“TO STIMULATE THE WORKING OF FCFS(FIRST COME FIRST SERVE), SSTF(SHORTEST SEEK TIME FIRST), SCAN, C-SCAN, LOOK AND C-SCAN DISC SCHEDULING ALGORITHMS”

JAPLEEN KAUR[1] , GOKUL JAMWAL[2], DHRUV GUPTA[3], REETIKESH BALI[4]
2021a1r033@mietjammu.in
2021a1r043@mietjammu.in
2021a1r041@mietjammu.in
2021a1r027@mietjammu.in
MODEL INSTITUTE OF ENGINEERING AND TECHNOLOGY,
KOT BHALWAL JAMMU, J&K, INDIA

ABSTRACT

Disc Scheduling algorithm is an algorithm that keeps and manages input and output systems. It is done by the Operating System to schedule I/O requests arriving for the disc. It is also known as I/O scheduling. The goal of disk scheduling algorithms is to maximize the throughput and minimize the response time. The present piece of investigation documents the comparative analysis of six different disk scheduling algorithms viz. First Come First Serve, Shortest Seek Time First, Scan, Look, C-Scan and C-look disk scheduling by comparing their head movement in different runs. The implementation is carried out in Turbo C by creating an interface to calculate total head movement of these six algorithms.

KEYWORDS- disk scheduling , seek time , rotational latency, transfer time, starvation.

1. INTRODUCTION

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms.

IMPORTANT TERMS RELATED TO DISC SCHEDULING-:

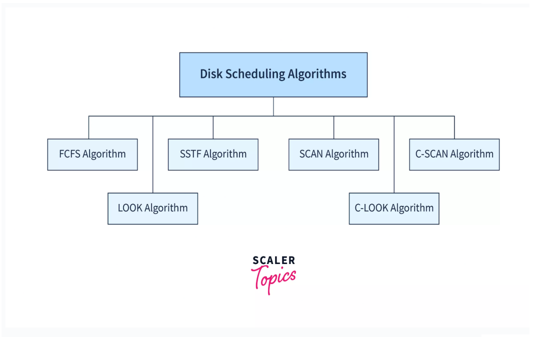
There are many terms that we need to know for a better understanding of Disk Scheduling. We are going to have a brief look at each of them one by one:

1. **Seek Time**: As we know, the data may be stored on various blocks of disk. To access these data according to the request, the disk arm moves and find the required block. The time taken by the arm in doing this search is known as "Seek Time".
2. **Rotational Latency**: The required data block needs to move at a particular position from where the read/write head can fetch the data. So, the time taken in this movement is known as "Rotational Latency". This rotational time should be as less as possible so, the algorithm that will take less time to rotate will be considered a better algorithm.
3. **Transfer Time**: When a request is made from the user side, it takes some time to fetch these data and provide them as output. This time taken is known as "Transfer Time".
4. **Disk Access Time**: It is defined as the total time taken by all the above processes. Disk access time = (seek time + rotational latency time + transfer time)
5. Disk Response Time: The disk processes one request at a single time. So, the other requests wait in a queue to finish the ongoing process

of request. The average of this waiting time is called "Disk Response Time".

6. **Starvation:** Starvation is defined as the situation in which a low-priority job keeps waiting for a long time to be executed. The system keeps sending high-priority jobs to the disk scheduler to execute first

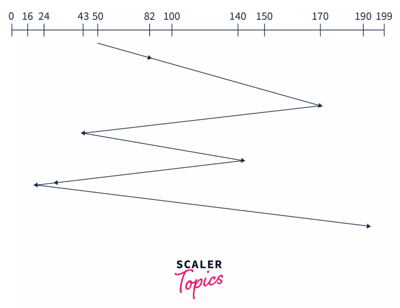
We have various types of Disk Scheduling Algorithms available in our system.Each one has its own capabilities and weak points



1. FCFS disk scheduling algorithm-

It stands for 'first-come-first-serve'. As the name suggests, the request which comes first will be processed first and so on. The requests coming to the disk are arranged in a proper sequence as they arrive. Since every request is processed in this algorithm, so there is no chance of 'starvation'.

Example: Suppose a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) of disk



are shown as in the given figure and the head start is at request 50.

Explanation: In the above image, we can see the head starts at position 50 and moves to request 82. After serving them the disk arm moves towards the second request that is 170 and then to the request 43 and so on. In this algorithm,, the disk arm will serve the requests in arriving order. In this way, all the requests are served in arriving order until the process executes.

"Seek time" will be calculated by adding the head movement differences of all the requests:

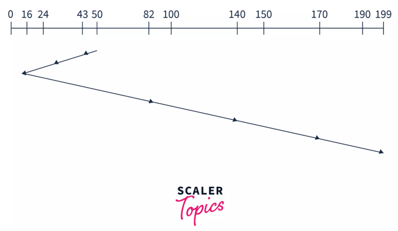
Seek time= "(82-50) + (170-82) + (170-43) + (140-43) + (140-24) + (24-16) + (190-16) = 642

- Advantages:
    1. Implementation is easy.
    2. No chance of starvation.
- Disadvantages:
    1. 'Seek time' increases.
    2. Not so efficient.

2. SSTF disk scheduling algorithm-

It stands for 'Shortest seek time first'. As the name suggests, it searches for the request having the least 'seek time' and executes them first. This algorithm has less 'seek time' as compared to FCFS Algorithm.

Example: : Suppose a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) are shown in the given figure and the head position is at 50.



Explanation: The disk arm searches for the request which will have the least difference in head movement. So, the least difference is (50-43). Here the difference is not about the shortest value but it is about the shortest time the head will take to reach the nearest next request. So, after 43, the head will be nearest to 24, and from here the head will be nearest to the request 16, After 16, the nearest request is 82, so the disk arm will move to serve request 82 and so on.

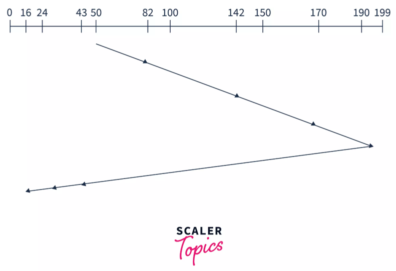
Hence, Calculation of Seek Time = (50-43) + (43-24) + (24-16) + (82-16) + (140-82) + (170-140) + (190-170) = 208

- Advantages:
    1. In this algorithm, disk response time is less.
    2. More efficient than FCFS.
- Disadvantages:
    1. Less speed of algorithm execution.
    2. Starvation can be seen.

3. SCAN disk scheduling algorithm:

In this algorithm, the head starts to scan all the requests in a direction and reaches the end of the disk. After that, it reverses its direction and starts to scan again the requests in its path and serves them. Due to this feature, this algorithm is also known as the "Elevator Algorithm".

Example: Suppose a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) are shown in the given figure and the head position is at 50. The 'disk arm' will first move to the larger values.



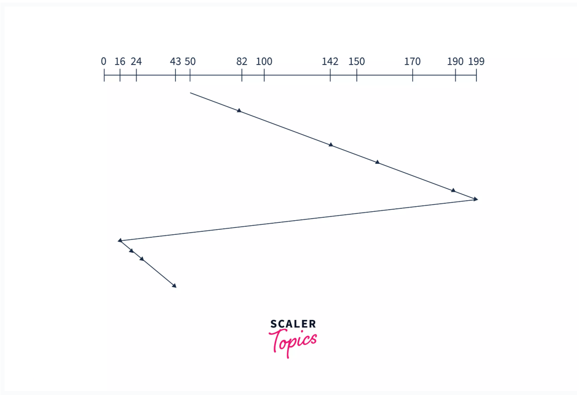
Explanation: In the above image, we can see that the disk arm starts from position 50 and goes in a single direction until it reaches the end of the disk i.e.- request position 199. After that, it reverses and starts servicing in the opposite direction until reached the other end of the disk. This process keeps going on until the process is executed. Hence, Calculation of 'Seek Time' will be like: (199-50) + (199-16) =332

- Advantages:
    1. Implementation is easy.
    2. Requests do not have to wait in a queue.
- Disadvantage:
    1. The head keeps going on to the end even if there are no requests in that direction.

4. C-SCAN disk scheduling algorithm:

It stands for "Circular-Scan". This algorithm is almost the same as the Scan disk algorithm but one thing that makes it different is that 'after reaching the one end and reversing the head direction, it starts to come back. The disk arm moves toward the end of the disk and serves the requests coming into its path. After reaching the end of the disk it reverses its direction and again starts to move to the other end of the disk but while going back it does not serve any requests.

Example: Suppose a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) are shown in the given figure and the head position is at 50.



Explanation: In the above figure, the disk arm starts from position 50 and reached the end(199), and serves all the requests in the path. Then it reverses the direction and moves to the other end of the disk i.e.- 0 without serving any task in the path. After reaching 0, it will again go move towards the largest remaining value which is 43. So, the head will start from 0 and moves to request 43 serving all the requests coming in the path. And this process keeps going.

Hence, Seek Time will be =(199-50) + (199-0) + (43-0) =391

Advantages:

1. The waiting time is uniformly distributed among the requests.
2. Response time is good in it.

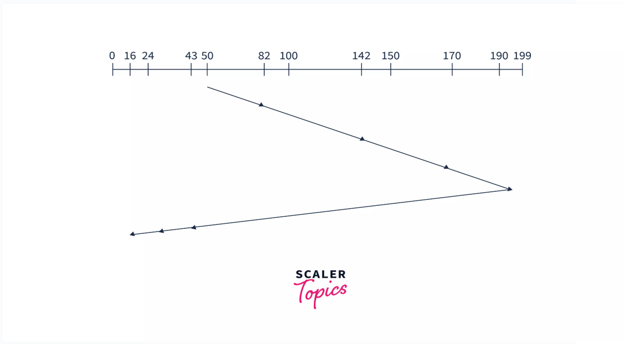
- Disadvantages:

1. The time taken by the disk arm to locate a spot is increased here.
2. The head keeps going to the end of the disk.

5. LOOK the disk scheduling algorithm:

In this algorithm, the disk arm moves to the 'last request' present and services them. After reaching the last requests, it reverses its direction and again comes back to the starting point. It does not go to the end of the disk, in spite, it goes to the end of requests.

Example a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) are shown in the given figure and the head position is at 50.



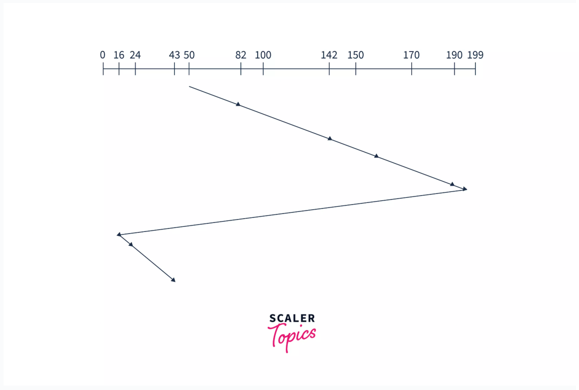
Explanation: The disk arm is starting from 50 and starts to serve requests in one direction only but in spite of going to the end of the disk, it goes to the end of requests i.e.-190. Then comes back to the last request of other ends of the disk and serves them. And again starts from here and serves till the last request of the first side. Hence, Seek time =(190-50) + (190-16) =314

- Advantages:
    1. Starvation does not occur.
    2. Since the head does not go to the end of the disk, the time is not wasted here.
- Disadvantage:
    1. The arm has to be conscious to find the last request.

6. C-LOOK disk scheduling algorithm:

The C-Look algorithm is almost the same as the Look algorithm. The only difference is that after reaching the end requests, it reverses the direction of the head and starts moving to the initial position. But in moving back, it does not serve any requests.

Example: Suppose a disk having 200 tracks (0-199). The request sequence(82,170,43,140,24,16,190) are shown in the given figure and the head position is at 50.



Explanation: The disk arm is starting from 50 and starts to serve requests in one direction only but in spite of going to the end of the disk, it goes to the end of requests i.e.-190. Then comes back to the last request of other ends of a disk without serving them. And again starts from the other end of the disk and serves requests of its path. Hence, Seek Time = (190-50) + (190-16) + (43-16) =341

- Advantages:
    1. The waiting time is decreased.
    2. If there are no requests till the end, it reverses the head direction immediately.
    3. Starvation does not occur.
    4. The time taken by the disk arm to find the desired spot is less.
- Disadvantage:
    1. The arm has to be conscious about finding the last request.

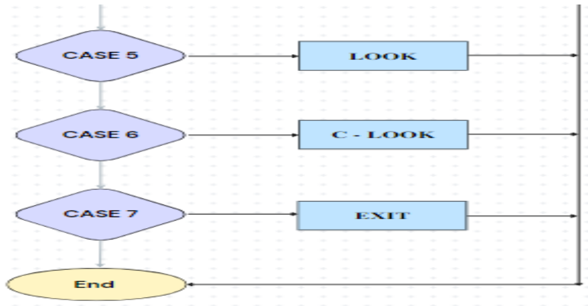
2.FLOWCHART-:

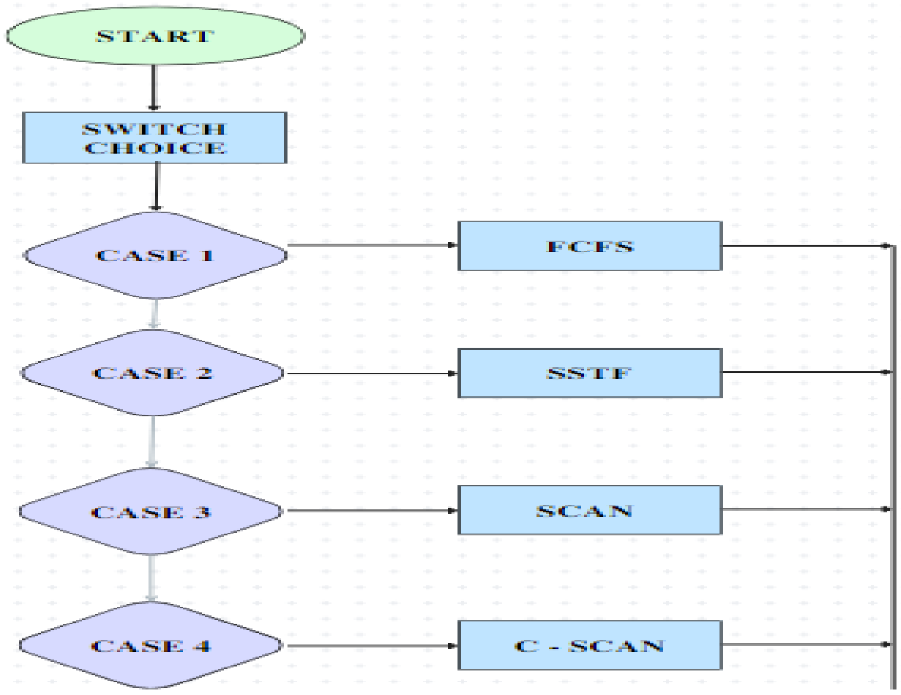
Overview

- To identify the disc scheduling algorithm followed by each process entering the CPU
- For process synchronization
- To maximize the efficiency of CPU

Flowchart

This visual map shows us. The working of FCFS,SSTF,SCAN,C-SCAN,LOOK,AND C-LOOK DISC -SCHEDULING ALOGORITHM





3.CONCLUSION:

The present paper analyzes the six disk scheduling algorithms viz. First Come First Serve,Shortest Seek Time First, Scan, Look, C-Scan and C-Look. The implementation was carried out using Turbo C. Different disk scheduling algorithms can be used depending upon the load of the system. The performance depends upon the number and types of requests.

5.SUMMARY:

We have calculated the average total head movement after entering the various runs for the requests of different algorithms because total head movement is the criteria for analyzing the disk scheduling algorithms. After comparing the total head movement of various algorithms, we have found that the Shortest Seek Time First disk scheduling algorithm has the least average head movement than the others discussed above in context to total head movement.

6. FUTURE WORK:

In the future, we will use random requests and some other parameters to compare the different disk scheduling algorithms.

7. REFERENCES:

[1] William Stallings, Operating Systems: Internals and Design Principles (India: Pearson,2009).

[2] Abraham Silberschatz, Peter B. Galvin and Greg Gagne, Operating System Concepts, Eight ed., UK: Wiley, 2010.

[3] Andrew S. Tanenbaum, Modern Operating Systems (USA: Prentice-Hall, Inc., 2001).

[4] John Ryan Celis et. al., “A Comprehensive Review for Disk Scheduling Algorithms,”International Journal of Computer Science, vol. 11, issue 1, no. 1, pp. 74-79, Jan 2014.

[5] Nidhi and Dayashankar Singh, “A New Optimized Real-Time Disk Scheduling Algorithm,” International Journal of Computer Applications, vol. 93, no. 18, pp. 7-12, May 2014.

[6] Hetal Paida, “Disk Scheduling,” International Journal of Advanced Research in Computer Science and Management Studies, vol. 1, issue 2, pp. 6-9, July 2013.

[7] Sarita Negi, Kulvinder Singh and Shubam Sahu, “Demand Based Disk Scheduling Using Circular Look Algorithm,” International Journal of Computer Science and Mobile Computing, vol. 4, issue 8, pp. 364-368, Aug. 2015.

[8] Priya Hooda and Supriya Raheja, "A New Approach to Disk Scheduling Using Fuzzy Logic," Journal of Computer and Communications, vol. 2, pp. 1-5, 2014.

[9] Sandipon Saha, Md. Nasim Akhter and Mohammod Abul Kashem, "A New Heuristic Disk Scheduling Algorithm," International Journal of Scientific & Technology Research, vol. 2, issue 1, pp. 49-53, Jan. 2013

[10] GEEKSFORGEEKS

[11] https://www.researchgate.net/publication/326331286_A_Comparative_Analysis_of_Disk_Scheduling_Algorithms