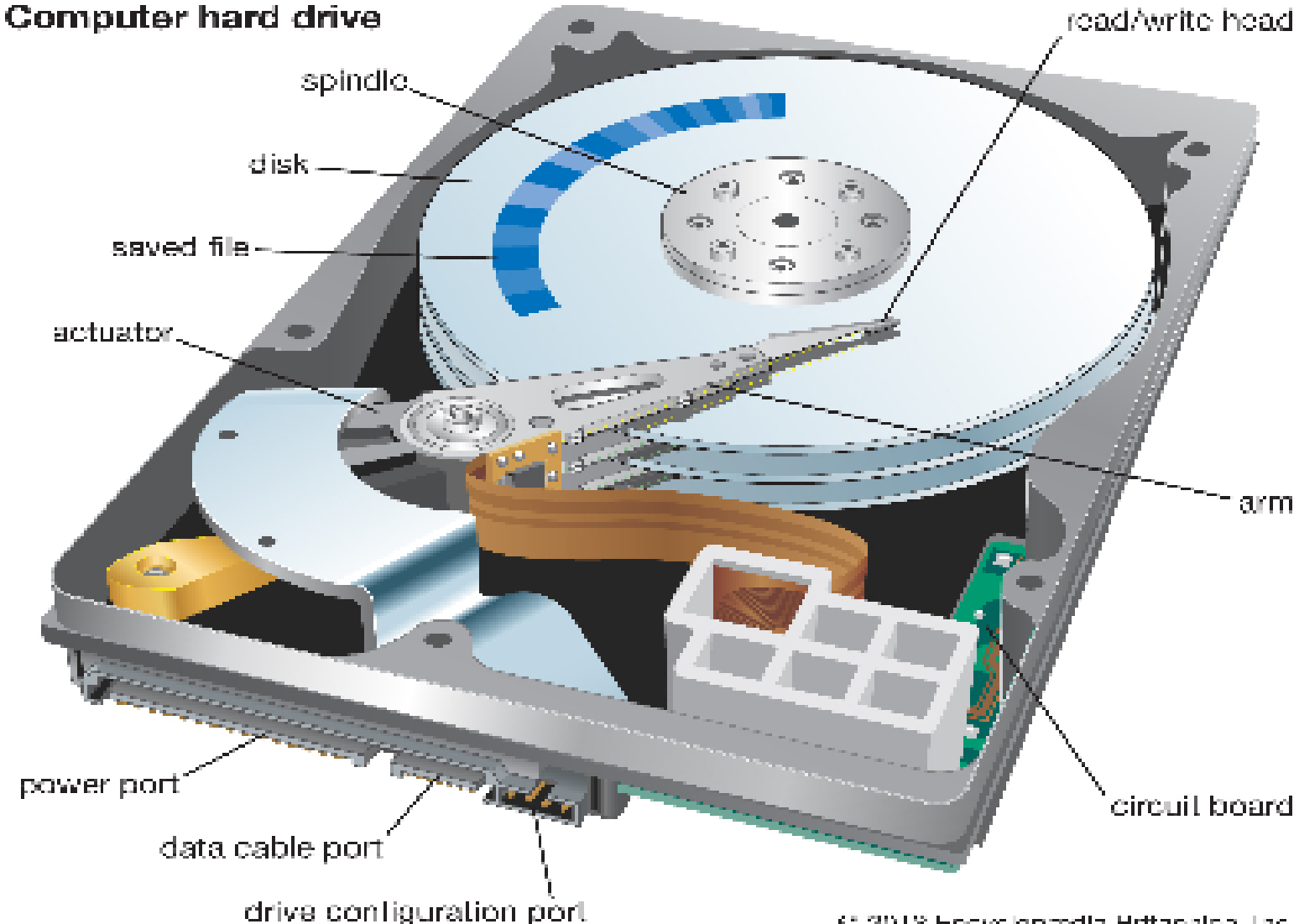
OPERATING SYSTEMS

Disk Scheduling

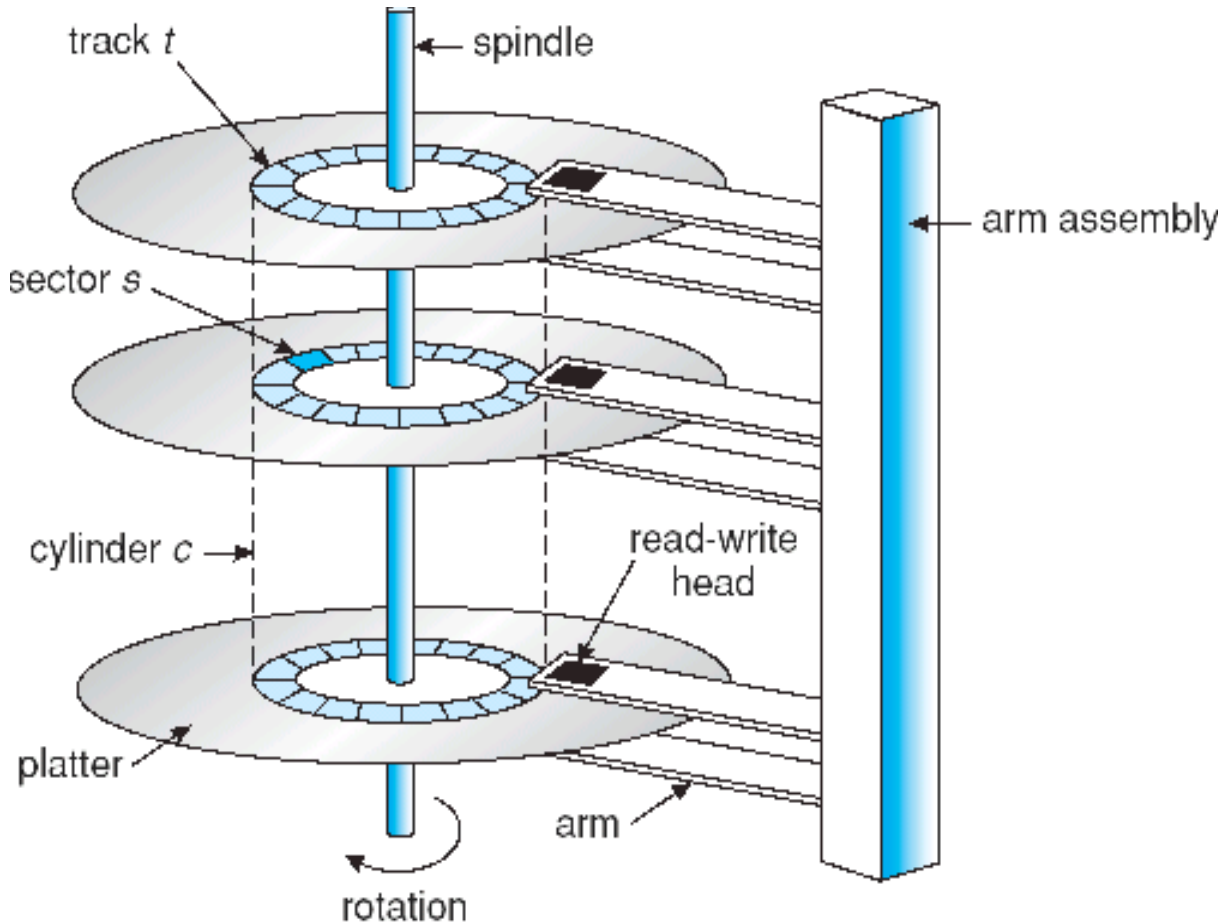
Disk Scheduling

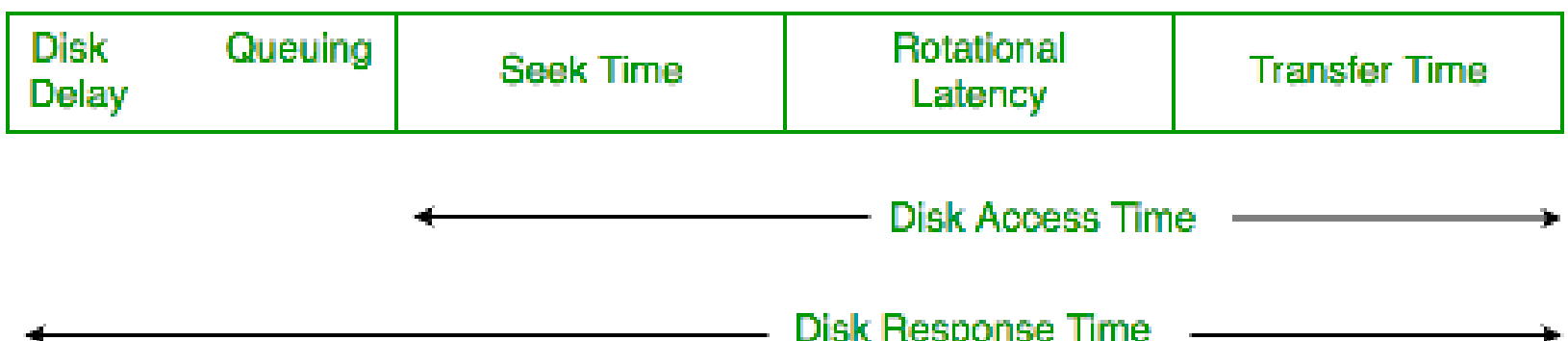
Magneti c Disks



- Each platter (Disk) has a flat circular shape like a CD or DVD.
- Generally diameter range from 1.8 to 5.25 inches.
- Both surface are covered by a magnetic material.
- R/W head is attached to disk arm.
- Surface of disk are divided into circular tracks.
- Tracks are further divided into sectors.
- Seek time: Time required to move the R/W head on the desired track.
- Disk Scheduling: In case of multiple I/O request, disk scheduling algorithm must decide which request must be executed first.

Disk Structure




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

Seek Time

- Seek time is the time taken in locating the disk arm to a specified track where the read/write request will be satisfied.

Rotational Latency

- It is the time taken by the desired sector to rotate itself to the position from where it can access the R/W heads.

Transfer Time

- It is the time taken to transfer the data.

Disk Access Time

- Disk access time is given as,
- Disk Access Time = Rotational Latency + Seek Time + Transfer Time

Disk Response Time

- It is the average time to spent by each request waiting for the IO operation.

Disk Scheduling

- As we know, a process needs two type of time, CPU time and IO time. For I/O, it requests the Operating system to access the disk.
- However, the operating system must be fare enough to satisfy each request and at the same time, operating system must maintain the efficiency and speed of process execution.
- The technique that operating system uses to determine the request which is to be satisfied next is called disk scheduling.

Purpose of Disk Scheduling

- The main purpose of disk scheduling algorithm is to select a disk request from the queue of IO requests and decide the schedule when this request will be processed.

Disk Scheduling Algorithms

- FCFS scheduling algorithm
- SSTF (shortest seek time first) algorithm
- SCAN scheduling
- C-SCAN scheduling
- LOOK Scheduling
- C-LOOK scheduling

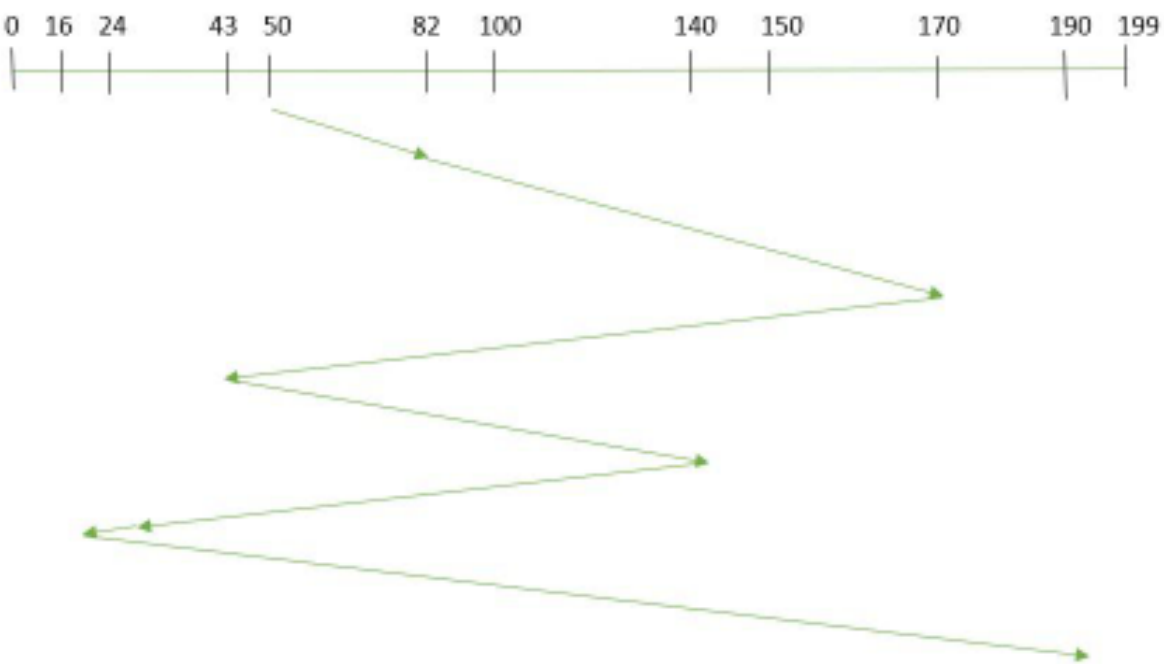
Disk Scheduling Algorithms

- We illustrate them with a I/O request queue (cylinders are between 0-199):

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

Suppose the order of request is- (82,170,43,140,24,16,190)

And current position of Read/Write head is : 50



So, total seek time:

=(82-50)+(170-82)+(170-43)+(140-43)+(

140-24)+(24-16)+(190-16)

=642

❑Advantages:

- Every request gets a fair chance
- No indefinite postponement

❑Disadvantages:

- Does not try to optimize seek time
- May not provide the best possible service

First Come First Serve (FCFS) Example

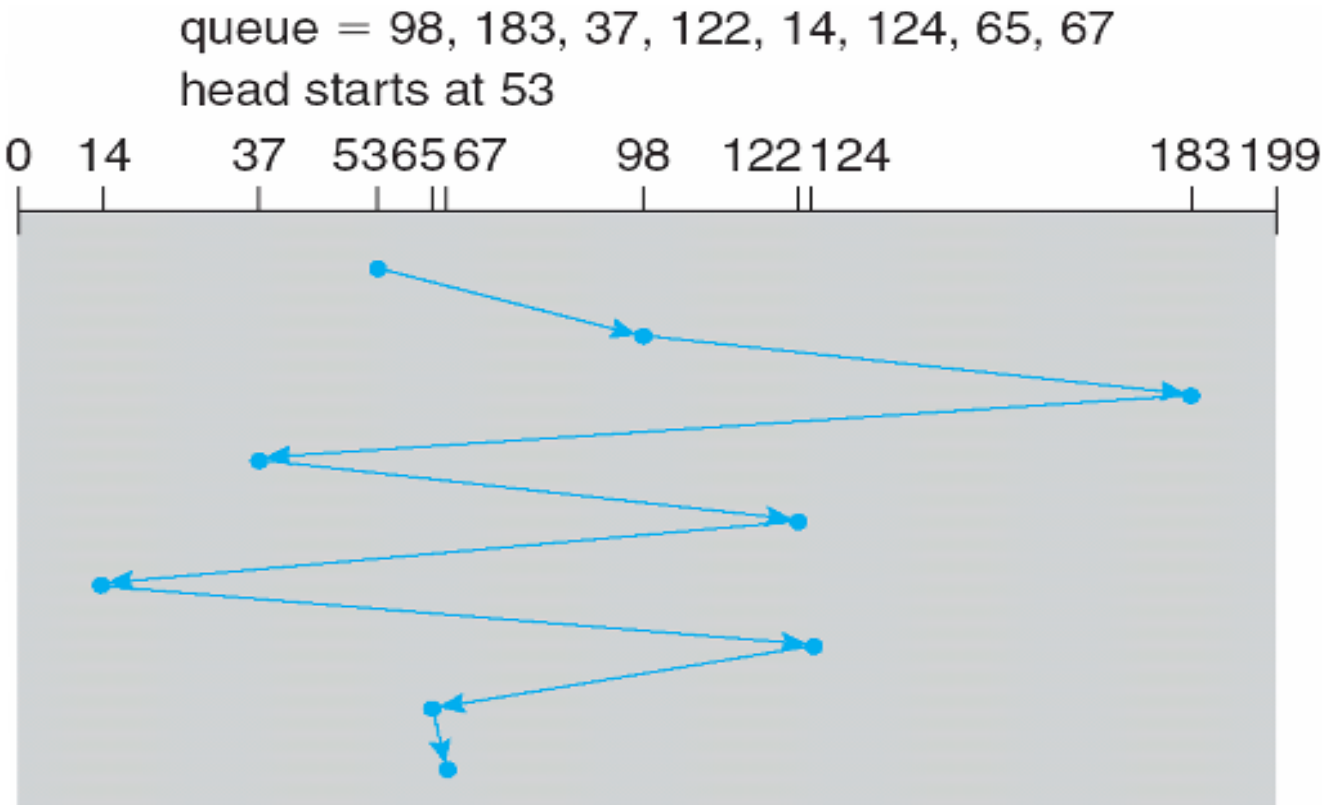
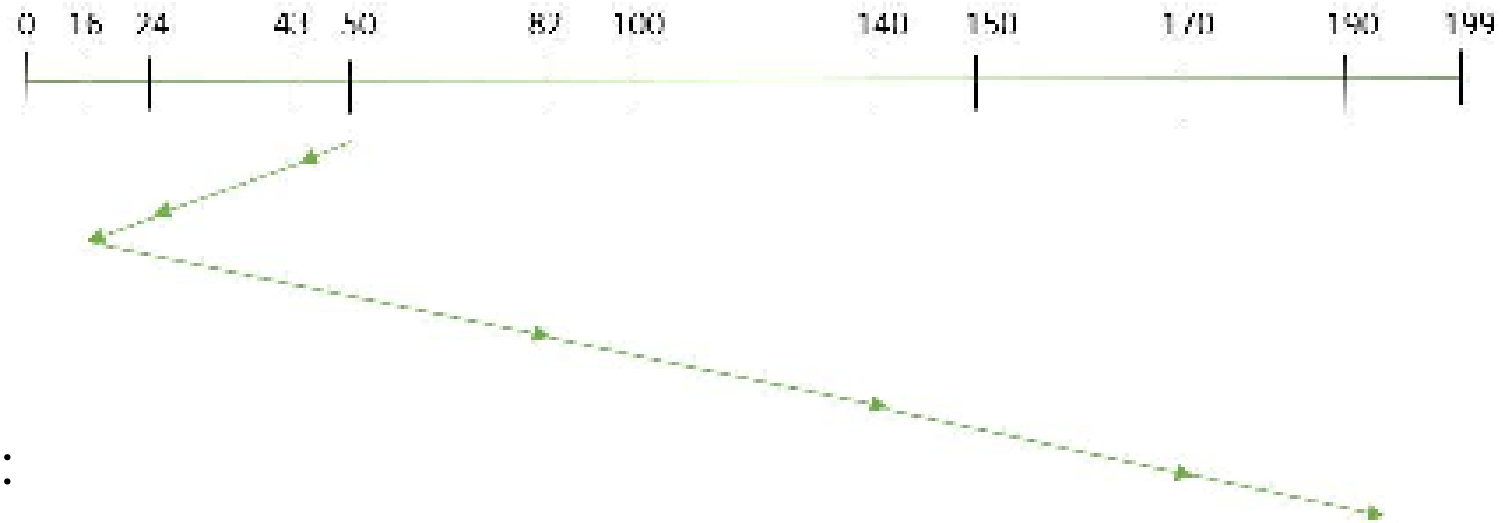


Illustration shows total head movement of 640 cylinders.

SSTF (Shortest Seek Time First)

Suppose the order of request is- (82,170,43,140,24,16,190)

And current position of Read/Write head is : 50



So, total seek time:

=(50-43)+(43-24)+(24-16)+(82-16)+(140-82)

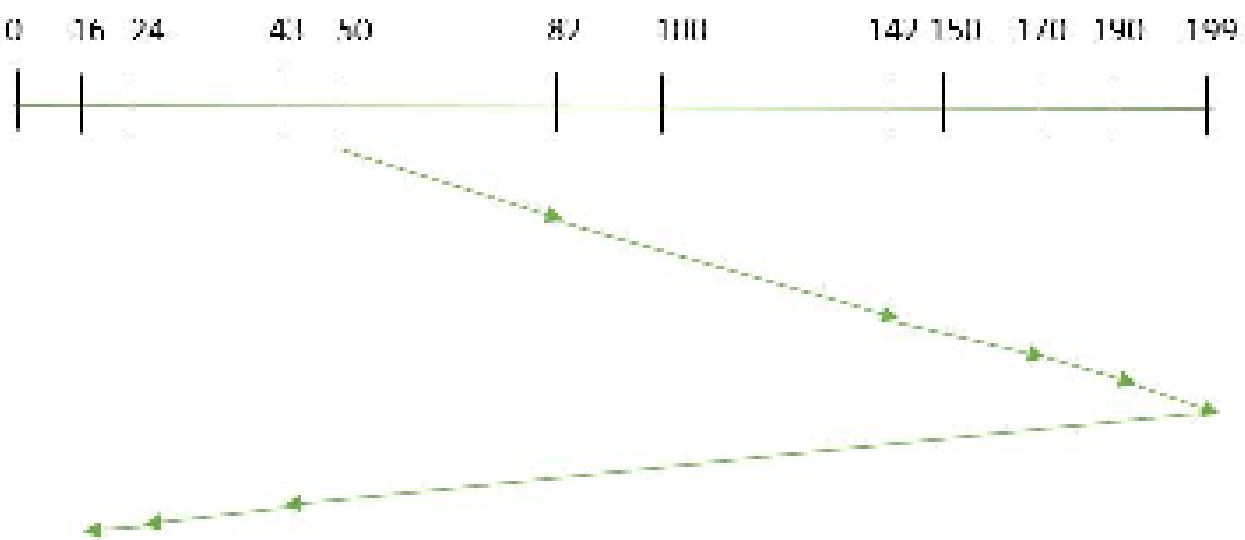
+(170-40)+(190-170)

=208

SCAN

- SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path.
- This algorithm works as an elevator and hence also known as **elevator algorithm.**
- As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.

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”.**



seek time is calculated as:

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

=332

Advantages:

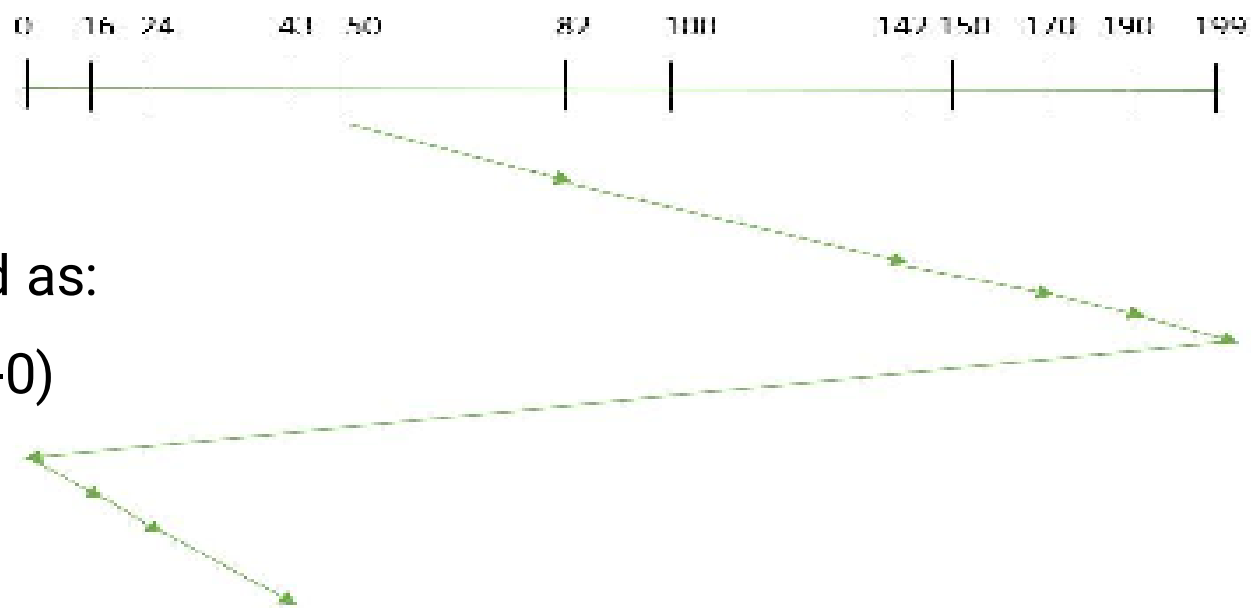
•High throughput

•Low variance of response time

•Average response time

CSCAN

- 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”.**



Seek time is calculated as:

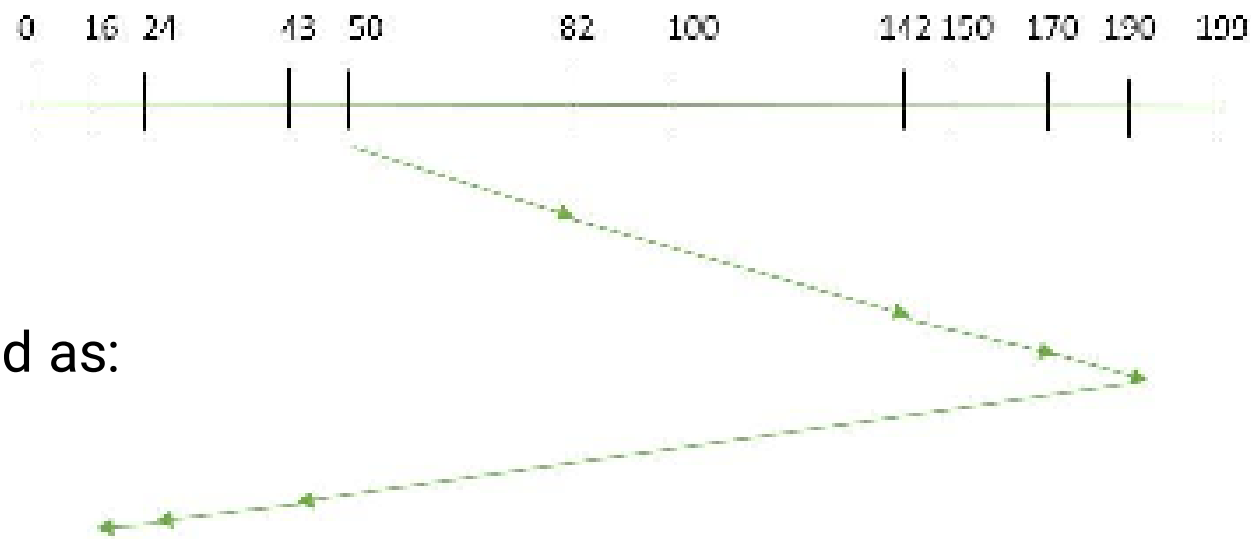
=(199-50)+(199-0)+(43-0)

=391

LOOK:

- It 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 which occurred due to unnecessary traversal to the end of the disk.

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”.**

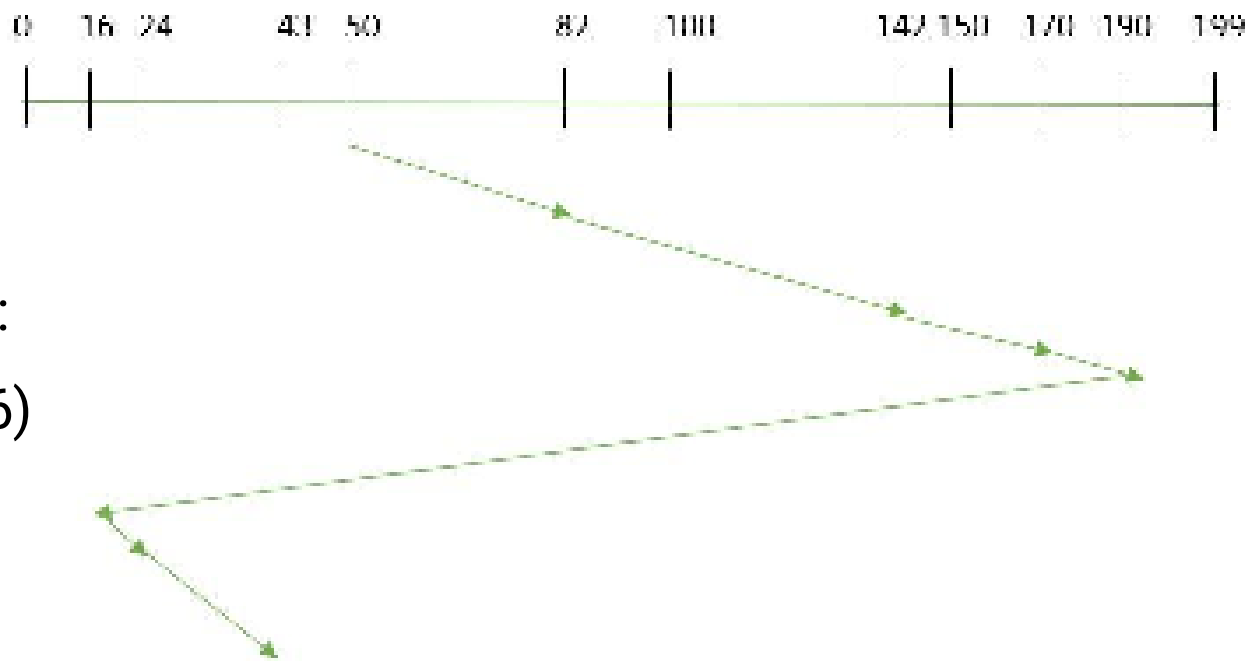


seek time is calculated as:

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

=314

CLOOK



seek time is calculated as:

=(190-50)+(190-16)+(43-16)

=341