Jaypee Institute of Information Technology, Noida

Project Report

Operating System and System Programming lab

[15B17CI472]

Comparative Analysis Of Disk Scheduling Algorithms

MEMBERS

1.	Khushboo Kumari	B13	19803003
2.	Kanistha	B13	19803008
3.	Ritawari Pareek	B13	19803009
4.	Kanishka Khullar	B13	19803011
5.	Pranav Gupta	B13	19803021

SUBMITTED TO

Dr Ashish Mishra

Problem Statement

Analysing and Simulating various Disk Scheduling Algorithms using given inputs.

Introduction

Operating systems use disk scheduling to schedule I/O requests that arrive at the disk. I/O scheduling is another name for disk scheduling.

- ➔ Disk scheduling is necessary because:
  - ❖ Several I/O requests may arrive from various processes, and the disk controller can only service one I/O request at a time. As a result, further I/O requests must wait in the queue and be scheduled.
  - ❖ Two or more requests may be separated by a significant distance, resulting in increased disk arm movement.
  - ❖ Hard drives are one of the slowest components in a computer system, thus they must be accessible quickly.

Important Terminologies

Seek Time: The time it takes for the disk arm to locate a specific track where data is to be read or written is known as seek time. As a result, the disk scheduling technique with the shortest average Seek time is preferable.

Rotational Latency: Rotational Latency is the time it takes for the required sector of the disk to rotate into position so that the read/write heads may be accessed. As a result, the disk scheduling approach with the shortest rotational latency is preferable.

Transfer Time: The time it takes to send data is known as the transfer time. It is determined by the disk's rotational speed and the number of bytes to be transferred.

Disk Access Time: Disk Access Time is defined as the summation of Seek Time, Rotational Latency, and Transfer Time.

Disk Response Time: Response Time is the average amount of time a request spends waiting for its I/O operation to complete. The average response time is the sum of all requests' responses. Variance Response Time is a metric that measures how quickly individual requests are handled in comparison to the average response time. As a result, the disk scheduling algorithm with the shortest variance response time is preferable.

Disk Scheduling Algorithms:

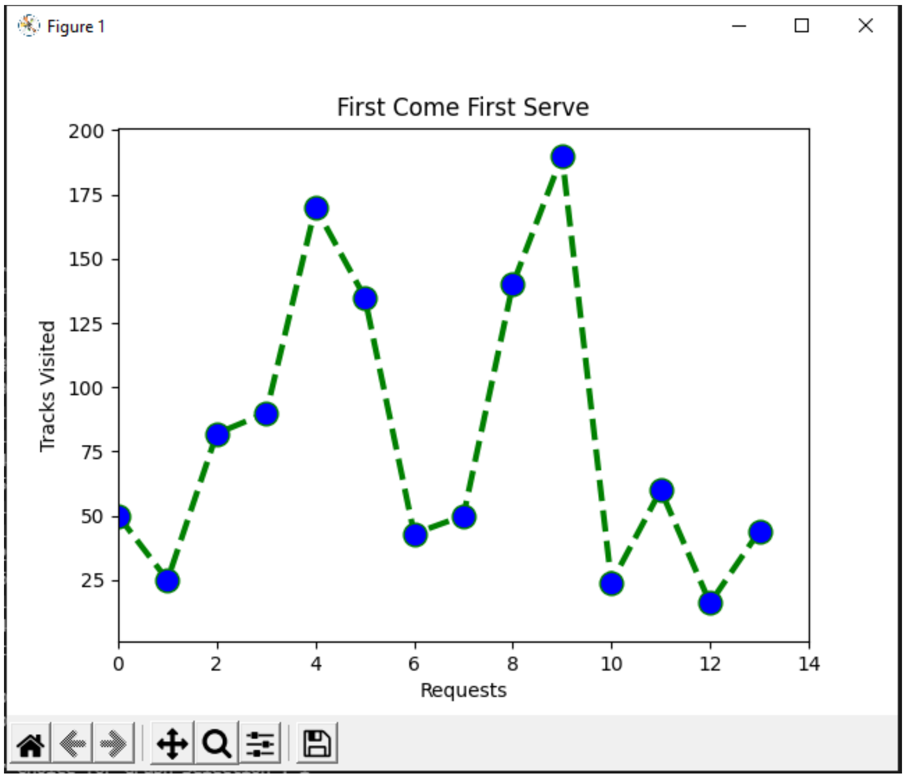
1. First Come First Serve (FCFS) - The FCFS Disk Scheduling Algorithm is the most basic of all the Disk Scheduling Algorithms. Requests in FCFS are handled in the order they come in the disk queue.

ADVANTAGES:

- ➔ The rationale behind the First Come First Serve algorithm is extremely simple: it processes process requests one by one in the order they arrive.
- ➔ As a result, First Come First Serve is a very simple and straightforward concept to grasp and practice.
- ➔ In FCFS, each and every process has a chance to run at some point, therefore there is no starvation.

DISADVANTAGES:

- ➔ This scheduling method is non-preemptive, which means it can't be halted in the middle of a task and will complete it in its entirety.
- ➔ Because FCFS is a non-preemptive scheduling technique, short processes at the back of the queue must wait for the long process at the front to finish before proceeding.
- ➔ FCFS's throughput isn't particularly efficient.
- ➔ FCFS is only used on tiny systems where input-output efficiency isn't a top priority.



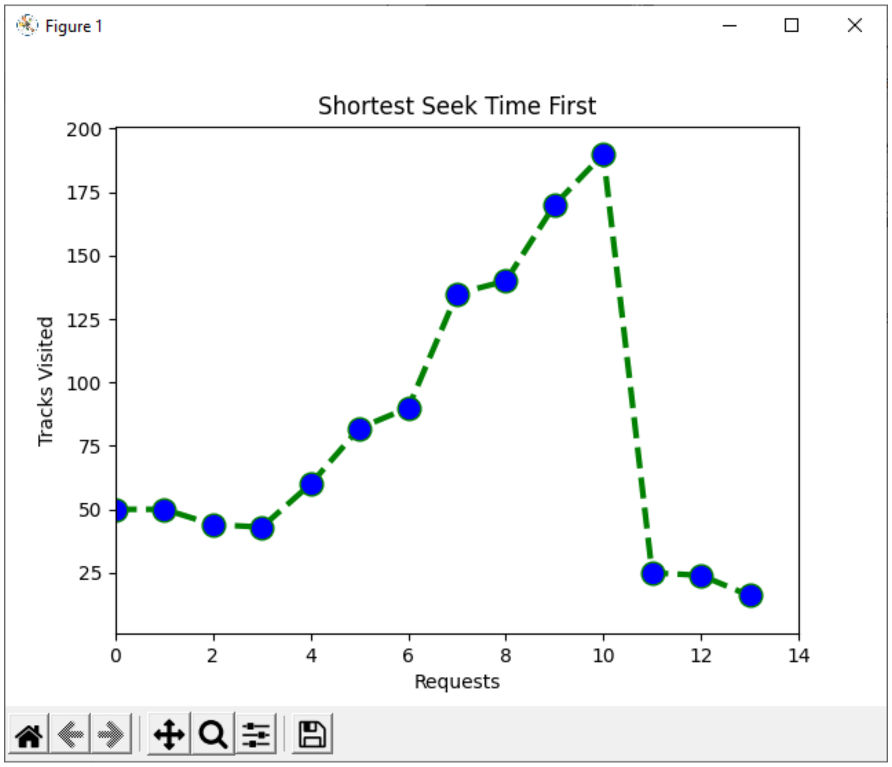
2. Shortest Seek Time First(SSTF) - The requests with the shortest seek times are processed first. As a result, each request's seek time is computed in advance in the queue, and then they are scheduled based on that determined seek time. As a result, the request that is closest to the disk arm will be processed first. SSTF is a significant improvement over FCFS since it reduces average response time and enhances system throughput.

ADVANTAGES:

- ➔ When compared to First Come First Serve, the total seek time is minimized.
- ➔ SSTF enhances and expands throughput.
- ➔ SSTF has a shorter average waiting time and response time.

DISADVANTAGES:

- ➔ The overhead of identifying the closest request in SSTF is significant.
- ➔ Requests that are too far away from the head may result in starvation.
- ➔ Response time and waiting time have a lot of variance in SSTF.
- ➔ The process is slowed by frequent changes in the Head's direction.



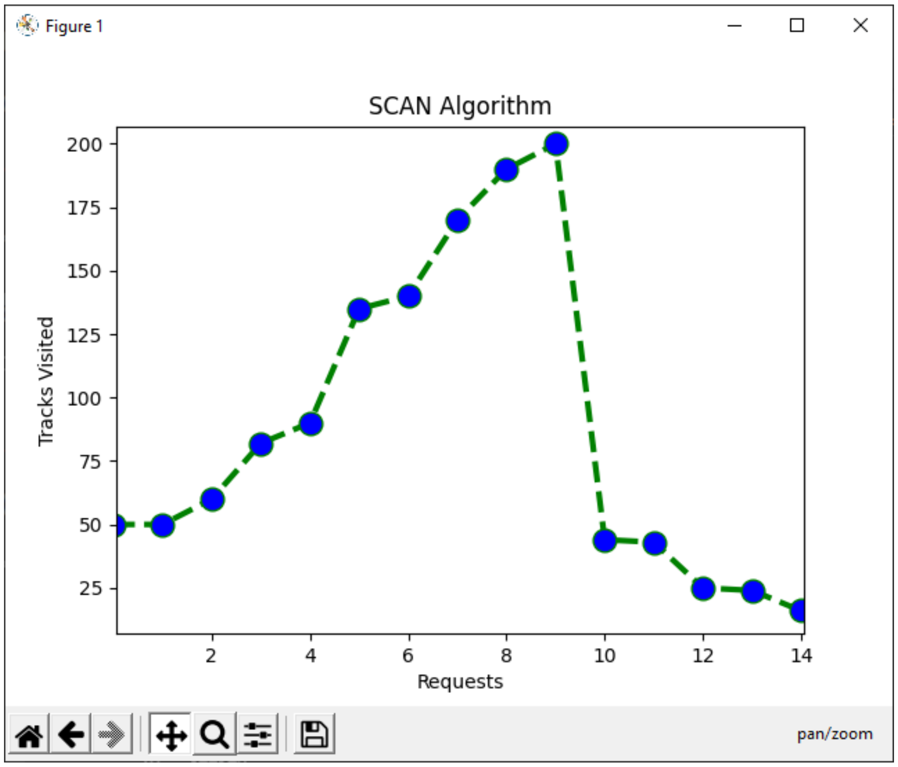
3. SCAN - In the SCAN algorithm, the disk arm moves in a specific direction and serves the requests that come in its way, then reverses direction and services the requests that come its way again after reaching the end of the disk. As a result, this algorithm is also known as the elevator algorithm because it acts as an elevator. As a result, requests at the midpoint receive greater attention, while those arriving after the disk arm must wait.

ADVANTAGES:

- ➔ The scan scheduling approach is straightforward and simple to comprehend and apply.
- ➔ The SCAN algorithm avoids starvation.
- ➔ Waiting time and response time are both affected as low variance occurs

DISADVANTAGES:

- ➔ For the cylinders that the head has just visited, there is a long wait.
- ➔ Despite the absence of requests to be served, the head goes to the end of the disk in SCAN.



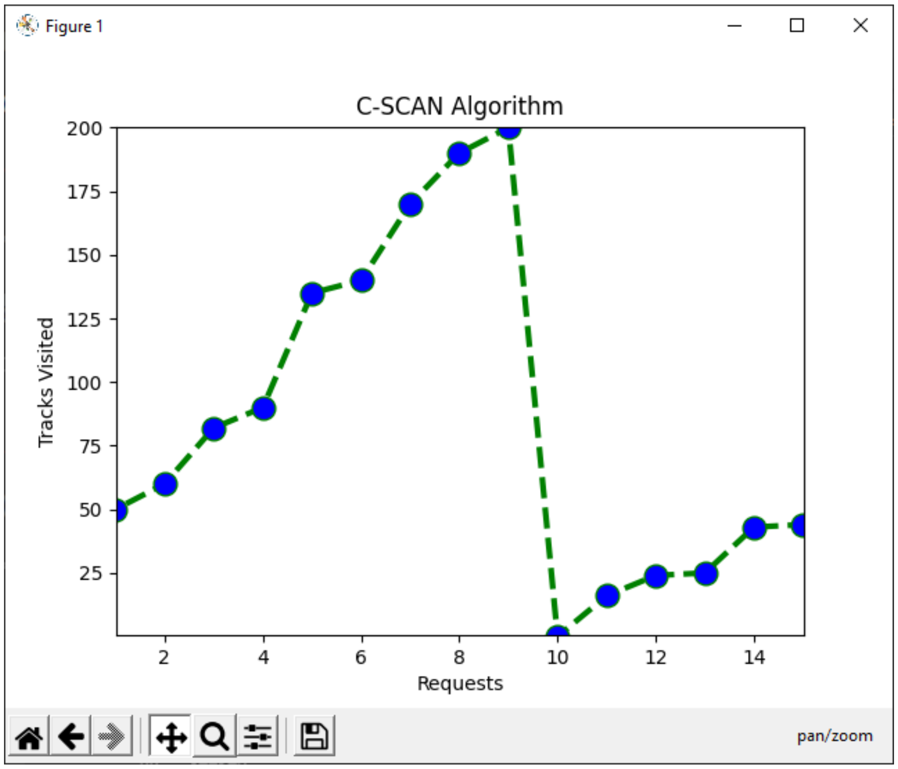
4. C-SCAN - Instead of reversing direction, the disk arm in the C-SCAN algorithm moves to the opposite end of the disk and begins serving requests from there. As a result, the disk arm moves in a circular pattern, and this process is comparable to the SCAN algorithm, hence the name C-SCAN (Circular SCAN).

ADVANTAGES:

- ➔ The C-SCAN Algorithm is the upgraded successor of the SCAN Scheduling Algorithm.
- ➔ The Head moves from one end of the disk to the other, serving all requests along the way.
- ➔ When compared to the SCAN Algorithm, C-SCAN reduces the time spent waiting for cylinders that have recently been visited by the head.
- ➔ A consistent amount of waiting time is supplied.
- ➔ There is a faster reaction time.

DISADVANTAGES:

- ➔ When compared to the SCAN Algorithm, C-SCAN causes more seek motions.
- ➔ Unlike the SCAN algorithm, the Head in C-SCAN will travel to the end of the disk even if there are no requests left to be served.



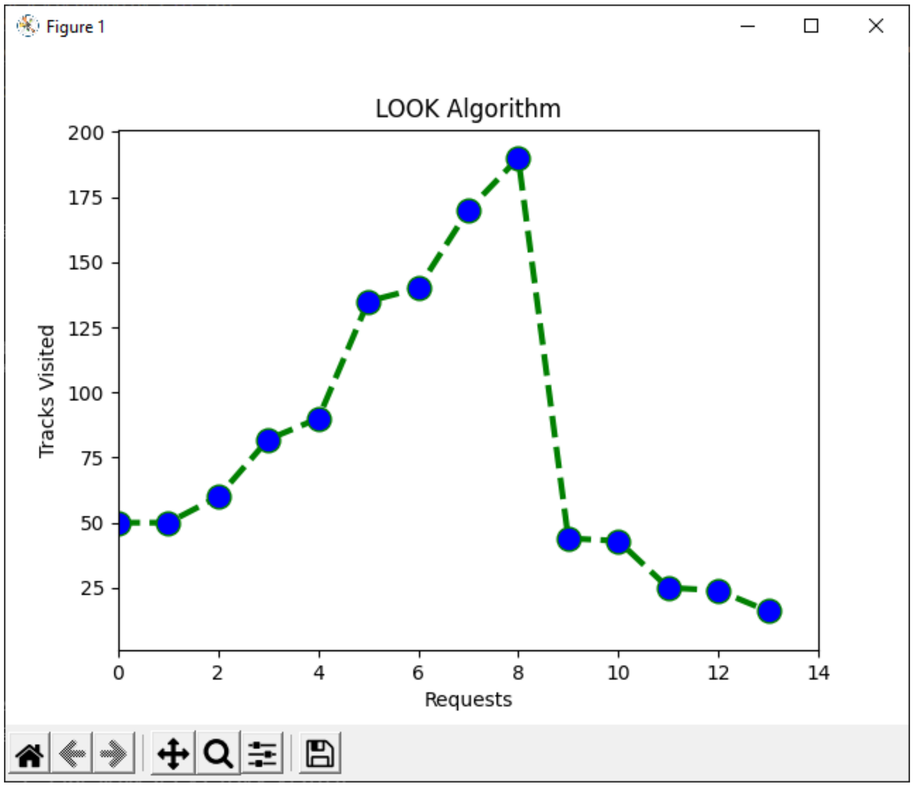
5. LOOK - It is identical to the SCAN disk scheduling algorithm, except that instead of travelling to the end of the disk, the disk arm only goes to the last request to be handled in front of the head and then reverses course from there. As a result, the extra time caused by unneeded traversal to the disk's end is avoided.

ADVANTAGES:

- ➔ Unlike the SCAN algorithm, the Head will not travel to the end of the disc if there are no requests remaining to be served.
- ➔ When compared to the SCAN Algorithm, this algorithm provides better results.
- ➔ The LOOK scheduling algorithm avoids starvation.
- ➔ Waiting time and response time have a low variance.

DISADVANTAGES:

- ➔ There is a cost associated with locating the final requests.
- ➔ Cylinders that Head has just visited must wait a long time.



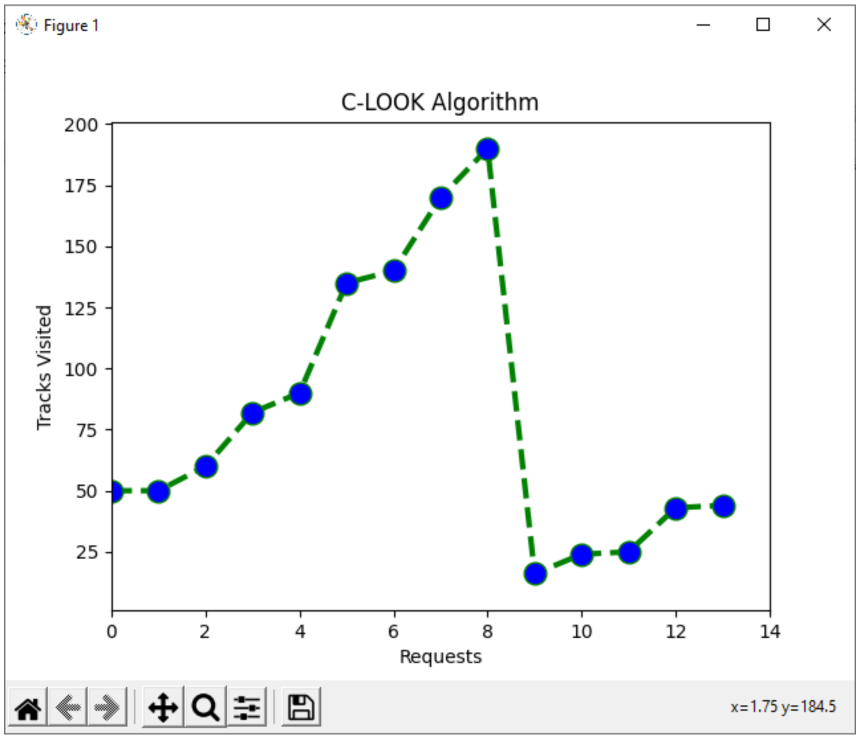
6. C-LOOK - C-LOOK is comparable to the C-SCAN disc scheduling method in the same way that LOOK is related to the SCAN algorithm. Despite travelling to the end, the disc arm in C-LOOK only goes to the last request to be serviced in front of the head and then to the opposite end's last request. As a result, it avoids the additional time caused by unneeded traversal to the disk's end.

ADVANTAGES:

- ➔ If there are no requests to be serviced, the head does not have to move all the way to the end of the disc in C-LOOK.
- ➔ In C-LOOK, there is less waiting time for cylinders that are only visited by the head.
- ➔ When compared to the LOOK Algorithm, C-LOOK performs better.
- ➔ In C-LOOK, starvation is avoided.
- ➔ Waiting time and response time have a low variance.

DISADVANTAGES:

- ➔ The overhead of detecting the end requests is evident in C-LOOK.



Tools And Technologies

Language Used: Python 3.

Libraries Used:

1. Heapq : (To Implement Heap Data Structure) In Python, the feature of this data structure is that the smallest of the heap elements gets popped each time (min-heap). The heap structure is preserved whenever elements are pushed or popped. Every time, the heap[0] element returns the smallest element.
2. Matplotlib: Matplotlib is a fantastic Python visualizing package that is simple to use. It is based on NumPy arrays and is intended to operate with the larger SciPy stack. It includes a variety of graphs such as line, bar, scatter, histogram, and others.

Editor Used: VS Code Editor

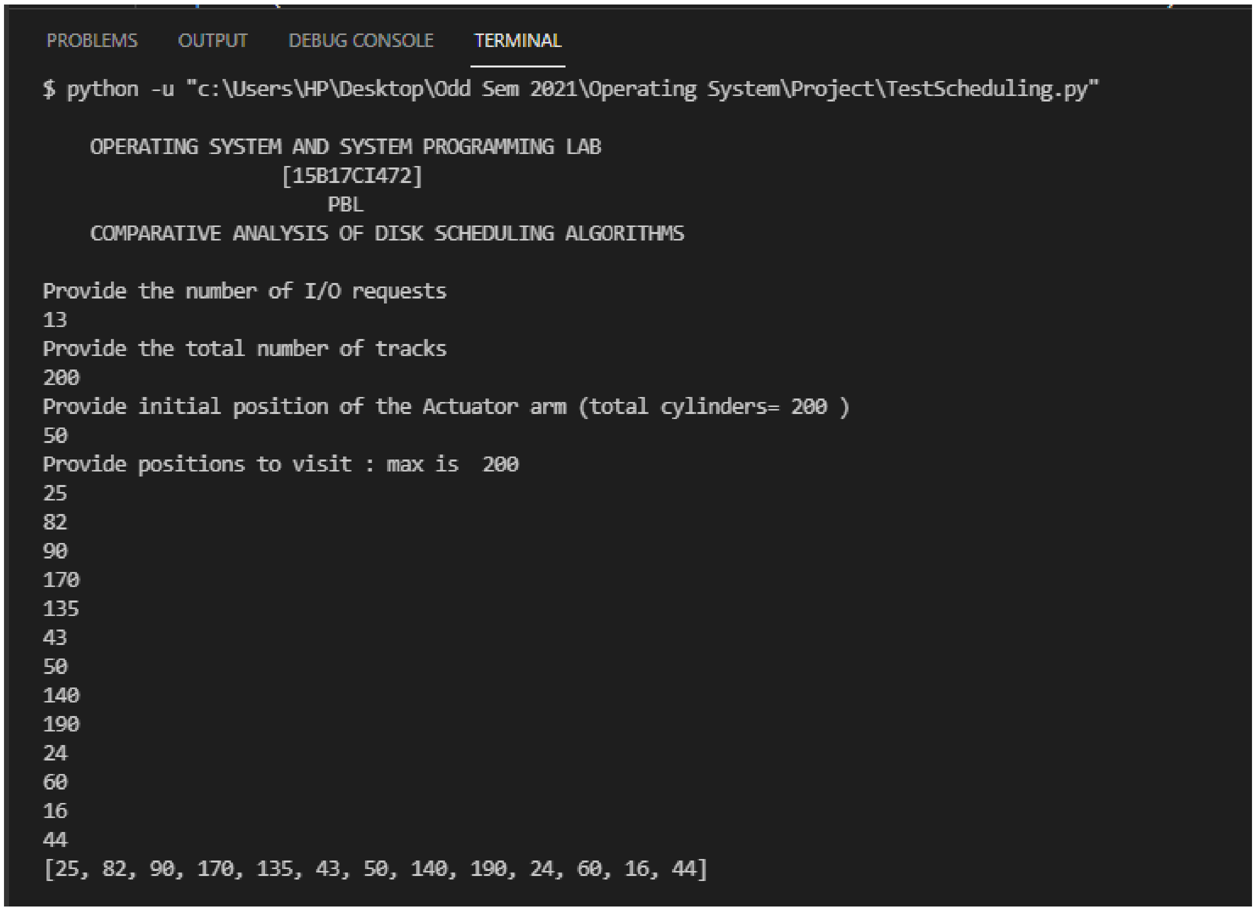
WORKFLOW DIAGRAM

Implementation of disk scheduling algorithms using VS CODE:

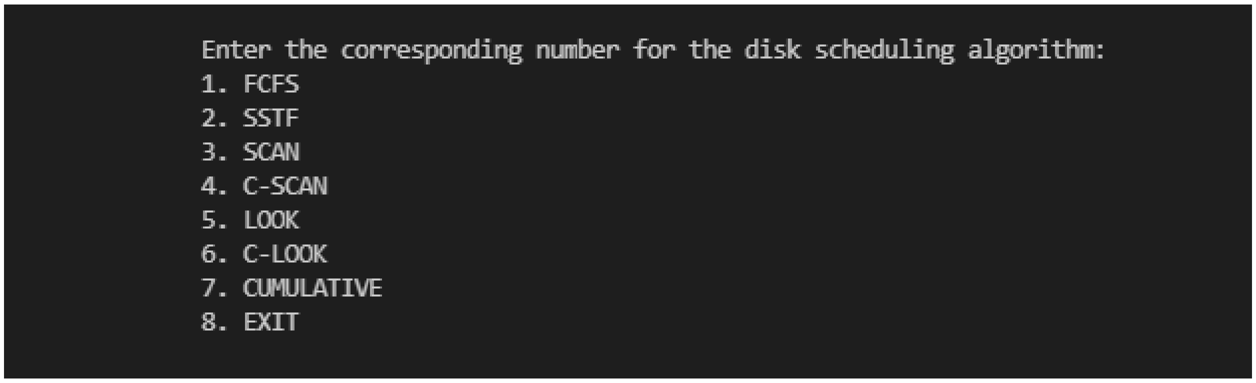
The implementation is done by developing an interface to find out head movements and total head movement in all six disk scheduling methods. The user must select an algorithm and provide basic information such as the total number of requests, initial head position, and disc queue requests. Additional fields for choosing the x vs y or y vs x are available for each graph.

The code basically has following functions

```
def FCFS(hp,requests,t):
def SSTF(hp,reqs, t):
def SCAN(hp,reqs, t):
def C_SCAN(hp,reqs, t):
def LOOK(hp,reqs, t):
def C_LOOK(hp,reqs, t):
def plotSeekSequence(x, y, title,t):
def plotSeekSequenceReverse(x, y, title, t):
def CumulativePlotDesign(x1, x2, x3, x4, x5, x6 ,y1, y2, y3, y4,
y5, y6):
def CumulativePlotDesignReverse(x1, x2, x3, x4, x5, x6 ,y1, y2,
y3, y4, y5, y6):
```

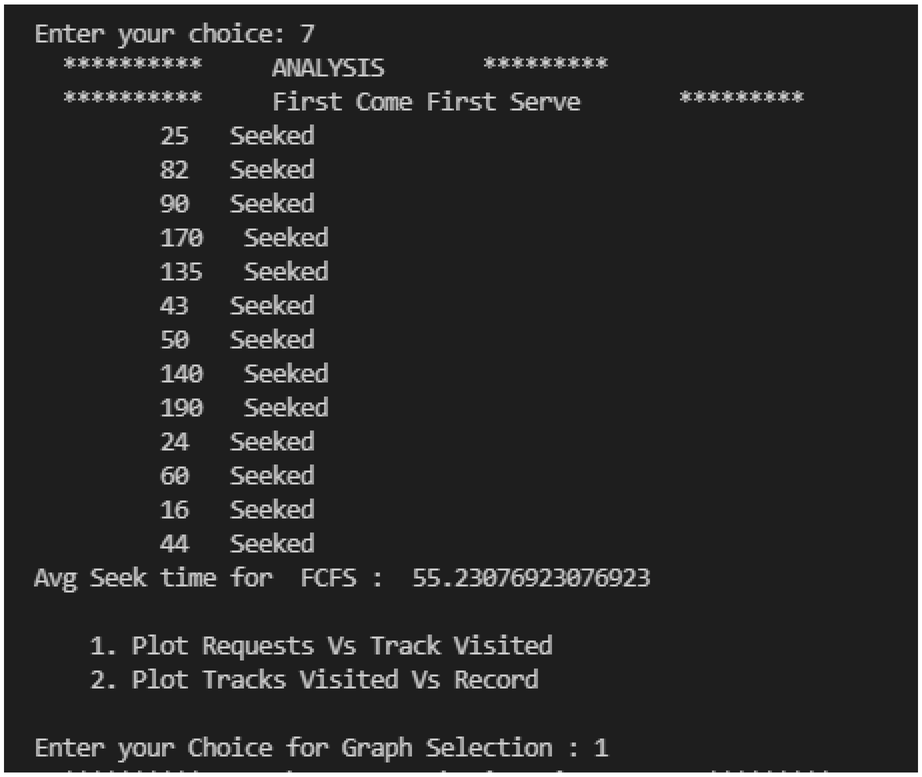


```
$ python -u "c:\Users\HP\Desktop\Odd Sem 2021\Operating System\Project\TestScheduling.py"

    OPERATING SYSTEM AND SYSTEM PROGRAMMING LAB
                    [15B17CI472]
                        PBL
    COMPARATIVE ANALYSIS OF DISK SCHEDULING ALGORITHMS

Provide the number of I/O requests
13
Provide the total number of tracks
200
Provide initial position of the Actuator arm (total cylinders= 200 )
50
Provide positions to visit : max is  200
25
82
90
170
135
43
50
140
190
24
60
16
44
[25, 82, 90, 170, 135, 43, 50, 140, 190, 24, 60, 16, 44]
```



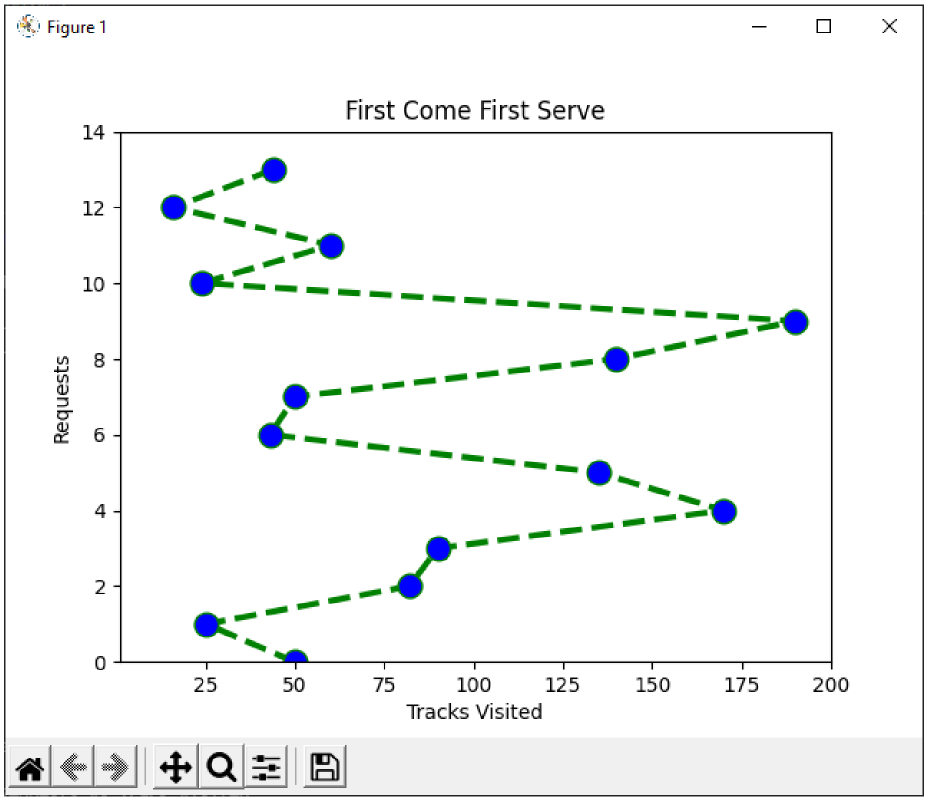
```
Enter the corresponding number for the disk scheduling algorithm:
1. FCFS
2. SSTF
3. SCAN
4. C-SCAN
5. LOOK
6. C-LOOK
7. CUMULATIVE
8. EXIT
```

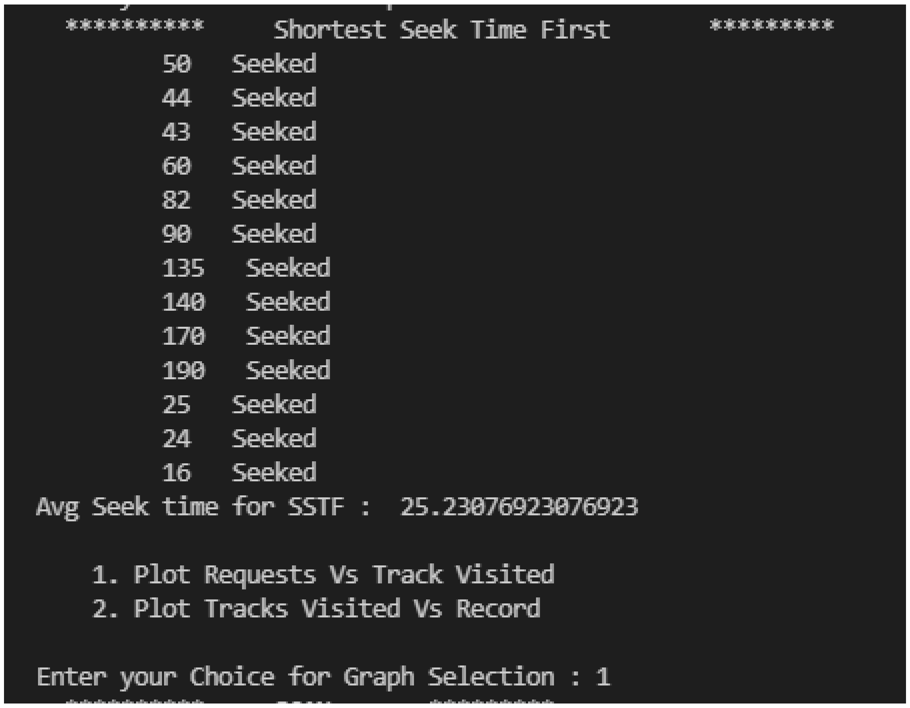
Snapshot Of Results

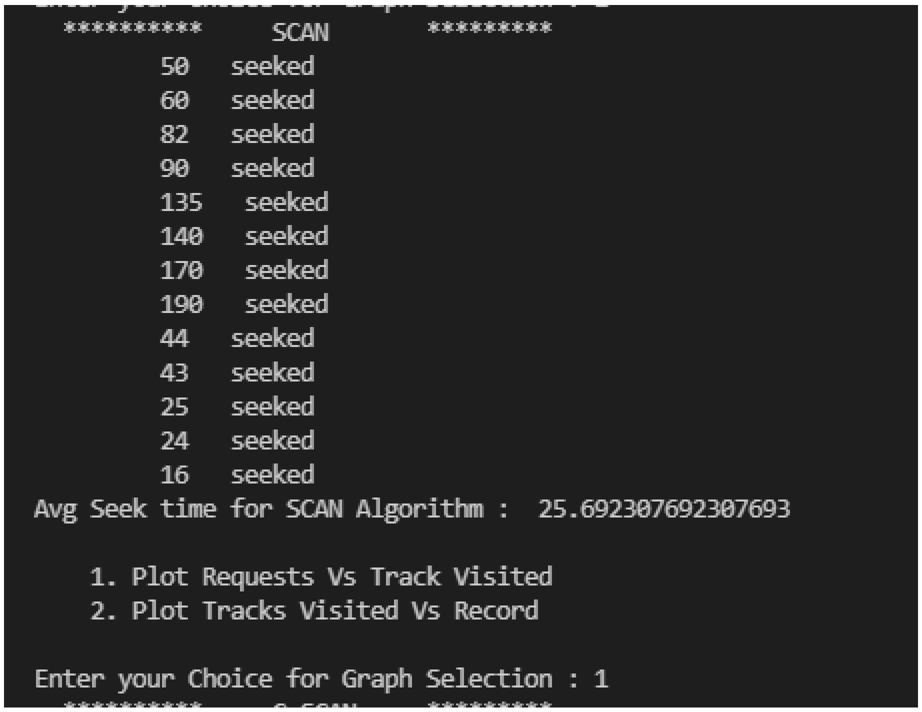


```
Enter your choice: 7
  **********     ANALYSIS       *********
  **********     First Come First Serve       *********
         25   Seeked
         82   Seeked
         90   Seeked
         170   Seeked
         135   Seeked
         43   Seeked
         50   Seeked
         140   Seeked
         190   Seeked
         24   Seeked
         60   Seeked
         16   Seeked
         44   Seeked
Avg Seek time for  FCFS :  55.23076923076923

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```

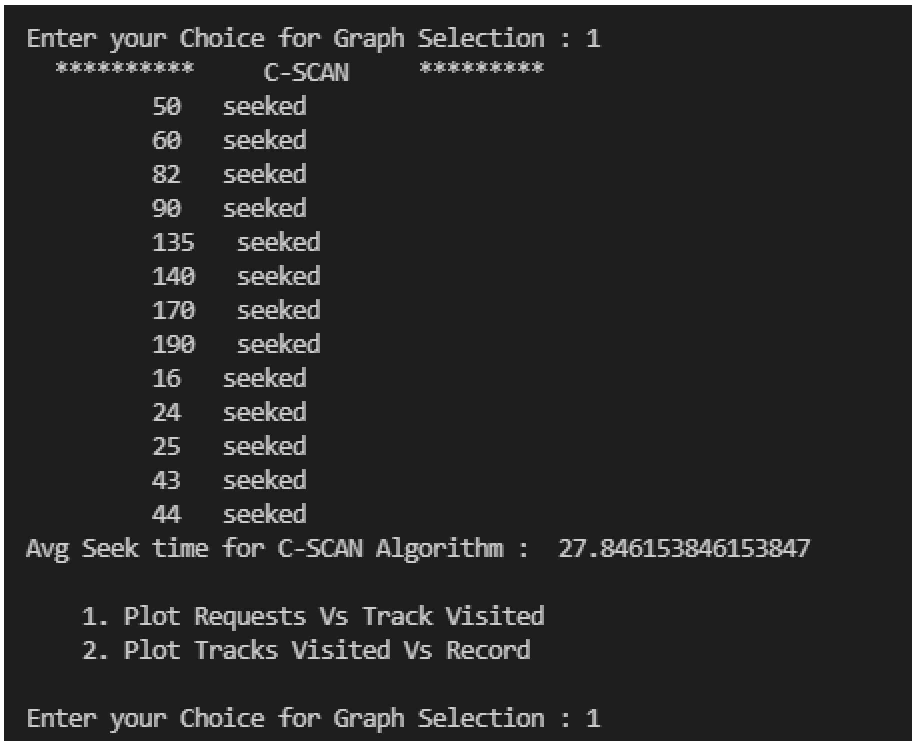
Additional Feature to plot Requests v/s Tracks



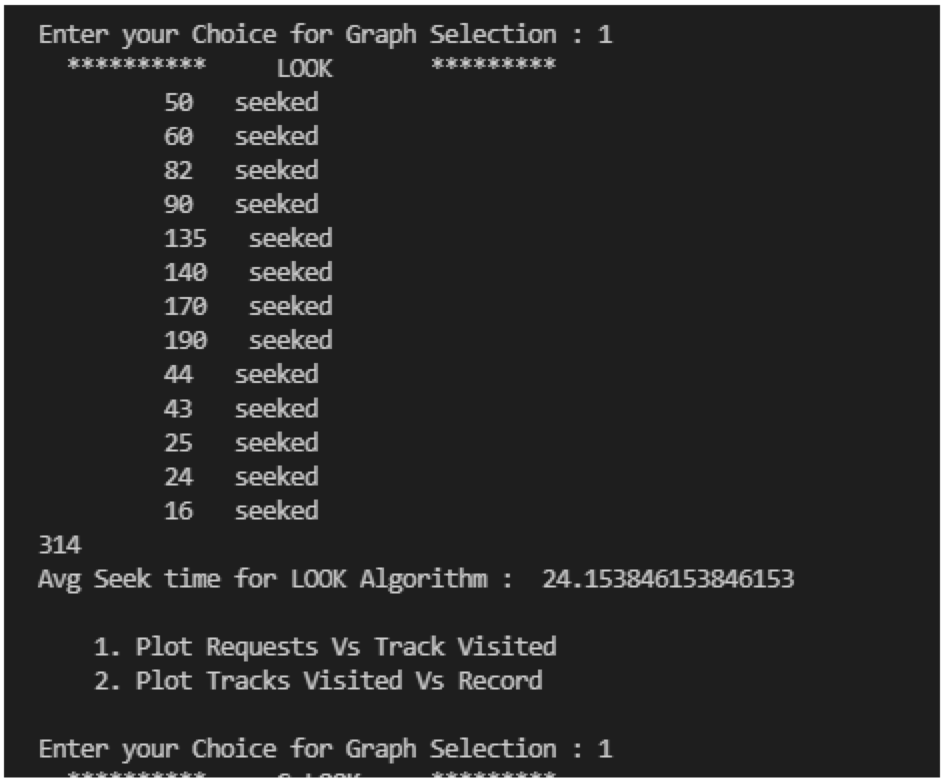

```
    **********      Shortest Seek Time First        *********
            50    Seeked
            44    Seeked
            43    Seeked
            60    Seeked
            82    Seeked
            90    Seeked
            135    Seeked
            140    Seeked
            170    Seeked
            190    Seeked
            25    Seeked
            24    Seeked
            16    Seeked
Avg Seek time for SSTF :  25.23076923076923

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```



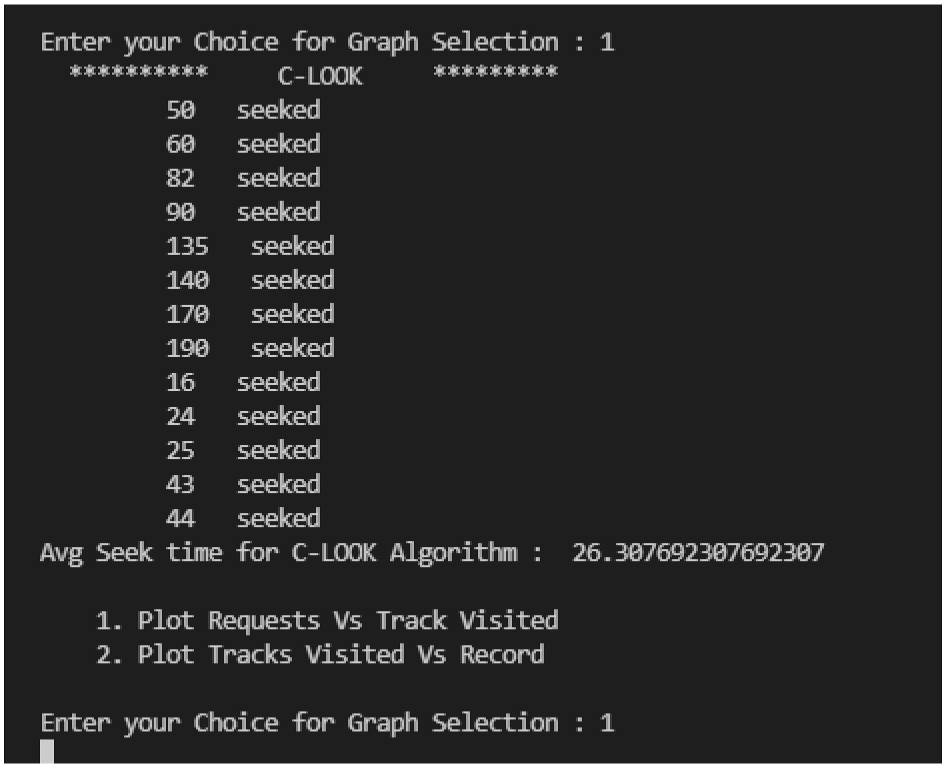
```
  **********     SCAN       *********
         50   seeked
         60   seeked
         82   seeked
         90   seeked
         135   seeked
         140   seeked
         170   seeked
         190   seeked
         44   seeked
         43   seeked
         25   seeked
         24   seeked
         16   seeked
Avg Seek time for SCAN Algorithm :   25.692307692307693

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```



```
Enter your Choice for Graph Selection : 1
  **********     C-SCAN     *********
         50   seeked
         60   seeked
         82   seeked
         90   seeked
         135   seeked
         140   seeked
         170   seeked
         190   seeked
         16   seeked
         24   seeked
         25   seeked
         43   seeked
         44   seeked
Avg Seek time for C-SCAN Algorithm :   27.846153846153847

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```



```
Enter your Choice for Graph Selection : 1
  **********     LOOK       *********
         50   seeked
         60   seeked
         82   seeked
         90   seeked
         135   seeked
         140   seeked
         170   seeked
         190   seeked
         44   seeked
         43   seeked
         25   seeked
         24   seeked
         16   seeked
314
Avg Seek time for LOOK Algorithm :  24.153846153846153

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```



```
Enter your Choice for Graph Selection : 1
  **********     C-LOOK     *********
         50   seeked
         60   seeked
         82   seeked
         90   seeked
         135   seeked
         140   seeked
         170   seeked
         190   seeked
         16   seeked
         24   seeked
         25   seeked
         43   seeked
         44   seeked
Avg Seek time for C-LOOK Algorithm :  26.307692307692307

    1. Plot Requests Vs Track Visited
    2. Plot Tracks Visited Vs Record

Enter your Choice for Graph Selection : 1
```

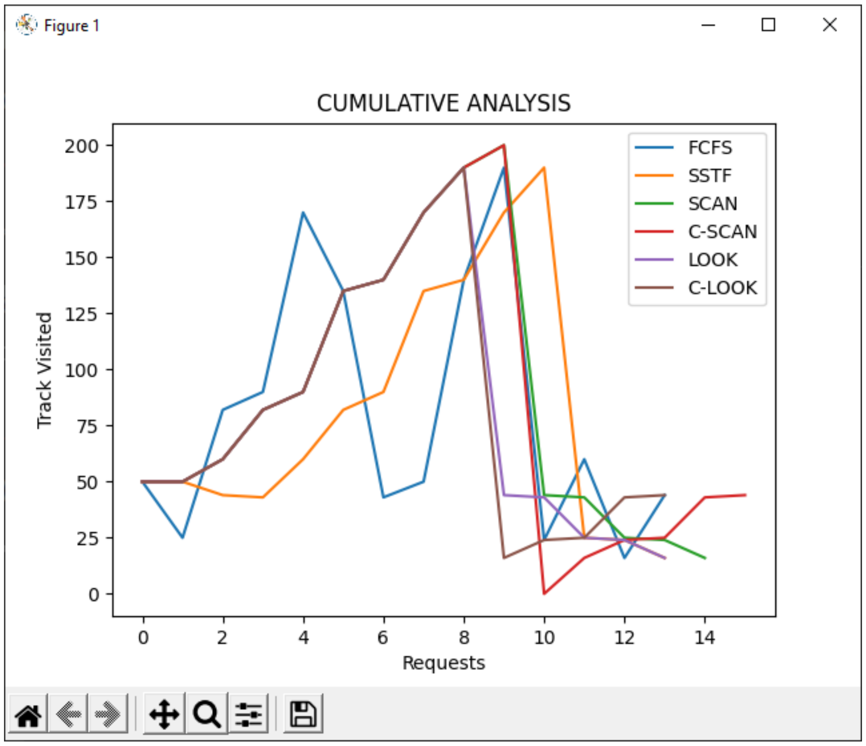


Figure 1
CUMULATIVE ANALYSIS
FCFS
SSTF
SCAN
C-SCAN
LOOK
C-LOOK
Track Visited
Requests
0
25
50
75
100
125
150
175
200
0
2
4
6
8
10
12
14

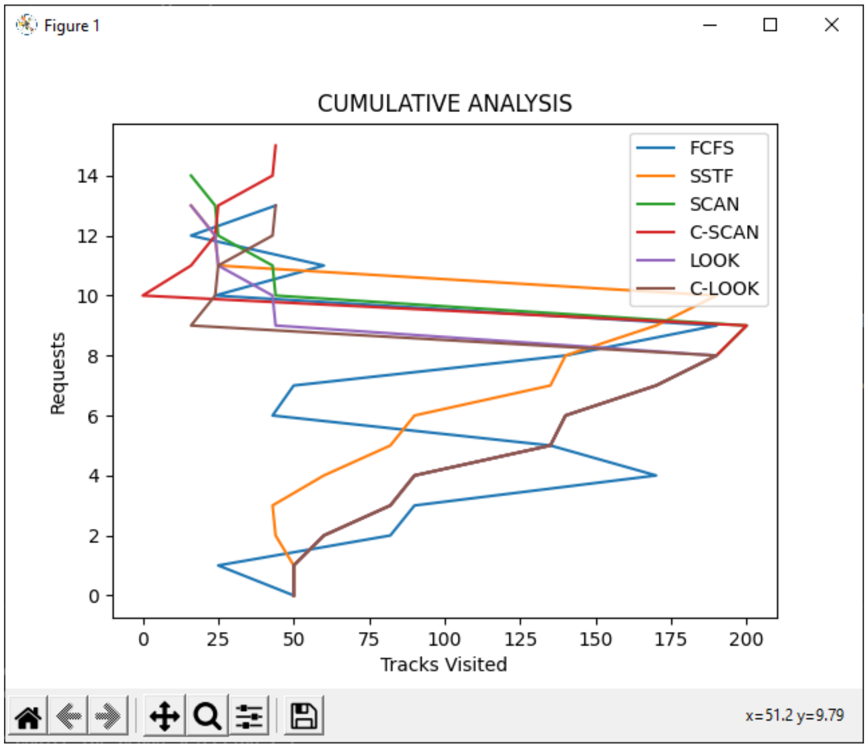


Figure 1
CUMULATIVE ANALYSIS
FCFS
SSTF
SCAN
C-SCAN
LOOK
C-LOOK
Requests
Tracks Visited
0
2
4
6
8
10
12
14
0
25
50
75
100
125
150
175
200
x=51.2 y=9.79

RESULTS

Because total head movement is the criterion for analysing disc scheduling methods, we estimated the average total head movement after entering the various runs for the requests of different algorithms. After evaluating the total head movement of several algorithms, we discovered that the LOOK Scheduling Algorithm has the least average head movement compared to the others listed above.

CONCLUSION

The current research examines six-disk scheduling techniques, namely First Come, First Serve, Scan, Look, C-Scan, and C-Look, and Scan, Look, C-Scan, and C-Look. Python was used to carry out the implementation. Depending on the system's load, different disk scheduling algorithms might be utilised. The amount and types of queries might affect the performance. Thus it will totally depend on the type of queries or the type of workload the simulator would be provided with.

REFERENCES

- William Stallings, Operating Systems: Internals and Design Principles (India: Pearson,2009).
- Abraham Silberschatz, Peter B. Galvin and Greg Gagne, Operating System Concepts,Eight ed., UK: Wiley, 2010.
- Andrew S. Tanenbaum, Modern Operating Systems (USA: Prentice-Hall, Inc., 2001).
- John Ryan Celis et. al., “A Comprehensive Review for Disk Scheduling Algorithms,”
- https://www.gatevidyalay.com/disk-scheduling-disk-scheduling-algorithms/
- http://www.cs.iit.edu/~cs561/cs450/disksched/disksched.html
- https://www.geeksforgeeks.org/disk-scheduling-algorithms/