

Chapter 11: File-System Interface



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

File Concept



- Contiguous logical address space
- Types:
 - Data
 - numeric
 - character
 - binary
 - Program

File Structure



- 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.
- Who decides:
 - Operating system
 - Program

File Attributes



- **Name** – only information kept in human-readable form.
- **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.
- Information about files are kept in the directory structure, which is maintained on the disk.

File Operations



- Create
- Write
- Read
- Reposition within file – file seek
- Delete
- Truncate
- Open(F_i) – search the directory structure on disk for entry F_i , and move the content of entry to memory.
- Close (F_i) – move the content of entry F_i in memory to directory structure on disk.

File Types – Name, Extension



| file type | usual extension | function |
|----------------|-----------------------------------|--|
| executable | exe, com, bin or none | read to run machine- language program |
| object | obj, o | compiled, machine language, not linked |
| source code | c, cc, java, pas, asm, a | source code in various languages |
| batch | bat, sh | commands to the command interpreter |
| text | txt, doc | textual data, documents |
| word processor | wp, tex, rrf, doc | various word-processor formats |
| library | lib, a, so, dll, mpeg, mov, rm | libraries of routines for programmers |
| print or view | arc, zip, tar | ASCII or binary file in a format for printing or viewing |
| archive | arc, zip, tar | related files grouped into one file, sometimes com- pressed, for archiving or storage |
| multimedia | mpeg, mov, rm | binary file containing audio or A/V information |

Access Methods



- **Sequential Access**

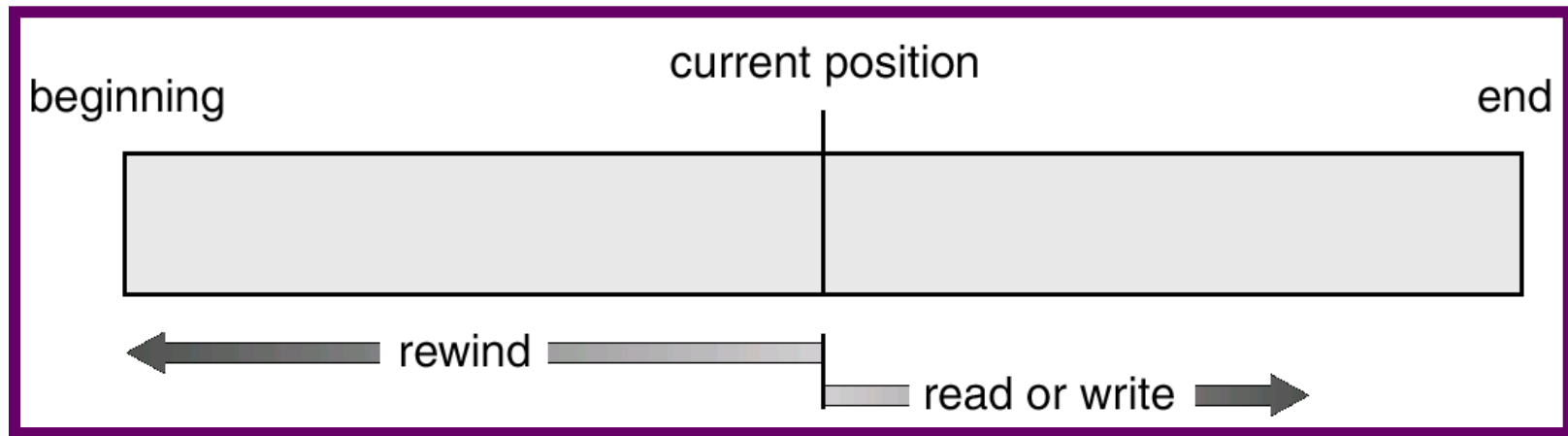
read next
write next
reset
no read after last write
(rewrite)

- **Direct Access**

read n
write n
position to n
read next
write next
rewrite n

n = relative block number

Sequential-access File

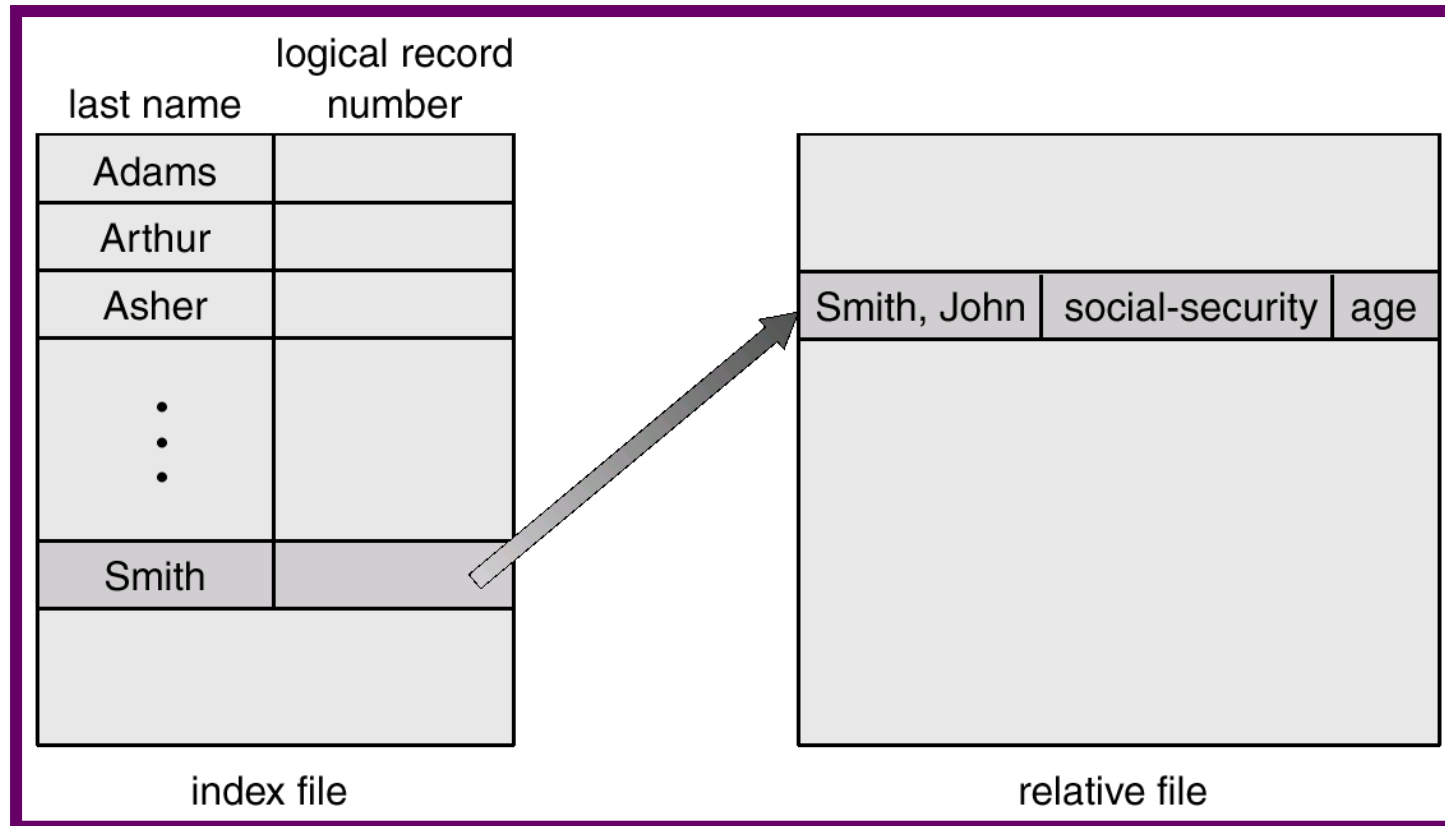


Simulation of Sequential Access on a Direct-access File



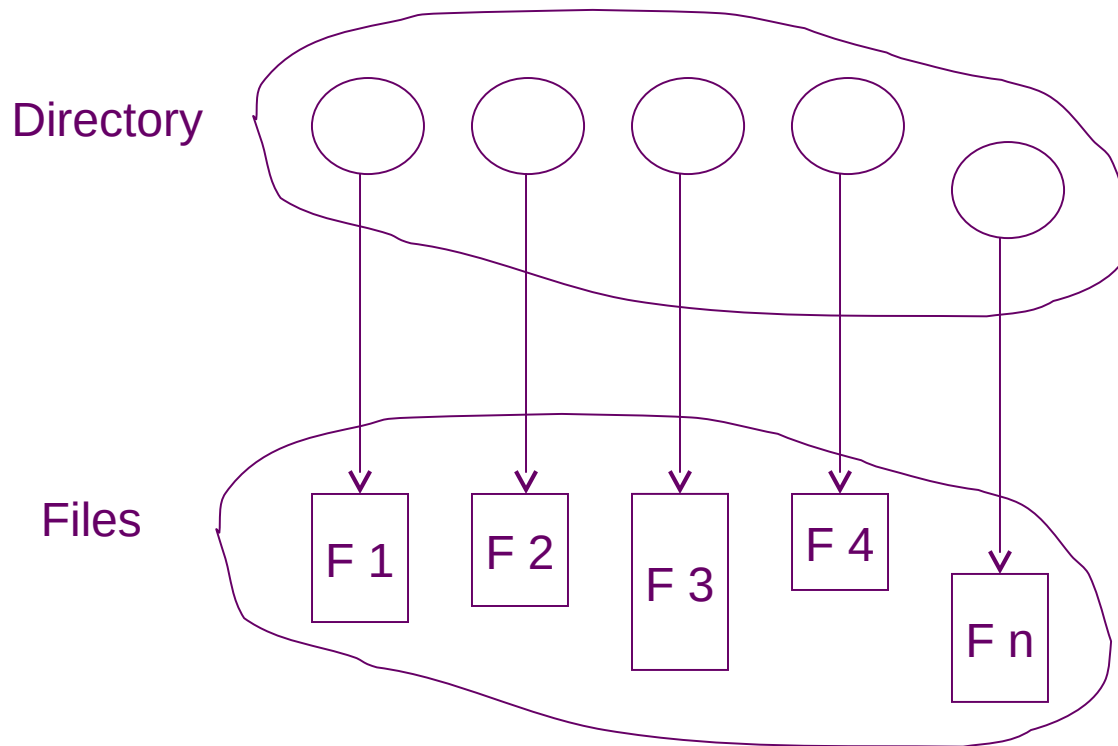
| sequential access | implementation for direct access |
|-------------------|---------------------------------------|
| <i>reset</i> | <i>cp = 0;</i> |
| <i>read next</i> | <i>read cp;</i> <i>cp = cp+1;</i> |
| <i>write next</i> | <i>write cp;</i> <i>cp = cp+1;</i> |

Example of Index and Relative Files



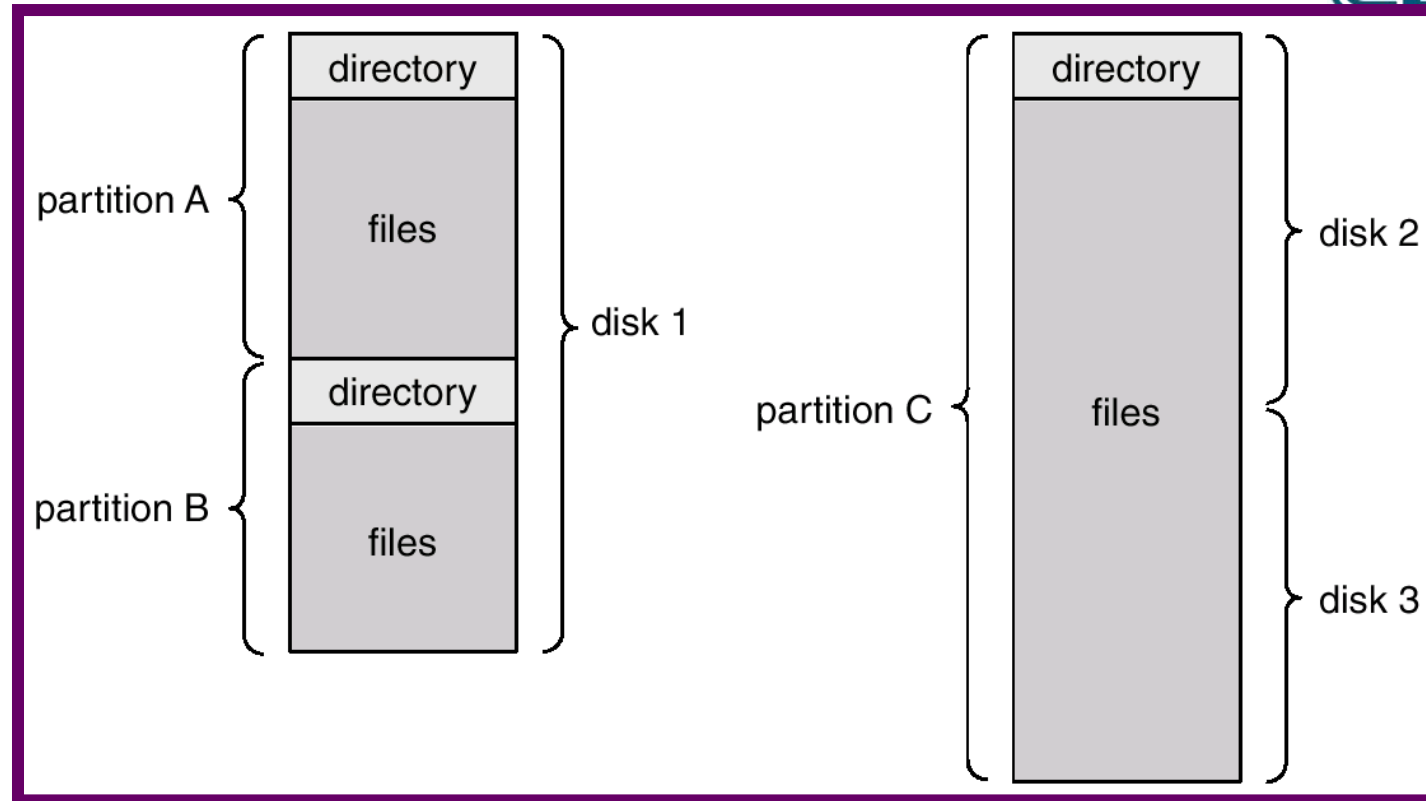
Directory Structure

- A collection of nodes containing information about all files.



Both the directory structure and the files reside on disk.
Backups of these two structures are kept on tapes.

A Typical File-system Organization



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)

Operations Performed on Directory



- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Organize the Directory (Logically) to Obtain

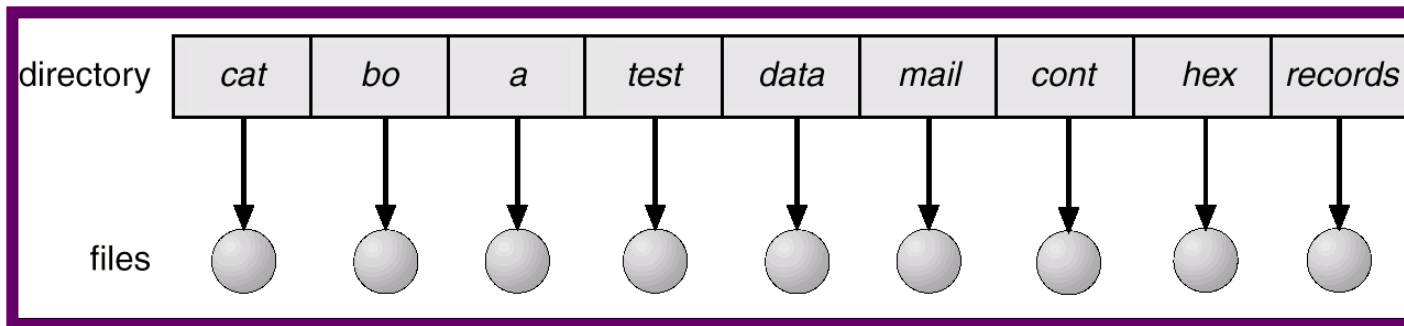


- **Efficiency** – locating a file quickly.
- **Naming** – convenient to users.
 - Two users can have same name for different files.
 - The same file can have several different names.
- **Grouping** – logical grouping of files by properties, (e.g., all Java programs, all games, ...)

Single-Level Directory



- A single directory for all users.

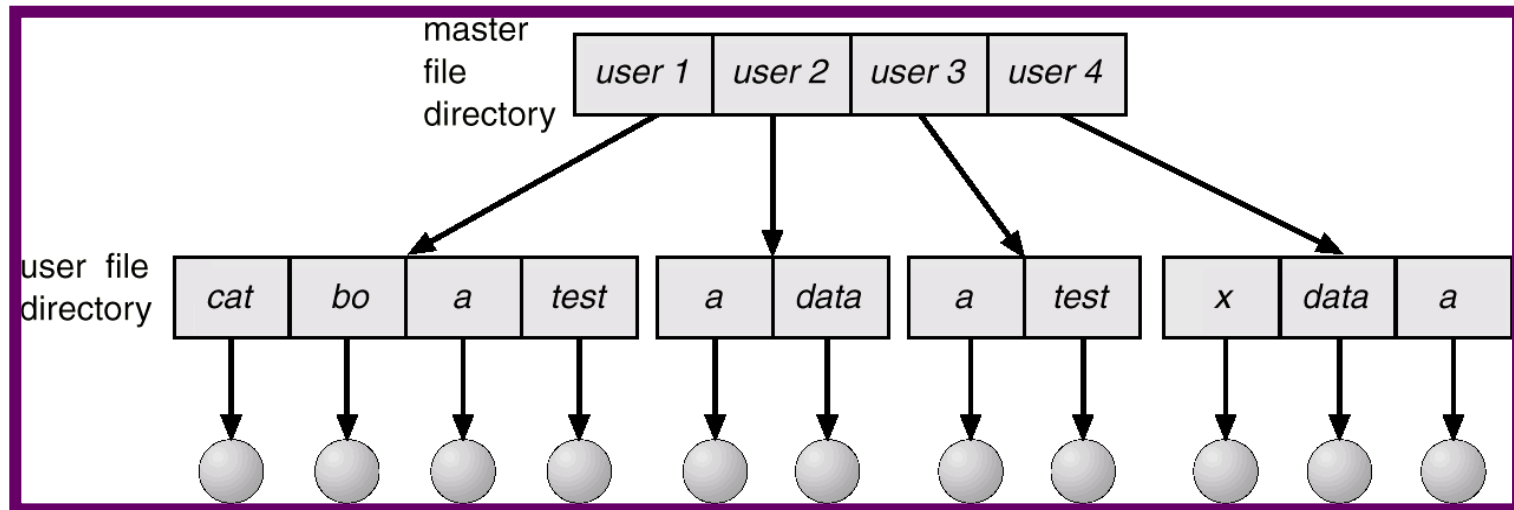


Naming problem

Grouping problem

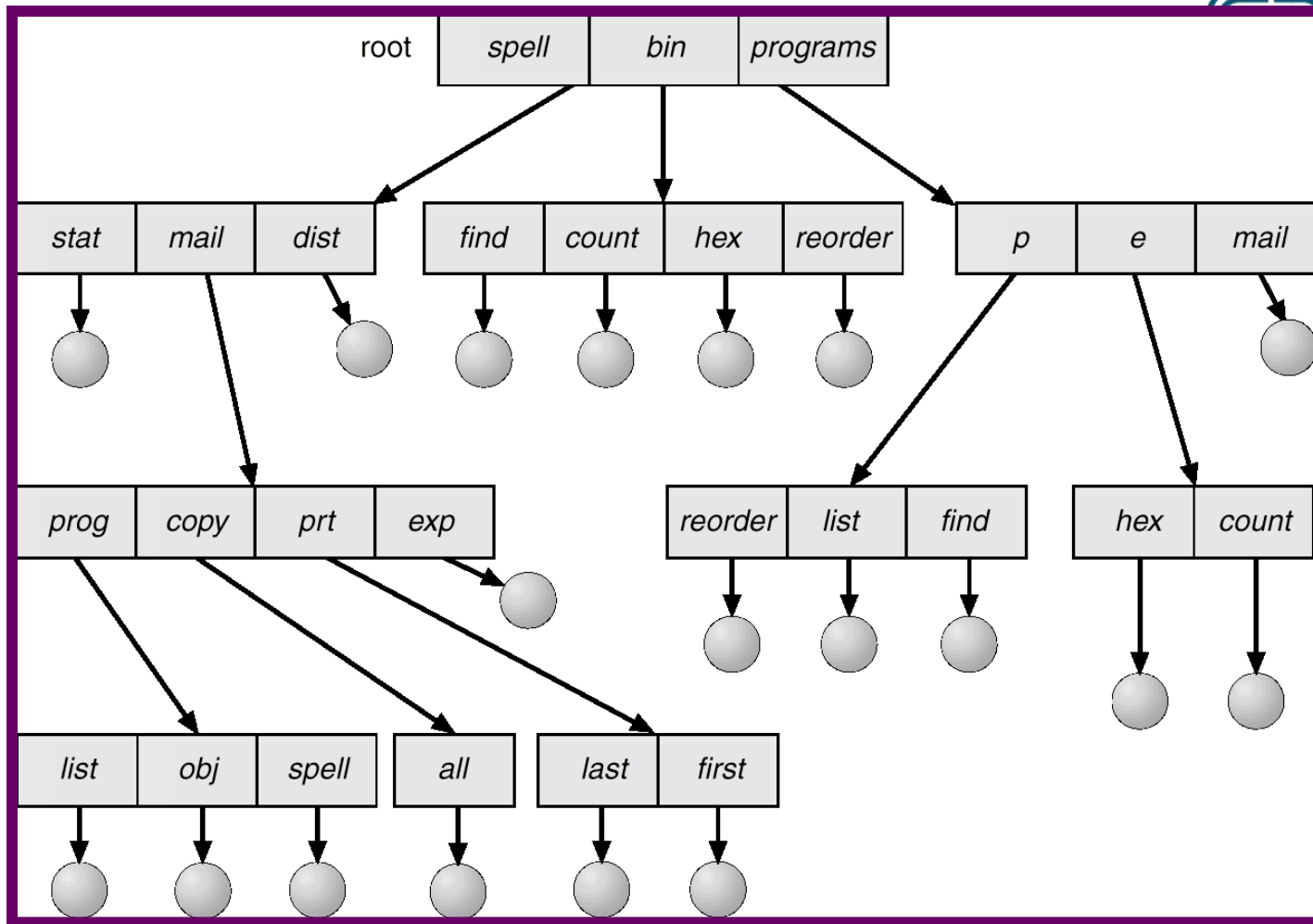
Two-Level Directory

- Separate directory for each user.



- Path name
- Can have the same file name for different user
- Efficient searching
- No grouping capability

Tree-Structured Directories



Tree-Structured Directories (Cont.)



- Efficient searching
- Grouping Capability
- Current directory (working directory)
 - **cd** /spell/mail/prog
 - **type** list

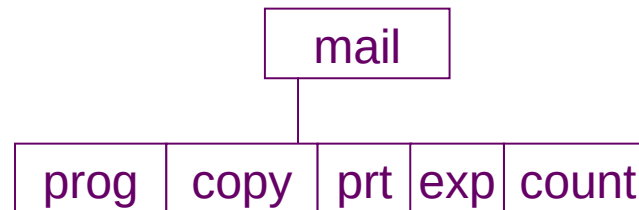
Tree-Structured Directories (Cont.)



- **Absolute** or **relative** path name
- Creating a new file is done in current directory.
- Delete a file
rm <file-name>
- Creating a new subdirectory is done in current directory.
mkdir <dir-name>

Example: if in current directory **/mail**

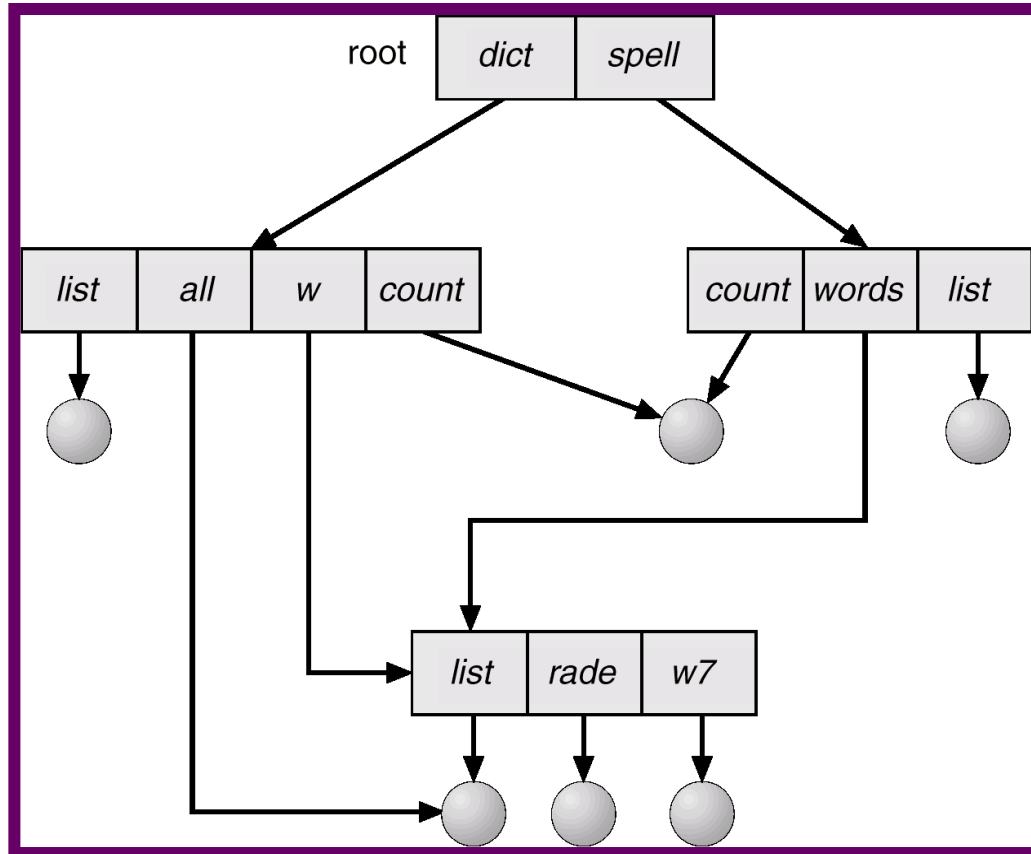
mkdir count



Deleting “mail” ⇒ deleting the entire subtree rooted by “mail”.

Acyclic-Graph Directories

- Have shared subdirectories and files.



Acyclic-Graph Directories (Cont.)

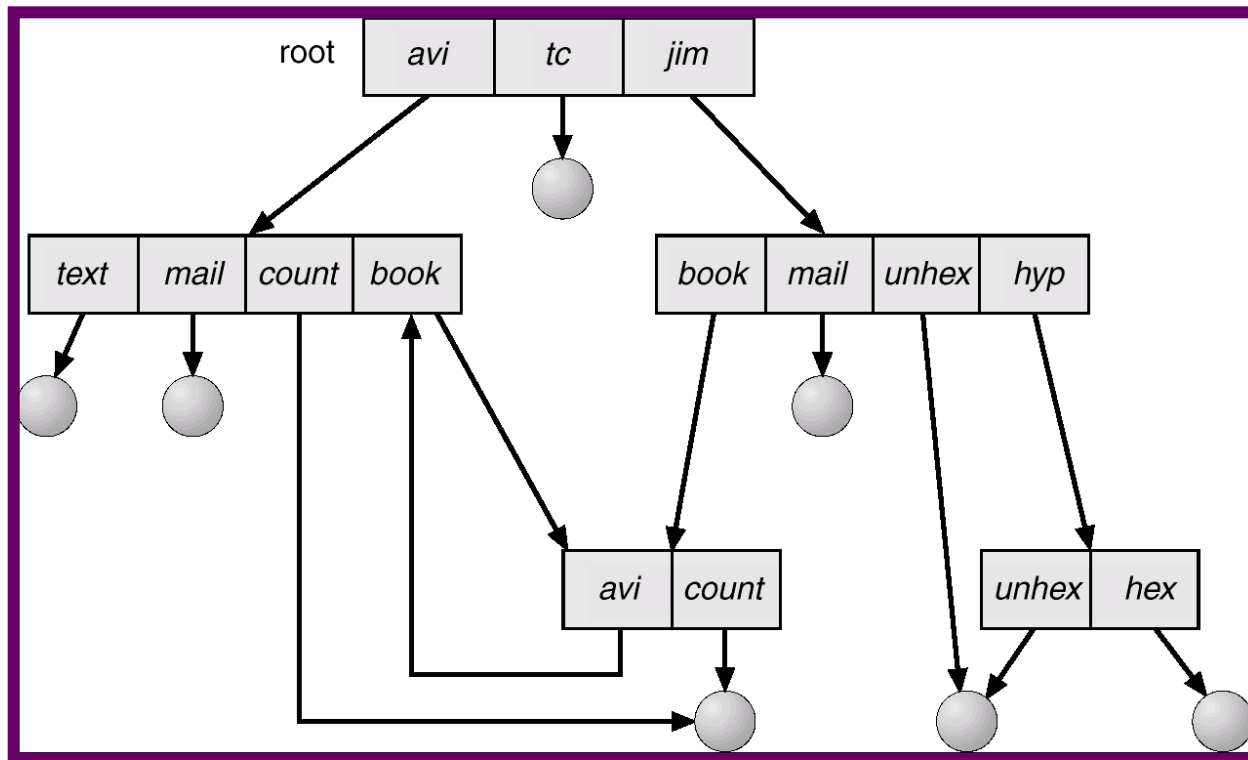


- Two different names (aliasing)
- If *dict* deletes *list* \Rightarrow dangling pointer.

Solutions:

- Backpointers, so we can delete all pointers. Variable size records a problem.
- Backpointers using a daisy chain organization.
- Entry-hold-count solution.

General Graph Directory



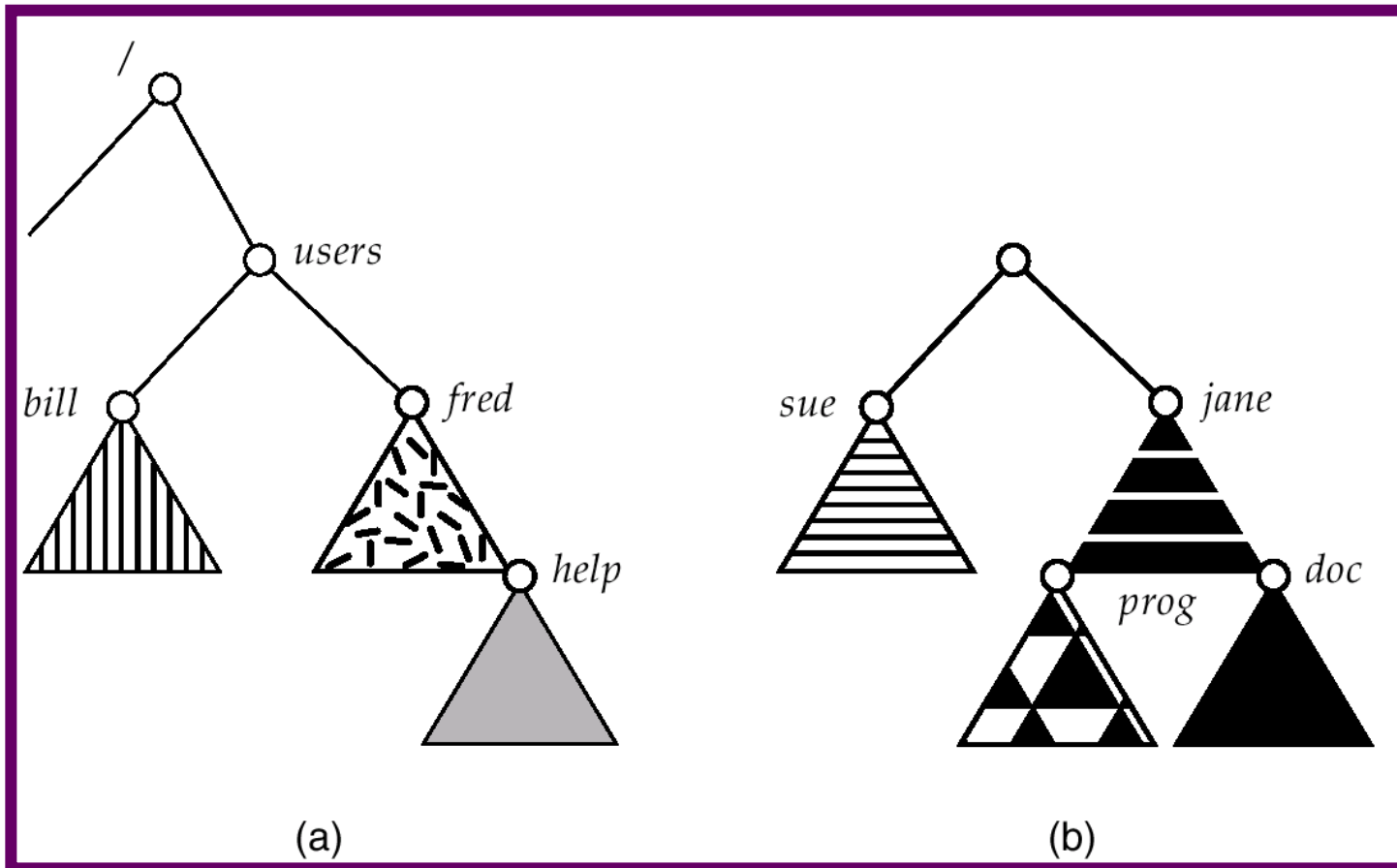
- How do we guarantee no cycles?
 - Allow only links to file not subdirectories.
 - Garbage collection.
 - Every time a new link is added use a cycle detection algorithm to determine whether it is OK.

File System Mounting

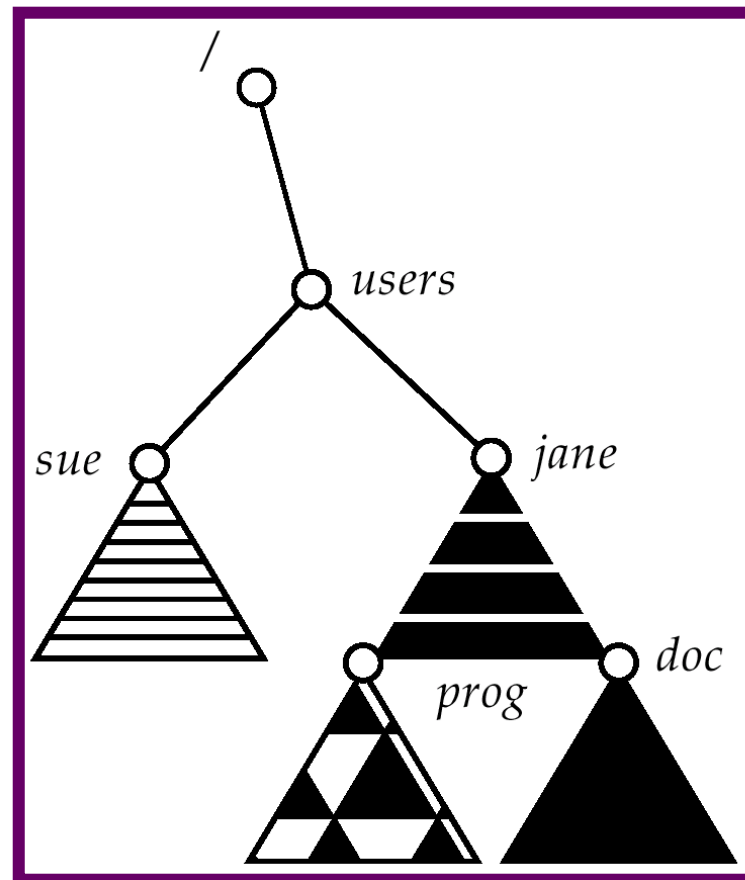


- A file system must be **mounted** before it can be accessed.
- A unmounted file system (I.e. Fig. 11-11(b)) is mounted at a **mount point**.

(a) Existing. (b) Unmounted Partition



Mount Point



File Sharing



- Sharing of files on multi-user systems is desirable.
- Sharing may be done through a *protection* scheme.
- On distributed systems, files may be shared across a network.
- Network File System (NFS) is a common distributed file-sharing method.

Protection



- File owner/creator should be able to control:
 - what can be done
 - by whom
- Types of access
 - Read
 - Write
 - Execute
 - Append
 - Delete
 - List

Access Lists and Groups



- Mode of access: read, write, execute
- Three classes of users

RWX

a) **owner access** 7 \Rightarrow 1 1 1

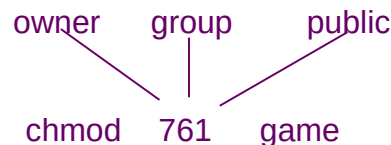
RWX

b) **group access** 6 \Rightarrow 1 1 0

RWX

c) **public access** 1 \Rightarrow 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.



Attach a group to a file

chgrp G game