Operating Systems

Course Code: **71203002004**

Disk & I/O Management

by -

Minal Rajwar

What is Disk Management?

Disk management is the part of the OS that organizes and controls data on storage devices (HDDs, SSDs, flash drives, etc.).

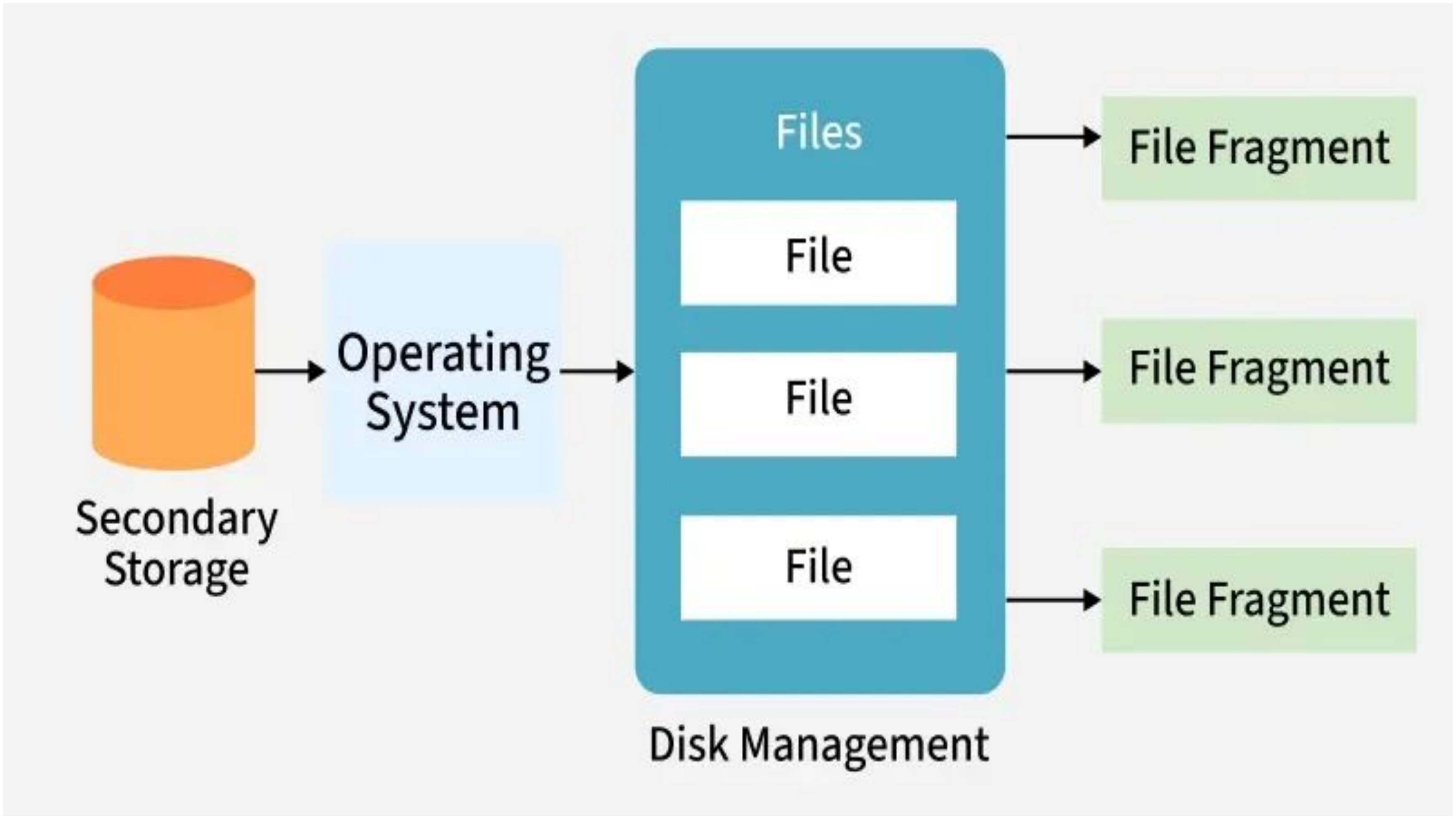
It ensures:

- Efficient use of storage space
- Safe and reliable read/write operations
- Faster data access

Role in OS Functions

The OS has four main management areas:

1. **Process Management** – runs processes.
2. **Memory Management** – manages RAM.
3. **File & Disk Management** – stores and organizes data on disk.
4. **I/O Management** – handles device communication.


Secondary Storage
Operating System
Files
File
File
File
File Fragment
File Fragment
File Fragment
Disk Management

Why Disk Management is Important

Secondary storage (disks, SSDs, tapes) stores programs and data cheaply and permanently.

Data is stored as **files**. Large files may be split (fragmented) across different disk locations.

The OS keeps track of file locations so data can be accessed quickly and safely.

Key Operations

Disk Formatting

- *Low-level formatting*: divides disk into sectors.
- *Logical formatting*: creates a file system (FAT, NTFS, ext4).
- Groups blocks into clusters for efficiency.

Booting from Disk

- A **bootstrap loader** (in ROM) loads the OS from the boot block.
- A disk with an OS is called a **boot disk**.

Bad Block Management

- Handles defective sectors.
- Methods: sector sparing, error recovery, or replacing the disk.

Common Disk Management Techniques

Partitioning – divide one disk into multiple logical drives.

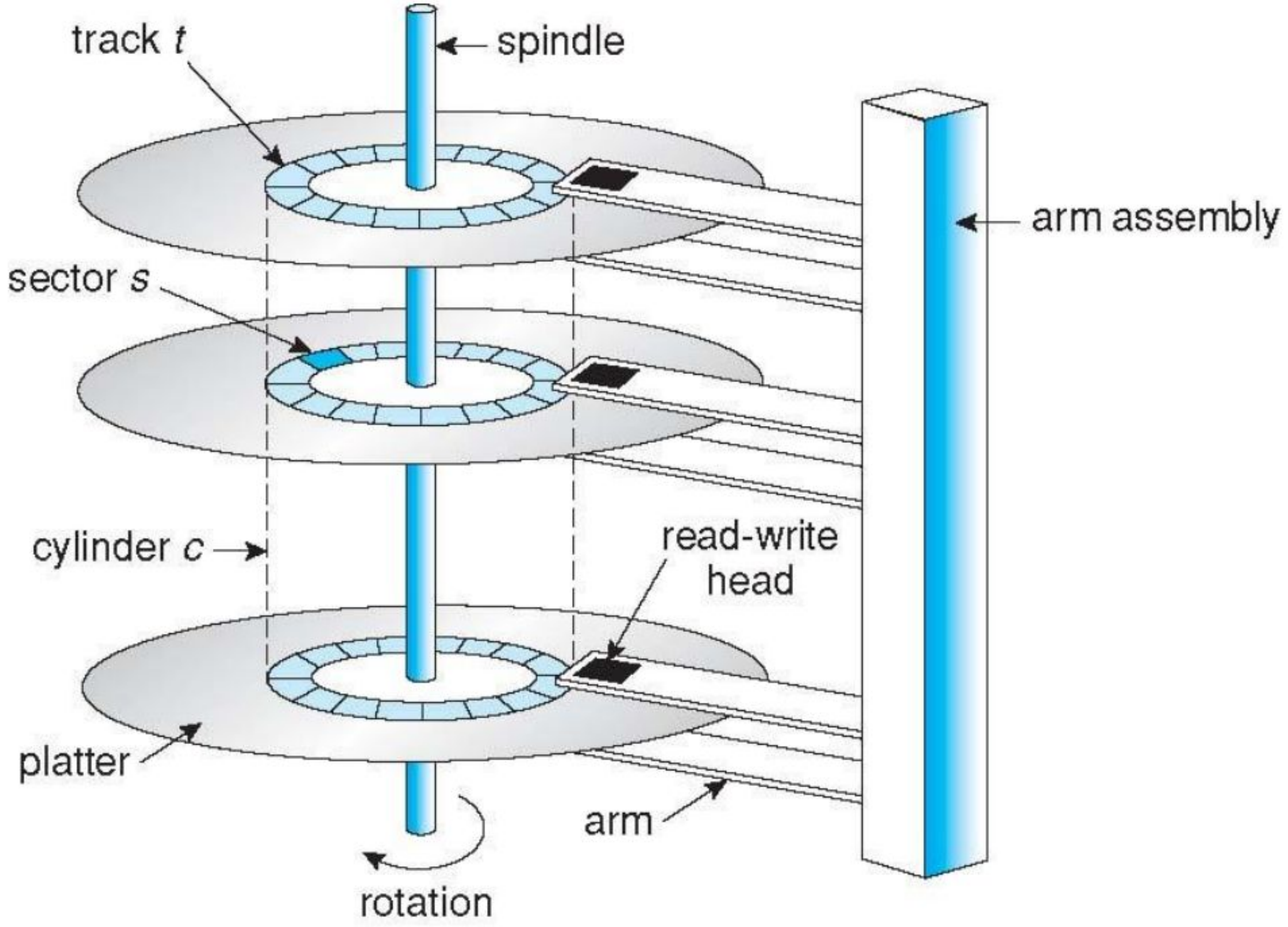
Formatting – prepare disk with a file system (erases all data).

File System Management – organizes files (FAT, NTFS, ext4).

Disk Space Allocation – allocates space (contiguous, linked, indexed).

Defragmentation – rearranges scattered file parts for faster access.

Moving Head Disk Mechanism



Disk Access Time

The **average access time** for a disk =

$$\text{Average Seek Time} + \text{Average Rotational Latency} + \text{Transfer Time} + \text{Controller Overhead}$$

Example:

- **Disk speed** = 3600 RPM
- **Seek time** = 10 ms
- **Controller overhead** = 0.1 ms
- **Track size** = 32 KB
- **Block size** = *k*

Step 1: Rotation Time

$$\text{Rotation time} = \frac{60}{\text{RPM}} = \frac{60}{3600} = 16.67\ \text{ms}$$

Step 2: Average Rotational Latency

Since latency = half a rotation on average:

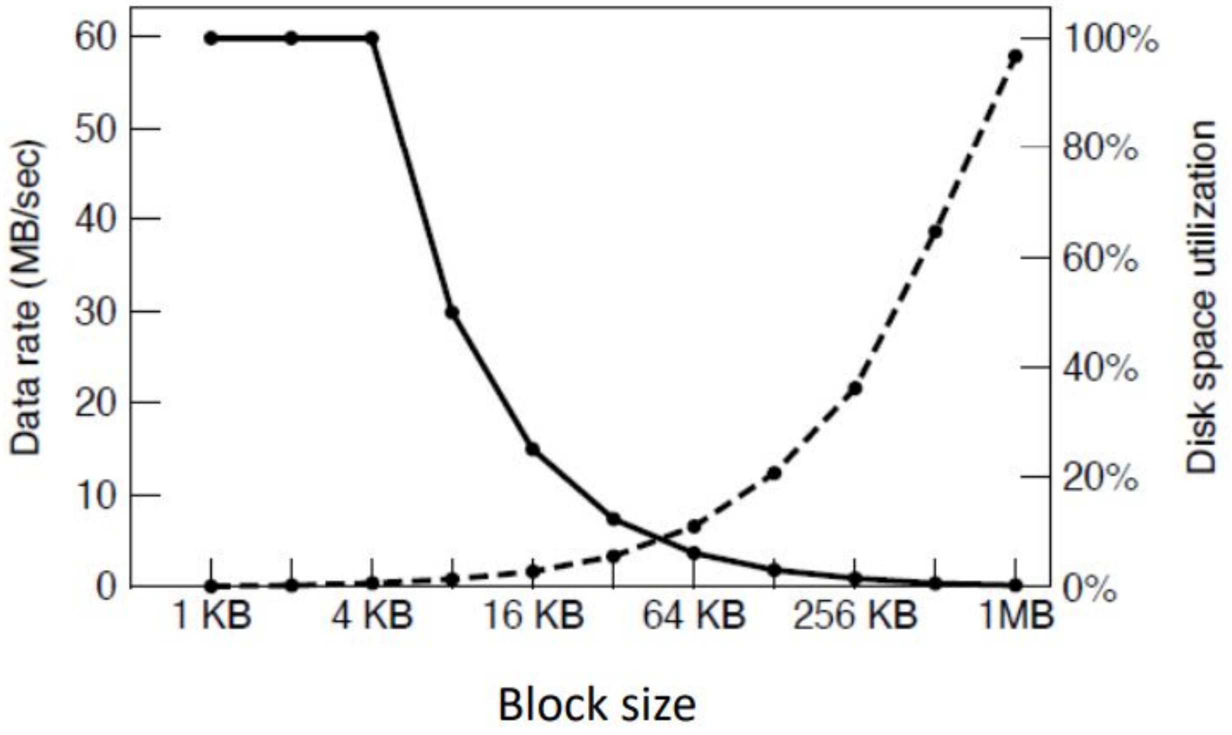
$$\text{Avg. latency} = \frac{16.67}{2} = 8.3\ \text{ms}$$

Step 3: Transfer Time

$$\text{Transfer time} = \left(\frac{\text{Block size (k)}}{\text{Track size (32KB)}}\right) \times 16.67\ \text{ms}$$

Data Rate vs Storage Efficiency

File size: 4KB



Step 4: Total Average Access Time

$$= 10\ \text{ms (seek)} + 8.3\ \text{ms (latency)} + \left(\frac{k}{32\ \text{KB}}\right) \times 16.67\ \text{ms} + 0.1\ \text{ms (overhead)}$$

Disk Scheduling

- OS aims to use disks efficiently by **reducing access time**.
- Access time depends heavily on **seek time** (head movement between tracks).
- **Seek time ≈ seek distance** (depends on how far the head must move).

Hence, disk scheduling algorithms (like FCFS, SSTF, SCAN, C-SCAN) are used to minimize **seek distance** → reduce total access time.

Disk Scheduling Overview

Sources of Disk I/O requests:

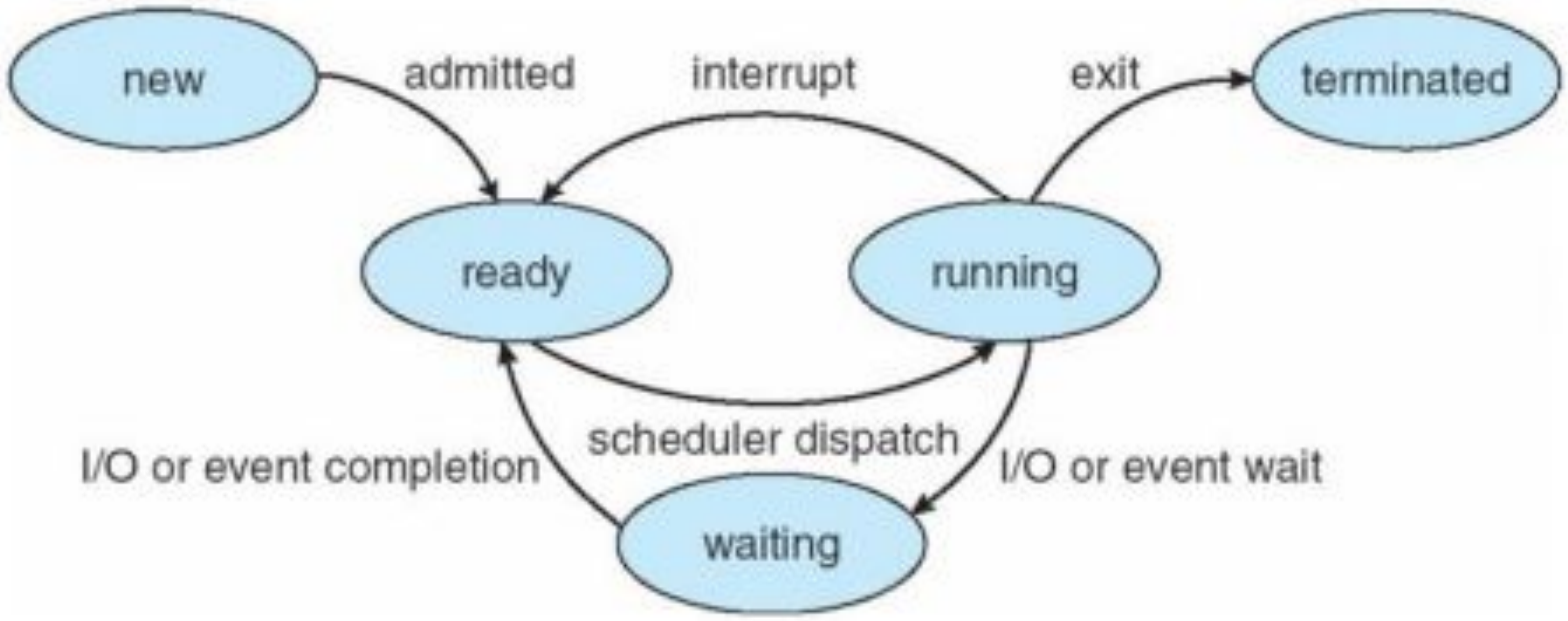
- System processes
- User processes

Request Handling:

- OS maintains a **queue of requests** for each disk or device.
- If the disk is **idle**, it can immediately handle the request.
- If the disk is **busy**, new requests are added to the queue.

Controller Role:

- Drive controllers often have **small buffers**.
- They can manage a queue of I/O requests of varying “depth.”


new
admitted
interrupt
exit
terminated
ready
running
scheduler dispatch
I/O or event completion
I/O or event wait
waiting

Disk Scheduling Overview

Example

- **Disk range** = 0 – 199 tracks
- **Request queue** = [98, 183, 37, 122, 14, 124, 65, 67]
- (Head starting position not given → usually assumed **head = 53** in such problems).

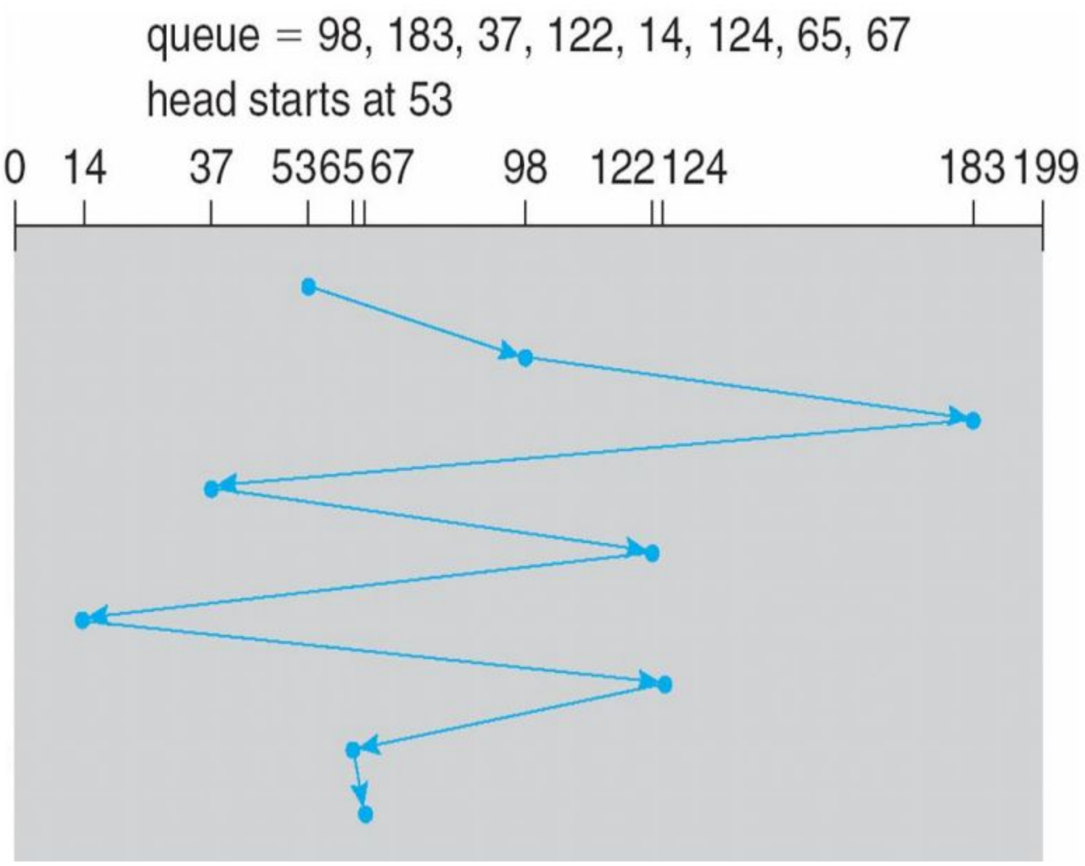
Common Disk Scheduling Algorithms

(a) FCFS (First Come First Serve)

- Serve requests in the order they arrive.
- Simple, but may cause long seek times.

Head = 53 → 98 → 183 → 37 → 122 → 14 → 124 → 65 → 67

Illustration shows total head movement of 640 cylinders



(b) SSTF (Shortest Seek Time First)

- Pick request nearest to current head.
- Reduces average seek time but may cause **starvation** (far requests delayed).

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

head starts at 53

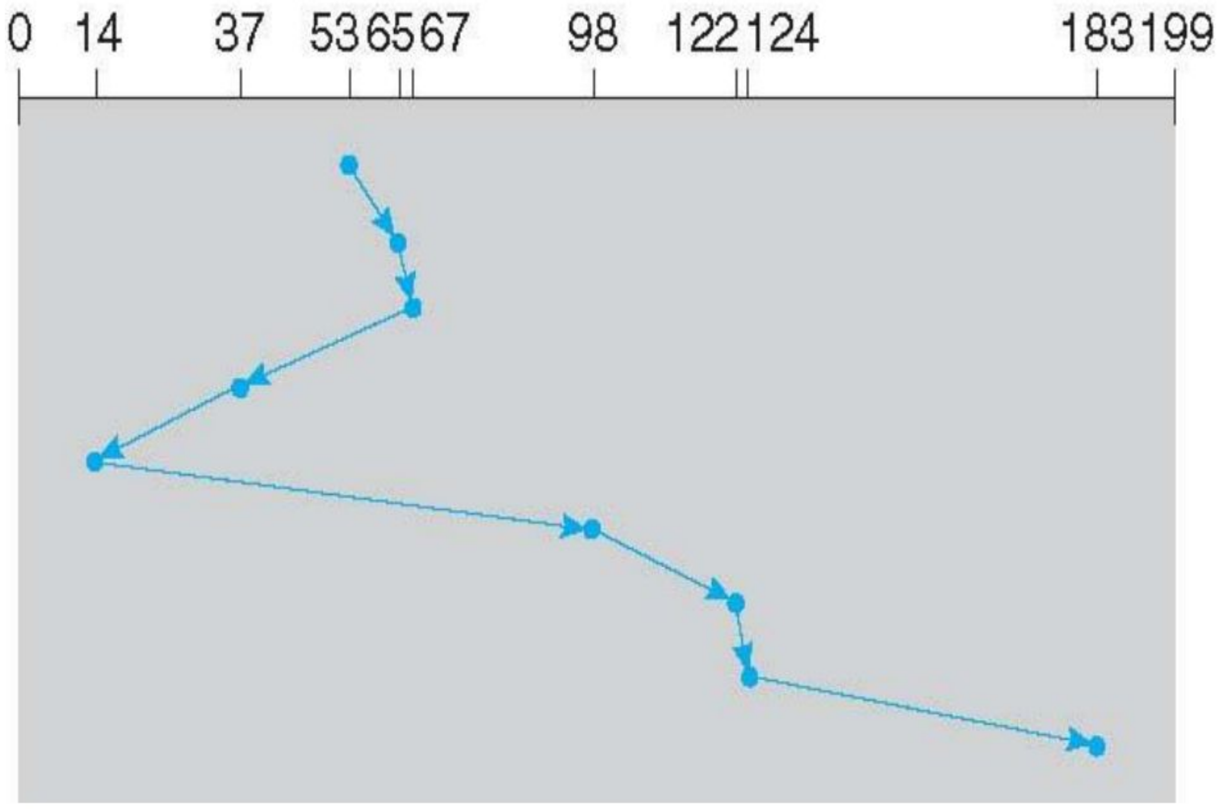
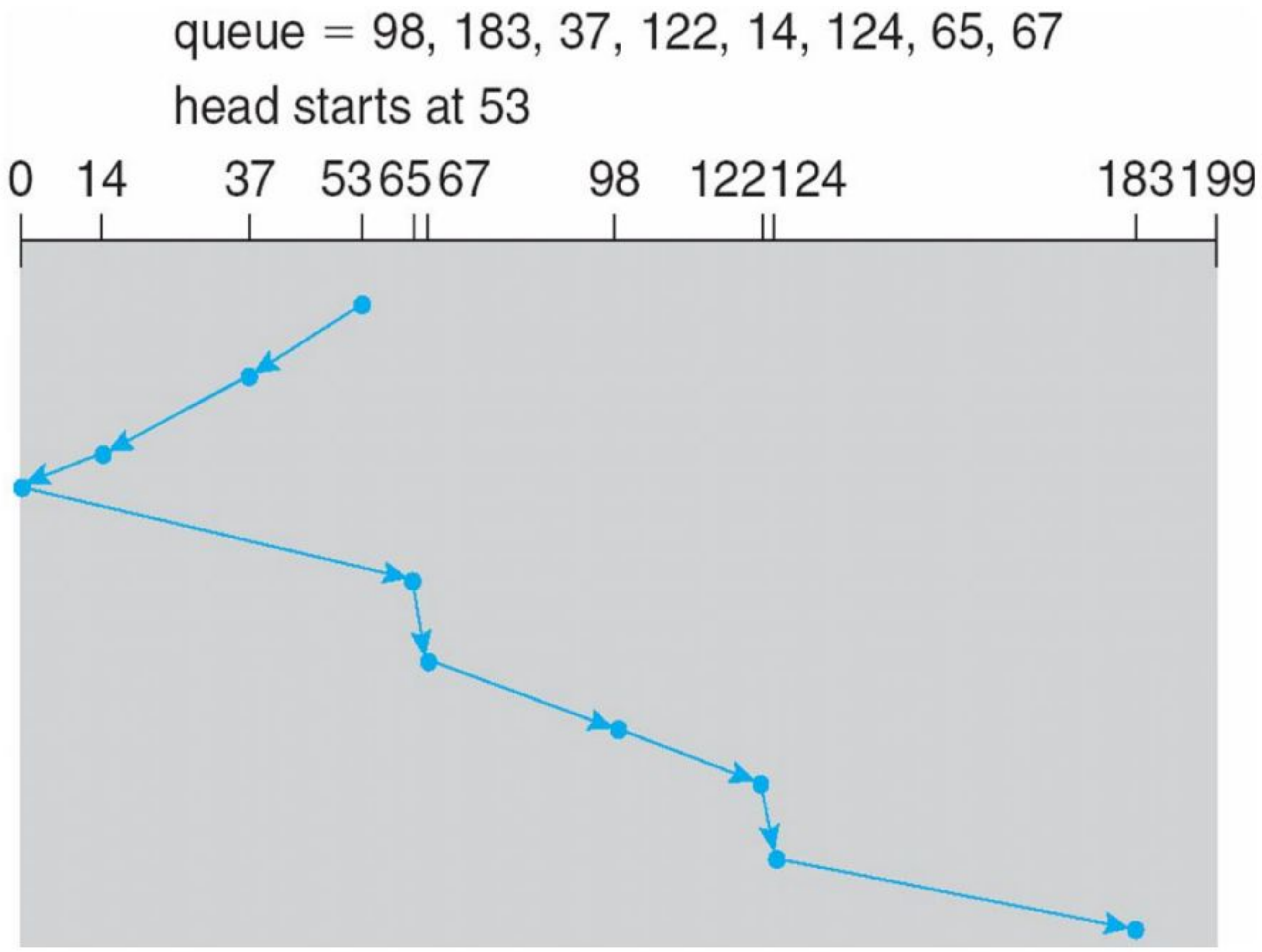


Illustration shows total head movement of 236 cylinders

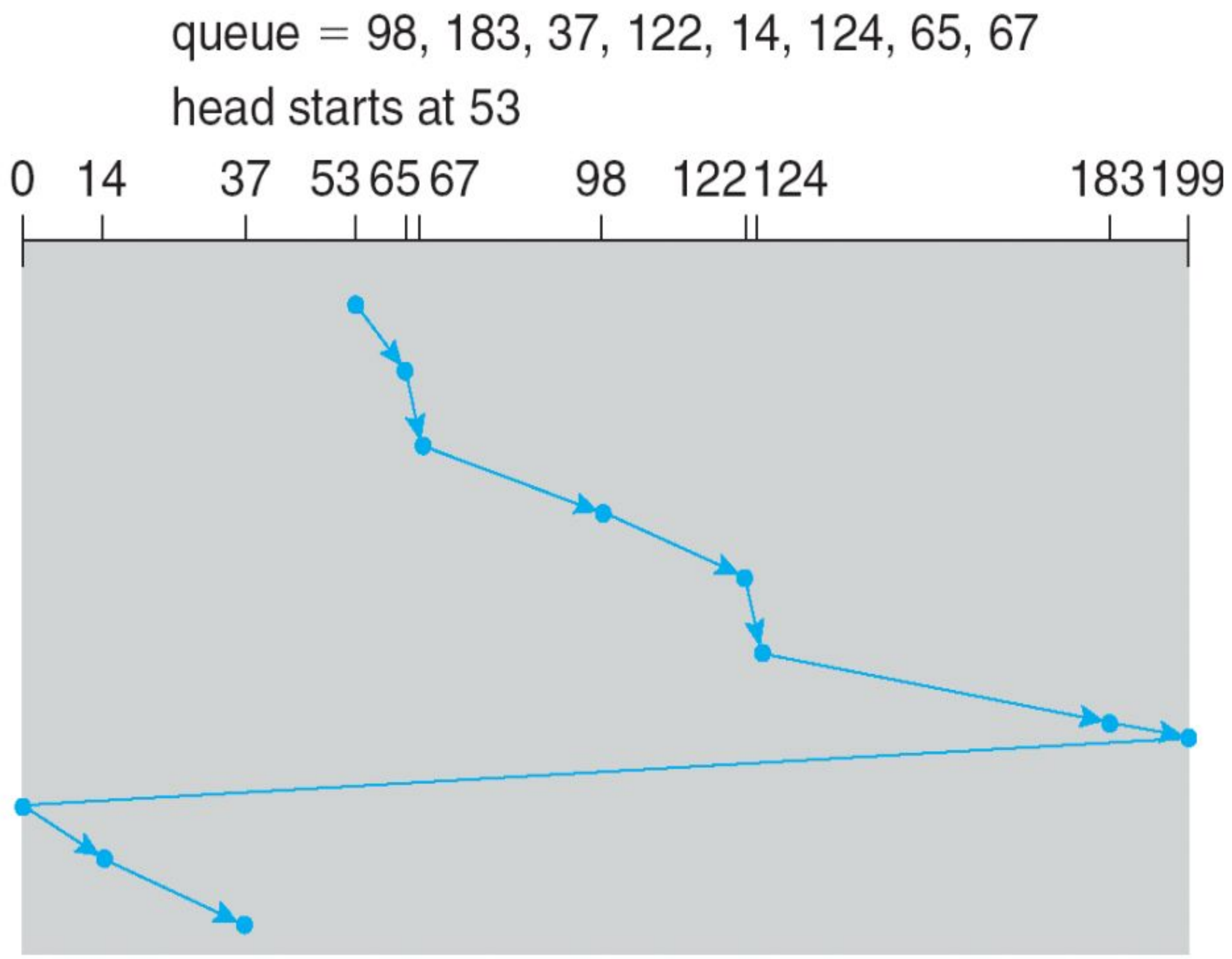
(c) SCAN (Elevator Algorithm)

- Head moves in one direction (say right), serving requests until the end, then reverses.
- Reduces starvation compared to SSTF.



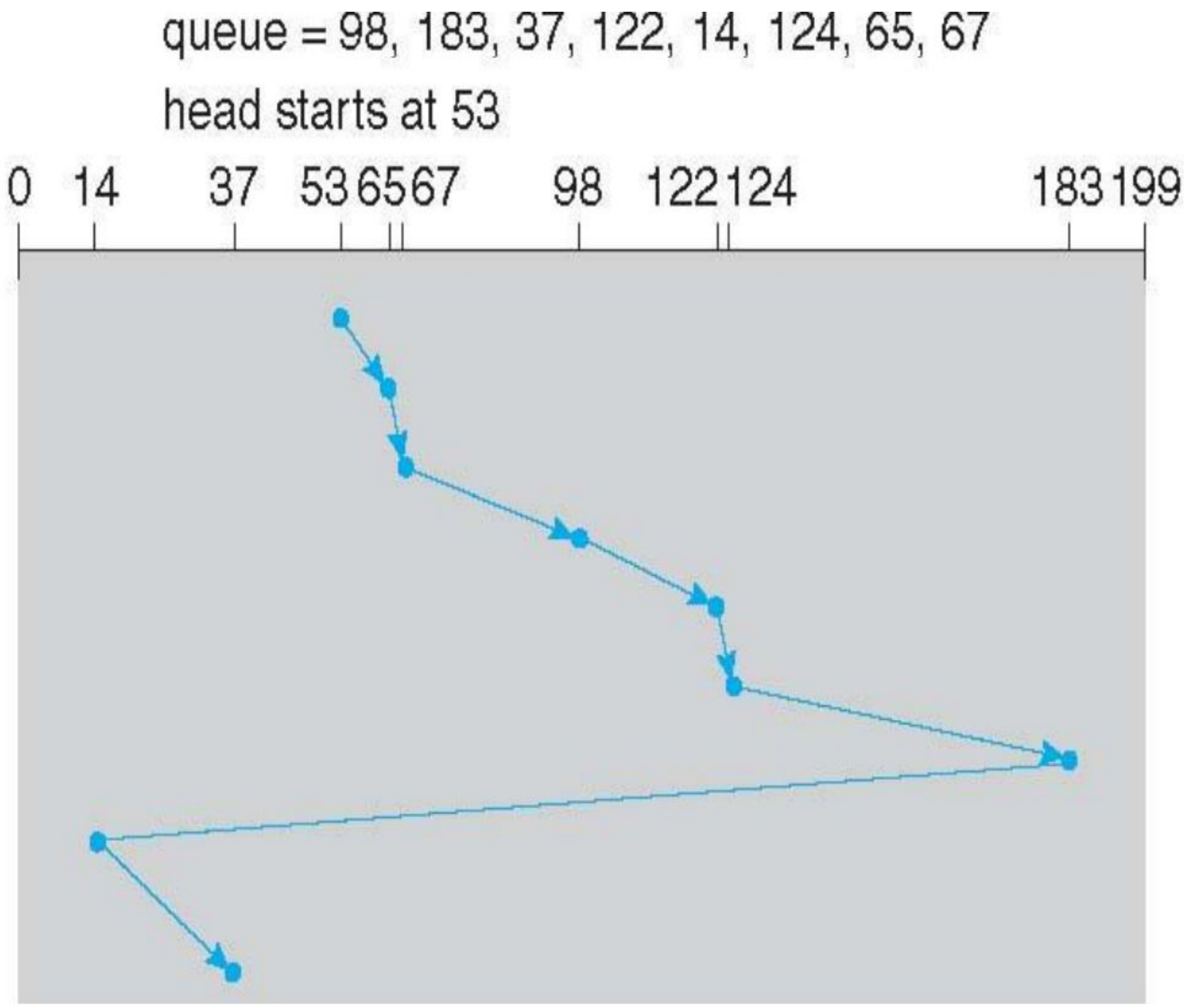
(d) C-SCAN (Circular SCAN)

- Head moves only in one direction, services requests, then jumps back to start.
- Provides **more uniform wait time**.



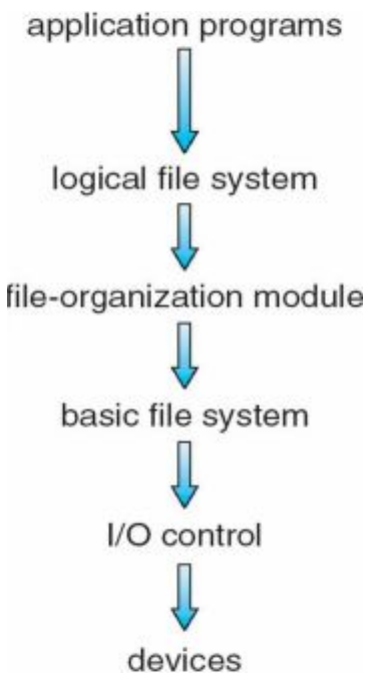
(e) LOOK & C-LOOK

- Same as SCAN / C-SCAN but head reverses or jumps back only up to the last request (not the physical end of disk).

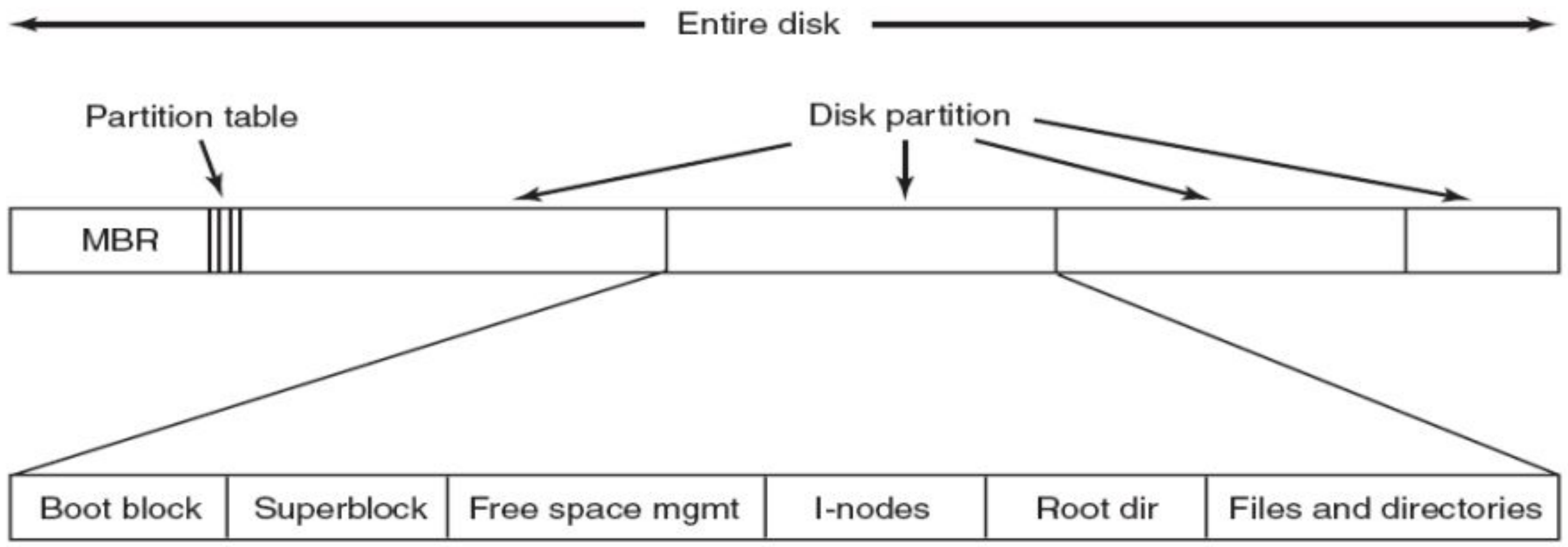


Selecting a Disk-Scheduling Algorithm

- Disk scheduling depends on the **file allocation method** (contiguous, indexed, etc.) and where directory/i-node information is stored.
- The scheduling algorithm should be a **separate OS module**, so it can be replaced if needed.
- **SSTF or LOOK** are commonly used as default algorithms.
- Many scheduling functions are handled inside the **disk controller**, since it knows the physical block details.
- OS-level scheduling may differ from controller scheduling because:
  - OS has other priorities (e.g., demand paging over normal I/O).
  - The disk controller doesn't know about these priorities.
  - The file organization module helps ensure requests are served in the right order.



Disk Layout


Entire disk
Partition table
Disk partition
MBR
Boot block
Superblock
Free space mgmt
I-nodes
Root dir
Files and directories

Disk Management

Low-Level (Physical) Formatting

- Disk is divided into **sectors** that the controller can read/write.
- Each sector stores: header + data + error correction code (ECC).
- Standard size = **512 bytes** (sometimes configurable).
- Maintains a **logical-to-physical address map**.

Logical Formatting (File System Creation)

- OS adds its own structures to manage files.
- Disk is divided into **partitions** (groups of cylinders).
- Each partition acts as a **logical disk**.
- File system structures created (FAT, directories, inodes, free space list).

Application I/O Interface

The OS provides a standard interface for all I/O devices.

Applications can open and use files without knowing disk hardware details.

New devices can be added without affecting applications or the OS structure.

DISCUSSION & REVISION

1. Which OS function manages storage devices like HDDs and SSDs?
2. Which disk scheduling algorithm picks the nearest request?
3. What is half of the disk rotation time called?
4. Which formatting creates a file system like FAT or NTFS?
5. In SCAN scheduling, the head moves like which machine?
6. What does the OS provide to make I/O devices appear uniform?

REFERENCES

- https://www.geeksforgeeks.org/operating-systems/disk-management-in-operating-system/
- https://cse.iitkgp.ac.in/~bivasm/os_notes/Disk_IO_v3.pdf