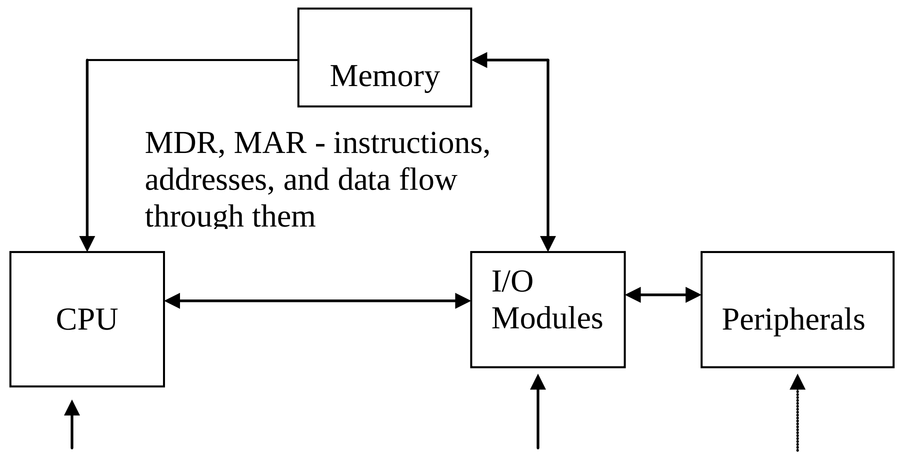
CIS-350
Infrastructure Technologies

Chapter 10: Computer Peripherals

Read chapter 10. (We will cover hard disk storage capacity and access time as well as displays. You need to read the rest.)



1. Registers (IR, PC, A), base registers for arithmetic on integer- and floating-point numbers	1. Interfaces, controllers, cards	Discuss now
2. Machine cycle-- F&E phase	2. Mainframe --I/O control units --channels	1. Devices that are separate from the basic computer 2. Not the CPU or memory. 3. Classified as input, output, and storage 4. Connect via ports or interface to systems bus 5. I/O devices, also storage devices --tapes --disks --CD-ROMs --displays --plotters --printers 6. PCs and mainframes have very similar devices
3. Addressing modes (will <u>not</u> talk about them)	3. Translate data and control peripherals	
4. Data, addresses, instruction flow	4. Issue <u>interrupts</u> to communicate with the CPU	
	5. Operate in an <u>asynchronous</u> manner, not synchronized by the clock like CPU and memory	
	6. <u>DMA</u> - Direct Memory Access, I/O module has direct access to memory - very important concept → releases the CPU to do other tasks	

The Hierarchy of Storage

Increasing storage capacity

Device	Typical access times
CPU registers	0.25 nsec
Cache memory (SRAM)	1-10 nsec
Conventional memory (DRAM)	10-50 nsec
Flash memory	120 μsec
Magnetic disk drive	10-50 msec
Optical disk drive	100-500 msec
Magnetic tape	0.5 and up sec

Increasing access times

Note the speed disparity: nanoseconds (memory) vs. milliseconds (disk drive)

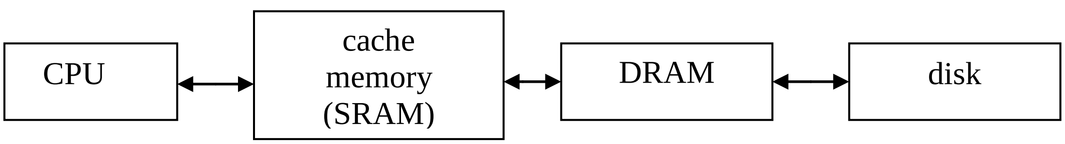
$1ns=10^{-9}s$ $\quad$ $1ms=10^{-3}s$

CPU clock speed - 2 GHz $\quad t = 1/f = 1/(2x10^{9}) = 0.5x10^{-9}s = 0.5ns$ (nanoseconds)

The speed of the computer also depends not only on the clock's speed, but also on

- the width and speed of buses
- speed of memory (the memory should keep up with the clock)

There has to be a synergy between all components.



DRAM & SRAM are volatile

The CPU will always attempt to access current instructions and data in cache memory before it looks for them in DRAM.

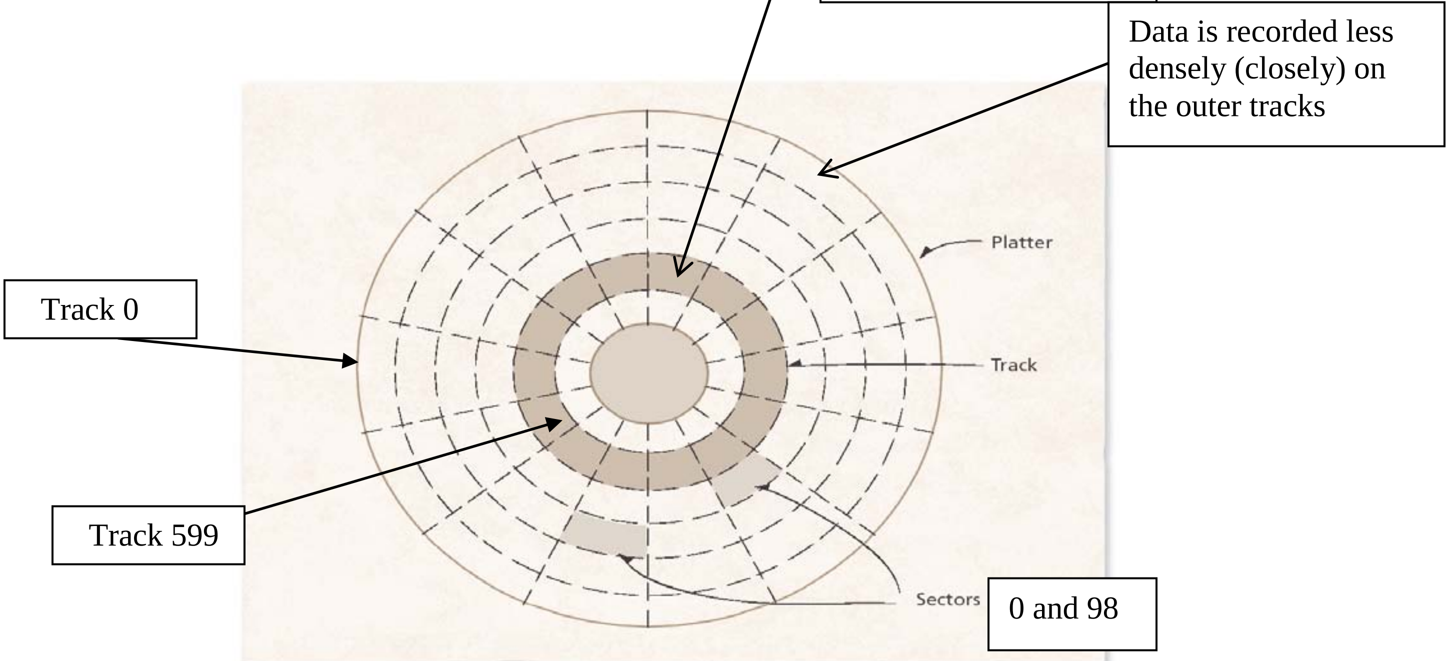
Reasons: speed & efficiency - cache memory is faster than DRAM

If not present in SRAM, the CPU will look in DRAM. If not present in DRAM, it will look on disk

permanent, stores data & programs, good for random and sequential access, data and programs must to be transferred to main memory to be processed by the CPU

Disk storage capacity

1. surface covered with magnetic substance
2. I/O
3. logically divided into tracks and sectors



Regardless of the track, the same angle is always swept out when a sector is accessed (360°/100 sectors = 3.6°/sector); thus, the transfer time is kept constant with the motor rotating at a fixed speed. This technique is called <u>Constant Angular Velocity (CAV)</u>. It has the advantage of simplicity and fast access. Hard disks generally store data in a CAV format.

The other alternative that might be used would be Constant Linear Velocity (CLV). This technique is more complicated because the motor would rotate with the variable speed; faster when the data is located on the outer sectors and slower when the data is located on the inner sectors.

Computation of Storage Capacity

1. <u>one sector</u>, usually = 512 bytes/characters is R/W at a time, or cluster = 2 or 4 sectors at a time.

   Note that in inner sectors, data is more densely packed.

2. Assume one track has 100 sectors, #-ed 0-99

3. <u>track capacity</u>:

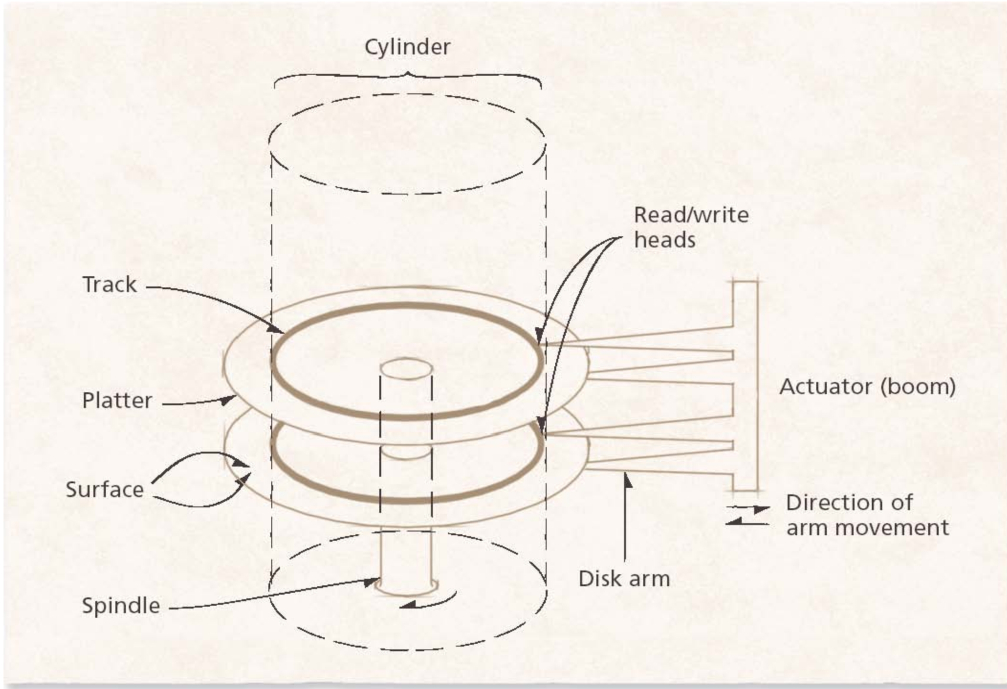
   100 sectors * 512 bytes = 51,200 bytes

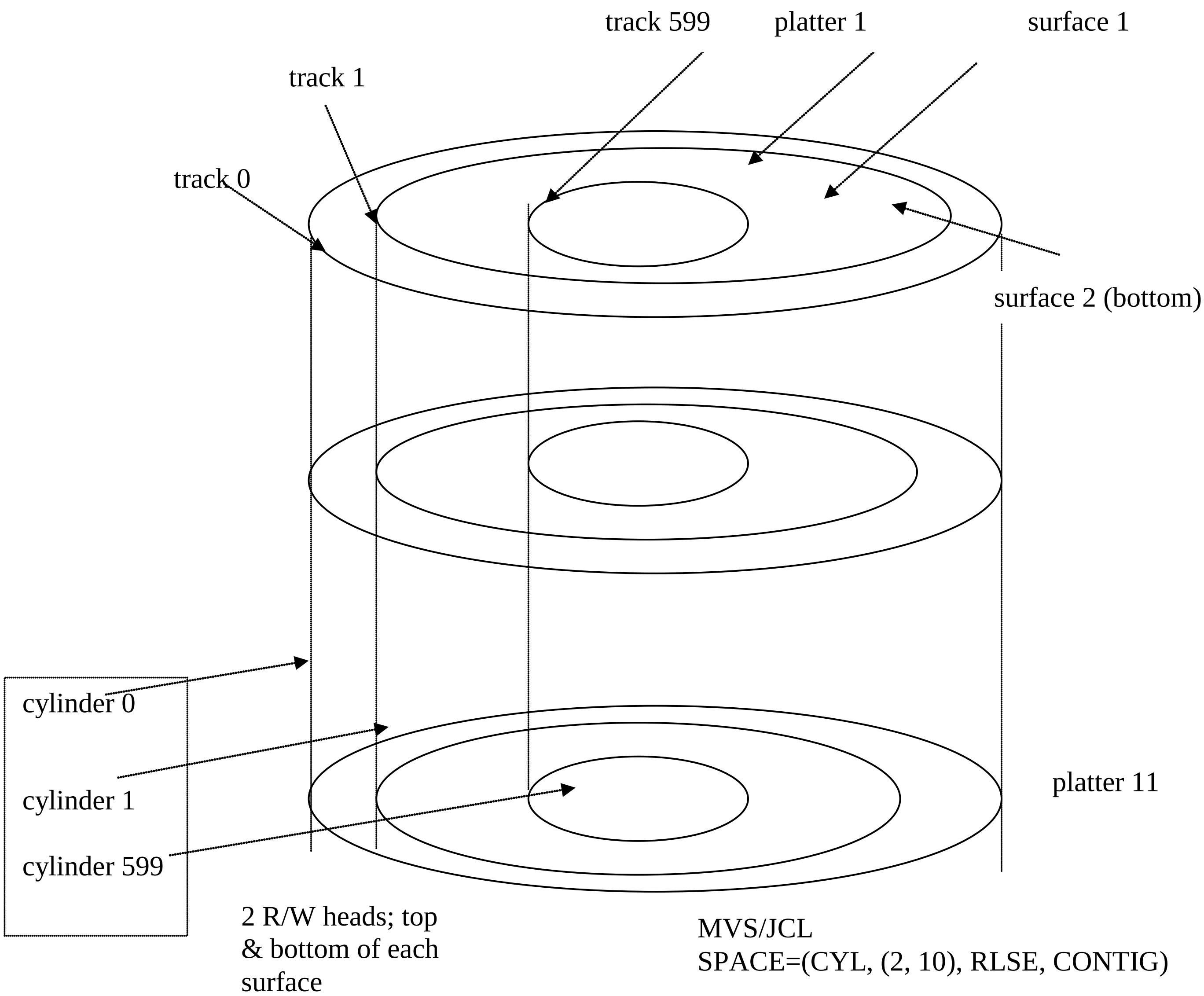
4. <u>surface capacity</u>: (assume 600 tracks)

   600 tracks * 51,200 bytes = 30,720,000 bytes = 30,000 KB = 29.3 MB
   (divided by 1024) (divided by 1024 again)

5. <u>platter capacity</u>: 2 surfaces

2 *30,000 KB = 60,000 KB





<u>Cylinder</u> - is a logical unit of disk storage, defined as all tracks with the same number across all disk surfaces

If data is allocated in cylinders, the R/W operations are faster.

6. <u>Cylinder capacity</u>:

a. # of recording surfaces

platter 1	- 1	1	1
platters 2-10	- 2	2*9	18
platter 11	- 1	1	1
			20

b. # of recording surfaces * track capacity (from page 3, point 3)

20 * 51,200 bytes = 1,024,000 bytes = 1000 KB = 0.9766 MB
(divide by 1024) (divide by 1024 again)

7. Capacity of the disk pack

# of tracks * capacity of the cylinder

600 * 1,000 KB = 600,000 KB = 585.94 MB = 0.572 GB

Disk access time

The average access time is the **sum** of:

1. Average seek time
2. Average latency time or rotational delay time
3. Transmission (transfer) time

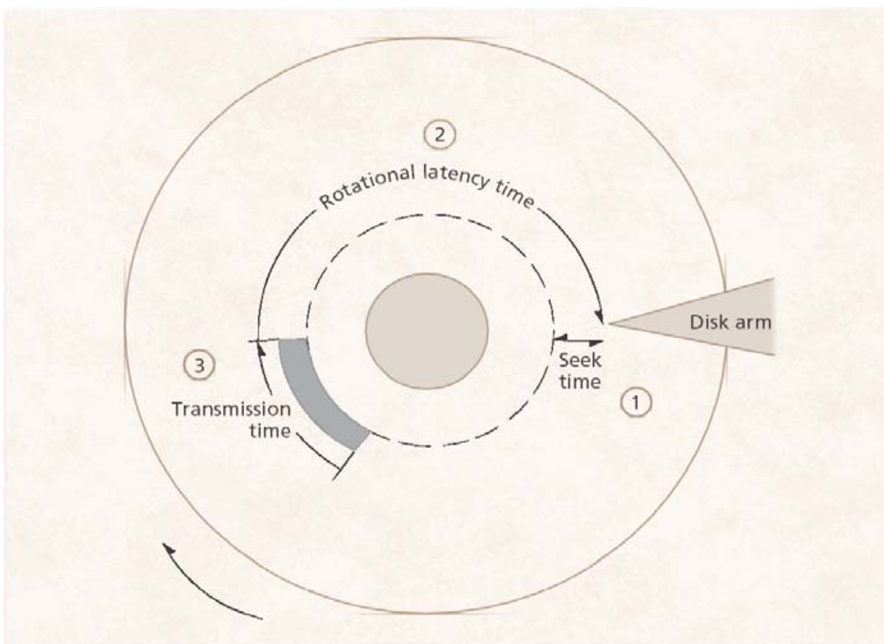


Figure 12.2 Hard disk track-to-track seek times and latency times.

Model (Environment)	Average Seek Time (ms)	Average Rotational Latency (ms)
Maxtor DiamondMax Plus 9 (High-end desktop)	9.3	4.2
WD Caviar (High-end desktop)	8.9	4.2
Toshiba MK8025GAS (Laptop)	12.0	7.14
WD Raptor (Enterprise)	5.2	2.99
Cheetah 15K.3 (Enterprise)	3.6	2.0

1. Average seek time = 10 ms

Depends on the speed of movement of the electromechanical parts on which the R/W heads are mounted.

2. Average latency = $\frac{1}{2} \cdot \frac{1}{\text{rotational speed}}$

on the average, the disk will have to rotate half way to read the the desired block

7200 revolutions/min = 120 revolutions/sec

Average latency = $\frac{1}{2} \times \frac{1}{120}$ s = 0.00416s = 4.16ms

Use the correct units: milliseconds, seconds

3. Transfer time for N sectors = $\frac{N}{\text{\# of sectors on a track} \times \text{rotational speed}}$

Transfer time for 1 sector = $\frac{1}{100 \times 120 \text{ revolution/sec}}$ = 0.083 ms

negligible time

(1 sector uses 1% of the track length;
100 sectors use 100 % of the track length)

Transfer time for 50 sectors = $\frac{50 \text{ sectors}}{100 \times 120}$ = 4.16 ms

the more sectors you transfer, the longer it takes

To compute access time, one needs to add (See in-class activity #5)

- the average seek time
- average latency, and
- transfer time

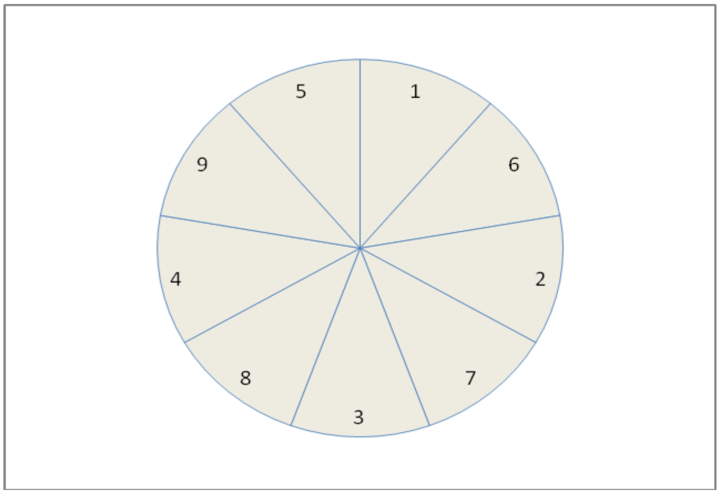
Average access time to (1 sector) =10ms + 4.16 ms + 0.083 ms = 14.243 ms

Average access time to (50 sectors) =10ms + 4.16 ms + 4.16 ms = 18.32 ms

Techniques used to improve disk access time

- disk interleaving
- disk arrays
- disk scheduling
- disk defragmentation
- disk compression

Disk interleaving



Disk is formatted this way so that the sectors are interleaved. In 2:1 interleaving each successive block to read/write is two blocks away. In 3:1 interleaving each successive block to read/write is three blocks away. It gives the disk controller additional time to perform the required DMA transfer between disk and memory and to be ready when the next consecutively numbered block arrives.

Disk arrays

used on large mainframes and servers running Windows Server OS or Unix/Linux

1. group multiple physical disks together to create a logical disk with a micro-controller coordinating the drives
2. reduce overall access time or increase system reliability (by storing exactly the same data - redundancy) thus protecting the data against loss

Disk space is not utilized well.

Note that in disk arrays data redundancy is desired. Recall the CIS-310 Database Design course. In a well designed database redundancy is not desired. Databases use only controlled redundancy (by introducing foreign keys).

The redundancy concept/technique is crucial to the operation of fault-tolerant computers/systems. They provide reduced service even after some of its components fail.

Examples of fault-tolerant systems:
- pacemakers in the heart (life and death value)
- airline reservation systems (business value)

use duplicated I/O systems
use duplicated CPUs

Two processors execute the same program in parallel. If a single CPU fails, another picks up the slack.

One type of disk array is known as <u>R.A.I.D.</u> -- <u>redundant array of inexpensive disks</u>.

<u>Two methods of implementing a disk array</u>

1. <u>mirrored array</u>
2. <u>striped array</u>

50% of disk space is utilized

<u>A mirrored array</u>

three - <u>33%</u>

- two or more disks store <u>exactly</u> the same data - disadvantage

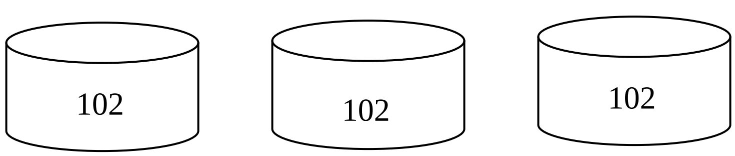
- <u>access time is reduced</u> because alternate blocks of data are <u>read</u> from different drives - advantage

2 disks → access time/2
3 disks → access time/3

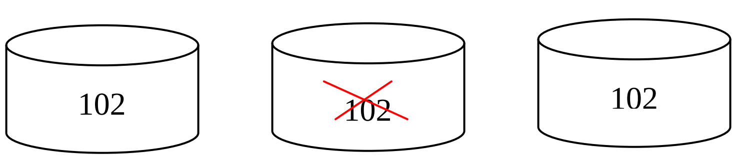
disks use independent controllers

reduced by the # of
disks in the array

<u>Increased system reliability</u> due to the method known as majority logic.



Data on three disks are identical. Majority is used in processing.



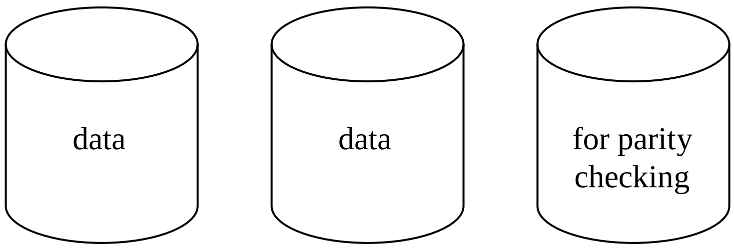
Data on two disks (1 and 3) are identical. Disk 2 is flagged as error. Majority is used in processing.

For two/three drives in the array, all data is written to two/three identical drives. If one drives fails, the system can continue, using the mirror drive. This approach is popular because it's easy to implement, but it doubles/triples storage costs. One can obtain improved read performance by reading from the alternate drive when the primary drive is busy.

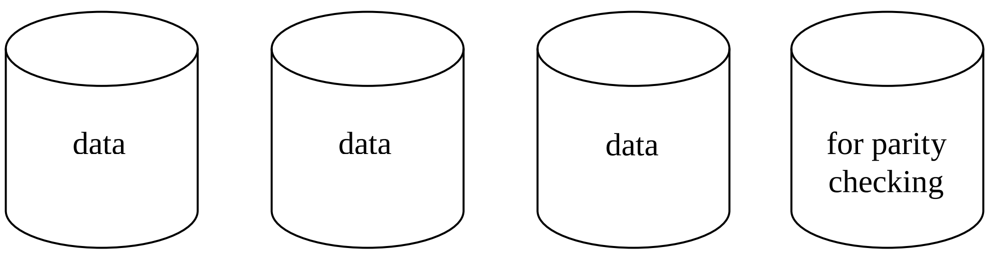
One can remove a disk from the array of disks while the computer is running without interrupting the operation of the system (should see it on the IT Tour to the Miller Technology Hall).

A striped array

-- requires a minimum of 3 or more disks
-- one disk drive is reserved for error checking



One disk is for parity checking for detecting and correcting potential errors in each block of data 66.6% utilization for 3 disks.



75% utilization for 4 disks.

Parity permits error correction. Slow writes due to parity computation. Fast reads. Bad performance on small data transfers. Good performance on large data transfers.

The performance level and features for mirrored and striped arrays depend really on the RAID Level: 0-5. There are subtle differences between each RAID level. See the textbook for more information.

Disk scheduling

Also, see section 18.8 on pp. 633-635 and Figure 18.25 on p. 636 in textbook.

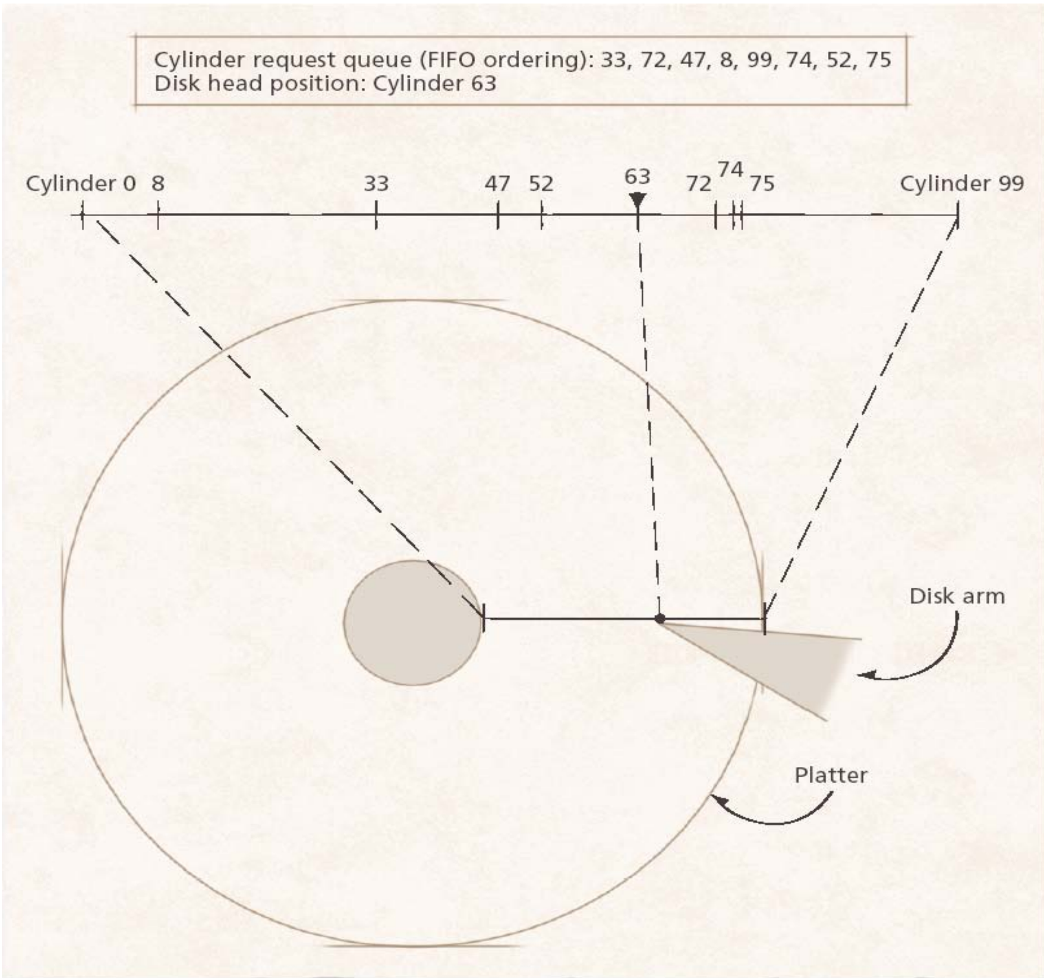
Goal – reduce the time spent seeking records can significantly improve throughput as well as the mean response time and variance of response times.

It involves careful examination of pending requests to determine the most efficient way to service them.

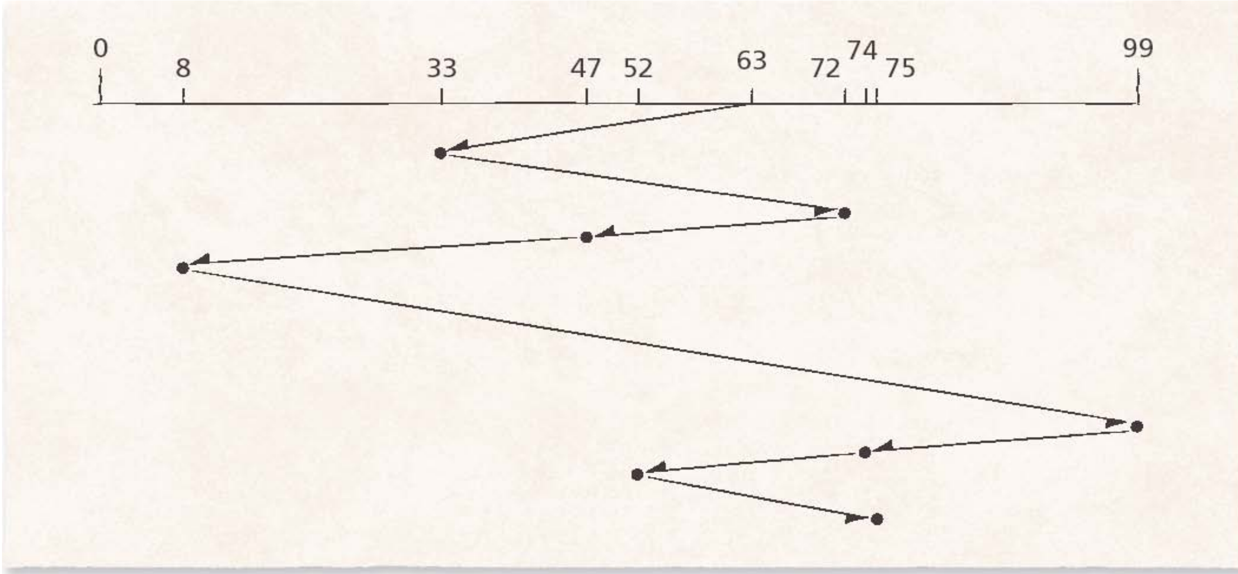
A disk scheduler examines the possible relationships among waiting R/W requests, then reorders the queue so that the requests will be serviced with minimum mechanical motion.

Among two most common types of scheduling: <u>seek time optimization</u> and <u>rotational delay optimization</u>, the former seems to be the most important.

Disk request pattern.

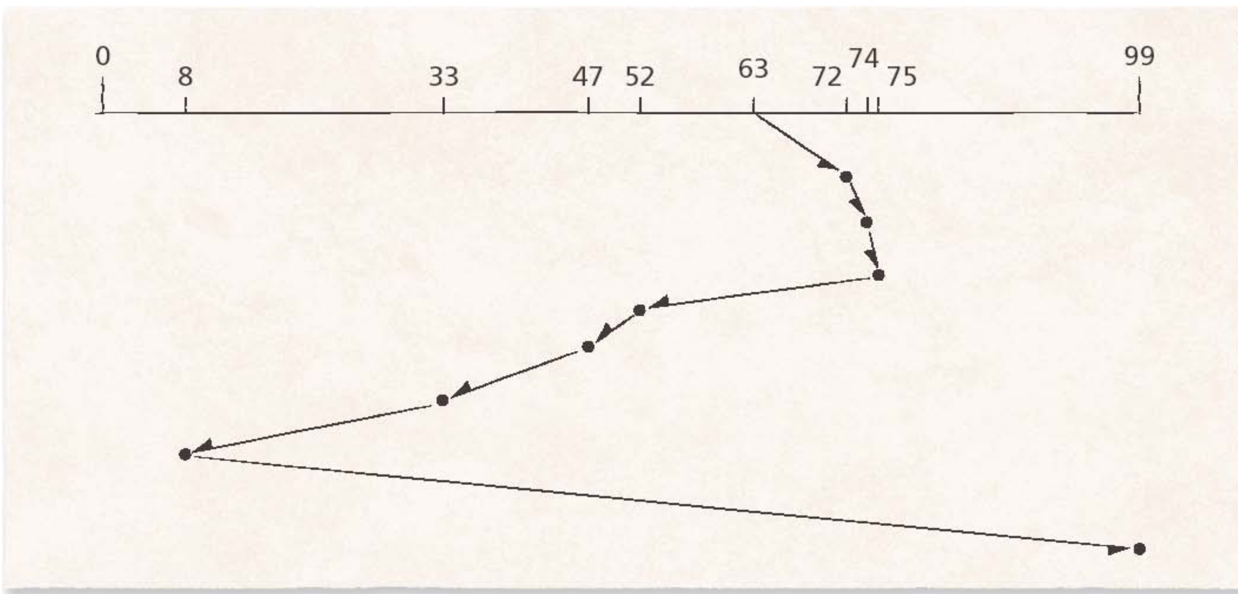


Seek pattern under the First-Come-First-Serve (FCFS) strategy.



The movement of the R/W head. Objective - minimize the distance traveled.

Seek pattern under the Shortest-Seek-Time-First (SSTF) strategy.



- Other ways of optimizing disk performance

  – **<u>Defragmentation</u>** (p. 563)

    As files and records are added and deleted, a disk's data tends to be dispersed throughout the disk, or fragmented. This increases access time. Operating systems provide defragmentation (or disk reorganization) programs that can be used periodically to reorganize files.

    - Place related data in contiguous sectors
    - Decreases number of seek operations required

- Partitioning can help reduce fragmentation because files are restricted to these partitions
- Partitions can lead to wasted memory

– **Compression**

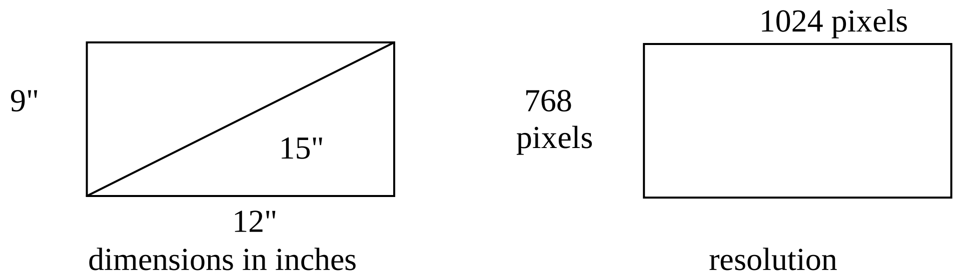
- Data consumes less disk space
- Improves transfer and access times
- Increased execution-time overhead to perform compression/decompression

<u>Displays</u> (Section 10.6, pp. 322-330, textbook - Englander)

1. An image is made up of pixels

2. The # of pixels determines the resolution of the screen, for example: 768 x 1024.

$9^2 + 12^2 = 15^2$
$81 + 144 = 225$



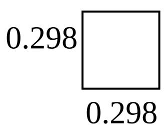
The <u>resolution</u> of commercial video monitors = 0.2 - 0.3 millimeter (mm).

How is this # obtained?

$$\frac{9''}{768} \quad \text{or} \quad \frac{12''}{1024} \cong 0.01172'' \cong 0.298 \text{ mm}$$

(1"=2.54 cm=25.4 mm)

The <u>display parameter</u> given in the technical data; should be small enough; maximum 4 or 5 pixels in 1 mm; smaller sizes cannot be distinguished by the human eye;



Size of the pixel

Display parameter is dots per inch (dpi).

Ex. 768 pixels/9"=85.3 pixels per inch.

Each pixel represents a color.

A color pixel is made up of a mixture of different intensifies of red, blue, and green (RBG).

<u>True color system of this resolution needs</u>

3 bytes x 768 x 1024 = 2,359,296 bytes ≅ 2.25 MB of video storage

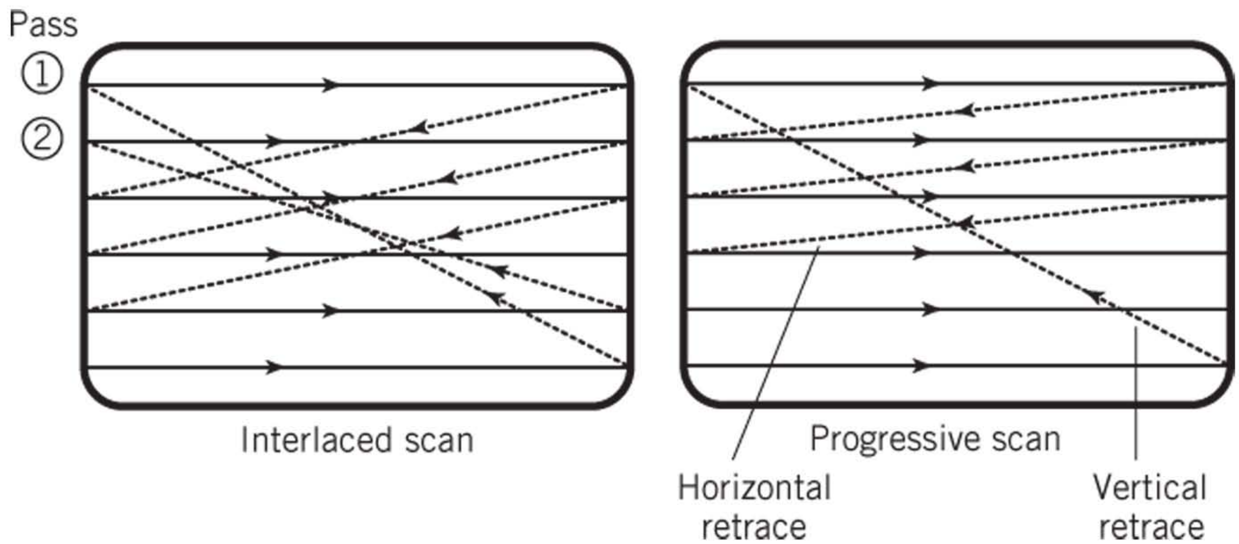
Each pixel needs 3 bytes in the true color system. Need one byte for 256 shades of Red. Need one byte for 256 shades of Blue. Need one byte for 256 shades of Green.

$2^8 \times 2^8 \times 2^8 = 2^{24} = 16{,}777{,}216$ colors.

<u>Scanning</u> – method of displaying pixels

The display is produced by scanning and displaying each pixel, one row at a time, from L → R, top to bottom. This method of displaying pixels is known as a <u>raster scan</u>.

This process is repeated more than 30 times/sec.



<u>Interlaced scan/display</u>: less demanding on the monitor, cheaper monitors, and the image may flicker.

<u>Noninterlaced scan</u>: row 1, 2, 3…, scanned, more demanding, more expensive monitors, the image holds on the screen, does not flicker

If you increase the resolution, the display will <u>flicker</u> on a poorer quality monitor.

A refresh rate parameter - good quality monitors 70 Hz - 80 Hz.