Journaling Vs. Soft Updates: Asynchronous Meta-data Protection in File Systems

Margo I. Seltzer, Gregory R. Ganger, M. Kirk McKusick, Keith A. Smith, Craig A. N. Soules, Christopher A. Stein

Presented By:
Abhijit Deodhar

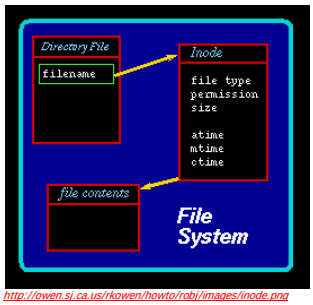
Overview

- Key Points
- Approaches
  - Soft updates
  - Journaling
- Comparison
- Evaluation

Key Points

- Meta-data integrity
- Journaling systems record metadata operations on an **auxiliary log**
- Soft updates uses **ordered writes**

FFS Organization



Source: http://owen.sj.ca.us/rkowen/howto/robj/images/inode.png

Meta-data Integrity

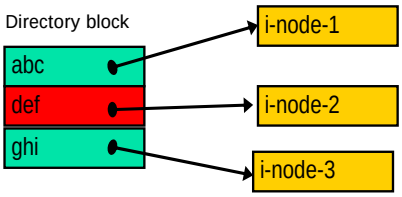
Old directory will contain link to new file

- Meta-data operations modify the **structure** of the file system
  - Creating, deleting or renaming files
- Why is it important?
  - Suppose you delete a file and subsequently crash
    - Privacy compromise ?
- File system recovery must be possible

Update Dependency Rules

1) Never point to a structure before it has been initialized. (inode < direntry)
2) Never reuse a resource before nullifying all previous pointers to it.(inode ptr to data blk < blk realloc)
3) Never reset the last pointer to a live resource before a new pointer has been set. (File rename)

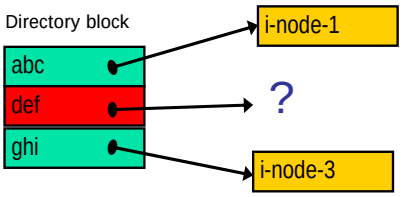
Deleting a File



Assume we want to delete file “def”

7

Deleting a File



Cannot delete i-node before directory entry “def”

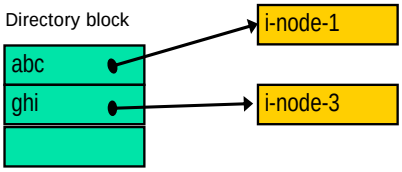
8

Deleting a File

- Correct sequence is
  - Write to disk directory block containing deleted directory entry “def”
  - Write to disk i-node block containing deleted i-node
- Leaves the file system in a consistent state

9

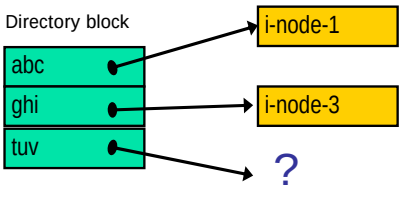
Creating a File



Assume we want to create new file “tuv”

10

Creating a File



Cannot write directory entry “tuv” before i-node

11

Creating a File

- Correct sequence is
  - Write to disk i-node block containing new i-node
  - Write to disk directory block containing new directory entry
- Leaves the file system in a consistent state

12

Synchronous Updates

- FFS
  - Guarantees durability
  - Meta-data operations done through blocking writes
- Problems
  - Increases cost of updates
  - Impacts file system performance
  - Recovery requires full file system scan

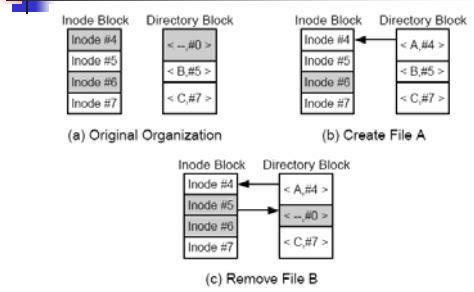
13

Soft Updates

- Delayed writes (write-back cache)
- Dependency information
  - Block level vs. Pointer level
- Advantage – faster file system recovery after crash
- Overhead - extra write
- When to write to disk?

14

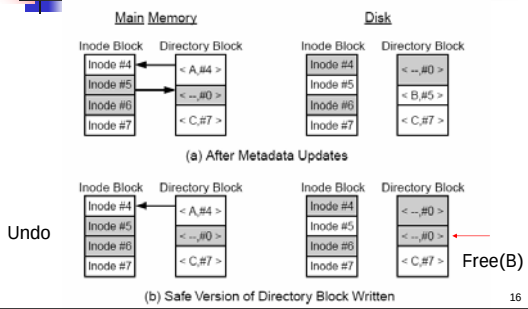
Cyclic Dependency



Source: Soft Updates: A solution to the metadata update problem in File Systems - Ganger

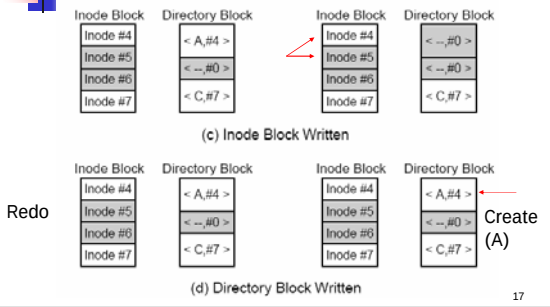
15

Undo/redo Operations



16

Undo/redo Operation Cont.



17

Journaling File Systems

- LFFS-file
  - Write-ahead logging (WAL)
    - Flush meta data to persistent storage before data
  - Circular buffer
    - Size of log proportional to amount of changing meta-data (usually KB)
  - Log space reclamation
    - Periodic syncer daemon
  - Forced checkpoints

18

Journaling File Systems

- LFFS-wafs
  - Auxiliary file system
  - Flexibility
    - Log on a separate high speed disk to reduce contention
  - Asynchronous logs for performance

19

Structure of Journal

- Contains three types of data blocks
  - Metadata: entire contents of a single block of filesystem metadata as updated by the transaction
  - Descriptor: describe other journal metadata blocks (where they really live on disk)
  - Header: contain head and tail of the journal, sequence number, the journal is a circular structure

20

LFFS Recovery

- Superblock has address of last checkpoint
  - LFFS-file has frequent checkpoints
  - LFFS-wafs much less frequent checkpoints
- First recover the log
- Read the log from logical end (backward pass) and undo all aborted operations
- Do forward pass and reapply all updates that have not yet been written to disk

21

File System Configurations

	File System Configurations
FFS	Standard FFS
FFS-async	FFS mounted with the async option
Soft-Updates	FFS mounted with Soft Updates
LFFS-file	FFS augmented with a file log log writes are asynchronous
LFFS-wafs-1sync	FFS augmented with a WAFS log log writes are synchronous
LFFS-wafs-1async	FFS augmented with a WAFS log log writes are asynchronous
LFFS-wafs-2sync	FFS augmented with a WAFS log log is on separate disk log writes are synchronous
LFFS-wafs-2async	FFS augmented with a WAFS log log is on a separate disk log writes are asynchronous

Table 1. File System Configurations.

22

System Comparison

- Semantics
  - Durability
  - Status of file system after a reboot
  - Guarantees about the data in files
  - Ability to provide atomicity

23

Feature Comparison

Feature	File Systems
Meta-data updates are synchronous	FFS, LFFS-wafs-[12]sync
Meta-data updates are asynchronous	Soft Updates LFFS-file LFFS-wafs-[12]async
Meta-data updates are atomic.	LFFS-file LFFS-wafs-[12]*
File data blocks are freed in back-ground	Soft Updates
New data blocks are written before inodes	Soft Updates
Recovery requires full file system scan	FFS
Recovery requires log replay	LFFS-*
Recovery is non-deterministic and may be impossible	FFS-async

Table 2. Feature Comparison.

24

Microbenchmark Results

Create Performance

- Drops in all curves.

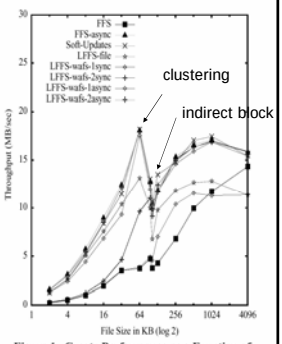


Figure 1. Create Performance as a Function of File Size.

Macrobenchmark Results

-Large data set exceeds cache

-Dependency rollbacks hit

	Unbatch	Expire	Total
	Absolute Time (in seconds)		
FFS-async	1282	640	1922
	Perf. Relative to FFS-async		
FFS	0.63	0.40	0.53
Soft-Updates	0.86	0.89	0.87
LFFS-file	0.95	0.95	0.95
LFFS-wafs-1sync	0.67	0.48	0.59
LFFS-wafs-1async	0.98	0.92	0.96
LFFS-wafs-2sync	0.91	0.67	0.81
LFFS-wafs-2async	0.97	0.95	0.96

Table 5. Netnews Results Normalized to FFS-async. These results are based on a single run, but we observed little variation between multiple runs of any configuration.

26

Results Summary

- Soft updates excels in meta-data intensive microbenchmarks
  - Background deletes
  - No disk I/O for 0-length file create/delete
- Macrobenchmark results are ambiguous

27

Evaluation

- Soft-updates – just another way of doing meta-data updates.
- Difficult to implement on FS which implements directories as b-tree / hashes.
- Updates may proceed out of order.
  - Eg. Create /dir1/a and then /dir2/b.
- Better parallelism due to fine-grained meta-data updates.

28

Evaluation

- Journaling widely adopted in industry (ext3, VxFS, JFS)
- Journaling alone is not sufficient to “solve” the meta-data update problem
  - Cannot realize its full potential when synchronous semantics are required
- Which method to choose -
  - Depends heavily on your file system design goals

29

Questions

30

Delete Performance

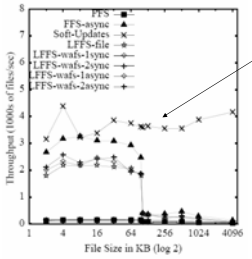


Figure 2. Delete Performance as a Function of File Size.

Background deletes.

31

0-length File Create/delete

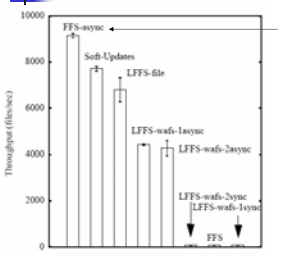


Figure 3. 0-length File Create/Delete Results in Files per Second.

Best case

-Asynchronous

-No disk IO

32

Approaches

- Soft Updates
- Journaling
- Hardware techniques
  - NVRAM
  - MRAM
- Log-structured file systems (LFS)

33