Filesystem

Disclaimer: some slides are adopted from book authors' slides with permission

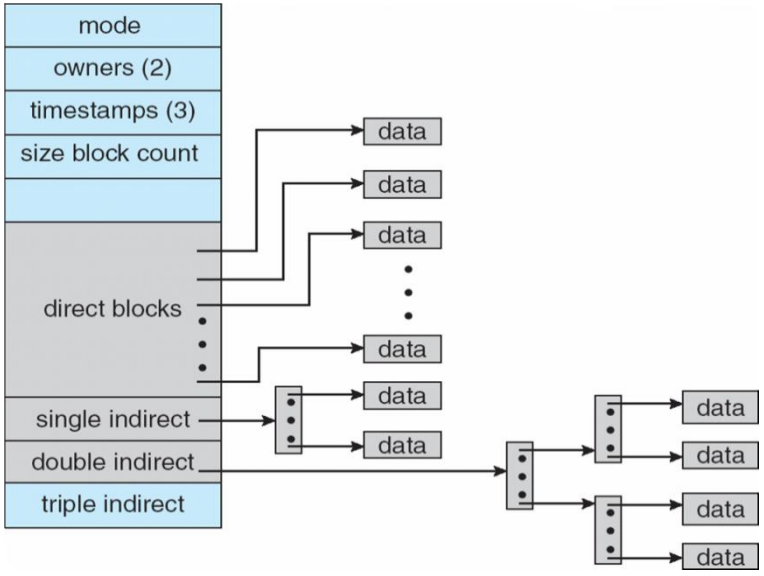
Concepts to Learn

- **Directory**
- **Caching**
- **Virtual File System**
- Putting it all together: FAT32 and Ext2
- Journaling
- Network filesystem (NFS)

KU THE UNIVERSITY OF KANSAS

How To Access Files?

- Filename (e.g., “project2.c”)
  - Must be converted to the file header (inode)



  - How to find the inode for a given filename?

Directory

- A special file contains a table of
  - Filename (directory name) & inode number pairs

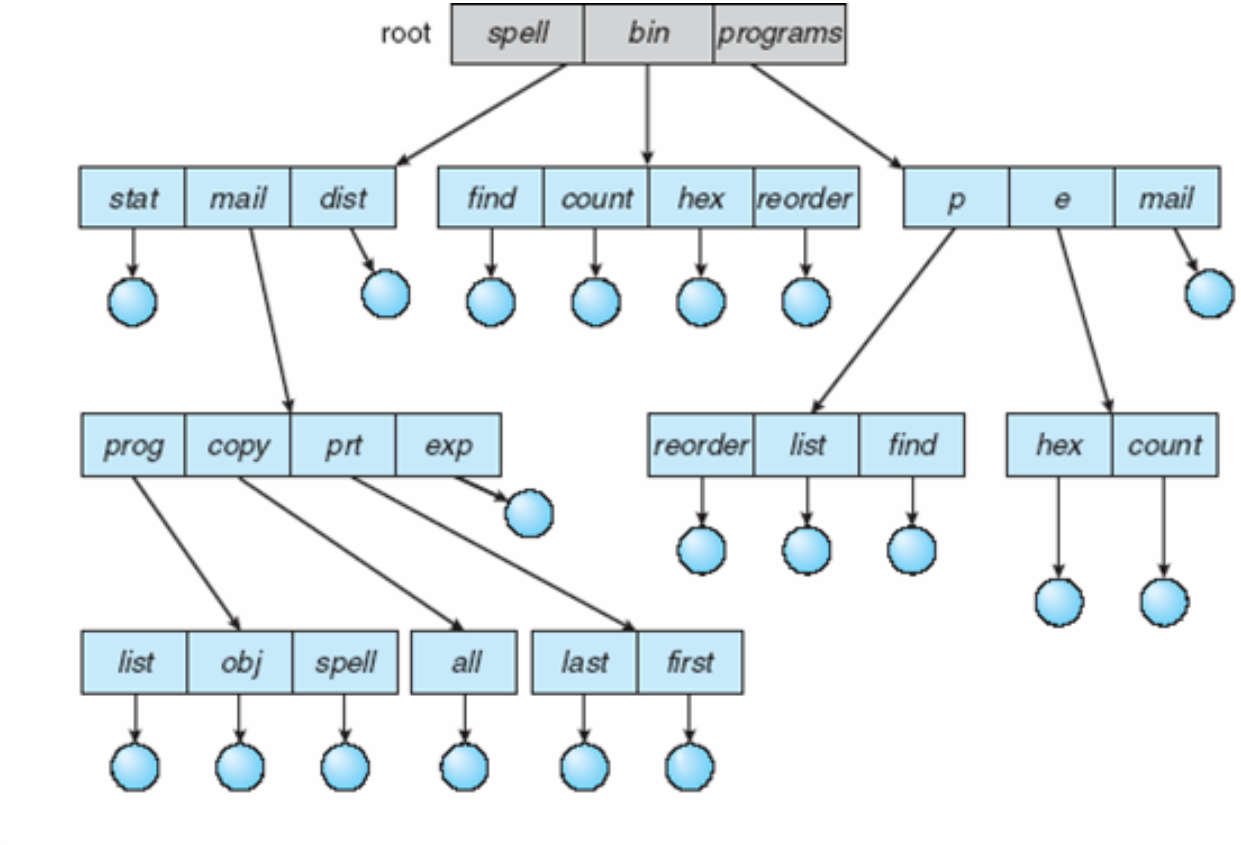
```
$ ls –i project2/
24242928 directory
25311615 dot_vimrc
25311394 linux-2.6.32.60.tar.gz
22148028 scheduling.html
25311610 kvm-kernel-build
22147399 project2.pdf
25311133 scheduling.pdf
25311604 kvm-kernel.config
25311612 reinstall-kernel
25311606 thread_runner.tar.gz
```

Inode number

Filename (or dirname)

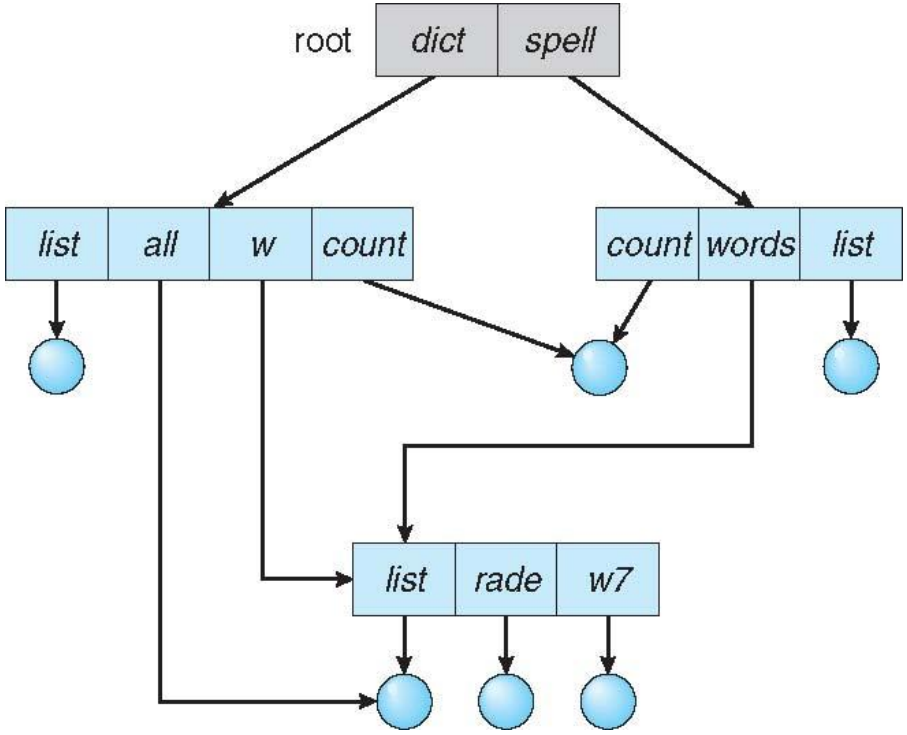
Directory Organization

- Typically a tree structure



Directory Organization

- Some filesystems support links → graph
  - Hard link: different names for a single file
  - Symbolic link: pointer to another file (“shortcut”)



Name Resolution

- Path
  - A unique name of a file or directory in a filesystem
    - E.g., /usr/bin/top
- Name resolution
  - Process of converting a path into an inode
  - How many disk accesses to resolve “/usr/bin/top”?

Name Resolution

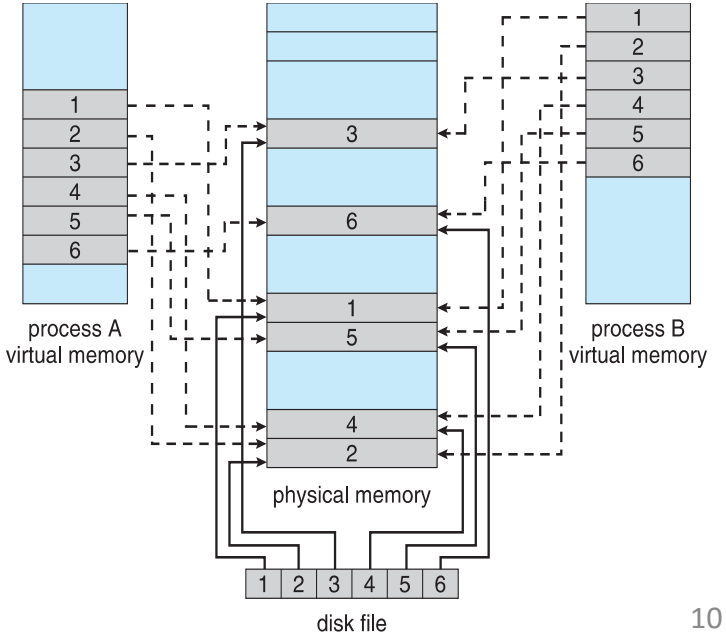
- How many disk accesses to resolve “/usr/bin/top”?
  - Read “/” directory inode
  - Read first data block of “/” and search “usr”
  - Read “usr” directory inode
  - Read first data block of “usr” and search “bin”
  - Read “bin” directory inode
  - Read first block of “bin” and search “top”
  - Read “top” file inode
  - Total 7 disk reads!!!
    - This is the minimum. Why? Hint: imagine 10000 entries in each directory

Directory Cache

- Directory name → inode number
  - Speedup name resolution process
    - When you first list a directory, it could be slow; next time you do, it would be much faster
  - Hashing
  - Keep only frequently used directory names in memory cache (how? LRU)

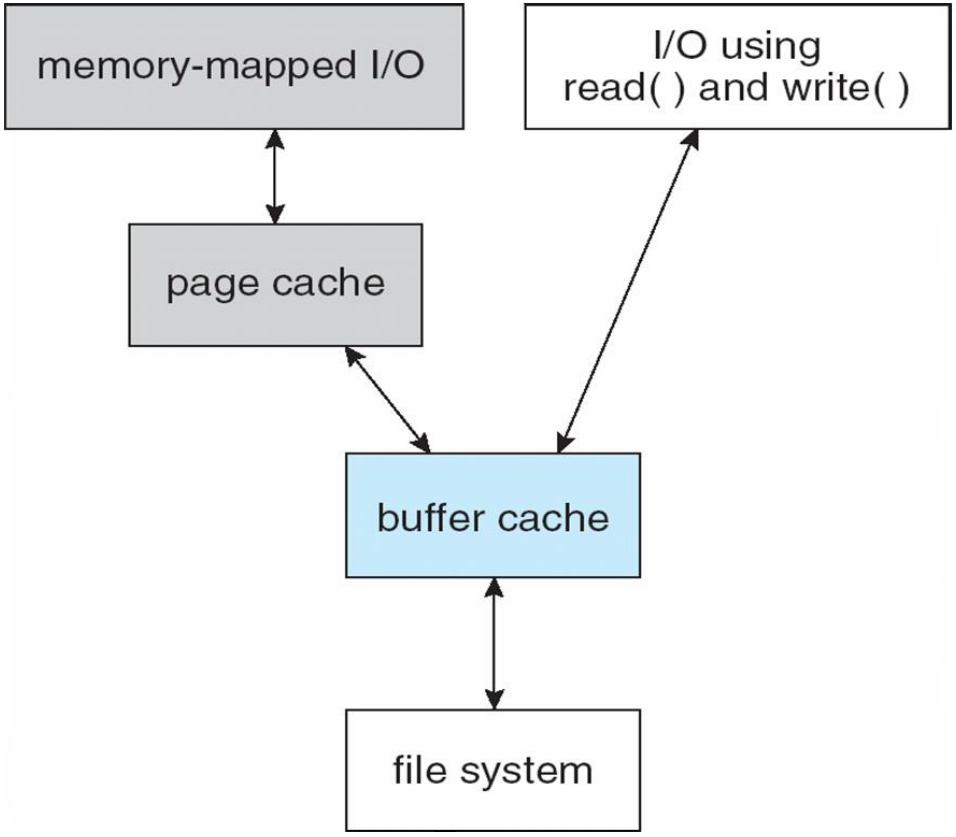
Filesystem Related Caches

- Buffer cache
  - Caching frequently accessed disk blocks
- Page cache
  - Remember memory mapped files?
  - Map pages to files using virtual memory

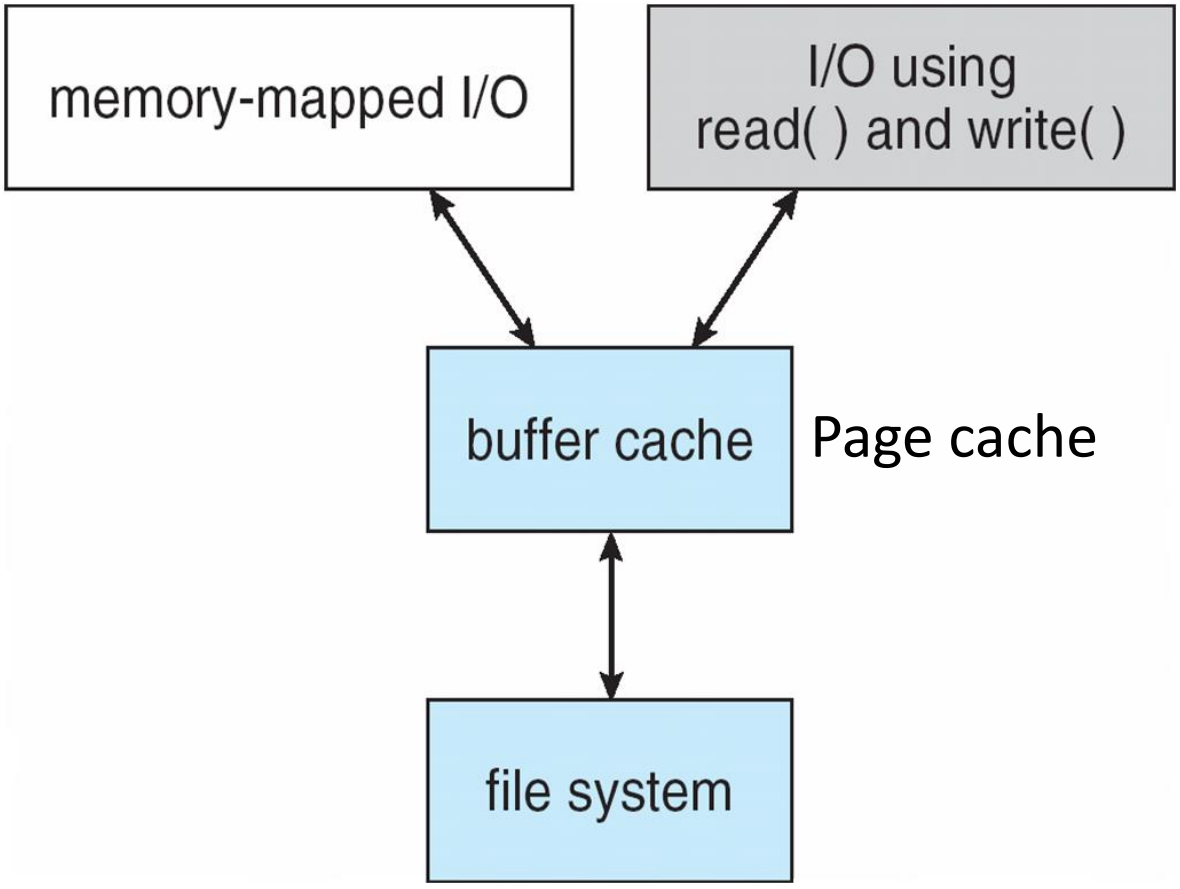


Non Unified Caches (Pre Linux 2.4)

- Problem: double caching

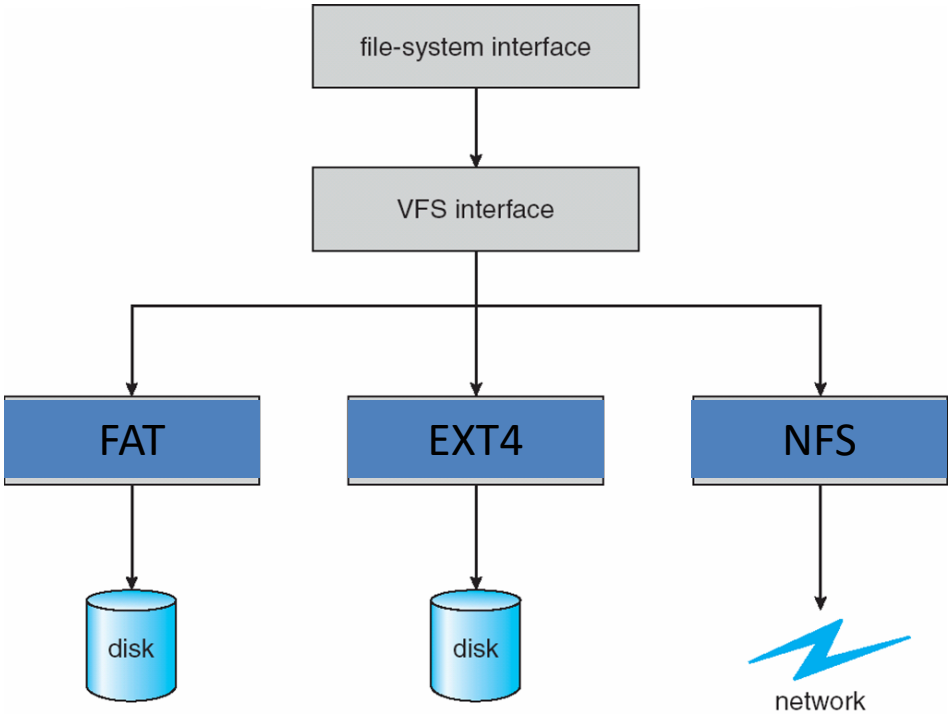


Unified Buffer Cache



Virtual Filesystem (VFS)

- Provides the same filesystem interface for different types of file systems



Virtual Filesystem (VFS)

- VFS defined APIs
  - **`int open(. . .)`**—Open a file
  - **`int close(. . .)`**—Close an already-open file
  - **`ssize t read(. . .)`**—Read from a file
  - **`ssize t write(. . .)`**—Write to a file
  - **`int mmap(. . .)`**—Memory-map a file
  - …
- All filesystems support the VFS apis

Storage System Layers (in Linux)