OPERATING SYSTEMS

FILE SYSTEM 1

10 File System

We will examine this chapter in three subtitles:

- Mass Storage Systems
- File System Interface
- File System Implementation

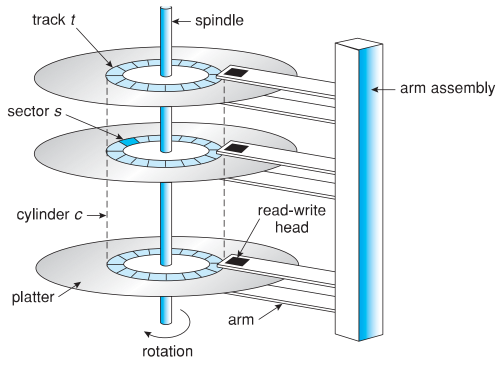
10.1 Mass Storage Systems

- Mass Storage Structure
- Performance
- Other Storage Devices
- Disk Management
- Disk Scheduling
- RAID

10.1.1 Mass Storage Structure

- Magnetic disks provide bulk of secondary storage
- It composes of some parts:
  - Platters, read/write heads, arms, spindle
- Disk carries the data by help of some logical units:
  - Cylinders, tracks, sectors
- The surface of a platter is logically divided into circular tracks, which are subdivided into sectors.
- The set of tracks that are at one arm position makes up a cylinder.

10.1.1 Mass Storage Structure

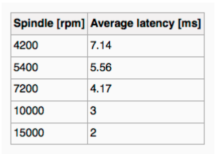


10.1.1 Mass Storage Structure

- Drives rotate at 60 to 250 times per second
- Transfer rate shows speed of data flow between drive and computer
- Positioning time (random-access time) is sum of times to move disk arm to desired cylinder (seek time) and to rotate under the disk head for desired sector (rotational latency)
- Drive attached to computer via I/O bus (EIDE, ATA, SATA, USB, Fibre Channel, SCSI, SAS, Firewire)

10.1.1 Mass Storage Structure

- Platters range from .85” to 14”
- Commonly 3.5”, 2.5”, and 1.8”
- Capacity from 30GB to 3TB
- Performance
  - Transfer Rate – theoretical – 6 Gb/sec
  - Effective Transfer Rate – real – 1Gb/sec
  - Seek time from 3ms to 12ms – 9ms common for desktop drives
  - Average seek time measured or calculated based on 1/3 of tracks
  - Latency based on spindle speed
    1 / (RPM / 60) = 60 / RPM
  - Average latency = ½ latency



Spindle [rpm]	Average latency [ms]
4200	7.14
5400	5.56
7200	4.17
10000	3
15000	2

10.1.2 Performance

- Average access time = average seek time + average latency
  - For a fastest disk 3ms + 2ms = 5ms
  - For a slow disk 9ms + 5.56ms = 14.56ms
- Transfer time = amount to transfer / transfer rate
- Average I/O time = average access time + transfer time
  + controller overhead

10.1.2 Performance

Example: Find the average I/O time to transfer a 4KB block on a 7200 RPM disk with a 5ms average seek time, 1Gb/sec transfer rate with a 0.1ms controller overhead.

average access time = 5ms + 4.17ms = 9.17ms
transfer time = 4KB/1Gbps = 32 / 1024 = 0.031ms

Average I/O time = 9.17ms + 0.031ms + 0.1ms
= 9.301ms

10.1.3 Other Storage Devices

Solid State Disks
- Nonvolatile memory used like a hard drive
- Can be more reliable than HDDs
- More expensive per MB
- Maybe have shorter life span
- Less capacity
- Much faster
- Busses can be too slow -> connect directly to PCI for example
- No moving parts, so no seek time or rotational latency

10.1.3 Other Storage Devices

Magnetic Tape
- Was early secondary-storage
- Relatively permanent and holds large quantities of data
- Access time slow
- Random access ~1000 times slower than disk
- Mainly used for backup, storage of infrequently-used data, transfer medium between systems
- Kept in spool and wound or rewound past read-write head
- Once data under head, transfer rates comparable to disk
  140MBps and greater
- 200GB to 1.5TB typical storage

10.1.4 Disk Management

- Before a disk can store data, it must be divided into sectors. Low-level formatting fills the disk with a special data structure for each sector.
  - The data structure for a sector typically consists of a header, a data area (usually 512 bytes in size), and a trailer
  - The header and trailer contain information used by the disk controller, such as a sector number and an error-correcting code
- To use a disk to hold files, the OS still needs to record its own data structures on the disk. It does so in two steps.
  - First step is to partition the disk into one or more groups of cylinders. Each treated as a logical disk
  - Then OS makes a file system as “logical formatting”

10.1.4 Disk Management

- After low-level formatting creates logical blocks, disk drive can be addressed as large one-dimensional array of logical blocks, where the logical block is the smallest unit of transfer.
- The one-dimensional array of logical blocks is mapped into the sectors of the disk sequentially.
  - Sector 0 is the first sector of the first track on the outermost cylinder
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost
- By using this mapping, a logical block number can be converted into a real disk address that consists of a cylinder number, a track number, and a sector number.

10.1.4 Disk Management

- On media that use constant linear velocity (CLV) like CD-ROM drive, the density of bits per track is uniform.
  - As we move from outer zones to inner zones, the number of sectors per track decreases.
  - Tracks in the outermost zone typically hold 40 percent more sectors than do tracks in the innermost zone.
  - The drive increases its rotation speed as the head moves from the outer to the inner tracks to keep the same rate of data moving under the head.
- Alternatively, the disk rotation speed can stay constant, and the density of bits decreases from inner tracks to outer tracks to keep the data rate constant. This method is used in hard disks and is known as constant angular velocity (CAV).

10.1.4 Disk Management

- Computers access disk storage in two ways:
  - Via I/O ports (or host-attached storage). It is common on small systems.
  - Via a remote host in a distributed file system; this is referred to as network-attached storage.
- Host-attached storage accessed through local I/O ports. These ports use several technologies:
  - The typical desktop PC uses an I/O bus architecture called IDE or ATA. This architecture supports a maximum of two drives per I/O bus
  - A newer, similar protocol that has simplified cabling is SATA.
  - High-end workstations and servers generally use more sophisticated I/O architectures, such as SCSI and fiber channel (FC)

10.1.4 Disk Management

- SCSI itself is a bus architecture
  - It supports up to 16 devices on one cable.
  - Generally, the devices include one controller card in the host and up to 15 storage devices (target).
  - It provides the ability to address up to 8 logical units in each target.
- FC is high-speed serial architecture operating on optical fiber
- A wide variety of storage devices are suitable for use as host-attached storage. Among these are hard disk drives, RAID arrays, and CD, DVD, and tape drives.

10.1.5 Disk Scheduling

- One of the responsibilities of the OS is to use the hardware efficiently. For the disk drives, meeting this responsibility entails having fast access time and large disk bandwidth.
- The access time has two major components
  - The seek time is the time for the disk arm to move the heads to the cylinder containing the desired sector.
  - The rotational latency is the additional time for the disk to rotate the desired sector to the disk head.
- The disk bandwidth is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer.

10.1.5 Disk Scheduling

- We can improve both the access time and the bandwidth by scheduling the servicing of disk I/O requests in a good order.
- For a multiprogramming system with many processes, the disk queue may often have several pending requests.
- Thus, when one request is completed, the OS chooses which pending request to service next.
- The OS can make this choice by DISK-SCHEDULING ALGORITHMS

10.1.5 Disk Scheduling

- Several algorithms exist to schedule the servicing of disk I/O requests
- We illustrate scheduling algorithms with a request queue

Example

Let's the queue be "98, 183, 37, 122, 14, 124, 65, 67"

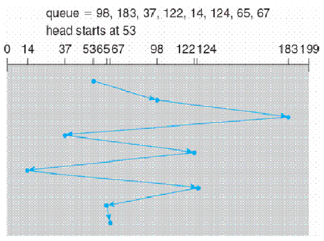
and the head pointer starts at 53

10.1.5 Disk Scheduling

Example "98, 183, 37, 122, 14, 124, 65, 67"

FCFS method finds total head movement of 640 cylinders

In First Come First Serve, all incoming requests are placed at the end of the queue.

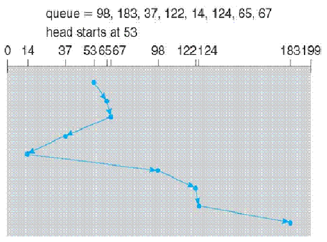


10.1.5 Disk Scheduling

Example "98, 183, 37, 122, 14, 124, 65, 67"

SSTF method finds total head movement of 236 cylinders

Shortest Seek Time First selects the request with the minimum seek time from the current head position. SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests

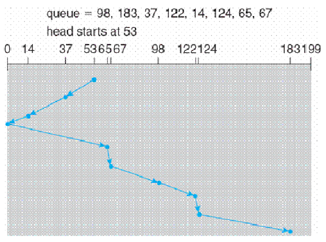


10.1.5 Disk Scheduling

Example "98, 183, 37, 122, 14, 124, 65, 67"

SCAN method finds total head movement of 208 cylinders

The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.

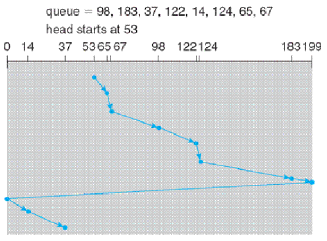


10.1.5 Disk Scheduling

Example "98, 183, 37, 122, 14, 124, 65, 67"

C-SCAN method finds total head movement of ??? cylinders

The head moves from one end of the disk to the other, servicing requests as it goes. When it reaches the other end, it immediately returns to the beginning of the disk.

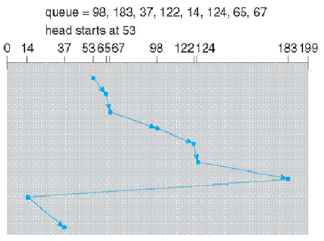


10.1.5 Disk Scheduling

Example "98, 183, 37, 122, 14, 124, 65, 67"

C-LOOK method finds total head movement of ??? cylinders

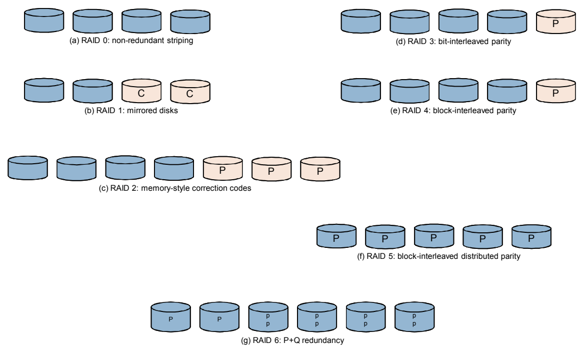
Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk



10.1.6 RAID

- RAID (redundant array of inexpensive disks) are commonly used to address the performance and reliability issues
- Extends the mean time to failure, repair and data loss
- RAID within a storage array can still fail if the array fails, so automatic replication of the data between arrays is common
- Frequently, a small number of hot-spare disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them
- The hot spare device runs as part of a working system. When a component fails, the hot spare is switched into operation immediately.

10.1.6 RAID


P
(a) RAID 0: non-redundant striping
(d) RAID 3: bit-interleaved parity
C
C
P
(b) RAID 1: mirrored disks
(e) RAID 4: block-interleaved parity
P
P
P
(c) RAID 2: memory-style correction codes
P
P
P
P
P
(f) RAID 5: block-interleaved distributed parity
P
P
p p
p p
p p
p p
(g) RAID 6: P+Q redundancy

10.2 File System Interface

- File Concept
- Access Methods
- Directory Structure
- File System Mounting
- File Sharing
- Protection
- Access Type

10.2.1 File Concept

- File is an abstract data collection that has a specific purpose. Almost all information stored in a computer must be in a file. Each file writes/reads contiguous blocks of bytes from a logically contiguous address space.
- Generally there are two types of files
  - Data (numeric, character, binary…)
  - Program
- According to contents defined by file's creator, there are many types (text, source, executable files…)

10.2.1 File Concept

- Information about files are kept in the directory structure, which is maintained on the disk
- Files have some attributes:
  - Name, Identifier, Type, Location, Size
  - Protection, Time-date, User identification
- Many variations, including extended file attributes such as file checksum

10.2.1 File Concept

- Name – only information kept in human-readable form
- Identifier – unique tag (number) identifies file within file system
- Type – needed for systems that support different types
- Location – pointer to file location on device
- Size – current file size
- Protection – controls who can do reading, writing, executing
- Time, date, and user identification – data for protection, security, and usage monitoring

10.2.1 File Concept

Some operations implemented on files
- Create – new file
- Write – at *write pointer* location
- Read – at *read pointer* location
- Seek – reposition within file
- Delete – from file system
- Open – search the directory structure on disk for entry, and move the content of entry to memory
- Close – move the content of entry in memory to directory structure on disk

10.2.1 File Concept

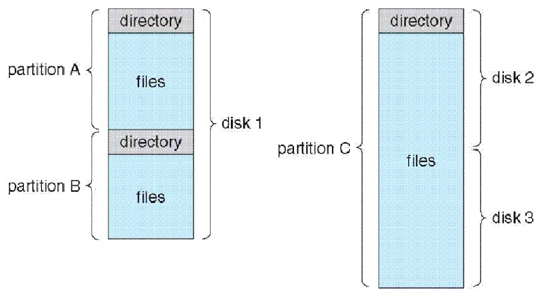
For opened files, some management rules are needed
- Open-file table: tracks open files
- File pointer: pointer to last read/write location, per process that has the file open
- File-open count: counter of number of times a file is open – to allow removal of data from open-file table when last processes closes it
- Disk location of the file: cache of data access information
- Access rights: per-process access mode information

10.2.2 Access Methods

There are two usual type of file access

- Sequential Access
  - read next, write next, reset
- Direct Access – file has fixed length logical records
  - read n, write n, position to n, read next, write next, rewrite n
    (n : Relative block number) Relative block numbers allow OS to decide where file should be placed

10.2.3 Directory Structure



10.2.3 Directory Structure

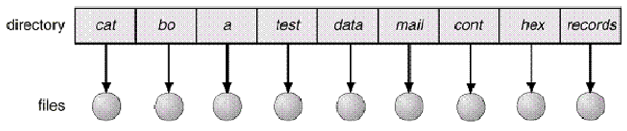
- To keep track of files, file systems normally have directories or folders. Usually, a directory is itself a file.
- The directory can be viewed as a symbol table that translates file names into their directory entries.
- A typical directory entry contains information (attributes, location, ownership) about a file.
- We want to be able
  - to insert entries,
  - to delete entries,
  - to search for a named entry,
  - to list all the entries in the directory.

10.2.3 Directory Structure

- The operations to perform on a directory:
  - **Search for a file** in a directory structure to find the entry filename.
  - **Create a file** as a new file and added to the directory.
  - **Delete a file** remove a file from the directory.
  - **List a directory** list the files in a directory and the contents of the directory entry for each file in the list.
  - **Rename a file** is used to change a filename when the contents or use of the file changes.

10.2.3.1 Single Level Directory

It is the simplest directory structure. All files are contained in the same directory, which is easy to support and understand.

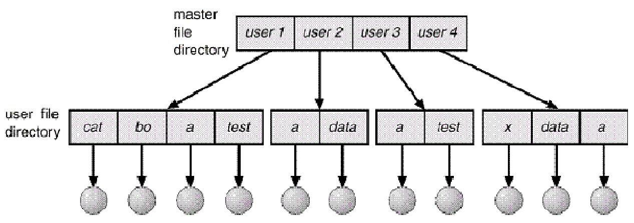


10.2.3.1 Single Level Directory

- On early personal computers, this system was common because there was only one user.
- A single-level directory has significant limitations depending on the number of files and system user.
  - **Naming problem:** convenient to users.
    - Two users can have same name for different files.
    - The same file can have several different names.
  - **Grouping problem:** logical grouping of files by properties, (e.g., all Java programs, all games, …)

10.2.3.2 Two Level Directory

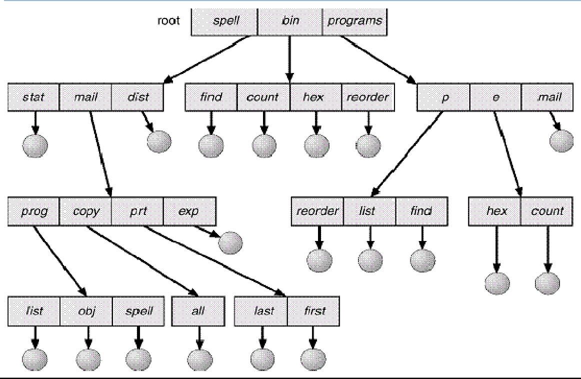
In the two-level directory structure, each user has his own user file directory (UFD). The UFDs have similar structures, but each lists only the files of a single user.



10.2.3.2 Two Level Directory

- In the two-level directory structure, each user has his own user file directory (UFD). The UFDs have similar structures, but each lists only the files of a single user.
- Although the two-level directory structure solves the name-collision problem, it still has disadvantages.
- When the users want to cooperate on some task and to access one another's files, they can not. Also the grouping problem is present.

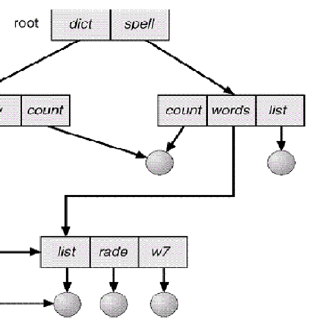
10.2.3.3 Tree Structured Directory



10.2.3.3 Tree Structured Directory

- It allows users to create their own subdirectories and to organize their files accordingly.
- The tree structure has a root directory, and every file in the system has a unique path name.
- All directories have the same internal format.
- Path names can be of two types:
  - An absolute path name begins at the root and follows a path down to the specified file, giving the directory names on the path.
  - A relative path name defines a path from the current directory.

10.2.3.4 Acyclic-Graph Directories



10.2.3.4 Acyclic-Graph Directories

- The acyclic graph is a natural generalization of the tree-structured directory scheme.
- Some directories should be shared. A shared directory or file will exist in the file system in two (or more) places at once.
- A classical tree structure prohibits the sharing of files or directories. An acyclic graph (cycle-free graph) allows directories to share subdirectories and files.

10.2.3.4 Acyclic-Graph Directories

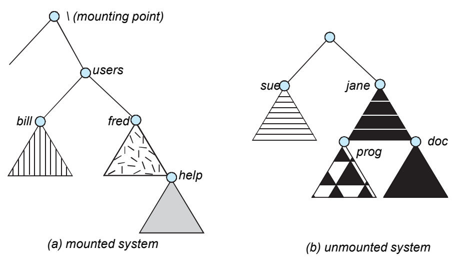
- Several problems must be considered carefully for an acyclic-graph directory structure.
  - A file may have multiple absolute path names. Thus, distinct file names may refer to the same file.
  - Another problem involves deletion. When can the space allocated to a shared file be deallocated and reused?
    - One possibility is to remove the file whenever anyone deletes it, but this action may leave dangling pointers to the now nonexistent file.
    - Worse, if the remaining file pointers contain actual disk addresses, and the space is subsequently reused for other files, these dangling pointers may point into the middle of other files.

10.2.4 File System Mounting

- A file system must be mounted before it is available.
- The OS is given the name of the device and the mount point (the location within the file structure) to mount it.
- Typically, a mount point is an empty directory. Next, the OS verifies that the device contains a file system.
- Finally, the OS notes in its directory structure that a file system is mounted at the specified mount point.
- This scheme enables the OS to traverse its directory structure.

10.2.4 File System Mounting

- To illustrate file mounting, consider the file system depicted in (a), where the triangles represent sub-trees of directories that are of interest.



10.2.5 File Sharing

- File sharing is very desirable for multiple users. When an OS is used by multiple users, some subjects (file sharing, file naming, and file protection) become pre-eminent.
- To implement sharing and protection, the system must maintain more file and directory attributes than are needed on a single-user system.
- Most systems have evolved to use the concepts of file (or directory) owner (or user) and group.
  - The owner is the user who can change attributes and grant access and who has the most control over the file.
  - The group attribute defines a subset of users who can share access to the file.

10.2.6 Protection

48

- When information is stored in a computer system, we want to keep it safe from physical damage (reliability) and improper access (protection).
- Reliability is generally provided by duplicate copies of files (copy disk files to tape).
- File systems can be damaged by hardware problems (such as errors in reading or writing), power surges or failures, head crashes, dirt, temperature extremes, and vandalism. Files may be deleted accidentally. Bugs in the file-system software can also cause file contents to be lost.

10.2.7 Access Type

49

- The need to protect files is a direct result of the ability to access files.
  - Systems that do not permit access to the files of other users do not need protection.
  - Alternatively, we could provide free access with no protection.
- Both approaches are too extreme. Third one is controlled access. Different types of operations may be controlled:
  - Read, Write, Execute, Append, Delete, List
- Other operations, such as renaming, copying, and editing the file, may also be controlled. These higher-level functions may be implemented by a system program that makes lower-level system calls.

10.2.7 Access Type

50

- Mode of access: read, write, execute
- Three classes of users on Unix / Linux

```
                                         RWX
a) owner access          7      ⇒       1 1 1
                                         RWX
b) group access          6      ⇒       1 1 0
                                         RWX
c) public access         1      ⇒       0 0 1
```

- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.

```
          o g p
chmod 761 game
```