HOW TO STORE DATA
data structure decisions define the algorithms that access data
ALGORITHMS

unordered

[7,4,2,6,1,3,9,10,5,8]

INDEX

DATA
ALGORITHMS

unordered

[7,4,2,6,1,3,9,10,5,8]
ALGORITHMS

INDEX

DATA
DATA STRUCTURES
DEFINE PERFORMANCE
register = this room
caches = this city
memory = nearby city
disk = Pluto

Jim Gray, Turing Award 1998
no perfect structure
no perfect structure
no perfect structure
no perfect structure

Array
Hash-Table
Linked-List
Sorted Array
Skip-List
Trie
B-tree
How do I make my **data system** run x times as fast? (sql,nosql,bigdata, ... )
How do I make my **data system** run x times as fast? (sql, nosql, bigdata, …)

How do I minimize my **bill** in the **cloud**?
How do I make my **data system** run $x$ times as fast?  

How do I minimize my **bill** in the **cloud**?  

How do I extend the **lifetime** of my hardware?
How do I make my **data system** run x times as fast?  

How do I minimize my **bill** in the **cloud**?

How do I extend the **lifetime** of my hardware?

How to accelerate **statistics** computation for data science/ML?
How do I make my **data system** run x times as fast? (sql, nosql, bigdata, ...)

How do I minimize my **bill** in the **cloud**?

How do I extend the **lifetime** of my hardware?

How to accelerate **statistics** computation for data science/ML?

How do I train my **neural network** x times faster?
NEW APPLICATIONS
NEW APPLICATIONS

existing systems need to change too
NEW APPLICATIONS

existing systems need to change too

WORKLOAD HARDWARE

ADAPT
NEW APPLICATIONS

existing systems need to change too

WORKLOAD

HARDWARE

ADAPT

IMPROVE WITHIN A BUDGET

WHAT WILL BREAK MY SYSTEM?

REASON
new applications

continuous need for new storage solutions

more data

new h/w
learning outcome
fundamental of storage
learning outcome
fundamental of storage

software engineering  data-driven startup  research
learning outcome
fundamental of storage

software engineering  data-driven startup  research
data structures, SQL, NoSQL, Big Data, Neural Networks, Statistics, Data Science
learning outcome
fundamental of storage

software engineering  data-driven startup  research
data structures, SQL, NoSQL, Big Data, Neural Networks, Statistics, Data Science

small set of principles across all fields
Think out of the Box
first 4 weeks: introduction to research problems/thinking through lectures
first 4 weeks: introduction to research problems/thinking through lectures

Reading research papers

Open ended projects/research
as of week 5: discussions/presentations
interaction: in and out of class
M/W/F OH/labs, Sat/Sun remote OH
There is no such thing as a wrong question/answer!!!!

as of week 5:

discussions/
presentations

interaction: in and out of class
M/W/F OH/labs, Sat/Sun remote OH
Recent Research Papers

Each student:
2 reviews per week/1 presentation

- review and slides should focus on
  - what is the problem
  - why is it important
  - why is it hard
  - why existing solutions do not work
  - what is the core intuition for the solution
  - solution step by step
  - does the paper prove its claims
  - exact setup of analysis/experiments
  - are there any gaps in the logic/proof
  - possible next steps

* follow a few citations to gain more background
Recent Research Papers

- Review and slides should focus on:
  - what is the problem
  - why is it important
  - why is it hard
  - why existing solutions do not work
  - what is the core intuition for the solution
  - solution step by step
  - does the paper prove its claims
  - exact setup of analysis/experiments
  - are there any gaps in the logic/proof
  - possible next steps

- Each student: 2 reviews per week/1 presentation

- Learn to judge constructively
- Learn to present
- Learn to prepare slides

* Follow a few citations to gain more background
semester project: due in the end of semester + a midway check in (early March, 10%)
semester project: due in the end of semester + a midway check in (early March, 10%)

systems project

individual project

NoSQL, in c/c++
systems project

individual project
NoSQL, in C/C++

research project

groups of three
NoSQL, Neural Networks
Periodic Table of Data Structures

semester project: due in the end of semester + a midway check in (early March, 10%)
semester project: due in the end of semester + a midway check in (early March, 10%)

**systems project**

**individual project**

**NoSQL, in c/c++**

**research project**

**groups of three**

**NoSQL, Neural Networks**

**Periodic Table of Data Structures**
ACM Special Interest Group In Data Management (SIGMOD)
Undergrad Research Competition

first prize in 2016, 2017, 2018, 2019
Adaptive Denormalization
Evolving Trees
Splaying LSM-Trees
Adaptive NoSQL
ACM Special Interest Group In Data Management (SIGMOD)
Undergrad Research Competition

first prize in 2016, 2017, 2018, 2019
Adaptive Denormalization
Evolving Trees
Splaying LSM-Trees
Adaptive NoSQL

Design continuums at CIDR 2019, two projects in SIGMOD 2020 finals
piazza forum

all announcements & discussions
as of week 2
link on class website - check out usage guidelines
piazza forum

all announcements & discussions
as of week 2
link on class website - check out usage guidelines

classes are recorded
(links on class website)
all announcements & discussions
as of week 2
link on class website - check out usage guidelines

classes are recorded
(links on class website)

NO LAPTOP/PHONE POLICY
class is based on participation!
piazza forum

all announcements & discussions
as of week 2
link on class website - check out usage guidelines

Project: 40%
Midway Check-in: 10%
Discussion: 20%
Presentation: 15%
Reviews: 15%

classes are recorded
(links on class website)

NO LAPTOP/PHONE POLICY
class is based on participation!
Get familiar with the very basics of traditional database architectures:

Get familiar with very basics of modern database architectures:

Get familiar with the very basics of modern large scale systems:

Check out: syllabus, preparation readings, project 0, systems project, online sections

http://daslab.seas.harvard.edu/classes/cs265/
Teaching Fellows:

Off class discussions are key! question on readings, ideas, help with code/analysis
Prerequisites

knowledge of algorithms, data structures, hardware, systems
Prerequisites

knowledge of algorithms, data structures, hardware, systems

Research track: open to CS165 students

Systems track allows taking the class without all prerequisites
Prerequisites

knowledge of algorithms, data structures, hardware, systems

Research track: open to CS165 students

Systems track allows taking the class without all prerequisites

(165/265 will not be offered in fall 2020/spring 2021)
questions on logistics?
Next few classes:

BASICS of storage

Intro to RESEACH topics

Discussion phase/presentation as of week 5
### Periodic Table of Data Structures

<table>
<thead>
<tr>
<th>Classes of Designs</th>
<th>B-Trees &amp; Variants</th>
<th>Tries &amp; Variants</th>
<th>LSM-Trees &amp; Variants</th>
<th>Differential Files</th>
<th>Membership Tests</th>
<th>Zone Maps &amp; Variants</th>
<th>Bitmaps &amp; Variants</th>
<th>Hashing</th>
<th>Base Data &amp; Columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partitioning</td>
<td>DONE</td>
<td>DONE</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td>DONE</td>
<td>DONE</td>
</tr>
<tr>
<td>Logarithmic Design</td>
<td>DONE</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fractional Cascading</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Log-Structured</td>
<td>DONE</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Buffering</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Differential Updates</td>
<td>DONE</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sparse Indexing</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adaptness</td>
<td>DONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DONE</td>
</tr>
</tbody>
</table>

@IEEE.EngBul18
Registers on chip cache

On board cache

Memory

Disk

CPU

Cache miss: looking for something which is not in the cache

Memory miss: looking for something which is not in memory

Memory wall

SRAM ~1ns

DRAM ~10ns

~100ns

Faster

Cheaper

Speed

Time

CPU

Mem
Don't miss!

- CPU
- Registers
- On-chip cache
- Disk

Cache miss: looking for something which is not in the cache

Memories:
- SRAM
- DRAM

Speeds:
- ~1 ns
- ~10 ns
- ~100 ns

Cache miss: looking for something which is not in the memory

Memory miss: looking for something which is not in memory
<table>
<thead>
<tr>
<th>Relative Location</th>
<th>Speed (x times)</th>
<th>Time (duration)</th>
</tr>
</thead>
<tbody>
<tr>
<td>registers</td>
<td>~0</td>
<td></td>
</tr>
<tr>
<td>this room</td>
<td>2x</td>
<td>1 min</td>
</tr>
<tr>
<td>on chip cache</td>
<td>10x</td>
<td>10 min</td>
</tr>
<tr>
<td>this building</td>
<td>100x</td>
<td>1.5 hours</td>
</tr>
<tr>
<td>on board cache</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New York</td>
<td>100Kx</td>
<td>2 years</td>
</tr>
<tr>
<td>memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pluto</td>
<td></td>
<td></td>
</tr>
<tr>
<td>disk</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Jim Gray, IBM, Tandem, DEC, Microsoft
ACM Turing award
ACM SIGMOD  Edgar F. Codd Innovations award
need to only read x... but have to read all of page 1

data value x

page1  page2  page3  ...

data move

CPU
registers
on chip cache
on board cache
memory
disk
query $x<5$

(size=120 bytes)
memory level $N$

memory level $N-1$

5 10 6 4 12 2 8 9 7 6 7 11 3 9 6 ...

page size: 5x8 bytes
query $x < 5$

scan

(memory level N)

(size=120 bytes)

5 10 6 4 12

(memory level N-1)

5 10 6 4 12 2 8 9 7 6 7 11 3 9 6 ...

page size: 5x8 bytes
query $x<5$

memory level N

memory level N-1

page size: 5x8 bytes
query $x < 5$

scan

(size=120 bytes)

memory level N

5 10 6 4 12

4

memory level N-1

5 10 6 4 12 2 8 9 7 6 7 11 3 9 6 ...

page size: 5x8 bytes
query \( x < 5 \)

Memory level N

\[
\begin{align*}
5 & \quad 10 & \quad 6 & \quad 4 & \quad 12 \\
2 & \quad 8 & \quad 9 & \quad 7 & \quad 6 \\
4 & \quad & & & \\
\end{align*}
\]

(size=120 bytes)

Memory level N-1

\[
\begin{align*}
5 & \quad 10 & \quad 6 & \quad 4 & \quad 12 \\
2 & \quad 8 & \quad 9 & \quad 7 & \quad 6 \\
7 & \quad 1 & \quad 1 & \quad 3 & \quad 9 & \quad 6 & \quad \ldots
\end{align*}
\]

Page size: 5x8 bytes
query: $x < 5$

memory level $N$

(memory level $N-1$)

Page size: 5x8 bytes
query $x < 5$

memory level N

(scan)

(size=120 bytes)

memory level N-1

(scan)

5 10 6 4 12
2 8 9 7 6
4 2

page size: 5x8 bytes
query $x < 5$

(size=120 bytes)

memory level N

memory level N-1

page size: 5x8 bytes
Query: $x < 5$

Scan

Memory level $N$

[size=120 bytes]

Memory level $N-1$

Page size: 5x8 bytes

80 bytes
query \( x<5 \)

Scan

(memory level N)

\( 7 \ 11 \ 3 \ 9 \ 6 \) \( 2 \ 8 \ 9 \ 7 \ 6 \) \( 4 \ 2 \ 3 \)

(size=120 bytes)

(memory level N-1)

\( 5 \ 10 \ 6 \ 4 \ 12 \) \( 2 \ 8 \ 9 \ 7 \ 6 \) \( 7 \ 11 \ 3 \ 9 \ 6 \) ...

page size: 5x8 bytes
query $x < 5$

(memory level N)

5 10 6 4 12

(size=120 bytes)

2 8 9 7 6

4 2 3

scan

(memory level N-1)

7 11 3 9 6

5 10 6 4 12

2 8 9 7 6

7 11 3 9 6

...
an oracle gives us the positions

query $x < 5$

(size=120 bytes)
memory level N

memory level N-1

5 10 6 4 12 2 8 9 7 6 7 11 3 9 6 ...

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

oracle

(size=120 bytes)
memory level N

memory level N-1

5 10 6 4 12
2 8 9 7 6
7 11 3 9 6
...

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

memory level N

memory level N-1

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

(oracle)

memory level N

memory level N-1

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

(memory level N)

(size=120 bytes)

(memory level N-1)

page size: 5x8 bytes
an oracle gives us the positions

query \( x < 5 \)

.oracle

(size=120 bytes)

memory level N

5 10 6 4 12

oracle

2 8 9 7 6

4 2

memory level N-1

5 10 6 4 12 2 8 9 7 6 7 11 3 9 6 ...

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

(memory level N)

oracle

5 10 6 4 12

(size=120 bytes)

memory level N-1

oracle

2 8 9 7 6

4 2

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

(size=120 bytes)

memory level N

memory level N-1

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

memory level N

memory level N-1

page size: 5x8 bytes
an oracle gives us the positions

query $x < 5$

memory level N

(size=120 bytes)

memory level N-1

page size: 5x8 bytes
an oracle gives us the positions

query $x<5$

page size: 5x8 bytes
when does it make sense to have an oracle
how can we minimize the cost

e.g., query \( x < 5 \)
algorithm system design = not just computation
CPU   DATA MOVEMENT   (ENERGY)   MEMORY REQUIREMENT   ROBUSTNESS   SPACE REQUIREMENT
<table>
<thead>
<tr>
<th>CPU</th>
<th>DATA MOVEMENT</th>
<th>(ENERGY)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEMORY REQUIREMENT</td>
<td>ROBUSTNESS</td>
<td>SPACE REQUIREMENT</td>
</tr>
</tbody>
</table>
SQL, NoSQL, Graph, Neural Nets, Statistics, Vision

CPU      DATA MOVEMENT      (ENERGY)
MEMORY REQUIREMENT
ROBUSTNESS      SPACE REQUIREMENT

TIME      ____      CLOUD COSTS
Get familiar with the very basics of traditional database architectures:

Get familiar with very basics of modern database architectures:

Get familiar with the very basics of modern large scale systems:

Check out: syllabus, preparation readings, project 0, systems project, online sections

http://daslab.seas.harvard.edu/classes/cs265/