Controlling File System Write Ordering

Nathan C. Burnett
Andrea C. Arpaci-Dusseau
Remzi H. Arpaci-Dusseau
University of Wisconsin - Madison

Why control write ordering?

- WAL requires control over write ordering
- How is it done now?
  - Application managed storage (raw device)
    - Makes management difficult
  - Direct I/O
    - Slow, not portable
  - fsync(), synchronous I/O
    - Slow
  - write() and hope
    - Consistency guarantees are lost

Approach

- Create interface to express ordering to OS
- What is the right interface?
  - Simple
  - Portable
  - Asynchronous
  - Fast

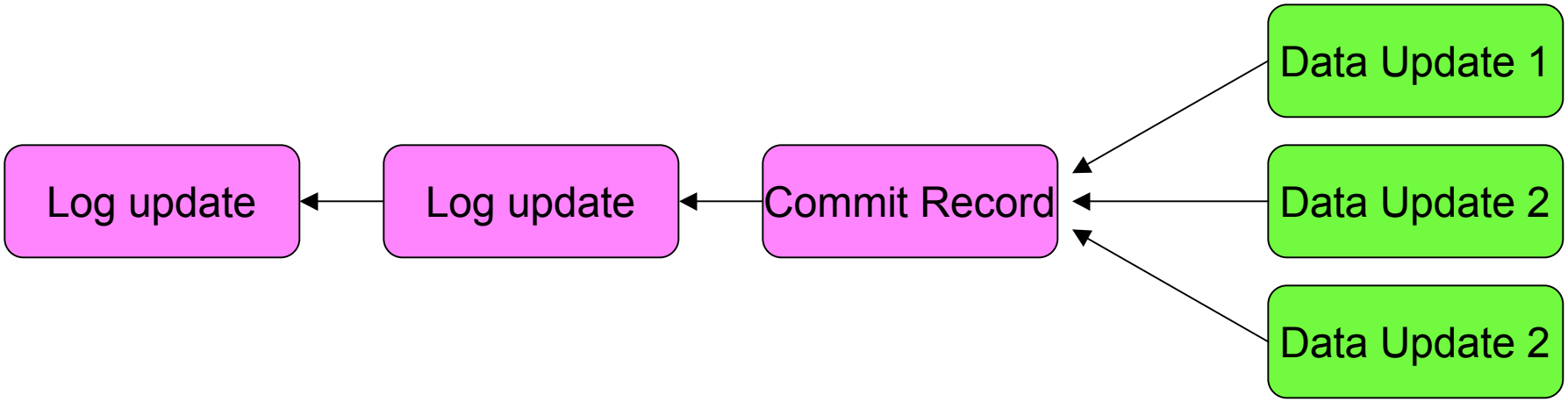
File System Barriers

- Added barrier() system call
- Writes do not get reordered across barrier
  - e.g. write(log, …), barrier(), write(data, …)
- Advantages
  - Easy to understand
  - Replaces fsync() and sync() for ordering
  - Asynchronous and fast
- But it still restrains OS I/O scheduling

Asynchronous Graphs

- Specify exactly when order matters
  - For two write ops, say which one goes first
  - Specify no ordering if it doesn't matter
- Generates graph of order dependencies
- Data will be written in order when needed
- OS is free to reorder other requests

A Quick Example



- Chain log updates so commit is last log update
- Ordering between data updates unspecified
- All data written *after* the log commit record

Current Status

- Barriers implemented in FreeBSD 5.4
- Exploring benefits in simple simulation
  - Simulates buffer cache and disk
  - Disk writes are either seq (fast) or not (slow)
- We can show for a transactional load:
  - agraphs requires fewer I/Os
  - agraphs requires fewer non-sequential I/Os

Performance Benefits

- Fewer writes overall
  - log writes are generally very small
  - fsync and barriers separate small writes
  - asynchronous graphs combines them
- Fewer random I/Os
  - delay log updates, 1 big I/O not 100 small I/Os

What's Next?

- Extend our simulator to include:
  - clustered writes
  - buffer cleaning daemons (syncd, bufd)
  - better disk model
- Implement agraphs in FreeBSD
  - Evaluate implementation complexity
  - Test performance on real & synth workloads

The End

- Comments?
- Questions?
- Jobs?

ncb@cs.wisc.edu

http://www.cs.wisc.edu/~ncb/