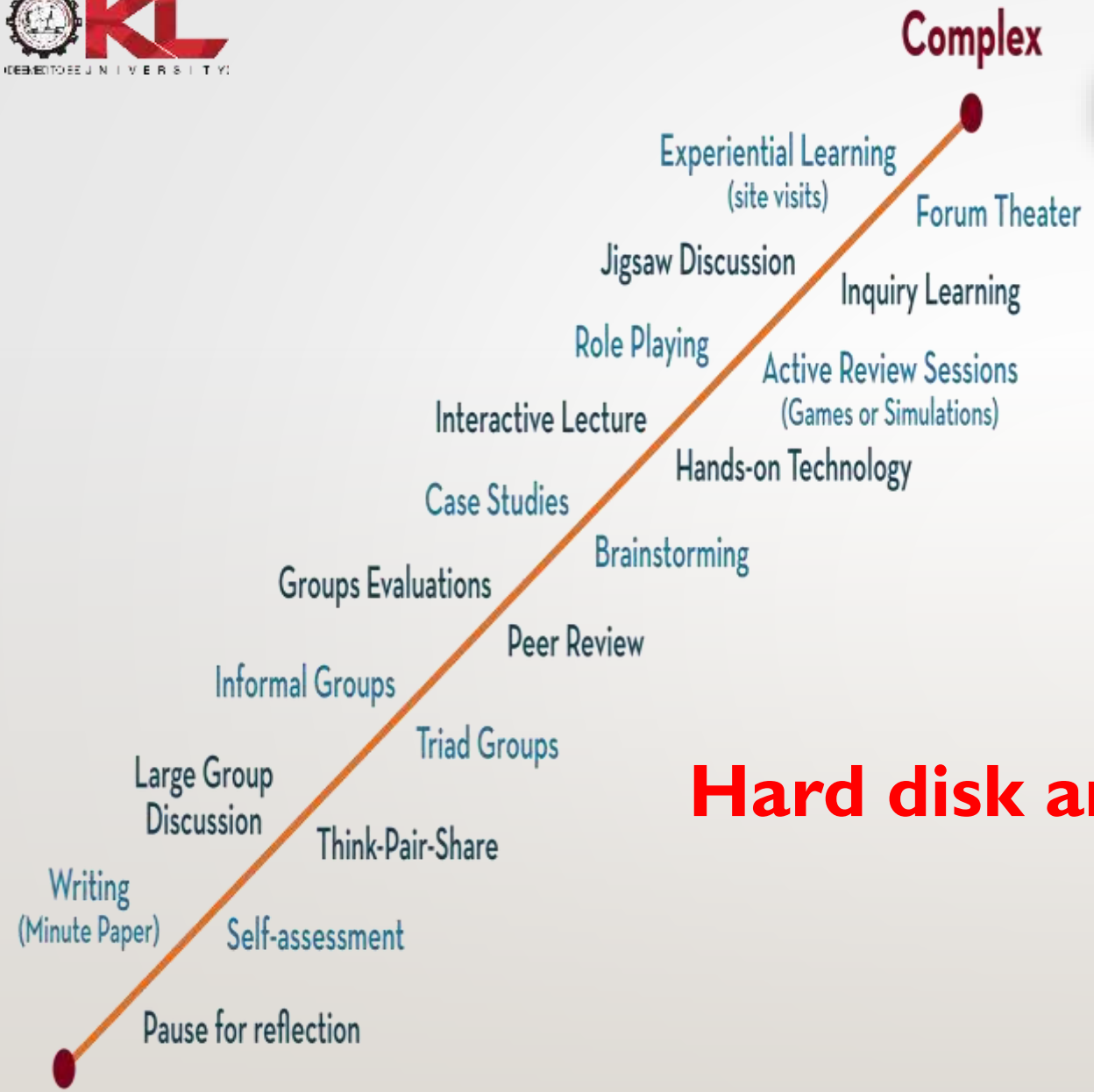


CO4

Session - 2

COURSE NAME: OPERATING SYSTEMS

COURSE CODE: 23CS2104R/A

Hard disk and Disk Scheduling Algorithms

AIM OF THE SESSION

To familiarize students with the basic concept of Hard Disk Drives.

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. Demonstrate what is meant by Hard Disk Drive.
2. Demonstrate what is meant by The Rotational Delay.
3. Describe the types of Disk Scheduling Algorithms.
4. Describe the Advantages and Disadvantages of Disk Scheduling Algorithms.

LEARNING OUTCOMES

At the end of this session, you should be able to:

1. Defines what is meant by Hard Disk.
2. Defines Disk Scheduling Algorithms
3. Summarizes the Role of Hard Disk.

HARD DISK DRIVER

- Hard disk driver have been **The main form of persistent data storage** in computer systems for decades.
  - The drive consists of a large number of **sectors** (512-byte blocks).

  Address Space :

  - We can view the disk with n sectors as an array of sectors; 0 to n-1.

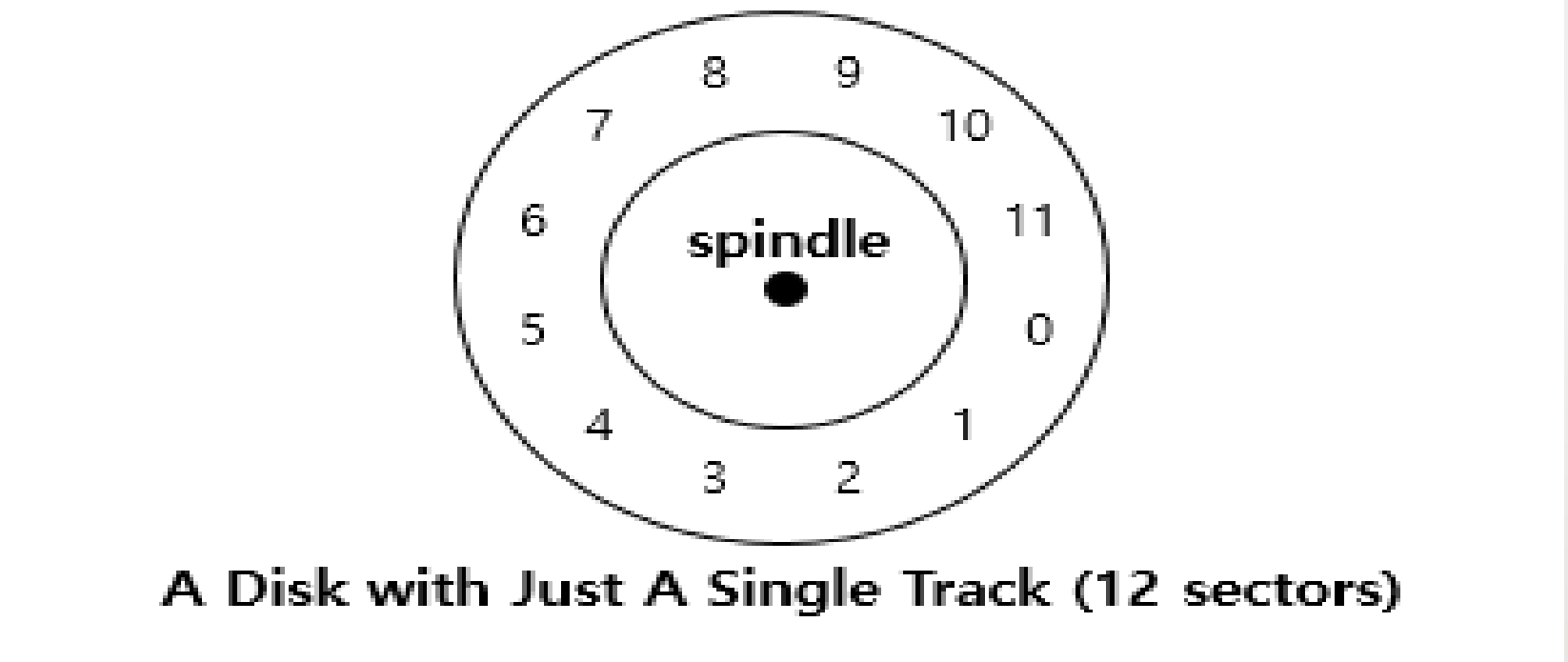
INTERFACE

- The only guarantee is that a single 512-byte write is **atomic**.

- Multi-sector operations are possible.
  - Many file systems will read or write 4KB at a time.
  - **Torn write**:
    - If an untimely power loss occurs, only a portion of a larger write may complete.

- Accessing blocks in **a contiguous chunk** is the fastest access mode.
  - A sequential read or write
  - Much faster than any more random access pattern.

BASIC GEOMETRY



A Disk with Just A Single Track (12 sectors)

Platter (Aluminum coated with a thin magnetic layer)

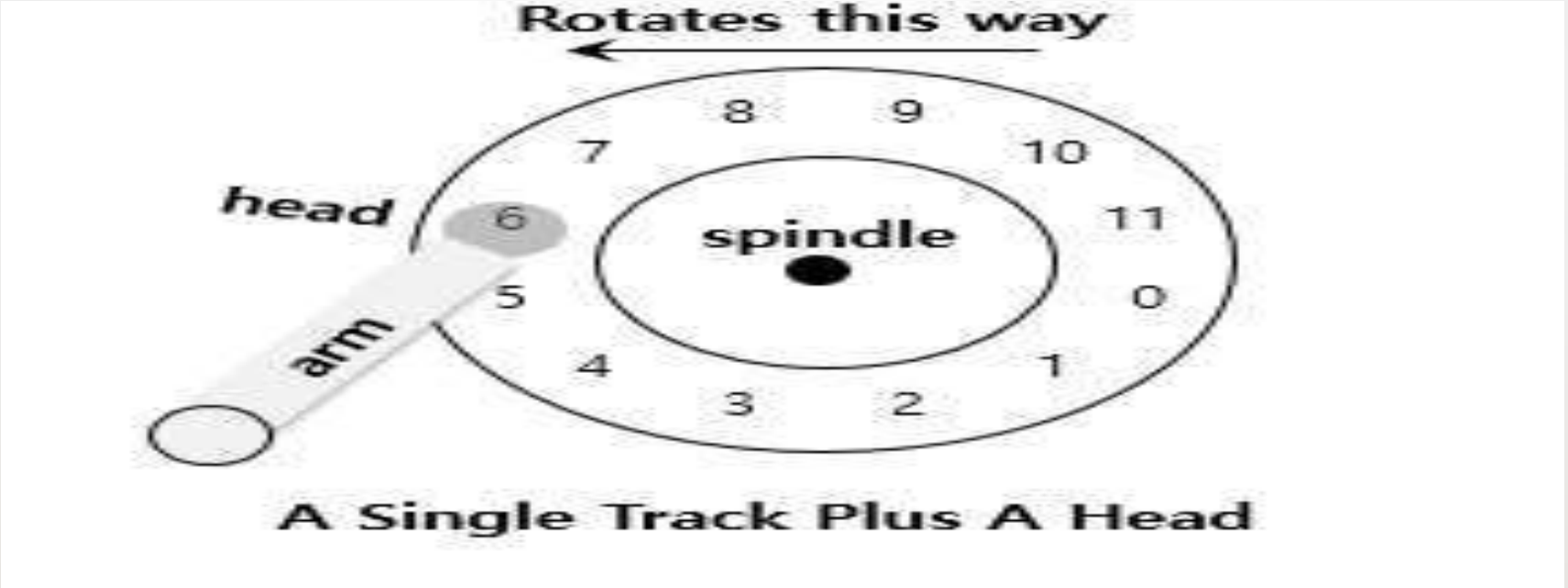
- A circular hard surface
- Data is stored persistently by inducing magnetic changes to it.
- Each platter has 2 sides, each called a **surface**.

BASIC GEOMETRY (CONT.)

- **Spindle**
  - The spindle is connected to a motor that spins the platters around.
  - The rate of rotations is measured in **RPM** (Rotations Per Minute).
    - Typical modern systems RPM values: 7,200 RPM to 15,000 RPM.
    - E.g..., 10000 RPM: A single rotation takes about 6 ms.

- **Track**
  - Concentric circles of sectors.
  - Data is encoded on each surface in a track.
  - A single surface contains many thousands and thousands of tracks.

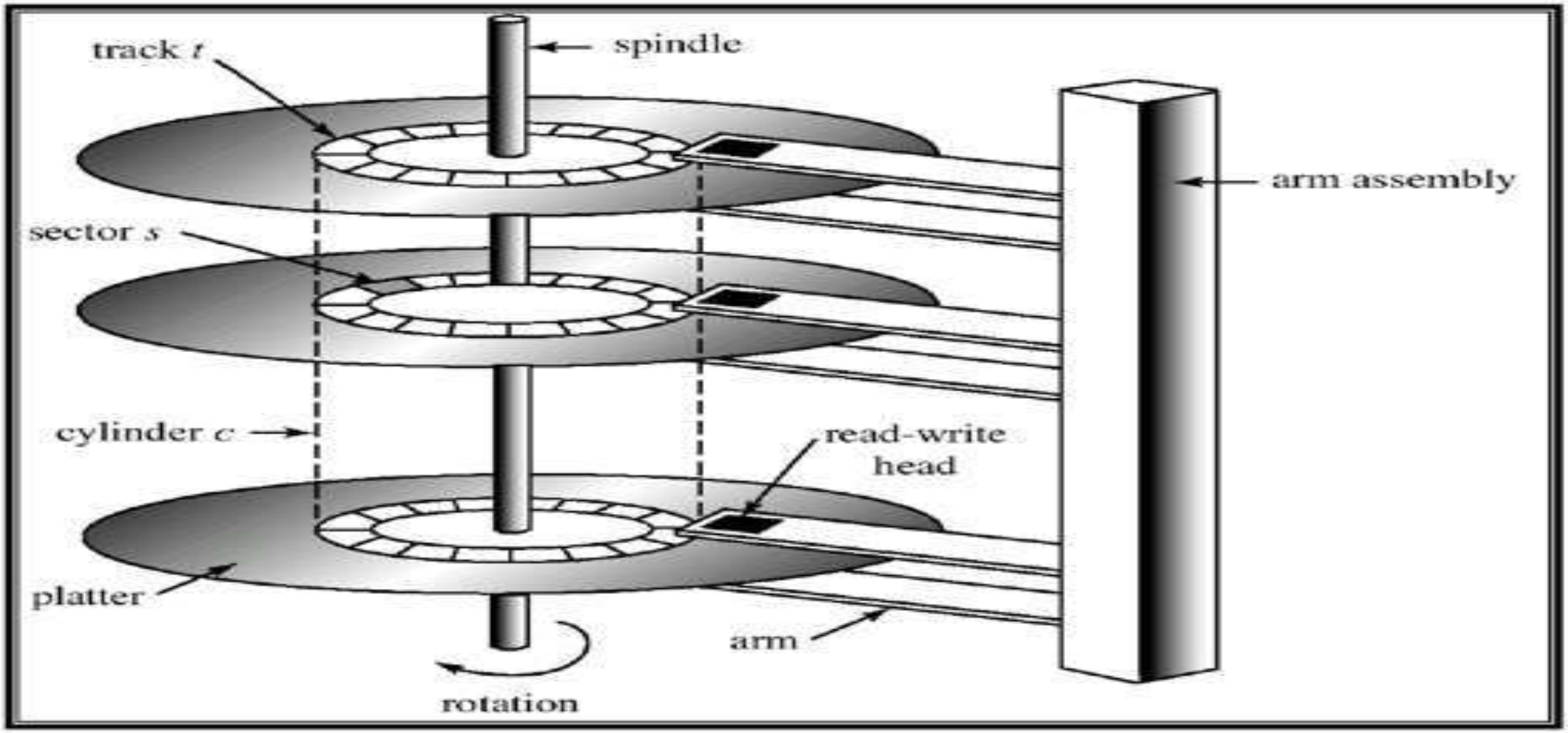
A SIMPLE DISK DRIVE



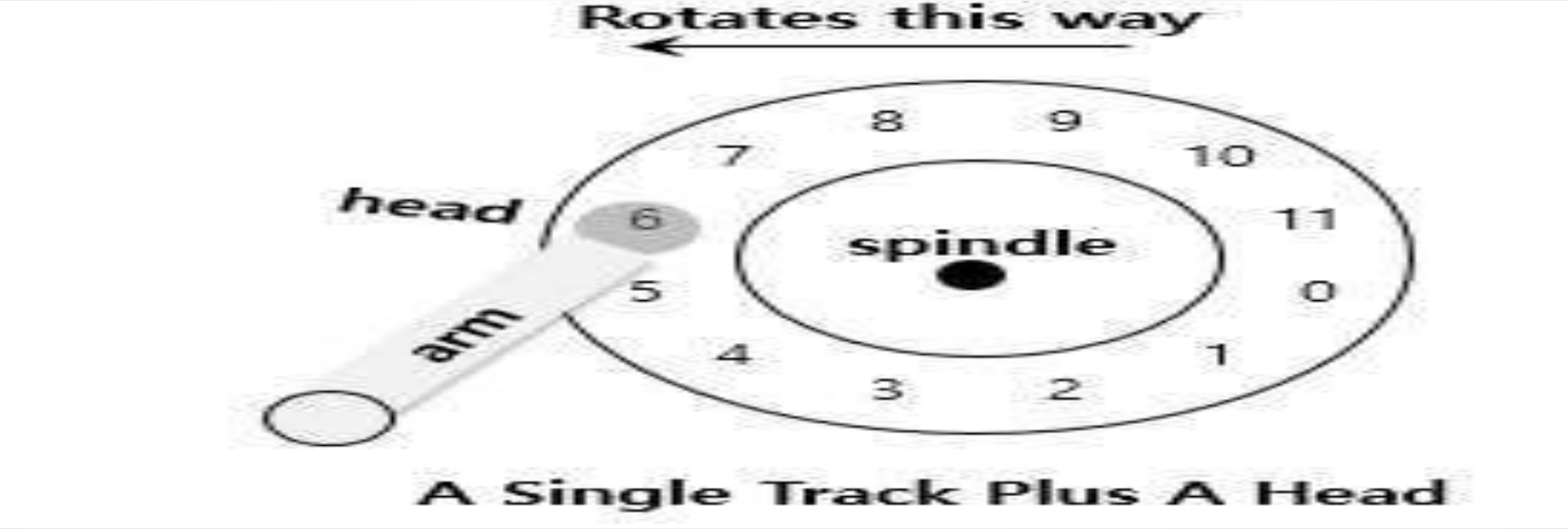
Disk head (One head per surface of the drive)

- The process of *reading* and *writing* is accomplished by the **disk head**.
- Attached to a single disk arm, which moves across the surface.

EXAMPLE OF A DISK



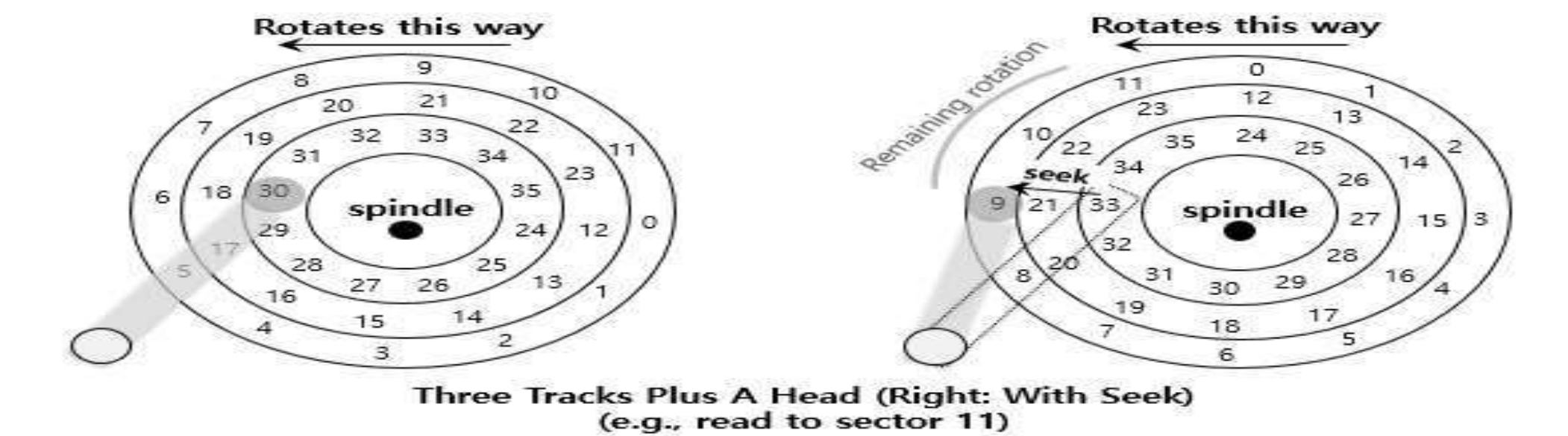
SINGLE-TRACK LATENCY: THE ROTATIONAL DELAY



Rotational delay: Time for the desired sector to rotate

- Ex) Full rotational delay is `R` and we start at `sector 6`
  - Read sector 0: Rotational delay = $\frac{R}{2}$
  - Read sector 5: Rotational delay = `R-1` (worst case.)

MULTIPLE TRACKS: SEEK TIME



Three Tracks Plus A Head (Right: With Seek)
(e.g., read to sector 11)

Seek: Move the disk arm to the correct track

Seek time: Time to move the head to the track containing the desired sector.

One of the most costly disk operations.

PHASES OF SEEK

- Acceleration → Coasting → Deceleration → Settling
  - **Acceleration**: The disk arm gets moving.
  - **Coasting**: The arm is moving at full speed.
  - **Deceleration**: The arm slows down.
  - **Settling**: The head is *carefully positioned* over the correct track.
    - The settling time is often quite significant, e.g., 0.5 to 2ms.

TRANSFER TIME

- The final phase of I/O
  - Data is either *read from* or *written* to the surface.

- Complete I/O time:
  - **Seek Time**
  - Waiting for the **rotational delay**
  - **Transfer Time**

CACHE (TRACK BUFFER)

- **Hold data** read from or written to the disk
  - Allow the driver to quickly respond to requests.
  - Small amount of memory (usually around 8 or 16 MB)

WRITE ON CACHE

- **Write back** (Immediate reporting)
  - Acknowledge a write has completed when it has **put the data in its memory**.
  - faster but dangerous

- **Write through**
  - Acknowledge a write has completed after the write has **actually been written to disk**.

I/O Time: Doing The Math

- I/O time ($T_{I/O}$): $T_{I/O} = T_{seek} + T_{rotation} + T_{transfer}$
- The rate of I/O ($R_{I/O}$): $R_{I/O} = \frac{Size_{Transfer}}{T_{I/O}}$

	Cheetah 15K.5	Barracuda
Capacity	300 GB	1 TB
RPM	15,000	7,200
Average Seek	4 ms	9 ms
Max Transfer	125 MB/s	105 MB/s
Platters	4	4
Cache	16 MB	16/32 MB
Connects Via	SCSI	SATA

© 2016 KL University – The contents of this presentation are an intellectual and copyrighted property of KL University. ALL RIGHTS RESERVED

KL ACCREDITED BY NAAC WITH A++ GRADE
nirf 2022 NATIONAL INSTITUTIONAL RANKING FRAMEWORK RANKED 27 AMONG ALL UNIVERSITIES
CATEGORY 1 UNIVERSITY BY MHRD, Govt. of India
43 YEARS OF EDUCATIONAL LEADERSHIP

I/O TIME EXAMPLE

- **Random workload**: Issue 4KB read to random locations on the disk
- **Sequential workload**: Read 100MB consecutively from the disk

		Cheetah 15K.5	Barracuda
T_{seek}		4 ms	9 ms
$T_{rotation}$		2 ms	4.2 ms
Random	$T_{transfer}$	30 microsecs	38 microsecs
	$T_{I/O}$	6 ms	13.2 ms
	$R_{I/O}$	0.66 MB/s	0.31 MB/s
Sequential	$T_{transfer}$	800 ms	950 ms
	$T_{I/O}$	806 ms	963.2 ms
	$R_{I/O}$	125 MB/s	105 MB/s

Disk Drive Performance: SCSI Versus SATA

PROBLEMS ON DISK

1. Consider a disk pack with the following specifications- 16 surfaces, 128 tracks per surface, 256 sectors per track, and 512 bytes per sector. What is the capacity of the disk pack?

Given-

- Number of surfaces = 16
- Number of tracks per surface = 128
- Number of sectors per track = 256
- Number of bytes per sector = 512 bytes
- Capacity of disk pack = Total number of surfaces x Number of tracks per surface x Number of sectors per track x Number of bytes per sector
- = 16 x 128 x 256 x 512 bytes
- = 2^{28} bytes
- = 256 MB

1.A hard disk has 63 sectors per track, 10 platters each with 2 recording surfaces and 1000 cylinders. The address of a sector is given as a triple (c, h, s), where c is the cylinder number, h is the surface number and s is the sector number. Thus, the 0th sector is addressed as (0, 0, 0), the 1st sector as (0, 0, 1), and so on The address <400,16,29> corresponds to sector number:

Solution:

The data in hard disk is arranged in the shown manner. The smallest division is sector. Sectors are then combined to make a track. Cylinder is formed by combining the tracks which lie on same dimension of platters.

Read write head access the disk. Head has to reach at a particular track and then wait for the rotation of the platte so that the required sector comes under it.

Here, each platter has two surfaces, which is the r/w head can access the platter from the two sides, upper and lower. So,<400,16,29> will represent 400 cylinders are passed(0-399) and thus,

for each cylinder 20 surfaces (10 platters *2 surface each)

and each cylinder has 63 sectors per surface.

Hence we have passed 0-399 = 400 * 20 * 63 sectors + In 400th cylinder we have passed 16 surfaces(0-15) each of which again contains 63 sectors per cylinder so 16 * 63 sectors. + Now on the 16th surface we are on 29th sector.

So, sector no = 400x20x63 + 16×63 + 29 = 505037.

DISK SCHEDULING

- **Disk scheduling** is done by operating systems to schedule I/O requests arriving for the disk. Disk scheduling is also known as I/O Scheduling.

Whenever a process needs I/O to or from the disk, it issues a system call to the operating system.

The Request Specifies several pieces of information :

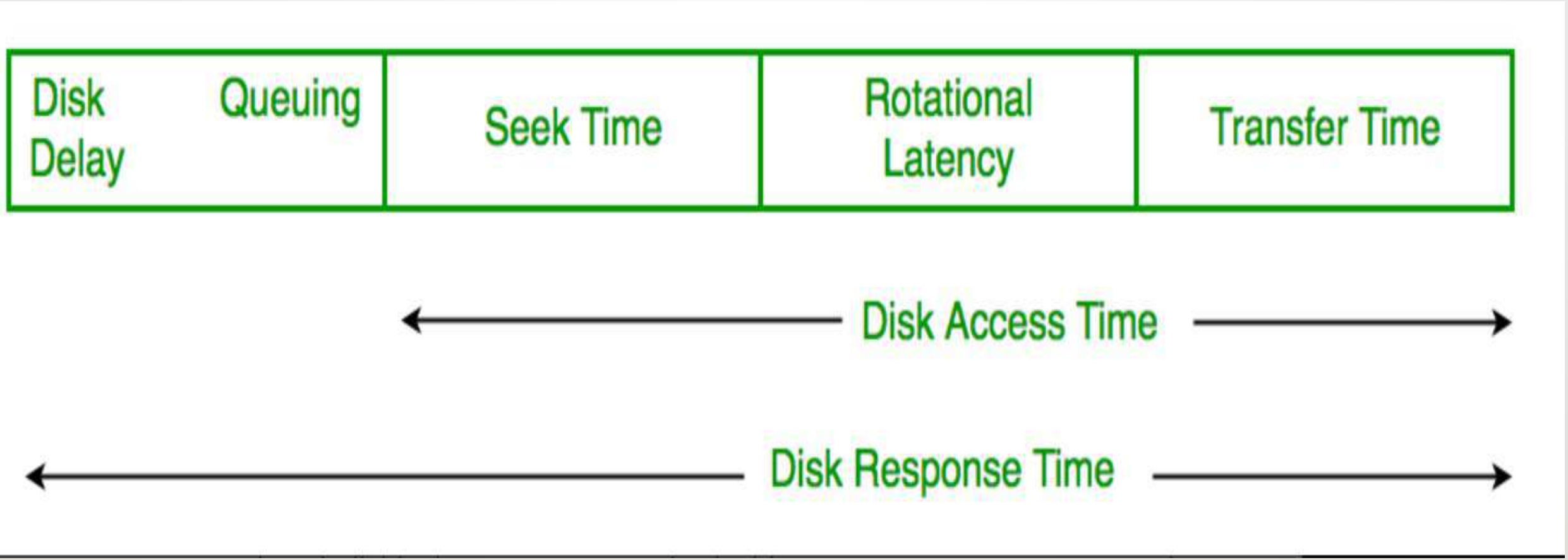
1. Whether This operation is in Input or Output.
2. What is the Disk Address for the Transfer?
3. What is the Memory Address for the Transfer?
4. What is the Number of Sectors to be Transferred?

If the desired disk drive and Controllers are available , The request can be serviced immediately.
Otherwise, any new Request for the service will be placed in the queue of pending requests for that drive.

KEY TERMS ASSOCIATED WITH DISK SCHEDULING

- **Seek Time:** Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or written. So the disk scheduling algorithm that gives a minimum average seek time is better.
- **Rotational Latency:** Rotational Latency is the time taken by the desired sector of the disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.
- **Transfer Time:** Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and the number of bytes to be transferred.
- **Disk Access Time:** Disk Access Time = Seek Time + Rotational Latency + Transfer Time

Total Seek Time = Total head Movement * Seek Time


Disk Delay
Queuing
Seek Time
Rotational Latency
Transfer Time
Disk Access Time
Disk Response Time

DISK SCHEDULING ALGORITHMS

- FCFS (First Come First Serve)
- SSTF (Shortest Seek Time First)
- SCAN (Elevator Algorithm)
- C-SCAN (Circular SCAN)
- LOOK
- C-LOOK

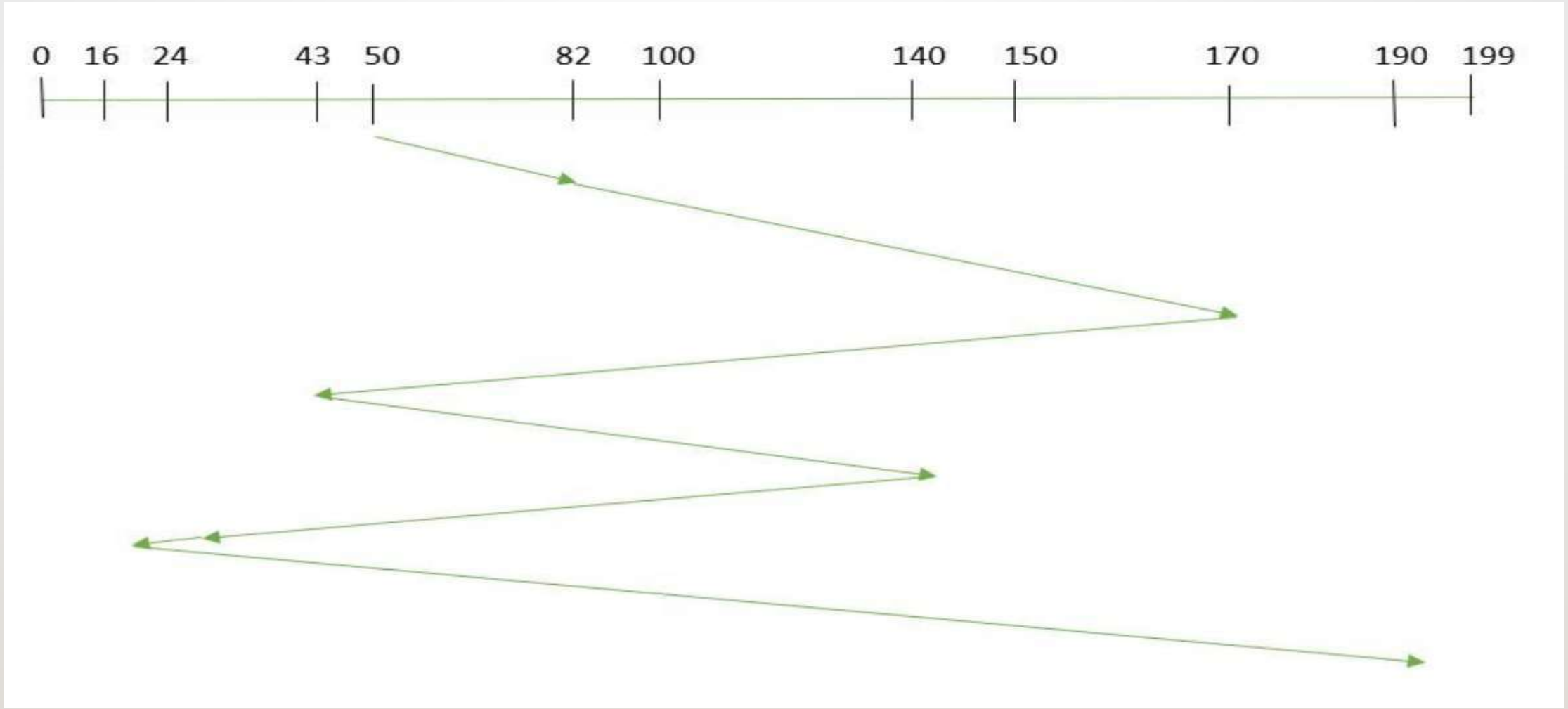
FCFS (FIRST COME FIRST SERVE)

FCFS is the simplest of all Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue. Let us understand this with the help of an example.

Example:

Suppose the order of request is- (82,170,43,140,24,16,190)
And current position of Read/Write head is: 50
So, Total overhead movement (Total distance covered by the disk arm)
= (82-50)+(170-82)+(170-43)+(140-43)+(140-24)+(24-16)+(190-16) =642

FCFS (First Come First Serve)



FCFS

Advantages

- Here are some of the advantages of First Come First Serve.
- Every request gets a fair chance
- No indefinite post ponement.

Disadvantages

- Here are some of the disadvantages of First Come First Serve.
- Does not try to optimize seek time
- May not provide the best possible service

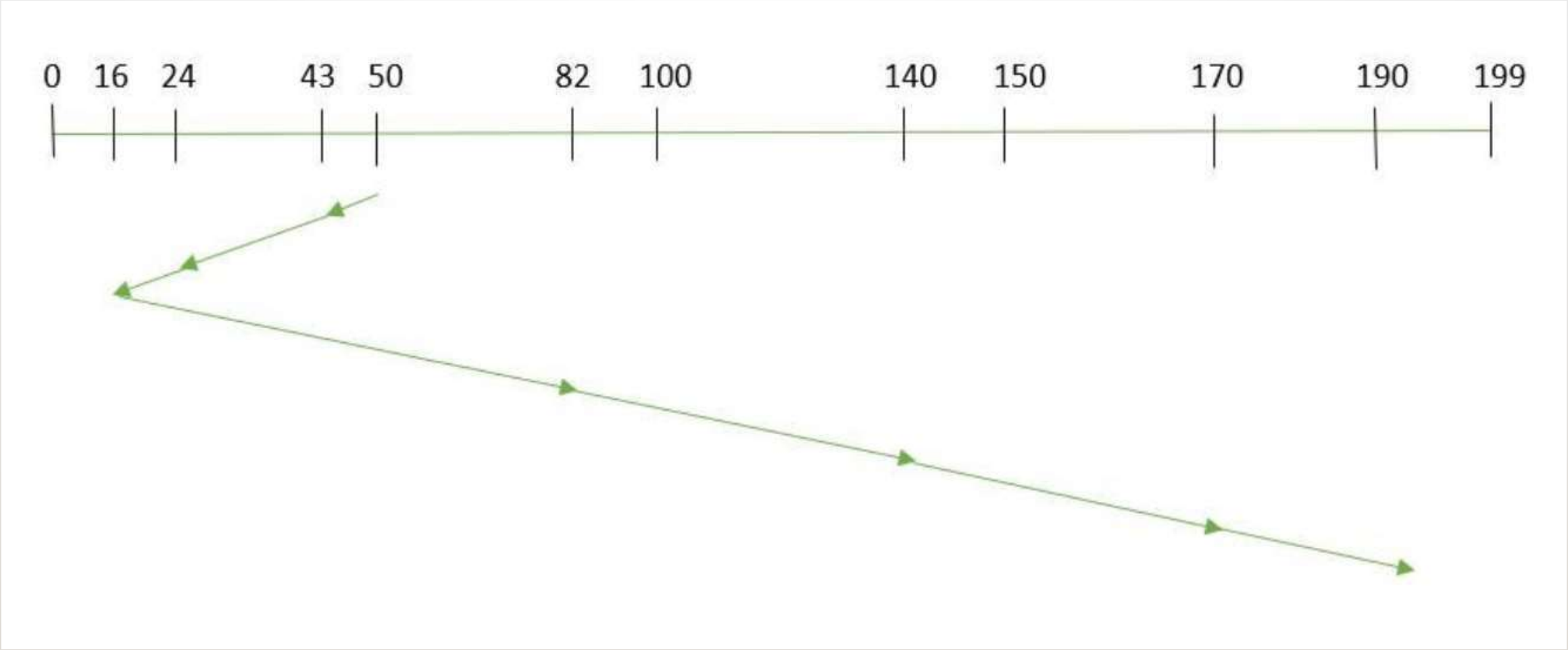
SSTF (SHORTEST SEEK TIME FIRST)

In SSTF (Shortest Seek Time First), requests having the shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time.

Example:

Suppose the order of request is- (82,170,43,140,24,16,190) And the current position of the Read/Write head is: 50 So,
Total overhead movement (total distance covered by the disk arm)
= (50-43)+(43-24)+(24-16)+(82-16)+(140-82)+(170-140)+(190-170) =208

SSTF (SHORTEST SEEK TIME FIRST)



SSTF

Advantages Here are some of the advantages of Shortest Seek Time First.

- The average Response Time decreases
- Throughput increases

Disadvantages

- Here are some of the disadvantages of Shortest Seek Time First.
- Overhead to calculate seek time in advance
- Can cause Starvation for a request if it has a higher seek time as compared to incoming requests
- The high variance of response time as SSTF favors only some requests

SCAN

In the SCAN algorithm the disk arm moves in a particular direction and services the requests coming in its path and after reaching the end of the disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works as an elevator and is hence also known as an **elevator algorithm.**

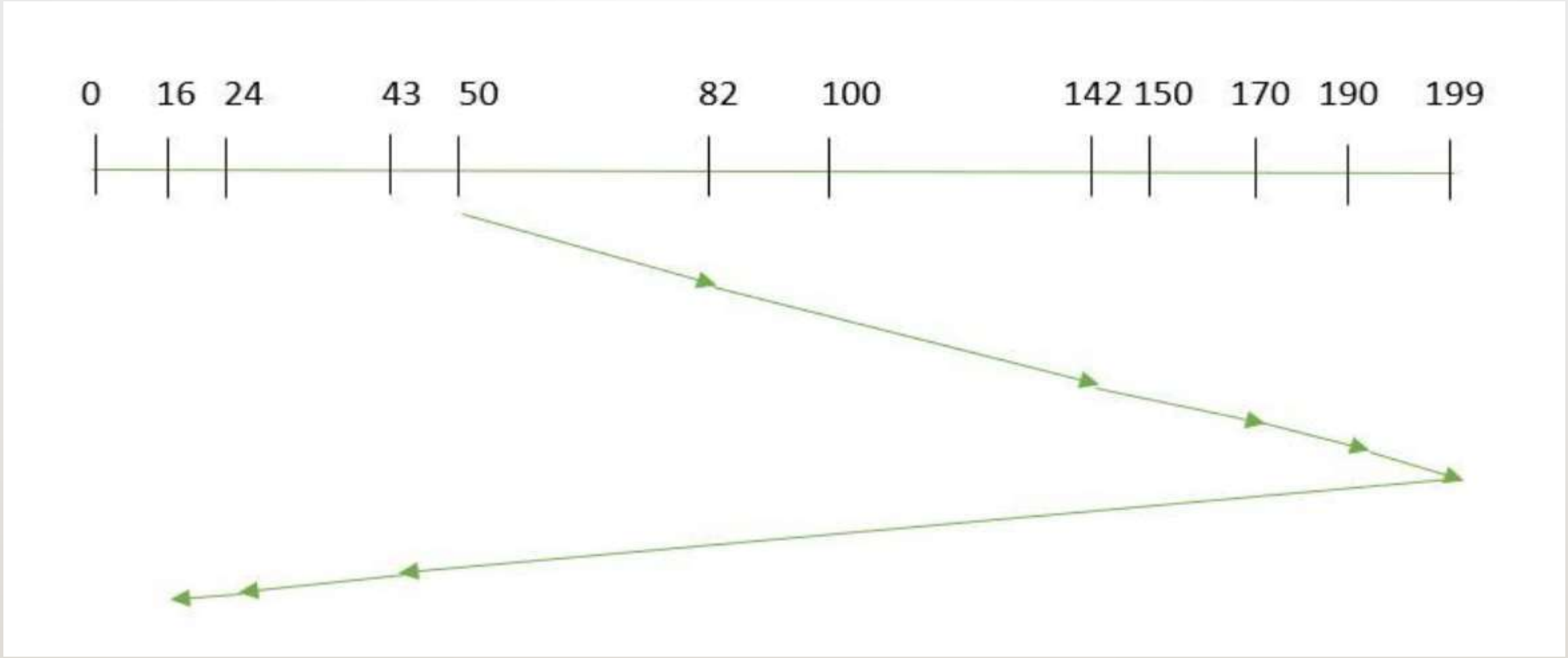
Example:

Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value".**

Therefore, the total overhead movement (total distance covered by the disk arm) is calculated as

= (199-50) + (199-16) = 332

SCAN



SCAN

Advantages

- Here are some of the advantages of the SCAN Algorithm.
- High throughput
- Low variance of response time
- Average response time

Disadvantages

- Here are some of the disadvantages of the SCAN Algorithm.
- Long waiting time for requests for locations just visited by disk arm

C-SCAN

In the SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction. So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.

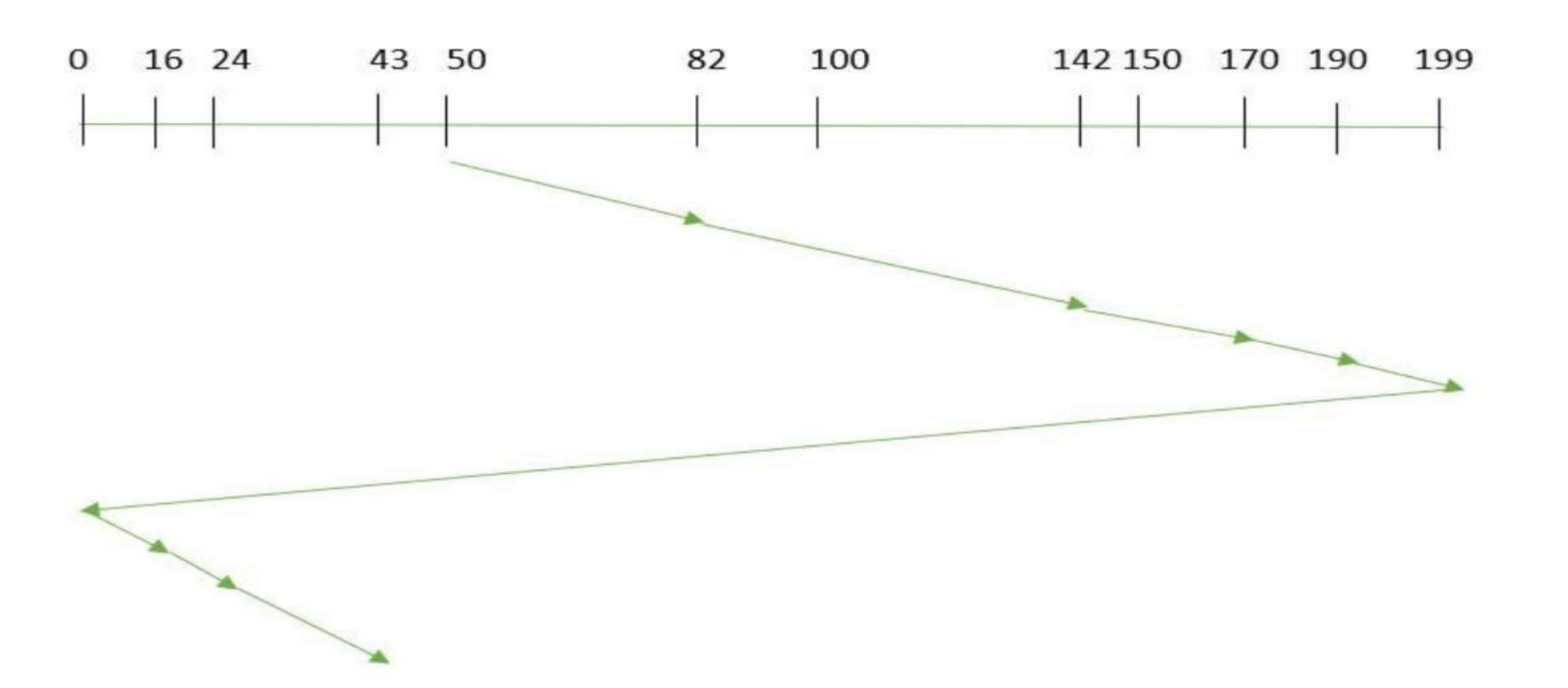
Example:

Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value".**

So, the total overhead movement (total distance covered by the disk arm) is calculated as:

=(199-50) + (199-0) + (43-0) = 391

C-SCAN



C-SCAN

Advantages:

- **Uniform wait times**: C-SCAN provides more uniform waiting times compared to SCAN, as the arm movement is unidirectional, ensuring all requests in one direction are served in one sweep.
- **Fairness**: Each request gets serviced without being bypassed or postponed for long due to the circular nature, ensuring fairness for all requests.
- **Reduced arm movement**: By moving the disk arm in one direction and jumping back to the start after reaching the end, it prevents back-and-forth movements, improving performance.

Disadvantages:

- **Longer wait times for some requests**: Some requests, especially those closer to the start of the disk, may experience longer wait times, as the disk arm must complete a full cycle before returning to serve them.
- **Idle time during reverse**: The jump back to the beginning of the disk can create idle time where no requests are served.

LOOK

LOOK Algorithm is similar to the SCAN disk scheduling algorithm except for the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only. Thus it prevents the extra delay that occurs due to unnecessary traversal to the end of the disk.

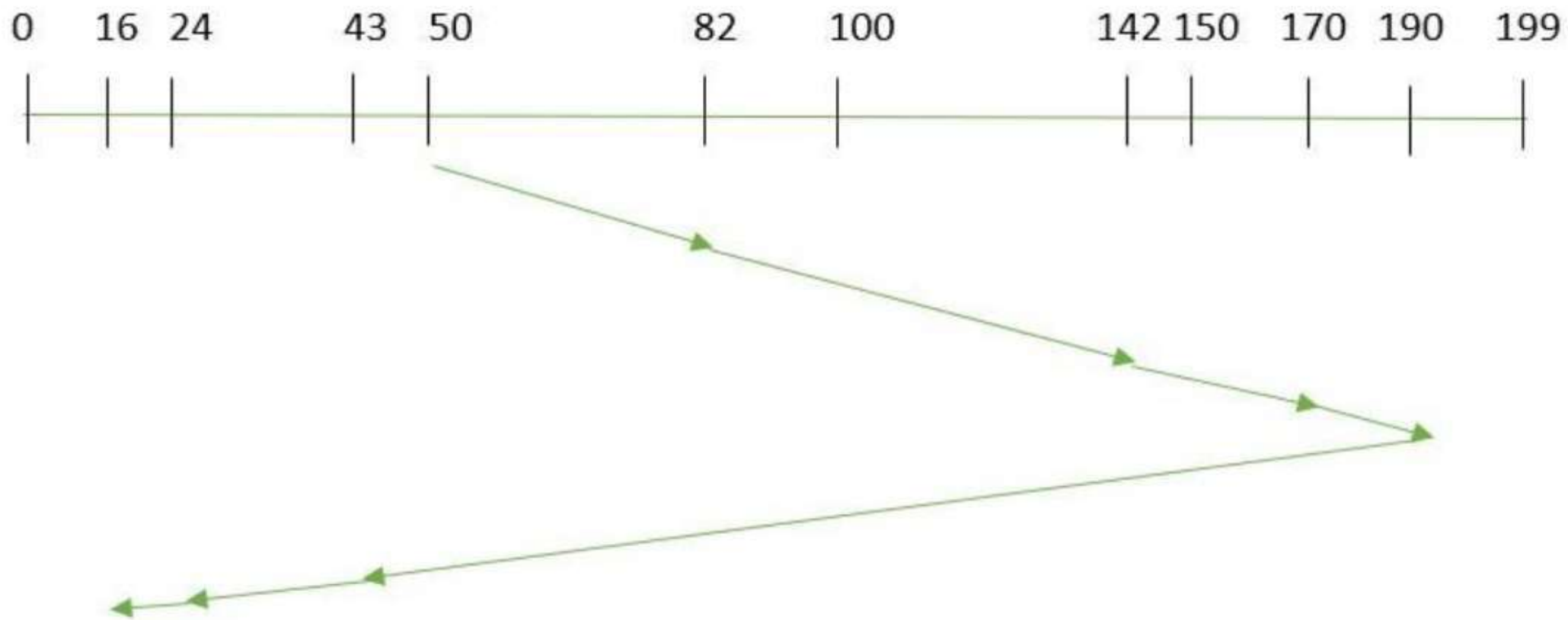
Example:

Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value".**

So, the total overhead movement (total distance covered by the disk arm) is calculated as:

= (190-50) + (190-16) = 314

LOOK



LOOK

Advantages:

- **Optimized movement**: Unlike SCAN, LOOK stops at the last request in each direction, avoiding unnecessary traversal to the physical end of the disk.
- **Better performance**: Because the arm does not always have to go to the far end of the disk, it can save time and improve overall performance.
- **Reduced average seek time**: LOOK often provides a lower average seek time because it minimizes unnecessary arm movements.

Disadvantages:

- **More variability in wait times**: Depending on where requests are located, some requests may experience significantly longer wait times than others.
- **Complexity**: Slightly more complex to implement compared to simpler algorithms like FCFS or SCAN.

C-LOOK

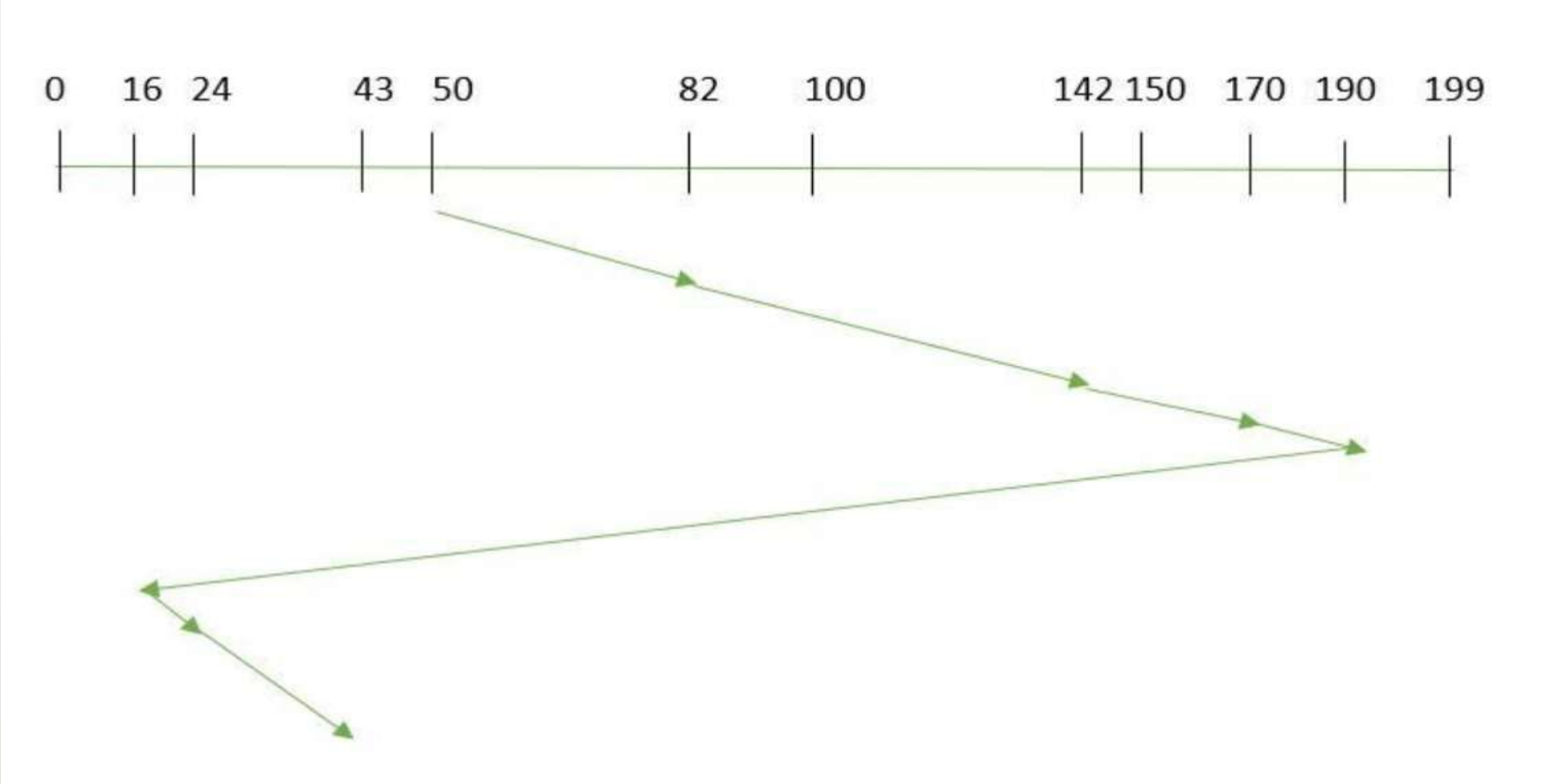
As LOOK is similar to the SCAN algorithm, in a similar way, C-LOOK is similar to the CSCAN disk scheduling algorithm. In CLOOK, the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end's last request.

Example:

Suppose the requests to be addressed are-82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move **"towards the larger value"**

So, the total overhead movement (total distance covered by the disk arm) is calculated as
= (190-50) + (190-16) + (43-16) = 341

C-LOOK



C-LOOK

Advantages:

- **Efficient movement**: Like C-SCAN, C-LOOK avoids back-and-forth movement, but it further reduces unnecessary arm movement by stopping at the furthest request.
- **Better response time**: By skipping the unnecessary end traversal, it can provide faster response times for disk I/O compared to C-SCAN.
- **Fairness**: Similar to C-SCAN, it ensures that all requests are served in a circular manner, providing fairness.

Disadvantages:

- **Long wait times for some requests**: Similar to C-SCAN, requests near the starting point may have to wait longer if they are not in the current direction of the disk arm.
- **Jump inefficiency**: The algorithm requires a jump back to the start after reaching the farthest request, which could still lead to idle times and inefficiencies in certain situations.

Input:

- **Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}**
- **Initial head position = 50**
- **Find Total Head Movements by Using FCFS, SSTF, SCAN, CSCAN, LOOK, CLOOK Algorithms.**

THANK YOU

Team - Operating System