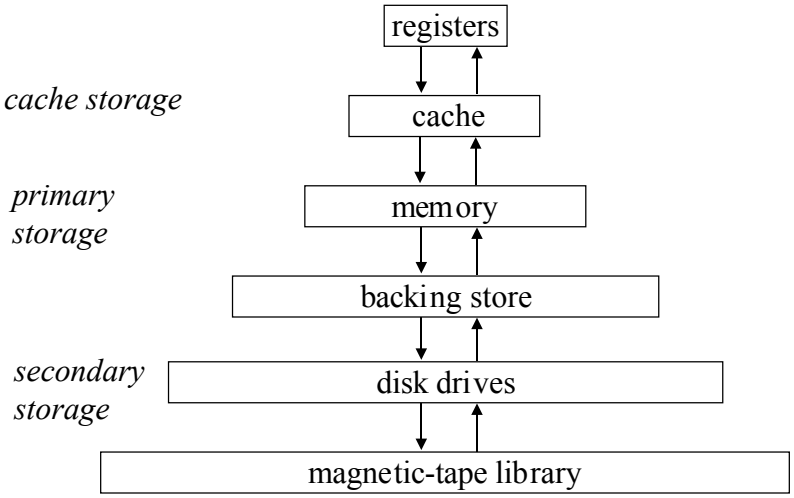
Silberschatz and Galvin Chapter 10

File-System Interface

File System Interface

- Physical storage mechanisms (see 2.3 and 2.4)
- Files--logical storage unit (abstract) that is independent of actual storage device
- Directory structure--organizing the collection of files
- Partitions--an additional division sometimes found
- File protection

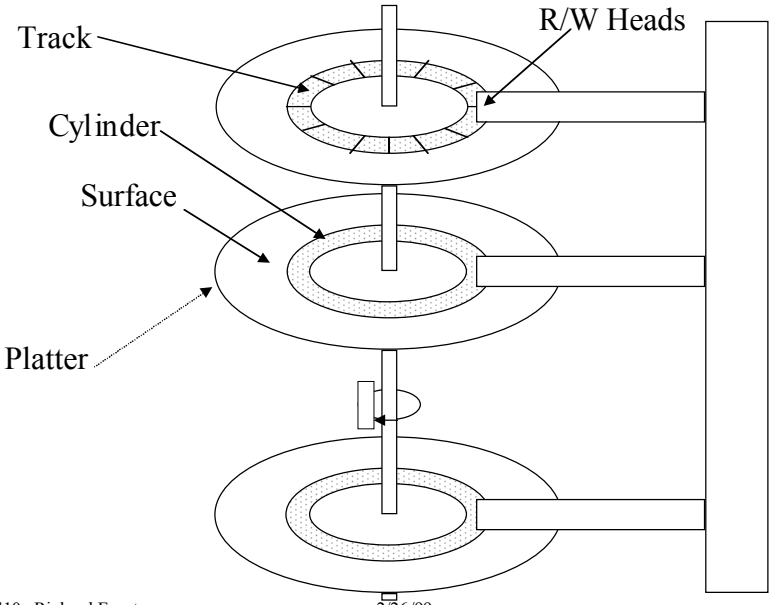
Storage Hierarchy



Primary and Secondary Storage

- Primary storage: small, volatile
- Secondary storage: large, nonvolatile. Able to hold very large amounts of data permanently. Example: magnetic disks, magnetic tapes, optical disks.
- Magnetic disks:

Disk Structure



Disk Terminology

- Cylinder: all tracks that can be accessed without moving the read/write head
- Tracks divided into blocks
- Fixed-size blocks define a sector
- Sector: smallest unit of information that can be transfered to/from disk

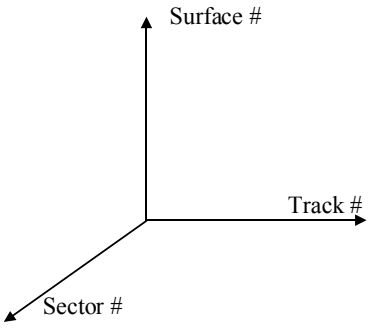
Disk Access Time

- seek time: time to position heads on cylinder (a fixed head disk does not require seek time but is more expensive than a moving-head disk)
- rotational latency: delay in accessing material once seek accomplished (time required to wait for data to rotate around under head)
- Transmission time: time to transfer information once it is under the head.
- access time = seek time + rotational latency
  +read/write transmission time
  seek time >> read/write time

Disk Addressing

- Disk address = (disk driver id,
  surface number,
  track number,
  sector number)

A disk is a three-dimensional array of sectors



Block address b, given cylinder i, surface j, sector k

$b = k + s * (j + i * t)$

s, the number of sectors per track
t, the number of tracks per cylinder (e.g., number of surfaces)

Accessing sequential disk addresses

- Question: what is the cost of accessing block b+1 given b?

Issues of Disk Management

- Allocation method
  - mapping: files ==> disk sectors/blocks
- Free space management
  - data structure and access method
- Disk head scan methods
  - implementation cost, run-time overhead, fairness

Magnetic Tape: Comparison

- 9 tracks on 1/2 inch wide tape (one bit per track)
- Variable length records (about 20-30000 bytes)
- Density in implementation varys from about 200 to 6250 bytes per inch (bpi) with higher densities more common now
- Tape speed: 20 to 200 inches per second. Takes time to come to speed and to stop. This can be megabytes/second when at speed, so DMA transfer may be required
- Inter-record gap is about .6 inch
- Can read or write at end but can only read in middle of tape (because of alignment of subsequent records)

Magnetic Tape

- Tape drives are not fully random-access devices.
- Disk can read/rewrite single sectors in middle of disk, can seek to locations relatively directly, etc.

File concept

- Contiguous logical address space
- Types:
  - Data
    - numeric
    - character
    - binary
  - Program
    - source
    - object (load image)
  - Documents

File Systems

- A file is a named collection of related information that is recorded on secondary storage
- File is a *logical* storage unit: independent of actual storage device; mapped by OS onto physical devices
- Generally *persistent* (e.g., across power failures); *nonvolatile*
- Referred to by *name* (for convenience of human users)
- May have *types* (e.g., source, data, object, executable)
- A file is an abstract data object with specific attributes and operations provided by the system

File attributes

- Name: symbolic name; the only information kept in human-readable form
- Type: if supported by system (more later)
- Location: pointer to device & location of the file on device
- Size: current size and perhaps the maximum allowed size
- Protection: access-control information (e.g., reading, writing, executing, etc.)
- Time, date, and user identification: e.g., creation, last modification, last use
- Usage count, owner, etc.

File attributes

- Kept in the directory structure
  - Also resides on secondary storage
  - 16 to 1000 bytes to store file attribute information
  - Directories may themselves be very large

File operations

- Create: allocate space, enter into directory
- Write: given name and information to be written (system keeps *write* pointer to file)
- Read: given file name and destination of information (system keeps *read* pointer)
- Reposition within a file (also known as file *seek*)
- Delete: release space and erase from directory
- Truncate: keeps directory entry/attributes but deletes contents (resets length to 0)
- Open/close file

File operations
Open/close files

- open/close file operations add/delete entry to open-file table. File accesses via open-file table
- Information associated with open file
  - file pointer (if no offset in read/write)
  - file open count (to keep from removing entry from open-file table prematurely)
  - disk location of the file

File Type

- By extension (.exe, .com, .bat, ...)
- By type attribute (e.g., Macintosh with type/creator attributes---creator identifies application to be launched).
- By "magic number" (as in Unix; embedded into file, at start)

Common file types

File type	**Usual Extension**	**Function**
Executable	exe, com, bin or none	ready-to-run machine-language program
Object	obj, o	compiled machine language, not linked
Source code	c, p, pas, f77, asm, a	source code in various languages
Batch	bat, sh	commands to the command interpreter
Text	txt, doc	textual data, documents

Common file types

File type	Usual Extension	Function
Word processor	wp, tex, rrf, …	various word processor formats
Library	lib, a	libraries of routines
Print or view	ps, dvi, gif	ASCII or binary file
Archive	arc, zip, tar	related files grouped into one file, sometimes compressed

File structure

- Potential structures
  - None - sequence of words, bytes
  - Simple record structure
    - Lines
    - Fixed length
    - Variable length
  - Complex Structures
    - Formatted document
    - Relocatable load file
- Can simulate last two with first method by inserting appropriate control characters. Operating system or program can establish file structure.

File Structure

- File type may indicate internal structure of file (e.g., source or object)
- IBM mainframe systems, for example, support a very wide range of access methods (see later)
- UNIX, MS-DOS, others, support only a minimal number of file structures. (UNIX files are sequence of 8-bit bytes)
- Macintosh resource fork and data fork
- Logical record size and physical block size (blocking factor)--packing a number of logical records into physical blocks. Block allocation results in internal fragmentation, in any case

File Access Methods

- Sequential Access (as in magnetic tape)
- Direct Access (or relative access). Logical records can be read/written in no particular order. read/write must include a block number as parameter
- Other access methods

Sequential access

read next

write next

reset or *skip n*

no *read* after last *write* (*rewrite*)

Direct access

read n

write n

position to n

- *read next*
- *write next*

rewrite n

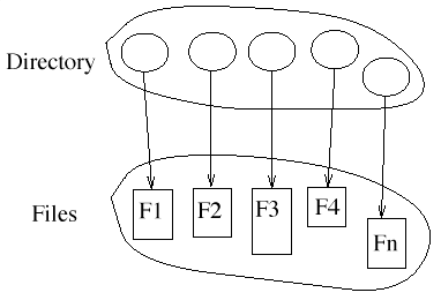
n = relative block number

Other access methods

- Indexed methods, e.g., ISAM (indexed sequential access method, which has file sorted on a defined key. Access look in master index for block number of secondary index. Secondary index read then binary searched for block with desired record. This block is then searched sequentially.)

Directory structure

- A collection of nodes containing information about all files
- Resides on disk, along with files



Information in a device directory

- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed (for archival)
- Date last updated (for dump)
- Owner ID (who pays)
- Protection information (discuss later)

Directory Operations

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system (e.g., for backup purposes)

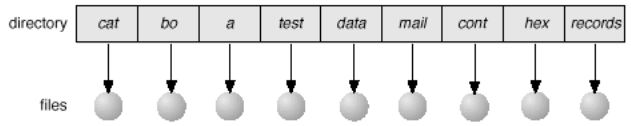
Goals in (logical) directory organization

- Efficiency--enable locating a file quickly
- Naming--convenient to users
  - Two users (or two directories) can have the same name for different files
  - The same file can have several different names
- Grouping--logical grouping of files by properties (e.g., extension, date changed, etc.)

Logical Directory Structures

- Single-level Directory
- Two-level Directory
- Tree-structured Directory
- Acyclic-graph Directory
- General-graph Directory

Single level directory



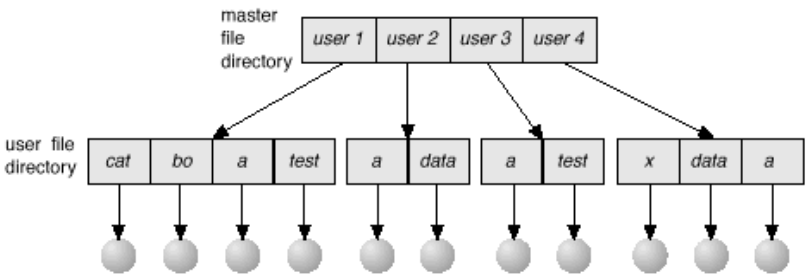
Single-Level Directory

- Advantage: simple to support and understand
- Disadvantage: files must have unique names (multiple users may clash; names may be limited in length; large directories may be hard to remember)

Two-Level Directory

- Each user has own user file directory (UFD)
- Master file directory (MFD) holds pointers to UFDs
- Disadvantage: discourages cooperation

Two-Level Directory

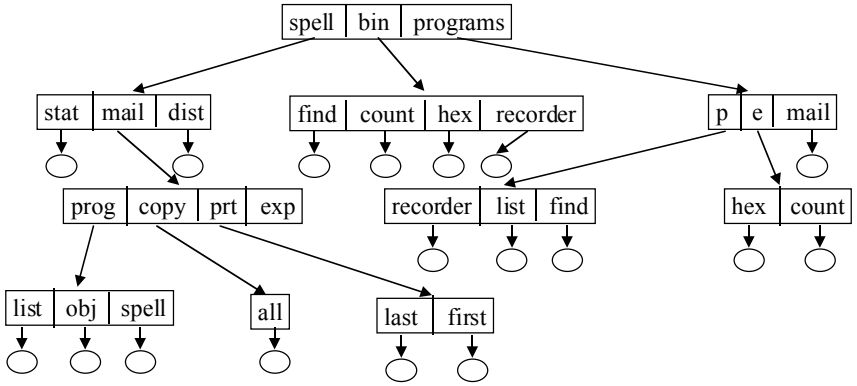


Tree-Structured Directories

- Natural generalization of two-level directories
- As in MS-DOS
- current directory, change directory operation
- policy decision: how to handle requests to delete a (non-empty) directory (prohibit, recurse, ...)

File System Organizations

- Objectives: Providing facility for locating, accessing, and protecting files in the system.
- Common approach: tree structured directory system
- Path name + file name has to be unique.

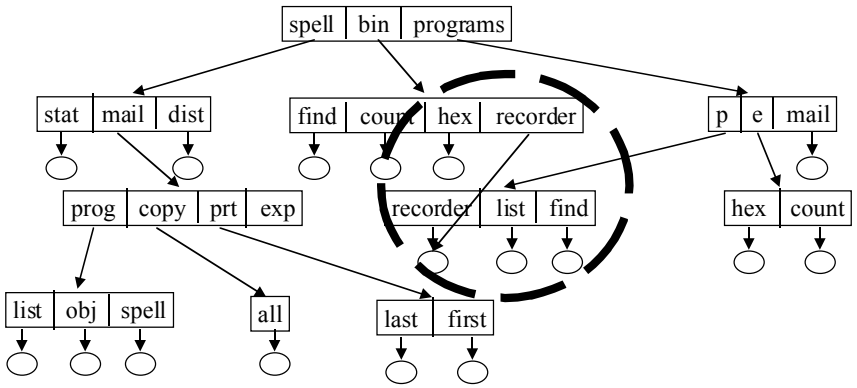


Acyclic-Graph Directories

- Add to tree-structured directories, for example, Unix **ln**
- permits the sharing of files and subdirectories
- the *same* file/subdirectory exists in the file system in two or more places at the same time

File System Organizations

recorder is now found in two locations in the file system



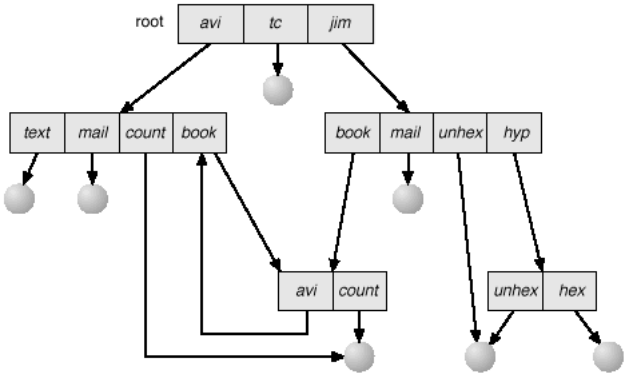
Acyclic-graph directory structures

- File now has multiple absolute path names (aliasing problem). Can backup avoid copying the same file twice?
- Deletion: when can the space be reclaimed? Reference counts, for example (or dangling links as another possible solution)
- Who gets charged for file space?
- Ensuring the absence of cycles
  - Allow links to file, not subdirectories (Unix hard links)

General Graph Directory

- As in acyclic-graph structure but also permits cycles!
- Traversal becomes more complicated (avoid traversing same entries repeatedly). How about ls -R?
- Self-referencing files create difficulties with reference counts. Garbage collection may be required. (Pass one: traverse and mark; pass two: collect).

General graph directory



Protection Mechanisms

- Simple protection by access right lists
  - read
  - write
  - execute
  - append
  - delete
  - list

Access lists and groups

- Mode of access: read, write, execute
- Classes of users
  - Owner
  - Group (membership carefully controlled)
  - Public
- Unix protection bits
  - RWXRWXRWX
  - O G P
- Operations: **chmod**, **chgrp**

Access lists and groups

- General access lists are even more flexible than Unix scheme
  - Example: allow everyone but one person to read a file
- Unix: interpretation of "x" bit for directories
- Unix: suid bit