An Hybridized Disk Scheduling Algorithm (HDSA)

[1]Akanmu, T. A., [2]Aadegoke, B. O. and [1]Oladoye, S. F.

[1]Department of Computer Science, Osun State Polytechnic, Iree.
[2]Departentment of Computer Engineering, The Federal Polytechnic, Ile-Oluji.

DOI: 10.29322/IJSRP.9.03.2019.p8789
http://dx.doi.org/10.29322/IJSRP.9.03.2019.p8789

Abstract- Management of disk scheduling is a very important aspect of operating system. Performance of the disk scheduling completely depends on how efficient is the scheduling algorithm to allocate services to the requests in a better manner. Many scheduling algorithms such as FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK are developed in recent years to optimised the system disk I/O performance. By reducing the number of head movement, we can improve the performance of disk I/O operation. In this paper, a new algorithm has been proposed to reduce the number of head movement. The proposed algorithm is designed from SSTF and SCAN algorithms. The existing scheduling algorithms and our proposed algorithm are applied to two different queues of disk I/O requests. The results show that, our algorithm and LOOK give the least head movement values in the case of first queue of disk I/O requests, but it gives the least head movement value in the second queue of disk I/O of requests. Conclusively, HDSA (Hybridized Disk Scheduling Algorithm) has reduced the head movement as compared to the existing Disk Scheduling Algorithms. it therefore maximizes the throughput for modern storage devices.

Index Terms- Disk Scheduling, FCFS, SSTF, SCAN, C-SCAN, LOOK, C-LOOK, Head movement

I. Introduction

Since the invention of the movable head disk, people have improved Input/Output (I/O) performance by intelligent scheduling of disk access. Processor speed and memory capacity are increasing several times faster than disk speed. This disparity suggests that disk I/O performance will become an important bottleneck (Sandipon, Akhter, and Abul-kashem, 2013).

Scheduling is a fundamental operating system function since almost all computer resources are scheduled before use. The disk is of course, one of the computer resources. For the disk drivers, meeting this responsibility entails having fast access time and large disk bandwidth. Processor speed and memory capacity are increasing by over 40% per year. In contrast, disk speed is increasing more gradually, growing by only 7% per year (Ruemmier, and Wilkes, 1994). Since this rate is unlikely to change substantially in the near future, I/O performance may become the system bottleneck. However, despite the difficulty of improving mechanical components, we can still aim to use the disk more efficiently. Researchers have demonstrated experimentally that sophisticated disk head scheduling algorithms can deliver higher throughput (Ruemmier and Wilkes, 1994). These past researches have focused almost exclusively on two types of workloads; synthetic workload; where disk requests are randomly and uniformly distributed across the disk and more recently traces; where the requests to an actual disk are recorded and used as a testing ground for algorithms. However, for these or for general workloads, researchers have made attempts to develop algorithms with provable performance guaranteed (Sandipon, *et al*, 2013).

The most basic algorithm a scheduler can implement is First-In First-Out (FIFO) scheduling or sometimes called First Come, First Served (FCFS). FIFO has a number of positive properties; it is clearly very simple and easy to implement. SSTF works similar to SJF, which selects the disk I/O request that requires the least movement of the disk arm from its current position, regardless of direction, it reduces total seek time compared to FCFS. SCAN algorithm states that the head moves from its current position towards one direction and serves all requests in its way until it reaches the last cylinder in its direction, it switches the direction to start serving the requests along all cylinders till it reaches the other end. Circular SCAN(C-SCAN) moves inwards servicing requests until it reaches the innermost cylinder; then jumps to the outside cylinder of the disk without servicing any requests. (Ammar, Abdulsalam and Ayed Salam, year). LOOK algorithm starts at first I/O request on the disk, and moves toward the last request on the other end, servicing requests until it gets to the other extreme request on the disk where the head movement is reversed and servicing continues. Circular LOOK (C-L0OK) is a version of C-SCAN where disk arm goes only as far as the last request in each direction, then, reverses direction immediately without first going all the way to the end of the disk. (Sandipon *et al.*, 2013). The disk scheduler is responsible for dynamically ordering the pending requests. By taking into account, the various delays associated with disk accesses, a scheduler can minimise the total positioning overhead while providing reasonable response times for individual requests (Bruce, Gregory and Yale, 1994). Management of disk scheduling is a very important aspect of operating system. Performance of the disk scheduling completely depends on how efficient is the scheduling algorithm to allocate services to the disk I/O requests in a better manner. The aim of disk scheduling algorithm is to reduce the seek time for a set of request. By reducing the average seeks time, the performance of disk I/O operation can be improved. The developed Hybridized Disk Scheduling algorithm (HDSA), has a reduced average seek time.

The disk I/O operations mainly depend on the computer system, the operating system, the nature of the I/O channel and disk controller hardware. The time taken to position the head at the desired track is called Seek Time. The time taken to reach the desired sector is called Latency Time or Rotational Delay. The

sum of seek time and rotational delay is called Access Time (Silberschatz, Galvin, and Gagne, 2008).

II. Related Work Done:

Many researches have been carried out to enhance the disk performance in recent years. Sourav, Sanjaya, and Imran, (2012) designed a disk scheduling algorithm called Optimized Disk Scheduling Algorithm. This algorithm after evaluation, shows better performance than existing disk scheduling algorithms. The average seek time was improved which increased the efficiency of the disk performance (Salim, Ali and Surekha, 2013) also improved the efficiency of the disk performance by evaluating the performance of conventional disk scheduling algorithms. They compared the performances of FCFS and SSTF algorithms to Optimised Disk Scheduling Algorithm (ODSA). Result showed that average seek time required by ODSA is lesser than the FCFS and SSTF algorithms. Sandipon *et al.*, (2013) presented a new Heuristic Disk Scheduling Algorithm which gives a better result by improving the efficiency of the disk performance.

III. Hybrid Disk Scheduling Algorithm (HDSA)

The service request queue is divided into two new queues P and Q. The first queue P contains requests that are lower than the initial disk head position h; while the second queue Q contains requests that are greater than initial disk head position h. Then the absolute difference between h and the lowest track request n of disk queue P and absolute difference between h and the highest track request m of disk queue Q, are calculated as x and y respectively. If x is greater than y, then scanning begins in disk request queue Q, servicing the requests using Shortest Seek Time First (SSTF) until the disk request queue Q gets exhausted, then the head movement reversed to disk request queue P, and continuing servicing requests using SSTF in the same way as it does on the request queue Q. Otherwise, if x is less than y then disk request queue P takes the lead, and servicing operations begin immediately on the other request queue Q after P has been completed; however, SSTF is performed on both queues to determine the total seek time or total head movements and the average seek time.

Algorithm of Hybrid disk scheduling algorithm (HDSA)
HSCAN (A, T, H, THM AST)

1. THM ← 0, J ← 0, n 0 ← m 0 ←
2. for J ← 0 To N-1 do
if(A[I] > h) then
Q[m] ← A[I], m m+1 ←
Else
P[n] ← A[J], n n+1 ←
x ← h -- smallest(p)
y ← largest(Q) – h
3. if (x > y) then
call HSSTF(P, t,n,h, THM, J),
call HSSTF (Q, t, m, t[J-1], THM, J),
else call HSSTF (Q, t, m, h, THM, J),
call HSST (P, t, n, t[J – 1], THM, J)
endif
4. AST ← THM/N
5. return AST, THM
6. Exit.

IV. Results and Discussion

The developed algorithm was employed on two disk I/O requests: disk of I/O request queue (10-199): 23, 89, 132, 42, 187 with the head current position at 100 and disk of I/O request queue (10-199): 36, 80, 120, 10, 15, 40, 188, 150, 168 with the head positioned at 130.

The results of the simulation of test case 1 is shown in Figure 1 and the distance covered by the read/write head is shown in Table 1. While SCAN algorithm had six (6) transition, all other six (6) algorithms, including the developed HDSA had five

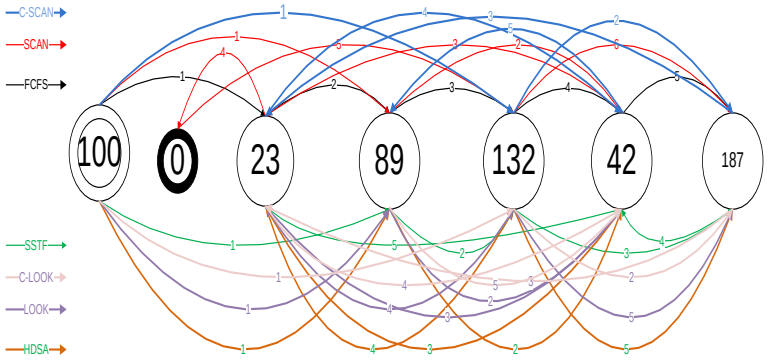

Figure 1: Transition diagram for Test Case_1

(5) transitions. Among the six algorithms, LOOK algorithm and the developed algorithm (HDSA) had a total of two hundred and forty-one (241) distance covered by the read/write head of the disk.

Table 1: Length of Transition for Test Case_1

Algorithms	Transitions						Total
	1	2	3	4	5	6	
FCFS	77	66	43	90	145		**421**
SCAN	11	47	19	23	132	55	**287**
SSTF	11	43	55	145	19		**273**
C-SCAN	11	19	9	164	55		**317**
LOOK	11	47	19	109	55		**241**
C-LOOK	32	55	164	19	47		**296**
HDSA	11	47	19	109	55		**241**

In the test case_2 as shown in Figure 2, the developed algorithm had the smallest distance coverage by the read/write head of the hard disk drive which was smaller in value than LOOK algorithm.

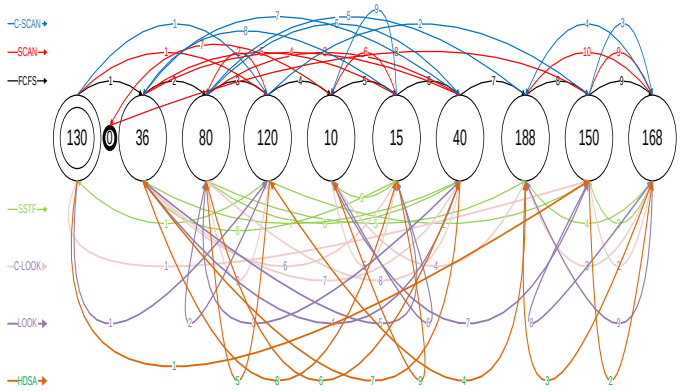

Figure 2: Transition diagram for Test Case_2

Table 2: Comparison of Length of Transition for Test Case_2

Algorithms	Transitions										Total
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**		
FCFS	94	44	40	110	5	25	148	38	18		**522**
SCAN	10	40	40	4	21	5	10	150	18	20	**318**
SSTF	10	10	18	20	88	40	4	21	5		**256**
C-SCAN	20	18	20	11	19	10	5	21	4	40	**388**
LOOK	10	40	40	4	21	5	140	18	20		**298**
C-LOOK	20	18	20	17	5	21	40	40			**342**
HDSA	20	18	20	68	40	40	4	21	5		**236**

Table 3: Summary of performance of the Algorithms

Algorithms	Test_case_1	Test_case_1
FCFS	421	582
SCAN	287	318
C-SCAN	273	388
SSTF	317	256
LOOK	241	298
C-LOOK	296	346
HDSA (Developed, 2018)	241	241

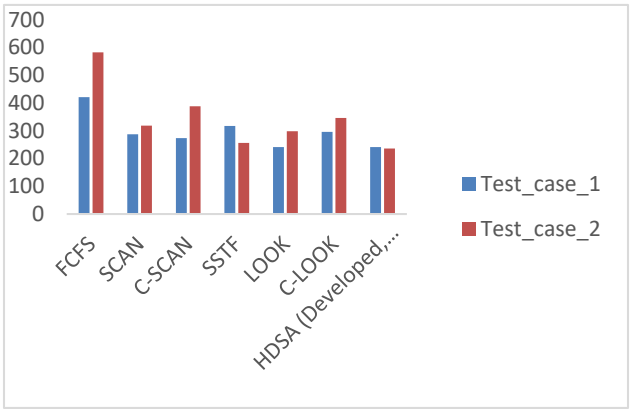

Figure 3: Performance of the algorithms

V. Conclusion

We have presented an Hybridized Disk Scheduling Algorithm (HDSA) which shows better performance than existing conventional disk scheduling algorithms such as FCFS, SSTF, SCAN, C-SCAN, LOOK and C-LOOK). The results of the experiment showed that the number of head movement of the HDSA was reduced compared to the conventional disk scheduling algorithms. This undoubtedly increase efficiency of the disk performance in computer resource management.

References

[1] Ammar Muqaddas, Hanady Abdulsalam, and Ayed Salam. S-look a preemptive disk scheduling algorithm for offline and online enviroments. Computer

[2] Engineering Department, Kuwait University, Kuwait.

[3] Bruce, L. Worthington, Gregory R. Ganger, and Yale N. Patt. Scheduling algorithms for modern disk drives. Appeared in the Proceedings of the ACM Sigmetrics Conference, May, 1994, pp.241-251.

[4] Ruemmier, C., and Wilkes, J. An introduction to disk drive modelling. Computer 27, .3(1994), 17-28.

[5] Salim, Y. A., Ali M.S. and Surekha, B. C. (2013). Performance evaluation of conventional disk scheduling algorithms. International Journal of Pure and

[6] Applied Research in Engineering and Technology, 2013, Volume 1(8): 568-575

[7] Sandipon Saha, Md Nasim Akhter, Mohammod Abul kashem. A new heuristic disk scheduling algorithm. International Journal of Scientific & Technology Research 2,1(2013), 49-53

[8] Silberschatz, A., Galvin, P. B. and Gagne, G. "Operating System Principles", 7th Edn., John Wiley and Sons, 2008, ISBN 978-81-265-0962-1.

[9] Sourav Kumar Bhoi, Sanjaya Kumar Panda & Immran Hossain Faruk (2012). Design and perfoemance evaluation of an optimized disk scheduling algorithm (ODSA). Internaltional Journal of Computer Applications (0975-8887) Volume 40- No. 11.

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

Authors

First Author – Akanmu, T. A, Department of Computer Science, Osun State Polytechnic, Iree.

Second Author – Aadegoke, B. O, Departentment of Computer Engineering, The Federal Polytechnic, Ile-Oluji.

Third Author – Oladoye, S. F, Department of Computer Science, Osun State Polytechnic, Iree.