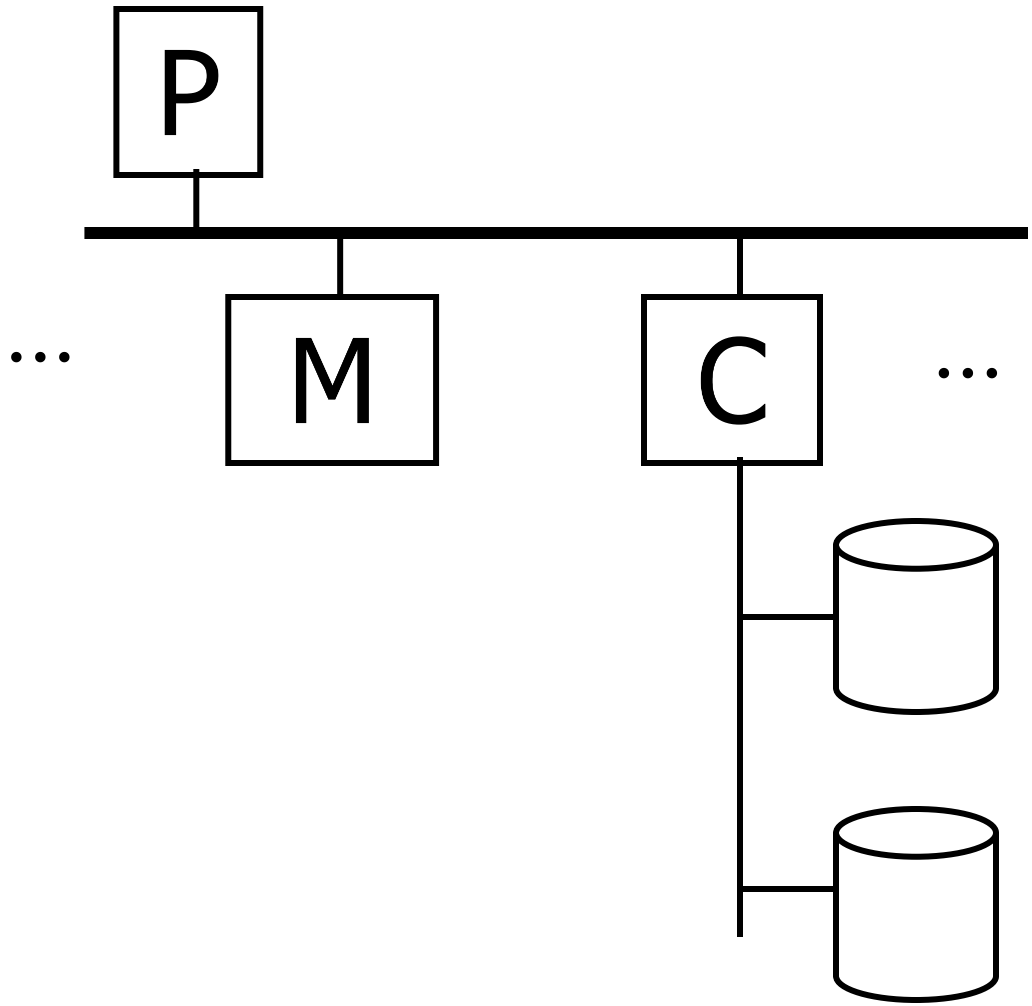
Hardware

Read Chap. 4 Riguzzi et al. Sistemi Informativi

Slides derived from those by Hector Garcia-Molina
Some images by Wikipedia

Outline

- Hardware: Disks
- Access Times
- Reliability
- RAID



Typical Computer

Secondary Storage

Processor

Fast, slow, reduced instruction set, with cache, pipelined…

Speed: 10,000 → 100,000 MIPS

Memory

Fast, slow, non-volatile, read-only,…

Access time: $10^{-6} \rightarrow 10^{-9}$ sec.

$1\ \mu s \rightarrow 1$ ns

Secondary storage

Hard Disks

Tertiary storage

Optical disks:

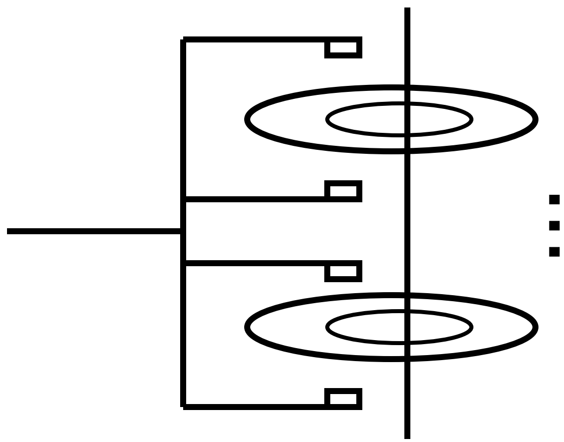
- CD-ROM
- DVD-ROM…

Tape

- Cartridges

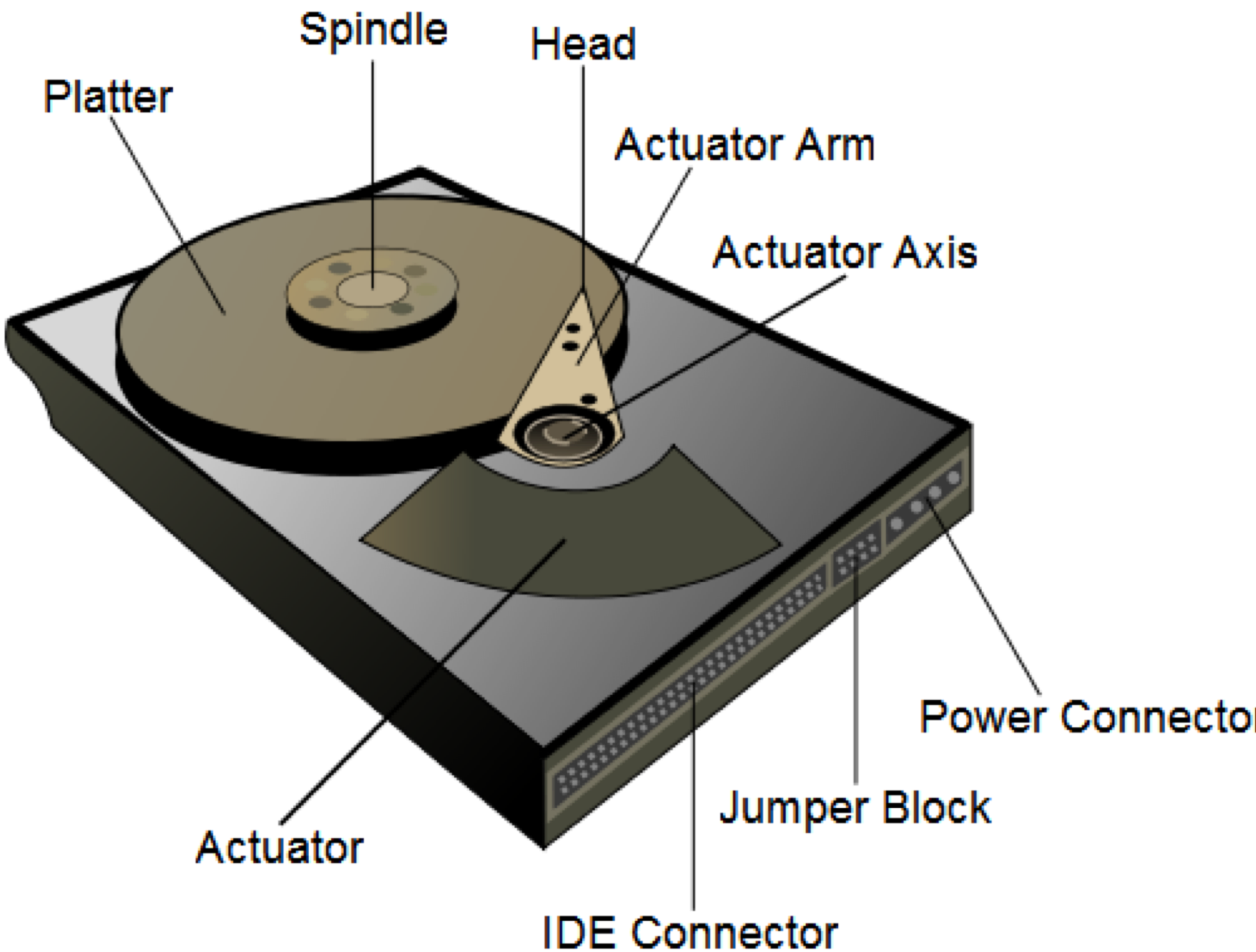
Robots

Focus on: "Typical Disk"

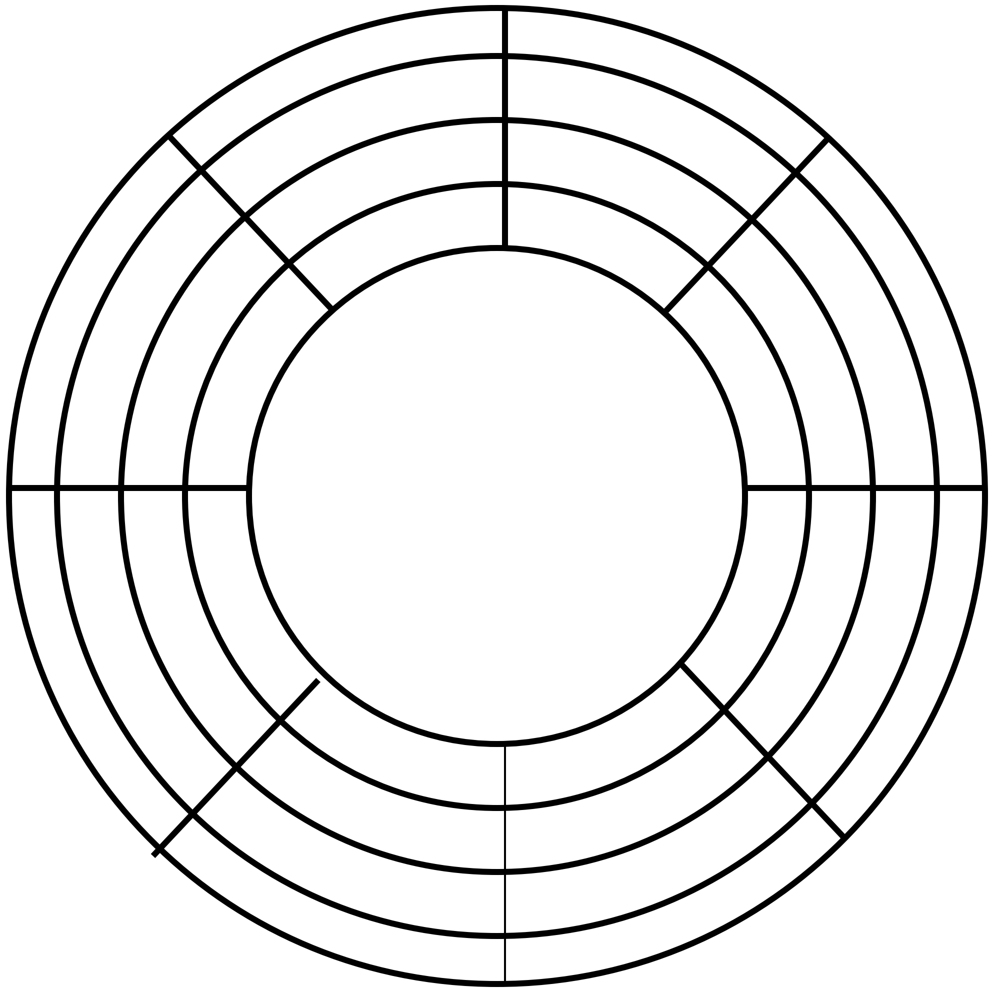


Terms: Platter, Surface, Head, Actuator
Cylinder, Track
Sector (physical),
Block (logical), Gap

Disk Architecture



Top View



“Typical” Numbers

Diameter: 1.8, 2.5 or 3.5 inches

(1 inch=2.54 cm)

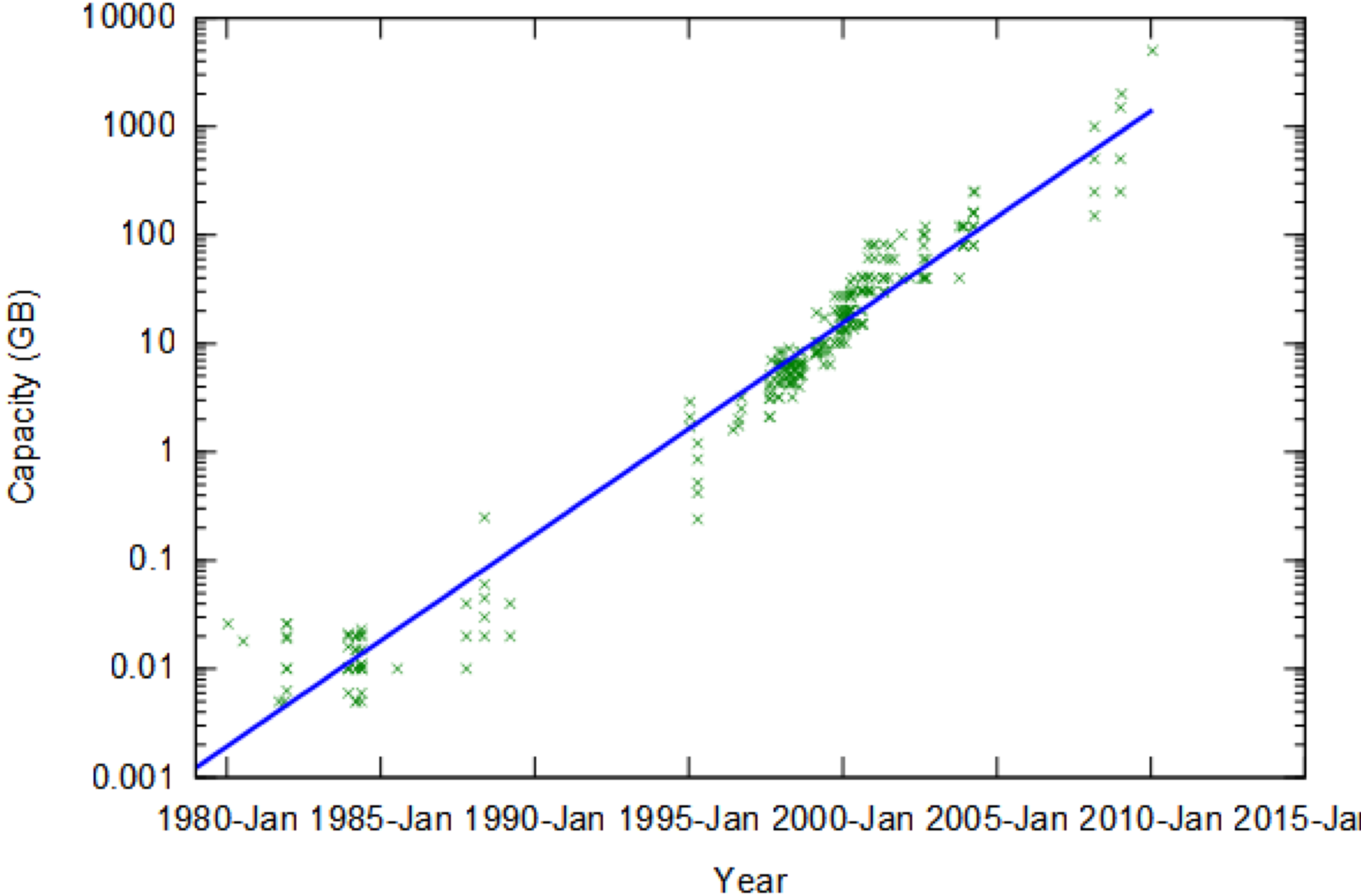
Cylinders: 10000 → 50000

Platters: 2 -> 7

Sector Size: 512B → 50KB

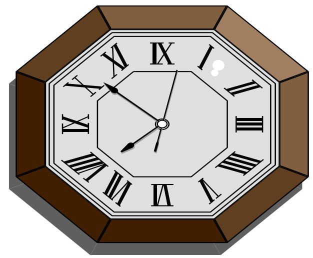
Capacity: 72 GB → 6TB

Capacity



10000
1000
100
10
1
0.1
0.01
0.001
Capacity (GB)
1980-Jan 1985-Jan 1990-Jan 1995-Jan 2000-Jan 2005-Jan 2010-Jan 2015-Jan
Year

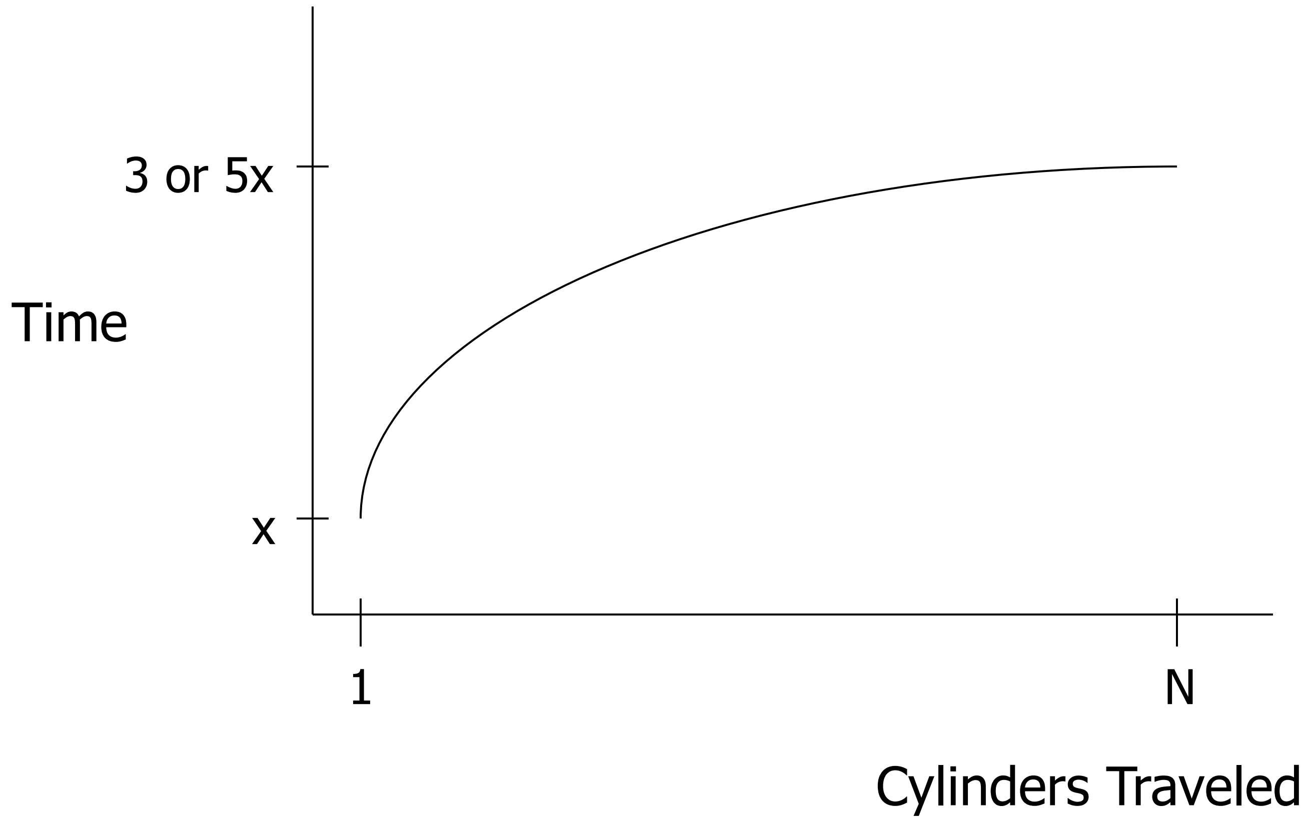
Disk Access Time

I want block X →  → block x in memory

?

Time = Seek Time +
Rotational Delay +
Transfer Time +
Other

Seek Time



Average Random Seek Time

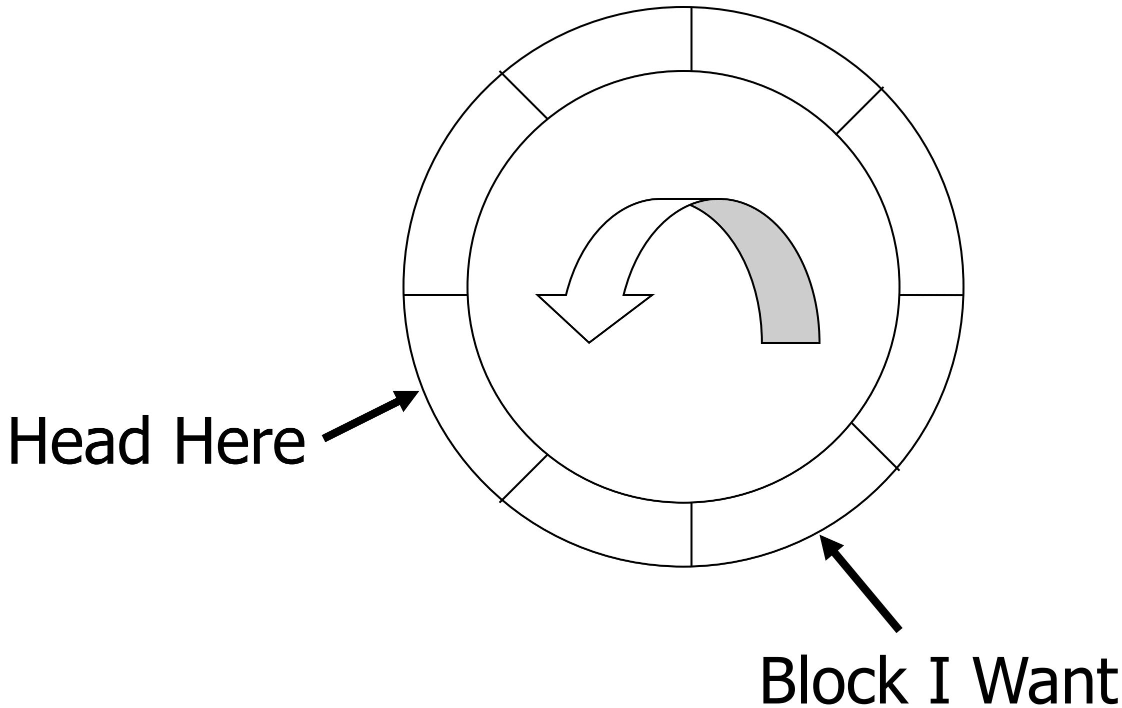
$$S = \frac{\sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} \text{SEEKTIME}(i \rightarrow j)}{N(N-1)}$$

"Typical" S: 3 ms $\rightarrow$ 10 ms

Seek time

- Average seek time ranges from under 4 ms for high-end server drives to 15 ms for mobile drives
- For the most common mobile drives si about 12 ms
- For the most common desktop drives is about 9 ms.

Rotational Delay



Average Rotational Delay

R = 1/2 revolution

“typical” R = 4.17 ms (7200 RPM)
R=3 ms (10000 RPM)
R=2 ms (15000 RPM)

Transfer Time

- transfer time: revolution/n. blocks per track

Other Delays

- CPU time to issue I/O
- Contention for controller
- Contention for bus, memory

"Typical" Value: 0

- So far: Random Block Access
- What about: Reading “Next” block?

Time to get block = revolution/blocks

Cost for Writing similar to Reading

…. unless we want to verify!
need to add (full) rotation +
revolution/blocks

- To <u>Modify</u> a Block?

<u>To Modify Block:</u>

(a) Read Block

(b) Modify in Memory

(c) Write Block

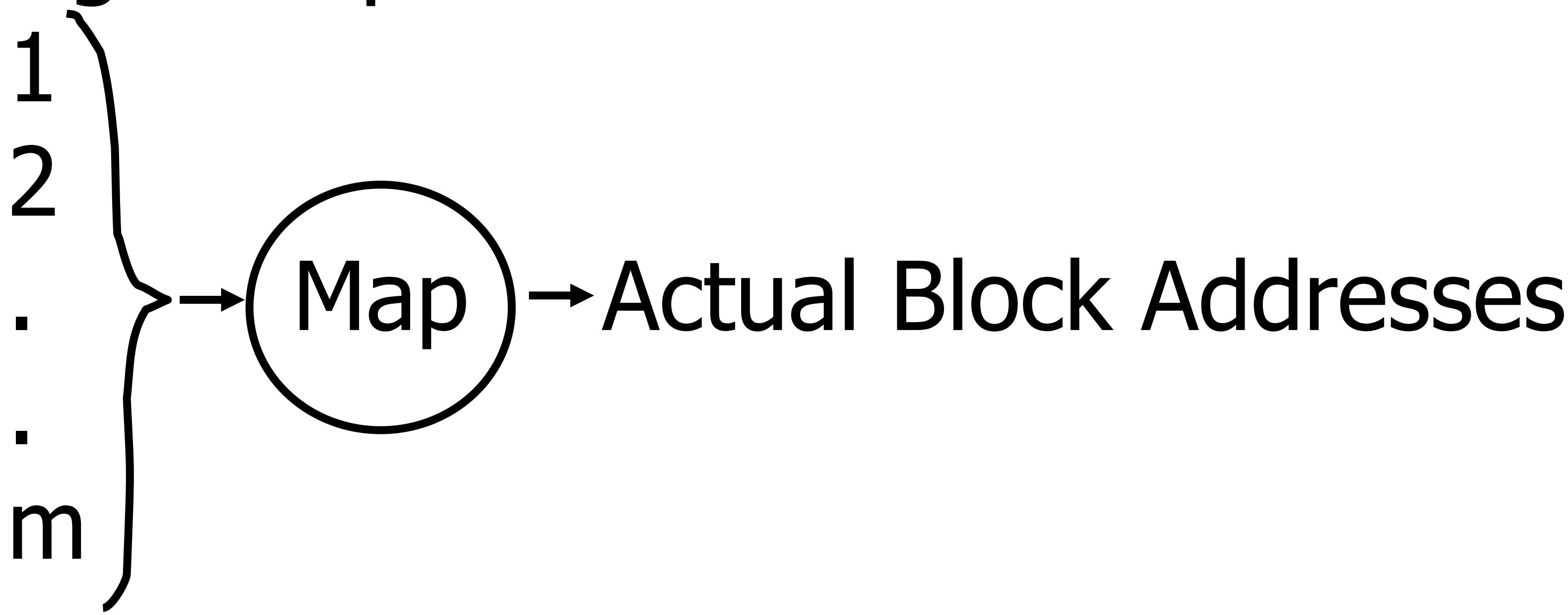
[(d) Verify?]

Block Address:

- Physical Device
- Cylinder (Track) #
- Surface #
- Sector

Complication: Bad Blocks

- Messy to handle
- Map via software from integer sequence



An Example

Megatron 747 Disk

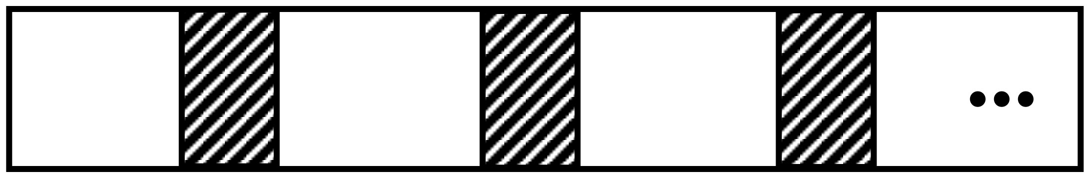
- 3.5 in diameter
- 8 platters, 16 surfaces
- 2^{14}=16,384 tracks per surface (16,384 cylinders)
- 2^{7}=128 sectors per track
- 2^{12}=4096 bytes per sector

- Capacity
  - Disk=$2^{4}*2^{14}*2^{7}*2^{12}=2^{37}$=128GB
  - Single track=$2^{7}*2^{12}$=512KB
- Rotation speed: 7200 RPM
- Average seek time: 8.5 ms

7200 RPM → 120 revolutions / sec

→ 1 rev. = 8.33 msec.

One track:



Time over useful data:(8.33)(0.9)=7.5 ms.

Time over gaps: (8.33)(0.1) = 0.833 ms.

Transfer time 1 sector = 7.5/128=0.059 ms.

Trans. time 1 sector+gap=8.33/128=0.065ms.

Burst Bandwith

4 KB in 0.059 ms.

BB = 4/0.059 = 68 KB/ms.

or

BB =68 KB/ms x 1000 ms/1sec

x 1MB/1024KB

= 68,000/1024 = 66.4 MB/sec

Sustained bandwith (over track)

512 KB in 8.33 ms.

SB = 512/8.33 = 61.5 KB/ms

or

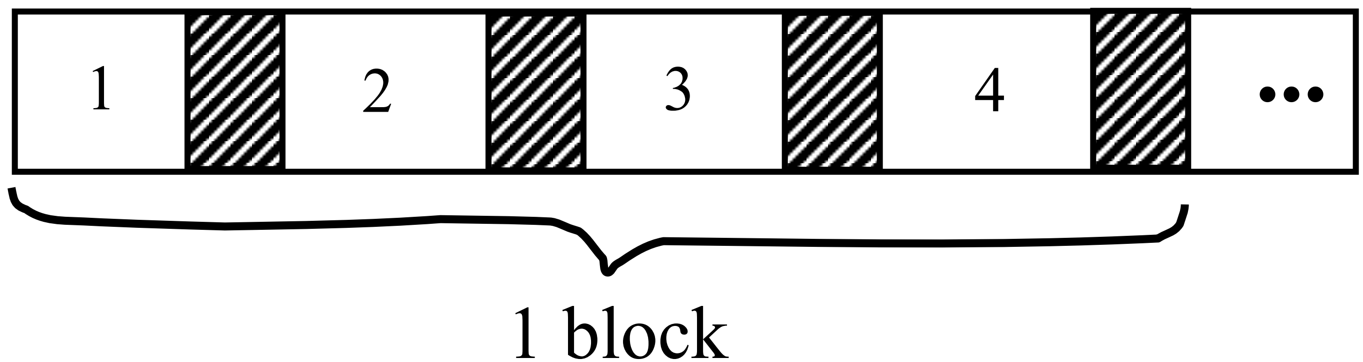
SB = 61.5 x 1000/1024 = 60 MB/sec.

T_1 = Time to read one random block

T_1 = seek + rotational delay + TT

$= 8.5 + (8.33/2) + 0.059 = 12.72$ ms.

Suppose OS deals with 16 KB blocks



$T_4 = 8.5 + (8.33/2) + 0.059*1 + (0.065) * 3 = 12.92$ ms

[Compare to $T_1 = 12.72$ ms]

T_T = Time to read a full track

(start at any block)

$T_T = 8.5 + (0.065/2) + 8.33^* = 16.86$ ms

to get to first block

* Actually, a bit less; do not have to read last gap.

Block Size Selection?

- Big Block $\rightarrow$ Amortize I/O Cost

Unfortunately...

- Big Block $\Rightarrow$ Read in more useless stuff!
  and takes longer to read
  waste of space

Reliability

- Measured by the Mean Time to Failure (MTTF):
  - Length of time by which 50% of a population of disks will have failed catastrophically (head crash, no longer readable)
  - For modern disks, the MTTF is 10 years
  - This means that, on average, after 10 years it will crash
  - We can assume that every year 5% of the disks fail (uniform distribution assumption)
    - Probability that a disk fails in one year P_F=5%=1/20

Disk Arrays

- Redundant Arrays of Inexpensive Disks (RAID)
- Two aims: increase speed and reliability

RAID 0

- Uses "block level striping"
  - Blocks that are consecutive for the OS are distributed evenly across different disks

RAID 0

A1	A2	consecutive blocks: A1-A8
A3	A4	
A5	A6	
A7	A8	

RAID 0

- Improves reading and writing speed
  - With two disks, two blocks can be read at the same time
  - A request for block "A1" would be serviced by disk 1. A simultaneous request for block A3 would have to wait, but a request for A2 could be serviced concurrently
- Reduces reliability: if one disk fails, the data is lost.

RAID 1

- Creates an exact copy (or **mirror**) of a set of data on two or more disks.
- Typically, a RAID 1 array contains two disks
- Improved
  - Reading speed: two blocks can be read at the same time
  - Reliability: if disk 1 crashes, we can use disk 2. We lose the data only if disk 2 crashes while we are changing disk 1 (which can be done in ~3 h)
- Writing speed remains the same

RAID 1

RAID 1	
A1	A1
A2	A2
A3	A3
A4	A4

RAID 4

- Uses block-level striping with a dedicated parity disk.

RAID 4

A1	A2	A3	Ap	Consecutive blocks
B1	B2	B3	Bp	A1-A3,B1-B3,
C1	C2	C3	Cp	C1-C3, D1-D3
D1	D2	D3	Dp	

Parity block

- Bit i of the block in position j on the parity disk is the parity bit of the bits in position i in the blocks in position j in the other disks
- Eg., blocks of one byte, blocks A1-A3

  Disk1 11110000

  Disk2 10101010

  Disk3 00111000

  Disk4 01100010 (parity disk)

RAID 4

- Improves reading time: multiple blocks can be read at the same time
- Improves reliability: if one disk fails, we can reconstruct its content (assuming the others are correct)

RAID 4

- Problem:
  - When writing a block, we need to read and write the parity disk's block
  - This creates a bottleneck

RAID 5

- Uses block-level striping with parity data distributed across all member disks.

RAID 5

A1	A2	A3	Ap
B1	B2	Bp	B3
C1	Cp	C2	C3
Dp	D1	D2	D3

RAID 5

- Reading and reliability as RAID 4
- Writing improved because the parity blocks are not all on one disk