CS323 Operating Systems
File System III

Yuanyuan Zhou

Lecture 25

3/31/2003

Content of this lecture

- Administrative announcements
- File systems basic concepts
- Summary

Directories in Unix

- Stored like regular files
  - Contents are file names and inode #s
  - Names are nul-terminated strings
- Logic
  - Separates file from location in tree
  - File can appear in multiple places
- What are the drawbacks?

Directory Structure Organization

- maps symbolic names into logical file names
  - search
  - create file
  - list directory
  - backup, archival, file migration

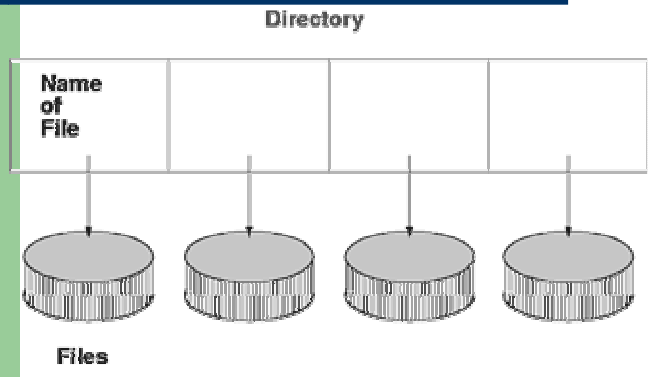
Directory Contents

- file name symbolic name
- file type indicates format of file
- location device and location
- size
- protection
- creation, access, and modification date
- owner identification

Directory Contents

- the following items may be stored on a per file, process basis
  - current read, write position
  - usage count

Single Level Directory



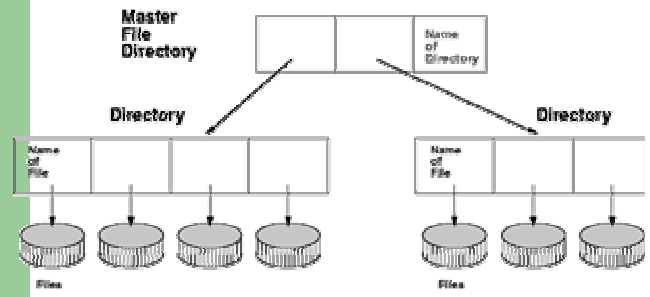
Problems With Single Level Directory

- more than one user
- large file systems
- moving files from one system to another
- name clashes
- modularity

Two-level Directory

- introduced to remove naming problems between users
- first level contains list of user directories
- second level contains user files
- system files kept in separate directory or level 1
- sharing accomplished by naming other users files

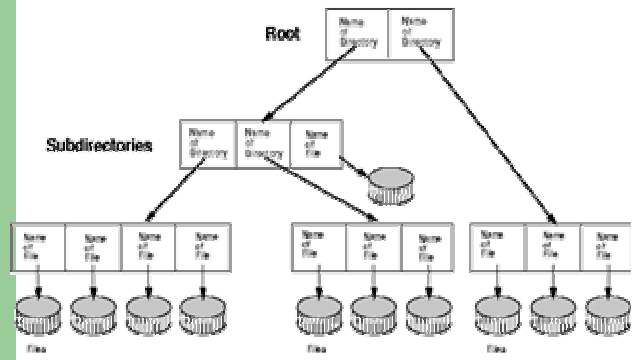
Two-level Directory



Tree Structured Directories

- arbitrary depth of directories
- leaf nodes are files
- interior nodes are directories
- path name lists nodes to traverse to find node
- use absolute paths from root
- use relative paths from current working directory pointer

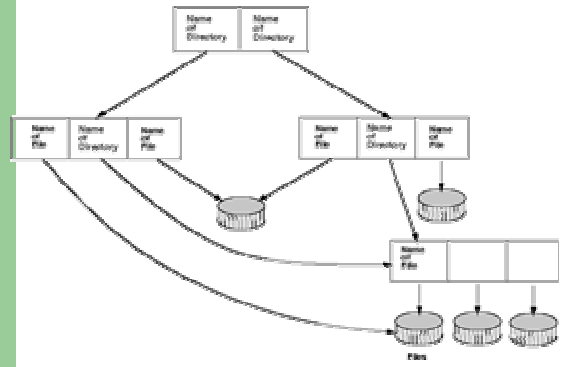
Tree Structured Directories



Acyclic Graph Structured Directories

- Acyclic graphs allow sharing
- two users can name same file
- implementation by links - use logical names of files (file system and file)
- implementation by symbolic links map pathname into a new pathname
- duplicate paths complicates backup copies
- need reference counts for hard links

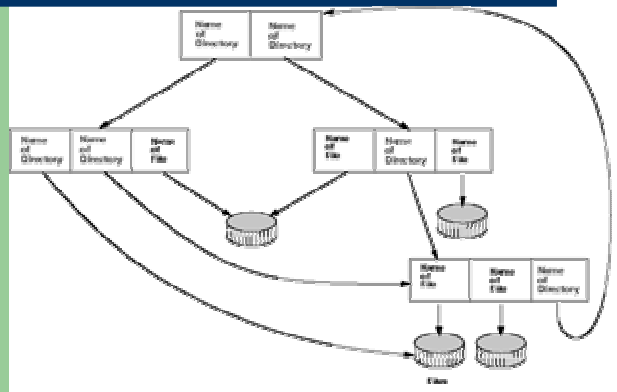
Acyclic Graph Structured Directories



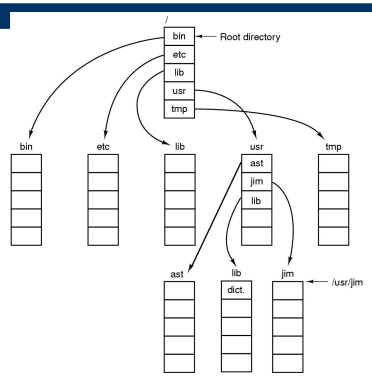
General Graph Structured Directories

- cycles
- more flexible
- more costly
- need garbage collection (circular structures)
- must prevent infinite searches

General Graph Structured Directories



Path Names



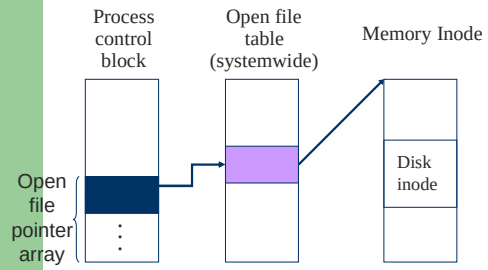
Directory Operations

1. Create
2. Delete
3. Opendir
4. Closedir
5. Readdir
6. Rename
7. Link
8. Unlink

Effects of Corruption

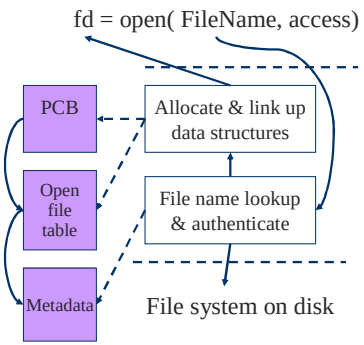
- inode – file gets “damaged”
  - Maybe some “free” block gets viewed
- Directory – “lose” files/directories
  - Might get to read deleted files
- Superblock – can’t figure out anything
  - This is why we replicate the superblock

Data Structures for A Typical File System



Opening A File

- File name lookup and authentication
- Copy the file metadata into the in-memory data structure, if it is not in yet
- Create an entry in the open file table (system wide) if there isn't one
- Create an entry in PCB
- Link up the data structures
- Return a pointer to user



Open-file Table Information

- File Pointer
  - current file position pointer
- File Open Count
  - counter which tracks the number of opens and closes. If files are closed, the OS must reuse the space in open-file table. Because multiple processes may open the same file, OS must wait until the last file closes before removing the entry.
- Disk Location
  - information needed to locate the file on disk.

Reading And Writing

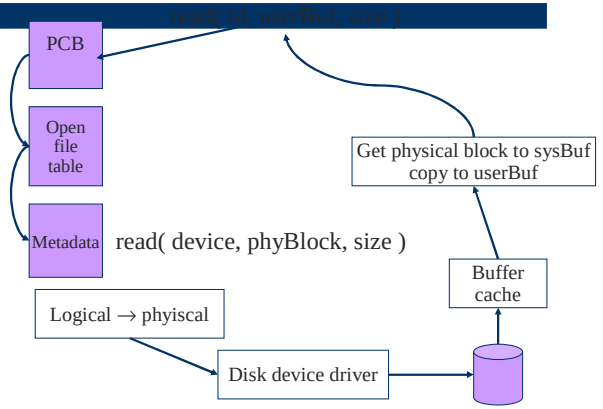
What happens when you…

- read 10 bytes from a file?
- write 10 bytes into an existing file?
- write 4096 bytes into a file?

Disk works on blocks (sectors)

- Can have temporary (ephemeral) buffers
- Longer lasting buffers = disk cache

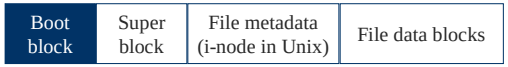
Reading A Block



Other File Issues

- Memory Mapping a File
  - Associate a part of the VM space with a section of a file.
  - Reads and writes to that memory region are then treated as reads and writes to the file.
- Internal File Structure
  - mapping between logical record and physical record
  - packing a number of logical records into physical blocks.
  - The logical record size, physical block size and packing technique determine how many logical records are in each physical block.
- Consistency Semantics
  - important criterion for evaluation of any file system that supports sharing of files.

A Disk Layout for A File System



- Superblock defines a file system
  - size of the file system
  - size of the file descriptor area
  - free list pointer, or pointer to bitmap
  - location of the file descriptor of the root directory
  - other meta-data such as permission and various times
- For reliability, replicate the superblock

File Usage Patterns

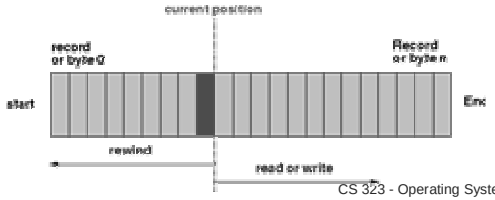
- How do users access files?
  - Sequential: bytes read in order
  - Random: read/write element out of middle of arrays
  - Whole file or partial file
- How are files used?
  - Most files are small
  - Large files use up most of the disk space
  - Large files account for most of the bytes transferred
- Bad news
  - Need everything to be efficient

Data Structures for Disk Management

- A "header" for each file (part of the file meta-data)
  - Disk sectors associated with each file
- A data structure to represent free space on disk
  - Bit map
    - 1 bit per block (sector)
    - blocks numbered in cylinder-major order, why?
  - Linked list
  - Others?
- How much space does a bit map need for a 4G disk?

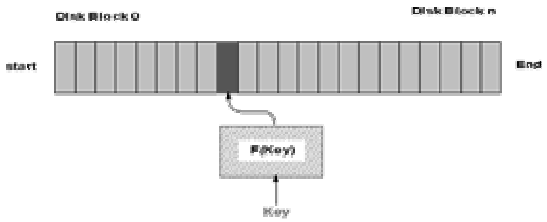
Sequential File Organization: Tape

- Organized like a tape; can be on a tape or direct access device.
- The records are stored and also retrieved in physical order.
- The ``next'' record physically follows the current one.
- Inserting a record means rewriting the file



Directed File Access Organization: Disk

- User maps key into disk address
- Access in physical order or random
- No notion of next record

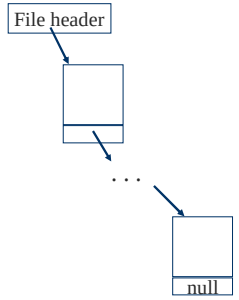


File Allocation

- Contiguous
- Non-contiguous (linked)
- Tradeoffs?

Linked Files

- File header points to 1st block on disk
- Each block points to next
- Pros
  - Can grow files dynamically
  - Free list is similar to a file
- Cons
  - random access: horrible
  - unreliable: losing a block means losing the rest



Contiguous Allocation

- Request in advance for the size of the file
- Search bit map or linked list to locate a space
- File header
  - first sector in file
  - number of sectors
- Pros
  - Fast sequential access
  - Easy random access
- Cons
  - External fragmentation
  - Hard to grow files