Rethink the Sync

Edmund B. Nightingale, Kaushik Veeraraghavan, Peter M. Chen, and Jason Flinn

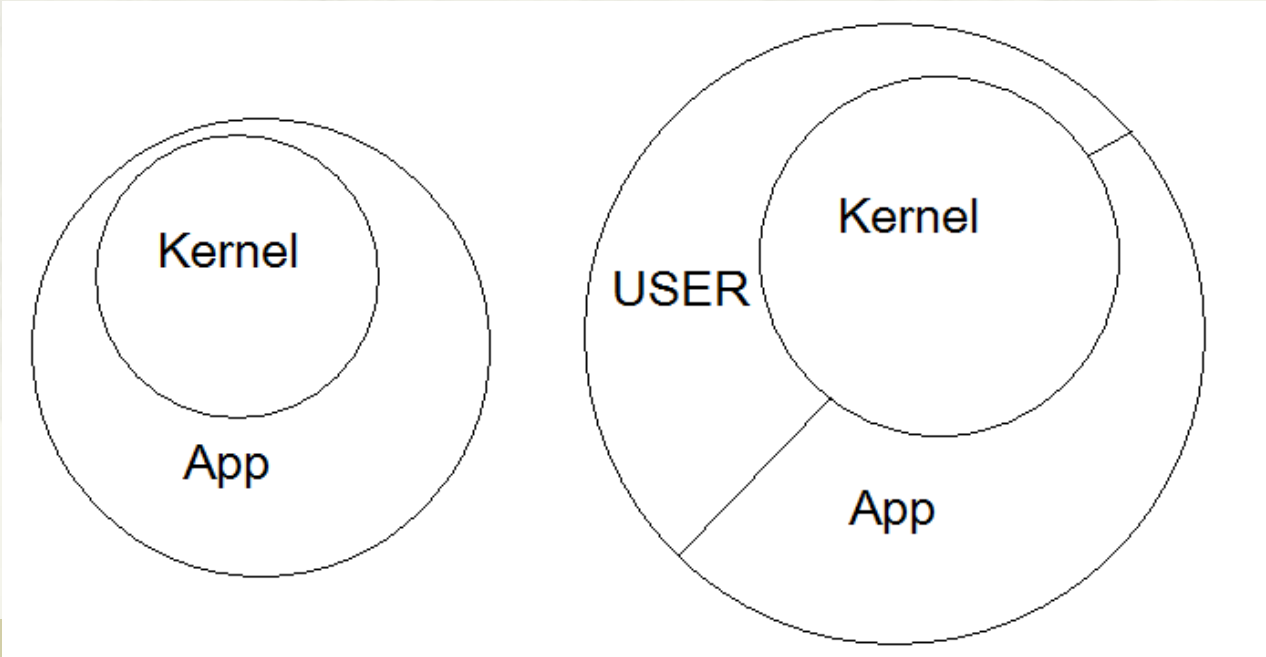
Presented by Yinzhi Cao

Introduction

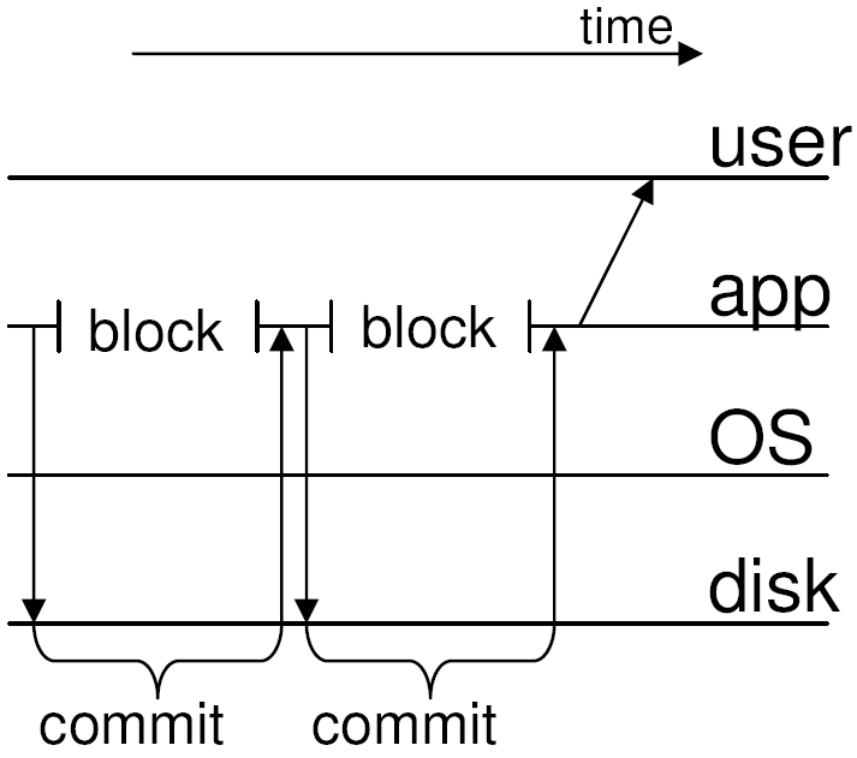
- Previous File System Approach
  - Synchronous File System
    - Durable
    - But Slow
  - Asynchronous File System
    - Fast
    - But not Safe

Introduction Continued

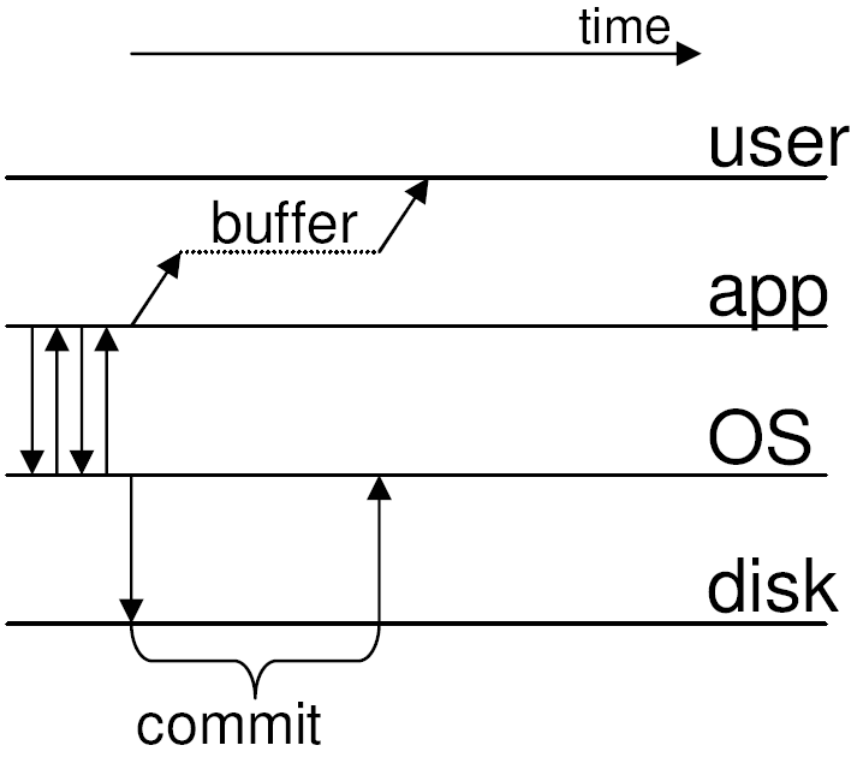
- New Approach: External Synchronous System
- Difference with Synchronous System
  - Application-centric View vs User-centric View



System Design(1)



(a) Synchronous I/O



(b) Externally synchronous I/O

System Design(2)

- Definition of Equivalence of an externally synchronous system and a synchronous one
  - The values of the external outputs are the same
  - The outputs occur in the same causal order

Happened Before Relation(From Wiki)

* The **happened-before** relation is a means of ordering events based on the causal relationship of two events in asynchronous distributed systems.

The happened-before relation is formally defined as:

- If events a and b occur on the same process, $a \rightarrow b$ if the occurrence of event a preceded the occurrence of event b
- If event a is the sending of a message and event b is the reception of the message sent in event a, $a \rightarrow b$.
- Transitivity property: for three events a, b, and c, if $a \rightarrow b$ and $b \rightarrow c$, then $a \rightarrow c$.

System Design(3)

- Improving performance
  - Modifications are group committed.
  - Buffering screen output
- System Requirement
  - Track causal relationship between file system modification and external output

System Design(4)

- Tradeoff between committing and delay committing
  - Immediately Low Throughput
  - Delaying High Latency
- Output-Triggered Commits

System Design(5)

- Limitation
  - Complicate Application-specific Recovery
  - User cannot see temporal expectations about modifications
  - Modifications to data in two different file systems may cause problems

Implementation(1)

- Speculator
  - Predict results of remote operation and let system run with predicted results
  - If different from predicted, roll back.
- From speculation to synchronization
  - No roll back and checkpoint
  - Dependency Tracking and Buffering

Implementation(2)

* Output-triggered commits
* 5-seconds commits

Implementation(3)

- Shared Memory
  - It is very complicated to deduct when writer is writing
  - This system just block all readers when one writer exists.

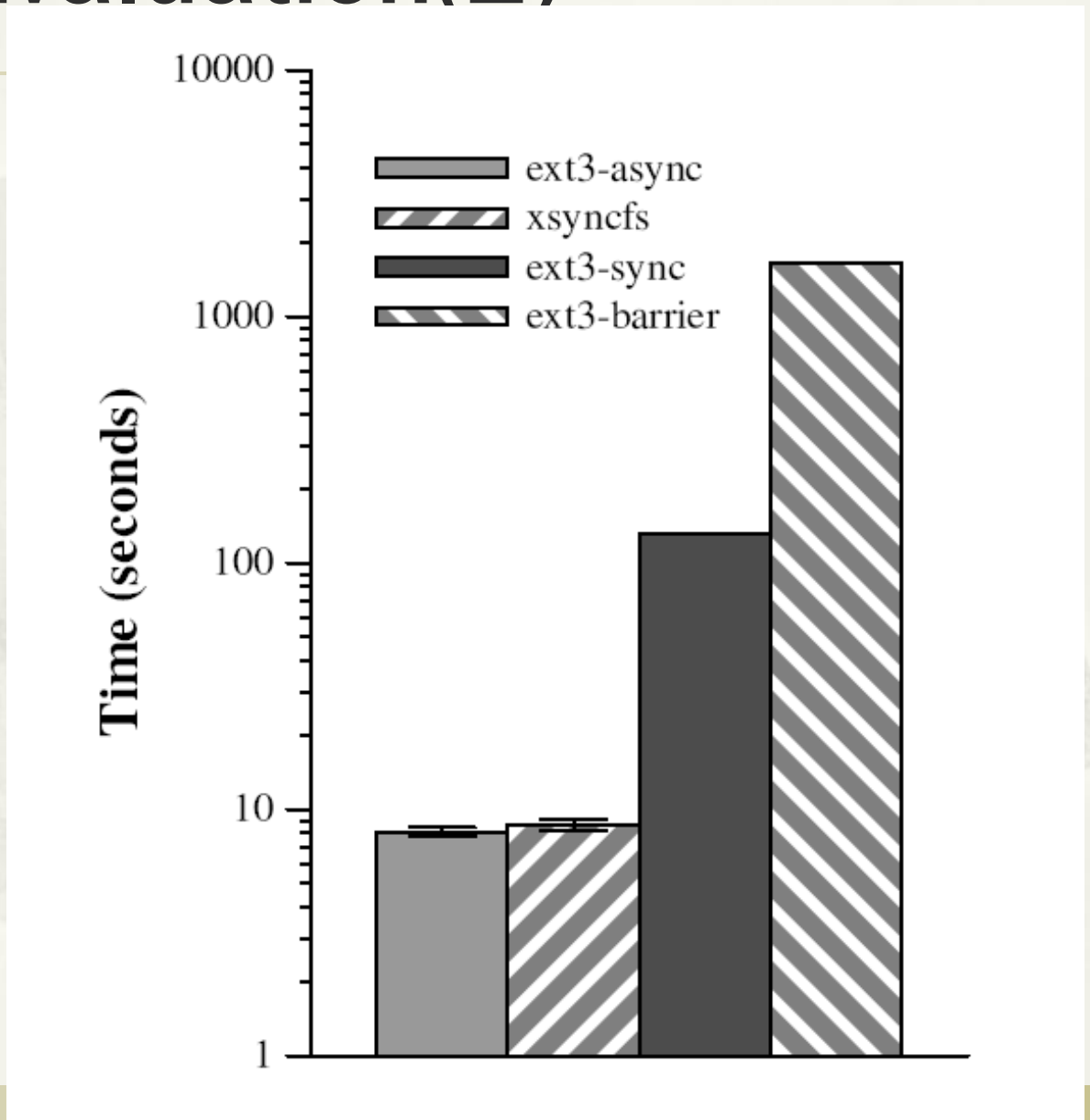
Evaluation(1)

* Durable

File system configuration	Data durable on `write`	Data durable on `fsync`
Asynchronous	No	Not on power failure
Synchronous	Not on power failure	Not on power failure
Synchronous with write barriers	Yes	Yes
External synchrony	Yes	Yes

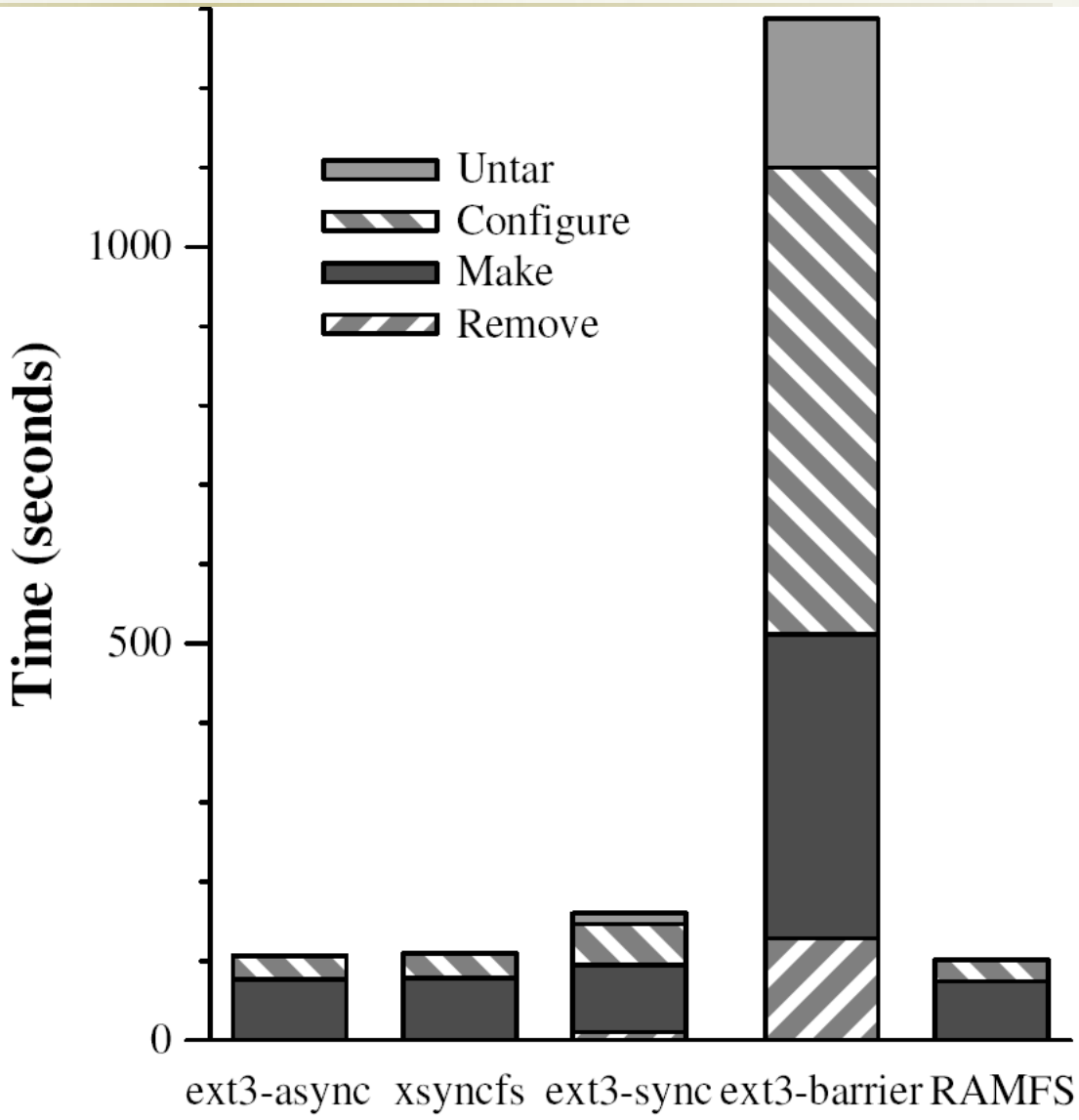
Evaluation(2)

* The PostMark benchmark



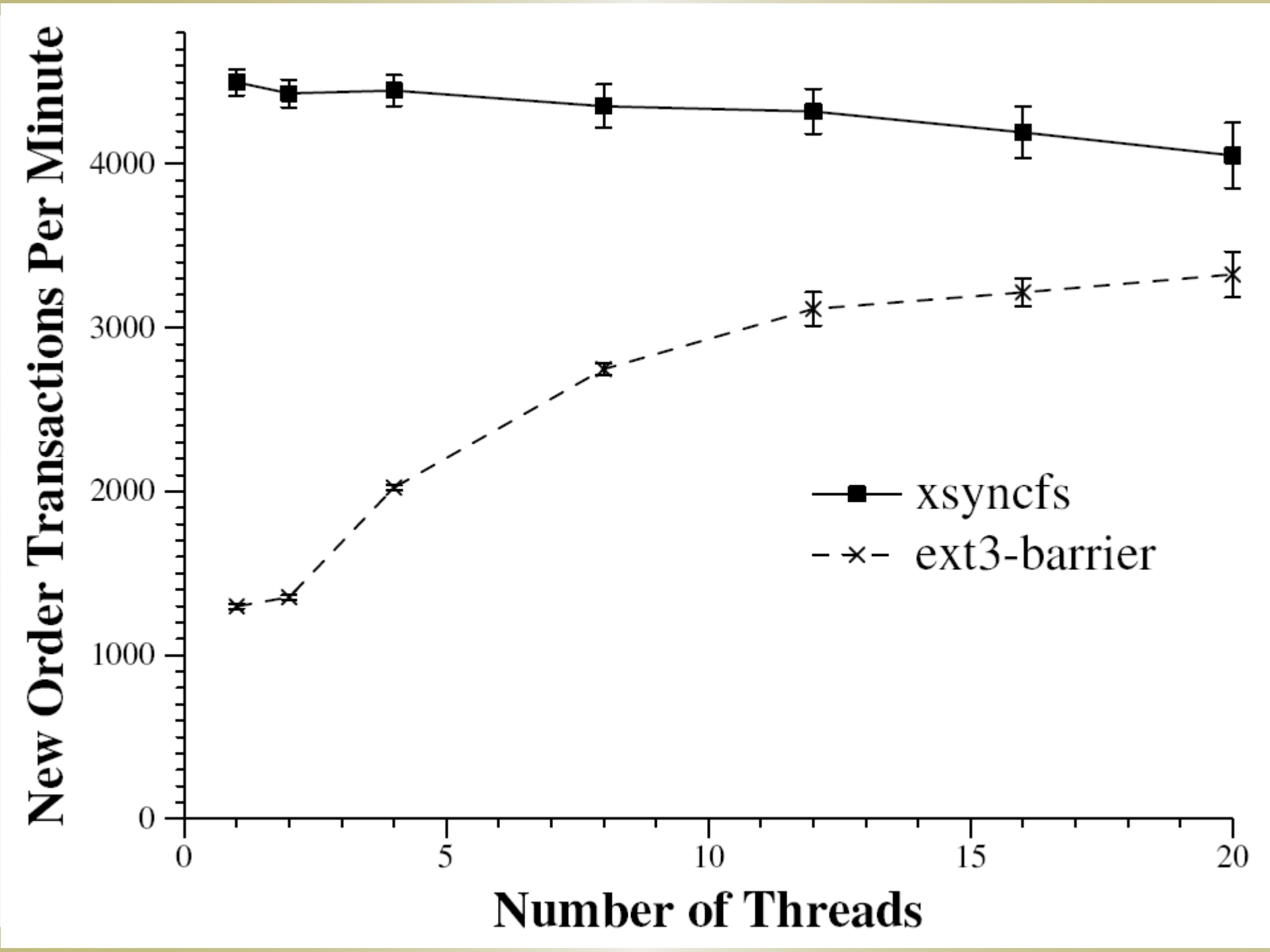
Evaluation(3)

- The Apache build benchmark



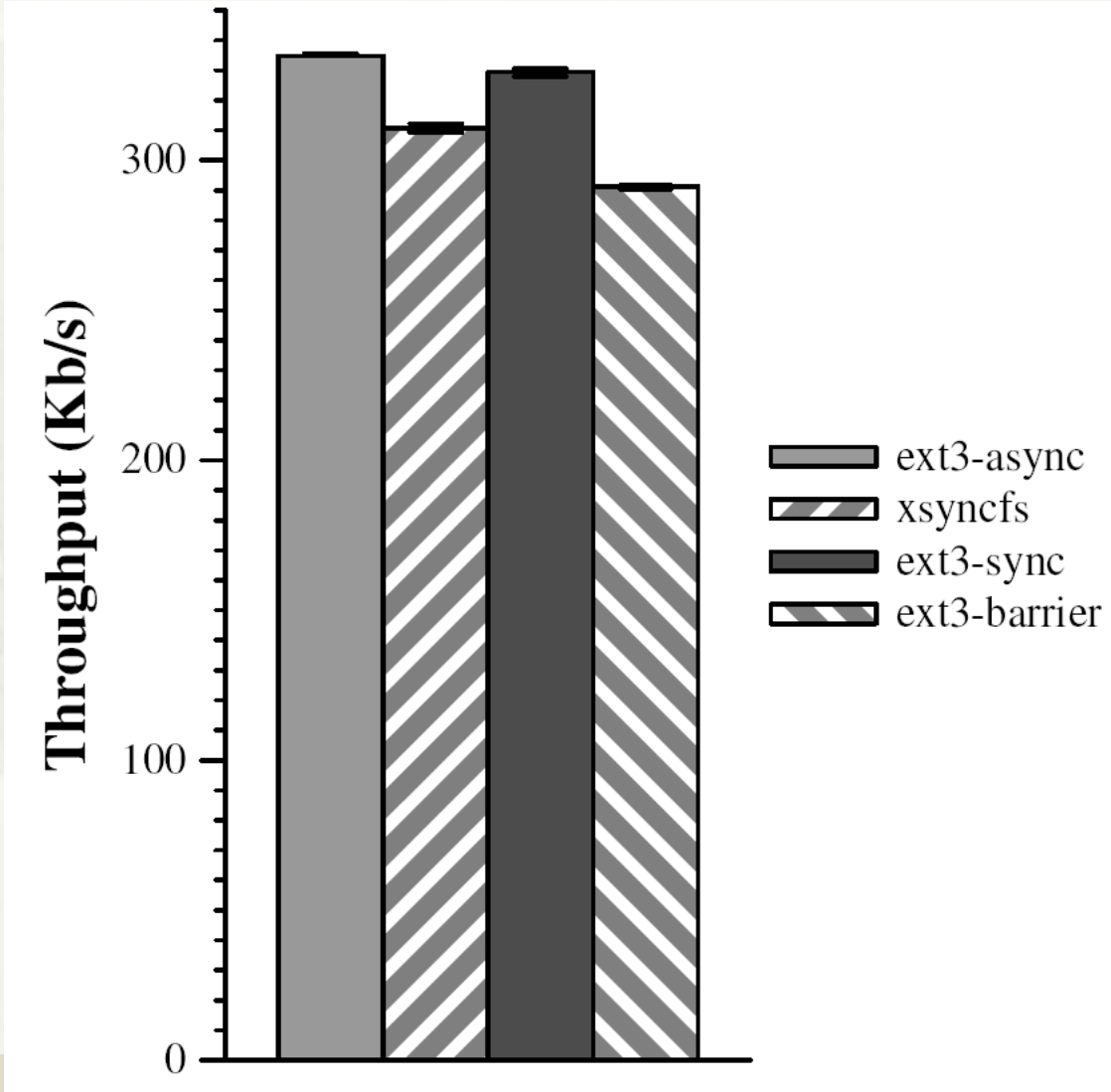
Evaluation(4)

* The MySQL benchmark


New Order Transactions Per Minute
0
1000
2000
3000
4000
0
5
10
15
20
Number of Threads
xsyncfs
ext3-barrier

Evaluation(5)

* The SPECweb99 benchmark



Evaluation(6)

* Benefit of output-triggered commits

Benchmark	Eager Commits	Output-Triggered Commits	Speedup
PostMark (seconds)	9.879 (±0.056)	8.668 (±0.478)	14%
Apache (seconds)	111.41 (±0.32)	109.42 (±0.71)	2%
MySQL 1 client (NOTPM)	3323 (±60)	4498 (±73)	35%
MySQL 20 clients (NOTPM)	3646 (±217)	4052 (±200)	11%
SPECweb99 (Kb/s)	312 (±1)	311(±2)	0%

Summary

- Strongest Point
  - Change of Point of View: From Application-Centric to User-Centric
- Weakest Point
  - They use other's code which may bring into redundancy.