Okeanos: Wasteless Journaling for Fast and Reliable Multistream Storage

Andromachi Hatzieleftheriou, Stergios V. Anastasiadis
Department of Computer Science
University of Ioannina, Greece

Outline

Motivation

Design

Implementation

Evaluation

Conclusions

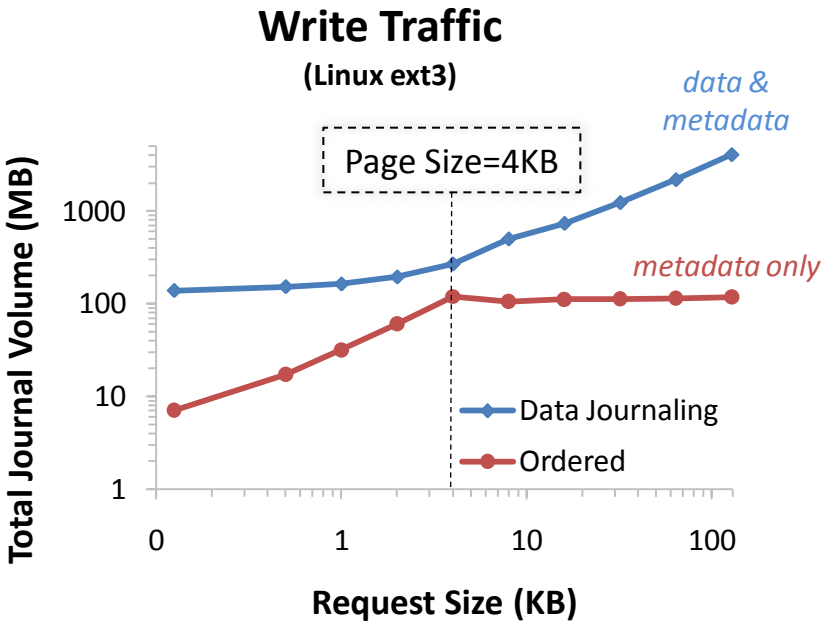
Motivation

Synchronous small writes

- critical for system and application reliability

Multistream concurrency

- effectively random I/O



In page-sized disk accesses

- async writes have good performance due to batching in memory
- sync writes result in wasteful traffic due to excessive full-page I/Os

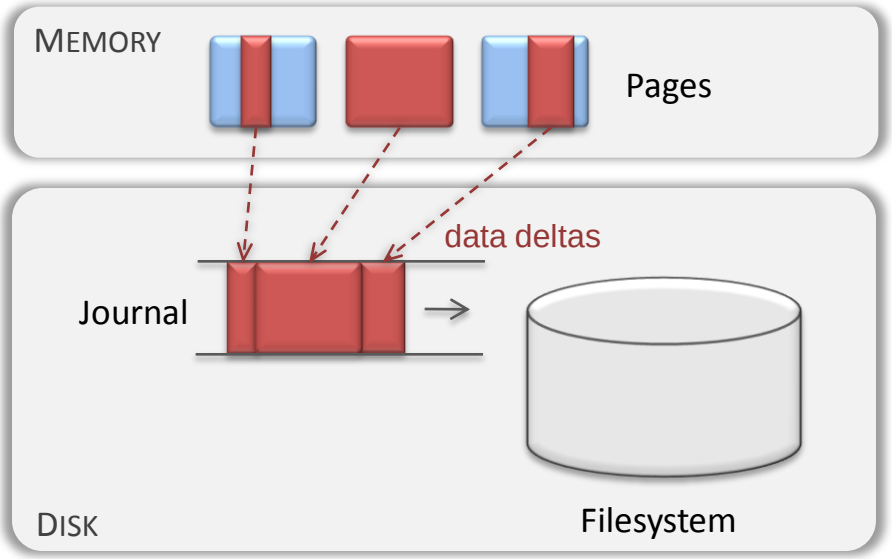
Design Goals

1. Reliable storage
   - keep data on disk
2. Inexpensive synchronous small writes
   - sequential disk throughput
3. Reduce disk bandwidth waste due to:
   - writes with high positioning overhead
   - unnecessary writes of unmodified data

Proposed approach:

- batch random small writes in memory
- journal data updates at subpage granularity

Wasteless Journaling



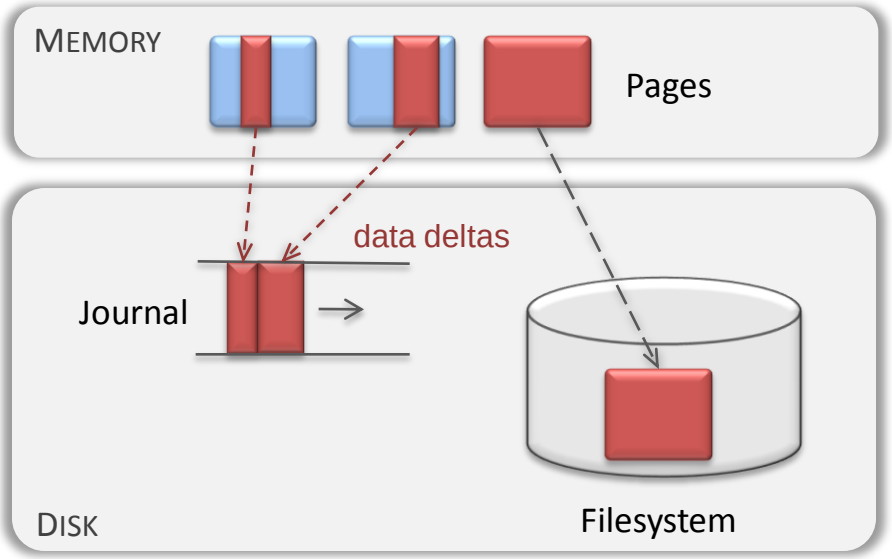
Idea:

1. Synchronously transfer data deltas from memory to journal
2. Occasionally move data blocks from memory to final location

Still wasteful!

- large writes → disk traffic duplication

Selective Journaling



Definition:

- write threshold differentiates requests by size

Idea:

1. Transfer large requests to final location without journaling of data
2. Treat small requests according to wasteless journaling

Consistency

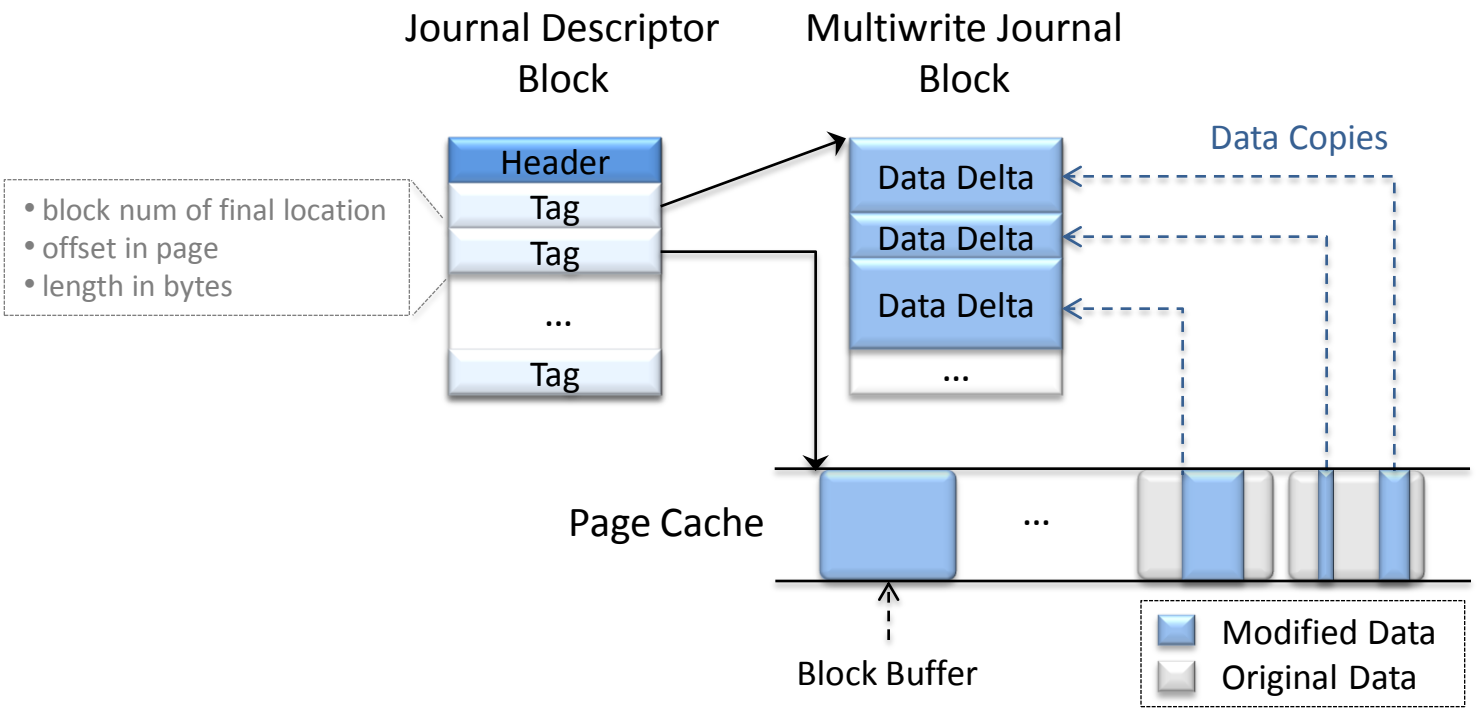
Wasteless Journaling:

- atomic updates of both data and metadata

Selective Journaling:

- data updates either journaled or not depending on request size
- consistency at least as strict as default ext3 journaling mode (ordered)

Prototype Implementation



Multiwrite journal block

- accumulates multiple subpage data updates

During recovery

- apply data deltas to corresponding final disk blocks

Experiments

Implemented in Linux kernel 2.6.18 ext3

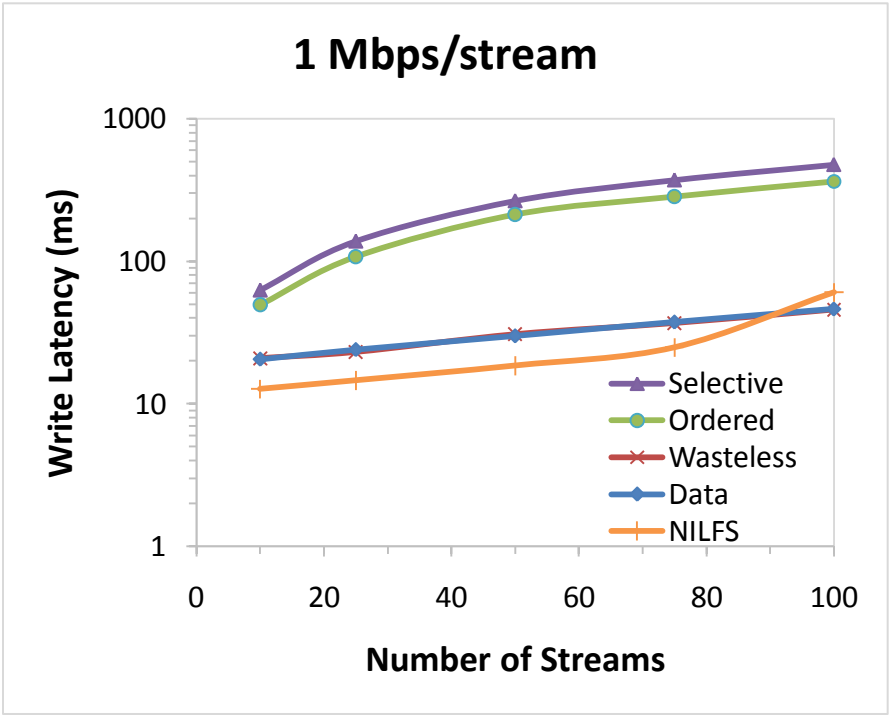
Experimentation Environment:

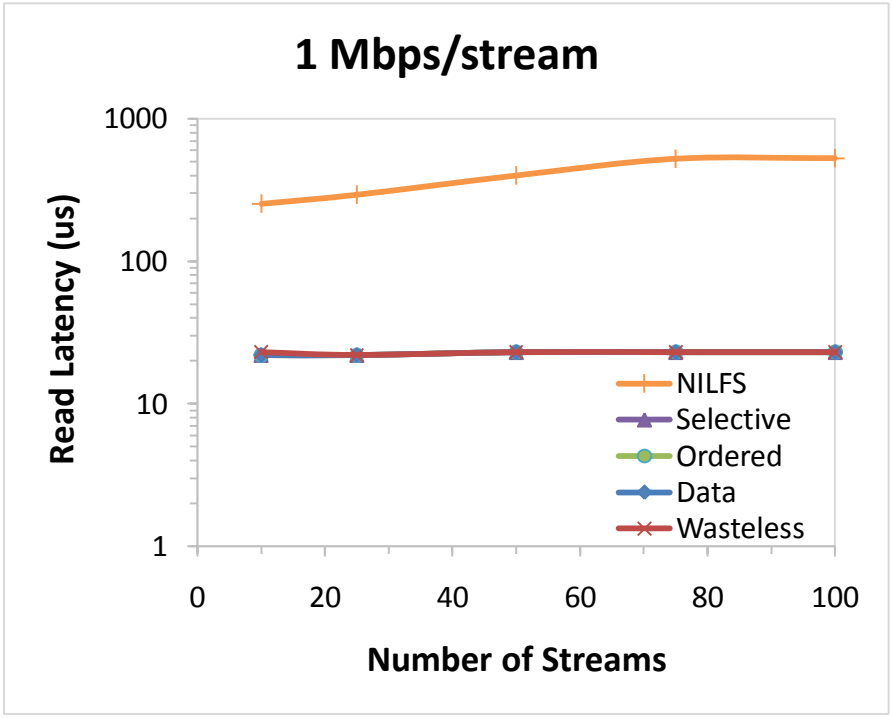
- x86-based servers
- quad-core 2.66GHz processor
- 3GB RAM
- Seagate Cheetah SAS 300GB 15KRPM disks

Workloads:

- Microbenchmarks
- Postmark
- MPIO-IO over PVFS2

Latency

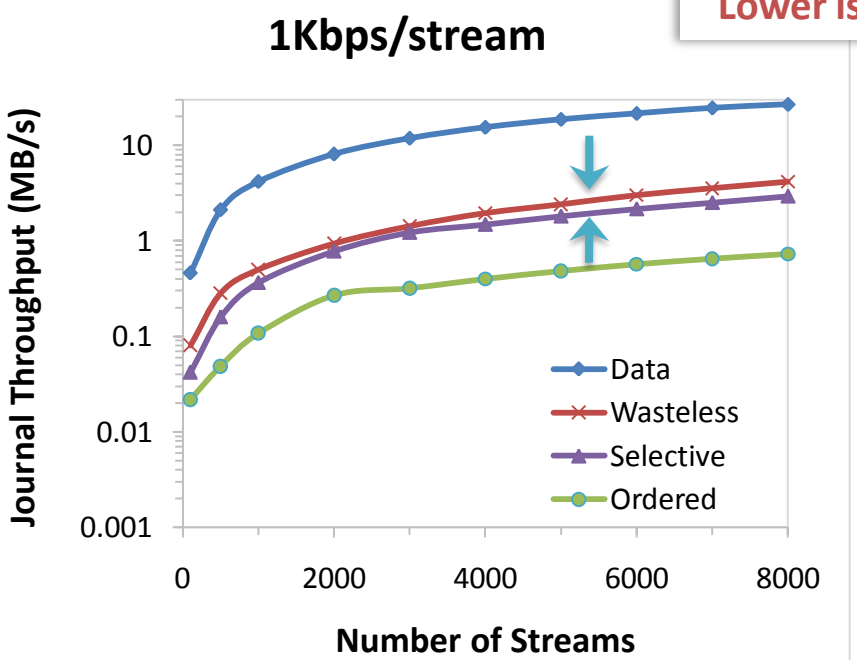


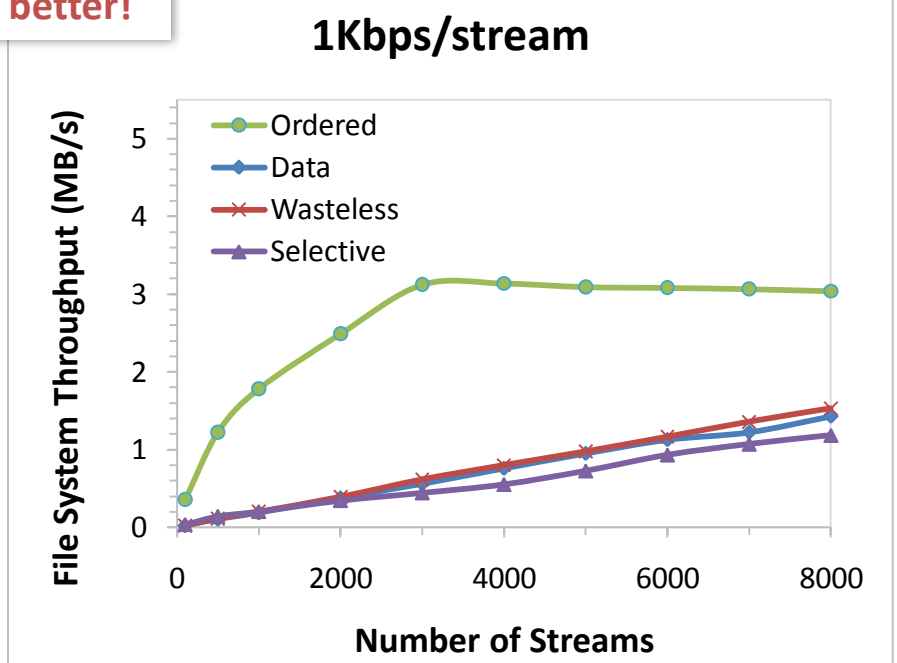


- Data & wasteless achieve substantially lower write latency
  - similar to NILFS (stable Linux port of LFS)
- NILFS read latency significantly higher due to poor storage locality!

Disk Traffic

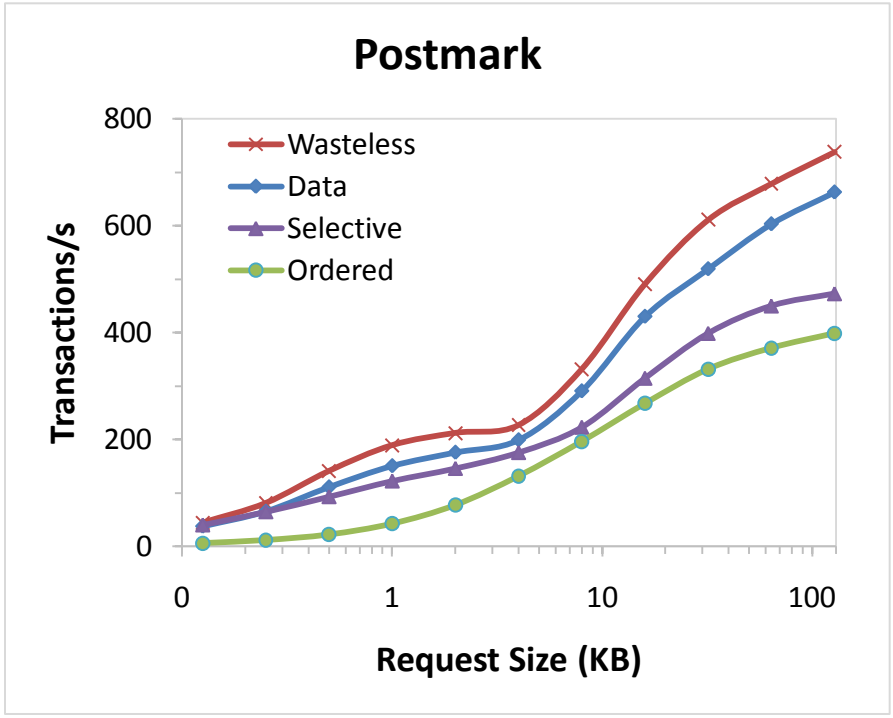
Lower is better!

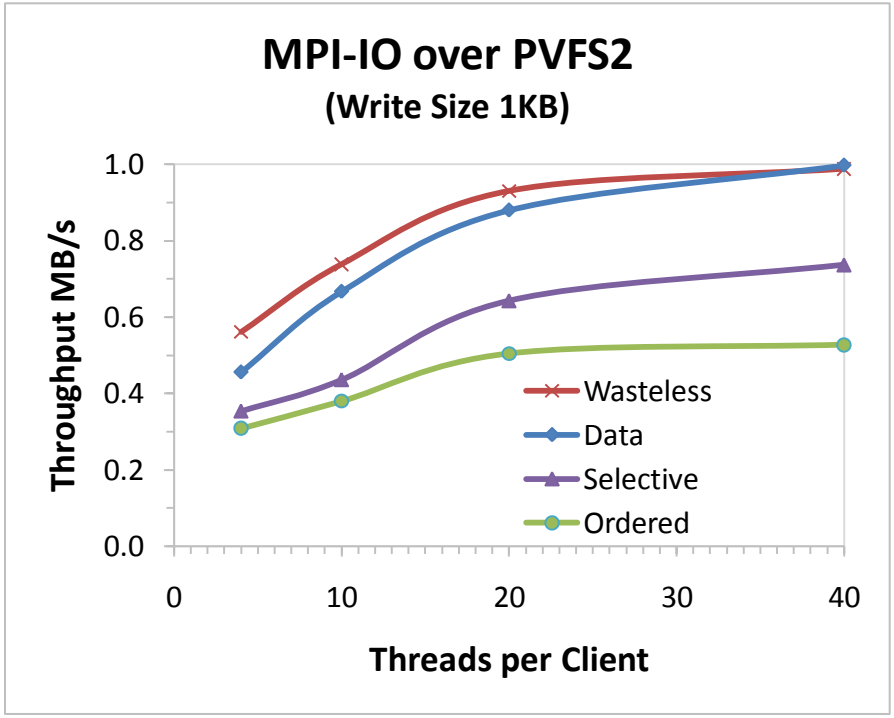




- Data journaling expensive in terms of journal traffic
- Ordered journaling incurs increased filesystem traffic
- Wasteless & selective substantially reduce journal and filesystem traffic

Application-Level Workloads





- Small files workload
  - wasteless increases transaction throughput
- Parallel I/O workload
  - 13 clients, 1 PVFS2 data server, 1 PVFS2 metadata server (15 machines)
  - wasteless doubles the throughput of parallel application checkpointing

Conclusions & Future Work

Key concept:

- apply subpage journaling of data updates to ensure reliability

Wasteless Journaling

- merges subpage writes into page-sized journal blocks

Selective Journaling

- journals only updates below a write threshold

Performance benefits demonstrated over ext3:

- reduced write latency
- improved transaction throughput
- avoided bandwidth waste

Future Work

- extent for virtualization environments and flash memory systems