Operating Systems

Chapter 5 File Systems and Storage Media

Tien Pham Van, Dr. rer. nat.

Compiled with reference to other presentations

Outlines

- File Systems
- Disks
- RAID
- SSD File Systems

File Systems

Files

- Files provide mechanism for online storage of and access to both data and programs of the OS and all the users
- A typical file system consists of:
  – Files (each storing related data)
  – Metadata (directory structure)

Files

- Information can be stored on various storage media
- OS provides a uniform logical view of information storage
- OS abstracts from the physical properties of its storage devices to define a logical storage unit, FILE!
- Files are mapped by the OS onto physical devices

Files

- File is a named collection of related information that is recorded on a secondary storage
- Smallest allotment of logical secondary storage
- Data cannot be written on secondary storage unless it is written on a file
- File is a very general concept

File System

- Collection of Files

  File contains related data

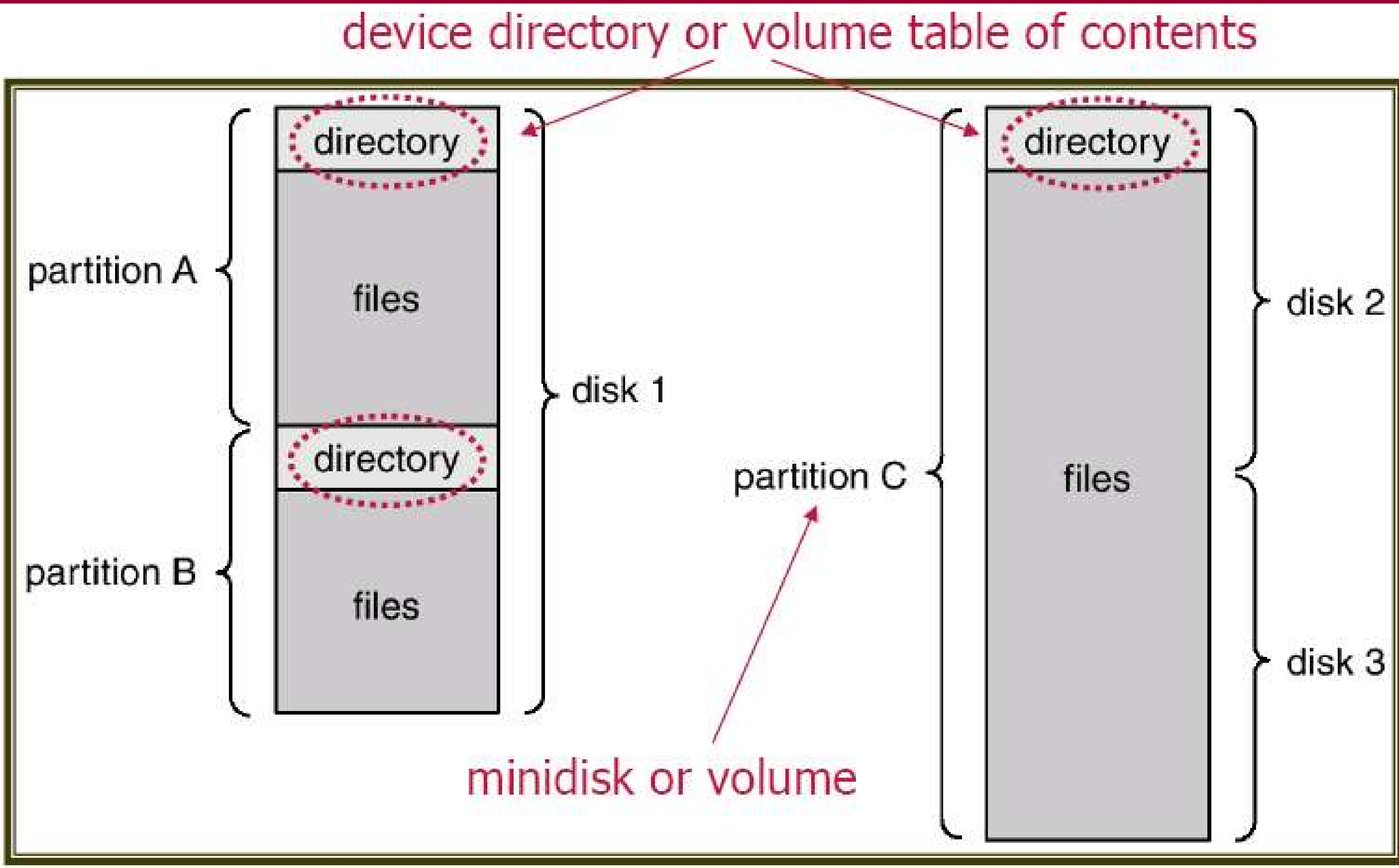
- Directory Structure

  Organizes & provides information about all files in the system

- Partitions

  Separates physically or logically large collections of directories

Typical File System



=> Every partition has a file system, which consists of directory and files

File Attributes

- Name
- Identifier (generally a number)
- Type
- Location (device: location)
- Size (bytes, words or blocks)
- Protection (rwx)
- Time, date, & user identification
  (creation, last modification, last use)

File Operations

- File is an ADT
- Must define operations on files
- File Operations
  - Creating
  - Writing
  - Reading
  - Repositioning within file (file seek)
  - Deleting
  - Truncating
- OS provides system calls for each operation

File Locking Example – Java API

```
import java.io.*;
import java.nio.channels.*;
public class LockingExample {
	public static final boolean EXCLUSIVE = false;
	public static final boolean SHARED = true;
	public static void main(String arsg[]) throws IOException {
		FileLock sharedLock = null;
		FileLock exclusiveLock = null;
		try {
			RandomAccessFile raf = new RandomAccessFile("file.txt", "rw");
			// get the channel for the file
			FileChannel ch = raf.getChannel();
			// this locks the first half of the file - exclusive
			exclusiveLock = ch.lock(0, raf.length()/2, EXCLUSIVE);
			/** Now modify the data . . . */
			// release the lock
			exclusiveLock.release();
```

File Locking Example – Java API (Cont.)

```
            // this locks the second half of the file - shared
            sharedLock = ch.lock(raf.length()/2+1, raf.length(),
                    SHARED);
            /** Now read the data . . . */
            // release the lock
            sharedLock.release();
        } catch (java.io.IOException ioe) {
            System.err.println(ioe);
        }finally {
            if (exclusiveLock != null)
            exclusiveLock.release();
            if (sharedLock != null)
            sharedLock.release();
        }
    }
}
```

File Types

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

Directory Structure

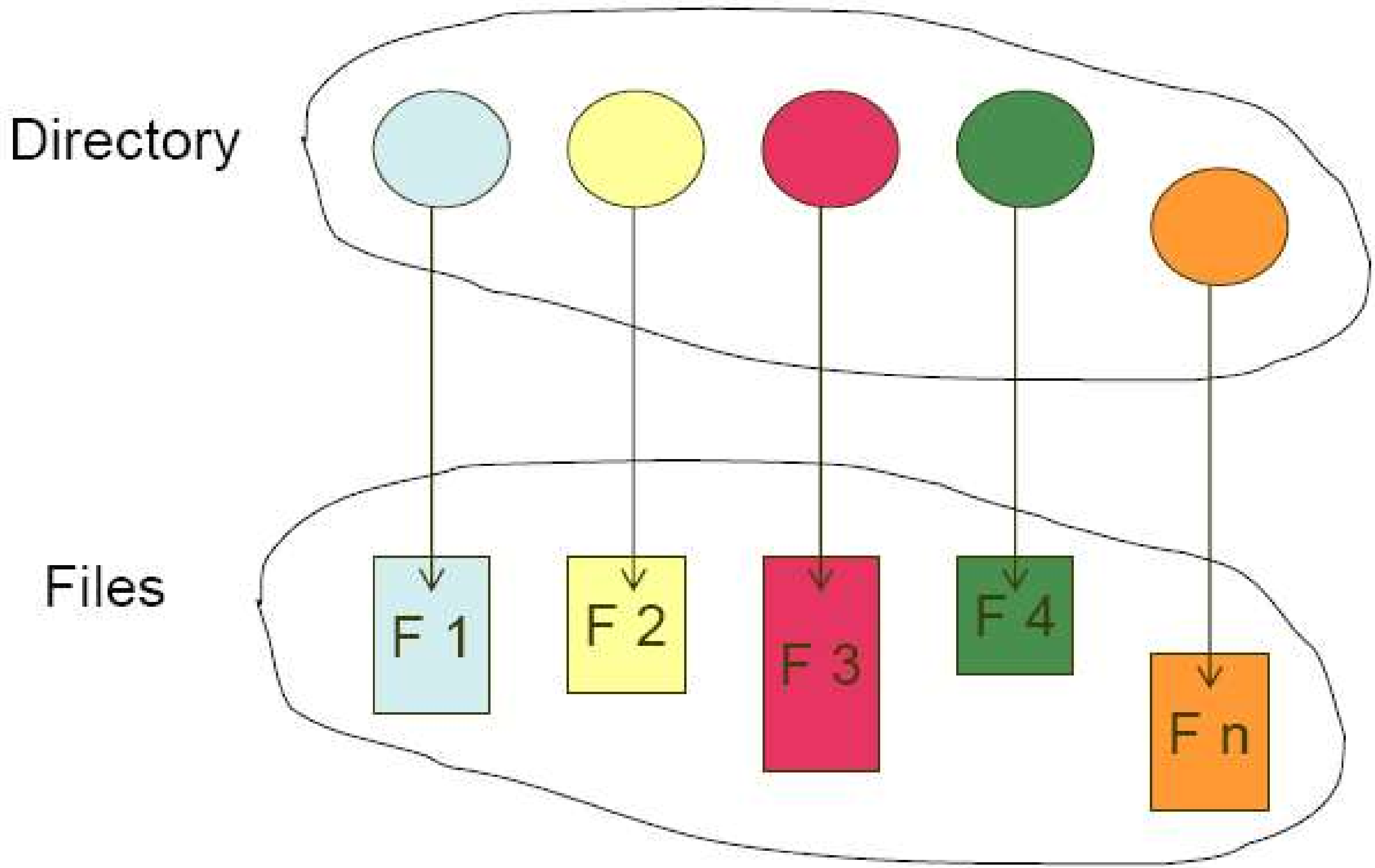
- Stores information about all files
- Resides on secondary storage
- Contains file’s name & Identifier
- Identifier in turn locates other file attributes
- More than 1KB of information for each file
- Directory structure size in MBs
- Backups of files and directory structure kept on tapes

Directory Structure

- Disks are split into one or more partitions
- Minidisks or Volumes
- Each disk contains atleast one partition
- Partitions can be larger than a disk
- Partitions are “*virtual disks*”
- Partition maintains information about files in “*Device Directory*”

Directory Structure

A collection of nodes containing information about all files.



Information in a Directory

- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed
- Date last updated
- Owner ID
- Protection information

Directory Operations

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

How to Organize a Directory?

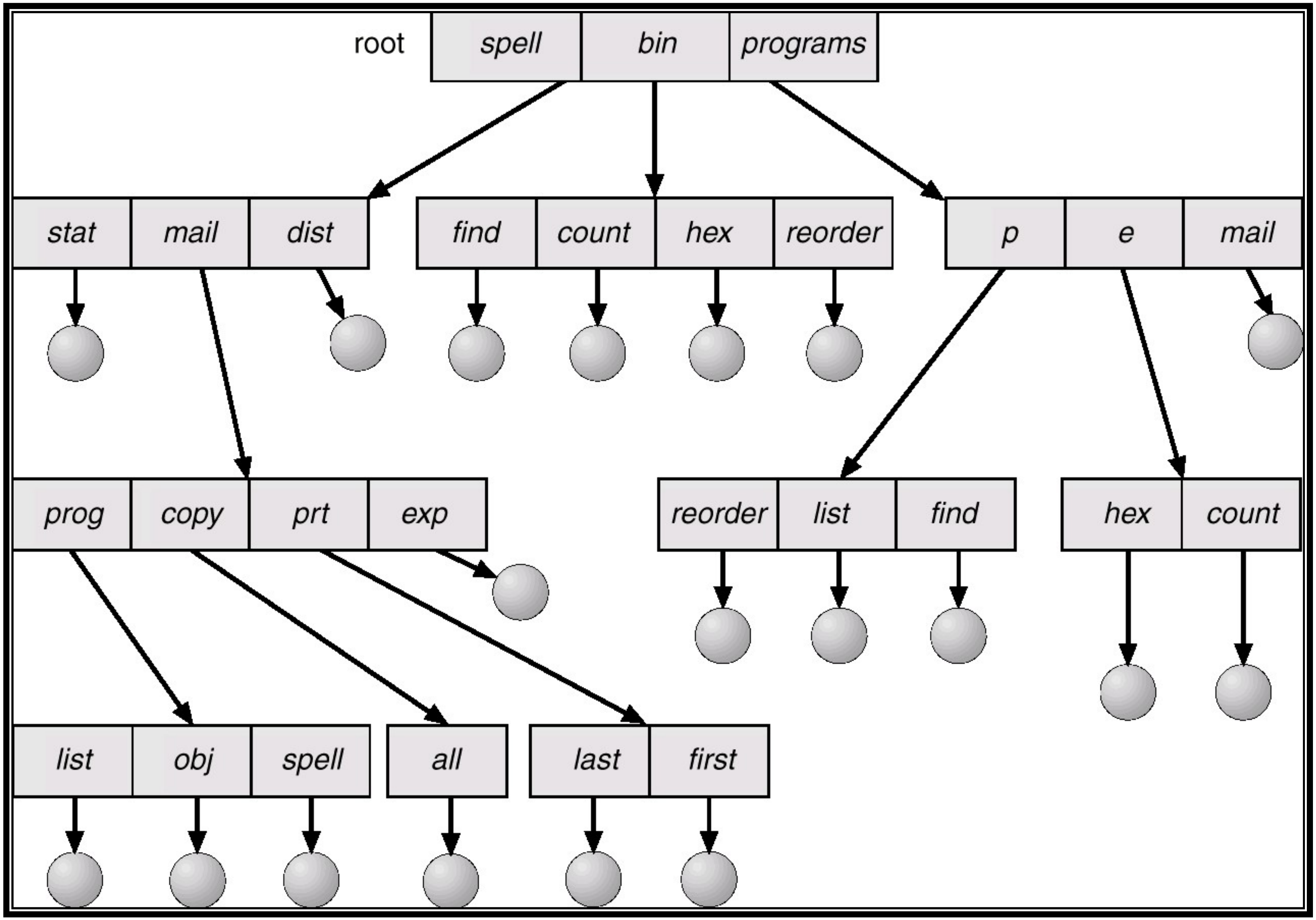
Issues

- 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, …)

Schemes for defining the logical structure of a directory:

- Single level directory
- Two level directory
- Tree structured directory
- Acyclic graph directory
- General graph directory

Tree-Structured Directory



Tree-Structured Directory

- Generalization of two-level directory (with arbitrary height)
- Each user has a current directory (working directory)
  - Can change current directory via cd command or system call
- Path names can be absolute or relative

Operations

- 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>

Advantages

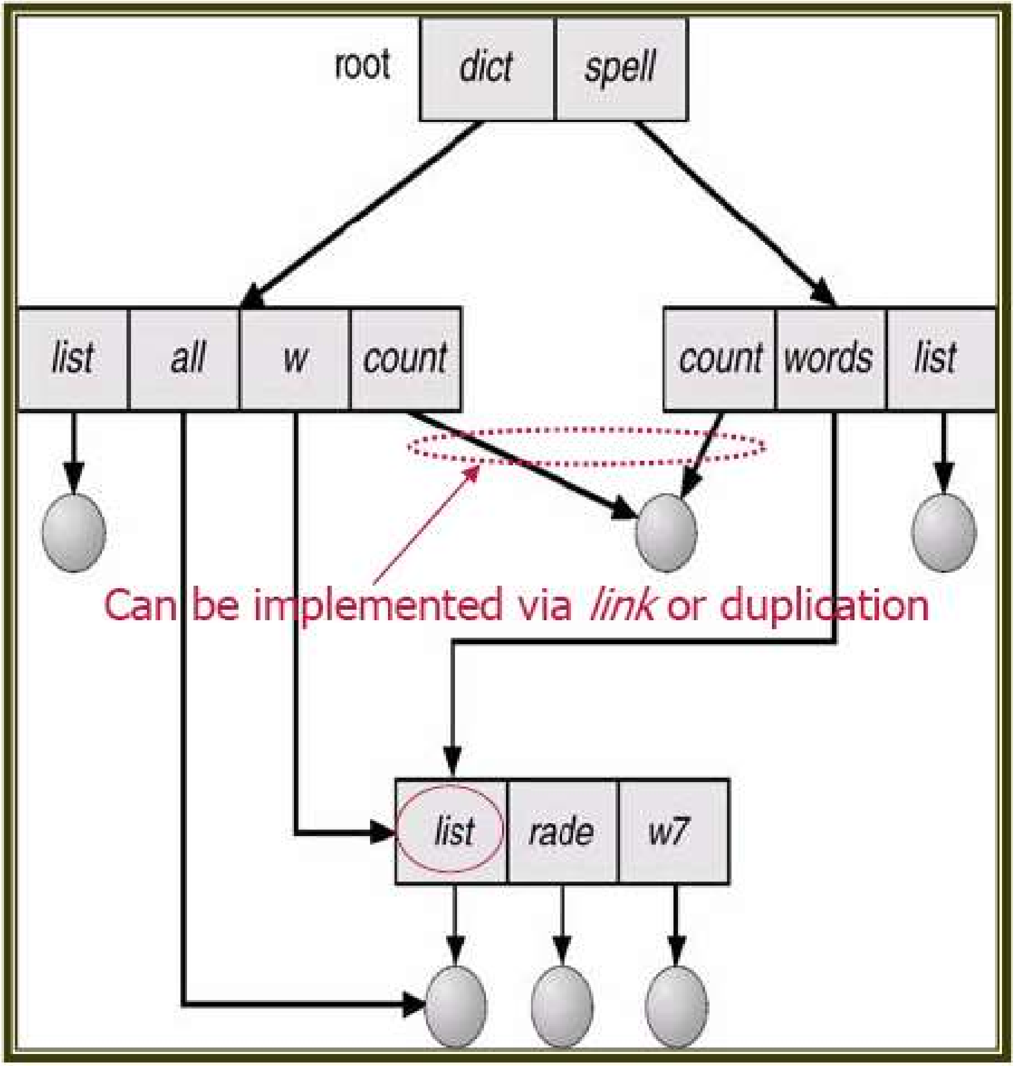
- Efficient searching
- Grouping Capability

Acyclic-Graph Directory

- 2 programmers working on a joint project
- File pertaining to project stored in a subdirectory of each user
- Shared subdirectory!
- Tree structure prohibits the sharing of files or directories!
- A shared file or dir will exist in the file system in two or more places at once
- AGD allows files & dirs to be shared

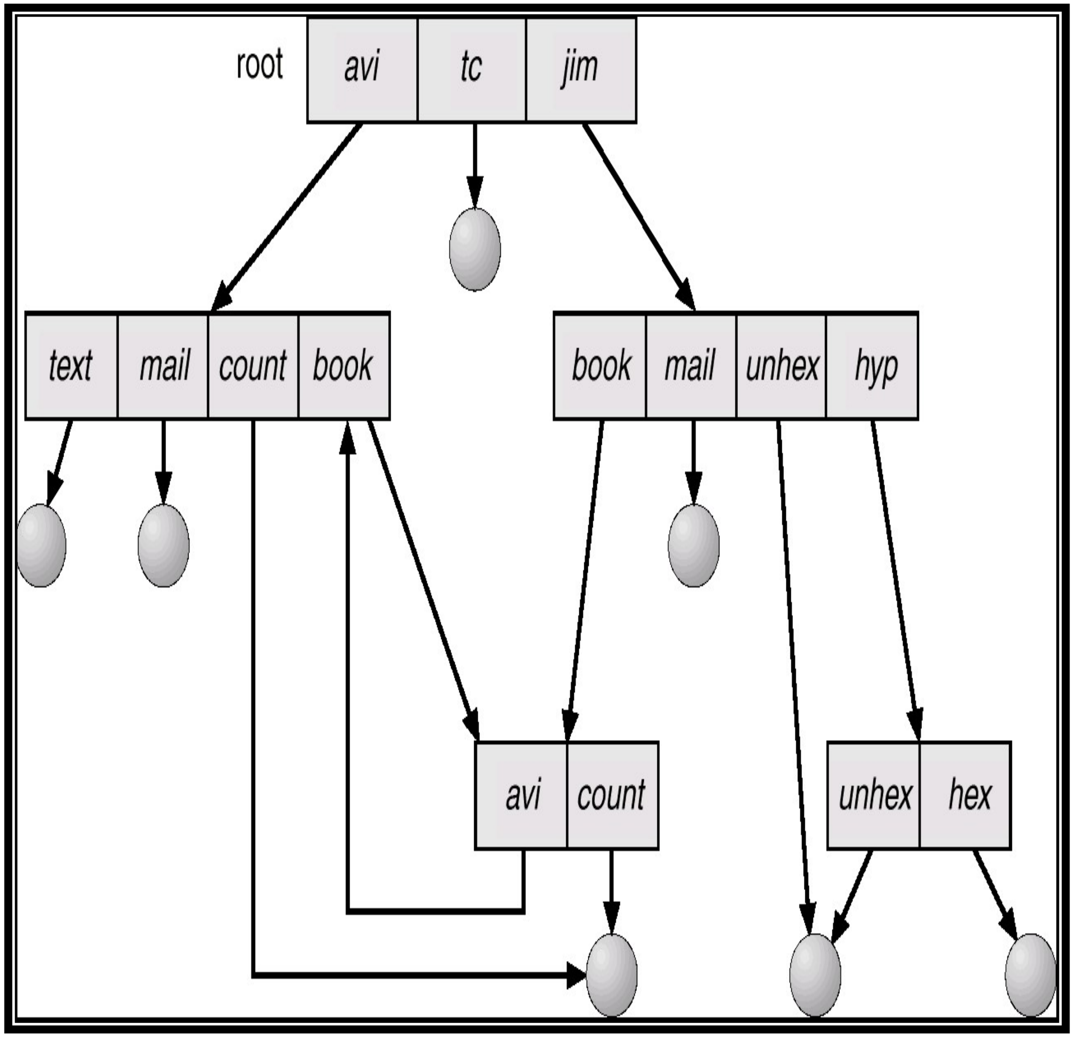
Acyclic-Graph Directory

- Have shared subdirectories and files using acyclic graph
- Two different names exist (aliasing problem)
- If ‘dict’ deletes list ⇒ dangling pointer (UNIX case)
- Solutions:
  - Preserve file until all references to it are deleted
    - Reference count solution



General-Graph Directory

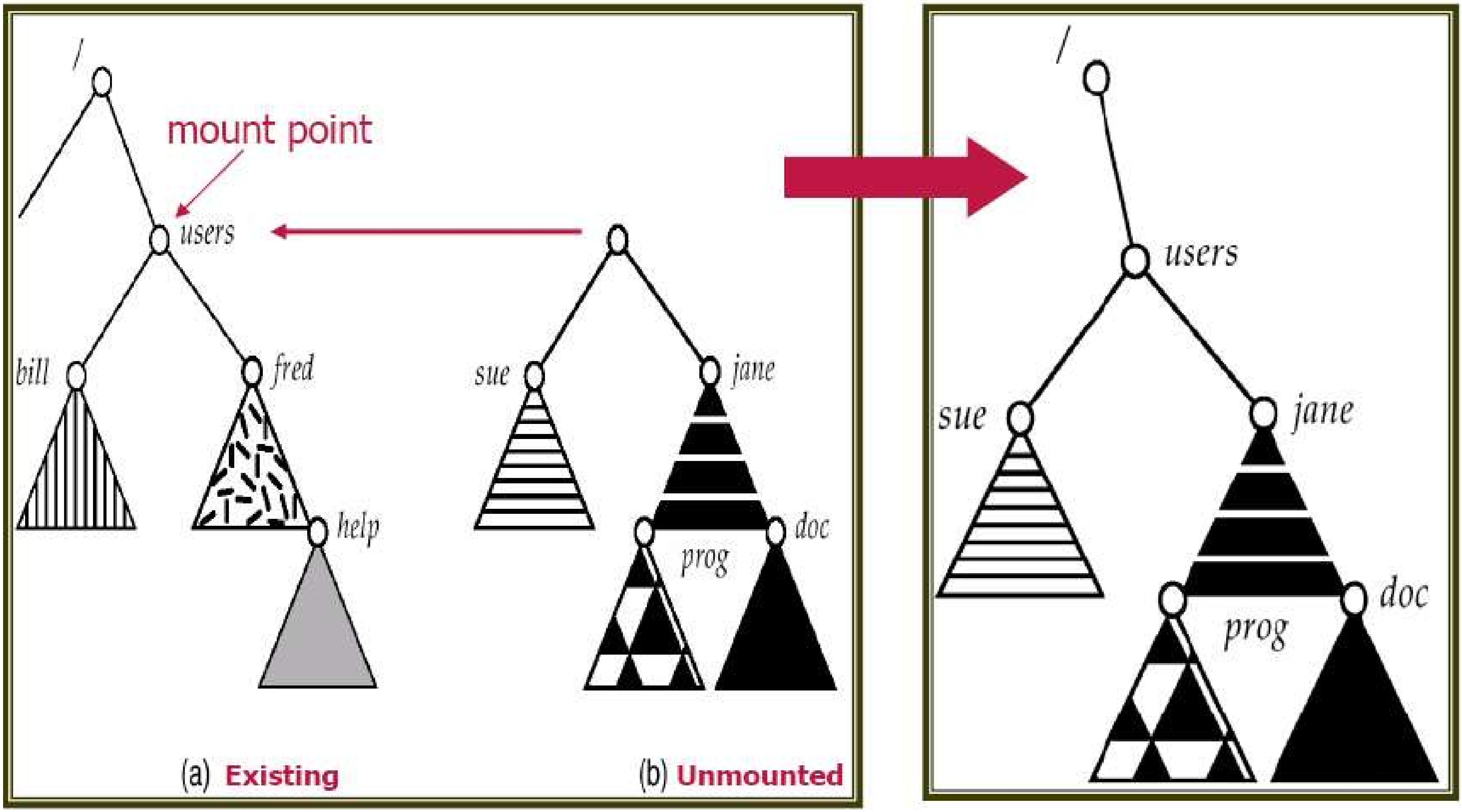
- Add links to an existing tree structured directory, resulting in a simple graph structure
- But, once we allow links to be added to an existing tree structure, the tree structure is destroyed, resulting in a simple graph structure



File System Mounting

- Just as a file must be opened before it is used, a file system must be mounted before it can be accessed
- Procedure?

File System Mounting



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

File Sharing: Remote File Systems

- Uses networking to allow file system access between systems
  - Manually via programs like FTP
  - Automatically, seamlessly using distributed file systems
  - Semi automatically via the world wide web
- Client-server model allows clients to mount remote file systems from servers
  - Server can serve multiple clients
  - Client and user-on-client identification is insecure or complicated
  - NFS is standard UNIX client-server file sharing protocol
  - CIFS is standard Windows protocol
  - Standard operating system file calls are translated into remote calls
- Distributed Information Systems (distributed naming services) such as LDAP, DNS, NIS, Active Directory implement unified access to information needed for remote computing

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 Control List (ACL)

- Various users may need different types of access to a file or directory
- The most general scheme is to associate with each file & directory an access-control list (ACL) specifying the user name & the types of access allowed for each user
- Advantages
  - Enable complex access methodology
- Disadvantages
  - Constructing such a list may be a tedious and unrewarding task, especially if we do not know in advance the list of users in the system
  - The directory entry, previously of fixed size, now needs to be of variable size, resulting in more complicated space management

Access Lists & Groups (UNIX)

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

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

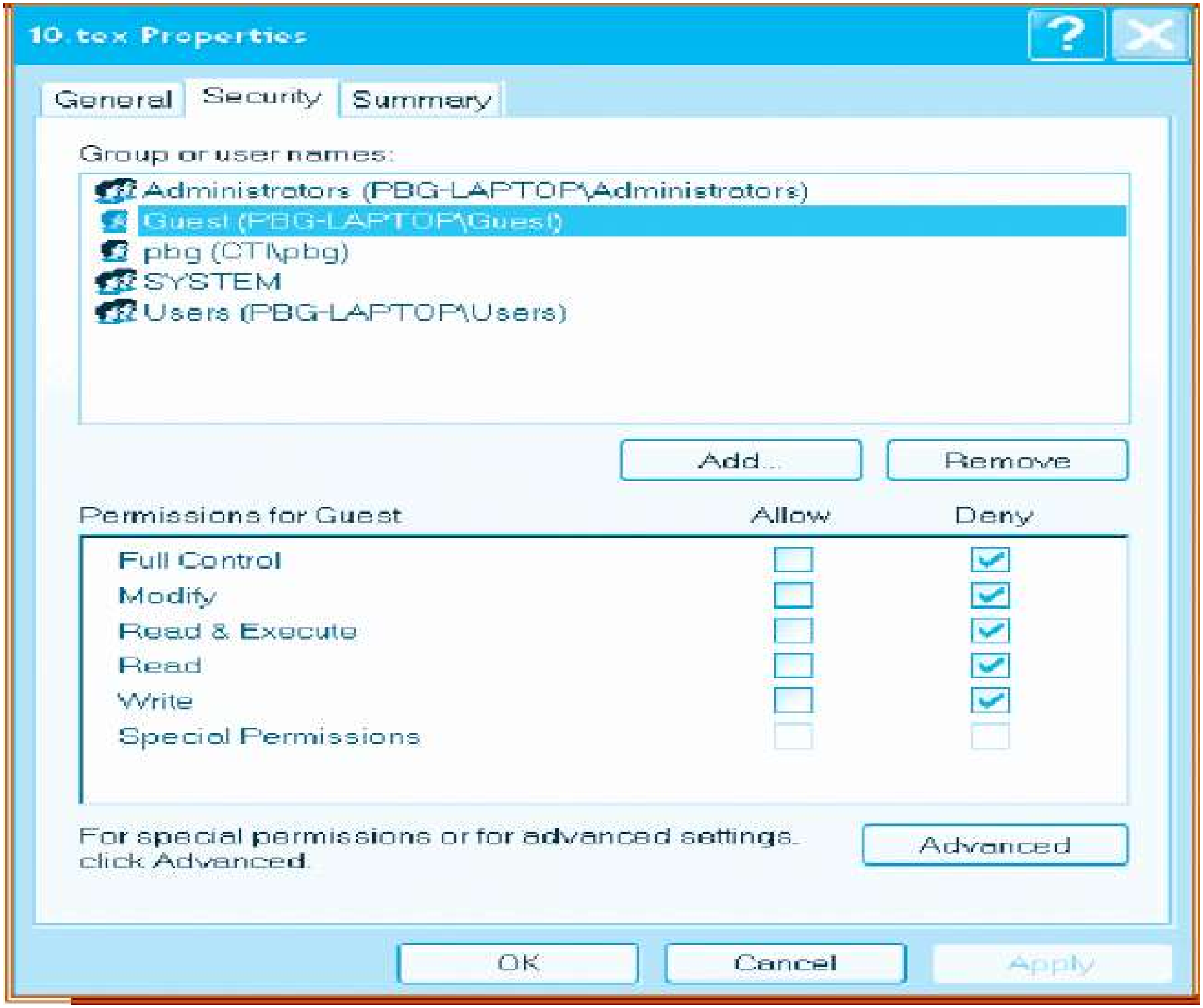
- Ask manager to create a group (unique name), say G, and add some users to the group
- For a particular file (say *game*) or subdirectory, define an appropriate access

owner group public

chmod 761 game

- Attach a group to a file using *chgrp G game*

Windows XP Access-control List Management



UNIX File System

- The filesystem is your interface to
  - physical storage (disks) on your machine
  - storage on other machines
  - output devices
  - etc.
- *Everything* in UNIX is a file (programs, text, peripheral devices, terminals, …)
- There are no drive letters in UNIX! The filesystem provides a *logical* view of the storage devices

Some Standard Directories

- / - root directory
- /bin – Home of all binaries
- /dev – device directory
- /etc – host-specific files and directories
- /home – users home directories
  - /home/grads/sgifford
- /lib – libraries for various languages
- /sbin – System administration utilities, tools,…
- /tmp – temporary files
- /var – Variable data that is changing

Output of ls

```
total 4
lrwxr-xr-x 1 user1 user 18 Aug 28 13:41 home ->
   /usr/people/bowman/
-rw-r--r-- 1 user1 user 94 Aug 28 13:42 nothing.txt
drwxr-xr-x 2 user1 user  9 Aug 28 13:40 test_dir/
```

Permissions Owner Group Modify date Filename

File type Size in bytes

- We'll keep coming back to this slide!

File Ownership

- Each file has a single owner
- `chown` command can be used to change the owner (usually only root user can use this command)
- There are also various *groups* to which users can belong
- Groups may have different permissions than everyone else

File Permissions

- Permissions used to allow/disallow access to file/directory contents
- Read (r), write (w), and execute (x)
- For owner, group, and world (everyone)
- `chmod <mode> <file(s)>`
  - `chmod 700 filetxt`
  - `chmod g+rw filetxt`

Typical File Systems

- [Explaining File Systems: NTFS, exFAT, FAT32, ext4 & More – YouTube](https://www.youtube.com/watch?v=_h30HBYxtws&list=PLNPBkTc2yReGb8ue-H00NDZrsSTl-ChlS&index=174)

https://www.youtube.com/watch?v=_h30HBYxtws&list=PLNPBkTc2yReGb8ue-H00NDZrsSTl-ChlS&index=174

Magnetic Disks

Disks Disk Hardware

Parameter	IBM 360-KB floppy disk	WD 18300 hard disk
Number of cylinders	40	10601
Tracks per cylinder	2	12
Sectors per track	9	281 (avg)
Sectors per disk	720	35742000
Bytes per sector	512	512
Disk capacity	360 KB	18.3 GB
Seek time (adjacent cylinders)	6 msec	0.8 msec
Seek time (average case)	77 msec	6.9 msec
Rotation time	200 msec	8.33 msec
Motor stop/start time	250 msec	20 sec
Time to transfer 1 sector	22 msec	17 μsec

Disk parameters for the original IBM PC floppy disk and a Western Digital WD 18300 hard disk

Disk Structure

- Disk drives are addressed as large 1-dimensional arrays of *logical blocks*, where the logical block is the smallest unit of transfer.
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially.
  - Sector 0 is the first sector of the first track on the outermost cylinder.
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.

Disk Performance Parameters

- To read or write, the disk head must be positioned at the desired track and at the beginning of the desired sector
- Seek time
  - Time it takes to position the head at the desired track
- Rotational delay or rotational latency
  - Time its takes for the beginning of the sector to reach the head

Timing of a Disk I/O Transfer

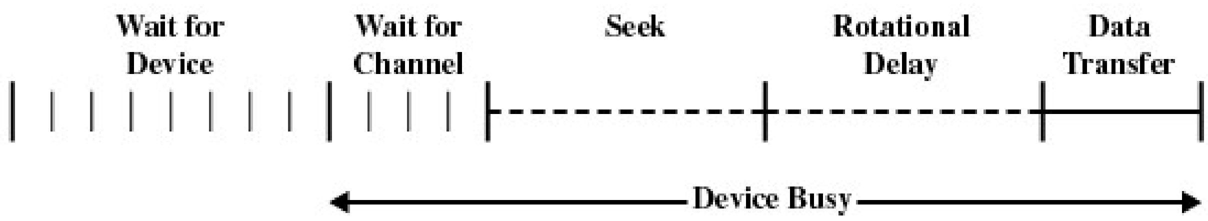


Figure 11.6 Timing of a Disk I/O Transfer

Disk Performance Parameters

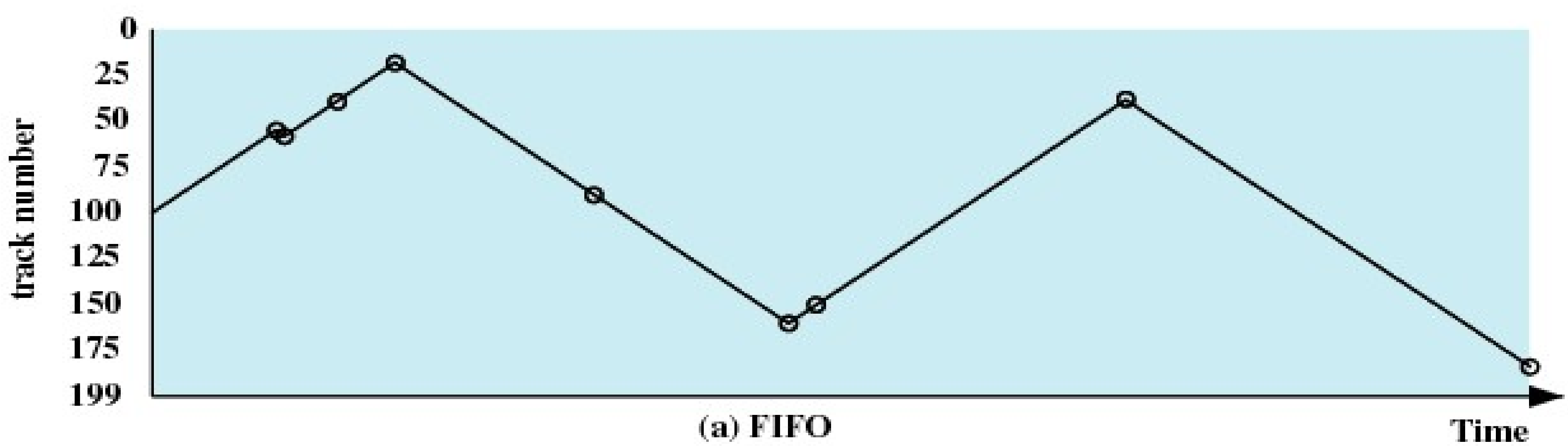
- Access time
  - Sum of seek time and rotational delay
  - The time it takes to get in position to read or write
- Data transfer occurs as the sector moves under the head

Disk Scheduling Policies

- Seek time is the reason for differences in performance
- For a single disk there will be a number of I/O requests
- If requests are selected randomly, we will poor performance

Disk Scheduling Policies

- First-in, first-out (FIFO)
  - Process request sequentially
  - Fair to all processes
  - Approaches random scheduling in performance if there are many processes



(a) FIFO

Embedded Networking Research Group
Email: tien.phamvan1@hust.edu.vn
School of Elec. and Telecom - Hanoi University of Science and Technology
C9-411, Dai Co Viet str. 1, HBT, Hanoi
Tel: +84-243-8693596

Disk Scheduling Policies

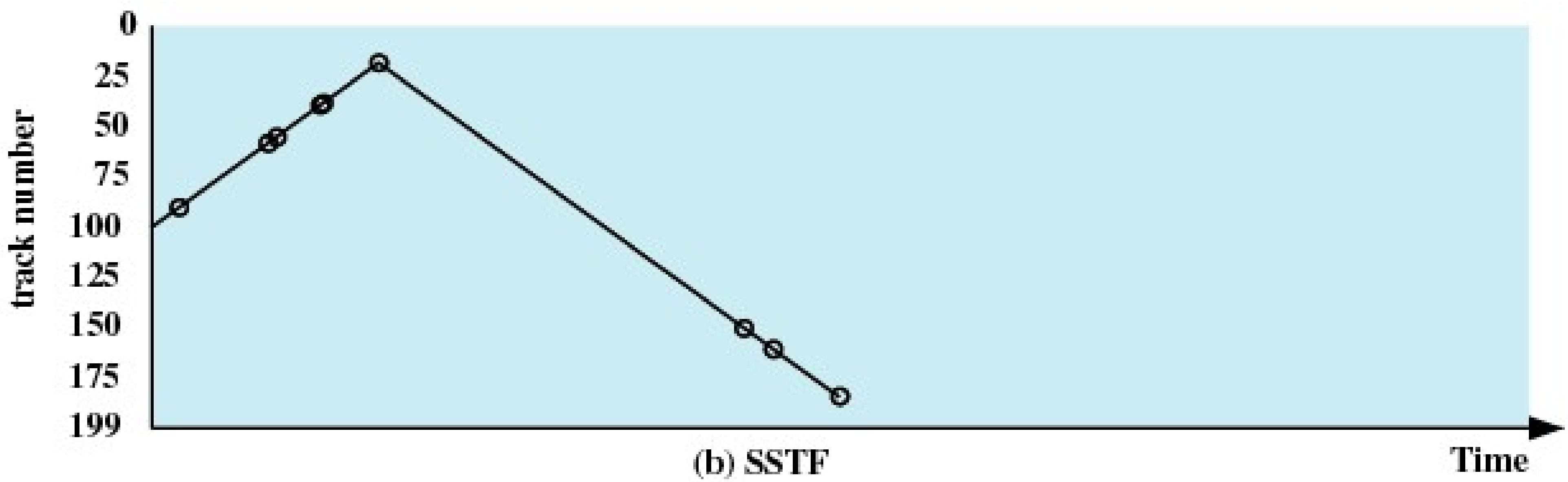
- Priority
  - Goal is not to optimize disk use but to meet other objectives
  - Short batch jobs may have higher priority
  - Provide good interactive response time

Disk Scheduling Policies

- Last-in, first-out
  - Good for transaction processing systems
    - The device is given to the most recent user so there should be little arm movement
  - Possibility of starvation since a job may never regain the head of the line

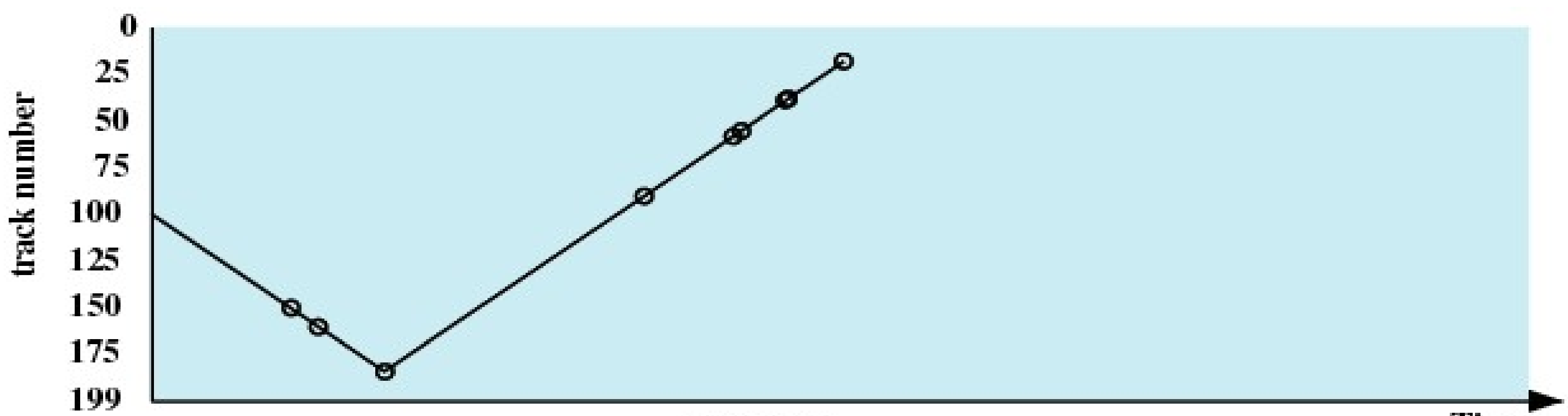
Disk Scheduling Policies

- Shortest Service Time First
  - Select the disk I/O request that requires the least movement of the disk arm from its current position
  - Always choose the minimum Seek time



Disk Scheduling Policies

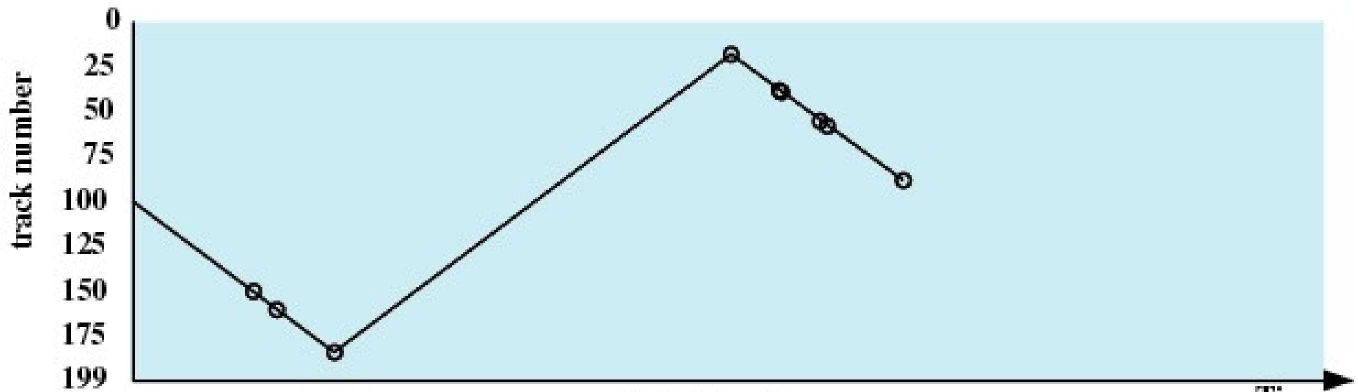
- SCAN
  - Arm moves in one direction only, satisfying all outstanding requests until it reaches the last track in that direction
  - Direction is reversed



(c) SCAN

Disk Scheduling Policies

- C-SCAN
  - Restricts scanning to one direction only
  - When the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again



(d) C-SCAN

Embedded Networking Research Group
Email: tien.phamvan1@hust.edu.vn
School of Elec. and Telecom - Hanoi University of Science and Technology
C9-411, Dai Co Viet str. 1, HBT, Hanoi
Tel: +84-243-8693596

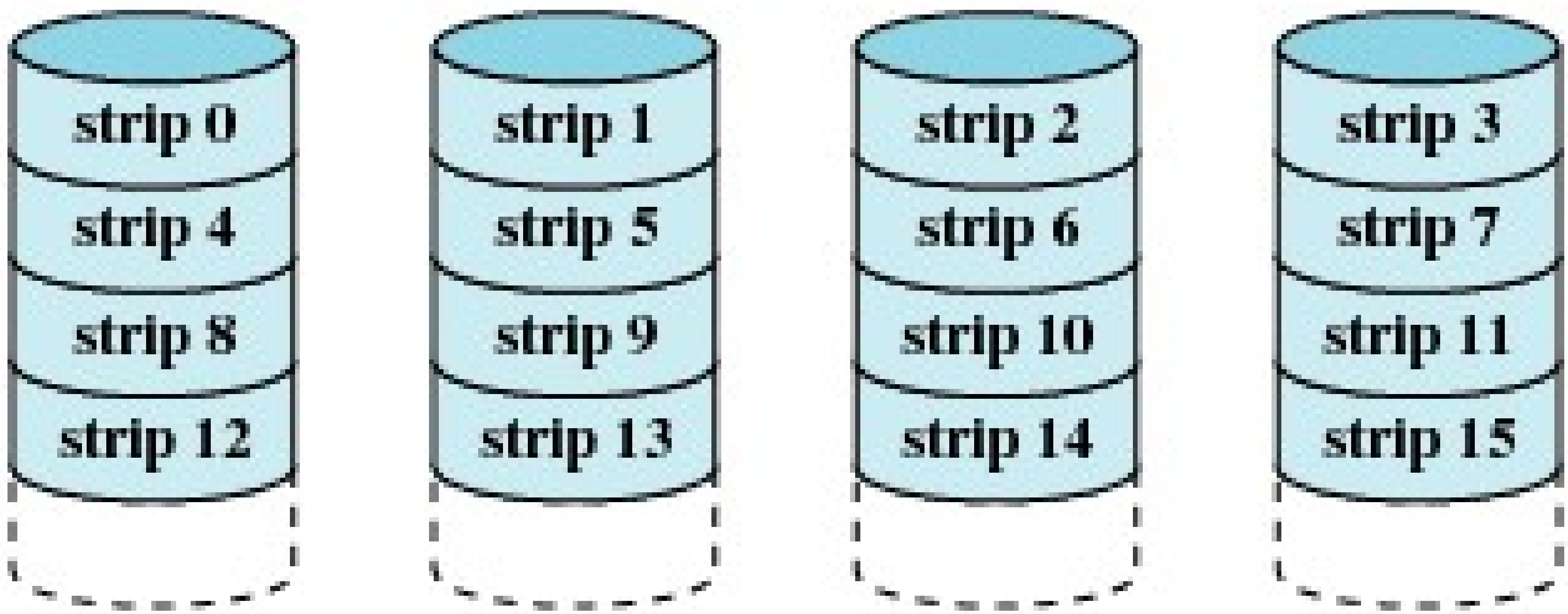
Disk Scheduling Policies

- N-step-SCAN
  - Segments the disk request queue into subqueues of length N
  - Subqueues are processed one at a time, using SCAN
  - New requests added to other queue when queue is processed
- FSCAN
  - Two queues
  - One queue is empty for new requests

RAID

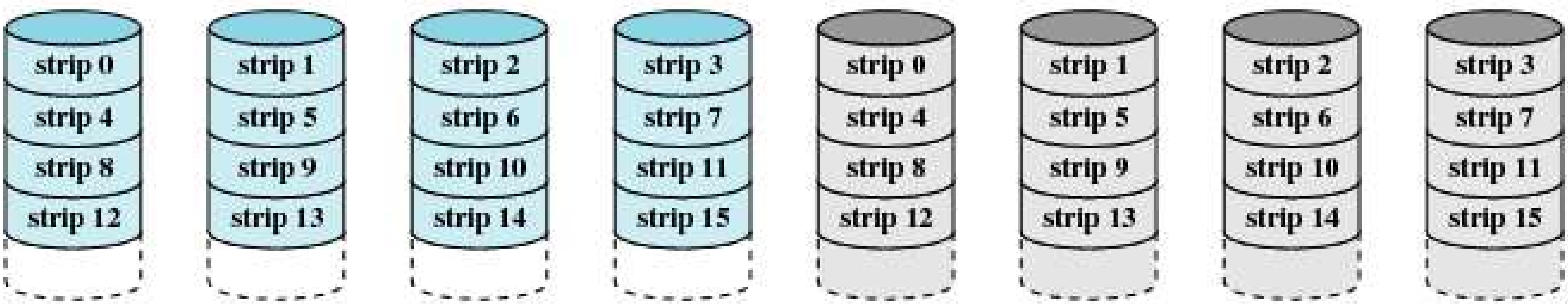
- Redundant Array of Independent Disks
- Set of physical disk drives viewed by the operating system as a single logical drive
- Data are distributed across the physical drives of an array
- Redundant disk capacity is used to store parity information

RAID 0 (non-redundant)



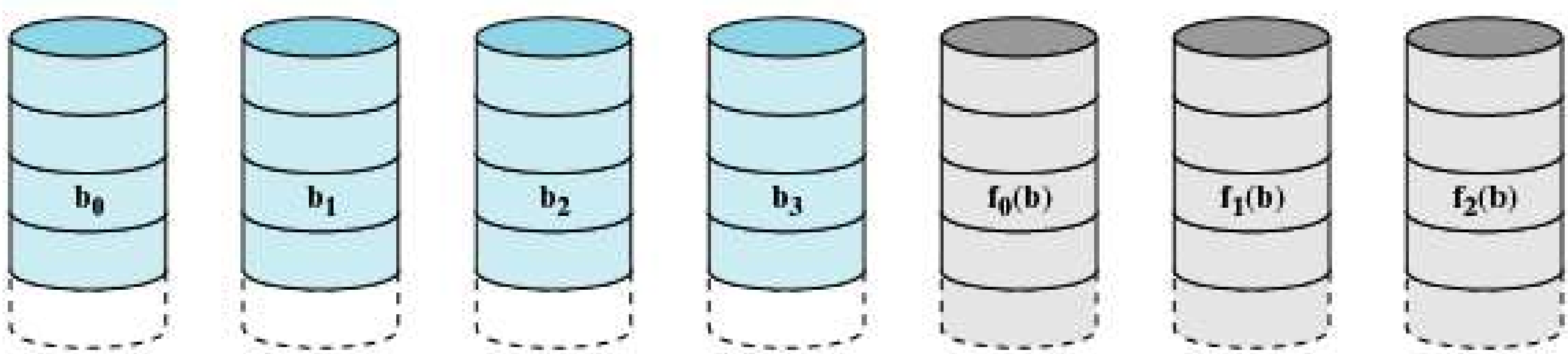
(a) RAID 0 (non-redundant)

RAID 1 (mirrored)



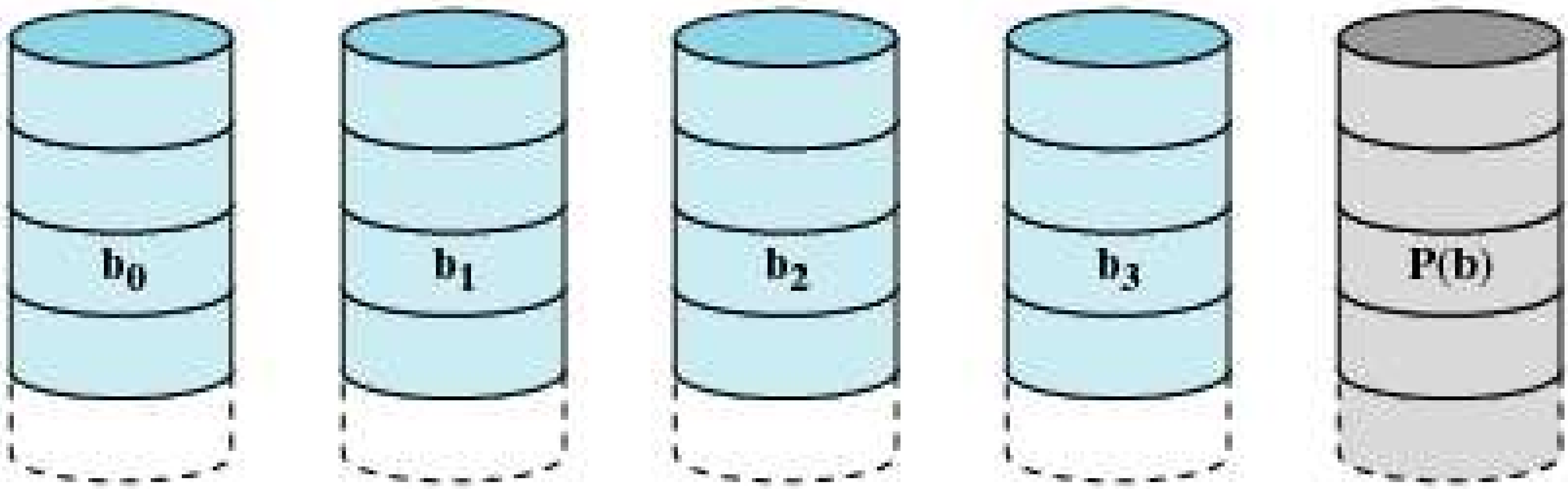
(b) RAID 1 (mirrored)

RAID 2 (redundancy through Hamming code)



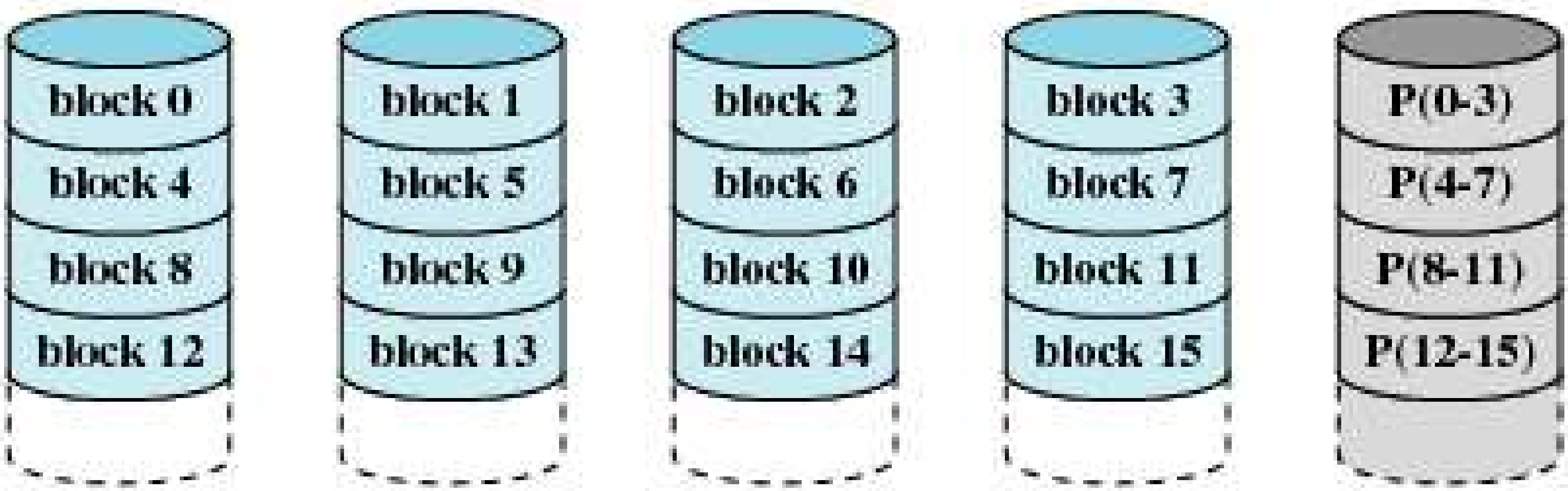
(c) RAID 2 (redundancy through Hamming code)

RAID 3 (bit-interleaved parity)



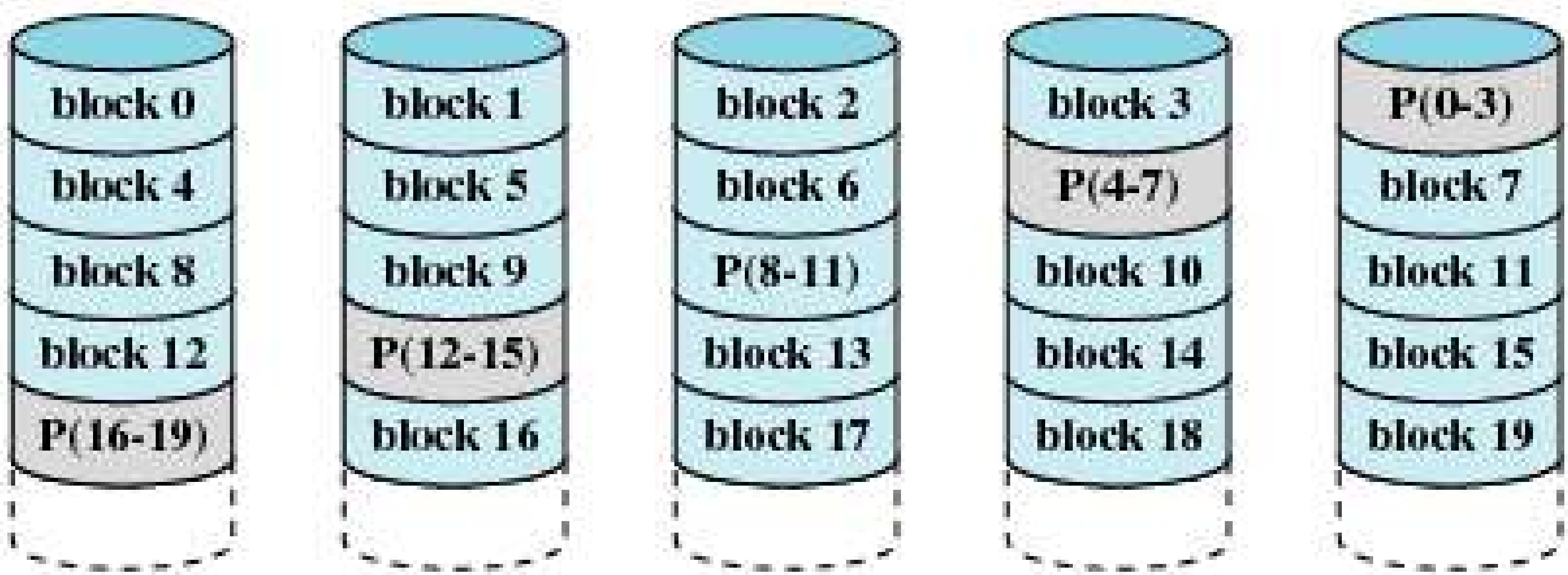
(d) RAID 3 (bit-interleaved parity)

RAID 4 (block-level parity)



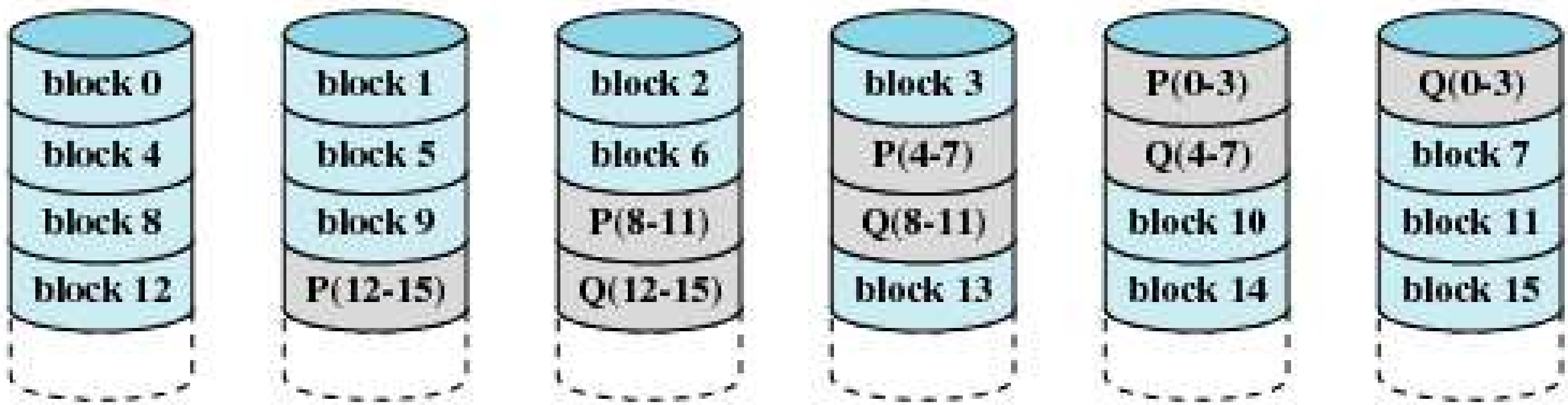
(e) RAID 4 (block-level parity)

RAID 5 (block-level distributed parity)



(f) RAID 5 (block-level distributed parity)

RAID 6 (dual redundancy)

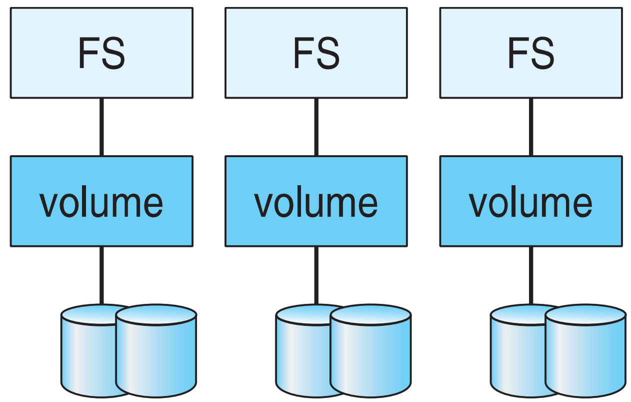


(g) RAID 6 (dual redundancy)

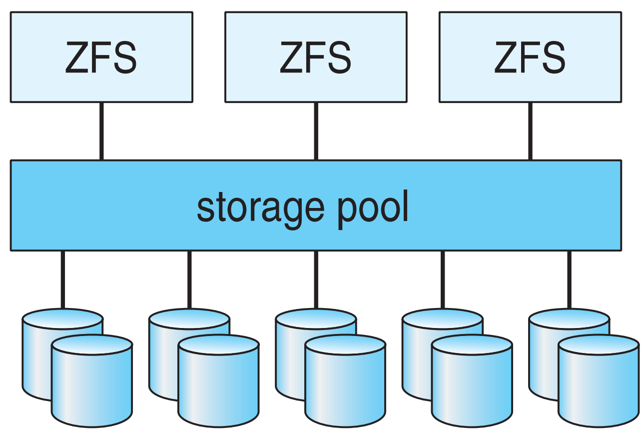
Extensions

- Solaris ZFS (Zettabyte (ZFS) file systems) adds **checksums** of all data and metadata
- Checksums kept with pointer to object, to detect if object is the right one and whether it changed
- Can detect and correct data and metadata corruption
- ZFS also removes volumes, partitions
  – Disks allocated in **pools**
  – Filesystems with a pool share that pool, use and release space like **`malloc()`** and **`free()`** memory allocate / release calls

Traditional and Pooled Storage



(a) Traditional volumes and file systems.



(b) ZFS and pooled storage.

Stable-Storage Implementation

- Write-ahead log scheme requires stable storage
- Stable storage means data is never lost (due to failure, etc)
- To implement stable storage:
  - Replicate information on more than one nonvolatile storage media with independent failure modes
  - Update information in a controlled manner to ensure that we can recover the stable data after any failure during data transfer or recovery
- Disk write has 1 of 3 outcomes
  1. **Successful completion -** The data were written correctly on disk
  2. **Partial failure -** A failure occurred in the midst of transfer, so only some of the sectors were written with the new data, and the sector being written during the failure may have been corrupted
  3. **Total failure -** The failure occurred before the disk write started, so the previous data values on the disk remain intact

Stable-Storage Implementation (Cont.)

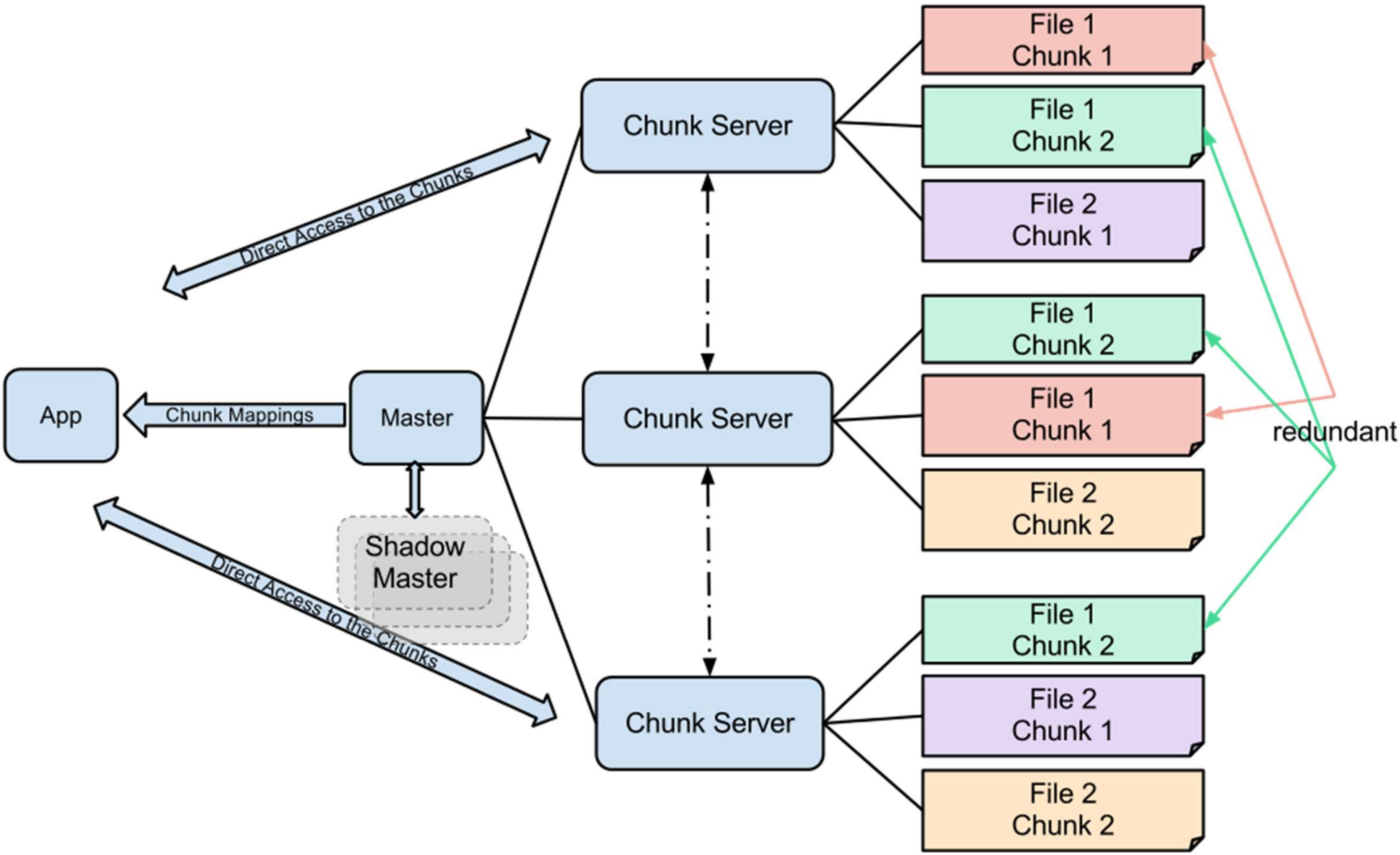
- If failure occurs during block write, recovery procedure restores block to consistent state
  - System maintains 2 physical blocks per logical block and does the following:
    1. Write to 1st physical
    2. When successful, write to 2nd physical
    3. Declare complete only after second write completes successfully

Systems frequently use NVRAM as one physical to accelerate

GFS – Google File Systems

- Used by Google
- A proprietary distributed file system developed by Google to provide efficient, reliable access to data using large clusters of commodity hardware
- Designed for system-to-system interaction, not for user-to-system

Chunk servers to replicate data automatically



GFS - operations

- https://www.youtube.com/watch?v=W-S0p_amT2A

Further reading

- Storage Area Network
- Introduction: https://www.youtube.com/watch?v=prkPpAPm4lA

Disk Cache

- Buffer in main memory for disk sectors
- Contains a copy of some of the sectors on the disk

Least Recently Used

- The block that has been in the cache the longest with no reference to it is replaced
- The cache consists of a stack of blocks
- Most recently referenced block is on the top of the stack
- When a block is referenced or brought into the cache, it is placed on the top of the stack

Least Recently Used

- The block on the bottom of the stack is removed when a new block is brought in
- Blocks don’t actually move around in main memory
- A stack of pointers is used

Least Frequently Used

- The block that has experienced the fewest references is replaced
- A counter is associated with each block
- Counter is incremented each time block accessed
- Block with smallest count is selected for replacement
- Some blocks may be referenced many times in a short period of time and the reference count is misleading

Electronic Disks (SSD)

Concept

- A solid-state storage device that uses integrated circuit assemblies to store data persistently, typically using flash memory
- No movable read–write heads as one used in hard disk drives (HDDs) and floppy disks
- SSDs store data in semiconductor cells: 1-4 bits per cell
- Quicker access time and lower latency

Concept

- RAM-based SSD: power-off?
- NAND flash-based: retaining data for a few years
- Hybrid drives or solid-state hybrid drives (SSHDs), such as Apple's Fusion Drive, combine features of SSDs and HDDs in the same unit using both flash memory and HDD in order to improve the performance of frequently-accessed data

File Systems

- Typically the same file systems used on HDD can also be used on SSD
- Some log-structured file systems (e.g. F2FS, JFFS2) help to reduce write amplification on SSDs, especially in situations where only very small amounts of data are changed