FAST'25 FEBRUARY 25-27, 2025
SANTA CLARA, CA

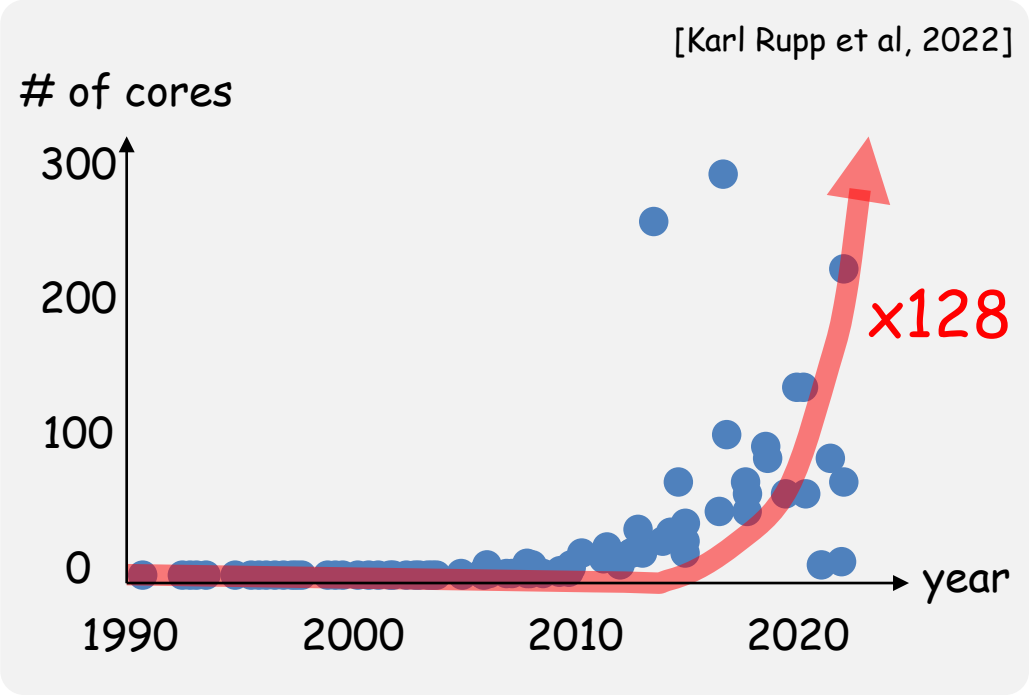
DJFS : Directory-Granularity Filesystem Journaling for CMM-H SSDs

Seung Won Yoo[1] Joontaek Oh[2] Myoengin Cheon[1] Bonmoo Koo[1]
Wonseb Jeong[3] Hyunsub Song[3] Hyeonho Song[3] Donghun Lee[3] YoujipWon[1]

[1]KAIST [2]University of Wisconsin-Madison [3]Samsung Electronics

Hardware Trends

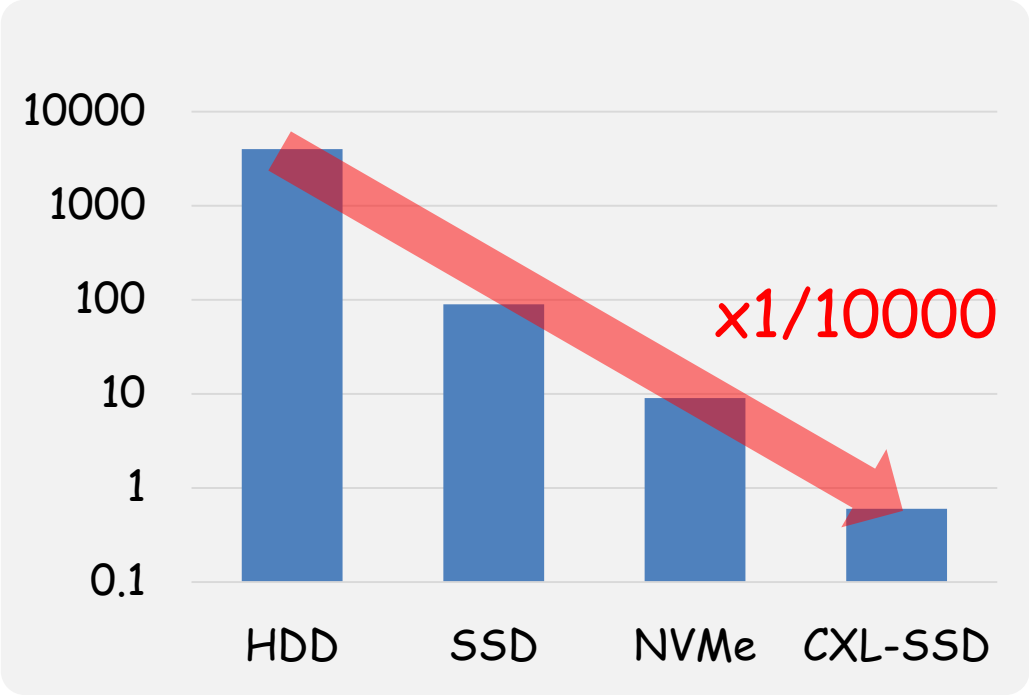
of cores is increasing!!



1990: Single Core (80386)

→ 2024: 128 Cores (Granite Rapids)

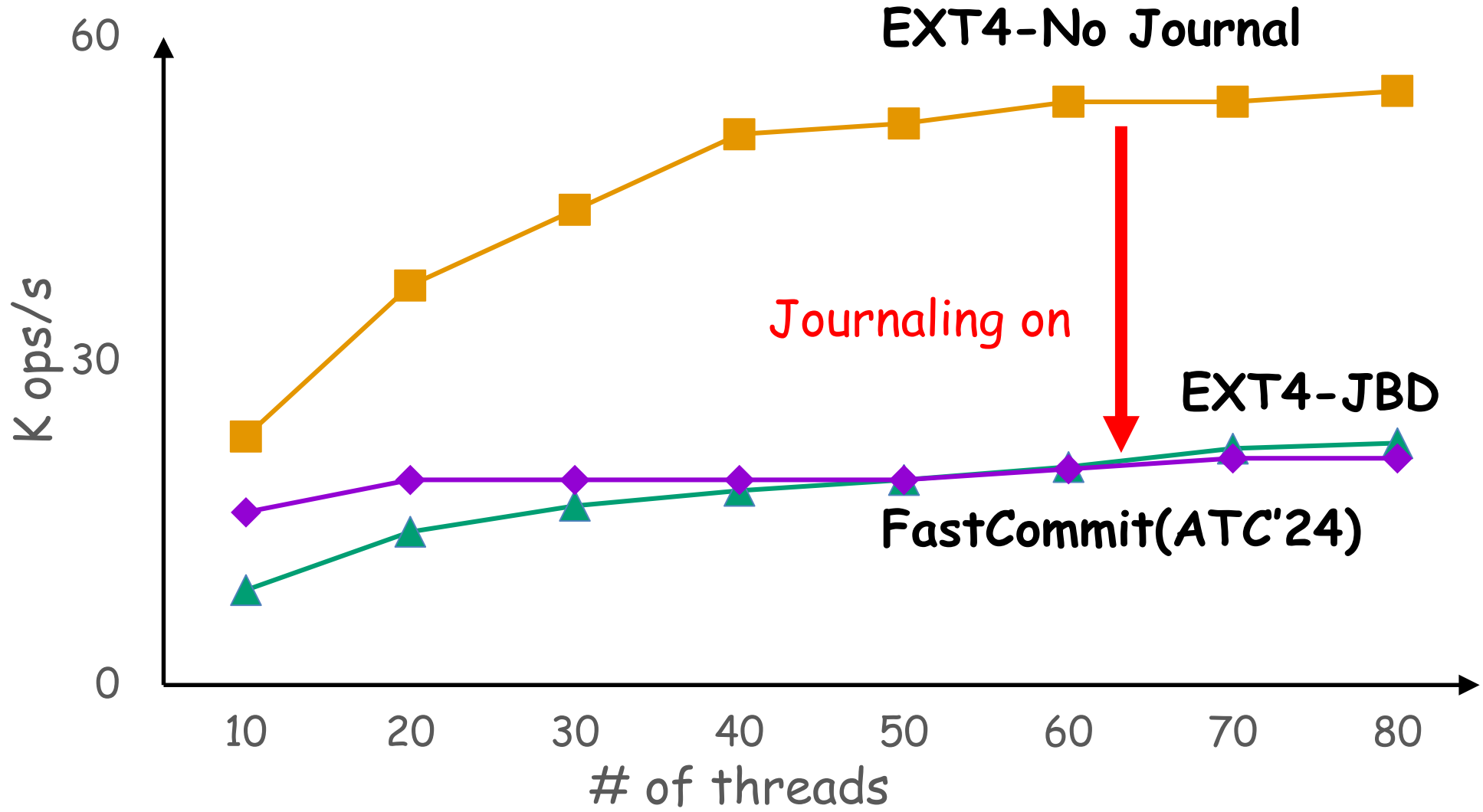
IO Latency is decreasing!!



2010: 4ms (HDD, WDVR200M)

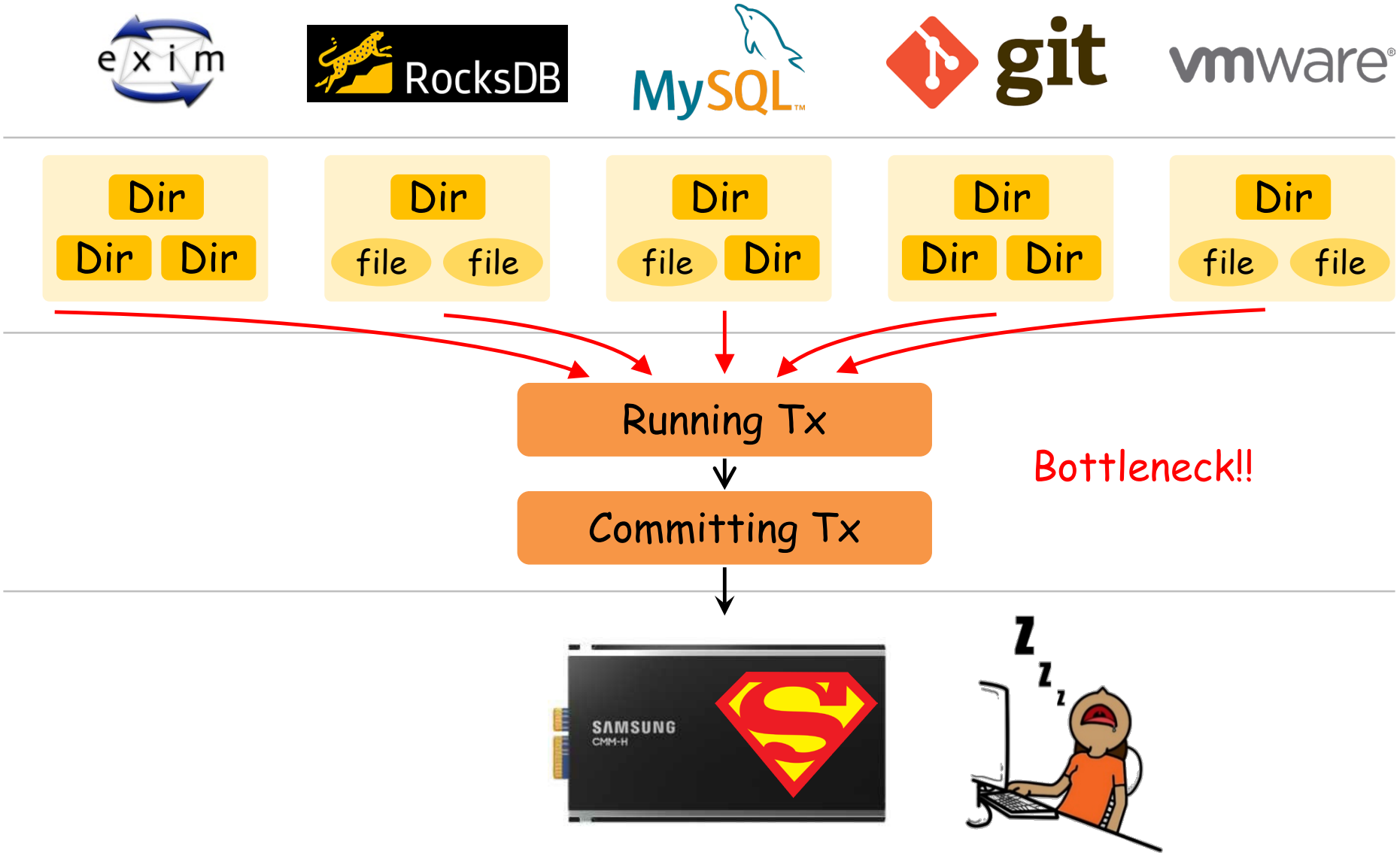
→ 2024: 0.6us (CMM-H)

Journaling Overhead

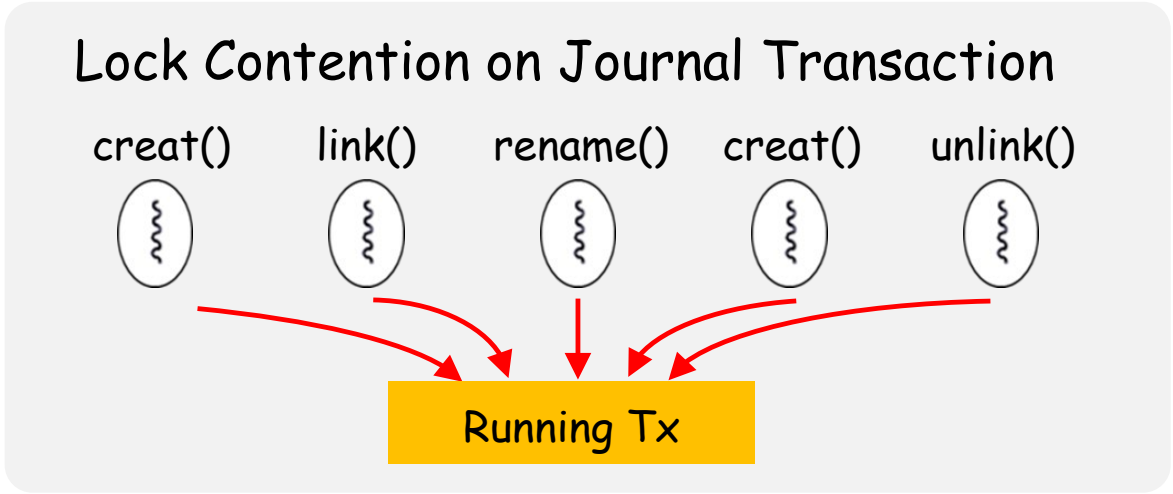


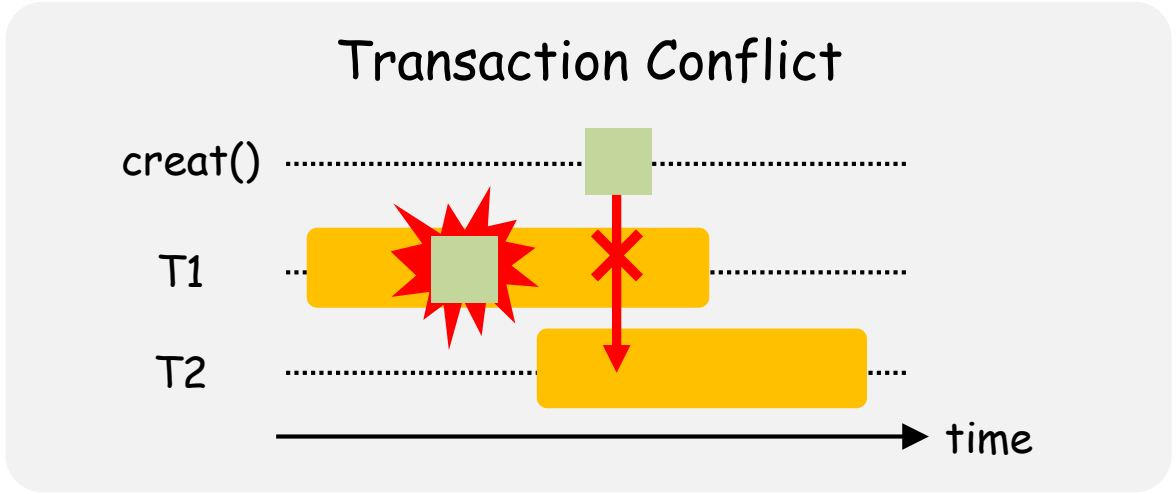
(SPR 88 Core, CMM-H, MDTest - file creation)

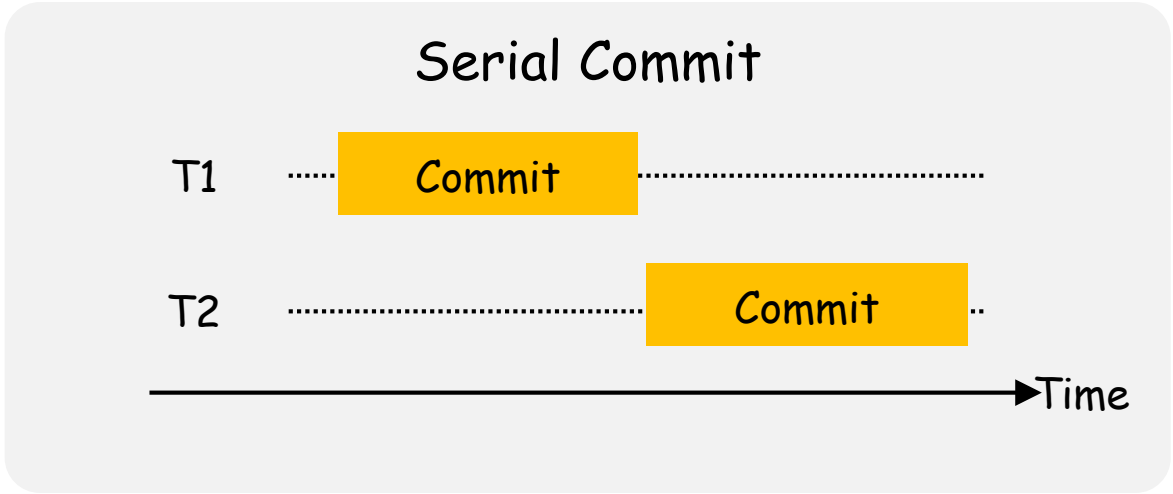
Journaling as a performance bottleneck

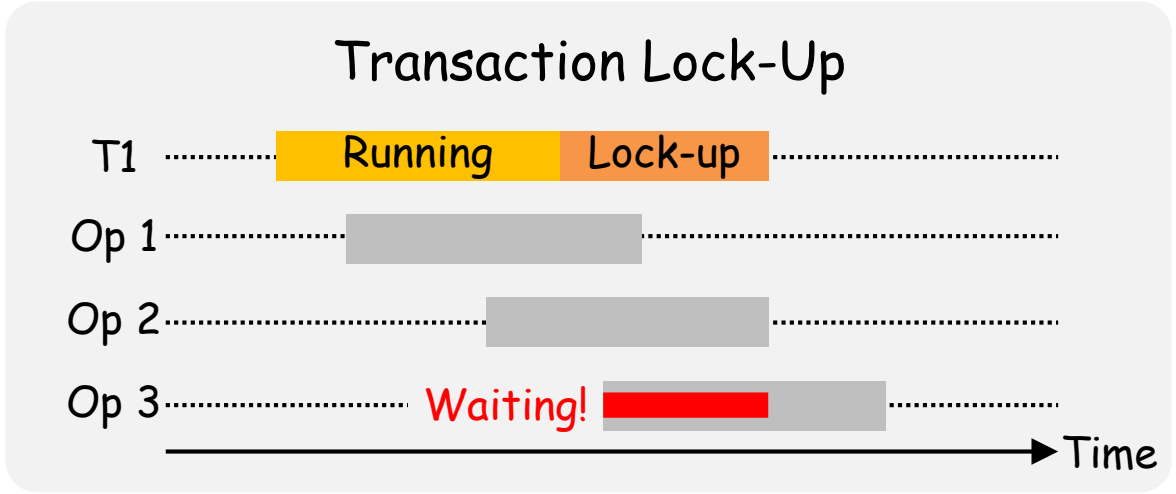


Main reasons for journaling bottleneck : EXT4

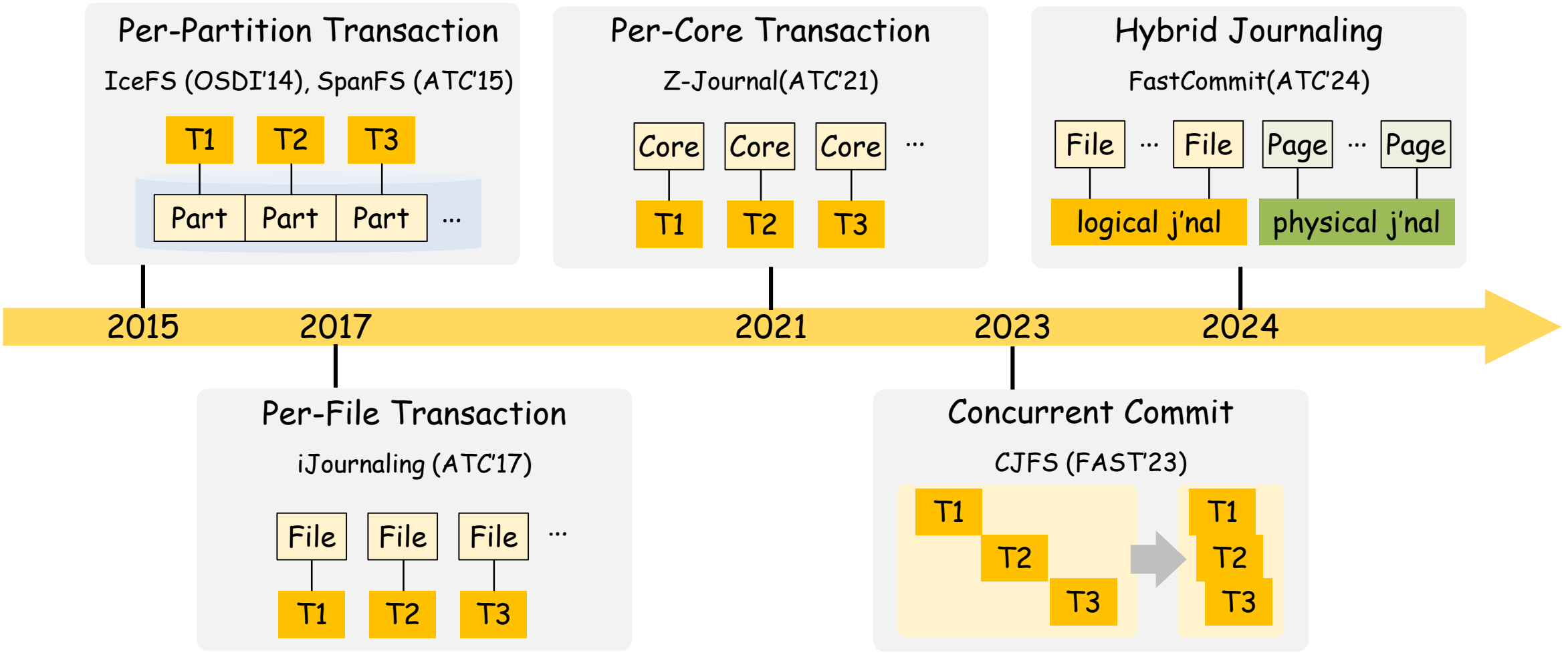




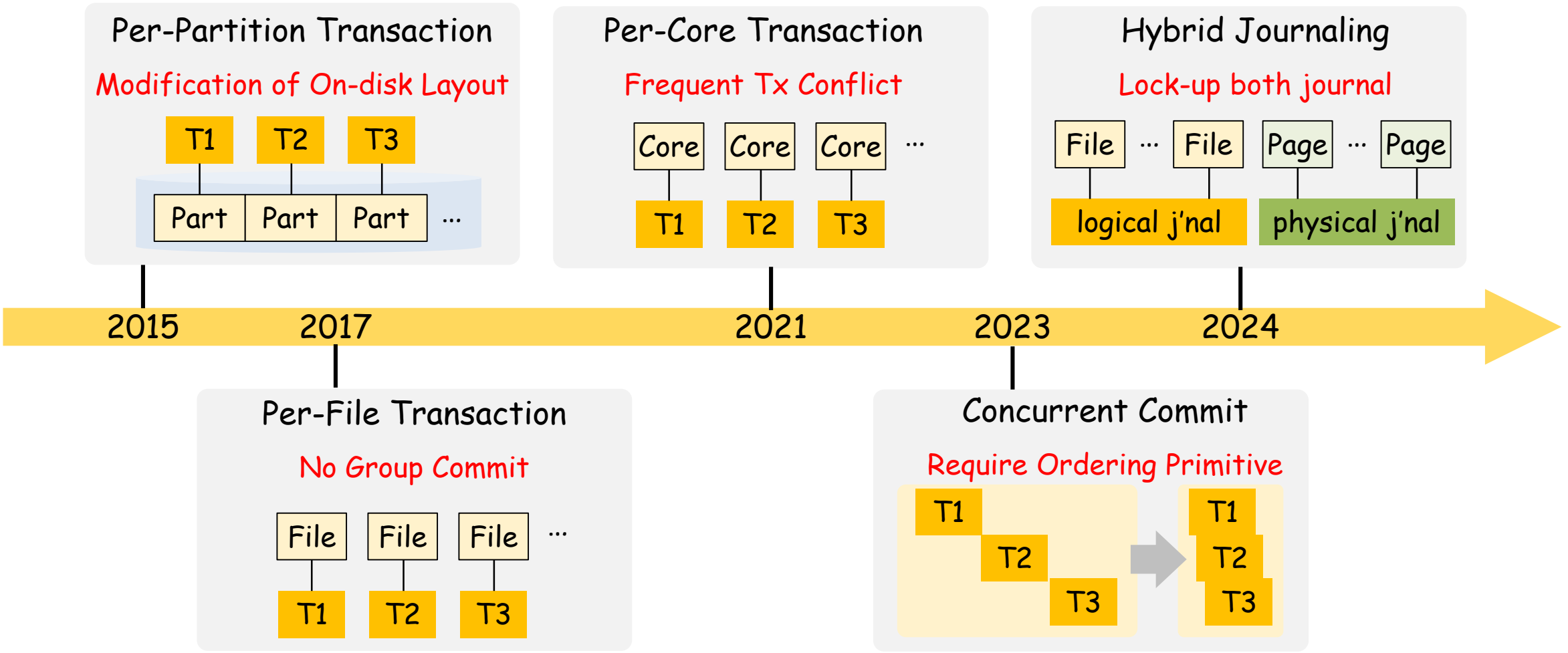




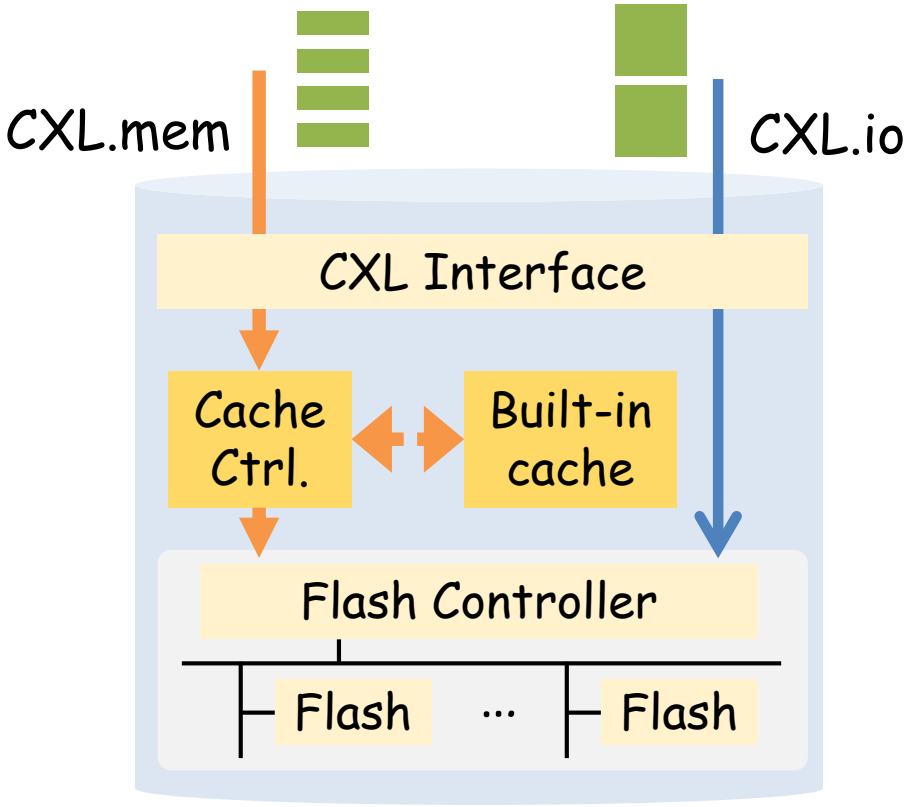
Existing Works for Journaling Scalability

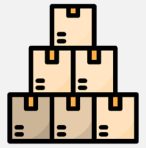


Existing Works for Journaling Scalability



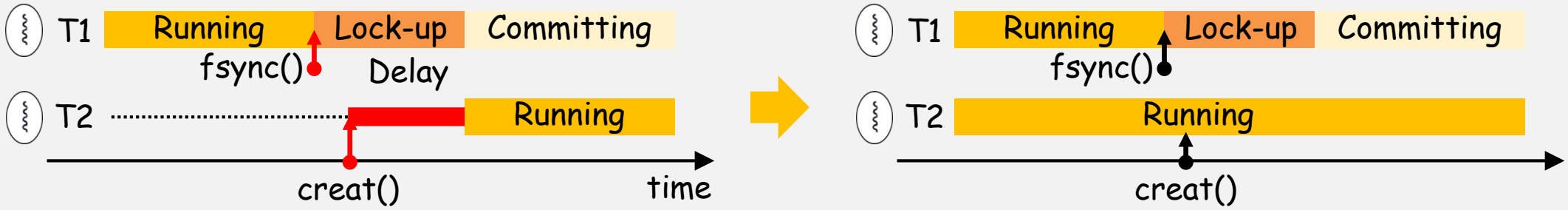
CMM-H : The Memory Semantic SSD



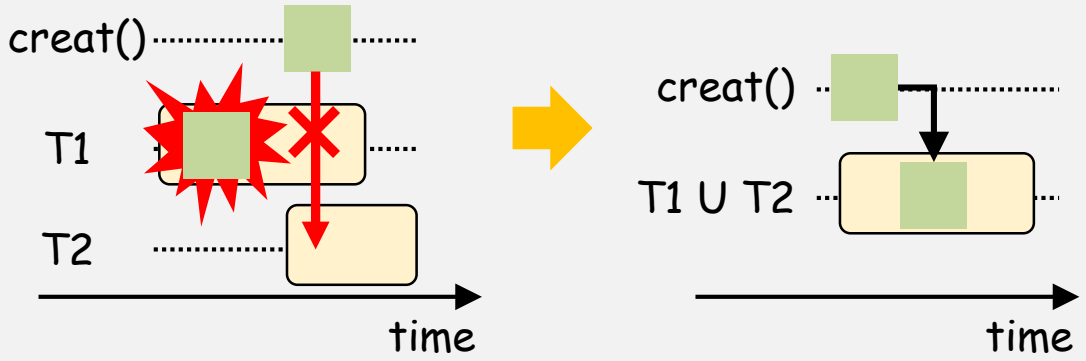
	CXL.io	CXL.mem
Analogous to		
Bandwidth		
Latency		
Use Case	User Data	Metadata Logging

Objectives for Scalable Journaling

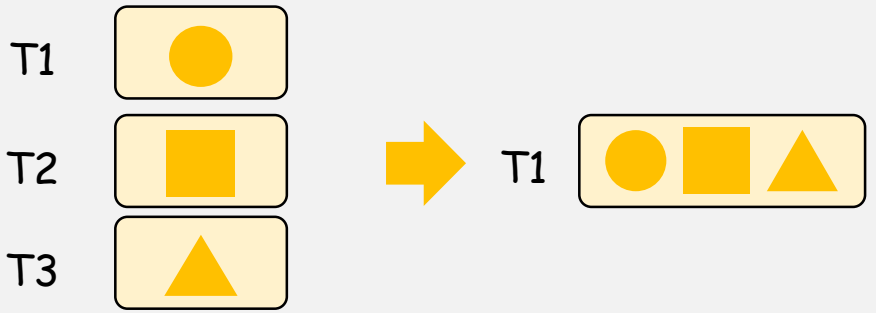
Minimize Transaction Lock-up Overhead



Minimize Transaction Conflicts



Maximize Compound Degree



File Updates in Modern Applications

We analyze the file update patterns in eight popular applications.


```
1. creat(/d/spool_data);
2. append(/d/spool_data);
3. fsync(/d/spool_data);
4. creat(/d/spool_hdr_tmp);
5. append(/d/spool_hdr_tmp);
6. fsync(/d/spool_hdr_tmp);
7. rename(/d/spool_hdr_tmp, spool_hdr);
```

Exim) journaling

```
1. while (/d/sst[@]) {
2.   creat(/d/newsst[@]);
3.   append(/d/newsst[@]);
4.   fsync(/d/newsst[@]);
5. }
6. fsync(/d);
7. unlink(/d/sst[@]);
8. append(/d/MANIFEST);
9. fsync(/d/MANIFEST);
```

RocksDB) Compaction

Dedicated Directory (Property D)

An application works on its own **dedicated directory**.


```
1. creat(/d/spl_data);
2. append(/d/spl_data);
3. fsync(/d/spl_data);
4. creat(/d/spl_hdr_tmp);
5. append(/d/spl_hdr_tmp);
6. fsync(/d/spl_hdr_tmp);
7. rename(/d/spl_hdr_tmp,/d/spl_hdr);
```

Exim

/spool

log

/mail

mail

RocksDB

/rocks

sst

wal

MySQL

/mysql

binlog

/redo

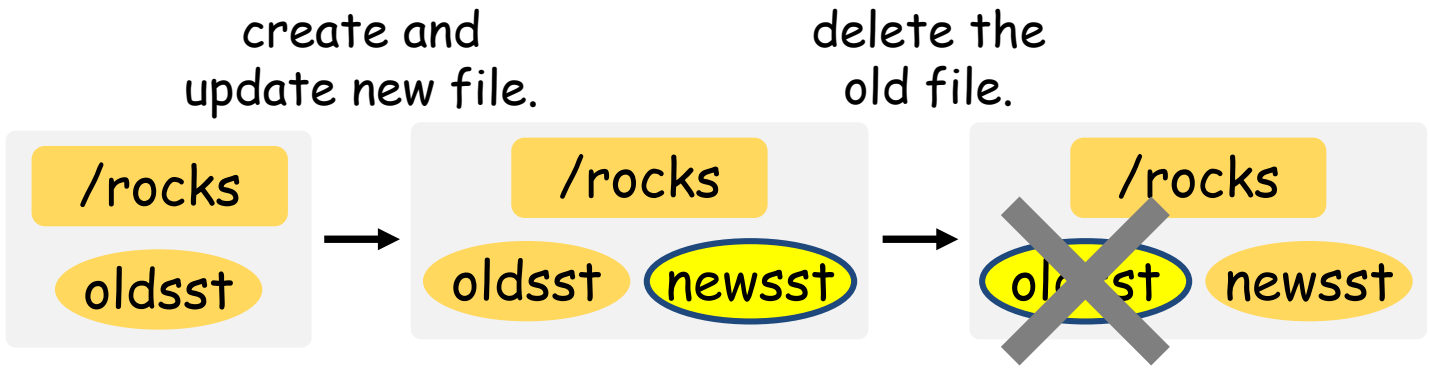
Directory Update (Property U)

All file update accompanies directory update.

- Create new file → Update New File → Delete Old File

RocksDB

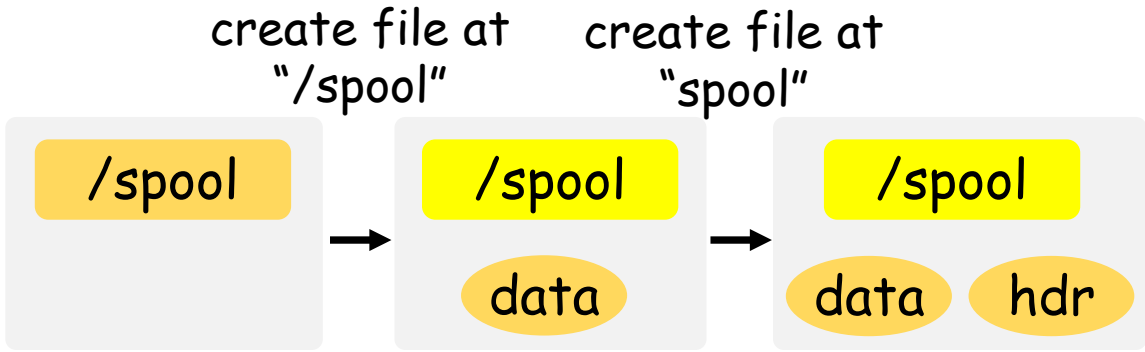
```
1. while (/d/sst[@]) {
2.   creat(/d/newsst[@]);
3.   append(/d/newsst[@]);
4.   fsync(/d/newsst[@]);
5. }
6. fsync(/d);
7. unlink(/d/sst[@]);
...
```



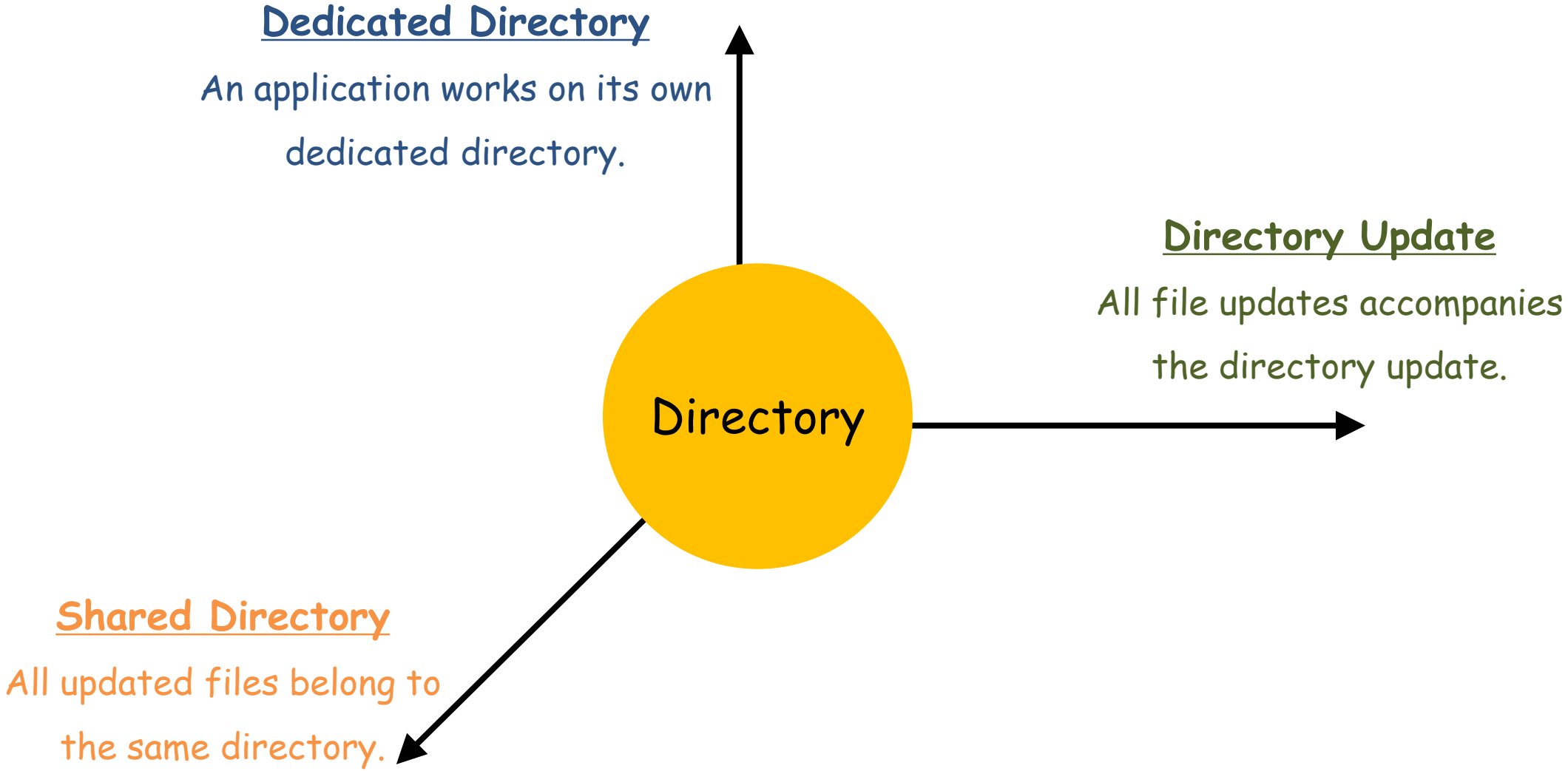
Shared Directory (Property S)

All updated files belong to the same directory.


```
1. creat(/d/spl_data);
2. append(/d/spl_data);
3. fsync(/d/spl_data);
4. creat(/d/spl_hdr_tmp);
5. append(/d/spl_hdr_tmp);
6. fsync(/d/spl_hdr_tmp);
7. rename(/d/spl_hdr_tmp,/d/spl_hdr);
```



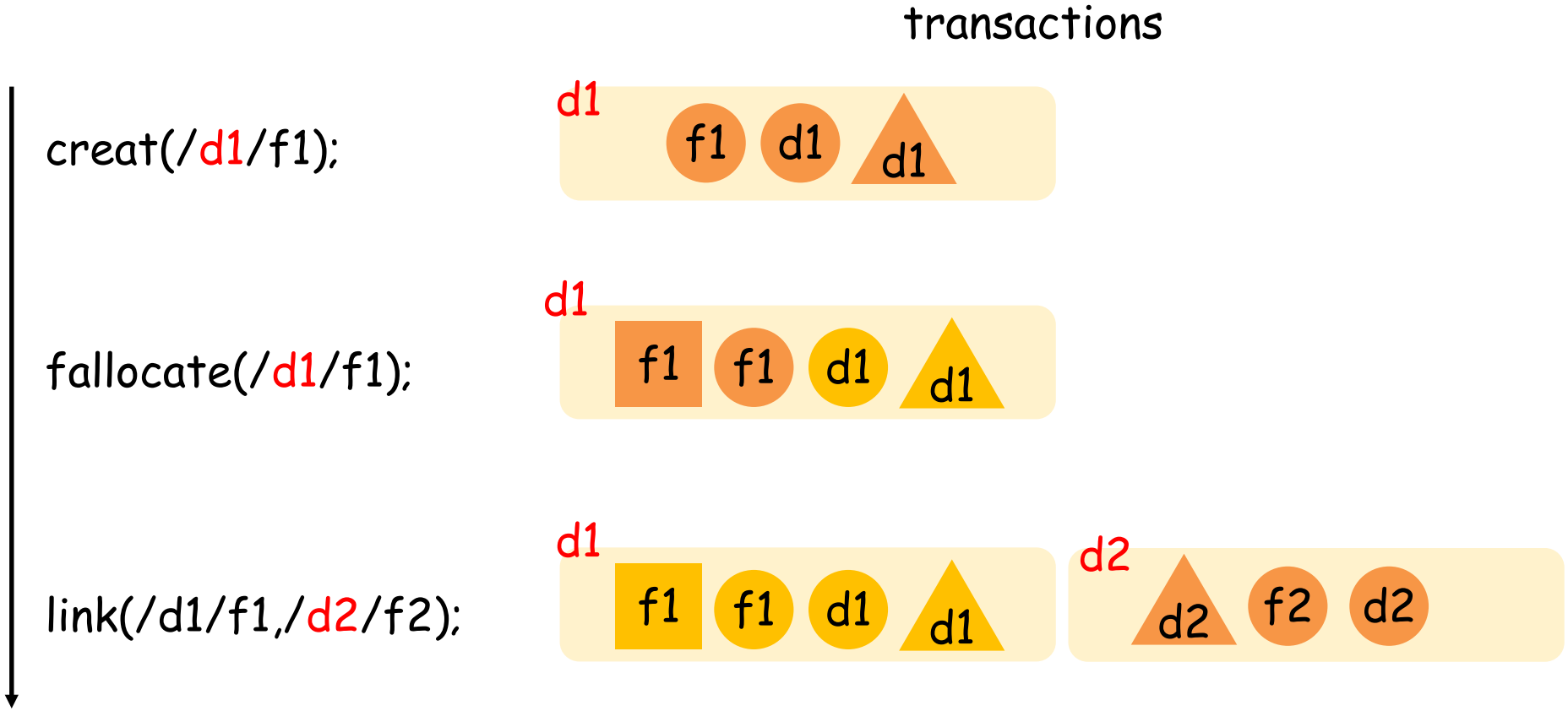
Characteristics of File Update Patterns



Daejeon Directory Journaling Filesystem, DJFS

Directory as a unit of journal transaction !

Directory as a unit of journal transaction

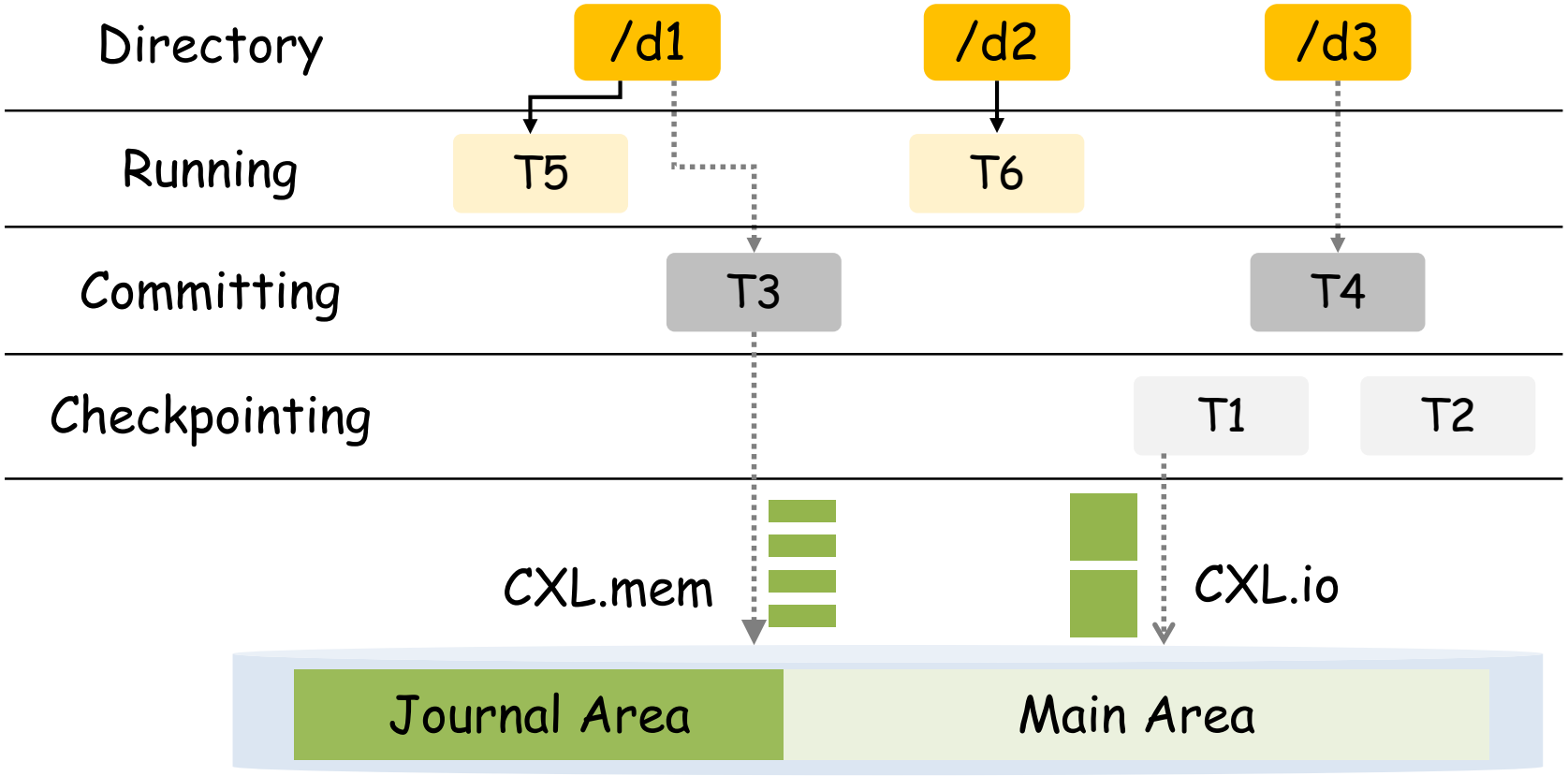


Analysis

Filesystem	Minimize Lock-Up	Minimize Conflict	Maximize Compound Degree
Single Running Tx. (JBD,WiP'98)	X	X	O
Per-Partition Tx. (SpanFS, ATC'15)	X	O	Δ
Per-file Tx. (iJournaling, ATC'17)	O	O	X
Per-Core Tx. (Z-Journal, ATC'21)	X	X	X
Dual Thread Design (CJFS, FAST'23)	X	O	O
Logical Logging (FastCommit, ATC'24)	X	Δ	O
DJFS	O	O	O

Overview

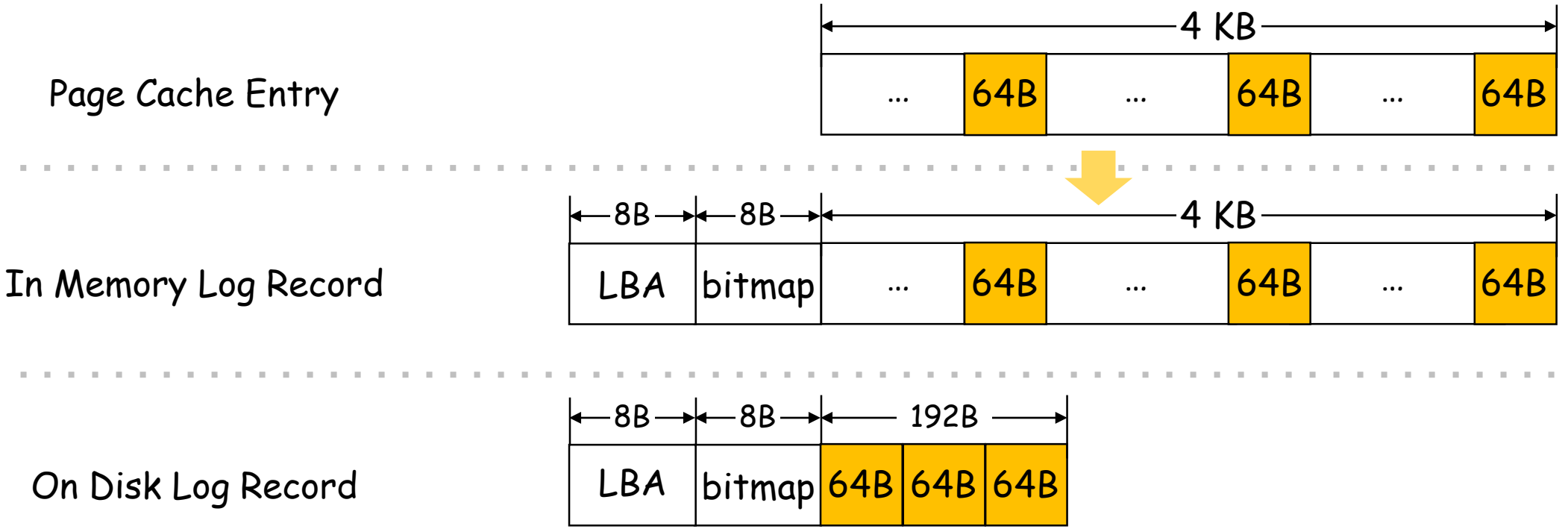
Each directory can have one running transaction and one committing transaction



Data Structures – Log Records

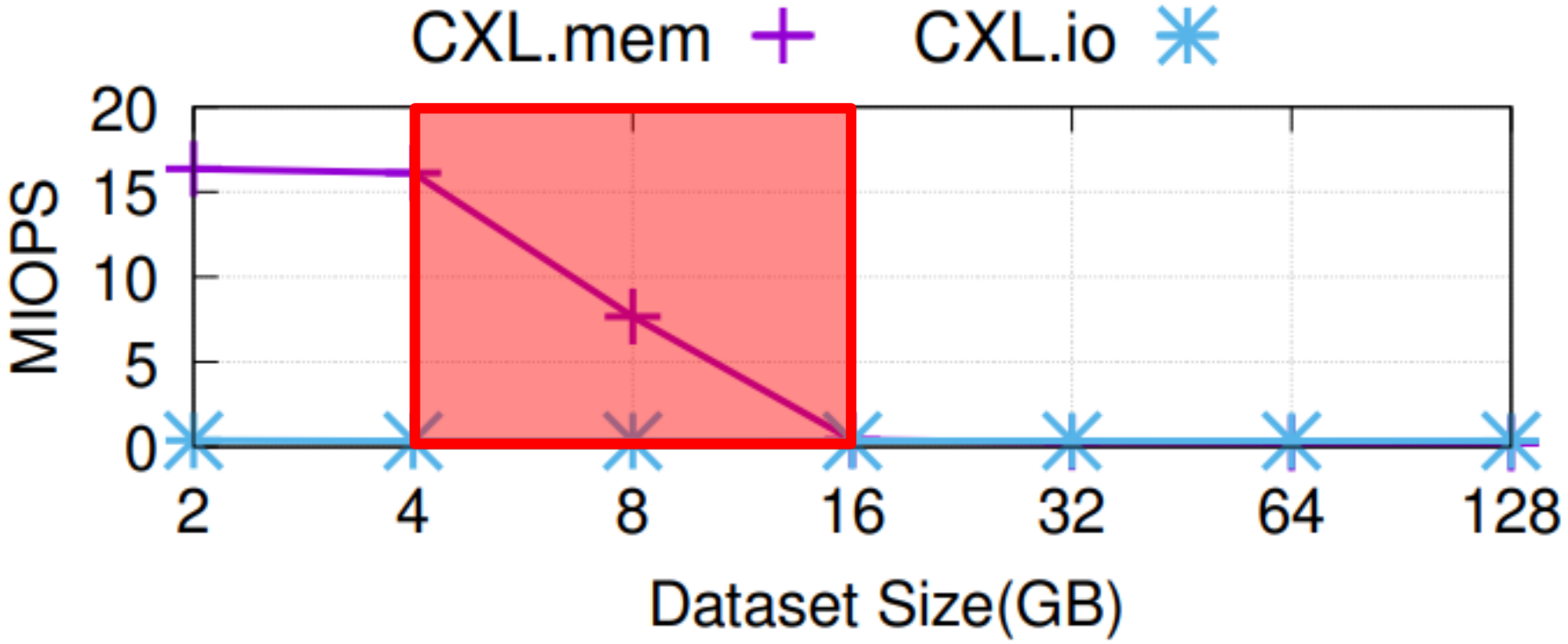
We journal only file metadata [iJournaling,ATC'17][FastCommit,ATC'24].

- inode) 256 bytes granularity logging.
- index and directory block) 64bytes granularity logging.



Access Locality in CMM-H

Journal region should fit in the cache of CMM-H.

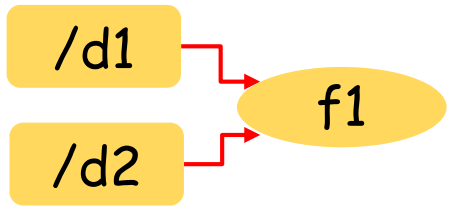


Random write throughput according to the dataset size
(CMM-H Prototype)

Challenges

What if a file has multiple links?

```
truncate("f1",…);
```



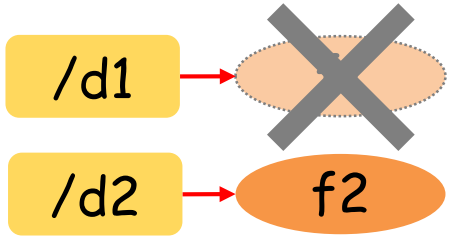
T1:/d1

T2:/d2

Transaction Selection

How to guarantee multi dir operation atomicity?

```
rename("/d1/f1",/d2/f2");
```



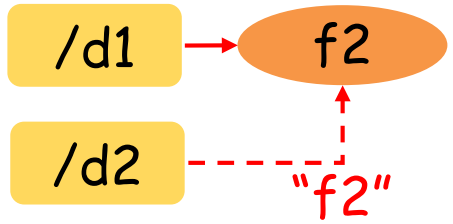
T1:/d1

T2:/d2

Multi Transaction Atomicity

What if a conflict occurs?

```
creat("/d1/f1");
link("/d1/f1","/d2/f2");
```



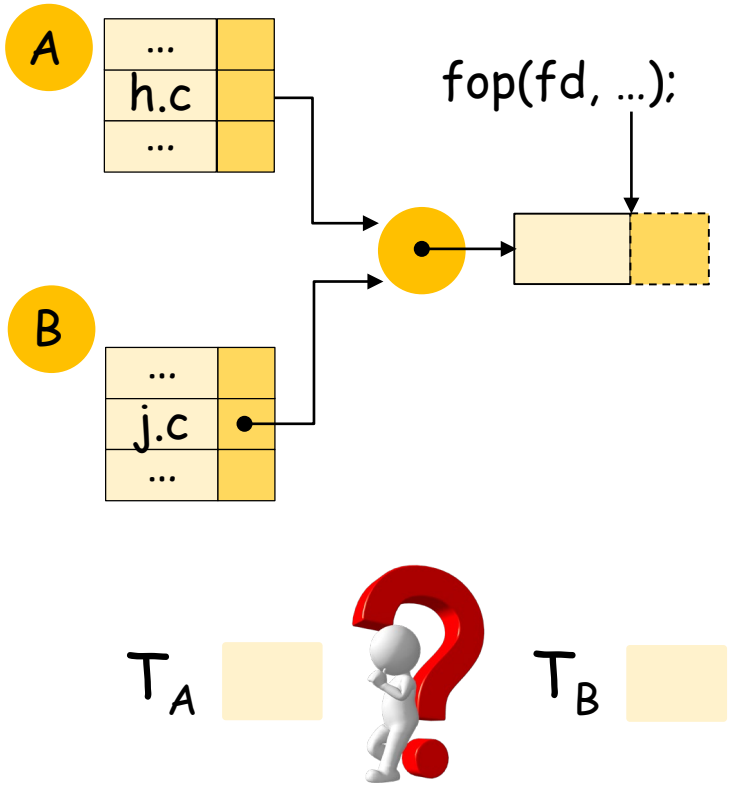
T1:/d1

T2:/d2

Commit Order

Path-based Transaction Selection

One transaction must be selected.

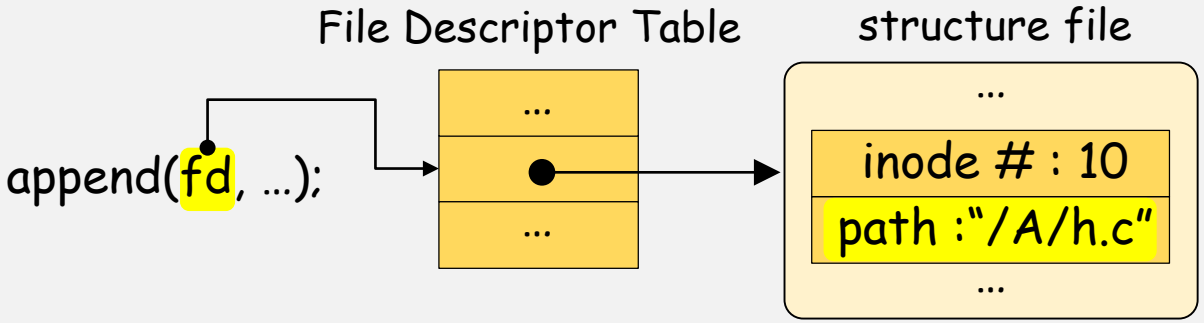


Path-based Transaction Selection

Path-based operation

creat("/A/h.c");

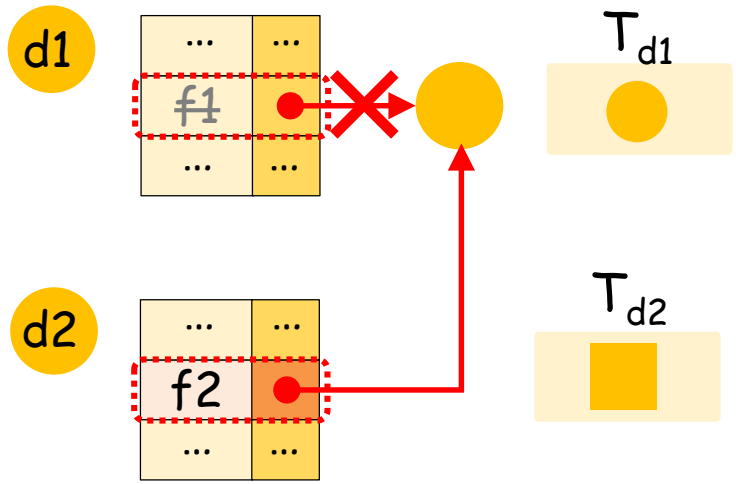
File descriptor based operation



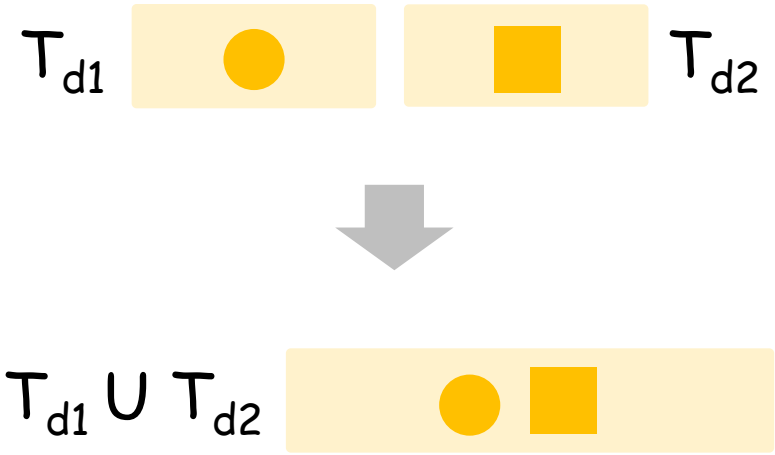
Transaction Coalescing

Atomicity of Multi Directory operation can be violated.

```
rename("/d1/f1",/d2/f2");
```



Coalesce Two Transactions!

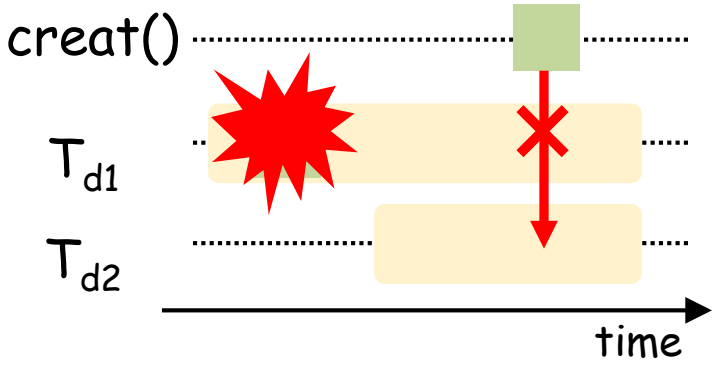


Transaction Conflict Resolution

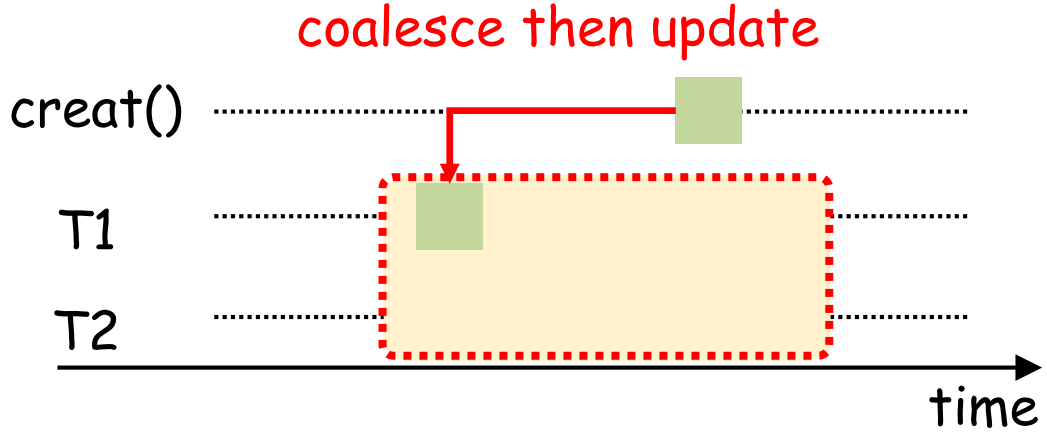
Running-to-Running Conflict

What if an updated object belong to another "running" transaction?

```
mkdir(/d1/d2);
creat(/d1/d2/f1);
```



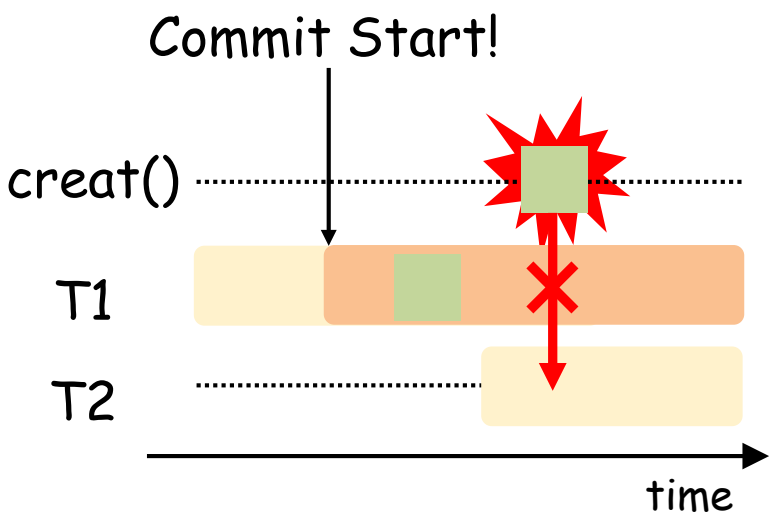
Coalesce Two Transactions!



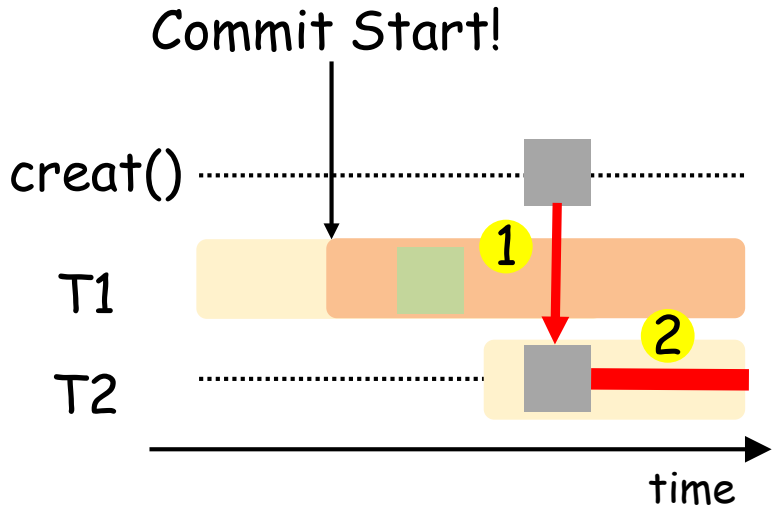
Transaction Conflict Resolution

Running-to-Committing Conflict

What if an updated object belong to another "committing" transaction?



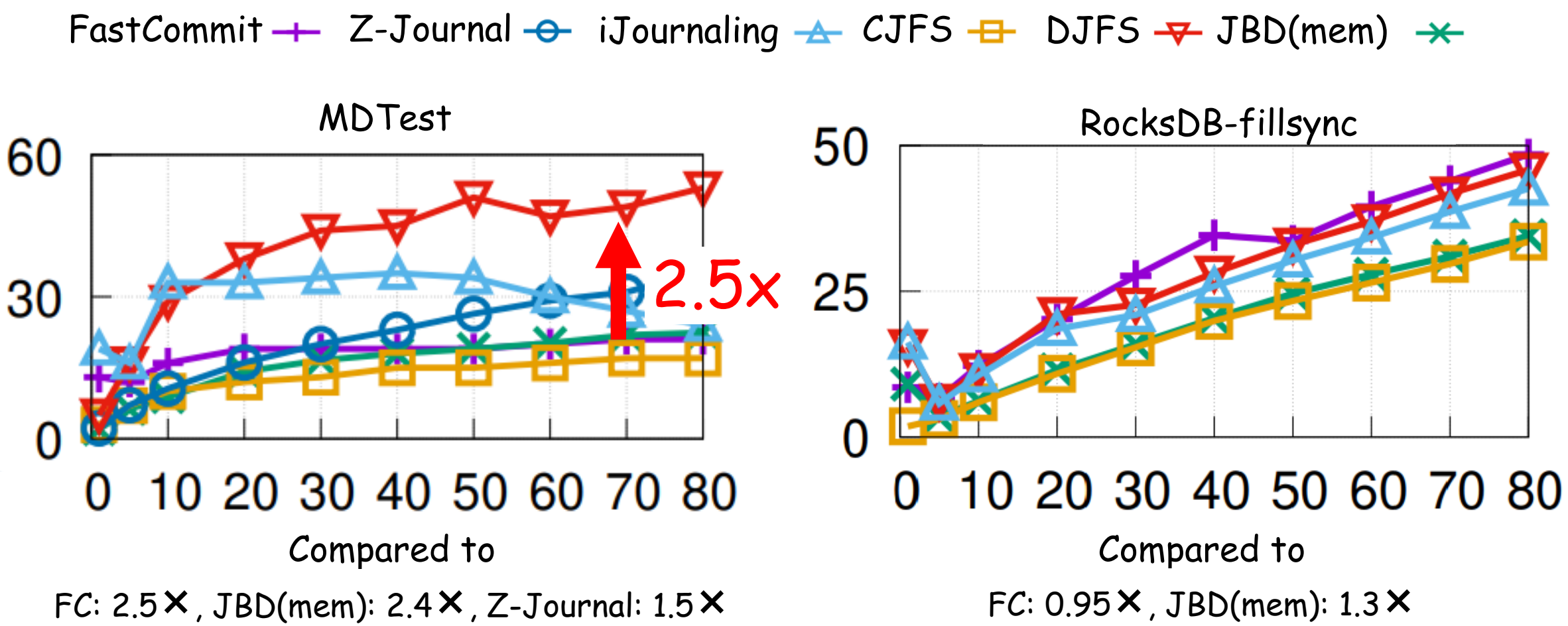
1 insert shadow copy 2 delay commit.



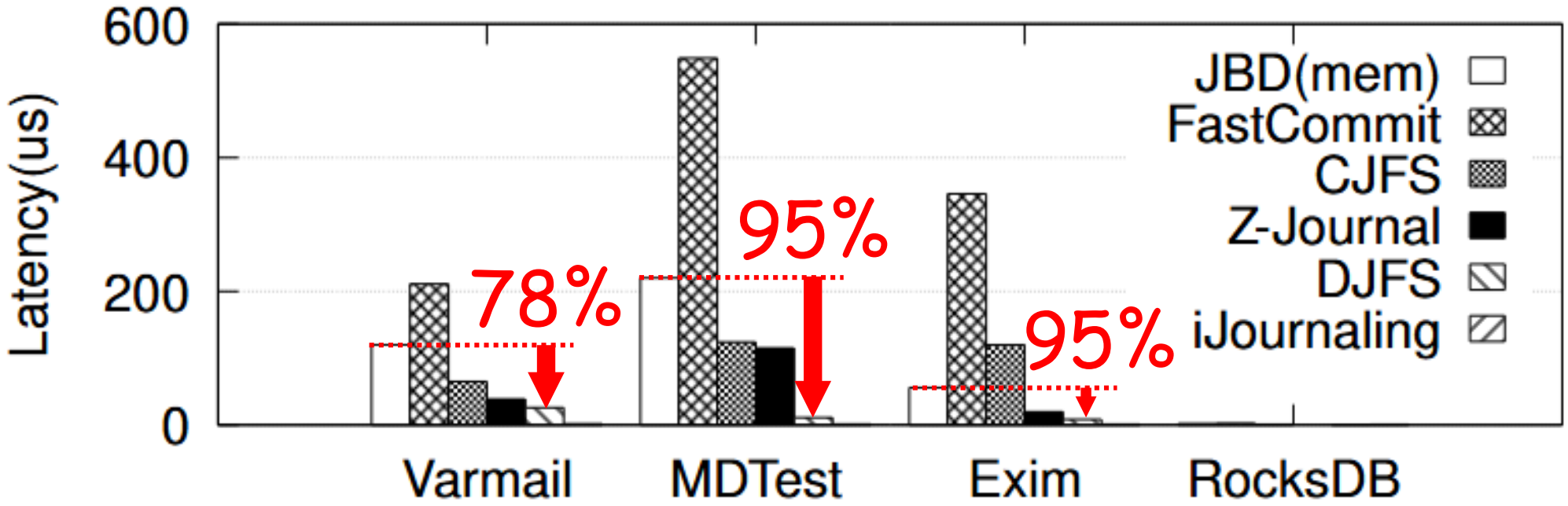
Evaluation Setup

- CPU : Intel SPR 88 Core (2 Socket X 44 core = 88 cores)
- Memory : 128GB DRAM
- Storage: Dual Mode CMM-H 2TB
- OS (Kernel) : Ubuntu 22.04 (Linux Kernel 5.18)
- Filesystem: JBD(mem), FastCommit, CJFS, Z-Journal, iJournaling, DJFS
  - Journal commit with .mem interface : JBD, FastCommit, iJournaling, DJFS
  - Journal commit with .io interface : CJFS, Z-Journal
- Workloads: Varmail, MDTest, Exim, RocksDB-fillsync

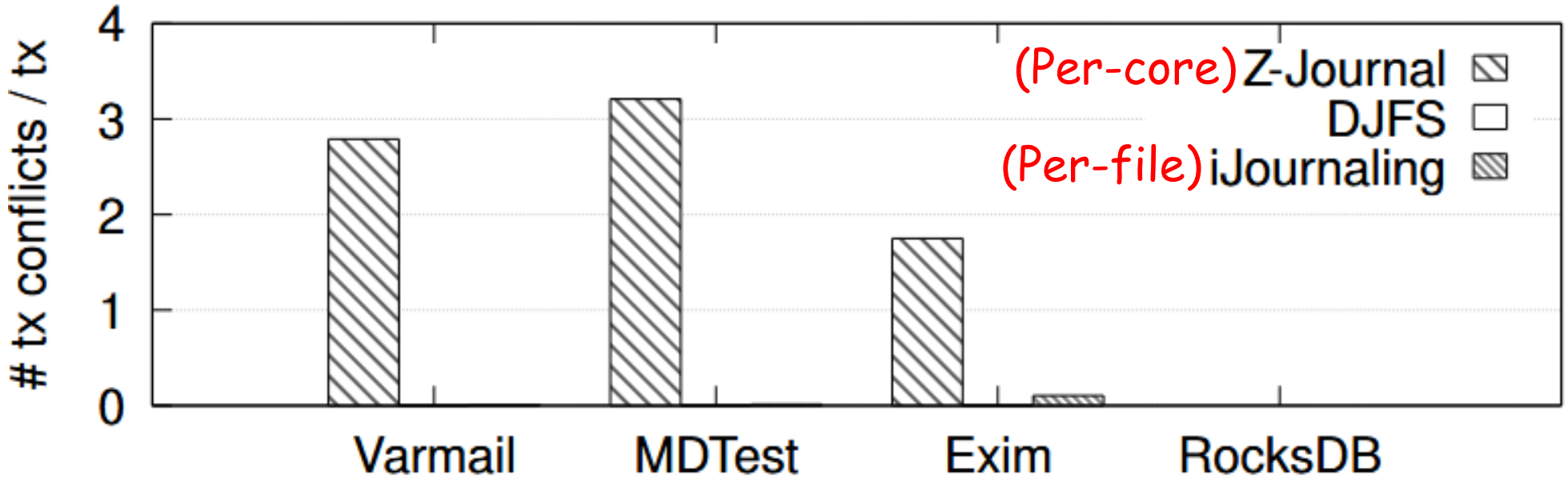
Macro Benchmarks



Transaction Lock-Up overhead



of Transaction Conflicts



Per-directory and per-file transaction almost eliminates transaction conflict.

Compound Degree at 80 Threads

Workload	Varmail	MDTest	Exim	fillsync
DJFS	4.2	2.0	1.9	1.0
iJournaling	0.9	1.0	1.3	0.6

Per-directory transaction shows higher coalescing degree than per-file transaction!

Conclusion

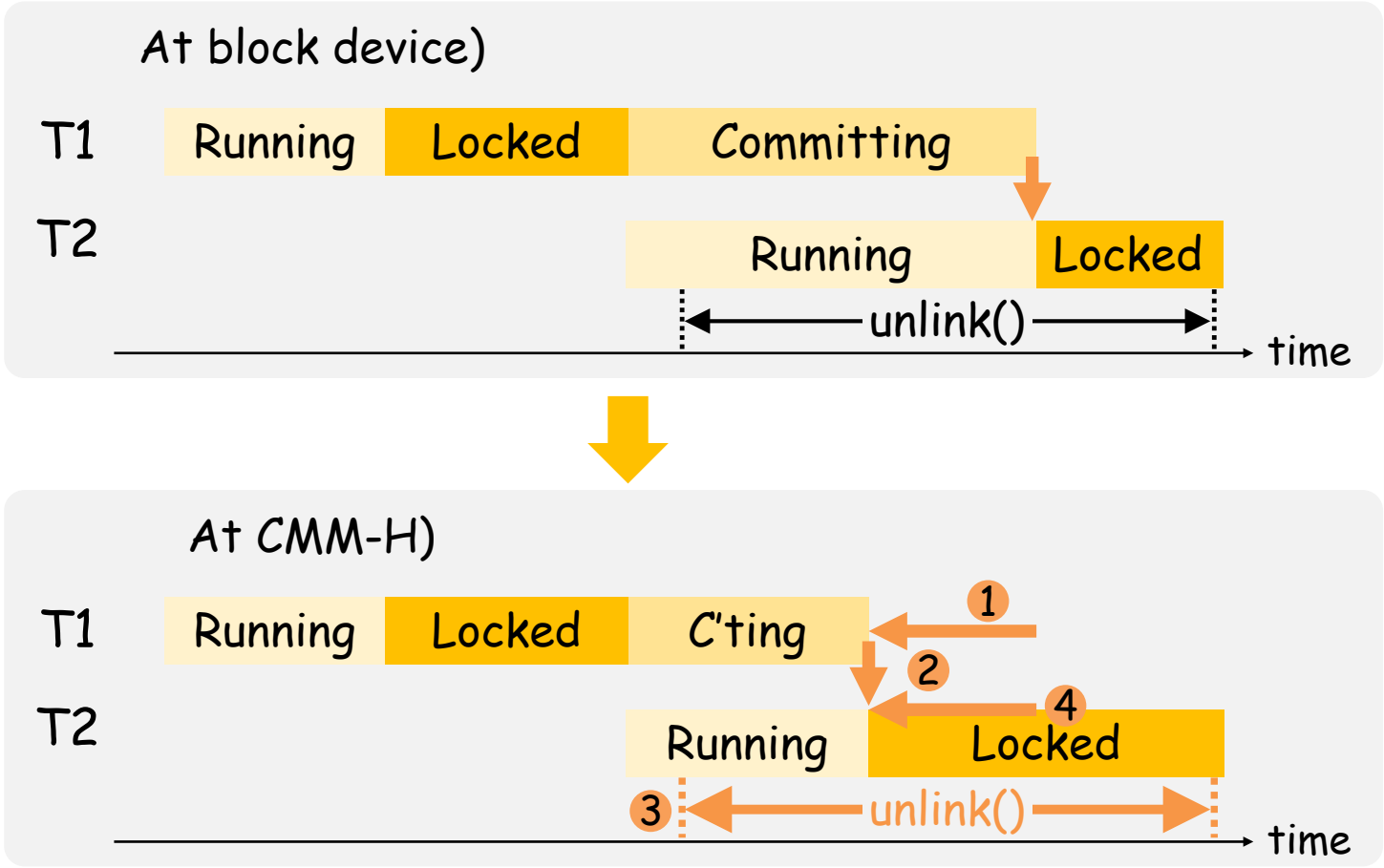
- Application's file update behavior is centered around a dedicated directory.
- We propose a new unit of filesystem journaling; directory.
- For CMM-H, we need a new journaling filesystem that can exploit the access locality of CMM-H.
- We propose DJFS, a directory-granularity journaling file system.

DJFS, Directory Journaling Filesystem

Question & Answer

Transaction Lock-Up becomes serious at CMM-H

As commit becomes shorter, lock-up becomes longer.



1. commit becomes shorter (decreases by 35%).

2. Lock up starts faster

3. Lock up ends same as before. (fop latency remains same.)

4. Lock up becomes longer (Portion : 6% → 18%).