EFFECTIVE DISK MANAGEMENT

PRESENTED BY

Arpit Jain | 2015047

Gautam Yadav| 2015093

UNDER GUIDANCE OF

Dr. Manish Kumar Bajpai

Research Papers referred

A New Optimized Real-Time Scheduling Algorithm

A New Heuristic Disk Scheduling Algorithm

An Improved FCFS (IFCFS) Disk Scheduling Algorithm

Disk Scheduling Algorithms

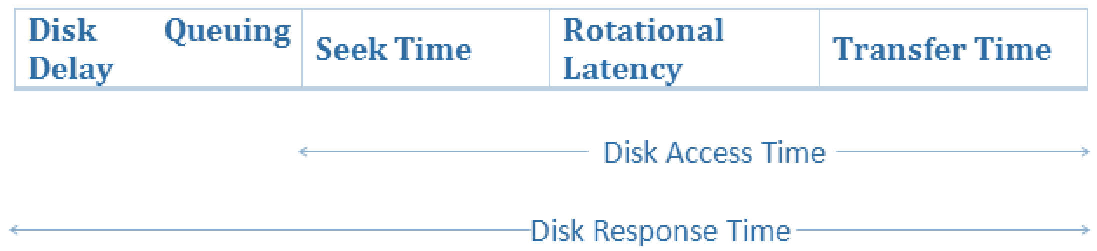
- **Disk scheduling** is done by operating systems to schedule I/O requests arriving for disk. Disk scheduling is also known as I/O scheduling.
- Disk scheduling is important because:

1. Multiple I/O requests may arrive by different processes and only one I/O request can be served at a time by disk controller. Thus other I/O requests need to wait in waiting queue and need to be scheduled.
2. Two or more request may be far from each other so can result in greater disk arm movement.
3. Hard drives are one of the slowest parts of computer system and thus need to be accessed in an efficient manner.

Some Important Terms

1. **Seek Time:** Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write. So the disk scheduling algorithm that gives minimum average seek time is better.
2. **Rotational Latency:** Rotational Latency is the time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.
3. **Transfer Time:** Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and number of bytes to be transferred.

4. **Disk Access Time:** Seek Time + Rotational Latency + Transfer Time



5. **Disk Response Time:** Response Time is the average of time spent by a request waiting to perform its I/O operation. Average Response time is the response time of the all requests. Variance Response Time is measure of how individual request are serviced with respect to average response time. So the disk scheduling algorithm that gives minimum variance response time is better.

A New Optimized Real-Time Scheduling Algorithm

Nidhi

Madan Mohan Malaviya University of Technology, Gorakhpur (UP)-273010, India

Dayashankar Singh

Madan Mohan Malaviya University of Technology, Gorakhpur (UP), India

• Abstract

In this proposed algorithm, initially the disk head is at the disk start position and has the direction towards the final disk position. It means initial head position and direction of head is always same. First we sort all the cylinders input blocks by using any sorting algorithm. Initially the head is at position 0 and sequentially moves and reached from this block to the highest input block number, servicing all the input request blocks in front of the head immediately.

• Algorithm

- Assume a[] is an array containing track numbers and x is the position of last input block.
- Inialize Head position is at 0. Assume h denotes the current head position.
- Sort input blocks of cylinder number in ascending order with the help of any sorting algorithm.
- Initially head position h is 0.
- for(i=0; i<=x; i++)
- Service the input request in front of head immediately.
- Total_head_movements = x;
- Return total_head_movements;

• Pros

1. Total number of head movements are always less than or equal to the N, where N is the position of last track on the disk.
2. If all the input blocks are concentrated near the position 0 (means in the first half of the disk), then this proposed algorithm gives best result.
3. It is very simple algorithm. Calculation of total head movements is very simple and is always equal to the x, where x is the position of last input block.

• Cons

1. Overhead time penalty to calculate and sort (Ascending order) input blocks of cylinder Numbers in advance.
2. Long Waiting Time for requests for locations just visited by the disk arm.
3. This algorithm assumes all the disk access calls to be registered before the actual processing starts. This doesn’t provide satisfactory performance when there are prompt disk access calls.

An Improved FCFS (IFCFS) Disk Scheduling Algorithm

Manish Kumar Mishra

Department of Information Technology, College of Computing and Informatics,

Haramaya University, Ethiopia

• Abstract

The improved FCFS (IFCFS) disk scheduling algorithm 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. After serving first request and the requests that were served on the way, disk head will move to the next request waiting in the queue. On the way going to serve this request, if there is any request waiting from the current disk head position to the next waiting request, will be served etc. IFCFS guarantees the performance improvement over FCFS.

• Algorithm

1. START
2. Make a queue of the I/O requests say REQUEST.
3. Do steps 4, 5 and 6 WHILE queue REQUEST becomes empty.
4. Move the disk head from the current position say K to the first request in the queue REQUEST. Serve the requests that are on the way of the disk head from the current position K to the first request in the queue REQUEST.
5. Serve the first request in the queue REQUEST.
6. Remove the requests from the queue REQUEST that are already served.
7. END

Pros

1. Improved version of FCFS.
2. Every Request gets a fair chance.
3. Average seek time has been reduced drastically.

Cons

1. Overhead time penalty to process the disk access queue at every point.
2. In the worst case, when the track positions are alternatively increasing and decreasing respectively, the time complexity reaches that of FCFS.

A New Heuristic Disk Scheduling Algorithm

Sandipon Saha, Md. Nasim Akhter, Mohammod Abul Kashem

Department of CSE, Dhaka University of Engineering & Technology

- ## Abstract

At first sorting in ascending order of all cylinders input blocks by using any sorting method. Find the distance between the smallest block number and current head position. Let it is P and again find the distance between the largest block number and current head position. Let it is Q. Sequentially move and reached head from these block to the highest block number. Else head moves sequentially from its current position to the highest block number in forward and again in backward which block is not visited. Then sequentially move and reached head from these block to the largest block number.

• Algorithm

New Heuristic Disk Scheduling Algorithm (a, n, count, h)

```
1.  // a [] is an array that contains cylinders number. N
    is the number of cylinder. Count is use
2.  // for counting head movement's. h denote the
    present head position.
3.  Sorting input blocks of cylinder number in
    ascending order by any sorting method.
4.  Input present head position h.
5.  Temp:=h;
6.  For  i :=1 to n do
    If(a[i]>=h) { Position=i; break;}
7.  Left_distance:=head-a [1];
    Right_distance:=a[n]-head;
8.  Count:=0;
9.  If (Left_distance< Right_distance)
        {

          For i:=position-1 to 1 step -1 do  {
          count:=count+Temp-a [i]; Temp:=a[i];
          }
           count :=count+[position]-a[1];
          For I: =position+1 to n do
            count: =count + a [i]-a [i-1];
      }
   Else
     {
        For i:=position to n do
        {count:=count+a[i]-head;
           head:=a[i];}
         Count:=count+a[n]-a[position-1];
       For i:=position-1 to  2 do
           Count:=count+a[i]-a[i-1];
     }
10. Return count;  // total head movement
```

• Pros

1. Average Response time reduces by a great factor. Calls are processed faster.
2. Throughput is better than other algorithms.
3. Optimum use of head movement when compared to other algorithms used.

• Cons

1. Sometime number of head movement is equal to SSTF or LOOK scheduling. Extra computation (For-Loop) in comparison to SSTF or LOOK scheduling is unneccessary.
2. When input blocks are stay in ascending order without sorting then FCFS is best. But in dynamic allocation of cylinder it is almost impossible.
3. Overhead time penalty to calculate and sort (Ascending order) input blocks of cylinder Numbers in advance.

Refrences

1. http://research.ijcaonline.org/volume93/number18/pxc3896046.pdf
2. http://www.ijstr.org/final-print/jan2013/A-New-Heuristic-Disk-Scheduling-Algorithm.pdf
3. http://research.ijcaonline.org/volume47/number13/pxc3880298.pdf