Concurrent Write Sharing: Overcoming the Bane of File Systems

Garth Gibson

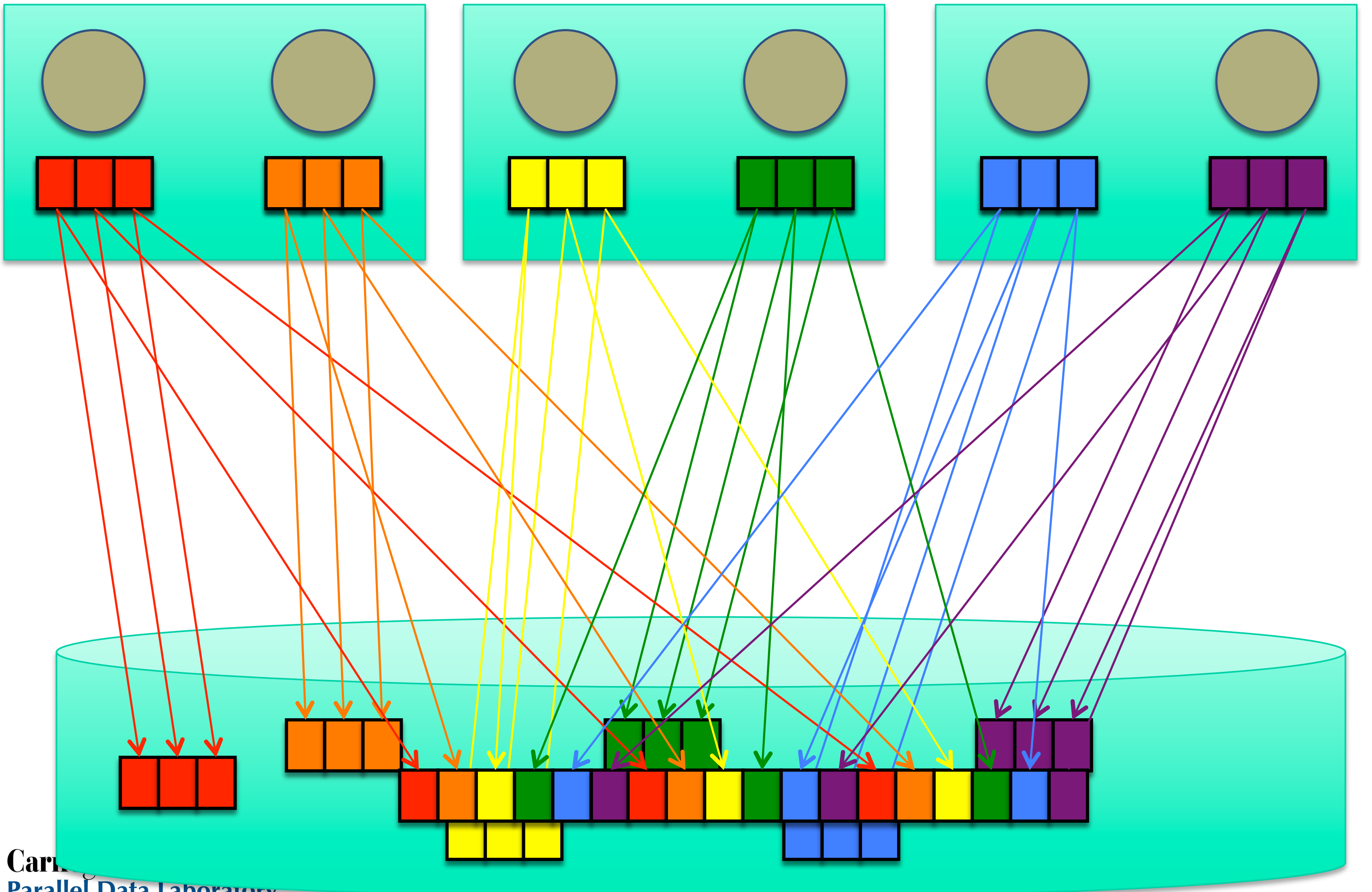
Professor, School of Computer Science, Carnegie Mellon University
Co-Founder and Chief Scientist, Panasas Inc.

Carnegie Mellon
Parallel Data Laboratory

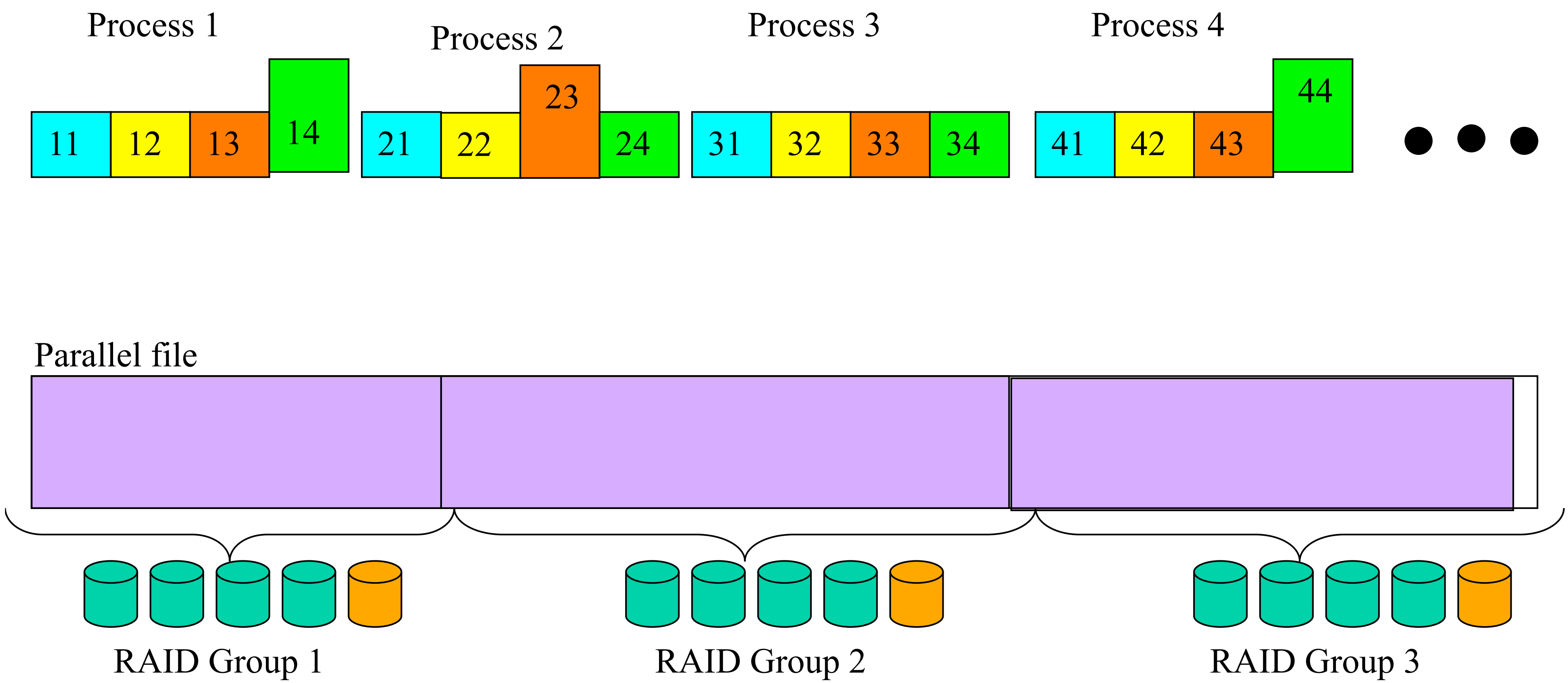
Agenda

- Simple File Systems Don't Do Write Sharing
- HPC Checkpointing: N-1 versus N-N concurrent write
  - N-1 has usability advantages & performance challenges
- PLFS: Parallel Log-structured File System
  - Library represents file as many logs of written values
  - In production at Los Alamos showing good benefits for important apps, brilliant benefits for benchmarks
- Eliminates write size & alignment problems
- Read performance doesn't suffer as expected
  - Index importing needs parallel impl.
- Backend abstracted to enable use of unusual storage, such as HPC on Hadoop HDFS

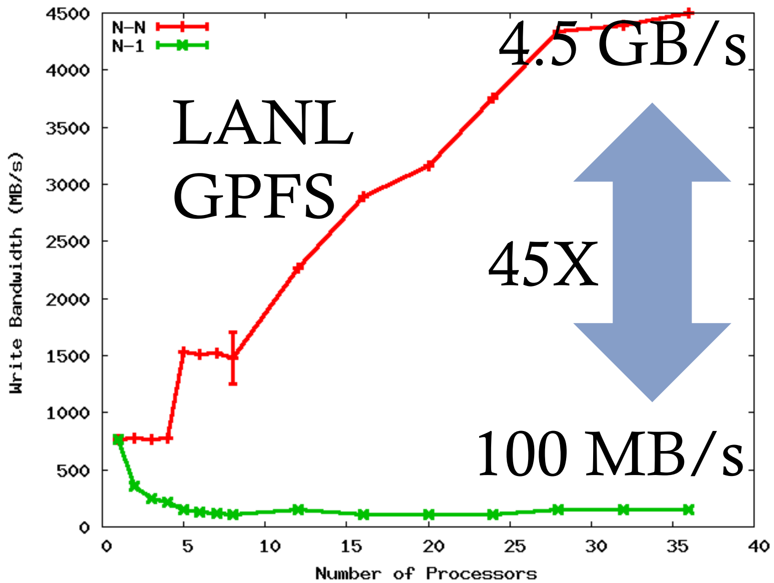
N-N File IO

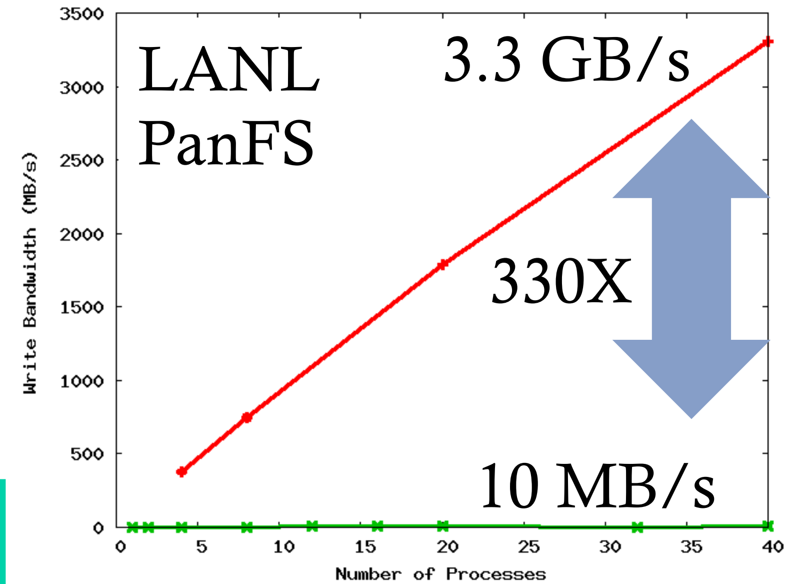


File Systems full of Locks for Consistency

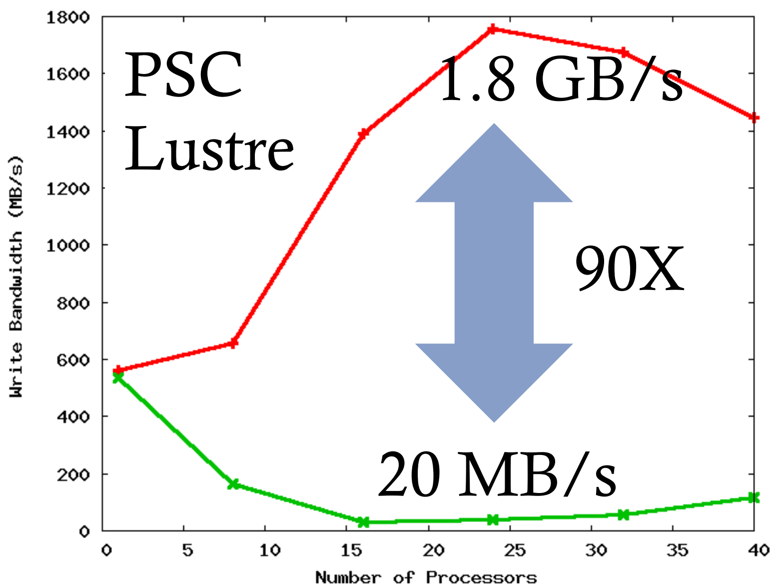


N-1 Concurrent Write Often Not Scalable





N-N
N-1



Cross graph comparisons not meaningful

N-1 versus N-N Checkpointing

- N-N writing easier for lock-happy file systems
- But many users prefer N-1 checkpoints
  - Prefer to manage 1 file, rather than thousands+
  - Can't avoid mapping because of N-M restarts
  - >50% LANL cycles use N-1 checkpointing
  - 2 of 8 open science apps written for Roadrunner
  - At least 8 of 23 parallel IO benchmarks including BTIO, FLASH IO, Chombo, QCD
- Some programmers switch but many don't
  - One app wrote 10K lines of code (bulkio) to "fix"

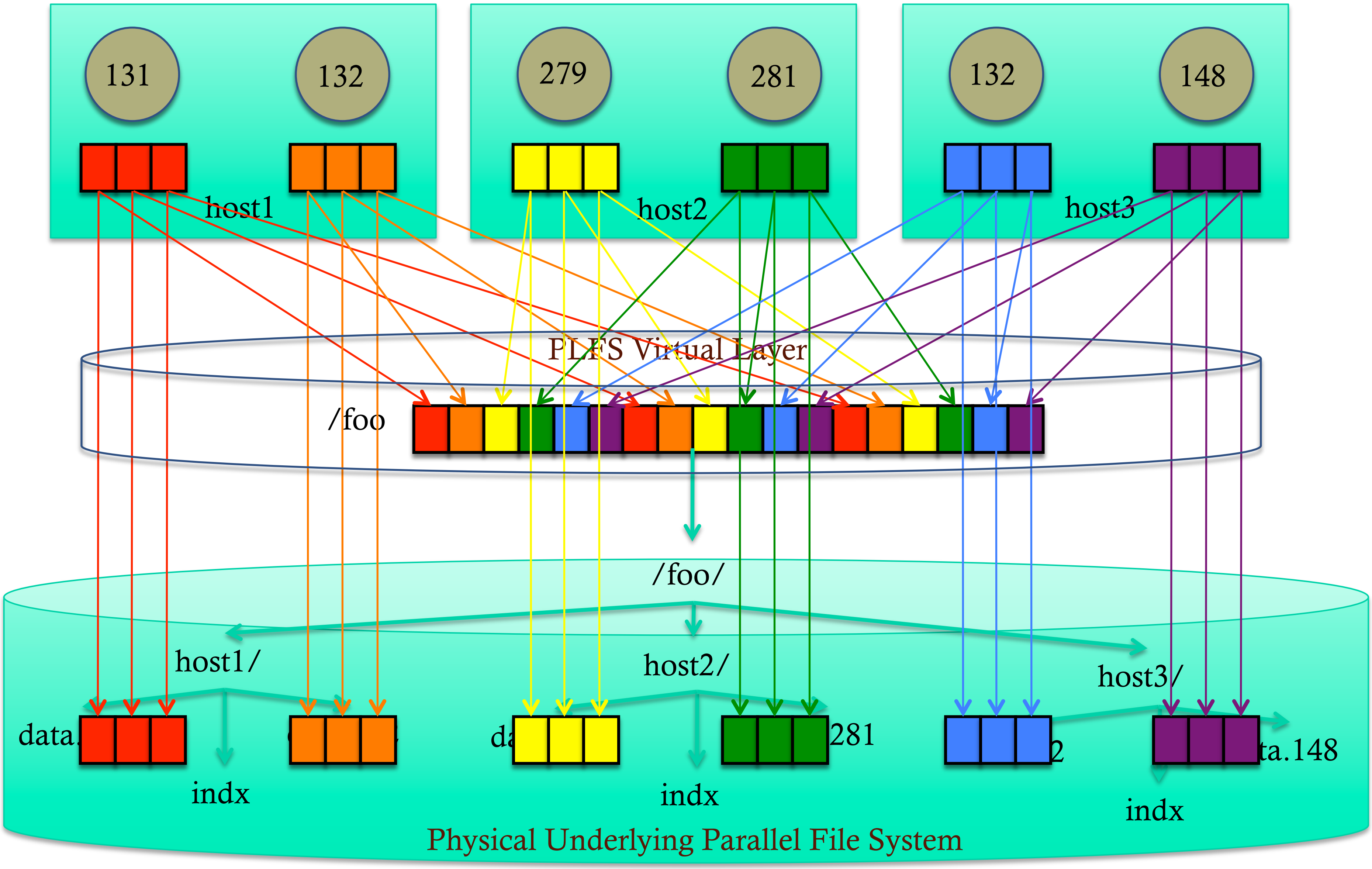
N-1 Write-Optimization via Log-Structuring

- 1991 LFS paper "write optimized" seeks during writes (instead of reads)
- Multiple projects emphasized eliminating seeks for checkpoint capture
  - PSC Zest "write where the head is" checkpointFS
  - ADIOS file formatting library uses delayed-write
  - PVFS experiments embedded log ordering
- In retrospect, log structured writing not as important as decoupling file system locks

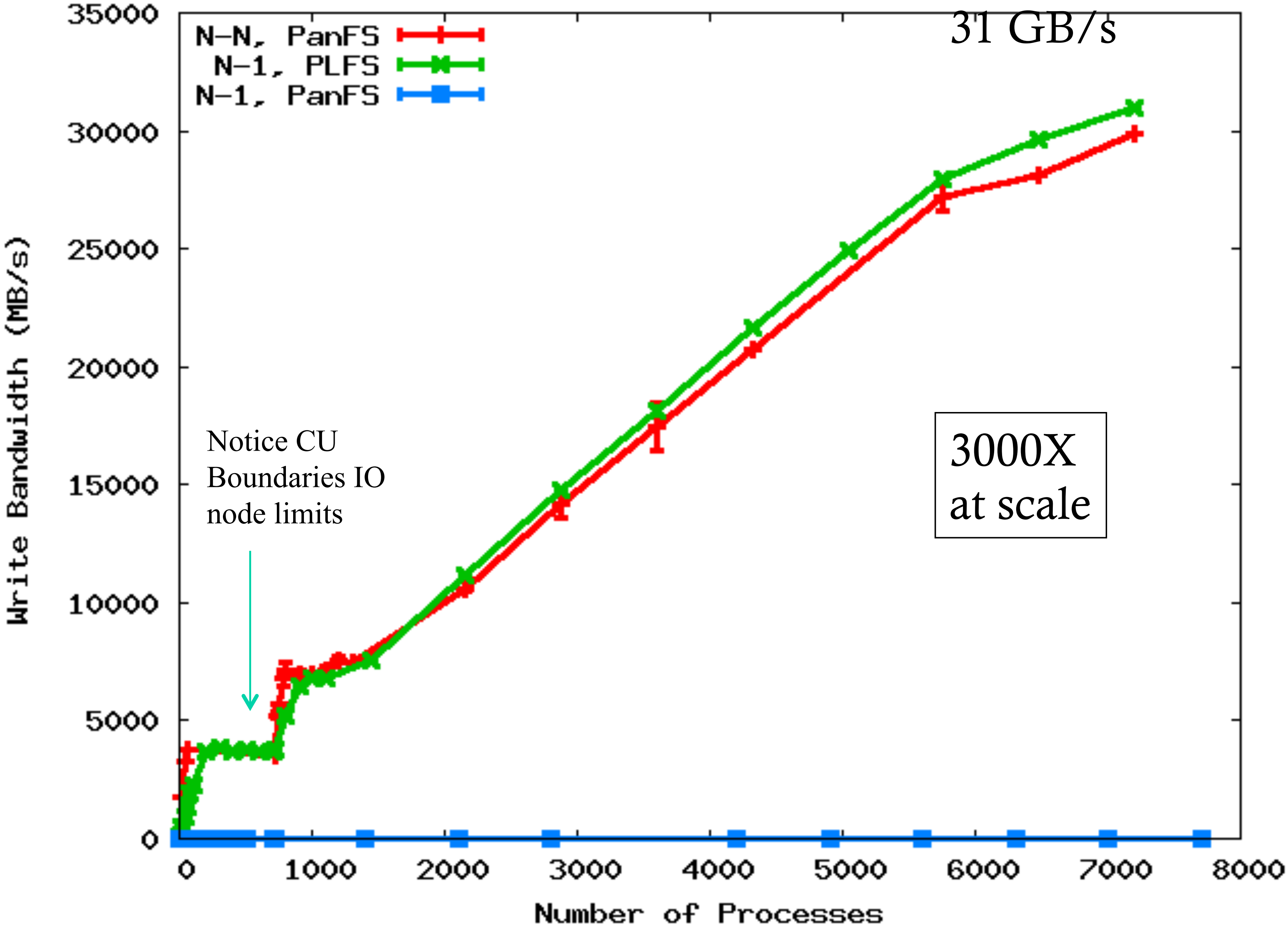
Parallel Log-structured File System

- Open source library for Fuse, MPI-IO
  - http://github.com/plfs (released v2.4 this week)
  - Part of FAST FORWARD plan for Exascale HPC
- Big team centered on Los Alamos Nat. Lab.
  - Gary Grider, Aaron Torres, Brett Kettering, Alfred Torrez, David Shrader, David Bonnie, John Bent, Sorin Faibish, Percy Tzelnic, Uday Gupta, William Tucker, Jun He, Carlos Maltzahn, Chuck Cranor
- This talk draws heavily on LA-UR-11-11964, SC09, PDSW09, Cluster12, CMU-PDL-12-115
  - Jun He in HPDC13 today (index management)
  - Other papers in PDSW12, DISCS12, 2x MSST12

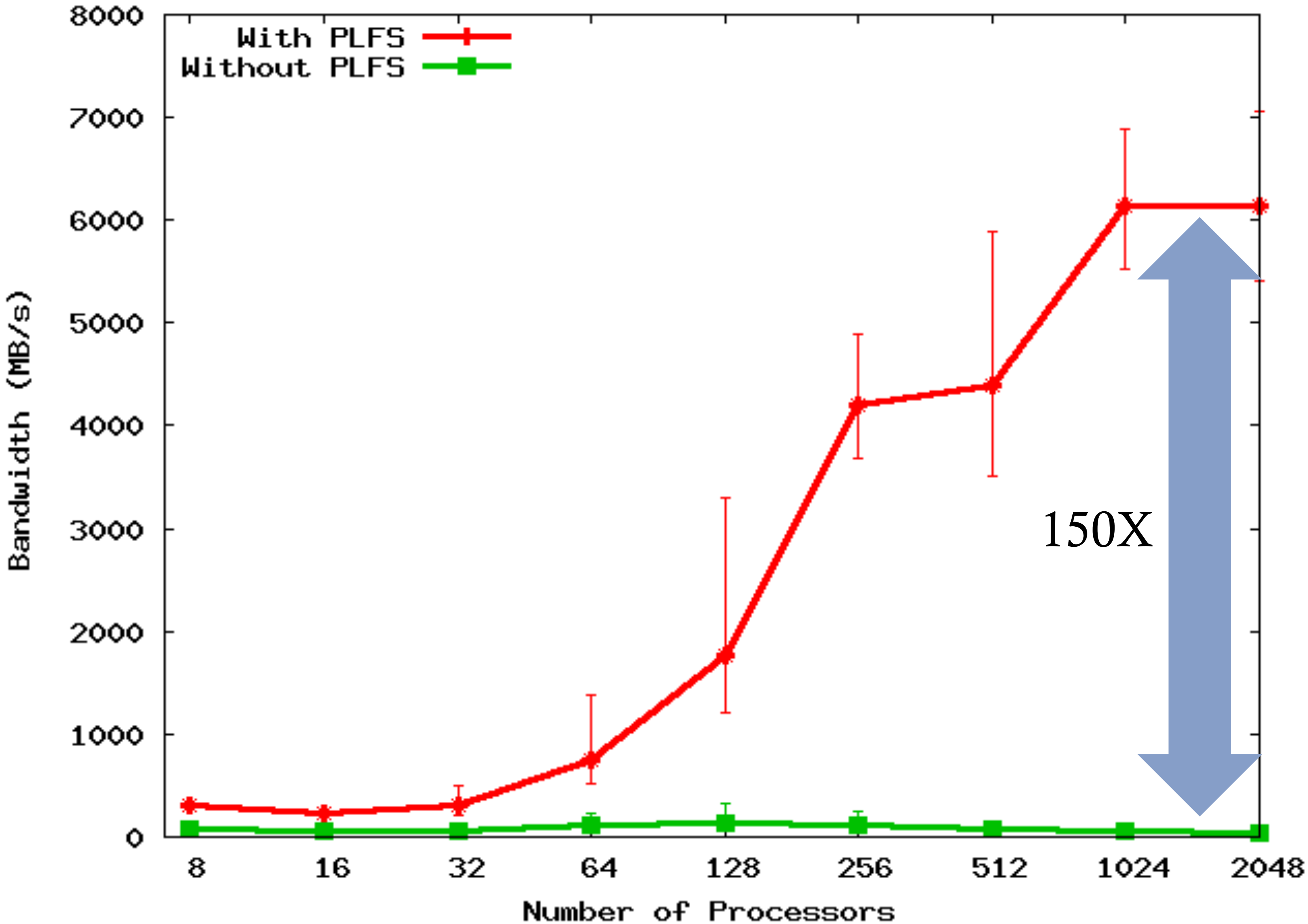
PLFS Decouples Logical from Physical



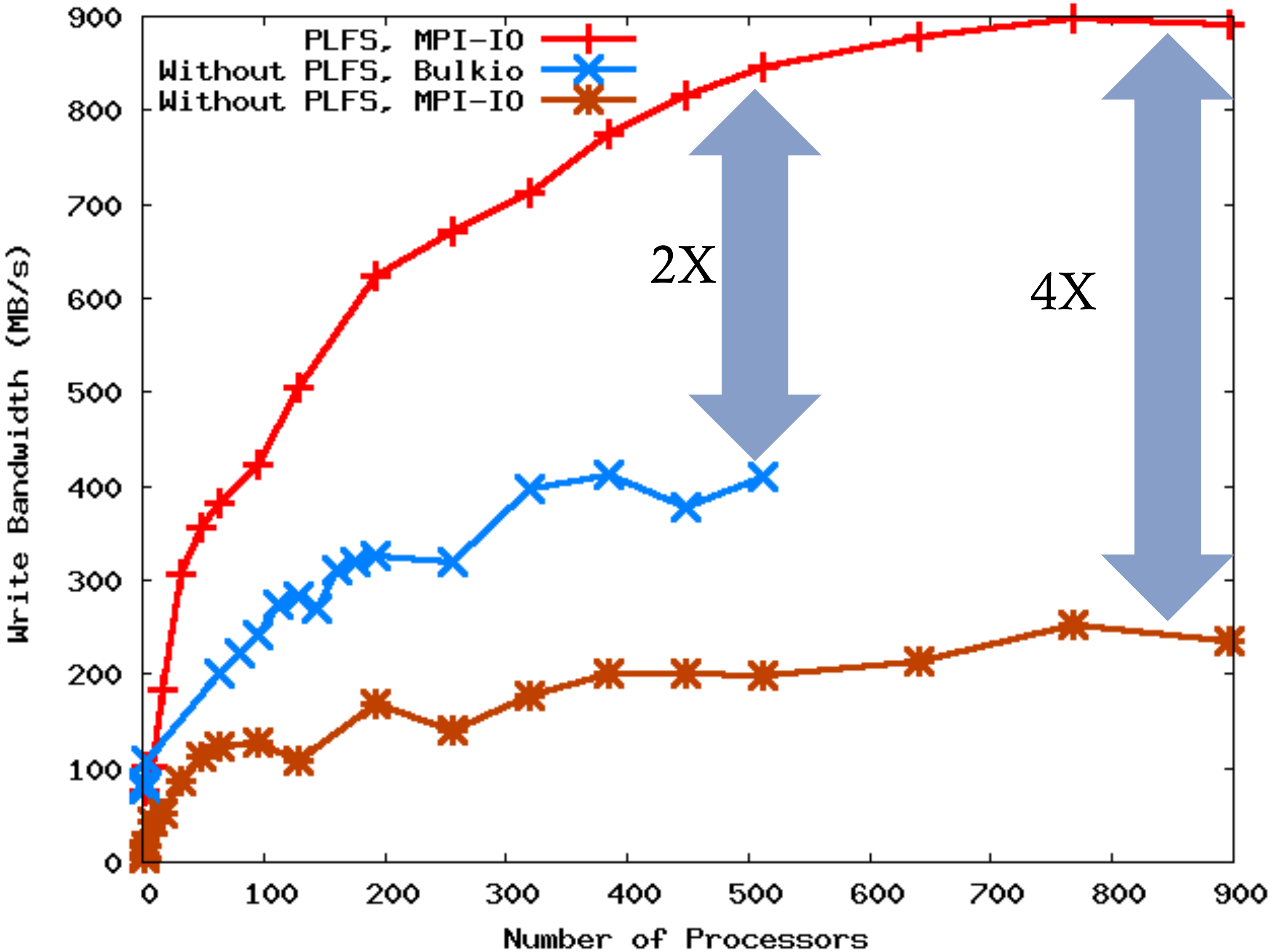
N-1 @ N-N BW in Simple Test



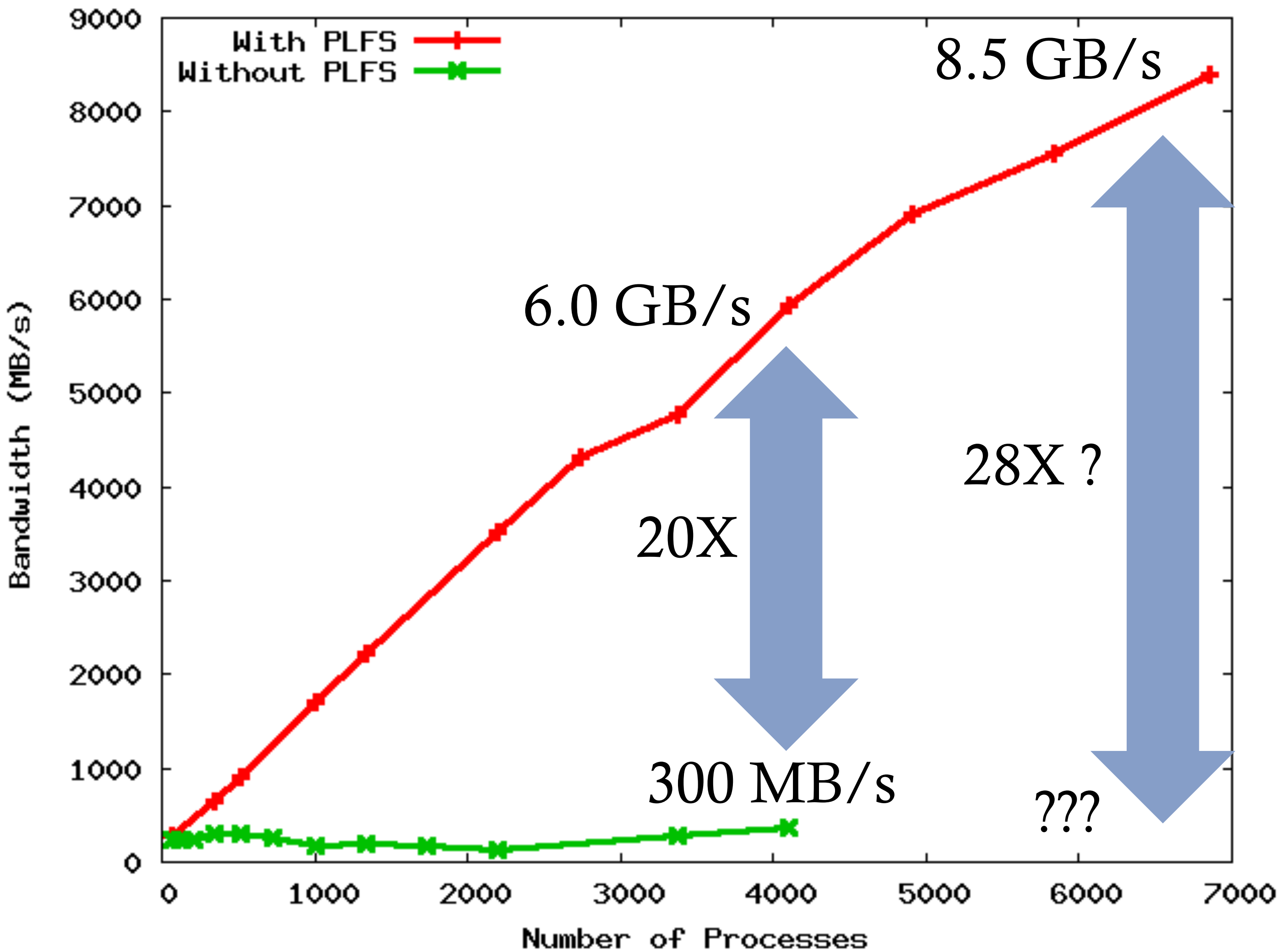
FLASH IO on Roadrunner



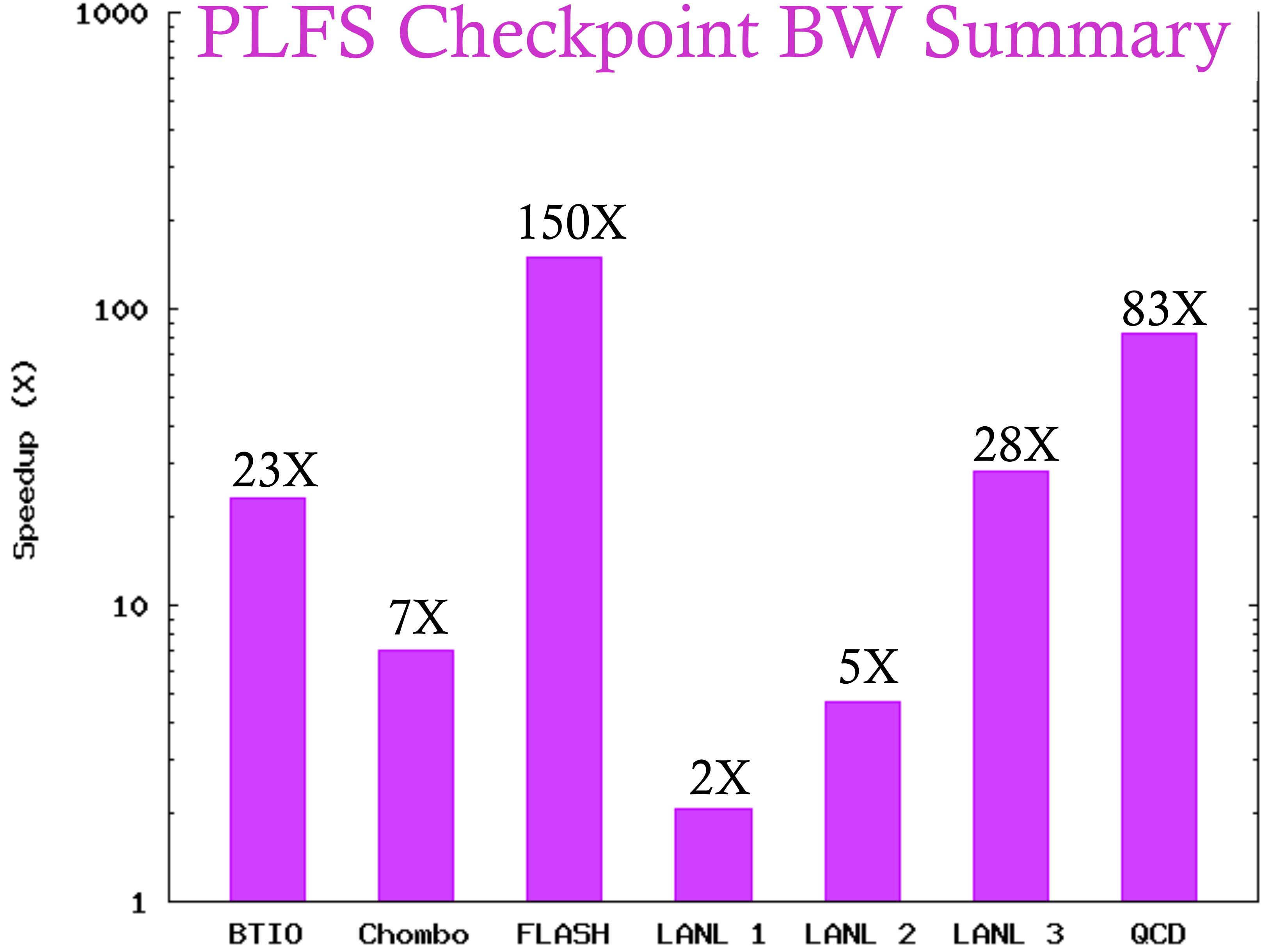
LANL1: 2X better than hand tuned library



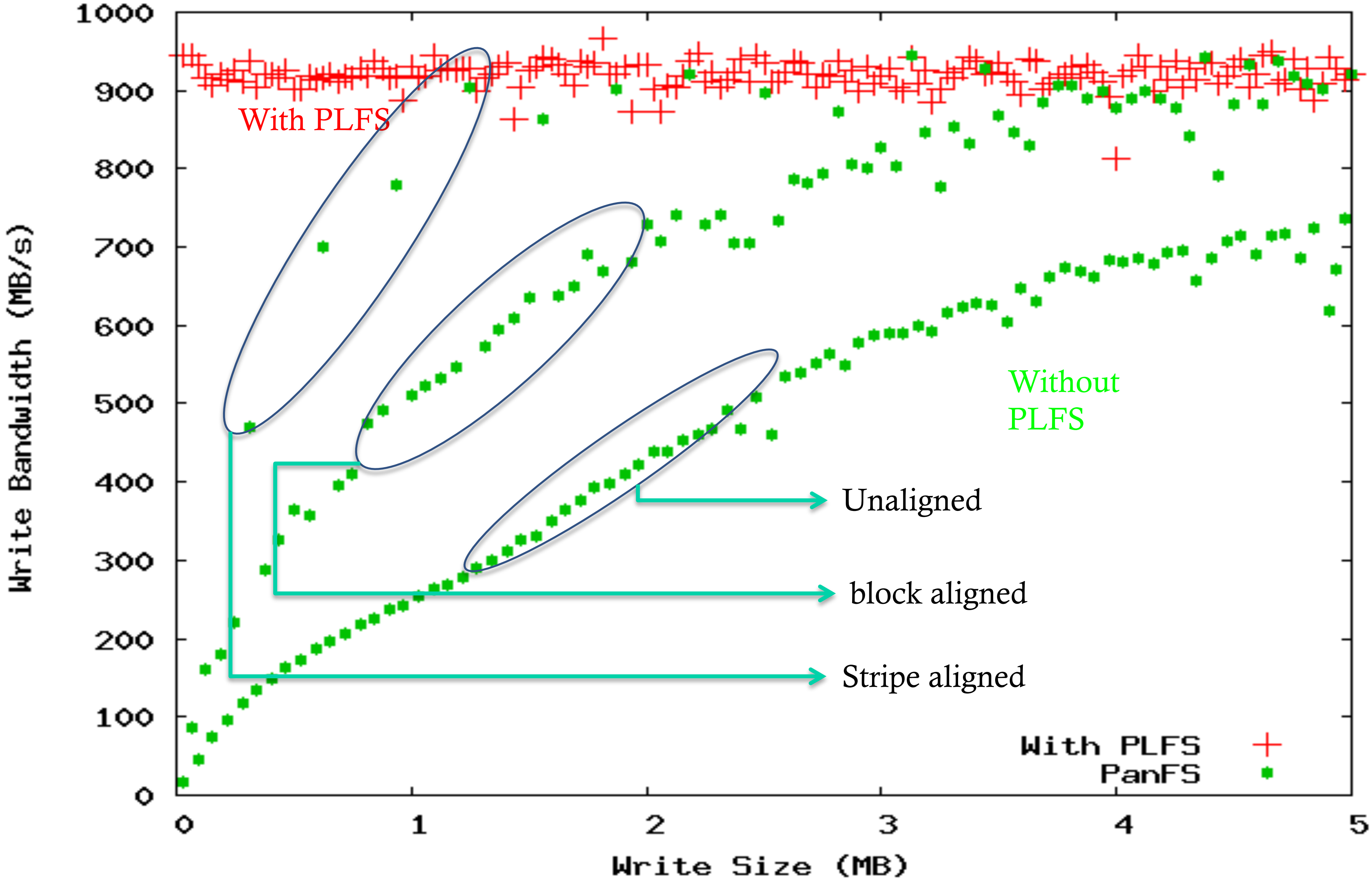
LANL3: More success in production



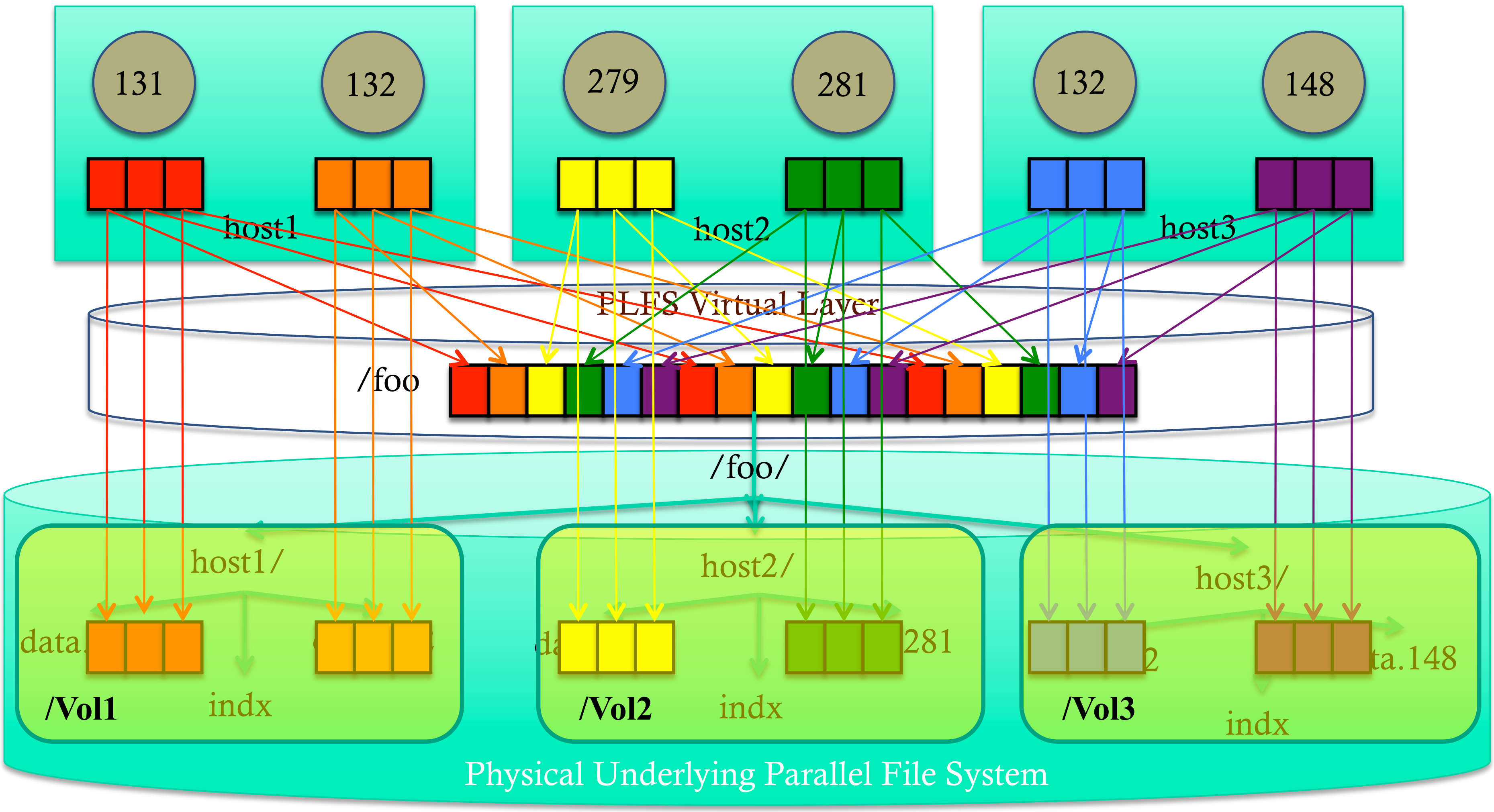
PLFS Checkpoint BW Summary

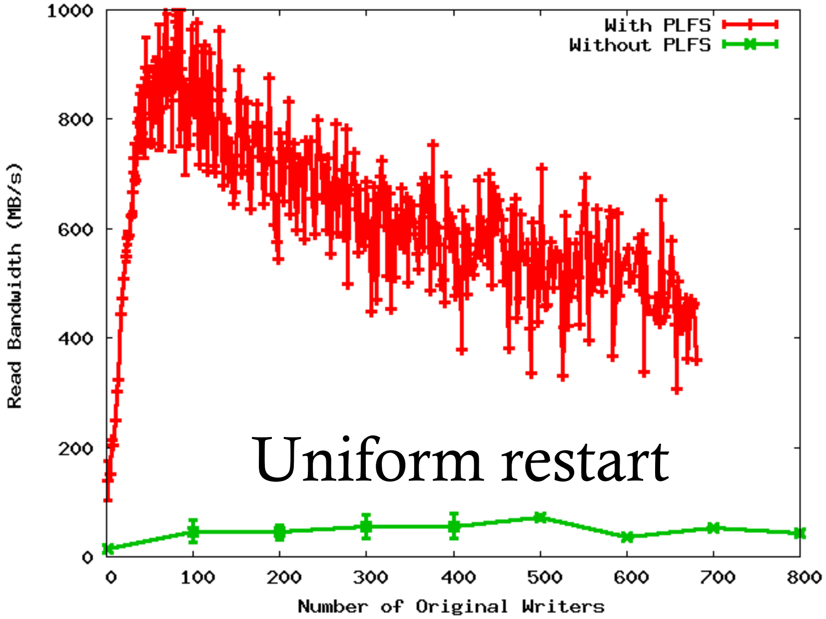


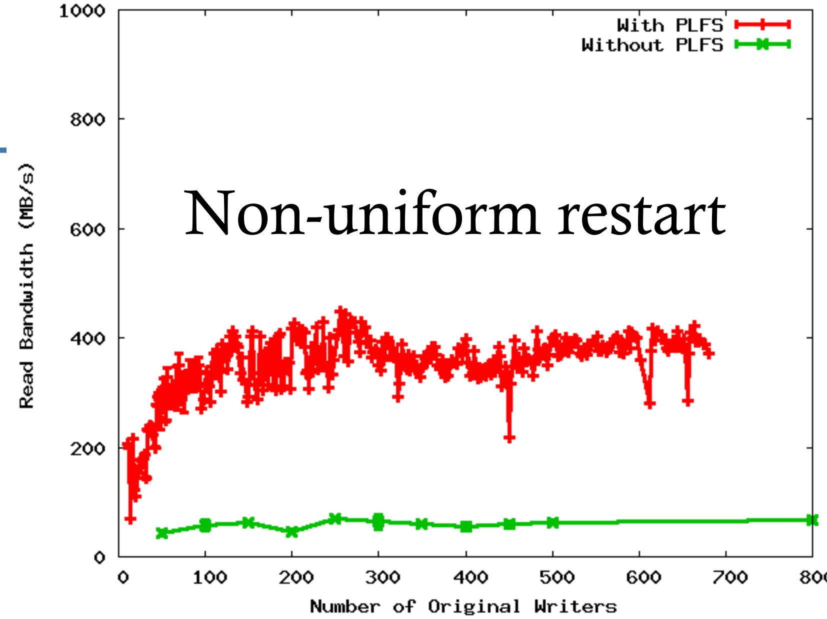
Does not Suffer "Alignment" Preferences



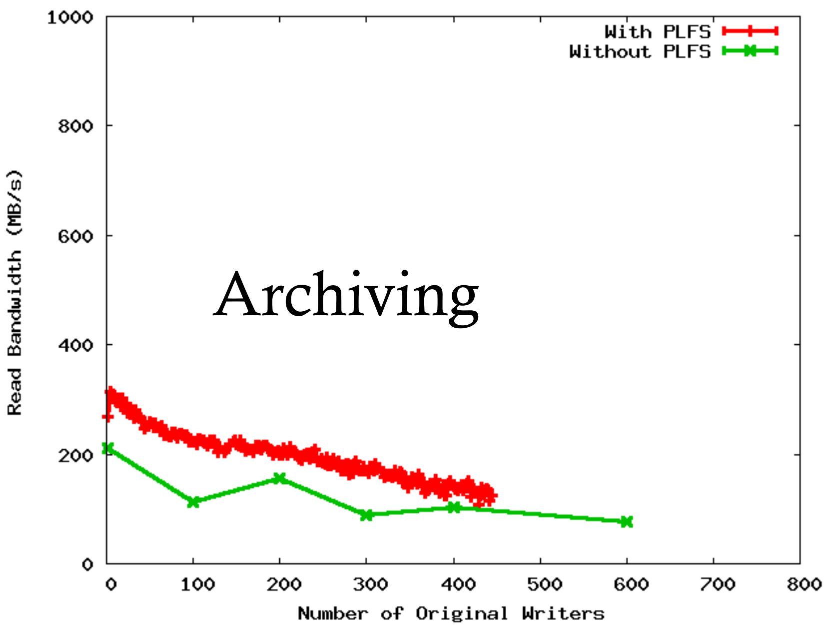
Distribute over Federated Backends





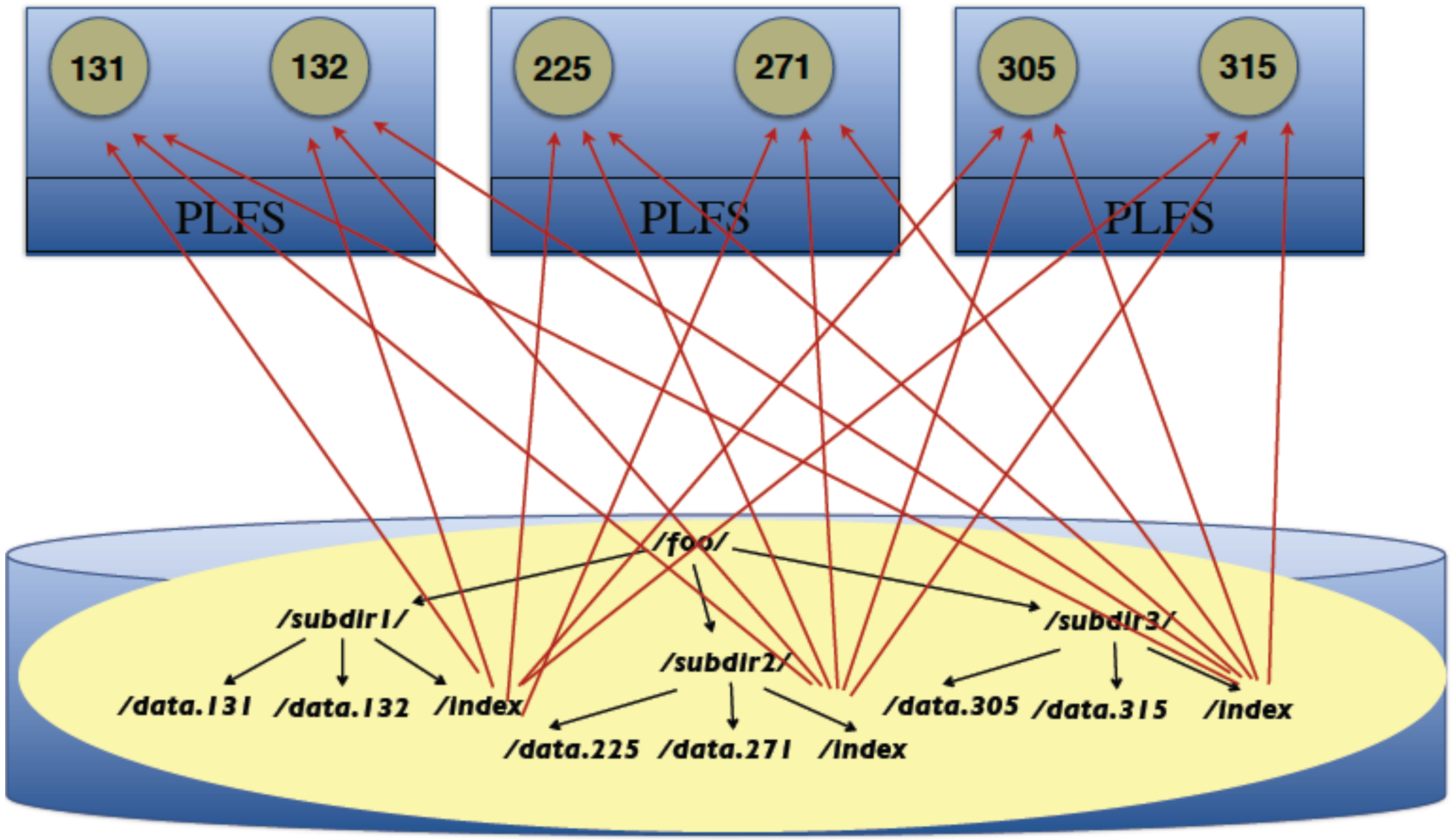


Read Bandwidth

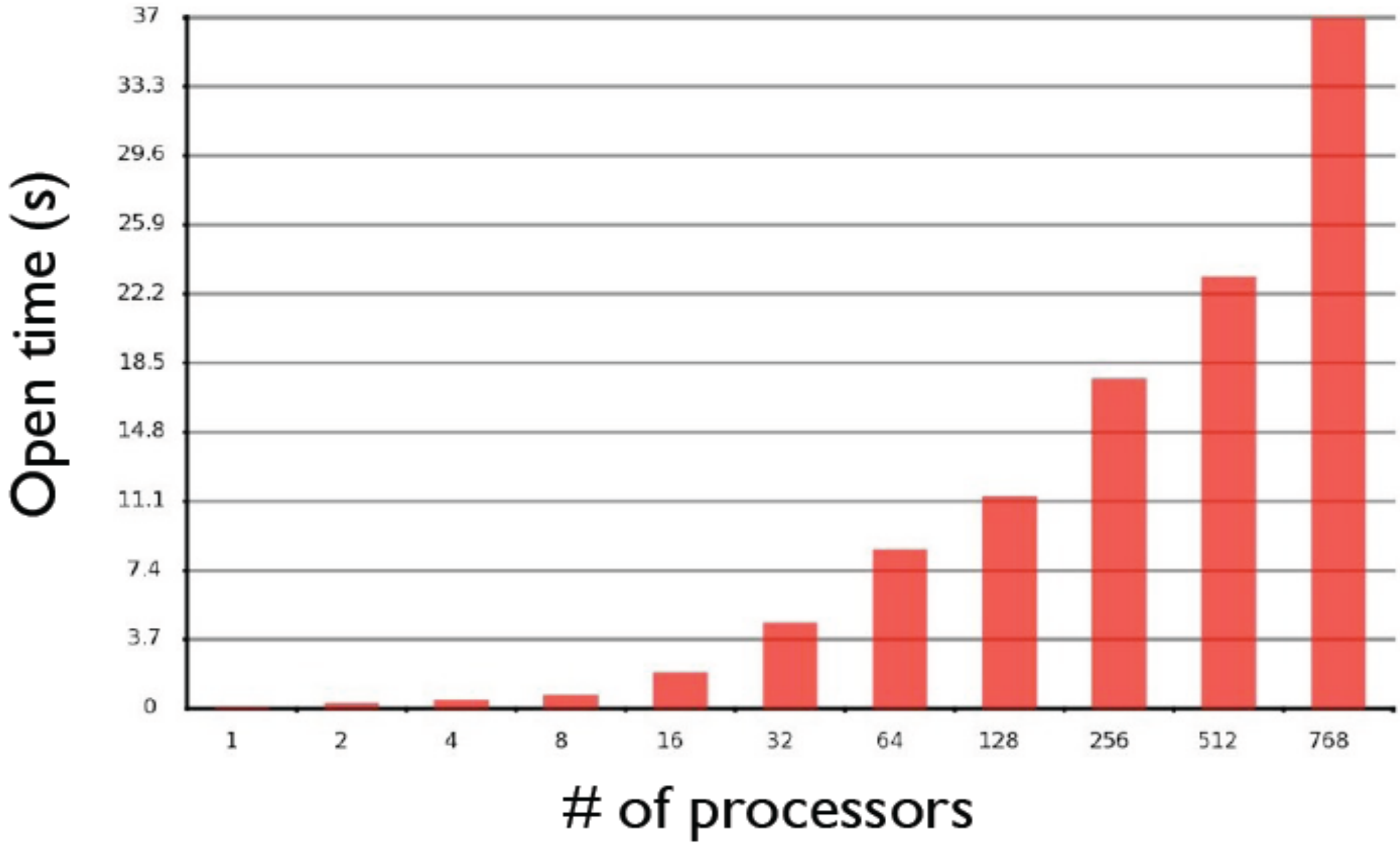


- Wrote increasing addresses
- Reads increasing addresses
- Result is mergesort with deep prefetch (one per log)
- So write optimized is also read optimized !

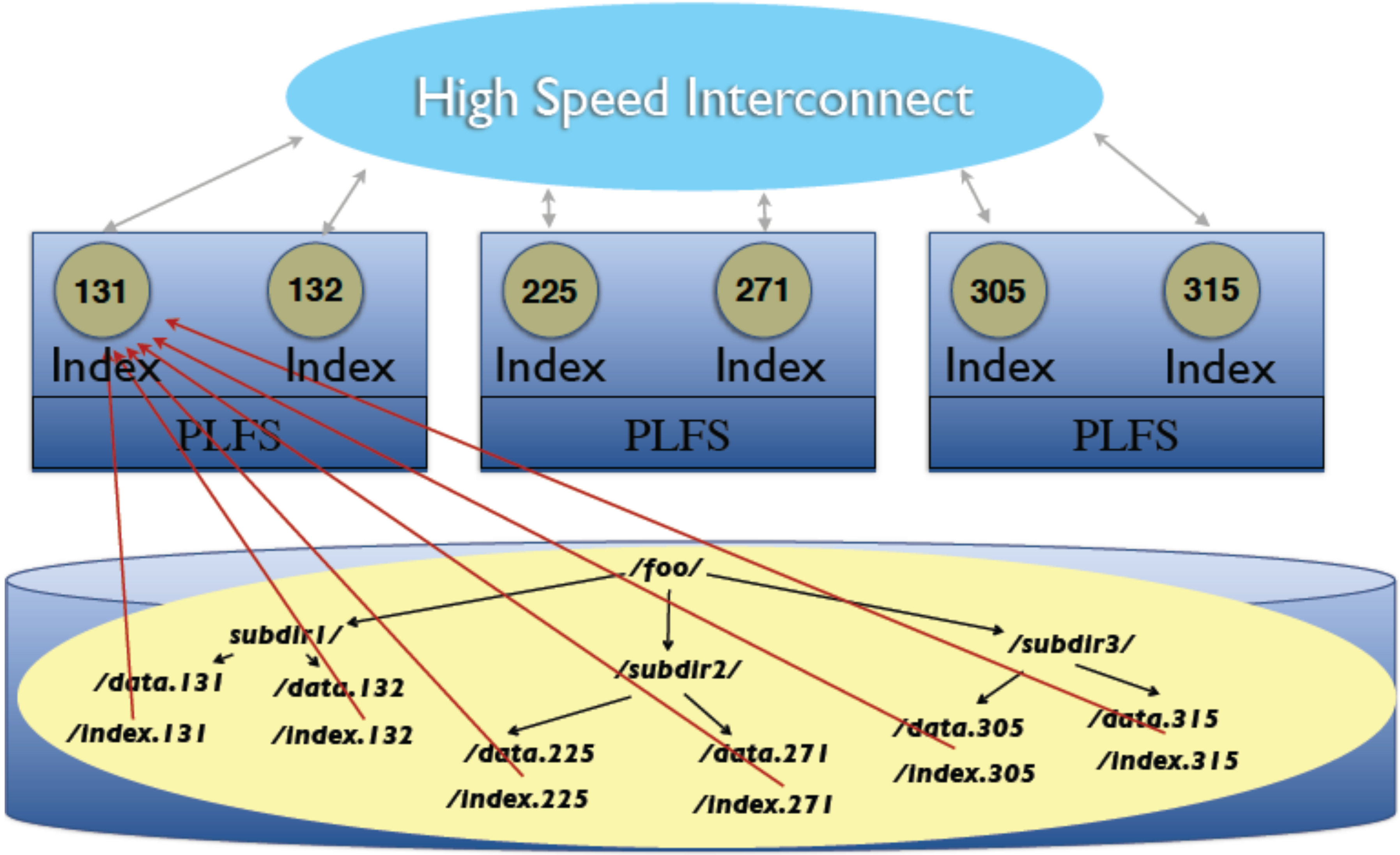
Open for Read does N Squared Index Reading



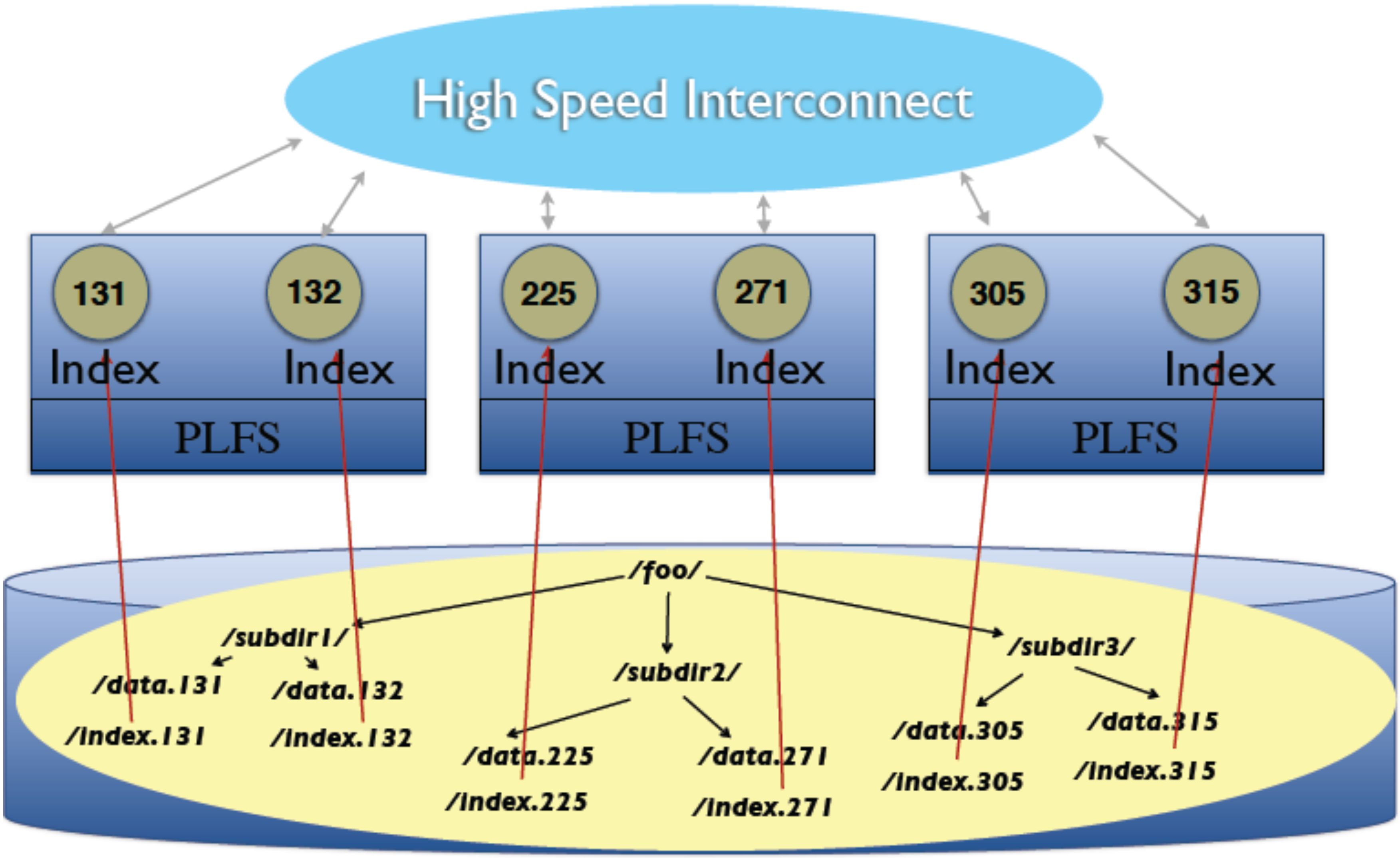
PLFS Open Times



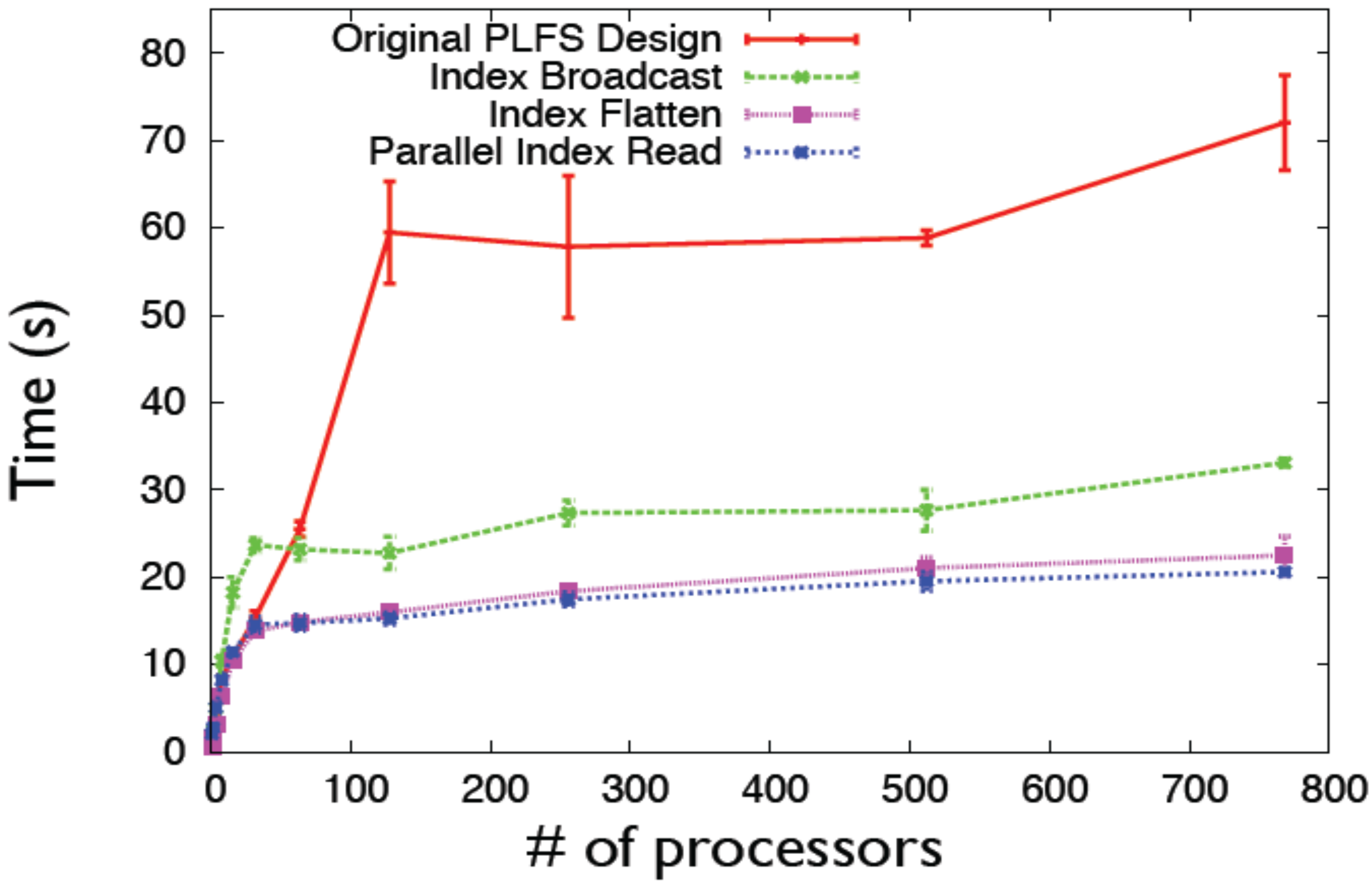
Index Broadcast



Parallel Index Read

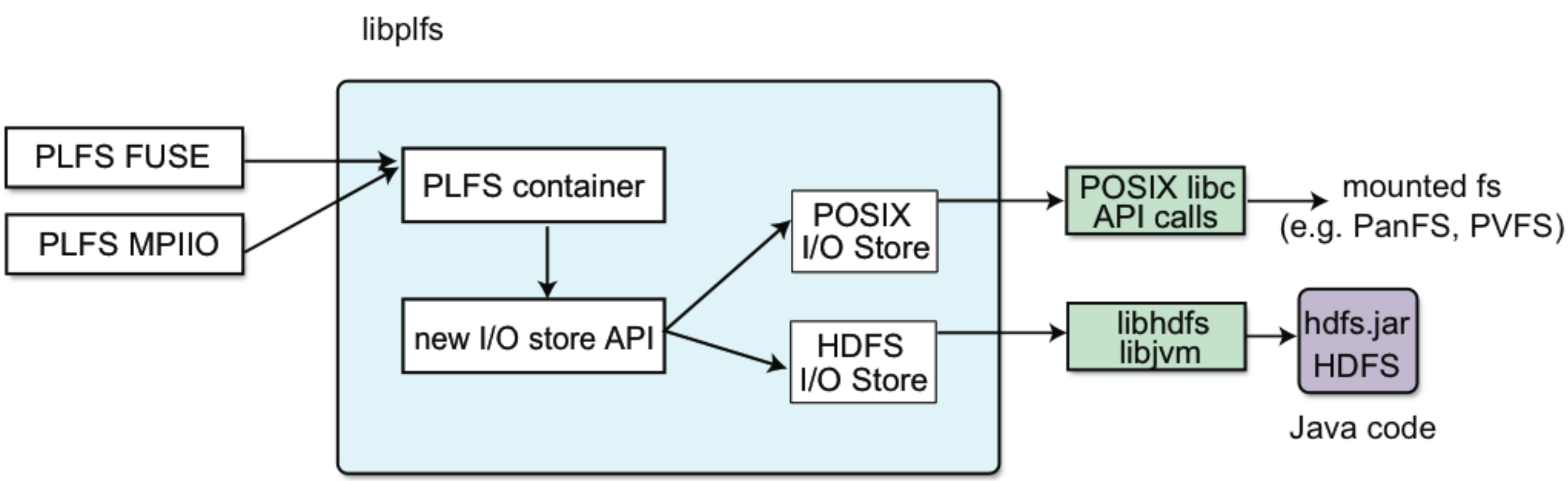


Read Open + Write Close Times



Abstracted Backend Storage

- PLFS uses abstracted backend storage
- E.g. HPC app w/ PLFS can run on a cloud with non-POSIX HDFS as native file system



PLFS Summary and Futures

- N-1 checkpoints feature intensive concurrent writing
- File systems choke on consistency preserving locks
- PLFS decouples concurrency with per-processor logs
  - Order of magnitude and larger wins for taking checkpoint
  - Insensitive to ideal write sizes or alignments
- Typical reading also faster because it mergesorts logs
  - Index construction on read can be parallelized
- Ongoing work with PLFS
  - N-N checkpointing benefits from hashing logs over federated backend storage systems – easy way to scalable metadata thruput
  - Burst buffers using NAND Flash for faster checkpoint can be managed by PLFS
  - In-burst-buffer local processing needs exposed structure