



Announcement:
**FAST: A New Sampling-
Based Algorithm for
Discovering Association Rules**
Prof. Peter Scheuermann
Northwestern University

Today (Monday, September 23, 2002)
3:30-4:30pm
CS 111

Chris Clifton - CS541 1



CS 541
Implementing a Database System

September 23, 2002
*Slides adapted from those developed by Stanford
University Prof. Hector Garcia-Molina*

Chris Clifton - CS541 2

Isn't Implementing a Database System Simple?

Relations \Rightarrow Statements \Rightarrow Results

Fall 2002

Chris Clifton - CS541

3

Introducing the

MEGATRON 3000

Database Management System

- The latest from Megatron Labs
- Incorporates latest relational technology
- UNIX compatible

Fall 2002

Chris Clifton - CS541

4

Megatron 3000 Implementation Details

! First sign non-disclosure agreement **!**

Megatron 3000 Implementation Details

- Relations stored in files (ASCII)
e.g., relation R is in /usr/db/R

```
Smith # 123 # CS  
Jones # 522 # EE  
⋮
```

Megatron 3000 Implementation Details

- Directory file (ASCII) in /usr/db/directory

```
R1 # A # INT # B # STR ...  
R2 # C # STR # A # INT ...  
:  
:
```

Fall 2002

Chris Clifton - CS541

7

Megatron 3000 Sample Sessions

```
% MEGATRON3000  
  Welcome to MEGATRON 3000!  
&  
:  
:  
& quit  
%
```

Fall 2002

Chris Clifton - CS541

8

Megatron 3000 Sample Sessions

```
& select *
  from R #

  Relation R
  A      B      C
  SMITH   123    CS

&
```

Megatron 3000 Sample Sessions

```
& select A,B
  from R,S
  where R.A = S.A and S.C > 100 #

  A      B
  123    CAR
  522    CAT

&
```

Megatron 3000 Sample Sessions

```
& select *  
  from R | LPR #  
&
```

Result sent to LPR (printer).

Megatron 3000 Sample Sessions

```
& select *  
  from R  
  where R.A < 100 | T #  
&
```

New relation T created.

Megatron 3000

- To execute “**select * from R where condition**”:
 - (1) Read dictionary to get R attributes
 - (2) Read R file, for each line:
 - (a) Check condition
 - (b) If OK, display

Fall 2002

Chris Clifton - CS541

13

Megatron 3000

- To execute “**select * from R
where condition | T**”:
 - (1) Process select as before
 - (2) Write results to new file T
 - (3) Append new line to dictionary

Fall 2002

Chris Clifton - CS541

14

Megatron 3000

- To execute “**select A,B from R,S where condition**”:
 - (1) Read dictionary to get R,S attributes
 - (2) Read R file, for each line:
 - (a) Read S file, for each line:
 - (i) Create join tuple
 - (ii) Check condition
 - (iii) Display if OK

What's wrong with the Megatron 3000 DBMS?

What's wrong with the Megatron 3000 DBMS?

- Tuple layout on disk
 - e.g., - Change string from 'Cat' to 'Cats' and we have to rewrite file
 - ASCII storage is expensive
 - Deletions are expensive

Fall 2002

Chris Clifton - CS541

17

What's wrong with the Megatron 3000 DBMS?

- Search expensive; no indexes
 - e.g., - Cannot find tuple with given key quickly
 - Always have to read full relation

Fall 2002

Chris Clifton - CS541

18

What's wrong with the Megatron 3000 DBMS?

- Brute force query processing

e.g., **select ***
from R, S
where R.A = S.A and S.B > 1000

- Do select first?
- More efficient join?

What's wrong with the Megatron 3000 DBMS?

- No buffer manager

e.g., Need caching

What's wrong with the Megatron 3000 DBMS?

- No concurrency control

Fall 2002

Chris Clifton - CS541

21

What's wrong with the Megatron 3000 DBMS?

- No reliability
 - e.g., - Can lose data
 - Can leave operations half done

Fall 2002

Chris Clifton - CS541

22

What's wrong with the Megatron 3000 DBMS?

- No security
 - e.g., - File system insecure
 - File system security is coarse

Fall 2002

Chris Clifton - CS541

23

What's wrong with the Megatron 3000 DBMS?

- No application program interface (API)
 - e.g., How can a payroll program get at the data?

Fall 2002

Chris Clifton - CS541

24

What's wrong with the Megatron 3000 DBMS?

- Cannot interact with other DBMSs.

Fall 2002

Chris Clifton - CS541

25

What's wrong with the Megatron 3000 DBMS?

- Poor dictionary facilities

Fall 2002

Chris Clifton - CS541

26

What's wrong with the Megatron 3000 DBMS?

- No GUI

Fall 2002

Chris Clifton - CS541

27

What's wrong with the Megatron 3000 DBMS?

- Lousy salesman!!

Fall 2002

Chris Clifton - CS541

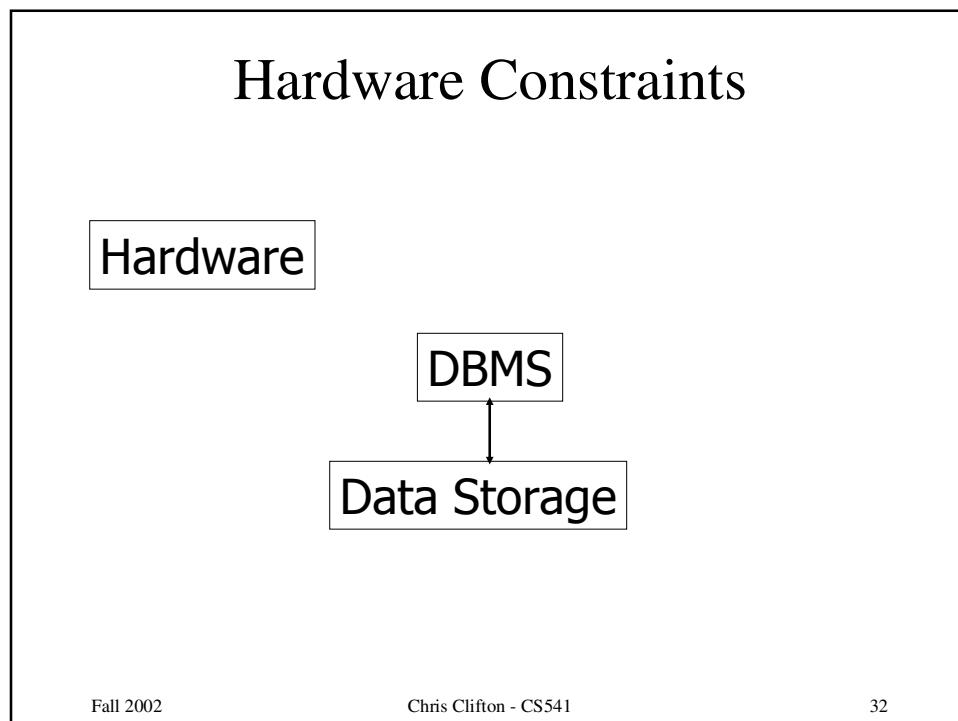
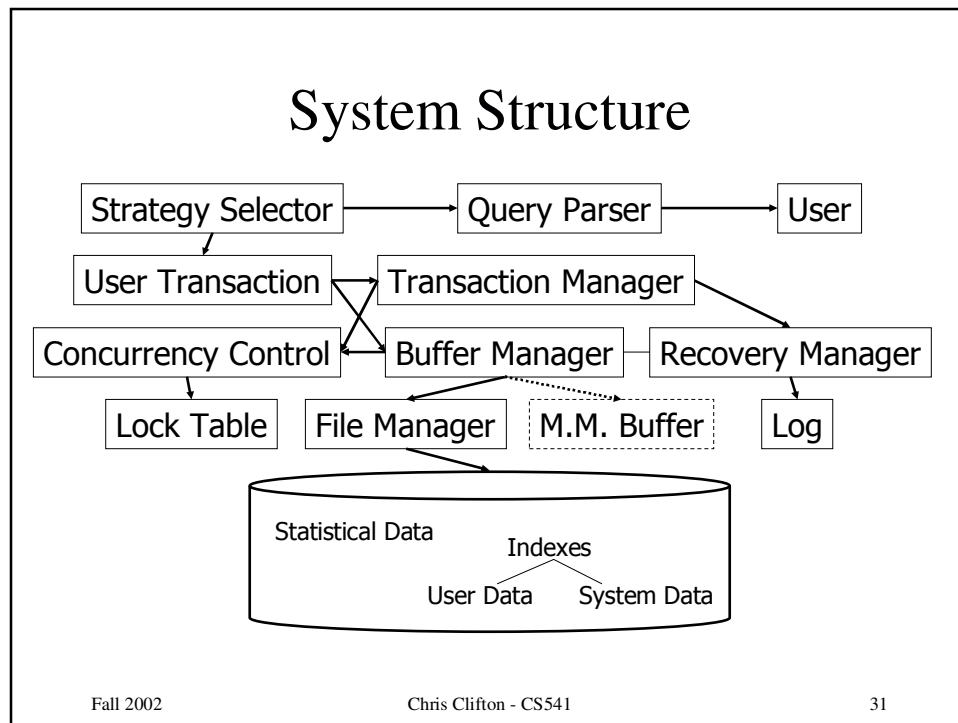
28

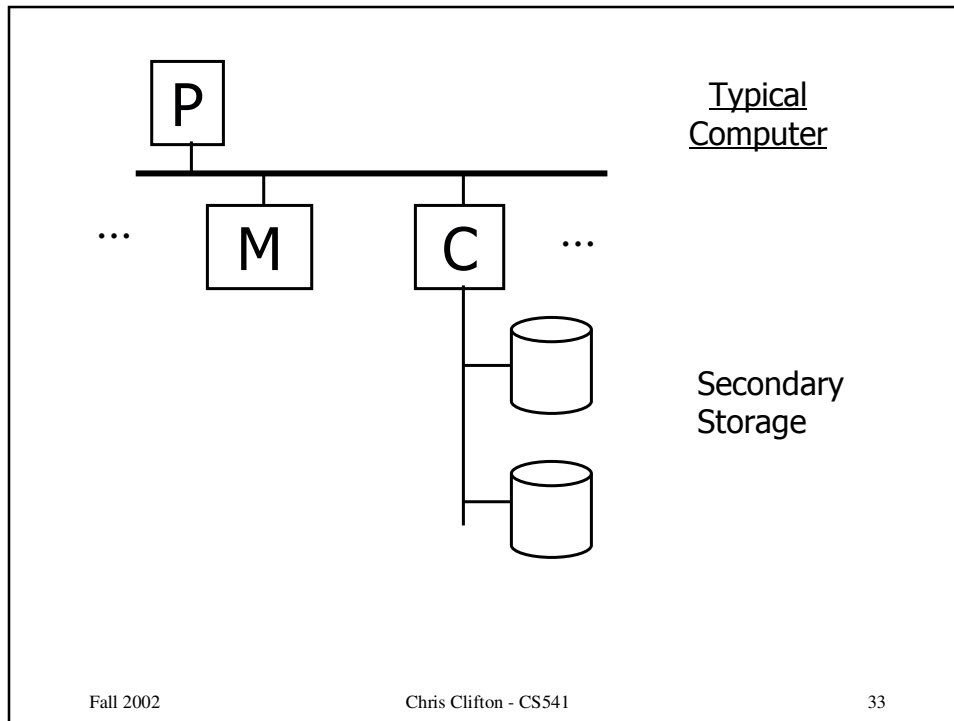
What do we need to know?

- **File & System Structure**
Records in blocks, dictionary, buffer management,...
- **Indexing & Hashing**
B-Trees, hashing,...
- **Query Processing**
Query costs, join strategies,...
- **Crash Recovery**
Failures, stable storage,...

What do we need to know?

- **Concurrency Control**
Correctness, locks,...
- **Transaction Processing**
Logs, deadlocks,...
- **Security & Integrity**
Authorization, encryption,...
- **Distributed Databases**
Interoperation, distributed recovery,...





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:

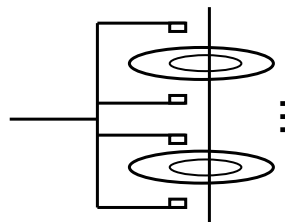
- Disk: Floppy (hard, soft)
Removable Packs
Winchester
Ram disks
Optical, CD-ROM...
Arrays
- Tape Reel, cartridge
Robots

Fall 2002

Chris Clifton - CS541

35

Focus on: "Typical Disk"



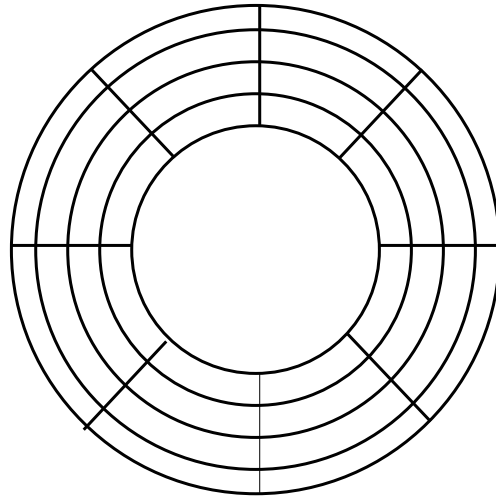
Terms: Platter, Head, Actuator
Cylinder, Track
Sector (physical),
Block (logical), Gap

Fall 2002

Chris Clifton - CS541

36

Top View



Fall 2002

Chris Clifton - CS541

37

"Typical" Numbers

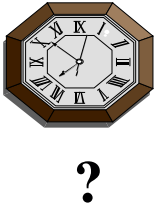
Diameter: 1 inch → 15 inches
Cylinders: 100 → 2000
Surfaces: 1 (CDs) →
(Tracks/cyl) 2 (floppies) → 30
Sector Size: 512B → 50K
Capacity: 360 KB (old floppy)
→ 30 GB (I use)

Fall 2002

Chris Clifton - CS541

38

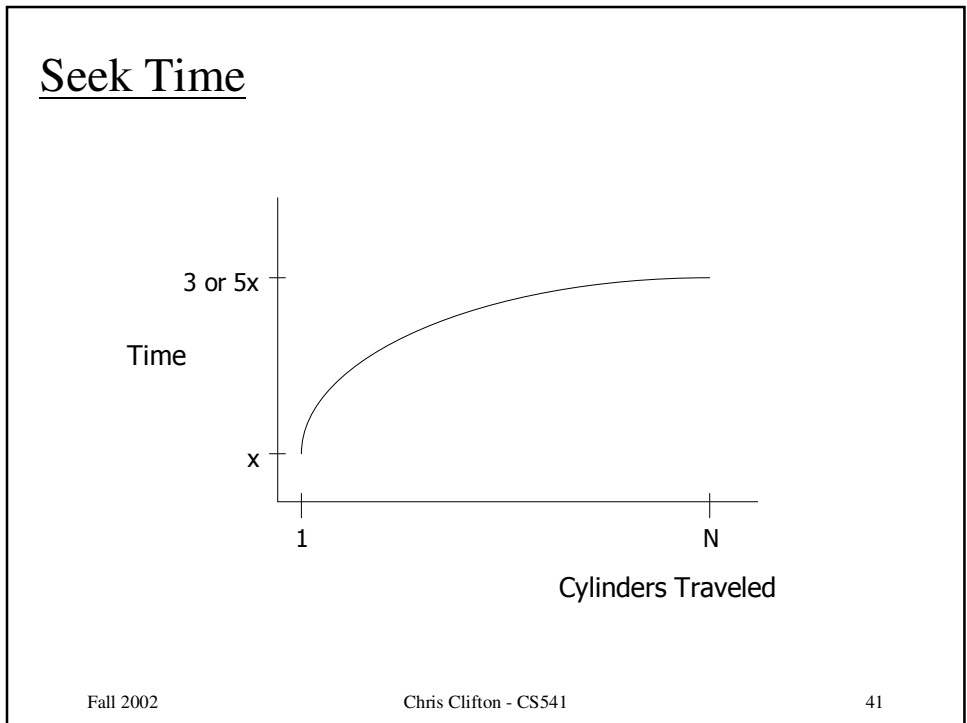
Disk Access Time

I want block X →  → block x in memory

Fall 2002 Chris Clifton - CS541 39

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

Fall 2002 Chris Clifton - CS541 40



Average Random Seek Time

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

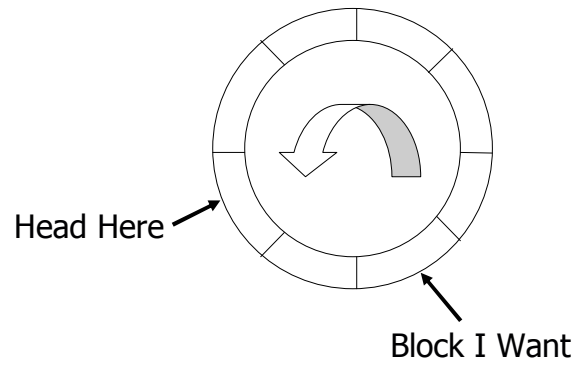
“Typical” S: 10 ms → 40 ms

Fall 2002

Chris Clifton - CS541

42

Rotational Delay



Fall 2002

Chris Clifton - CS541

43

Average Rotational Delay

$R = 1/2$ revolution

"typical" $R = 8.33$ ms (3600 RPM)

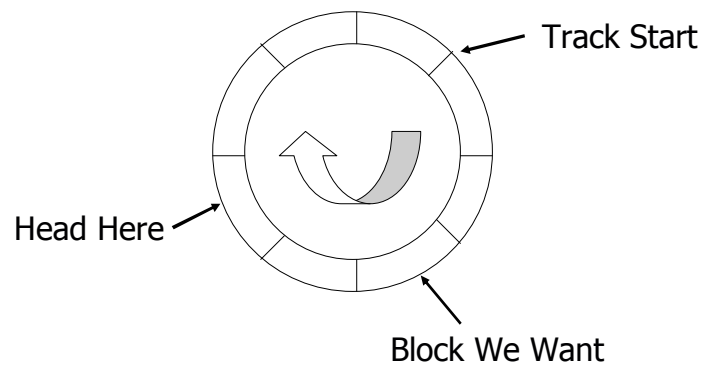
Fall 2002

Chris Clifton - CS541

44

Complication

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



Fall 2002

Chris Clifton - CS541

45

Transfer Rate: t

- “typical” t : 1 → 3 MB/second
- transfer time: block size

 t

Fall 2002

Chris Clifton - CS541

46

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 = $\frac{\text{Block Size}}{\text{t}}$ + 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
_____ t

Fall 2002

Chris Clifton - CS541

51

- To Modify a Block?

To Modify Block:

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

Fall 2002

Chris Clifton - CS541

52

Block Address:

- Physical Device
- Cylinder #
- Surface #
- Sector

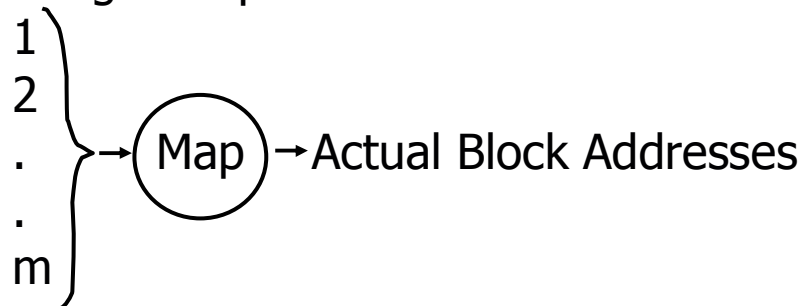
Fall 2002

Chris Clifton - CS541

53

Complication: Bad Blocks

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



Fall 2002

Chris Clifton - CS541

54

An Example Megatron 747 Disk (old)

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

Fall 2002

Chris Clifton - CS541

55

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

Fall 2002

Chris Clifton - CS541

56

3600 RPM 60 revolutions / sec
 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 Bandwith

_____ 1 KB in 0.117 ms.

BB = $1/0.117 = 8.54$ KB/ms.

or

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

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

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

or

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

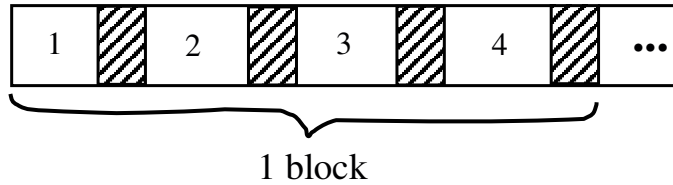
T_1 = Time to read one random block

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

$$= 25 + (16.66/2) + .117 = 33.45 \text{ ms.}$$

↙
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 $T_1 = 33.45 \text{ ms}$]

Fall 2002

Chris Clifton - CS541

61

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.

Fall 2002

Chris Clifton - CS541

62

The NEW Megatron 747 (Example 2.1 book)

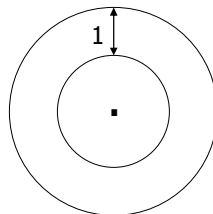
- 8 Surfaces, 3.5 Inch diameter
 - ◆ outer 1 inch used
- $2^{13} = 8192$ Tracks/surface
- 256 Sectors/track
- $2^9 = 512$ Bytes/sector

Fall 2002

Chris Clifton - CS541

63

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



Fall 2002

Chris Clifton - CS541

64

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

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

Fall 2002

Chris Clifton - CS541

65

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

Fall 2002

Chris Clifton - CS541

66

Outline

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



Fall 2002

Chris Clifton - CS541

67

Optimizations (in controller or O.S.)

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

Fall 2002

Chris Clifton - CS541

68

Double Buffering

Problem: Have a File

- » Sequence of Blocks B1, B2

Have a Program

- » Process B1
- » Process B2
- » Process B3

⋮

Fall 2002

Chris Clifton - CS541

69

Single Buffer Solution

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

Fall 2002

Chris Clifton - CS541

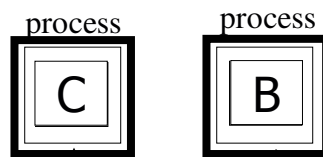
70

Say P = time to process/block
 R = time to read in 1 block
 n = # blocks

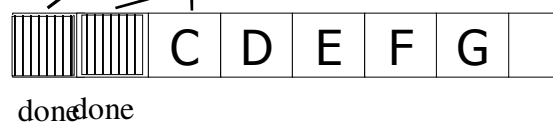
Single buffer time = $n(P+R)$

Double Buffering

Memory:



Disk:



Say $P \geq R$

What is processing

$P = \text{Processing time/block}$ $R = \text{IO time/block}$ $n = \# \text{ blocks}$

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

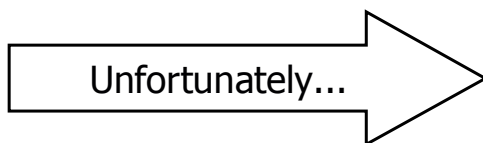
Fall 2002

Chris Clifton - CS541

73

Block Size Selection?

- Big Block → Amortize I/O Cost



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

Fall 2002

Chris Clifton - CS541

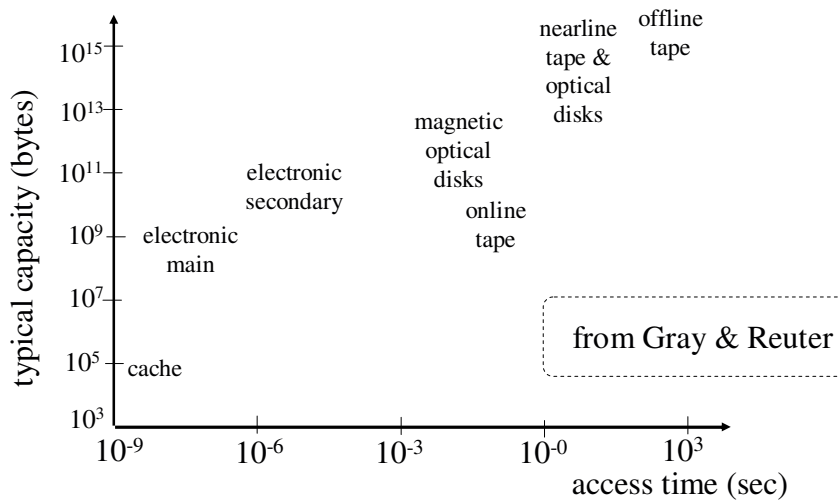
74

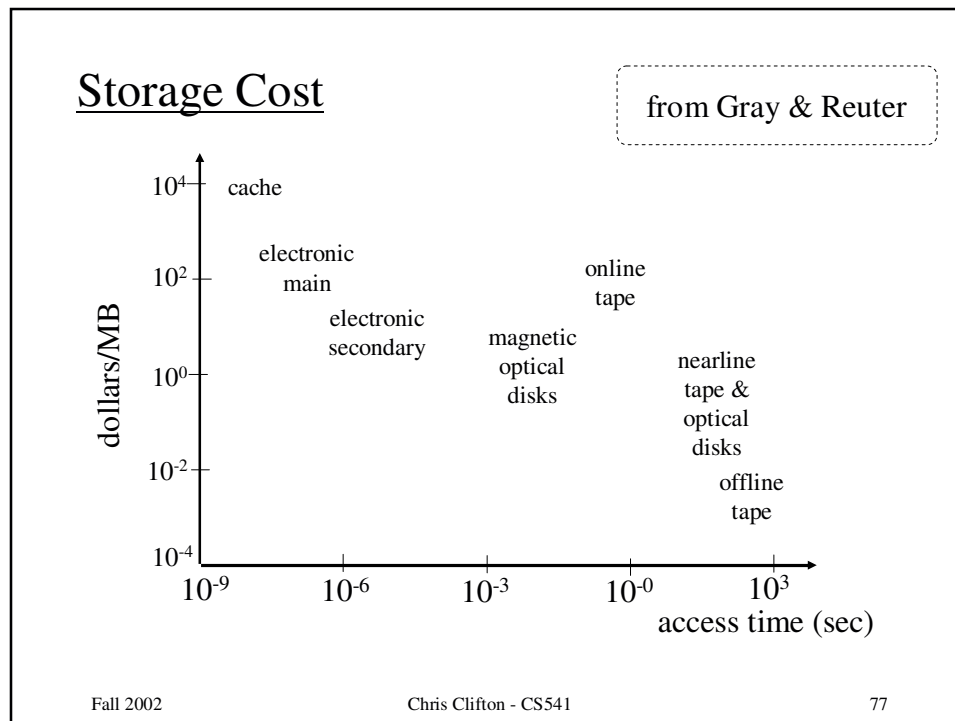
Trend

Trend

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

Storage Cost





Using secondary storage effectively (Sec. 2.3)

- 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 2.5)

- Partial → Total
- Intermittent → Permanent

Fall 2002

Chris Clifton - CS541

79

Coping with Disk Failures

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

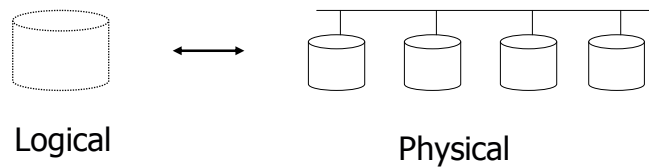
Fall 2002

Chris Clifton - CS541

80

At what level do we cope?

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



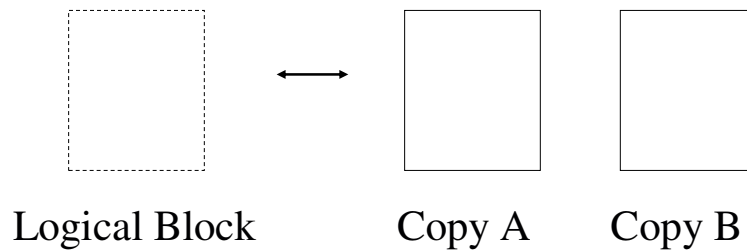
Fall 2002

Chris Clifton - CS541

81

Operating System

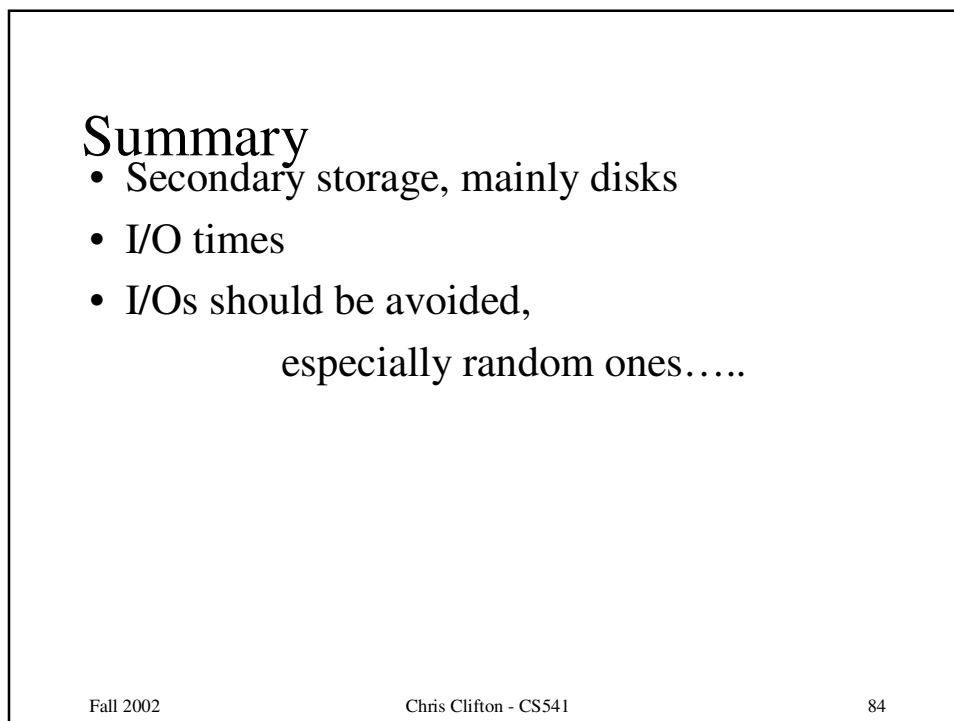
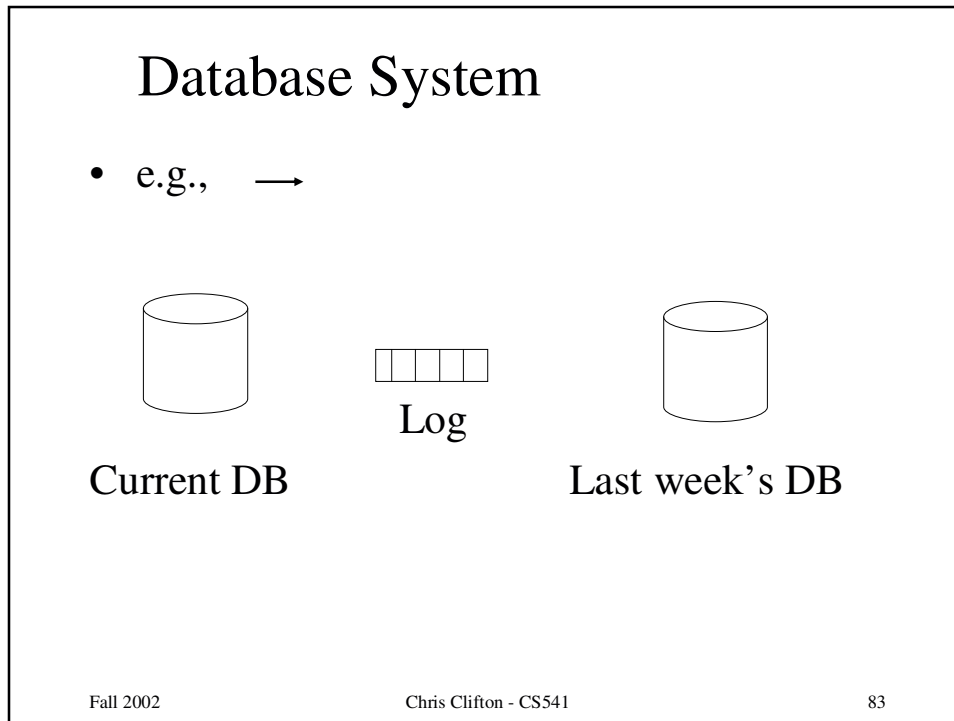
→ e.g., Stable Storage



Fall 2002

Chris Clifton - CS541

82



Outline

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

