Disk Scheduling

This tutorial is prepared for those that need assistance in Disk Scheduling Algorithms.

Concept: In operating systems, seek time is very important. Since all device requests are linked in queues, the seek time is increased causing the system to slow down. Disk Scheduling Algorithms are used to reduce the total seek time of any request. The purpose of this material is to provide one with help on disk scheduling algorithms. **Hopefully with this, one will be able to get a stronger grasp of what disk scheduling algorithms do**.

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.

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.

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 Response Time:** Response Time is the average of time spent by a request waiting to perform its I/O operation. Average Response time is the response time of the all requests. Variance Response Time is measure of how individual request are serviced with respect to average response time. So the disk scheduling algorithm that gives minimum variance response time is better.

Some popular disk scheduling algorithms are outlined below:

1. **First Come-First Serve (FCFS):**

All incoming requests are placed at the end of the queue. Whatever number that is next in the queue will be the next number served.

2. **Shortest Seek Time First (SSTF):**

In this case request is serviced according to next shortest distance.

3. **Elevator (SCAN):**

This approach works like an elevator does. It scans down towards the nearest end and then when it hits the bottom it scans up servicing the requests that it didn't get going down. If a request comes in after it has been scanned it will not be serviced until the process comes back down or moves back up.

4. **Circular SCAN (C-SCAN):**

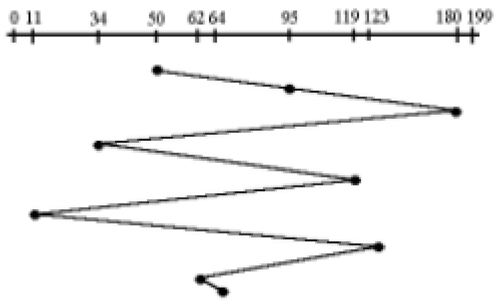
Circular scanning works just like the elevator to some extent. It begins its scan toward the nearest end and works it way all the way to the end of the system. Once it hits the bottom or top it jumps to the other end and moves in the same direction. Keep in mind that the huge jump doesn't count as a head movement.

5. **LOOK:**

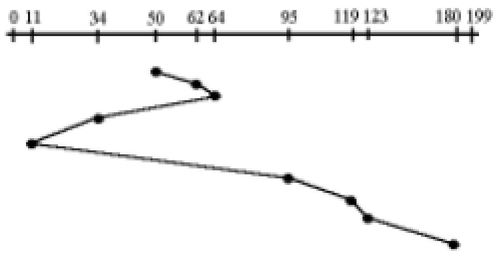
It is similar to the SCAN disk scheduling algorithm except for the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only. Thus it prevents the extra delay which occurred due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

- **Example:** Given the following queue -- 95, 180, 34, 119, 11, 123, 62, 64 with the Read-write head initially at the track 50 and the tail track being at 199 let us now depict the different algorithms.

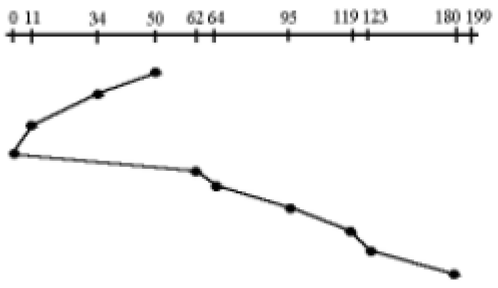
1. FCFS



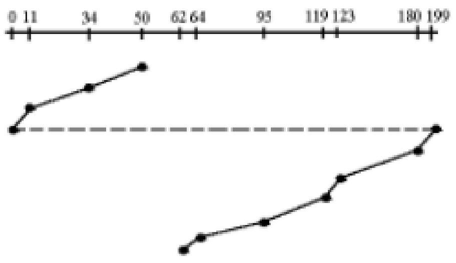
2. SSTF



3. SCAN



4. C-SCAN



Consider a disk scheduling algorithm **ALGO_1** that is an enhanced version of C-SCAN, where the scanning doesn't go past the last request in the direction that it is moving. It too jumps to the other end but not all the way to the end. Just to the furthest request.

Question: Write a code for **ALGO_1,** and plot the variation of total seek time for each algorithm, for the following order of requests. Comment on your answer.

For a disk with 200 track:

Order 1. (82, 170, 43, 140, 24, 16, 190)

Order 2. 180, 34, 119, 11, 123, 62, 64

Order 3. 89, 52, 61, 87, 25, 45, 21, 67, 90, 4, 50

For a disk with 100 track:

Order 3. 45, 21, 67, 90, 4, 50, 89, 52, 61, 87, 25