CS 265
Stratos Idreos
BIG DATA SYSTEMS
NoSQL | Neural Networks | SQL | Graph | Data Science
Today:
Go quickly over logistics again

Introduce self-designing systems concept

Start understanding self-designing through data structures/KV-stores

Rough intro into key-value stores
learning outcome
Fundamentals of storage
data structures, SQL, NoSQL, Neural Networks, Data Science, Images, LLMs
learning outcome
Fundamentals of storage

data structures, SQL, NoSQL, Neural Networks, Data Science, Images, LLMs

Self-designing systems

Automated system design: cloud cost, hardware, data & app requirements
How do I make my **data system** run x times as fast? (sql,nosql,bigdata, ...)

---

[Logo: DASlab @ Harvard SEAS]
How do I make my **data system** run x times as fast?  

How do I minimize my **bill** in the **cloud**?
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 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 to accelerate **statistics** computation for data science/ML?

How do I train my **neural network/LLM** $x$ times faster?
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 to accelerate **statistics** computation for data science/ML?

How do I train my **neural network/LLM** x times faster?

How can I do 10x **Image AI inference**?
first 4-5 weeks: Stratos/Sanket/Utku
Basic background
Self-designing systems
Neural network systems
Image AI systems
Research thinking

afterwards:
Students present research papers
One paper per class (ML systems)
In-class research/systems discussion
Research reviews
Research/systems projects
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 (mid March,10%)
semster project: due in the end of semester + a midway check in (mid March, 10%)
semester project: due in the end of semester + a midway check in (mid March, 10%)

systems project

individual project
NoSQL, in c/c++
MLsys, in pytorch

research project

groups of max three
Adaptivity/Performance
Across all subject areas
systems project

individual project
NoSQL, in c/c++
MLsys, in pytorch

research project

groups of max three
Adaptivity/Performance
Across all subject areas

semester project: due in the end of semester + a midway check in (mid March, 10%)
all announcements & discussions
as of week 2
link on class website - check out usage guidelines
all announcements & discussions
as of week 2
link on class website - check out usage guidelines

classes are recorded
(links on canvas)
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 canvas)
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 canvas)

NO LAPTOP/PHONE POLICY
class is based on participation!
In class discussions is a critical component and learning outcome

Think creatively
Fail quickly
Incrementally solve
In class discussions is a critical component and learning outcome

Think creatively
Fail quickly
Incrementally solve

DailyOH/labs,
Sat/Sun remote OH
In class discussions is a critical component and learning outcome

Think creatively
Fail quickly
Incrementally solve

DailyOH/labs, Sat/Sun remote OH

There is no such thing as a wrong question/answer!!!!
In class discussions is a critical component and learning outcome.

Think creatively
Fail quickly
Incrementally solve


There is no such thing as a wrong question/answer!!!!
In class discussions is a critical component and learning outcome.

Every few classes - pick some technique and try it out.


There is no such thing as a wrong question/answer!!!!
Questions on logistics?
algorithm/system design = not just computation

50-80% of end-to-end time is due to storage-related decisions
algorithm/system design = not just computation

50-80% of end-to-end time is due to storage-related decisions
DATA STRUCTURES
DEFINE PERFORMANCE

2023

speed

DATA MOVEMENT

COMPUTE
register = this room
caches = this city
memory = nearby city
disk = Pluto

Jim Gray, Turing Award 1998
Self-designing Systems
The problem: as the big data/AI world keeps changing…
The problem: as the big data/AI world keeps changing…

there is a continuous need for new data systems

but it is **extremely hard to design & build new systems**
How do we design a data system that is $X$ times faster for a workload $W$?
How do we design a data system that is \textit{X times faster for a workload W}? 

How do we design a data system that allows for control of \textit{cloud cost}?
How do we design a data system that is \textit{X} times faster for a workload \textit{W}? \\

How do we design a data system that allows for control of \textit{cloud cost}? \\

What happens if we introduce \textit{new application feature} \textit{Y}? \\

Should we \textit{upgrade} to new version \textit{Z}? \\

What will \textit{break} our system?
BOTTLENECK: SUB-OPTIMAL DATA SYSTEMS

What happens if we introduce **new application feature** Y?

Should we **upgrade** to new version Z?

What will **break** our system?
BOTTLENECK: SUB-OPTIMAL DATA SYSTEMS

huge cloud cost
environmental impact
How do I make my data system run X times faster? How do I control my bill on the cloud?

BOTTLENECK: SUB-OPTIMAL DATA SYSTEMS

huge cloud cost  expensive transitions
environmental impact
How do I make my data system run X times faster?

How do I control my bill on the cloud?

NEED TO DESIGN NEW DATA SYSTEMS

BOTTLENECK: SUB-OPTIMAL DATA SYSTEMS

huge cloud cost
expensive transitions
application feasibility
environmental impact
BOTTLENECK: SUB-OPTIMAL DATA SYSTEMS

huge cloud cost   expensive transitions   application feasibility   environmental impact

complexity
how we BUILD systems
BUILD

GET $N$ EXPERT DESIGNERS
GIVE THEM $T$ TIME
HOPE FOR THE BEST
build

get $N$ expert designers

give them $T$ time

hope for the best

design is an art
BUILD

GET $N$ EXPERT DESIGNERS
GIVE THEM $T$ TIME
HOPE FOR THE BEST

design is an art
Design: 6-7 years
Reasoning: months/impossible

GET \( N \) EXPERT DESIGNERS
GIVE THEM \( T \) TIME
HOPE FOR THE BEST

design is an art
1 design/research skills do not scale

- data
  - applications
  - systems

years

- design skills

[Stratos' Guess]

years
2 no one knows everything out there

NoSQL storage

The log-structured merge-tree (LSM-tree)

No one knows everything out there

The log-structured merge-tree (LSM-tree)
2 no one knows everything out there

NoSQL storage

P. O’Neil, E. Cheng, D. Gawlick, E, O’Neil
The log-structured merge-tree (LSM-tree)
The HIPPO method
"highest paid person's opinion"
standard “solution”

expose knobs
Some possible ideas
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?
   Yes, but only around a narrow design space.
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision? Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications?
1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision? Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications? Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

Some possible ideas
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?
   Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications?
   Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

3. Aren’t learned system components able to adapt even more?
Some possible ideas

1. **Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?**
   Yes, but only around a narrow design space.

2. **Aren’t adaptive data systems architectures able to adapt to new applications?**
   Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

3. **Aren’t learned system components able to adapt even more?**
   Yes, better than #2 (e.g., data adaptivity), but still only around a narrow design space.
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision? Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications? Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

3. Aren’t learned system components able to adapt even more? Yes, better than #2 (e.g., data adaptivity), but still only around a narrow design space.

4. Can’t we just throw ML into the problem? ChatGPT?
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?
   Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications?
   Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

3. Aren’t learned system components able to adapt even more?
   Yes, better than #2 (e.g., data adaptivity), but still only around a narrow design space.

4. Can’t we just throw ML into the problem? ChatGPT?
   Yes, but the programming design space is massive. A correct design is not a desired one.
Some possible ideas

1. Aren’t data systems already “adaptive”, e.g., optimizer makes the best online decision?
   Yes, but only around a narrow design space.

2. Aren’t adaptive data systems architectures able to adapt to new applications?
   Yes, better than #1 (e.g., query adaptivity), but still only around a narrow design space.

3. Aren’t learned system components able to adapt even more?
   Yes, better than #2 (e.g., data adaptivity), but still only around a narrow design space.

4. Can’t we just throw ML into the problem? ChatGPT?
   Yes, but the programming design space is massive. A correct design is not a desired one.

These ideas can lead to better systems but we need something more to FIND FAST THE BEST POSSIBLE DESIGN
SELF-DESIGNING SYSTEMS

Automatically invent & build the perfect system for any new application
massive design space of system designs
massive design space of system designs

system = a set of low-level design decisions
few existing designs

massive design space of system designs

system = a set of low-level design decisions
few existing designs

massive design space of system designs

system = a set of low-level design decisions
massive design space of system designs

reasoning: understand all the design decisions & their impact
HOW
——
DO WE
——START——
HOW DO WE START

- SLAs
- robustness
- concurrency
- data types
- complex operations
- multi-tenancy
- indexing
- optimizer
- cloud
- hardware

SLAs, robustness, concurrency, data types, complex operations, multi-tenancy, indexing, optimizer, cloud, hardware.
DATA

INDEX

DATA

—— HOW ——

—— DO WE ——

—— START ——
ALGORITHMS

data structure decisions define the algorithms that access data
ALGORITHMS

unordered

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

unordered

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

DATA INDEX

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

ordered: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
no perfect structure
no perfect structure
no perfect structure

- Read
- Update
- Memory

- differential
- approximate

- point tree

memory amplification

EDBT 2016
SIGMOD 2016
point read

range read

update

memory
NoSQL systems are the backbone of the BigData and AI era

- LSM-tree
- KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE
NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer
NoSQL systems are the backbone of the BigData and AI era

LSM-tree   FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
KV-stores  MACHINE LEARNING, SQL, CRYPTO, SCIENCE
NoSQL systems are the backbone of the BigData and AI era

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE
NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE
NoSQL systems are the backbone of the BigData and AI era

LSM-tree  FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
KV-stores  MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer  filters  fences  cache

level 0  level 1  level 2  level 3
NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores
FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer
filters
fences
cache

diverse
data structures

level 0
level 1
level 2
level 3
NoSQL systems are the backbone of the BigData and AI era

- LSM-tree
- KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE

diverse data structures

buffer
filters
fences
cache

interactions

level 0
level 1
level 2
level 3
NoSQL systems are the backbone of the BigData and AI era

LSM-tree

KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer

filters

fences

cache

diverse data structures

interactions

hardware

level 0

level 1

level 2

level 3
NoSQL systems are the backbone of the BigData and AI era

LSM-tree

KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer

filters

fences

cache

diverse data structures

interactions

hardware

parallelism

level 0

level 1

level 2

level 3
NoSQL systems are the backbone of the BigData and AI era

*LSM-tree*  FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN  MACHINE LEARNING, SQL, CRYPTO, SCIENCE

*KV-stores*
NoSQL systems are the backbone of the BigData and AI era

LSM-tree and KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN
MACHINE LEARNING, SQL, CRYPTO, SCIENCE

buffer
diverse data structures

filters
interactions

fences
hardware

robustness
cache
cloud cost

parallelism
SLAs

There exist numerous variations of NoSQL KV-stores
LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft)
There exist numerous variations of NoSQL KV-stores. LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft).
There exist numerous variations of NoSQL KV-stores. LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft).
There exist numerous variations of NoSQL KV-stores. LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft)

Constant and increasing efforts for new system designs as applications & hardware change
Requirements/Goals

- data & queries
- performance
- budget

Context

- SLA
- hardware
- parallelism
- budget
- cloud cost
- SLAs

diverse data structures
Requirements/Goals

- data & queries
- performance
- budget

Context

- interactions
- hardware
- parallelism
- robustness cloud cost SLAs
Requirements/Goals

- diverse data structures
- interactions
- hardware
- parallelism
- robustness cloud cost SLAs

Context

- data & queries
- performance
- budget
- SLA

Best DATA SYSTEM design & code
what-if reasoning
data & queries

performance

budget

$$\text{SLA}$$

what-if reasoning

design1

perf1

cost1

DASlab
© Harvard SEAS
what-if reasoning
AUTO DESIGN
“IS THERE A CALCULUS OF DATA STRUCTURES
by which one can choose the appropriate representation
and techniques for a given problem?” (SIAM, 1978)

[P vs NP, average case, constant factors vs asymptotic, low bounds]
Is there a calculus of data structures by which one can choose the appropriate representation and techniques for a given problem?" (SIAM, 1978) 

\[P \text{ vs } NP, \ \text{average case, constant factors vs asymptotic, low bounds}\]
the grammar of data systems design
the grammar of data systems design

action is the most holy form of ultimate theory

I hope for nothing
I fear nothing
I am free

Nikos Kazantzakis, philosopher
I hope for nothing
I fear nothing
I am free

Nikos Kazantzakis, philosopher
words
alphabet

Nikos Kazantzakis, philosopher

the grammar of data systems design

action is the most holy form of ultimate theory

I hope for nothing
I fear nothing
I am free
the grammar of data systems design

action is the most holy form of ultimate theory

I hope for nothing
I fear nothing
I am free

Nikos Kazantzakis, philosopher
Nikos Kazantzakis, philosopher

the grammar of data systems design

action is the most holy form of ultimate theory

I hope for nothing
I fear nothing
I am free
the grammar of data systems design

Grammar:
- sentences
- words
- data structures
- alphabet
- principles

Nikos Kazantzakis, philosopher

I hope for nothing
I fear nothing
I am free
Nikos Kazantzakis, philosopher

The grammar of data systems design

Action is the most holy form of ultimate theory.

I hope for nothing
I fear nothing
I am free
the grammar of data systems design

New

I hope for nothing
I fear nothing
I am free

Nikos Kazantzakis, philosopher
<table>
<thead>
<tr>
<th>grammar/sentences</th>
<th>interactions</th>
<th>data structures</th>
<th>principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>words</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alphabet</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Nikos Kazantzakis, philosopher

the **grammar** of data systems design

action is the most holy form of theory

which are “all” possible data systems we may ever invent?

I hope for nothing
I fear nothing
I am free

Nikos Kazantzakis, philosopher
Trillions of possible data structures
Data Calculator @SIGMOD 2018
Trillions of possible data structures
Data Calculator @SIGMOD 2018

New NoSQL systems: 1000x faster
Cosine @PVLDB 2022 and Limousine @SIGMOD 2024
Trillions of possible data structures
Data Calculator @SIGMOD 2018

New NoSQL systems: 1000x faster
Cosine @PVLDB 2022 and Limousine @SIGMOD 2024

Synthesized statistics, 10x faster ML
Data Canopy @SIGMOD 2017
Trillions of possible data structures
Data Calculator @SIGMOD 2018

New NoSQL systems: 1000x faster
Cosine @PVLDB 2022 and Limousine @SIGMOD 2024

Synthesized statistics, 10x faster ML
Data Canopy @SIGMOD 2017

10x faster Neural Networks
MotherNets @MLSys 2020, and M2 @MLSys 2023
Trillions of possible data structures
Data Calculator @SIGMOD 2018

New NoSQL systems: 1000x faster
Cosine @PVLDB 2022 and Limousine @SIGMOD 2024

Synthesized statistics, 10x faster ML
Data Canopy @SIGMOD 2017

10x faster Neural Networks
MotherNets @MLSys 2020, and M2 @MLSys 2023

10x faster Image AI
Image Calculator, SIGMOD 2024
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 1, online sections

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

Research papers: late next week

Second systems project: late next week

Research projects: with the research lectures (week 4-5)

Expect to start systems/research project mid Feb
CS 265
Stratos Idreos
BIG DATA SYSTEMS
NoSQL | Neural Networks | SQL | Graph | Data Science