北京邮电大学软件学院

School Of software Engineering Of BUPT

Operating Systems

Lecture 9 ： File-System Interface

Jinpengchen

Email: jpchen@bupt.edu.cn

Catalog Description

- File Concept
- Access Methods (访问方式)
- Directory Structure (目录结构)
- Protection

Chapter Objectives

- To explain the function of file systems
- To describe the interfaces to file systems
- To discuss file-system design tradeoffs, including access methods, and directory structures
- To explore file-system protection

File Concept

- OS provides a uniform logical view of information storage despite the various storage media (nonvolatile).
- A file is a logical storage unit.
  - A file is a named collection of related information that is recorded on secondary storage.
  - Types:
    - ✓ Data: numeric; character; binary
    - ✓ Program
  - In general, a file is a sequence of bits, bytes, lines, or records.
    - ✓ The meaning is defined by the file's creator and user.
  - A file has a certain defined structure, which depends on its type.
    - ✓ Example: text files, source files, object files, executable files
  - Contiguous logical address space

File Concept

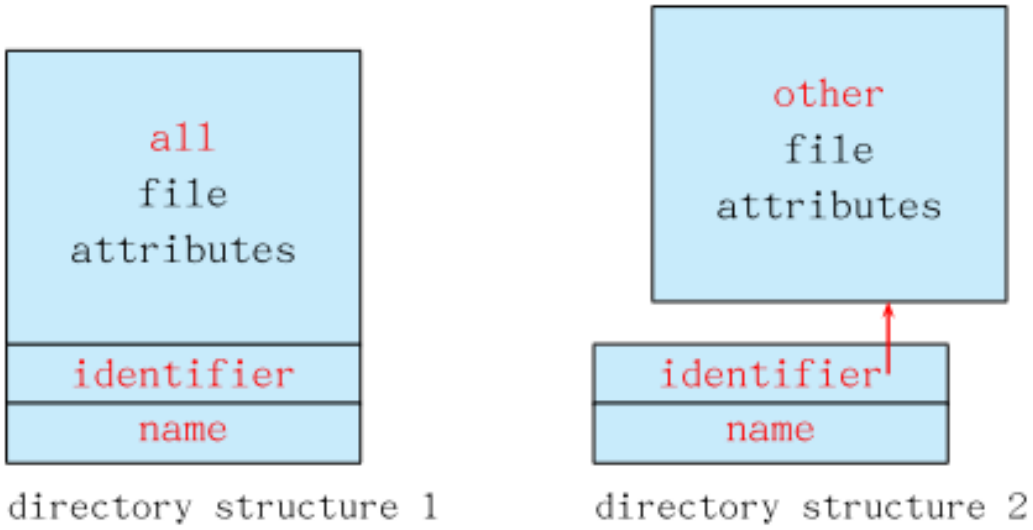
- File attributes
- File operations
- File types
- File structures
- Internal file structure

File Attributes

- **A file's attributes vary from one OS to another but typically consist of these:**
  - Name-The only information kept in human-readable form
    - ✓ A name is usually a string of characters, such as “example.c”
    - ✓ Uppercase vs. lowercase: care or not care
  - Identifier-Unique tag, usually a number, identifies file within FS
    - ✓ The non-human-readable name for the file
  - Type-- Needed for systems that support different types
  - Location-- A pointer to file location on device
  - Size-- Current file size; may also include MAX size
  - Protection-- Access-control(访问控制) information: who can do reading, writing, executing
  - Time, date, and user identification -- Data for protection, security, and usage monitoring

File Attributes

- Information about files are kept in the directory structure, which is also maintained on the secondary storage



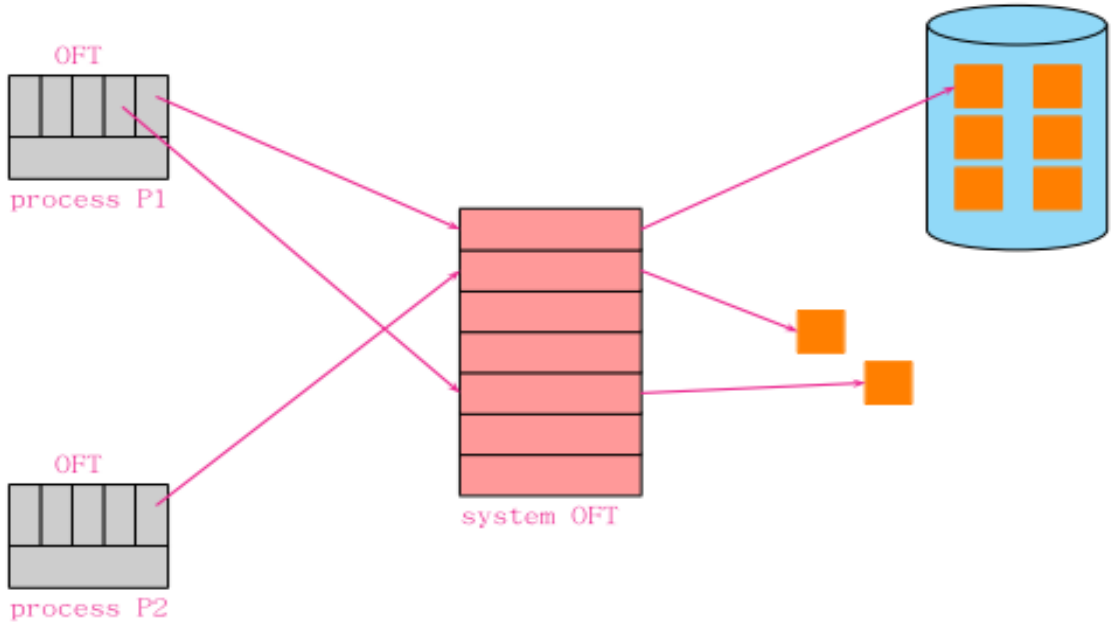
- Typically, a directory entry only consists of the file's name and its unique identifier. The identifier in turn locates the other file attributes.

File Operations(文件操作)

- **File is an abstract data type. OS provides the 6 basic system calls**
  - Create : allocate space + create an directory entry
  - Write: write pointer
  - Read: read pointer
  - Reposition within file: also known as seek
  - Delete: release space + erase the directory entry
  - Truncate: file len=0; release space; all other attributes remain unchanged
- **others:**
  - For file: append, rename
  - For file attribute: chown, chmod, . . .
  - For directory & directory entries:
    - ✓ Open(Fi)--search the directory structure on disk for entry Fi, and move the content of entry to memory
    - ✓ Close(Fi)-- move the content of entry Fi in memory to directory structure on disk

File Operations(文件操作)

- **Open Files & Open-File Table**
  - Open-file table, OFT: a small table containing information about all open files
  - Several processes may open the same file at the same time ⇒2-levels: a per-process table & a system-wide table with process-independent information



File Operations(文件操作)

- **Open Files & Open-File Table**
  - Several pieces of data are needed to manage open files:
    - ✓ File pointer: pointer to last read/write location, process-dependent
    - ✓ 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: the information needed to locate the file on disk, always is kept in memory
    - ✓ Access rights: per-process access mode information

File Structure

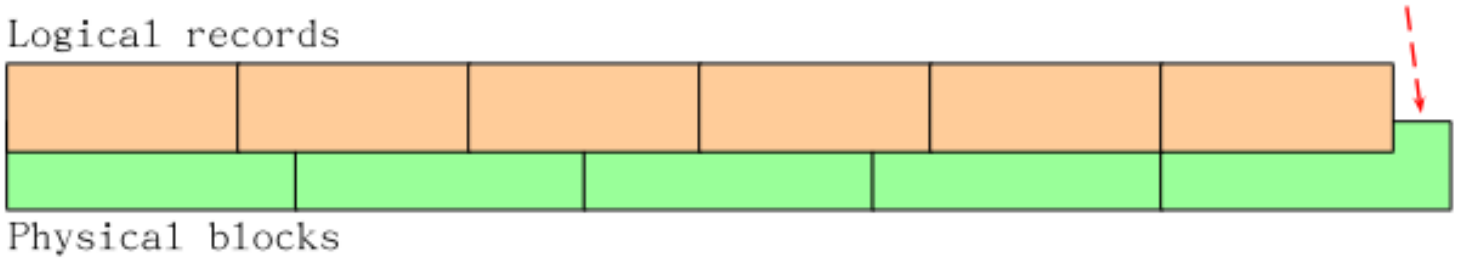
- **Open Files & Open-File Table**
  - Sometimes, file types can indicate the internal structure of file
  - File structures(文件结构)(逻辑上)
    - ✓ None - sequence of words, bytes
    - ✓ Simple record structure
      - Lines
      - Fixed length;
      - Variable length
    - ✓ Complex Structures
      - Formatted document
      - Relocatable load file
  - Can simulate last two with first method by inserting appropriate control characters

File Structure

- **System-supported file structures**
  - Most modern OSes support a minimal number of file structures directly
    - ✓ Example: UNIX sees every file as a sequence of 8-bit bytes
  - Benefits:
    - ✓ Applications have more flexibility
    - ✓ Simplifies the OS

File Structure

- **Internal file structure**
  - How to locate an offset within a file?
    - ✓ Logical file (record) (vary in length)
    - → Physical block (fixed size)
  - Solution: Packing -- packing a number of logical records into physical blocks.
    - ✓ Pack & unpack: convert between logical records and physical blocks
    - ✓ Internal fragmentation will occur



Catalog Description

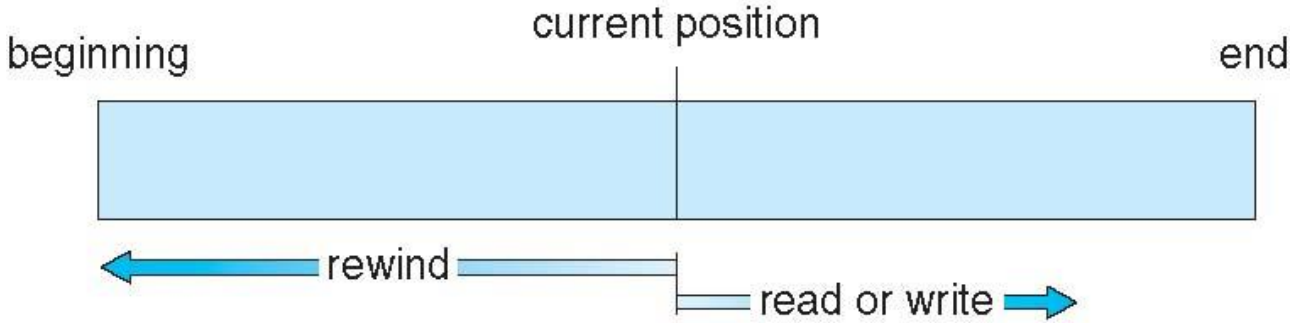
- File Concept
- Access Methods (访问方式)
- Directory Structure (目录结构)
- Protection

Access Methods(访问方式)

- **Files store information. When it is used, this information must be accessed and read into computer memory**
- **On a logical perspective of users, access a file of records**
  - Sequential Access (顺序访问方式)
  - Direct Access (直接访问方式)
  - Indexed Access (索引访问方式)

Access Methods(访问方式)

- **Sequential Access (顺序访问方式): the simplest access method. Information in the file is processed in order, one record after the other.**
  - This is a most common access mode.
    - ✓ For example: editors, compilers
  - A tape model of file
- **File operations & the effect on file pointer**
  - read/write next
  - reset
  - rewind/forward n



Direct Access（直接访问方式）

- **Direct Access (直接访问方式): Information in the file is processed in no particular order.**
  - File is made up of a numbered sequence of fixed-length logical records
    - ✓ A disk model of a file, allow random access, immediate access
    - ✓ For example: databases, or an airline-reservation system
  - Can move quickly to any record location by supplying a relative record number(n)
    - ✓ Read n & Write n,
      File pointer = L * n, 0 ⩽ n ⩽ N, where N is the last record number, L is the fixed length of each record.
    - ✓ = Position n & read/write next

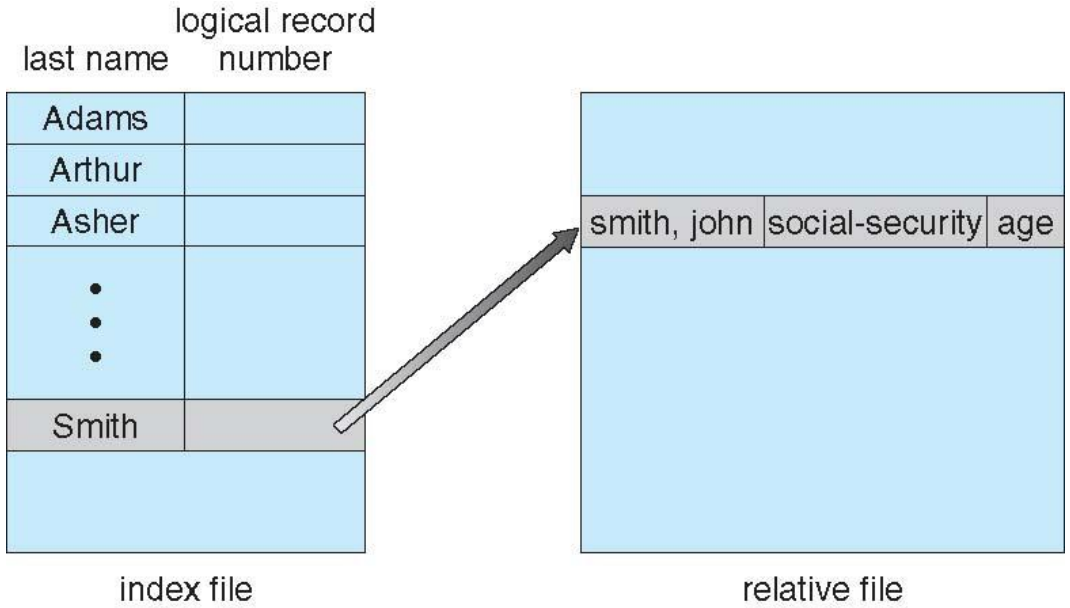
Direct Access（直接访问方式）

- **Simulation of Sequential Access on Direct-access File**

sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp* + 1;
write next	*write cp*; *cp* = *cp* + 1;

Indexed Access（索引访问方式）

- **To improves search time and reduce I/O**
  - Make an index file for the file, which contains pointers to various records
  - Search the index file first, and then use the pointer to access the file directly and to find the desired record.



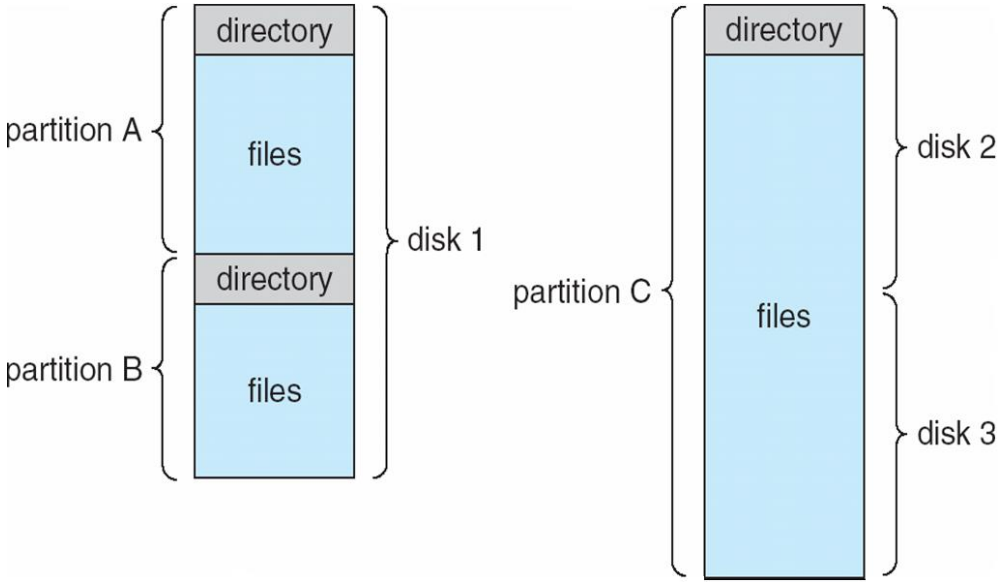
  - With large files, the index file itself may become too large to be kept in memory ⇒ Multi-level index table

Catalog Description

- File Concept
- Access Methods (访问方式)
- Directory Structure (目录结构)
- Protection

A Typical File-system Organization

- **Partition (mini-disks, volumes)**
  - One disk
  - Part of a disk: provide separate logical spaces on one disk
  - N disks: group several disks into a single logical space

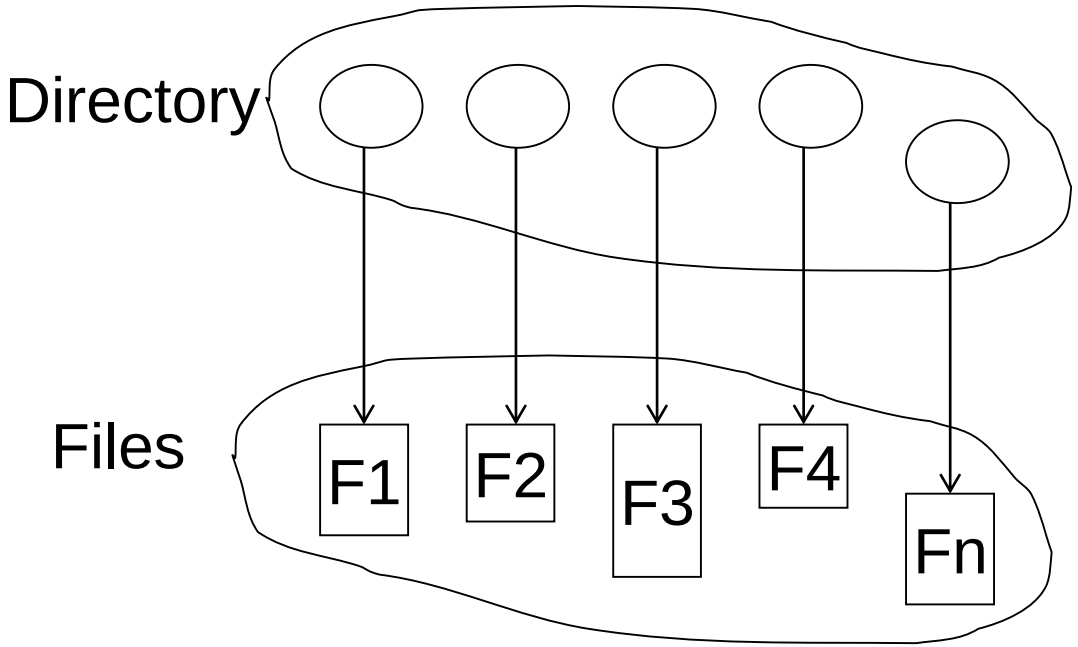


- **Partition = files + directories**
  - Directory: holds file information (name, location, size, type, ...) for all files in that partition

Directory Overview

- **Directory:**

  A collection of nodes containing information about all files



- **Directory + files: all reside on disk**
- **Backups of these two structures are kept on tapes**

Directory Overview

- Information in a directory entry

File attributes

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

In DOS

- Directory entry = FCB (file control block)
- 32 bytes each
- May cost many I/O operations to search for an entry

In UNIX

- Inode: Store most of file attributes
- Directory entry = file name + a pointer to the inode
- 16 bytes each

Directory Overview

- Operations performed on directory

Search for a file	⇒	Search in the table for an entry
Create a file	⇒	Insert an entry
Delete a file	⇒	Delete an entry
List a directory	⇒	Modify an entry
Rename a file	⇒	. . .
Traverse the file system		

Directory Overview

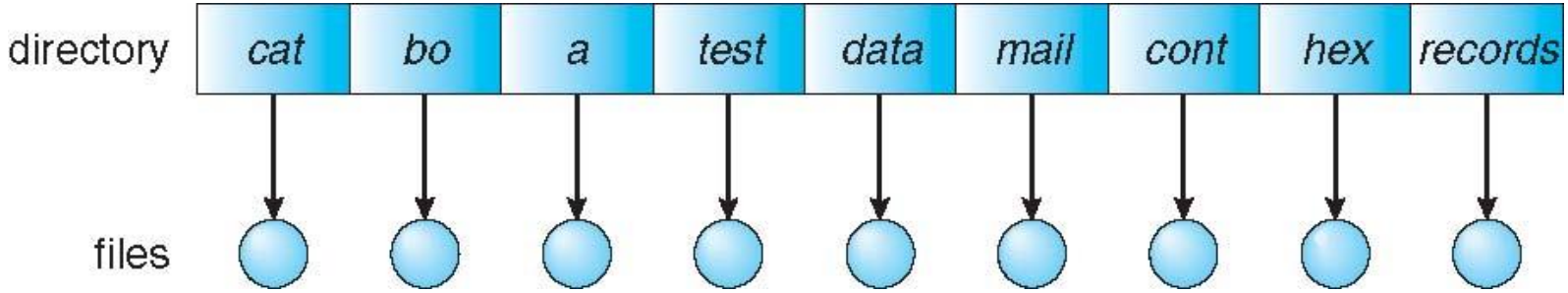
- **Organize the Directory (Logically) to Obtain**
  - Efficiency – locating a file quickly
  - Naming – convenient to users
    - ✓ Two users can have same name for different files
    - ✓ The same file can have several different names
  - Grouping – human convention
    - ✓ logical grouping of files by properties, (e.g., all Java programs, all games,…)

Directory Overview

- **Directory Structures (目录结构)**
  - Single-level directory (单层目录)
  - Two-level directory (双层目录)
  - Tree-structured directory (树型结构目录)
  - Acyclic-graph directory (无环图目录)
  - General-graph directory (通用图目录)

Directory Overview

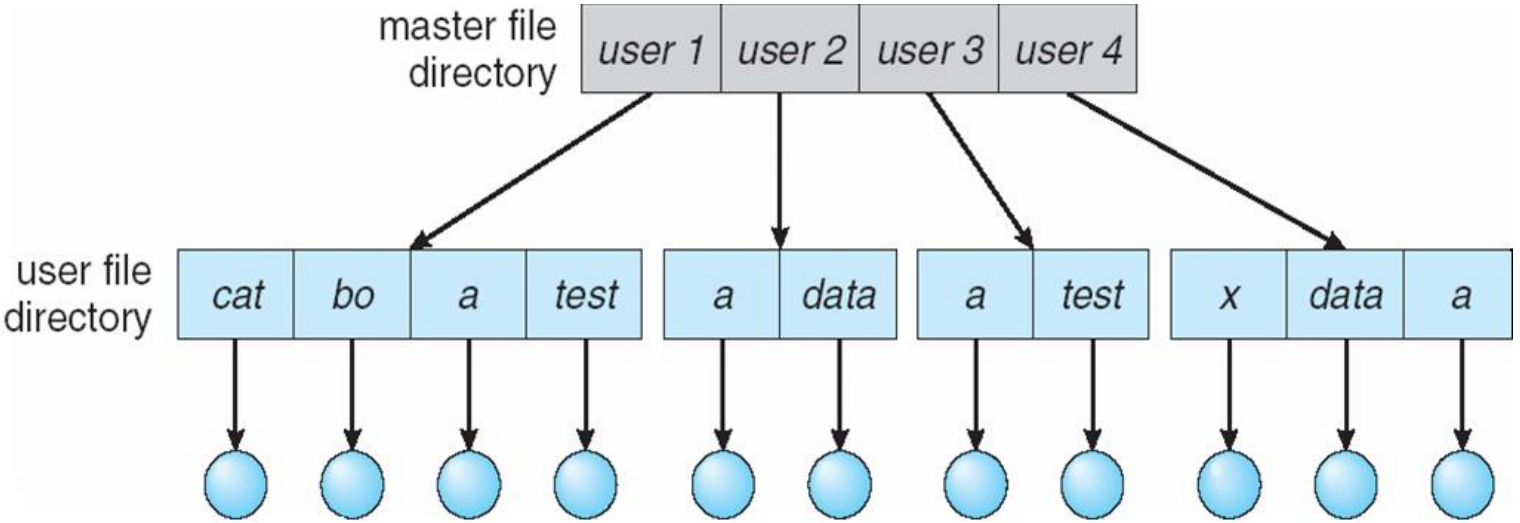
- **A single directory for all users**



- **Easy to support and understand.**
  - But if there are large numbers of files and/or users…
    - ✓ Very low searching speed, O(N)
    - ✓ Naming problem
      Small naming space & Name collision
      MS-DOS: 11 bytes for filename
      UNIX: 256 bytes
    - ✓ protection VS sharing;
    - ✓ grouping problem

Two-Level Directory(双层目录)

- **Separate directory for each user**
  - User File Directory, UFD
    - ✓ Each entry owns information for a user's file
  - Master file directory, MFD
    - ✓ Each entry contains: (1) User name, (2) A pointer to his UFD

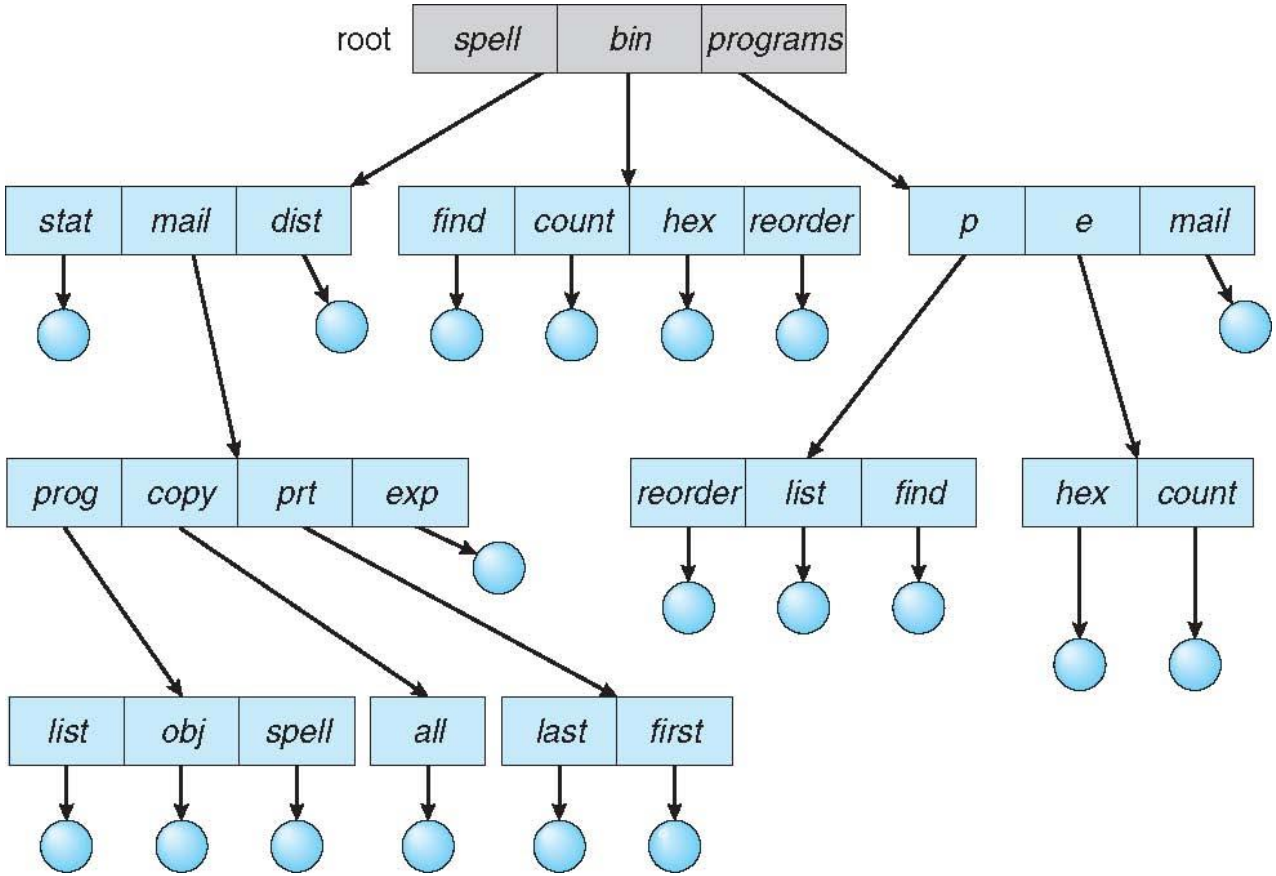


Two-Level Directory(双层目录)

- **Can have the same file name for different user**
- **Efficient searching**
- **No grouping capability**
- **Easy management**
  - Add/delete a user
- **Security VS. Sharing**
  - MFD, system administrator
  - UFD, isolated from other users
  - Directory tree & path name
  - How to share? E.g. system-wide files (data, program, …)
    - ✓ copy for each user?
    - ✓ searching path

Tree-Structured Directories(树型结构目录)

- **Root directory (根目录) & directory (目录) & subdirectory (子目录)**

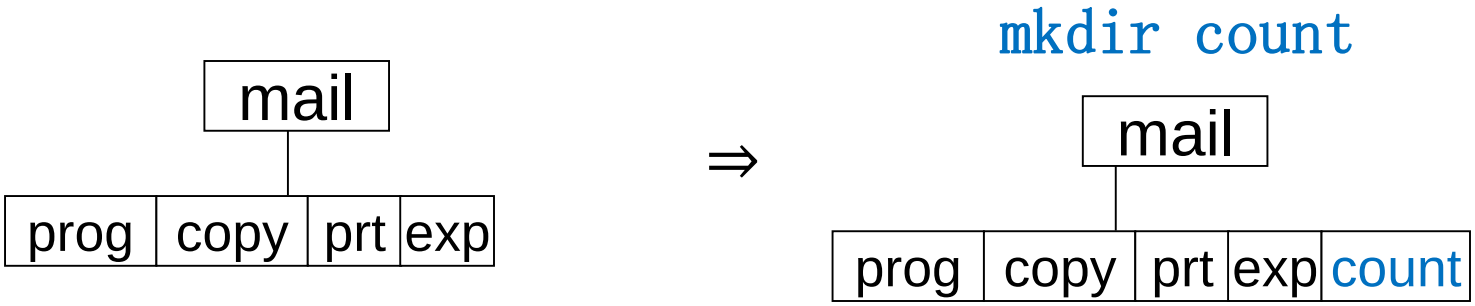


Tree-Structured Directories(树型结构目录)

- **Regular file VS. subdirectory**
  - Treat a subdirectory like another file
  - Use a special bit in the directory entry to distinguish a file (0) from a subdirectory (1)
- **Current directory (当前目录) (working/searching directory)**
  - Creating a new file is done in current directory.
  - Initial current directory
- **Absolute vs. relative path names (绝对/相对路径名)**
  - /spell/words/rade
  - ../spell/words/rade

Tree-Structured Directories(树型结构目录)

- **Operations**
  - Change current directory: cd /spell/mail/prog
  - Delete a file: rm <file-name>
  - List a dictory: ls
  - Create a new directory: mkdir <dir-name>
    - ✓ Example: if in current directory /mail



  - Delete a directory
    - ✓ MS-DOS (only empty directory) VS. UNIX (optional)

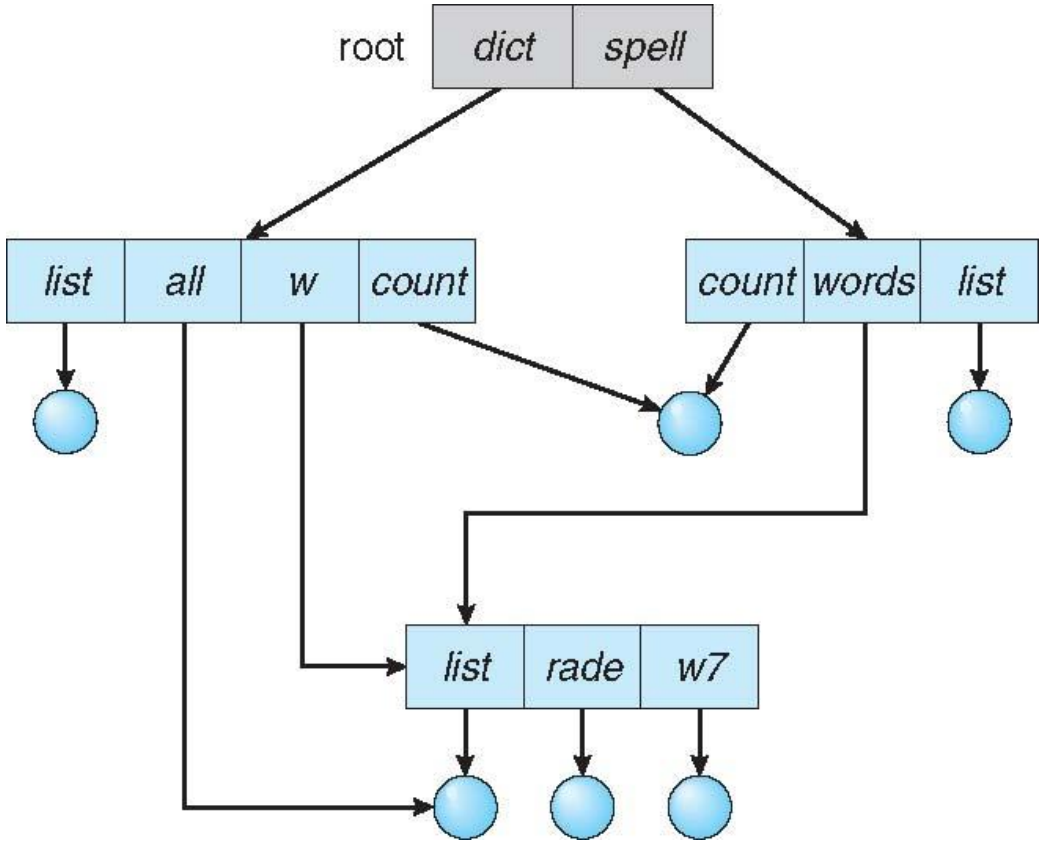
Tree-Structured Directories(树型结构目录)

- **Efficient searching**
- **Grouping Capability**
- **The tree structure**

  prohibits（阻止）the sharing of files and directories.

Acyclic-Graph Directories (无环图目录)

- **Have shared subdirectories and files, with no cycles**
- **The same file or directory may be in two different directories, having two different names (aliasing)**



Acyclic-Graph Directories (无环图目录)

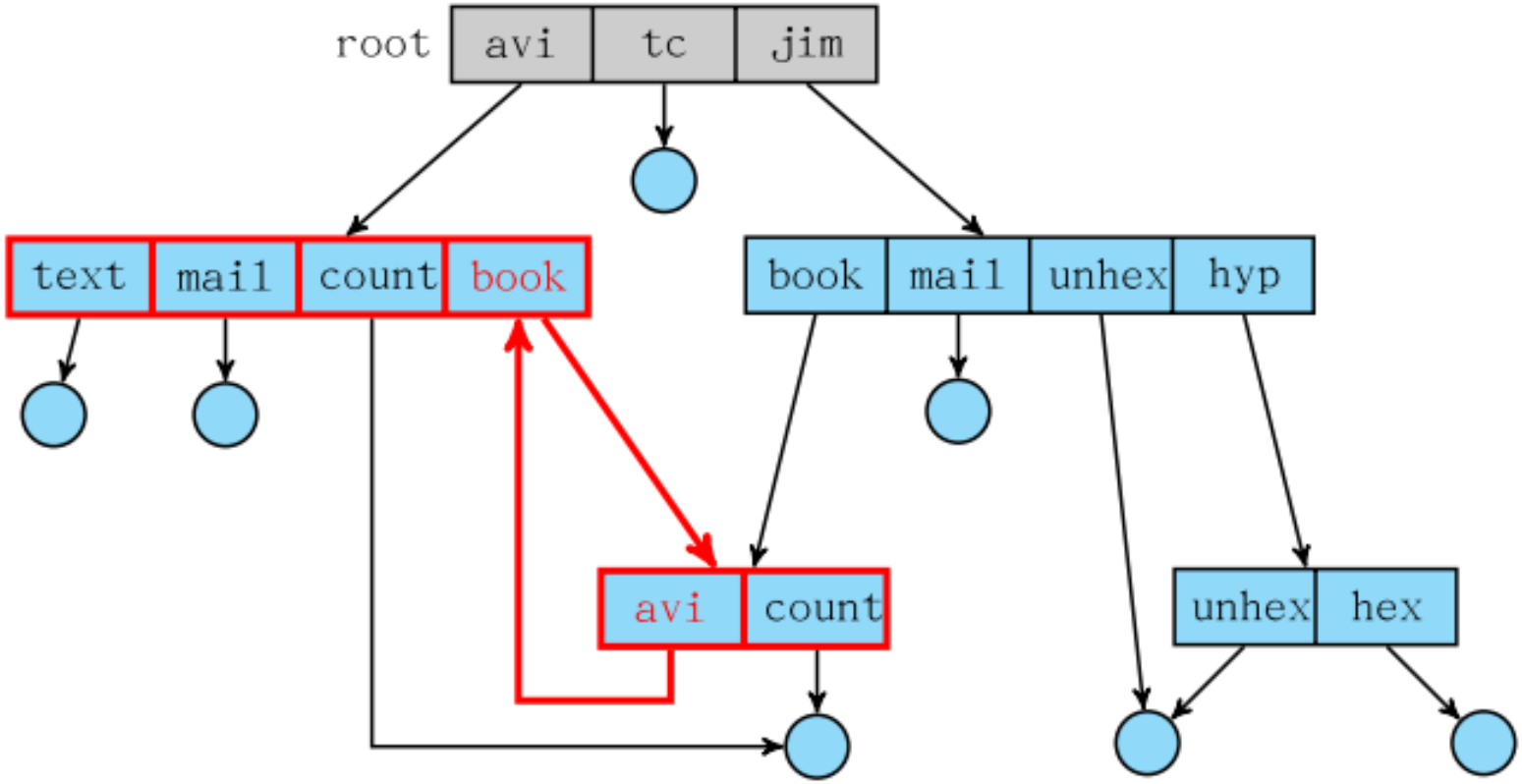
- **Implementation**
  - Symbolic links (符号链接)
    - ✓ A special new directory entry (link)
    - ✓ The content of such file is the path name of the real file/directory
  - Duplicates directory entries
    - ✓ Hard to maintain consistency

Acyclic-Graph Directories (无环图目录)

- **Traversing problem**
  - Different names, actual only one file
  - traverse more than once
- **Deleting problem**
  - If direct deletes list ⇒ dangling pointer
  - or preserve the file until all reference to it are deleted
  - Solutions:
    - ✓ File-reference list
    - ✓ Reference count: hard link (硬链接) in UNIX
  - How to ensure there are no cycles?

General Graph Directory (通用图目录)

- **If we allow cycles existed in directory**



General Graph Directory (通用图目录)

- **The traversing problem and deleting problem still exists, even more complicatedly**
  - Infinite loop
    - ✓ limit the access number of a directory while for a search
  - Garbage & garbage collection
- **How do we guarantee no cycles?**
  - Allow only links to file not subdirectories
  - Every time a new link is added use a cycle detection algorithm to determine whether it is OK

Catalog Description

- File Concept
- Access Methods (访问方式)
- Directory Structure (目录结构)
- File sharing (文件共享)
- Protection

Protection

- **Reliability (可靠性)**
  - Guarding against physical damage
  - File systems can be damaged by
    - ✓ Hardware problems, power surges or failures, head crashed, dirt, temperature extremes, or Vandalism
  - Generally provided by duplicate copies of files (disk→tape,…)
- **Protection (保护，安全性)**
  - Guarding against improper access

Protection in multi-user system

- **The need to protect files is a direct result of the ability to access files (of other users).**
  - Complete protection with prohibiting access
  - Free access with no protection
  - Controlled access. √
- **Controlled access: limiting the types of file access that can be made**
  - Types of access: Read/Write/Execute/Append/Delete/ List
  - Higher-level functions may also be controlled: rename/copy/edit/. . .
- **File owner/creator should be able to control:**
  - what can be done? by whom?
- **Many protection mechanisms have been proposed.**

Access control(访问控制)

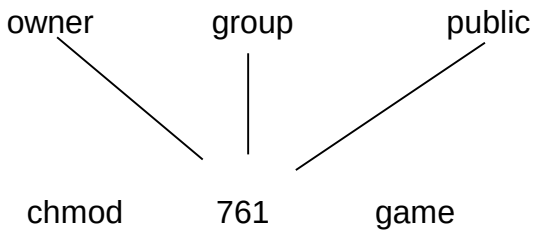
- The most common approach to the protection problem: ID-dependent access
  - Make access dependent on the ID of the user
- The most general scheme to implement ID-dependent access: Access control list (访问控制列表, ACL)
  - Associate with each file and directory an access list.
    - ✓ Access list specifies for each listed (allowed) user name and the types of (allowed) access allowed.
    - ✓ Stored in each directory entry
- **Length problem**

Solution: Three classes of users

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

Access control(访问控制)

- **About group:**
  - 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.



  - Attach a group to a file

    chgrp G game