Operating Systems Z02-54c

Task 2 Discheduling algorithms

FCFS, SSTF, SCAN, C-SCAN, EDF, FD-SCAN

Mustafa Tayyip BAYRAM 257639

OVERVIEW

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.

Disk scheduling is a policy of operating system to decide which I/O request is going to be satisfied foremost. The goal of disk scheduling algorithms is to maximize the throughput and minimize the response time.

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

Terms

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.

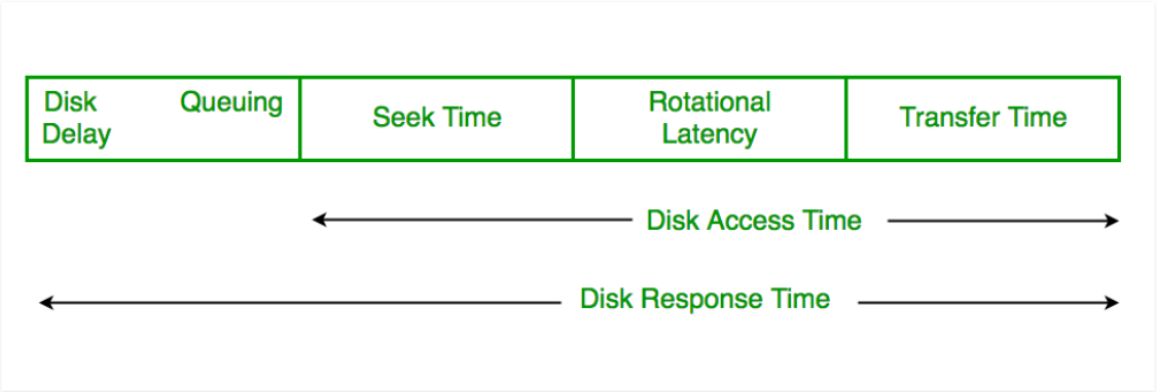
Rotational Latency : It is the time taken by the desired sector to rotate itself to the position from where it can access the R/W heads.

Transfer Time : It is the time taken to the transfer the data.

Disk Accesss Time : Disk access time is given as,

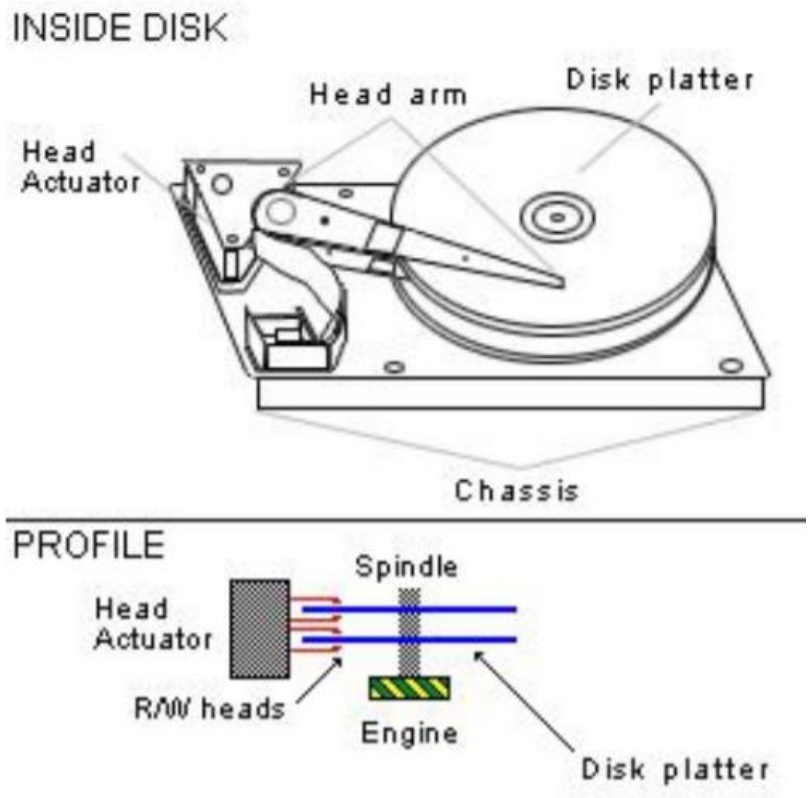
Disk Access Time = Rotational Latency + Seek Time + Transfer Time

Disk Response Time : It is the average of time spent by each request waiting for the IO operation.



Purpose of Disk Scheduling

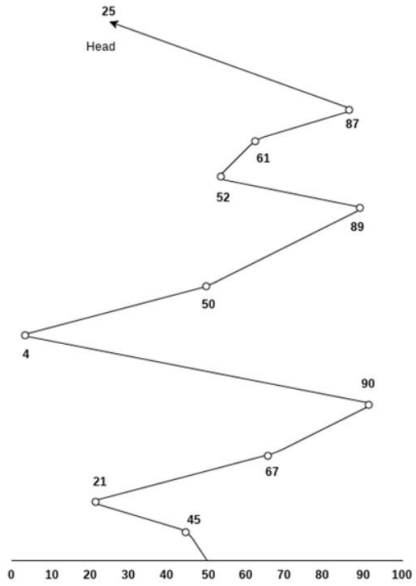
The main purpose of disk scheduling algorithm is to select a disk request from the queue of IO requests and decide the schedule when this request will be processed.



Disk Scheduling Algorithms

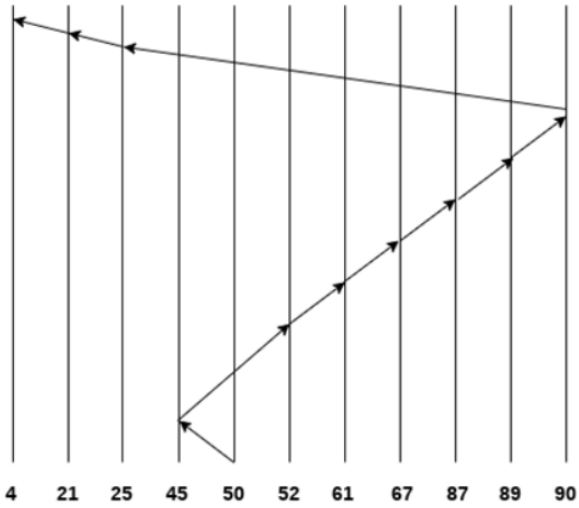
1. **FCFS (First Come – First Serve) :** It is the simplest Disk Scheduling algorithm. It services the IO requests in the order in which they arrive. There is no starvation in this algorithm, every request is serviced.
    - Simplest, perform operations in order requested
    - no reordering of work queue
    - no starvation: every request is serviced
    - Doesn't provide fastest service

    - Example : Consider the following disk request sequence for a disk with 100 tracks 45, 21, 67, 90, 4, 50, 89, 52, 61, 87, 25



Number of cylinders moved by the head = (50-45)+(45-21)+(67-21)+(90-67)+(90-4)+(50-4)+(89-50)+(61-52)+(87-61)+(87-25) = 5 + 24 + 46 + 23 + 86 + 46 + 49 + 9 + 26 + 62= 376

2. **SSTF (Shortest Seek Time First) :** Shortest seek time first (SSTF) algorithm selects the disk I/O request which requires the least disk arm movement from its current position regardless of the direction. It reduces the total seek time as compared to FCFS. It allows the head to move to the closest track in the service queue.
    - Like SJF, select the disk I/O request that requires the least movement of the disk arm from its current position, regardless of direction
    - reduces total seek time compared to FCFS.
    - Disadvantages
        - starvation is possible; stay in one area of the disk if very busy
        - switching directions slows things down
        - Not the most optimal
    - Example : Consider the following disk request sequence for a disk with 100 tracks 45, 21, 67, 90, 4, 89, 52, 61, 87, 25
      Head pointer starting at 50. Find the number of head movements in cylinders using SSTF scheduling.



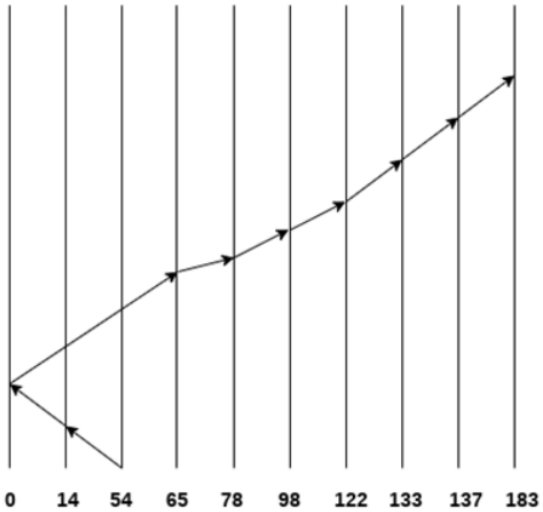
Number of cylinders = 5 + 7 + 9 + 6 + 20 + 2 + 1 + 65 + 4 + 17 = 136

3. **SCAN :** It is also called as Elevator Algorithm. In this algorithm, the disk arm moves into a particular direction till the end, satisfying all the requests coming in its path,and then it turns backand moves in the reverse direction satisfying requests coming in its path.It works in the way an elevator works, elevator moves in a direction completely till the last floor of that direction and then turns back.
    - Example : Consider the following disk request sequence for a disk with 100 tracks

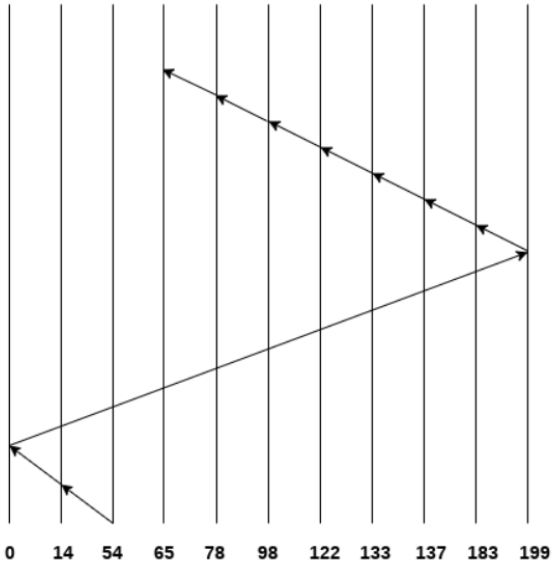
      98, 137, 122, 183, 14, 133, 65, 78

      Head pointer starting at 54 and moving in left direction. Find the number of head movements in cylinders using SCAN scheduling.

      Number of Cylinders = 40 + 14 + 65 + 13 + 20 + 24 + 11 + 4 + 46 = 237



4. **C-SCAN:** In C-SCAN algorithm, the arm of the disk moves in a particular direction servicing requests until it reaches the last cylinder, then it jumps to the last cylinder of the opposite direction without servicing any request then it turns back and start moving in that direction servicing the remaining requests**.**
    - **Example : Consider the following disk request sequence for a disk with 100 tracks**
    - **98, 137, 122, 183, 14, 133, 65, 78**
    - **Head pointer starting at 54 and moving in left direction. Find the number of head movements in cylinders using C-SCAN scheduling.**



No. of cylinders crossed = 40 + 14 + 199 + 16 + 46 + 4 + 11 + 24 + 20 + 13 = 387

5. **EDF :** The Earliest Deadline First algorithm is an analog of FCFS. Requests are orders according to deadline and the request with the earliest deadline is serviced first. Assigning priorities to transactions an Earliest Deadline policy minimizes the number of late transactions in systems operating under low or moderate levels of resources and data contention. This is due to Earliest Deadline steeply degrades in an overloaded system. This is because, under heavy loading, transactions gain high priority only when they are close to their deadlines. Gaining high priority at this late stage may not leave sufficient time for transactions to complete before their deadlines. Under heavy loads, then, a fundamental weakness of the Earliest Deadline priority policy is that it assigns the their deadlines, thus delaying other transactions that might still be able to meet their deadlines .
    - Example : Consider a disk head is currently at cylinder 75 and the queue of cylinders (ordered according to deadlines) is 98, 183, 105. Under strict EDF scheduling, the disk head will move from 75, to 98, to 183, and then back to 105. Note that the head passes over cylinder 105 as it travels from 98 to 183. It is possible that the disk scheduler could have serviced the request for cylinder 105 en route to cylinder 183 and still preserved the deadline requirement for cylinder 183.

6. **FD-SCAN :** In Feasible Deadline-Scan (FD-SCAN), the track location of the request with earliest feasible deadline is used to determine the scan direction. A deadline is feasible if we estimate that it can be met. Each time that a scheduling decision is made, the read requests are examined to determine which have feasible deadlines given the current head position. The request with the earliest feasible deadline is the target and determines the scanning direction. The head scans toward the target servicing read requests along the way. These requests either have deadlines later than the target request or have unfeasible deadlines, ones that cannot be met. If there is no read request with a feasible deadline, then FD-SCAN simply services the closest read request. Since all request deadlines have been missed, the order of service is no longer important for meeting deadlines.

Software Design

Simulation program will take requests as an input and can select algorithm, When request values are run, the graph will appear according to the selected algorithm. Project will be made with WPF.