Disk Management

Overview of Mass-Storage Structure

10.1.1 Magnetic Disks

- Traditional magnetic disks have the following basic structure:
    - One or more ***platters*** in the form of disks covered with magnetic media. ***Hard disk*** platters are made of rigid metal, while "***floppy***" disks are made of more flexible plastic.
    - Each platter has two working ***surfaces.*** Older hard disk drives would sometimes not use the very top or bottom surface of a stack of platters, as these surfaces were more susceptible to potential damage.
    - Each working surface is divided into a number of concentric rings called ***tracks.*** The collection of all tracks that are the same distance from the edge of the platter, (i.e. all tracks immediately above one another in the following diagram) is called a ***cylinder***.
    - Each track is further divided into ***sectors,*** traditionally containing 512 bytes of data each, although some modern disks occasionally use larger sector sizes. (Sectors also include a header and a trailer, including checksum information among other things. Larger sector sizes reduce the fraction of the disk consumed by headers and trailers, but increase internal fragmentation and the amount of disk that must be marked bad in the case of errors.)
    - The data on a hard drive is read by read-write ***heads.*** The standard configuration (shown below) uses one head per surface, each on a separate ***arm***, and controlled by a common ***arm assembly*** which moves all heads simultaneously from one cylinder to another. (Other configurations, including independent read-write heads, may speed up disk access, but involve serious technical difficulties.)
    - The storage capacity of a traditional disk drive is equal to the number of heads (i.e. the number of working surfaces), times the number of tracks per surface, times the number of sectors per track, times the number of bytes per sector. A particular physical block of data is specified by providing the head-sector-cylinder number at which it is located.

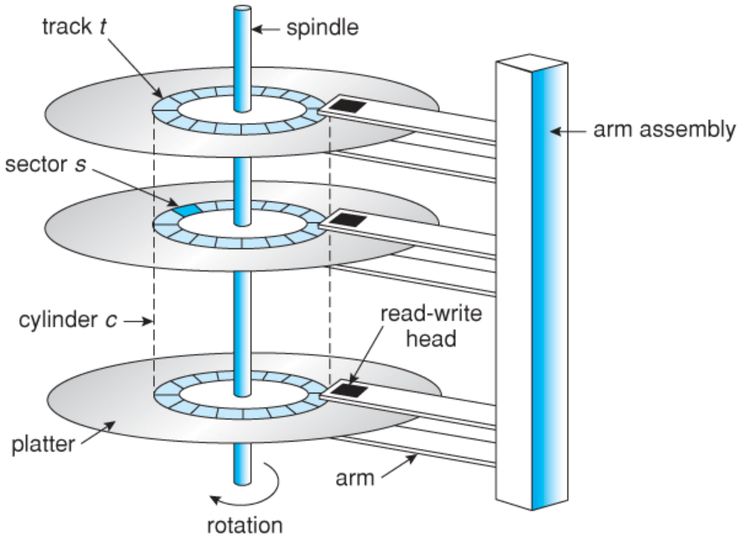

Figure 10.1 - Moving-head disk mechanism.

- In operation the disk rotates at high speed, such as 7200 rpm (120 revolutions per second.) The rate at which data can be transferred from the disk to the computer is composed of several steps:
    - The ***positioning time***, a.k.a. the ***seek time*** or ***random access time*** is the time required to move the heads from one cylinder to another, and for the heads to settle down after the move. This is typically the slowest step in the process and the predominant bottleneck to overall transfer rates.
    - The ***rotational latency*** is the amount of time required for the desired sector to rotate around and come under the read-write head.This can range anywhere from zero to one full revolution, and on the average will equal one-half revolution. This is another physical step and is usually the second slowest step behind seek time. (For a disk rotating at 7200 rpm, the average rotational latency would be 1/2 revolution / 120 revolutions per second, or just over 4 milliseconds, a long time by computer standards.
    - The ***transfer rate***, which is the time required to move the data electronically from the disk to the computer. (Some authors may also use the term transfer rate to refer to the overall transfer rate, including seek time and rotational latency as well as the electronic data transfer rate.)
- Disk heads "fly" over the surface on a very thin cushion of air. If they should accidentally contact the disk, then a ***head crash*** occurs, which may or may not permanently damage the disk or even destroy it completely. For this reason it is normal to ***park*** the disk heads when turning a computer off, which means to move the heads off the disk or to an area of the disk where there is no data stored.
- Floppy disks are normally ***removable***. Hard drives can also be removable, and some are even ***hot-swappable***, meaning they can be removed while the computer is running, and a new hard drive inserted in their place.
- Disk drives are connected to the computer via a cable known as the ***I/O Bus.*** Some of the common interface formats include Enhanced Integrated Drive Electronics, EIDE; Advanced Technology Attachment, ATA; Serial ATA, SATA, Universal Serial Bus, USB; Fiber Channel, FC, and Small Computer Systems Interface, SCSI.
- The ***host controller*** is at the computer end of the I/O bus, and the ***disk controller*** is built into the disk itself. The CPU issues commands to the host controller via I/O ports. Data is transferred between the magnetic surface and onboard ***cache*** by the disk controller, and then the data is transferred from that cache to the host controller and the motherboard memory at electronic speeds.

10.4 Disk Scheduling

- As mentioned earlier, disk transfer speeds are limited primarily by ***seek times*** and ***rotational latency.*** When multiple requests are to be processed there is also some inherent delay in waiting for other requests to be processed.
- ***Bandwidth*** is measured by the amount of data transferred divided by the total amount of time from the first request being made to the last transfer being completed, (for a series of disk requests.)
- Both bandwidth and access time can be improved by processing requests in a good order.
- Disk requests include the disk address, memory address, number of sectors to transfer, and whether the request is for reading or writing.

10.4.1 FCFS Scheduling

- ***First-Come First-Serve*** is simple and intrinsically fair, but not very efficient. Consider in the following sequence the wild swing from cylinder 122 to 14 and then back to 124:

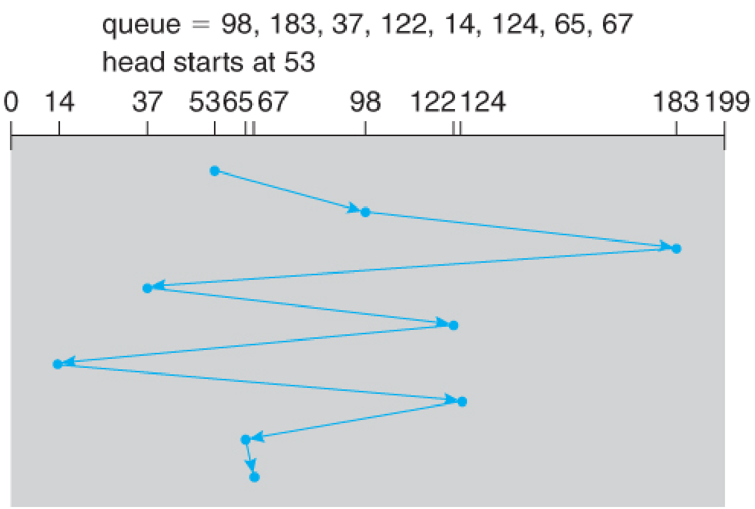

Figure 10.4 - FCFS disk scheduling.

10.4.2 SSTF Scheduling

- ***Shortest Seek Time First*** scheduling is more efficient, but may lead to starvation if a constant stream of requests arrives for the same general area of the disk.
- SSTF reduces the total head movement to 236 cylinders, down from 640 required for the same set of requests under FCFS. Note, however that the distance could be reduced still further to 208 by starting with 37 and then 14 first before processing the rest of the requests.

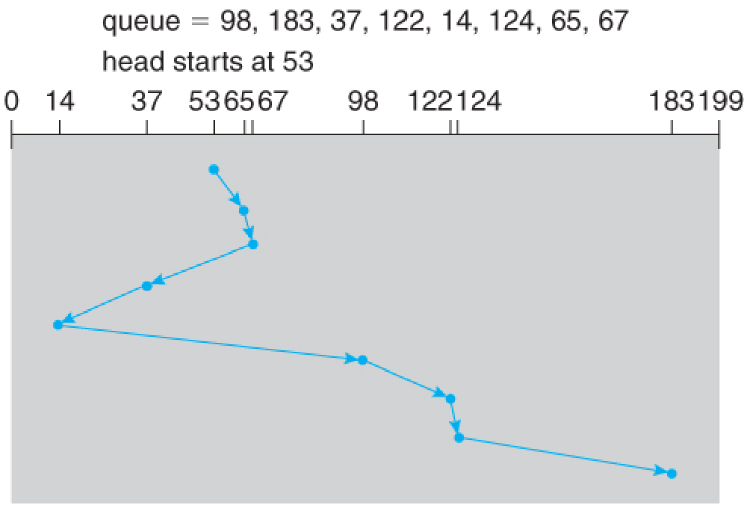

Figure 10.5 - SSTF disk scheduling.

10.4.3 SCAN Scheduling

- The ***SCAN*** algorithm, a.k.a. the ***elevator*** algorithm moves back and forth from one end of the disk to the other, similarly to an elevator processing requests in a tall building.

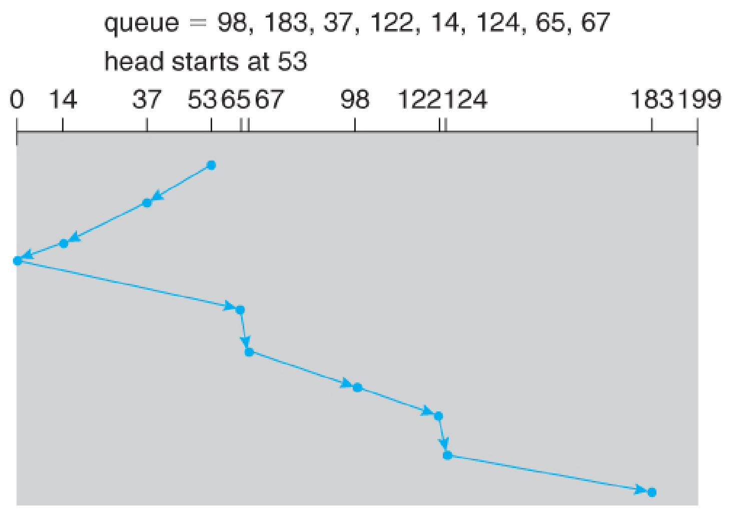

Figure 10.6 - SCAN disk scheduling.

- Under the SCAN algorithm, If a request arrives just ahead of the moving head then it will be processed right away, but if it arrives just after the head has passed, then it will have to wait for the head to pass going the other way on the return trip. This leads to a fairly wide variation in access times which can be improved upon.
- Consider, for example, when the head reaches the high end of the disk: Requests with high cylinder numbers just missed the passing head, which means they are all fairly recent requests, whereas requests with low numbers may have been waiting for a much longer time. Making the return scan from high to low then ends up accessing recent requests first and making older requests wait that much longer.

10.4.4 C-SCAN Scheduling

- The ***Circular-SCAN*** algorithm improves upon SCAN by treating all requests in a circular queue fashion - Once the head reaches the end of the disk, it returns to the other end without processing any requests, and then starts again from the beginning of the disk:

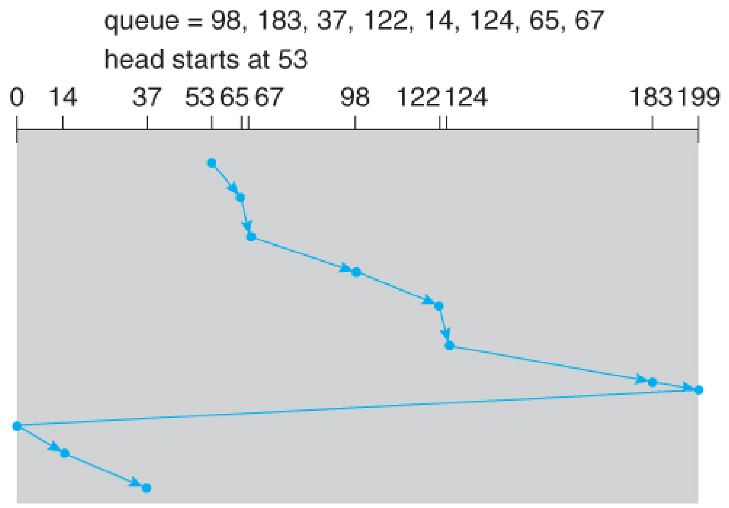

Figure 10.7 - C-SCAN disk scheduling.

12.4.5 LOOK Scheduling

- ***LOOK*** scheduling improves upon SCAN by looking ahead at the queue of pending requests, and not moving the heads any farther towards the end of the disk than is necessary. The following diagram illustrates the circular form of LOOK:

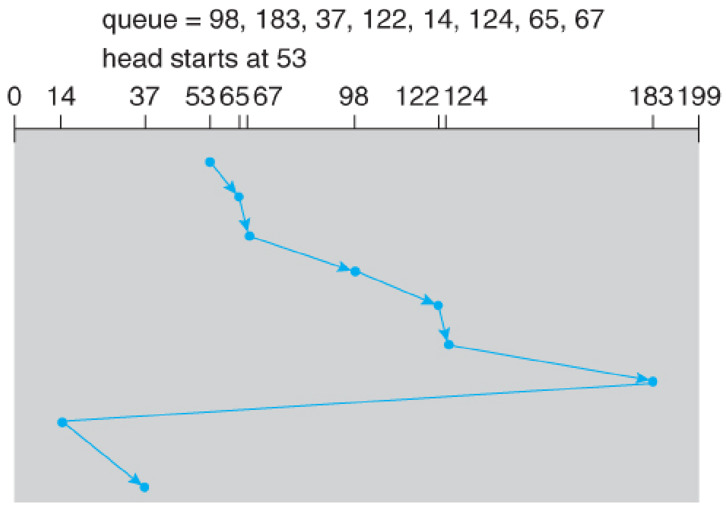

Figure 10.8 - C-LOOK disk scheduling.

10.4.6 Selection of a Disk-Scheduling Algorithm

- With very low loads all algorithms are equal, since there will normally only be one request to process at a time.
- For slightly larger loads, SSTF offers better performance than FCFS, but may lead to starvation when loads become heavy enough.
- For busier systems, SCAN and LOOK algorithms eliminate starvation problems.
- The actual optimal algorithm may be something even more complex than those discussed here, but the incremental improvements are generally not worth the additional overhead.
- Some improvement to overall filesystem access times can be made by intelligent placement of directory and/or inode information. If those structures are placed in the middle of the disk instead of at the beginning of the disk, then the maximum distance from those structures to data blocks is reduced to only one-half of the disk size. If those structures can be further distributed and furthermore have their data blocks stored as close as possible to the corresponding directory structures, then that reduces still further the overall time to find the disk block numbers and then access the corresponding data blocks.
- On modern disks the rotational latency can be almost as significant as the seek time, however it is not within the OSes control to account for that, because modern disks do not reveal their internal sector mapping schemes, (particularly when bad blocks have been remapped to spare sectors.)
  - Some disk manufacturers provide for disk scheduling algorithms directly on their disk controllers, (which do know the actual geometry of the disk as well as any remapping), so that if a series of requests are sent from the computer to the controller then those requests can be processed in an optimal order.
  - Unfortunately there are some considerations that the OS must take into account that are beyond the abilities of the on-board disk-scheduling algorithms, such as priorities of some requests over others, or the need to process certain requests in a particular order. For this reason OSes may elect to spoon-feed requests to the disk controller one at a time in certain situations.