Disk Scheduling Algorithms

(CS537 – Spring 2015 :: Kanlue Zhang)

Abstract

This report starts from the structure of a disk and the hardware activities involved in the data retrieval from direct attached storage device, then introduces and compares multiple Disk Scheduling algorithms such as First Come First Serve (FCFS), Shortest Seek Time First (SSTF), SCAN, LOOK, Circular SCAN (C-SCAN) and C-LOOK Scheduling algorithm. In order to show the differences between these algorithms, serval examples are provided and concluded with analysis in the end.

1. Introduction

Since the emergence of computers, hardware technology has been evolving at a tremendous pace. This evolution includes storage technologies. The current trend in storage technology is miniaturization for portability and increased storage capacity. Due to the volatile characteristic of the CPU register, Cache, and Main Memory, the use of secondary storage devices such as Disk came into existence.

In a movable-head disk, access may take the form of a write or a read operation performed by the access arm, which holds the read/write head. Since the invention of movable head disk, the Input and Output (I/O) performance has been improved by implementing proper and intelligent scheduling of disk accesses. Disk scheduling involves a careful examination of pending requests to determine the most efficient way to service the requests. Some of Disk Scheduling algorithms are First Come First Serve (FCFS), Shortest Seek Time First (SSTF), SCAN, LOOK, C-SCAN and C-LOOK Scheduling algorithm which is further discussed in this report.

2. Disk Structure

2.1 Physical Structure

A disk is a platter, made of metal or plastic with a magnet coating on it, and in circular shape. It is possible to store information by recording it magnetically on the platters. A conducting coil, called head, which is a relatively small device, facilitates the data recording on and retrieval from the disk. In a disk system, head rotates just above both surfaces of each platter. All heads, being attached to a disk arm, move collectively as a unit. To enable a read and write operation, the platter rotates beneath the stationary head.

Data are organized on the platter in tracks, which are in the form of concentric set of rings. In media using constant linear velocity, the track densities are uniform (bits per linear inch of track). The outermost zone has about 40 percent more sectors than innermost zone. The rotation speed increases as the head moves from the outer to the inner tracks to keep the same data transfer rate. This method is also used in CD-ROM and DVD-ROM drives. In these types of media, the storage capacity of the disk is maximized by zoning application. A zone consists of adjacent cylinders having the same track densities (sector per track).

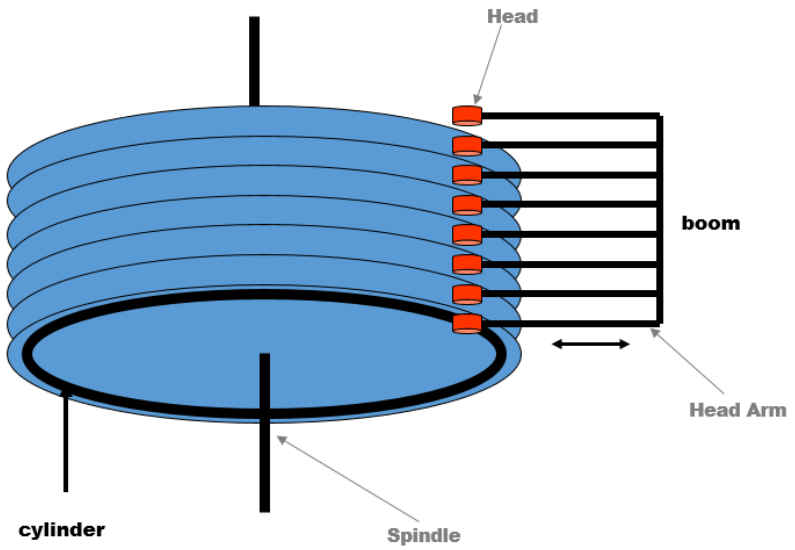

Fig. 1 Disk Mechanism

There are three elements of a disk known as cylinder, track and sector/block. Tracks have subdivisions, called sectors. Data are transferred to and from the disk in blocks, size of which are typically smaller than the capacity of the track. Block-size regions on the disk where data are recorded, are called sectors each having 512 bytes capacity for most disk drives. The request locations are defined with the physical

block addresses over these sectors. Adjacent sectors are separated by in track gaps in order to avoid imposing unreasonable precision requirements on the system.

A common disk drive has a capacity in the size of gigabytes. While the set of tracks that are at one arm position forms a cylinder, in a disk drive there may be thousands of concentric cylinders.

2.2 Working Principle

In a movable-head disk, where there is only one access arm to service all the disk tracks, the time spent by the Read and Write (R/W) head to move from one track to another is called Seek Time . This is differing from a fixed-head disk wherein there is an assigned arm for every track. Seek time can be compared to the movement of a space shuttle from one planetary orbit to another. There are two kinds of seek either inward or outward seek depending on the current location of the R/W head. Moving towards the outer portion of the disk corresponds to an outward seek while moving towards the center of the disk considered an inward seek.

Once it reaches the desired track, the disk will rotate to find the block of data to be accessed and position it before the R/W head. The time spent in moving the disk from undesired block to desired block is known as rotational delay or latency time. Rotational delay positions the tip of the R/W head at the beginning of the sector1 where the data is to be read or written.

As soon as the R/W head is positioned at the beginning of the data block, it is electrically switched on in line with the preparation for read or writes function. Once it is switched on, that's the time the read or write functions begin.

Finally, the disk will again rotate in order for the data block to get ahead of under read/write. The data is then read from or written to the disk; consequently it is called transfer time.

3. Disk Scheduling Algorithms

Disk scheduling algorithms are used to allocate the services to the I/O requests on the disk. Since seeking disk requests is time consuming, disk scheduling algorithms try to minimize this latency.

If desired disk drive or controller is available, request is served immediately. If busy, new request for service will be placed in the queue of pending requests. When one request is completed, the Operating System has to choose which pending request to service next. The OS relies on the type of algorithm it needs when dealing and choosing what particular disk request is to be processed next. The objective of using these algorithms is keeping Head movements to the amount as possible. The less the head to move, the faster the seek time will be. To see how it works, the different disk scheduling algorithms will be discussed and examples are also provided for better understanding on these different algorithms.

3.1 First Come First Serve (FCFS)

It is the simplest form of disk scheduling algorithms. The I/O requests are served or processes according to their arrival. The request arrives first will be accessed and served first. Since it follows the order of arrival, it causes the wild swings from the innermost to the outermost tracks of the disk and vice versa . The farther the location of the request being serviced by the read/write head from its current location, the higher the seek time will be.

Example:

Given the following track requests in the disk queue, compute for the Total Head Movement2 (THM) of the read/write head:

95, 180, 34, 119, 11, 123, 62, 64

Consider that the read/write head is positioned at location 50. Prior to this track location 199 was serviced. Show the total head movement for a 200 track disk (0-199).

Solution:

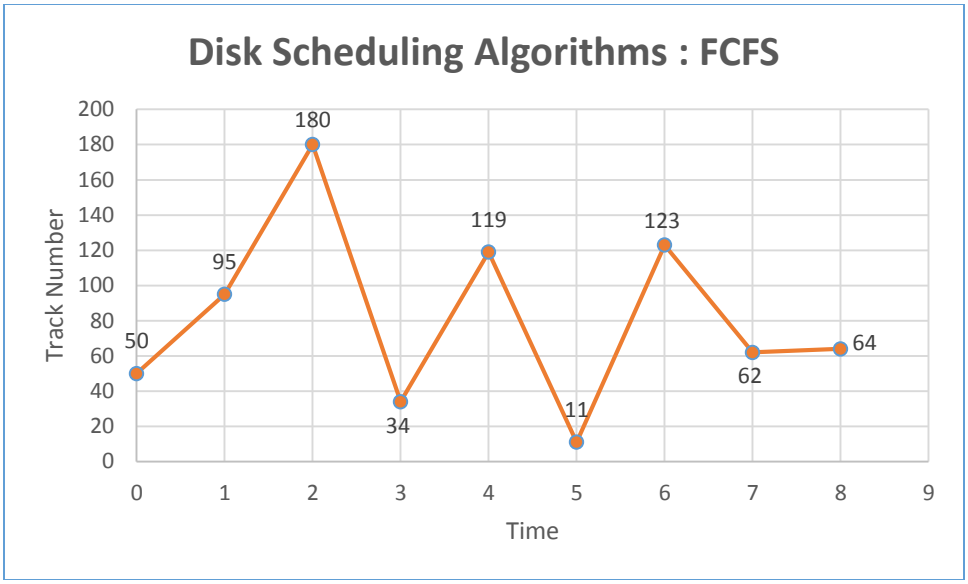

Fig. 2 FCFS Representation

Total Head Movement Computation:

(THM) = (180 - 50) + (180-34) + (119-34) + (119-11) + (123-11) + (123-62) + (64-62)

= 130 + 146 + 85 + 108 + 112 + 61 + 2

(THM) = 644 tracks

Assuming a seek rate of 5 milliseconds is given, we compute for the seek time using the formula:

Seek Time = THM * Seek rate = 644 * 5 ms

Seek Time = 3,220 ms

There are some requests that are far from the current location of the R/W head which causes the access arm to travel from innermost to the outermost tracks of the disk or vice versa.

In this example, it had a total of 644 tracks and a seek time of 3,220 milliseconds. Based on the result, this algorithm produced higher seek rate since it follows the arrival of the track requests.

3.2 Shortest Seek Time First (SSTF)

This algorithm is based on the idea that that he R/W head should proceed to the track that is closest to its current position. The process would continue until all the track requests are taken care of. Using the same sets of example in FCFS the solution are as follows:

Solution:

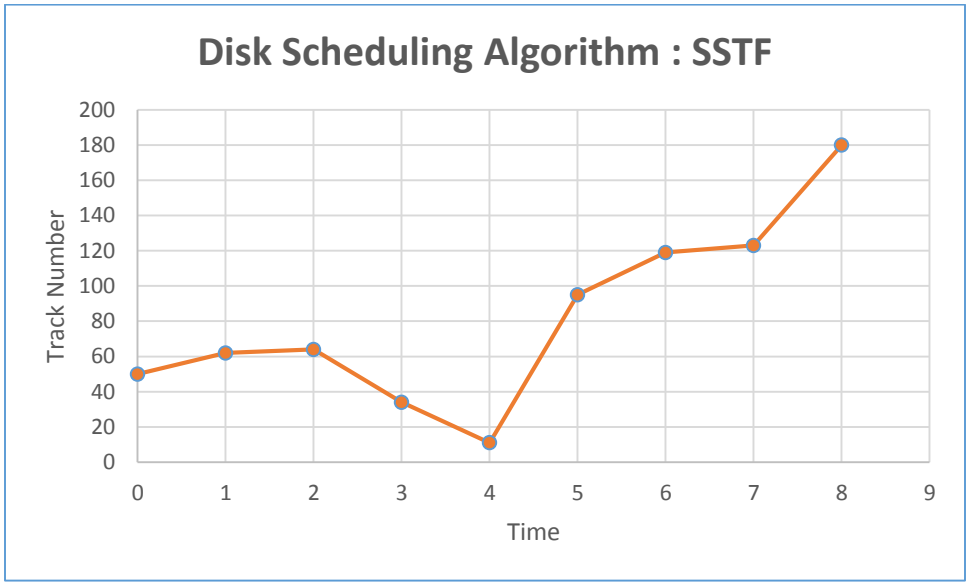

Fig. 3 SSTF Representation

(THM) = (64-50) + (64-11) + (180-11) = 14 + 53 + 169

(THM) = 236 tracks

Seek Time = THM * Seek rate = 236 * 5ms

Seek Time = 1,180 ms

In this algorithm, request is serviced according to the next shortest distance. Starting at 50, the next shortest distance would be 62 instead of 34 since it is only 12 tracks away from 62 and 16 tracks away from 34. The process would continue up to the last track request. There are a total of 236 tracks and a seek time of 1,180 ms, which seems to be a better service compared with FCFS which there is a chance that starvation3 would take place. The reason for this is if there were lots of requests closed to each other, the other requests will never be handled since the distance will always be greater.

3.3 SCAN Scheduling Algorithm

This algorithm is performed by moving the R/W head back-and-forth to the innermost and outermost track. As it scans the tracks from end to end, it process all the requests found in the direction it is headed. This will ensure that all track requests, whether in the outermost, middle or innermost location, will be traversed by the access arm thereby finding all the requests. This is also known as the Elevator algorithm. Using the same sets of example in FCFS the solution are as follows:

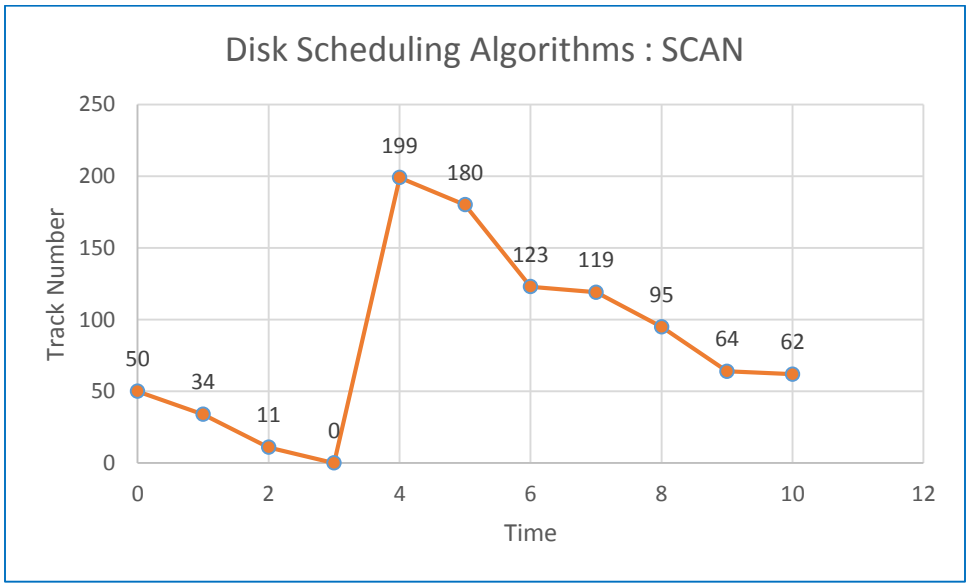

Fig. 4 SCAN Representation

(THM) = (50-0) + (180-0) = 50 + 180

(THM) = 230

Seek Time = THM * Seek rate = 230 * 5ms

Seek Time = 1,150 ms

This algorithm works like an elevator does. In the algorithm example, it scans down towards the nearest end and when it reached the bottom it scans up servicing the requests that it did not get going down. If a request comes in after it has been scanned, it will not be serviced until the process comes back down

or moves back up. This process moved a total of 230 tracks and a seek time of 1,150. This is optimal than the previous algorithm.

3.4 LOOK Scheduling Algorithm

This algorithm is similar to SCAN algorithm except for the end-to-end reach of each sweep. The R/W head is only tasked to go the farthest location in need of servicing. This is also a directional algorithm, as soon as it is done with the last request in one direction it then sweeps in the other direction. Using the same sets of example in FCFS the solution are as follows:

Solution:

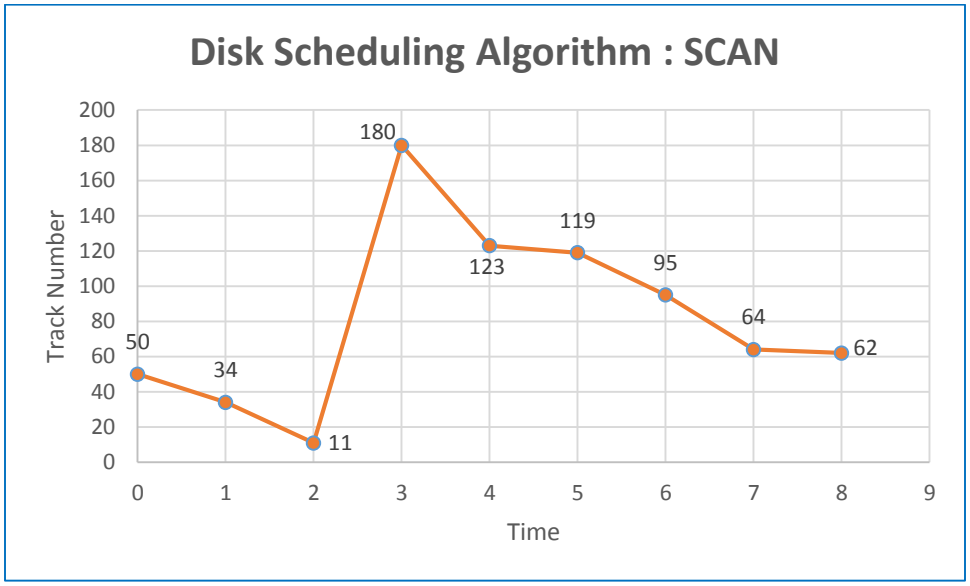

Fig. 5 LOOK Representation

(THM) = (50-11) + (180-11) = 39 + 169

(THM) = 208 tracks

Seek Time = THM * Seek rate = 208 * 5ms

Seek Time = 1,040 ms

This algorithm has a result of 208 tracks and a seek rate of 1,040 milliseconds. This algorithm is better than the previous algorithm.

3.5 Circular SCAN (C-SCAN) Algorithm

This algorithm is a modified version of the SCAN algorithm. C-SCAN sweeps the disk from end-to-end, but as soon it reaches one of the end tracks it then moves to the other end track without servicing any requesting location. As soon as it reaches the other end track it then starts servicing and grants requests headed to its direction. This algorithm improves the unfair situation of the end tracks against the middle tracks. Using the same sets of example in FCFS the solution are as follows:

Solution:

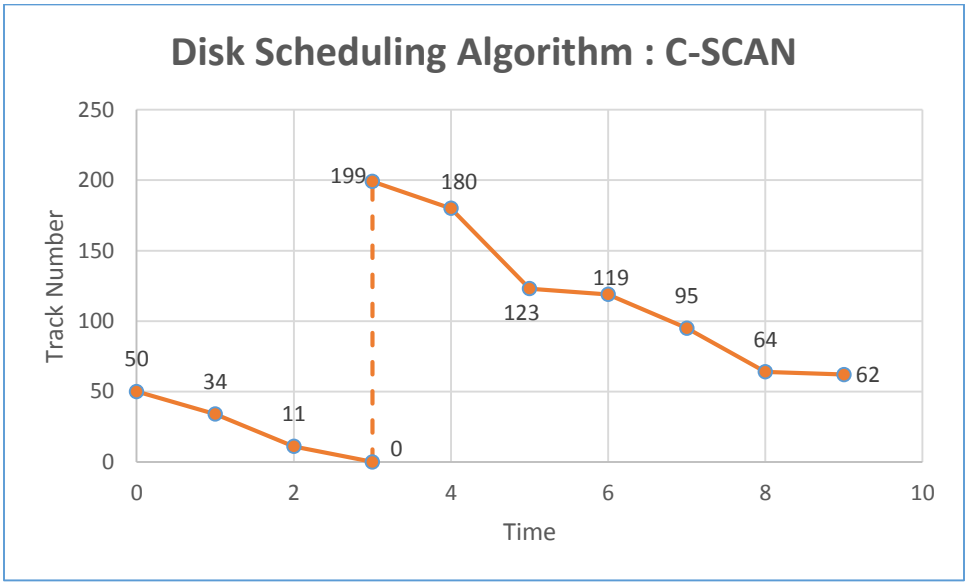

Fig. 6 C-SCAN Representation

Notice that in this example an alpha3 symbol (α) was used to represent the dash line. This return sweeps is sometimes given a numerical value which is included in the computation of the THM. As analogy, this can be compared with the carriage return lever of a typewriter. Once it is pulled to the right most direction, it resets the typing point to the leftmost margin of the report. A typist is not supposed to type during the movement of the carriage return lever because the line spacing is being adjusted. The

frequent use of this lever consumes time, same with the time consumed when the R/W head is reset to its starting position.

Assume that in this example, α has a value of 20ms, the computation would be as follows:

(THM) = (50-0) + (199-62) + α = 50 + 137 + 20

(THM) = 207 tracks

Seek Time = THM * Seek rate = 187 * 5ms

Seek Time = 935 ms

The computation of the seek time excluded the alpha value because it is not an actual seek or search of a disk request but a reset of the access arm to the starting position.

3.6 C-LOOK Scheduling Algorithm

Circular LOOK is like a C-SCAN which uses a return sweep before processing a set of disk requests. It does not reach the end of the tracks unless there is a request, either read or write on such disk location similar with the LOOK algorithm. Using the same sets of example in FCFS the solutions are as follows:

Solution:

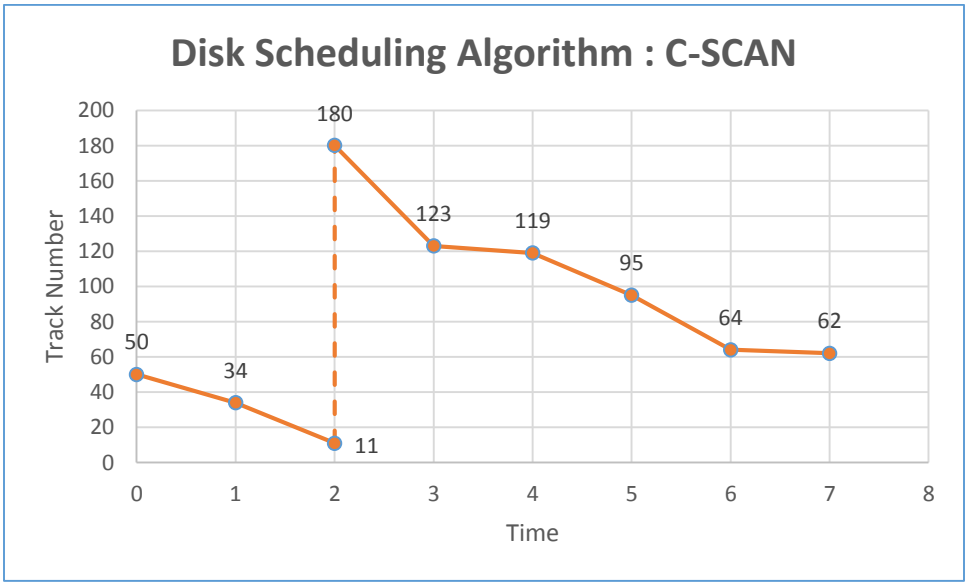

Fig. 7 C-LOOK Representation

Note: using the same value of the alpha in C-SCAN.

(THM) = (50-11) + (180-62) + α = 39 + 118 + 20

(THM) = 177 tracks

Seek Time = THM * Seek rate = 157 * 5ms

Seek Time = 785 ms

Based on the example, this algorithm does not go past the last request in the direction that it is headed. It jumps to the other end but not all the way to the end. It has total head movement of 157 tracks (excluding the alpha) and a seek rate of 785 milliseconds. From 644 tracks in FCFS, C-LOOK reduced it down to 157 tracks.

4. Analysis

Disk scheduling algorithms are used to allocate the services to the I/O requests on the disk and improve its performance. Different qualities exists on these algorithms based on the given examples and computations. Several disadvantages also occur on these different algorithm and these are:

- The FCFS performs operations in order requested. No reordering of work queue since it processed disk requests according to its arrival. There is no starvation and all the requests are serviced but it doesn't provide fastest service.
- The Shortest Seek Time First (SSTF) selects the disk I/O request that requires the least movement of the disk access arm from its current position regardless of direction. It also reduces the seek time compared to FCFS but in this algorithm, I/O requests at the edges of the disk surface may get starved.
- The SCAN algorithm go from the outside to the inside servicing requests and then back from the outside to the inside servicing requests. It also reduces variance compared to SSTF.

- The Circular SCAN (C-SCAN) moves from one end of the disk to the other, servicing requests. When other end is reached, it immediately returns to the beginning of the disk, without servicing any requests. This algorithm treats the cylinders as a circular list that wraps around from the last cylinder to the first one. It also provides a more uniform wait time than SCAN.
- In LOOK scheduling algorithm, the arm goes only as far as the final request in each direction. The direction reverses immediately, without going all the way to the end of the disk.
- The Circular LOOK (C-LOOK) algorithm is similar to C-SCAN. The disk head also goes as far as the last request in its direction then reverses its direction immediately without first going all the way to the end of the disk.

When selecting a Disk Scheduling algorithm, performance depends on the number and types of requests. SSTF is common and has a natural appeal. SCAN gives better performance than FCFS and SSTF. From the given examples, a SCAN change is seen from 644 total head movements to just 157. There is now an understanding as to why an operating system truly relies on the type of algorithm it needs when it is dealing with multiple processes.

The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary. Either SCAN or C-LOOK is a reasonable choice for the default algorithm.