WUHAN NATIONAL LABORATORY FOR OPTOELECTRONICS 武汉光电国家研究中心

華中科技大學

LAD19
Paris, France 23.09. - 25.09.19
LUSTRE ADMINS & DEVS WORKSHOP

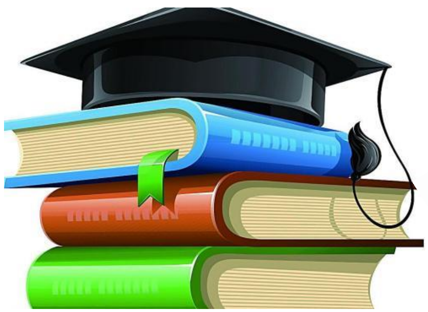
NVRAM-oriented Lustre Persistent Cache on Client

Lingfang Zeng, Xi Li*, and Wen Cheng
Wuhan National Laboratory for Optoelectronics (WNLO)
Huazhong University of Science and Technology (HUST)
***DDN / Whamcloud**

With Contributions from

- Yingjin Qian, Shuichi Ihara, Carlos Aoki Thomaz, and Shilong Wang @ **DDN**
- Andreas Dilger @ **DDN/Whamcloud**
- Tim Süß, and André Brinkmann @ **JGU**
- Chunyan Li, Fang Wang, and Dan Feng @ **HUST**
- LPCC
  - **SC2019**

Outline



BACKGROUND

PROBLEM & TERMINOLOGY & OBJECTIVES

METHODS

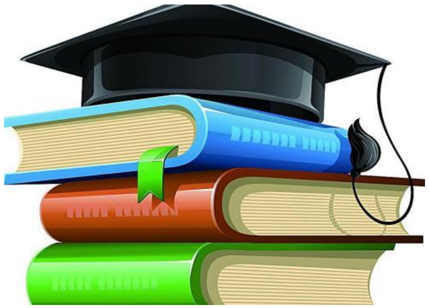
HIERARCHICAL PERSISTENT CLIENT CACHING

IMPLEMENTATION

RW-PCC & RO-PCC & RULE-BASED TRIGGERING & POLICY ENGINE

EVALUATIONS

EXPERIMENT & RESULTS

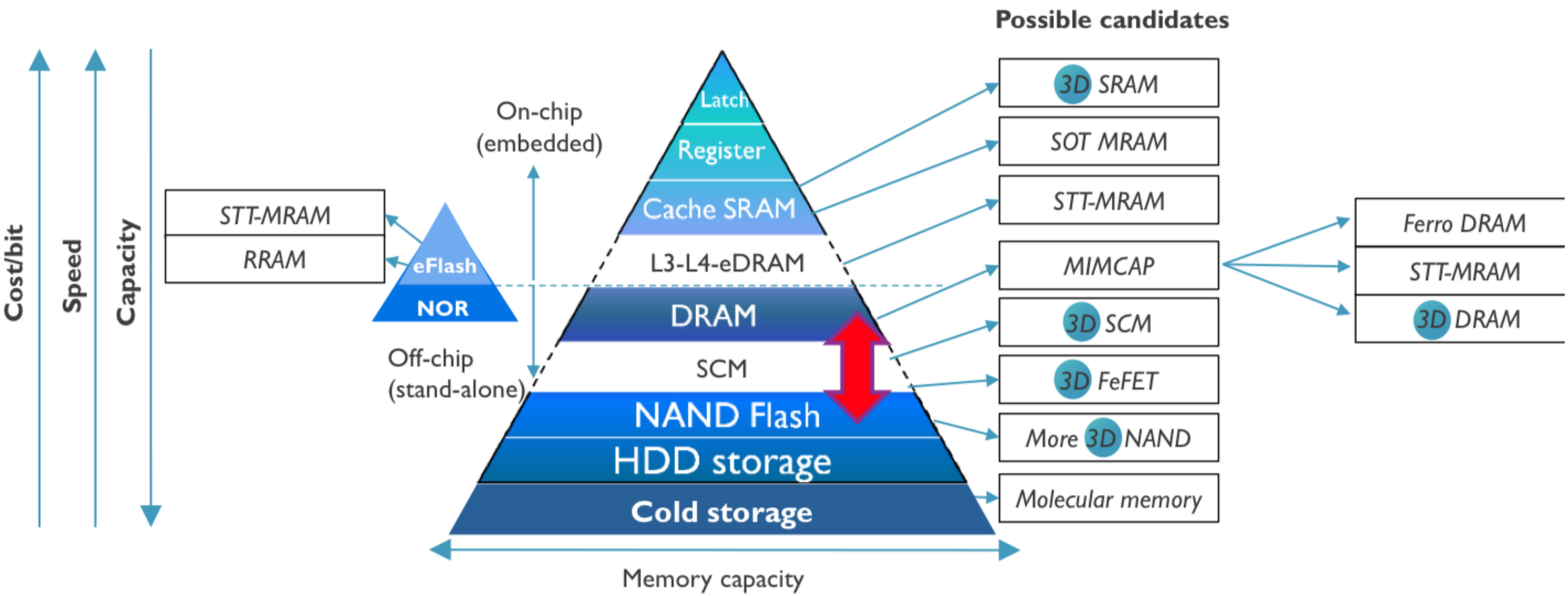


01

BACKGROUND

PROBLEM & TERMINOLOGY & OBJECTIVES

Hierarchical Storage Management (HSM)

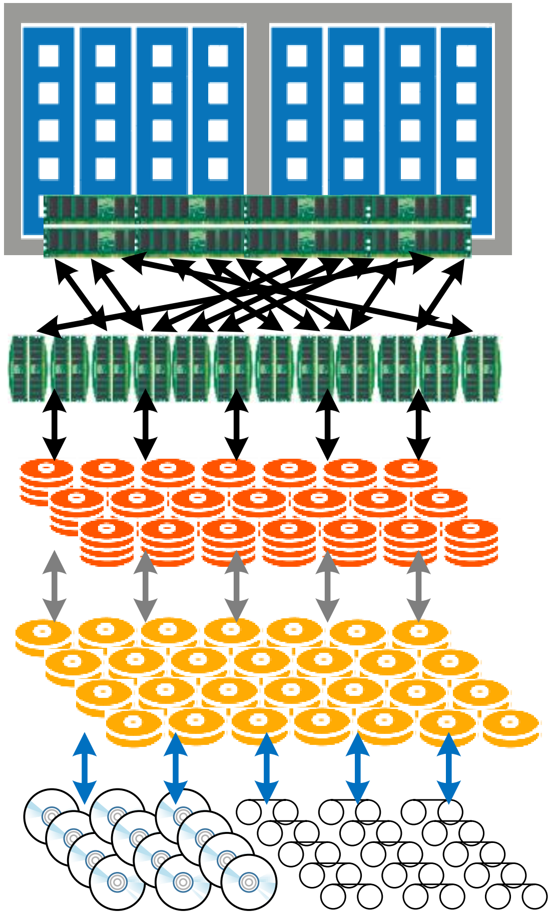


HPC workloads were too big to be stored only on flash

https://semiengineering.com/a-new-memory-contender/
https://storageswiss.com/2018/01/10/enterprise-needs-to-learn-from-hpc-environments/

HSM Tier

- Compute servers
  - HBM
  - NVRAM/SCM
- Performance storage
  - DRAM
  - SSD
  - (performance HDD)
- Capacity storage
  - DRAM
  - Capacity HDD



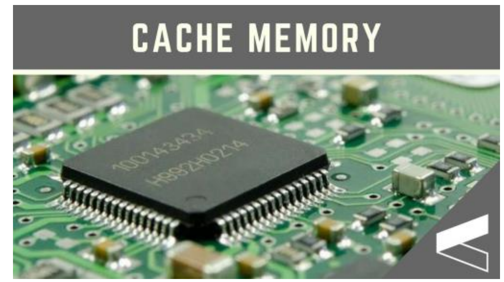
Lang's Law: the more tiers, the more tears

Problems

- Performance
  - Speed defines the winner
  - Cache
- Utilization rate (Lustre client devices)
  - NVMe
  - Flash-based SSD
  - NVRAM/SCM
- Data consistency
- Transparency

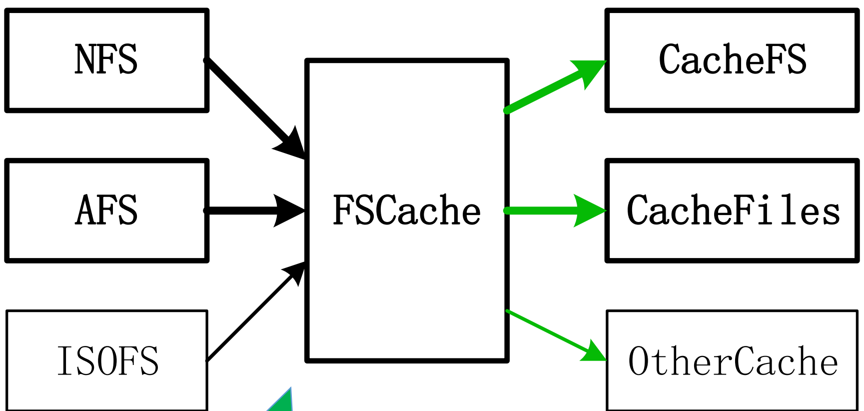
Industry and Academic Solutions

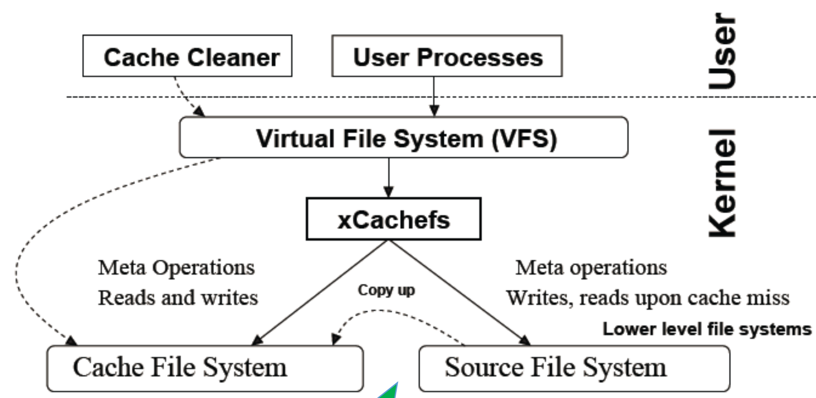
- Andrew File System [TOCS'88, CMU]
- Coda File System [TOCS'88, CMU]
- FS-Cache [Linux Symposium'06, Red Hat]
- BWCC [CLUSTER'12, CAS]
- Nache [FAST'07, RU & IBM]
- Panache [FAST'10, IBM]
- Mercury [MSST'12, NetApp]
- GPFS' LROC [IBM]
- TRIO [CLUSTER'15, FSU & ORNL & AU]
- BurstFS [SC'16, FSU & LLNL]
- MetaKV [IPDPS'17, FSU & LLNL]



- Dmcache [TOCS'88, CMU]
- Xcachefs [SBU, 2005]
- FlashCache [CASES'06, UM]
- Bcache [LWN, 2010]

Related Work



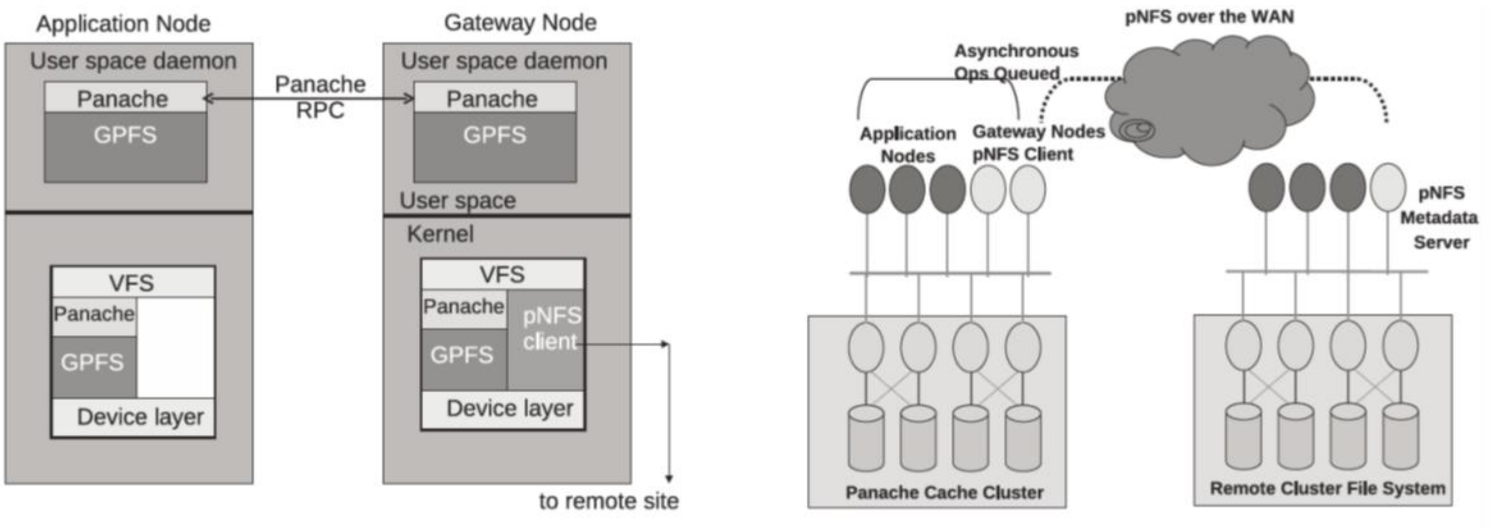


- Read-only cache
- Tolerate I/O failures in cache

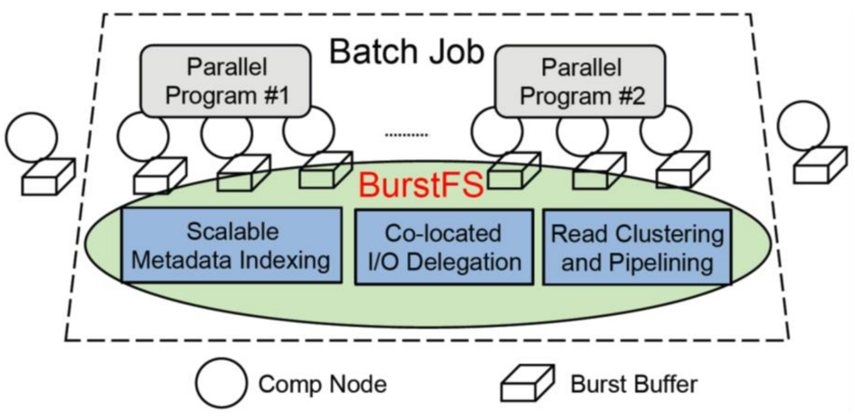
- File system meta-operations (both cache and source)

Reference: Howells, FS-Cache: A Network Filesystem Caching Facility, Red Hat UK Ltd.
Sivathanu+, A Versatile Persistent Caching Framework for File Systems, Stony Brook University, Technical Report FSL-05-05.

Related Work



(a) Node Block Diagram (b) Cache Cluster Architecture



Reference: Eshel+, Panache: A parallel file system cache for global file access, FAST'10
Wang+,An ephemeral burst-buffer file system for scientific applications, SC'16

Lustre File System

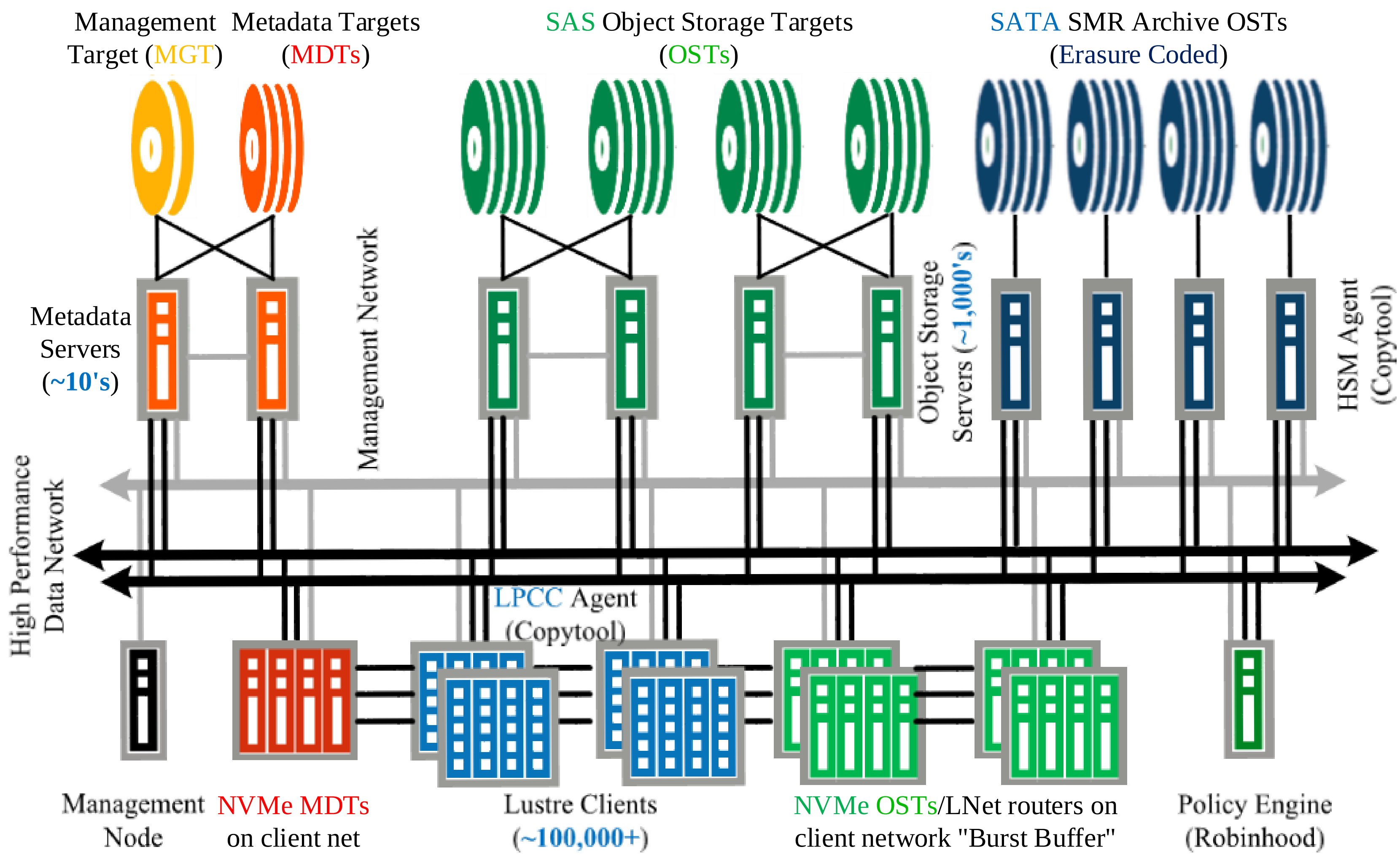
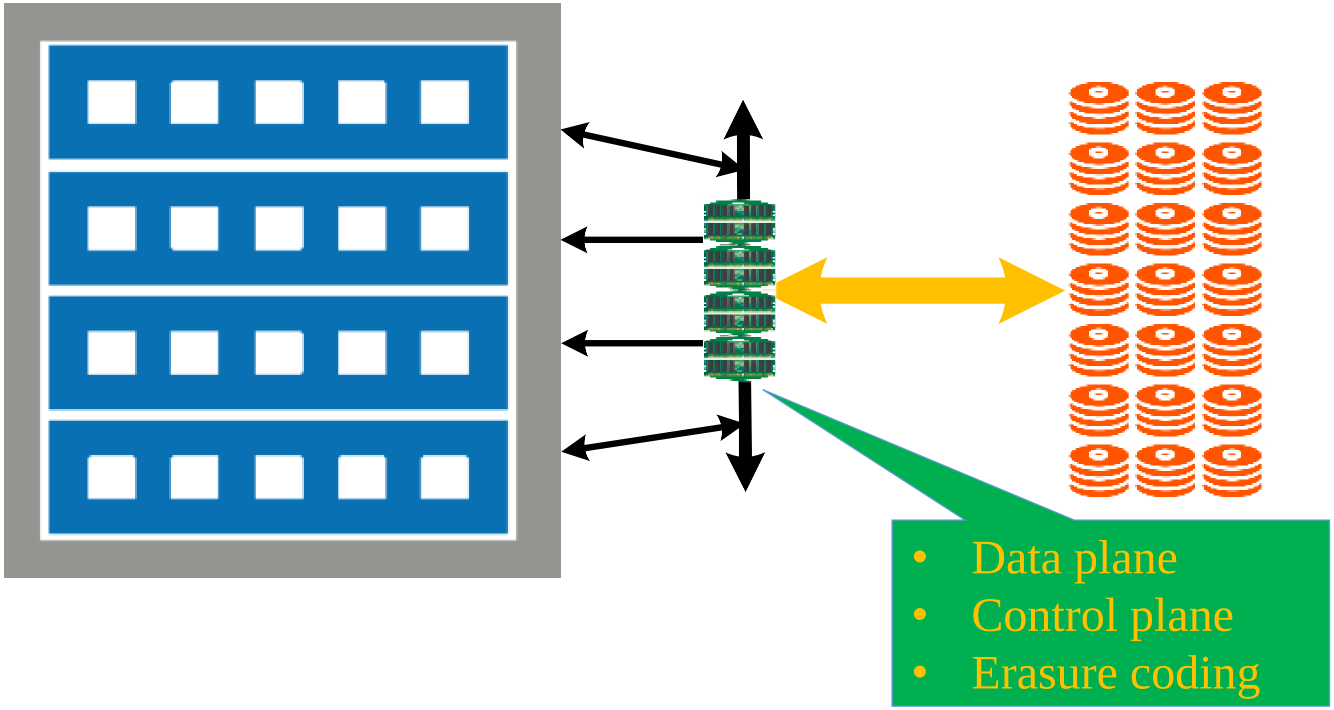


Figure based on Andreas Dilger's Lustre User Group 2018 presentation: Lustre 2.12 and beyond (see http://opensfs.org/lug-2018-agenda/)

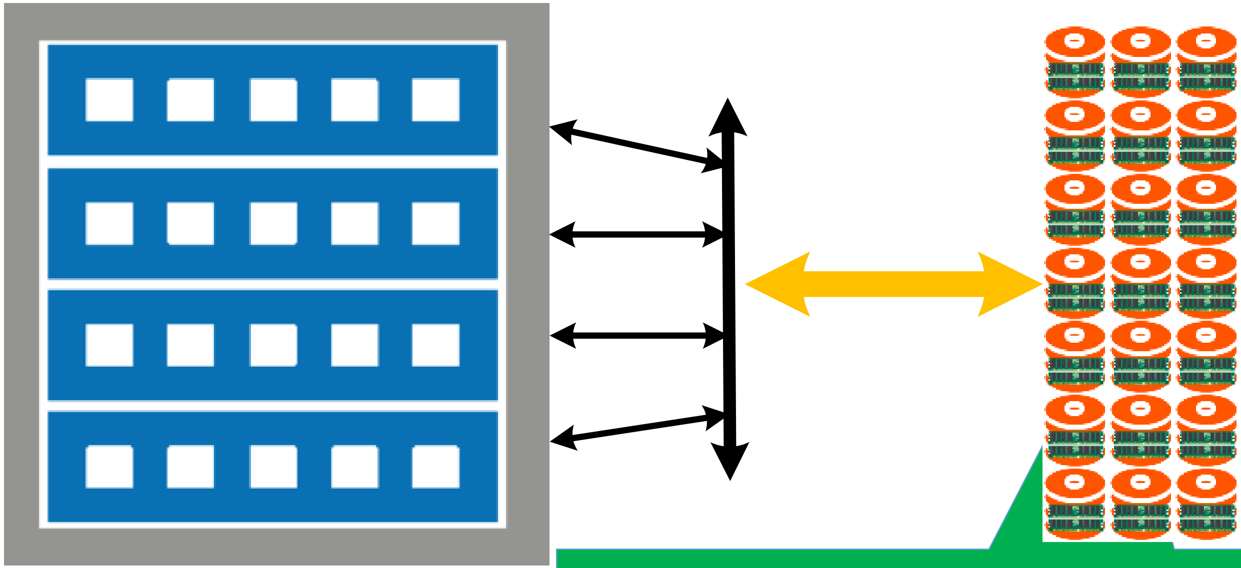
HSM Tier

- Shared
  - DDN IME @ ICHEC
  - Cray Trinity @ LANL



HSM Tier

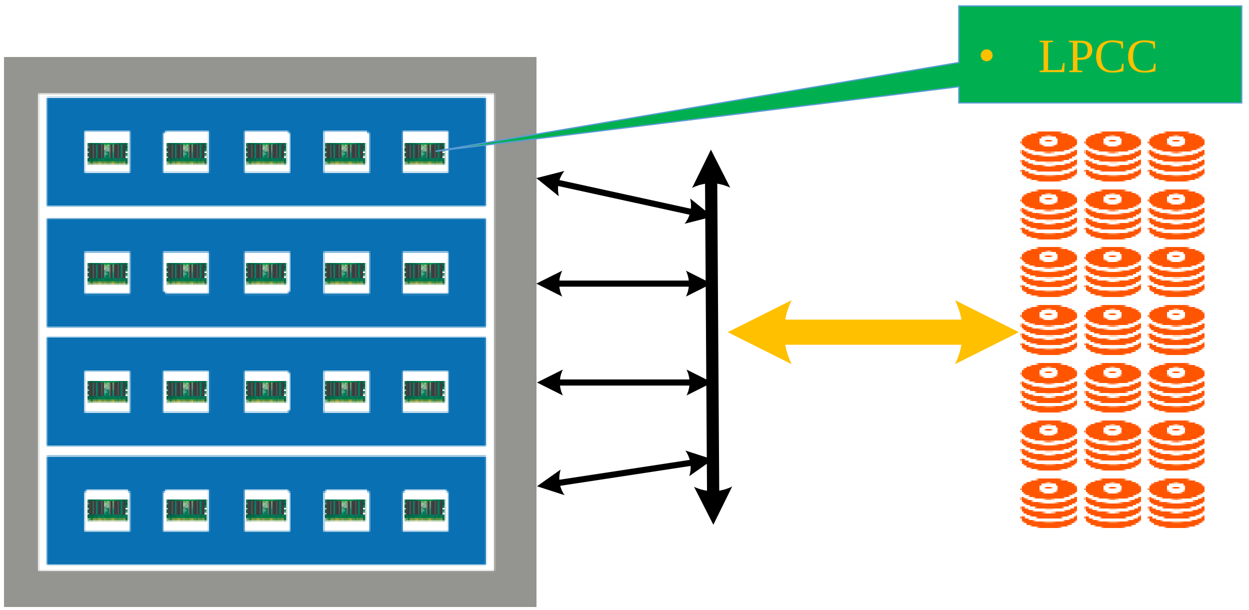
- Shared
  - DDN IME @ ICHEC
  - Cray Trinity @ LANL
- Server-side
  - Seagate Nytro NXD @ Sanger



- Storage-side flash acceleration
- I/O histogram
- Performance statistics
- Dynamic flush

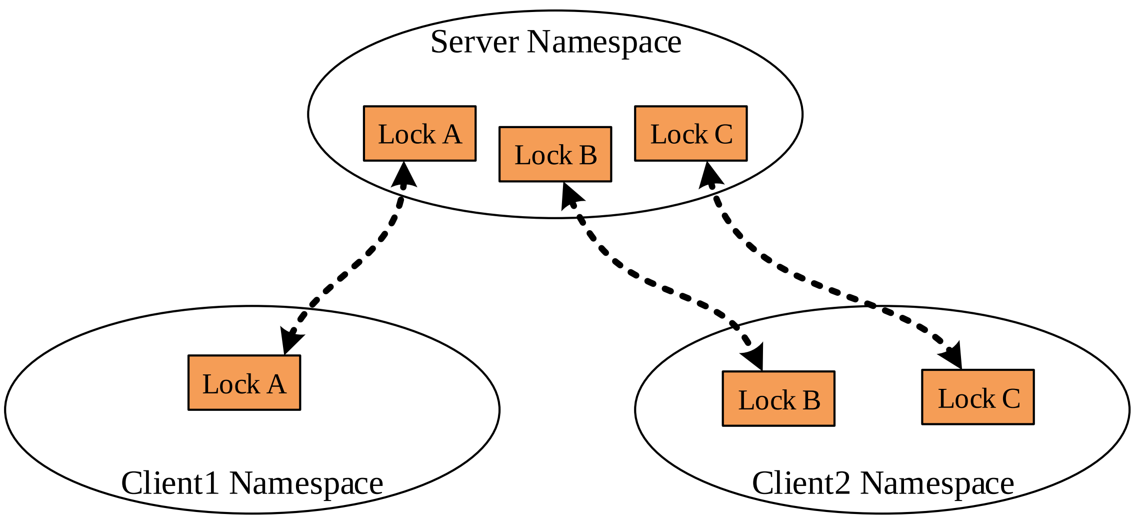
HSM Tier

- Shared
  - DDN IME @ ICHEC
  - Cray Trinity @ LANL
- Server-side
  - Seagate Nytro NXD @ Sanger
- Client-side
  - **Lustre Persistence Client Cache (LPCC)**



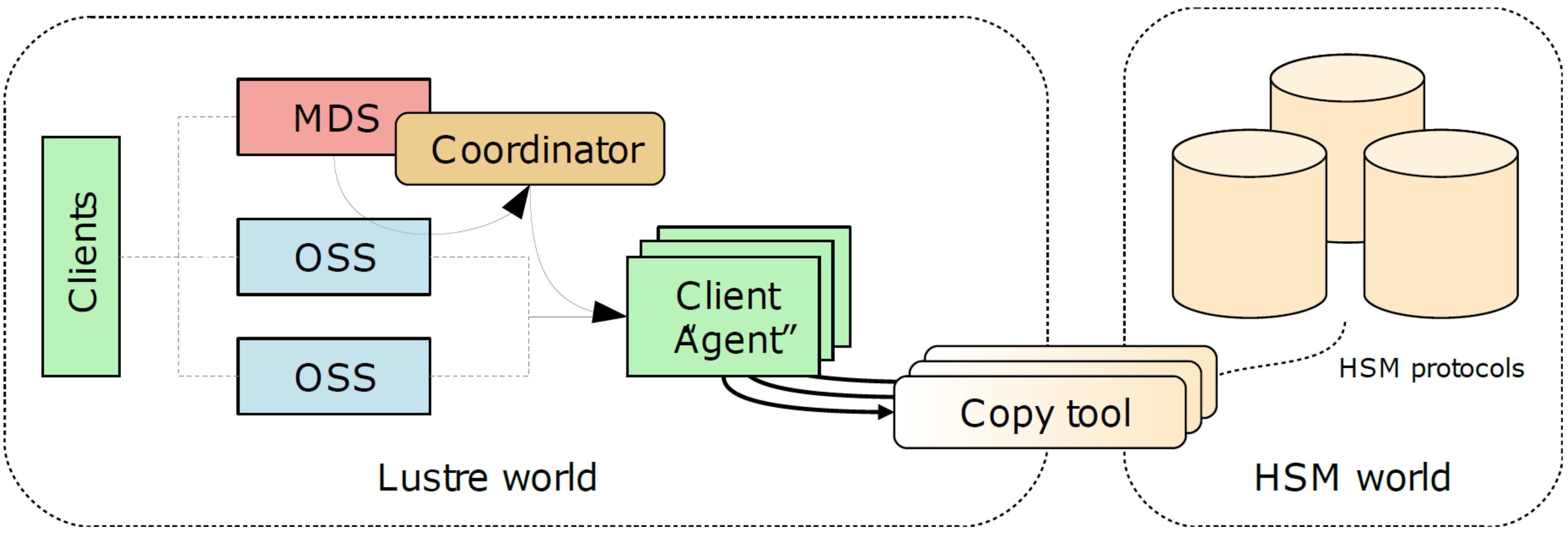
Lustre's DLM and Layout Lock

- Distributed lock manager (DLM)
  - Data and metadata consistency
  - A separate namespace
- Excusive mode (EX) lock
- Concurrent read mode (CR) lock
- *L.Gen* field



Lustre HSM

- Agents – Lustre file system clients running Copytool
- Coordinator – Act as an interface between the policy engine, the metadata server(MDS) and the Copytool



Refererence: Lustre manual. http://doc.lustre.org/lustre_manual.pdf

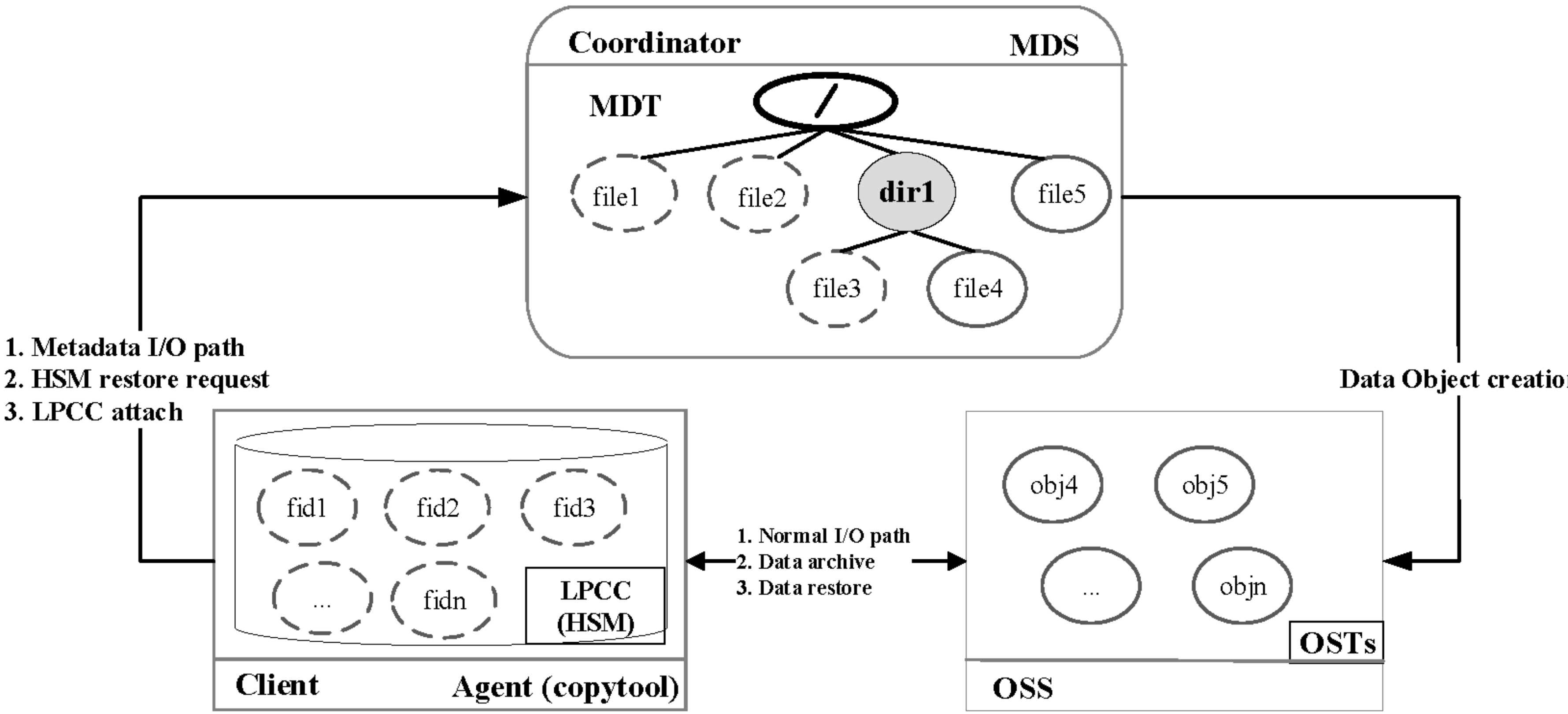
Key Idea

- Logical two-tier (with physical multitier)
  - Simple and efficient architecture (memory vs. disk)
- A global namespace
  - Space efficient
- Latencies and lock conflicts can be significantly reduced
- Caching reduces the pressure on (OSTs)
  - small or random I/Os can be regularized to big sequential I/Os and temporary files do not even need to be flushed to OSTs.

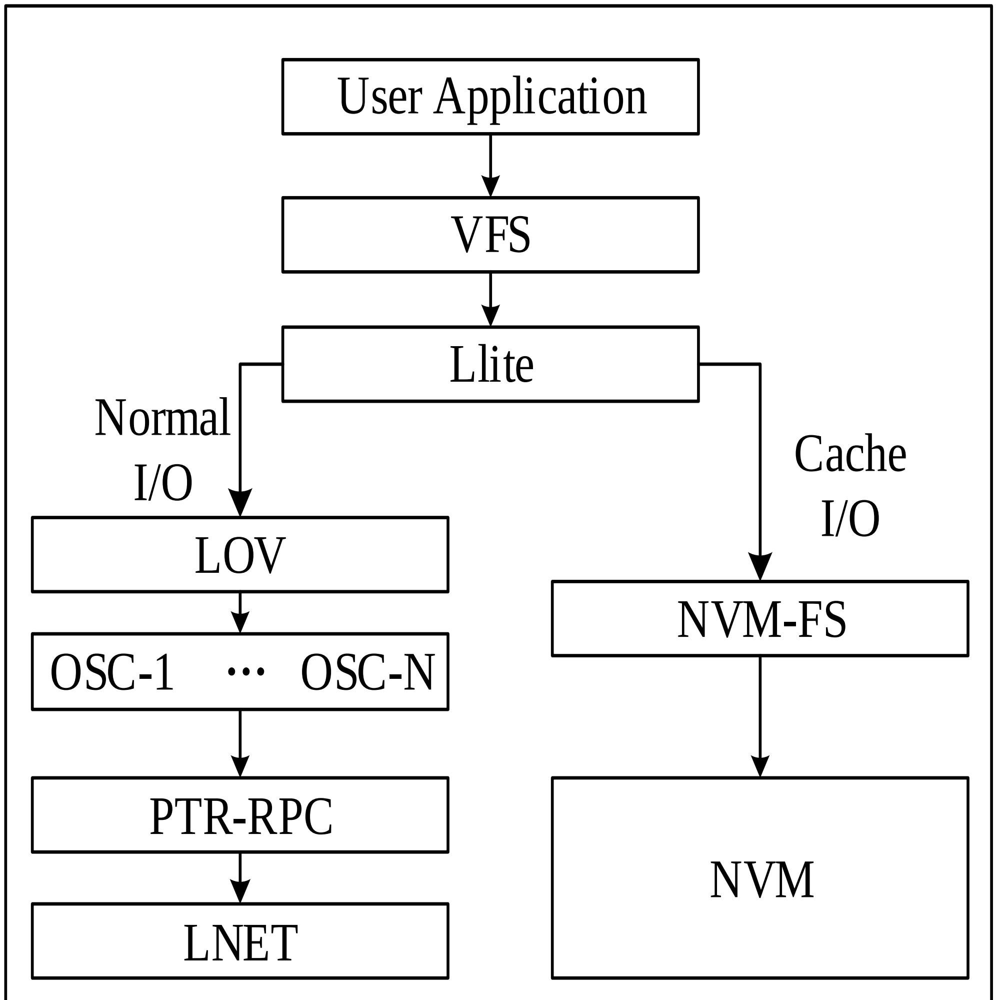
02 METHODS

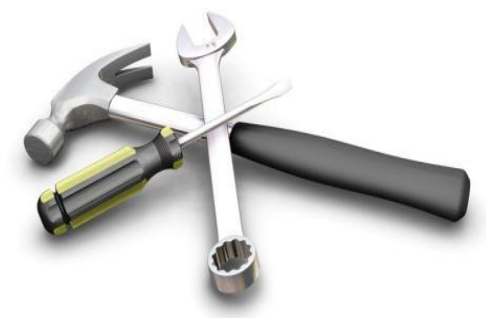
HIERARCHICAL PERSISTENT CLIENT CACHING

Overview of LPCC Architecture



I/O Path

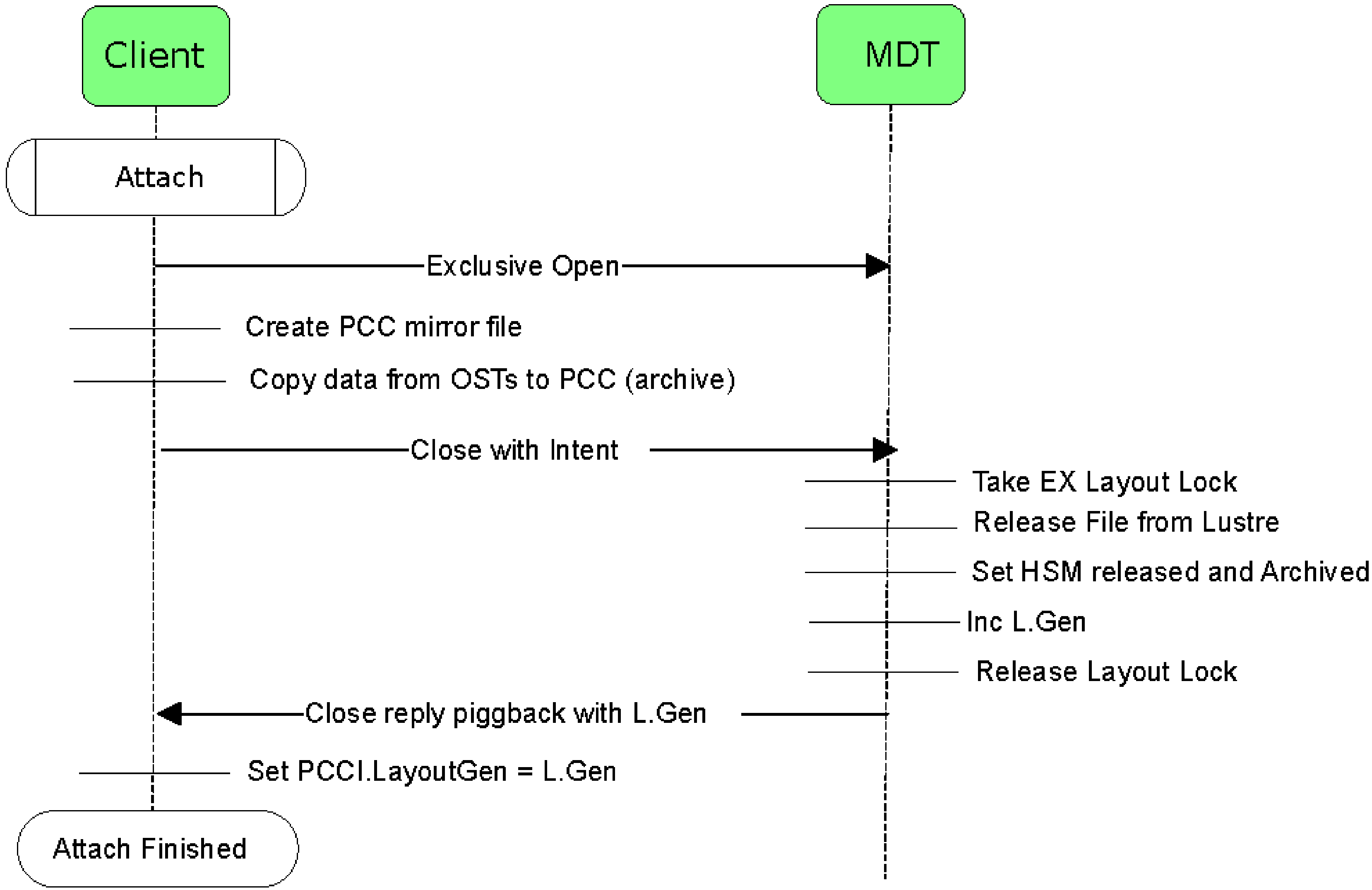

User Application
VFS
Llite
Normal I/O
Cache I/O
LOV
OSC-1 ··· OSC-N
PTR-RPC
LNET
NVM-FS
NVM



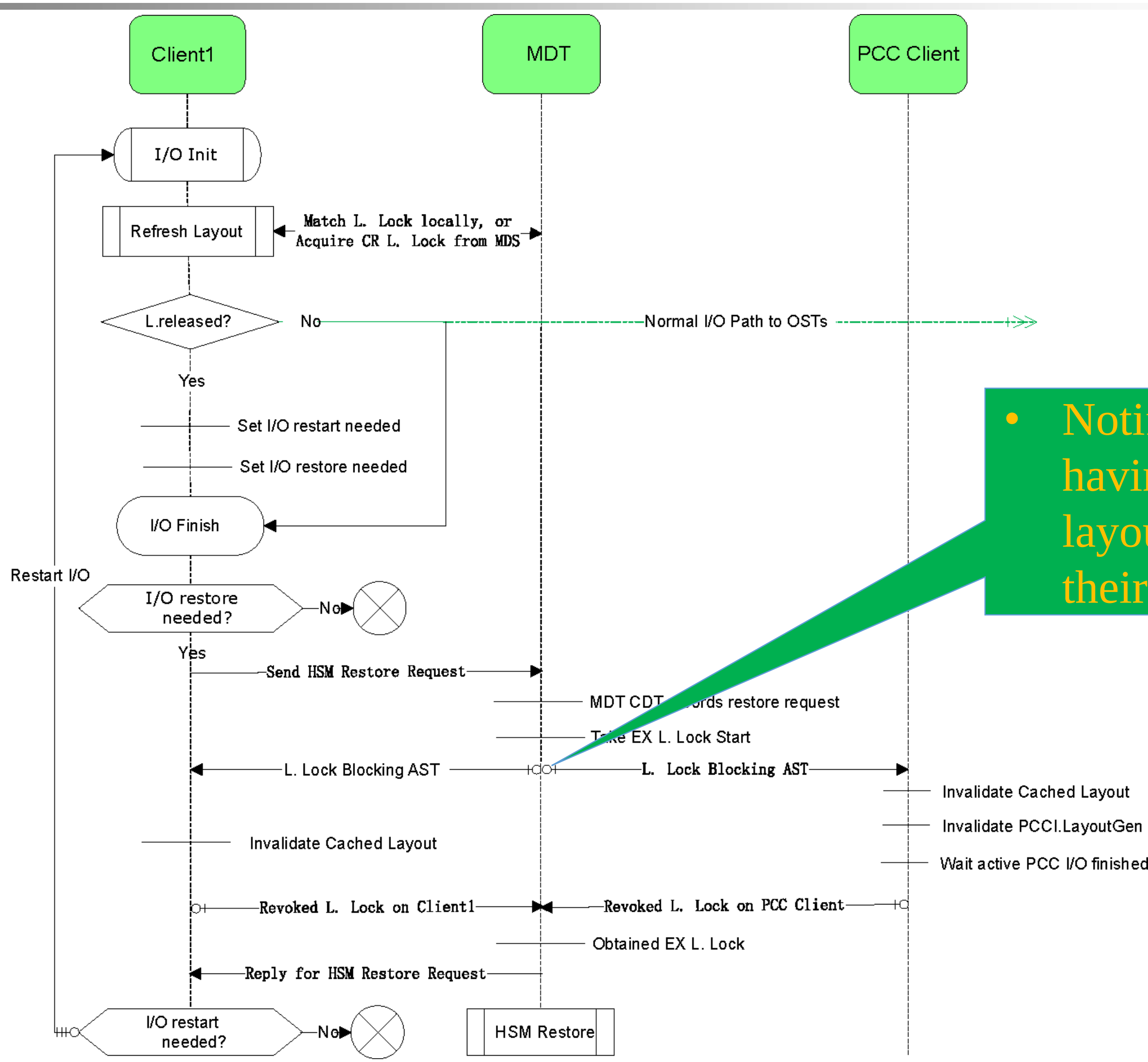
03 IMPLEMENTATION

RW-PCC & RO-PCC & RULE-BASED TRIGGERING & POLICY ENGINE

Lustre Read-Write PCC Caching (attach)

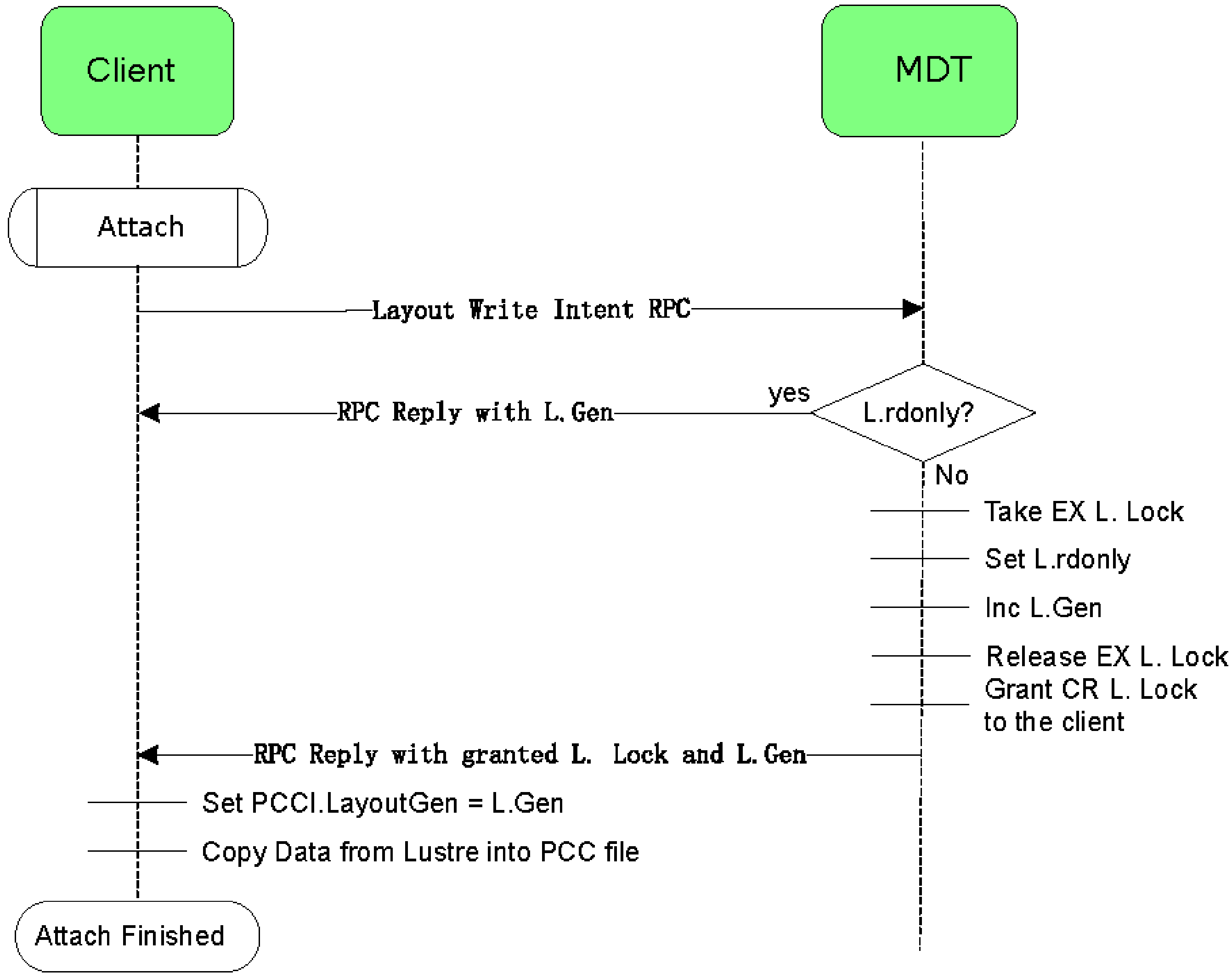


Lustre Read-Write PCC Caching (restore)

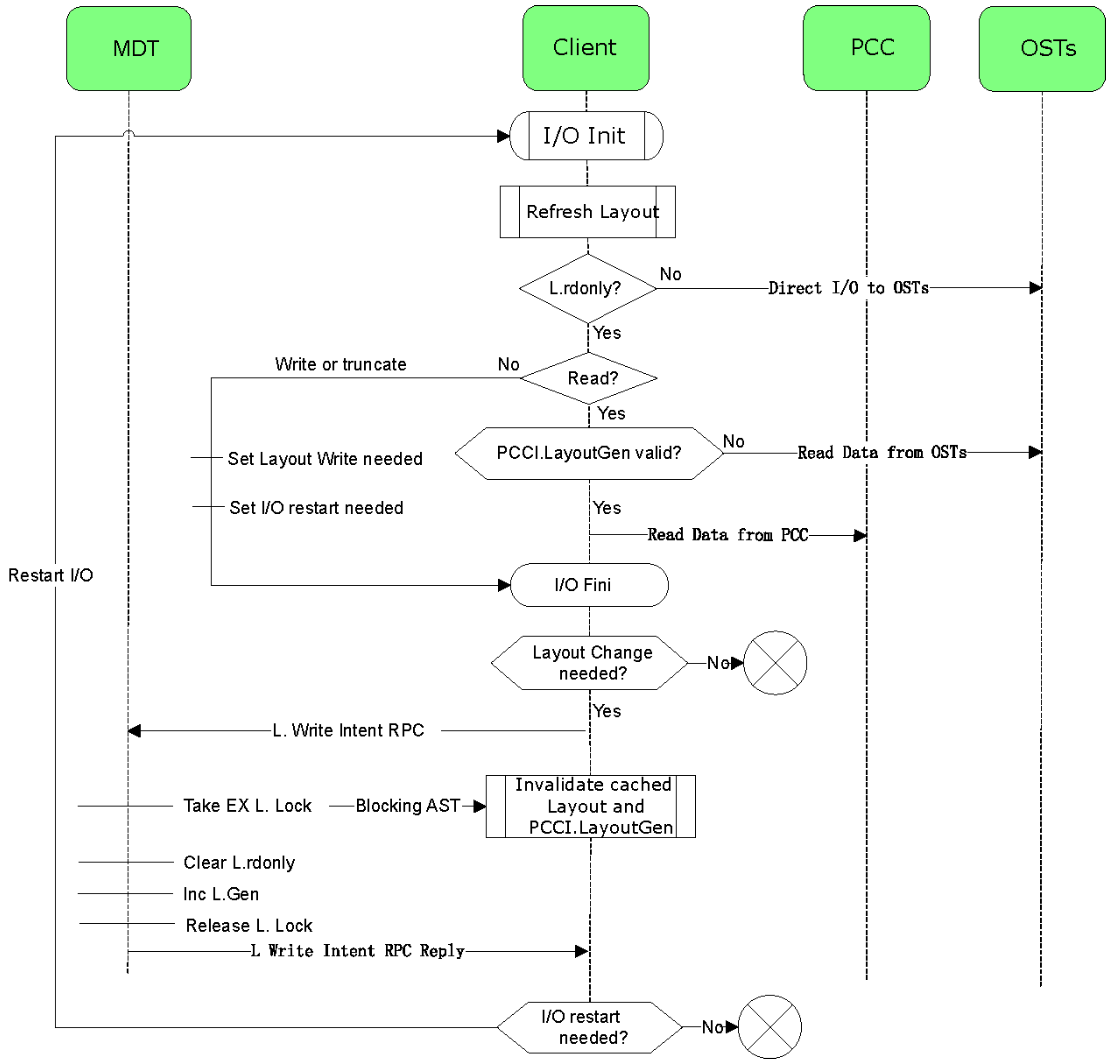


- Notify all clients having cached the layout to invalidate their layouts

Lustre Read-only PCC Caching (attach)



Lustre Read-only PCC Caching (I/O flow)



Rule-based Persistent Client Caching

- Different user, groups, and projects or filenames
  - E.g. (projid={500,1000} & fname=*.h5),(uid=1001)
- Quota limitation
  - Cache isolation
- Auto LPCC caching mechanism

Cache Prefetching and Replacement

- Policy engine
  - Manage data movement
- Lustre changelogs
  - Periodic prefetching decision
- LRU and SIZE

04 EVALUATIONS

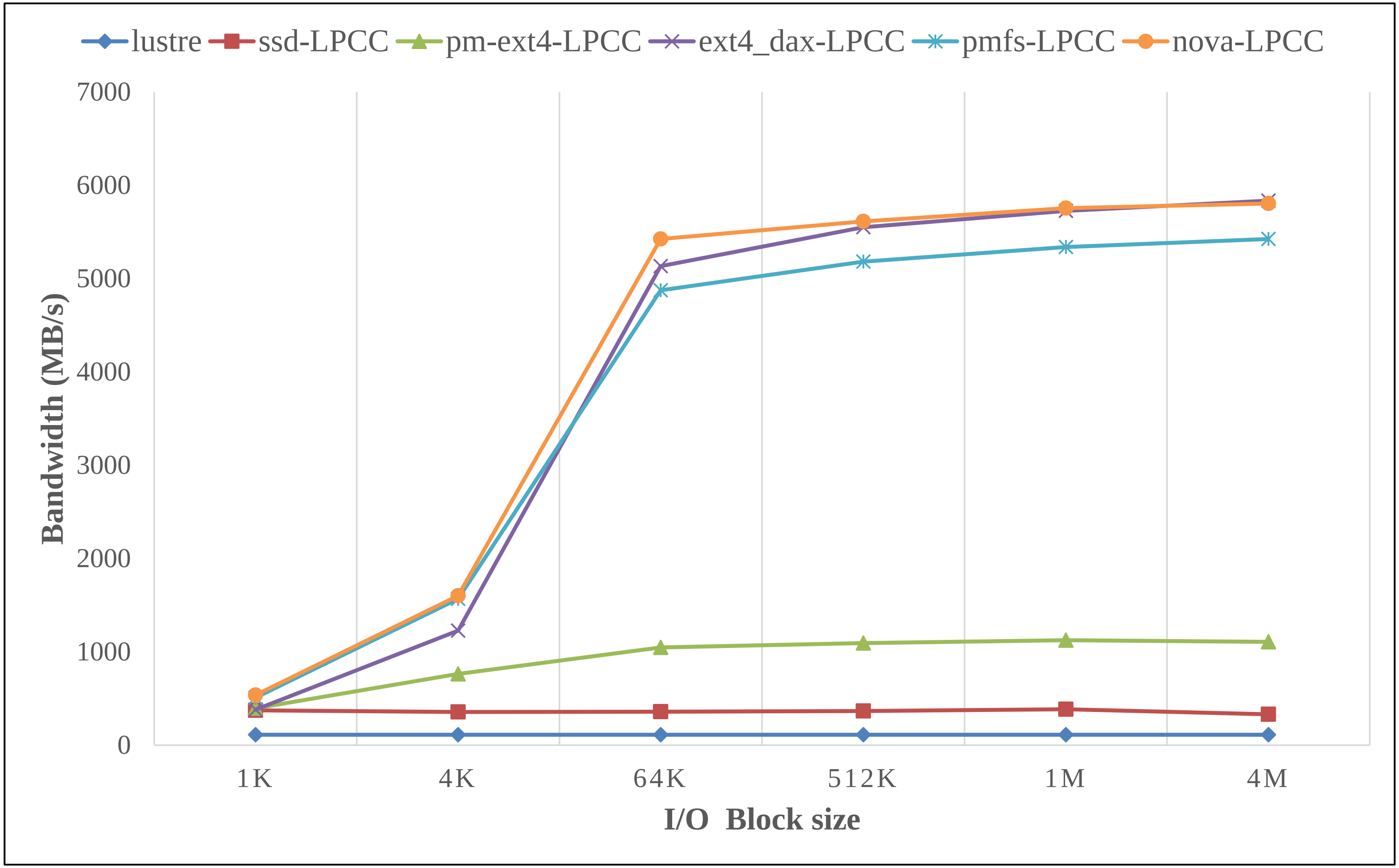
EXPERIMENT & RESULTS

Evaluation Setup

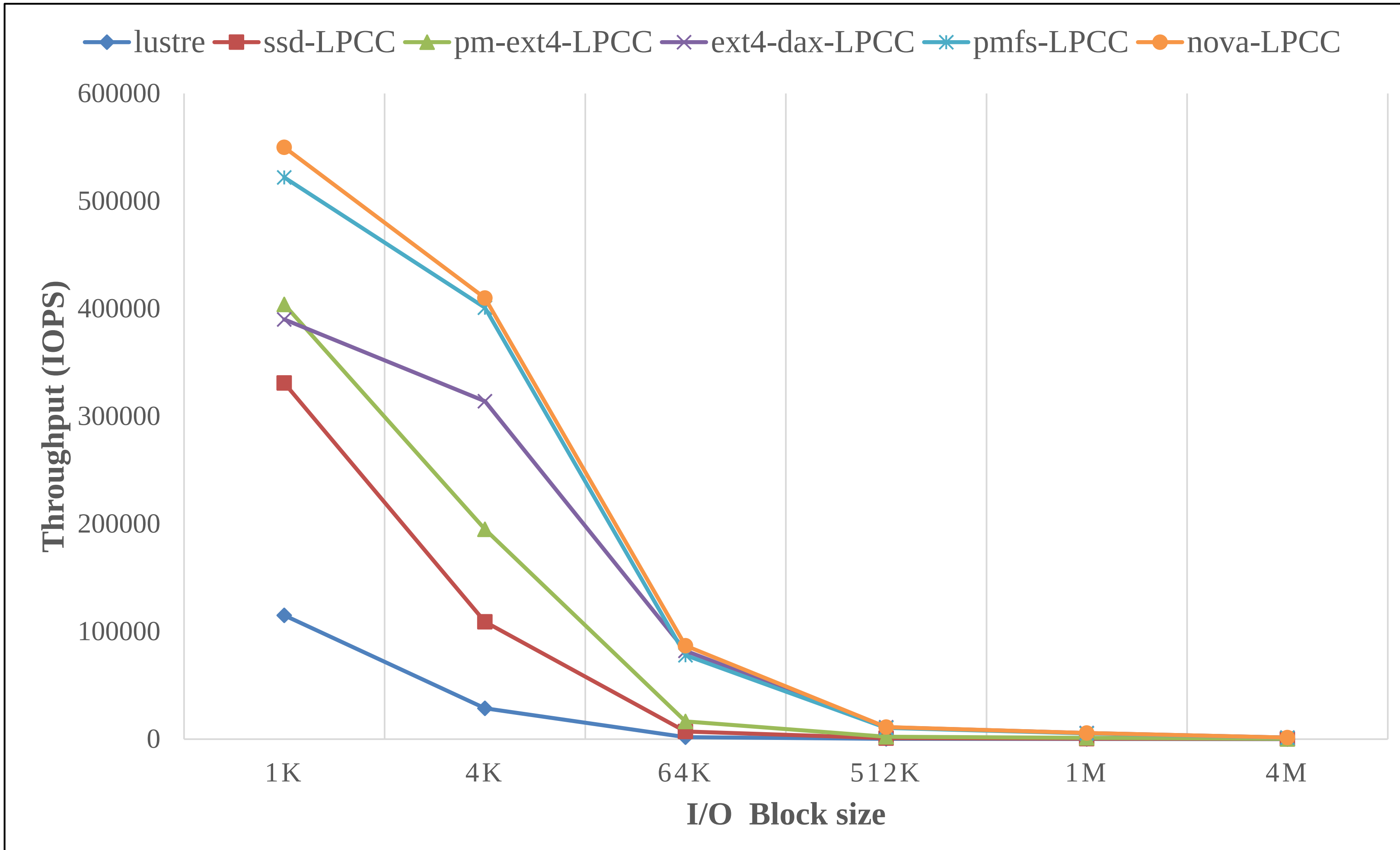
- Simulate DRAM to NVRAM
  - DRAM (28GB)/NVM(100GB)
  - Write latency 200ns
- Clients(8)
- OSS(3)
- MDS(1)

Server	Inspur SA5248L
CPU	Intel(R) Xeon(R) CPU E5-2620, 2.00GHz
DRAM	128GB
SSD	Kingston SA400S37/240G
HDD	SAS Disks
OS	Centos-7.5
Kernel	3.10.0-862.6.3
Network	1GB Ethenet
Lustre	Lustre 2.11.53
Fio	fio-3.1
Filebench	filebench 1.5-alpha3
IOR	IOR-3.2.0

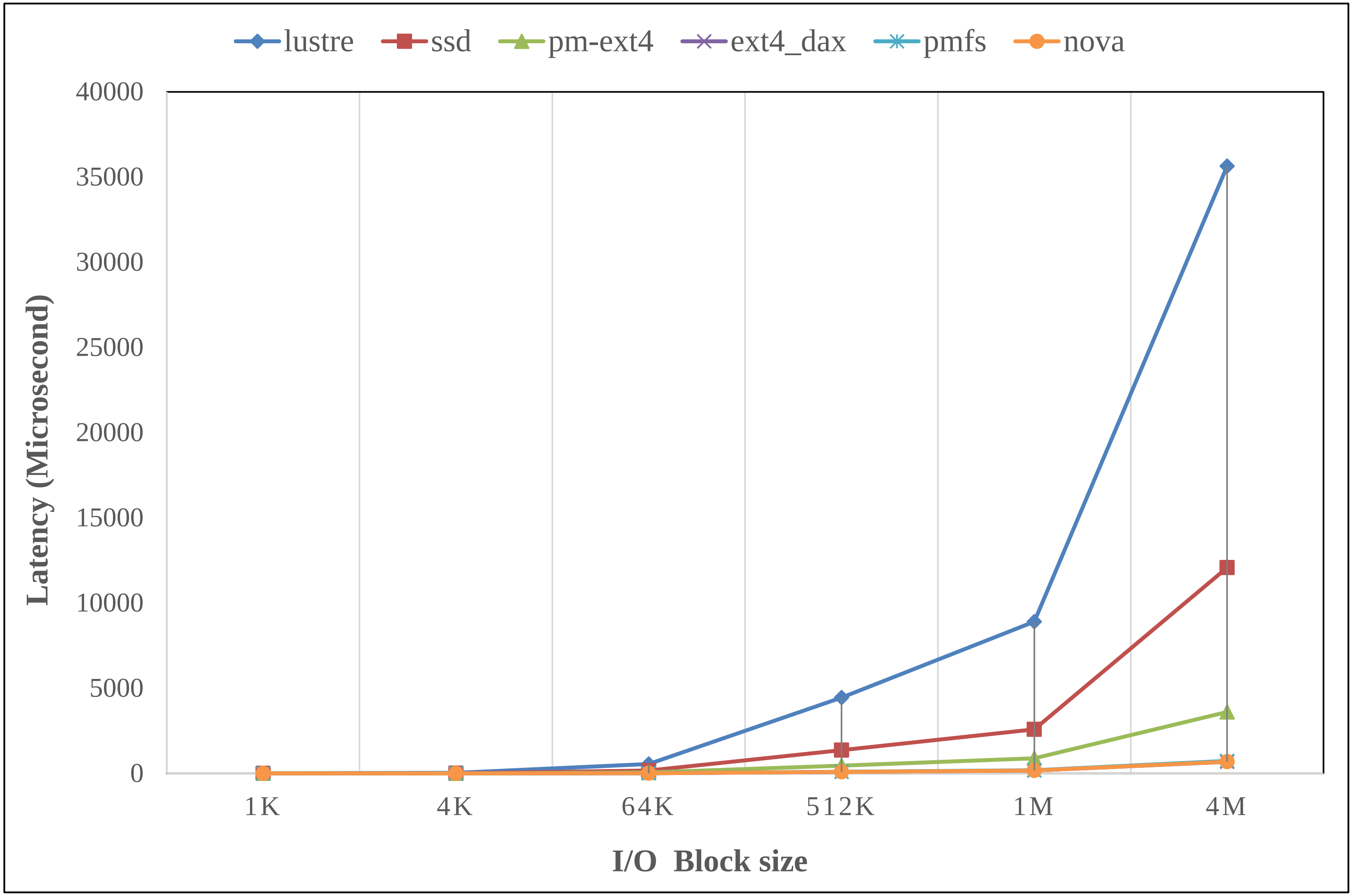
Single Client Read Performance (fio)



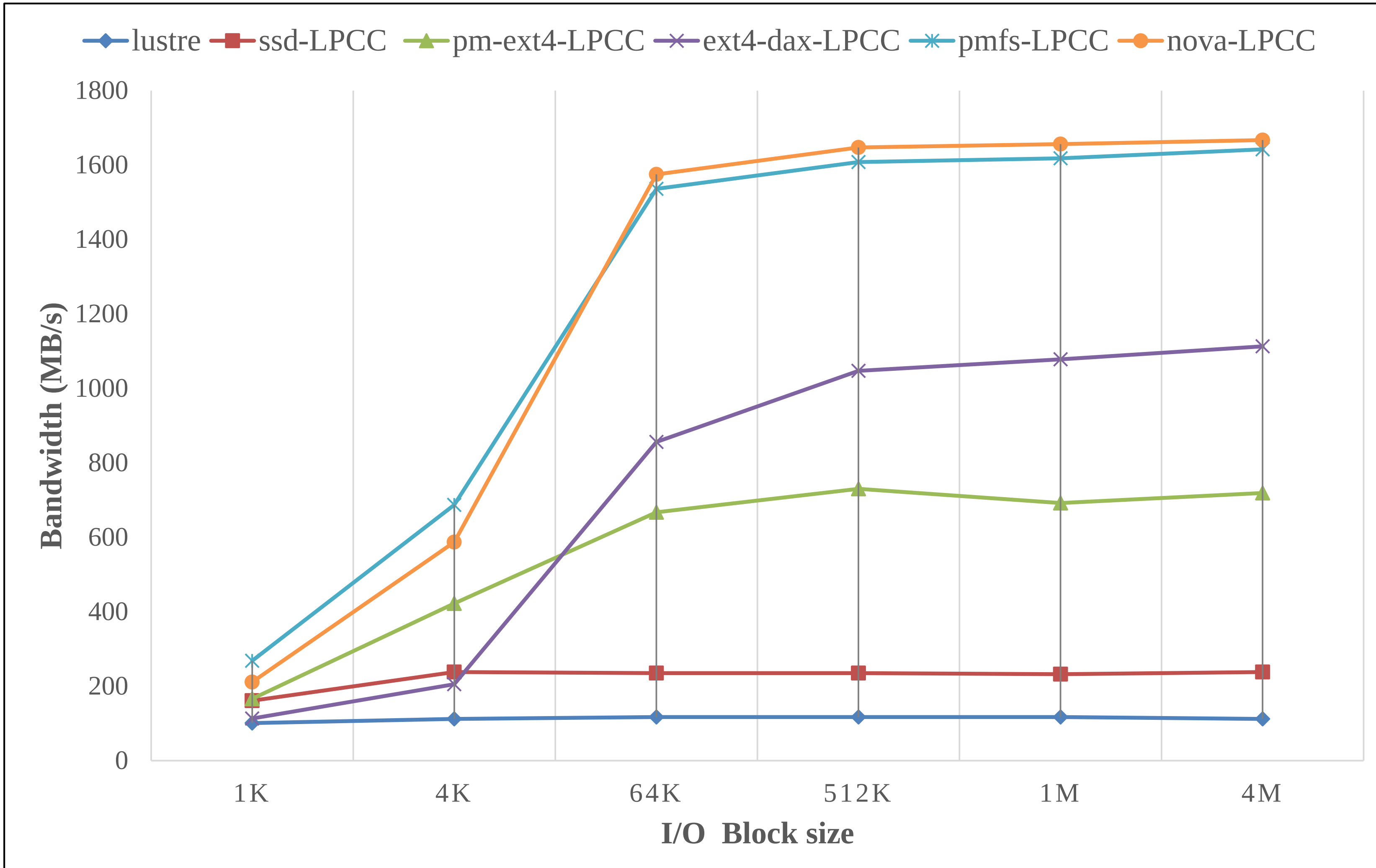
Single Client Read Performance (fio)



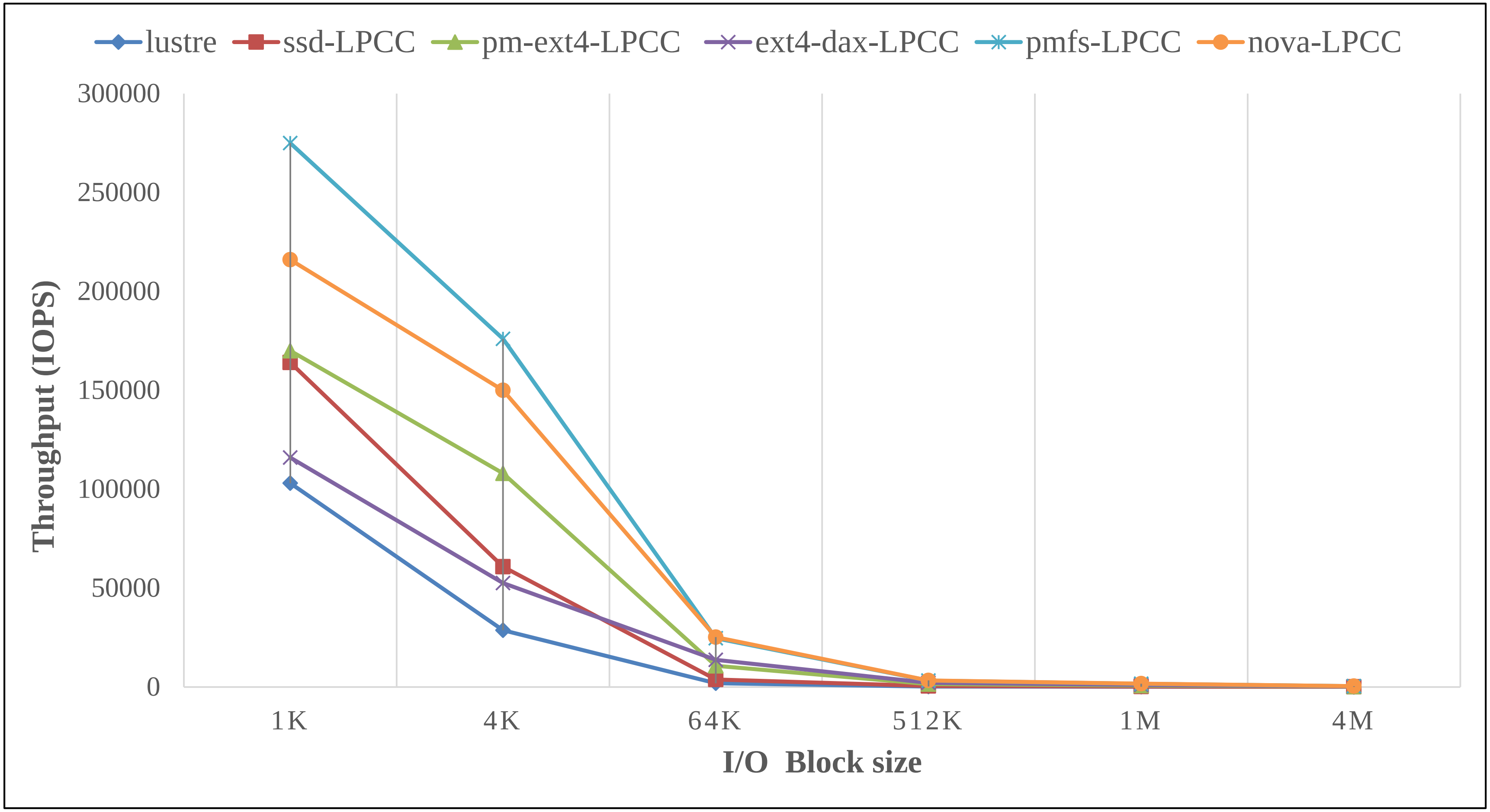
Single Client Read Performance (fio)



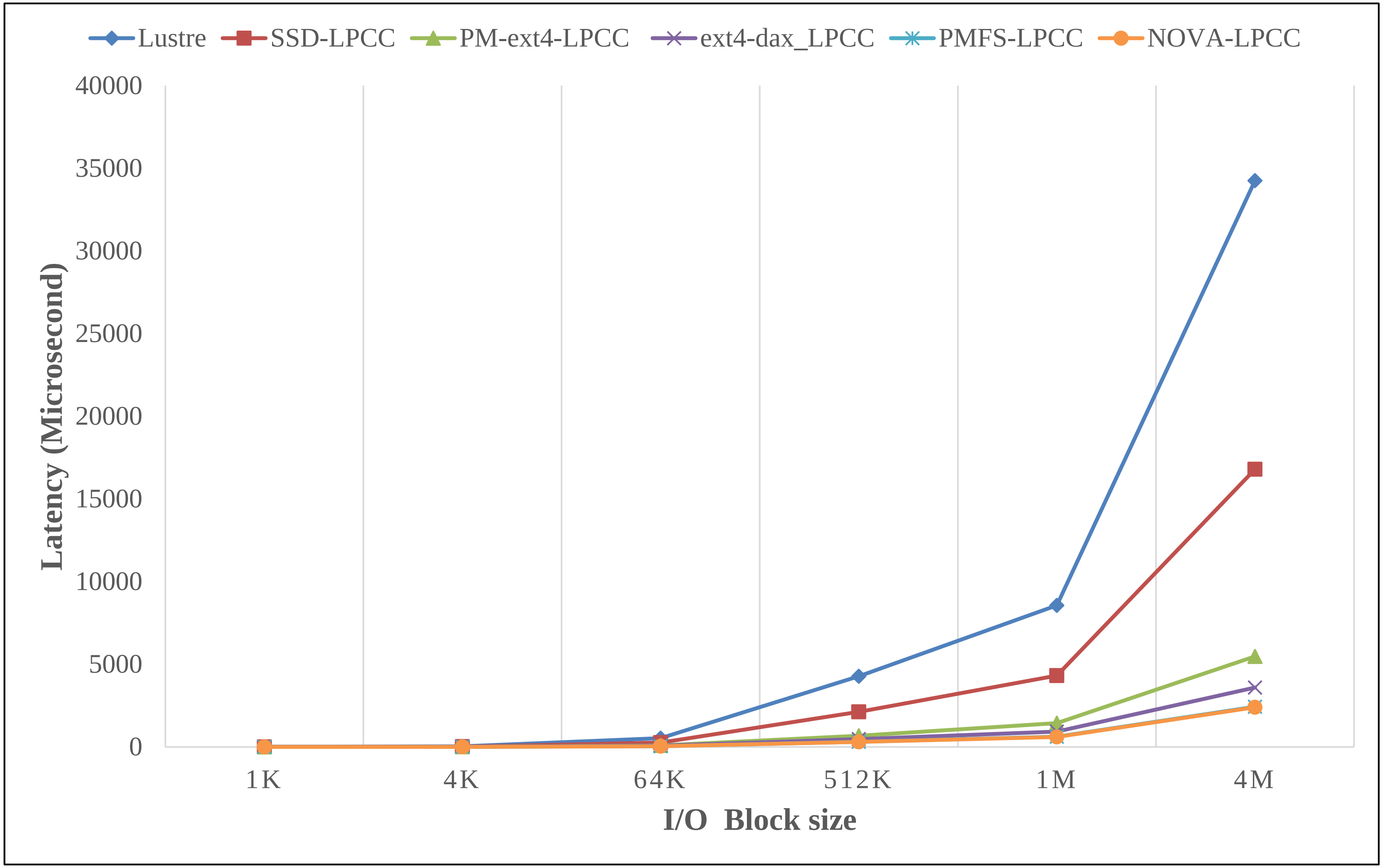
Single Client Write Performance (fio)



Single Client Write Performance (fio)

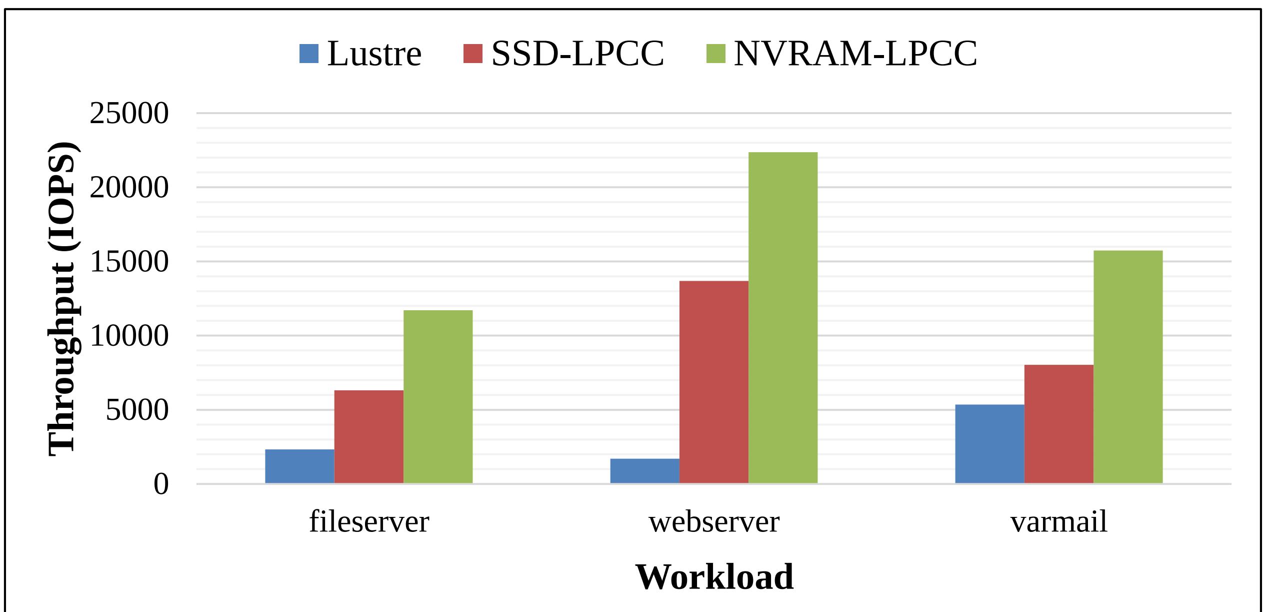


Single Client Write Performance (fio)



Write Performance (RW-PCC, filebench)

Workload	Average file size	Num. of Files	Read-write ratio
fileserver	128KB	0.3Million	1:2
webwerver	64KB	0.3Million	10:1
varmail	32KB	0.3Million	1:1

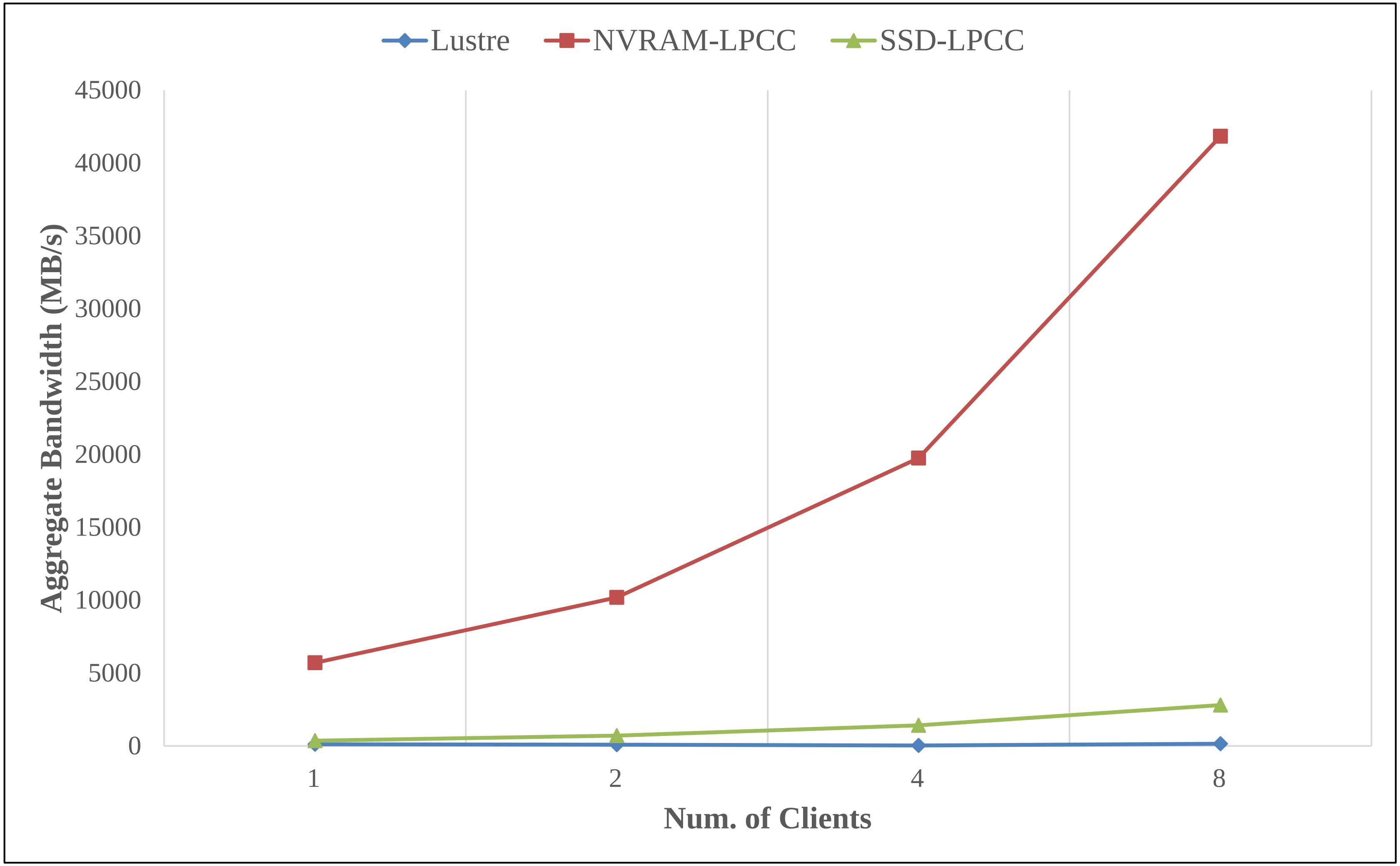


Fileserver:
NVRAM-LPCC / SSD-LPCC vs. Lustre
Throughput:
4x
2x

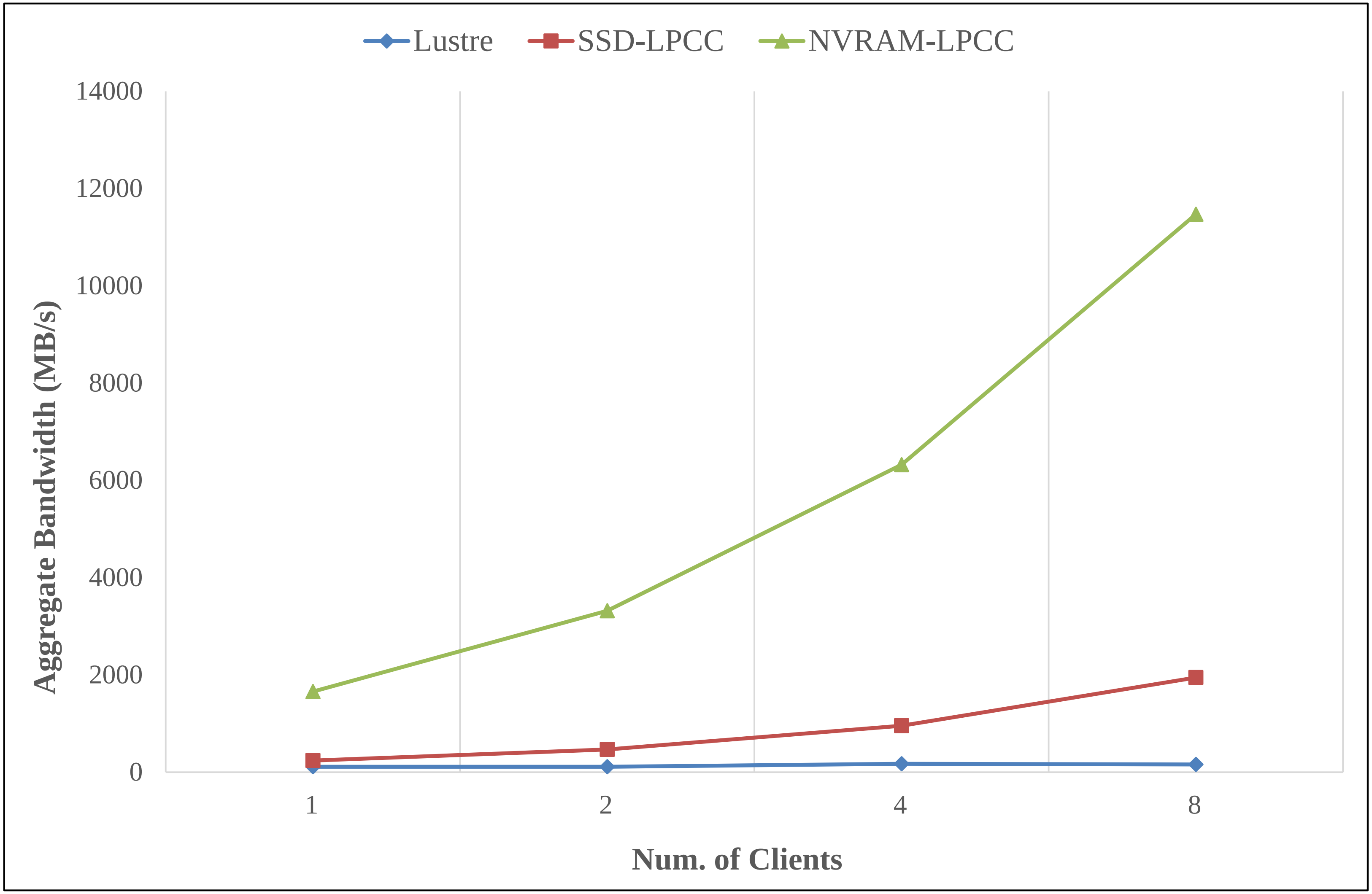
Webserver:
NVRAM-LPCC / SSD-LPCC
Vs. Lustre
9.2x
8.3x

Varmial:
NVRAM-LPCC / SSD-LPCC vs. Lustre
1.94x
1.5x

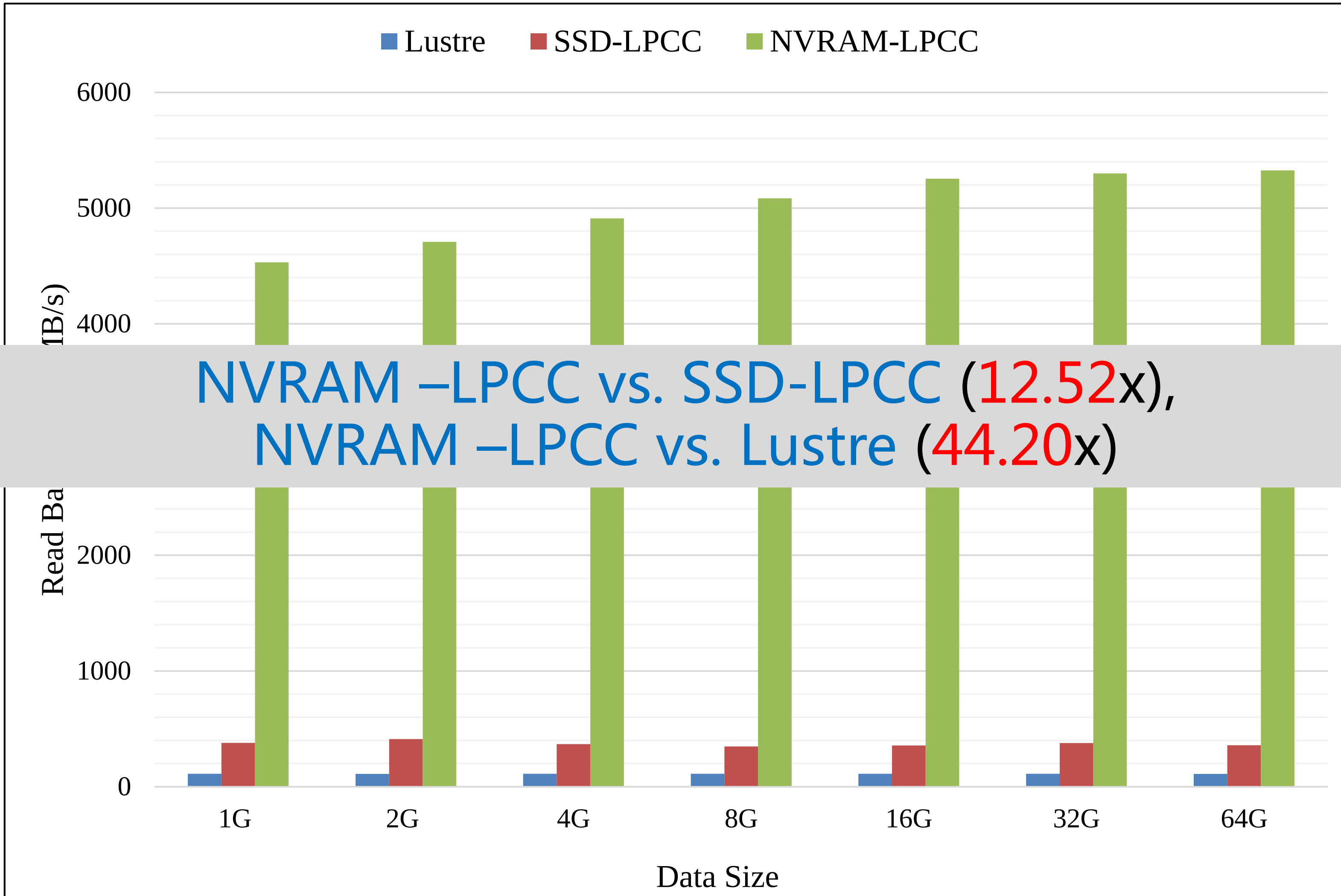
Read Scale Test (RW-PCC, IOR)



Write Scale Test (RW-PCC, IOR)

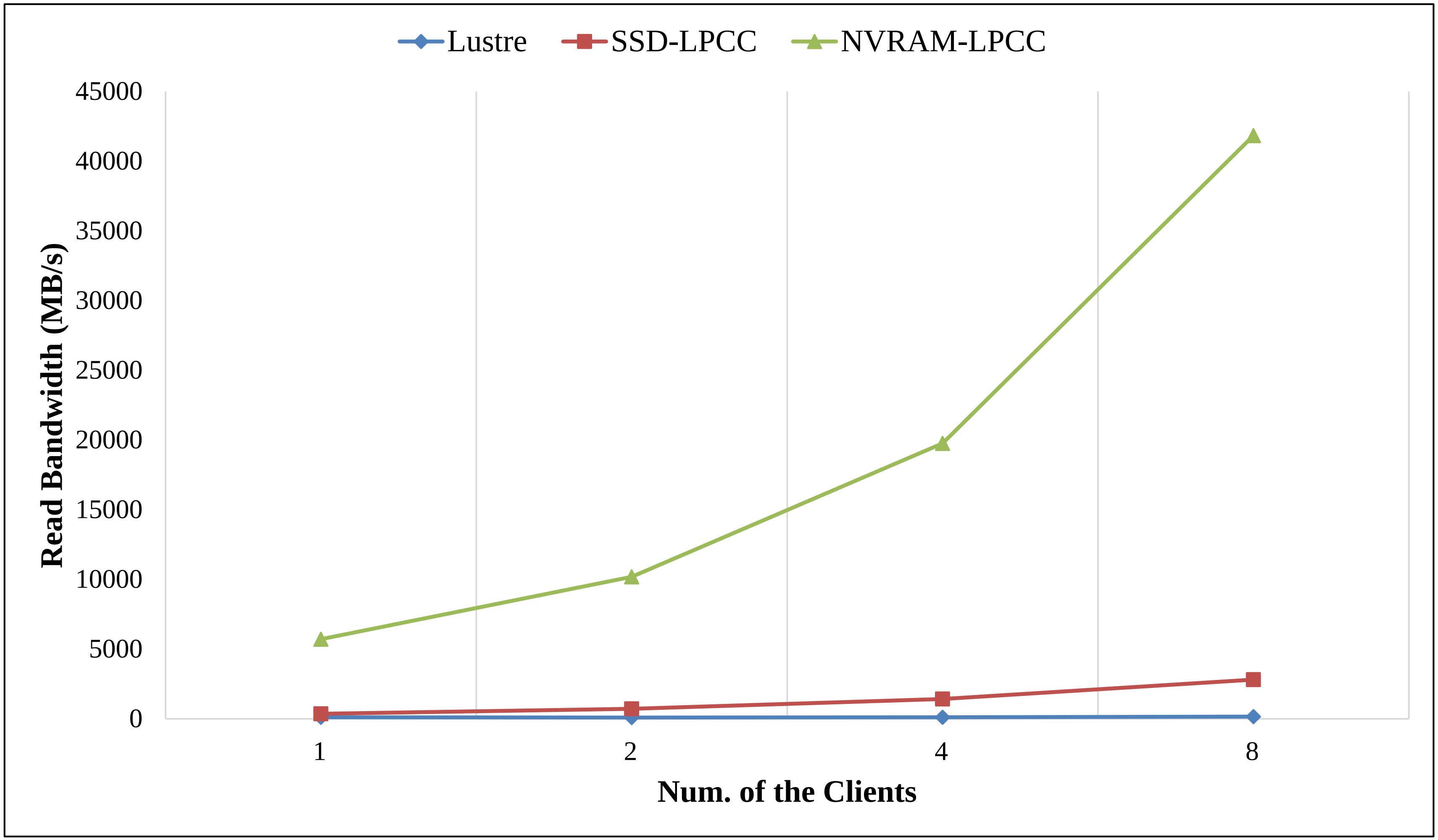


RO-PCC Scalability (I/O block size=1MB)



NVRAM –LPCC vs. SSD-LPCC (12.52x),
NVRAM –LPCC vs. Lustre (44.20x)

RO-PCC Scalability



Summary

- The performance of the cache medium has a great influence on the LPCC performance
  - Kingston SA400S37/240G SSD (proposed)
  - 512GB Samsung 840 PRO SSD (SC19)
- High performance based on NVRAM
  - Less overhead, and network latencies and lock conflicts significantly reduced
  - Reduce the pressure on the OSTs
  - No page cache
  - Optimized flush (Load/Store)
  - NVRAM-LPCC >> pm-ext4-LPCC

Thanks!

LAD19
Paris, France 23.09. - 25.09.19
LUSTRE ADMINS & DEVS WORKSHOP

NVRAM-oriented Lustre Persistent Cache on Client

Lingfang Zeng[*], Xi Li[#], and Wen Cheng[*]
[*]Wuhan National Laboratory for Optoelectronics (WNLO)
[*]Huazhong University of Science and Technology (HUST)
[#]DDN / Whamcloud