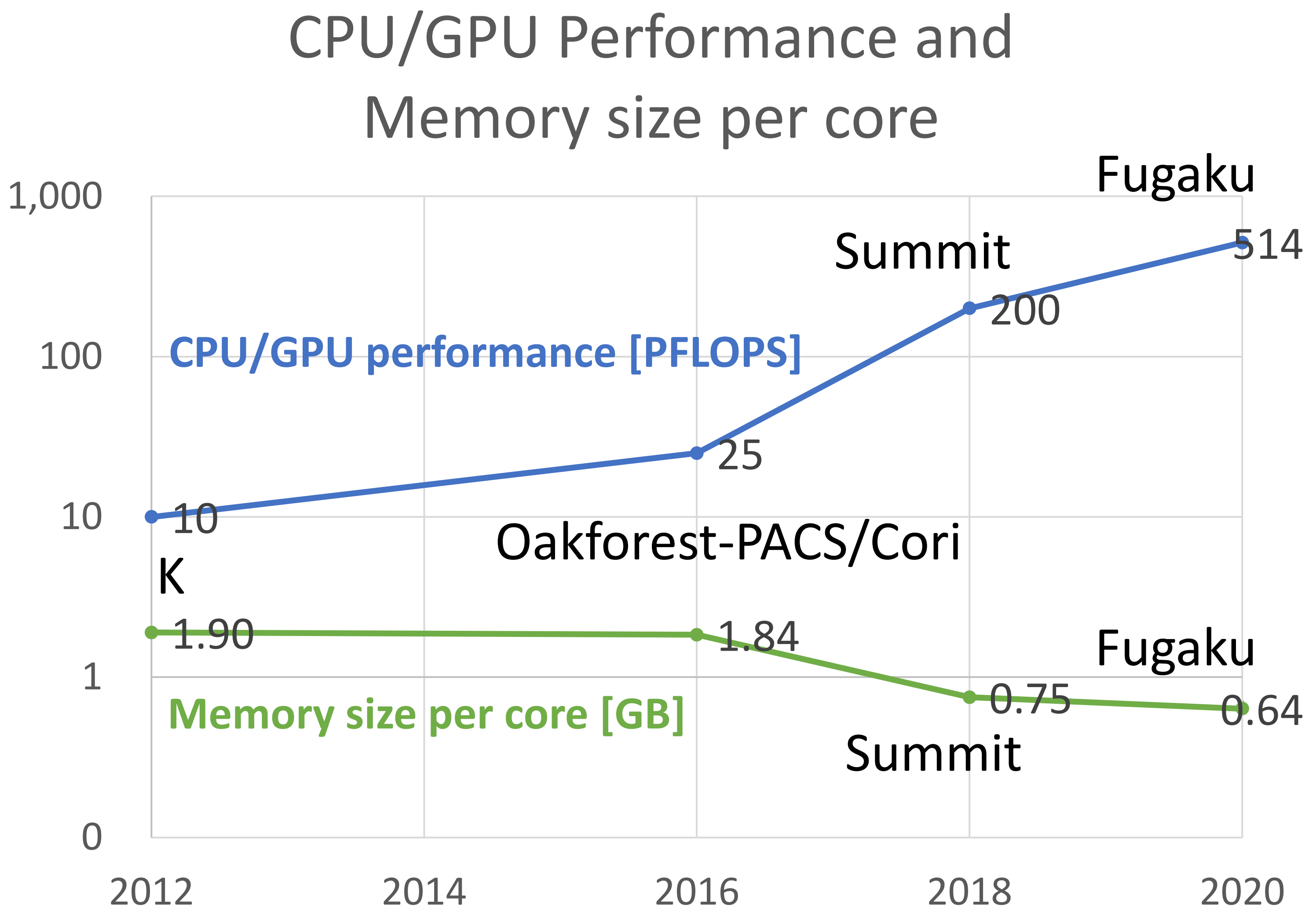
Persistent Memory Supercomputer Pegasus for data-driven and AI-driven Science

Osamu Tatebe

Center for Computational Sciences, University of Tsukuba

Pegasus background

- CPU performance 50x, but memory size 3.8x in 8 years
- It matters for Data-driven and AI-driven Science
  - Memory size and Storage performance are really important
- Introduce Persistent Memory
  - Memory mode for memory size and direct mode for storage performance

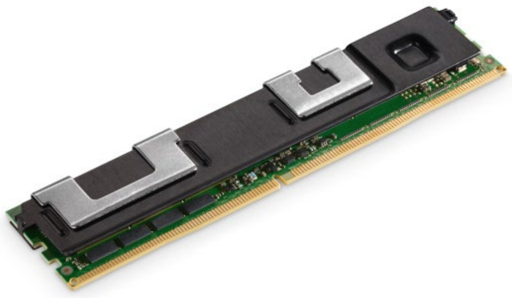


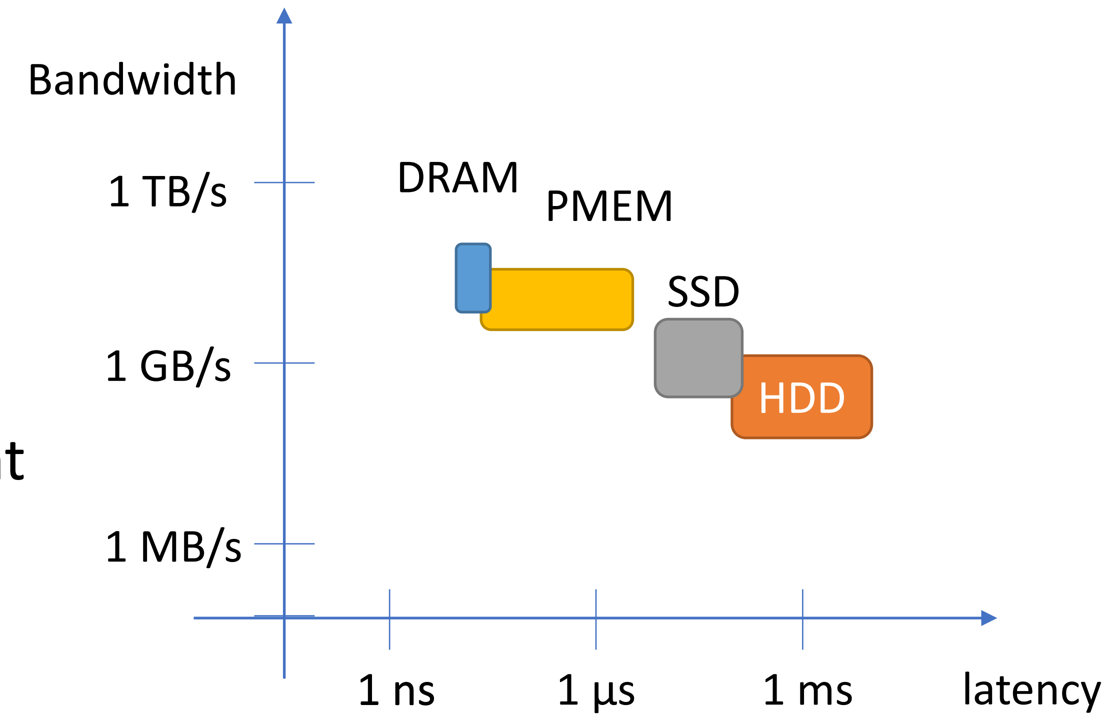
Design Goal of Pegasus

- Accelerates large-scale data analysis and big data AI by utilizing persistent memory for large memory space and high performance storage
- Fosters new fields of large-scale data analysis, new applications of big data AI, and system software research

Persistent Memory

- One order better cost performance
- Minimum latency is ~60 ns (similar to DRAM)
- Half of bandwidth
- Memory mode
  - Larger memory space without much performance penalty
- App direct mode
  - Direct access to byte-addressable persistent memory and high-performance storage





Pegasus Highlights

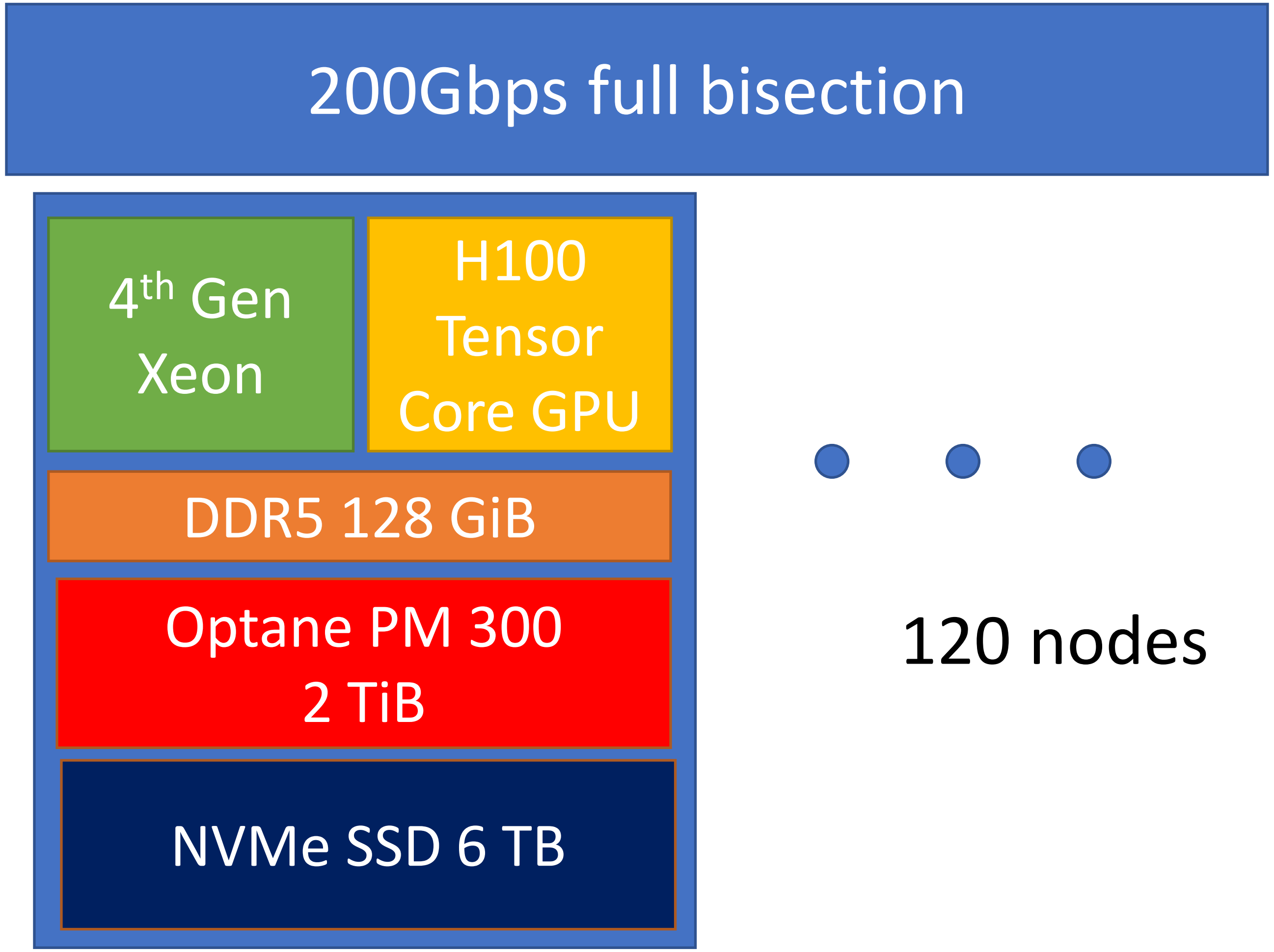
- Plans to build with Intel 4th Gen Xeon, NVIDIA H100 Tensor Core GPU with PCIe, and 2 TiB Intel Optane PM 300 series will strongly drive Big Data and AI
- The world's first system with NVIDIA H100 PCIe GPUs connected via PCIe Gen5
- First system announced in Japan that will utilize NVIDIA Quantum-2 InfiniBand networking

Pegasus Specification

NEC LX B1000E Blade Enclosure

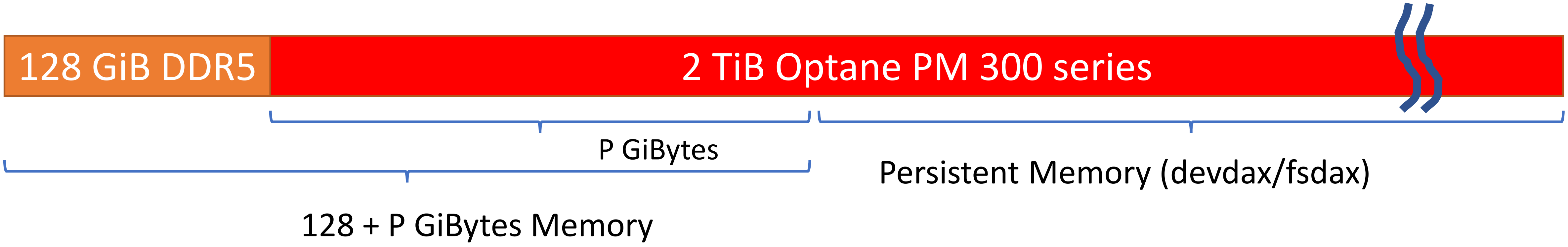
NEC LX 102Bk-6

- Will be installed in 4Q 2022
- Total Performance
  - 120 nodes, > 6.1 PFlops, 240 TiB Pmem
- Node specification
  - Intel 4th Gen Xeon Scalable Processor
  - 51 TFlops NVIDIA H100 Tensor Core GPU
  - 128 GiB DDR5 DRAM
  - 2 TiB Optane PM 300 series (16x DRAM)
  - 6 TB NVMe SSD (7 GB/s)
- Interconnection Network
  - NVIDIA Quantum-2 InfiniBand platform (200 Gbps) full bisection
- Parallel File System
  - 7.1 PByte DDN EXAScaler (40 GB/s)



How to use Persistent Memory

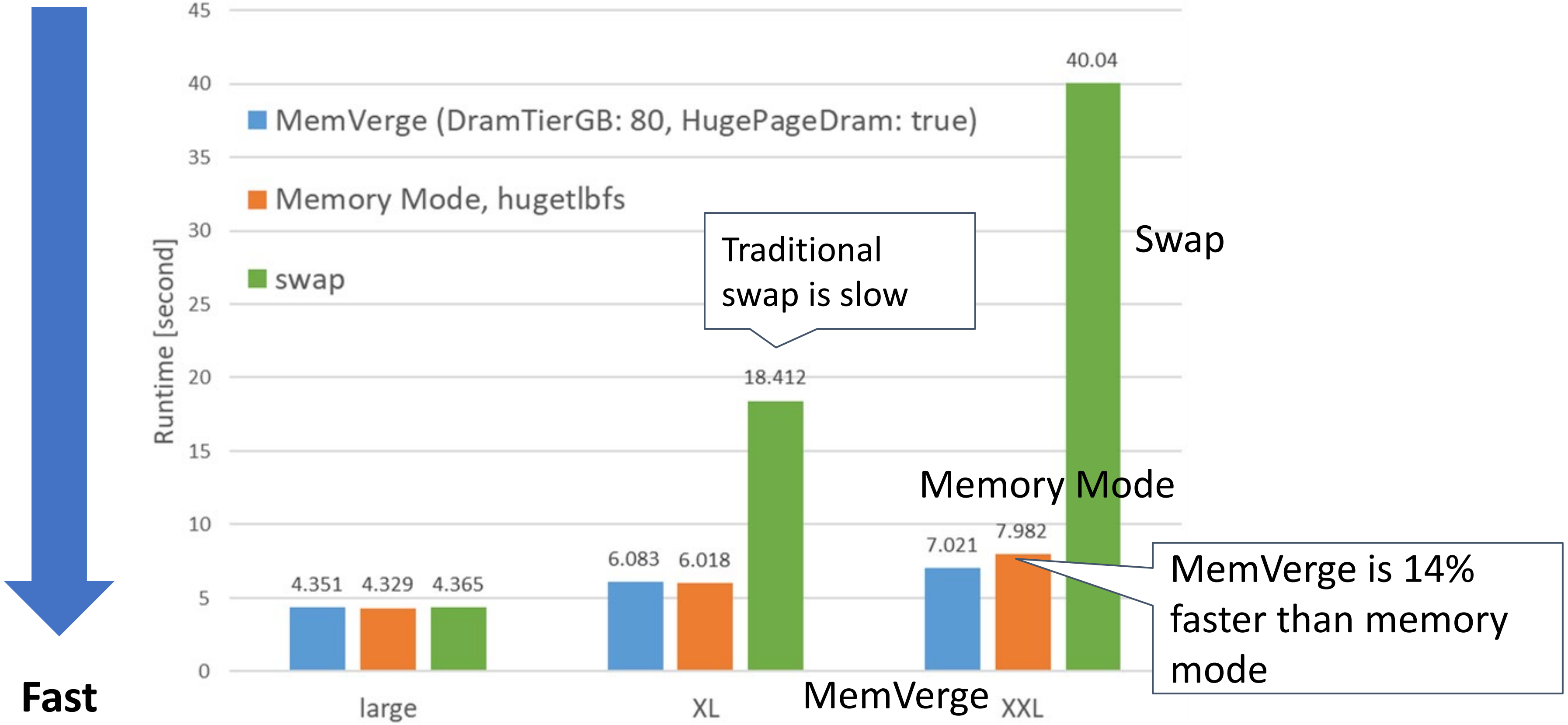
- All PM regions are configured in App direct mode
- Users can specify PM size as extended memory and as persistent memory



- MemVerge Memory Machine is used for memory extension

XSBench Application Benchmark (on Optane PM 100)

- Proxy application of the Monte Carlo neutron transport algorithm
- Memory usage: large 5.6 GB, XL 120GB, XXL 252GB
- Memory access pattern: random read

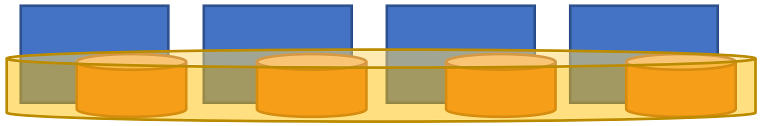


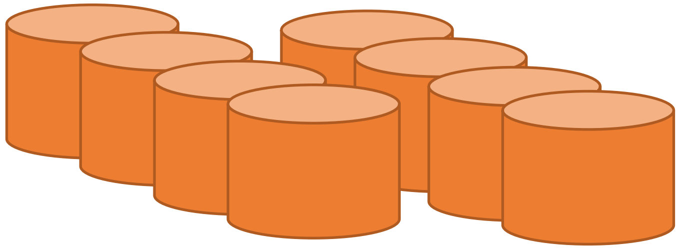
Comparison of Cygnus and Pegasus

	Cygnus (2019)	Pegasus (2022)
PFLOPS (DP)	2.3	> 6.1 (2.7x)
CPU	0.16	?.?? (?.?x)
GPU	2.18	6.12 (2.8x)
FPGA (SP)	0.64	0
Memory (TiB)	15.2	15.36 (1.01x)
Pmem (TiB)	0	240
Storage (PB)	2.4	7.1 (3.0x)

Research of Ad hoc parallel file system

- Temporal parallel file system using node-local storage
- Fill the performance gap between CPU/GPU and storage





- We are developing CHFS ad hoc file system to utilize persistent memory
  - No metadata server, no sequential processing for performance and scalability

Design Goal of CHFS [HPC Asia 2022]

- Utilize persistent memory performance
  - In-memory persistent key-value store (not block-based file system)
- Reduce metadata overhead and achieve scalable performance improvement
  - No dedicated metadata server (no additional lookup for metadata)
  - no sequential processing and no central data structure
- Improve single-shared-file performance
  - File is divided into fixed-size chunks to distribute a single file among servers and to avoid lock contentions
- Based on highly parallel distributed key-value store without any central data structure

Design of File System

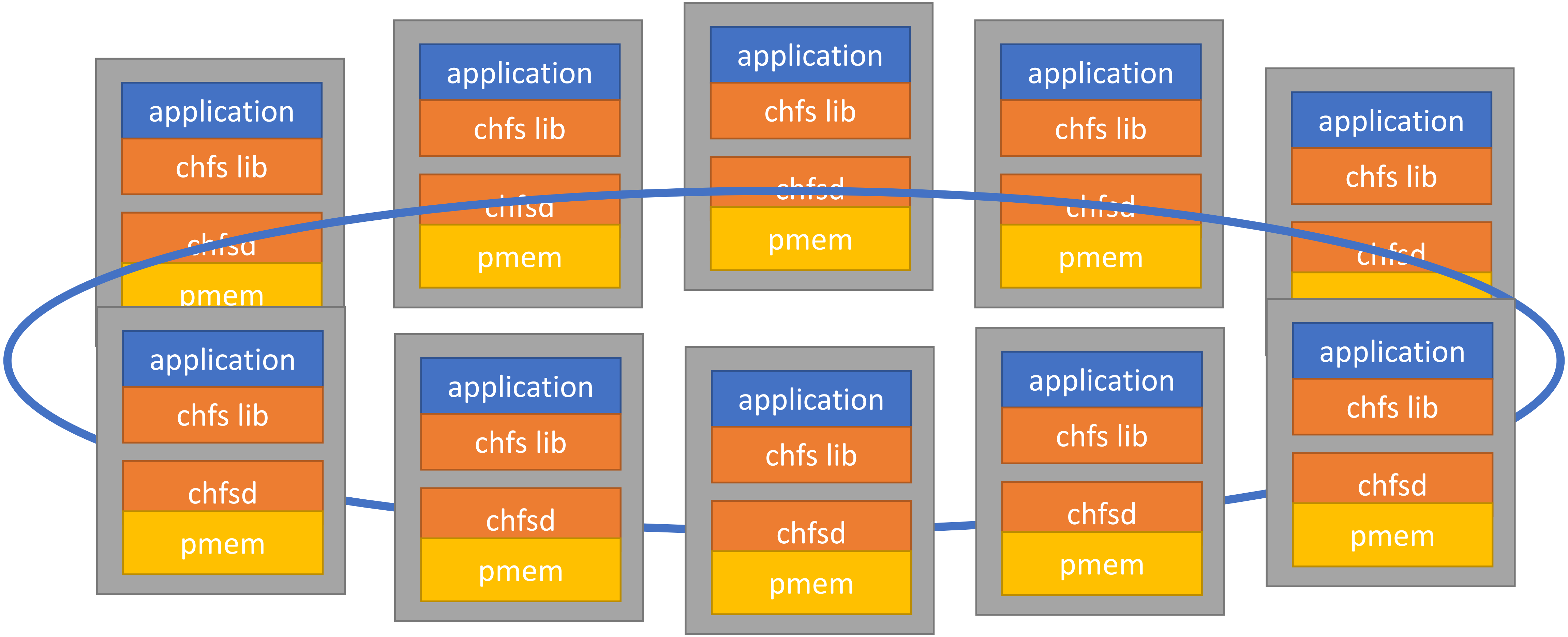
- All data is stored in a highly parallel distributed KV store
- A single Key-Value format in CHFS for a file chunk and a directory

Key	Value	
Full path and chunk number	Metadata	File data

Metadata (64 bytes)
mode, uid, gid, size chunk size mtime, ctime

- For a directory, there is no chunk number and no file data
- No file-level information in KV store, such as total chunk numbers and total file size to avoid sequential processing

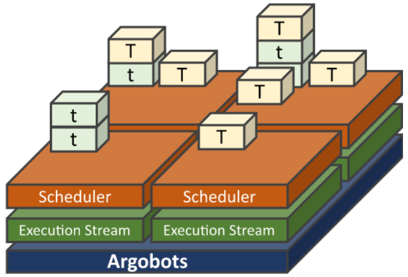
System Architecture of CHFS



Compute nodes

Implementation of CHFS

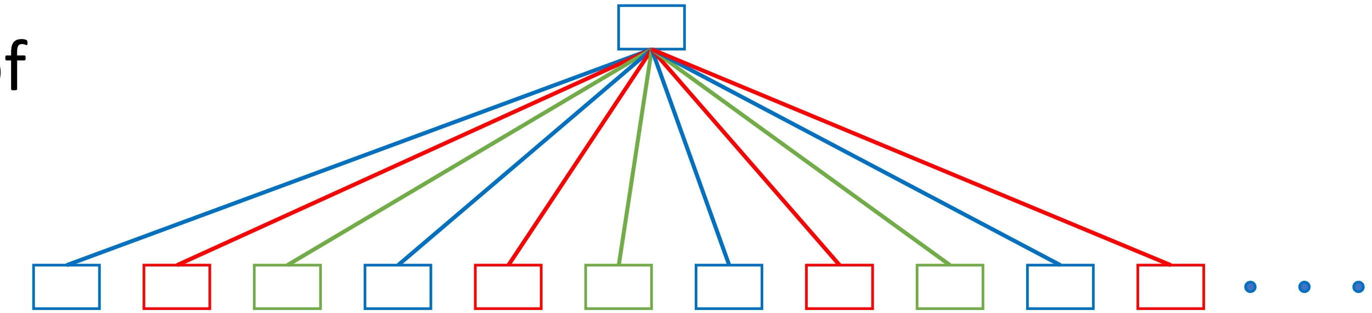
- Mochi-Margo [JCST 2020]
  - https://mochi.readthedocs.io/en/latest/
  - Communication library using Mercury and Argobots
- Mercury [Cluster 2013]
  - Async RPC, RDMA communication library
  - libfabric, UCX, shared memory plug-ins
- Argobots [IEEE TPDS 2018]
  - Light-weight thread library
- Pmemkv – persistent key-value store in PMDK



Performance Evaluation of CHFS

- IO500 benchmark
  - BW: IOR easy/hard – 4 benchmarks
  - MD: MDtest easy/hard, Find – 8 benchmarks
- Score is the geometric mean of benchmark results
- 4-node Pmem cluster and 78-node Cygnus

MDtest hard

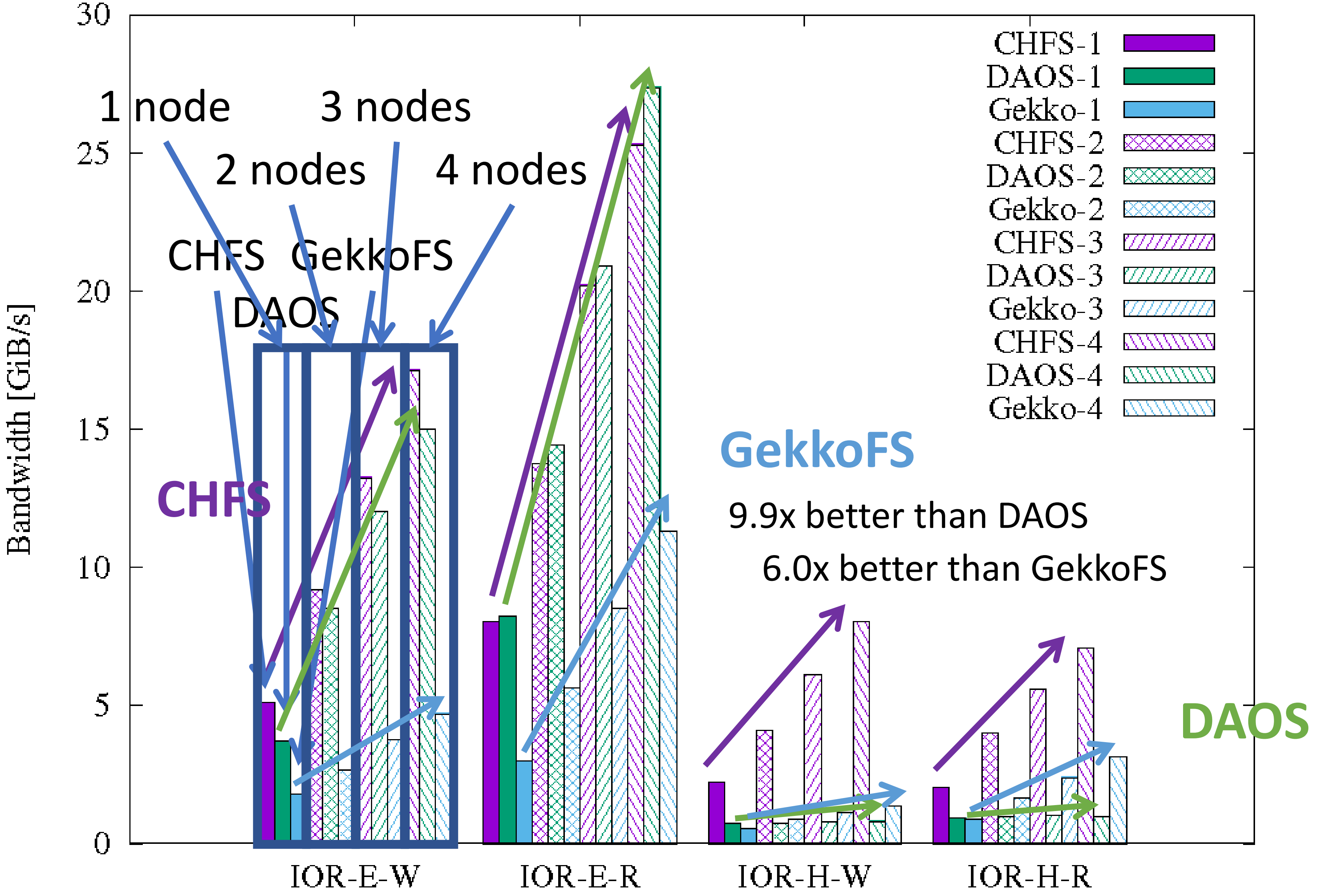


3,901 B

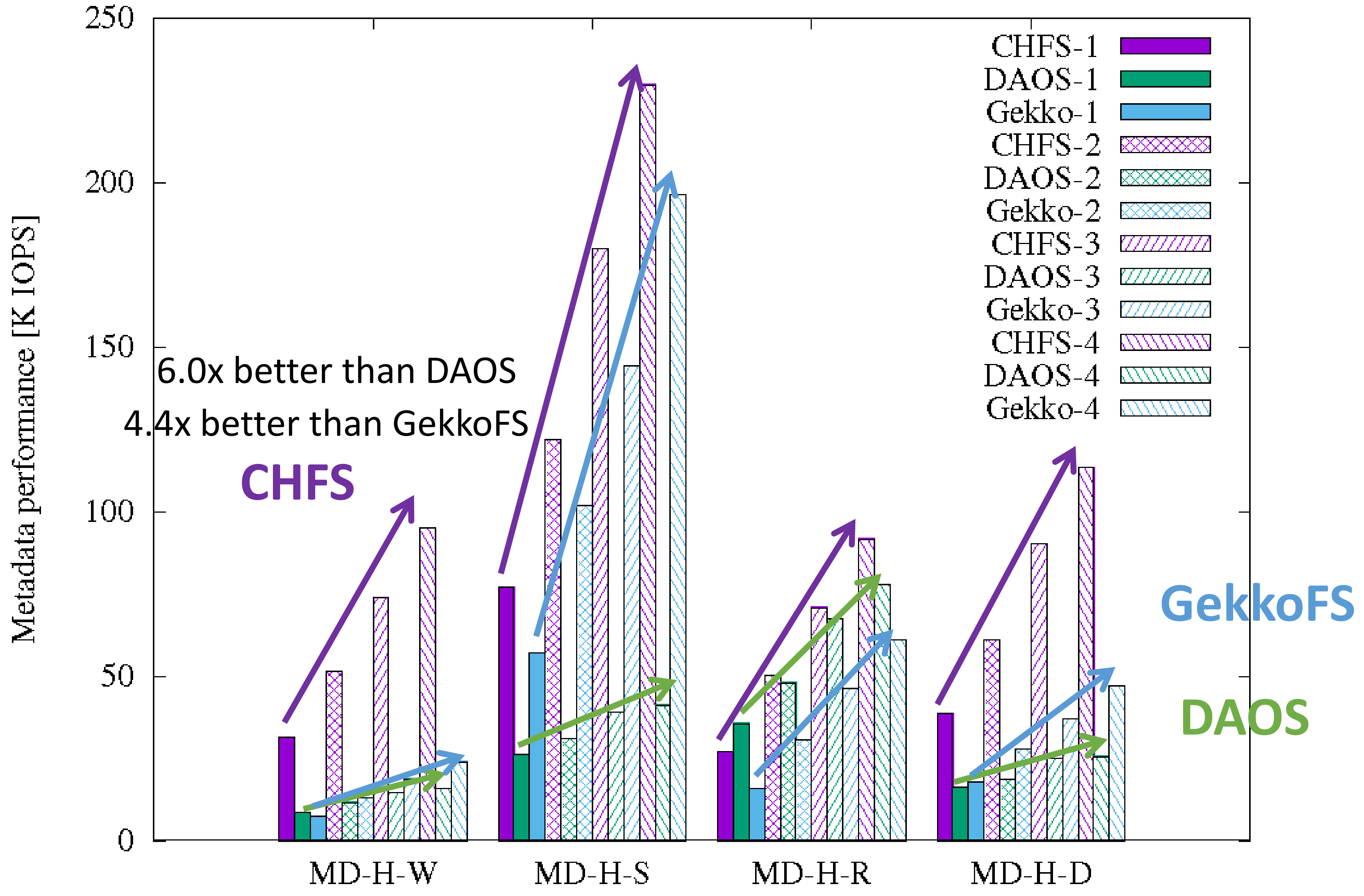
IOR hard

47,008 B

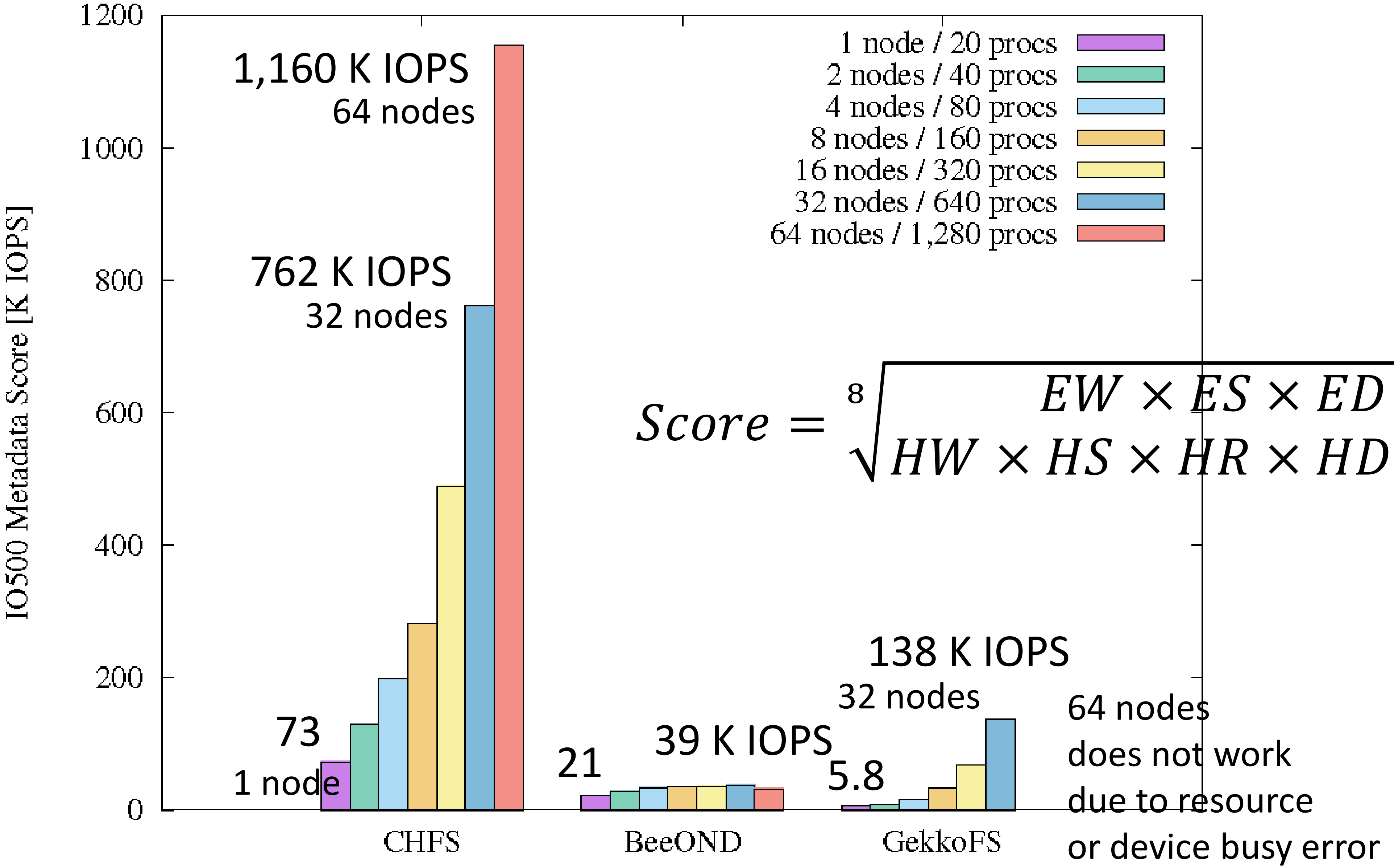
IO500 Bandwidth (CHFS/DAOS/GekkoFS)



IO500 Metadata Hard (CHFS/DAOS/GekkoFS)



IO500 Metadata on Cygnus (CHFS/BeeOND/GekkoFS)



$$Score = \sqrt[8]{\begin{matrix} EW \times ES \times ED \times \\ HW \times HS \times HR \times HD \times FIND \end{matrix}}$$

3	ISC20	Intel	Wolf	Intel	DAOS	10	420	758.71	164.77	3,493.56
4	ISC21	Lenovo	Lenovo-Lenox	Lenovo	DAOS	10	960	612.87	105.28	3,567.85
5	ISC20	TACC	Frontera	Intel	DAOS	10	420	508.88	79.16	3,271.49
6	ISC21	National Supercomputer Center in GuangZhou	Venus2	National Supercomputer Center in GuangZhou	kapok	10	480	474.10	91.64	2,452.87
7	ISC20	Argonne National Laboratory	Presque	Argonne National Laboratory	DAOS	10	380	440.64	95.80	2,026.80
8	ISC21	Supermicro		Supermicro	DAOS	10	1,120	415.04	112.17	1,535.63
9	SC19	NVIDIA	DGX-2H SuperPOD	DDN	Lustre	10	400	249.50	86.97	715.76
10	SC20	EPCC	NextGENIO	BSC & JGU	GekkoFS	10	3,800	239.37	45.79	1,251.32
11	ISC21	Olympus Storage Technology Innovation Lab	OceanStor	Huawei	OceanFS	10	960	220.10	69.49	697.15
12	SC20	Johannes Gutenberg University Mainz	MOGON II	JGU (ADA-FS)& BSC (NEXTGenIO)	GekkoFS	10	240	167.64	22.97	1,223.59
13	SC20	DDN	DIME	DDN	IME	10	110	161.53	101.60	256.78
14	SC19	WekaIO	WekaIO	WekaIO	WekaIO Matrix	10	2,610	156.51	56.22	435.76
15	ISC21	University of Tsukuba	Cygnus	OSS	CHFS	10	240	148.69	30.39	727.61
16	ISC21	Joint Institute of Nuclear Research	Govorun	RSC	DAOS	10	160	132.06	20.19	863.69
17	SC20	TACC	Frontera	DDN	IME	10	280	109.91	176.23	68.55

14	SC19	WekaIO	WekaIO	WekaIO	WekaIO Matrix	10	2,610	156.51	56.22	435.76
15	ISC21	University of Tsukuba	Cygnus	OSS	CHFS	10	240	148.69	30.39	727.61
16	ISC21	Joint Institute of Nuclear Research	Govorun	RSC	DAOS	10	160	132.06	20.19	863.69
17	SC20	TACC	Frontera	DDN	IME	10	280	109.91	176.23	68.55

#15 in 10 node list

#23 in full list

Design Goal of CHFS/Cache [ESSA 2022]

- Based on CHFS ad hoc PFS [HPC Asia 2022]
  - Exploit node-local persistent memory
  - High metadata performance, high bandwidth
  - Scalable performance
  - A separate FS from the backend PFS
- Design a caching FS by synchronizing with backend PFS
  - Not sacrifice metadata performance
  - Relax consistency between backend PFS in easy-to-understand semantics

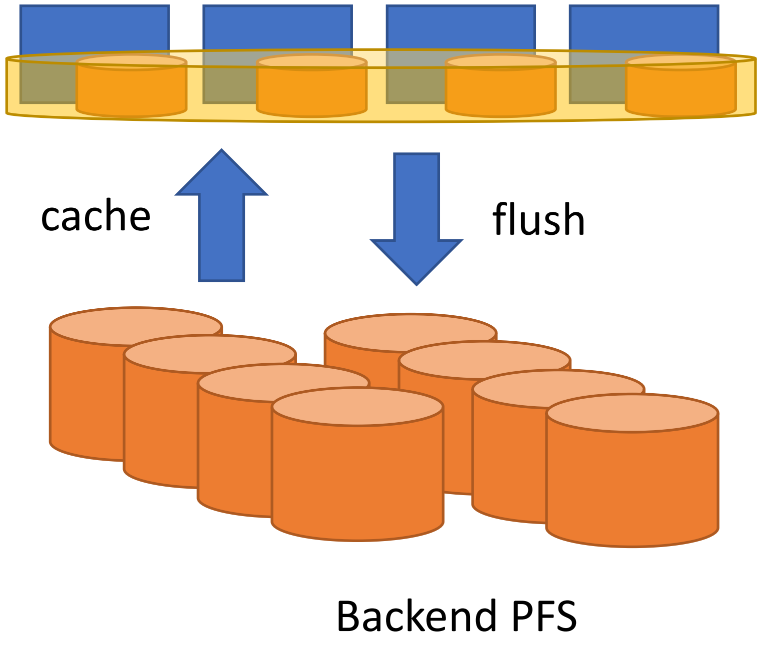
IO500 Score of Oakforest-PACS

	BW [GiB/s]	Metadata [kIOP/s]	Total
Lustre	21.4	88.78*	42.18*
IME	471.25	21.85	101.48

Relaxation of Consistency

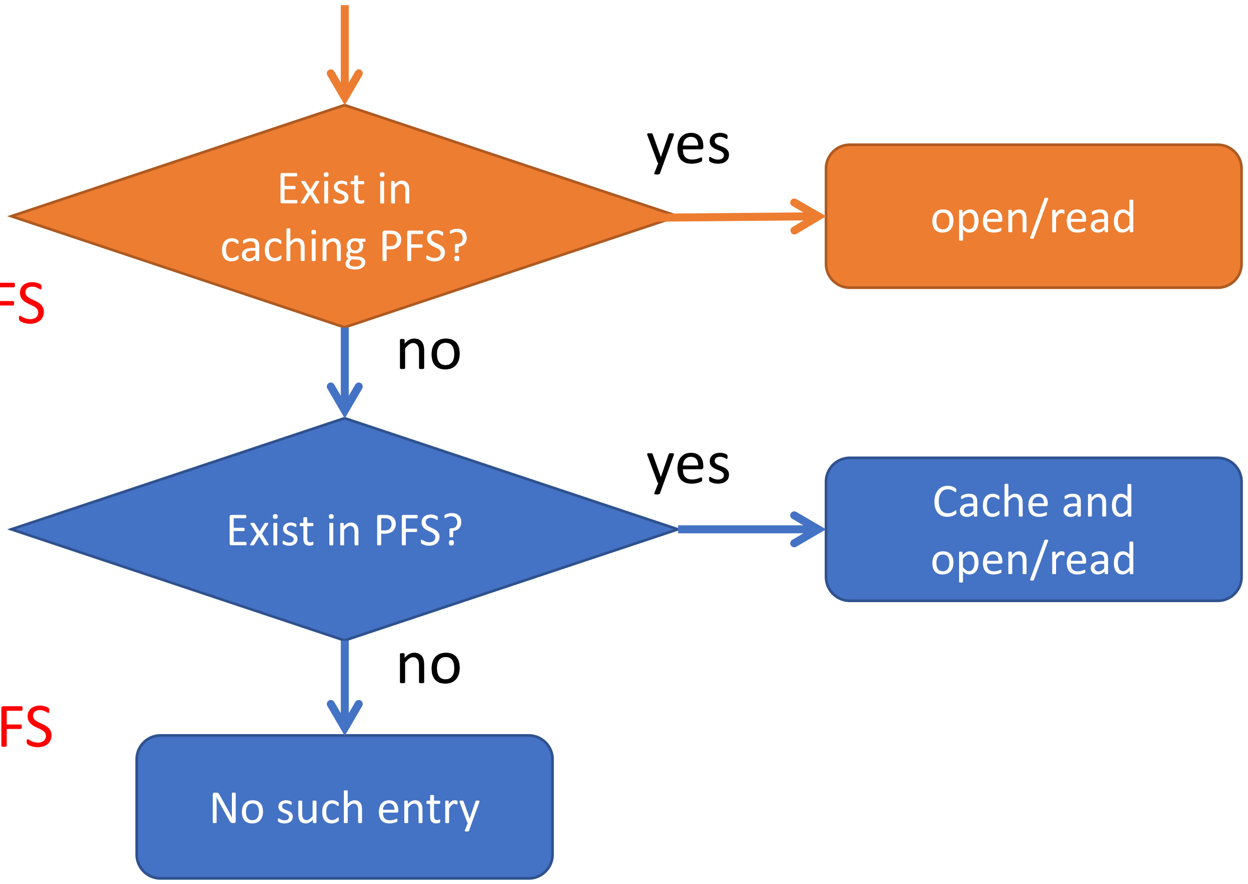
<u>Assumptions</u>

1. Input data not changed during the execution
2. Creation of new entries succeeds
3. Before updating an existing file, the file is read once
4. Updates not reflected until flushed
   (updates can be accessed from login nodes)
5. Flushing performed before the job terminates

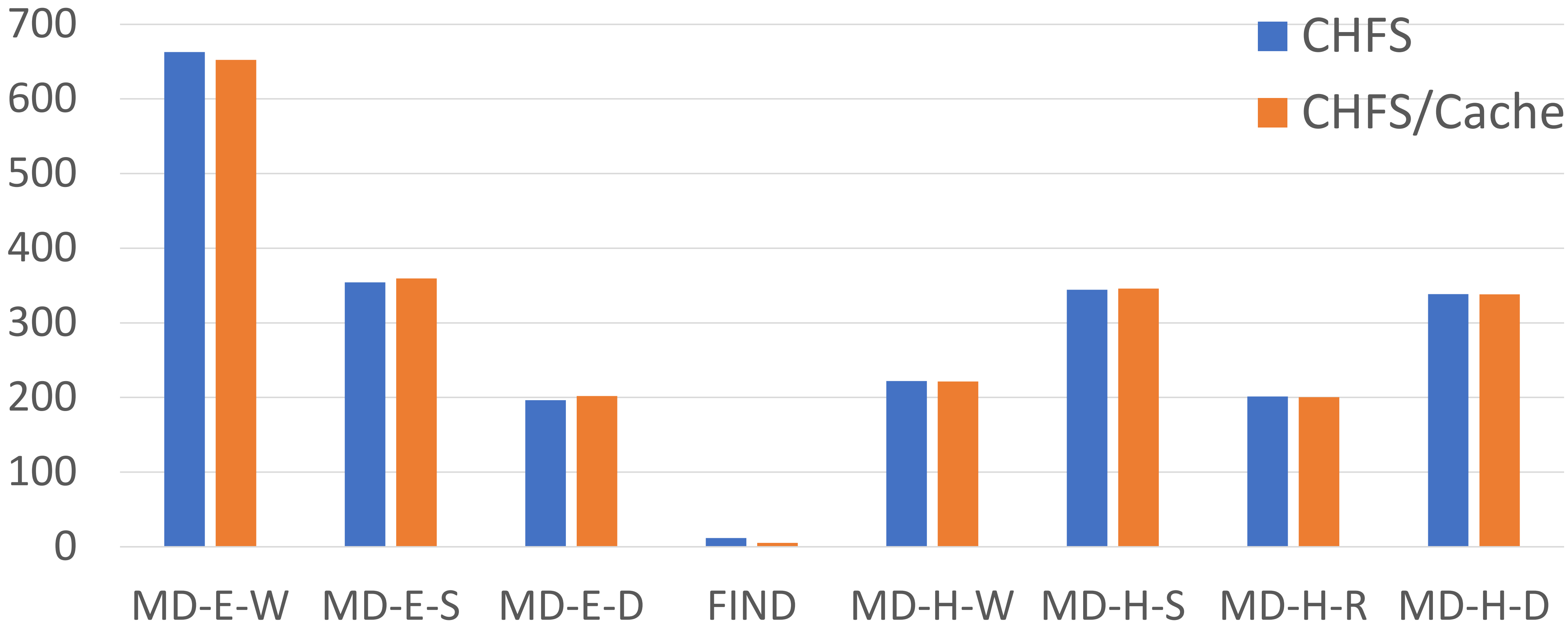


Design of File Operation

- File Open and Read
  - Right figure (From Assumption 1)
  - If exist in caching PFS,
    no checking performed for backend PFS
- File Creation and Write
  - Creates and writes in caching PFS (From Assumptions 2-4)
  - No checking performed for backend PFS



Metadata performance on Cygnus P nodes (4 nodes)



Overhead is less than 3% except FIND. It is 55% for FIND, but not problematic

Summary

- Pegasus will be introduced in 4Q 2022
  - Big memory and high-performance storage for data-driven and AI-driven science
- Research of ad hoc parallel file system
  - Better and scalable performance utilizing persistent memory
  - #15 in 2021 June IO500 10 node list, #23 in full list
- Design CHFS/Cache caching PFS
  - Solve the metadata performance problem
  - Better bandwidth and metadata performance than the backend PFS