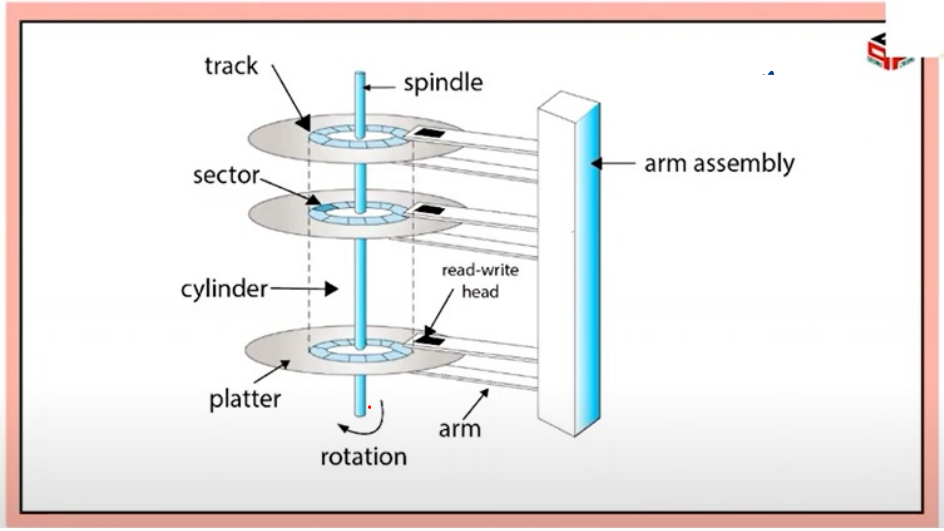
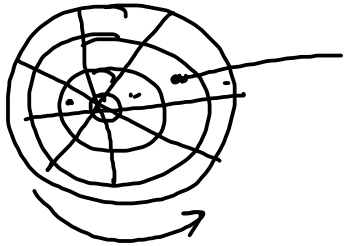
Operating Systems

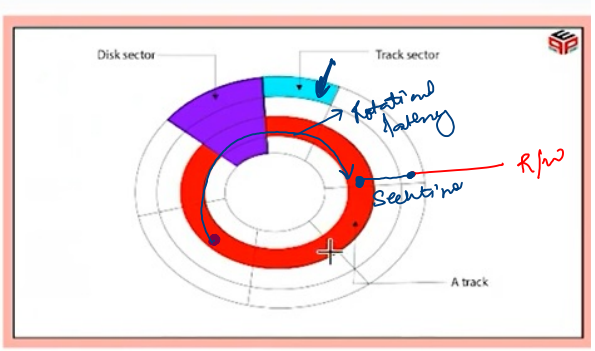
Disk Scheduling

1. Characteristics of Disk Scheduling
2. Goals of Disk Scheduling
3. Disk Scheduling Algo.
   a) FCFS
   b) SSTF
   (c) SCAN
   (d) C-SCAN
   (e) LOOK
   (f) CLOOK

Disk architecture







Characteristics

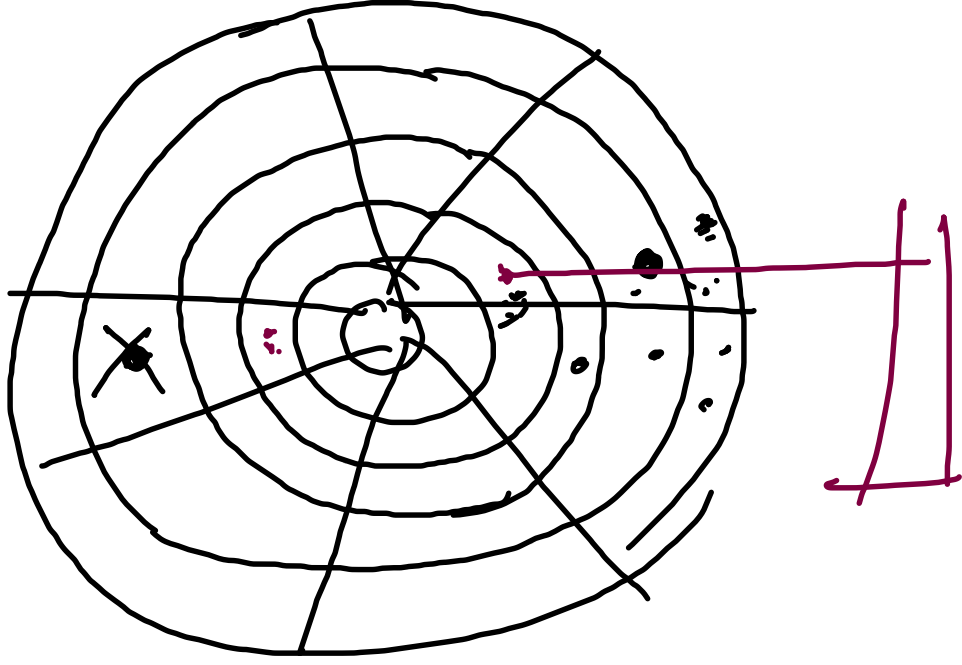
1. **Seek time :-** amount of time taken to move read/write (r/w) head from its current position to desired track.

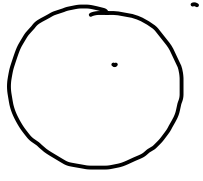
2. **Rotational latency -** Time taken to rotate the track to reach the desired sector.

3. **Transfer time :-** Amount of time taken to transfer the data.

→ depends upon rotating speed of disk & no. of bytes to be transferred

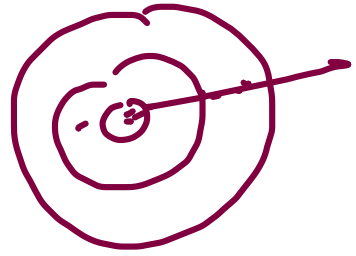
Total disk access time = Seek time + Rotational latency + Data transfer time





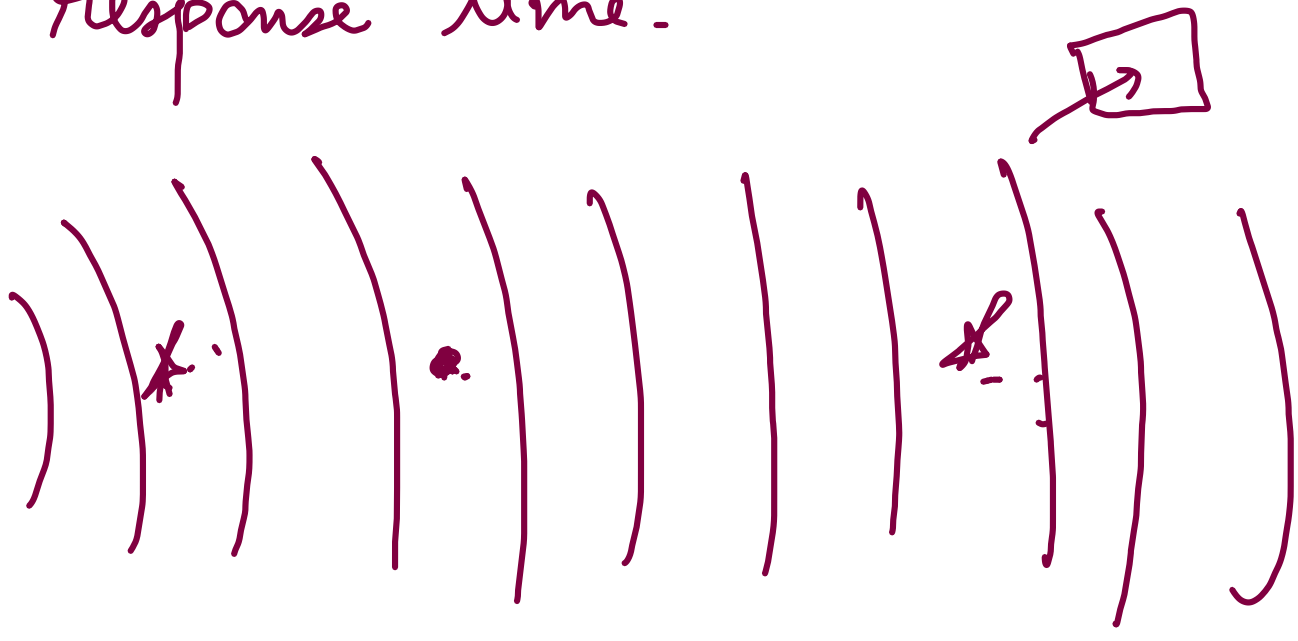
Goals:-

1) Reduce seek time



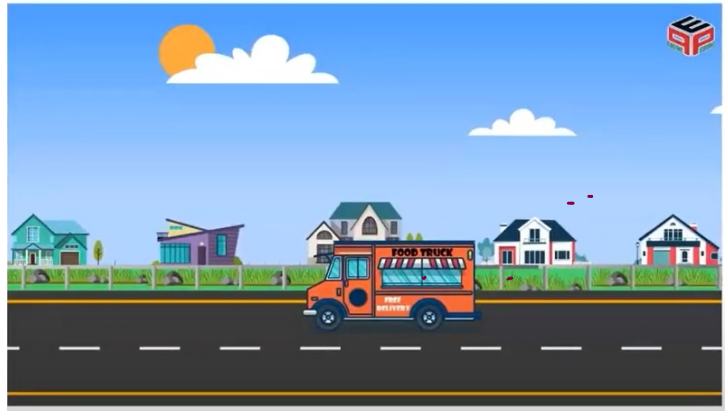
Disk sector

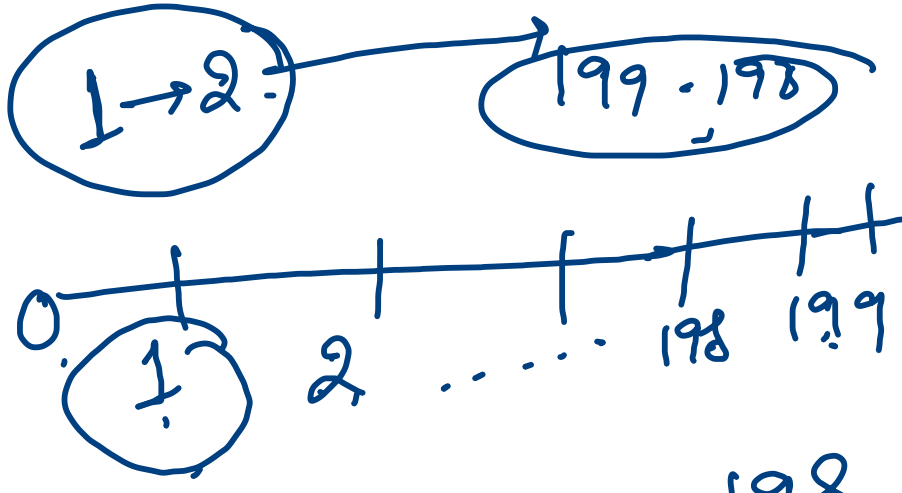
2) Reduce response time.



FCFS - (First come first serve)

Servicing I/O requests in the order in which they arrive.





Adv :- 1) Fair chance ✓
2) No starvation ✓

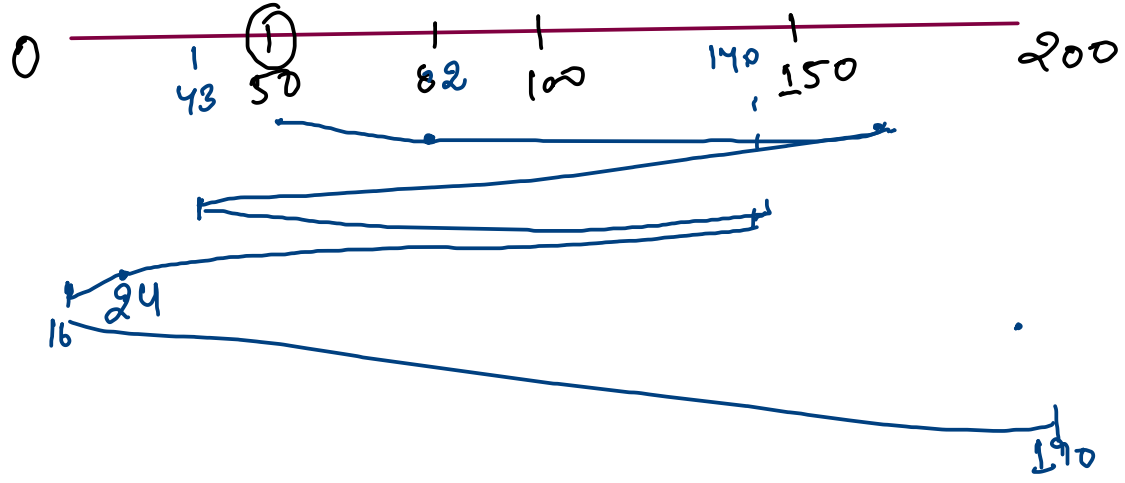
Dis :- 1) Optimization of seek time.
2) Best service

1 → 199 - 2 - 198

1 → 2 - (1sec)

Request queue 82, 170, 43, 140, 24, 16, 190

Initial R/w head = 50



Seek time X

Seek time

50

(82-50) + (170-82)

+ (170-43) +

(140-43) + ...

SSTF - Shortest Seek time first

→ move the head to the closest track in the service queue.

→ select the requests with the min. seek time

Adv :- 1) Optimize seek time ⇒

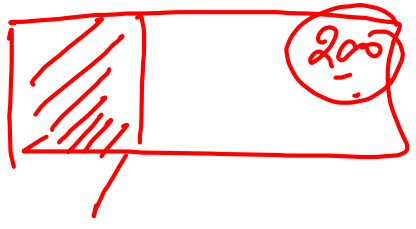
2) Red. response time ⇒

Dis :- Starvation

Shortest seek ti⇒

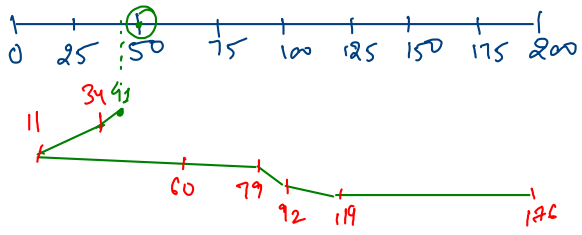
1, 200, 10, 5, 7

5, 6, 7, 10, 11



ready queue = (176, 79, 34, 60, 92, 11, 41, 114)

current pos of R/w head = 50



1) Seek time ↓

2) Response time X

Total Seek time

⇒

(50-41) + (41-34)

+ (34-11) +

(60-11) + (79-60)

+ (92-79) +

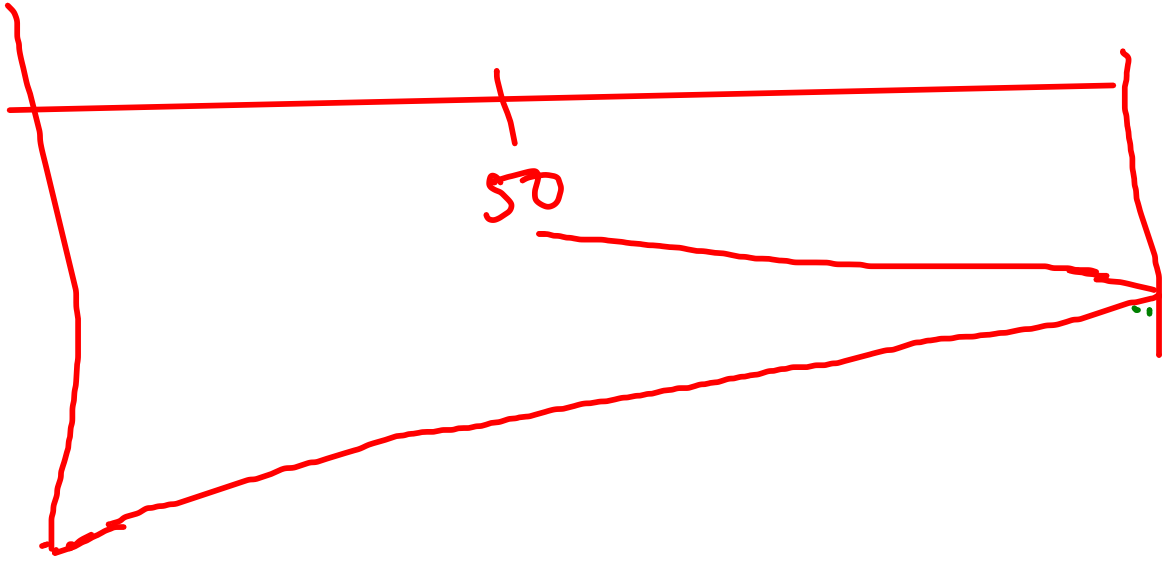
(119-92) + (176-119)

SCAN (elevators algo)

→ disk arm moves into a particular direction & services the requests coming in its path & after reaching the end of the disks, it reverses its direction & again services the requests arriving in its path.

Adv:- 1) Avg. response time ↓
2) Seek time ↓

Dis- Starvation (in some cases.)



Ready queue ⇒ 98, 183, 37, 122, 14, 124, 65, 67

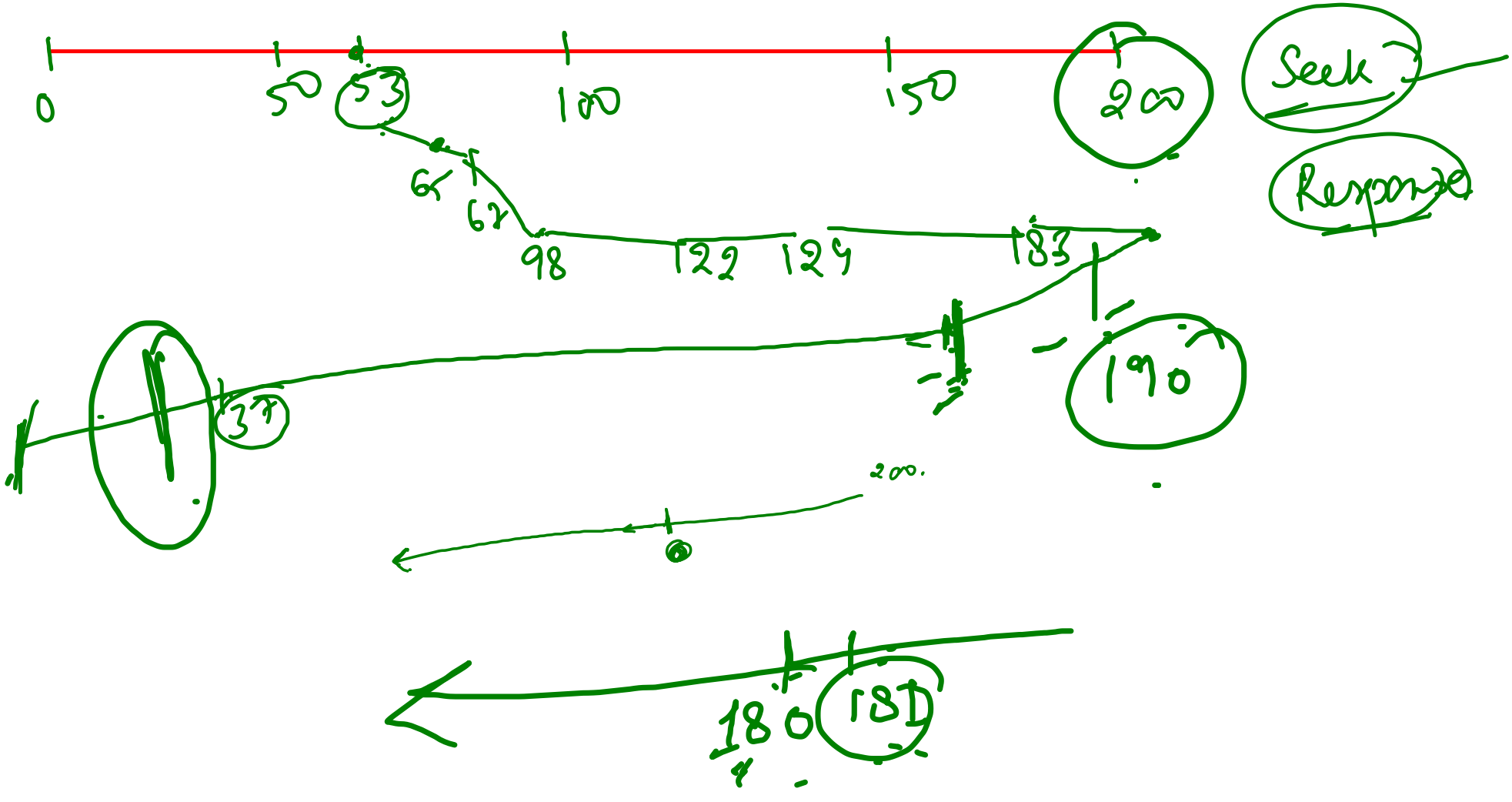
R/W pointer is at 53.

150

Scan

→ Right

Scan



Seek

Response

Ready queue ⇒ 98, 183, 37, 122, 14, 124, 65, 67

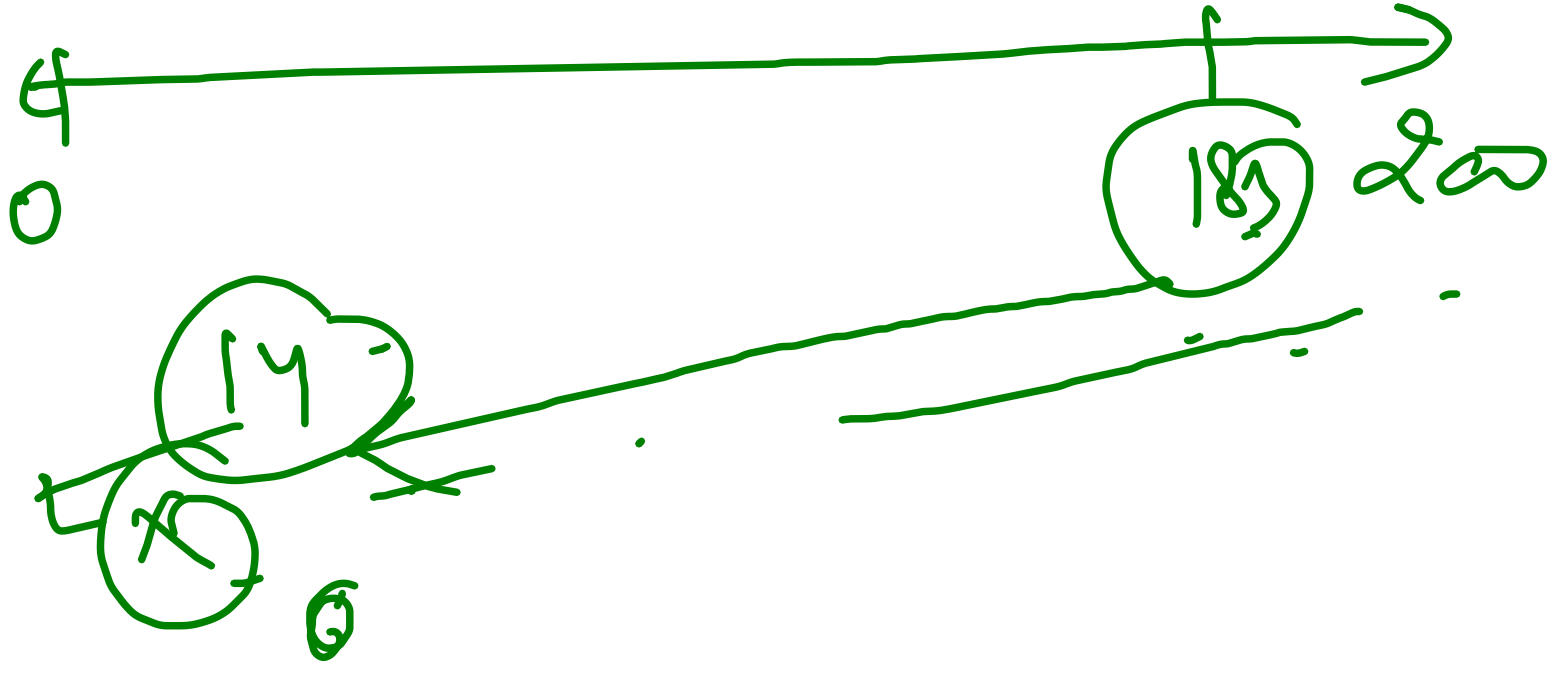
R/W pointer is at 53.

↪ Right

Look

C-Look

C-Scan

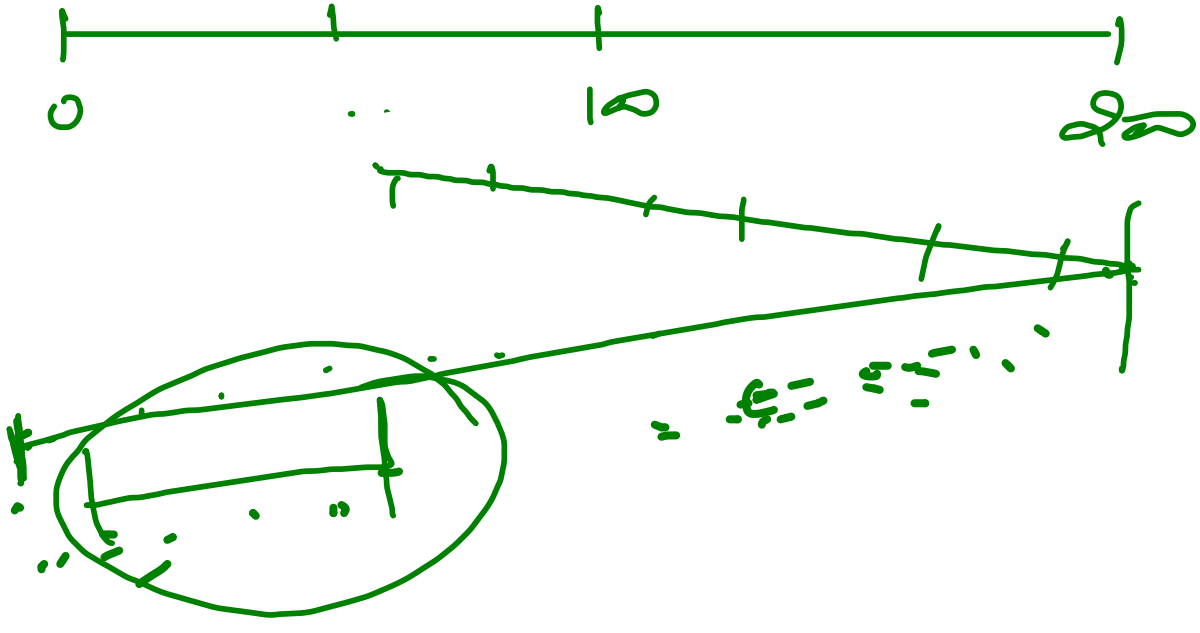


C-Scan

C-Scan

- → to avoid too much waiting time, because it might be possible too much requests are waiting at the other end.
- → Uniform wait time
- → heads seeks to the beginning w/o servicing any I/O

Starvation



Look

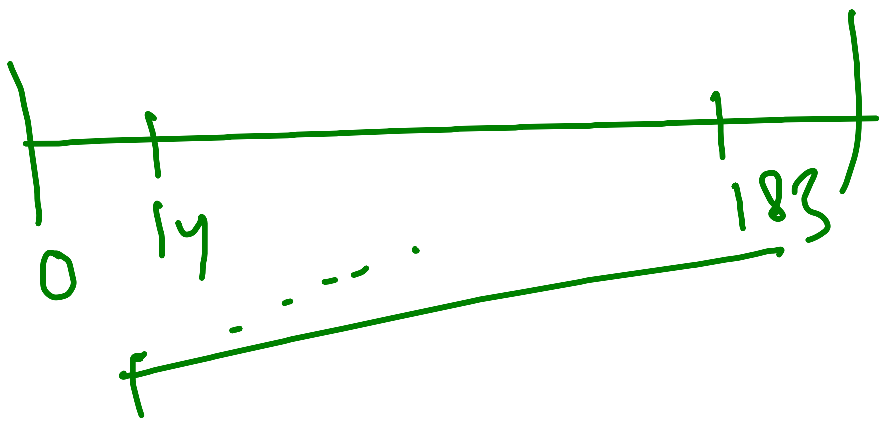
→ common-sense improvement over SCAN.

→ move the head as far as the last request in that direction.

→ prevents extra delay.

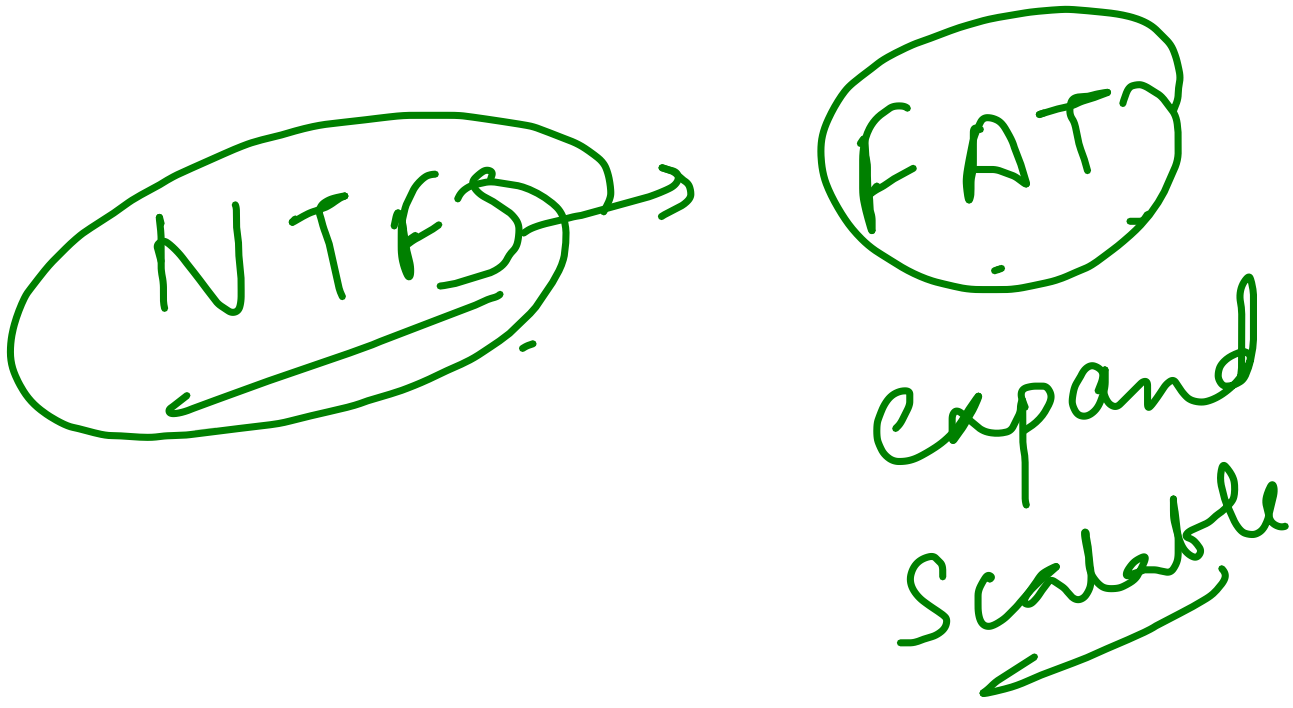
C- Look

Same as C-Scan EXCEPT instead of seeking to the starting of disk we seek to the lowest track with scheduled I/O.



0 20 30

Strategy	Advantages	Disadvantages
FCFS	• Easy to implement • Sufficient for light loads	• Doesn't provide best average service • Doesn't maximize throughput
SSTF	• Throughput better than FCFS • Tends to minimize arm movement	• May cause starvation of some requests • Localizes under heavy loads
SCAN/LOOK	• Eliminates starvation • Throughput similar to SSTF • Works well with light to moderate loads	• Needs directional bit • More complex algorithm to implement • Increased overhead
C-SCAN/ C-LOOK	• Works well with moderate to heavy loads • No directional bit • Small variance in service time • C-LOOK doesn't travel to unused tracks	• May not be fair to recent requests for high-numbered tracks • More complex algorithm than N-Step SCAN, causing more overhead


NTFS
FAT
expand
Scalable