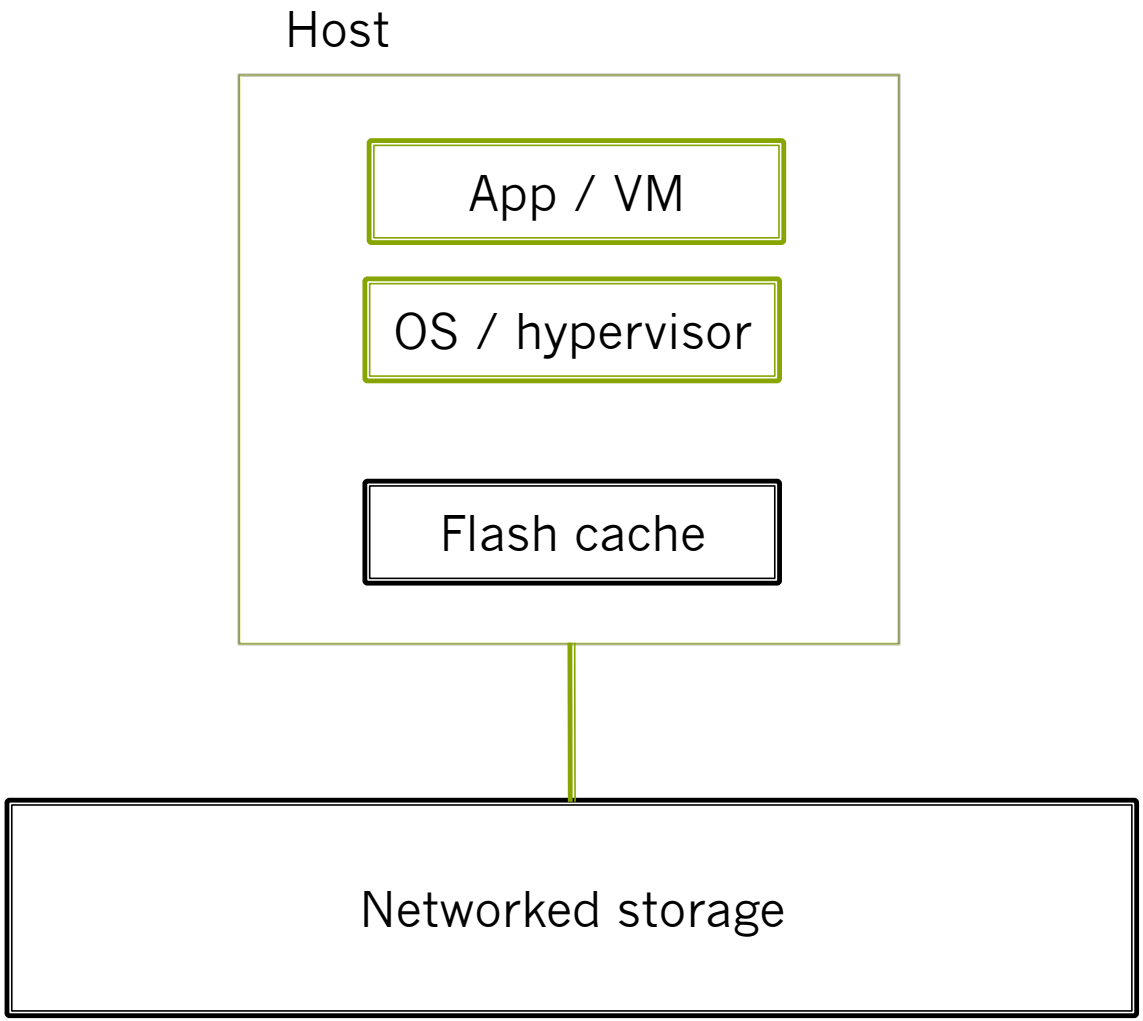
Write Policies for Host-side Flash Caches

Ricardo Koller, FIU (currently at VMware)
Leonardo Marmol, FIU
Raju Rangaswami, FIU
Swaminathan Sundararaman, FusionIO
Nisha Talagala, FusionIO
Ming Zhao, FIU

Introduction

- Virtualized environments typically use networked storage
- Higher VM consolidation demands
- Flash caches used to alleviate VMs performance demands
- Recent literature has argued in favor of write-through [Byan12, VFCache12, Hensbergen06, Srinivasan12].

Architecture

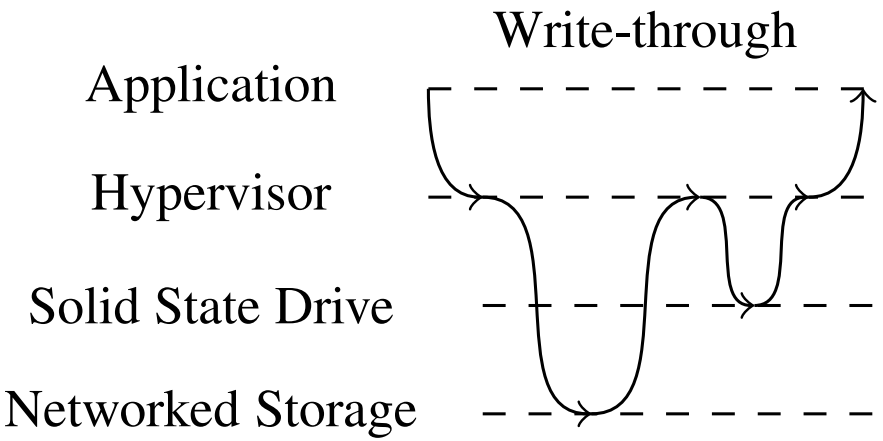


Outline

1. Write-back caching motivation
2. Proposed write-back modifications
   1. Ordered write-back
   2. Journaled write-back
3. Consistency analysis
4. Evaluation
5. Summary and conclusions

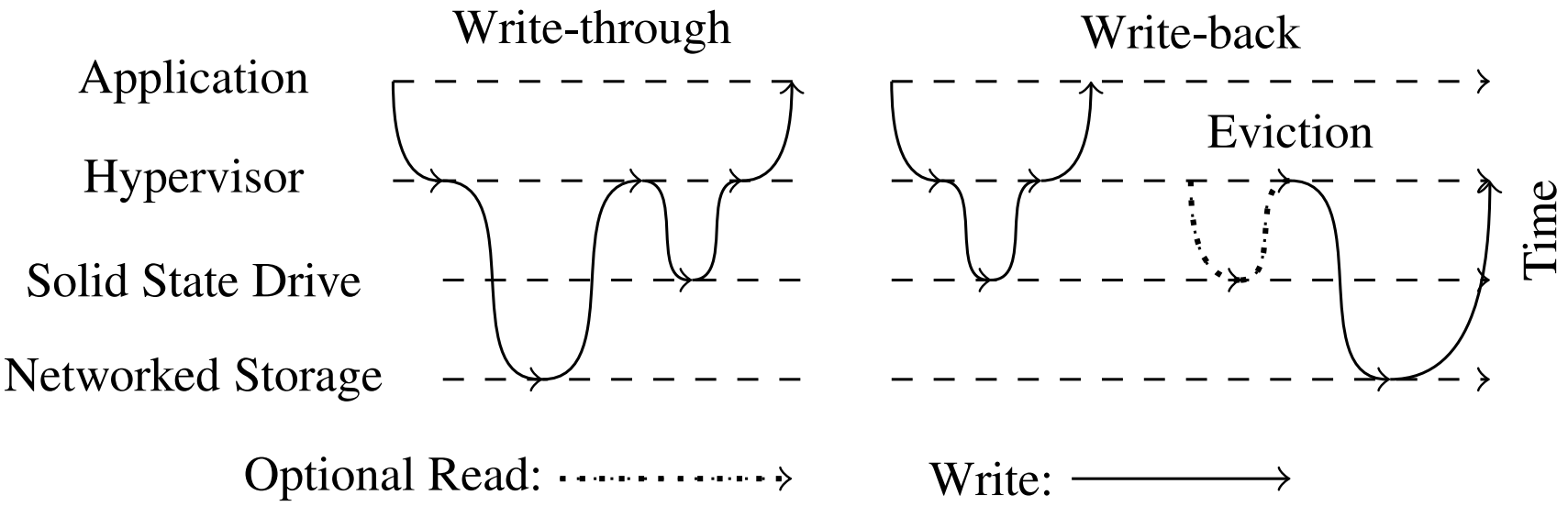
Write caching

- The advantage of caching a read is unquestionable
- What about writes?



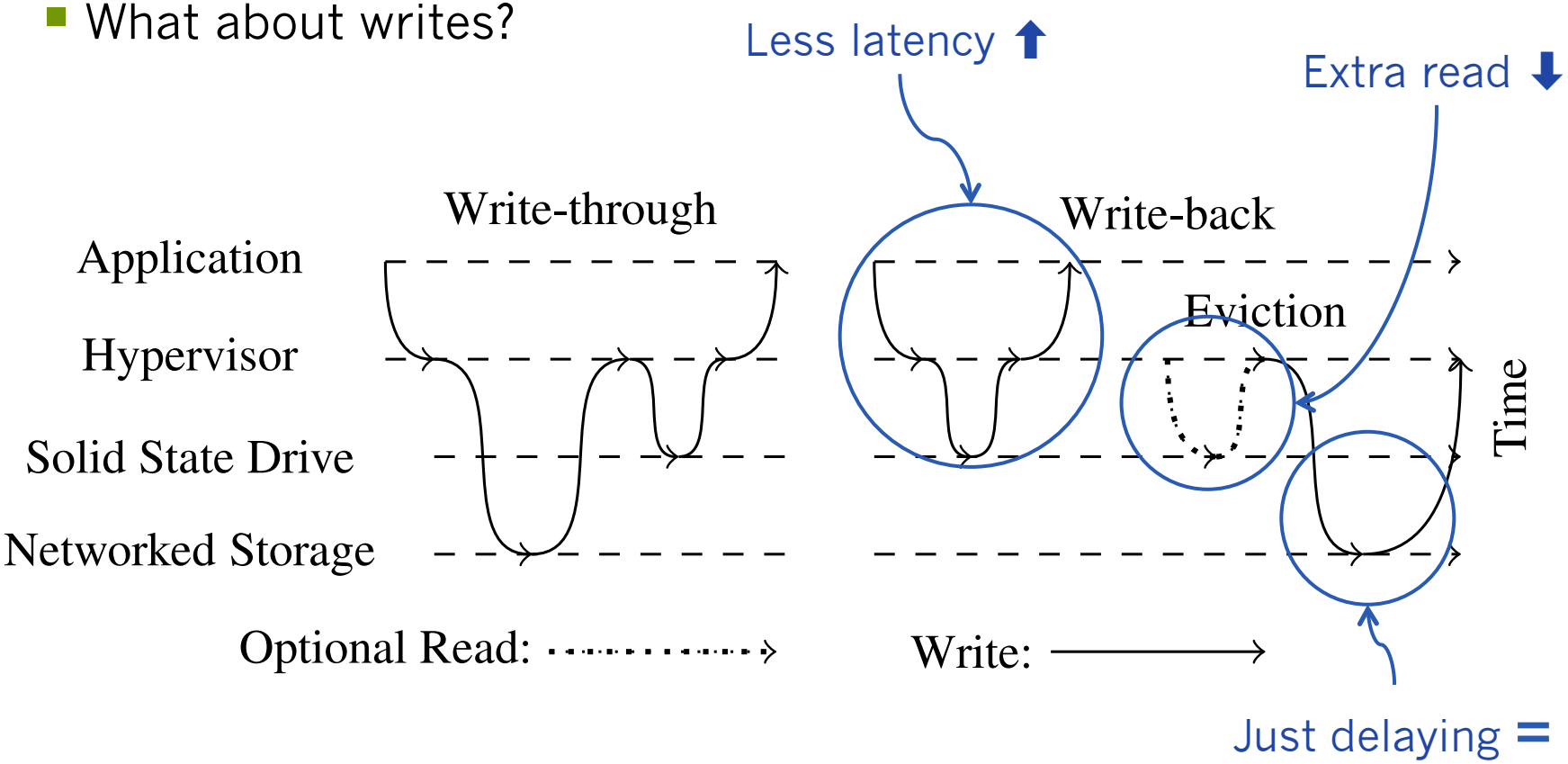
Write caching

- The advantage of caching a read is unquestionable
- What about writes?



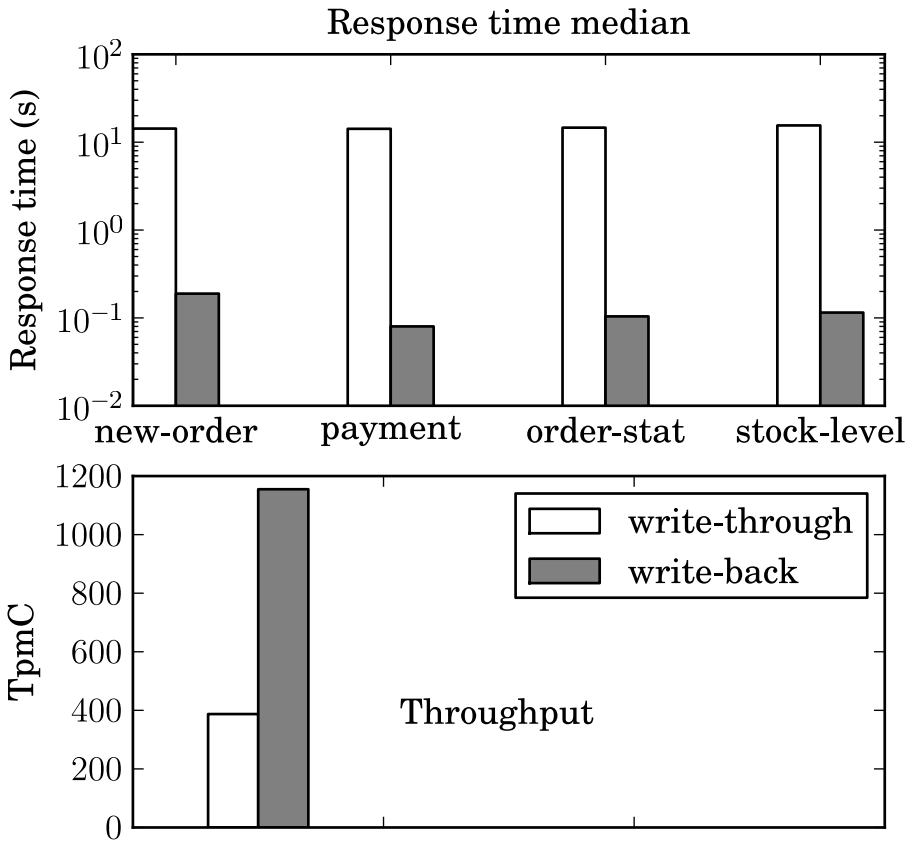
Write caching performance

- The advantage of caching a read is unquestionable
- What about writes?



Write through vs. write back: Performance

TPCC results: write-back performs better than write-through



75x

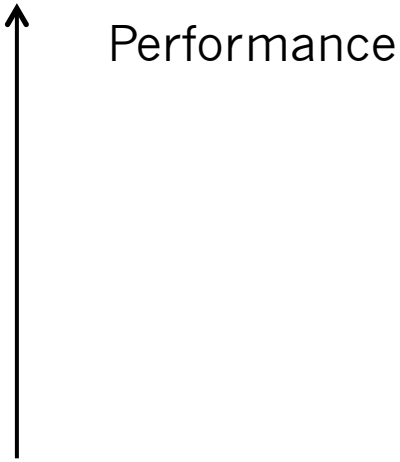
3x

Reasons for write-back good performance

- Lower latency
  - Writes do not need to reach networked storage
- Write coalescing
  - Reduction of writes to networked storage
- Parallel evictions
  - Batched writes to networked storage

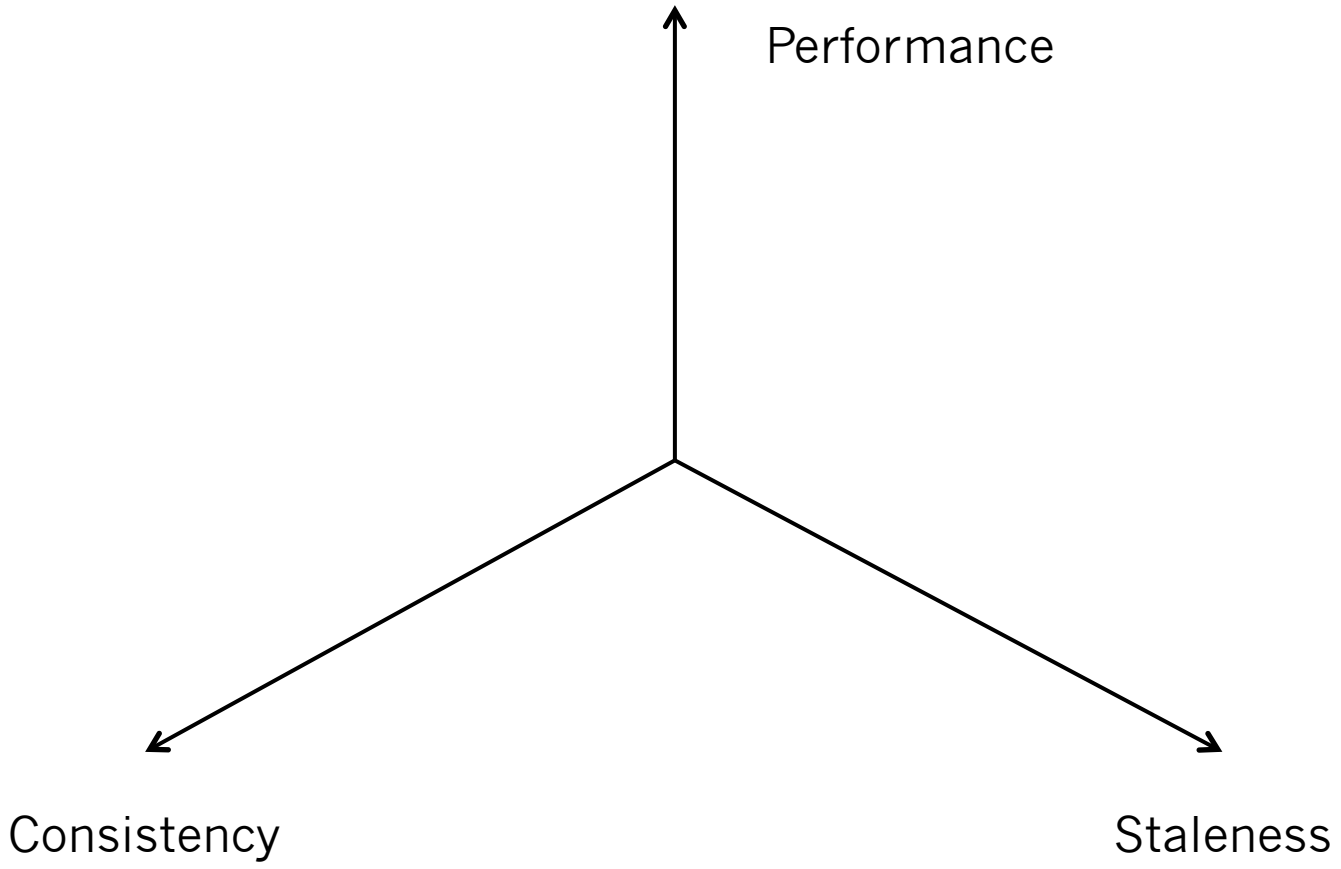
Tradeoffs in conventional write caching policies

- Performance is not the only dimension

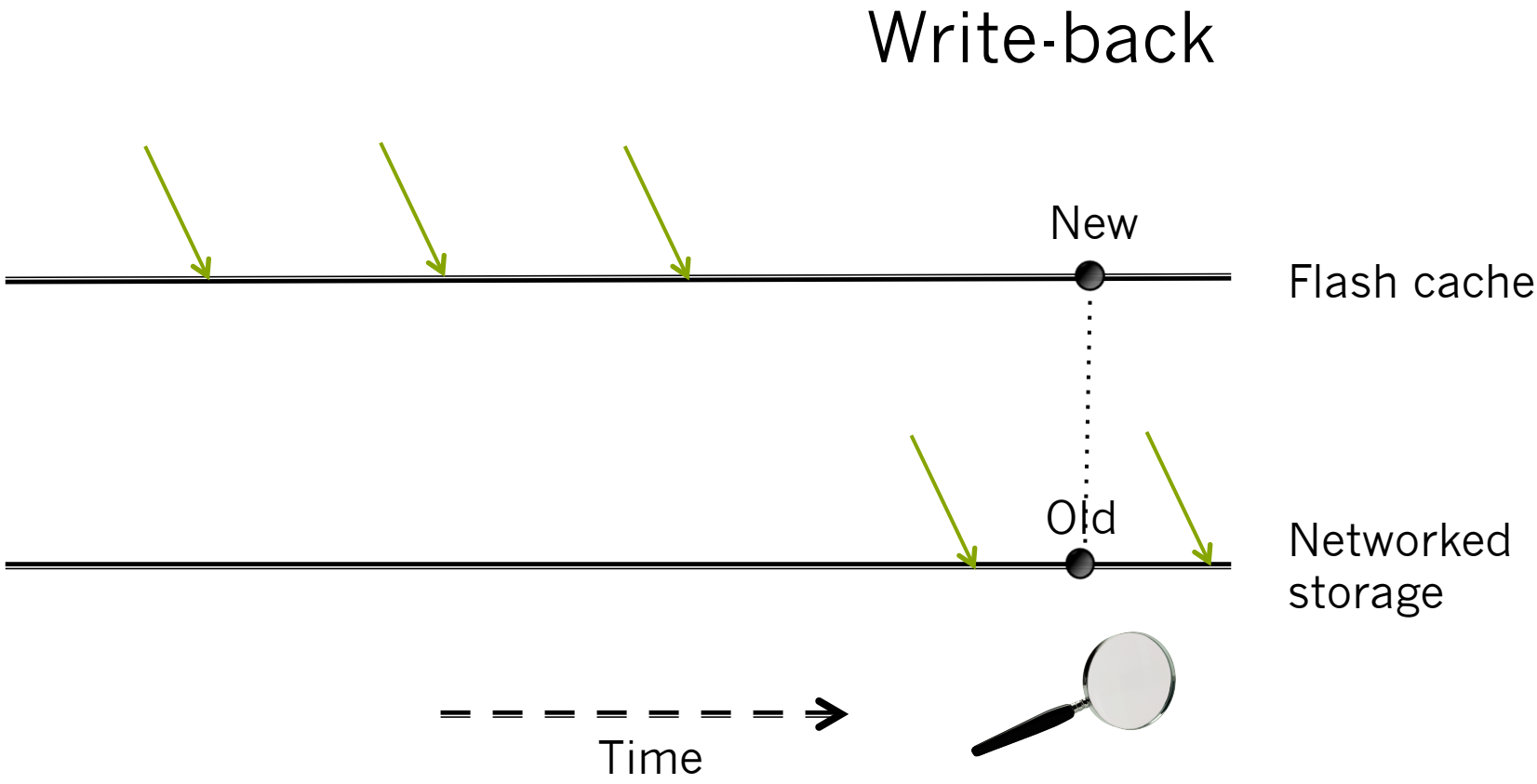


Tradeoffs in conventional write caching policies

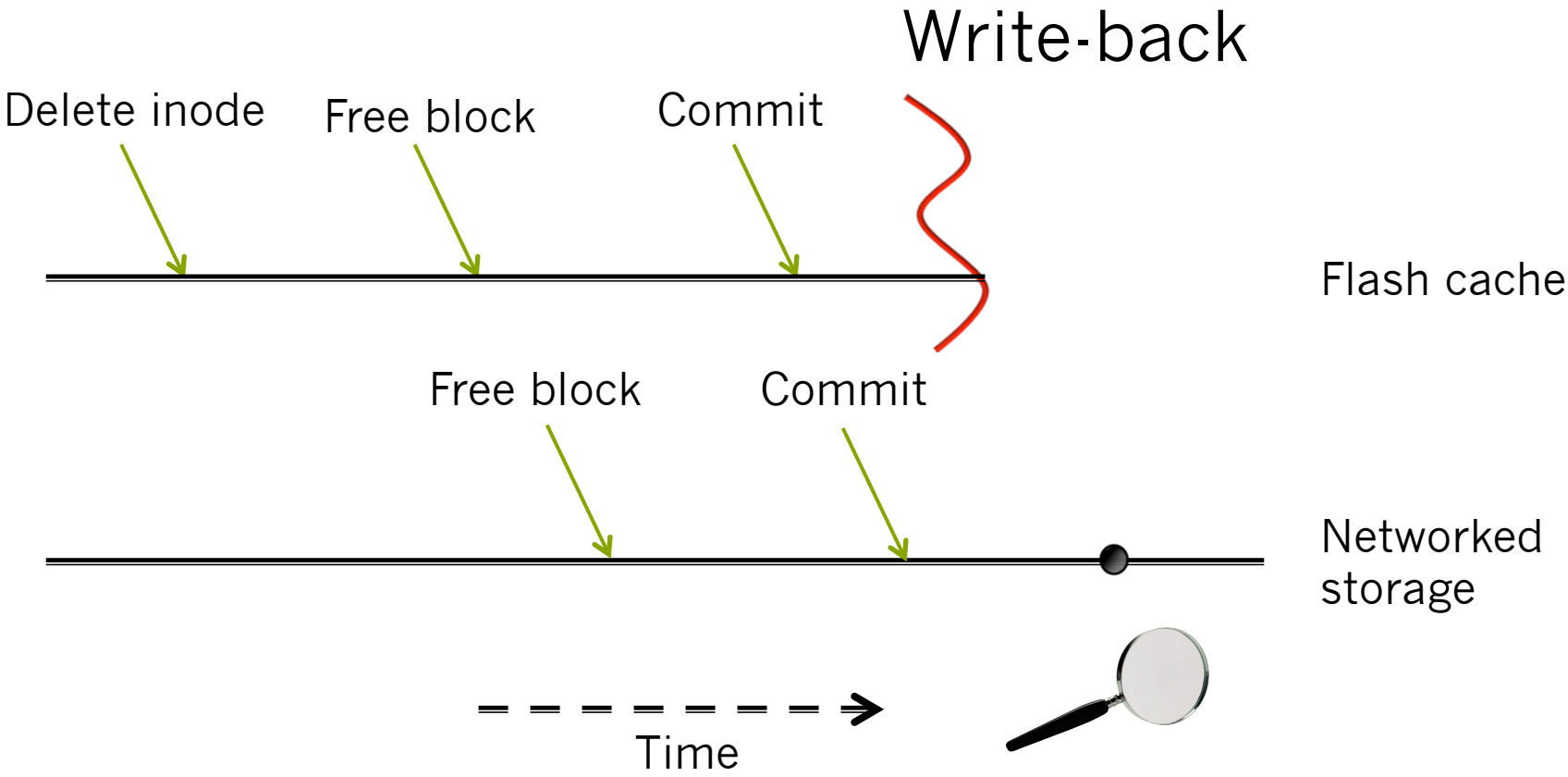
- Performance is not the only dimension



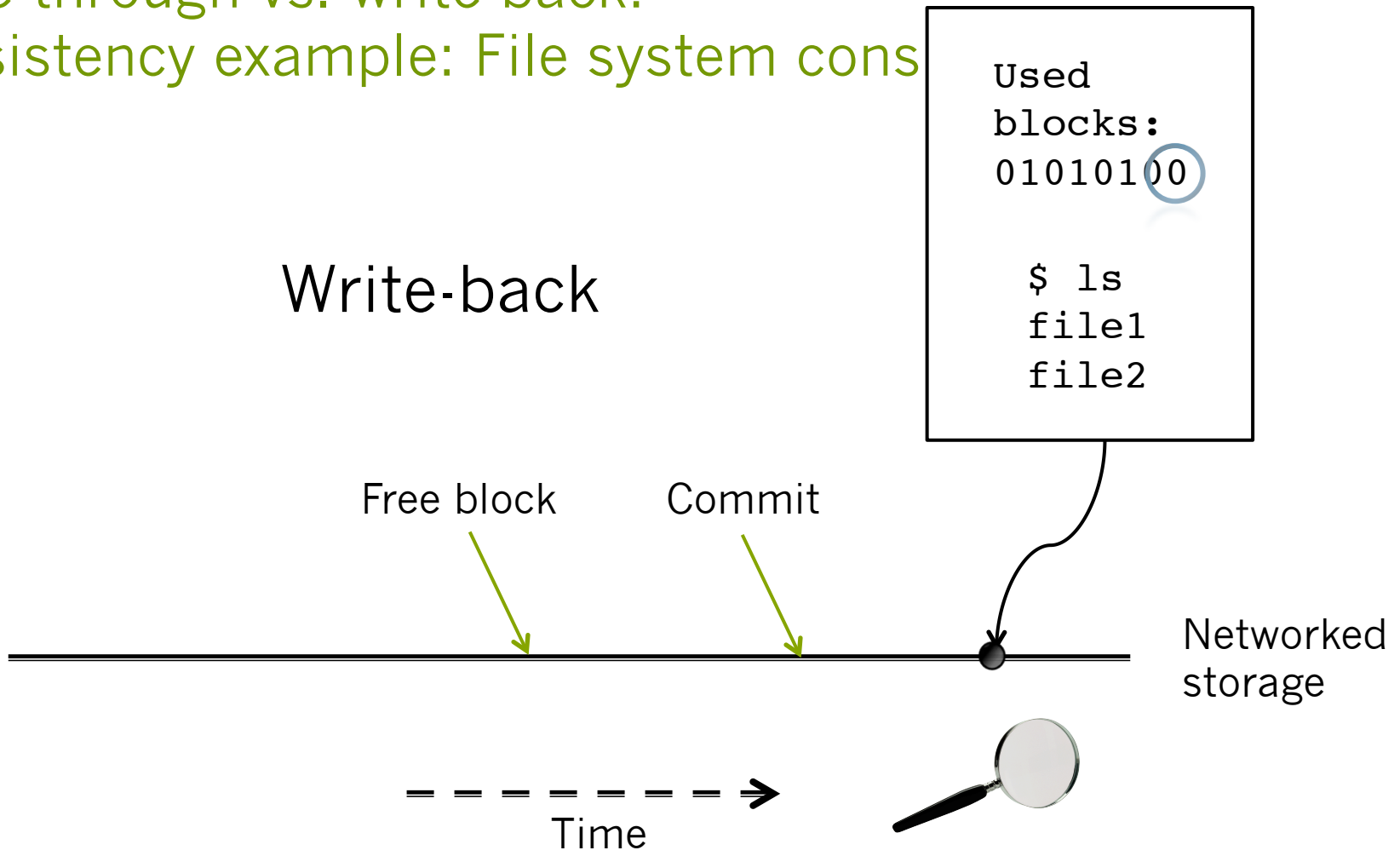
Write through vs. write back: Staleness



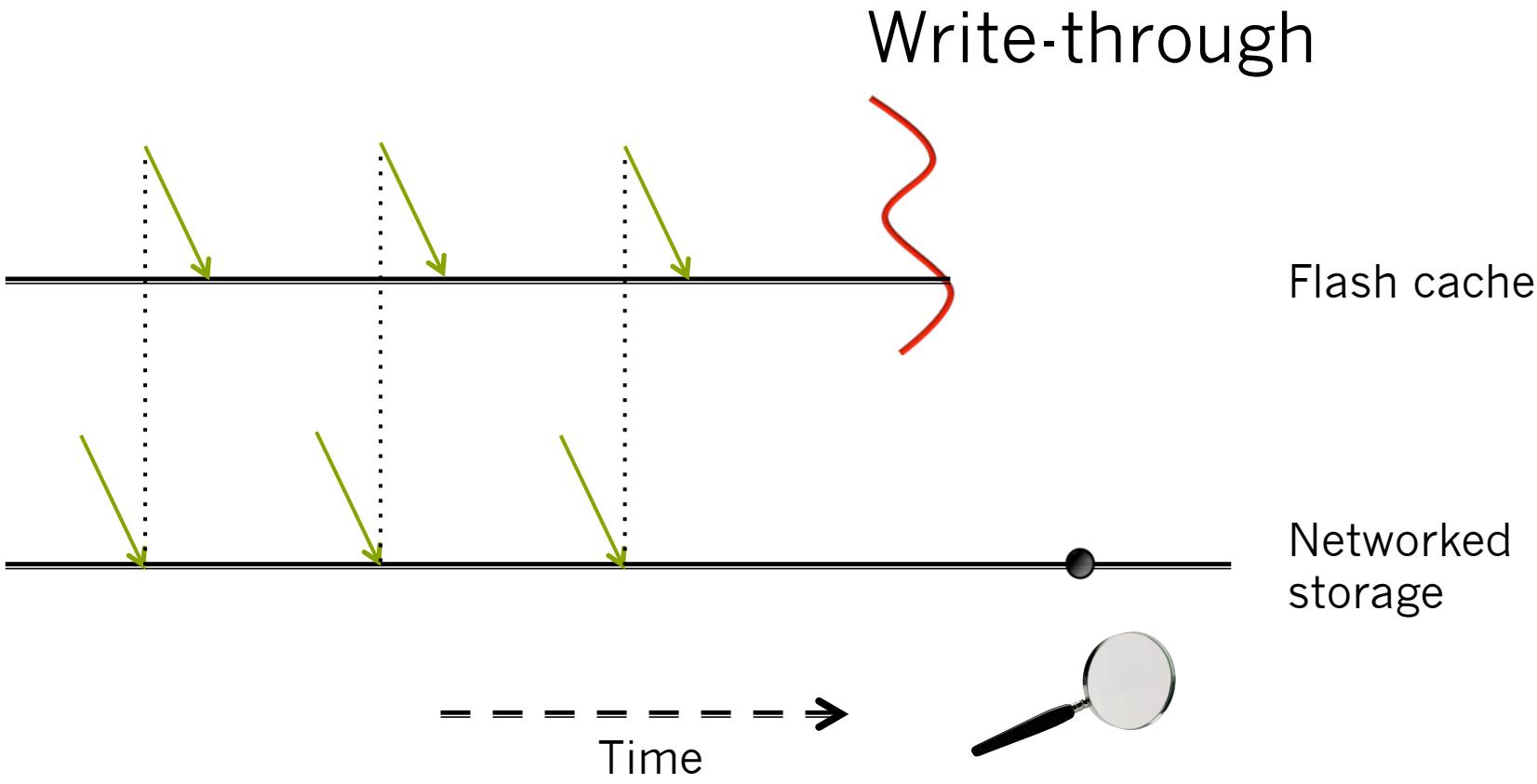
Write through vs. write back:
Consistency example: File system consistency



Write through vs. write back:
Consistency example: File system cons

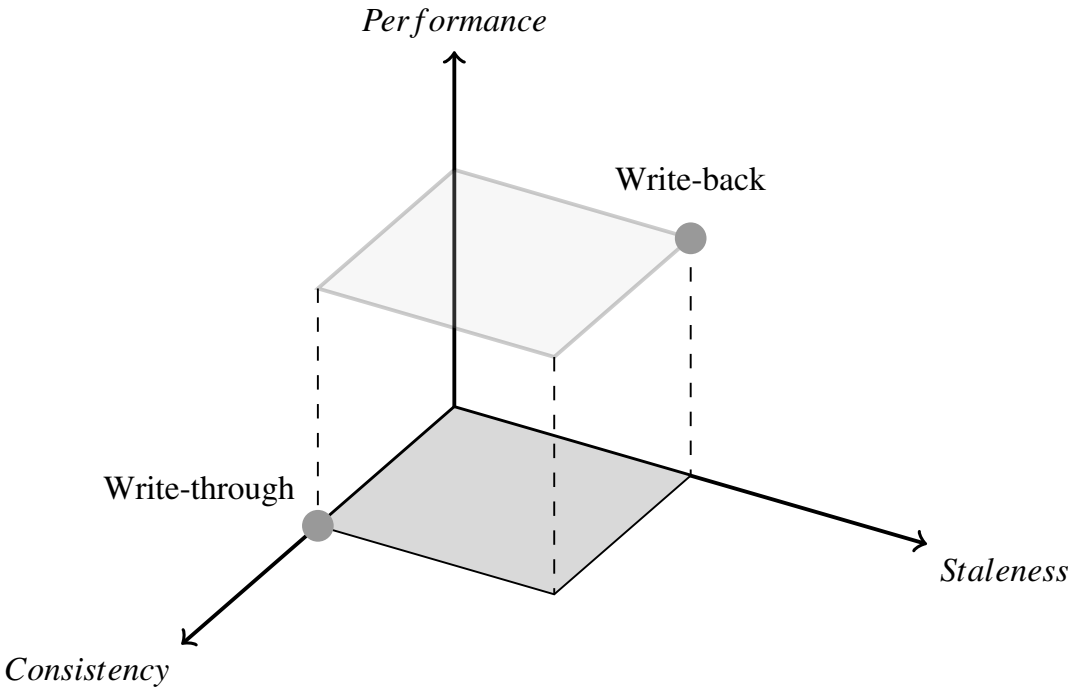


Write through vs. write back: Consistency and no staleness



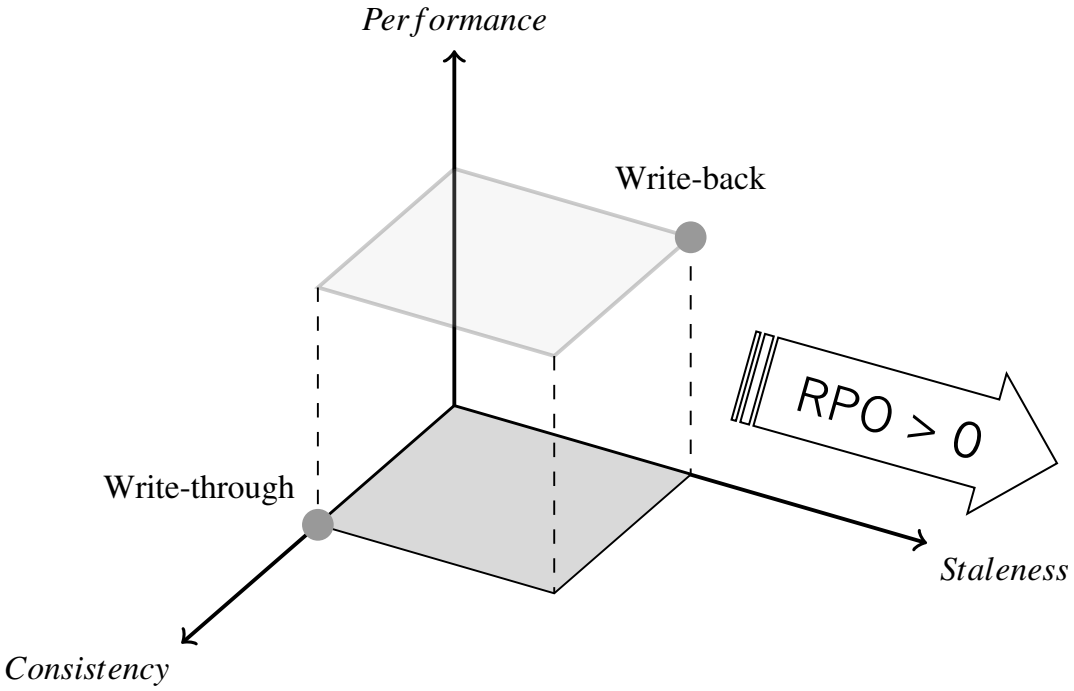
Other options?

- Write-back and write-through are opposites



Other options?

- Asynchronous remote mirroring ensure consistency but delays writes to the remote server [Ji03, Patterson02]
- Some applications [Keeton04] allow RPO (Recovery Point Objective) > 0

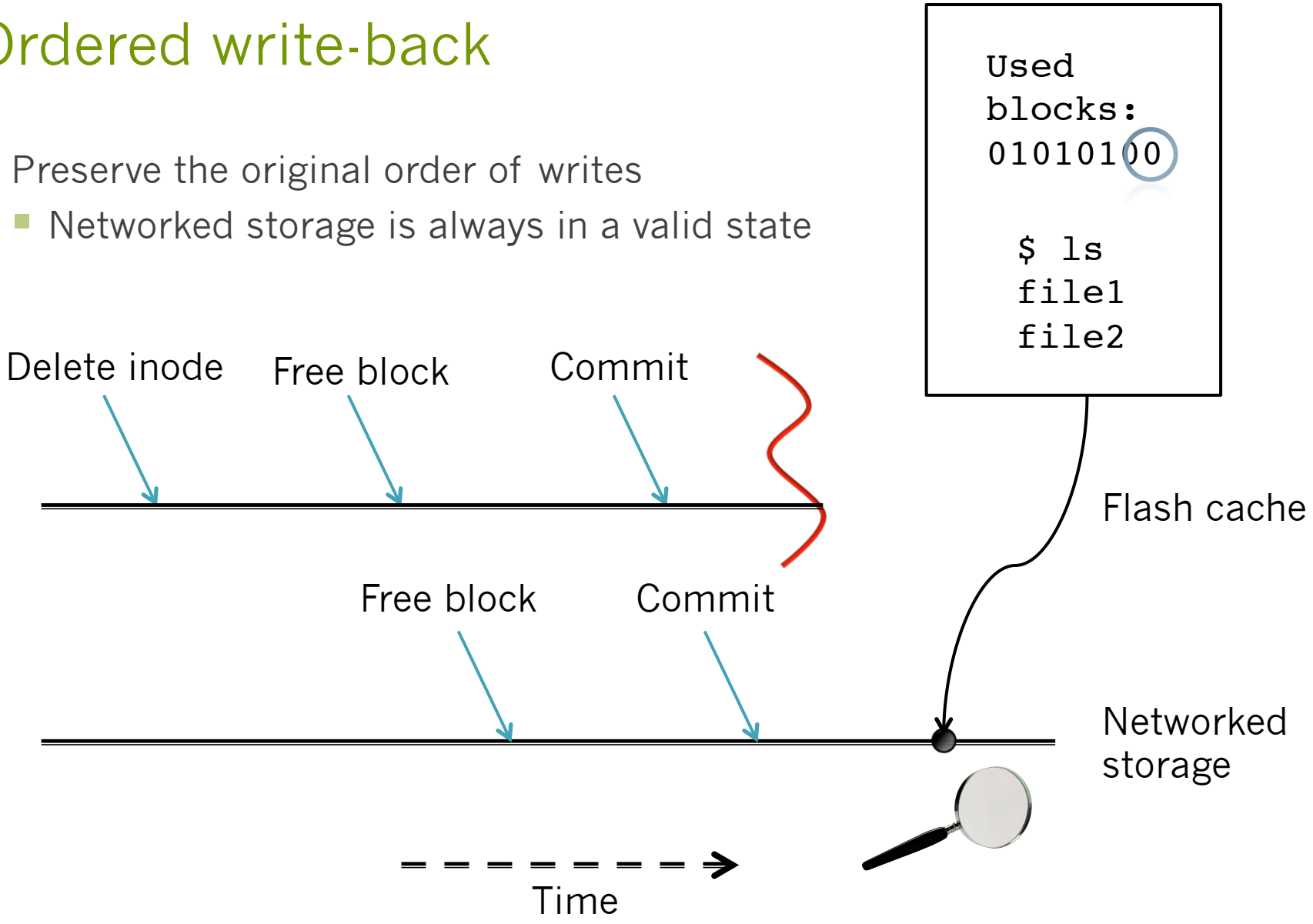


Desired characteristics of new write-back

1. At least point-in-time consistency at the networked storage
2. Delay writes to networked storage
3. Write coalescing
4. Parallel evictions

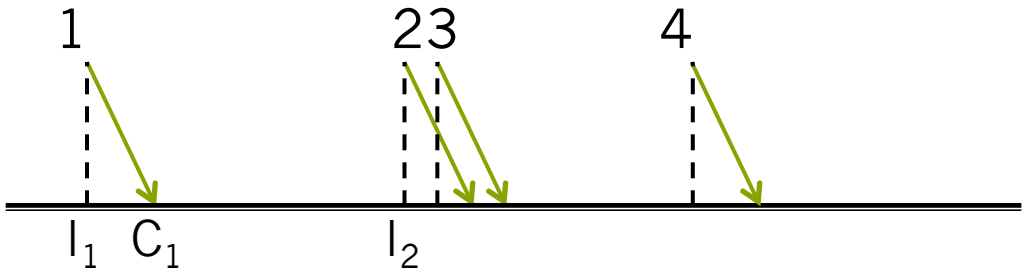
Ordered write-back

- Preserve the original order of writes
  - Networked storage is always in a valid state



Storing I/O Ordering using Completion-issue invariants

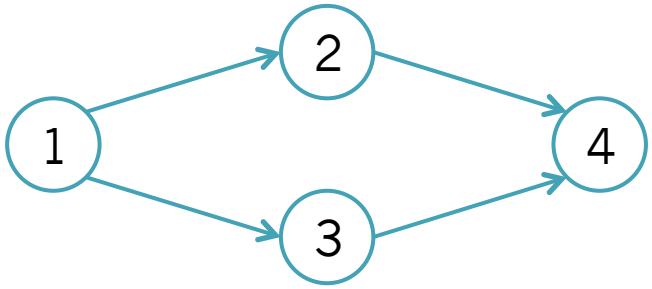
- Completion-issue invariants:



- 2 issued (I_2) after 1 completes (C_1)
- 3 issued after 1 completes
- 4 issued after 1, 2, and 3 complete

Storing I/O Ordering using Completion-issue invariants

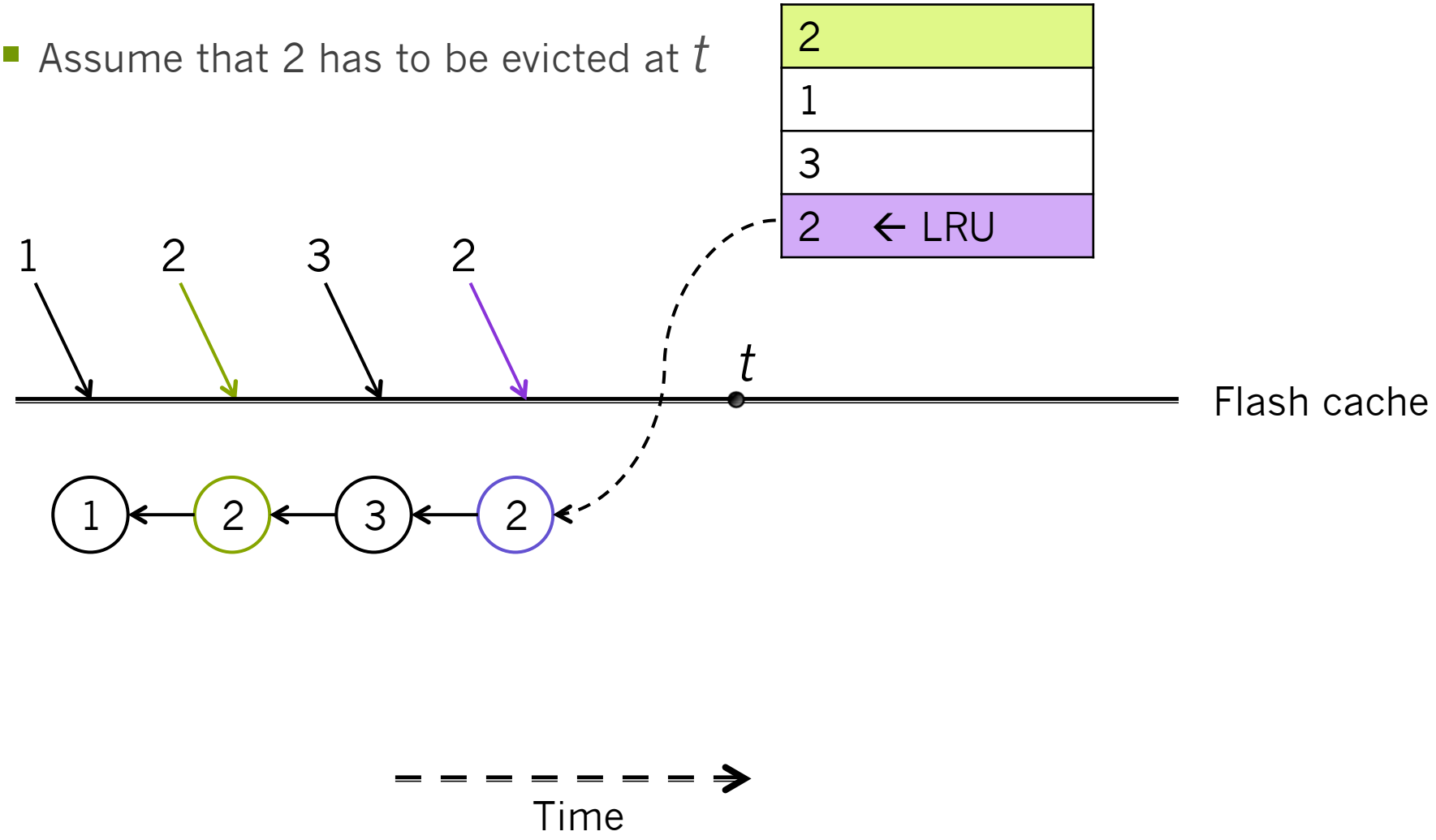
- Completion-issue invariants stored in a graph:



- 2 issued (I_2) after 1 completes (C_1)
- 3 issued after 1 completes
- 4 issued after 1, 2, and 3 complete

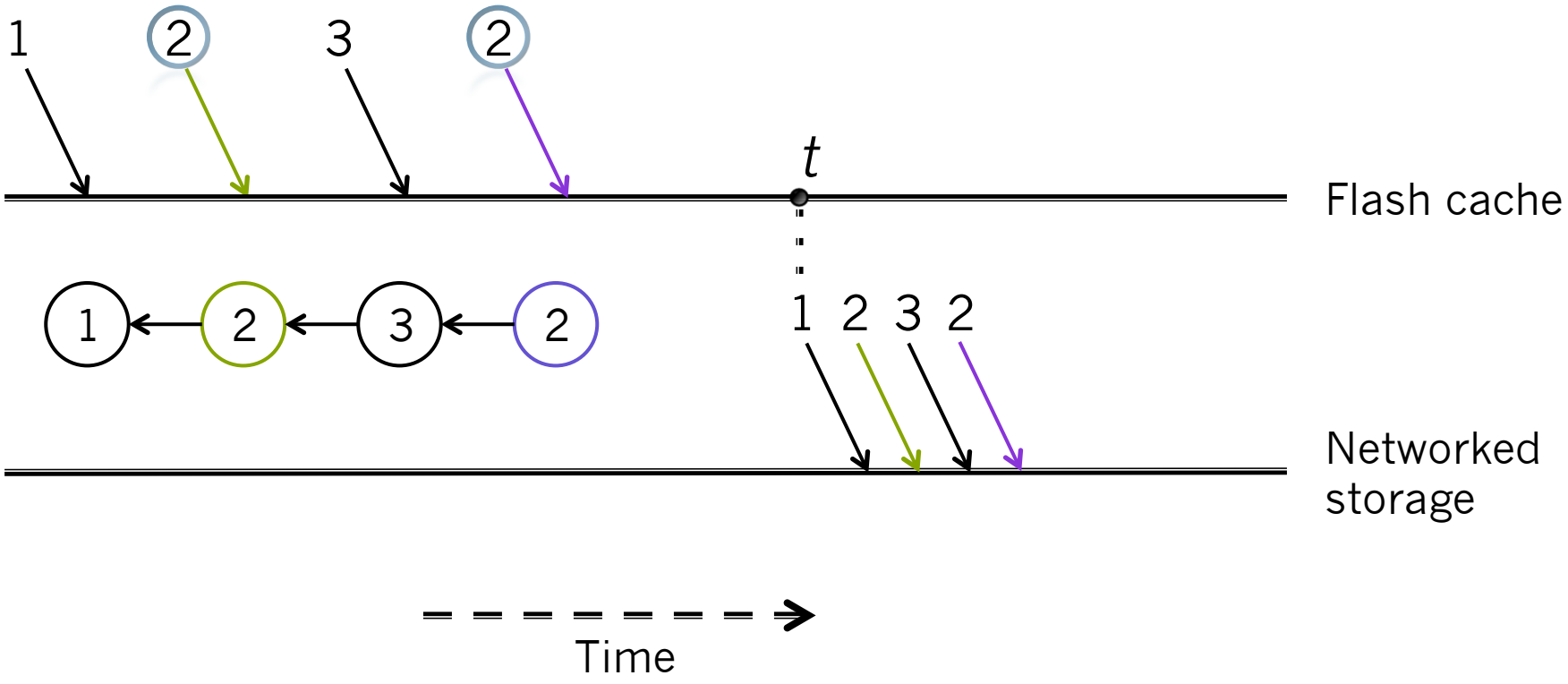
Ordered Write-back

- Assume that 2 has to be evicted at t

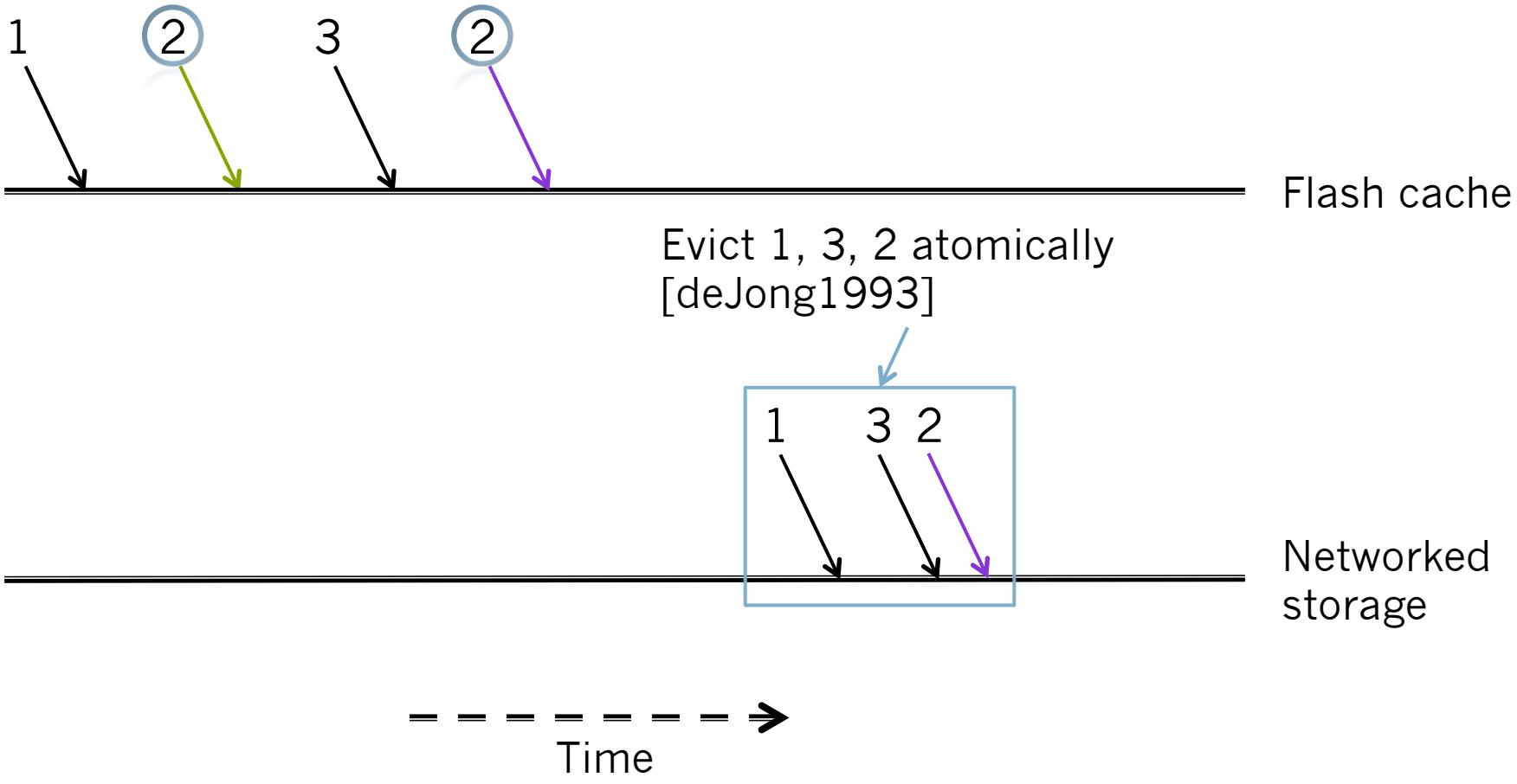


Ordered evictions: problem

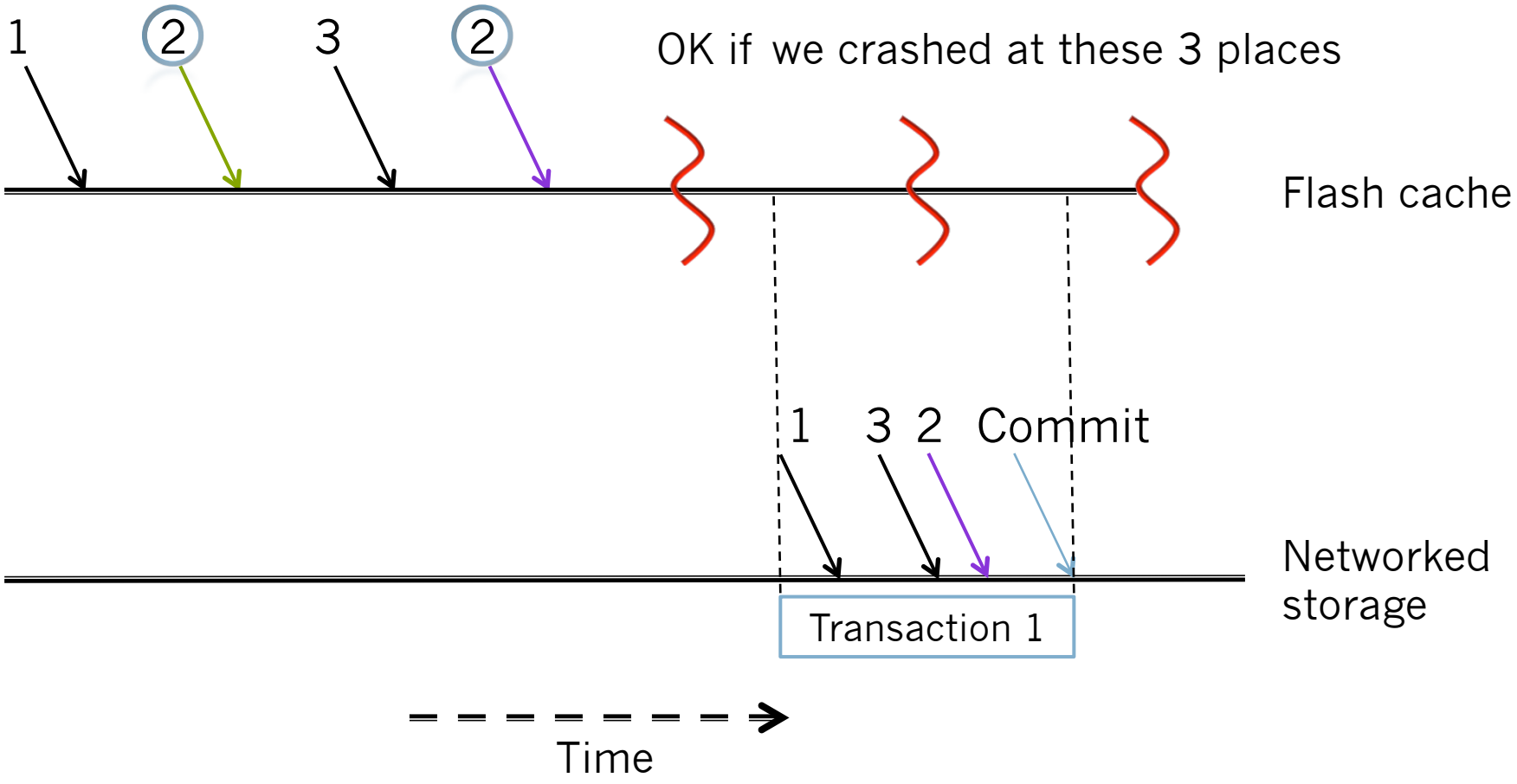
- No write coalescing
  - Block 2 has to be stored and evicted twice



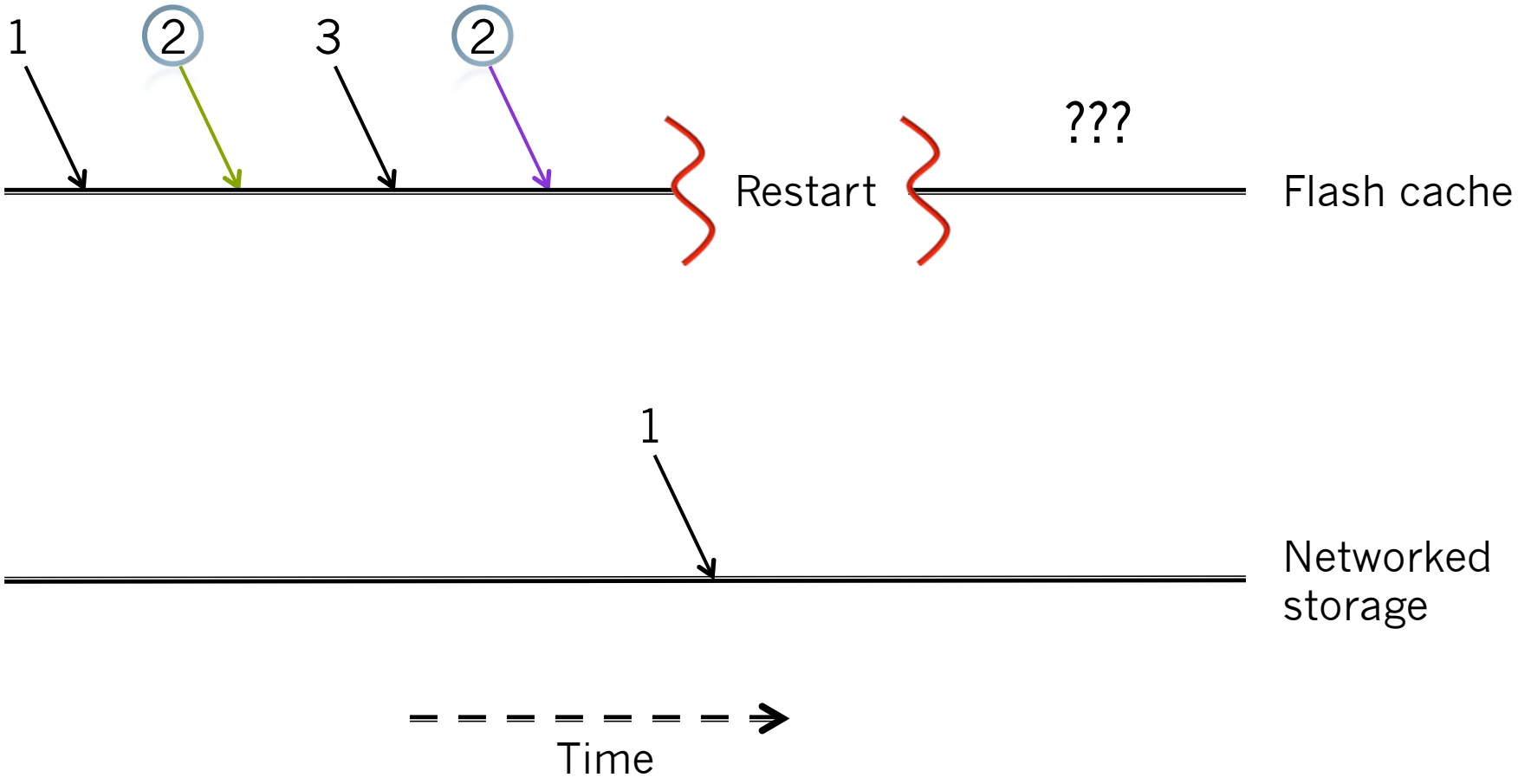
Journaled Write-back



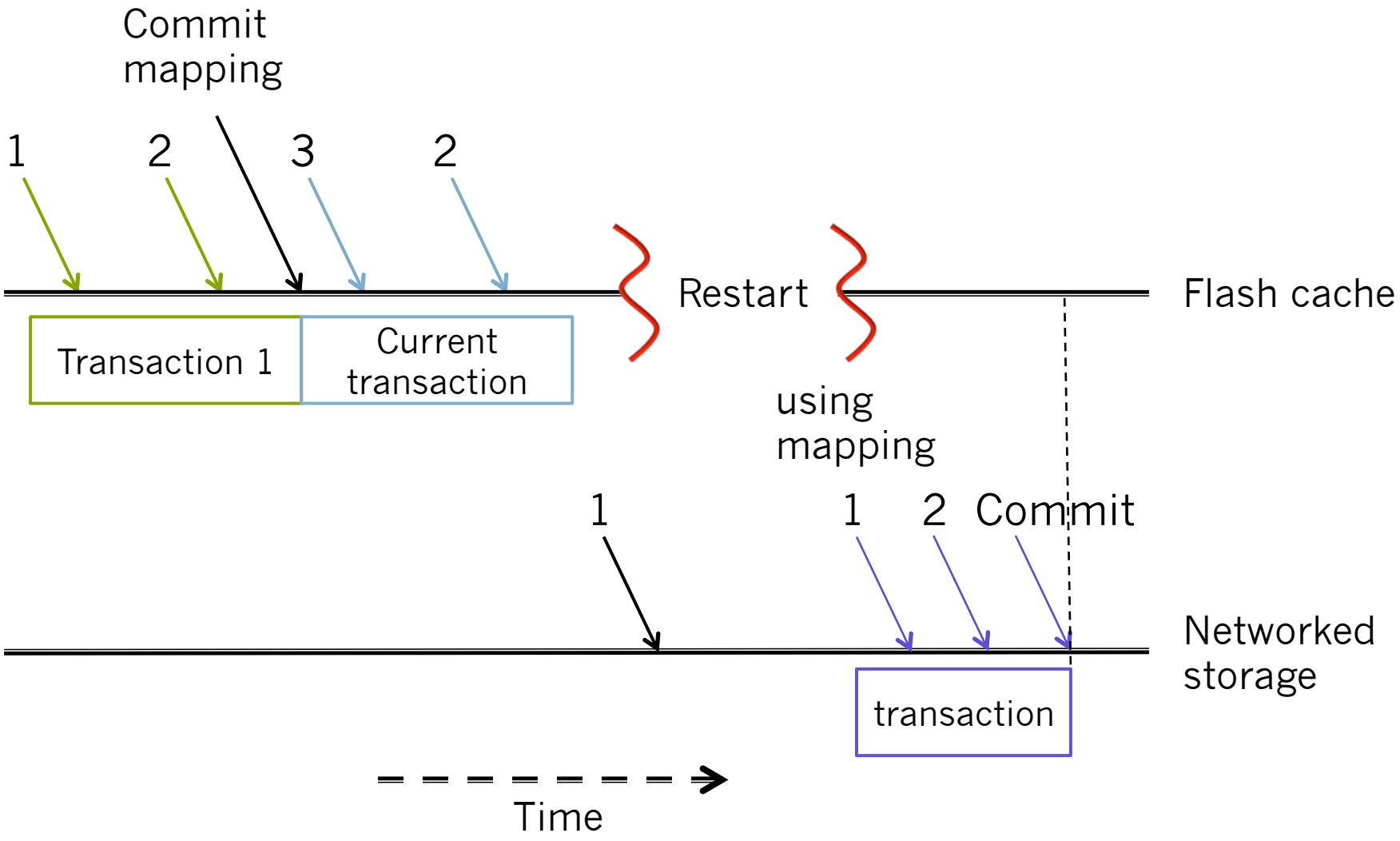
Storage-side journal



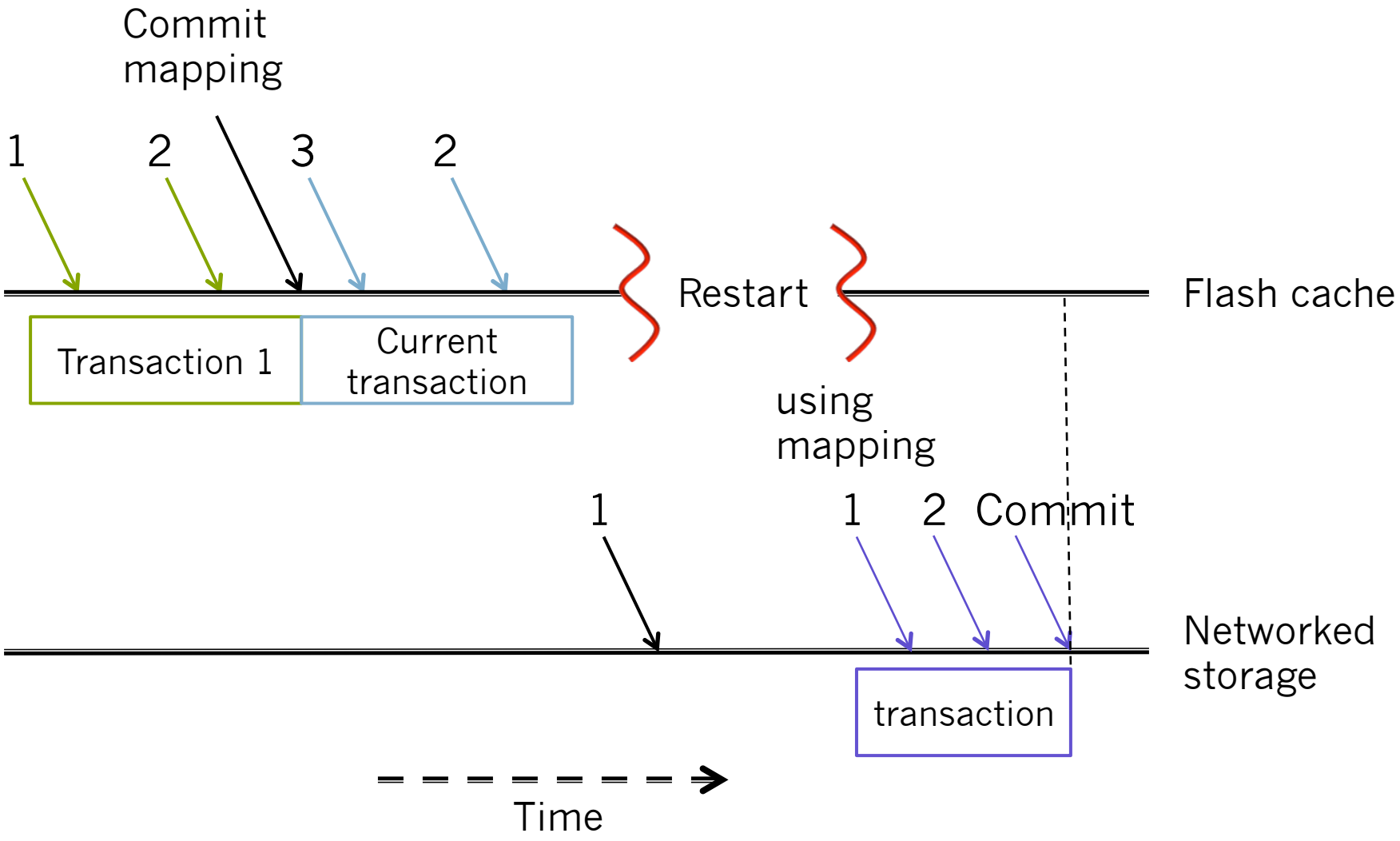
Restart the host



Host-side journal

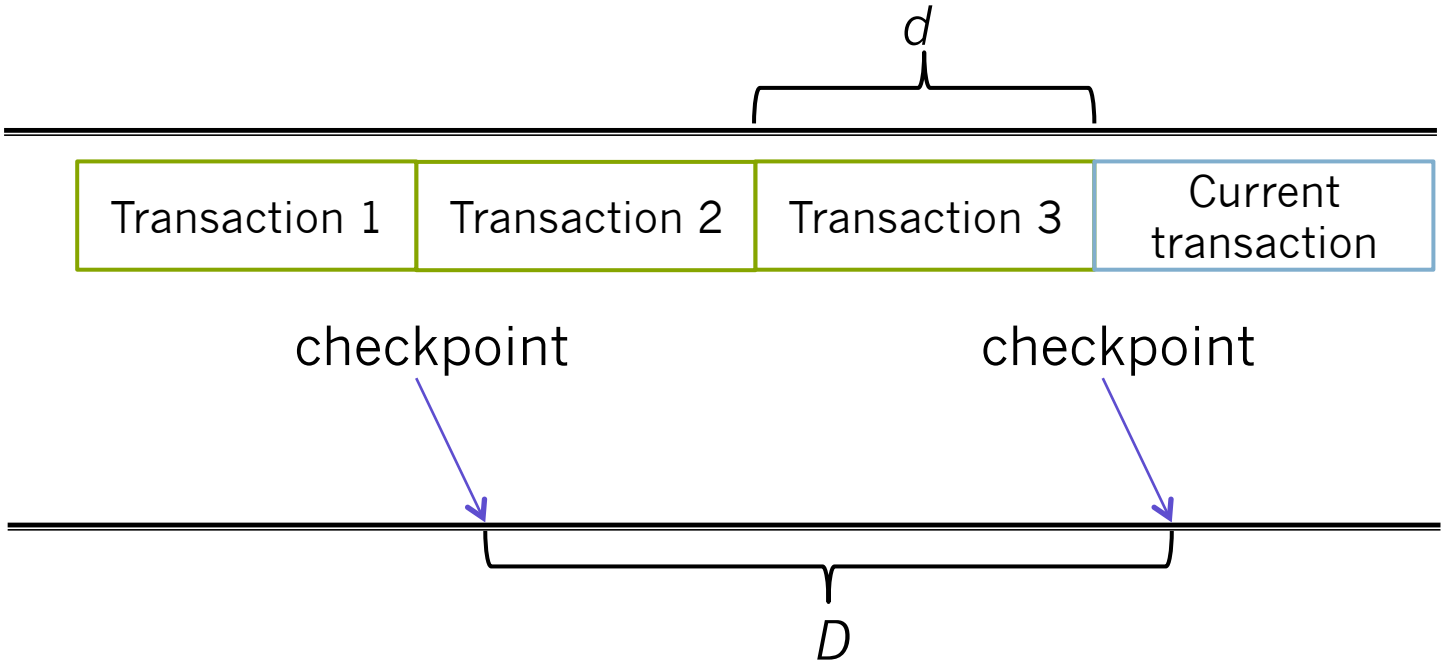


Journaled write-back



Dual staleness control

- Staleness at host crash with SSD accessible *(d)*
  - Frequency of host-side commits
- Staleness at host crash with SSD un-accessible *(D)*
  - Frequency of storage-side checkpoints

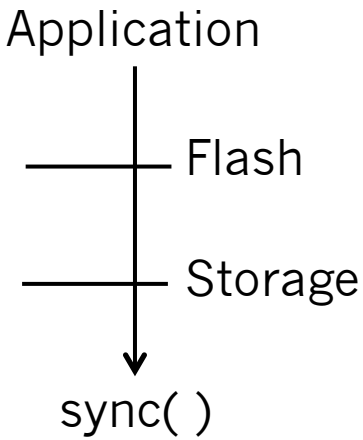


Consistency analysis

- Point-in-time consistency
  - Ordered write-back
  - Journaled write-back
- Transactional consistency (block layer)
  - Write-through
- Application consistency
  - Journaled write-back with application hints

Consistency analysis

- Point-in-time consistency
  - Ordered write-back
  - Journaled write-back
- Transactional consistency (block layer)
  - Write-through
- Application consistency
  - Journaled write-back with application hints



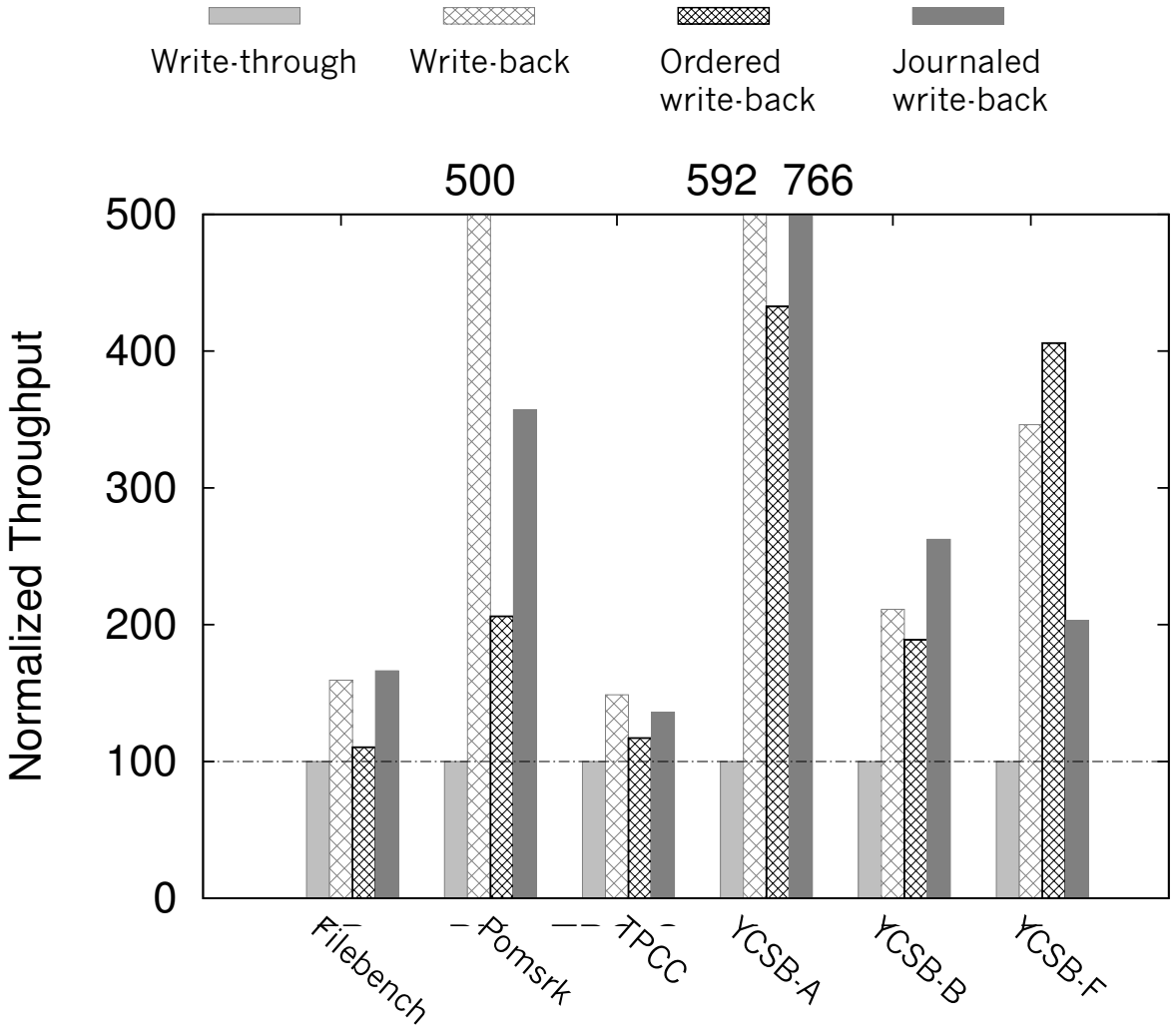
Implementation and evaluation setup

- Linux module intercepts I/Os at the block layer
  - 4 KB blocks and ARC as the replacement algorithm
- Local SSD (OCZ Revodrive PCI-e)
- Networked storage is Enterprise ISCSI Target, array of 2 disks at 7200 RPM
- Filebench, Postmark, YCSB (Yahoo Cloud Serving Benchmark), TPC-C

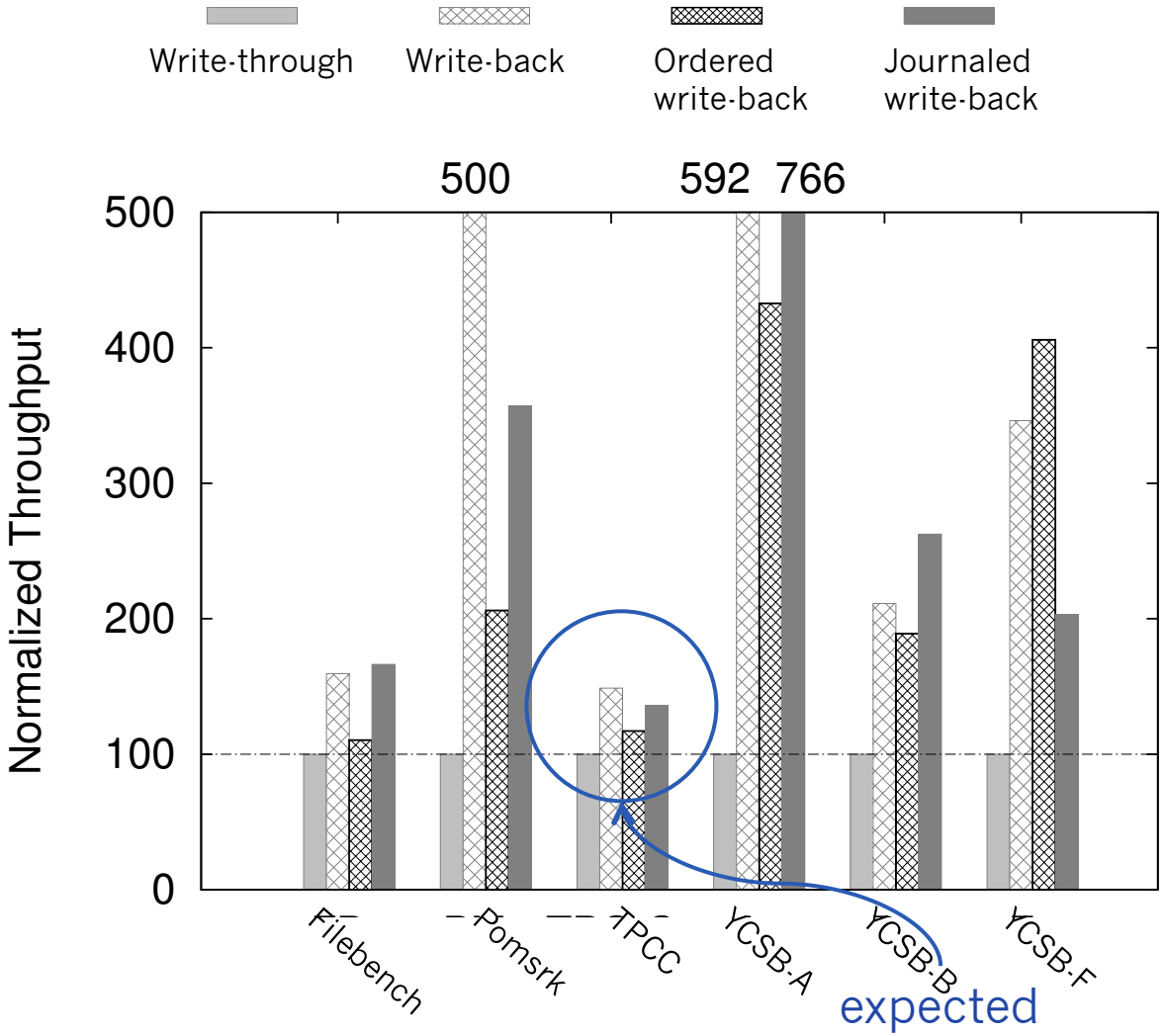
Performance comparison and analysis

Controlled staleness

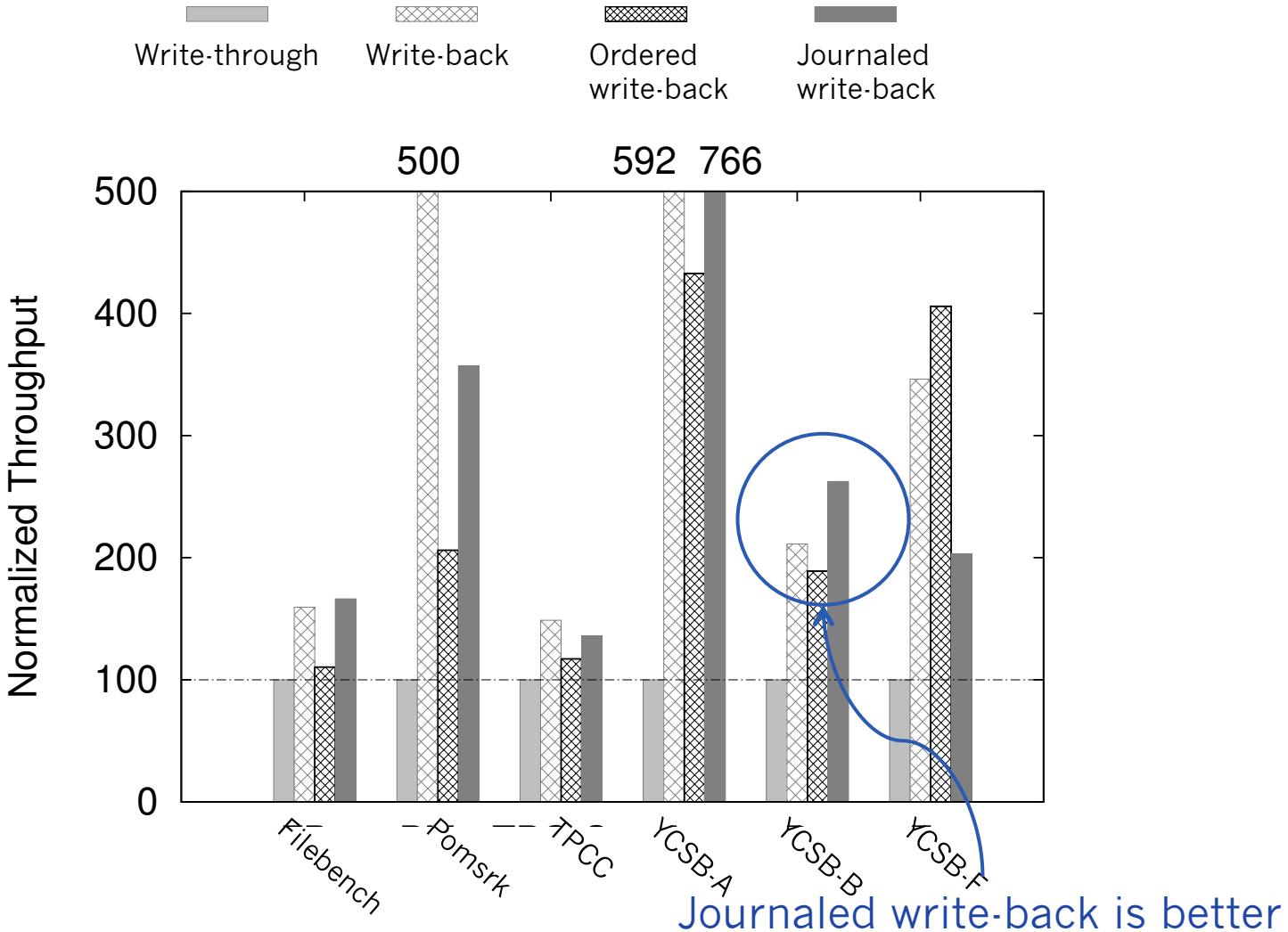
Performance comparison



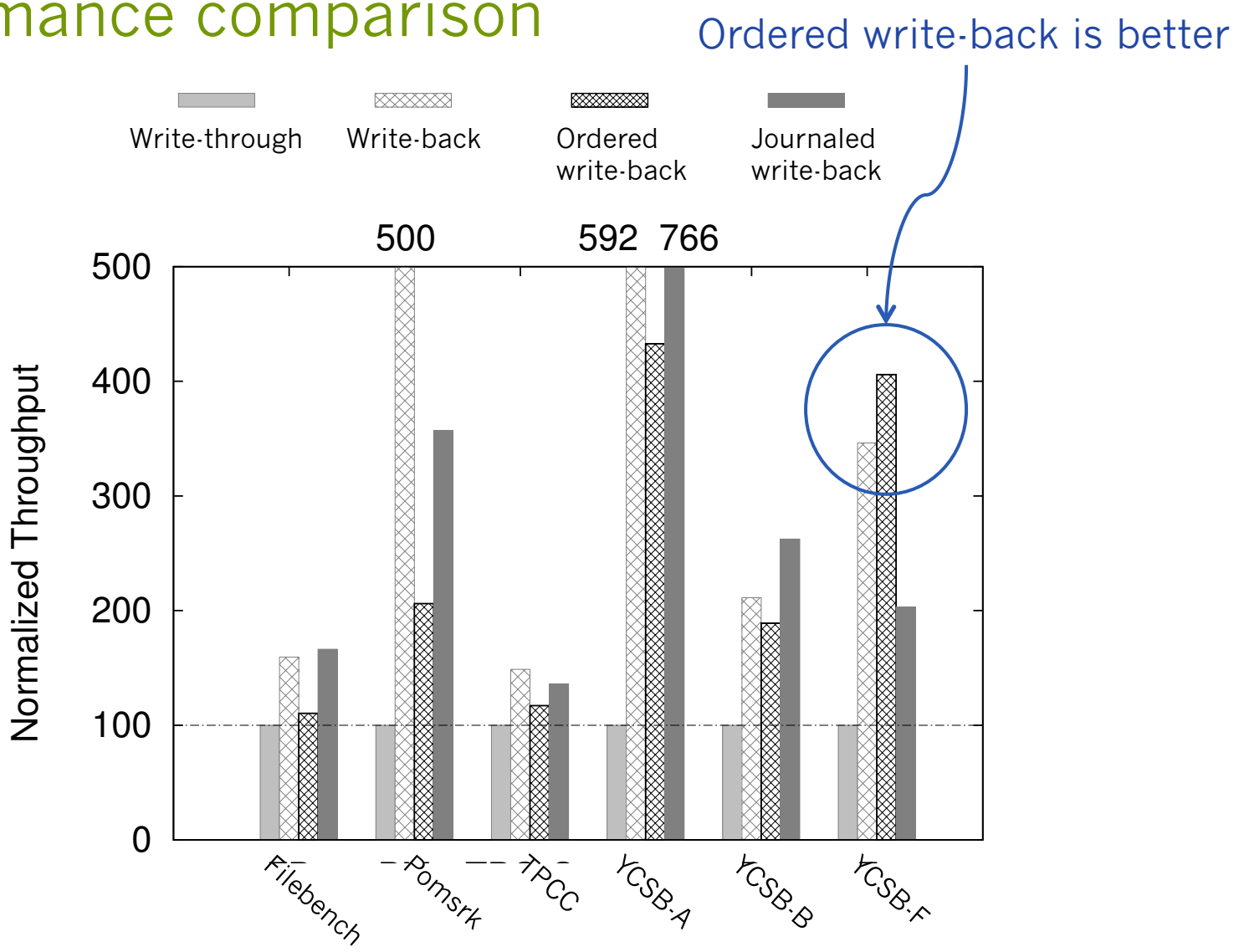
Performance comparison



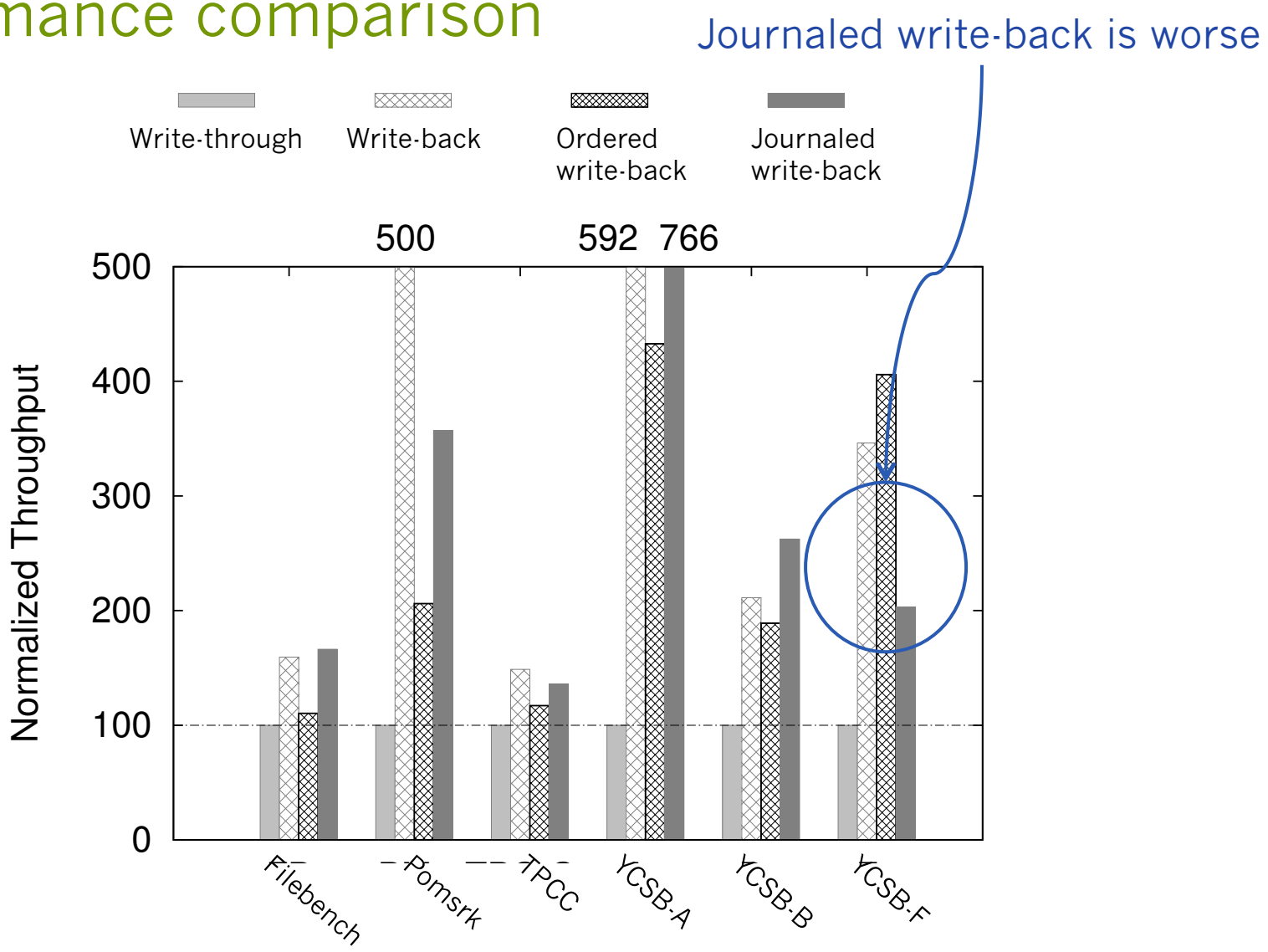
Performance comparison



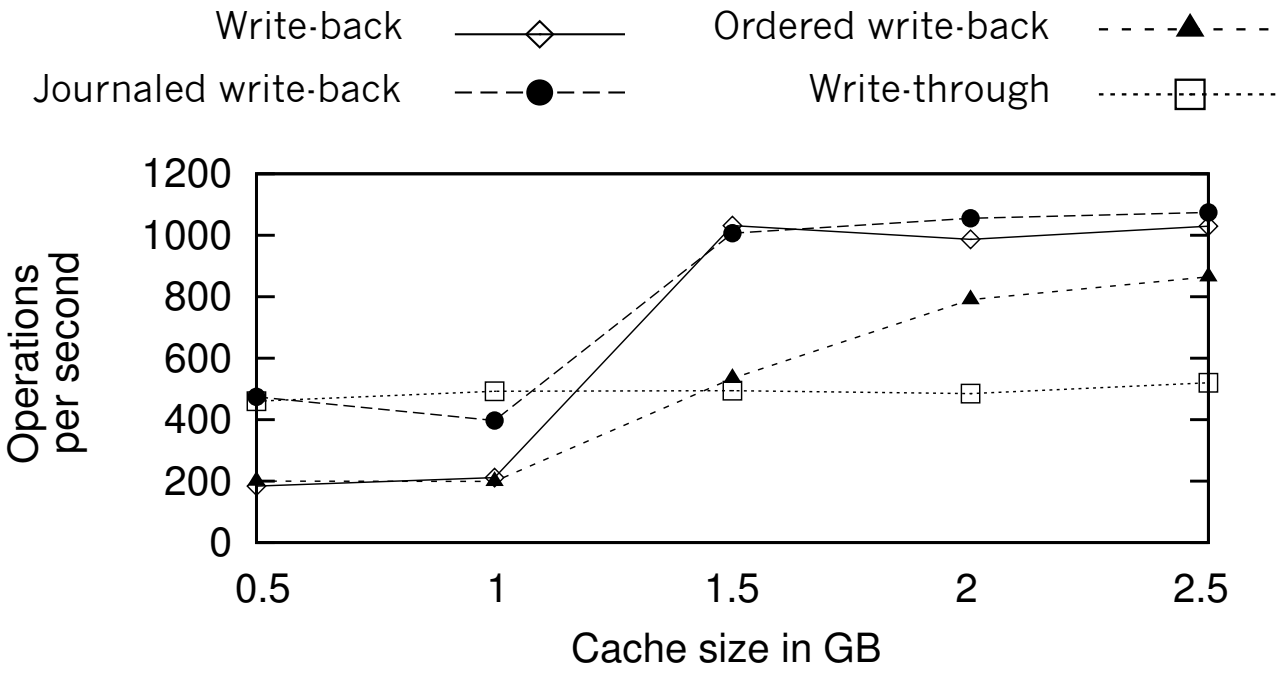
Performance comparison



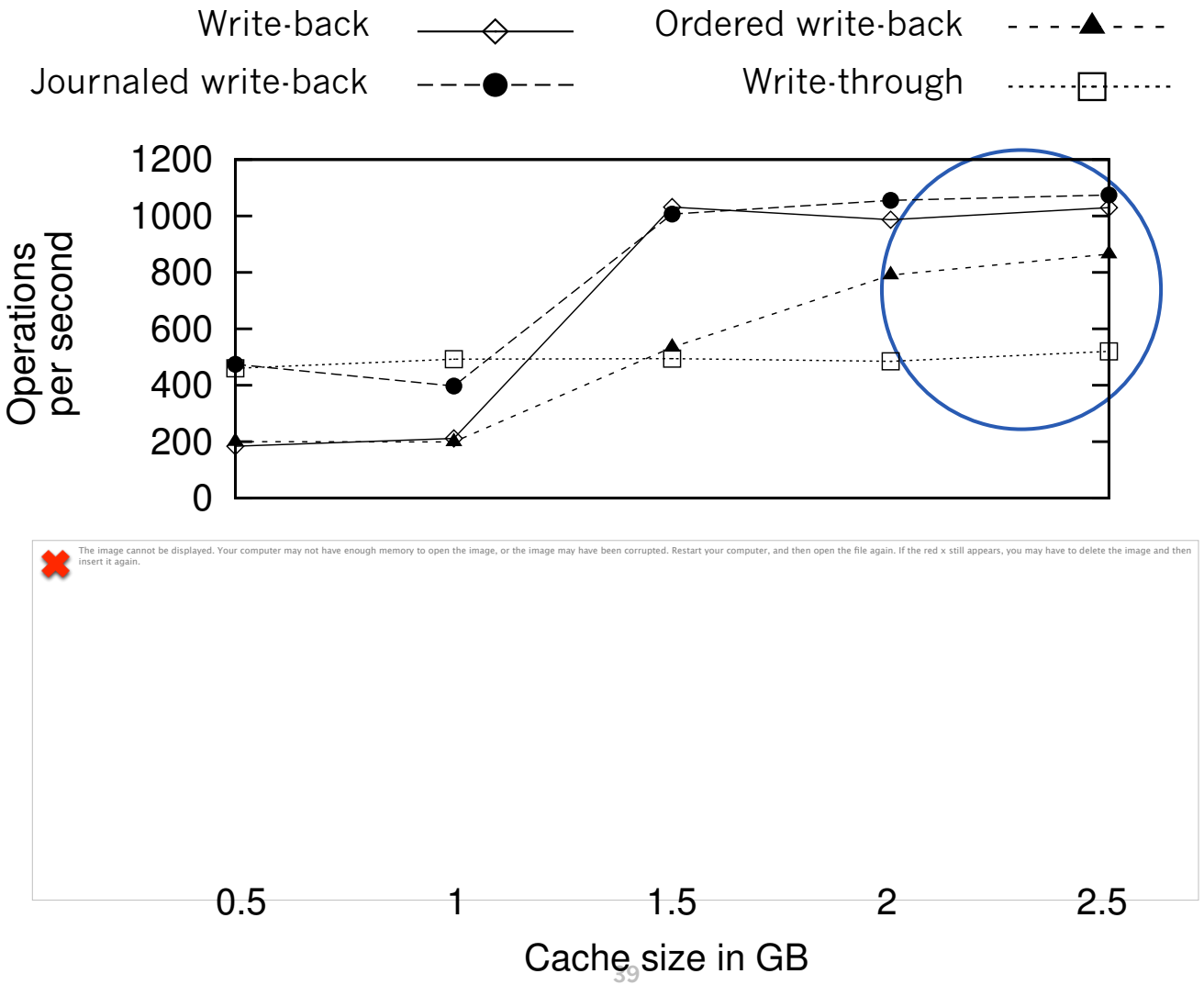
Performance comparison



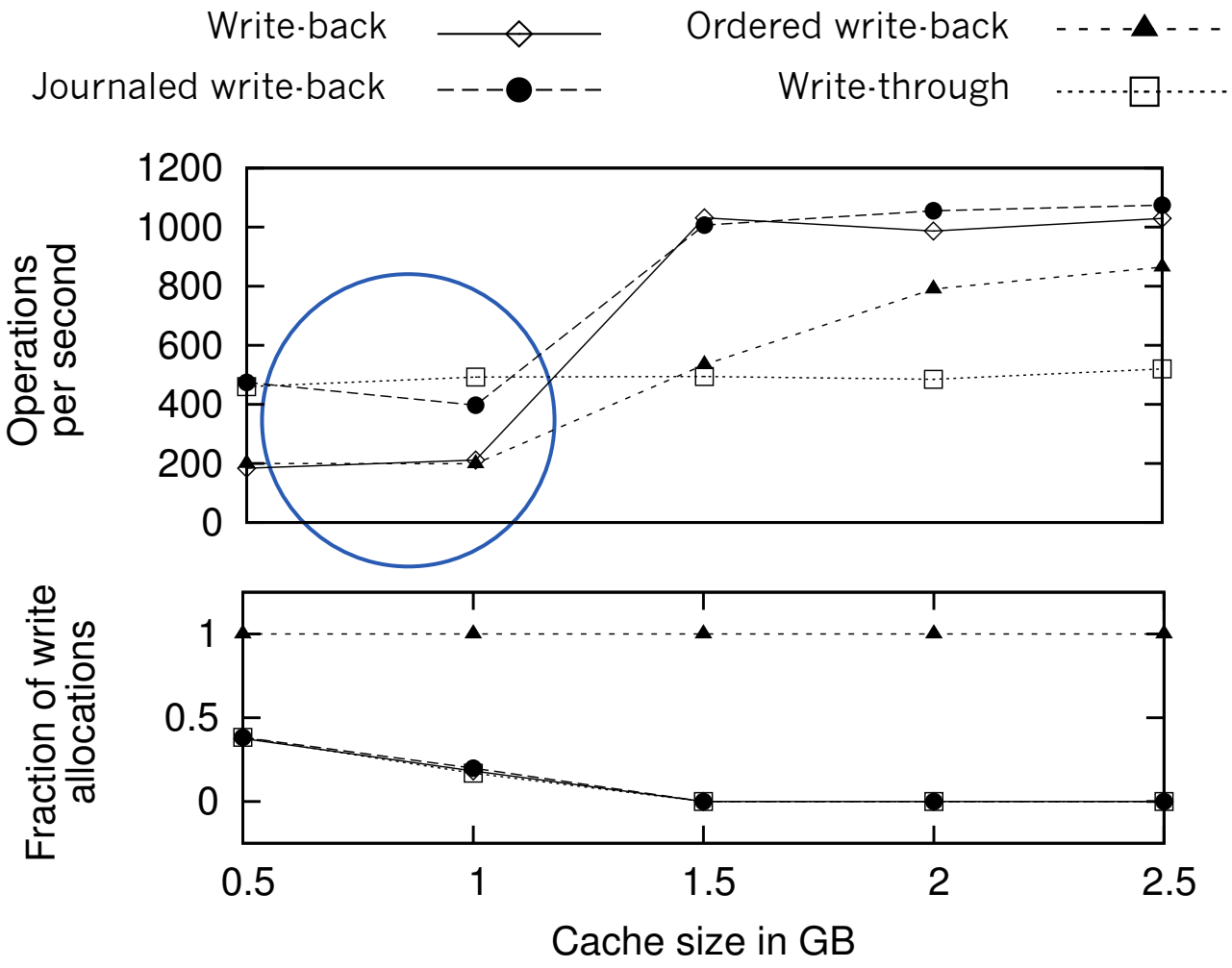
Performance comparison
Filebench fileserver



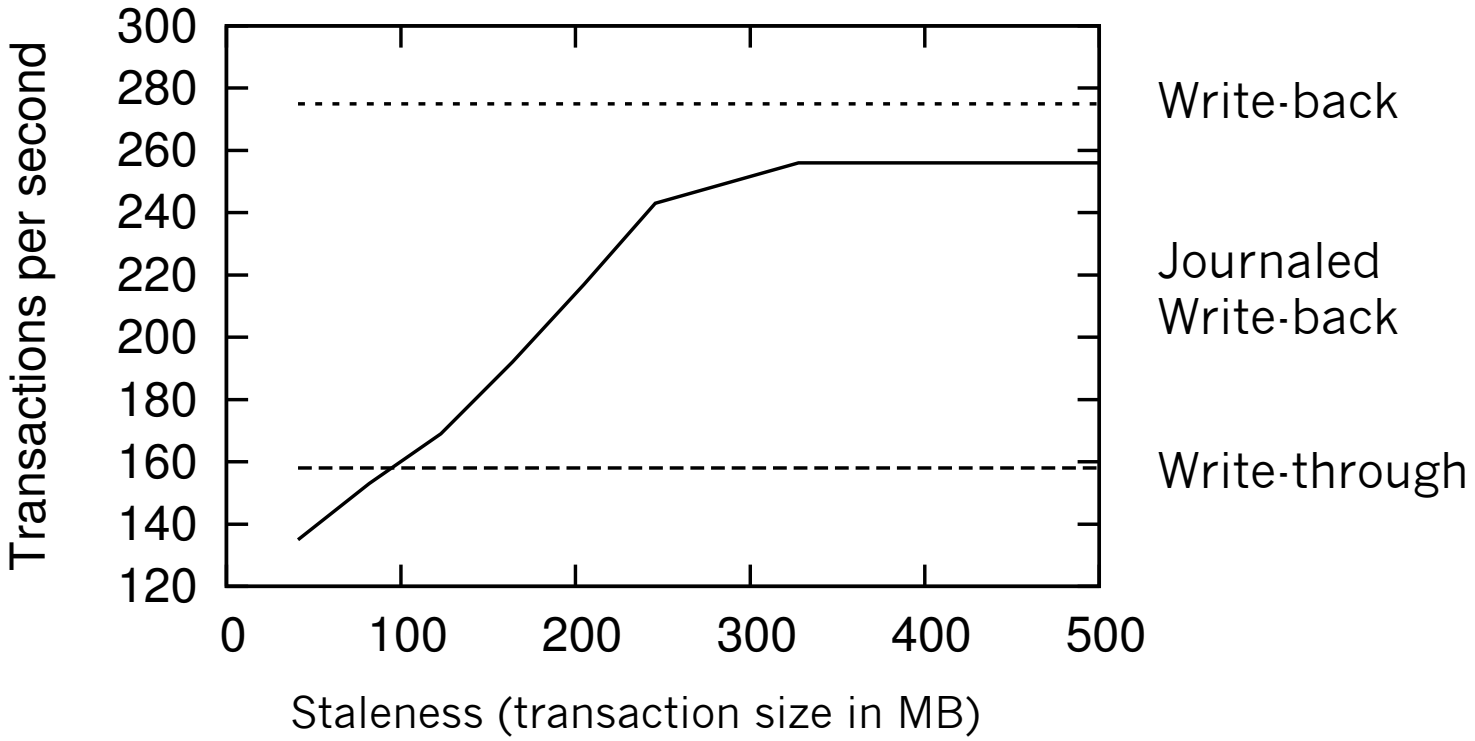
Performance comparison
Filebench fileserver



Performance improvements - Filebench



Controlled staleness
Filebench fileserver



Summary and conclusions

- Write-through and write-back are extreme opposites in terms of performance, staleness and consistency
- Ordered write-back provides point-in-time consistency at some performance cost
- Journaled write-back provides point-in-time consistency at no performance cost
- A variant of journaled write-back provides application consistency by using hints from the application

Thank you

Questions?