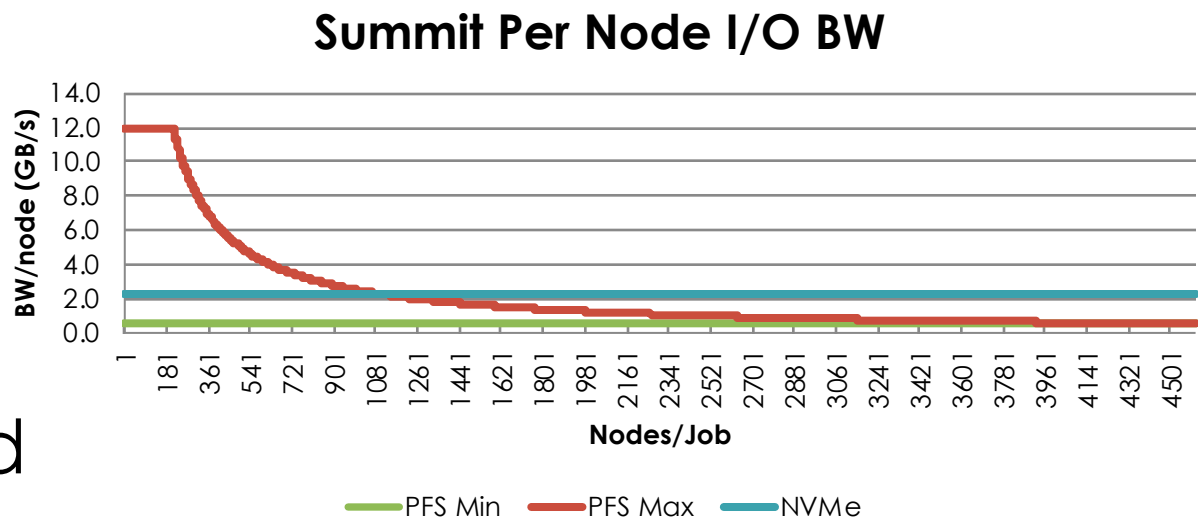
Christopher Zimmer

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

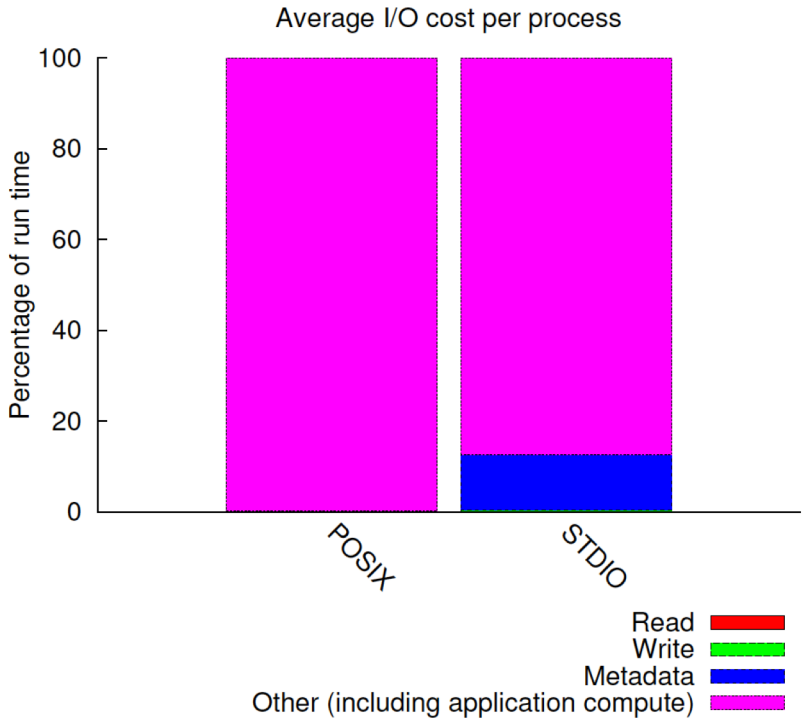
When to use the Burst Buffers

- Alpine GPFS Performance
  - Per node 12-14 GB/s (Without core isolation)
  - Aggregate 2.5 TB/s
    - Full system job will achieve 550 MB/s per node
- Node Local NVME
  - Samsung PM1725A
    - Write 2.1 GB/s
    - Read 5.5 GB/s
  - Scales linearly with Job Size
- Realistically benefit is realized
  - 150 Nodes



When to use continued:

- 8000 Files to GPFS
  - 5 TB of data written
  - 12 seconds spent creating files
  - <1 second spent writing files
- High IOP read workload (Full System)
  - 4k Random Reads (GPFS) ~100million
  - 4k Random Reads (NVMe) ~4.5 Billion
    - 1M per device



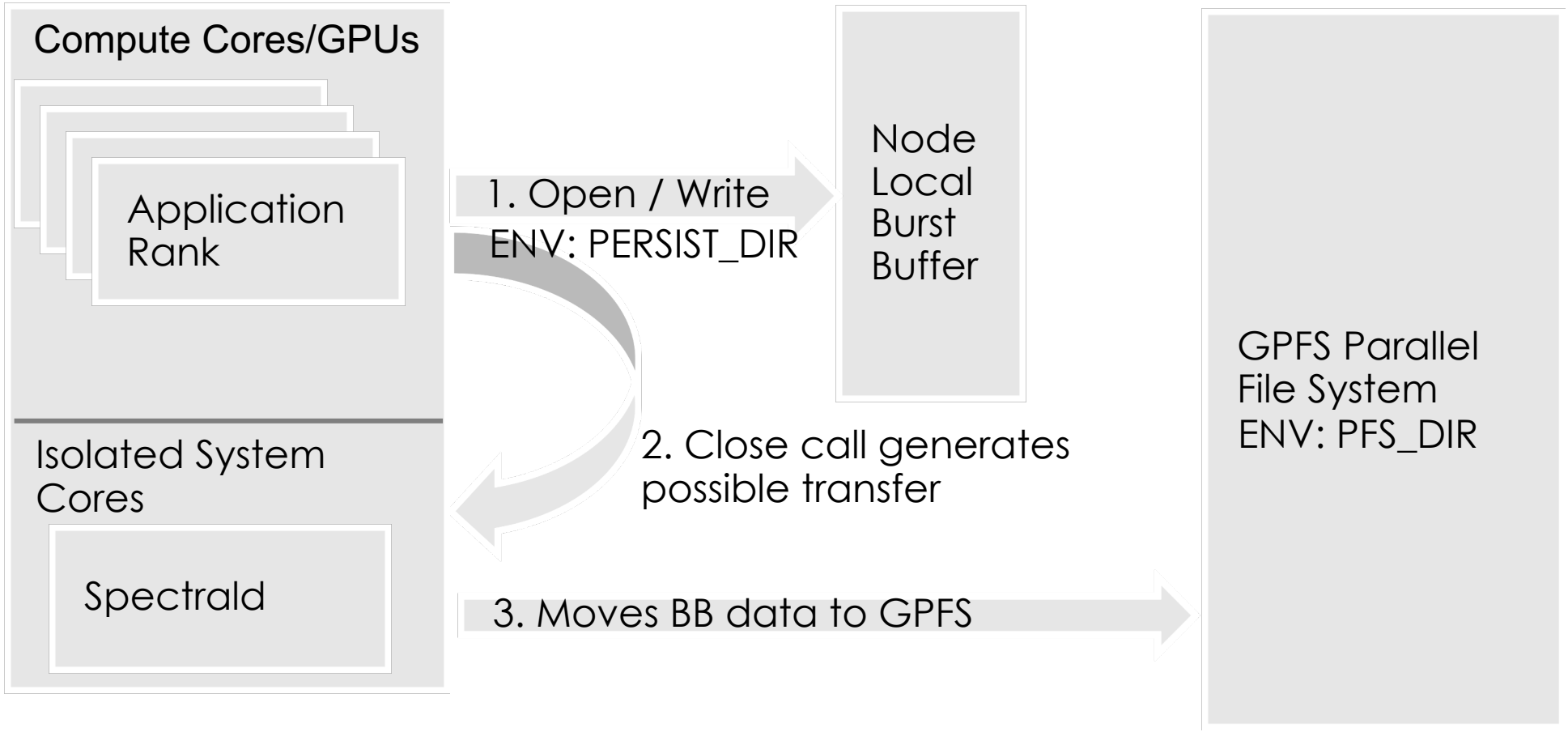
Modes of Use

- Scratch File System -alloc_flags “nvme”
  - Good for ML/DL Workloads
  - Self managed checkpointing
  - Ephemeral storage
- Spectral –alloc_flags “spectral”
  - File per process checkpoints
  - Iterative output
- SymphonyFS –alloc_flags “symphonyfs”
  - Shared file output

Spectral

- On node copy agent
  - Runs on isolated cores as system agent
- Application Interface Transparent
  - Node code modifications (LD_PRELOAD)
    - Changes limited job scripts
  - Application only reasons about a single namespace
- Preserves portability with single namespace
- Non-shared files
- Transaction log of file movement states (Can be used to determine when finished after application run)

Spectral Data Flow



Lammps Example

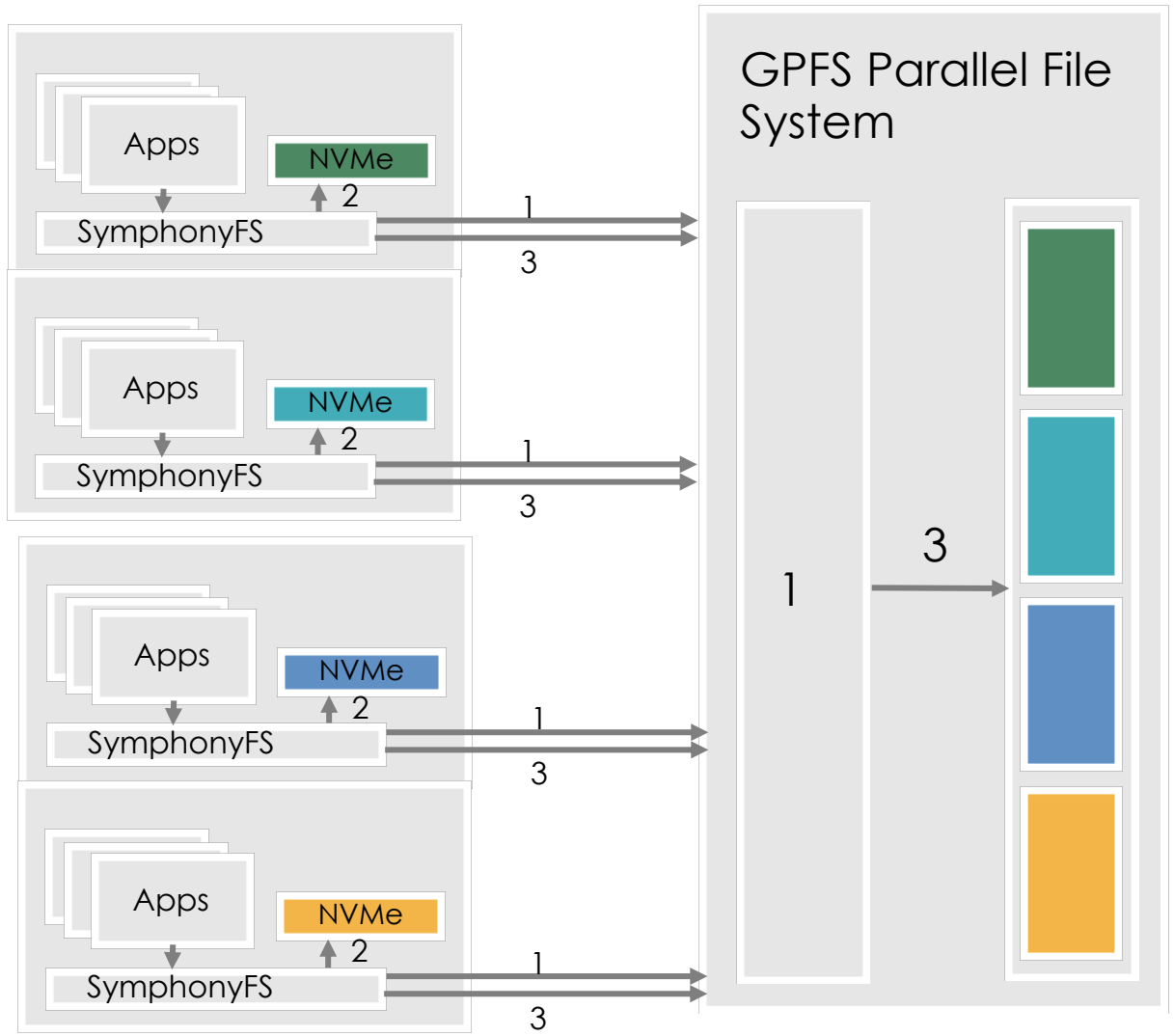
```
#Changes for Spectral
export PERSIST_DIR=${BBPATH}
export PFS_DIR=$PWD/restart
export OMPI_LD_PRELOAD_PREPEND=/ccs/techint/home/cjzimmer/Spectral-Test/build/src/libspectral.so

jsrun --nrs 16 --rs_per_host 1 --tasks_per_rs 6 ${BINARY} -v number_of_atoms ${natoms} -v output_dir ${PERSIST_DIR} < ${LAMMPS_INPUT_FILE}
```

- PERSIST_DIR sets where lammps actually writes to via config file
- When spectrald detects a write into PERSIST_DIR, the data is moved to PFS_DIR outside of the application
- OMPI_LD_PRELOAD_PREPEND will be set by module load

SymphonyFS

- Single (FUSE) name space
- File per process, shared file
- Operational model:
  - Uses NVMe as a write-back extent cache
  - Log based file system
  - Reconstructs file on parallel file system

SymphonyFS Data Flow



1. File open (Meta data handled through GPFS)'
2. Apps write data (buffered into NVMe's)
3. Upon file close and flush. SymphonyFS reconstitutes file on GPFS.

SymphonyFS

- Limitations
  - Non-overlapping writes
  - Read after write
    - Must be flushed to parallel file system
      - Reads come from parallel file system

Questions/Interest in early access?

- Spectral - zimmercj@ornl.gov

- SymphonyFS – brumgardcd@ornl.gov