class 2

from bits to systems

prof. Stratos Idreos

HTTP://DASLAB.SEAS.HARVARD.EDU/CLASSES/CS165/
today

logistics, goals, etc

big data & systems (cont’d)

designing a data system algorithm: what can go wrong

quick overview: logical/physical/system design
Marcin Zukowski
founder @Snowflake
previously founder @Vectorwise

10/17
Johannes Gehrke

Microsoft

previously prof at Cornell

10/24
HV Jagadish

Prof University of Michigan

10/6
research oriented, discussion based, material based on research papers
labs, sections, and cs nights

**Sections**: video only / 1-2 videos a week
first 2 sections on Project 0 and basics of C are online

**Labs (& CS Night)**: every day
help with debugging and everything else
starting Tuesday Sep 6

**OH**: every day 3-4pm with Stratos at MD139
away next week but most likely I will be able to hold OH remotely most days
monitor Piazza and learn to use Zoom
http://daslab.seas.harvard.edu/classes/cs165/project.html

code and testing infrastructure available as of today

treat it as a good starting point
you have complete design freedom
no libraries allowed

 collaboration encouraged! final deliverable is personal

 face-to-face/hands-on evaluation

 stay in touch!
James & Wilson

spring 2014, somewhere at Harvard, late at night working on cs165 project
spring 2014, somewhere at Harvard, late at night working on cs165 project

several months later…

“This class is invaluable.”

“One of the greatest intellectual journeys to date.”
“TAKE THIS CLASS IF YOU WANT TO ACTUALLY LEARN SOMETHING AND BECOME A TRUE COMPUTER SCIENTIST!”

“The course is amazing! It's relaxed, it teaches you about modern database systems, it introduces you to current research topics, it's freaking incredible!”

“CS is awesome and Databases rock! I came in not being very confident in my cs skills or my understanding of how memory hierarchy and large systems work and now everything is changed! I also just feel very inspired to explore more in the area of databases in the future! Loved this class!”
FAQ

should I take the class now or next year (or never)?
  it depends (but mostly up to you and your energy levels)
  delaying can be better if you not senior
  unless you have research ambitions

  can I take it pass fail?
    no

  can I audit?
    yes (but limited slots)
cs165 diff from 2015

- cleaned up DSL
- cleaned up starter code
- give out parser code
- revamped testing infrastructure
- performance tests (=bigger data)
- benchmark implementation
- project 0
- midway check-in 10%
- class participation 20%
- more TFs (lab class status)
- addition of new research papers from 2015/2016
we purposely build up slowly in class

from your side:
Project 0 next week
and basic environment/tools for project
then start with P1 the week after once we discuss the basics of storage
data

star(id, name, distance, density, …)

[1, star1, x1, y1, …]
[2, star2, x2, y2, …]
[3, star3, x3, y3, …]
[4, star4, x4, y4, …]
...

astronomer
data

star(id, name, distance, density, …)

[1, star1, x1, y1, …]
[2, star2, x2, y2, …]
[3, star3, x3, y3, …]
[4, star4, x4, y4, …]
...

data size  store - access

10s/100s  paper - just look at it!

data collection is the key
data

\[
\text{star(id, name, distance, density, \ldots)}
\]

\[
[1, \text{star1, x1, y1, \ldots}]
\]

\[
[2, \text{star2, x2, y2, \ldots}]
\]

\[
[3, \text{star3, x3, y3, \ldots}]
\]

\[
[4, \text{star4, x4, y4, \ldots}]
\]

\[
\ldots
\]

data size

<table>
<thead>
<tr>
<th>store - access</th>
</tr>
</thead>
<tbody>
<tr>
<td>10s/100s</td>
</tr>
<tr>
<td>paper - just look at it!</td>
</tr>
</tbody>
</table>

data collection is the key

| 10s/100s       |
| K/M            |
| PC files - shell/excel |

learn a bit how computers work
data

star(id, name, distance, density, …)

[1, star1, x1, y1, …]
[2, star2, x2, y2, …]
[3, star3, x3, y3, …]
[4, star4, x4, y4, …]
...

data size  store - access

10s/100s  paper - just look at it!

K/M  PC files - shell/excel

data collection is the key

learn a bit how computers work

need a bit more tailored analysis

need serious programming skills

PC files - custom
data

star(id, name, distance, density, …)

[1, star1, x1, y1, …]
[2, star2, x2, y2, …]
[3, star3, x3, y3, …]
[4, star4, x4, y4, …]
...

<table>
<thead>
<tr>
<th>data size</th>
<th>store - access</th>
</tr>
</thead>
<tbody>
<tr>
<td>10s/100s</td>
<td>paper - just look at it!</td>
</tr>
<tr>
<td>K/M</td>
<td>PC files - shell/excel</td>
</tr>
</tbody>
</table>

data collection is the key

learn a bit how computers work

need a bit more tailored analysis

need serious programming skills

exploration - many users/updates

data sys. - declarative

data-driven analysis
“Increasingly, scientific breakthroughs will be powered by advanced computing capabilities that help researchers manipulate and explore massive datasets.

The speed at which any given scientific discipline advances will depend on how well its researchers collaborate with one another, and with technologists, in areas of eScience such as databases, workflow management, visualization, and cloud computing technologies.”
big data V’s
(it is not about size only)

volume  velocity  variety  veracity

actually none of that is really new…

new:
our ability to gather and store machine generated data
broad understanding that we cannot just manually get value out of data
declarative interface

ask “what” you want

db system

the system decides “how” to best store and access data
a data system **stores** data
and **provides access** to data

& makes knowledge generation easy
a data system **stores** data
and **provides access** to data

& makes knowledge generation easy
A data system **stores** data and **provides access** to data, and makes knowledge generation easy.
(here is where all the magic happens!)

data system kernel

cs165/265 student
a data system is a massive collection of algorithms and data structures

more than one ways to do the same thing

optimization: a smart way to dynamically decide which way to choose for each query

no good way to model a system
a data system is a massive collection of algorithms and data structures

more than one ways to do the same thing

optimization: a smart way to dynamically decide which way to choose for each query

no good way to model a system

Research Problem 1
so what is a good data system?

it depends...

conflicting goals

application requirements

hardware

performance

budget

energy profile
“Three things are important in the database world: \textit{performance, performance, and performance}”

Bruce Lindsay, IBM
ACM SIGMOD  Edgar F. Codd Innovations award 2012
a "simple" example

assume an array of $N$ integers:
find all positions where $value>x$

exists in all systems: sql, nosql, newsql

select operator

even the simplest tasks are actually far from trivial

no obvious solutions; just a taste of what to come
assume an array of $N$ integers
find all positions where $value > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```
assume an array of $N$ integers: find all positions where $value > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if (data[i] > x)
        res[j++] = i
```

res[j] = i;
j += (data[i] > x)
assume an array of $N$ integers: find all positions where $value > x$

res=new array[data.size]

$j=0$

for (i=0; i<data.size; i++)
    if data[i] > x
        res[j++] = i

what if only 1% qualifies?
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

```
res=new array[data.size]
j=0
for (i=0; i<data.size; i++)
    if data[i]>x
        res[j++]=i
```

what if only 1% qualifies?
Assume an array of $N$ integers: find all positions where $\text{value} > x$.

```java
res = new array[data.size]
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

What if only 1% qualifies?
assume an array of \( N \) integers:
find all positions where \( value > x \)

```java
res = new array[data.size]
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?

memory

data
assume an array of \( N \) integers: find all positions where \( value > x \)

```
res = new array[data.size]
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

```
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?

memory
assume an array of $N$ integers: find all positions where $value > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?
assume an array of $N$ integers: find all positions where value $> x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?
assume an array of $N$ integers: find all positions where $value > x$

```plaintext
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
  if data[i] > x
    res[j++] = i
```

what if only 1% qualifies?
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for } (i=0; \ i<\text{data.size}; \ i++)
\]

\[
\quad \text{if } \text{data}[i] > x
\]

\[
\quad \text{res}[j++] = i
\]

what if only 1\% qualifies?
assume an array of $N$ integers: find all positions where $value > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?

memory

- data
- copy res
assume an array of $N$ integers: find all positions where $value > x$

```java
res = new array[data.size]
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

what if only 1% qualifies?

but how can we know?
assume an array of $N$ integers: find all positions where $value > x$

res = new array[data.size]

$j = 0$
for ($i = 0; i < data.size; i++$)
  if $data[i] > x$
    $res[j++] = i$

what if 90% qualifies?

result size = qualifying values * w bytes
assume an array of $N$ integers: find all positions where $\text{value} > x$

res = new array [data.size]

$j = 0$

for ($i = 0; i < \text{data.size}; i++$)
  if (data[i] > x)
    res[$j++$] = i

what if 90% qualifies?

result size = qualifying values * $w$ bytes

bit vector for res?

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

\[
\text{res = new array[data.size]}
\]

\[
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
\]

what if 90% qualifies?

result size = qualifying values * \( w \) bytes

bit vector for \( \text{res} \)?

if statements = bad, bad, bad
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

and we haven’t even started discussing about how to find the qualifying values…

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for } (i = 0; i < \text{data.size}; i++)
\]

\[
\text{if } \text{data}[i] > x
\]

\[
\text{res}[j++] = i
\]

what if 90% qualifies?

result size = qualifying values \( \times w \) bytes

bit vector for res?

if statements = bad, bad, bad
assume an array of $N$ integers: find all positions where $value > x$

res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

```java
res = new array[data.size]
j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

NUMA architectures? SIMD functionality? & what about result writing?

not as simple as spinning off \( N \) threads…
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for (i=0; i<\text{data.size}; i++)}
\]

\[
\text{if data}[i] > x
\]

\[
\text{res}[j++] = i
\]

\( N \gg 1 \text{ queries in parallel} \)

q1,q2,q3
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for } (i = 0; i < \text{data.size}; i++)
\]

\[
\text{if } \text{data}[i] > x
\]

\[
\text{res}[j++] = i
\]

\( N \gg 1 \) queries in parallel

q1, q2, q3
assume an array of $N$ integers: find all positions where $\text{value} > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

$N >> 1$ queries in parallel
assume an array of \( N \) integers: find all positions where \( \text{value} > x \)

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for } (i=0; i<\text{data.size}; i++)
\]

\[
\quad \text{if } \text{data}[i] > x
\]

\[
\quad \text{res}[j++] = i
\]
assume an array of $N$ integers: find all positions where $value > x$

\[
\text{res} = \text{new array}[\text{data.size}]
\]

\[
j = 0
\]

\[
\text{for } (i=0; i<\text{data.size}; i++)
\]

\[
\quad \text{if data}[i] > x
\]

\[
\quad \text{res}[j++] = i
\]

$N >> 1$ queries in parallel

$q1, q2, q3 \quad q4$

$q4$
assume an array of $N$ integers: find all positions where $value > x$

```java
res = new array[data.size]

j = 0
for (i = 0; i < data.size; i++)
    if data[i] > x
        res[j++] = i
```

```java
res = new array[10]
j = 0

if data[0] > x res[j++] = 0
if data[1] > x res[j++] = 1
if data[2] > x res[j++] = 2
if data[3] > x res[j++] = 3
if data[4] > x res[j++] = 4
if data[6] > x res[j++] = 6
if data[7] > x res[j++] = 7
if data[8] > x res[j++] = 8
if data[9] > x res[j++] = 9
```
cost: data touched & computation
random access &
page-based access

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
assume an array of $N$ integers: find all positions where $value >$ 

option 1: **scan** all data

option 2: use a **tree** (do not consider tree generation costs)

**which one is best?**

groups 3-4 students: try to keep noise level down, take turns speaking, etc.
it is not obvious that the tree would give the best result even if we ignore the building cost

random access at the granularity of one page at a time

data movement is a HUGE cost component

it depends on the query and the data

>1 ways to have a tree (implicit tree, linked leaves or not, etc.)

and happens with updates, >>1 concurrent queries, etc.
actually this was your second open research problem of the day!

systems are so fast= too little time to optimize

we will do all of these in great detail: the examples so far are meant to demonstrate the kinds of problems and discussion we will be doing
it all starts with how we store data

**every bit matters**
C and no libraries
MAIN-MEMORY OPTIMIZED DATA SYSTEMS
MAIN-MEMORY OPTIMIZED DATA SYSTEMS
history/timeline

~1960s

philosophy/sciences

papyrus/paper

~2015

dbs

dbs vs OSs

dbs

... dbs vs nosql

declarative/physical/logical independence/recovery/consistency

dbs vs hand code

history/timeline
why noSQL

as apps become more complex
as apps need to be more scalable

newSQL
why noSQL

regardless of system name/data model it is the same design space and the same design principles

db
- complex
- legacy
- tuning
- expensive
- fast
... 

noSQL
- simple
- clean
- just enough
- cheap/free
- slow

as apps become more complex
as apps need to be more scalable

newSQL
design

- **logical** design
- **physical** design
- **system** design
piazza forum

all announcements & discussions
(link on class website)
slides are not notes!
slides are mainly there to trigger discussion

note keeping is your task:
starting class 3 we will do collaborative note taking:
http://tinyurl.com/CS165-Fall2016-Notes
and check syllabus/website!
stratos away next week

NO CLASS ON SEP 7
TFs will hold a lab

NEXT CLASS ON SEP 12
4pm, Pierce 301

NO OH NEXT WEEK
every day labs +
OH with Stratos on demand
via Skype or Zoom
reading

textbook: chapters 1, 3 (-3.5), 5 (-5.8, -5.9)
intro + relational model + SQL

browse “the Fourth Paradigm”

next couple of classes

data models
low level details - system design
data layouts and column-stores
readings for next 3 classes

Architecture of a Database System  (Sections 1, 2, 3, 4)
by J. Hellerstein, M. Stonebraker and J. Hamilton

The Design and Implementation of Modern Column-store Database Systems
by D. Abadi, P. Boncz, S. Harizopoulos, S. Idreos, S. Madden
games

thousands of users - tons of data
frequent updates - tons of queries
analytics in the background
for game intelligence

W. White, A. Demers, C. Koch, J. Gehrke and R. Rajagopalan

Scaling games to epic proportions

ACM SIGMOD International Conference on Management of Data, 2007
Notes to remember

traditional algorithm analysis does not capture all costs
both data movement and processing matter
“mundane” steps such as incrementing a loop counter or materializing results can have a huge effect
there are no simple tasks
all systems are the “same”: just data and access patterns
from bits to dbs

DATA SYSTEMS

prof. Stratos Idreos