FILE-SYSTEM INTERFACE

S. Lakshmi Priya, AP/CSE, SSNCE

Topics to be discussed

- Basic file concepts
  - Types
  - Structure
  - Internal Structure
- Access Methods
  - Sequential Access
  - Direct Access
- Directory Structure
- File system mounting
- File Sharing
- Protection

Basic File Concepts

1.File Attributes & operations
2.File Types
3.File Structure
4.Internal structure

File Concept

- A file is a named collection of related information that is recorded on secondary storage.
- Contiguous logical address space
- Types:
  - Programs
    - Source and object forms
  - Data
    - numeric, character, alphanumeric, binary
- Many different types of information may be stored in a file – source programs, object pgms, images, sound recordings etc.,

Basic concepts contd.,

- File has a certain defined structure according to its type.
  - Text file – sequence of characters organized into lines
  - Source file – sequence of subroutines and functions.
  - Object file - sequence of bytes organized into blocks understandable by linker.
  - executable file – series of code sections that the loader can bring into memory and execute.

1.File Attributes

- **Name** – only information kept in human-readable form.
- **Identifier** – unique tag, usually a number, identifies the file within the 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.
- Information about files are kept in the directory structure, which is maintained on the disk.

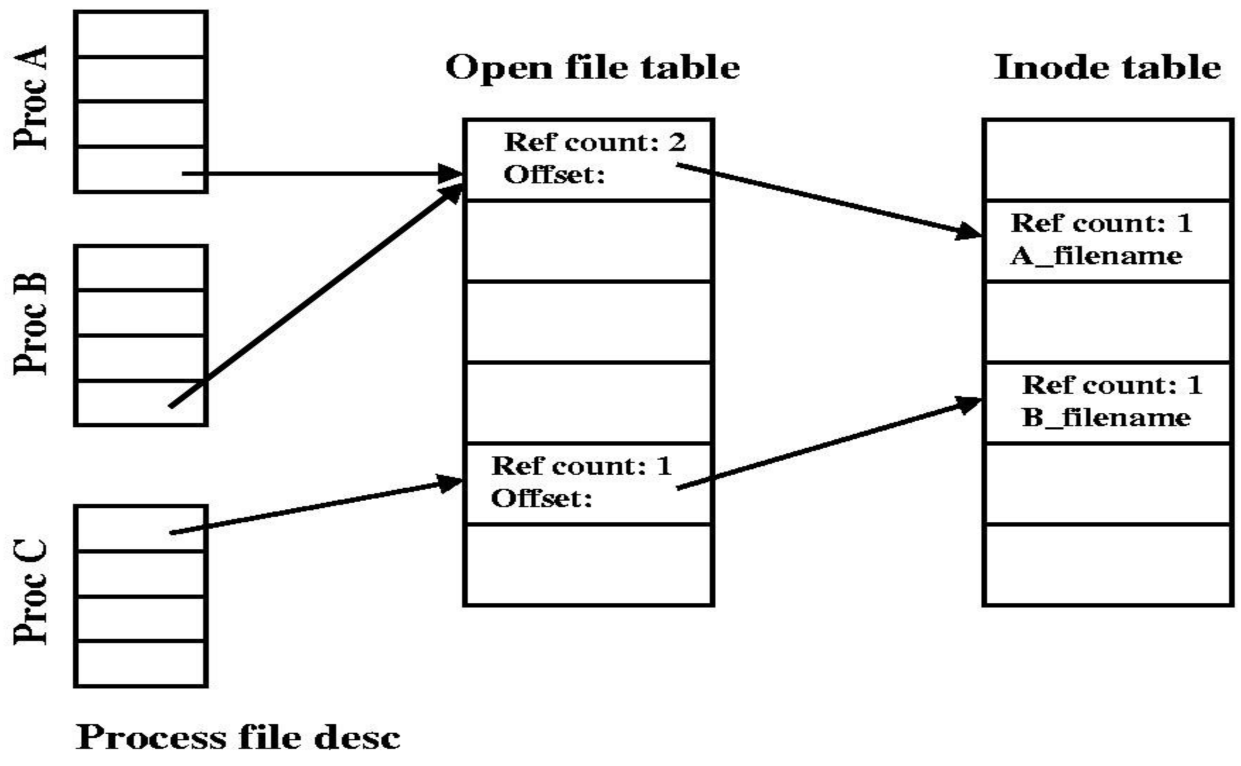
1.File Operations

- Create – find space & make a new dir entry
- Write – find entry & write data starting from write pointer
- Read - find entry & read data starting from read pointer
- Reposition within file – file seek – update the current-file-position.
- Delete – delete the dir entry
- Truncate – keep the name & attributes but erase the contents.
- 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.

Data Structures

- Open file table
  - List of all open files.
  - Open system call will return a pointer to the entry in the open-file-table.
- In a multiuser system like UNIX, several users may open the same file at the same time.
- Two level of internal tables :
  - Per process table
    - Tracks all the files that a process has open – info regarding the use of the file by the process.
    - Maintains current file pointer, access info and accounting info.
  - System –wide table
    - Each entry in per process table points to this table.
    - Contains process independent info – location of file on disk, access dates etc.,

Data Structures

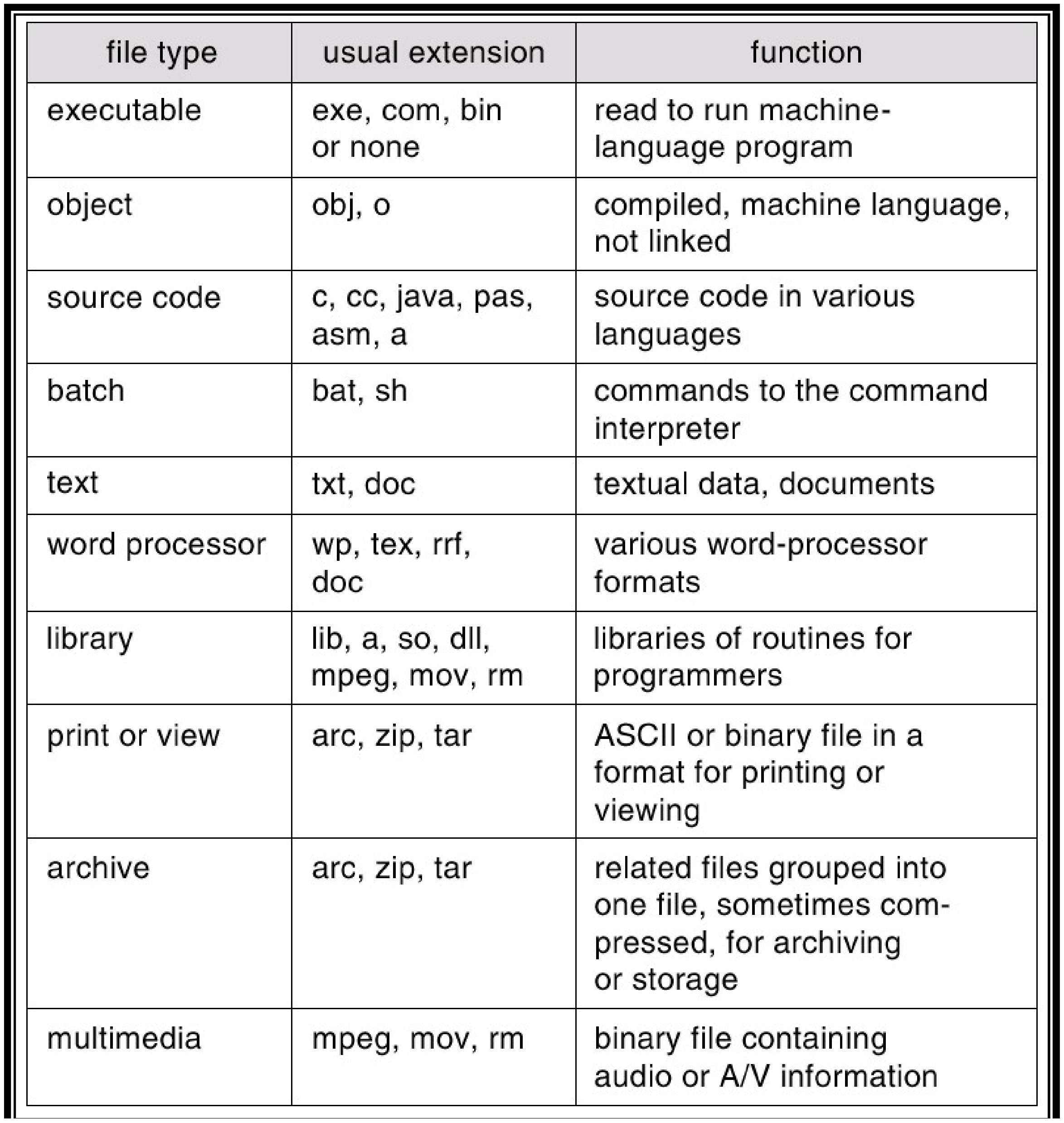


- If a process opens a file that is being used by another process, then an entry is made in the process file table and the count is incremented by one in the corresponding entry of system-wide table.

Data structures

- Several pieces of info are associated with an open file.
  - File pointer
  - File open count
  - Disk location of the file
  - Access rights.

2.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 compressed, for archiving or storage
multimedia	mpeg, mov, rm	binary file containing audio or A/V information

File Types

- Name.extension
- Tops-20 OS:
  - The source code after modification is automatically recompiled.
  - The OS distinguishes the source file from object file
- Apple Macintosh
  - Each file has a creator attribute containing the name of the program that created it
  - Eg: word file – creator – Word processor
  - When the file icon is double clicked, it automatically opens in the word processor.
- UNIX
  - Has a magic number stored at the beginning of the file to indicate roughly the type of the file.

3.File Structure

- File types are also used to indicate the internal structure of the file.
- Eg: exe file must a specific structure to determine where in memory to load the file and what the location of the first instruction is.
- Diasdv of OS supporting multiple file structures:
  - Resulting size of OS is larger.
  - Eg: if 5 diff structures, then every file has to be defined in any one of the structures.
- Some file systems adopt minimum number of structures
  - Eg: UNIX, MS-DOS and others

File Structure

- UNIX
  - considers each file to be a sequence of 8 bit bytes
  - Max flexibility for OS but little support
  - Every application program must include its own code to interpret an input file into the appropriate structure.
- MAC
  - Expects files to contain two parts: a resource fork and a data fork
  - Resource fork : info of interest to user
  - Eg: Labels of any buttons displayed by the program. Another user can modify it.
  - Data fork: program code or data- traditional file contents.

Too few structures make programming inconvenient, whereas too many cause operating system bloat and programmer confusion.

4.Internal File Structure

- Locating an offset within a file is complicated.
- All disk I/O is done in units of one block(physical record).
- Physical record size may vary from the logical record size.
- Packing
  - Accommodate the logical records into physical blocks.
- UNIX:
  - All files are a stream of bytes.
  - Logical record is one byte.
  - OS packs or unpacks bytes into physical disk blocks.
  - Causes internal fragmentation.

Access methods

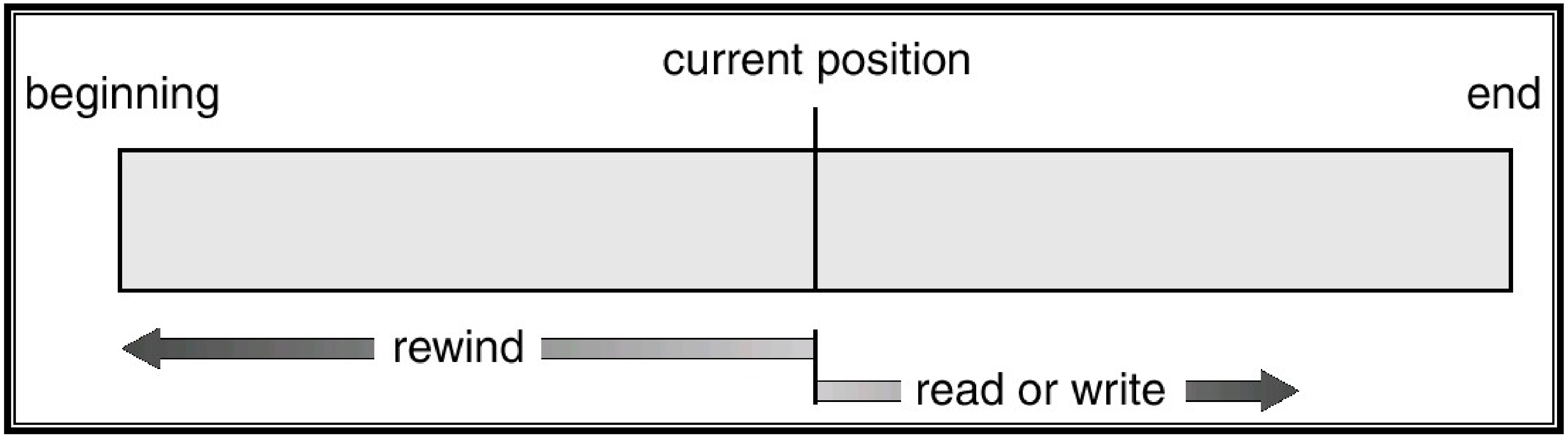
1.Sequential Access

2.Direct Access

3. Other Access Methods

1.Sequential Access

- Information in file is processed in order, one record after the other.
  - Read pointer – *read next*
  - Write pointer – *write next*
  - Reset to any position on file – move forward or backward n number of records



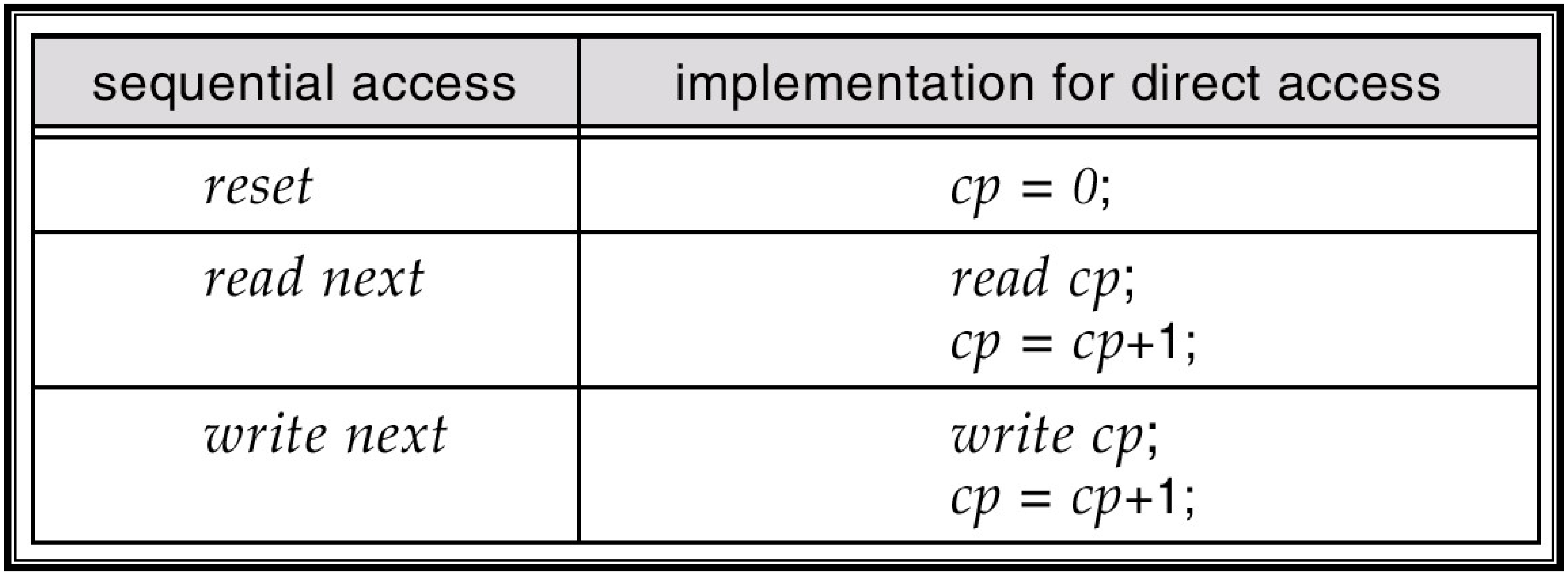
2.Direct Access or Relative Access

- Based on disk model of a file since disks allow random access to any file block.
- File is viewed as a numbered sequence of blocks or records.
- No restrictions on the order of reading and writing
- All file operations must include the block number as a parameter.
  - read n as opposed to read next
  - write n as opposed to write next

    OR

  - position to n
    - read next
    - write next
- Where n is the relative block number – an index relative to the beginning of the file. (starts at 0 or 1)

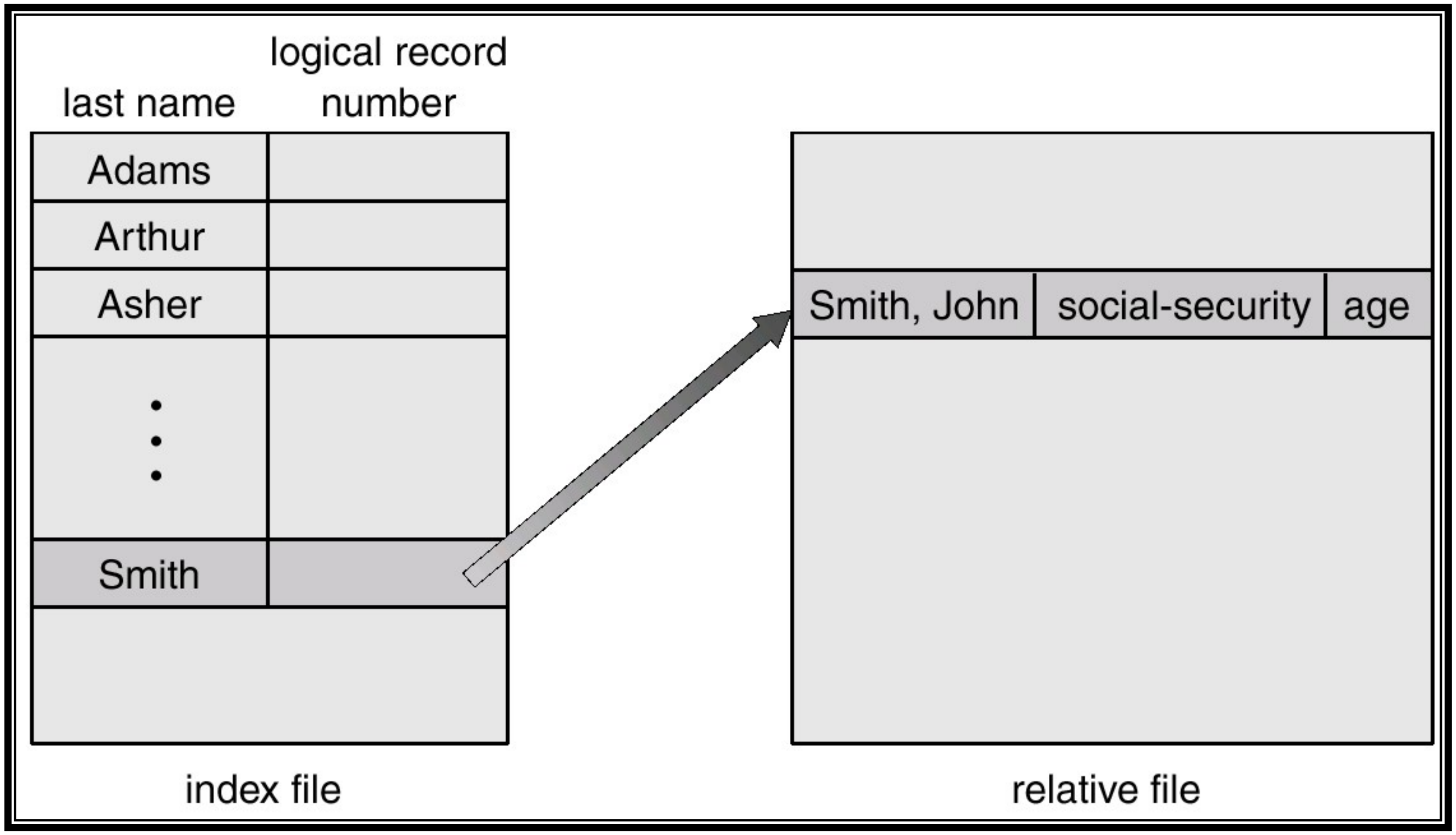
Simulation of Sequential Access on a Direct-access File



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

3. Other Methods - Example of Index and Relative Files

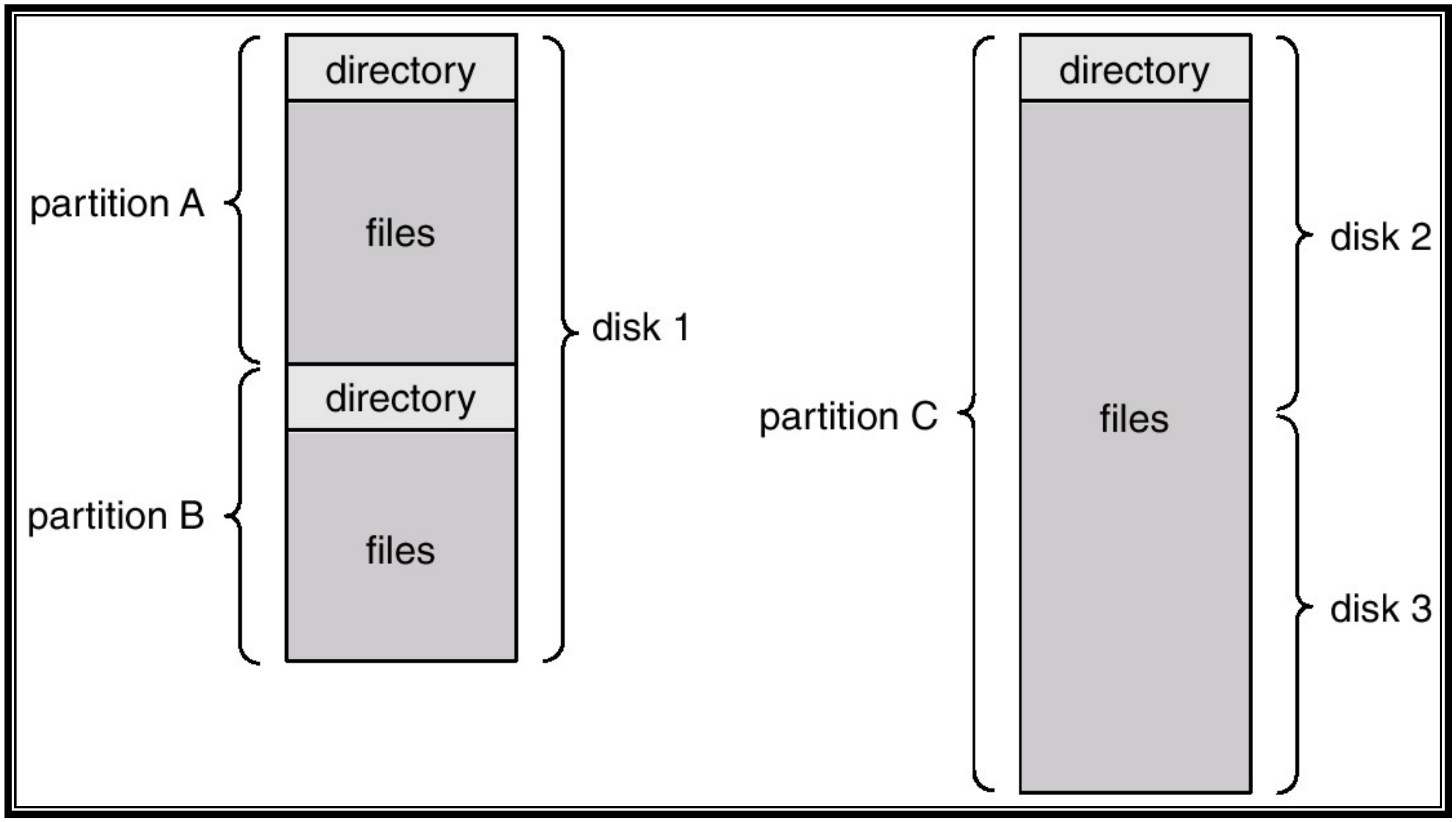
- Create an index for the file (as in a book)
- For large files, the index file itself may become too large. So use two level indices.



Directory Structure

- To manage all data and files in the system we need to organize it.
- Done in two parts
  - Partition the disk
  - For each partition maintain a directory(device directory or volume table of contents) which has info about all the files in that partition.
- The directory can be viewed as a symbol table that translates file names into their directory entries.
- 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
- Date last updated
- Owner ID
- Protection information

Operations Performed on Directory

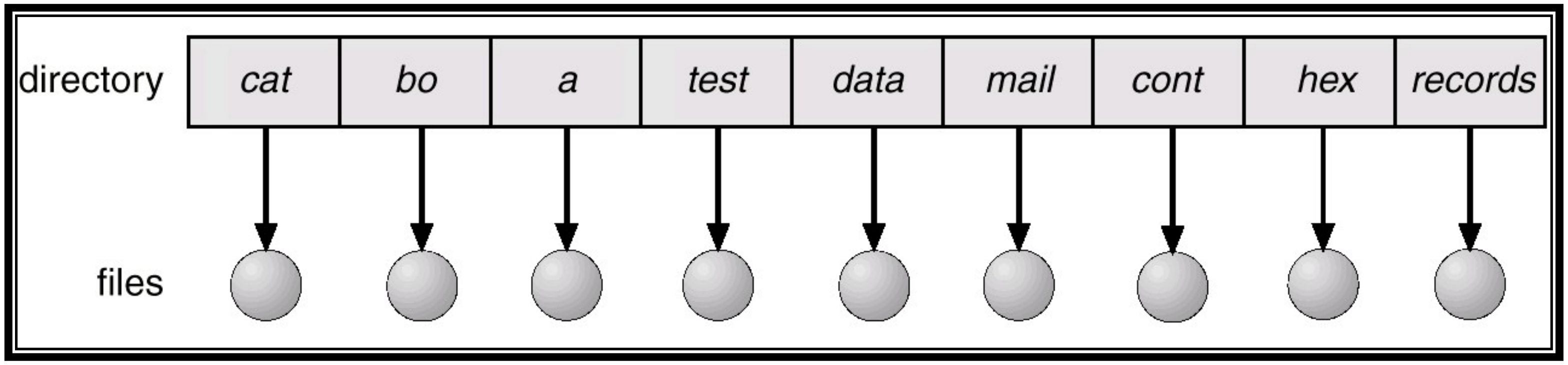
- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Organize the Directory (Logically)

- 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.

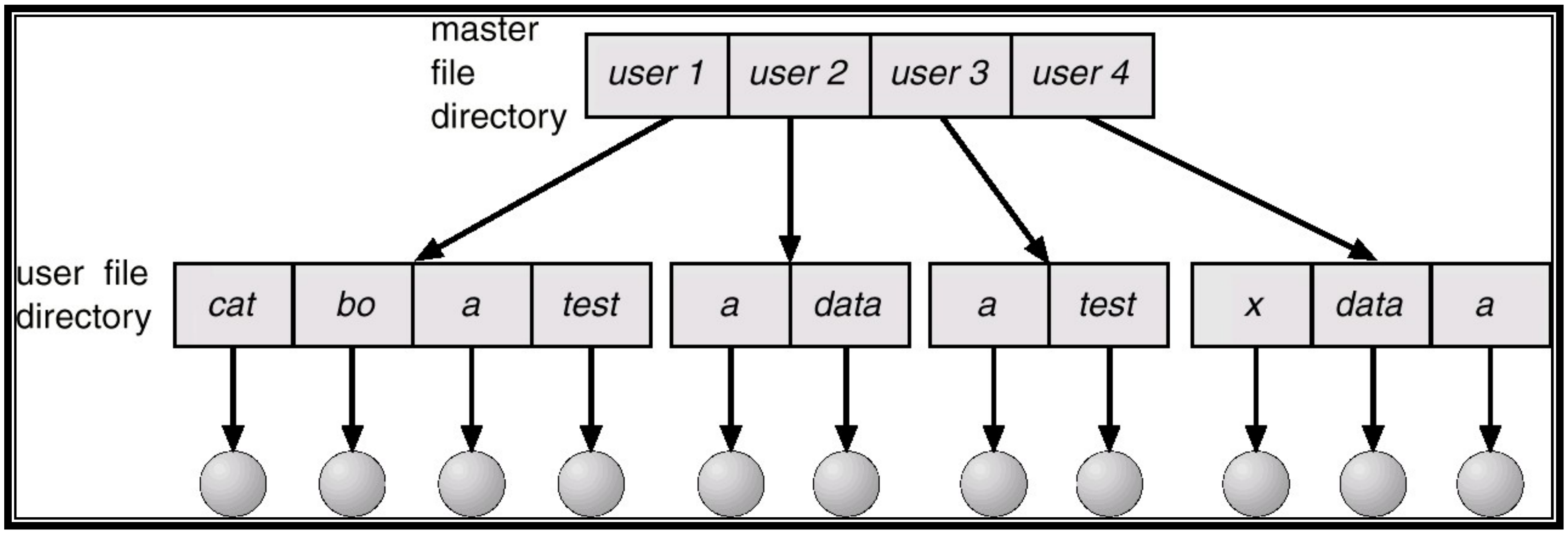


Naming problem – unique names

Grouping problem

Two-Level Directory

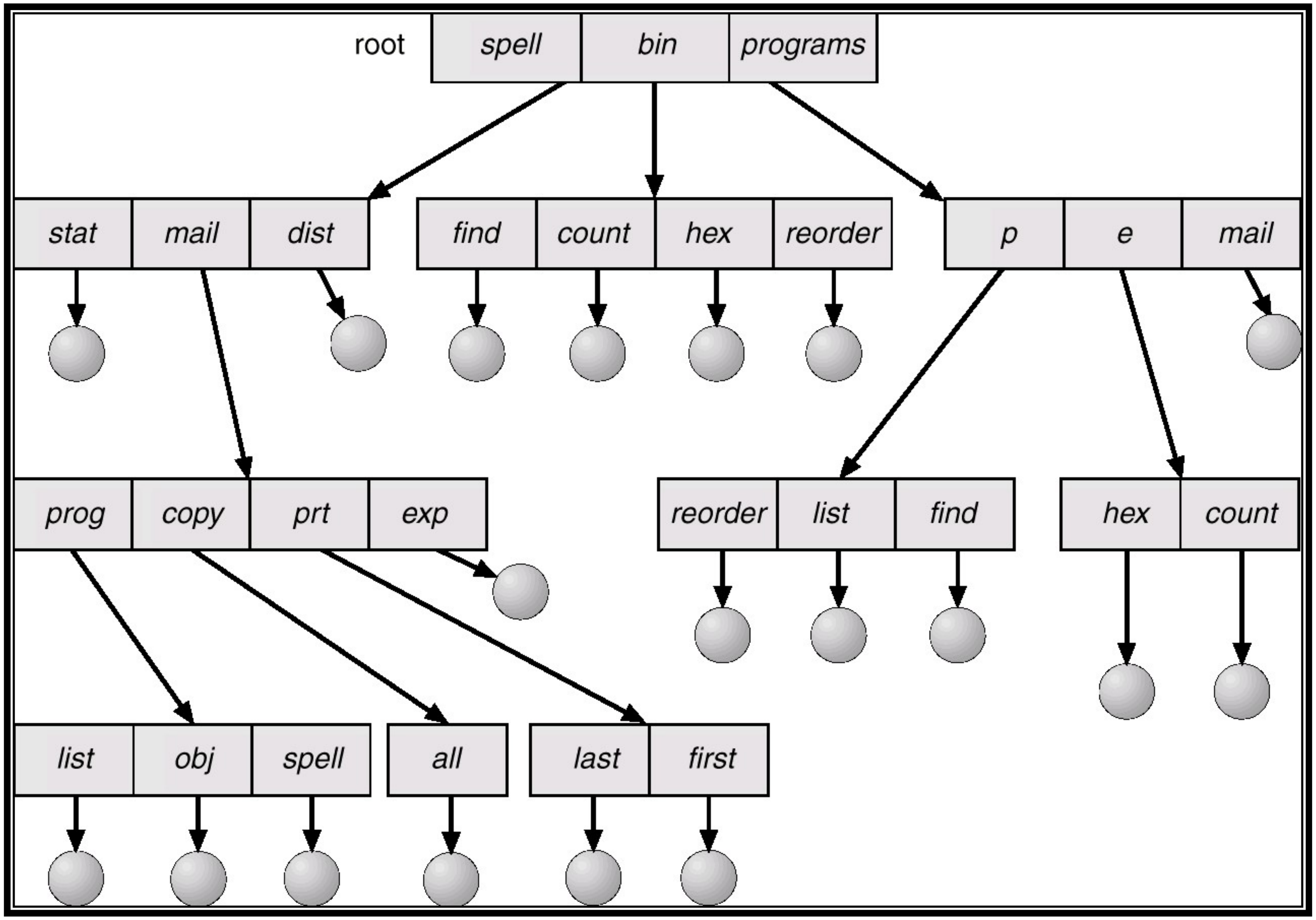
- Separate directory for each user.
  - MFD(Master File Directory) – which has all the users
  - UFD(User File Directory) – for every user
- For all file operations, the USD of the specified user alone is searched.



Two-level Directory

- Disadvantages:
  - Isolates the user completely.
  - If access to other user's file be permitted he should specify the path to the other file.
- Path name: user name+ filename
  - In MS-DOS : partition is specified as C:\user1\f1.txt
- Advantages:
  - Can have the same file name for different user
  - Efficient searching

Tree-Structured Directories



Tree-Structured Directories (Cont.)

- Extension of two-level tree.
- Advantages:
  - Efficient searching
  - Grouping Capability
- A directory can contain subdirectories (one bit is used to distinguish files(0) and subdirectories(1))
- Current directory (working directory)
  - A file is searched in the current directory. If not present specify the pathname or change working directory.
    - **cd** /spell/mail/prog
    - **type** list

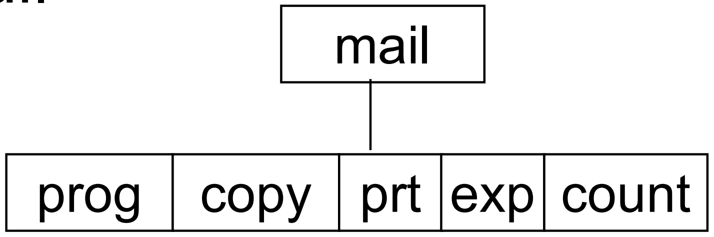
Tree-Structured Directories (Cont.)

- Absolute path(from root) or relative path name(from pwd)
- 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

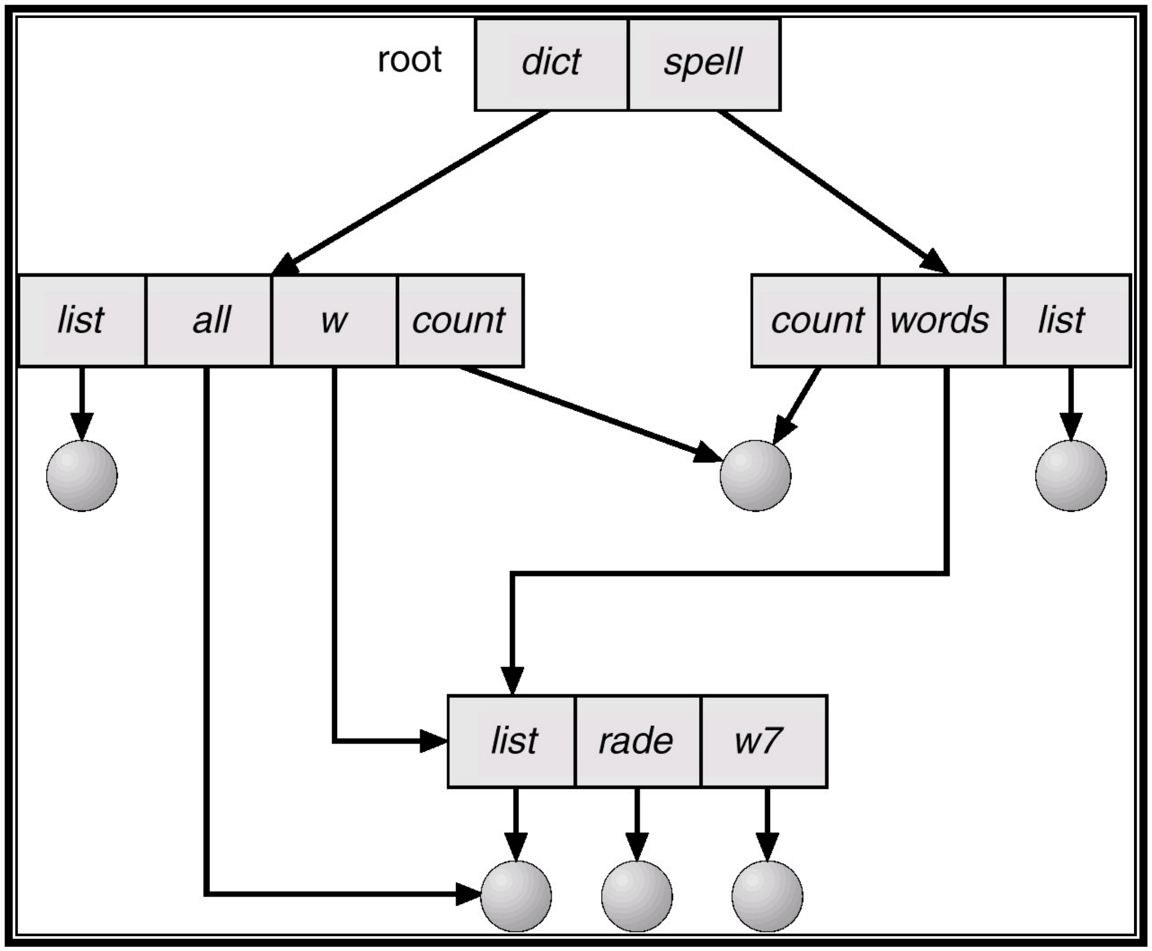
  mkdir count



Deleting "mail" ⇒ deleting the entire subtree rooted by "mail".

Acyclic-Graph Directories

- Have shared subdirectories and files.



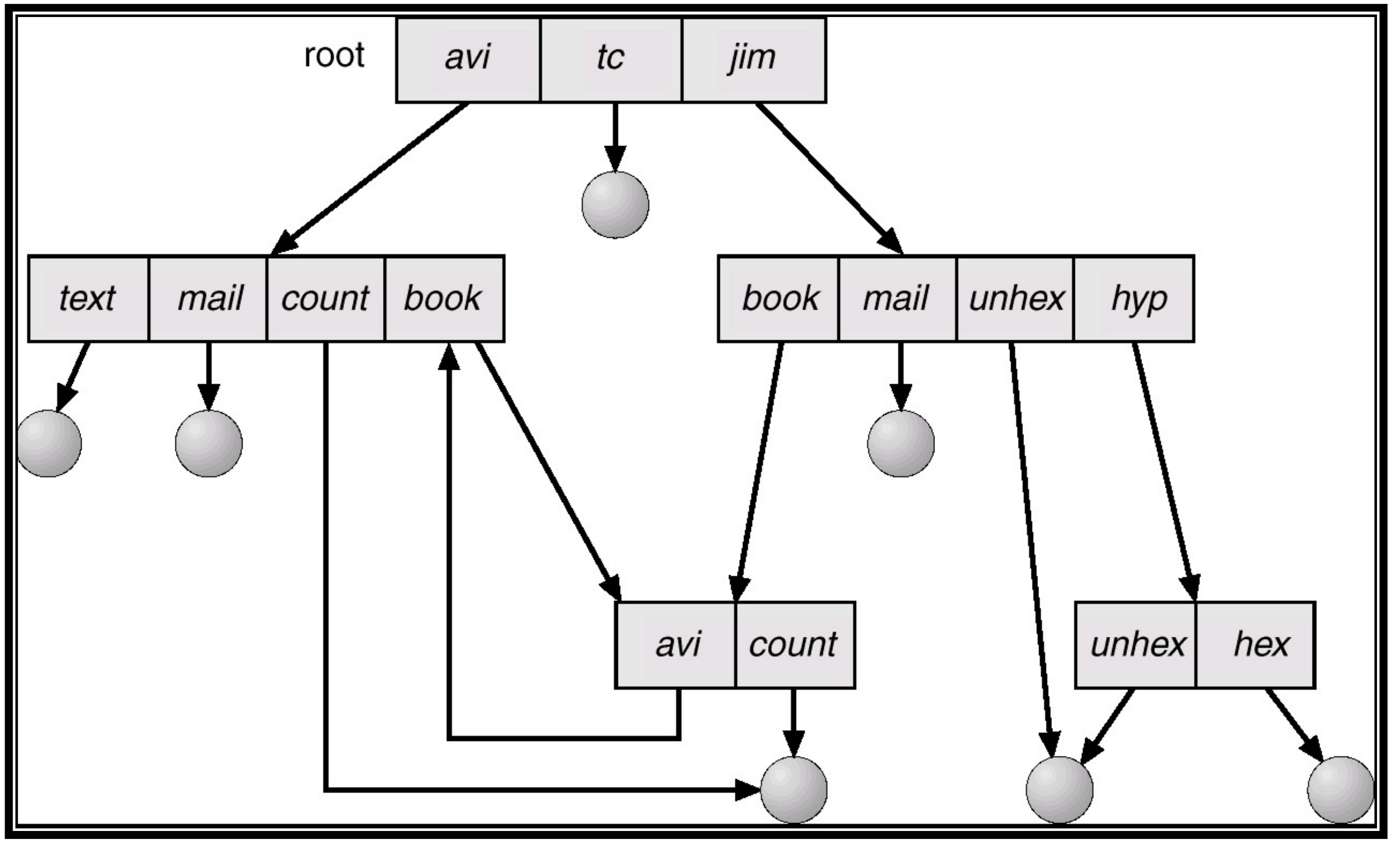
Acyclic-Graph Directories (Cont.)

- Not two copies of the same file but same file under two different names (aliasing). Changes in one is immediately reflected in the other.
- Implementation:
  - UNIX:
    - as a link – pointer to another file.
    - If a file is referenced and its current dir entry is marked as link then we obtain the original pathname for the file.
  - Creating duplicate copies
    - maintaining consistency of the file is difficult.
    - Problem in deletion
    - If *dict* deletes count $\Rightarrow$ dangling pointer.

Solutions for deleting a link

- Solutions:
  - Symbolic links –
    - just delete the link. if file entry itself is deleted leave the links dangling.
    - User's responsibility to remove those links.(can be checked when the file is referenced next)
    - Maintain a file reference list – for each time the file is referenced make an entry in this list. If list is empty, delete the file.
  - Non-Symbolic links (hard links):
    - Do not need to keep the entire list. Just keep track of the number of references made.(count)
    - If count is zero, delete the entry
    - Implemented in Unix

General Graph Directory



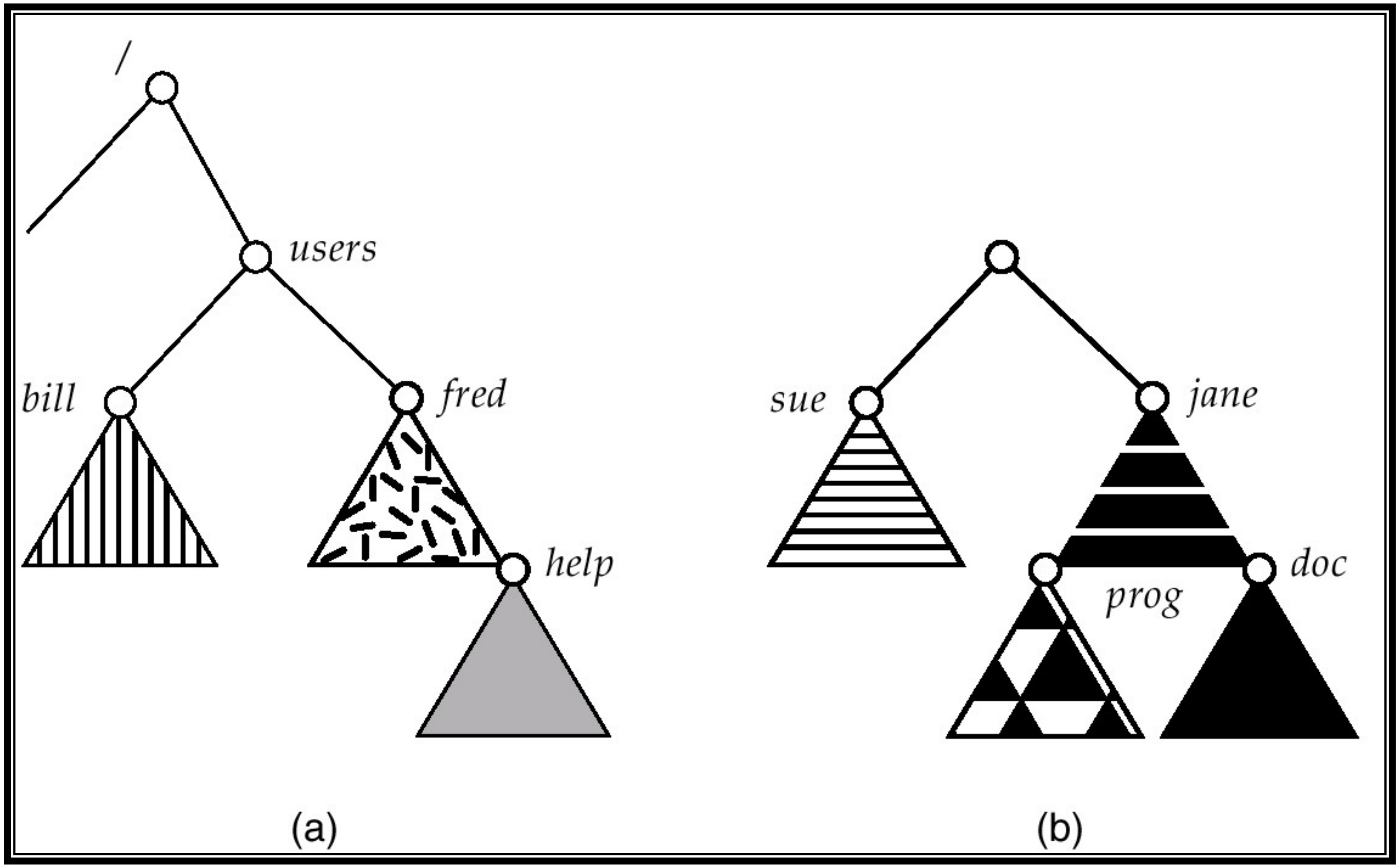
General Graph Directory (Cont.)

- If cycles are present in the graph, searching will be complicated as it might result in an infinite loop.
- How do we guarantee no cycles?
  - Allow only links to file not subdirectories.
  - Garbage collection –
    - Traverse the entire file system. Mark everything that is accessed.
    - In the next pass, move the unmarked items to a list of free space.
  - Every time a new link is added use a cycle detection algorithm to determine whether it is OK to add that link.
    - But these algorithms can be expensive.

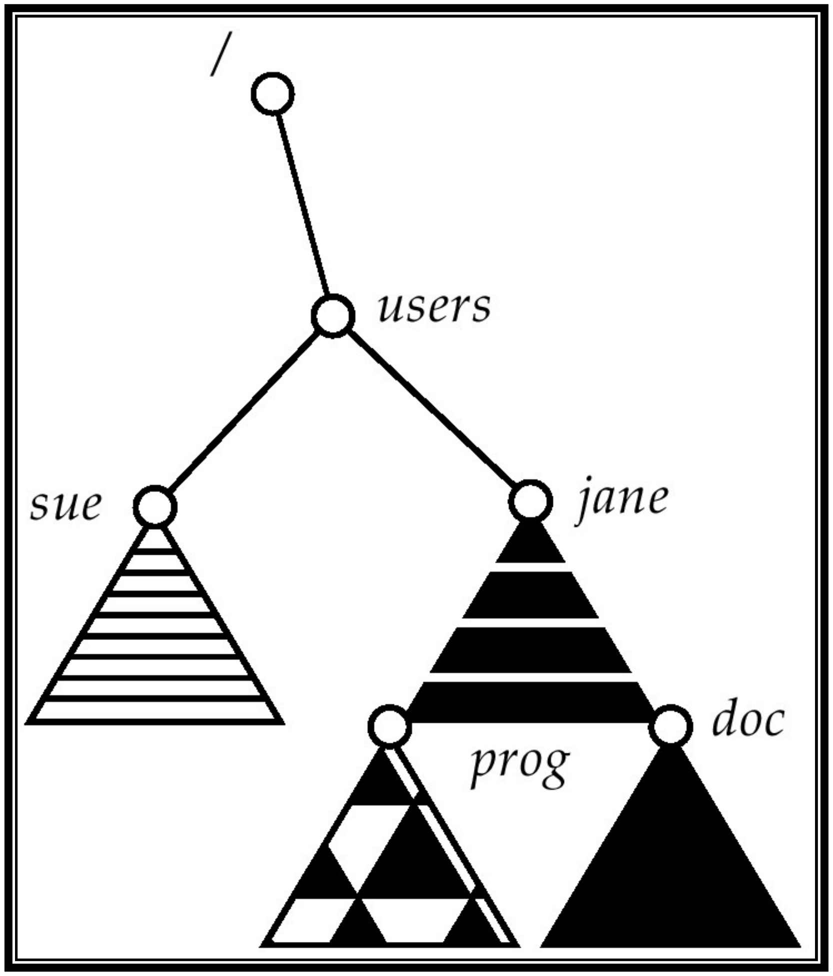
File System Mounting

- A file system must be **mounted** before it can be accessed.
- A un-mounted file system (I.e. Fig. 11-11(b)) is mounted at a **mount point.**
- Accessing the mounted filesystem:
  - mount point is an empty dir
  - If not, hide the original files and directories of the mount point until the mounted file system is unmounted.
- UNIX: manual mounting
- Windows family: Separate drive is assigned automatically when a device is mounted.

(a) Existing. (b) Unmounted Partition



Mount Point



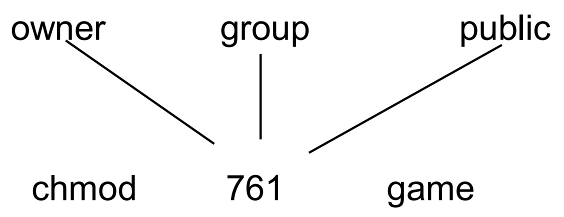
Protection

- Information should be kept safe from
  - physical damage (reliability)
    - have more no. of copies
  - improper access(protection).
- Two types of protection:
  - Prohibit Access
  - Controlled Access
- Types of operations that may be controlled:
  - Read
  - Write
  - Execute
  - Append
  - Delete
  - List file attributes

Access Lists and Groups

- Maintain a access control list(ACL) – may be very long
- To condense the length of acl restrict to three classes of users :
- 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


owner
group
public
chmod
761
game