Chapter 13
File System Interface

CS 3423 Operating Systems
Fall 2019
National Tsing Hua University

Outline

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

File Concept

- Different meanings
  - User's view: unit of data they can store and move
  - OS's view: unit of named data on some nonvolatile storage
- Logical vs. Physical storage unit
  - File: logically contiguous space
  - physical: disk sector, track, platter, ..
- Contents defined by file's creator
  - Consider text file, source file, executable file

File Attributes (1/2)

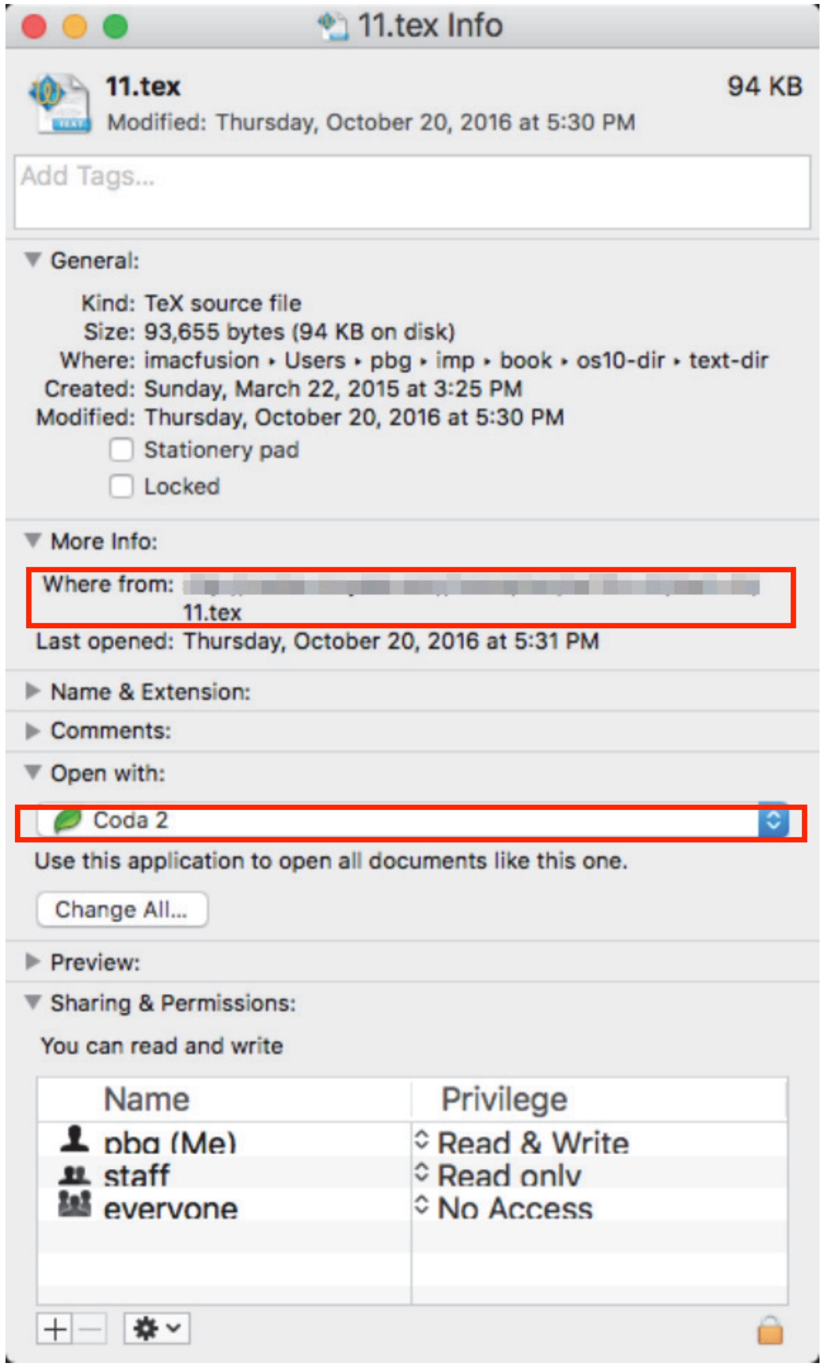
- Name
  - human-readable string, not part of content
- Identifier
  - unique tag (#) identifies file within file system
- Type
  - for systems that support different types
- Location
  - pointer to file location on device
- Size
  - current file size, in #bytes, #words, #blocks, possibly max
- Protection
  - controls who can read, write, execute

File Attributes (2/2)

- Access info (Timestamps & User ID)
  - Time, date, and user identification – data for protection, security, and usage monitoring
- Keeping metadata
  - In the directory structure, maintained on disk
  - extended file attributes such as file checksum
  - Could also be kept in a registry or metadata file

File info Window (macOS)

- Extended file attributes
  - Apps that can open the file
  - URL the file was downloaded from
  - User label, File icon
  - File's Checksum
- File info may be lost when file is transmitted (e.g., email attachment)
  - Some file info is stored in directory, rather than as part file content



11.tex Info

11.tex 94 KB
Modified: Thursday, October 20, 2016 at 5:30 PM

Add Tags...

▼ General:

Kind: TeX source file
Size: 93,655 bytes (94 KB on disk)
Where: imacfusion ▸ Users ▸ pbg ▸ imp ▸ book ▸ os10-dir ▸ text-dir
Created: Sunday, March 22, 2015 at 3:25 PM
Modified: Thursday, October 20, 2016 at 5:30 PM
□ Stationery pad
□ Locked

▼ More Info:

Where from: [redacted] 11.tex
Last opened: Thursday, October 20, 2016 at 5:31 PM

▶ Name & Extension:

▶ Comments:

▼ Open with:

Coda 2

Use this application to open all documents like this one.

Change All...

▶ Preview:

▼ Sharing & Permissions:

You can read and write

Name	Privilege
pbg (Me)	Read & Write
staff	Read only
everyone	No Access

Open File attributes

- Per-Process
  - Open-file table: tracks open files
  - File pointer: pointer to last read/write location in file
  - Access rights: per-process access mode information
- OS System-Wide
  - File-open count: # times a file is open
    - when last processes closes the file (count=0), allows removal of data from the open-file table
  - Disk location of the file: cache of data access information

File Operations

- Create
- Write – at write pointer location
- Read – at read pointer location
- Reposition within file - seek
- Delete -- from directory; reclaim space when no more directory contains the file
- Truncate -- write over file & update (instead of recreate) attributes
- 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

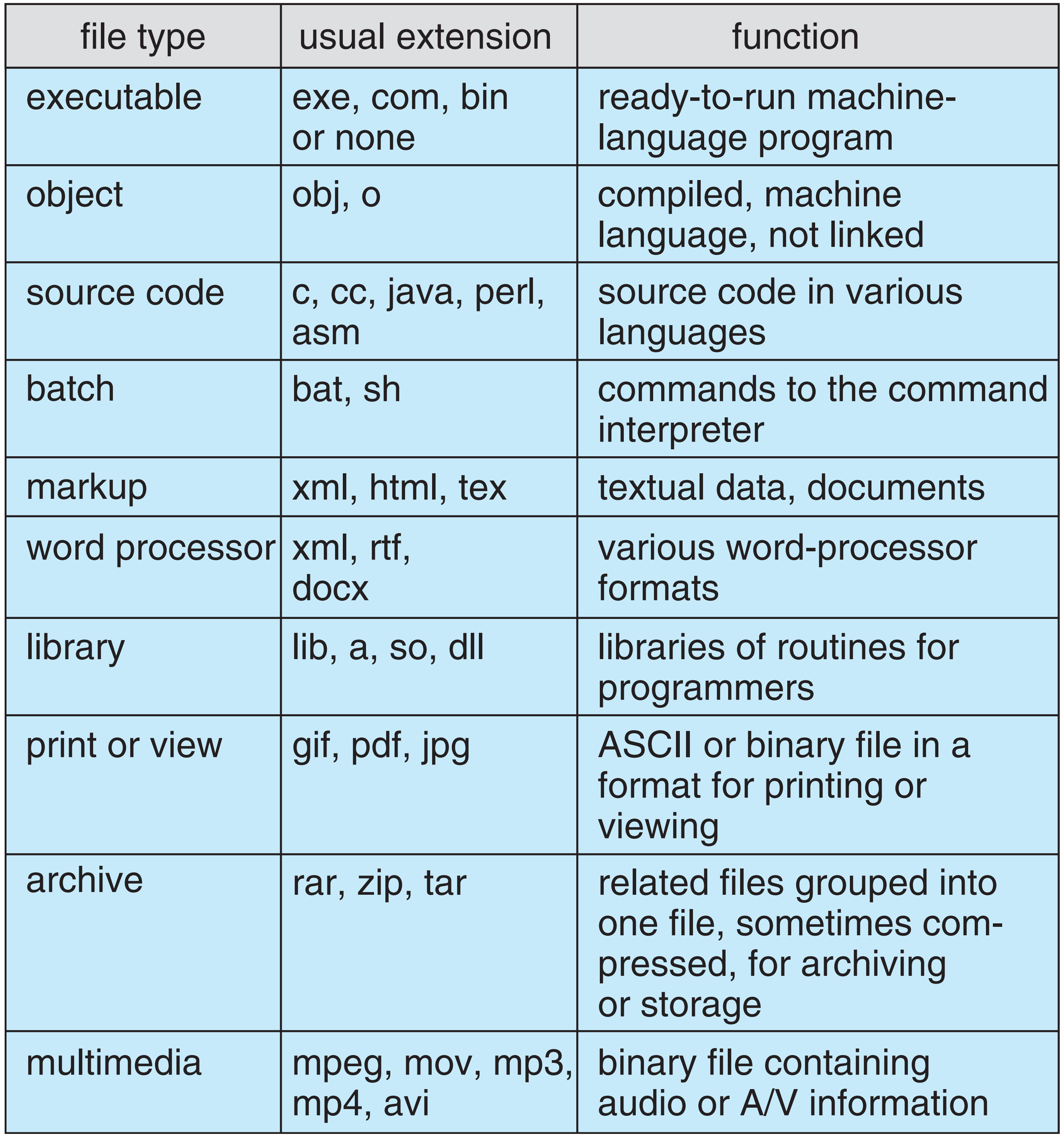
Locking of Open Files

- Provided by some operating systems and file systems
    - Similar to reader-writer locks
    - **Shared lock** similar to reader lock – several processes can acquire concurrently
    - **Exclusive lock** similar to writer lock
- Mandatory or advisory file-locking mechanisms
    - **Mandatory** – access is denied depending on locks held and requested
    - **Advisory** – processes can find status of locks and decide what to do

File types

- could be in file attribute
  - creator attribute => let the app figure out. OS just launches the app with file as argument
- Magic number
  - beginning of some binary files, esp. media
  - image, audio, PDF,
- Unix "file" command guesses file type
  - based on name, header/magic number, content sample

File Types – Name, Extension



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, cc, java, perl, asm	source code in various languages
batch	bat, sh	commands to the command interpreter
markup	xml, html, tex	textual data, documents
word processor	xml, rtf, docx	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	gif, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	rar, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, mp3, mp4, avi	binary file containing audio or A/V information

File Structure

- None - sequence of words, bytes
- Simple record structure
  - Lines (entries), fixed length or variable length
- Complex Structures
  - Formatted document
  - Relocatable load file
- Can simulate last two with first method by inserting appropriate control characters
- File structure may be decided by OS or application program
  - Bad idea for OS to dictate more than a few file structure!

Access Methods

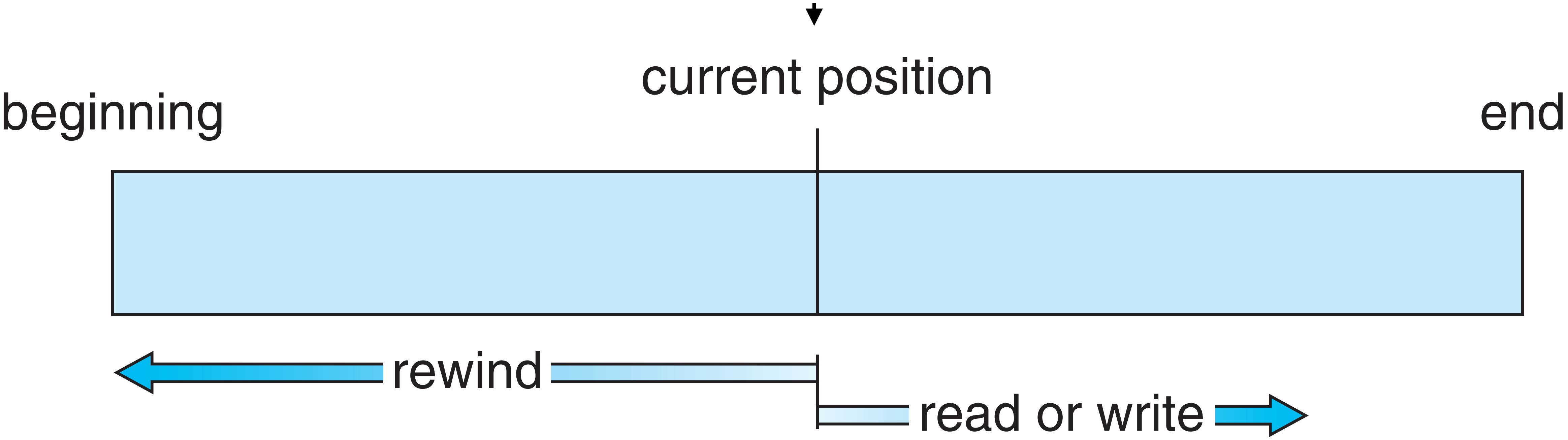
- Sequential Access
  - read next, write next
  - reset (to beginning)
  - no read after last write (rewrite)
- Direct Access – file is fixed-length logical records
  - read *n*, write *n*, position to *n*
  - read next record, write next record
  - rewrite *n*, where *n* = relative block number

Sequential-access File

```
fh.seek(0)
# move to beginning
```

```
fh.tell()
# get cure pos
```

```
fh.seek(0,
         whence=2)
# move to end
```



current position

beginning

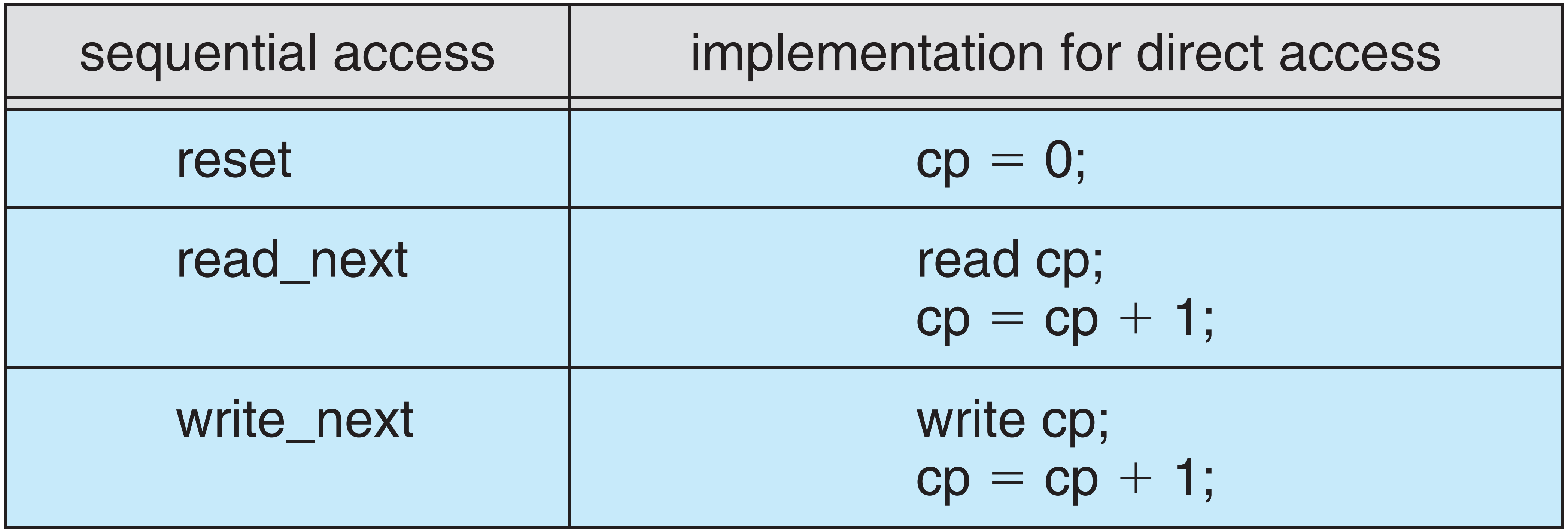
end

rewind

read or write

```
fh.seek(delta, whence=1)
# move relative to current position
```

Simulation of Sequential Access on Direct-access File



sequential access	implementation for direct access
reset	cp = 0;

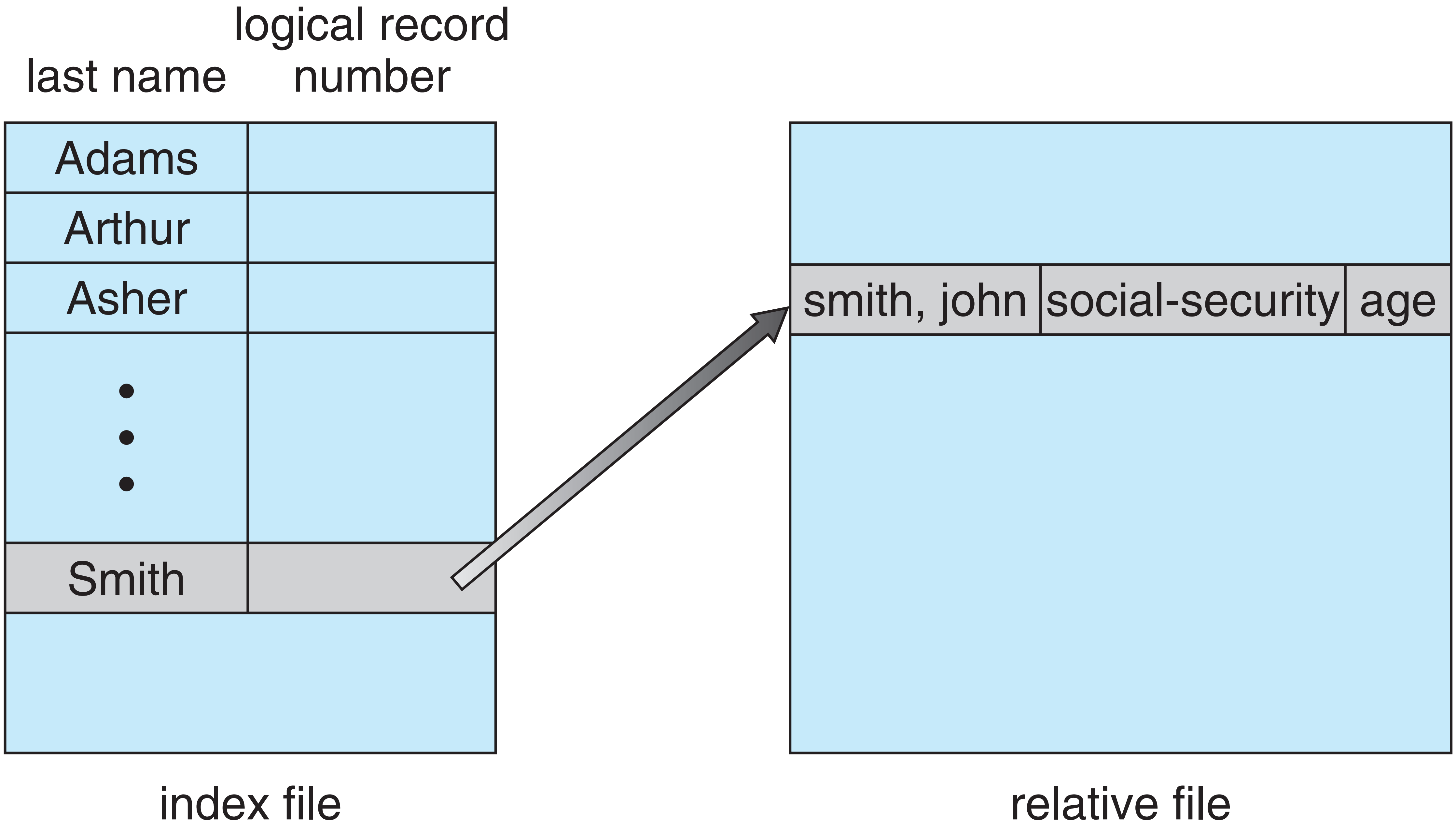
read_next	read cp;
	cp = cp + 1;

write_next	write cp;
	cp = cp + 1;

Other Access Methods: index

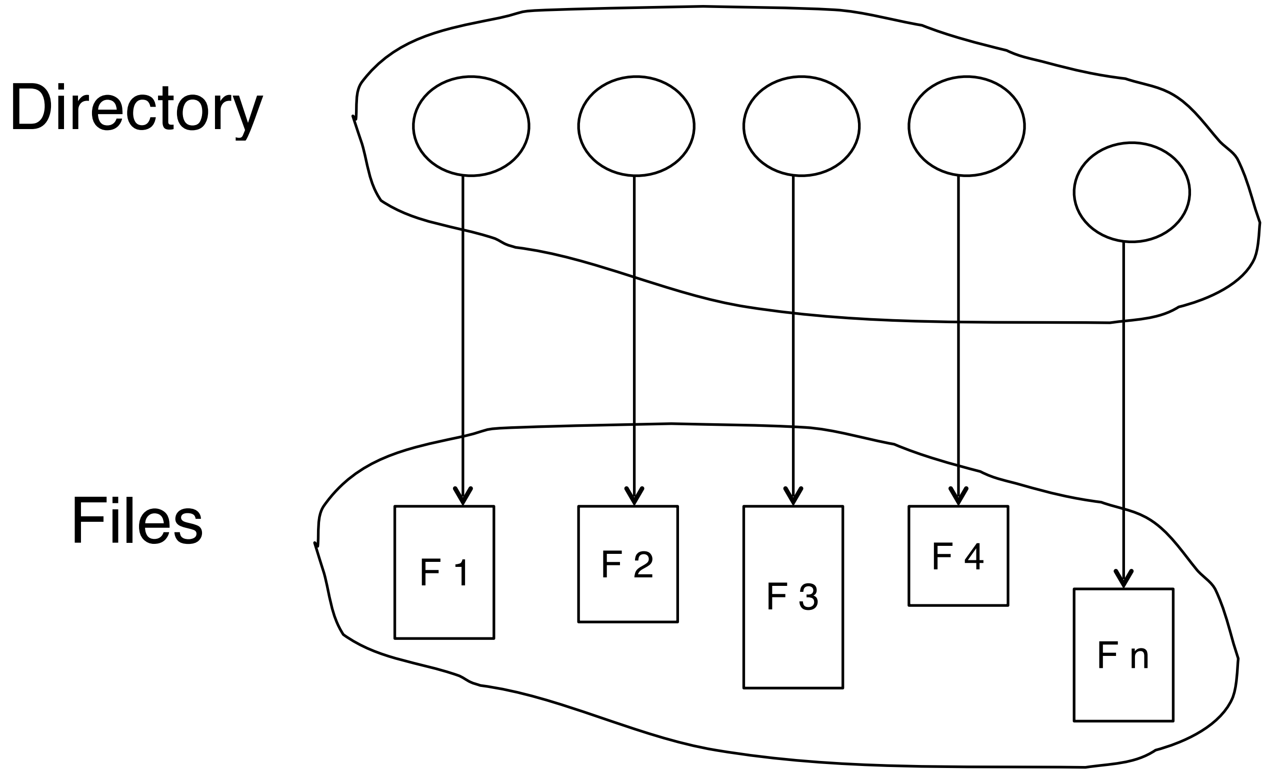
- Purpose
  - for fast determination of location of data to be operated on
  - (consider UPC code plus record of data about that item)
  - If too large, index (in memory) of the index (on disk)
- IBM indexed sequential-access method (ISAM) - by OS
  - Small master index, points to disk blocks of secondary index
  - File kept sorted on a defined key
- VMS provides **index** and **relative files**
  - as another example (see next slide)

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

Directories

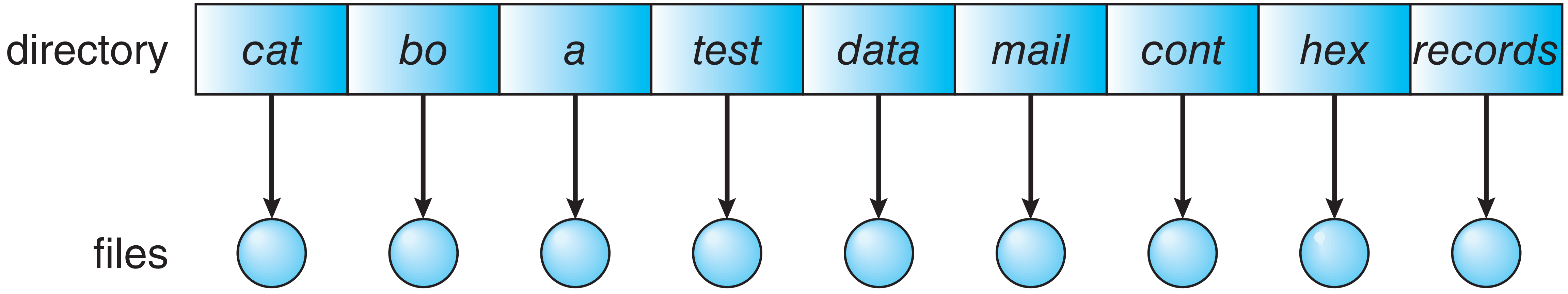
- "Folders" - containers of other files (and directories)
- Objective
  - Efficiency – locating a file quickly
- Functions
  - Naming – convenient to users
    - Two users can have same name for different files
    - The same file can have several different names
  - Grouping – logical grouping of files by properties, (e.g., all Java programs, all games, …)

Operations Performed on Directory

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

Single-Level Directory

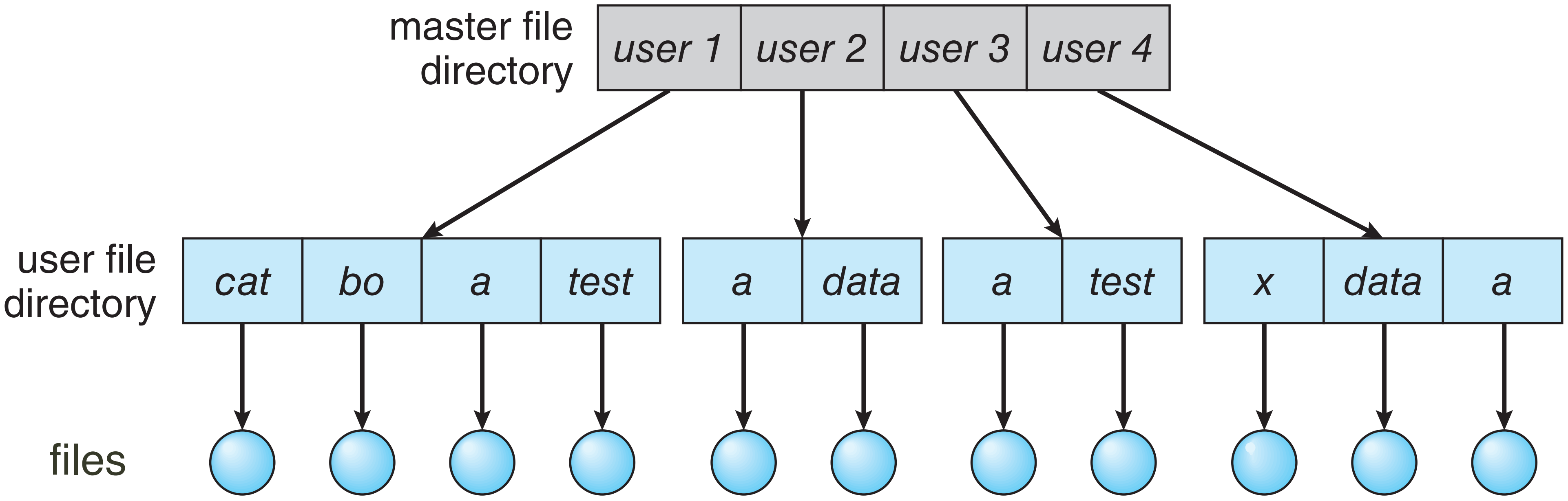
- A single directory for all users
  - (e.g. 1st Mac file system)



- Naming problem
- Grouping problem

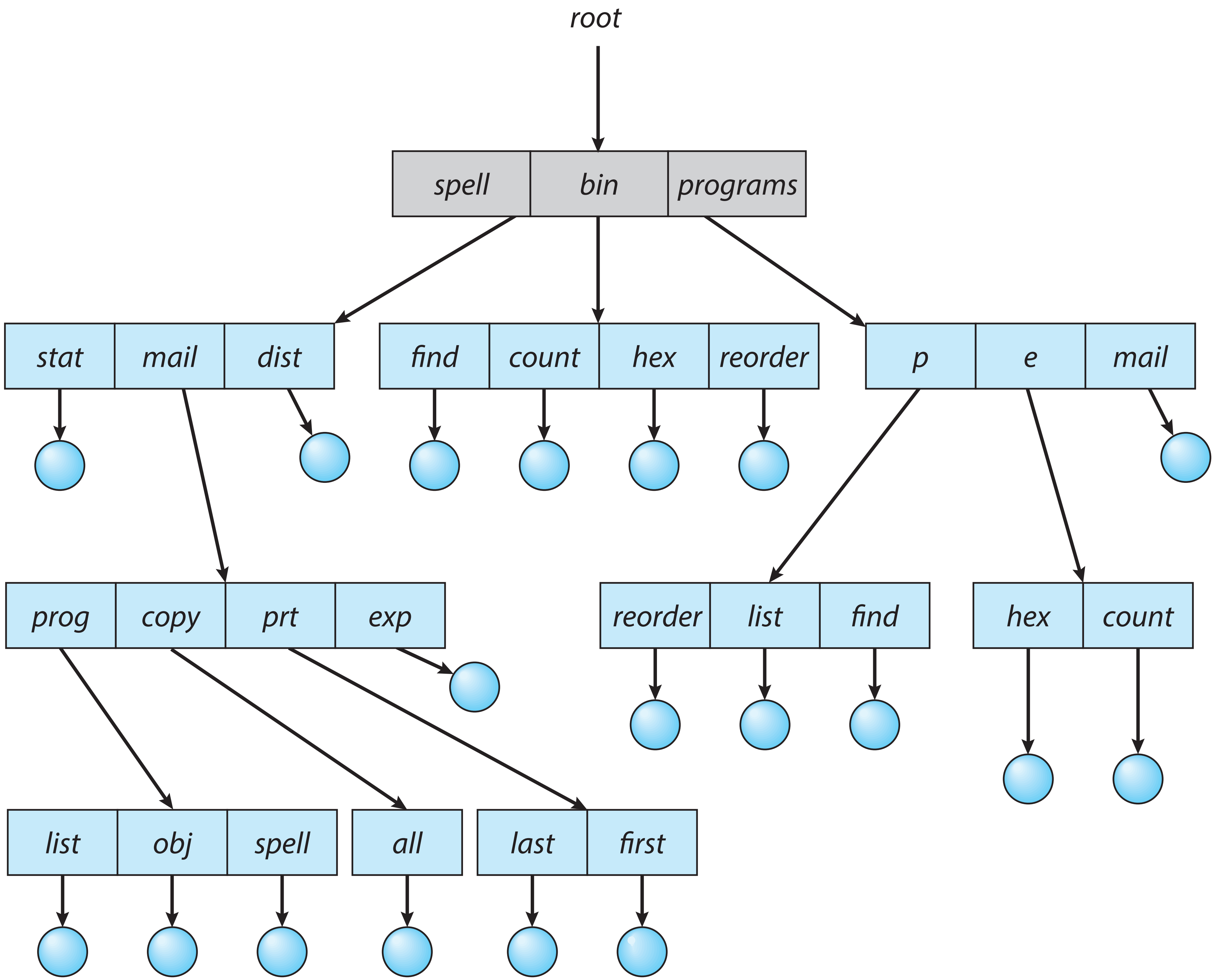
Two-Level Directory

- Separate directory for each user



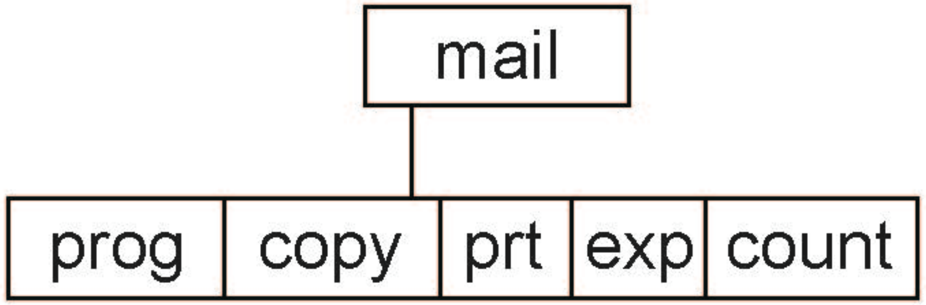
- Path name
- Can have the same file name for different user
- Efficient searching
- No grouping capability

Tree-Structured Directories



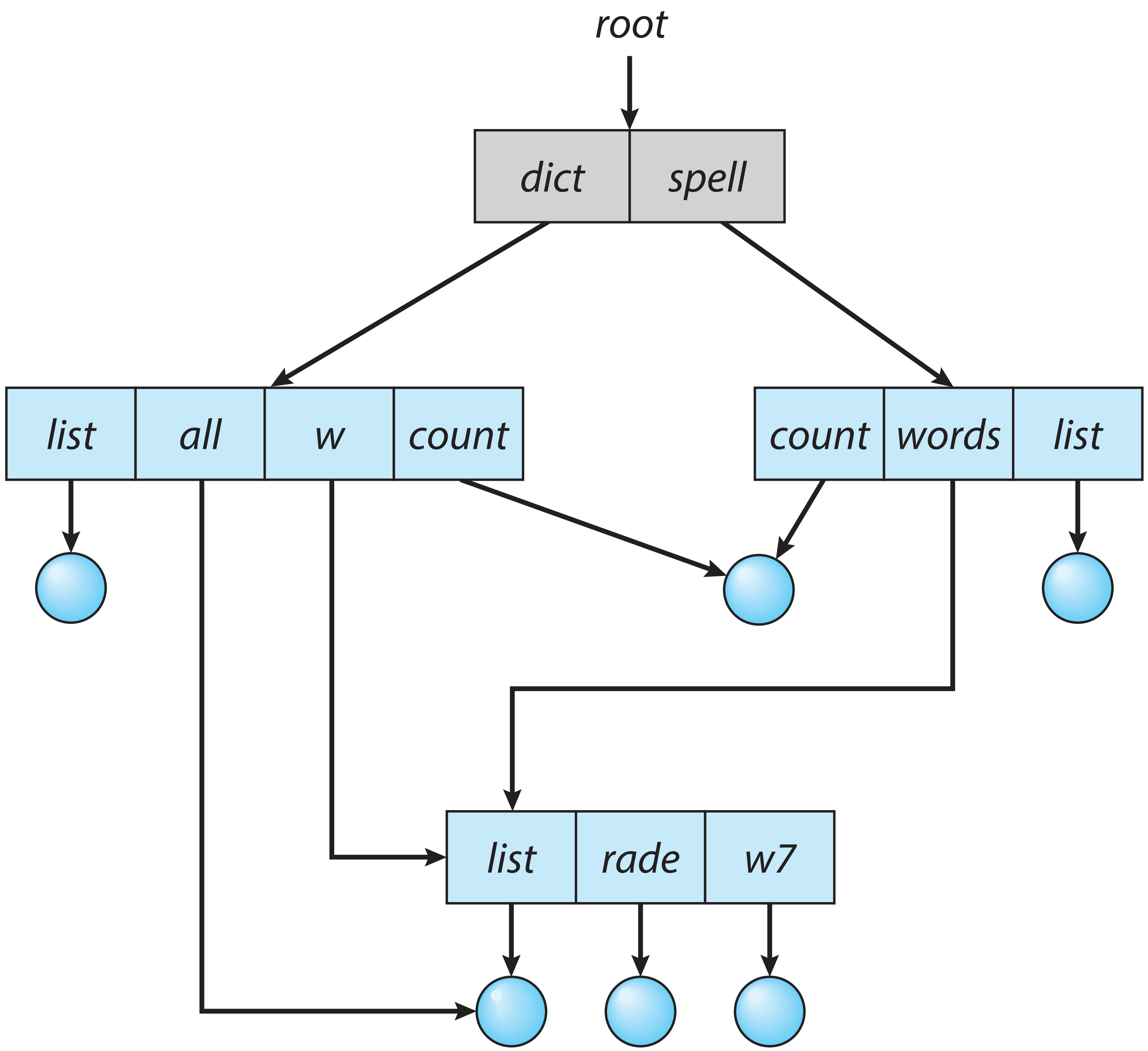
Tree-Structured Directories (Cont'd)

- Absolute or relative path name
- Creating a new file is done in current directory
- Delete a file
    - rm <file-name>
- Creating a new subdirectory
    - mkdir <dir-name>
- Removing a directory
    - rmdir <dir-name>: directory must be empty



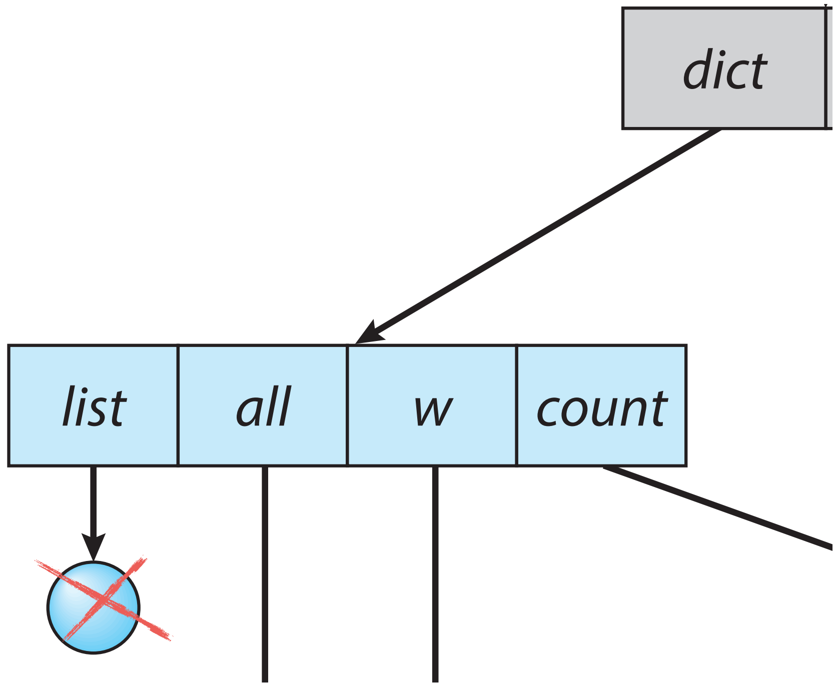
Acyclic-Graph Directories

- Have shared subdirectories and files

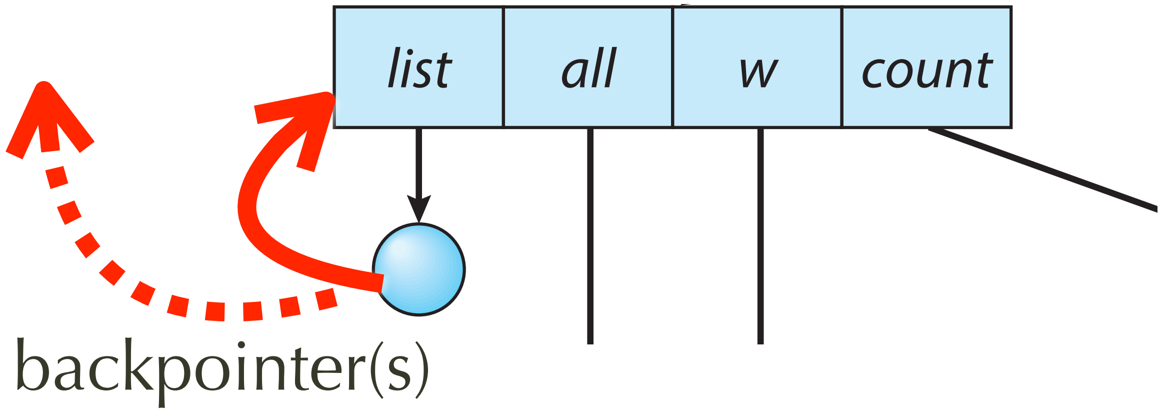


Acyclic-Graph Directories (Cont.)

- Deletion
  - If `list` file is deleted ⇒ `dict` directory now contains a dangling pointer!
  - need to preserve file until all references to file have been deleted.

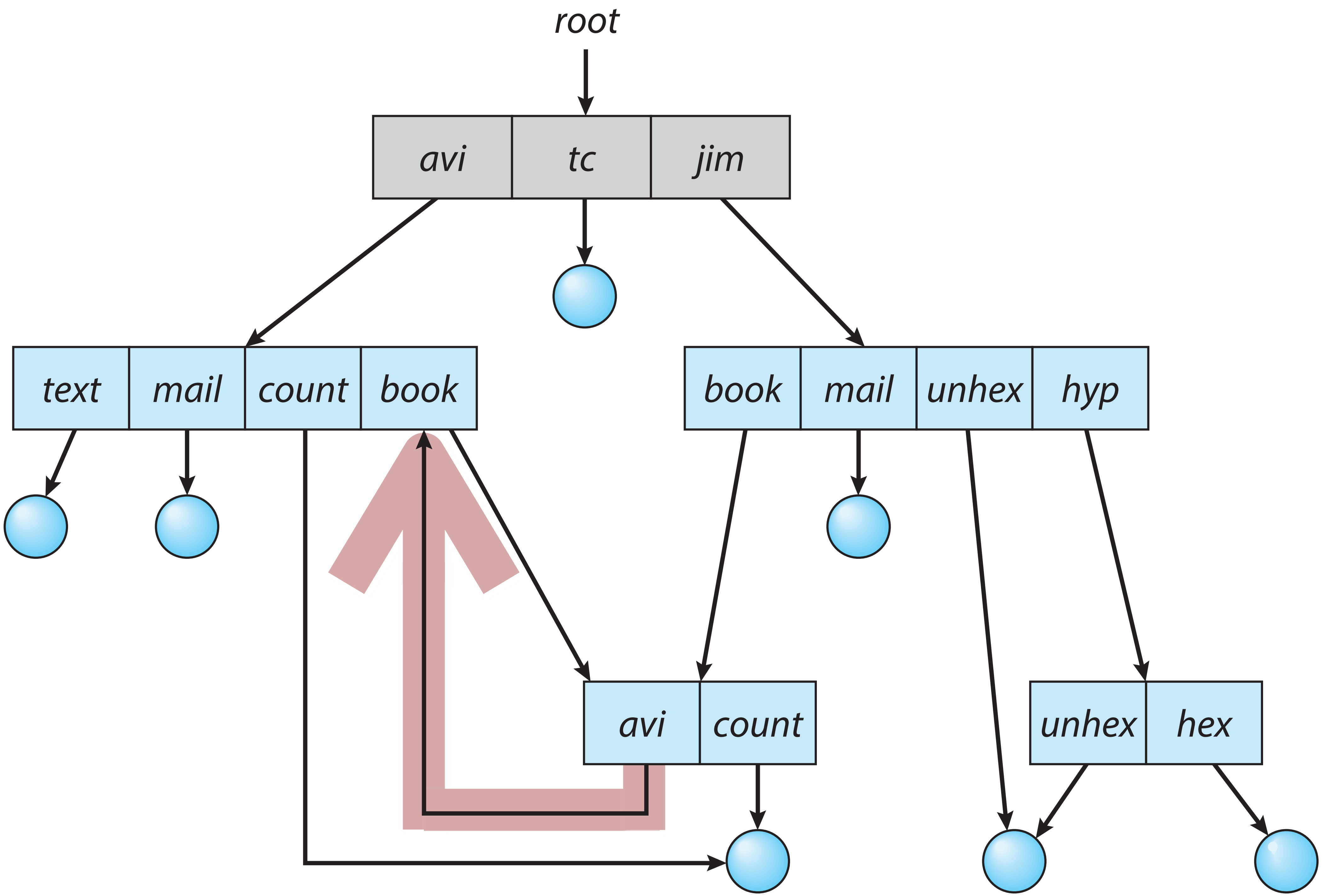


- Solution 1: Backpointer
  - but how many? Variable size records can be a problem
  - Backpointers using a daisy chain organization

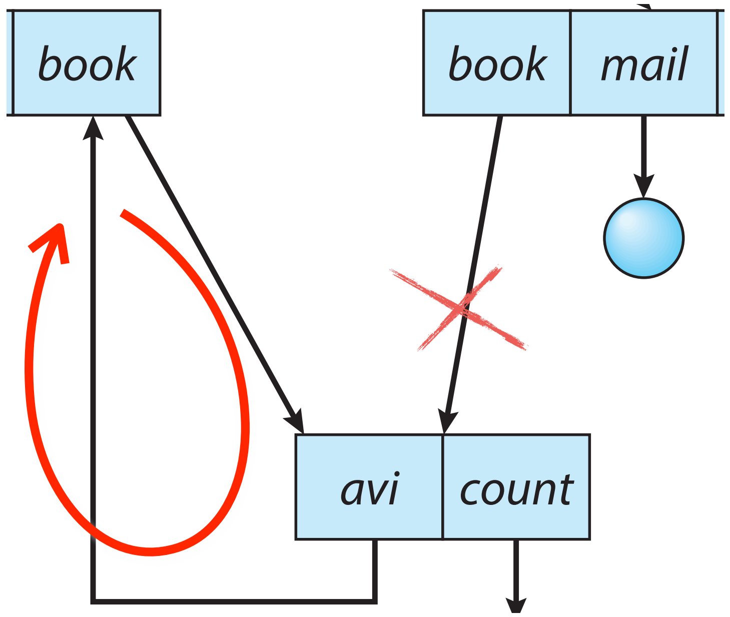


- Solution 2 (simpler): count #refs

General Graph Directory



General Graph Directory (Cont.)



- Issues with cycles
  - refer count 0 doesn't imply file accessible
  - Same problem as garbage collection
- Solutions
  - Garbage collection: 2 passes needed (mark and sweep) => expensive, rarely used
  - Allowing only (hard) links to file, not to directories => acyclic
  - Every time a new link is added, use a cycle detection algorithm to determine whether it is OK => expensive
  - When traversing directories, skip links => simpler

Protection vs. Reliability

- Protection
  - Owner of file controls what operations, by whom
  - Ops: Read, Write, Execute, Append, Delete, List
- Reliability
  - Backup copies, extra bits for error correction and detection
  - Physical isolation on different disks

Access Control

- Control access to file and directory based on
  - user name, type(s) of access
  - possibly also time and location allowed
  - possibly password protection, per-file or per-directory
- Most file systems use Access Control List
  - users and allowed operations per file

Access Control List (ACL)

- Content of ACL:
  - List of users
  - allowed modes of access: read, write, execute
- Problem:
  - too complex to specify!
- Unix Solution: condensed version
  - owner-group-public ACL

Protection Schemes when sharing Files on Multi-user system

- User ID (owner)
  - identify users, allowing permissions and protections of files & directories to be per-user
- Group ID
  - group = set of users (e.g., students, admin, faculty, …) => only sysadmin can create group!!
  - permitting each file & dir to define group-access rights
- Permission for a file or directory is defined for
  - an Owner, a Group, and Others

A Sample UNIX Directory Listing

```
-rw-rw-r--    1 pbg   staff      31200   Sep 3 08:30     intro.ps
drwx------    5 pbg   staff        512   Jul 8 09.33     private/
drwxrwxr-x    2 pbg   staff        512   Jul 8 09:35     doc/
drwxrwx---    2 jwg   student      512   Aug 3 14:13     student-proj/
-rw-r--r--    1 pbg   staff       9423   Feb 24 2017     program.c
-rwxr-xr-x    1 pbg   staff      20471   Feb 24 2017     program
drwx--x--x    4 tag   faculty      512   Jul 31 10:31    lib/
drwx------    3 pbg   staff       1024   Aug 29 06:52    mail/
drwxrwxrwx    3 pbg   staff        512   Jul 8 09:35     test/
```

r : "read" (read file content, or read directory listing!)

w : "write" (write file content, or modify directory -- add, delete, mv file)

x : "execute" a file or "enter" a directory (but might not be able to read)

d : "directory"

Setting file access in Unix

- Setting group
  - `$ chgrp` *groupName fileOrDir*
- Setting mode
  - `$ chmod` *modebits fileOrDir*
  - e.g., `$ chmod 761 myFile`

			RWX	
a) **owner access**	7	⇒	1 1 1	(RWX)
b) **group access**	6	⇒	1 1 0	(RW)
c) **public access**	1	⇒	0 0 1	(X)

- Can also do `chmod +r`, `chmod -r`, `chmod a+r`, etc