Directory

Directory Overview

- Directory similar to symbol table translating file names to their directory entries
  - Can be organized in many ways
- Organization needs to support operations including:
  - Search for a file or multiple files
  - Create a file
  - Delete a file
  - List a directory
  - Rename a file
  - Traverse the file system

Directory Organization

- Should have the features:
  - 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, …) or arbitrarily

Directory – A Special Kind of File

- A tool for users & applications to organize and find files
  - User-friendly names
  - Names that are meaningful over long periods of time
- The data structure for OS to locate files (i.e., containers) on disk

Directory structures

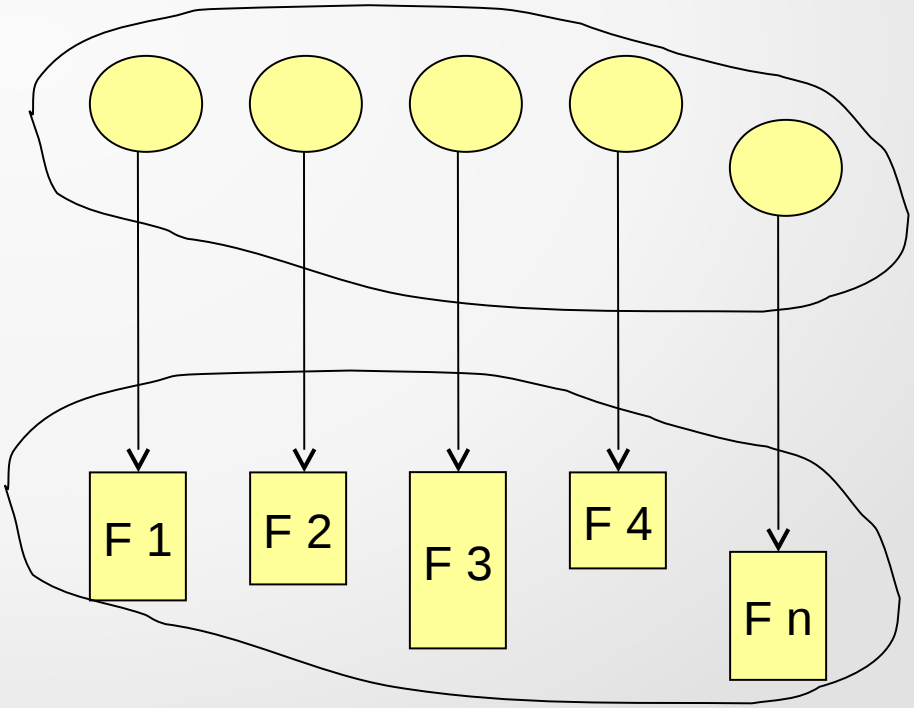
- Single level
  - One directory per system, one entry pointing to each file
  - Small, single-user or single-use systems
    - PDA, cell phone, etc.
- Two-level
  - Single “master” directory per system
  - Each entry points to one single-level directory per user
  - Uncommon in modern operating systems
- Hierarchical/tree
  - Any directory entry may point to
    - Individual file
    - Another directory
  - Common in most modern operating systems

Directory Considerations

- *Efficiency* – locating a file quickly.
- *Naming* – convenient to users.
  - Separate users can use same name for separate files.
  - The same file can have different names for different users.
  - Names need only be unique within a directory
- *Grouping* – logical grouping of files by properties
  - e.g., all Java programs, all games, …

Directories

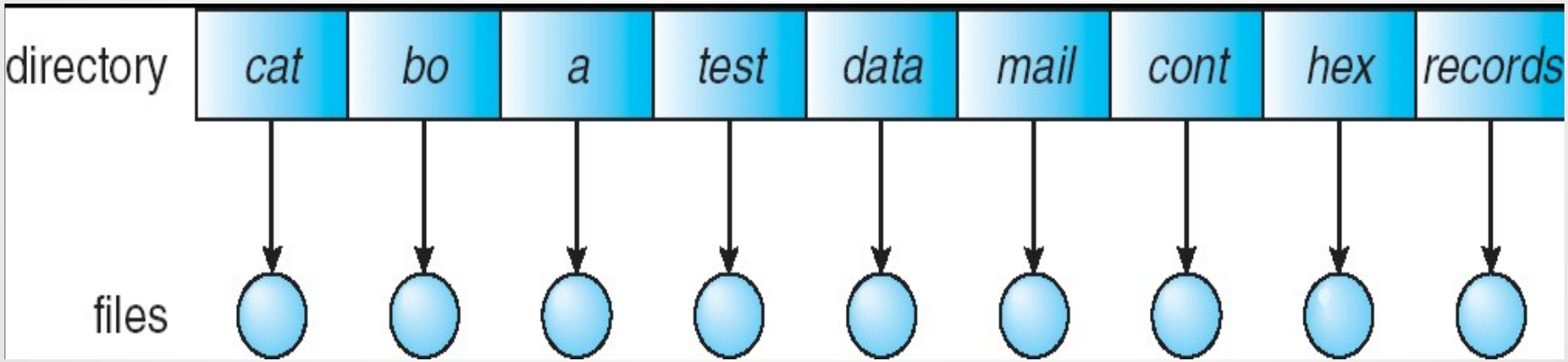
- Directories/folders keep track of files
- Is a symbol table that translates file names to directory entries
- Usually are themselves files
- How to structure the directory to optimize all of the following:
  - Search a file
  - Create a file
  - Delete a file
  - List directory
  - Rename a file
  - Traversing the FS



Single-level Directory

- One directory for all files in the volume
  - Called root directory

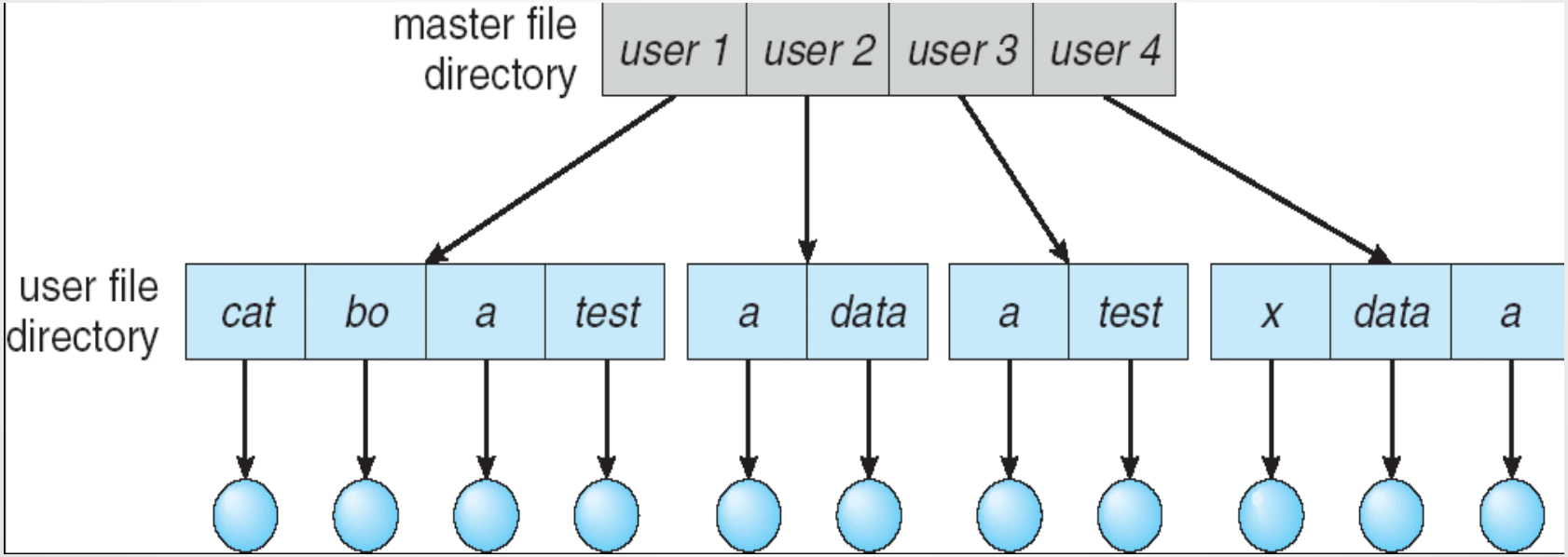
  Used in early PCs, even the first supercomputer CDC 6600



- Pros: simplicity, ability to quickly locate files
- Cons: Naming problem, Grouping problem

Two-level directory

- Each user has a separate directory

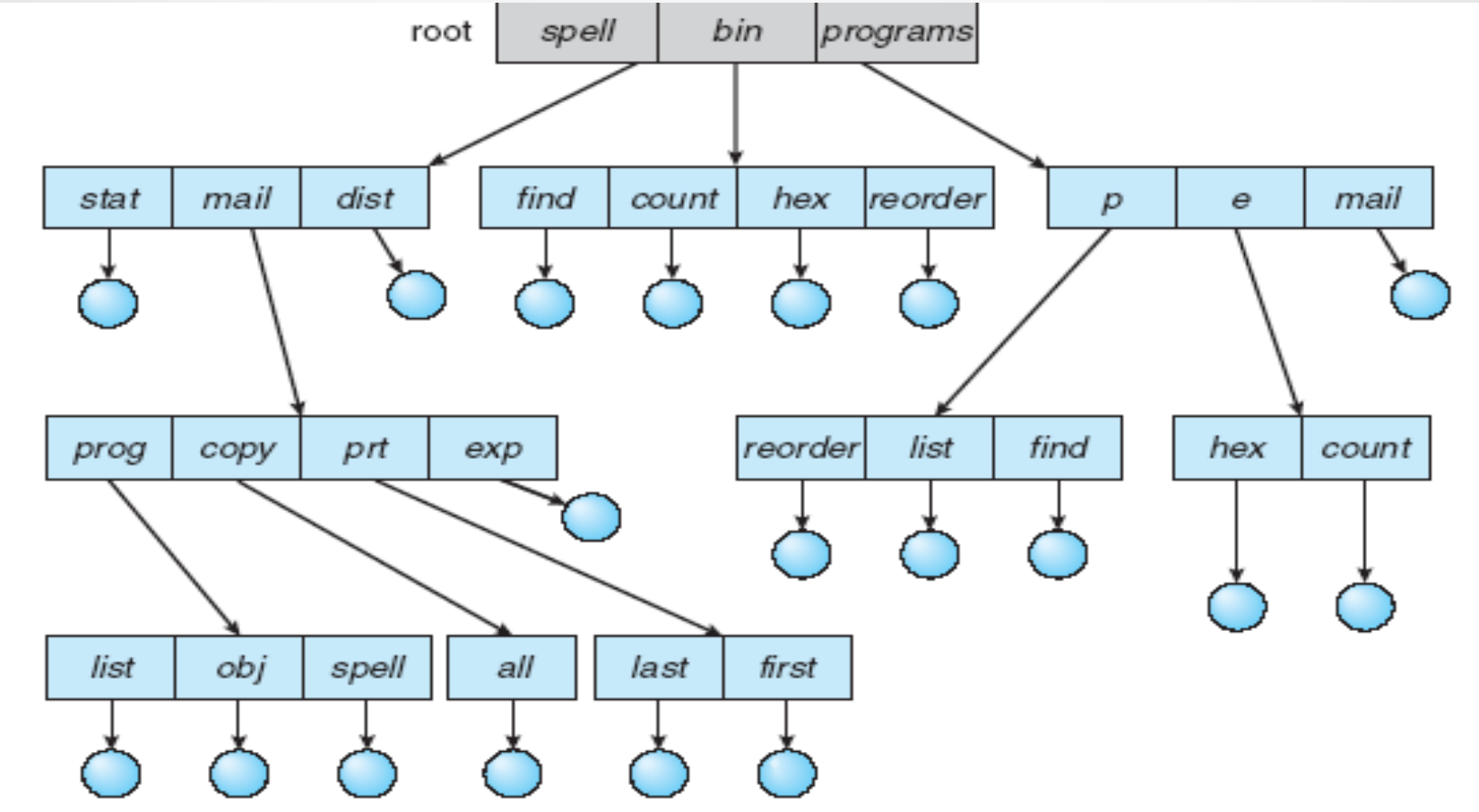


Two-level directory

- Pros:
  - Solves name collision.
  - Can have the same file name for different user.
  - Efficient searching.
  - Isolation
- Cons:
  - No grouping capability
  - May not allow a user to access other users’ files

Tree-structured Directory

- Directory is now a tree of arbitrary height



Tree-structured Directory

- Directory contains files and sub-directories
- A bit in directory entry differentiates files from subdirectories
- This ensures:
    - Efficient searching
    - Grouping Capability
    - Current directory (working directory)
        - cd /spell/mail/prog
        - type list

Tree-Structured Directories (Cont.)

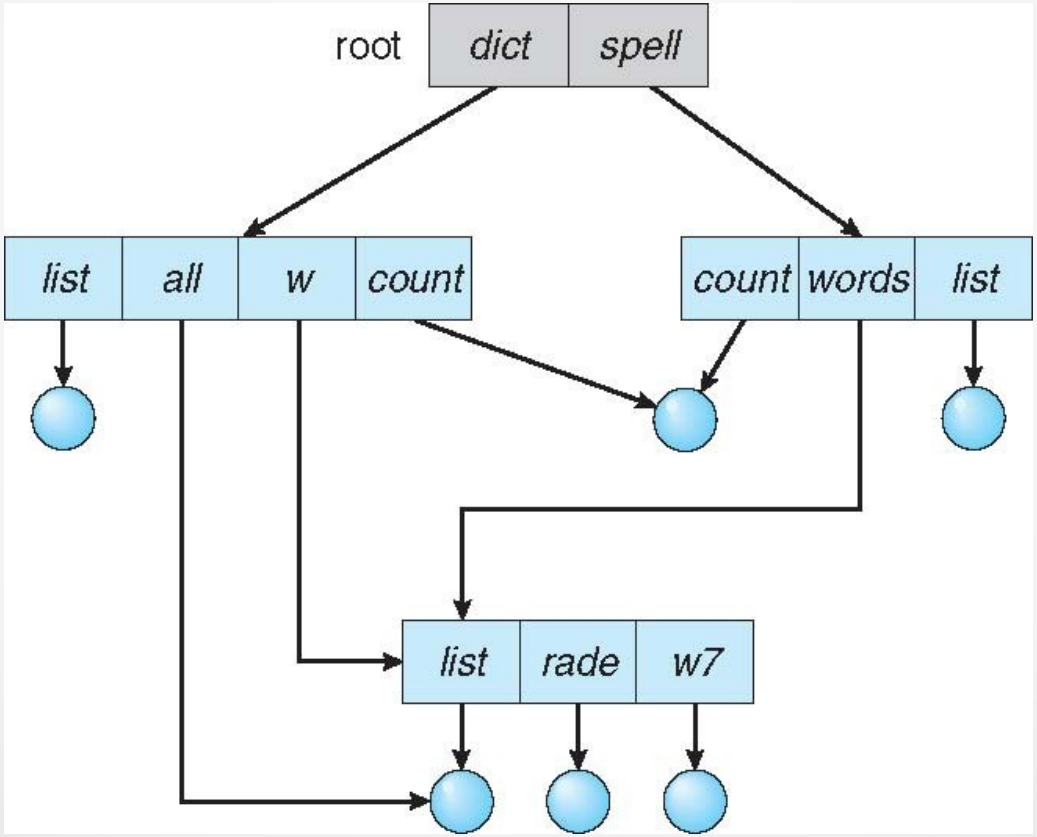
- Users can create directories within their directory
- Directory can then contain files or other directories
- Directory can be another file with defined formatting and attribute indicating its type
- Separate system calls to manage directory actions

Path: Directories & File

- Path: A unique name of a file or directory in a filesystem
  - E.g., /usr/bin/top
- Absolute path or full path: The path that points to the same location on one file system regardless of working directory or combined path.
  - It is usually written in reference to a root directory.
  - Ex: /home/users/c/computerhope/public_html/cgi-bin
- Relative path: The path related to the current working directory.
  - Ex: /public_html/cgi-bin

Acyclic-Graph Directories

- Have shared sub-directories and files



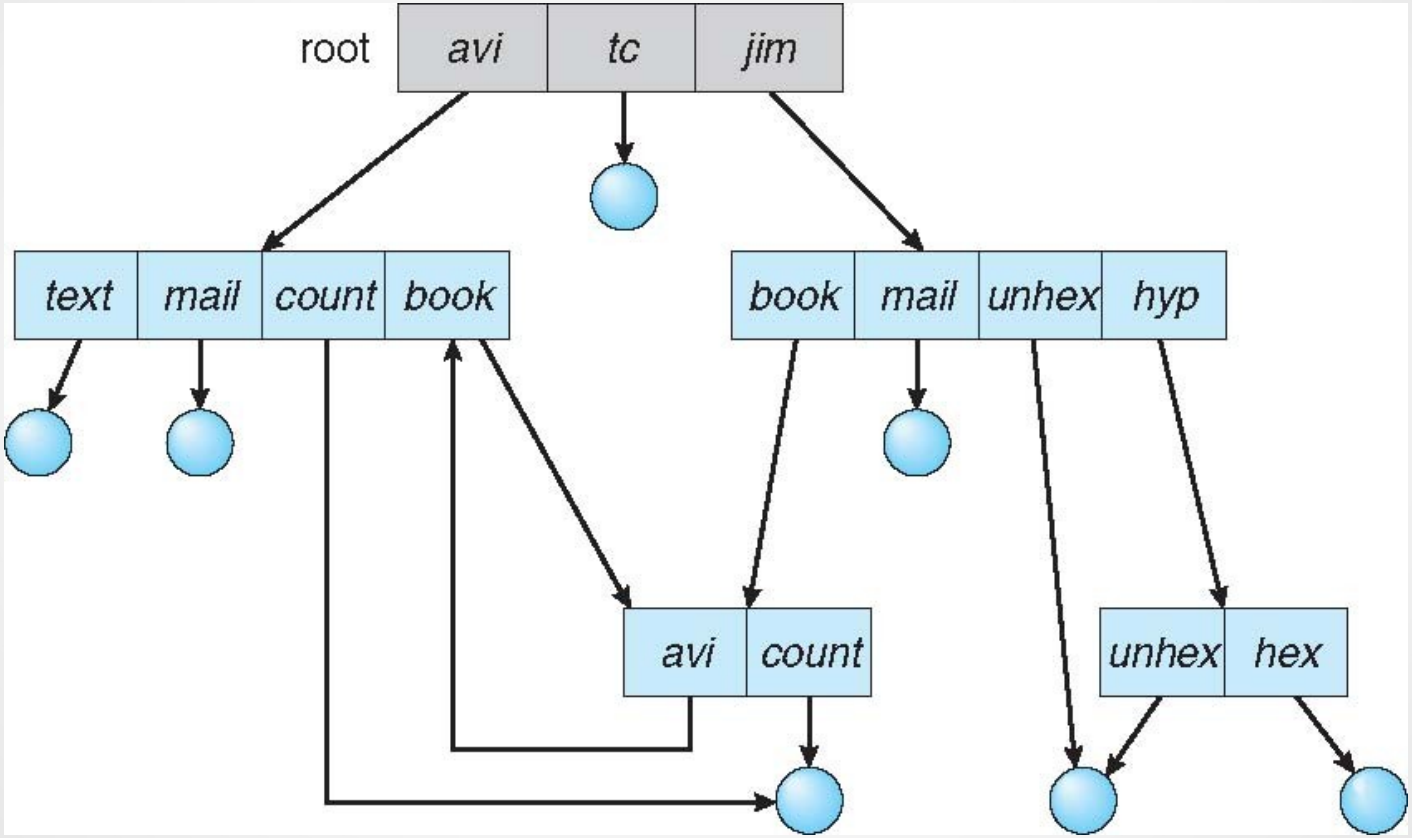
Acyclic-Graph Directories (Cont.)

- Adds ability to directly share directories between users.
- But can now have multiple absolute paths to the same file.
- Two different names (aliasing)
  - If dict deletes list: dangling pointer
- Solutions:
  - Back pointers, so we can delete all pointers
    - Variable size records a problem
  - Entry-hold-count solution

Acyclic-Graph Directories (Cont.)

- New directory entry type
  - Link – another name (pointer) to an existing file
    - Indirect pointer
    - Delete link separate from the files
    - Hard and symbolic
- Resolve the link – follow pointer to locate the file

General Graph Directory



General Graph Directory (Cont.)

- 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
  - Or just bypass links during directory traversal