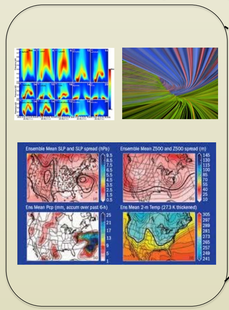
BurstFS: A Distributed Burst Buffer File System for Scientific Applications

Teng Wang[1] Kathryn Mohror[2] Adam Moody[2] Weikuan Yu[1] Kento Sato[2]

Florida State University[1] Lawrence Livermore National Laboratory[2]

Problem Statement

Exascale Computing



Too Much Data Pressure!

Supercomputer

Intermediate Data

Solution!

Storage System

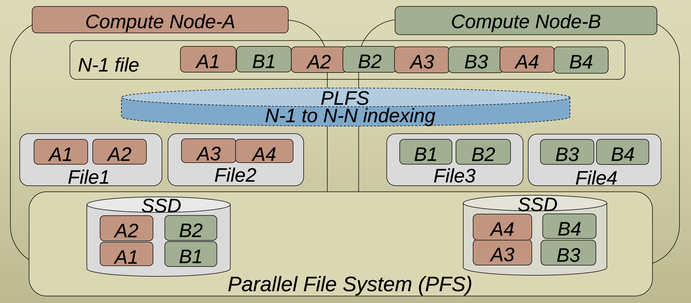
OUR GOAL

Design a distributed file system on top of node-local BBs to handle bursty I/O workloads for scientific applications.

*Compute Node with local **burst buffer (BB)***

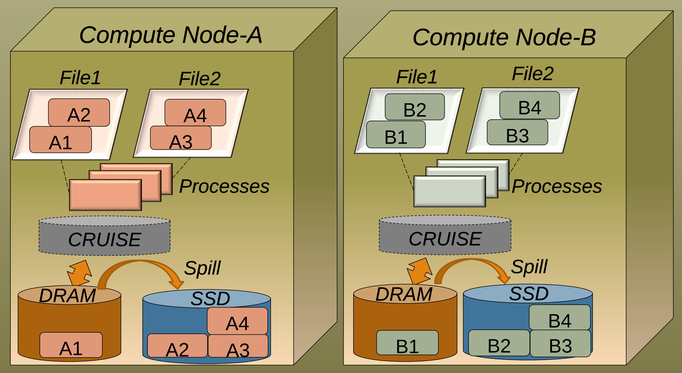
Existing Solutions

1. *Intercepting I/O using PLFS.*



- Establishes a PFS on all BBs.
- Supports N-1, N-N patterns.
- Suffers from PFS contention.
- High metadata overhead.

2. *Intercepting I/O using CRUISE.*

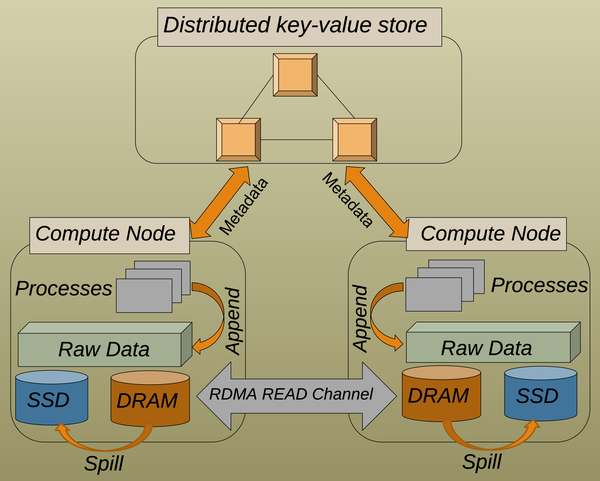


- Uses both local DRAM and SSD.
- Scales with local read and write.
- Less efficient in N-1 writes.
- No support of remote read and write.

Our Solution

1. *Complementing PLFS-based solution.*

- Local writes incur less contention.
- Local writes deliver high scalability.
- Service metadata using distributed key-value store.



2. *Complementing CRUISE-based solution.*

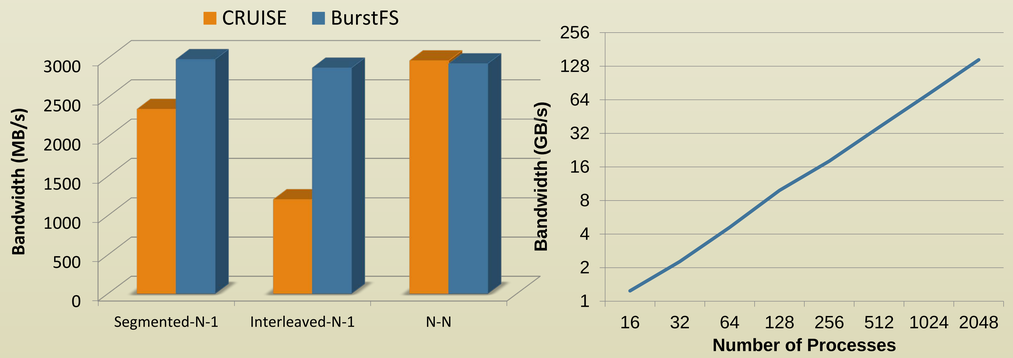
- Uses log-structured write, which works well in both N-1 and N-N writes.
- Provides efficient, RDMA-based read service.

Key features

- Log-structured write and RDMA-based read.
- Distributed key-value store for metadata service.

Preliminary Results

- Catalyst cluster with 324 nodes, 128GB DRAM, 512GB SSD.



- Write Bandwidth.
  - Compare CRUISE and BurstFS on N-1, N-N write patterns, two processes on the same node write to local DRAM.
- Scalability.
  - Use 16 processes on each node write to their local SSD.

Conclusions & Future Work

- **Conclusions**
  - Analyzed the representative I/O solutions for node-local BB.
  - Proposed BurstFS, a distributed burst buffer file system on node-local BBs.
  - Complement existing solutions with optimized write, read, metadata service.
  - Initial evaluation demonstrates good write bandwidth and scalability.
- **Future Work**
  - Provide efficient RDMA-based read support.
  - Spread metadata workload using distributed key-value store.
  - Evaluate against scientific I/O workload.

References

- R. Rajachandrasekar, A. Moody, K. Mohror, and D. K. Panda. A 1 PB/s file system to checkpoint three million MPI tasks. In HPDC'13. ACM, 2013.
- J. Bent, G. Gibson, G. Grider, B. McClelland, P. Nowoczynski, J. Nunez, M. Polte, and M. Wingate. PLFS: a checkpoint filesystem for parallel applications. In SC'09. ACM, 2009.

Acknowledgement

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. (LLNL-ABS-675972)