



UNIVERSITY OF ASIA PACIFIC

Course Title: Operating System Lab

Course Code: CSE 406

Date of Submission: 22th August,2025

Submitted by:

Name: Md.Sarjil Hasan

Reg: 22101168

Sec : D

Submitted to:

Atia Rahman Orthi

Lecturer

University of Asia Pacific

Lab 6: SSTF Disk Scheduling Algorithm.

Problem Statement:

In a multiprogramming environment, several processes often request disk I/O operations at the same time. Since the disk arm can serve only one request at a time, an efficient disk scheduling algorithm is required to decide the sequence of servicing requests. The **Shortest Seek Time First (SSTF)** disk scheduling algorithm is designed to minimize the total head movement by always choosing the request closest to the current head position.

Objective:

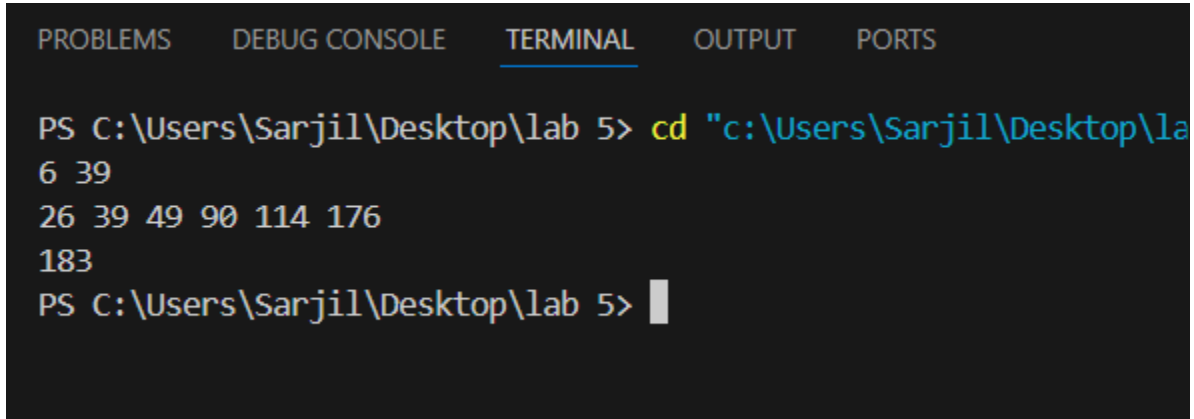
The main objectives of implementing SSTF disk scheduling are:

1. To study how the disk head movement is optimized using SSTF.
2. To calculate performance measures like **total head movement** and **average seek time**.
3. To compare its performance against simple algorithms like FCFS.
4. To understand the advantages and limitations of SSTF scheduling.

Code:

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  using namespace std;
5  int main()
6  {
7      int n, head;
8      cin >> n >> head;
9      vector<int> a(n);
10     vector<bool> visited(n);
11     int total_movement = 0;
12     int current = head;
13     for(int i=0;i<n;i++){
14         cin>>a[i];
15     }
16     for (int i = 0; i < n; i++)
17     {
18         int idx = -1;
19         int min_dist = INT32_MAX;
20         for (int j = 0; j < n; j++)
21         {
22             if (!visited[j])
23             {
24                 int dist = abs(current - a[j]);
25                 if (dist < min_dist)
26                 {
27                     min_dist = dist;
28                     idx = j;
29                 }
30             }
31         }
32         visited[idx] = true;
33         total_movement+=min_dist;
34         current=a[idx];
35     }
36     cout << total_movement << endl;
37     return 0;
38 }
```

Output:

A screenshot of a terminal window with a dark background. At the top, there are five tabs: 'PROBLEMS', 'DEBUG CONSOLE', 'TERMINAL' (which is selected and underlined), 'OUTPUT', and 'PORTS'. The terminal shows a command prompt 'PS C:\Users\Sarjil\Desktop\lab 5>' followed by a 'cd' command. Below this, several numbers are displayed on separate lines: '6 39', '26 39 49 90 114 176', and '183'. The prompt 'PS C:\Users\Sarjil\Desktop\lab 5>' is shown again at the bottom with a cursor.

Discussion:

The Shortest Seek Time First (SSTF) Algorithm is a disk scheduling technique that selects the pending request closest to the current position of the disk head. This reduces unnecessary movement of the disk arm and improves average response time compared to FCFS.

- If the current head is at position H , SSTF checks all pending requests and chooses the one with the minimum seek distance $|H - \text{request}|$.
- After servicing the chosen request, the head moves to that position, and the process repeats until all requests are served.

This approach works similarly to the Shortest Job Next (SJN) CPU scheduling algorithm, but applied to disk I/O requests. While SSTF reduces average seek time significantly, it has one major drawback: starvation. A request that is far away from the current head position may remain unserved for a long time if nearer requests keep arriving.

Thus, SSTF is a good balance between simplicity and performance, but not ideal in systems where fairness is critical.

Advantages:

1. **Improved performance** – Reduces total head movement compared to FCFS.
2. **Better average seek time** – Services closer requests first, minimizing delays.
3. **Efficient in moderate workloads** – Works well when requests are not extremely scattered.

Disadvantages

1. **Starvation problem** – Requests far from the current head may suffer indefinite waiting.
2. **Not fair** – Unlike FCFS, it does not guarantee equal service to all requests.
3. **More complex** – Requires searching the queue for the nearest request each time.

Conclusion:

The SSTF disk scheduling algorithm provides a more efficient solution than FCFS by minimizing head movement and improving average response time. However, it can lead to starvation for distant requests if closer ones keep arriving. In practice, SSTF is useful for improving disk performance, but in systems where fairness and starvation avoidance are critical, advanced algorithms like **SCAN**, **C-SCAN**, or **LOOK** are often preferred.

GIT-HUB Link:

<https://github.com/Sarjil-SarZzz/CSE406-LAB.git>