Operating Systems Group
`https://os.itec.kit.edu`

Fundamental OS Design Considerations for CXL-based Hybrid SSDs

Daniel Habicht, Yussuf Khalil, Lukas Werling, Thorsten Gröninger, and Frank Bellosa

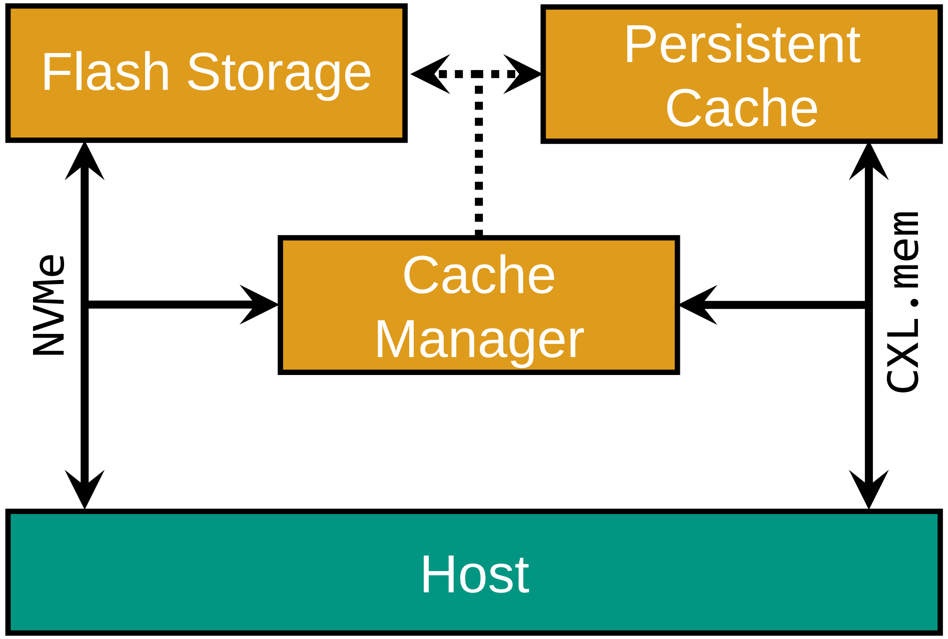
Hybrid SSDs as Cost-Effective PM

✅ cost-effective (Flash ≫ cache)

❌ no uniform access due to cache

❌ existing OS abstractions unsuitable

Our contribution:
OS-centric hybrid SSD management

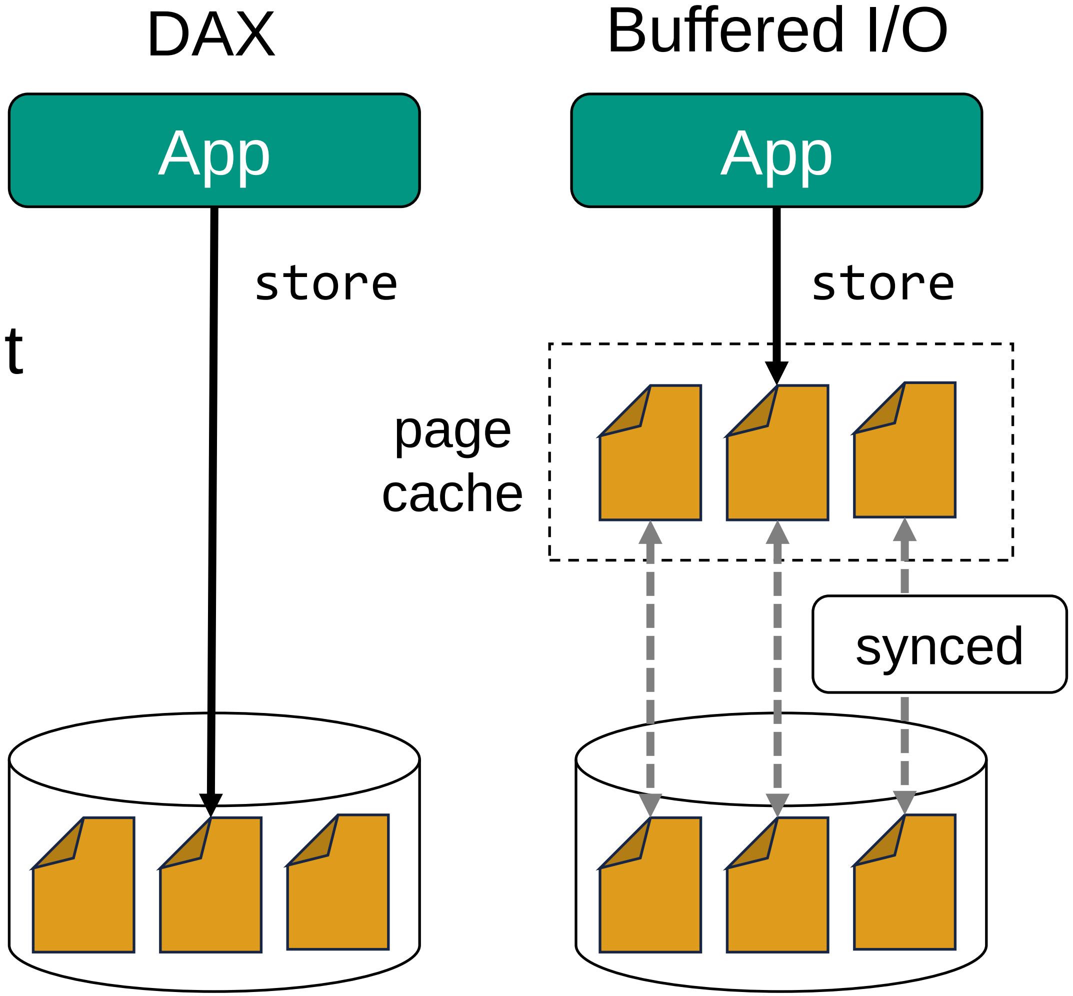


Linux Direct Access (DAX)

- DAX = (volatile) page cache bypass

- Per-inode DAX flag
  ❌ no fine-granular resource management
  ❌ pressure on small on-device cache

- Assumes non-blocking access
  ❌ CPU stalls on cache miss



Linux Direct Access (DAX)

- DAX = (volatile) page cache bypass

- Per-inode DAX flag

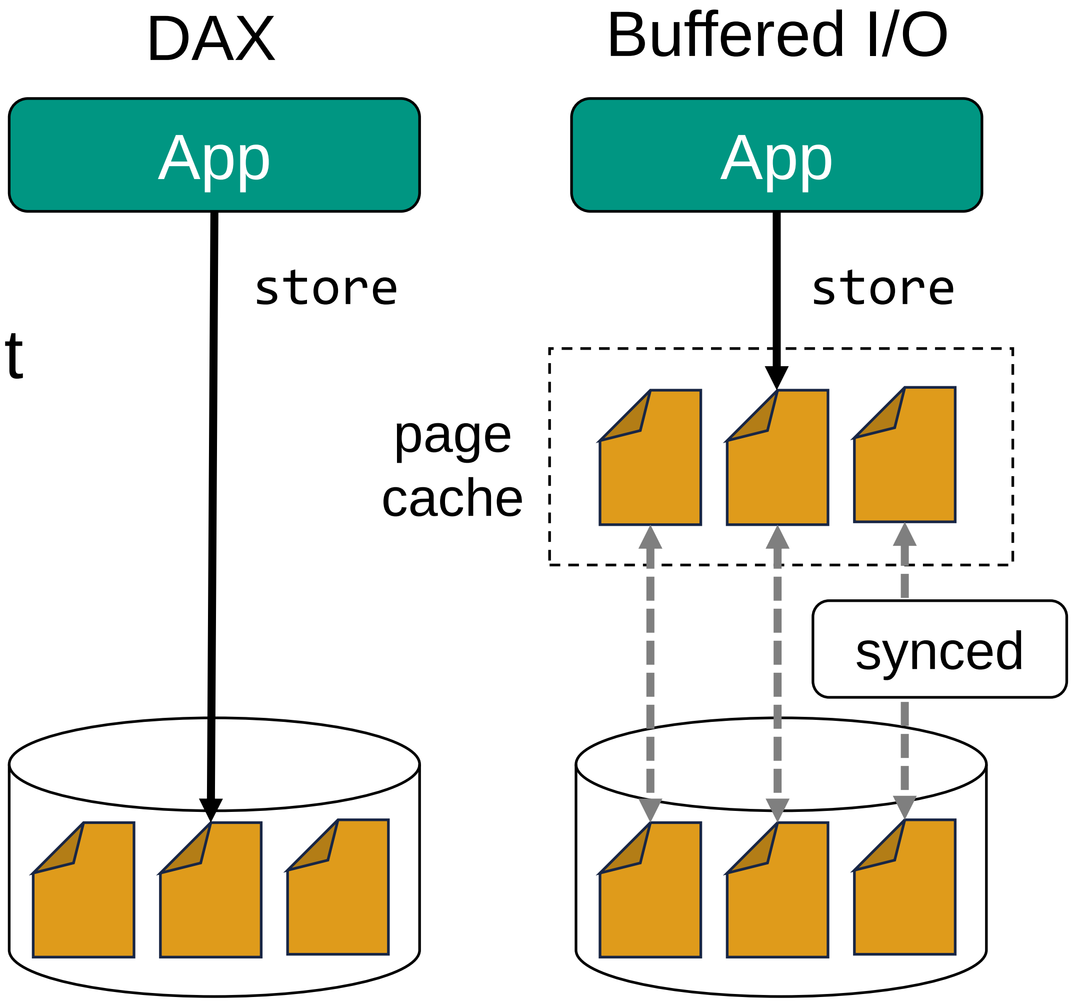
❌ no fine-granular resource management

❌ pressure on small on-device cache

- Assumes non-blocking access

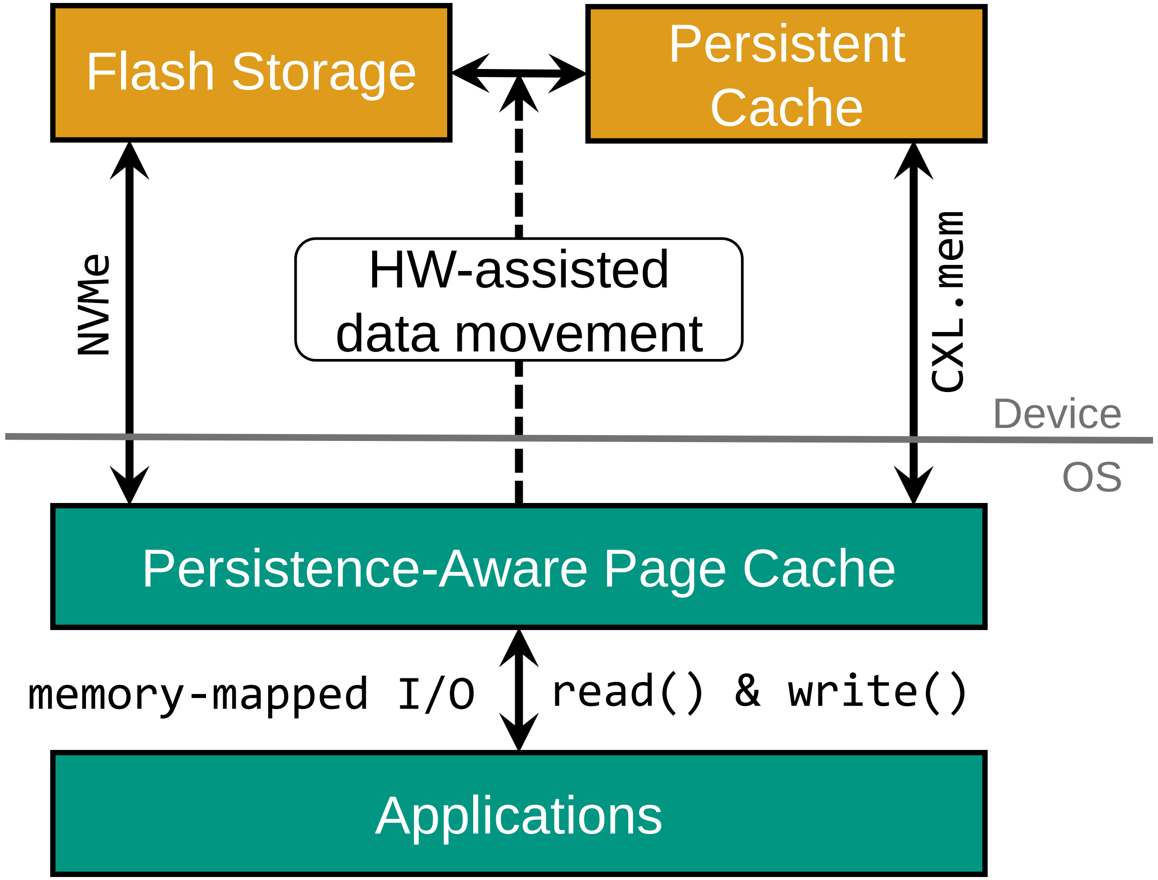
❌ CPU stalls on cache miss

➔ Existing DAX support unsuitable for hybrid SSDs

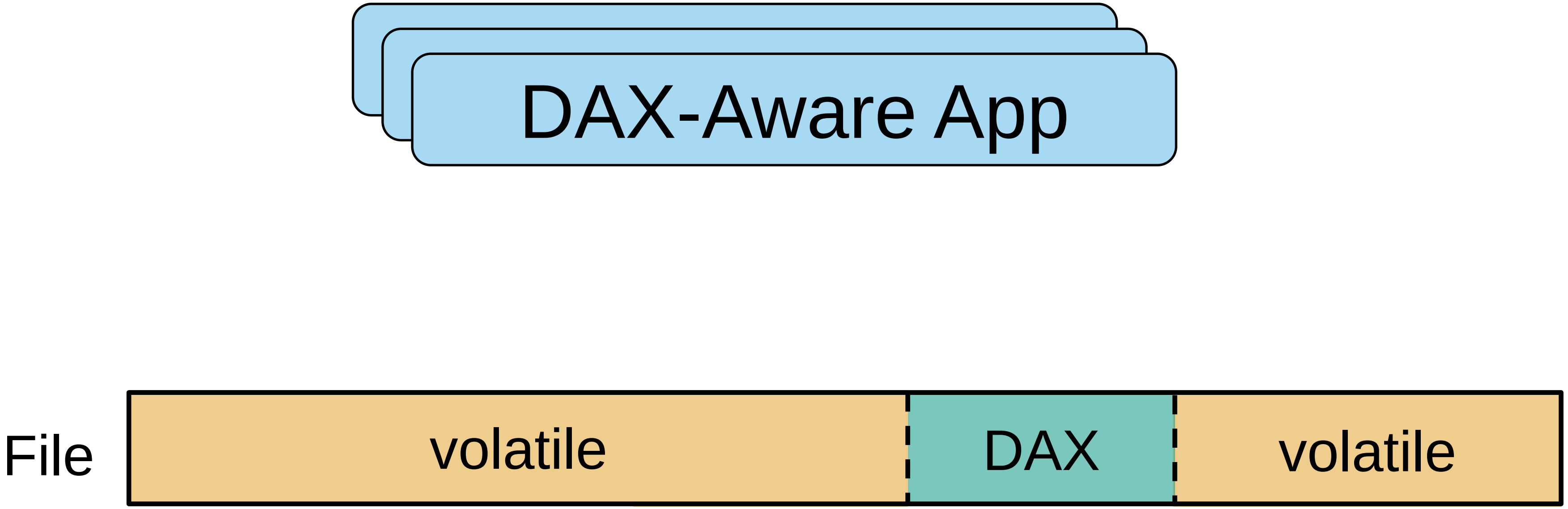


Our Solution: Persistence-Aware Page Cache

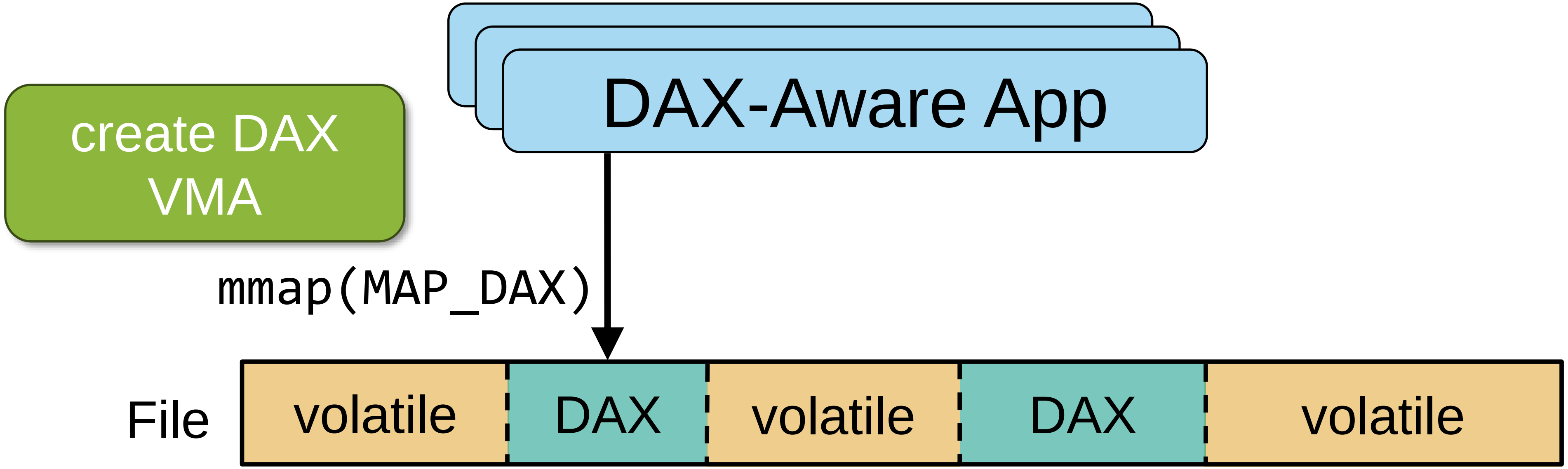
- OS-centric cache management
  - Host page tables reflect cache state
  - Hardware-assisted data movement
  - Expose resource management to apps
- Leverage persistence of DAX pages for lightweight `fsync()`



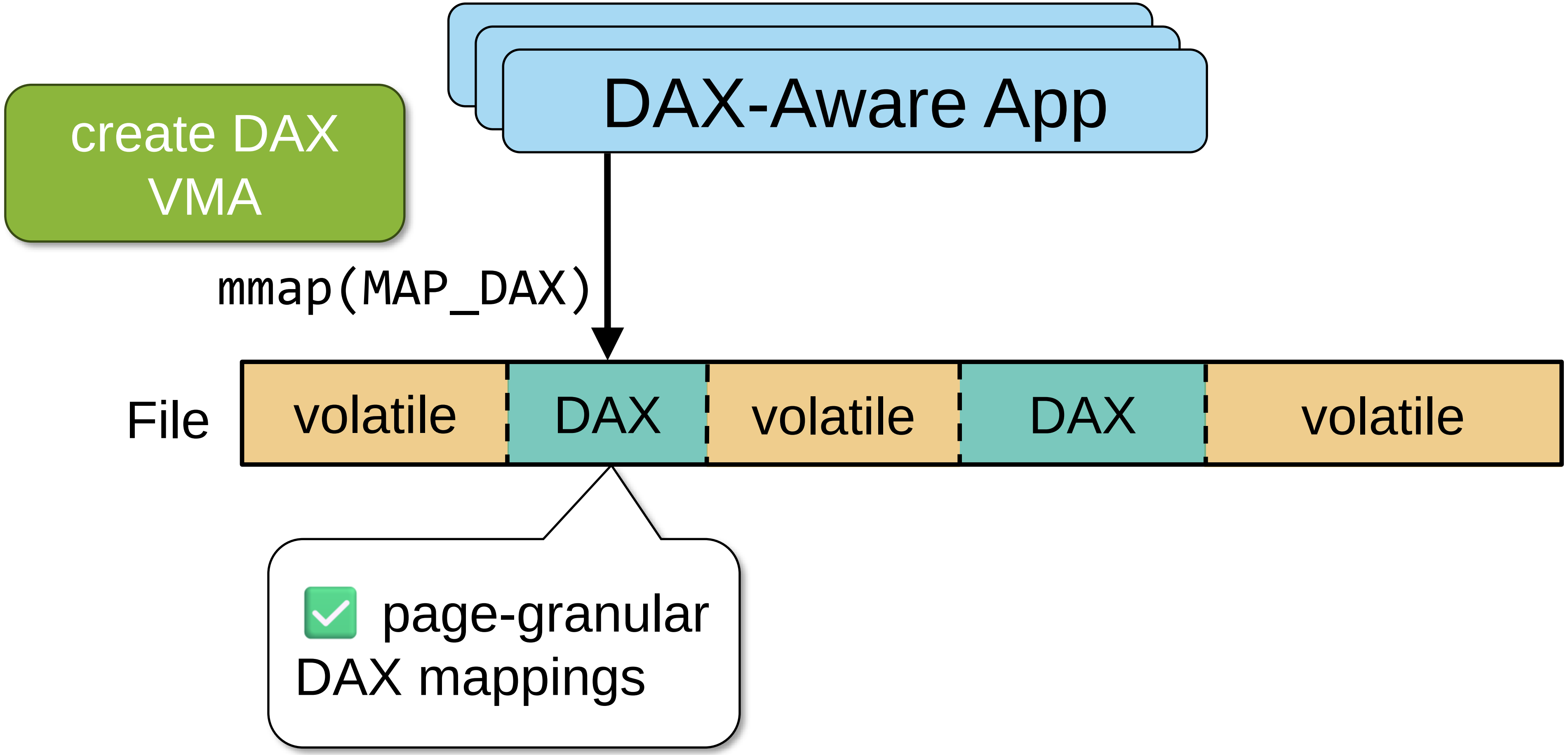
Fine-Granular Resource Management



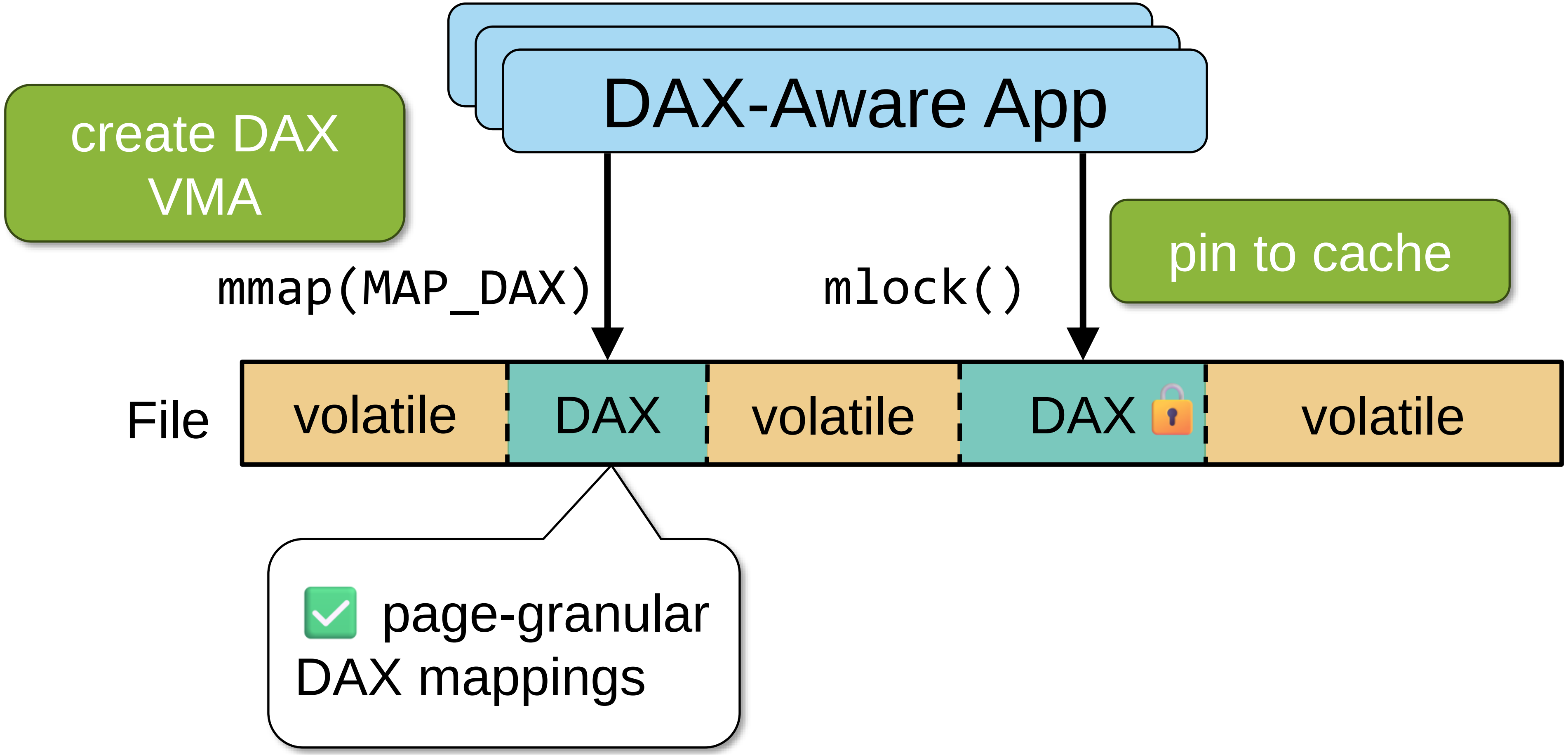
Fine-Granular Resource Management



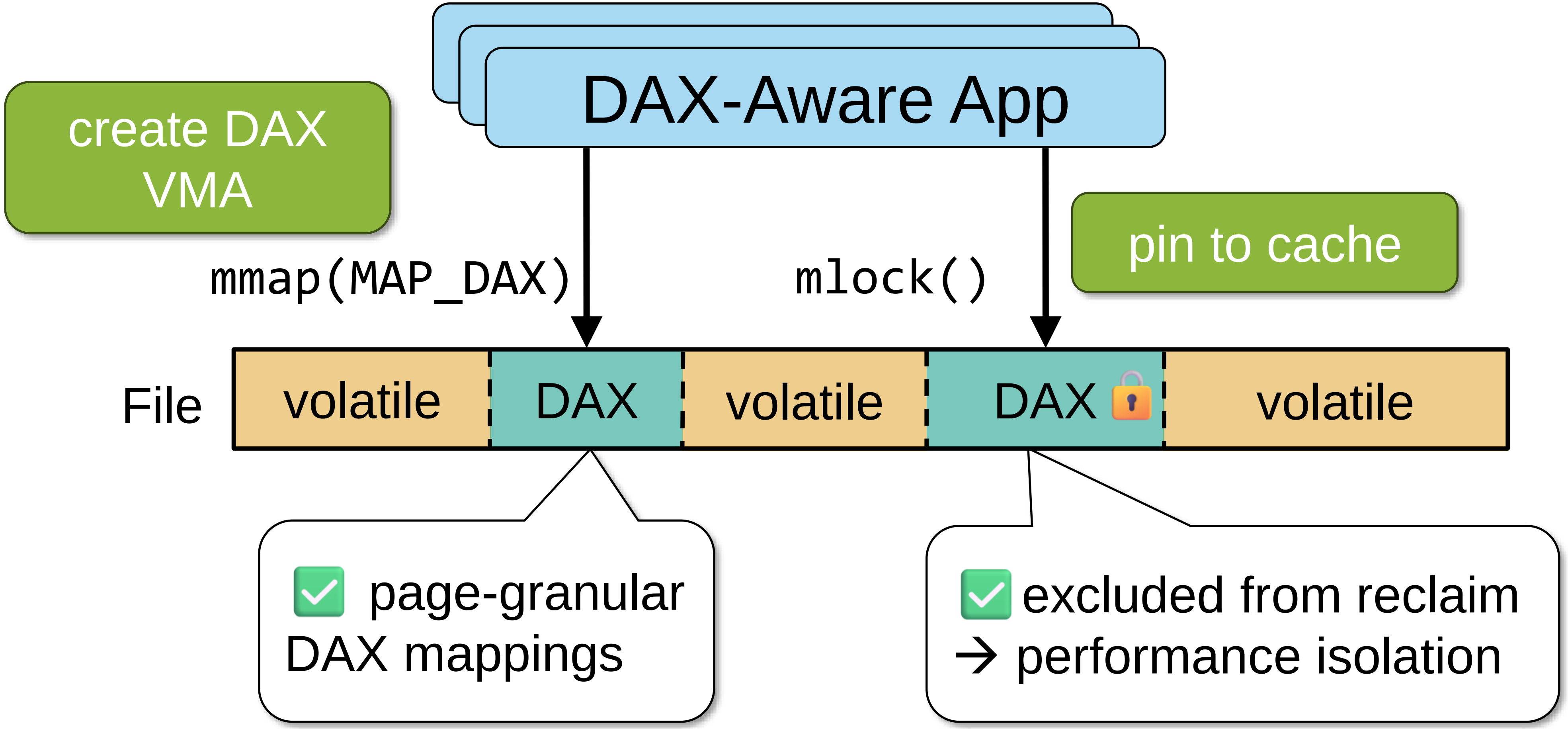
Fine-Granular Resource Management



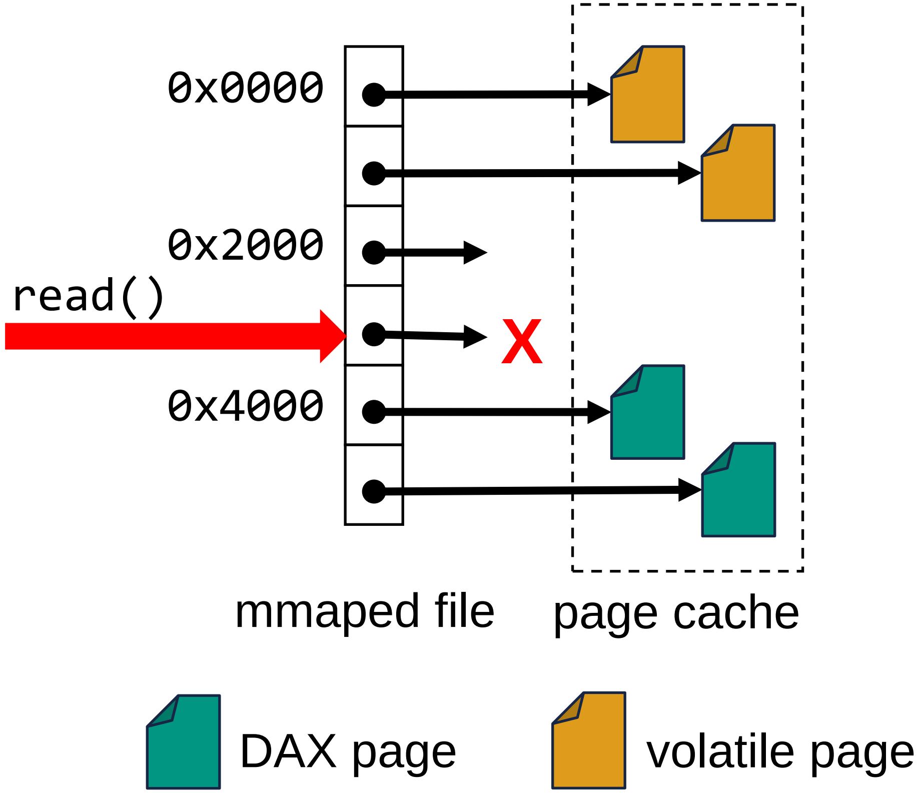
Fine-Granular Resource Management



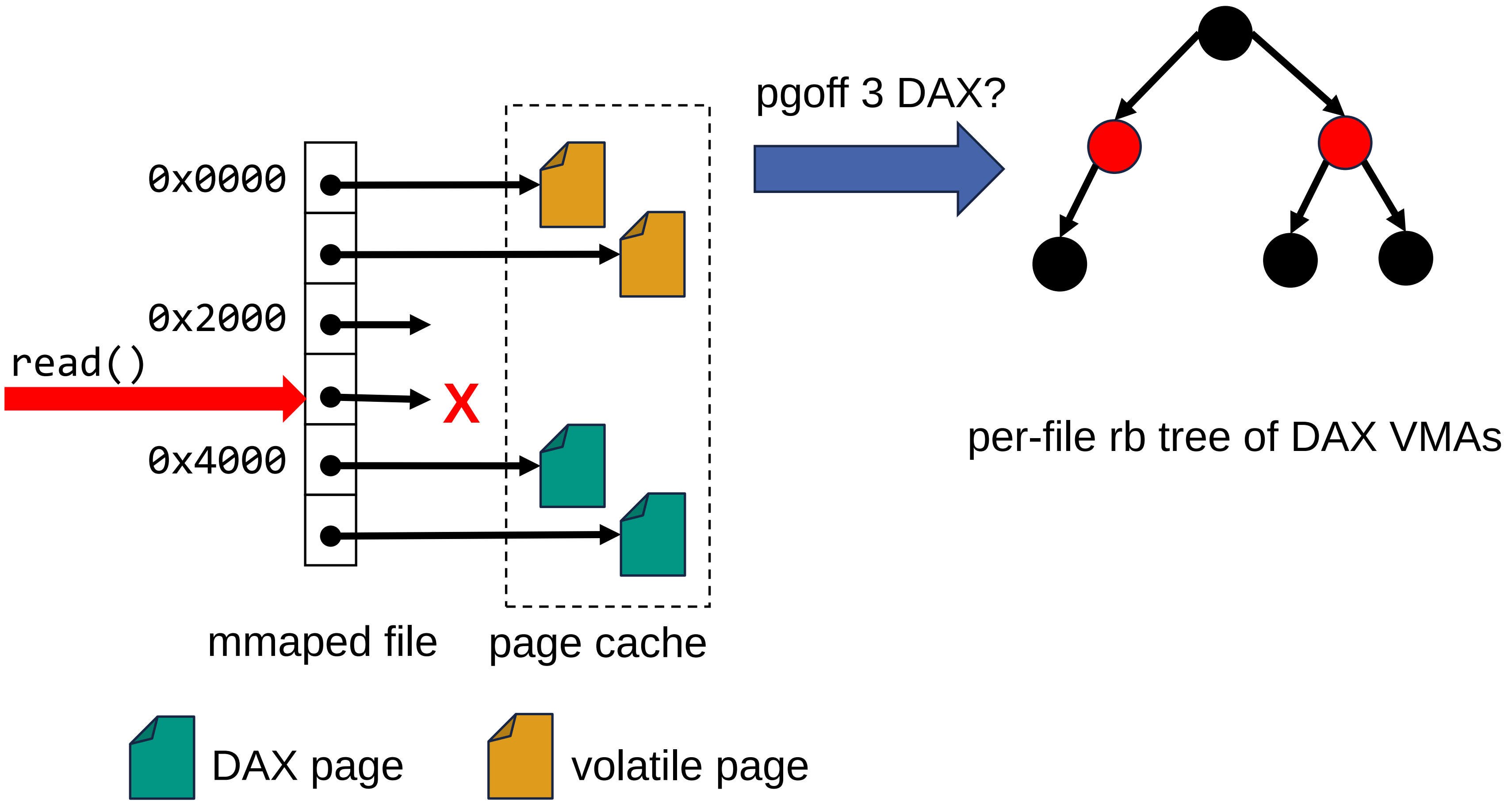
Fine-Granular Resource Management



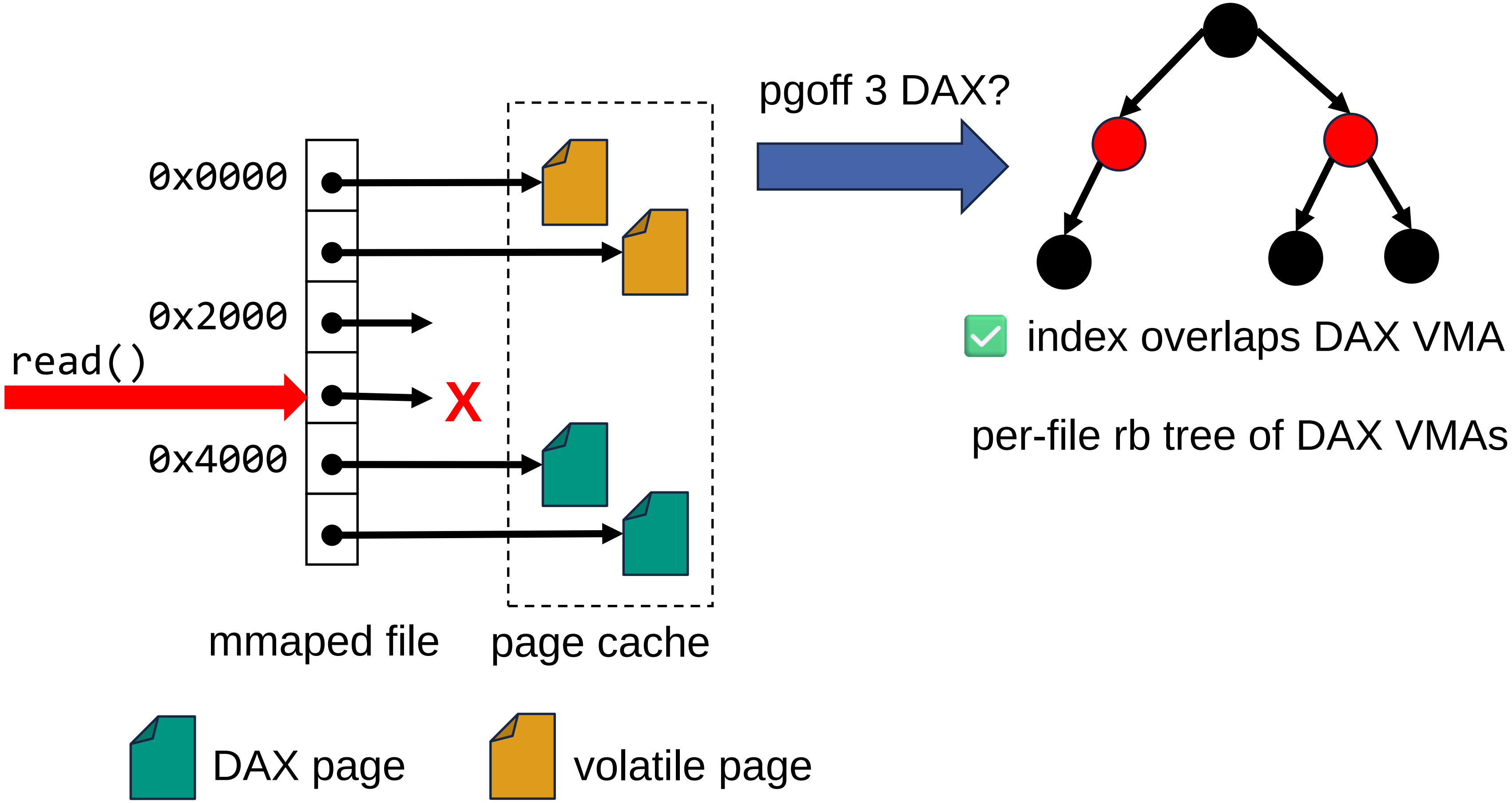
DAX-Aware Page Cache Allocation



DAX-Aware Page Cache Allocation



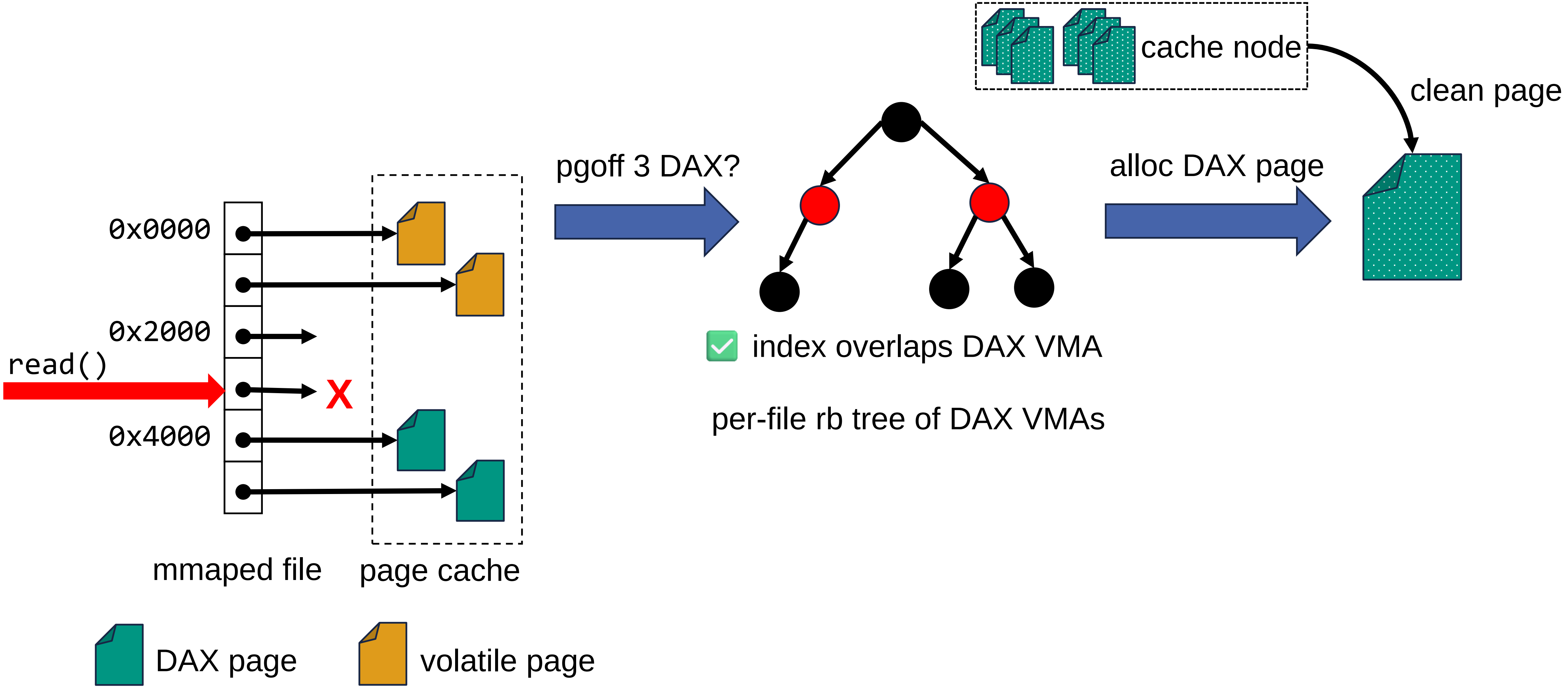
DAX-Aware Page Cache Allocation



✓ index overlaps DAX VMA

per-file rb tree of DAX VMAs

DAX-Aware Page Cache Allocation



cache node

clean page

pgoff 3 DAX?

alloc DAX page

0x0000

0x2000

read()

X

0x4000

✓ index overlaps DAX VMA

per-file rb tree of DAX VMAs

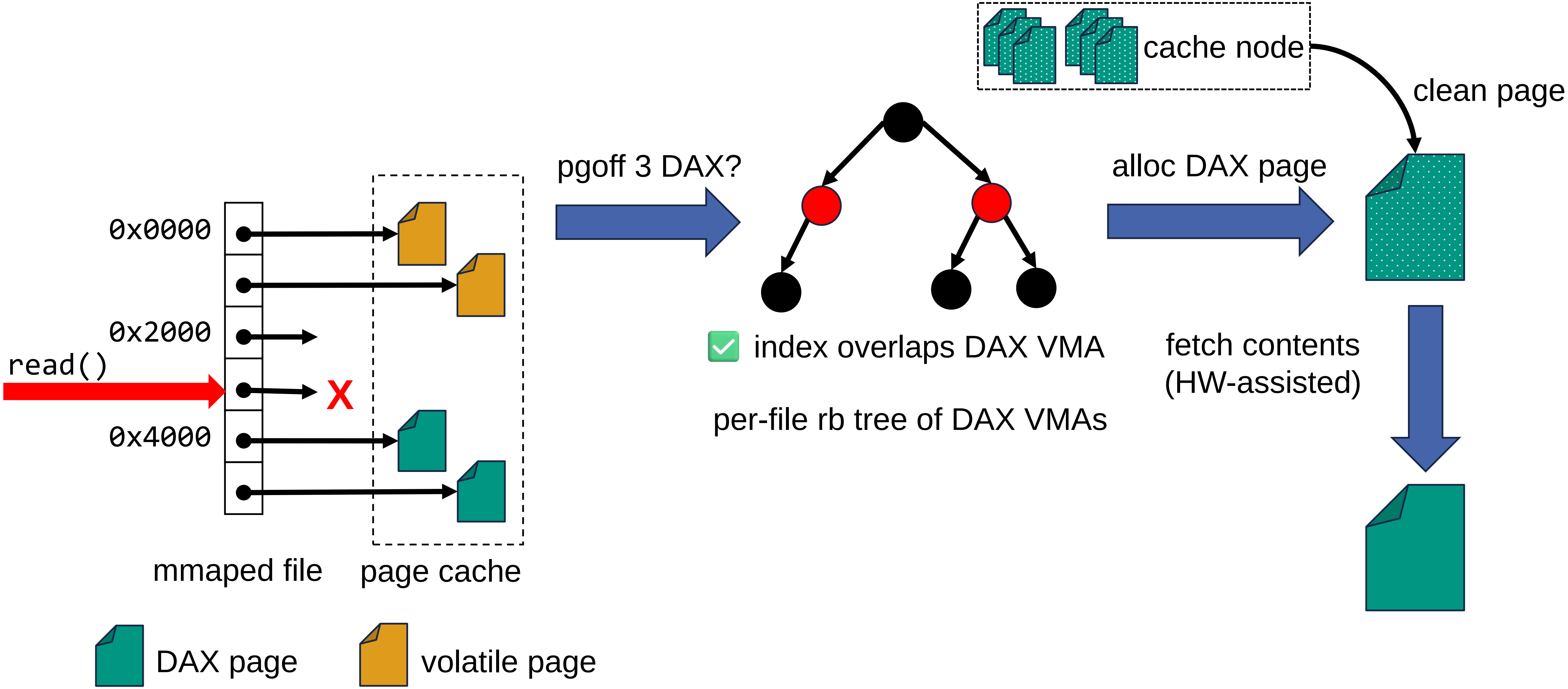
mmaped file

page cache

DAX page

volatile page

DAX-Aware Page Cache Allocation



cache node

clean page

pgoff 3 DAX?

alloc DAX page

0x0000

0x2000

read()

X

0x4000

✓ index overlaps DAX VMA

per-file rb tree of DAX VMAs

fetch contents (HW-assisted)

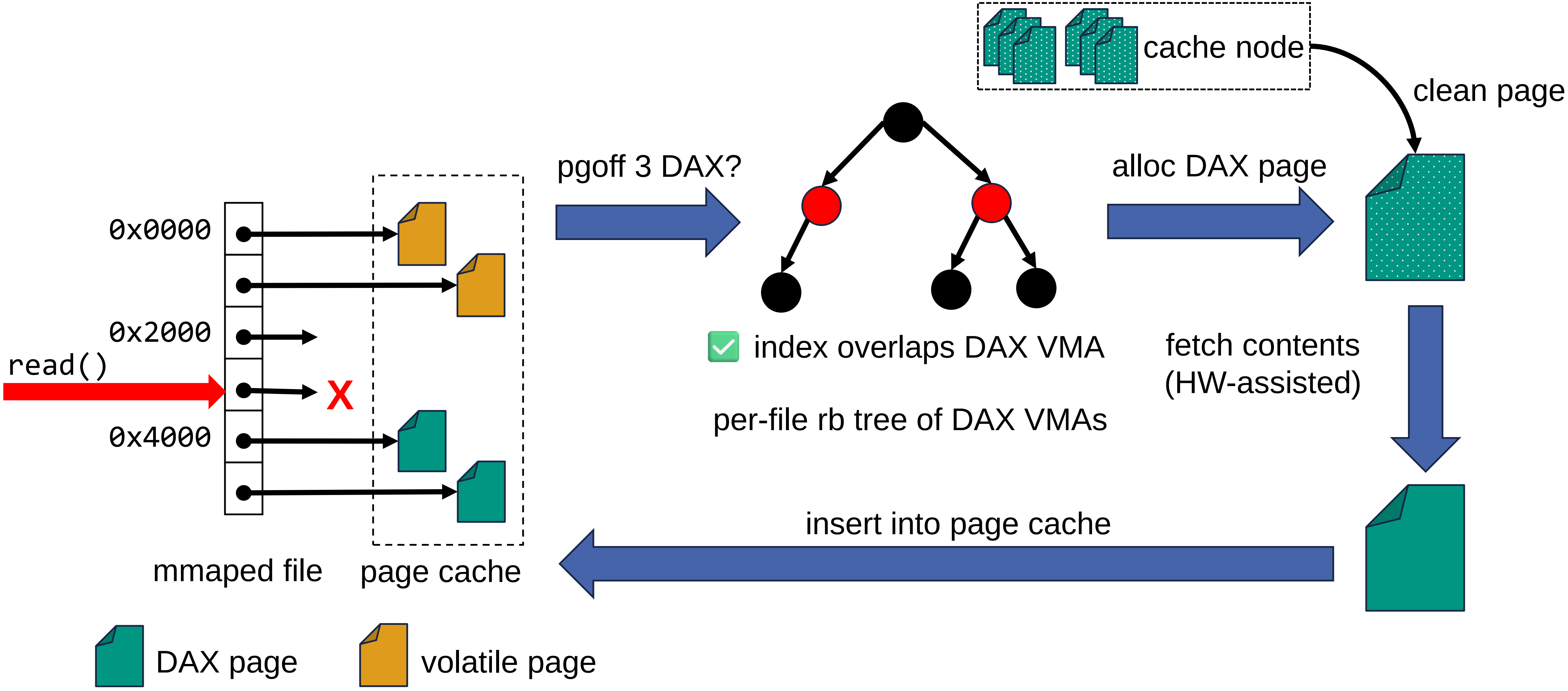
mmaped file

page cache

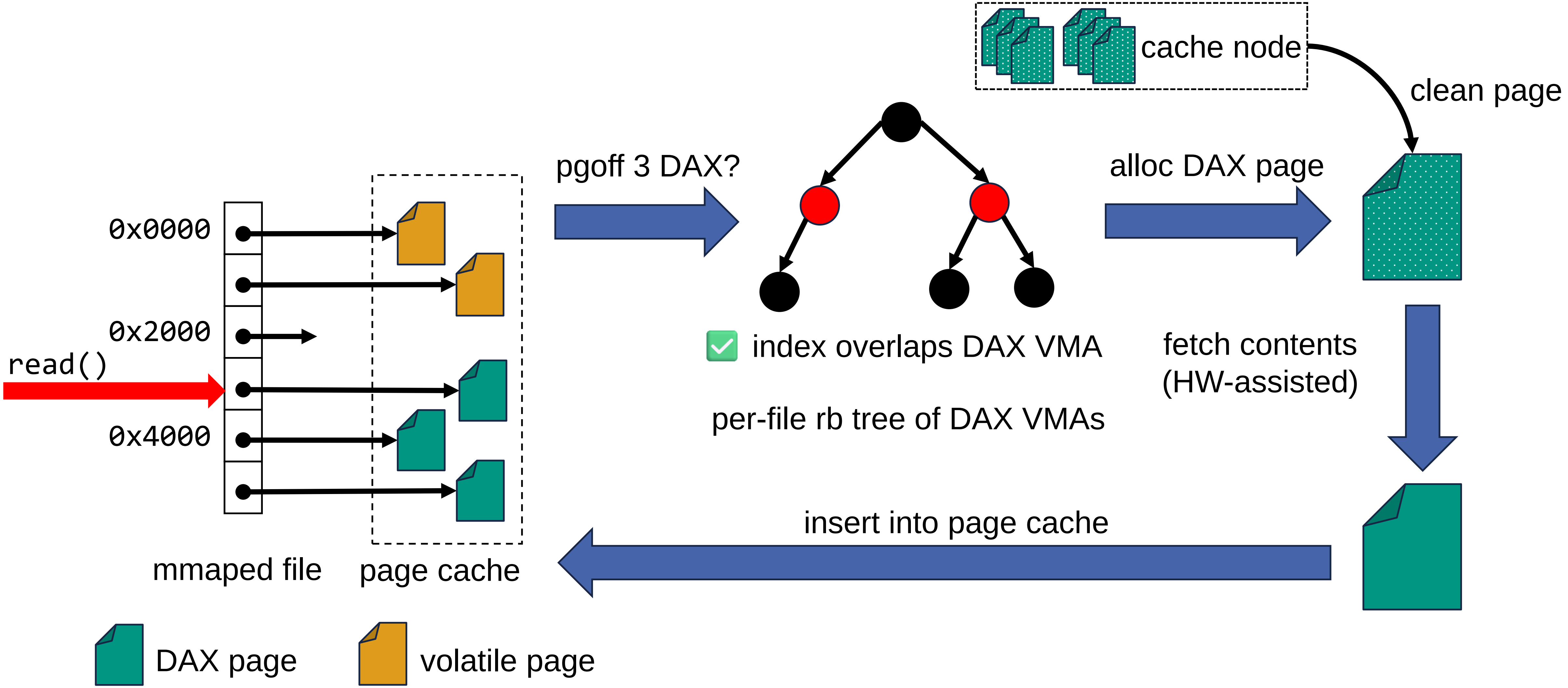
🟩 DAX page

🟧 volatile page

DAX-Aware Page Cache Allocation

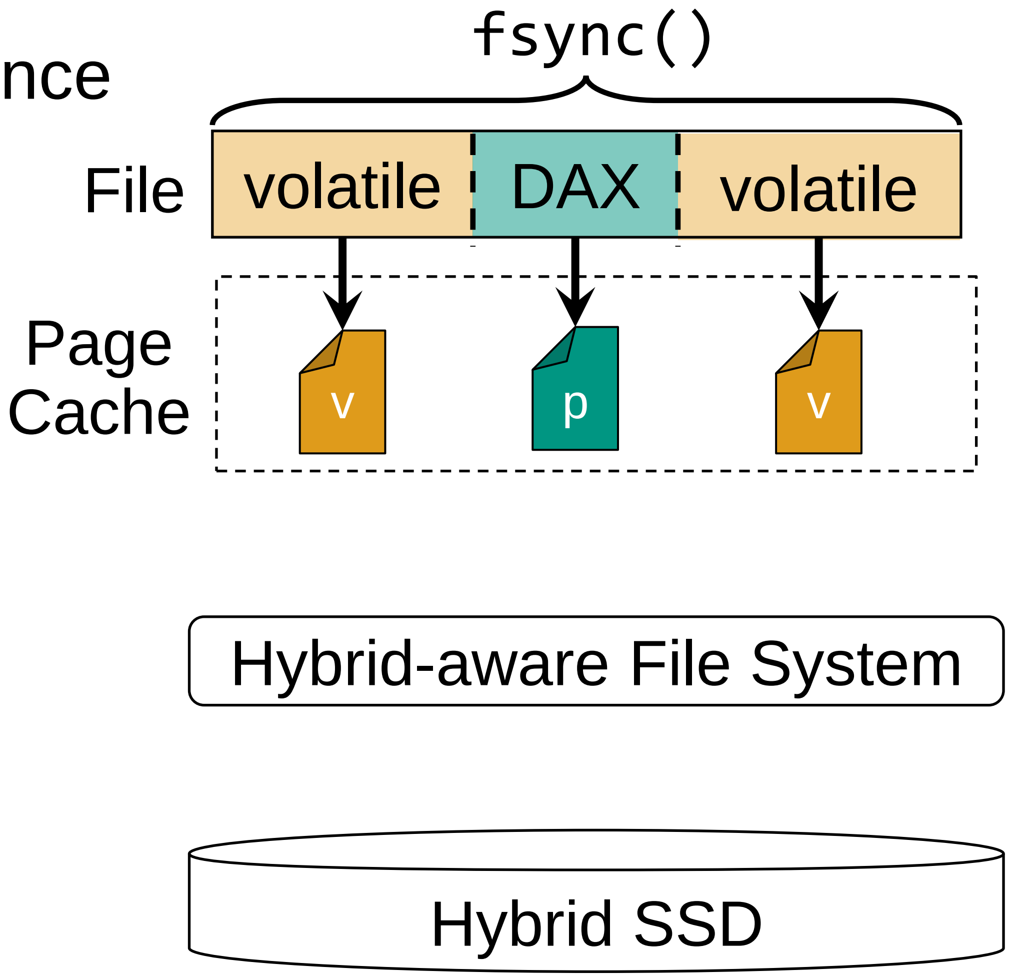


DAX-Aware Page Cache Allocation



Lightweight Synchronous Writeback

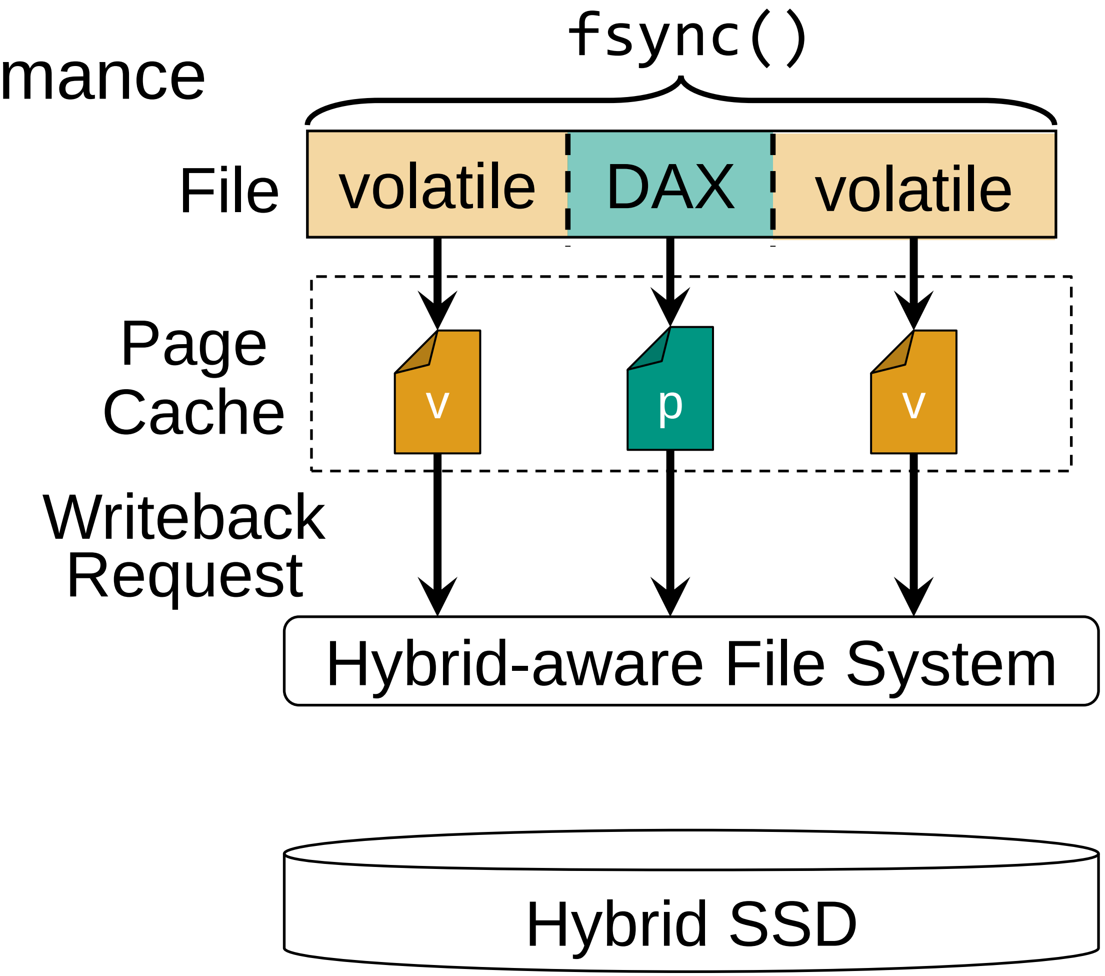
- Synchronous writeback critical for performance
  - On-device cache guarantees persistence
    →skip writeback of DAX pages
  - DAX pages remain dirty



Lightweight Synchronous Writeback

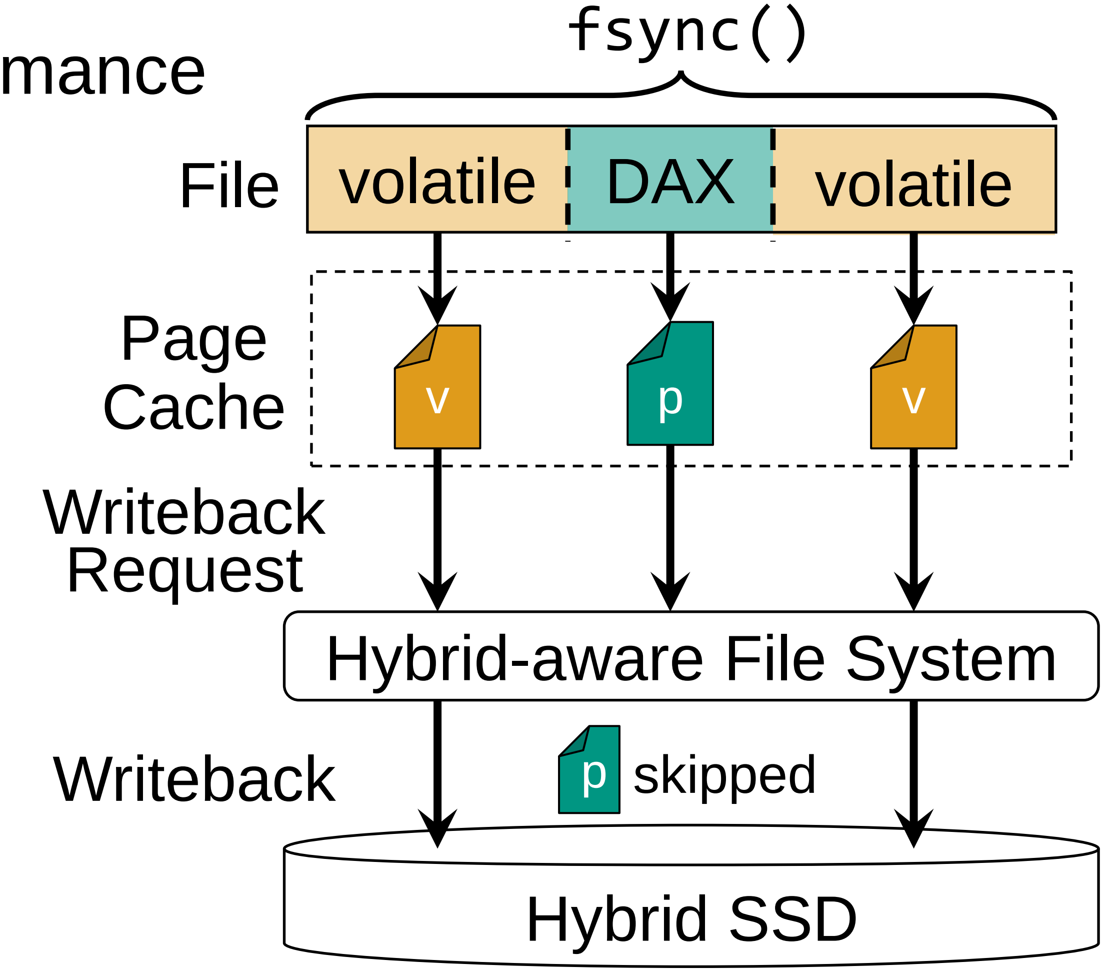
- Synchronous writeback critical for performance
  - On-device cache guarantees persistence

  →skip writeback of DAX pages
  - DAX pages remain dirty



Lightweight Synchronous Writeback

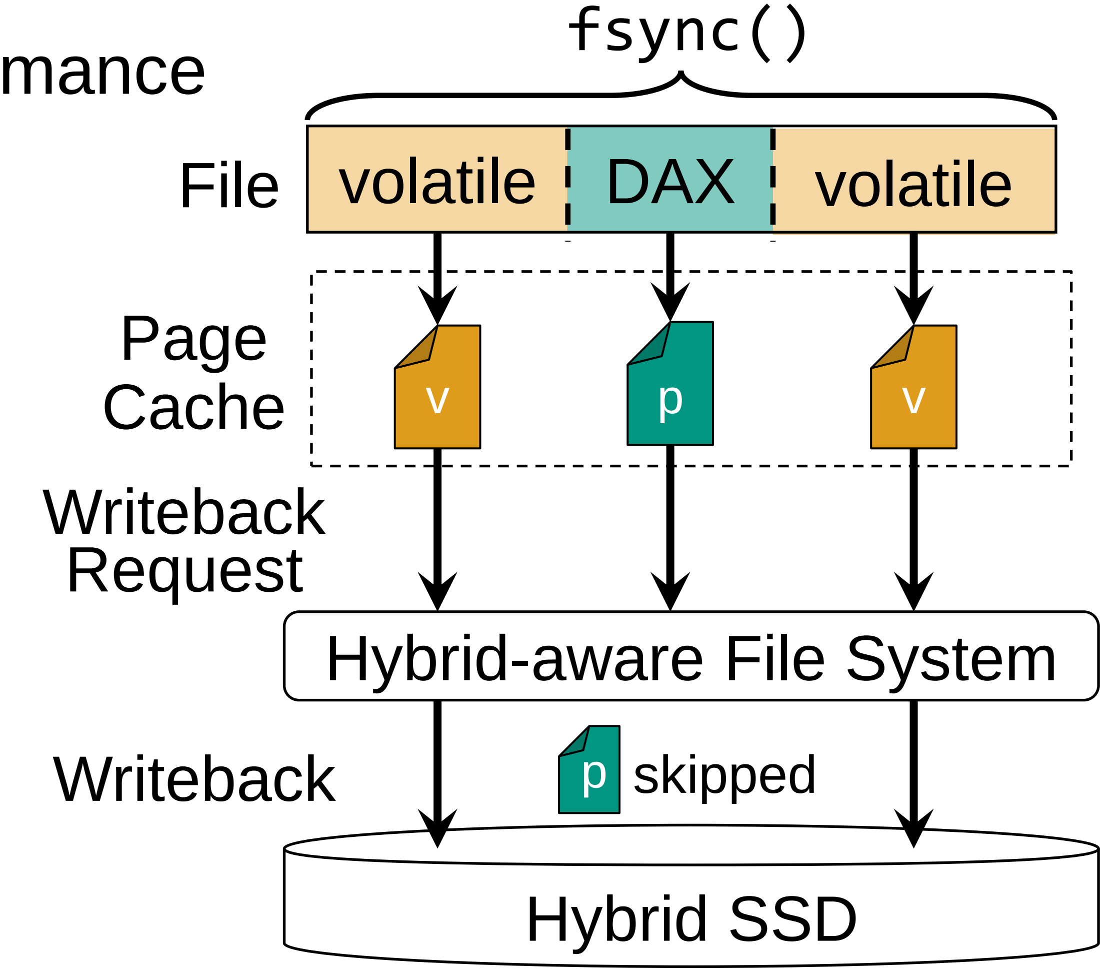
- Synchronous writeback critical for performance
  - On-device cache guarantees persistence
    →skip writeback of DAX pages
  - DAX pages remain dirty



Lightweight Synchronous Writeback

- Synchronous writeback critical for performance
  - On-device cache guarantees persistence
    →skip writeback of DAX pages
  - DAX pages remain dirty

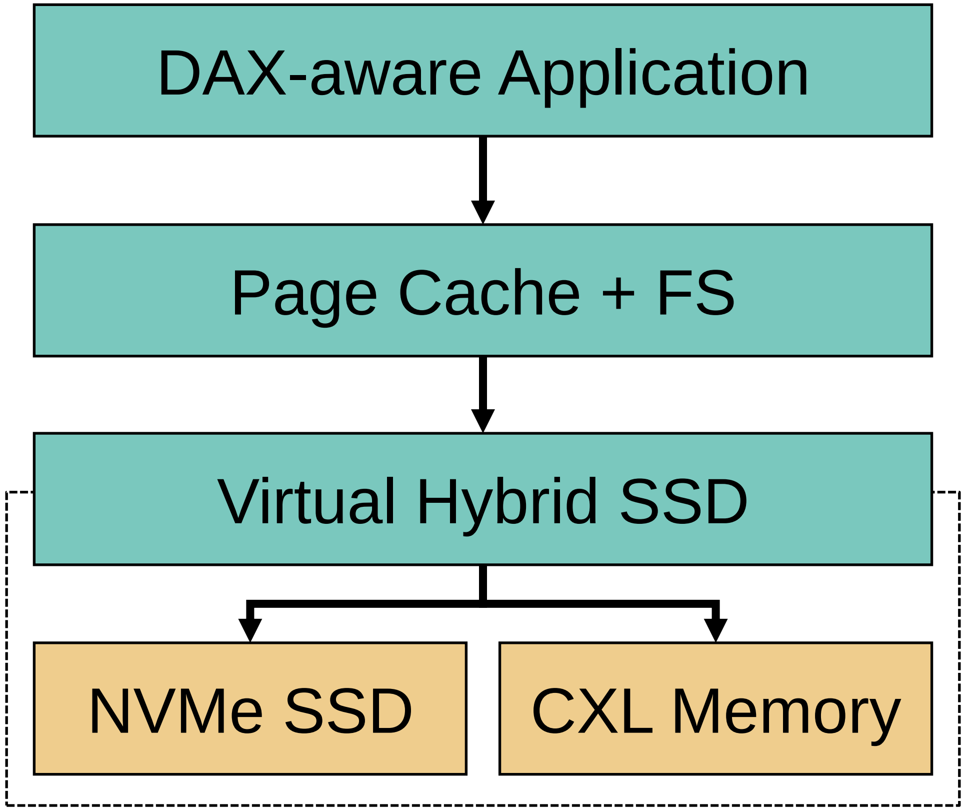
- Asynchronous writeback unchanged
  - Performance not critical
  - Clean pages beneficial for reclaim



Evaluation

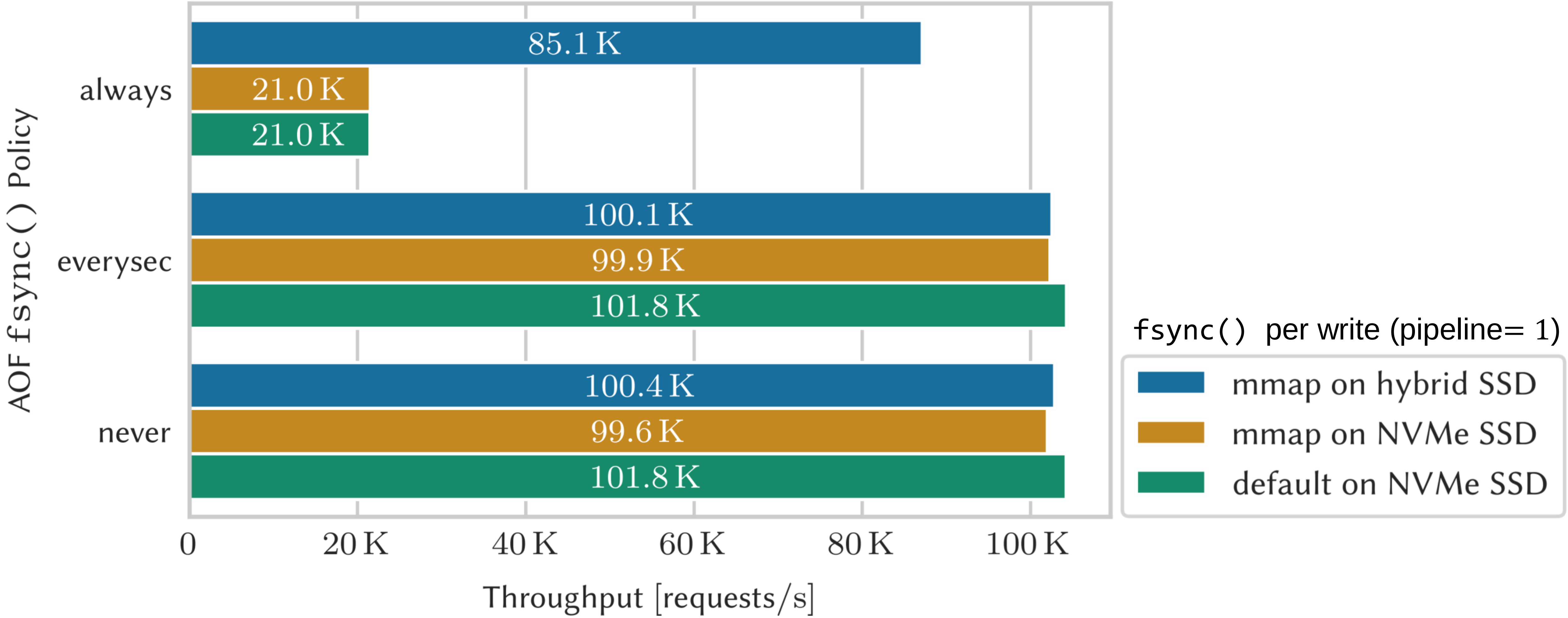
- Emulated hybrid SSD (SSD + CXL mem)

- *Valkey* with Append-Only-File (AOF)
  - AOF writeback policy determines overhead
  - Write-only workload evaluated (worst-case)
  - mmap AOF backend for hybrid SSD

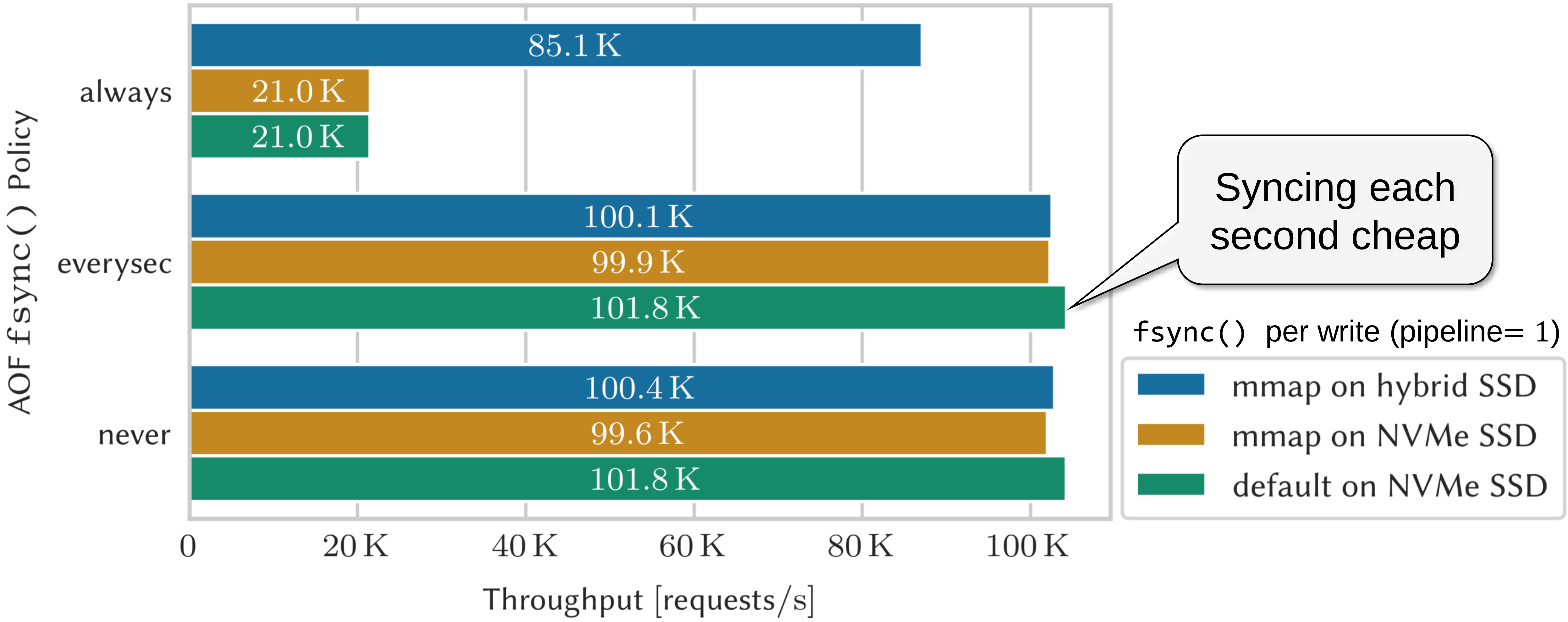


✅ Throughput, tail latencies, CPU and energy efficiency improved

Valkey AOF Throughput

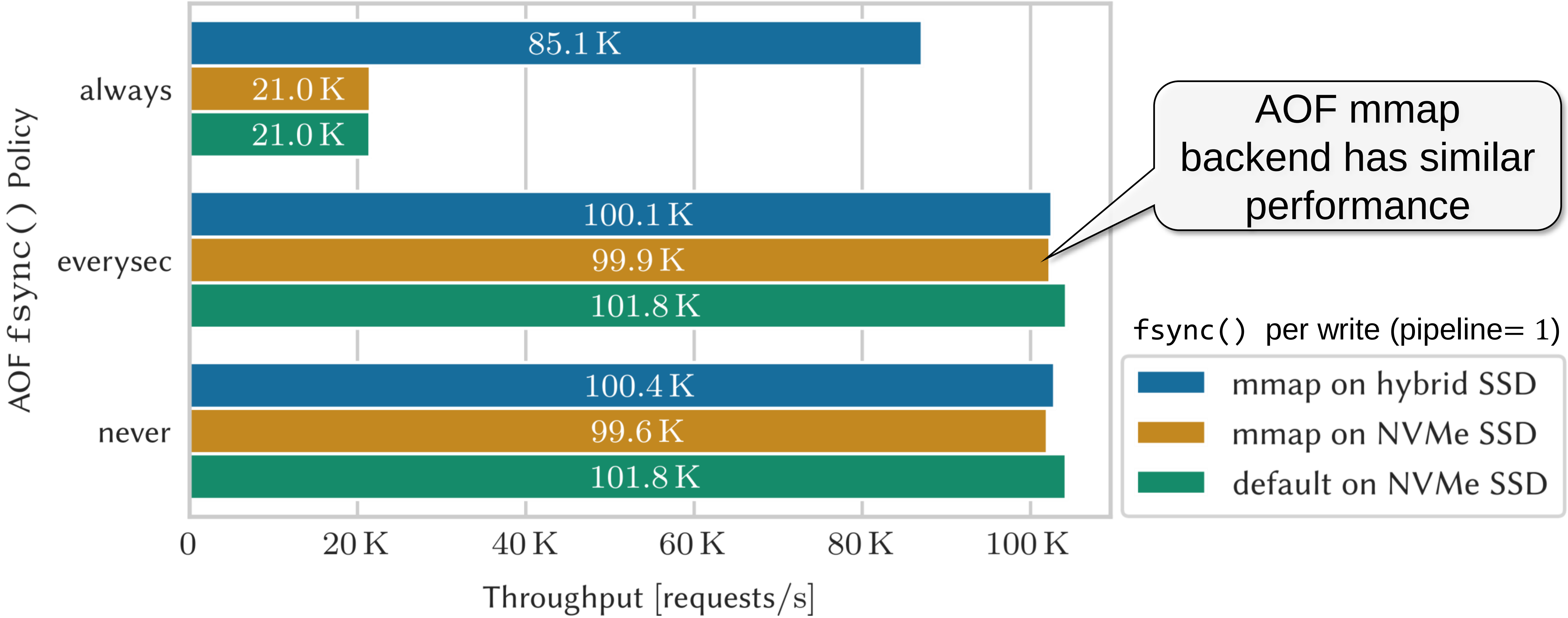


Valkey AOF Throughput



Syncing each second cheap

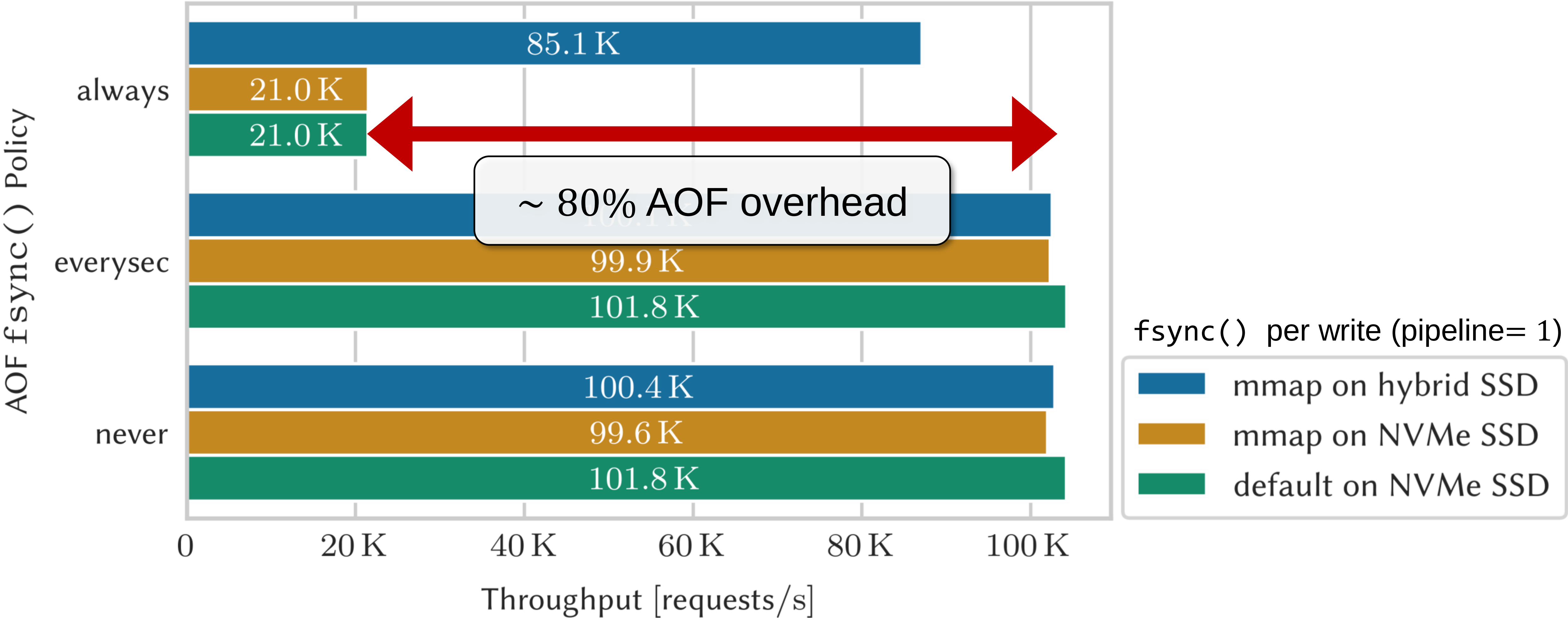
Valkey AOF Throughput



AOF mmap backend has similar performance

`fsync()` per write (pipeline= 1)

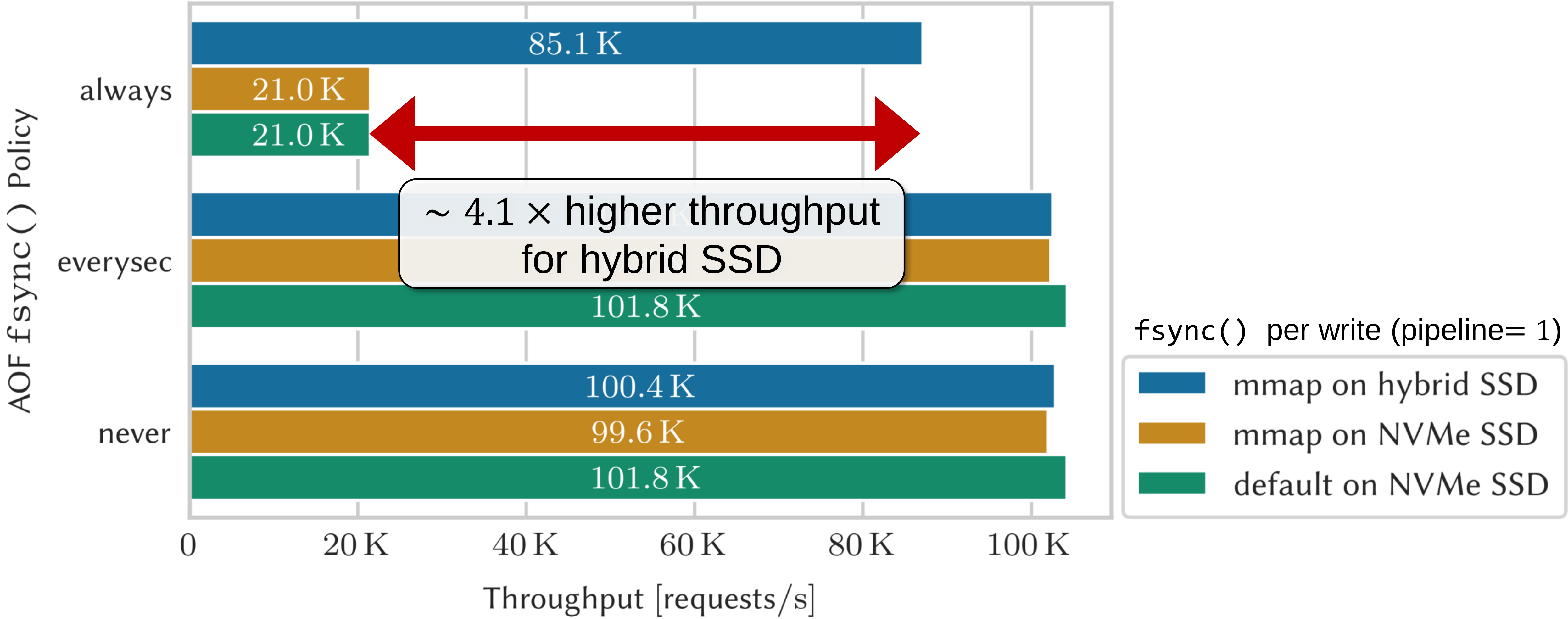
Valkey AOF Throughput



~ 80% AOF overhead

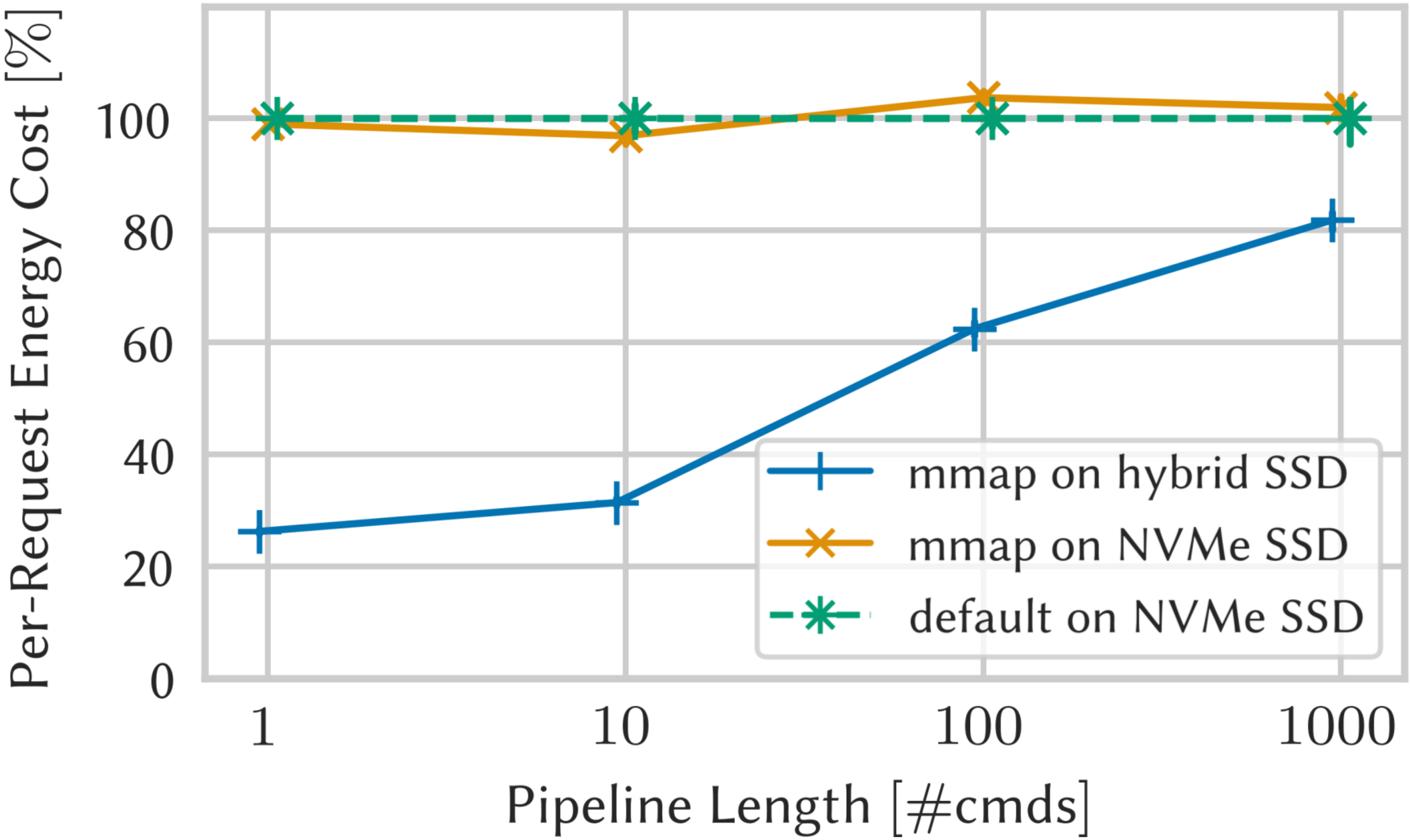
`fsync()` per write (pipeline= 1)

Valkey AOF Throughput

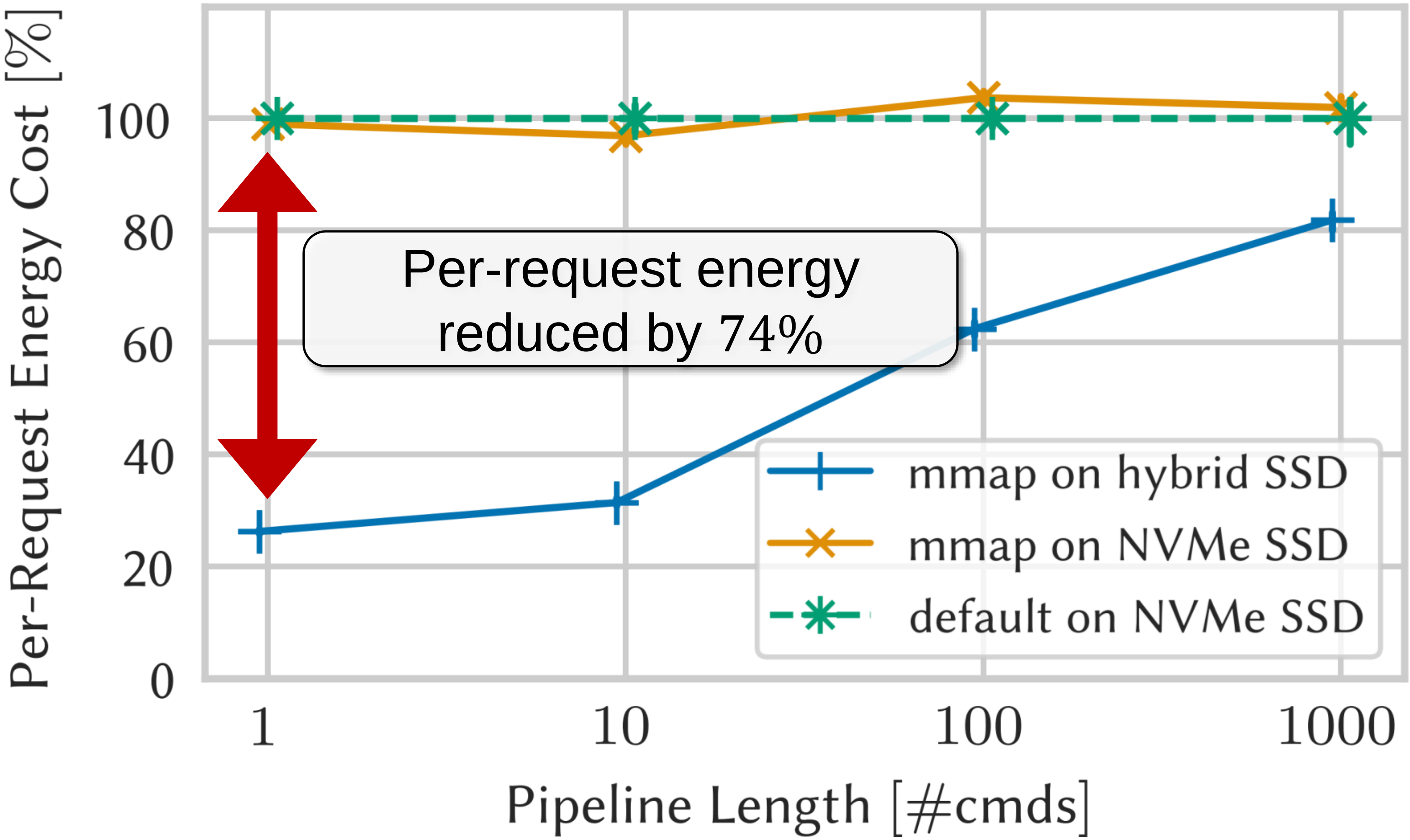


~ 4.1 × higher throughput for hybrid SSD

Valkey Per-Request Energy Consumption

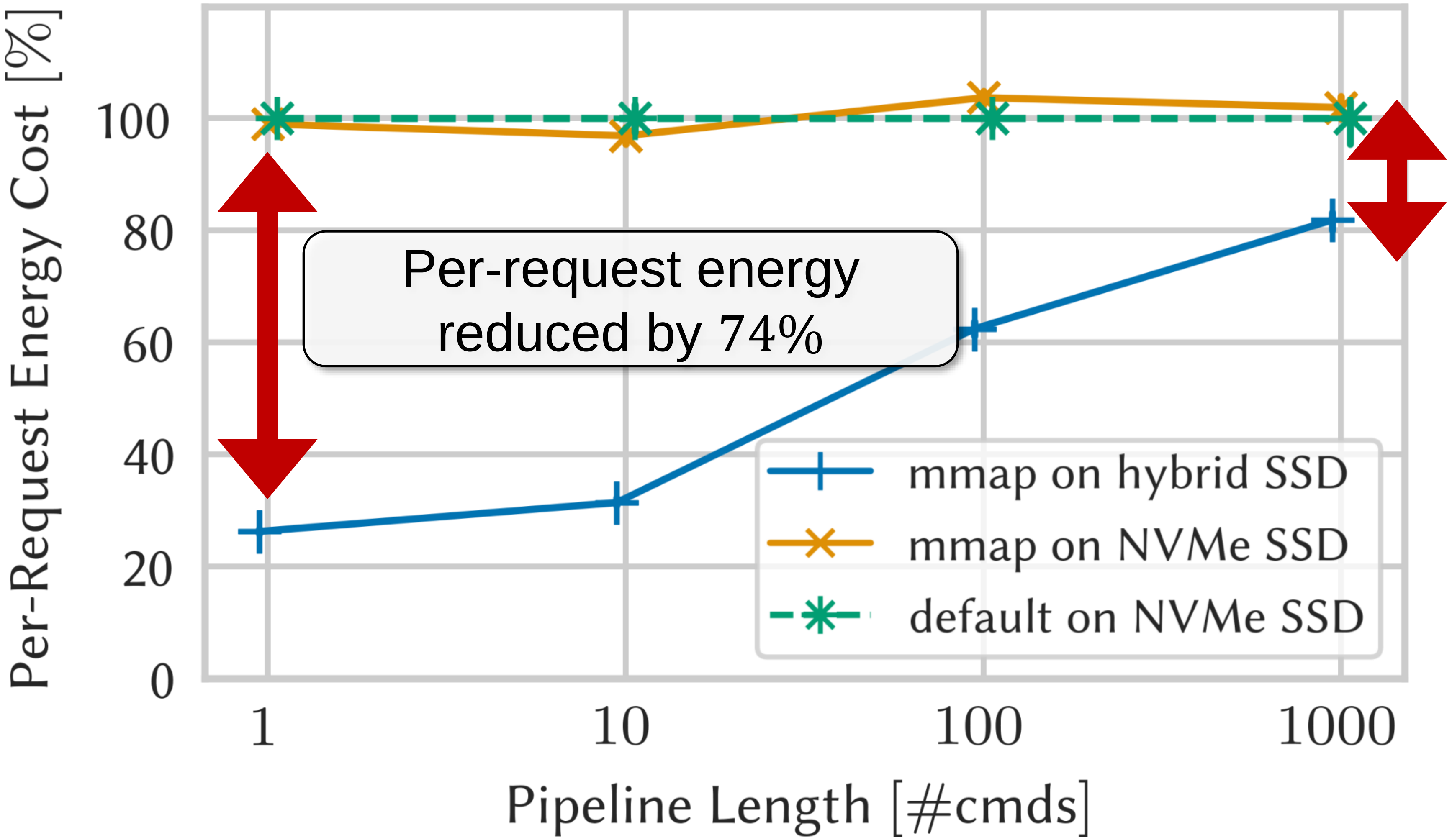


Valkey Per-Request Energy Consumption



Per-request energy reduced by 74%

Valkey Per-Request Energy Consumption



Per-request energy reduced by 74%

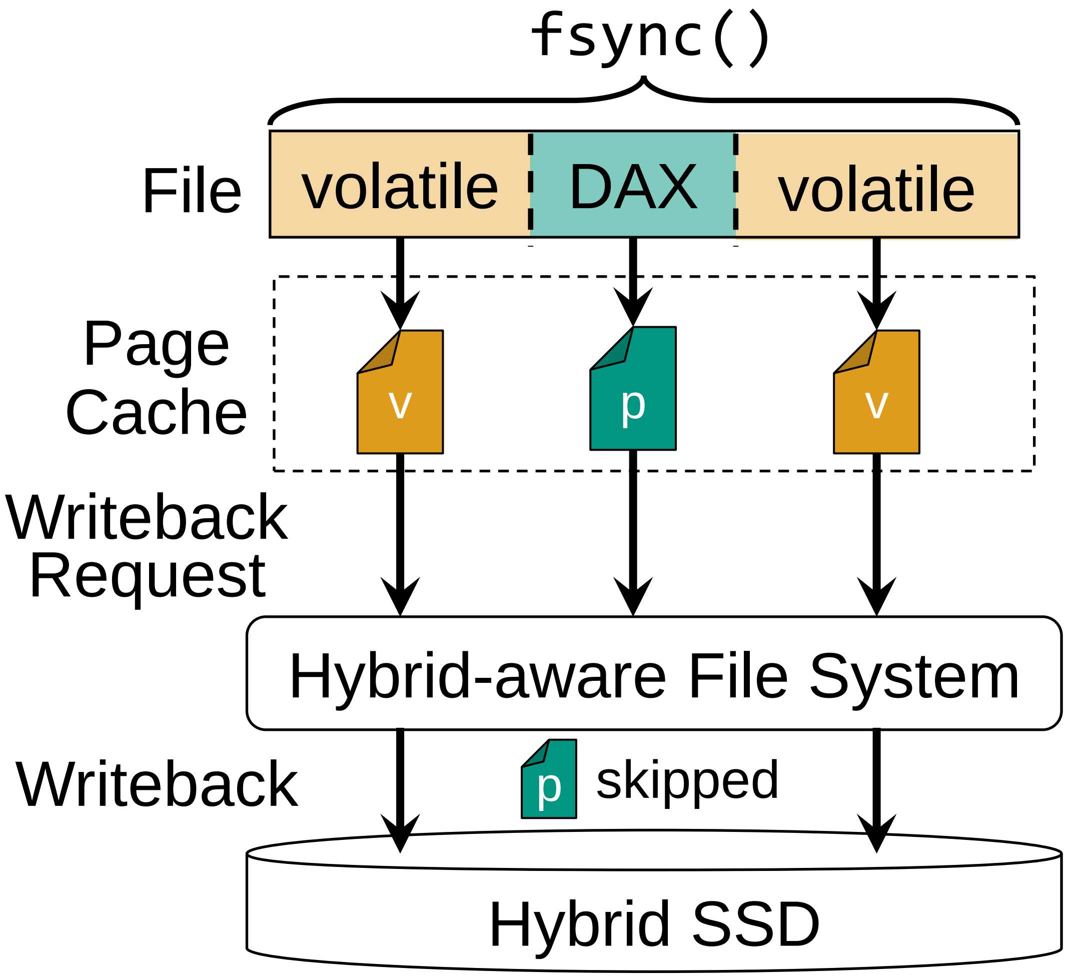
Fewer `fsync()` calls → benefit of hybrid SSD decreases

Future Work

- 💡 Transparently establish of DAX mappings
- 💡 Study hardware design space for cache management
- 💡 Reevaluate on real-world hybrid SSDs
- 💡 Explore hybrid SSDs in consumer context

Summary

- Hybrid SSD = NVMe + CXL.mem
- Existing OS abstractions unsuitable
  - Limited control over resource usage
  - CPU stalled on cache miss
- Our design:
  - Fine-granular resource management
  - Cache managed by OS
- Up to 4.1 × higher *Valkey* throughput and 78% lower energy consumption


fsync()
File
volatile
DAX
volatile
Page
Cache
v
p
v
Writeback
Request
Hybrid-aware File System
Writeback
p
skipped
Hybrid SSD