Course plan

OPERATING SYSTEM

Disk scheduling

Objectives:

- To select a disk request from the queue of IO requests and decide the schedule when this request will be processed.
- To reduce the seek time.
- To Increase the throughput: the average number of requests satisfied per time unit.
- To Maximize response time: the average time that a request must wait before it is satisfied.
- To use the hardware efficiently using fast access time and large disk bandwidth that depends on the relative positions of the read-write head and the requested data.

Outcomes:

- The disk scheduling algorithm will improve the performance of disk i/o by reducing average seek time compared to the existing disk scheduling algorithm.

Prerequisites:

- Magnetic Disk Structure
- One single unit of disk is called platter
- Each platter is divided into circular shaped tracks.
- Each track is further divided into sectors.
- The speed of the disk is measured with the help of transfer rate and random access time.
- Seek time
- Rotational latency

Plan for the lecturer delivery

1. **Teaching aid both Blackboard and Presentation Via LCD**
2. **Topic should be start with Definition, as follows**

There are many Disk Scheduling Algorithms but before discussing them let's have a quick look at some of the important terms:

- 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 write. So the disk scheduling algorithm that gives minimum average seek time is better.

- Rotational Latency: Rotational Latency is the time taken by the desired sector of 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 number of bytes to be transferred.
- Disk Access Time: Disk Access Time is:

```
Disk Access Time = Seek Time +
                   Rotational Latency +
                   Transfer Time
```

Disk Scheduling Algorithms:

- **FCFS**: FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.
- **SSTF**: In SSTF (Shortest Seek Time First), requests having 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.
- **SCAN**: In 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.
- **CSCAN**: In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction.
- **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.
- **CLOOK**: As LOOK is similar to SCAN algorithm, in similar way, CLOOK is similar to 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.

3. **Practical Reality of the topic can be explained with the following:**

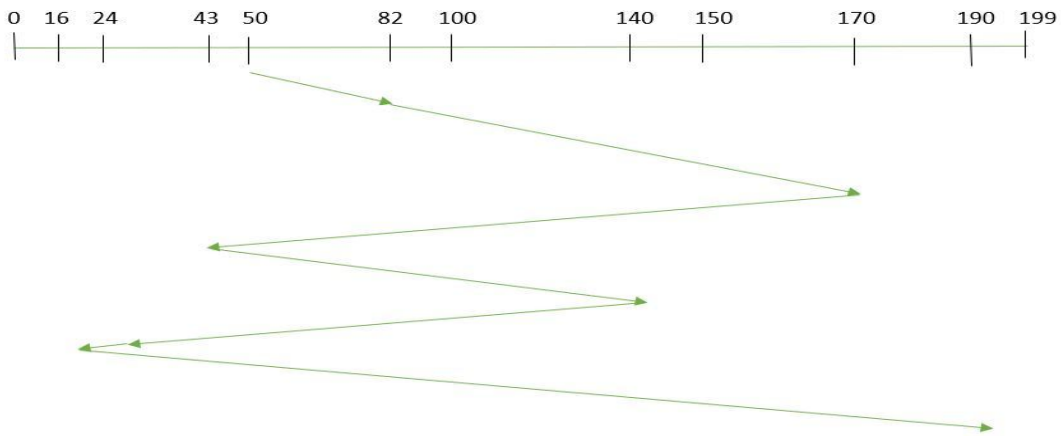
FCFS Disk Scheduling algorithm:

FCFS is the simplest of all the 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 seek time:

=(82-50)+(170-82)+(170-43)+(140-43)+(140-24)+(24-16)+(190-16)

=642

5.COMPARISON OF VARIOUS DISK SCHEDULING ALGORITHMS

FCFS Scheduling: It is a fair scheduling policy. It prevents starvation and gives low overhead. This is acceptable when the load on a disk is light. It has extremely low throughput due to lengthy seeks.

SSTF Scheduling: SSTF Scheduling has higher throughput and lower response time than FCFS Policy. It is reasonable solution for batch processing system. Sometimes, it does not ensure fairness because with this scheduling starvation is possible. This policy is generally not acceptable for interactive systems. It leads to higher variances of response times.

SCAN Scheduling: It offers an improved variance of response time. The drawback of this scheduling policy is that it does not change the direction until edge of disk is reached. Starvation is still possible in this scheduling. Under a light load, SCAN policy is best.

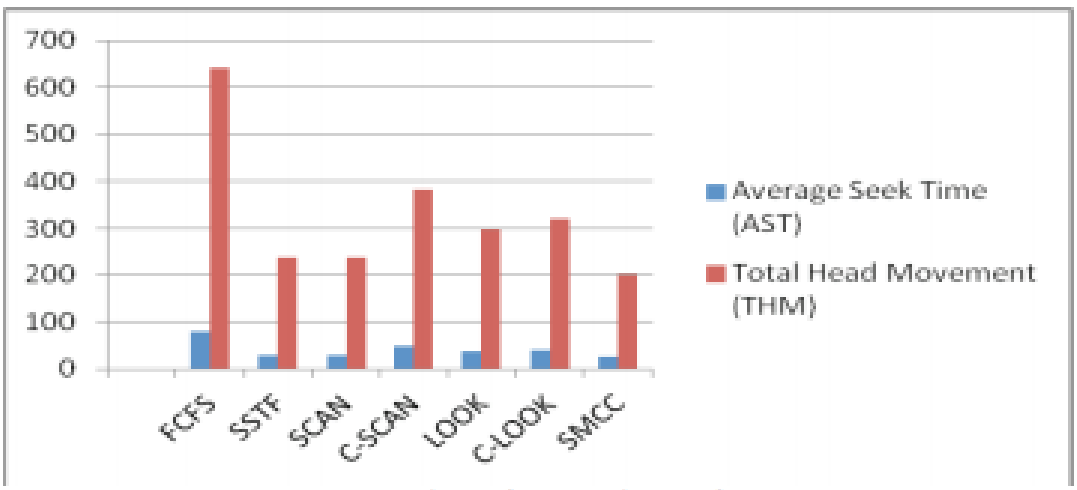
C-SCAN Scheduling: It maintains high level of throughput while further limiting variance of response times by avoiding discrimination against the innermost and outermost cylinders.

LOOK Scheduling: The main advantage of this scheduling is that it only performs sweeps large enough to service all requests. It improves efficiency by avoiding unnecessary seek operation. It gives high throughput.

C-LOOK Scheduling: It gives lower variance of response time than LOOK, at the expense of throughput.

Comparison of Disk Scheduling Algorithms

(a) FIFO (starting at track 100)		(b) SSTF (starting at track 100)		(c) SCAN (starting at track 100, in the direction of increasing track number)		(d) C-SCAN (starting at track 100, in the direction of increasing track number)	
Next track accessed	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**
55	45	90	10	150	50	150	50
58	3	58	32	160	10	160	10
39	19	55	3	184	24	184	24
18	21	39	16	90	94	18	166
90	72	38	1	58	32	38	20
160	70	18	20	55	3	39	1
150	10	150	132	39	16	55	16
38	112	160	10	38	1	58	3
184	146	184	24	18	20	90	32
Average seek length	55.3	**Average seek length**	27.5	**Average seek length**	27.8	**Average seek length**	35.8



6. Final conclusion:

- The performance of disk scheduling algorithm depends heavily on the total number of head movement, seek time and rotational latency.
- With the classical approach of disk scheduling algorithm, few algorithms like SSTF and LOOK will be the most efficient algorithm compared to FCFS, SCAN, C-SCAN and C-LOOK disk scheduling algorithm with respect to these parameters.

8. Important points for understanding:

Disk scheduling is important because:

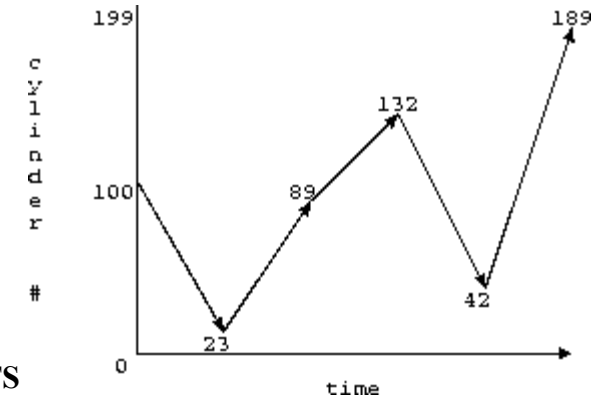
- Multiple I/O requests may arrive by different processes and only one I/O request can be served at a time by the disk controller. Thus other I/O requests need to wait in the waiting queue and need to be scheduled.
- Two or more request may be far from each other so can result in greater disk arm movement.

- Hard drives are one of the slowest parts of the computer system and thus need to be accessed in an efficient manner.

9.Examples of Disk Scheduling Algorithms

- Work Queue: 23, 89, 132, 42, 187
- there are 200 cylinders numbered from 0 - 199
- the diskhead stars at number 100

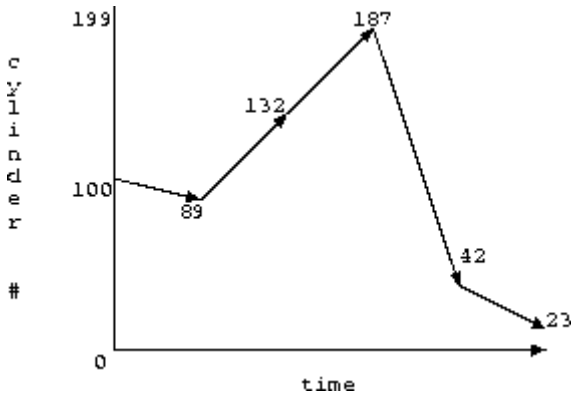
1. **FCFS**



- total time is estimated by total arm motion

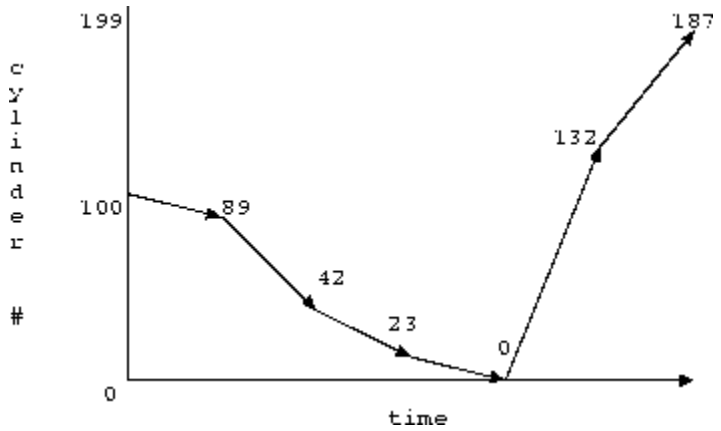
$$| 100 - 23 | + | 23 - 89 | + | 89 - 132 | + | 23 - 132 | + | 132 - 42 | + | 42 - 187 |= 77$$

2. **SSTF**



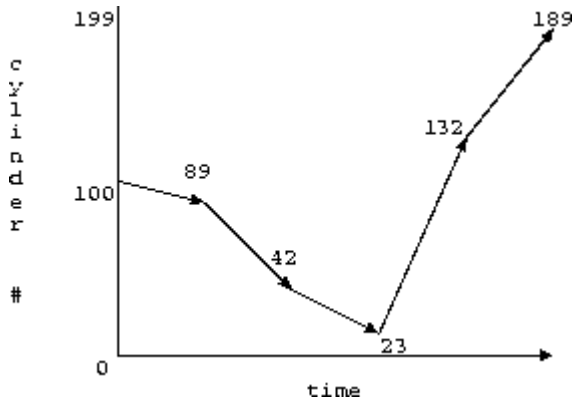
$$| 100 - 89 | + | 89 - 132 | +132 - 187 | + | 187 - 42 | + | 42 - 23 |= 11 + 43 + 55 + 145 + 19 = 273$$

3. **SCAN**
    - assume we are going inwards (i.e., towards 0)



$$| 100 - 89 | + | 89 - 42 | + | 42 - 23 | + | 23 - 0 | + | 0 - 132 | + | 132 - 187 | = 11 + 47 + 19 +$$

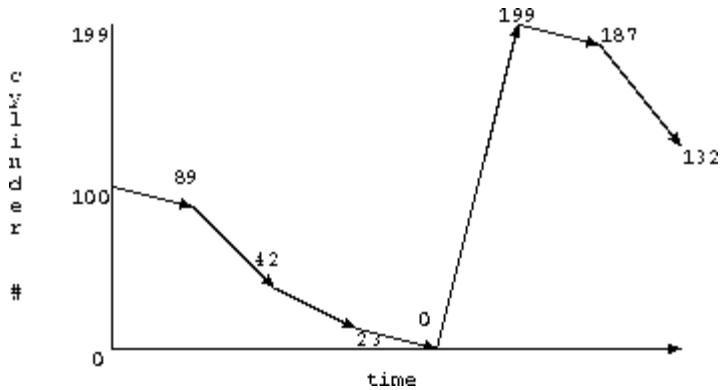
4. **LOOK**



$$| 100 - 89 | + | 89 - 42 | + | 42 - 23 | + | 23 - 132 | + | 132 - 187 | = 11 + 47 + 19 + 109 + 55 = 241$$

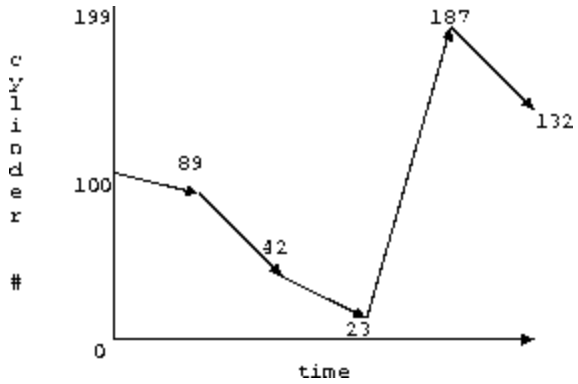
- reduce variance compared to SCAN

5. **C-SCAN**



$$| 100 - 89 | + | 89 - 42 | + | 42 - 23 | + | 23 - 0 | + | 0 - 199 | + | 199 - 187 | + | 187 - 132 | = 11 +$$

6. **C-LOOK**



$$|100-89|+|89-42|+|42-23|+|23-187|+|187-132|=11+47+19+164+55=$$