File Systems

Chapter 10

File

Basic concepts

- Blocks of data accessed by name and stored in a contiguous logical address space on a disk drive
  - Logical block address (LBA)
  - disk looks like an array
- Physical address(es) may not be contiguous or sequential
- A disk block can be written in place
- Disk drive can directly access any block (move the head and wait for the disk to rotate under the head)

File

Continued

- Types
  - Data
    - Text (ASCII)
    - Binary
  - Program (executable)
- Each file is described by metadata
  - file permissions
  - size
  - time stamps
  - data (physical location)
  - ownership / access control list (ACL)
- Structure (determined by operating system or application)
  - Lines
  - Records (fixed or variable length)
  - Formatted document

File Access

Directory / Folder

Organize files and other directories

- Contains names of files and directories
  - Names must be unique
  - May have a parent directory
  - May have zero or more children directories
- Organizes content
- Maps name to a file control block (FCB) containing metadata

File System Design

Two views

User View
- File
- File attributes
- File operations
  - Read
  - Write
  - Etc.
- Directory structure

Physical View
- Organization of directories
- File information and security
- Location of data on drive
- Management of free blocks
### Directory Organization

How to structure multiple directories

- **Linear list**
  - Simple to implement
  - Inefficient operation: $O(n)$
- **Tree**
  - Efficient operation: $O(\log n)$ search and insert; $O(n)$ delete
  - Automatic ordering/sorting
- **Hash table / linear list**
  - Efficient search: $O(1)$
  - Must deal with collisions
  - Fixed table size (rehash needed to adjust table size)

---

### Hierarchical File System

Tree structured directories (Unix/Linux example)

```
/  (root directory)
  ├── tmp
  │    └── docs
  ├── dir
  │    └── mail
  ├── bin
  │    └── canort
  │        └── dab
  │        └── bob
  │        └── fred
  ├── etc
  └── home
```

- `/` name separator and top of tree
- `.=` (dot) current directory
- `..` (dot dot) parent of current directory
- **Absolute or full path names begin with `/` (root)**
- **Relative path names begin with a directory, a file, or ...**

---

### Physical Organization

The hardware view

- Platter, cylinder, and sector map to a LBA

---

### File System vs. Drive

**Physical vs logical drive**

- 1 Drive partitioned into multiple file systems
- 1 File system spanning multiple drives

---

### Contiguous Allocation

Organizing blocks on a disk

- Each file in a set of contiguous blocks
- Simple: only start & length
- High performance
- Supports random access
- Random access: $b + i$
- Wastes space (allocation)
- Files can't grow
- External fragmentation
- Free space
- Extents can help

---

### Linked Allocation

Organizing blocks on a disk

- Simple: need only start
- No wasted space
- No random access
- Sequential access may be inefficient
  - But true for most algorithms
  - Overhead for pointers
  - Clusters can help
File Allocation Table (FAT)

Alternate linked allocation
- 1 entry / disk block
- Indexed by block number
- Contains number of next block in list
- Unused marked by a special value
- Directory contains starting block
- Better random access
- FAT is cached

Indexed Allocation

Organizing blocks on a disk
- One index block / file
- Good random access
  - i-th entry -> i-th block
- Increased pointer overhead
- Index block structure
  - linked
  - multilevel
  - combined

Defragmentation

Reorganizing blocks on a disk
- Blocks belonging to one file are reorganized to improve read time
  - Can help a computer to boot faster
  - Can make programs load faster
  - Can make data files load faster

fragmented disk:

after defragmentation: