Mass-Storage Systems

:Galvin- Gagne
Jul-Nov 2019
Moumita Patra

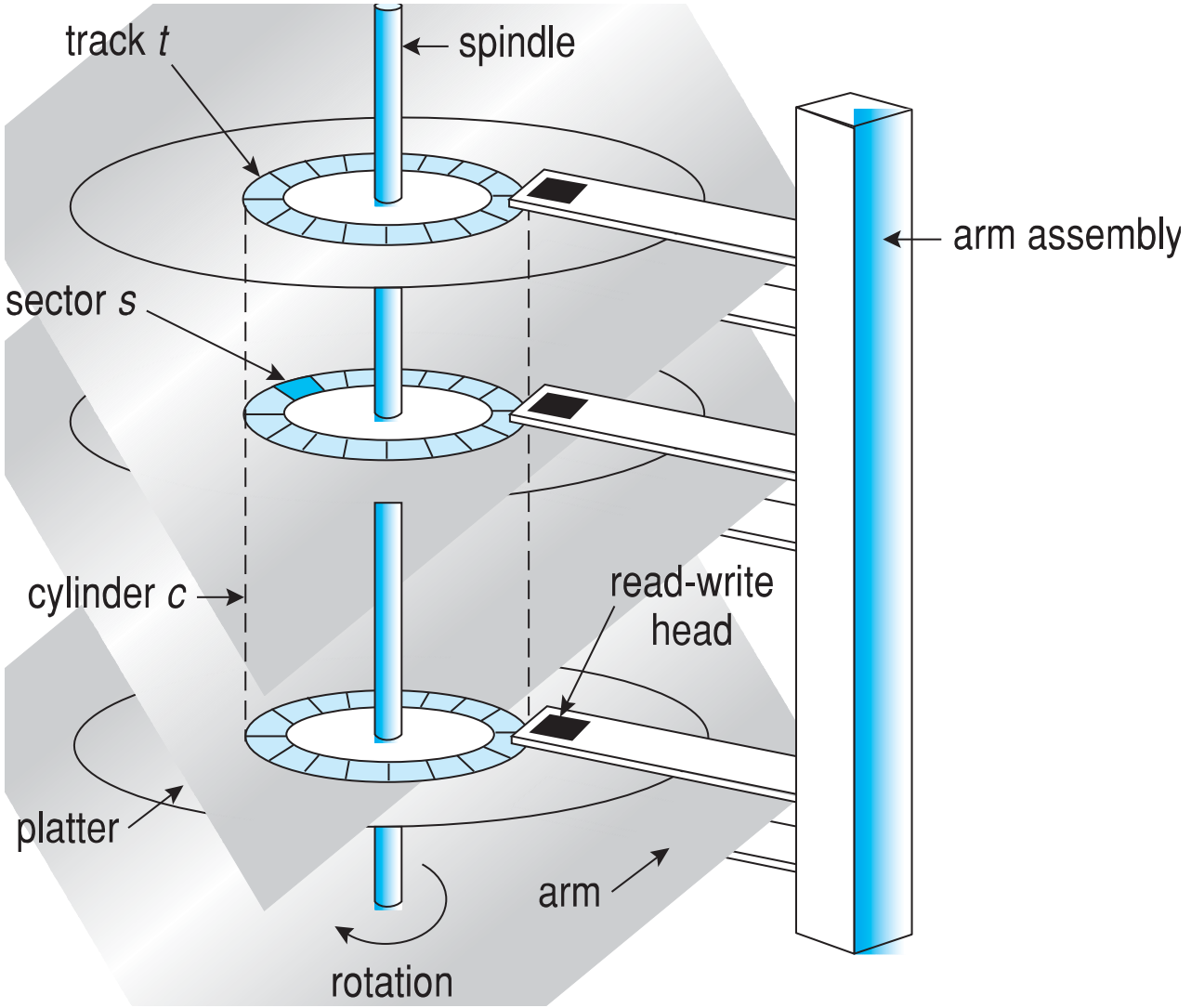
Objectives

- To describe the physical structure of secondary storage devices and its effects on the use of the devices
- To explain the performance characteristics of mass-storage devices
- To evaluate disk scheduling algorithms

- **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**)
- **Head crash** results from disk head making contact with the disk surface -- That's bad
- Disks can be removable

- Drive attached to computer via **I/O bus**
- Busses vary, including **ATA**, **SATA**, **USB**, **Fibre Channel**, **SCSI, SAS, Firewire**
- **Host controller** in computer uses bus to talk to **disk controller** built into drive or storage array
- Information is stored by recording it magnetically on the platters

Moving-head Disk Mechanism



Hard Disks

- Platters range from .85” to 14” (historically)
- Commonly 3.5”, 2.5”, and 1.8”
- Range from 30 GB to 3 TB per drive
- Performance
  - Transfer rate- theoretical- 6 Gb/sec
  - Effective transfer rate- real- 1 Gb/sec
  - Seek time from 3 ms to 12 ms – 9 ms common for desktop drives

Hard Disk Performance

- Access latency = Average access time = average seek time + average latency
- Average I/O time = average access time + (amount to transfer/ transfer rate) + controller overhead

Solid State Disks

- Non volatile memory used like hard drive
- Many technology variations- DRAM with a battery, flash-memory technologies like single-level cell (SLC) and multilevel cell (MLC) chips
- Can be more reliable than HDDs
- Consume less power
- More expensive per MB
- May have shorter life span
- Less capacity
- Faster, because no access latency (as no moving parts)
- Busses can be too slow -> may connect directly to PCI

Magnetic Tape

- Was early secondary-storage medium
- Relatively permanent and holds large quantities of data
- Slow access time
- 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 are comparable to 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
- Low-level formatting creates **logical blocks** on physical media
- 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
- Logical to physical address should be easy
  - Except for bad sectors

Disk Scheduling

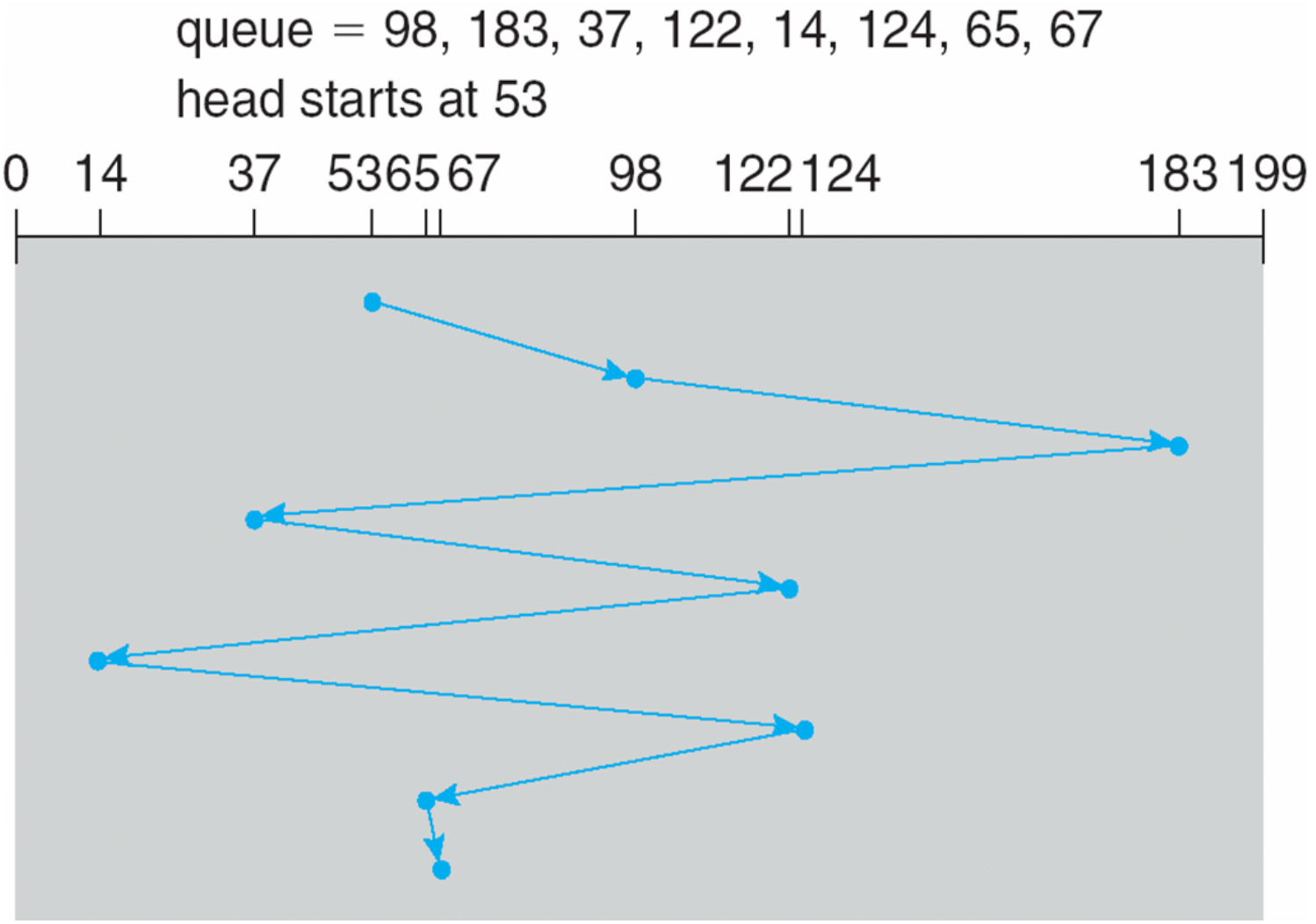
- The OS is responsible for using hardware efficiently- for the disk drives, this means having a fast access time and disk bandwidth
- Minimize seek time
- Seek time => seek distance
- **Disk bandwidth**- total number of bytes transferred divided by total time between first request of service and the completion of the last transfer

- There are many sources of disk I/O request -
  - OS
  - System processes
  - Users processes
- I/O request includes input or output mode, disk address, memory address, number of sectors to transfer
- OS maintains queue of requests, per disk or device
- Idle disk can immediately work on I/O request, busy disk means work must queue
- Optimization algorithms only make sense when a queue exists

- **NOTE**: Drive controllers have small buffers and can manage a queue of I/O requests
- Several algorithms exist to schedule the servicing of disk I/O requests
- Analysis is true for one or many platters

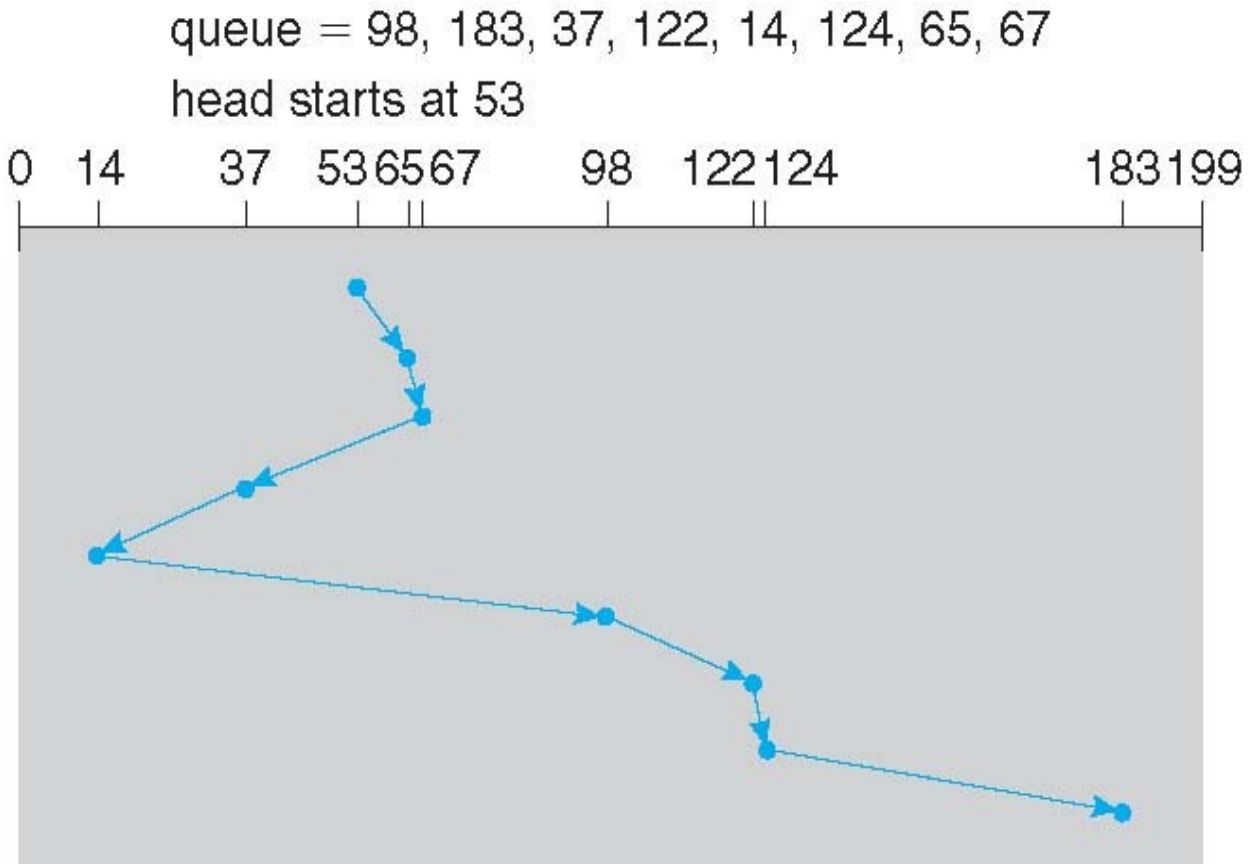
FCFS

Total head movement of 640 cylinders



Shortest Seek Time First (SSTF)

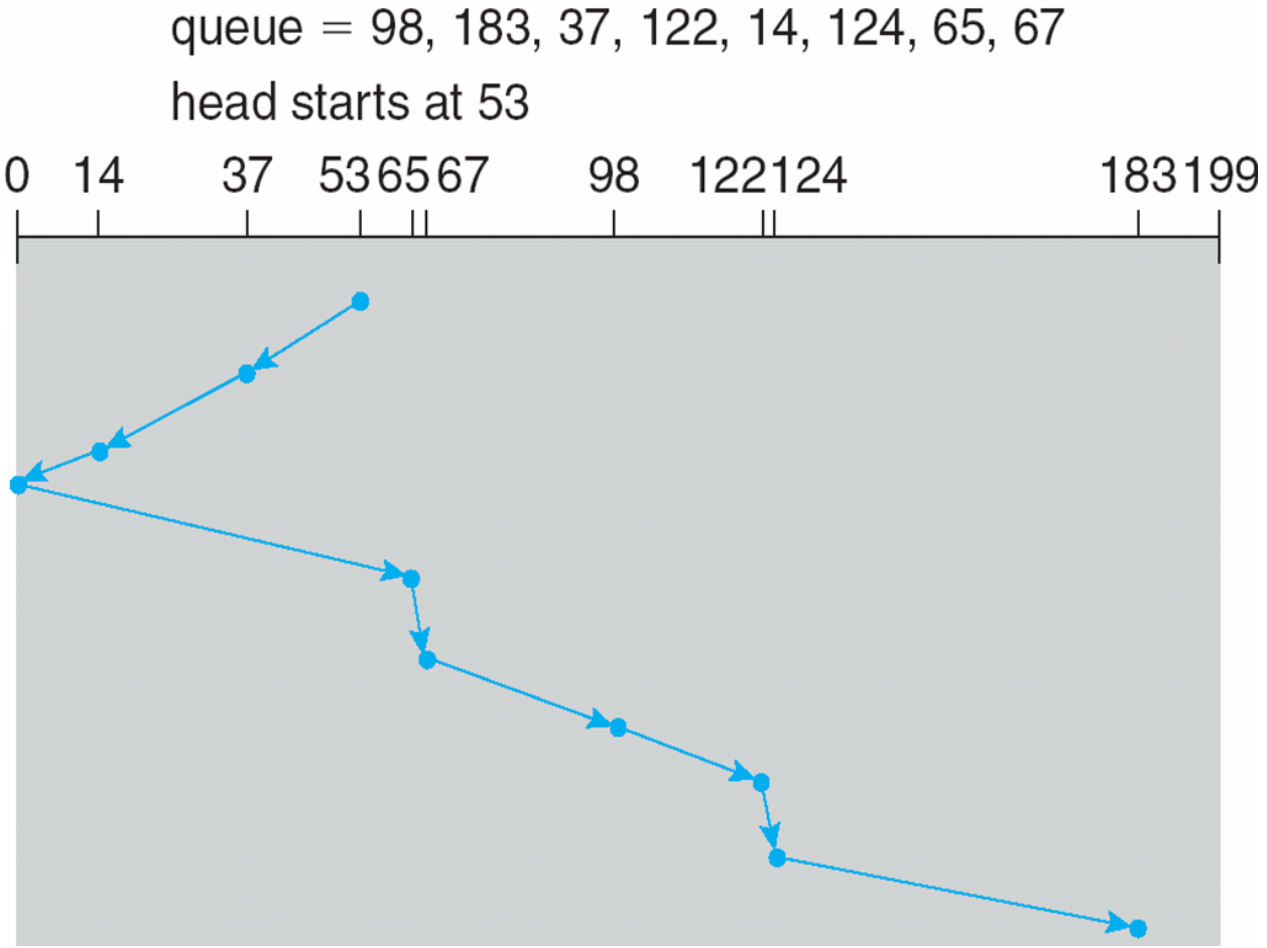
- Selects request with the minimum seek time from the current head position
- Like SJF scheduling
- May cause starvation of some requests



SCAN

- Disk arm starts at on end of the disk, moves towards the other end while servicing requests
- Once the other end is reached, head movement is reversed and servicing continues
- Also called elevator algorithm
- If requests are uniformly dense, largest density at other end of disk and those wait longest

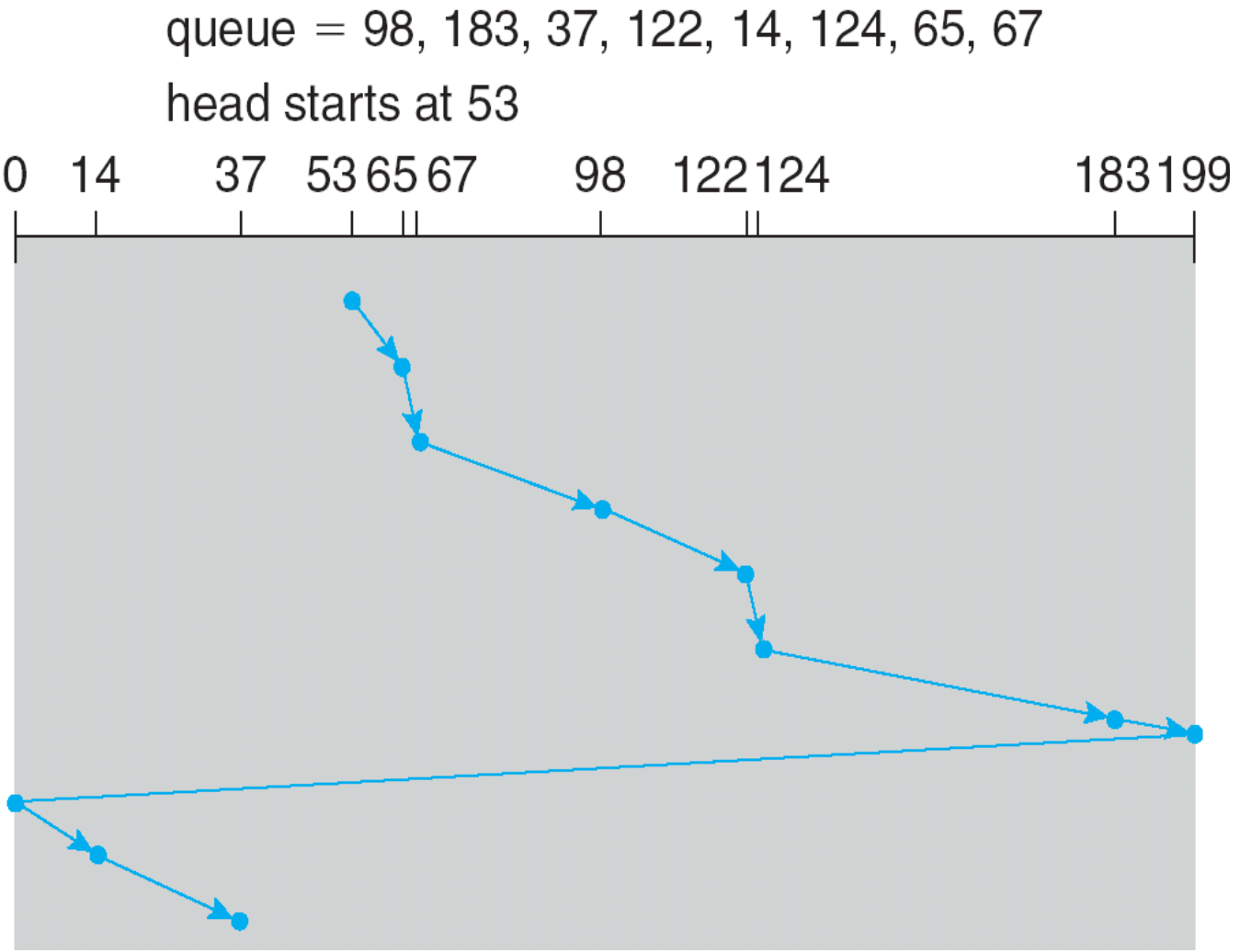
SCAN


queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
0
14
37
53
65
67
98
122
124
183
199

C-SCAN

- Provides a more uniform wait time than SCAN
- Head moves from one end of the disk to the other, servicing requests as it goes
- When it reaches the other end, it immediately returns to the beginning of the disk, without servicing any requests on the return trip
- Treats the cylinders as a circular list that wraps around from the cylinder to the first one

C-SCAN


queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
0
14
37
53
65
67
98
122
124
183
199

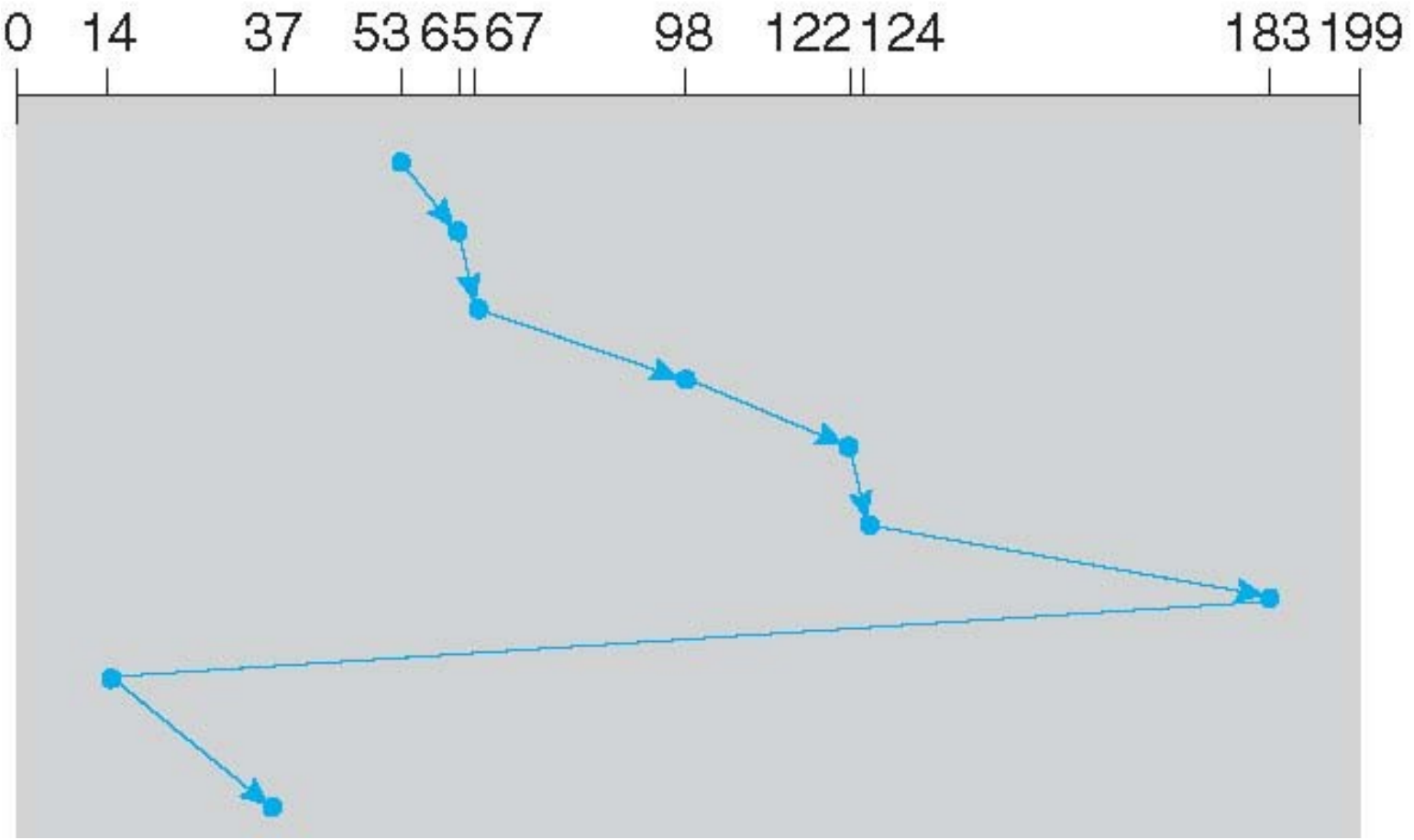
C-LOOK

- LOOK – a version of SCAN
- C-LOOK- a version of C-SCAN
- 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

C-LOOK

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

head starts at 53



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

- Performance depends on the number and types of requests
- Requests for disk service can be influenced by the file-allocation method and metadata layout

- 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
- What about rotational latency?

➔ Difficult for OS to calculate

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
- To use a disk to hold files, the operating system still needs to record its own data structures on the disk
  - **Partition** the disk into one or more groups of cylinders, each treated as a logical disk
  - **Logical formatting** or “making a file system”- the OS stores the initial file system data structures onto the disk.
  - To increase efficiency most file systems group blocks into **clusters**
    - Disk I/O done in blocks
    - File I/O done in clusters

Disk Management (Cont.)

- Raw disk- Some OS give special programs the ability to use a disk partition as a large sequential array of logical blocks, without any file-system data structures. This array is called raw disk.
- I/O to raw disk is called raw I/O.
- Raw disk access for applications that want to do their own block management, keep OS out of the way (databases for example)

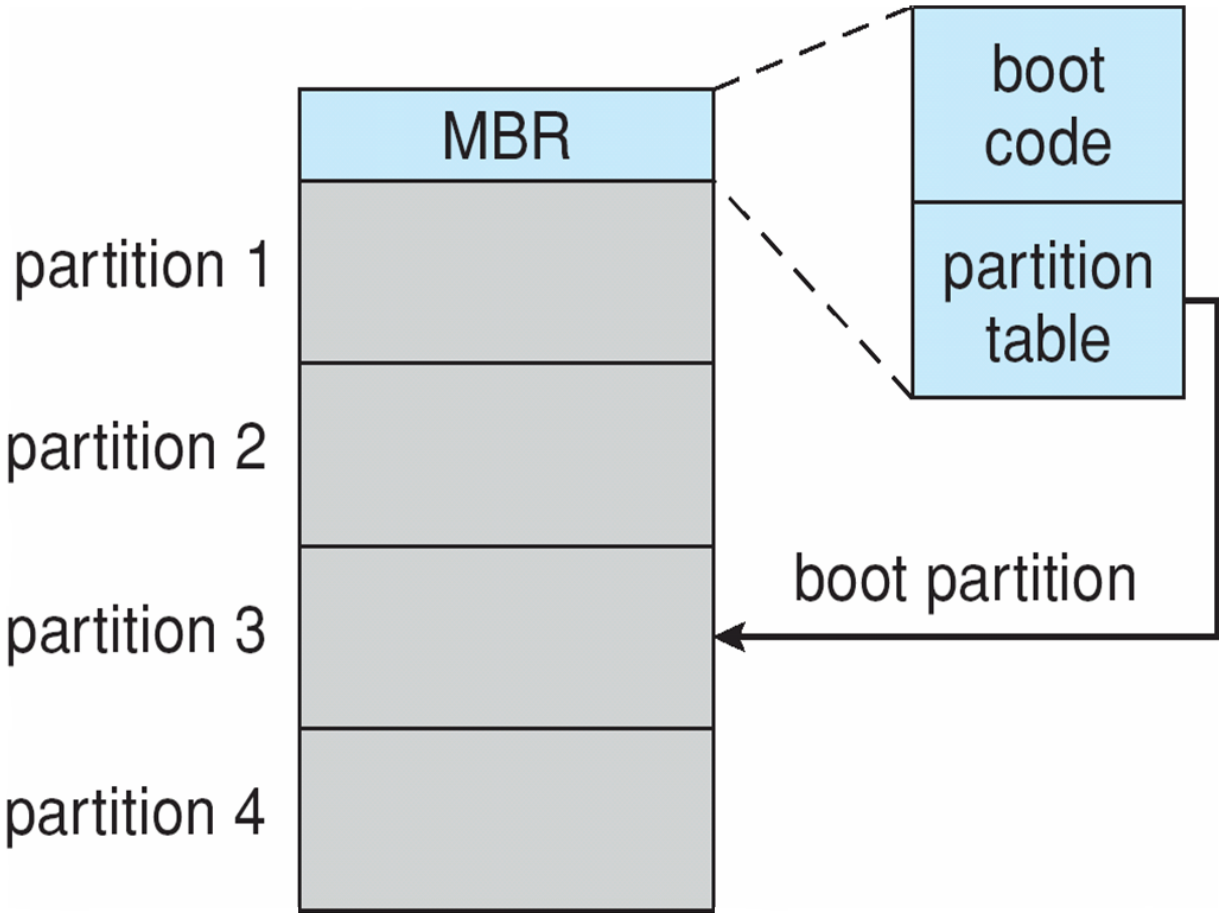
Boot Block

- Bootstrap program initializes all aspects of the system and starts the OS.
- Bootstrap program tends to be very simple.
- It finds the OS kernel on the disk, loads the kernel into memory, and jumps to an initial address to begin the OS execution.
- Bootstrap is stored in the ROM because
  - ROM needs no initialization
  - It is at a fixed location that the processor can start executing when powered up or reset.
  - Since ROM is read-only, it cannot be infected by a computer virus.
- **Problems**
  - Changing the bootstrap code requires changing the ROM hardware chips.
- Thus, most systems store a tiny bootstrap loader program in the boot ROM whose only job is to bring in a full bootstrap program from disk.
- The full bootstrap program is stored in "boot blocks" at a fixed location on the disk.

- Boot block initializes the system
- Disk that has a boot partition is called a boot disk or system disk.
- Code in the boot ROM instructs the disk controller to read the boot blocks into memory (no device drivers are loaded at this point) and then starts executing the code.
- **Bootstrap loader** program is stored in boot blocks of boot partition.

- In a Windows system
  - Hard disk can be divided into partitions and one partition is identified as boot partition- contains the OS and device drivers.
  - The boot code is placed in the first sector on the hard disk, known as Master Boot Record (MBR).
  - Booting begins by running the code in the system's ROM. This code directs the system to read boot code from MBR.
  - MBR also contains a table listing the partitions for the hard disk and a flag indicating which partition the system is to be booted from.
  - Once the system identifies the boot partition, it reads the first sector from that partition (known as boot sector) and continues with the remainder of the boot process.

Booting from a Disk in Windows



Bad Blocks

- If failure is complete, disk nees to be replaced and its contents need to be restored from backup media to the new disk.
- More often, one or more sectors become defective.
- Most disks have bad blocks even when they come from factory.

Swap-Space Management

- Swap space management is a low-level task of the OS.
- Swap space - Virtual memory uses disk space as an extension of main memory.
- Since disk access is slower than memory access, using swap space decreases performance.
- Therefore, objective is to design and implement swap space to provide best throughput.
- Swap space can be carved out of normal file system, or, can be in a separate disk partition (raw)
- It is safe to overestimate swap space than underestimate it.
- What if a system runs out of swap space?

RAID

- RAID- Redundant Array of Independant Disks
- Creating a disk system to provide
- Reliability
- Availability
- Performance
- Capacity

RAID Structure

- **Structuring RAID:** RAID storage can be structured in a variety of ways
- A system can have disks directly attached to its buses
- An intelligent host controller can control multiple attached disks and can implement RAID on those disks in hardware.
- A storage array or RAID array can be used.
- RAID Array: A standalone unit with its own controller cache (usually) and disks. It is attached to the host via one or more standard controllers.

Improvement of Reliability via Redundancy

- **“Chance that some disk out of N disks will fail is much higher than the chance of a specific single disk fail”**
- Then how is RAID going to help?
  - **Redundancy:** Store extra information that can be used in the event of failure of a disk so that data is not lost
  - Increases the **mean time to failure**
- **Mirroring-** a logical disk consists of 2 physical disks and every write is carried out on both disks
- MTTF of a mirrored volume depends on 2 factors:

➔ MTTF of the individual disk

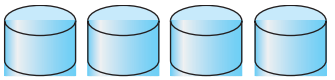
➔ MTTR

- **Mean time to repair –** exposure time when another failure could cause data loss

Improvement via Parallelism

- Parallel access to multiple disks improves performance- throughput and reduced response time
- Disk **striping** uses a group of disks as one storage unit
- **Bit-level striping:** splitting the bits of each byte across multiple disks.
- **Block-level striping:** blocks of a file are striped across multiple disks. Most common.
- RAID is arranged into different levels

RAID Levels



(a) RAID 0: non-redundant striping.

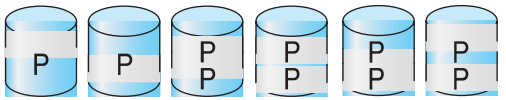
(b) RAID 1: mirrored disks.

(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.

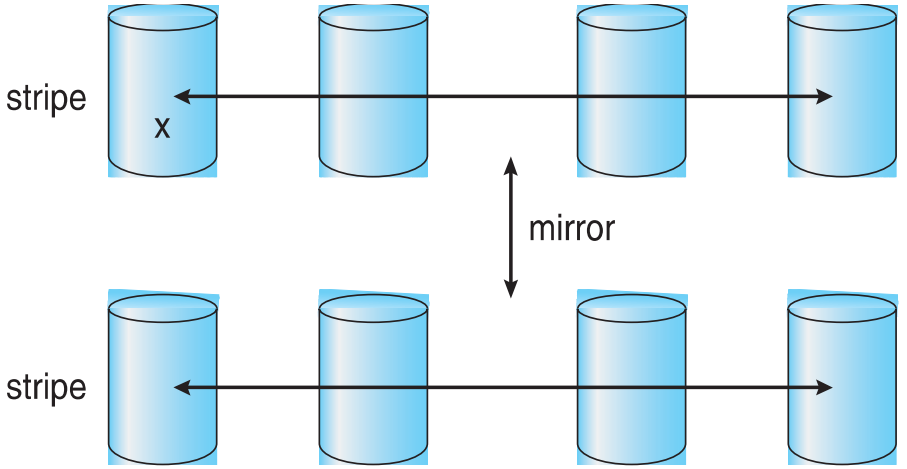


(g) RAID 6: P + Q redundancy.

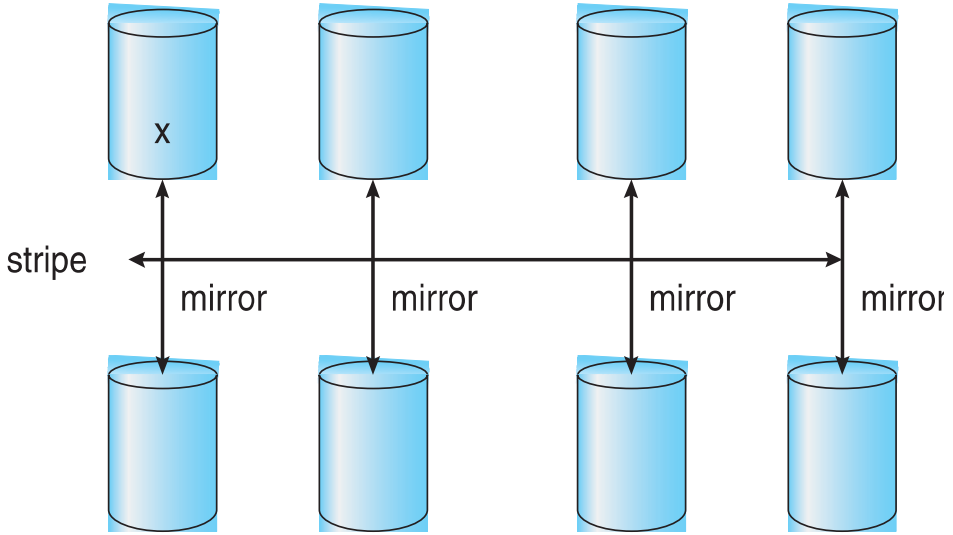
RAID Levels

- RAID level 0: disk arrays with striping at the level of blocks but without any redundancy
- RAID 1: **Mirroring** or **shadowing.** Keeps duplicate of each disk
- RAID 2: Bit level striping + ECC
- RAID 3: Byte level striping + Parity
- RAID 4: Block level striping + Parity
- RAID 5: striping + distributed parity
- RAID 6: Like RAID 5 but with ECC such as REED-Solomon Codes

- Striped mirrors (**RAID 1+0**) or mirrored stripes (**RAID 0+1**) provides high performance and high reliability
- RAID 0 +1: Generally gives better performance than RAID 5
- Expensive because doubles the number of disks for storage
- RAID 1 + 0: Disks are mirrored in pairs, resulting mirrored pairs are striped
- With a failure, a single disk is unavailable but the disk that mirrors it is still available, as are all the rest of the disks



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



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

Which RAID level to select?

- Time required to rebuild performance after failure
- Other decisions to be made:
- Number of disks in a RAID set?
- How many disks should be protected by each parity bit?
-