Advanced Operating Systems
COEN 383-01 Project-5

REPORT

GROUP - 5

(Aaron Pham, Ananya Pagadala, Ankush Rai, Harsh Jitendrakumar Rao,
Harsha Ramesh Naik, Keya Bharatbhai Patel)

Objective:

The purpose of Project 5 is to implement and analyze the performance of various disk scheduling algorithms under simulated disk I/O conditions. The goal is to compare the efficiency of each algorithm in terms of total head movement required to fulfill a queue of disk access requests.

We simulate a disk with 5000 cylinders (numbered 0–4999), starting at cylinder 2255. A fixed queue of 10 pending I/O requests is processed using six different scheduling strategies. Each team member is responsible for implementing one algorithm under a shared modular structure. The algorithms implemented include:

- First-Come First-Serve (FCFS)
- Shortest Seek Time First (SSTF)
- SCAN (Elevator Algorithm)
- LOOK
- C-SCAN

Here is a concise overview of each Disk-scheduling algorithms strategy:

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

Handles requests in the exact order they arrive. It is simple but can lead to inefficient long jumps if requests are far apart.

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

Picks the request closest to the current head position. It reduces total movement but may cause starvation for distant requests.

3. **SCAN (Elevator Algorithm):**

Moves the disk arm in one direction, servicing all requests until it reaches the end, then reverses direction. It provides better overall efficiency than FCFS.

4. **LOOK:**

Similar to SCAN, but the head only travels as far as the final request in the current direction, avoiding unnecessary movement to the disk's end.

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

Services requests in one direction only. Once it reaches the end, it jumps back to the beginning and continues in the same direction. Ensures uniform wait times.

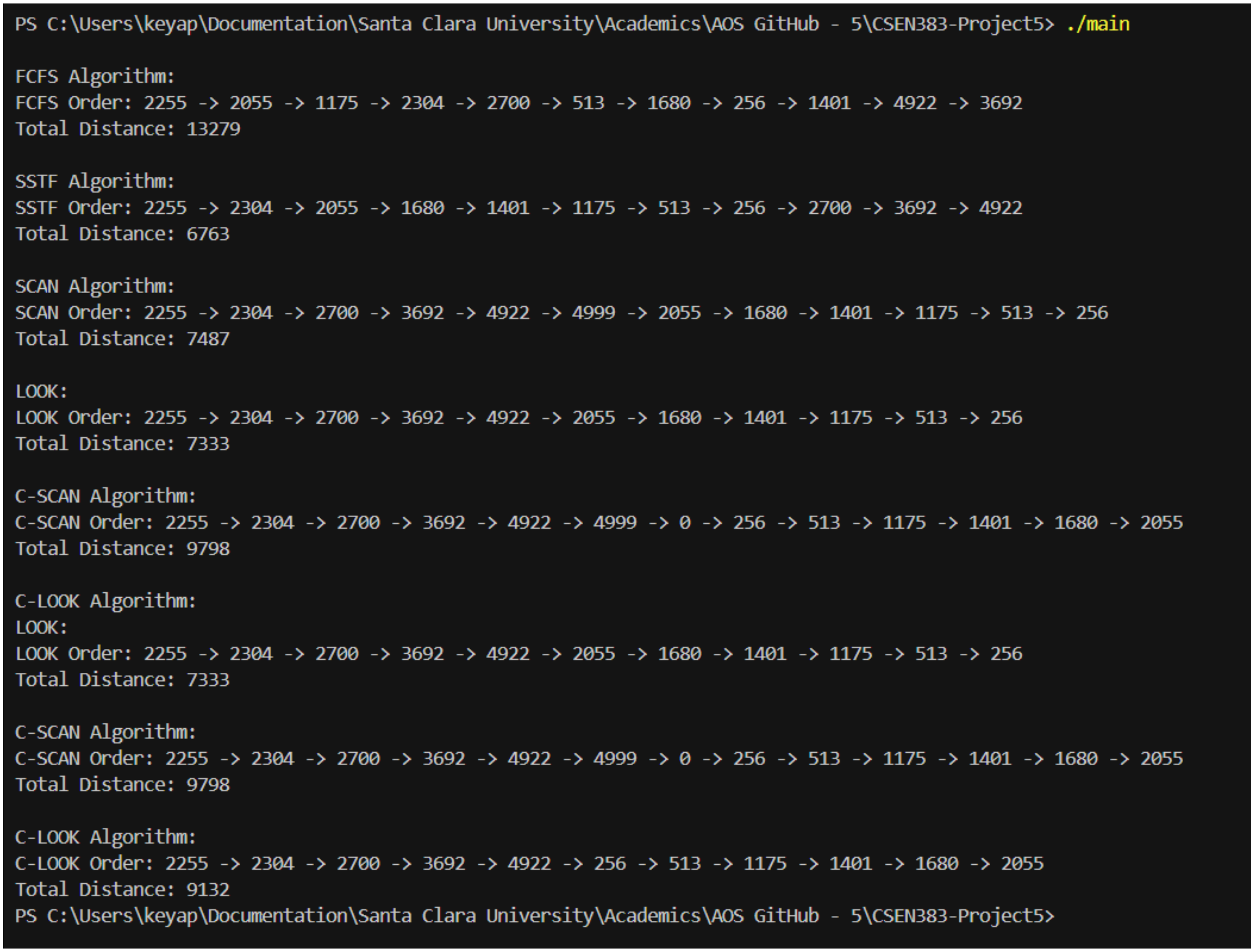
6. **C-LOOK:**

Like C-SCAN, but jumps back only to the lowest pending request rather than the physical start. It avoids extra travel when there are no requests at the disk's edges.

Output:

Total Distance Moved For Each Algorithm:

1. **FCFS - 13279 Cylinders**
2. **SSTF - 6763 Cylinders**
3. **SCAN - 7487 Cylinders**
4. **LOOK - 7333 Cylinders**
5. **C-SCAN - 9798 Cylinders**
6. **C-LOOK - 9132 Cylinders**



```
PS C:\Users\keyap\Documentation\Santa Clara University\Academics\AOS GitHub - 5\CSEN383-Project5> ./main

FCFS Algorithm:
FCFS Order: 2255 -> 2055 -> 1175 -> 2304 -> 2700 -> 513 -> 1680 -> 256 -> 1401 -> 4922 -> 3692
Total Distance: 13279

SSTF Algorithm:
SSTF Order: 2255 -> 2304 -> 2055 -> 1680 -> 1401 -> 1175 -> 513 -> 256 -> 2700 -> 3692 -> 4922
Total Distance: 6763

SCAN Algorithm:
SCAN Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 4999 -> 2055 -> 1680 -> 1401 -> 1175 -> 513 -> 256
Total Distance: 7487

LOOK:
LOOK Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 2055 -> 1680 -> 1401 -> 1175 -> 513 -> 256
Total Distance: 7333

C-SCAN Algorithm:
C-SCAN Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 4999 -> 0 -> 256 -> 513 -> 1175 -> 1401 -> 1680 -> 2055
Total Distance: 9798

C-LOOK Algorithm:
LOOK:
LOOK Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 2055 -> 1680 -> 1401 -> 1175 -> 513 -> 256
Total Distance: 7333

C-SCAN Algorithm:
C-SCAN Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 4999 -> 0 -> 256 -> 513 -> 1175 -> 1401 -> 1680 -> 2055
Total Distance: 9798

C-LOOK Algorithm:
C-LOOK Order: 2255 -> 2304 -> 2700 -> 3692 -> 4922 -> 256 -> 513 -> 1175 -> 1401 -> 1680 -> 2055
Total Distance: 9132
PS C:\Users\keyap\Documentation\Santa Clara University\Academics\AOS GitHub - 5\CSEN383-Project5>
```

Conclusion:

The analysis of various Disk Scheduling Algorithms—FCFS, SSTF, SCAN, LOOK, C-SCAN, and C-LOOK—demonstrates significant variations in efficiency based on the total seek time. FCFS, being the simplest, results in the highest seek time due to its non-optimized approach, making it inefficient for real-world applications. SSTF significantly reduces seek time by serving the closest request first but suffers from starvation issues. SCAN and LOOK improve upon SSTF by ensuring fairness and reducing unnecessary movements, though they still involve significant seek distances.

Among all algorithms, C-SCAN proves to be the most efficient, ensuring a uniform wait time while minimizing the total seek distance. C-LOOK further optimizes this by avoiding redundant movement to the disk edges. Overall, the choice of the scheduling algorithm depends on system needs: C-SCAN or LOOK are ideal for balancing efficiency and fairness, while SSTF is suitable for minimizing seek time but may not be fair to all requests.