LocoFS

NHPCC Room 300 | Friday, December 8th, 2017 | Daniel Shao

Design and Implementation

Loosely-Coupled Arch.

Consists of: Client, DMS, FMS, Object Store
- DMS: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- FMS: File Metadata Server. Multiple

KV Pattern: HASHING
- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Flattened Directory Tree

Motivation: DESTROY directory tree
- Backward Directory Entry Organization
- Client Caching: only directories' metadata

Rename Discussion

Problem: hashing
- File: only metadata needs relocation
- Directory: its metadata as well as all successors' metadata need relocation.

Decoupled File Metadata

Motivation:
- Large-Value access
- (De)Serialization

Tech:
- Fine-grained File Metadata
- Indexing Metadata Removal
- (De)Serialization

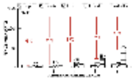
Motivation

Problem with FS Directory Tree in DFS

Motivation

Gap between FS Metadata and KV Store

Existing file systems have much lower performance than KV Stores:
- It has been confirmed that more than half operations are about metadata in file systems.
- KV Stores have great advantages on small objects



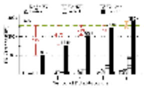
Q & A

Evaluation

Metadata Performance

- mdtest: 1 million files each time

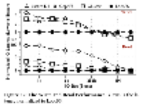
1. Latency
2. Throughput
3. Bridging gap



Evaluation

Full System Performance

Benchmark:
not mentioned



LocoFS

NHPCC Room 300 | Friday, December 8th, 2017 | Daniel Shao

Design and Implementation

Loosely-Coupled Arch.

Consists of: **Client, DMS, FMS, Object Store**
- **DMS**: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- **FMS**: File Metadata Server. Multiple

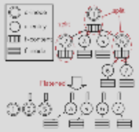
KV Pattern: **HASHING**
- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

Problem: hashing
- File: only metadata needs relocation
- Directory: its metadata as well as all successors' metadata need relocation.

Flattened Directory Tree

Motivation: DESTROY directory tree
- Backward Directory Entry Organization
- Client Caching: only directories' metadata



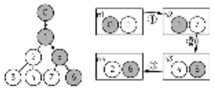
Decoupled File Metadata

Motivation:
- Large-Value access
- (De)Serialization

Tech:
- Fine-grained File Metadata
- Indexing Metadata Removal
- (De)Serialization

Motivation

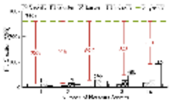
Problem with FS Directory Tree in DFS



Motivation

Gap between FS Metadata and KV Store

Existing file systems have much lower performance than KV Stores:
- It has been confirmed that more than half operations are about metadata in file systems.
- KV Stores have great advantages on small objects



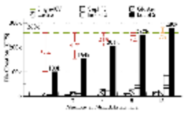
Q & A

Evaluation

Metadata Performance

- mdtest: 1 million files each time

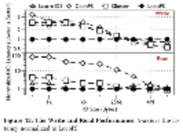
1. Latency
2. Throughput
3. Bridging gap



Evaluation

Full System Performance

Benchmark:
not mentioned



LocoFS

NHPCC Room 300 | Friday, December 8th, 2017 | Daniel Shao

Design and Implementation

Loosely-Coupled Arch.

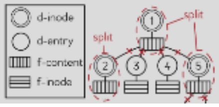
Consists of Client, DMS, FMS, Object Store

- DMS: Directory Metadata Server. Only 1
  - Enough to hold around 100 million

Flattened Directory Tree

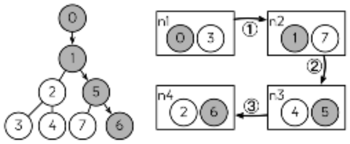
Motivation: DESTROY directory tree

- Backward Directory



Motivation

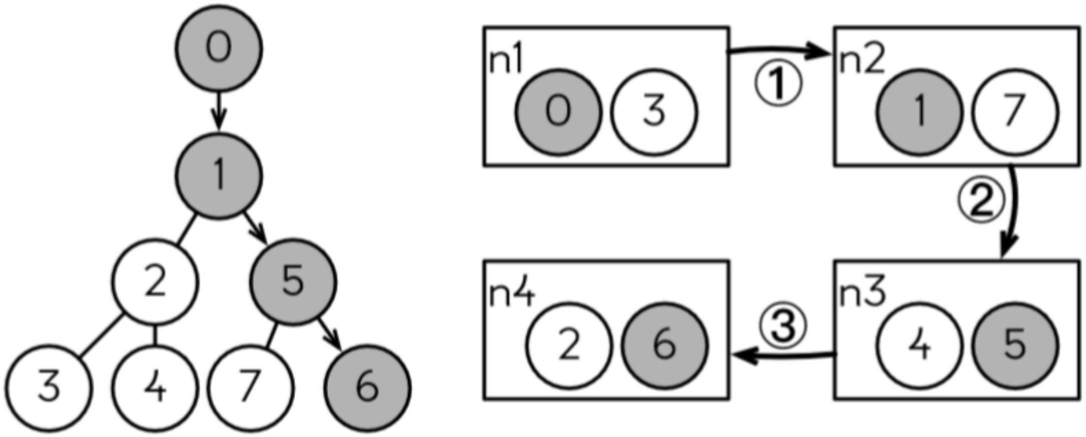
Problem with FS Directory Tree in DFS



Motivation

Motivation

Problem with FS Directory Tree in DFS

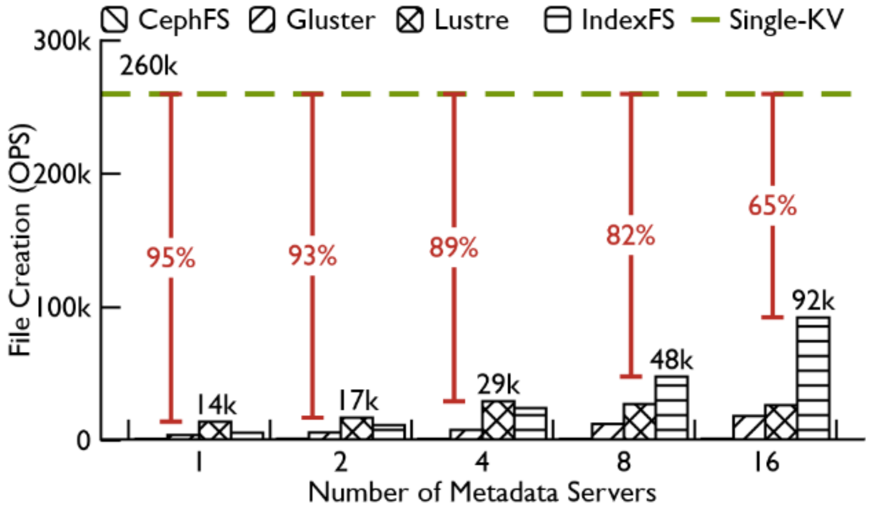


Motivation

Gap between FS Metadata and KV Store

Existing file systems have much lower performance than KV Stores:

- It has been confirmed that more than half operations are about metadata in file systems.
- KV Stores have great advantages on small objects



Design and Implementation

Loosely-Coupled Arch.

Consists of: **Client, DMS, FMS, Object Store**

- **DMS**: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- **FMS**: File Metadata Server. Multiple

KV Pattern: **HASHING**

- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

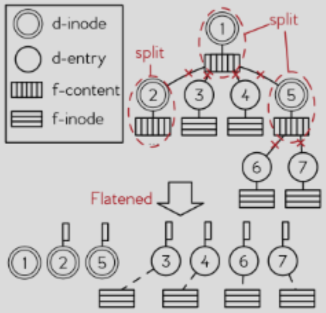
Problem: hashing

- File: only metadata needs relocation
- Directory: its metadata as well as all successors' metadata need relocation.

Flattened Directory Tree

Motivation: DESTROY directory tree

- Backward Directory Entry Organization
- Client Caching: only directories' metadata



Decoupled File Metadata

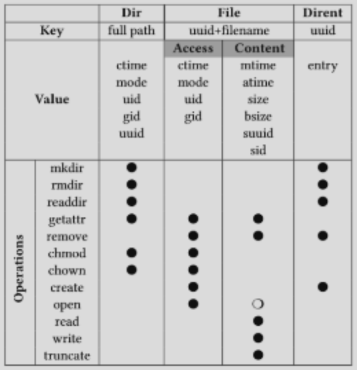
Motivation:

- Large-Value access
- (De)Serialization

Tech:

- Fine-grained File Metadata
- Indexing Metadata Removal
- (De)Serialization

Table 1: Metadata Access in Different File Operations



		Dir	**File**		**Dirent**
Key		full path	uuid+filename		uuid
			Access	**Content**	
Value		ctime mode uid gid uuid	ctime mode uid gid	mtime atime size bsize suuid sid	entry
Operations	mkdir	●			●
	rmdir	●			●
	readdir	●			●
	getattr	●	●	●	
	remove		●	●	●
	chmod	●	●		
	chown	●	●		
	create		●		●
	open		●	○	
	read			●	
	write			●	
	truncate			●	

● stands for field updating in an operation.
○ stands for optional field updating in an operation (different file system have different implementations).

Design and Implementat

Loosely-Coupled Arch.

Consists of: **Client, DMS, FMS, Object Store**

- **DMS**: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- **FMS**: File Metadata Server. Multiple

KV Pattern: **HASHING**

- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

Flatten

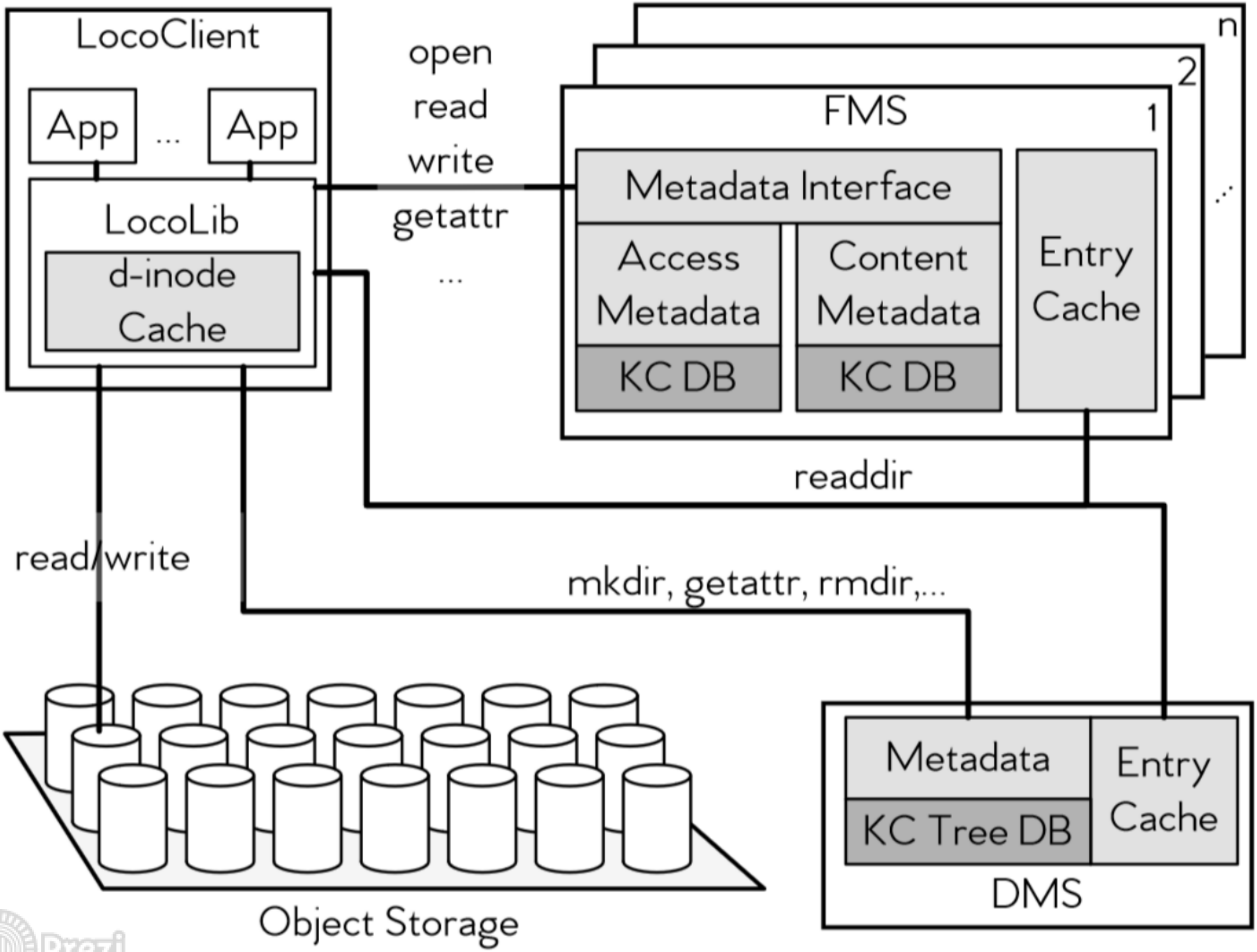
Motivation:
directory tre

- Backwar
  Entry Or
- Client C
  directori

Decou

Motivation:

- Large-V


LocoClient
App
...
App
LocoLib
d-inode Cache
open
read
write
getattr
...
FMS
1
2
n
...
Metadata Interface
Access Metadata
KC DB
Content Metadata
KC DB
Entry Cache
readdir
read/write
mkdir, getattr, rmdir,...
Metadata
KC Tree DB
Entry Cache
DMS
Object Storage

Design and Implementati

Loosely-Coupled Arch.

Consists of: **Client, DMS, FMS, Object Store**

- **DMS**: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- **FMS**: File Metadata Server. Multiple

KV Pattern: **HASHING**

- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

Flattene

Motivation: D
directory tree

- Backward
  Entry Orga
- Client Cac
  directories'

Decoupl

Motivation:

- Large-Valu
  (De)Serial

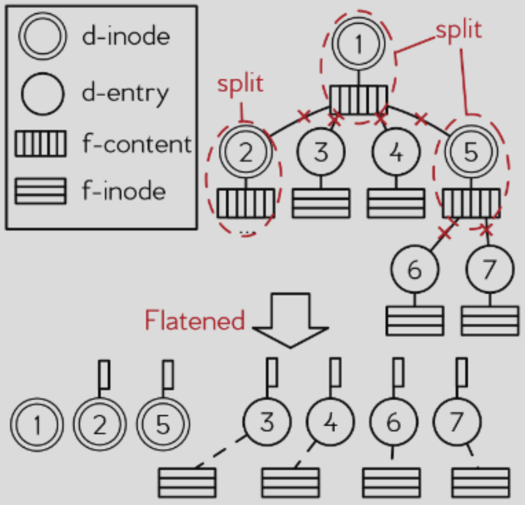
ementation

on

Flattened Directory Tree

Motivation: DESTROY directory tree

- Backward Directory Entry Organization
- Client Caching: only directories' metadata



Decoupled File Metadata

Mot

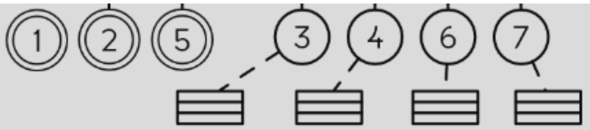
Pro

Mot

Gap

Existin
perforn
- It h
thar

directories' metadata



Decoupled File Metadata

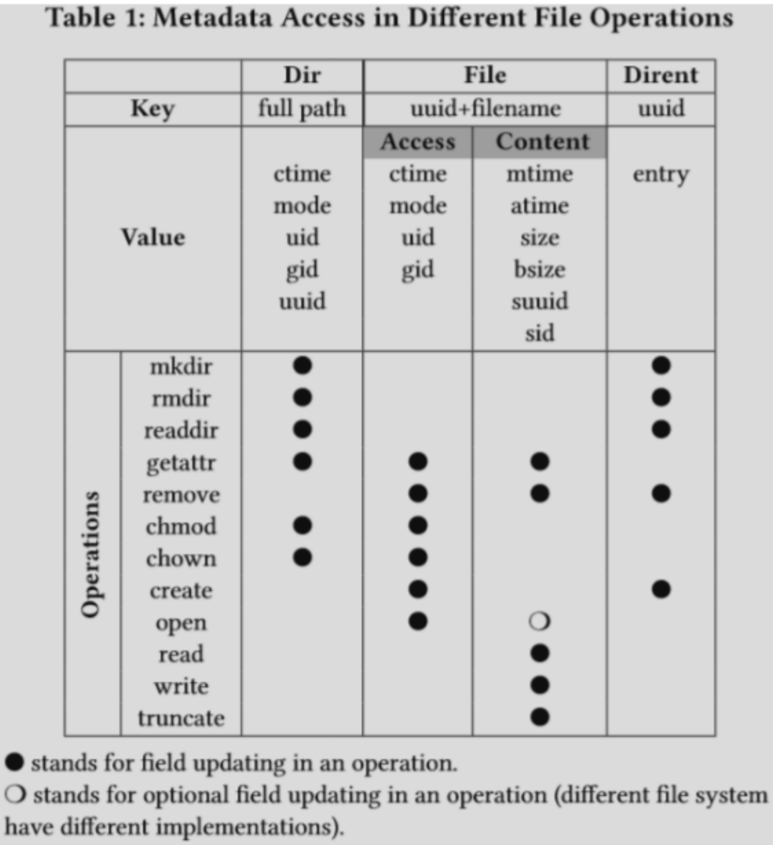


Table 1: Metadata Access in Different File Operations

		Dir	File		Dirent
Key		full path	uuid+filename		uuid
			Access	Content	
Value		ctime mode uid gid uuid	ctime mode uid gid	mtime atime size bsize suuid sid	entry
Operations	mkdir	●			●
	rmdir	●			●
	readdir	●			●
	getattr	●	●	●	
	remove		●	●	●
	chmod	●	●		
	chown	●	●		
	create		●		●
	open		●	○	
	read			●	
	write			●	
	truncate			●	

● stands for field updating in an operation.
○ stands for optional field updating in an operation (different file system have different implementations).

Motivation:

- Large-Value access
- (De)Serialization

Tech:

- Fine-grained File Metadata
- Indexing Metadata Removal
- (De)Serialization

- Enough to hold around 100 million directories in 32GB memory.
- Simple ACL Management.

- **FMS**: File Metadata Server. Multiple

KV Pattern: **HASHING**

- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

Problem: hashing

- File: only metadata needs relocation
- Directory: its metadata as well as all successors' metadata need relocation.

- Backward Di
Entry Organiza
- Client Caching
directories' me

Decouple

Motivation:

- Large-Value a
- (De)Serializat

Tech:

- Fine-grained Metadata
- Indexing Meta Removal
- (De)Serializat

Evaluation

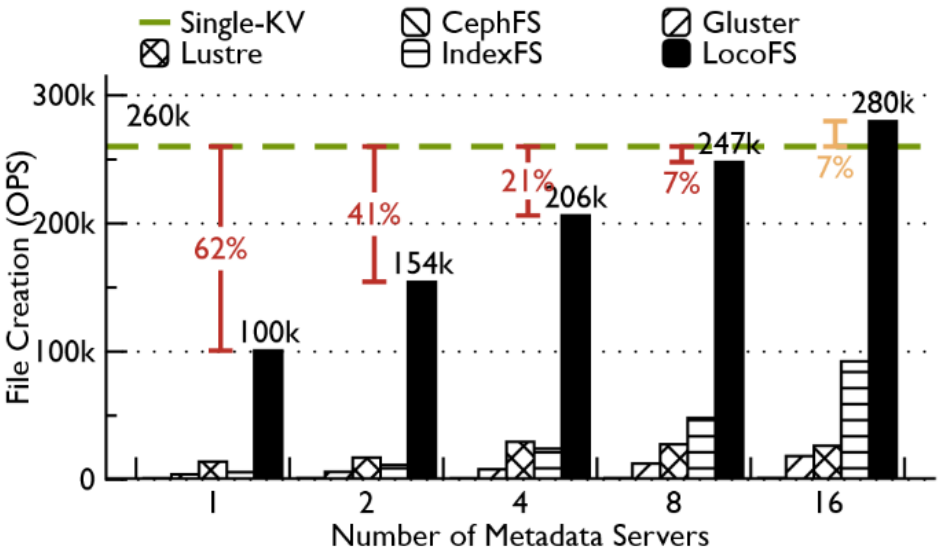
Evaluation

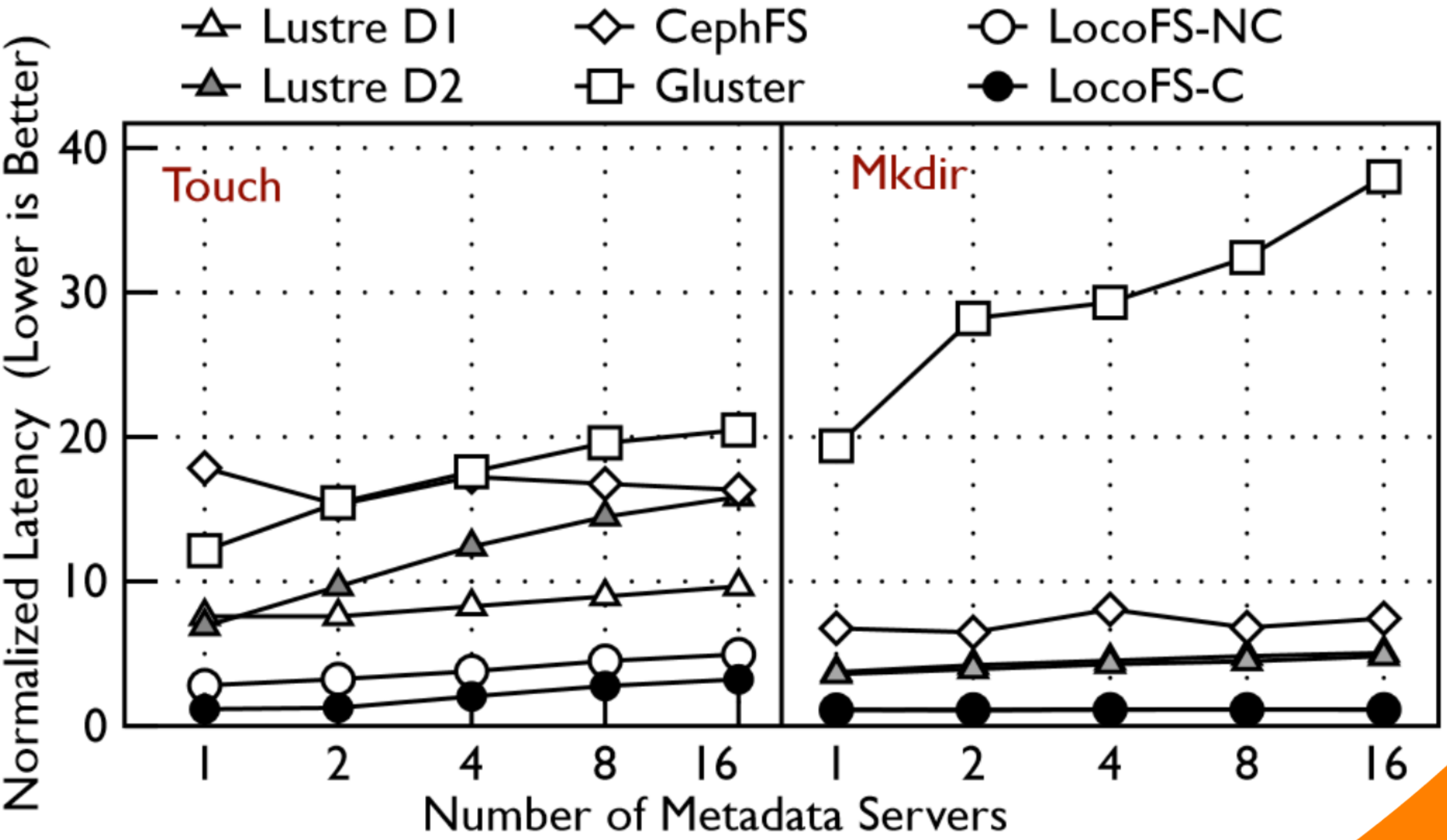
Evaluation

Metadata Performance

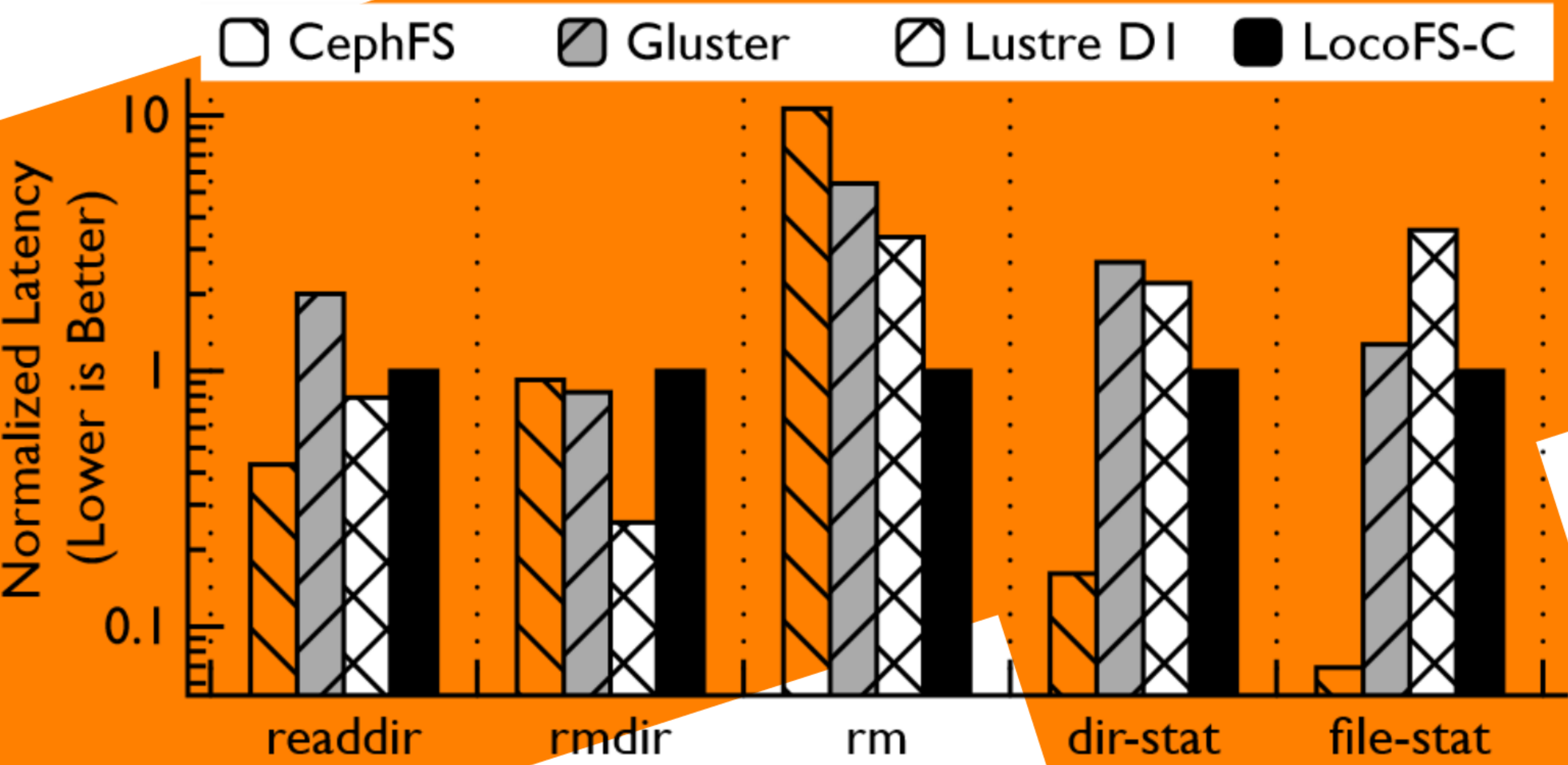
- mdtest: 1 million files each time

1. Latency
2. Throughput
3. Bridging gap





Lustre D1
Lustre D2
CephFS
Gluster
LocoFS-NC
LocoFS-C
Touch
Mkdir
Normalized Latency (Lower is Better)
0
10
20
30
40
1
2
4
8
16
1
2
4
8
16
Number of Metadata Servers

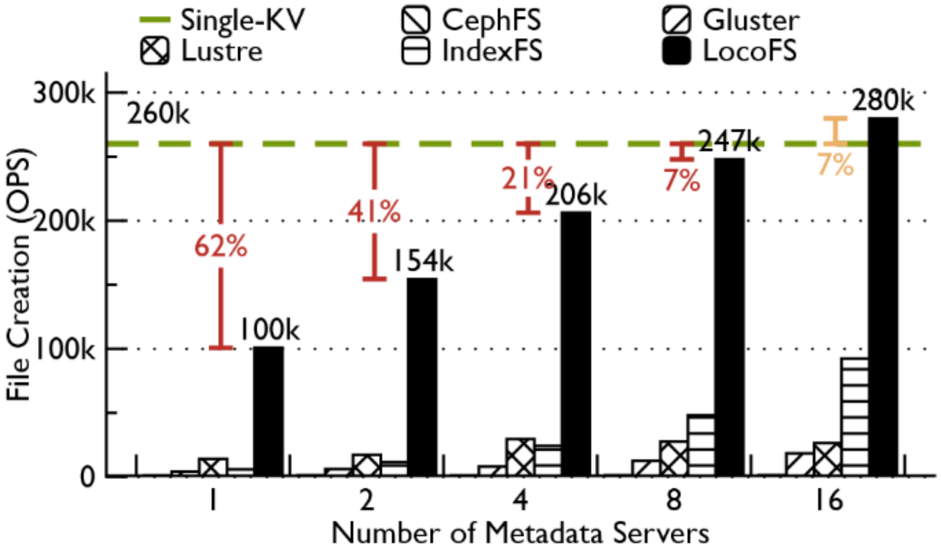

CephFS
Gluster
Lustre D1
LocoFS-C
Normalized Latency
(Lower is Better)
10
1
0.1
readdir
rmdir
rm
dir-stat
file-stat

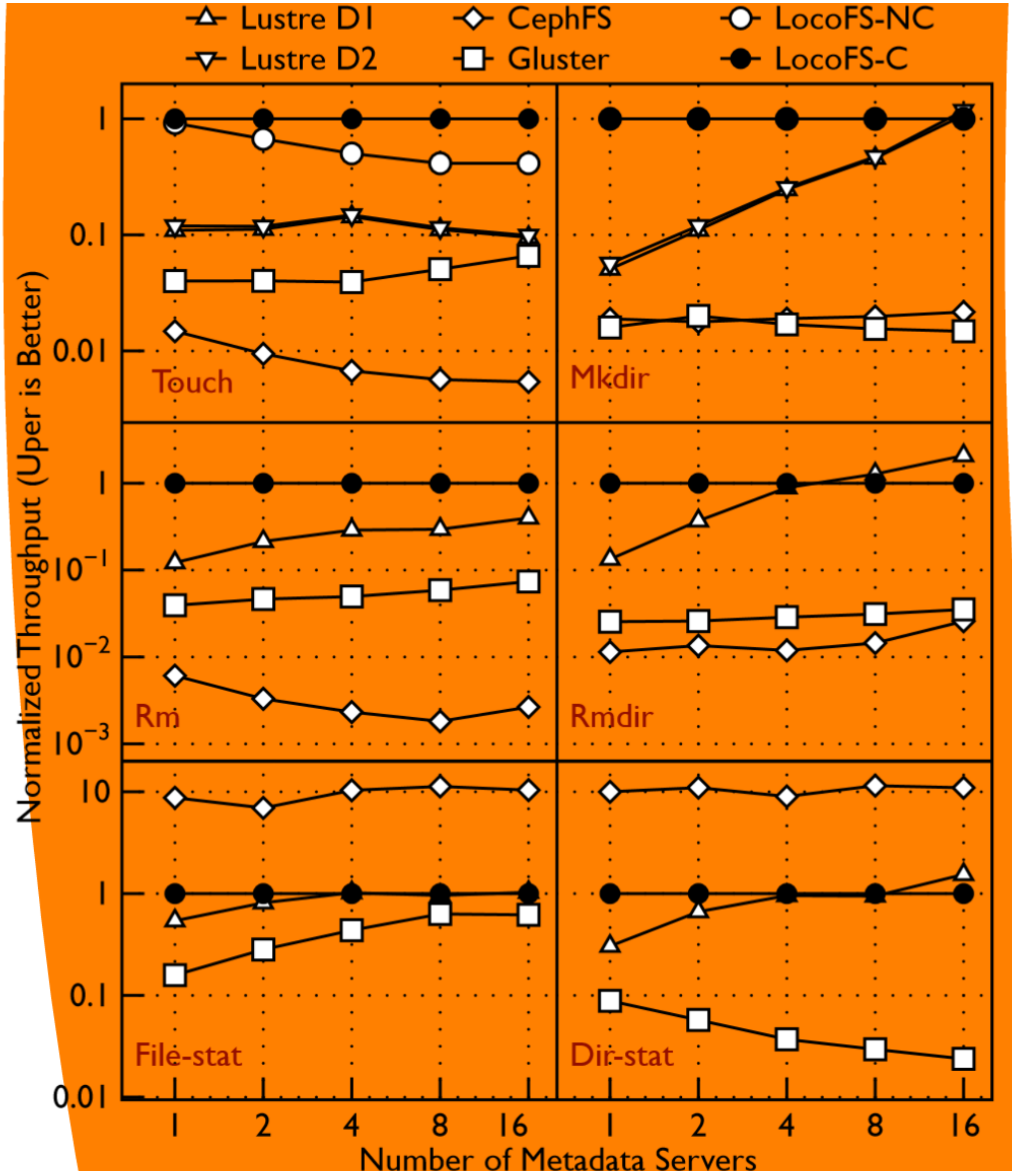
Evaluation

Metadata Performance

- mdtest: 1 million files each time

1. Latency
2. Throughput
3. Bridging gap



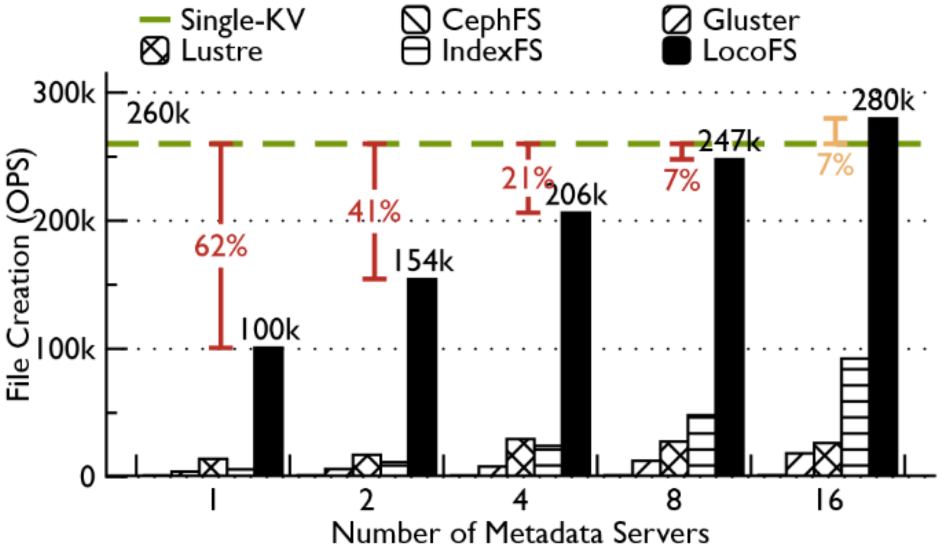

Lustre D1
CephFS
LocoFS-NC
Lustre D2
Gluster
LocoFS-C
Normalized Throughput (Uper is Better)
Touch
Mkdir
Rm
Rmdir
File-stat
Dir-stat
1
0.1
0.01
10^{-1}
10^{-2}
10^{-3}
10
1
2
4
8
16
Number of Metadata Servers

Evaluation

Metadata Performance

- mdtest: 1 million files each time

1. Latency
2. Throughput
3. Bridging gap



Evaluation

Full System Performance

Benchmark:

not mentioned

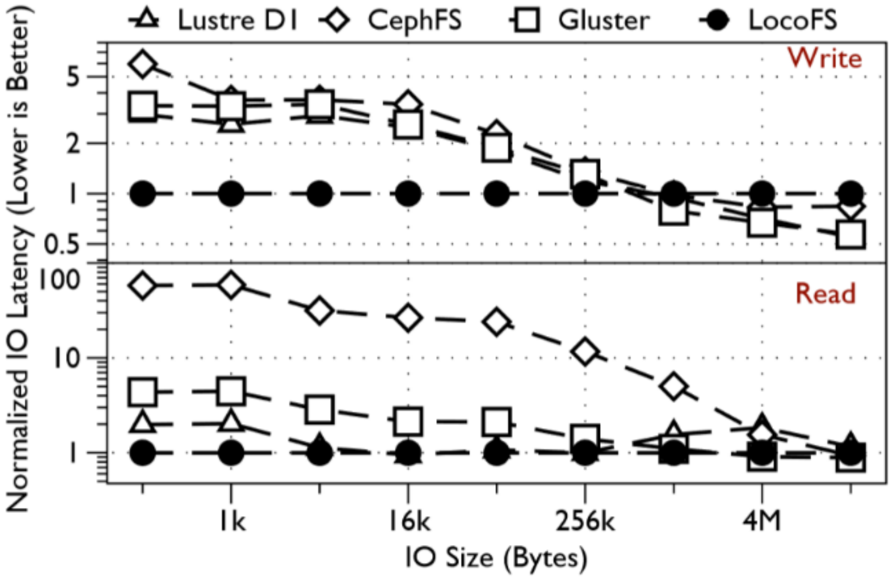


Figure 12: The Write and Read Performance. Y-axis is the latency normalized to LocoFS.

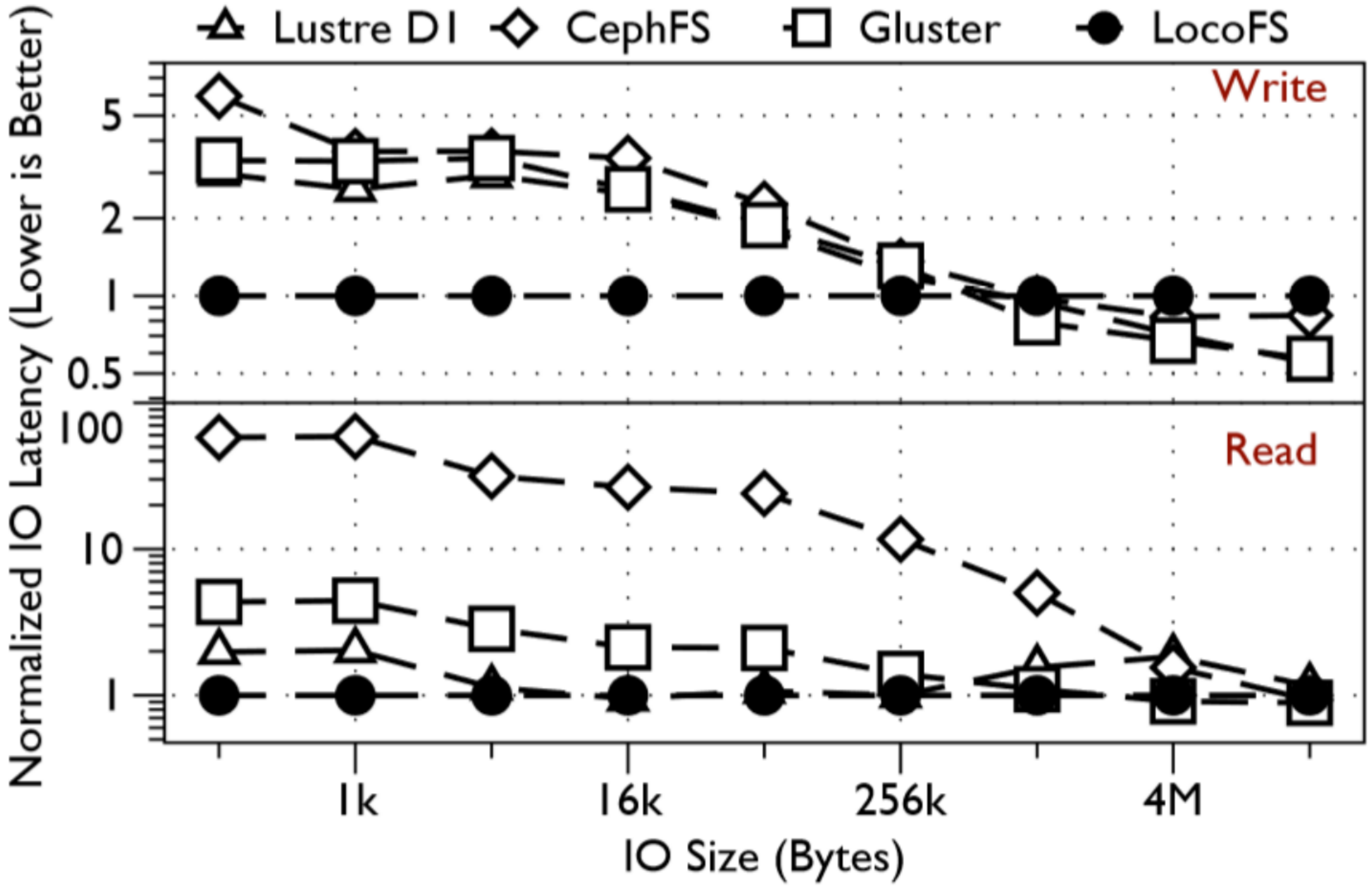


Figure 12: The Write and Read Performance. Y-axis is the latency normalized to LocoFS.

Evaluation

Full System Performance

Benchmark:
not mentioned

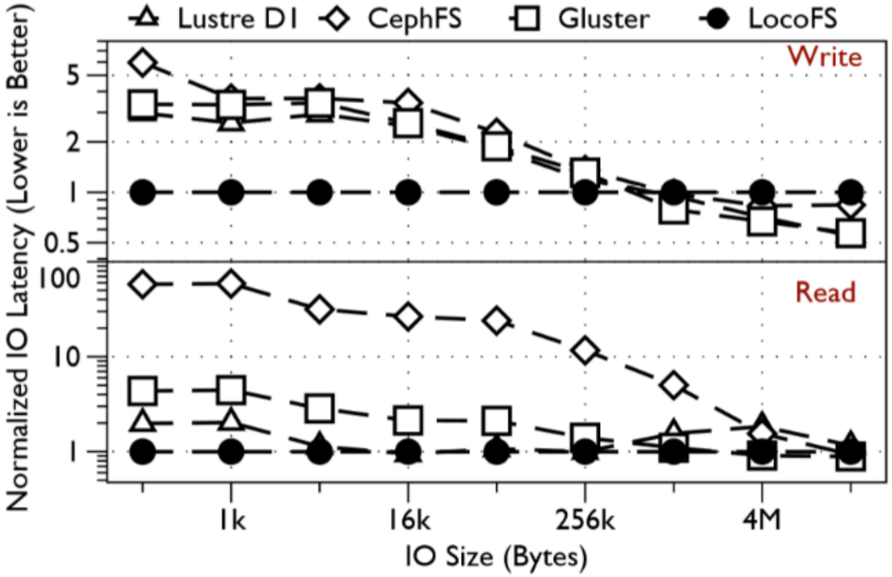
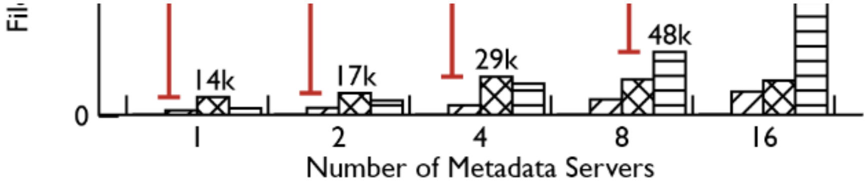


Figure 12: The Write and Read Performance. Y-axis is the latency normalized to LocoFS.

- KV Stores have great advantages on small objects



Q & A

LocoFS

NHPCC Room 300 Friday, December 8th, 2017 Daniel Shao

Design and Implementation

Loosely-Coupled Arch.

Consists of: Client, DMS, FMS, Object Store
- DMS: Directory Metadata Server. Only 1
  - Enough to hold around 100 million directories in 32GB memory.
  - Simple ACL Management.
- FMS: File Metadata Server. Multiple

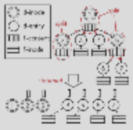
KV Pattern: HASHING
- DMS: full pathname → directory metadata
- FMS: dir_uuid+filename → file metadata

Rename Discussion

Problem: hashing
- File: only metadata needs relocation
- Directory: its metadata as well as all successors' metadata need relocation.

Flattened Directory Tree

Motivation: DESTROY directory tree
- Backward Directory Entry Organization
- Client Caching: only directories' metadata



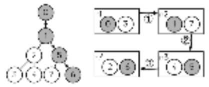
Decoupled File Metadata

Motivation:
- Large-Value access
- (De)Serialization

Tech:
- Fine-grained File Metadata
- Indexing Metadata Removal
- (De)Serialization

Motivation

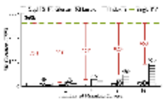
Problem with FS Directory Tree in DFS



Motivation

Gap between FS Metadata and KV Store

Existing file systems have much lower performance than KV Stores:
- It has been confirmed that more than half operations are about metadata in file systems.
- KV Stores have great advantages on small objects



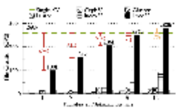
Q & A

Evaluation

Metadata Performance

- mdtest: 1 million files each time

1. Latency
2. Throughput
3. Bridging gap



Evaluation

Full System Performance

Benchmark:
not mentioned