CS 525: Advanced Database Organization
02: Hardware

Boris Glavic

Slides: adapted from a course taught by Hector Garcia-Molina, Stanford InfoLab
Outline

• Hardware: Disks
• Access Times
• Example - Megatron 747
• Optimizations
• Other Topics:
  – Storage costs
  – Using secondary storage
  – Disk failures
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: $10^{-6} \rightarrow 10^{-9}$ sec.
$1 \mu s \rightarrow 1 \text{ ns}$
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 $\rightarrow$ 15 inches
Cylinders: 100 $\rightarrow$ 2000
Surfaces: 1 (CDs) $\rightarrow$
(Tracks/cyl) 2 (floppies) $\rightarrow$ 30
Sector Size: 512B $\rightarrow$ 50K
Capacity: 360 KB (old floppy) $\rightarrow$ 1 TB (I use)
Disk Access Time

I want block X

? 

block x in memory
Time = Seek Time + Rotational Delay + Transfer Time + Other
Seek Time

Time

3 or 5x

Cylinders Traveled

x

1  N
Average Random Seek Time

\[ S = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \text{SEEKTIME (i → j)}_{j \neq i}}{N(N-1)} \]
Average Random Seek Time

\[ S = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} \text{SEEKTIME} (i \rightarrow j)}{N(N-1)} \]

“Typical” \( S \): 10 ms \( \rightarrow \) 40 ms
Rotational Delay

Head Here

Block I Want
Average Rotational Delay

R = 1/2 revolution

“typical” R = 8.33 ms (3600 RPM)
Transfer Rate: $t$

- “typical” $t$: 10’s $\rightarrow$ 100’s MB/second
- transfer time: block size $t$
Other Delays

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

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

“Typical” Value: 0
Other Delays (now and near future)

- Increasing amount of parallelism
- Contention can become a problem
- -> need rethink approach to scale
• So far: Random Block Access
• What about: Reading “Next” block?
If we do things right (e.g., Double Buffer, Stagger Blocks...)

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

- 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 + **Block size**

```
• **To Modify a Block?**
• 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

1
2
. .
M

\rightarrow \text{Map} \rightarrow \text{Actual Block Addresses}
An Example  Megatron 747 Disk (old)

• 3.5 in diameter
• 3600 RPM
• 1 surface
• 16 MB usable capacity (16 \times 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:
3600 RPM → 60 revolutions / sec
→ 1 rev. = 16.66 msec.

One track:

```
... \[\text{shaded} \] \[\text{white} \] \[\text{shaded} \] \[\text{white} \] ...
```

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 KB/ms.

or

BB = 8.54 KB/ms × 1000 ms/1 sec × 1 MB/1024 KB
   = 8540/1024 = 8.33 MB/sec
**Sustained bandwidth** (over track)

128 KB in 16.66 ms.

\[
SB = \frac{128}{16.66} = 7.68 \text{ KB/ms}
\]

or

\[
SB = 7.68 \times \frac{1000}{1024} = 7.50 \text{ MB/sec.}
\]
$T_1 = \text{Time to read one random block}$

$T_1 = \text{seek} + \text{rotational delay} + \text{TT}$

$= 25 + (16.66/2) + .117 = 33.45 \text{ ms.}$
Suppose OS deals with 4 KB blocks

\[ T_4 = 25 + \left(\frac{16.66}{2}\right) + (0.117) \times 1 + (0.130) \times 3 = 33.83 \text{ ms} \]

[Compare to \( T_1 = 33.45 \text{ ms} \)]
\[ T_T = \text{Time to read a full track} \]
\[ (\text{start at any block}) \]
\[ T_T = 25 + \left( \frac{0.130}{2} \right) + 16.66^* = 41.73 \text{ ms} \]

\[ \sqrt{\text{to get to first block}} \]

\[ * \text{Actually, a bit less; do not have to read last gap.} \]
The **NEW** Megatron 747

- 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 2.3)

- 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
- On Disk Cache
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 $P = \text{time to process/block}$
$R = \text{time to read in 1 block}$
$n = \# \text{blocks}$

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

Memory:

Disk:

- A
- B
- C
- D
- E
- F
- G

process
Double Buffering

Memory:

Disk:

process

done
Double Buffering

Memory:

Disk:

process

done
Double Buffering

Memory:

Disk:

process C

process B

done done done
Say $P \geq R$

<table>
<thead>
<tr>
<th>$P$ = Processing time/block</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R$ = IO time/block</td>
</tr>
<tr>
<td>$n$ = # blocks</td>
</tr>
</tbody>
</table>

What is processing time?
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 = $n(R+P)$
Disk Arrays

- RAIDs (various flavors)
- Block Striping
- Mirrored

logically one disk
On Disk Cache

Diagram:
- P
- M
- C
- Two cache nodes connected to C

Block Size Selection?

• Big Block → Amortize I/O Cost, Less Management Overhead

Unfortunately...

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

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

- Cache (typical capacity: $10^3$ bytes)
- Electronic main
- Electronic secondary
- Magnetic optical disks
- Online tape
- Nearline tape & optical disks
- Offline tape

From Gray & Reuter
Storage Cost

from Gray & Reuter

<table>
<thead>
<tr>
<th>Storage Category</th>
<th>access time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cache</td>
<td>10^{-2}</td>
</tr>
<tr>
<td>main electronic</td>
<td>10^{-1}</td>
</tr>
<tr>
<td>secondary electronic</td>
<td>10^0</td>
</tr>
<tr>
<td>magnetic optical disks</td>
<td>10^{-1}</td>
</tr>
<tr>
<td>online tape</td>
<td>10^1</td>
</tr>
<tr>
<td>nearline tape &amp; optical disks</td>
<td>10^2</td>
</tr>
<tr>
<td>offline tape</td>
<td>10^3</td>
</tr>
</tbody>
</table>

dollars/MB

from Gray & Reuter
Using secondary storage effectively

- Example: Sorting data on disk
- Conclusion:
  - I/O costs dominate
  - Design algorithms to reduce I/O

- Also: How big should blocks be?
Five Minute Rule

• THE 5 MINUTE RULE FOR TRADING MEMORY FOR DISC ACCESSES
  Jim Gray & Franco Putzolu
  May 1985

• The Five Minute Rule, Ten Years Later
  Goetz Graefe & Jim Gray
  December 1997
Five Minute Rule

• Say a page is accessed every X seconds
• CD = cost if we keep that page on disk
  – $D = cost of disk unit
  – I = numbers IOs that unit can perform per second
  – In X seconds, unit can do XI IOs
  – So  CD = $D / XI
Five Minute Rule

- Say a page is accessed every X seconds
- CM = cost if we keep that page on RAM
  - $M = cost of 1 MB of RAM
  - P = numbers of pages in 1 MB RAM
  - So CM = $M / P
Five Minute Rule

• Say a page is accessed every X seconds
• If CD is smaller than CM,
  – keep page on disk
  – else keep in memory
• Break even point when CD = CM, or

\[ X = \frac{\$D \cdot P}{\$M} \]
Using ‘97 Numbers

- $P = 128$ pages/MB (8KB pages)
- $I = 64$ accesses/sec/disk
- $D = 2000$ dollars/disk (9GB + controller)
- $M = 15$ dollars/MB of DRAM

- $X = 266$ seconds (about 5 minutes)
  (did not change much from 85 to 97)
Disk Failures

- Partial $\rightarrow$ Total
- Intermittent $\rightarrow$ 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
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 + formulas
  – Sequential vs. random
• I/Os should be avoided, especially random ones…..
• OS optimizations
• Disk errors
Outline

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

• Disk Access is the main limiting factor

• However, to implement fast DBMS
  – need to understand other parts of the hardware
    • Memory hierarchy
    • CPU architecture: pipelining, vector instructions, OOE, ...
    • SSD storage
  – need to understand how OS manages hardware
    • File access, VM, Buffering, ...
Memory Hierarchy

- **CPU Register** (< 1KB, 1 cycle)
- **L1 Cache** (10 KB’s, few cycles)
- **L2 Cache** (e.g., 512 KB, 2-10 x L1)
- **L3 Cache** (MB)
- **Main Memory** (GB, 100’s cycles)
Memory Hierarchy

- **Compare**: Disk vs. Main Memory
- Reduce accesses to main memory
- Cache conscious algorithms
Increasing Amount of Parallelism

- Contention on, e.g., Memory
- NUMA
- Algorithmic Challenges
  - How to parallelize algorithms?
  - Sometime: Completely different approach required
  - -> Rewrite large parts of DBMS
New Trend: Software/Hardware Co-design

- Actually, revived trend: database machines (80’s)
- New goals: power consumption
- Design specific hardware and write special software for it
- E.g., Oracle Exadata, Oracle Labs