

CS 277: Database System Implementation

Notes 02: Hardware

Arthur Keller

Outline

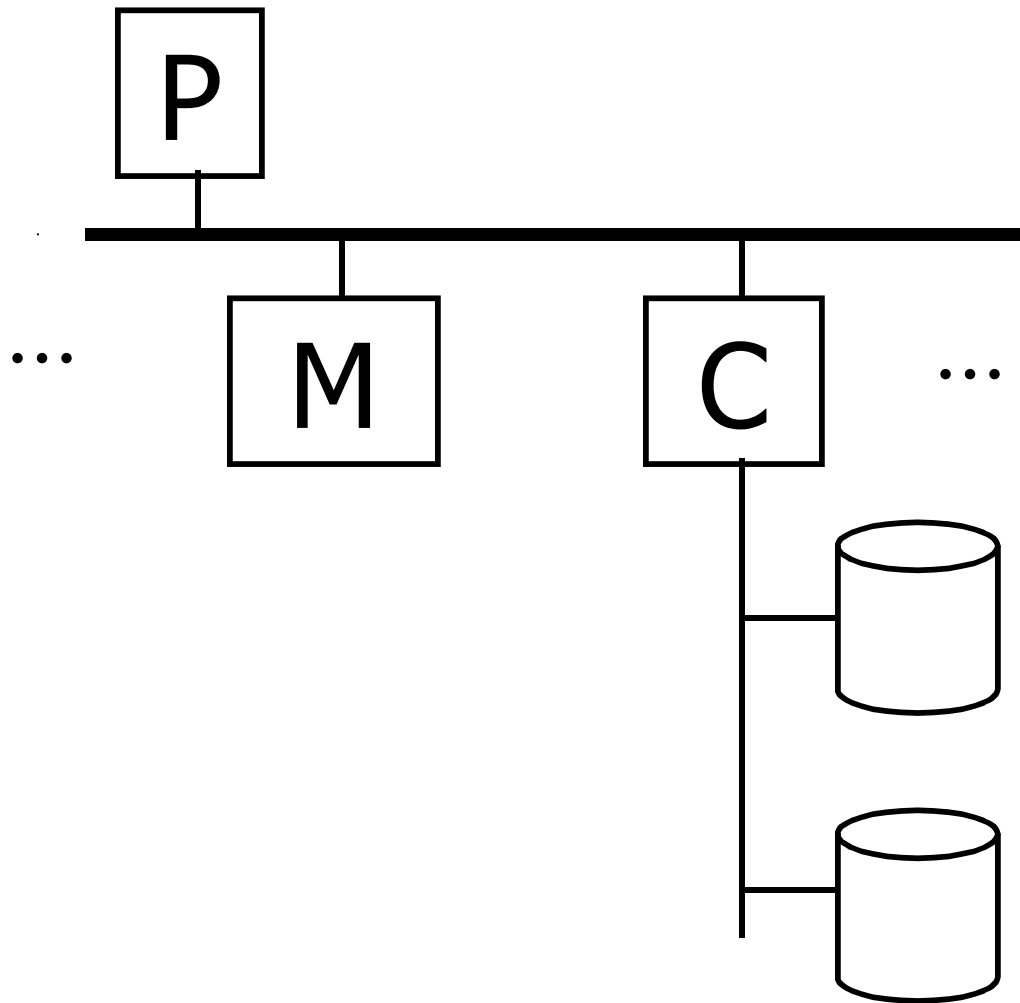
- Hardware: Disks
- Access Times
- Example - Megatron 747
- Optimizations
- Other Topics:
 - Storage costs
 - Using secondary storage
 - Disk failures

Hardware

DBMS

Data Storage





Typical
Computer

Secondary
Storage

Processor

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

Speed: 100 → 500 → 1000 MIPS

Memory

Fast, slow, non-volatile, read-only,...

Access time: 10^{-6} → 10^{-9} sec.

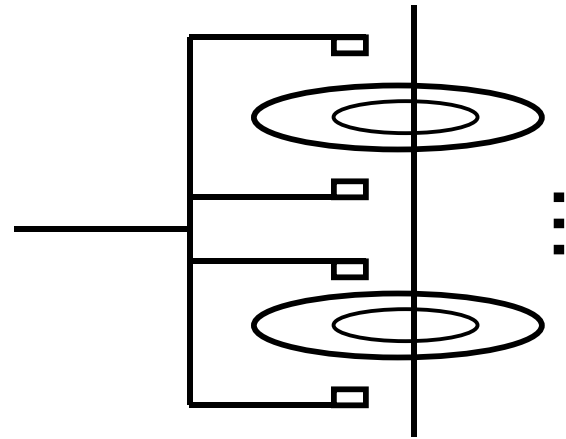
1 μ s → 1 ns

Secondary storage

Many flavors:

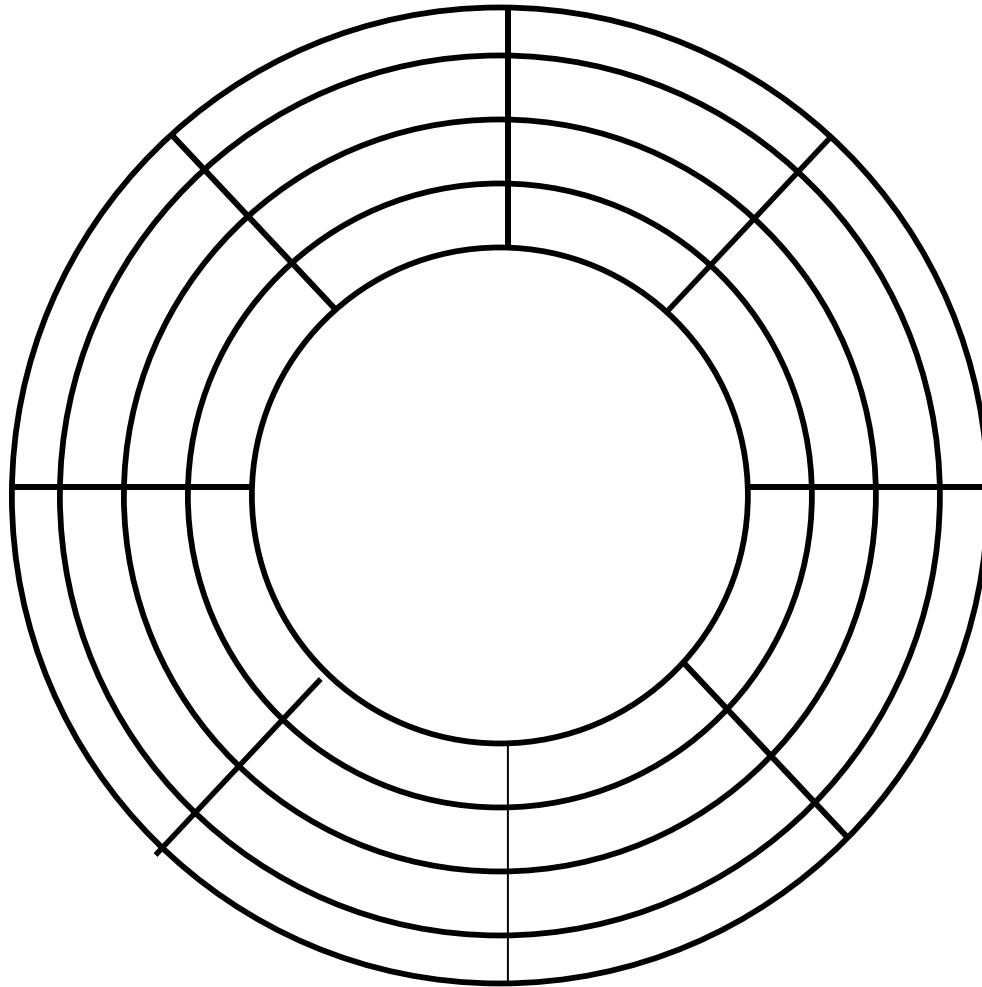
- Disk: Floppy (hard, soft)
Removable Packs
Winchester
Ram disks
Optical, CD-ROM...
Arrays
- Tape Reel, cartridge
Robots

Focus on: “Typical Disk”



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

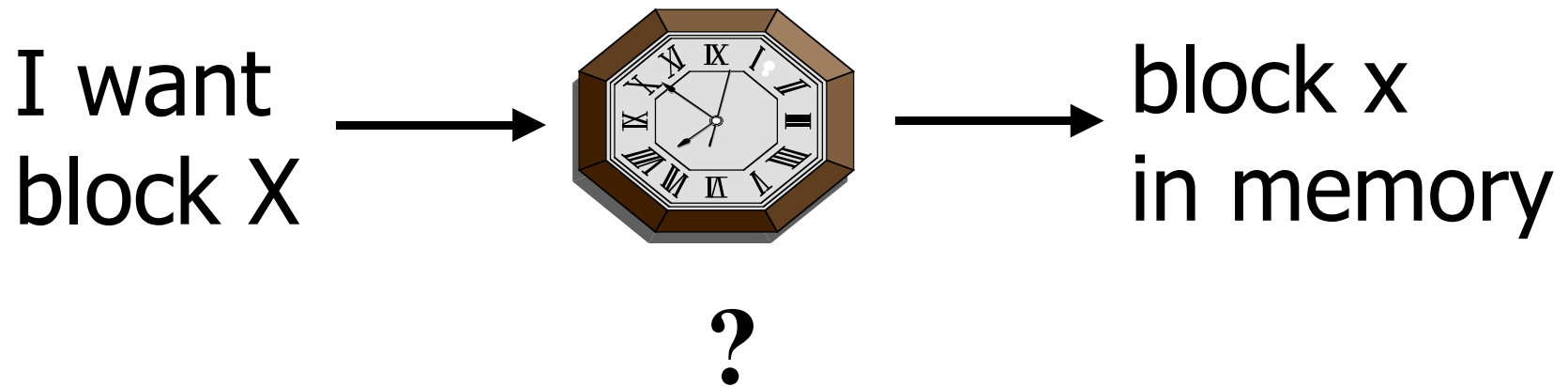
Top View



“Typical” Numbers

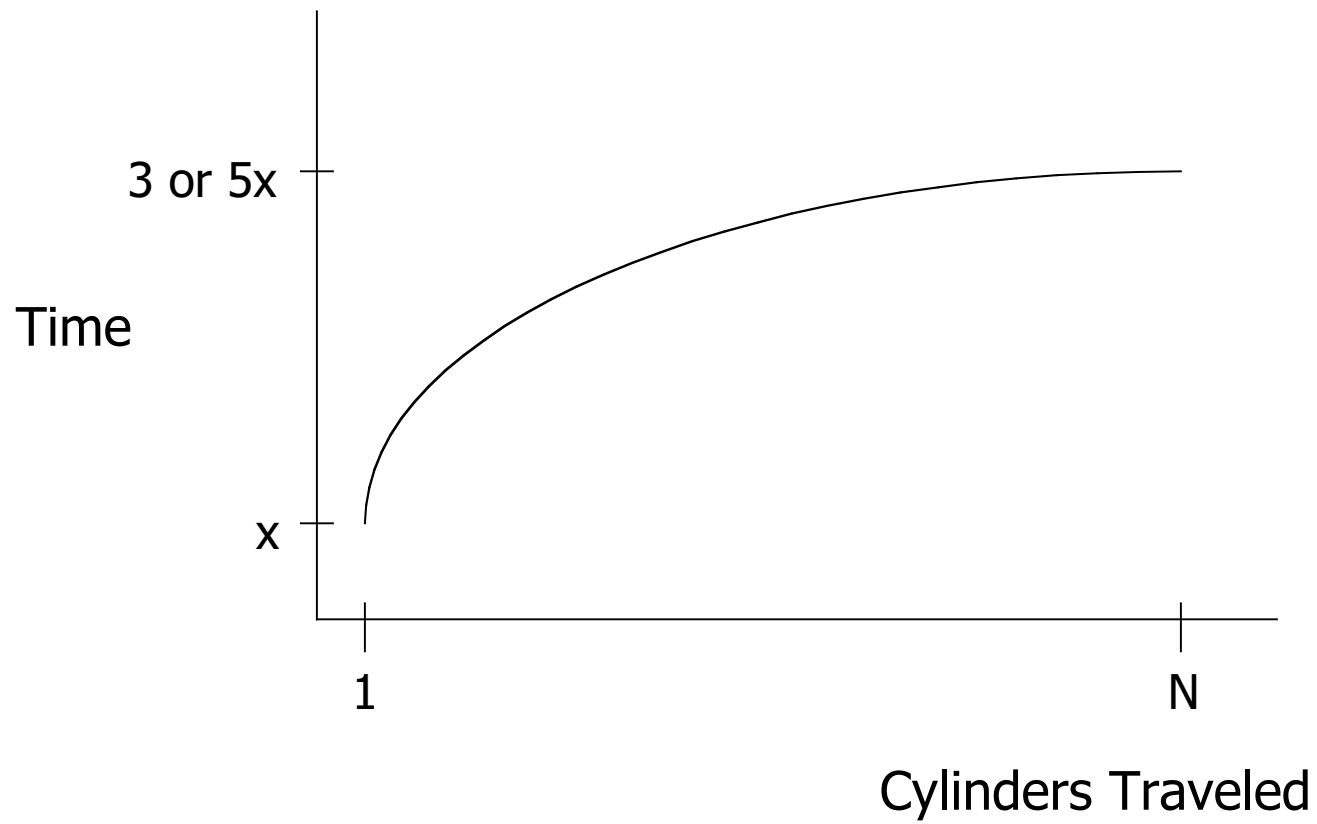
Diameter:	1 inch → 15 inches
Cylinders:	100 → 2000
Surfaces:	1 (CDs) →
(Tracks/cyl)	2 (floppies) → 30
Sector Size:	512B → 50K
Capacity:	360 KB (old floppy) → 70 GB (I use)

Disk Access Time



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

Seek Time

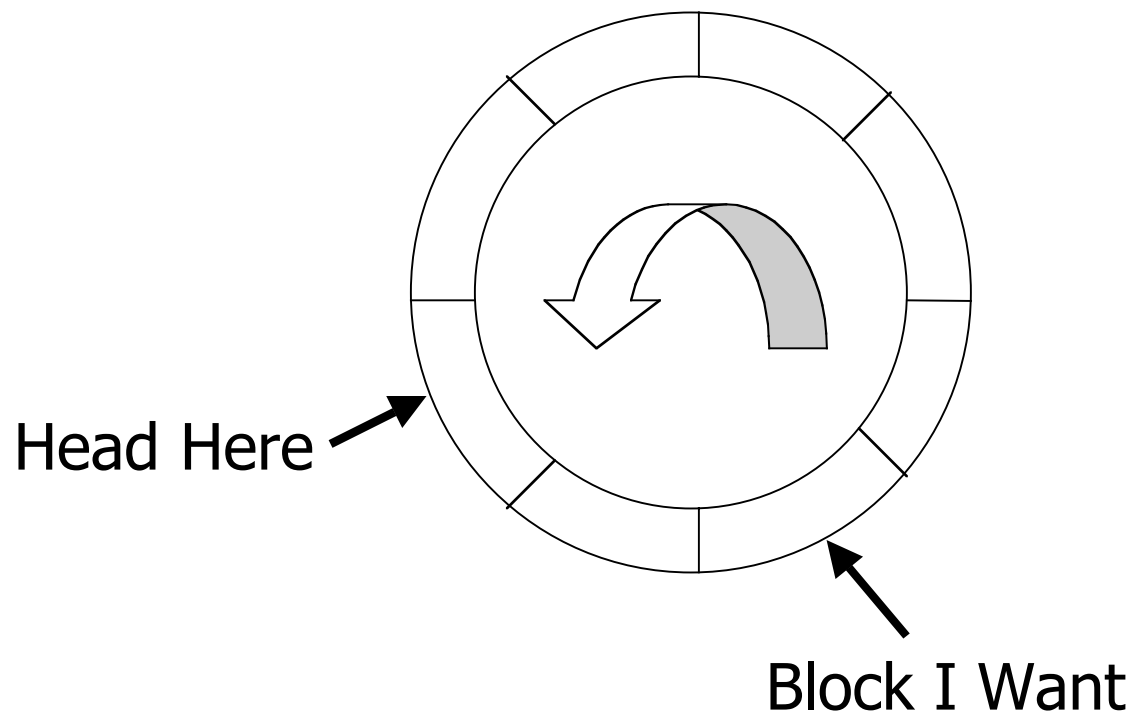


Average Random Seek Time

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

“Typical” S: 10 ms \rightarrow 40 ms

Rotational Delay



Average Rotational Delay

$R = 1/2$ revolution

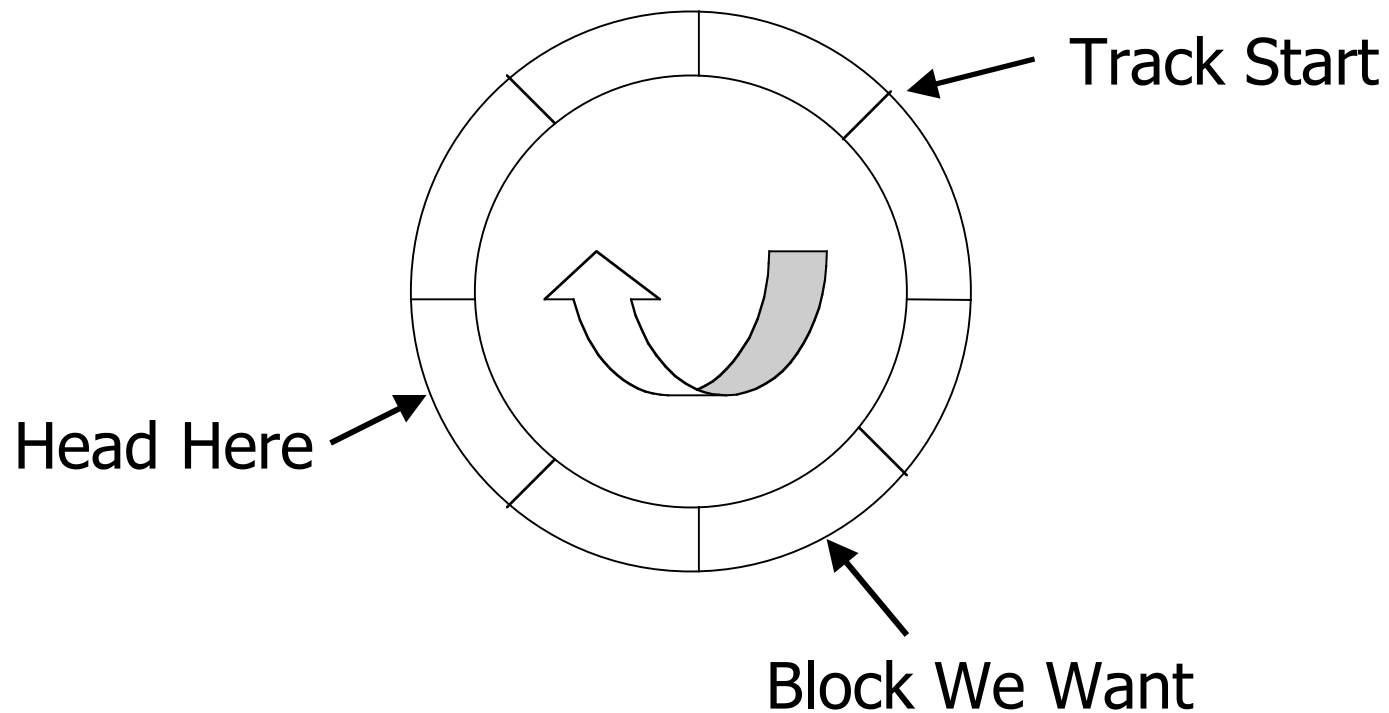
“typical” $R = 8.33$ ms (3600 RPM)

Faster disks now 7200 RPM ($R = 4.17$ ms)

Fastest disks now 10,000 RPM ($R = 3$ ms)

Complication

- May have to wait for start of track before we can read desired block



Transfer Rate: t

- “typical” t: 1 → 3 MB/second
- transfer time: $\frac{\text{block size}}{t}$

Other Delays

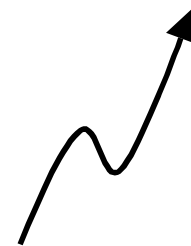
- CPU time to issue I/O
- Contention for controller
- Contention for bus, memory

“Typical” Value: 0

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

If we do things right (e.g., Double Buffer, Stagger Blocks,...)

Time to get block = $\frac{\text{Block Size}}{t}$ + Negligible



- skip gap
- switch track
- once in a while, next cylinder

Rule of Thumb

Random I/O: Expensive
Sequential I/O: Much less

- **Ex:** 1 KB Block
 - » Random I/O: ~ 20 ms.
 - » Sequential I/O: ~ 1 ms.

Cost for Writing similar to Reading

.... unless we want to verify!
need to add (full) rotation + $\frac{\text{Block size}}{t}$

- To Modify a Block?

To Modify Block:

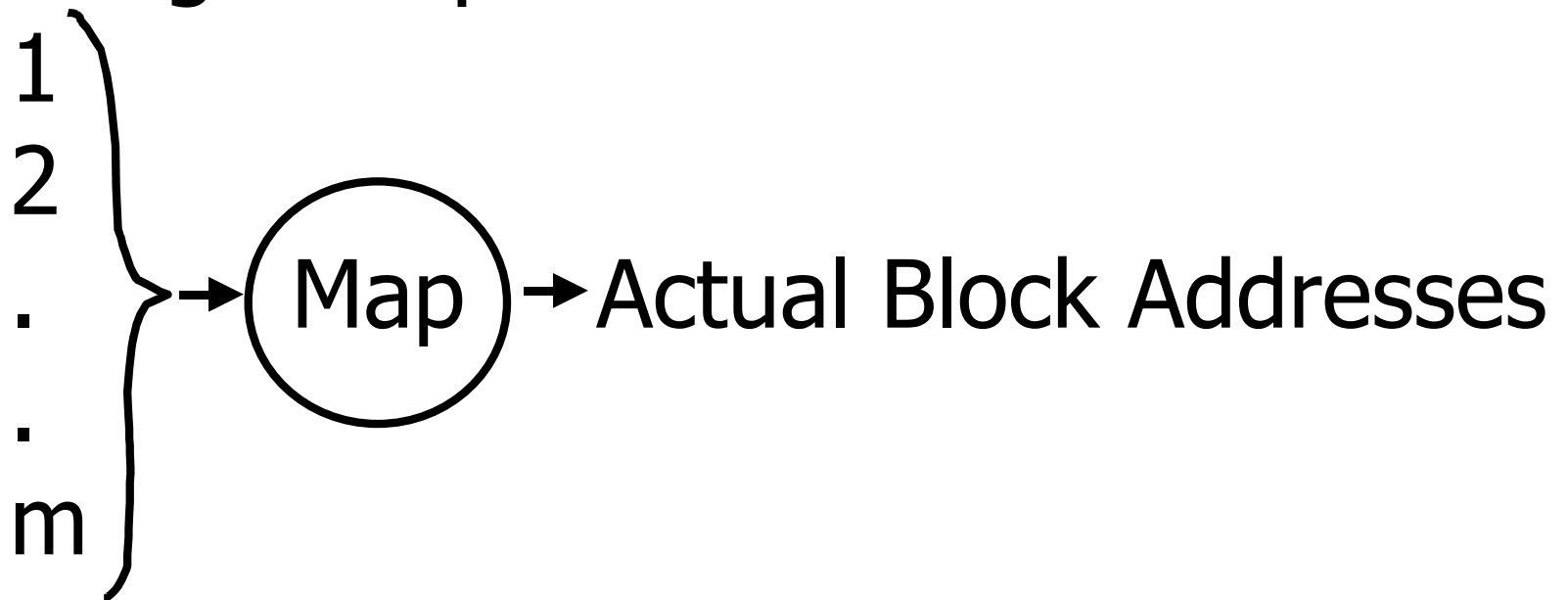
- (a) Read Block
- (b) Modify in Memory
- (c) Write Block
- [(d) Verify?]

Block Address:

- Physical Device
- Cylinder #
- Surface #
- Sector

Complication: Bad Blocks

- Messy to handle
- May map via software to integer sequence



An Example

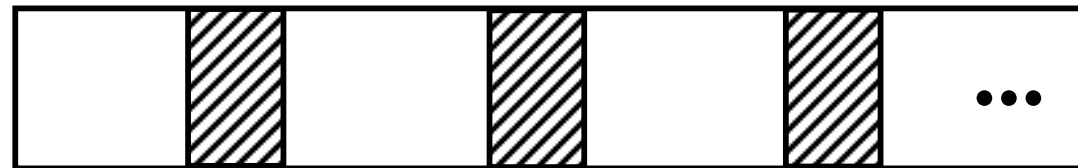
Megatron 747 Disk (old)

- 3.5 in diameter
- 3600 RPM
- 1 surface
- 16 MB usable capacity (16×2^{20})
- 128 cylinders
- seek time: average = 25 ms.
adjacent cyl = 5 ms.

- 1 KB blocks = sectors
- 10% overhead between blocks
- capacity = 16 MB = $(2^{20})16 = 2^{24}$
- # cylinders = 128 = 2^7
- bytes/cyl = $2^{24}/2^7 = 2^{17} = 128$ KB
- blocks/cyl = 128 KB / 1 KB = 128

3600 RPM \rightarrow 60 revolutions / sec
 \rightarrow 1 rev. = 16.66 msec.

One track:



Time over useful data: $(16.66)(0.9) = 14.99$ ms.

Time over gaps: $(16.66)(0.1) = 1.66$ ms.

Transfer time 1 block = $14.99/128 = 0.117$ ms.

Trans. time 1 block+gap = $16.66/128 = 0.13$ ms.

Burst Bandwidth

1 KB in 0.117 ms.

$$BB = 1/0.117 = 8.54 \text{ KB/ms.}$$

or

$$\begin{aligned} BB &= 8.54 \text{ KB/ms} \times 1000 \text{ ms/1sec} \times 1 \text{ MB/1024 KB} \\ &= 8540/1024 = 8.33 \text{ MB/sec} \end{aligned}$$

Sustained bandwidth (over track)
128 KB in 16.66 ms.


$$SB = 128/16.66 = 7.68 \text{ KB/ms}$$

or

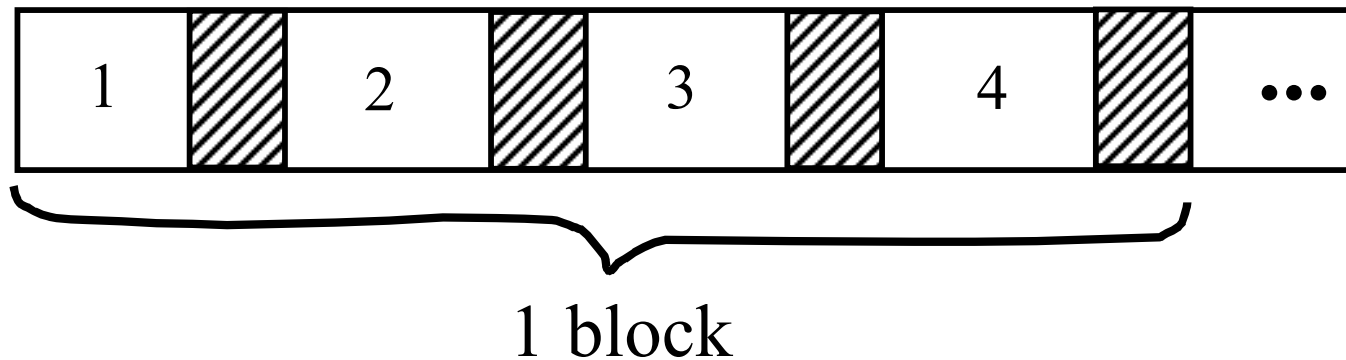
$$SB = 7.68 \times 1000/1024 = 7.50 \text{ MB/sec.}$$

T_1 = Time to read one random block

$$\begin{aligned} T_1 &= \text{seek} + \text{rotational delay} + TT \\ &= 25 + (16.66/2) + .117 = 33.45 \text{ ms.} \end{aligned}$$

 assuming we do not have to wait for track start

Suppose OS deals with 4 KB blocks



$$T_4 = 25 + (16.66/2) + (.117) \times 1 \\ + (.130) \times 3 = 33.83 \text{ ms}$$

[Compare to $T_1 = 33.45 \text{ ms}$]

T_T = Time to read a full track
(start at any block)

$$T_T = 25 + (0.130/2) + 16.66^* = 41.73 \text{ ms}$$



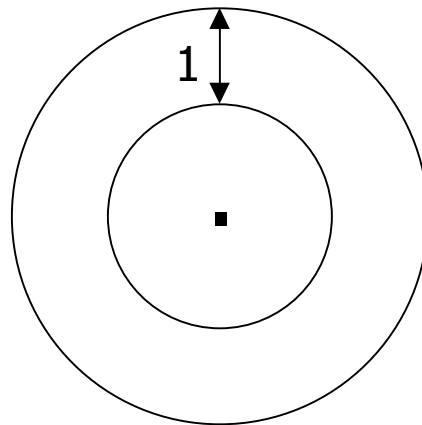
to get to first block

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

The NEW Megatron 747 (Example 11.1 book)

- 8 Surfaces, 3.5 Inch diameter
 - outer 1 inch used
- $2^{13} = 8192$ Tracks/surface
- 256 Sectors/track
- $2^9 = 512$ Bytes/sector

- 8 GB Disk
- If all tracks have 256 sectors
 - Outermost density: 100,000 bits/inch
 - Inner density: 250,000 bits/inch



- Outer third of tracks: 320 sectors
- Middle third of tracks: 256
- Inner third of tracks: 192

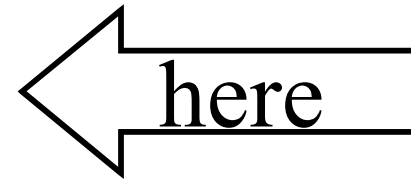
- Density: 114,000 → 182,000 bits/inch

Timing for new Megatron 747 (Ex 11.3)

- Time to read 4096-byte block:
 - MIN: 0.5 ms
 - MAX: 33.5 ms
 - AVE: 14.8 ms

Outline

- Hardware: Disks
- Access Times
- Example: Megatron 747
- Optimizations
- Other Topics
 - Storage Costs
 - Using Secondary Storage
 - Disk Failures



Optimizations (in controller or O.S.)

- Disk Scheduling Algorithms
 - e.g., elevator algorithm
- Track (or larger) Buffer
- Pre-fetch
- Arrays
- Mirrored Disks

Double Buffering

Problem: Have a File

» Sequence of Blocks B1, B2

Have a Program

» Process B1

» Process B2

» Process B3

⋮

Single Buffer Solution

- (1) Read B1 → Buffer
- (2) Process Data in Buffer
- (3) Read B2 → Buffer
- (4) Process Data in Buffer ...

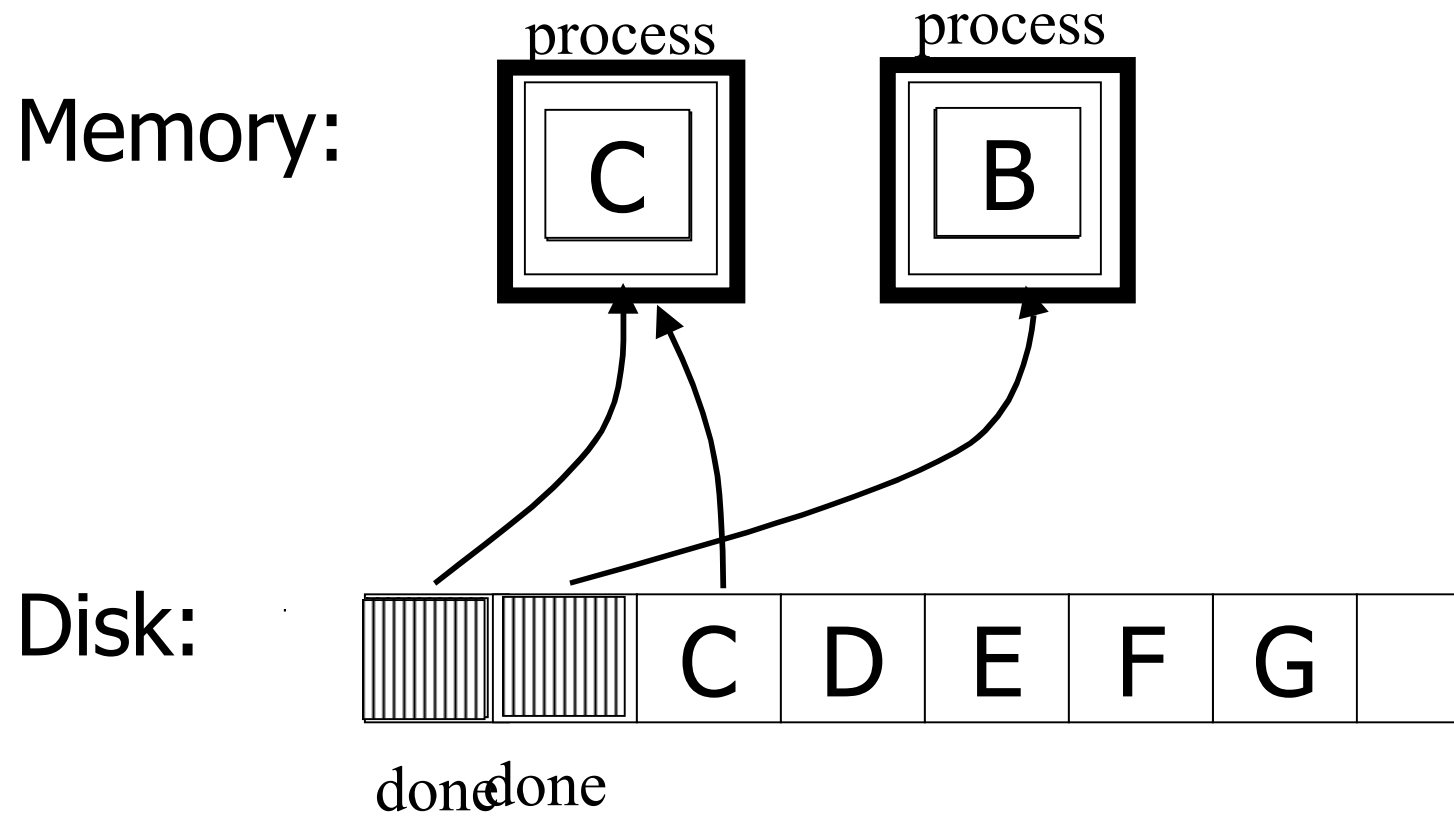
Say P = time to process/block

R = time to read in 1 block

n = # blocks

Single buffer time = $n(P+R)$

Double Buffering



Say $P \geq R$

P = Processing time/block

R = IO time/block

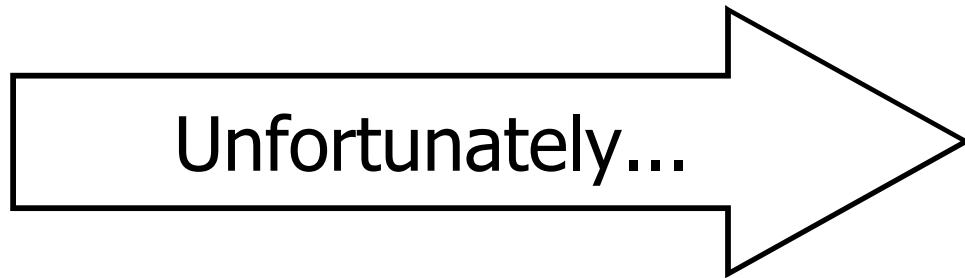
n = # blocks

What is processing time?

- Double buffering time = $R + nP$
- Single buffering time = $n(R+P)$

Block Size Selection?

- Big Block → Amortize I/O Cost

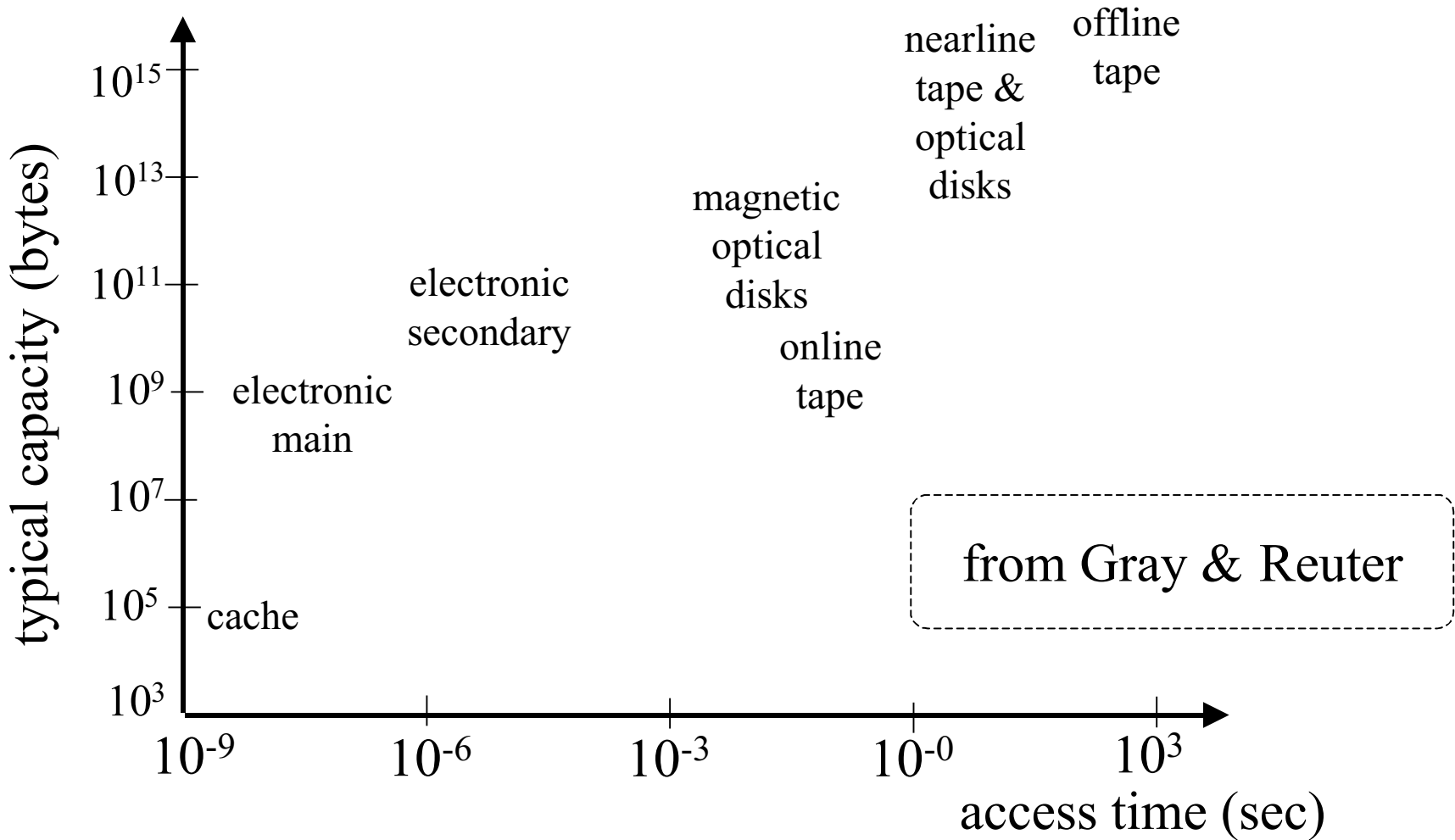


- Big Block ⇒ Read in more useless stuff!
and takes longer to read

Trend

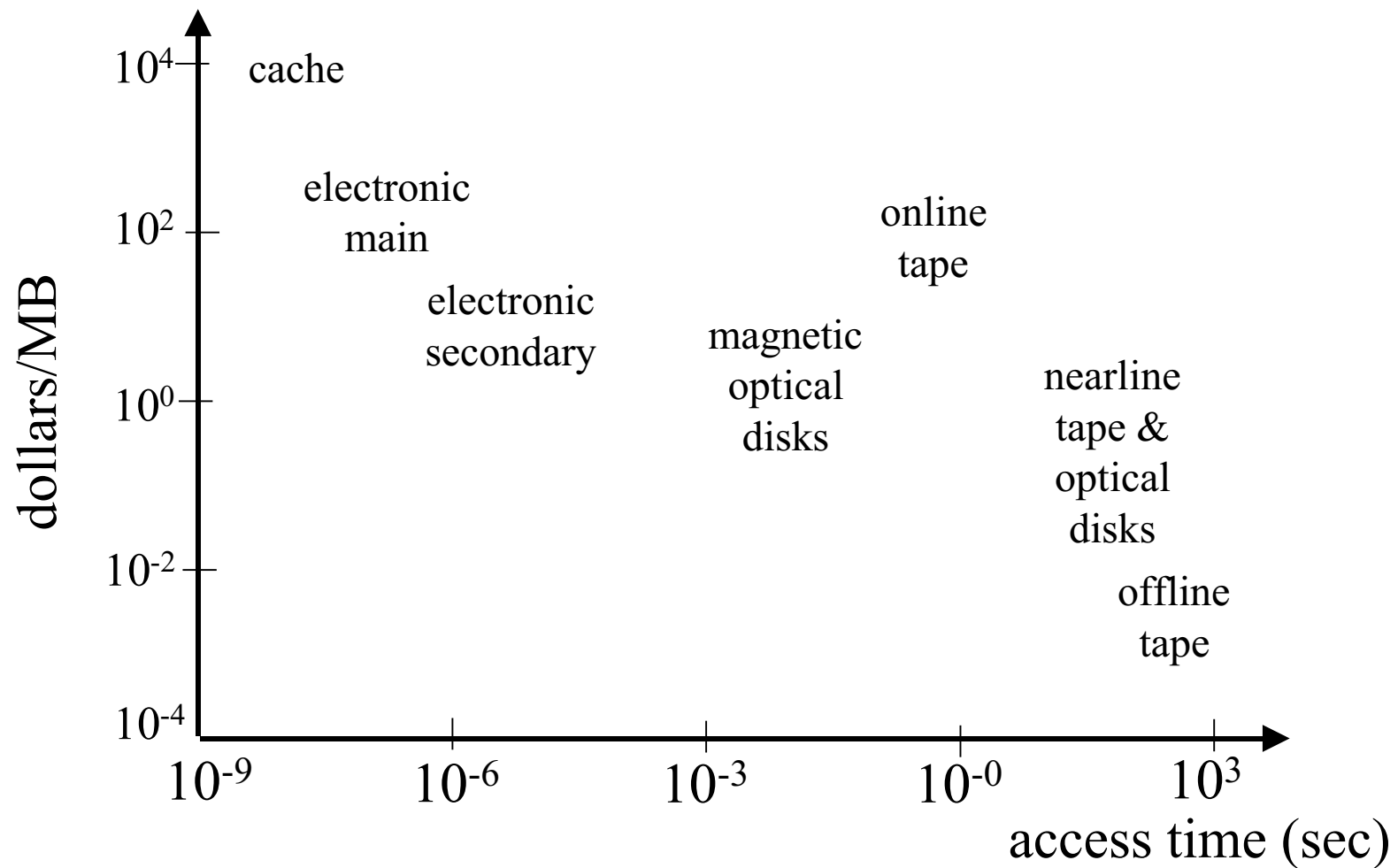
- As memory prices drop,
blocks get bigger ...

Storage Cost



Storage Cost

from Gray & Reuter



Using secondary storage effectively

(Sec. 11.3)

- Example: Sorting data on disk
- Conclusion:
 - I/O costs dominate
 - Design algorithms to reduce I/O
- Also: How big should blocks be?

Disk Failures (Sec 11.5)

- Partial → Total
- Intermittent → Permanent

Coping with Disk Failures

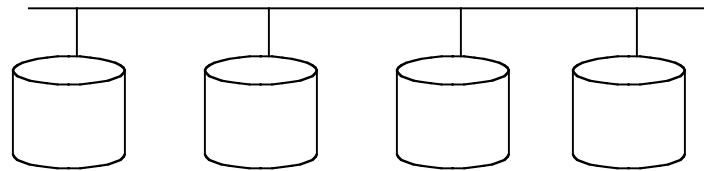
- Detection
 - e.g., Checksum
- Correction
 - ⇒ Redundancy

At what level do we cope?

- Single Disk
 - e.g., Error Correcting Codes
- Disk Array



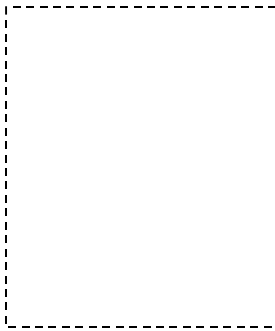
Logical



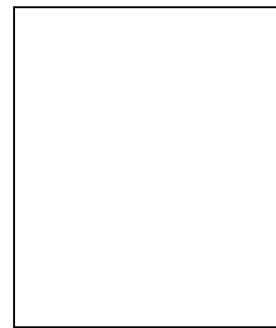
Physical

→ Operating System

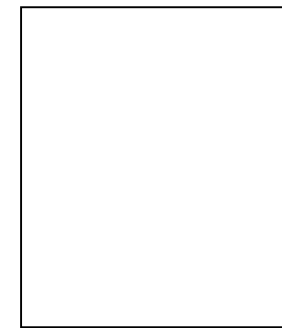
e.g., Stable Storage



Logical Block



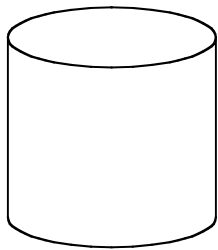
Copy A



Copy B

→ Database System

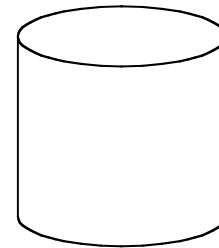
- e.g.,



Current DB



Log



Last week's DB

Summary

- Secondary storage, mainly disks
- I/O times
- I/Os should be avoided,
especially random ones.....

Outline

- Hardware: Disks
- Access Times
- Example: Megatron 747
- Optimizations
- Other Topics
 - Storage Costs
 - Using Secondary Storage
 - Disk Failures

