

Distributed File Systems

CS 256/456
Dept. of Computer Science, University
of Rochester

11/6/2008

CSC 2/456

1

Last class: Log-structure File Systems and Solid State Drives

- Log-structure file systems
 - Improve individual disk write throughput when using large caches for reads
 - Improve reliability and recovery overhead via journaling
- Solid State Drives
 - Fast reads, random access
 - Slow write/erase (in blocks) cycle
 - Finite number of writes requiring wear leveling

11/6/2008

CSC 2/456

2

Log-Structured File Systems

- With CPUs faster, memory larger
 - buffer caches can also be larger
 - most of read requests can come from the memory cache
 - thus, most disk accesses will be writes
 - poor disk performance when most writes are small
- LFS Strategy [Rosenblum&Ousterhout SOSP1991]
 - structures entire disk as a log
 - always write to the end of the disk log
 - when updates are needed, simply add new copies with updated content; old copies of the blocks are still in the earlier portion of the log
 - periodically purge out useless blocks

11/6/2008

CSC 2/456

3

Delayed Writes and Data Loss at Machine Crash

- Writes are commonly delayed for better performance
 - data to be written is cached
- A sudden machine crash may result in a loss of data
 - a completed write does not mean the data is safely stored on storage
- fsync() - flush all delayed writes to disk
 - fsync() may not even be totally safe with delayed writes on disk controller buffer cache

11/6/2008

CSC 2/456

4

Consistency: Weaker Form of Reliability

- File system operations are not atomic; a sudden machine crash may leave the file system in an inconsistent state
- Consistency checking and fix (fsck, scandisk)
 - it takes too long
- Journaling file system:
 - maintain a dedicated journal that logs all operations
 - the logging happens before the real operation
 - each logging is made to be atomic
 - after the completion of an operation, its entry is removed from the journal
 - at the recovery time, only journal entries need to be examined ⇒ fast recovery
 - similar to transactions in database systems
 - performance impact?

11/6/2008

CSC 2/456

5

File System Caching

- File content is cached in memory buffer for later reuse
 - what is the basic unit of such caching?
 - Disk blocks vs. clusters vs. pages
- Replacement policy for file system buffer cache
 - LRU replacement is one possibility; but sequential access is very likely in file system I/O
 - MRU or free-behind

11/6/2008

CSC 2/456

6

File System Prefetching

- File content is read ahead of time for anticipated use in the near future
- Often sequential (based on past access history on the file)
- What is the advantage of file prefetching?
- What is the danger of file prefetching?
- A balanced scheme that provides a competitive performance to the optimal scheme [Li et al. EuroSys 2007]

11/6/2008

CSC 2/456

7

Informed Prefetching

- Informed prefetching - prefetching while utilizing some information about application data access pattern
- Application I/O hints [Cao et al. 1994] [Patterson et al. 1995]
- Automatic I/O hints based on speculative execution [Chang&Gibson 2000], [Fraser&Chang 2003]

11/6/2008

CSC 2/456

8

Buffer Cache in Main Memory

- Memory-mapped I/O naturally share page cache with the virtual memory system
- Problems:
 - double buffering
 - inconsistencies

```

    graph TD
        VM[virtual memory] --> VMC[virtual memory page cache]
        MMIO[memory-mapped I/O] --> VMC
        VMC --> Disk[disk]
        FSDIO[file system direct I/O] --> FSC[file system block cache]
        FSC --> Disk
    
```

11/6/2008 CSC 2/456 9

Unified Buffer Cache & Unified Virtual Memory

- A unified buffer cache uses the same page cache to store [Pai et al. 1999]
 - virtual memory pages
 - memory-mapped pages
 - file system direct I/O data

```

    graph TD
        VM[virtual memory] --> UBC[unified buffer page-based]
        MMIO[memory-mapped I/O] --> UBC
        FSDIO[file system direct I/O] --> UBC
        UBC --> Disk[disk]
    
```

11/6/2008 CSC 2/456 10

Multi-level I/O Buffer

- buffer cache in the main memory
- track cache on the disk controller

```

    graph TD
        MM[Main Memory Cache] --> TC[Track Cache on Disk Controller]
        TC --> Disk[(Disk)]
    
```

11/6/2008 CSC 2/456 11

Example File Systems

- MS-DOS/Windows – file allocation table (FAT), NTFS
- Linux – VFS, ext2fs, ext3fs
- Berkeley - FFS
- ...

11/6/2008 CSC 2/456 12

Network vs. Distributed Operating Systems

- Network operating system:
 - Users aware of multiple machines
 - Data/computation migration -> user's responsibility
- Distributed operating system:
 - Access to remote resources similar to local
 - Data/computation migration under control of OS

11/6/2008

CSC 2/456

13

Data Migration

- Whole files (automated ftp): e.g., AFS first version
- On-demand (demand paging): e.g., NFS, SMB, AFS newer versions

11/6/2008

CSC 2/456

14

Computation Migration

- RPC
- Messages (more concurrency)

11/6/2008

CSC 2/456

15

Process Migration

- Rationale
 - Load balancing
 - Computation speedup
 - Hardware preference
 - Software preference
 - Data access
- Either automated (transparent) or user specifies how the process should migrate

11/6/2008

CSC 2/456

16

Distributed File Systems Issues

- Naming and transparency (location transparency versus location independence)
 - Host:local-name
 - Attach remote directories (mount)
 - Single global name structure
- Remote file access
 - Remote-service mechanism
 - Stateful vs. stateless
 - Caching and coherence
 - Cache update policy (write through vs. delayed write)
 - Client-initiated vs. server-initiated
- Reliability and file replication
 - Naming transparency
 - Availability vs. consistency

11/6/2008

CSC 2/456

17

NFS (Network File System)

- Deals with heterogeneity using RPC/XDR
- Stateless – no open and close of files
- Interface transparent to user via VFS

11/6/2008

CSC 2/456

18

The Andrew File System (AFS)

- Unified namespace (one /afs for everybody)
- One read-write copy
- Protection using access control lists (ACLs)
- Security using Kerberos 5 authentication

11/6/2008

CSC 2/456

19