Storage Management, Protection and Security

Module 7

Dr. Naveenkumar J

Associate Professor,

PRP- 217 - 4

Hard Disk Drive

- A Hard Disk Drive (HDD) is a **non-volatile data storage device** commonly found in computers, laptops, servers, and many electronic devices.

- It **uses magnetic storage** to **store and retrieve digital information** by **spinning multiple rigid disks (platters) coated with magnetic material**.

HDD Key Characteristics

- **Function**: Stores operating systems, applications, documents, media files, and virtually all user and system data.

- **Structure**: Contains spinning platters, read/write heads, actuator arm, controller electronics, and is housed in a protective casing.

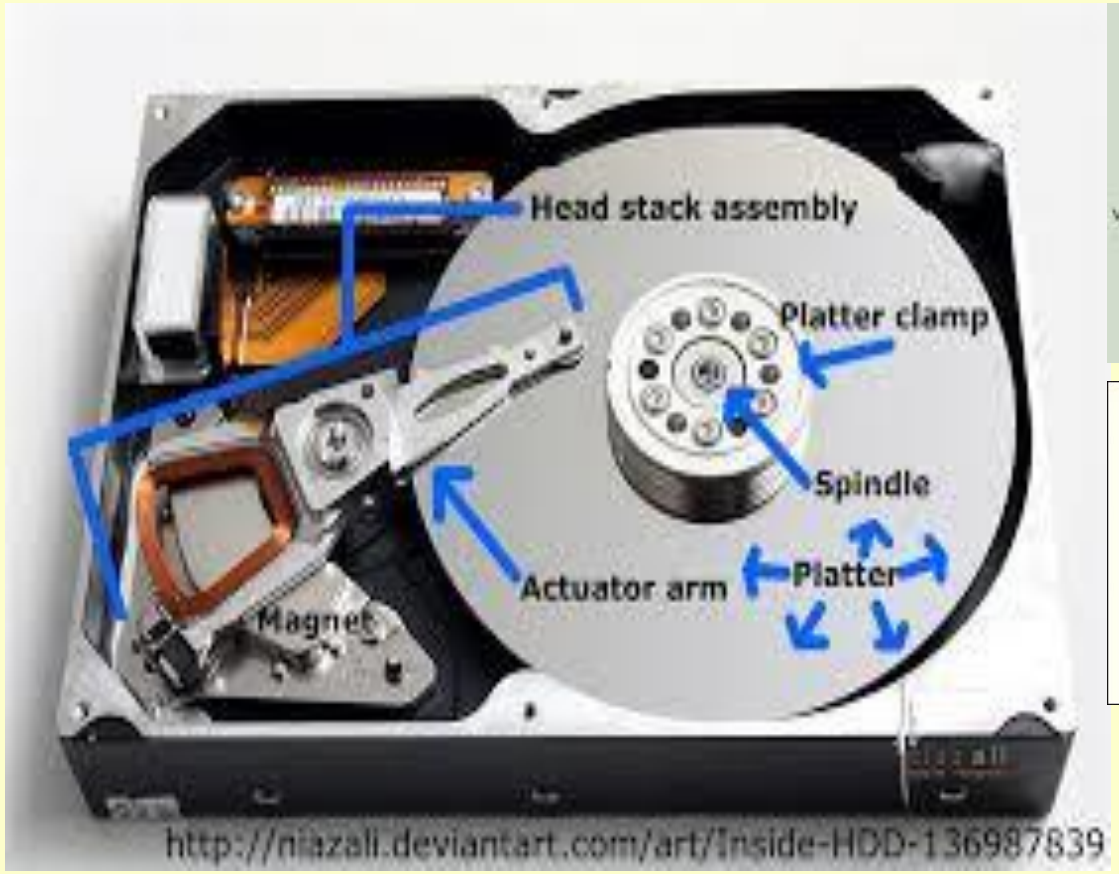
- **Working Principle**: Data is encoded as tiny magnetized zones on the platters. As platters spin, read/write heads move very close to the surface to read or change the magnetization, representing binary data (0s and 1s).

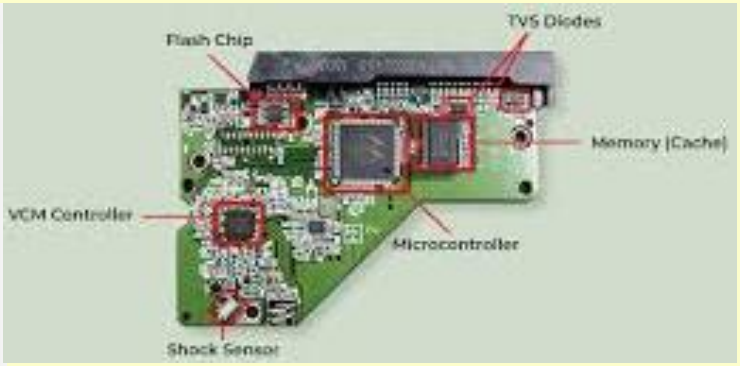
- **Capacity**: Modern HDDs range from hundreds of gigabytes (GB) to multiple terabytes (TB)

HDD – Simple Working

- The platters spin at high speed (5400–15000 RPM).
- **Read/write heads** attached to arms quickly move to locate the right place on the platter.
- The **controller board manages all operations and communicates** with the computer via a connector (SATA, IDE, SCSI, etc.).
- Data stays intact even when the device is powered off (non-volatile).

HDD – Components





HDD – Components

- **Platters**
  - Flat, circular disks (usually made of aluminum or glass, coated with magnetic material) where data is actually stored.
  - A standard HDD contains multiple platters stacked vertically; each surface can store data.

- **Spindle**
  - The rotating axis/wheel holding the platters in place.
  - Spins the platters at high speeds (typically 5400, 7200, or 10,000+ RPM), enabling rapid data access.

HDD – Components

- **Read/Write Arm (Actuator Arm)**
  - A thin, pivoting metallic arm that moves across the platters.
  - Positions the read/write head over the correct track on the platter for reading or writing data.

- **Actuator (Voice Coil Motor)**
  - The device (often an electromagnet) that controls the precise movement of the read/write arm.
  - Allows rapid, accurate movement to the required data location.

HDD – Components

- **Read/Write Head**
  - Microscopic electromagnetic devices located at the tip of each actuator arm.
  - Flies just above the platter surface, reading or altering the magnetic fields to store/retrieve bits.
  - Each platter surface has its own dedicated head.

- **Controller Board (PCB)**
  - The electronic circuit board mounted on the underside of the drive.
  - Manages all internal communication, power regulation, and data transfer between the HDD and host (computer).

HDD – Components

- **Connector Interface**
  - External data and power connectors (SATA, PATA/IDE, SCSI, etc.) for connecting to the motherboard and power supply.

HDD – Components

Component	Function/Role
Casing	Protection and physical support
Platters	Magnetic data storage
Spindle	Rotates platters for data access
Read/Write Arm	Positions the heads over data tracks
Actuator	Moves the arm precisely
Read/Write Head	Reads/writes magnetic data
Controller Board	Controls logic, communication, power
Connector	Connects power/data to computer

Disk Scheduling Algorithm

- A disk scheduling algorithm is a strategy used by operating systems to **decide the order in which disk I/O** (input/output) **requests are serviced**.

- These algorithms are vital in multi-tasking systems or environments with heavy disk usage, because multiple read/write requests may arrive simultaneously or out of sequence.

Disk Scheduling Algorithm – Why important

- Minimizes overall waiting time for disk accesses.
- Reduces mechanical movement of the disk's read/write heads, decreasing seek time and improving performance.
- Prevents starvation and ensures fairness among requests.

Disk Scheduling Algorithm – Goals

- Optimize:
  - **Seek time** (time to move disk head between tracks)
  - **Rotational latency** (time for required sector to rotate under head)
  - **Throughput** (number of serviced requests per time unit)
  - **Fairness** (avoid neglecting distant requests)

Disk Scheduling Algorithm – Metrics

- **Seek Time**
  - The time taken for the **disk's read/write head to move from its current position to the track containing the desired data**.
- Example
  - Suppose the head is at track 20 and your data is at track 80. If each track jump takes 0.5 ms, total seek time = 60 tracks × 0.5 ms = 30 ms.

Disk Scheduling Algorithm – Metrics

- **Rotational Latency**
  - The time it takes for the desired disk sector to rotate under the read/write head after the head has reached the correct track. It depends on disk rotation speed.

- Example:
  - A disk spins at 7200 RPM (Revolutions Per Minute). One full rotation takes 60sec/7200 = 0.0083sec = 8.3ms
  - The average rotational latency is half of this: 4.15 ms.

Disk Scheduling Algorithm – Metrics

- **Transfer Time**
  - The time required to actually transfer (read/write) the data once the head is positioned correctly.
- **Example:**

If the disk can transfer data at 100 MB/s and you want to read 1 MB, Transfer Time = $\frac{1\ MB}{100\ MB/s} = 0.01\ sec = 10\ ms.$

Disk Scheduling Algorithm – Metrics

- **Throughput**
  - The number of disk requests serviced in a given period (requests per second).

- Example:
  - If 70 I/O requests are completed in 10 seconds,

$$\text{Throughput} = \frac{70}{10} = 7\ requests/sec.$$

Disk Scheduling Algorithm – Metrics

- **Fairness**
  - The extent to which requests—whether near or far from the current head position—are serviced without undue delay or starvation.

- **Example:**
  - SSTF may be fast for short seeks, but is less fair (may starve far requests), while SCAN and C-SCAN offer balanced fairness.

Disk Scheduling AlgorithmS

Algorithm	Definition
FCFS (First-Come, First-Served)	Requests are processed in the exact order they arrive. Simple and fair, but not always efficient.
SSTF (Shortest Seek Time First)	The request closest to the current position of the disk arm is handled first, which minimizes immediate head movement.
SCAN (Elevator)	The disk arm moves in one direction, servicing all requests along its path. When it reaches the end, it reverses and services requests on the way back, like an elevator.
C-SCAN (Circular SCAN)	The disk arm moves in one direction, servicing requests until it reaches the end. Then, it quickly returns to the start (without servicing on the way back) and repeats.
LOOK	The disk arm moves only as far as the last request in each direction, then reverses, instead of going all the way to the physical end of the disk.
C-LOOK	The arm goes from the current position to the furthest request in one direction, then jumps to the furthest request in the other direction and continues, skipping over empty parts of the disk.

Disk Scheduling Algorithm

- Suppose the head of moving disk with 200 tracks, numbered 0 to 199 is currently serving a request at track 132 and has just finished a request at track 120.
- If the queue of requests is kept in the FIFO order 38, 55, 86, 123, 147, 91, 177, 115, 94, 150, 100, 175 and 130, 185.
- What is total head movement to satisfy these request for the all-disk scheduling algorithms?

Disk Scheduling Algorithm

- **Given Data**
  - Tracks: 0 to 199
  - Current Head Position: 132
  - Previous Head Position: 120 (**Indicates the head is moving towards higher track numbers**)
  - Request Queue (FIFO): 38, 55, 86, 123, 147, 91, 177, 115, 94, 150, 100, 175, 130, 185

Disk Scheduling Algorithm - FCFS

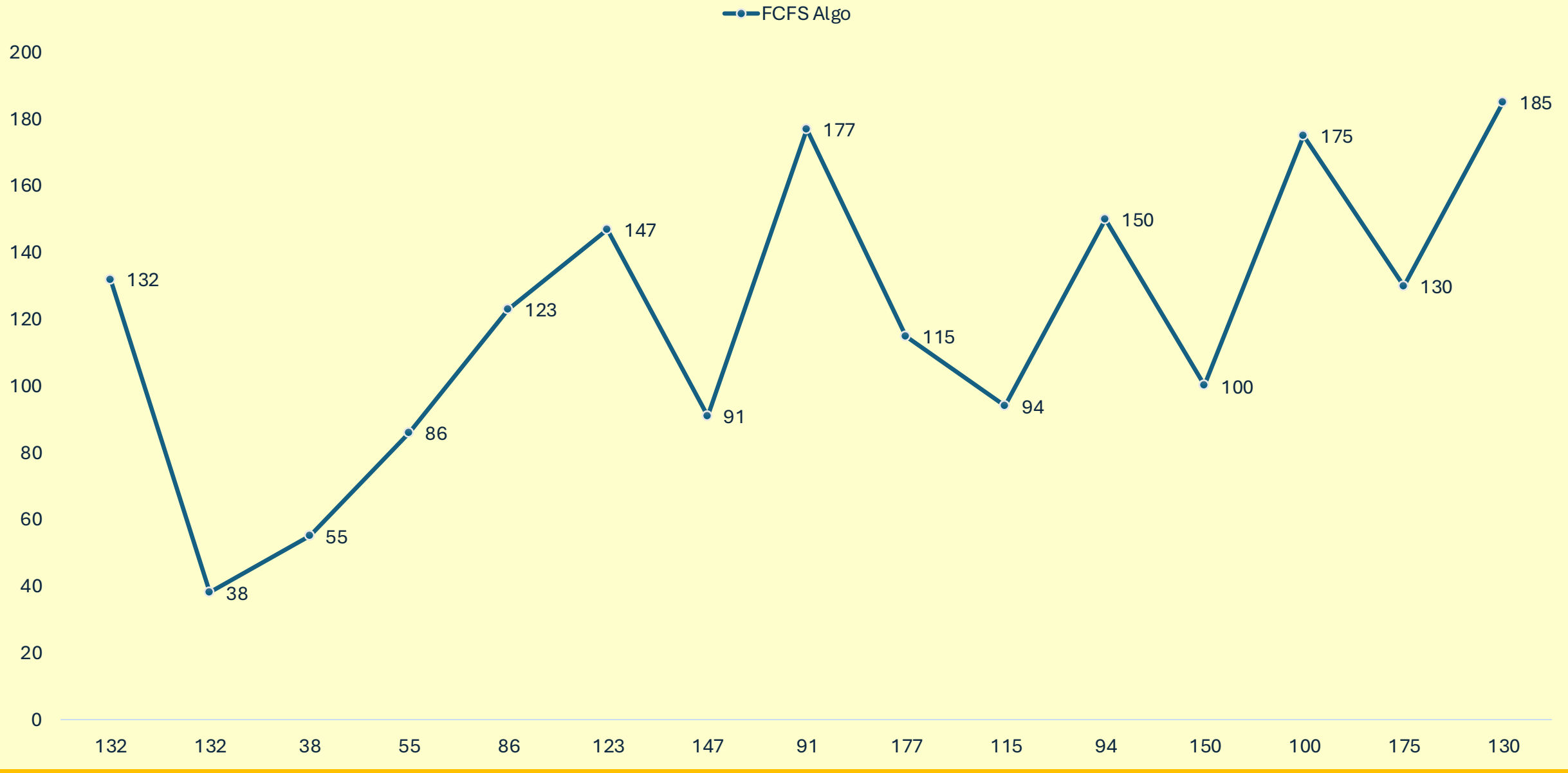
- In the FCFS algorithm, the requests are serviced in the order they appear in the queue.

- **Path: 132 → 38 → 55 → 86 → 123 → 147 → 91 → 177 → 115 → 94 → 150 → 100 → 175 → 130 → 185**

Disk Scheduling Algorithm - FCFS

Step	Current Head Position	Request Serviced	Head Movement	Total Movement
1	132	38	|132 - 38| = 94	94
2	38	55	|38 - 55| = 17	111
3	55	86	|55 - 86| = 31	142
4	86	123	|86 - 123| = 37	179
5	123	147	|123 - 147| = 24	203
6	147	91	|147 - 91| = 56	259
7	91	177	|91 - 177| = 86	345
8	177	115	|177 - 115| = 62	407
9	115	94	|115 - 94| = 21	428
10	94	150	|94 - 150| = 56	484
11	150	100	|150 - 100| = 50	534
12	100	175	|100 - 175| = 75	609
13	175	130	|175 - 130| = 45	654
14	130	185	|130 - 185| = 55	**709**

Disk Scheduling Algorithm - FCFS

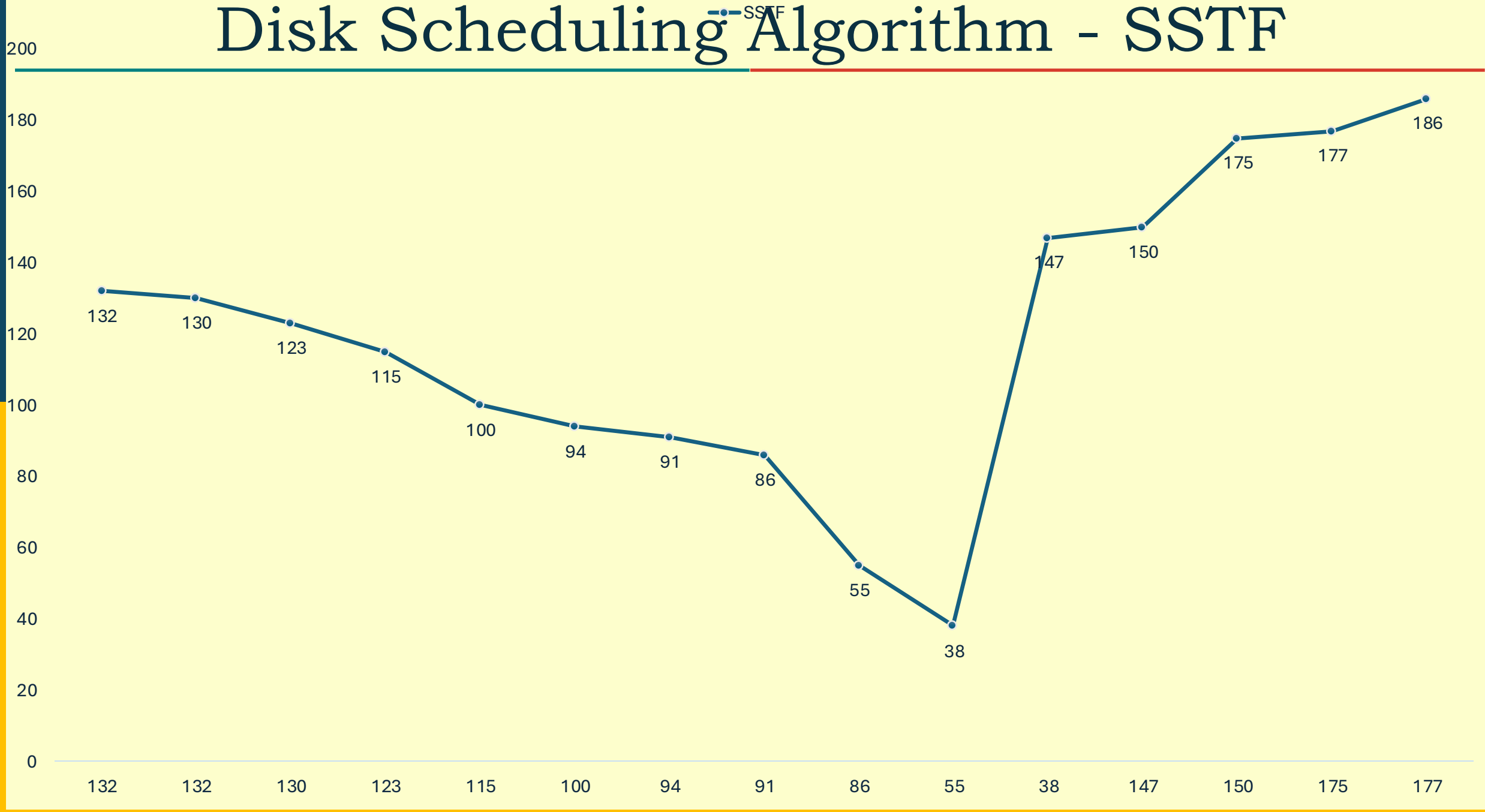


Disk Scheduling Algorithm - SSTF

Step	Current Head Position	Request Serviced	Head Movement	Total Movement
1	132	130	|132 - 130| = 2	2
2	130	123	|130 - 123| = 7	9
3	123	115	|123 - 115| = 8	17
4	115	100	|115 - 100| = 15	32
5	100	94	|100 - 94| = 6	38
6	94	91	|94 - 91| = 3	41
7	91	86	|91 - 86| = 5	46
8	86	55	|86 - 55| = 31	77
9	55	38	|55 - 38| = 17	94
10	38	147	|38 - 147| = 109	203
11	147	150	|147 - 150| = 3	206
12	150	175	|150 - 175| = 25	231
13	175	177	|175 - 177| = 2	233
14	177	185	|177 - 185| = 8	**241**

The head moves to the closest pending request.

Disk Scheduling Algorithm - SSTF



Disk Scheduling Algorithm - SCAN

Current Position	Direction	Path Segment & Requests Serviced	Justification	Head Movement	Total Movement
132	↑	132 → 147 → 150 → 175 → 177 → 185	Continue in initial direction (UP).	|185 - 132| = 53	53
185	↑	185 → 199	Continue to end of disk.	|199 - 185| = 14	67
199	↓	Reverse Direction	At disk end, reverse to ↓.	0	67
199	↓	199 → 130 → 123 → ... → 38	Service all remaining requests.	|199 - 38| = 161	**228**

Sweep in one direction servicing all requests, then reverse at the disk's end and sweep back.

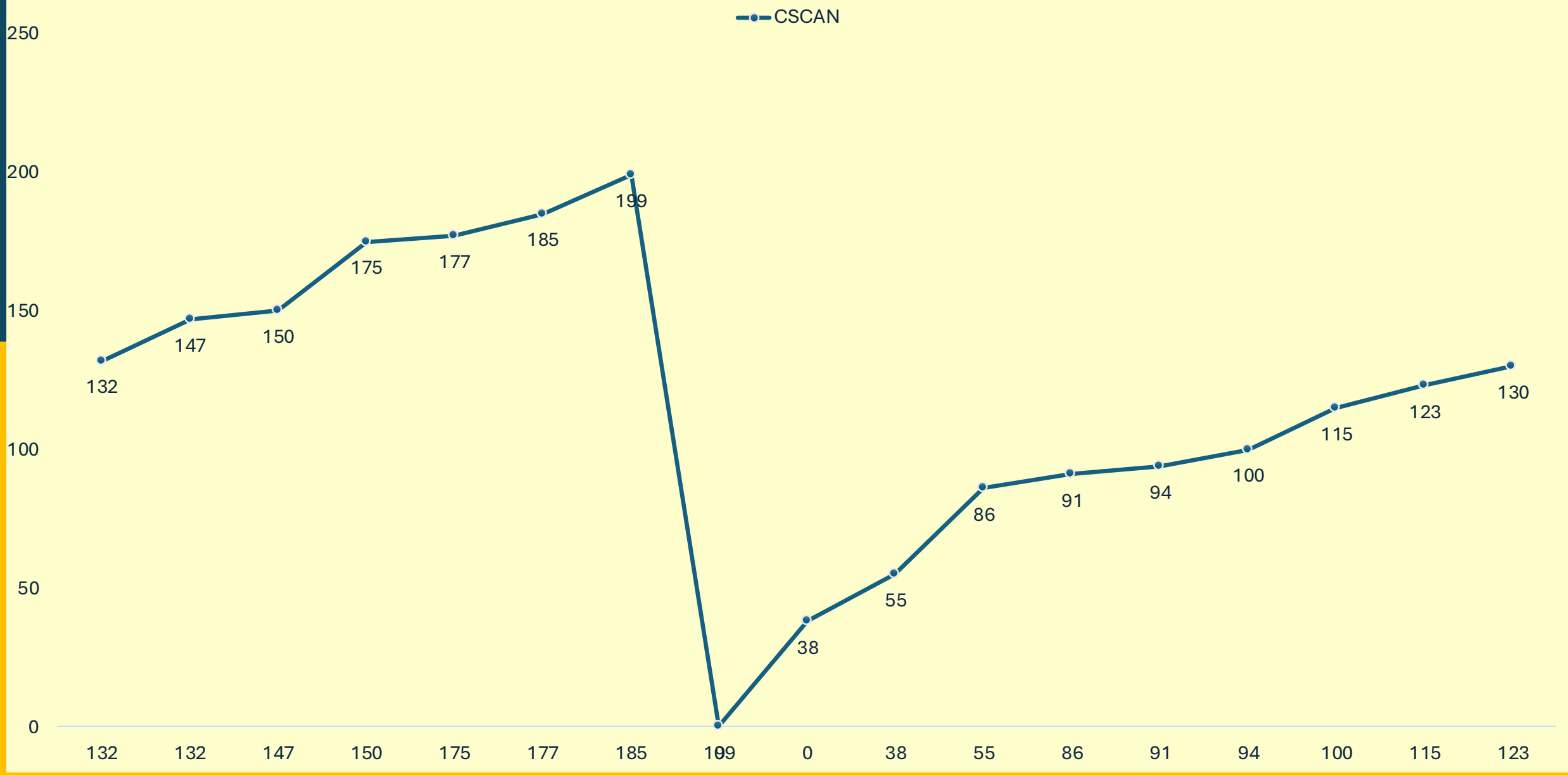
Disk Scheduling Algorithm - SCAN

Disk Scheduling Algorithm - CSCAN

Current Position	Direction	Path Segment & Requests Serviced	Justification	Head Movement	Total Movement
132	↑	132 → 147 → 150 → 175 → 177 → 185	Continue in initial direction (UP).	|185 - 132| = 53	53
185	↑	185 → 199	Continue to end of disk.	|199 - 185| = 14	67
199	JUMP	199 → 0	At disk end, jump to start.	|199 - 0| = 199	266
0	↑	0 → 38 → 55 → ... → 130	Service requests from start in UP direction.	|130 - 0| = 130	**396**

Sweep in one direction only. After reaching the end, jump back to the start and sweep again.

Disk Scheduling Algorithm - CSCAN



Disk Scheduling Algorithm - Look

Current Position	Direction	Path Segment & Requests Serviced	Justification	Head Movement	Total Movement
132	↑	132 → 147 → 150 → 175 → 177 → 185	Continue in initial direction (UP).	|185 - 132| = 53	53
185	↓	Reverse Direction	At last request (185), reverse to ↓.	0	53
185	↓	185 → 130 → 123 → ... → 38	Service all remaining requests.	|185 - 38| = 147	**200**

Like SCAN, but reverse direction after the last request in that direction (don't go to the end).

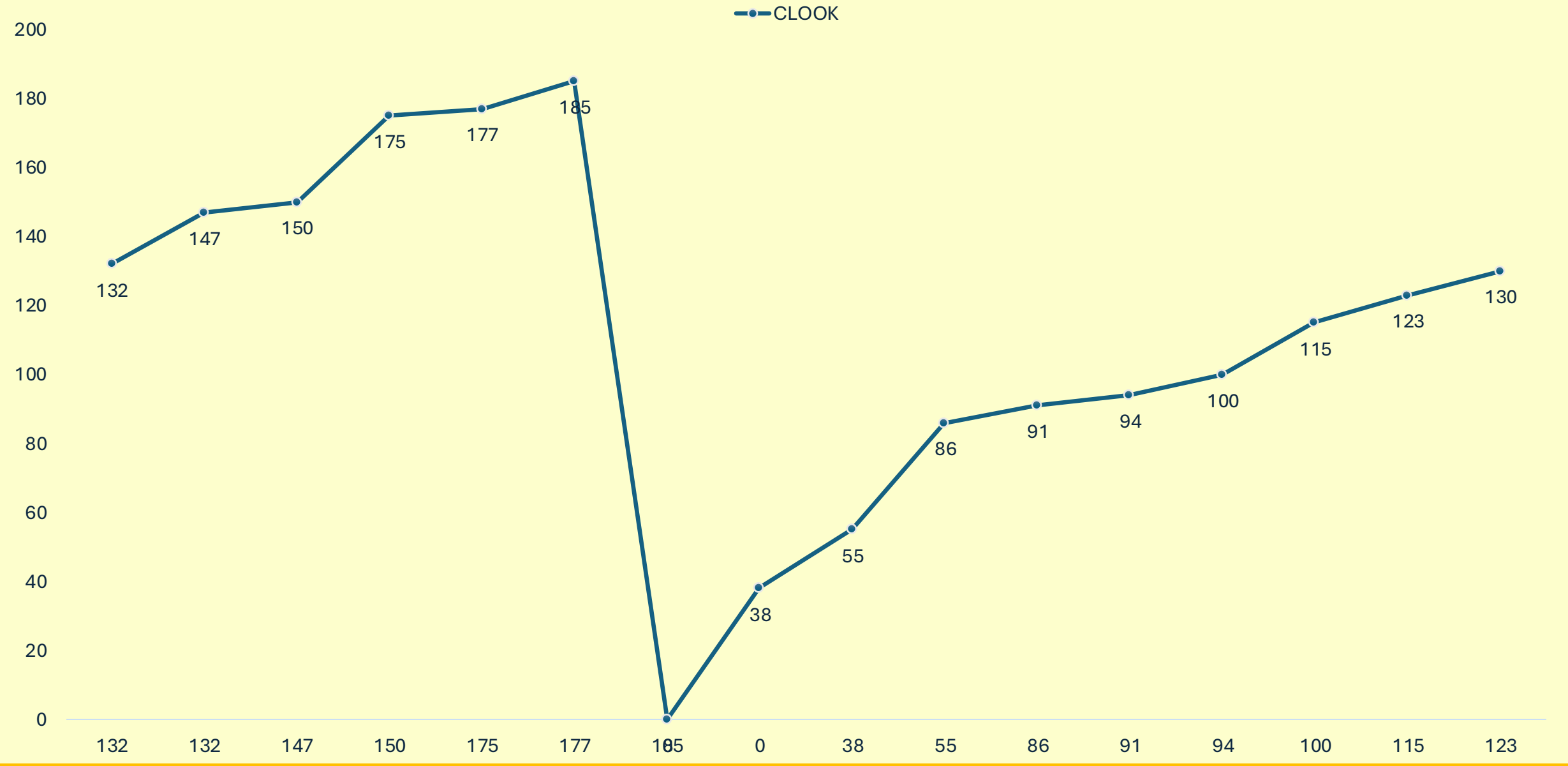
Disk Scheduling Algorithm - Look

Disk Scheduling Algorithm - CLook

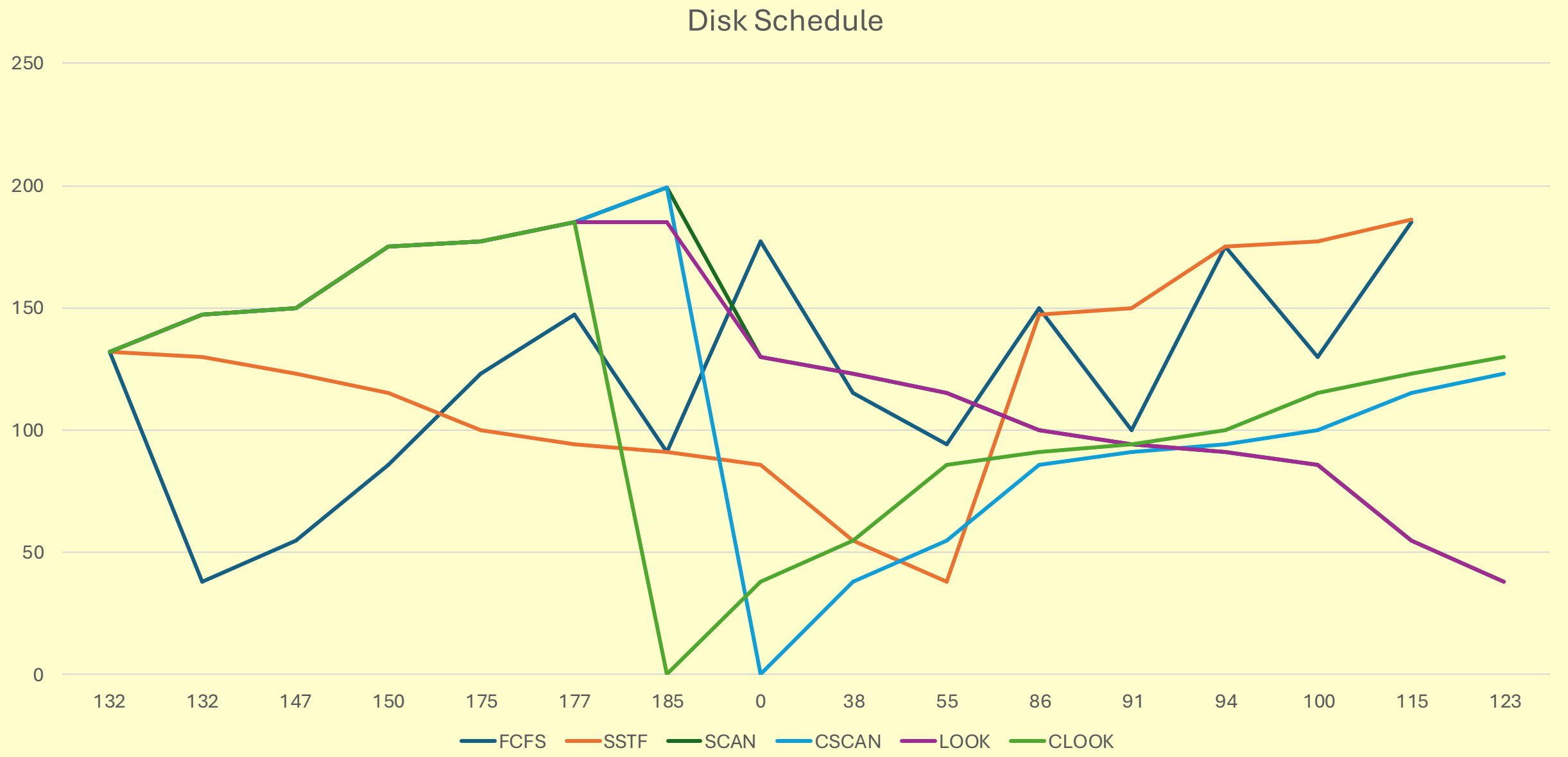
Current Position	Direction	Path Segment & Requests Serviced	Justification	Head Movement	Total Movement
132	↑	132 → 147 → 150 → 175 → 177 → 185	Continue in initial direction (UP).	|185 - 132| = 53	53
185	JUMP	185 → 38	At last request, jump to first in queue.	0 (jump, not sweep)	53
38	↑	38 → 55 → 86 → … → 130	Service remaining requests from new start.	|130 - 38| = 92	**145**

Like C-SCAN, but jump from the last request in one direction to the first request in the queue.

Disk Scheduling Algorithm - CLook



Disk Scheduling Algorithm - All



Disk Scheduling Algorithm

- Suppose the head of moving disk with 1000 tracks, numbered 0 to 999 is currently serving a request at track 132 and has just finished a request at track 120.
- If the queue of requests is kept in the FIFO order 358, 555, 861, 123, 147, 911,177, 115, 994, 150 , 100, 175, and 130, 185
- what is total head movement to satisfy these request for the following disk scheduling algorithms?
  - (a) FCFS (b)LOOK (C) SCAN