



# Operating System Practice

Che-Wei Chang

[chewei@mail.cgu.edu.tw](mailto:chewei@mail.cgu.edu.tw)

Department of Computer Science and Information  
Engineering, Chang Gung University

# Advanced Operating System Concepts

- Chapter 10: File System
- Chapter 11: Implementing File-Systems
- ● Chapter 12: Mass-Storage Structure
- Chapter 13: I/O Systems
- Chapter 14: System Protection
- Chapter 15: System Security

# Study Items

- ▶ Overview of Mass Storage Structure
- ▶ Disk Structure
- ▶ Disk Attachment
- ▶ Disk Scheduling
- ▶ Disk Management
- ▶ Swap-Space Management
- ▶ RAID Structure

# Overview of Mass Storage Structure

- ▶ Magnetic disks provide bulk of secondary storage of modern computers
  - Drives rotate at 60 to 250 times per second
  - Transfer rate is rate at which data flow between drive and computer
  - Positioning time (random-access time) is time to move disk arm to desired cylinder (seek time) and time for desired sector to rotate under the disk head (rotational latency)
- ▶ Drive attached to computer via I/O bus
  - Busses vary, including ATA, SATA, USB, SCSI, SAS, Firewire
  - Host controller in computer uses bus to talk to disk controller

# Solid-State Disk

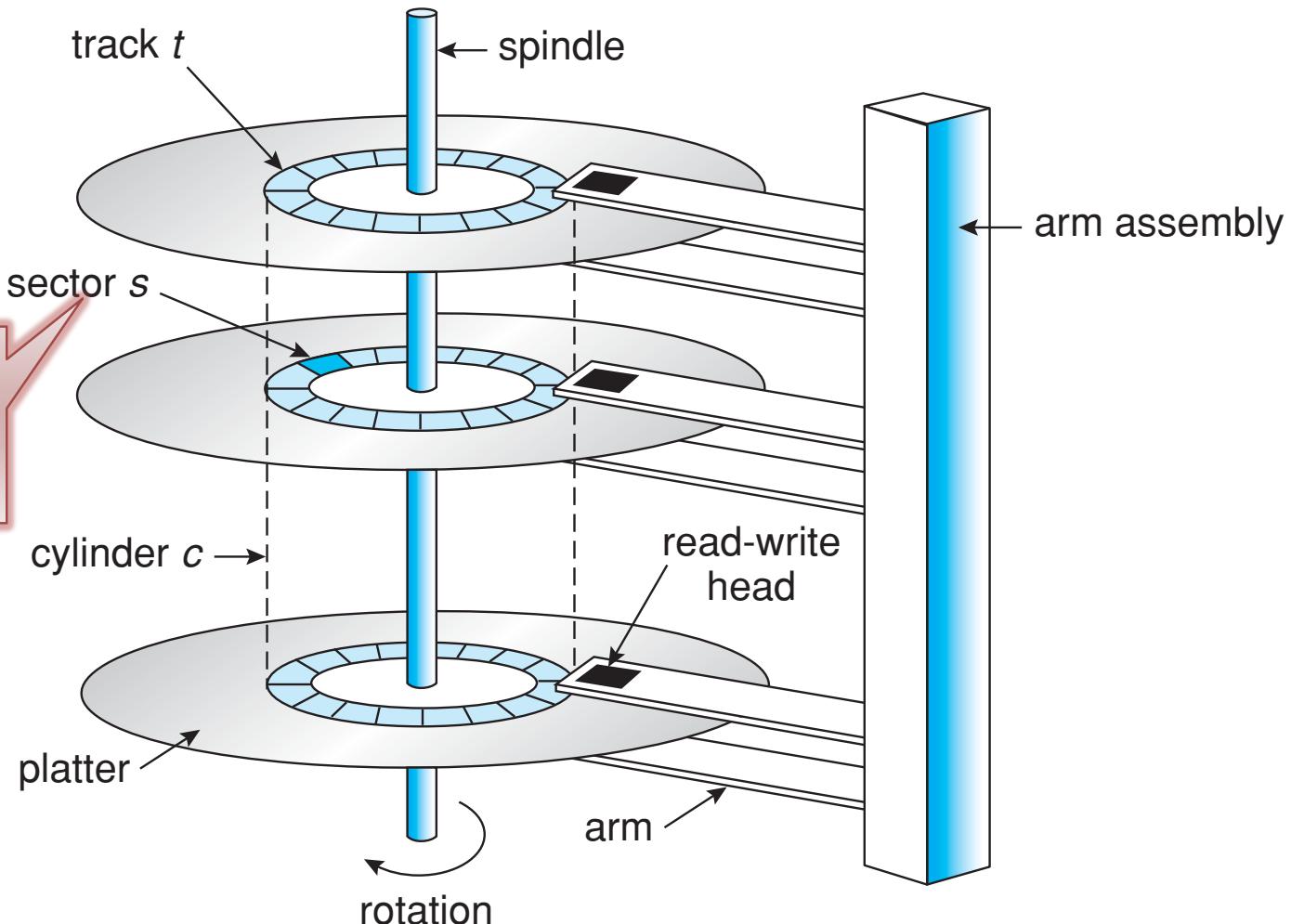
- ▶ Nonvolatile memory used like a hard drive
  - Many technology variations
- ▶ Can be more reliable than HD
- ▶ More expensive than HD
  - Less capacity
  - But much faster
- ▶ May have shorter life span
- ▶ Busses can be too slow
- ▶ **No moving parts**, so no seek time or rotational latency

# Magnetic Tape

- ▶ Early secondary-storage medium
- ▶ Relatively permanent and holds large quantities of data
- ▶ Random access ~1000 times slower than disk
- ▶ Mainly used for **backup**, storage of infrequently-used data, **transfer medium between systems**
- ▶ Once data under head, transfer rates comparable to disk
  - 140MB/sec or greater

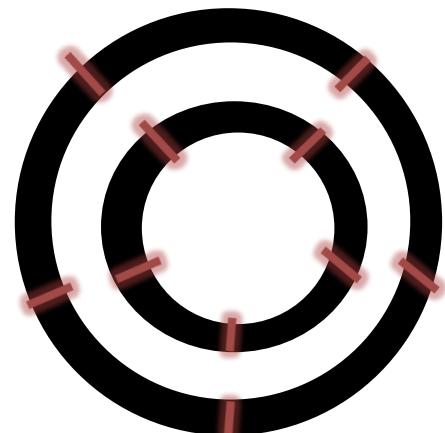
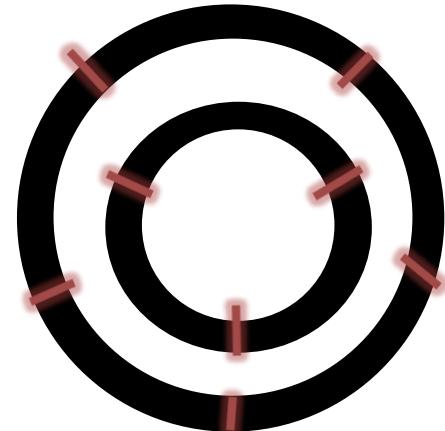
# Moving-Head Disk Mechanism

The size of a sector is from 512B to 4KB



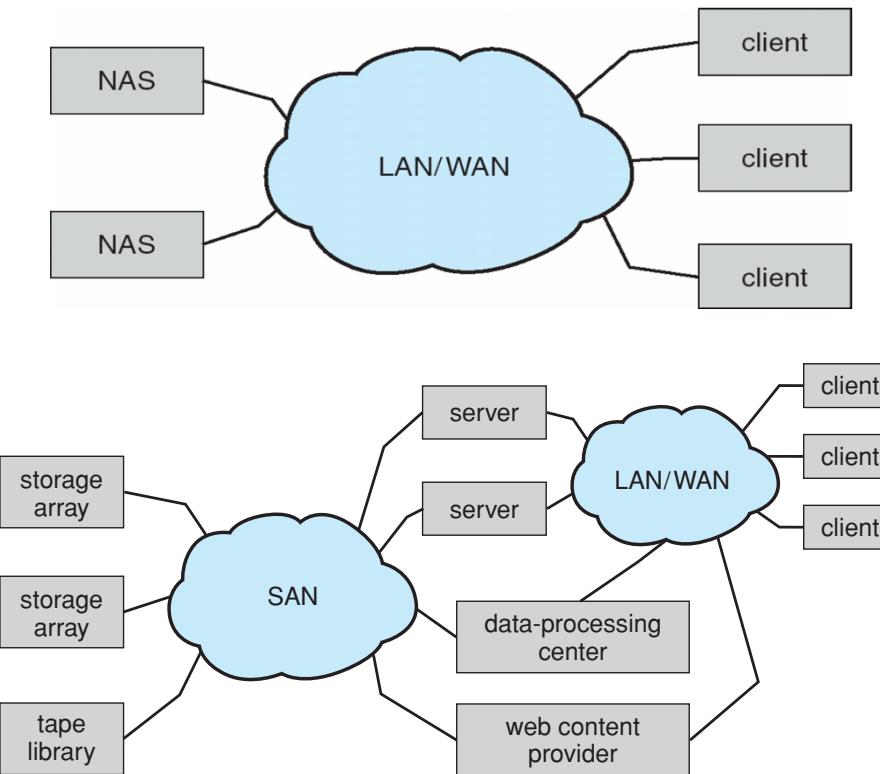
# Disk Structure

- ▶ Constant Linear Velocity (CLV)
  - The outermost track typically hold 40 percent more sectors than the innermost track
  - The drive increases its rotation speed as the head moves from the outer to the inner tracks
  - The same rate of data moving is kept
  - CD and DVD adopt this approach
- ▶ Constant Angular Velocity (CAV)
  - All tracks have the same number of sectors
  - Tracks have different densities of sectors
  - The same rate of data moving is kept
  - HD adopts this approach



# Disk Attachment

- ▶ Host-Attached Storage
  - Storage is accessed through I/O ports talking to I/O busses
- ▶ Network-Attached Storage (NAS)
  - NAS is **storage made available over a network** rather than over a local connection (such as a bus)
- ▶ Storage-Area Network (SAN)
  - SAN is **a private network** using storage protocols rather than networking protocol connecting servers and storage units



# Disk Scheduling

- ▶ The disk I/O request specifies several pieces of information:
  - Whether this operation is input or output
  - What the disk address for the transfer is
  - What the memory address for the transfer is
  - What the number of sectors to be transferred is
- ▶ When there are multiple request pending, a good disk scheduling algorithm is required
  - Fairness: which request is the most urgent one
  - Performance: sequential access is preferred

Cylinders	1	2	3	4	5	6	7
Requests	5	7	2	6	4	1	3

Resort the requests?

# Magnetic Disk Performance

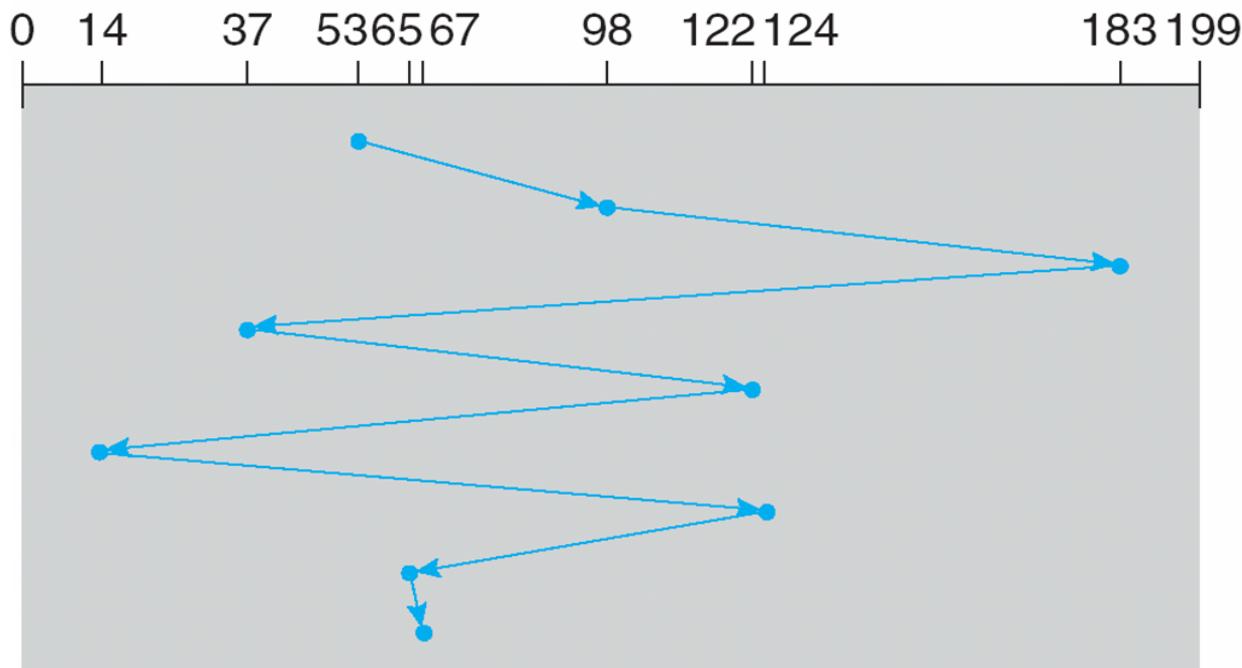
- ▶ Access Latency = Average access time = average seek time + average rotation latency
  - For fastest disk  $3\text{ms} + 2\text{ms} = 5\text{ms}$
  - For slow disk  $9\text{ms} + 5.56\text{ms} = 14.56\text{ms}$
- ▶ Average I/O time = average access time + (amount to transfer / transfer rate) + controller overhead

# FCFS Scheduling

- ▶ FCFS: first come, first serve
- ▶ FCFS scheduling is fair but might with low throughput

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53

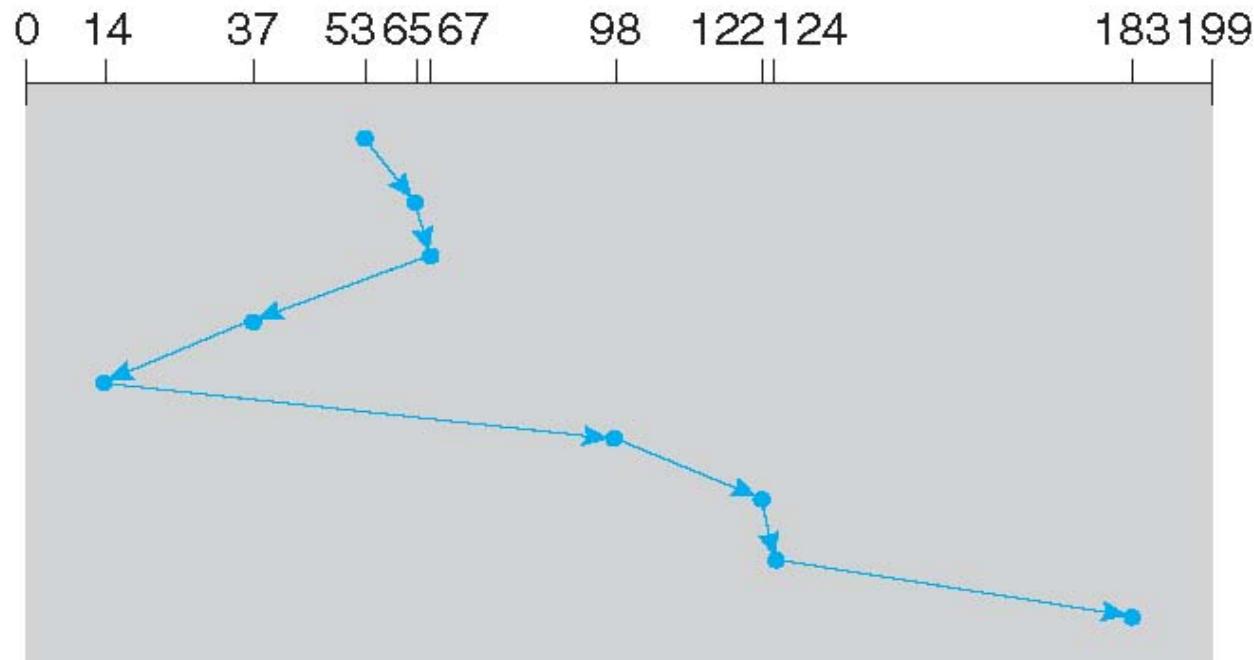


# SSTF Scheduling

- ▶ SSTF: shortest seek time first
- ▶ SSTF scheduling serves the request with shortest seek time

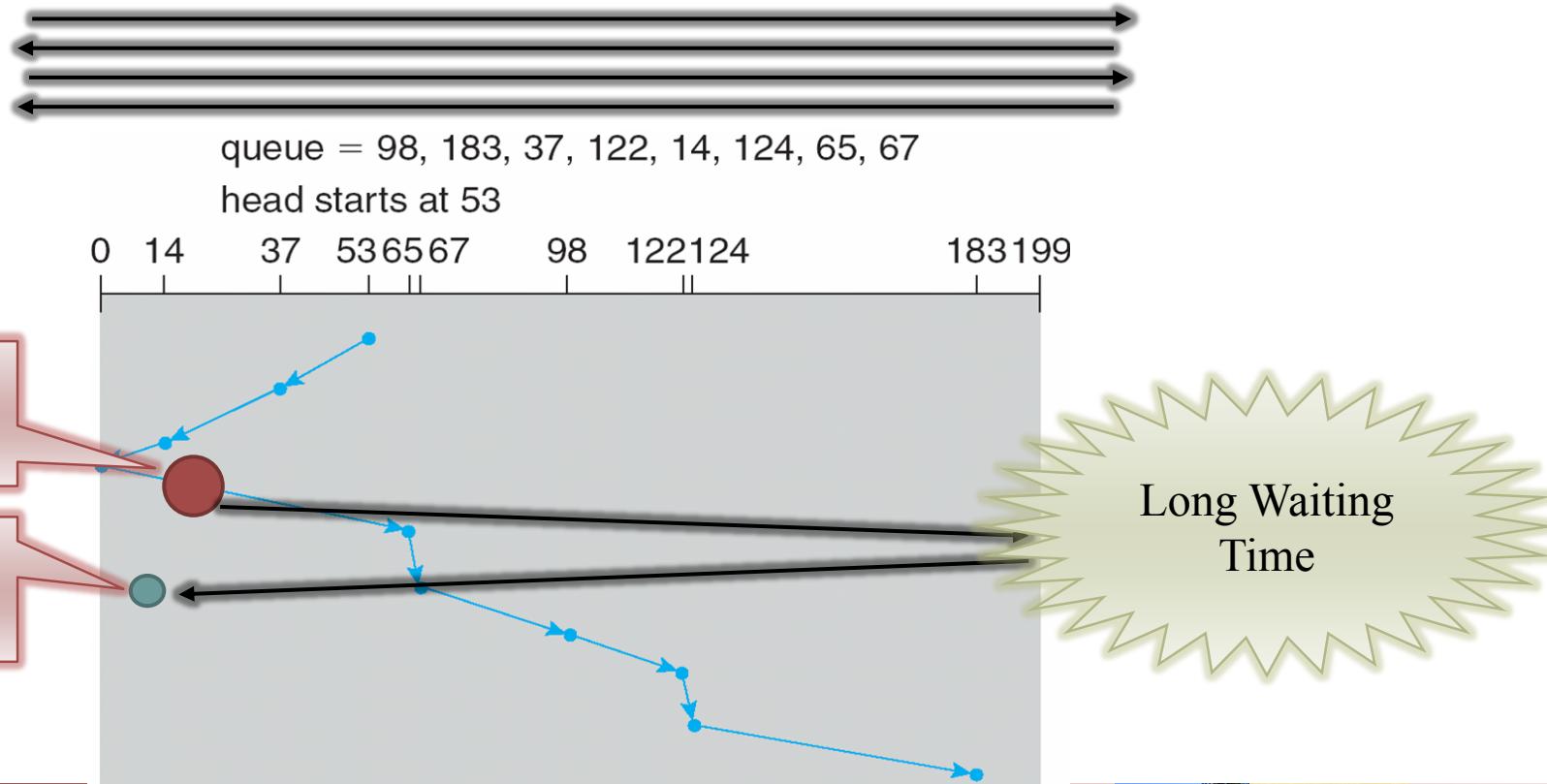
queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



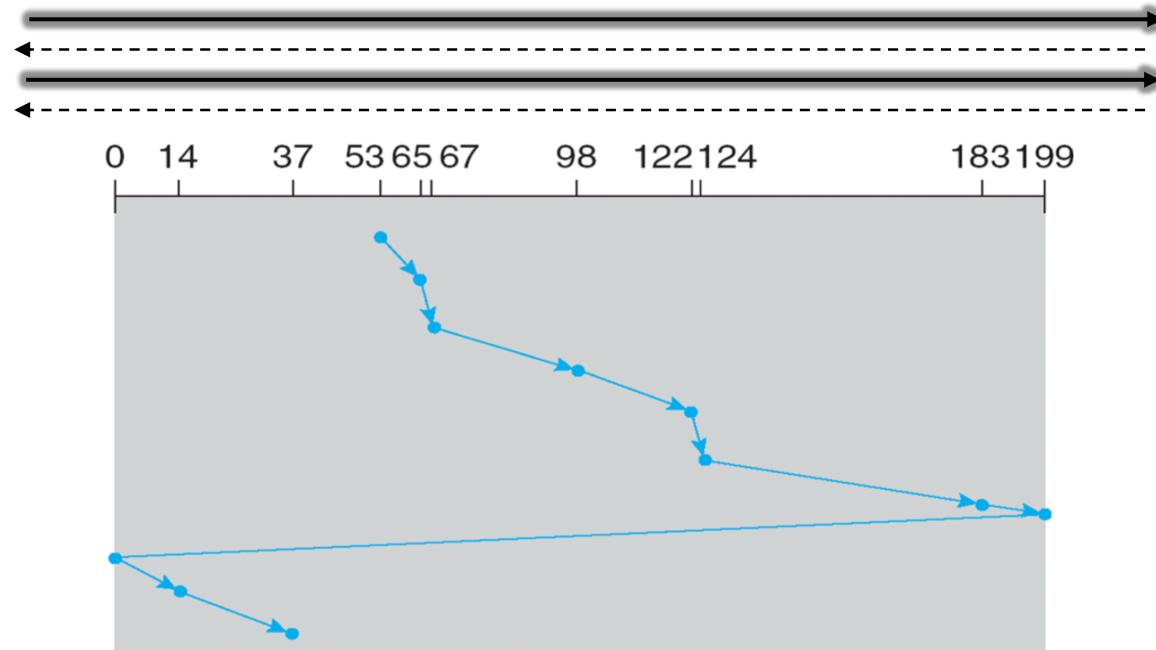
# SCAN Scheduling

- ▶ SCAN scheduling (also called the elevator algorithm) starts at one end and moves toward the other end



# C-SCAN Scheduling

- ▶ C-SCAN (Circular SCAN) scheduling starts at only one end and provides a more uniform wait time than SCAN scheduling

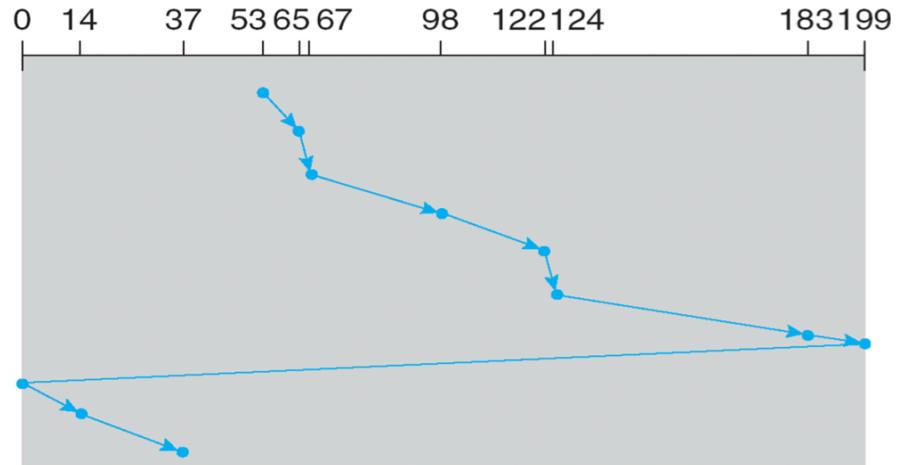


# LOOK and C-LOOK Scheduling

- ▶ LOOK scheduling starts at one end and moves toward the other end, and **looks for a request** before continuing to move in a given direction
- ▶ C-LOOK scheduling starts at only one end, and **looks for a request** before continuing to move in a given direction
- ▶ Arm only goes as far as the last request in each direction, then reverses direction immediately, **without first going all the way to the end of the disk**

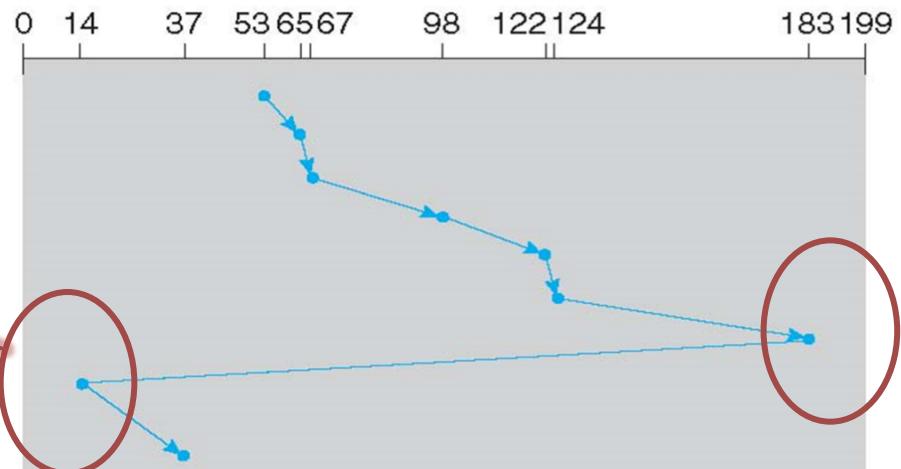
# Examples of C-SCAN and C-LOOK

C-SCAN



C-LOOK

Reduce the moving time



# Selecting a Disk-Scheduling Algorithm

- ▶ SSTF is common and has a natural appeal
- ▶ SCAN and C-SCAN perform better for systems that place a heavy load on the disk
  - Less starvation
- ▶ Requests for disk service can be influenced by the file-allocation method
- ▶ The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary
- ▶ Either SSTF or LOOK is a reasonable choice for the default algorithm
- ▶ FCFS is good for SSD

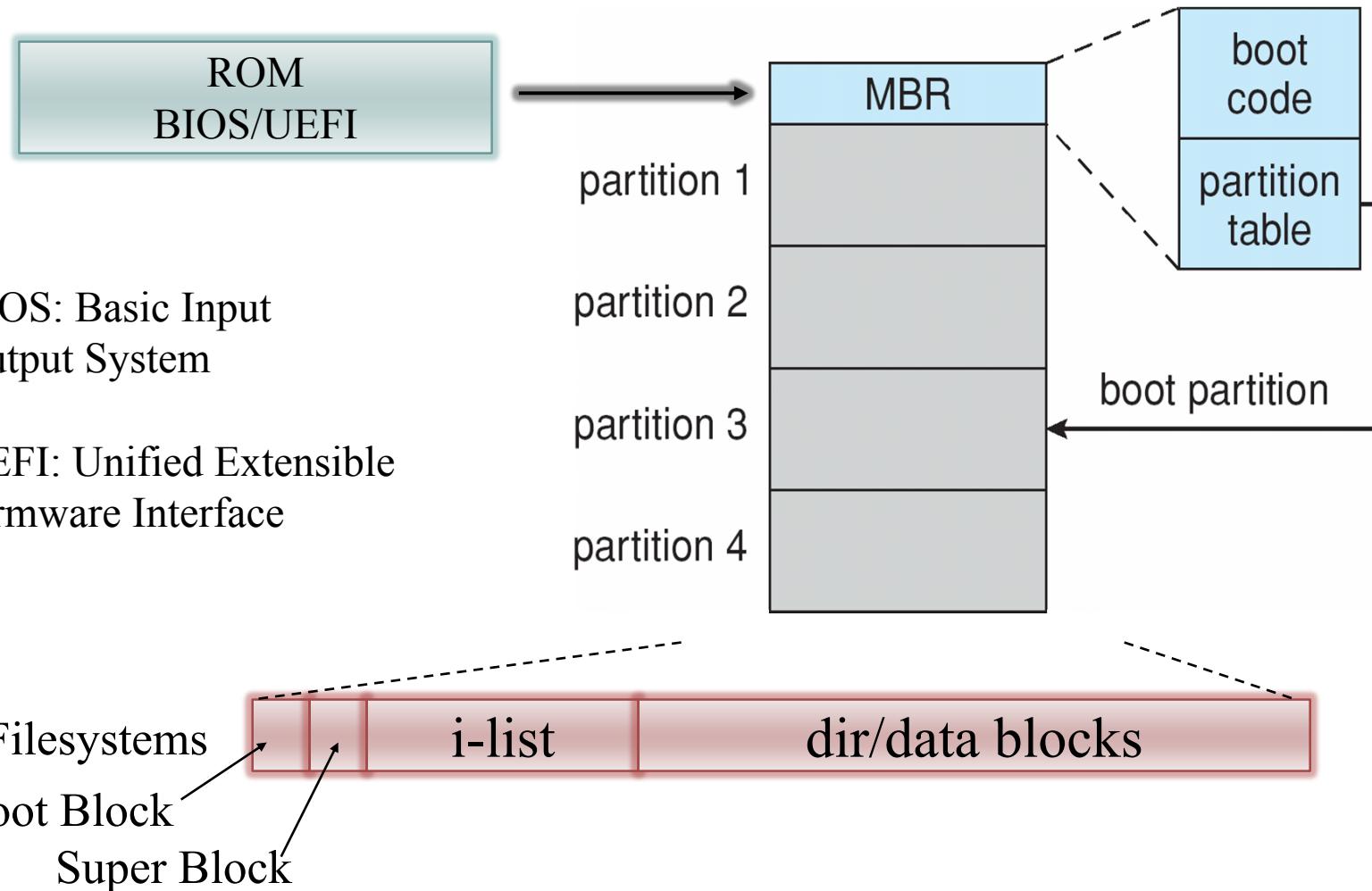
# Disk Management

- ▶ Low-level formatting, or physical formatting — Dividing a disk into sectors that the disk controller can read and write
  - Each sector can hold header information, plus data, plus error correction code (ECC)
  - Usually 512 bytes of data but can be selectable
- ▶ Partition the disk into one or more groups of cylinders, each treated as a logical disk
- ▶ Logical formatting — making a file system
  - To increase efficiency most file systems group blocks into clusters
    - Disk I/O done in blocks
    - File I/O done in clusters
- ▶ Raw disk access for apps that want to do their own block management, keep OS out of the way (databases for example)

# Bootstrapping from a Disk

BIOS: Basic Input  
Output System

UEFI: Unified Extensible  
Firmware Interface



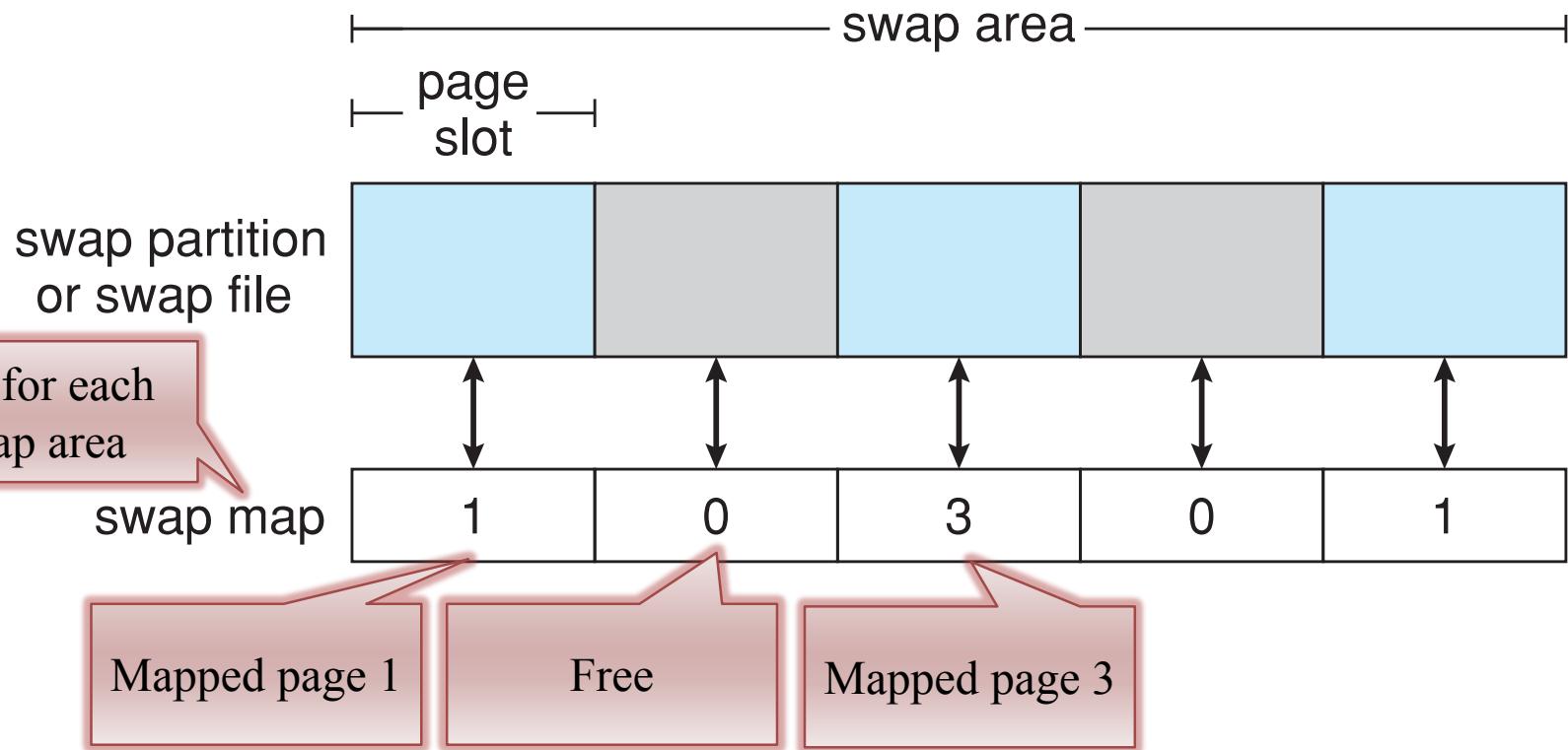
# Bad Blocks

- ▶ A bad block: some bits of data in the block is corrupted
- ▶ Soft error: a bad block can be recovered by ECC
- ▶ Hard error: a bad block results in lost data
- ▶ Spared sectors are for bad block replacement
  - For example, one spared sector per 100 normal sector, let 97<sup>th</sup> block is a bad block
  - Sector sparing:
    - Use the spared sector to replace the 97<sup>th</sup> block
  - Sector slipping:
    - 97→98, 98→99, 99→100, 100→spared sector

# Swap-Space Management

- ▶ Swap-space: virtual memory uses disk space as an extension of main memory
  - Less common now due to memory capacity increases
- ▶ Swap-space can be carved out of the normal file system, or, more commonly, it can be in a separate disk partition (raw)
- ▶ Some systems allow multiple swap spaces

# Data Structures for Swapping on Linux Systems



# RAID Structure

- ▶ RAID: redundant array of inexpensive disks
  - Multiple disk drives provides reliability via **redundancy**
  - RAID Increases the **mean time to data loss**
- ▶ **Mean time to failure**: the average time for the first hard error
- ▶ **Mean time to repair**: the exposure time when another failure could cause data loss
- ▶ **Mean time to data loss**: the average time for the first data loss
- ▶ If mirrored disks fail independently, consider each disk with 100,000 hours mean time to failure and 10 hours mean time to repair
  - Mean time to failure of any of the two disks:  $100,000/2 = 50,000$
  - The possibility for another disk to fail within the 10 hours:  $10/100,000$
  - **Mean time to data loss is  $5 * 10^8$  hours, or 57,000 years!**

# Improvement in Performance via Parallelism

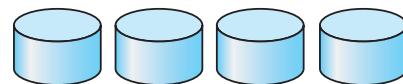
- ▶ Bit-level striping
  - Bit-level striping can be generalized to include a number of disks that either is a multiple of 8 or divides 8
  - For example, if we use an array of four disks, bits  $i$  and  $4 + i$  of each byte go to disk  $i$
- ▶ Block-level striping
  - blocks of a file are striped across multiple disks
  - For example, with  $n$  disks, block  $i$  of a file goes to disk  $(i \bmod n) + 1$

# RAID

- ▶ RAID schemes improve performance and improve the reliability of the storage system by storing redundant data
  - Mirroring or shadowing (RAID 1) keeps duplicate of each disk
  - Striped mirrors (RAID 1+0) or mirrored stripes (RAID 0+1) provides high performance and high reliability
- ▶ Block interleaved parity (RAID 4, 5, 6) uses much less redundancy
- ▶ Frequently, a small number of hot-spare disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them

# RAID Levels

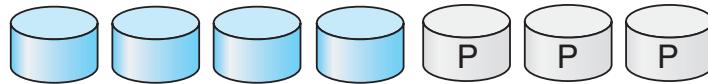
In the figures, P indicates error-correcting bits and C indicates a second copy of the data



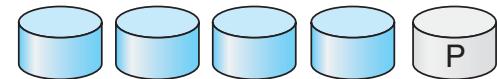
(a) RAID 0: non-redundant striping.



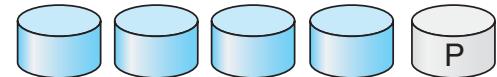
(b) RAID 1: mirrored disks.



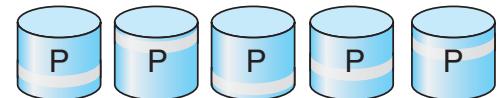
(c) RAID 2: memory-style error-correcting codes.



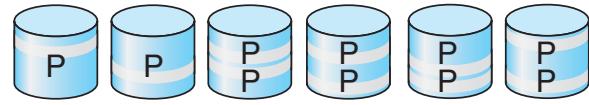
(d) RAID 3: bit-interleaved parity.



(e) RAID 4: block-interleaved parity.



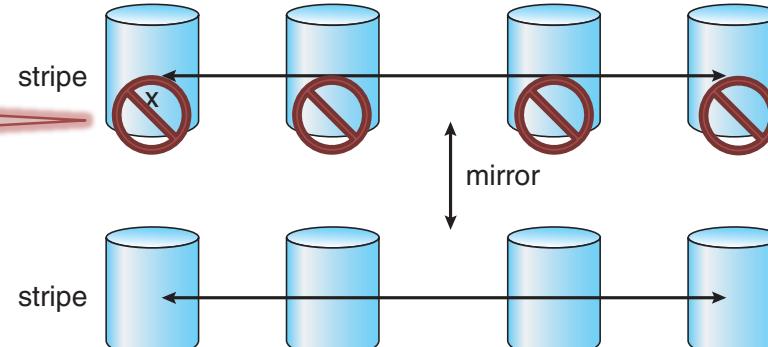
(f) RAID 5: block-interleaved distributed parity.



(g) RAID 6: P + Q redundancy.

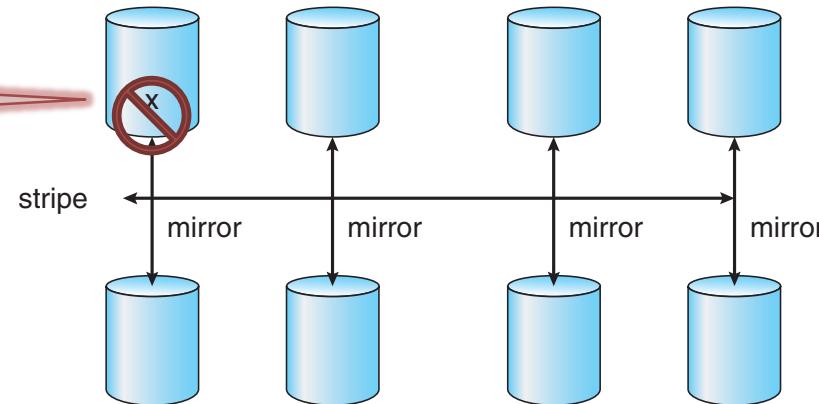
# RAID (0+1) and (1+0)

One Fail  
Four Offline



a) RAID 0 + 1 with a single disk failure.

One Fail  
One Offline



b) RAID 1 + 0 with a single disk failure.