Major Half Served First (MHSF) Disk Scheduling Algorithm

Manish Kumar Mishra
Department of Information Technology
College of Computing and Informatics
Haramaya University, Dire Dawa, Ethiopia

ABSTRACT
I/O performance has been improved by proper scheduling of disk accesses since the time movable head disk came into existence. Disk scheduling is the process of carefully examining the pending requests to determine the most efficient way to service the pending requests. Scheduling algorithms generally concentrate on reducing seek times for a set of requests, because seek times tend to be an order of magnitude greater than latency times. Some important scheduling algorithms are First-Come-First-Served (FCFS), Shortest Seek Time First (SSTF), SCAN, Circular Scan (C-SCAN) and LOOK. This paper proposes a new disk scheduling algorithm called Major Half Served First (MHSF). Simulation results show that using MHSF the service is fast and seek time has been reduced drastically.

Keywords
Disk Scheduling, Seek Time, Average Seek Time, FCFS, SSTF, MHSF.

1. INTRODUCTION
In multiprogrammed computing systems, inefficiency is often caused by improper use of rotational storage devices such as disk. In this type of system, many processes may be generating requests for reading and writing disk records. Sometimes these processes make requests faster than they can be serviced by the moving head-disks, as a result waiting lines or queues build up for each device (H. M. Deitel, 2002). Which process should be selected next for service, is an important question, because it affects the effectiveness of the service. The main aim of the disk scheduling algorithms is to reduce or minimize the seek time for a set of requests (Sourav et al., 2012). The disk performance can be optimized by installing a magnetic disk that can result in high transfer rates. Magnetic disk is a collection of platters. Information is stored by recording it magnetically on the platters. A read-write disk head is located on top of each surface of every platter. The heads are attached to a disk arm that moves all the heads as a unit. The surface of a platter is logically divided into circular tracks, which are subdivided into sectors (A. Silberschatz et al., 2005). A cylinder is made up of set of tracks that are at one arm position. Disks are currently four orders of magnitude slower than main memory, so many researches are going on to enhance the efficiency of disks (William, 2007). Scheduling algorithms for moving head-disks have been studied for many years, but which algorithm is "best" is still an open question (Robert and Stephen, 1987). Most scheduling algorithms in use today are variations of a few central themes. By reducing the average seek time we can improve the performance of disk I/O operation. This study proposed a new algorithm, Major Half Served First (MHSF), which is an improvement of SSTF. MHSF takes less average seek time as compare to SSTF and FCFS disk scheduling algorithms.

1.1 Disk performance parameters
The disk I/O operations mainly depend on the computer system, the operating system, and the nature of the I/O channel and disk controller hardware (C. Staelin et al., 2009). The time taken to position the disk arm at the desired cylinder is called the Seek Time, and the time for the desired sector to rotate to the disk head is called the Rotational Latency. The sum of seek time and rotational latency is known as Access Time. The transfer time mainly depends on the rotational speed of the disk. The total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer is called the disk Bandwidth (A. Silberschatz et al., 2005). These are the disk performance parameters and they can be improved by scheduling the servicing of disk I/O requests in a good order.

1.2 Disk scheduling algorithms
Disk scheduling algorithms are used to allocate the services to the I/O requests on the disk. Some important scheduling algorithms are First-Come-First-Served (FCFS), Shortest Seek Time First (SSTF), SCAN, Circular Scan (C-SCAN) and LOOK. FCFS is the simplest form of disk scheduling algorithm. In this scheduling, I/O requests are served as per their arrival. The request that arrive first, is served first so the name First-Come-First-Served. In SSTF algorithm, the request with the minimum seek time from the current head position is served first. In this algorithm, I/O requests at the edges of the disk surface may get starved (A. L. N. Reddy et al., 2005). SSTF gives substantial improvement over FCFS. In SCAN algorithm, the disk arm starts from one end of the disk and moves to the other end of the disk. While moving from one end to the other end of the disk, it serves the requests as it reaches each cylinder. When it reaches to other end, the direction of head movement is reversed. SCAN gives better performance than FCFS and SSTF. In C-SCAN, the disk head moves from one end to the other end of the disk, serving the request along the way. When the disk head reaches to the other end, it immediately returns back to the beginning of the disk. In return trip, it does not serve any request. The waiting time increases in C-SCAN (Sourav et al., 2012). In LOOK algorithm, the arm goes only as far as the final request in each direction (A. Silberschatz et al., 2005). The direction reverses immediately, without going all the way to the end of the disk.

1.3 Related work done
In the recent years many researches has been done for enhancing the disk performance. (Manish, 2012) proposed an improvement in existing FCFS disk scheduling algorithm which works similar to FCFS but with a small improvement. IFCFS move the disk head with the intention to serve the first I/O request. On the way going to serve the first request, if there is any request waiting from the current disk head position to the first request, will be served. (Z. Dimitrijevic et al., 2005) have presented Semi-preemptible I/O, which divides disk I/O requests into small temporal units of disk commands to improve the preemptibility of disk access. (Cheng - Han et al., 2008) propose a novel real-time disk-scheduling algorithm called WRR - SCAN (Weighted-Round-Robin-SCAN) to provide quality guarantees for all in-service streams encoded at variable bit rates and bounded response

times for aperiodic jobs. (B. L. Worthington et al., 1994) examined theimpact of complex logical-to-physical mappings and large prefetching caches on scheduling effectiveness. (A. Muqaddas et al., 2009) made a simulator (Disksims). (W. A. Burkhard and J. D. Palmer, 2002) reduced the required flash memory by a factor of more than thirty there by reducing the manufacturing cost per drive.

2. MAJOR HALF SERVEDD FIRST (MHSF) ALGORITHM

My proposed MHSF algorithm is an improvement of SSTF disk scheduling algorithm. The aim of MHSF is to improve the disk performance by reducing average seek time. The MHSF disk scheduling algorithm checks number of requests on lower half area and upper half area from the present disk head position. If number of requests on lower half area is more than the number of requests on upper half area then first it serves lower half area requests and then it serves upper half area requests. If number of requests on upper half area is more than the number of requests on lower half area then first it serves upper half area requests and then it serves lower half area requests. If number of requests on lower half area and upper half area are equal then MHSF checks the present disk head position. If disk head is in lower half area then first it serves lower half area requests then it serves upper half area requests. If disk head is in upper half area then first it serves upper half area requests then it serves lower half area requests. MHSF selects either lower half requests or upper half requests depend on the number of requests pending on both areas. MHSF serves the pending requests of selected area using SSTF algorithm. Means the request that is closer to the present disk position will be served first in selected area.
Following is the proposed MHSF disk scheduling algorithm

Step 1. START
Step 2. Make a queue of the I/O requests say REQUEST.
Step 3. Check number of requests on lower half area and upper half area from the present disk head position.
Step 4. IF number of requests on lower half area is more than the number of requests on upper half area THEN first serve lower half area requests and then serve upper half area requests ELSEIF number of requests on upper half area is more than the number of requests on lower half area THEN first serve upper half area requests and then serve lower half area requests
Step 5. IF number of requests on lower half area and number of requests on upper half area are equal THEN check the present disk head position. IF disk head is in lower half area then first serve lower half area requests then serve upper half area requests ELSE first serve upper half area requests then serve lower half area requests
Step 6. END

3. PERFORMANCE EVALUATION

3.1 Experiments performed

For performance evaluation of my proposed MHSF algorithm, I have taken three different cases. Six, eight and ten I/O requests have been taken into consideration in case 1, case 2 and case 3 respectively. In each case, the experimental results of proposed MHSF algorithm have been compared with SSTF and FCFS algorithms.

Case 1: The disk queue with request for I/O to blocks on cylinders 15, 50, 35, 22, 5 and 12 has been taken into consideration. MHSF checks number of requests on lower half and upper half from the present disk head position. If disk head is presently at cylinder 30 then lower half has 4 and upper half has 2 requests.

MHSF algorithm will serve lower half requests first and then upper half requests using SSTF since lower half requests are more than upper half requests. Disk head first move to cylinder 22 since it is closet from the present disk head position in lower half. Once disk head is at cylinder 22, the next closet request is at cylinder 15. After serving request at cylinder 15, disk head moves to cylinder 12 and then it serves the last request at cylinder 10 in lower half area. All the requests in the lower half area have been served.

Now disk head is ready to serve upper half requests. In upper half area, first request that is close to the present disk head position is at cylinder 35. Disk head moves to cylinder 35 then the next closet request is at cylinder 50. The total head movement is 70 cylinders. Using the same example request queue, the total head movement is 80 cylinders in SSTF and 102 cylinders in FCFS. Table 1 shows the comparison of result of proposed MHSF with SSTF and FCFS algorithms.

Figure 1, Figure 2 and Figure 3 shows the representation of MHSF, SSTF and FCFS respectively. Figure 4 shows the comparison of average seek time of MHSF, SSTF and FCFS.

Table 1. Comparison of MHSF, SSTF and FCFS (Case 1)

Algorithms	Total Head Movement	Average Seek Time
MHSF	70	11.67
SSTF	80	13.33
FCFS	102	17

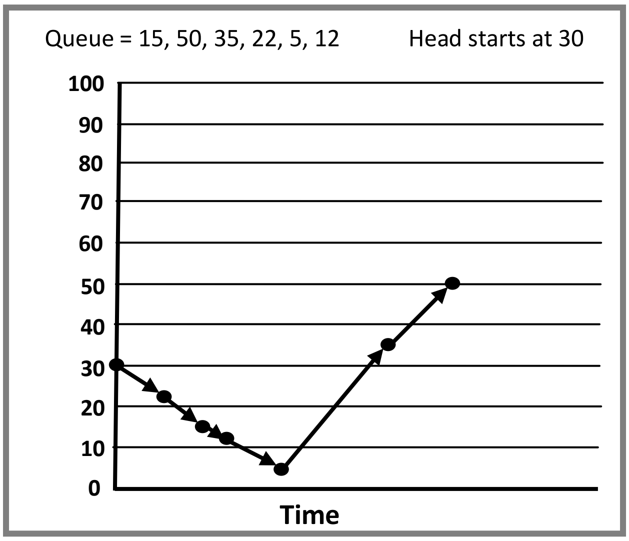

Fig 1: Representation of MHSF (Case 1)

y

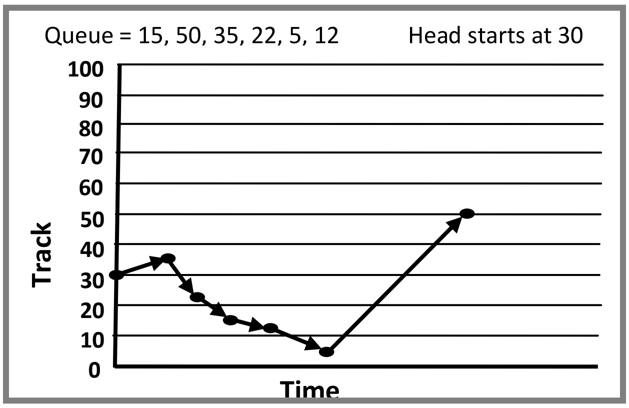

Fig 2: Representation of SSTF (Case 1)

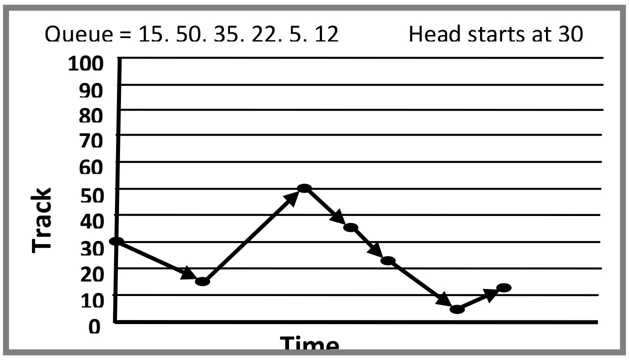

Fig 3: Representation of FCFS (Case 1)

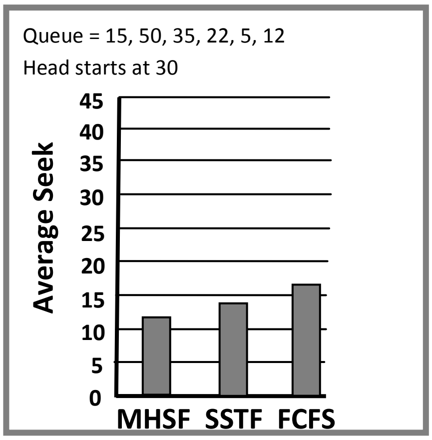

Fig 4: Comparison of Average Seek Time (Case 1)

Case 2: The disk queue with request for I/O to blocks on cylinders 50, 10, 90, 75, 100, 80, 65 and 5 has been taken into consideration. MHSF checks number of requests on lower half and upper half from the present disk head position. If disk head is presently at cylinder 55 then lower half has 3 and upper half has 5 requests. MHSF algorithm will serve upper half requests first and then lower half requests using SSTF since upper half requests are more than lower half requests. Disk head first move to cylinder 65 since it is closet from the present disk head position in upper half. Once disk head is at cylinder 65, the next closet request is at cylinder 75. After serving request at cylinder 75, disk head moves to cylinder 80, 90 and then it serves the last request at cylinder 100 in upper half area. All the requests in the upper half area have been served. Now disk head is ready to serve lower half requests. In lower half area, first request that is close to the present disk head position is at cylinder 50. Disk head moves to cylinder 50 then the next closet request is at cylinder 10 and finally request at cylinder 5 is served. The total head movement is 140 cylinders. Using the same example request queue, the total head movement is 150 cylinders in SSTF and 260 cylinders in FCFS. Table 2 shows the comparison of result of proposed MHSF with SSTF and FCFS algorithms.

Figure 5, Figure 6 and Figure 7 shows the representation of MHSF, SSTF and FCFS respectively. Figure 8 shows the comparison of average seek time of MHSF, SSTF and FCFS.

Table 2. Comparison of MHSF, SSTF and FCFS (Case 2)

Algorithms	Total Head Movement	Average Seek Time
MHSF	140	17.50
SSTF	150	18.75
FCFS	260	32.50

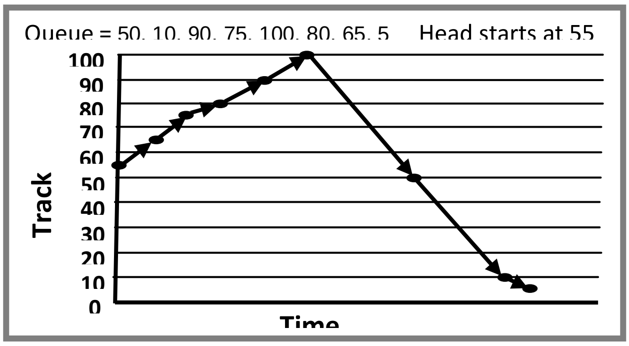

Fig 5: Representation of MHSF (Case 2)

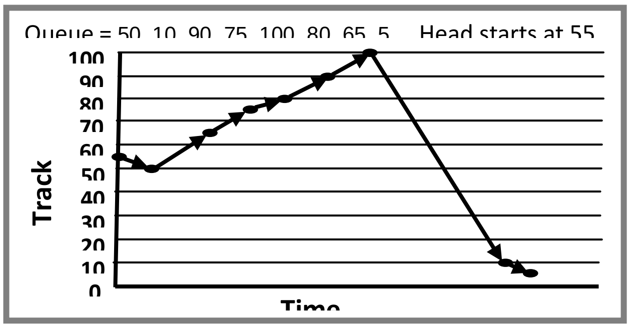

Fig 6: Representation of SSTF (Case 2)

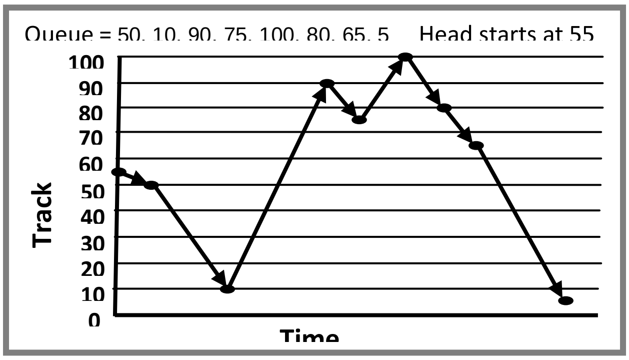

Fig 7: Representation of FCFS (Case 2)

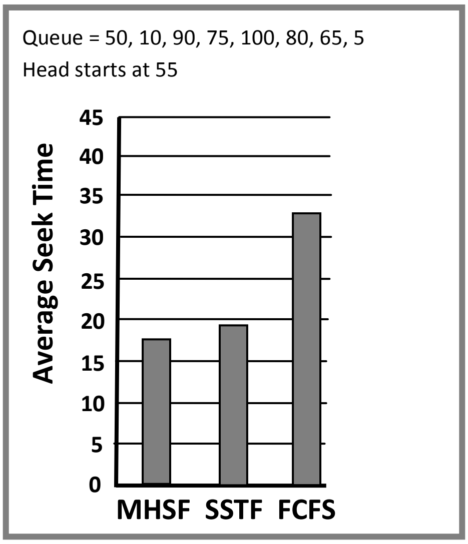

Fig 8: Comparison of Average Seek Time (Case 2)

Case 3: The disk queue with request for I/O to blocks on cylinders 20, 30, 5, 95, 85, 55, 90, 100, 25 and 15 has been taken into consideration. MHSF checks number of requests on lower half and upper half from the present disk head position. If disk head is presently at cylinder 45 then lower half has 5 and upper half has also 5 requests. Since lower half and upper half have same number of requests, the disk head position will be check against minimum and maximum track numbers. The minimum track number is 0 and the maximum track number is 100 on each platter. Since disk head is presently in the lower half of the track numbers available, MHSF algorithm selects lower half requests. Disk head first move to cylinder 30 since it is closet from the present disk head position in lower half. Once disk head is at cylinder 30, the next closet request is at cylinder 25. After serving request at cylinder 25, disk head moves to cylinder 20, 15 and then it serves the last request at cylinder 5 in lower half area. All the requests in the lower half area have been served. Now disk head is ready to serve upper half requests. In upper half area, first request that is close to the present disk head position is at cylinder 55. Disk head moves to cylinder 55 then the next closet request is at cylinder 85. After serving request at cylinder 85, disk head moves to cylinder 90, 95 and finally request at cylinder 100 is served. The total head movement is 135 cylinders. Using the same example request queue, the total head movement is 155 cylinders in SSTF and 320 cylinders in FCFS. Table 2 shows the comparison of result of proposed MHSF with SSTF and FCFS algorithms.

Figure 9, Figure 10 and Figure 11 shows the representation of MHSF, SSTF and FCFS respectively. Figure 12 shows the comparison of average seek time of MHSF, SSTF and FCFS.

Table 3. Comparison of MHSF, SSTF and FCFS (Case 3)

Algorithms	Total Head Movement	Average Seek Time
MHSF	135	13.5
SSTF	155	15.5
FCFS	320	32.0

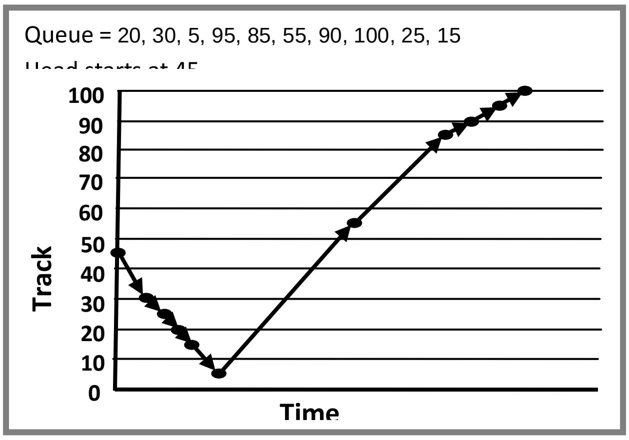

Fig 9: Representation of MHSF (Case 3)

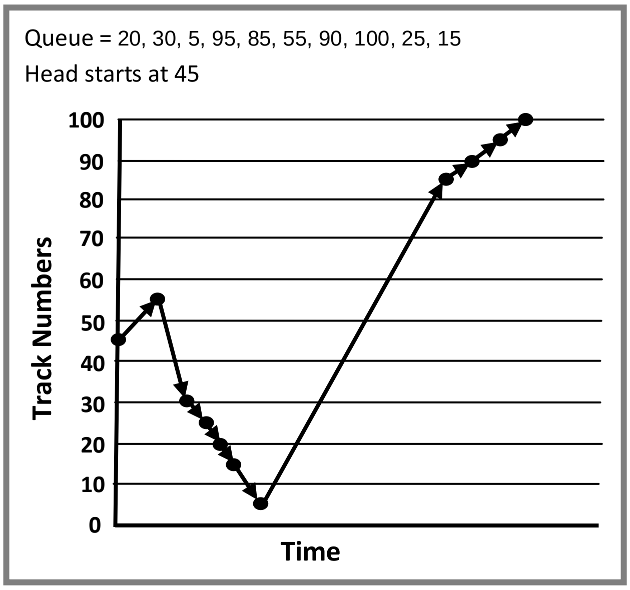

Fig 10: Representation of SSTF (Case 3)

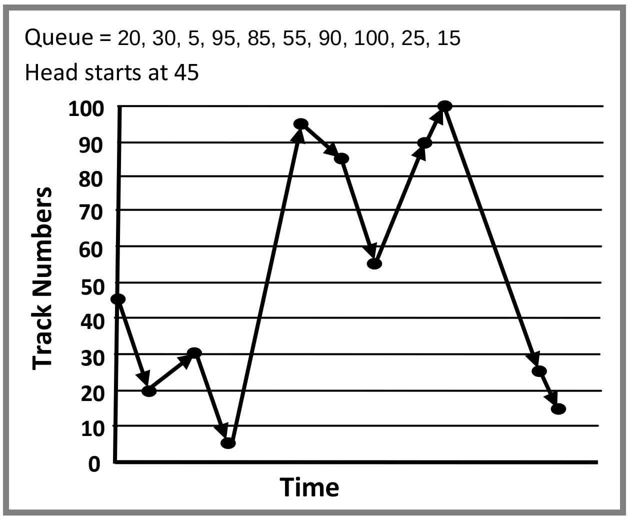

Fig 11: Representation of FCFS (Case 3)

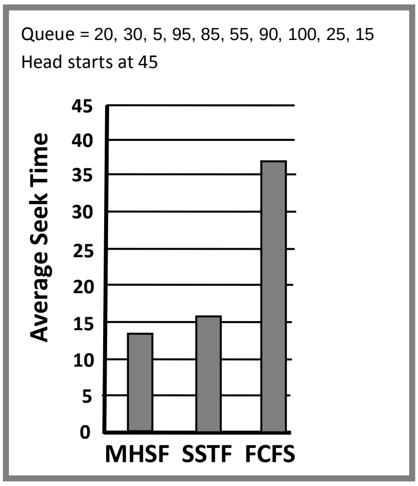

Fig 12: Comparison of Average Seek Time (Case 3)

4. CONCLUSION

This paper presented a new disk scheduling algorithm. Experimental results shows that the proposed MHSF disk scheduling algorithm is giving better performance than SSTF and FCFS disk scheduling algorithms. The average seek time has been reduced by this algorithm which increases the efficiency of the disk performance. This algorithm can be implemented to improve the performance in real time systems.

5. REFERENCES

[1] A. L. N. Reddy, Jim Wyllie, K. B. R. Wijayaratne, 2005. Disk Scheduling in a Multimedia I/O System. ACM Transactions on Multimedia Computing, Communications and Applications. Vol. 1, No. 1, 37-59.

[2] A. Muqaddas, H. Abdulsalam, A. Salman, 2009. S-LOOK: A Preemptive Disk Scheduling Algorithm for Offline and Online Environments. CSIT, Lviv, Ukraine. 15-17th Oct, 1-4.

[3] A. Silberschatz, P. B. Galvin, G. Gagne, 2005. Operating System Concepts, seventh ed. John Wiley and Sons Inc, New Delhi.

[4] B. L. Worthington, G. R. Ganger, Y. N. Patt, 1994. Scheduling Algorithms for Modern Disk Drives. Appeared in the Proceedings of the ACM Sigmetrics Conference. 241-251.

[5] C. Staelin, G. Amir, D. B. Ovadia, R. Dagan, M. Melamed, D. Staas, 2009. Real-time disk scheduling algorithm allowing concurrent I/O requests. HP Laboratories. HPL-344.

[6] C. Tsai, T. Huang, E. Chu, C. Wei, Y. Tsai, 2008. An Efficient Real-Time Disk-Scheduling Framework with Adaptive Quality Guarantee. IEEE Transactions on computers. Vol. 57, No. 5.

[7] H. M. Deitel, 2002. Operating Systems, second ed. Pearson Education Pte. Ltd., New Delhi.

[8] Manish Kumar Mishra, 2012. An Improved FCFS (IFCFS) Disk Scheduling Algorithm. International Journal of Computer Applications. Vol. 47, No. 13, 20-24.

[9] Robert Geist, Stephen Daniel, 1987. A Continuum of Disk Scheduling Algorithms. ACM Transactions on Computer Systems. Vol. 5, No. 1, 77-92.

[10] Sourav Kumar Bhoi, Sanjaya Kumar Panda, Imran Hossain Faruk, 2012. Design and Performance Evaluation of an Optimized Disk Scheduling Algorithm (ODSA). International Journal of Computer Applications. Vol. 40, No. 11, 28-35.

[11] W. A. Burkhard, J. D. Palmer, 2002. Rotational Position Optimization (RPO) Disk Scheduling. FAST, Monterey, California. 28-29th Jan.

[12] William Stallings, 2007. Operating Systems, fourth ed. Pearson Education Pte. Ltd., New Delhi.

[13] Z. Dimitrijevic, R. Rangaswami, E. Y. Chang, 2005. Support for Preemptive Disk Scheduling. IEEE Transactions on computers. Vol. 54, No. 10.