CS 265

Big Data & AI Systems

NoSQL | Neural Networks | Image AI | LLMs | Data Science

Scope: End-to-end AI systems
Topics: LLMs, Context, Agents, RAG
Inspiration: Research + Industry
Technical: Storage/Computation/Self-designing
Projects:

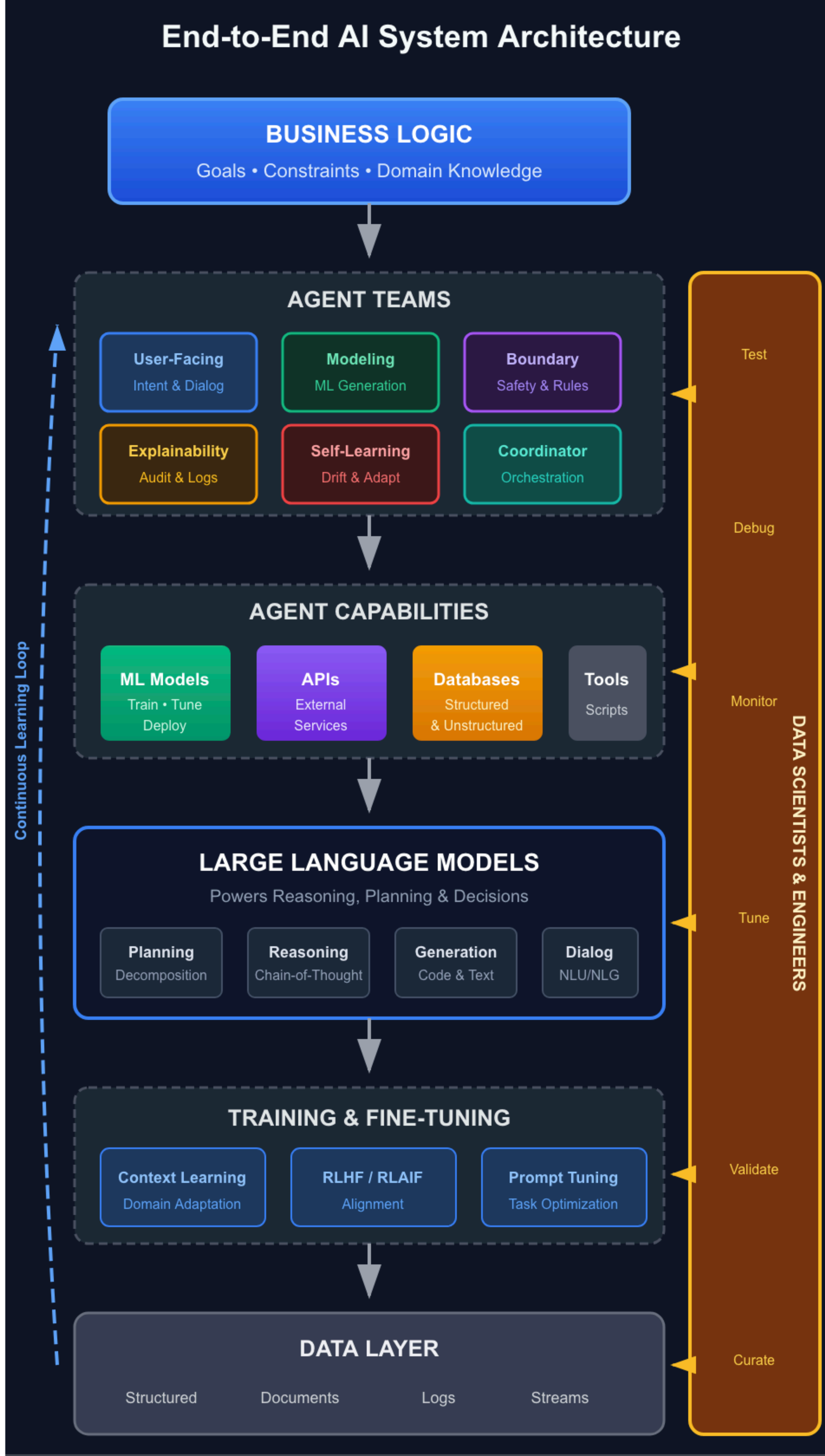
Systems (LLM core, or design)
Research (LLM compiler, RAG, Image, Fine-tuning, Context Management)
Research is open to 165 & systems students but eventually open to all

Timeline:

5 weeks of introduction
then reading research papers

Goals: Develop to an “AI systems person”
Info: http://daslab.seas.harvard.edu/classes/cs265/

DASlab @ Harvard SEAS




Industry Evolution Path
VALUE & COMPLEXITY
1
Data Foundation
STORE & MANAGE
Data-first approach
2
Analytics
INSIGHT & REPORTING
Business intelligence
3
Productivity Automation
BASIC AGENTS
Workflow efficiency
4
Core Automation
BUSINESS USE CASES
Mission-critical functions
5
Custom ML Models
DOMAIN-SPECIFIC AI
Tailored intelligence
6
Systems of Intelligence
FULL INTEGRATION
Complex orchestration
MATURITY JOURNEY →

USE AI AGGRESSIVELY, BUT NEVER OUTSOURCE THE HARD PART

The Equation of Learning

Struggle + Effort + Repetition = Friction + Pattern Recognition = Unique Learning + Skills

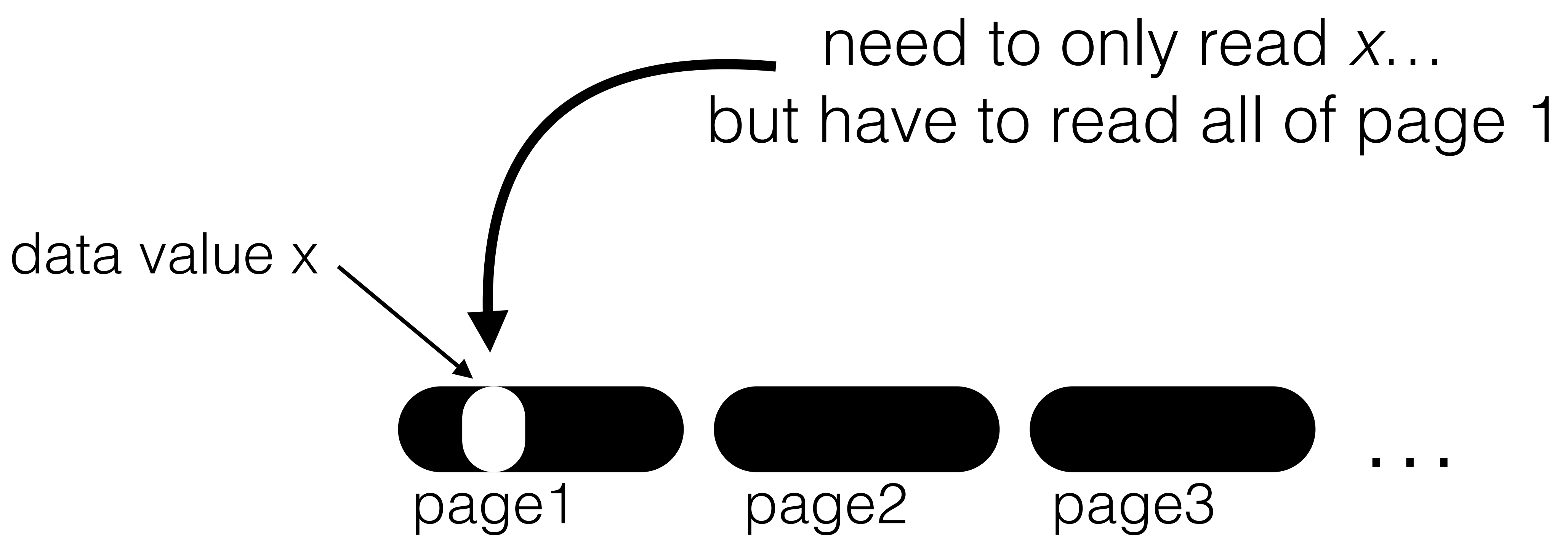
USE AI AGGRESSIVELY, BUT NEVER OUTSOURCE THE HARD PART

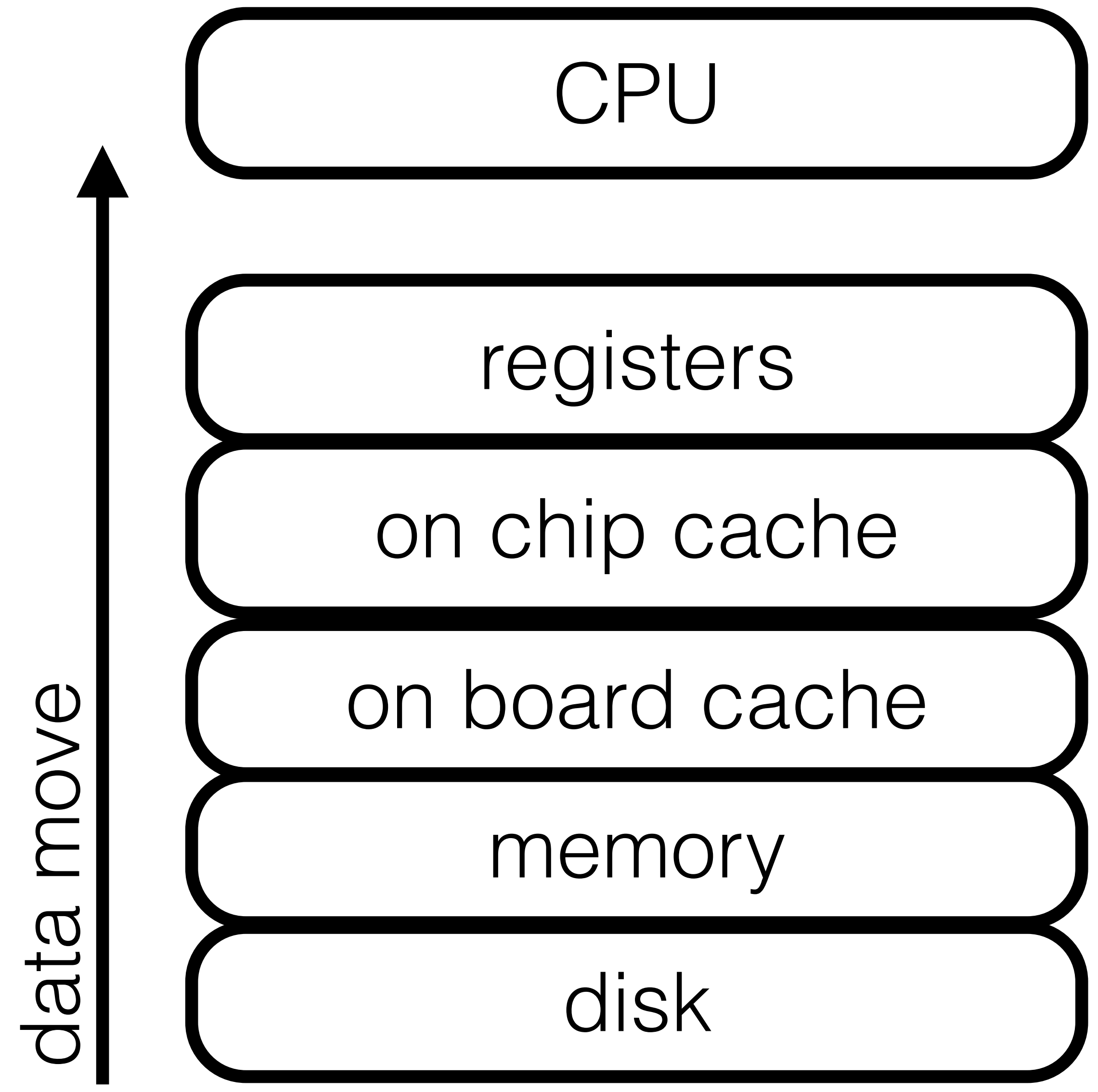
The Equation of Learning

Struggle + Effort + Repetition = Friction + Pattern Recognition = Unique Learning + Skills

Every time you push through confusion, you're building irreducible intellectual capital

If AI removes friction before your brain has learned from it,
you've traded learning for convenience


need to only read *x*…
but have to read all of page 1
data value x
page1
page2
page3
…


CPU
registers
on chip cache
on board cache
memory
disk
data move

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 →

5 10 6 4 12

(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

5 10 6 4 12

4

(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

DASlab
@ Harvard SEAS

 40 bytes

query x<5

scan

5 10 6 4 12

4

(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

 40 bytes

query x<5

scan → scan →

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

(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

DASlab @ Harvard SEAS

40 bytes

query x<5

scan → scan →

5 10 6 4 12

2 8 9 7 6

4 2

(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

80 bytes

query x<5

scan

scan

5 10 6 4 12

2 8 9 7 6

4 2

(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

DASlab @ Harvard SEAS

 80 bytes

query x<5

2 8 9 7 6 | 4 2

(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

DASlab @ Harvard SEAS

80 bytes

query x<5

scan →

(size=120 bytes)

memory level N

7 11 3 9 6 | 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

 80 bytes

query x<5

scan

(size=120 bytes)
memory level N

7 11 3 9 6 | 2 8 9 7 6 | 4 2 3

memory level N-1

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

page size: 5x8 bytes

120 bytes

query x<5

scan

(size=120 bytes)
memory level N

7 11 3 9 6 | 2 8 9 7 6 | 4 2 3

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

(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

5 10 6 4 12

(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 →

5 10 6 4 12

4

(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  40 bytes

query x<5

oracle →

5 10 6 4 12

4

(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

40 bytes

query x<5

oracle → oracle →

(size=120 bytes)

memory level N

5 10 6 4 12	2 8 9 7 6	4

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

40 bytes

query x<5

oracle → oracle →

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

(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

80 bytes

query x<5

oracle

oracle

(size=120 bytes)

memory level N

5 10 6 4 12

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  80 bytes

query x<5

(size=120 bytes)

memory level N

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  80 bytes

query x<5

oracle →

(size=120 bytes)
memory level N

7 11 3 9 6 | 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

DASlab @ Harvard SEAS

an oracle gives us the positions

80 bytes

query x<5

oracle

7 11 3 9 6

2 8 9 7 6

4 2 3

(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  120 bytes

query x<5

oracle →

(size=120 bytes)
memory level N

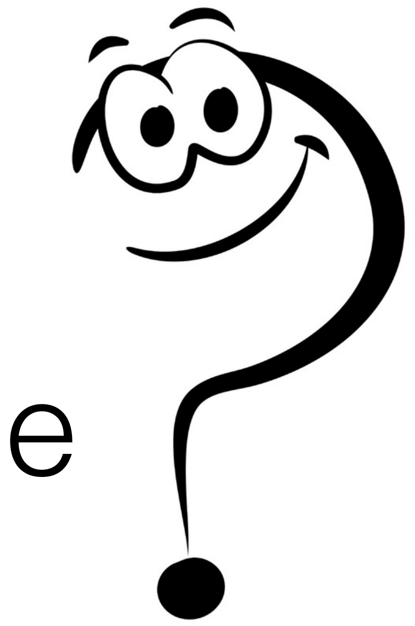
7 11 3 9 6 | 2 8 9 7 6 | 4 2 3

memory level N-1

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

page size: 5x8 bytes

when does it make sense to have an oracle
how can we minimize the cost



e.g., **query** x<5

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

algorithm/system design = not just computation

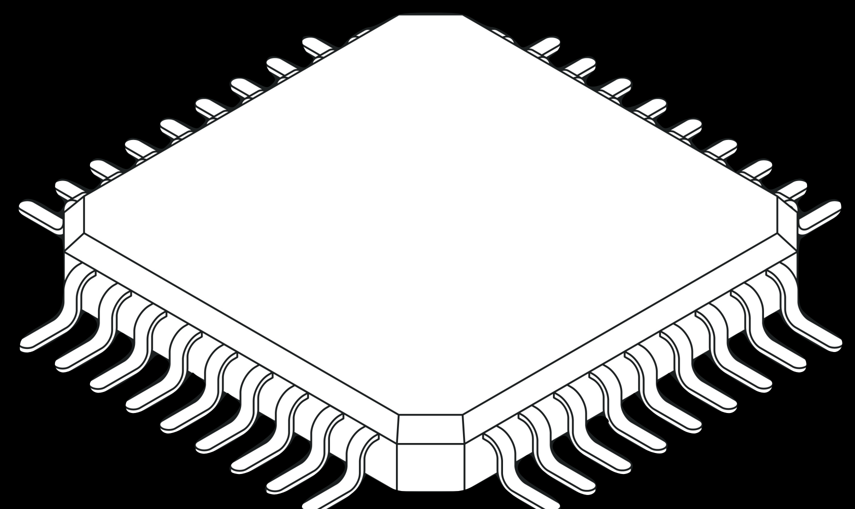
algorithm/system design = not just computation

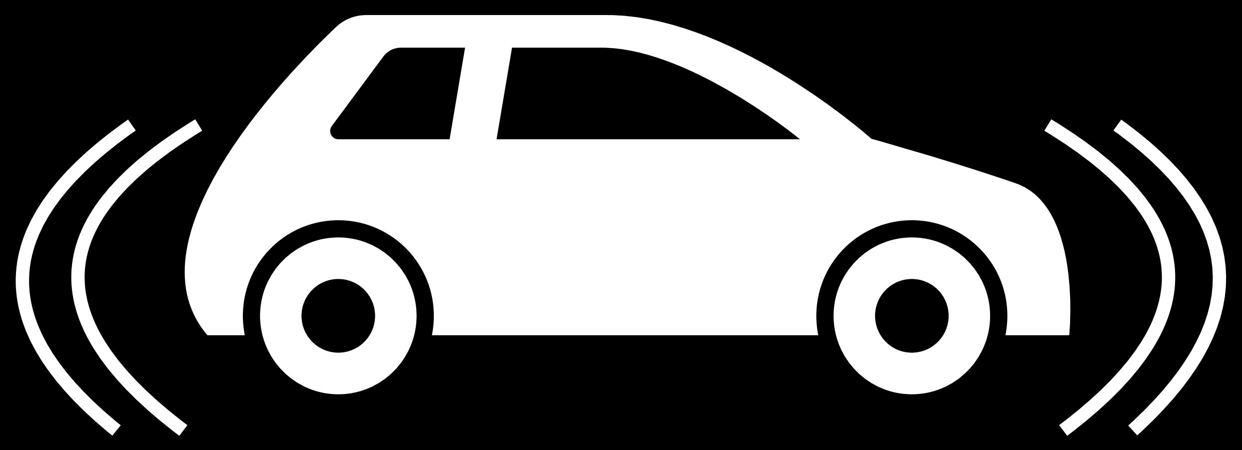
Is there maybe a perfect system? Nope…

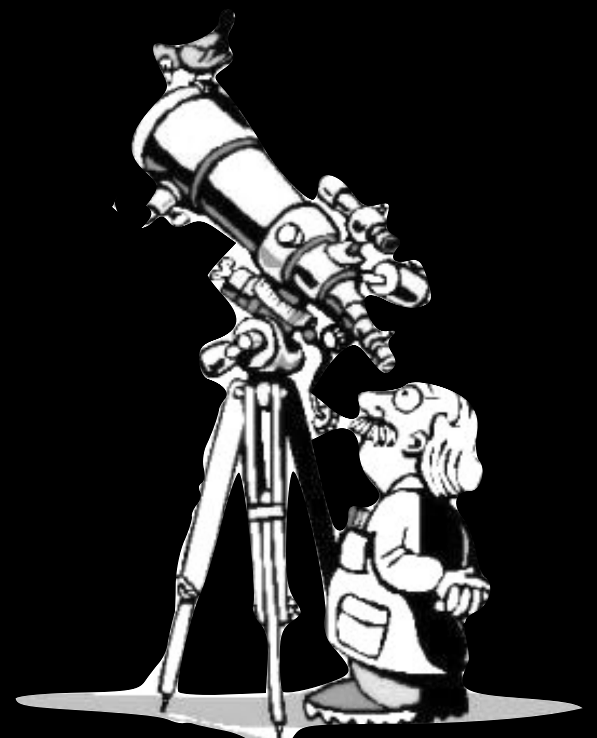
Intro into high-level ideas for

Self-designing Systems

The problem: as the big data/AI world keeps changing…

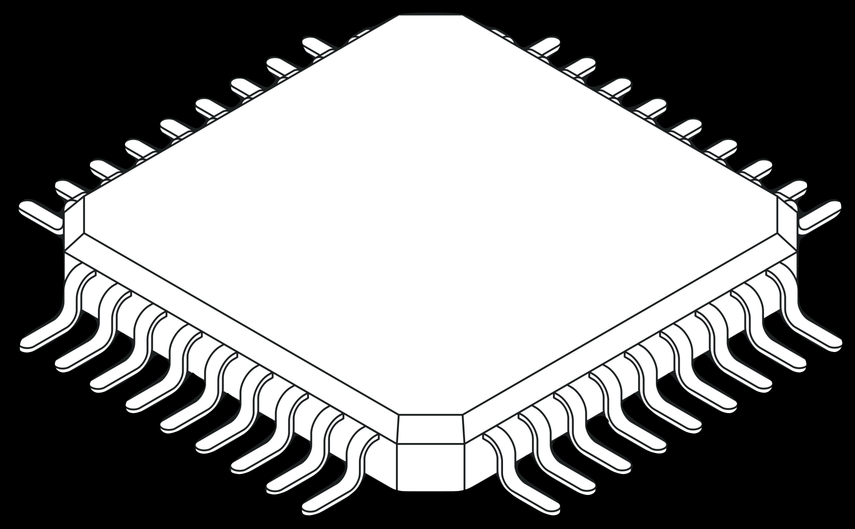


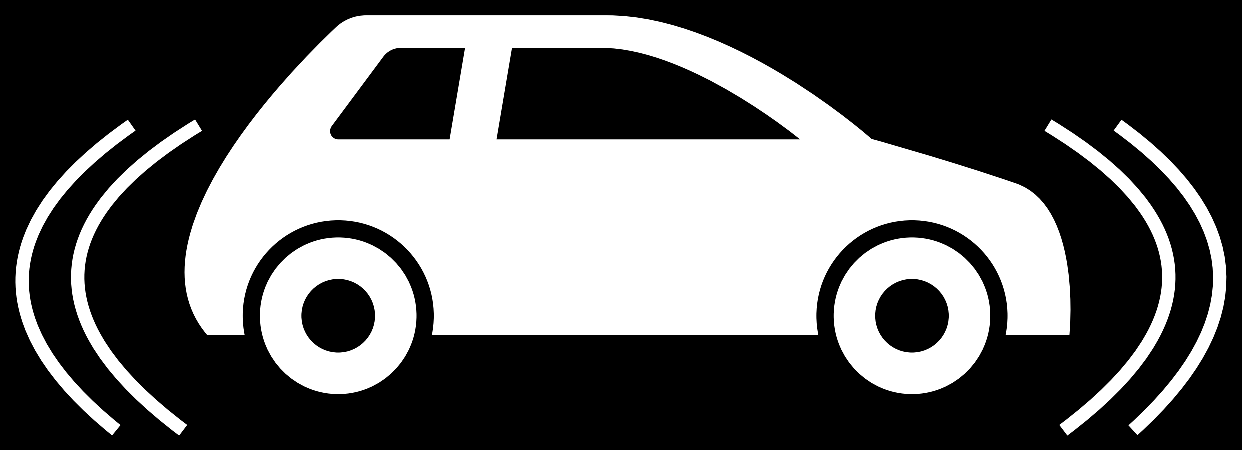


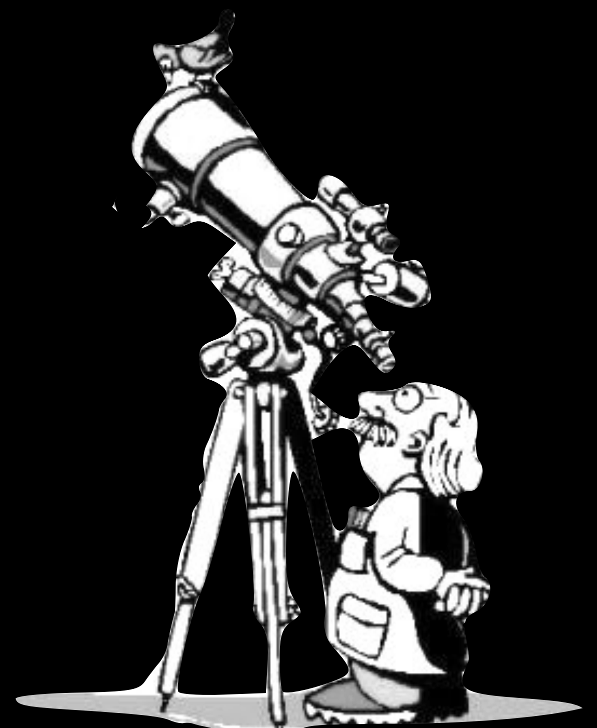


The problem: as the big data/AI world keeps changing…

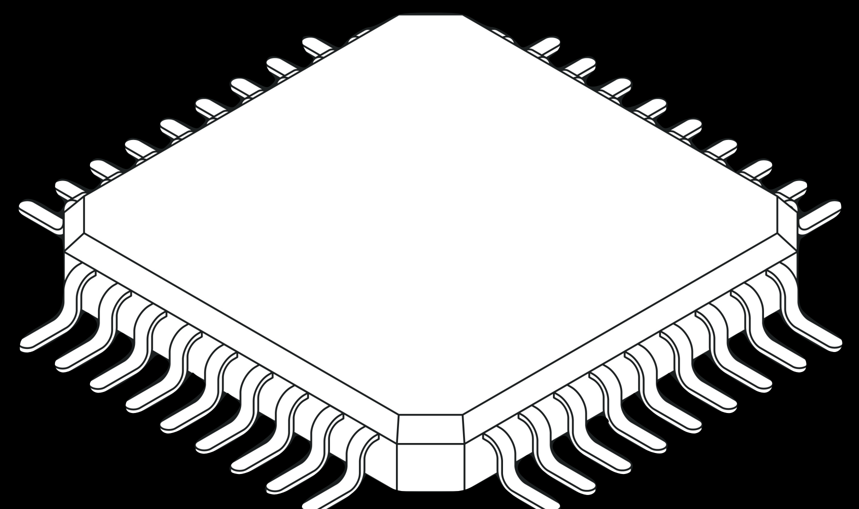
there is a continuous need for new data systems
but it is **extremely hard to design & build new systems**

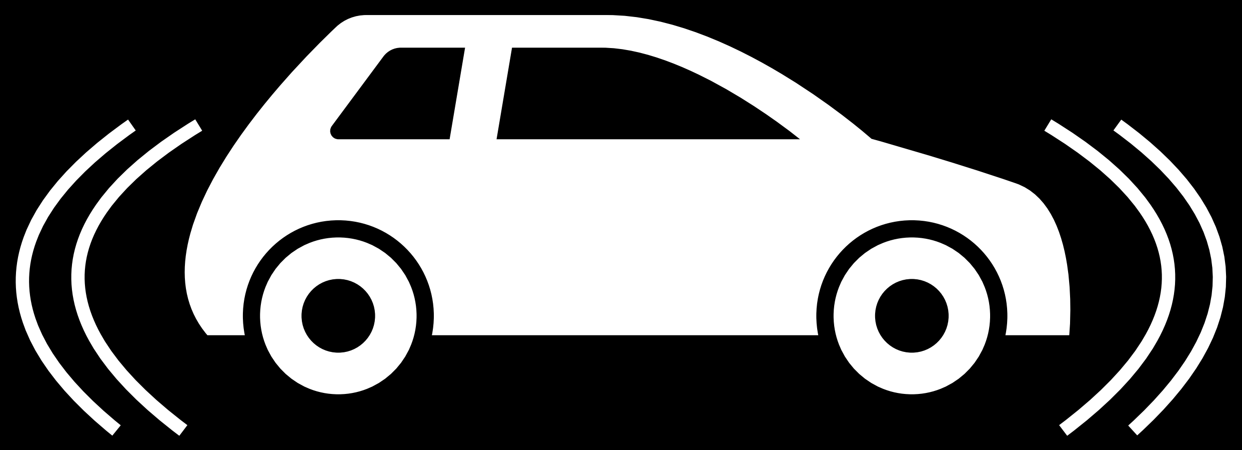


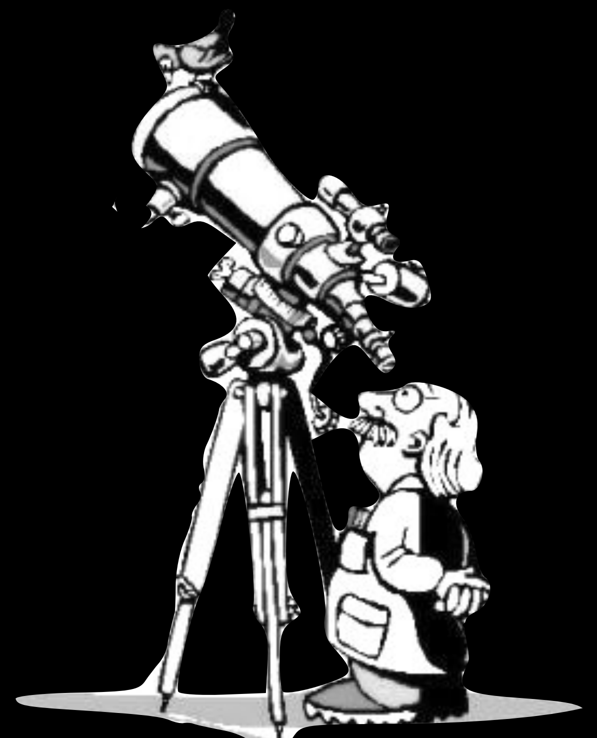




How do we design a system that is **X times faster for a workload W**?

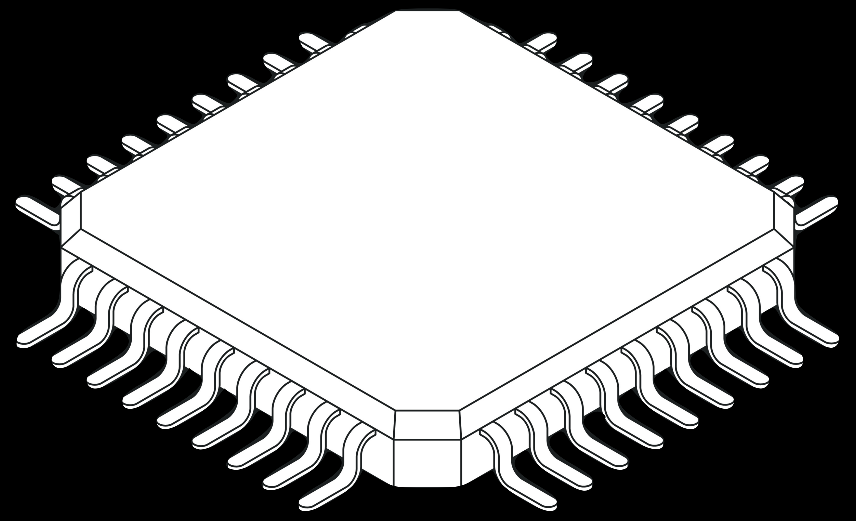


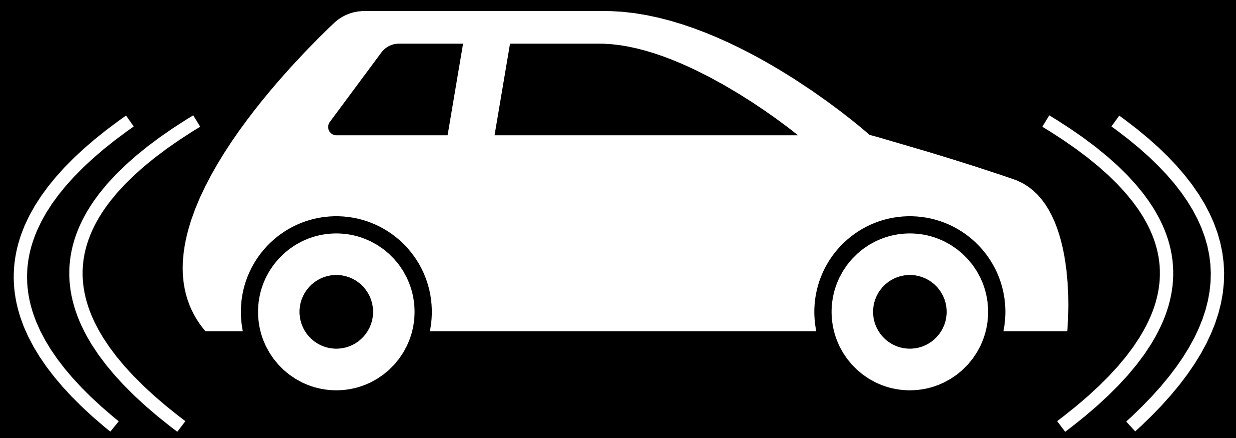


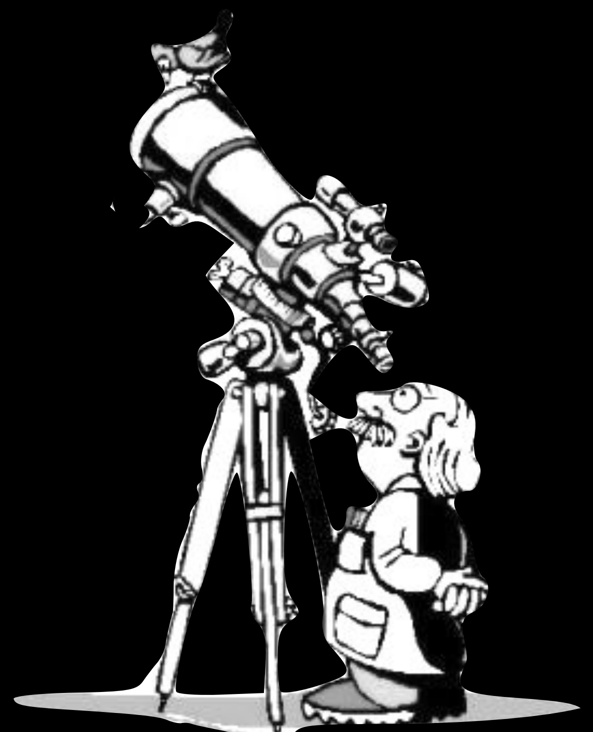


How do we design a system that is **X times faster for a workload W**?

How do we design a system that allows for control of **cloud cost**?



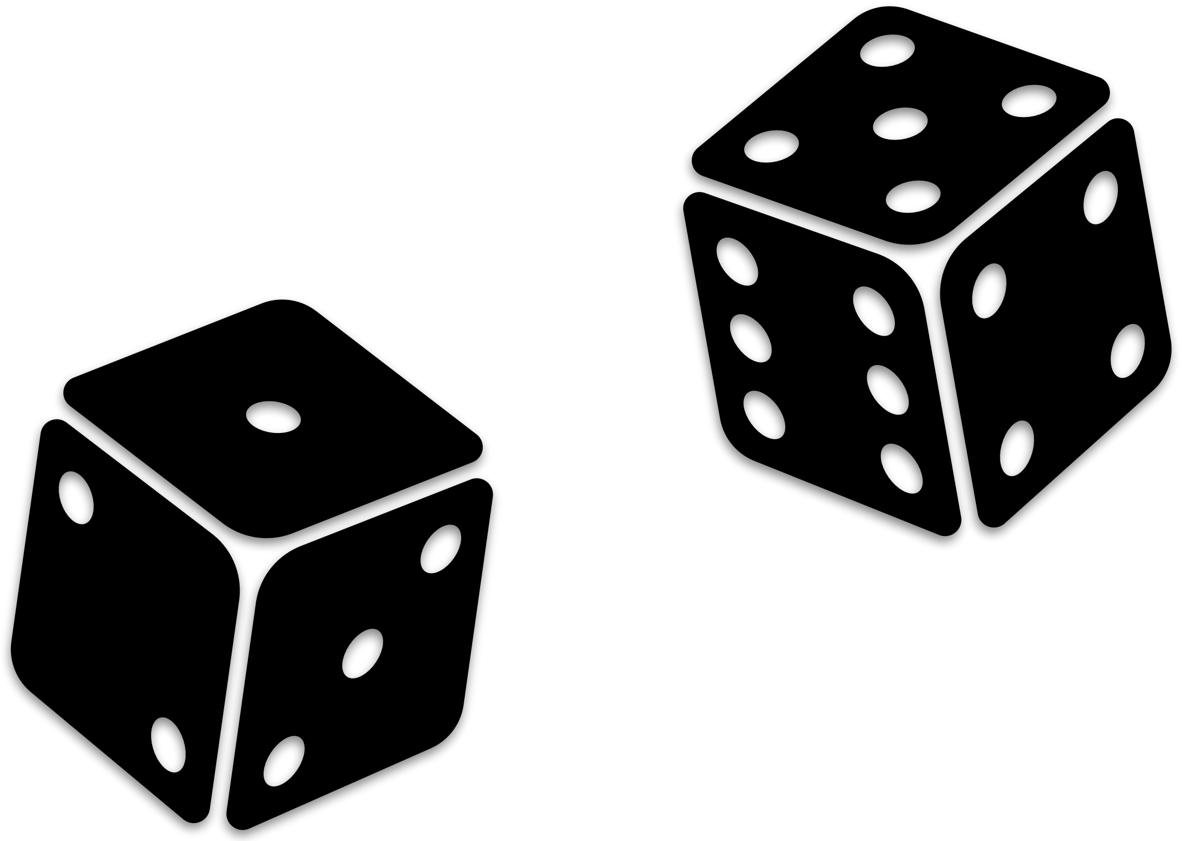




How do we design a system that is **X times faster for a workload W**?

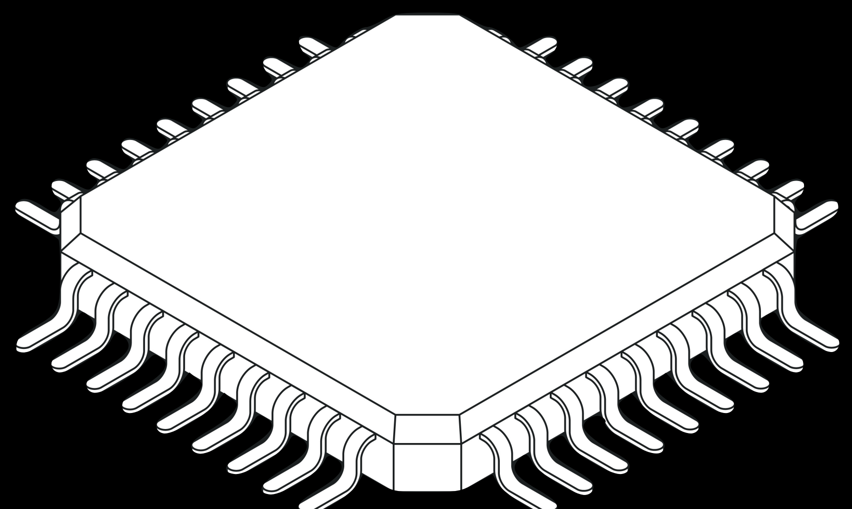
How do we design a system that allows for control of **cloud cost**?

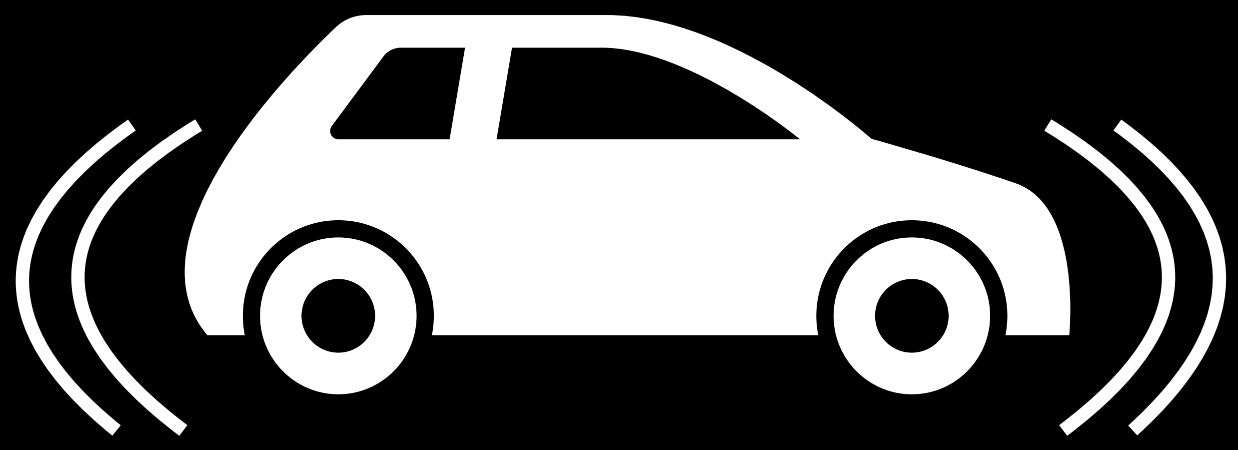
What happens if we introduce **new application feature** Y?

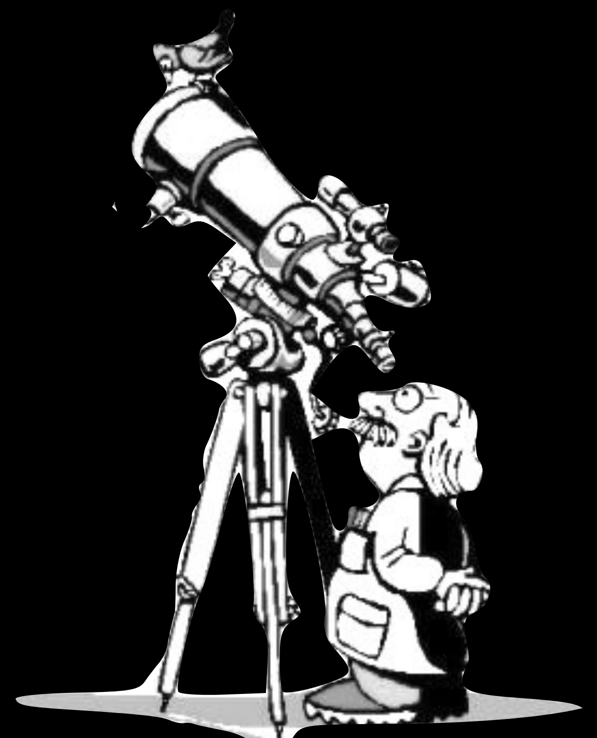


Should we **upgrade** to new version Z?

What will **break** our system?





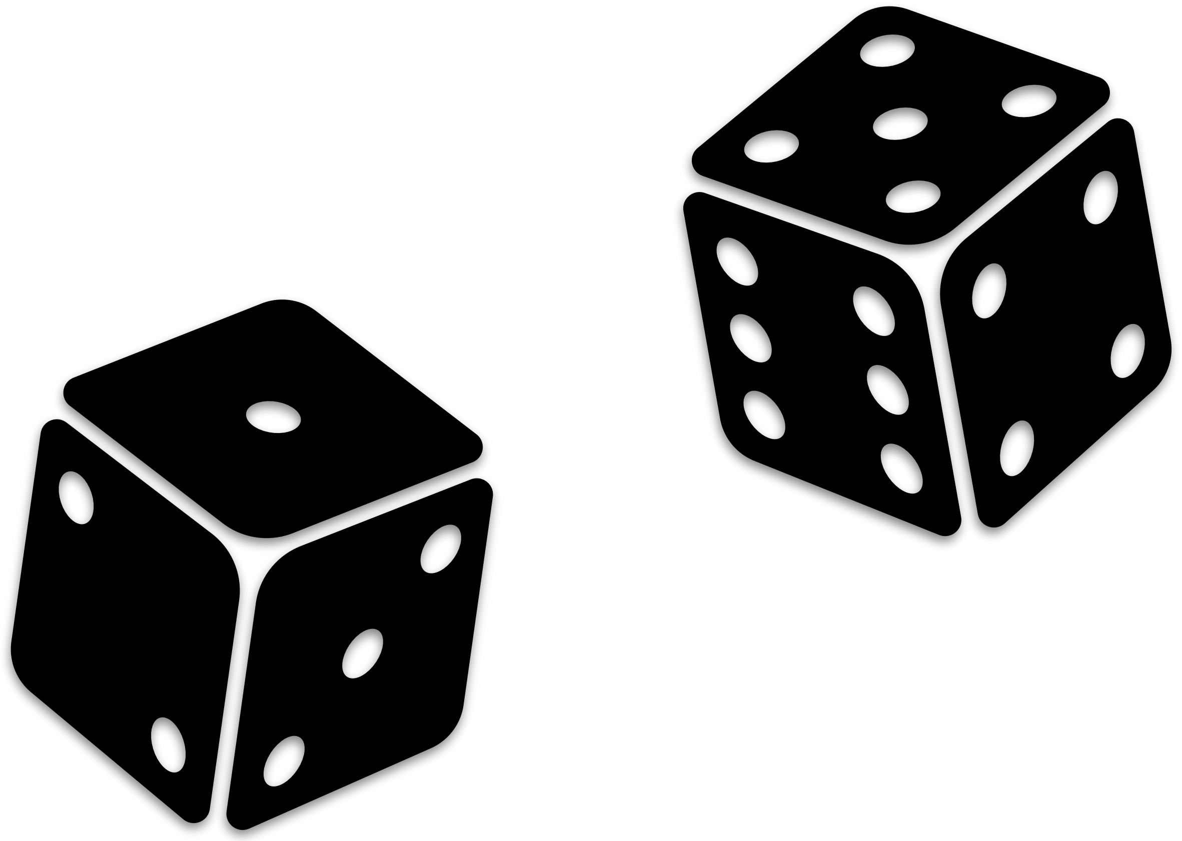


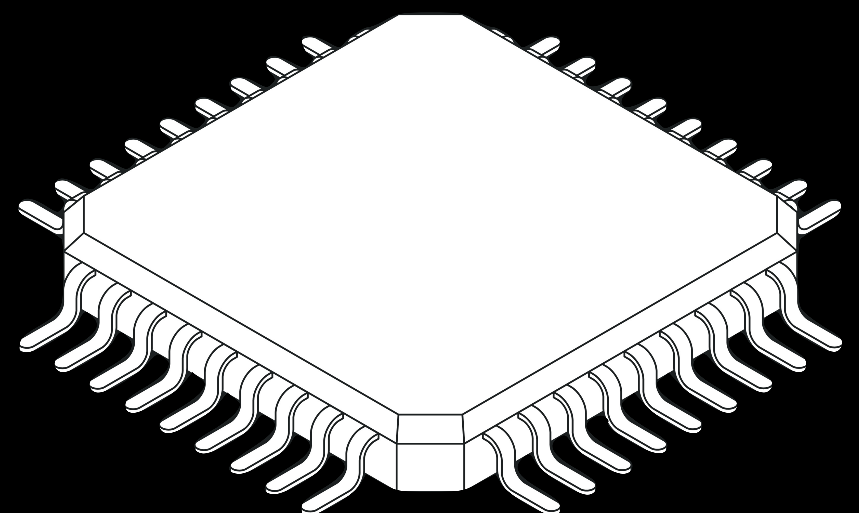
BOTTLENECK: SUB-OPTIMAL SYSTEMS

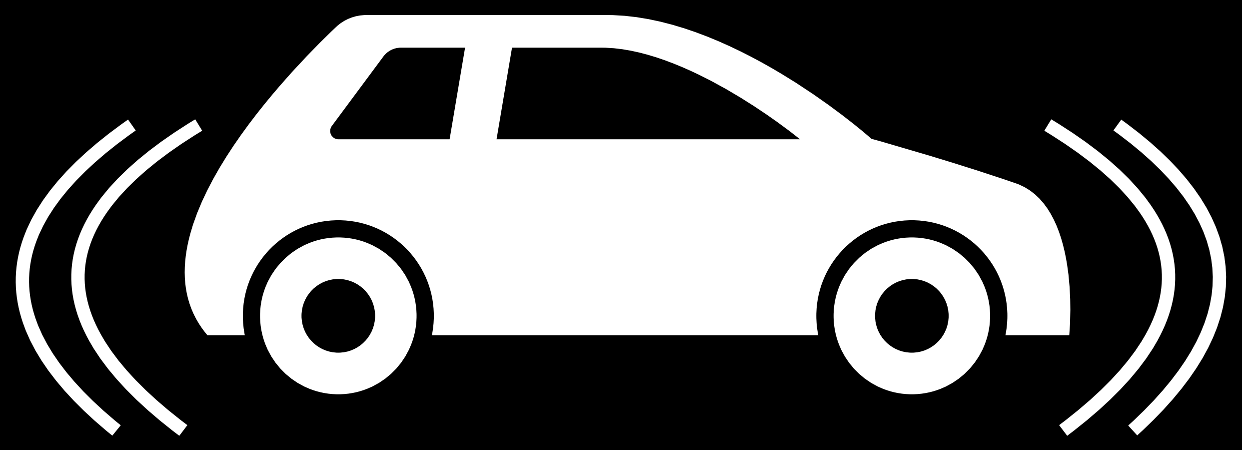
What happens if we introduce **new application feature** Y?

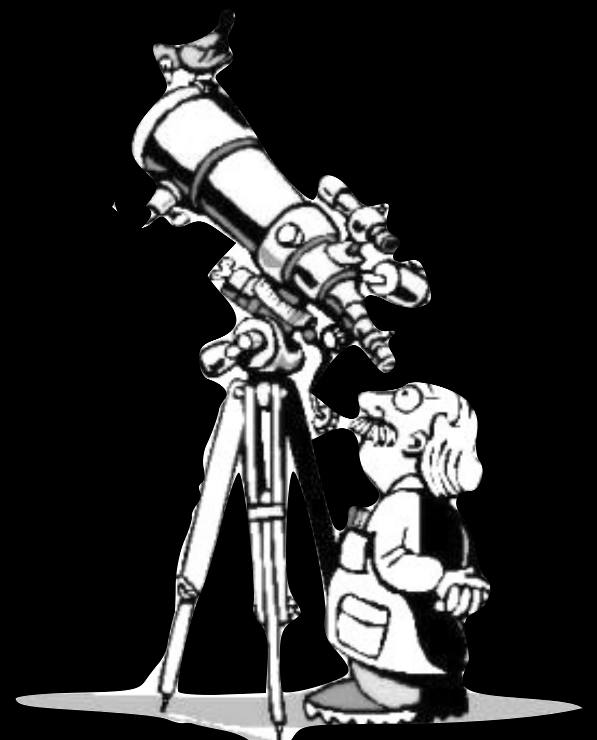
Should we **upgrade** to new version Z?

What will **break** our system?





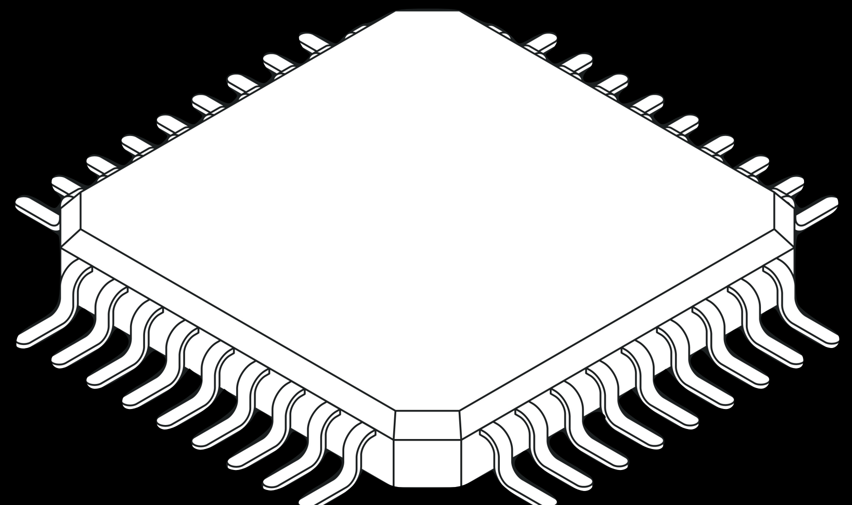


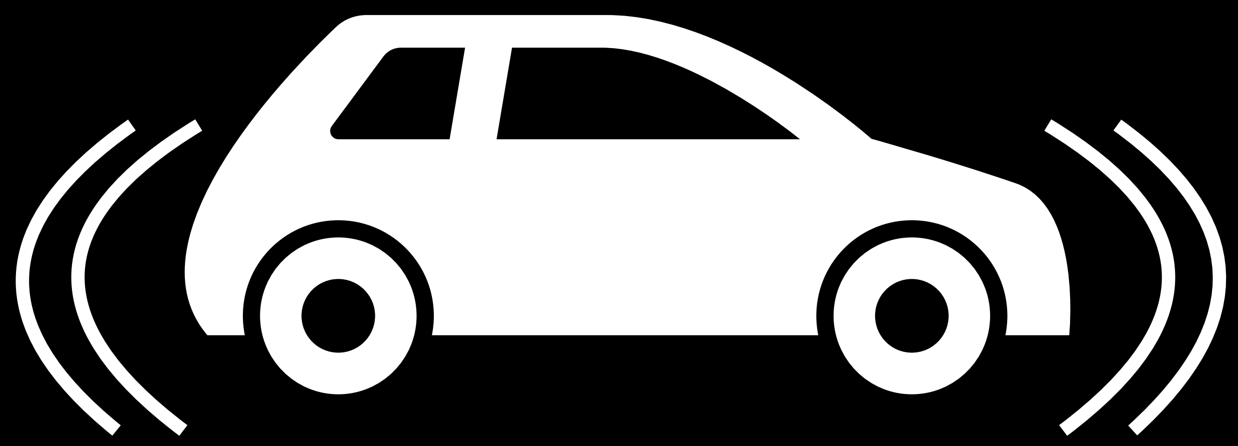


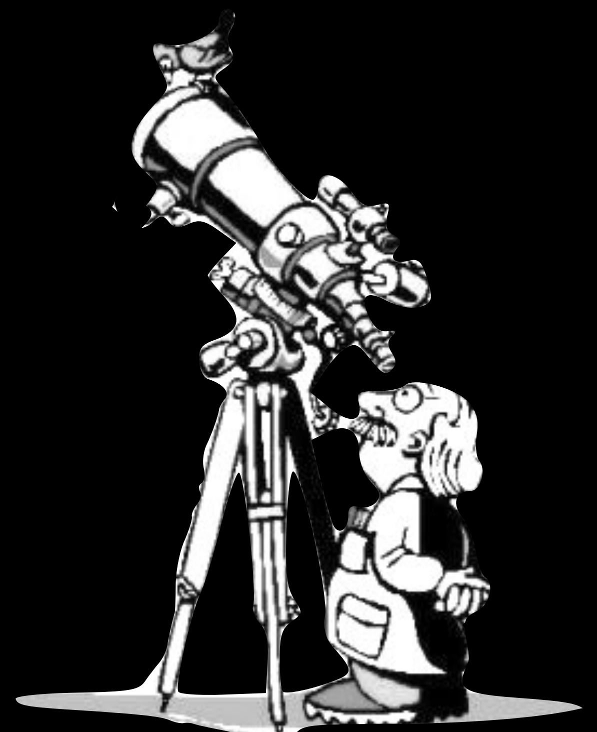
BOTTLENECK: SUB-OPTIMAL SYSTEMS

huge cloud cost

environmental impact





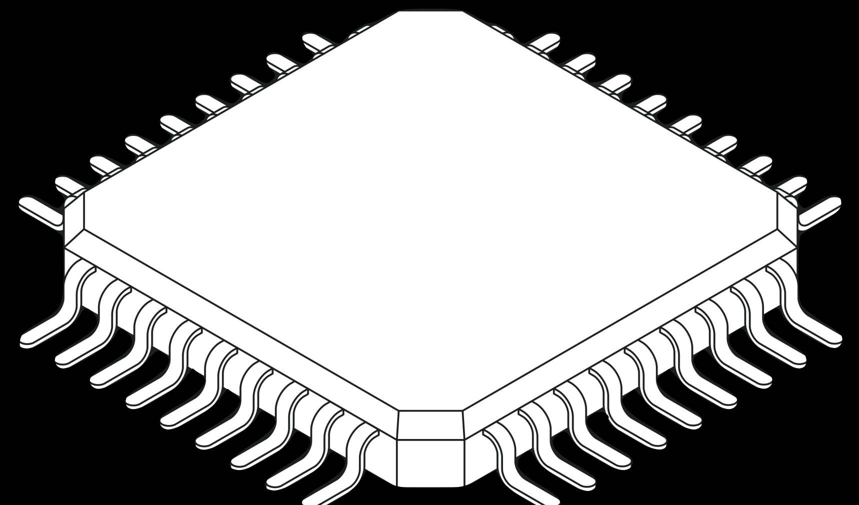


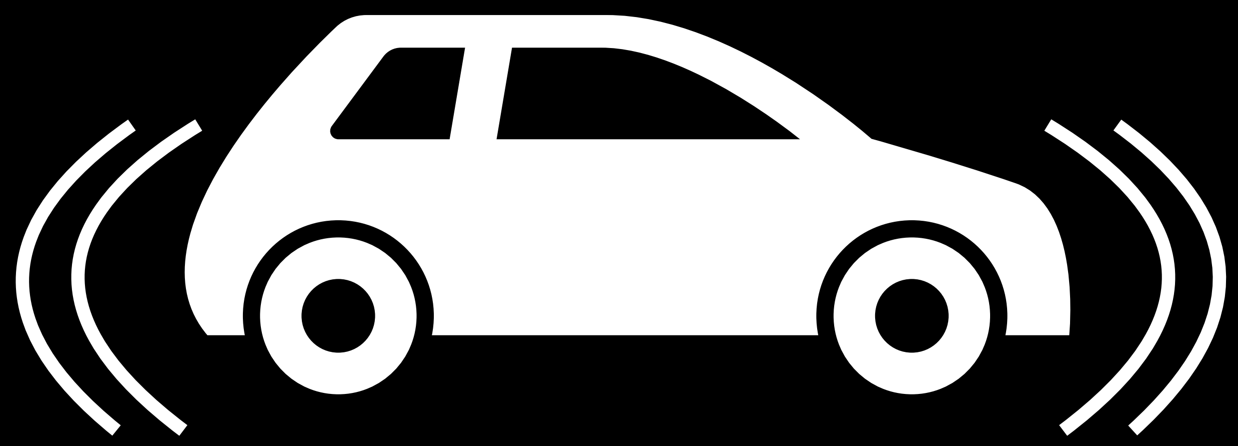
BOTTLENECK: SUB-OPTIMAL SYSTEMS

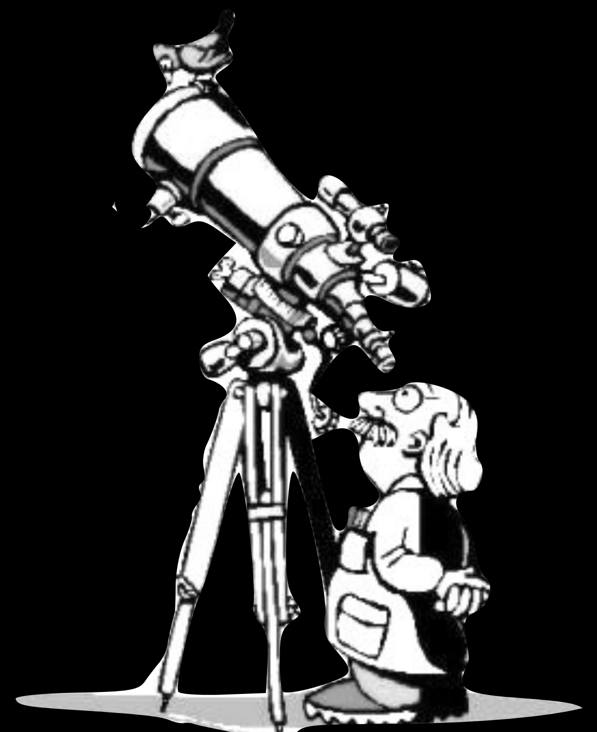
expensive transitions

huge cloud cost

environmental impact







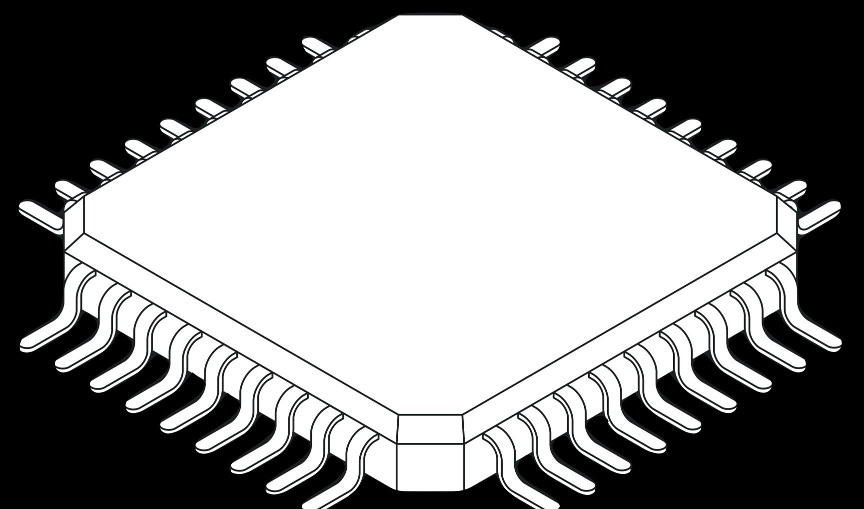
BOTTLENECK: SUB-OPTIMAL SYSTEMS

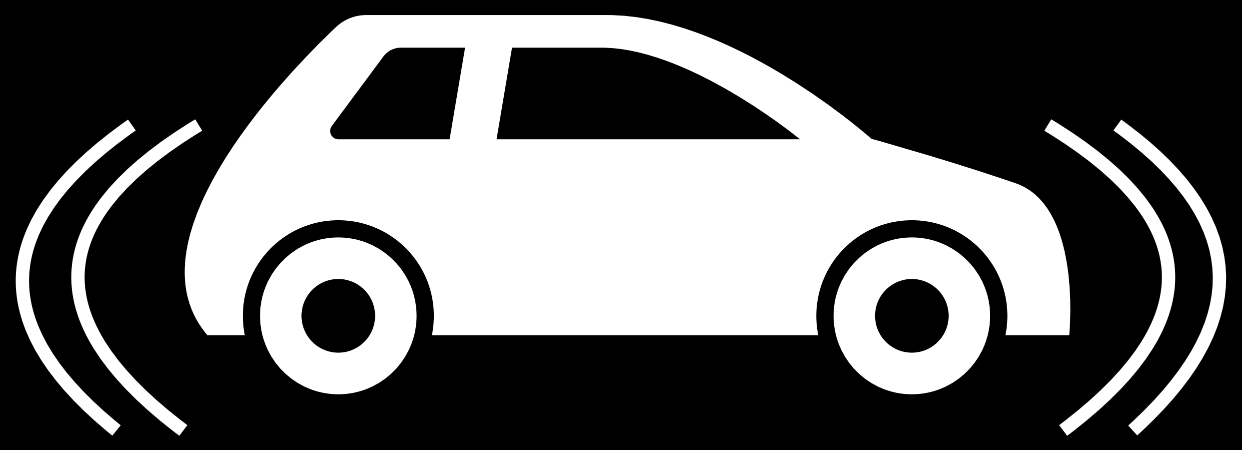
expensive transitions

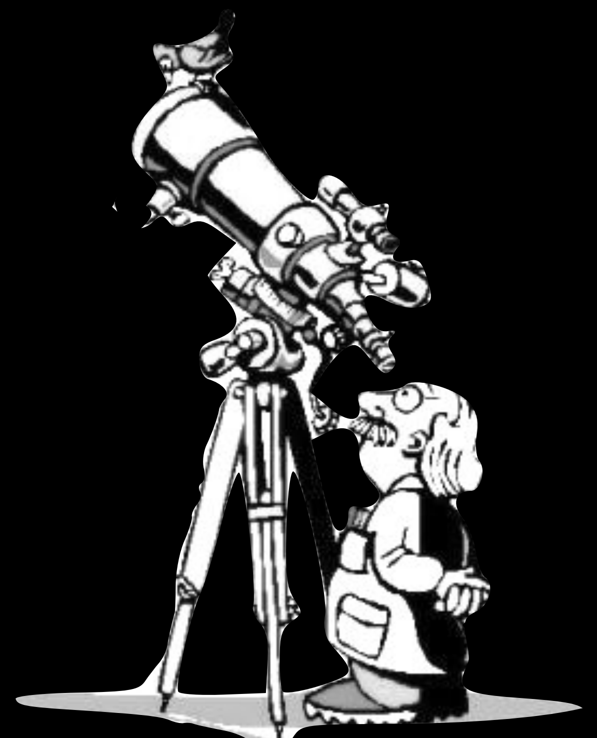
huge cloud cost

application feasibility

environmental impact





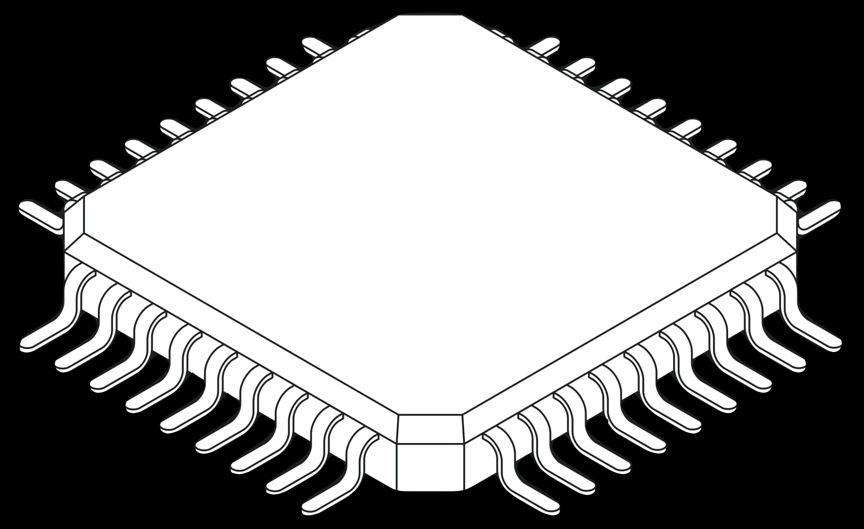


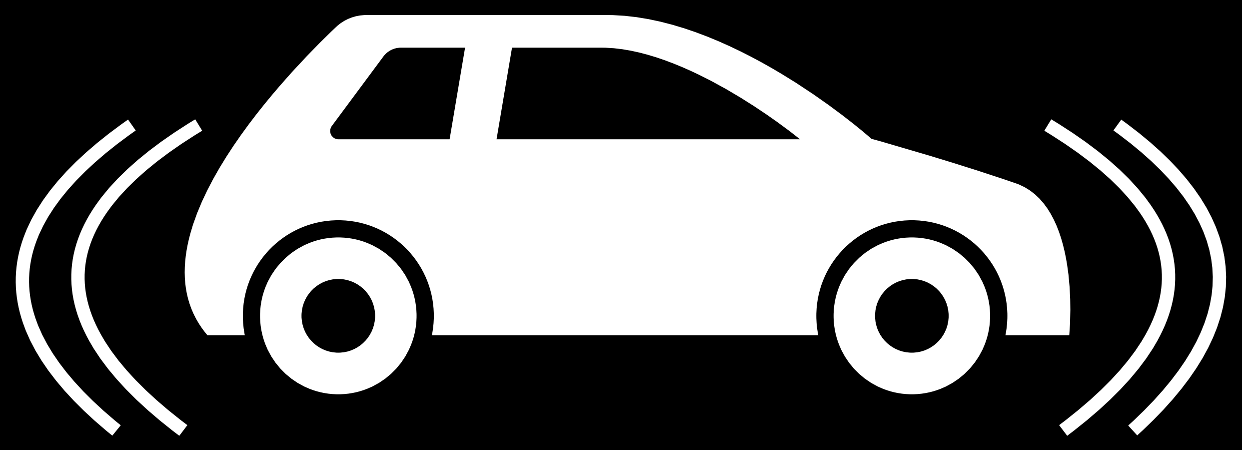
BOTTLENECK: SUB-OPTIMAL SYSTEMS

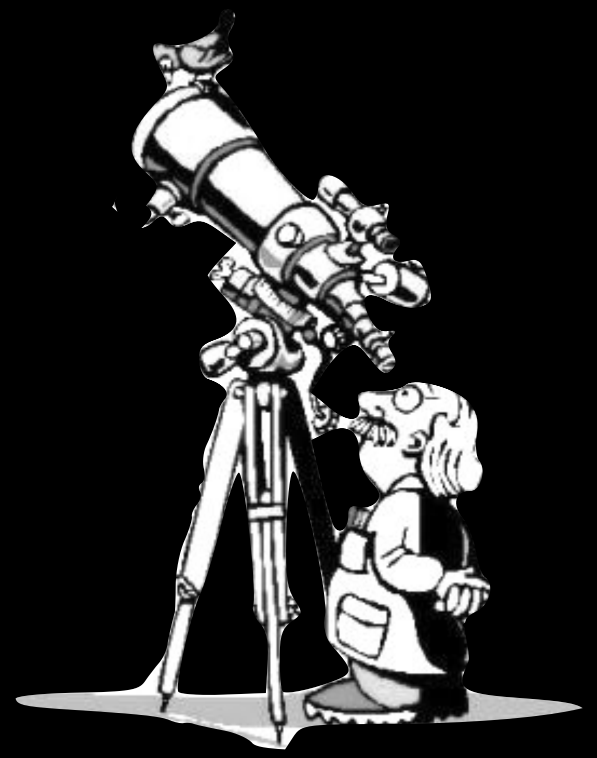
huge cloud cost   expensive transitions   application feasibility   environmental impact

complexity

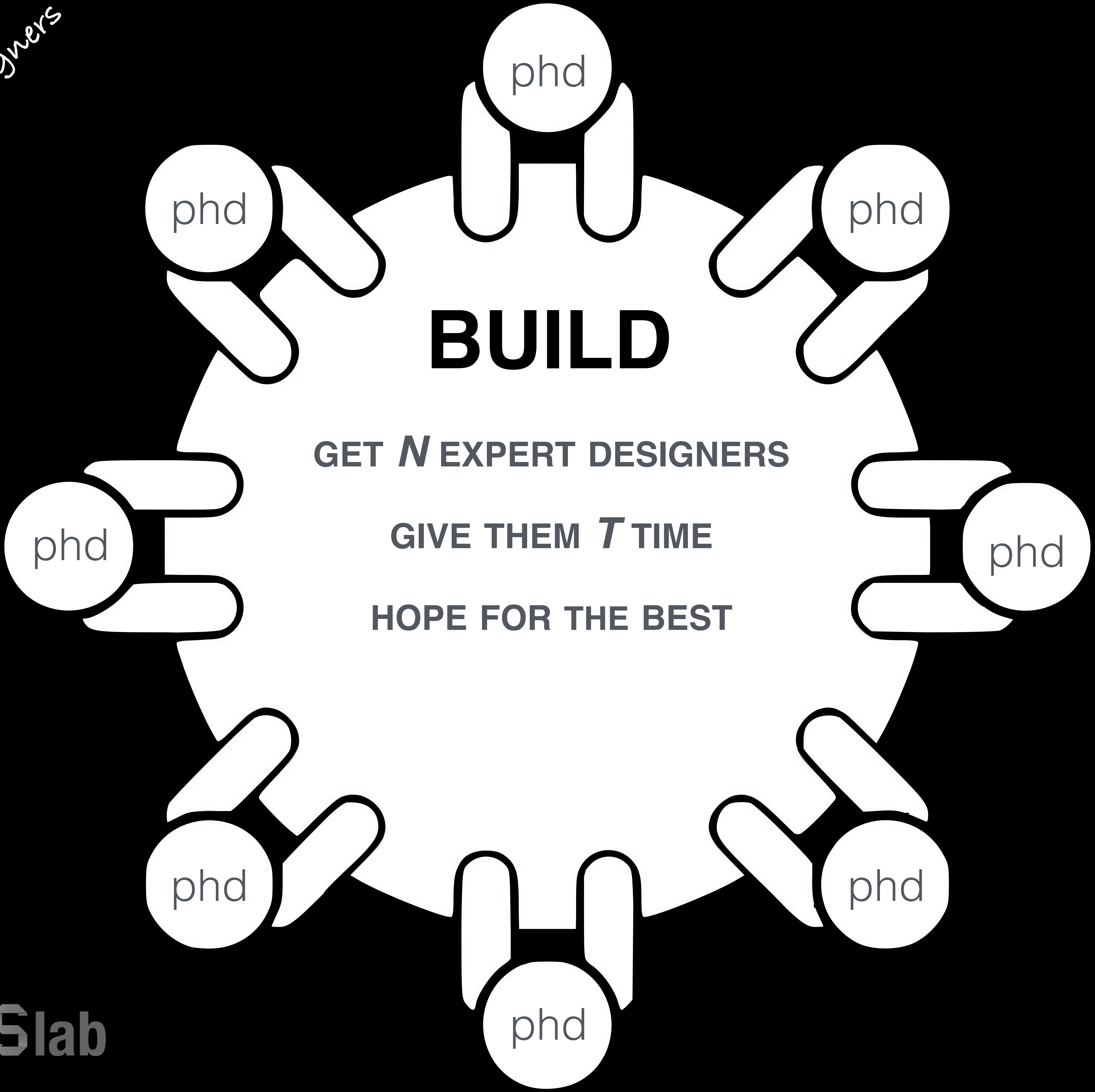
how we **BUILD** systems

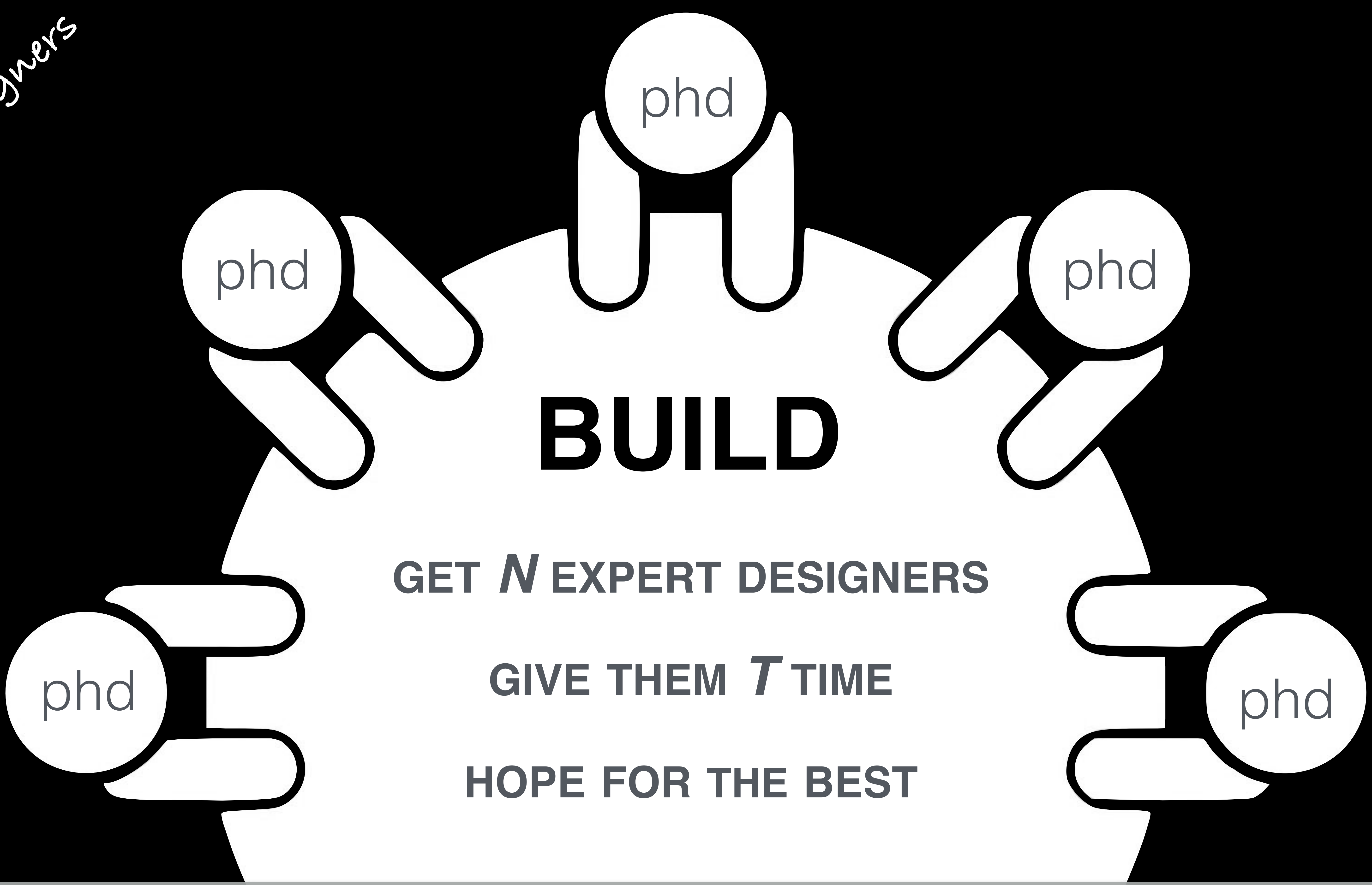


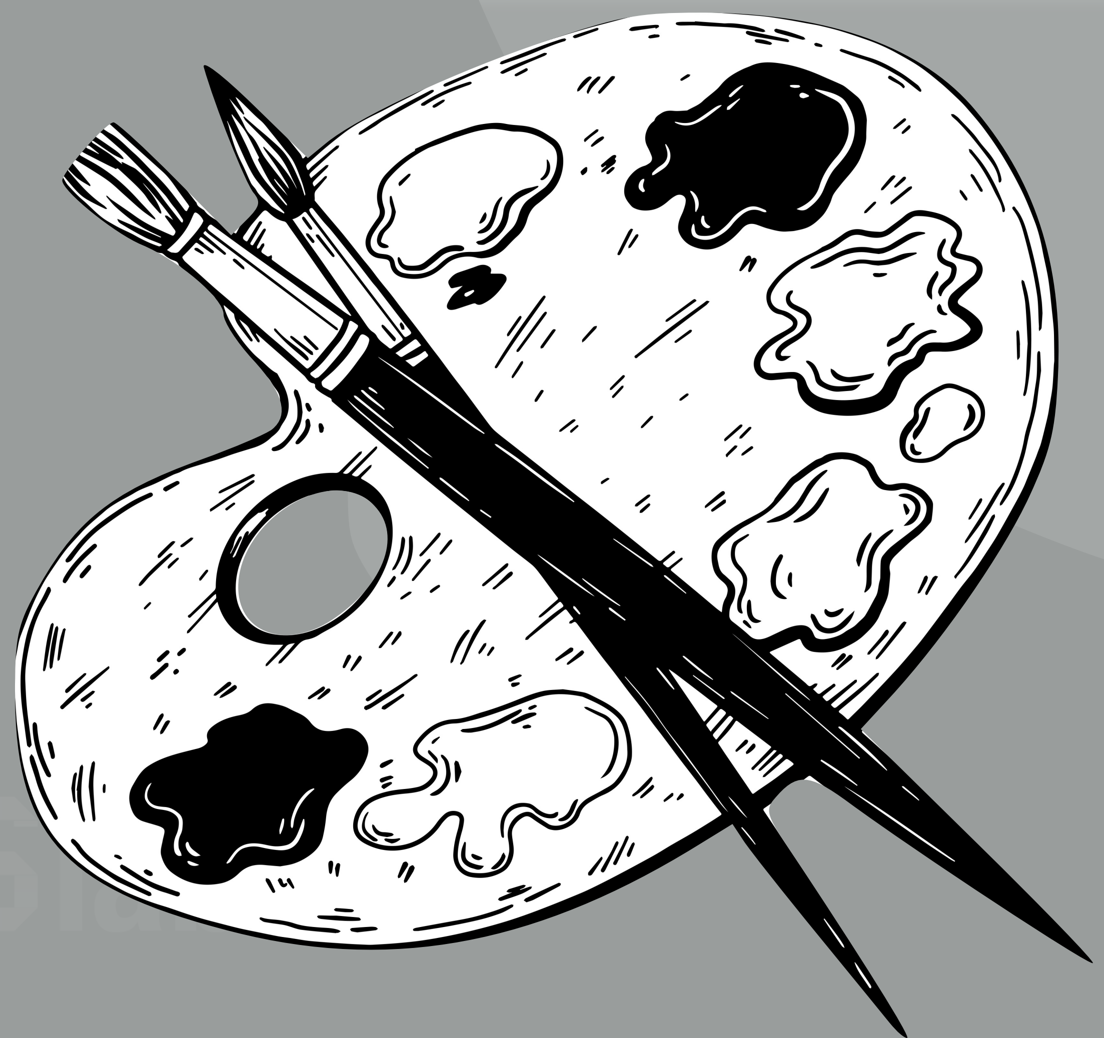




the dinning
systems designers







design is an art

the dinning
systems designers

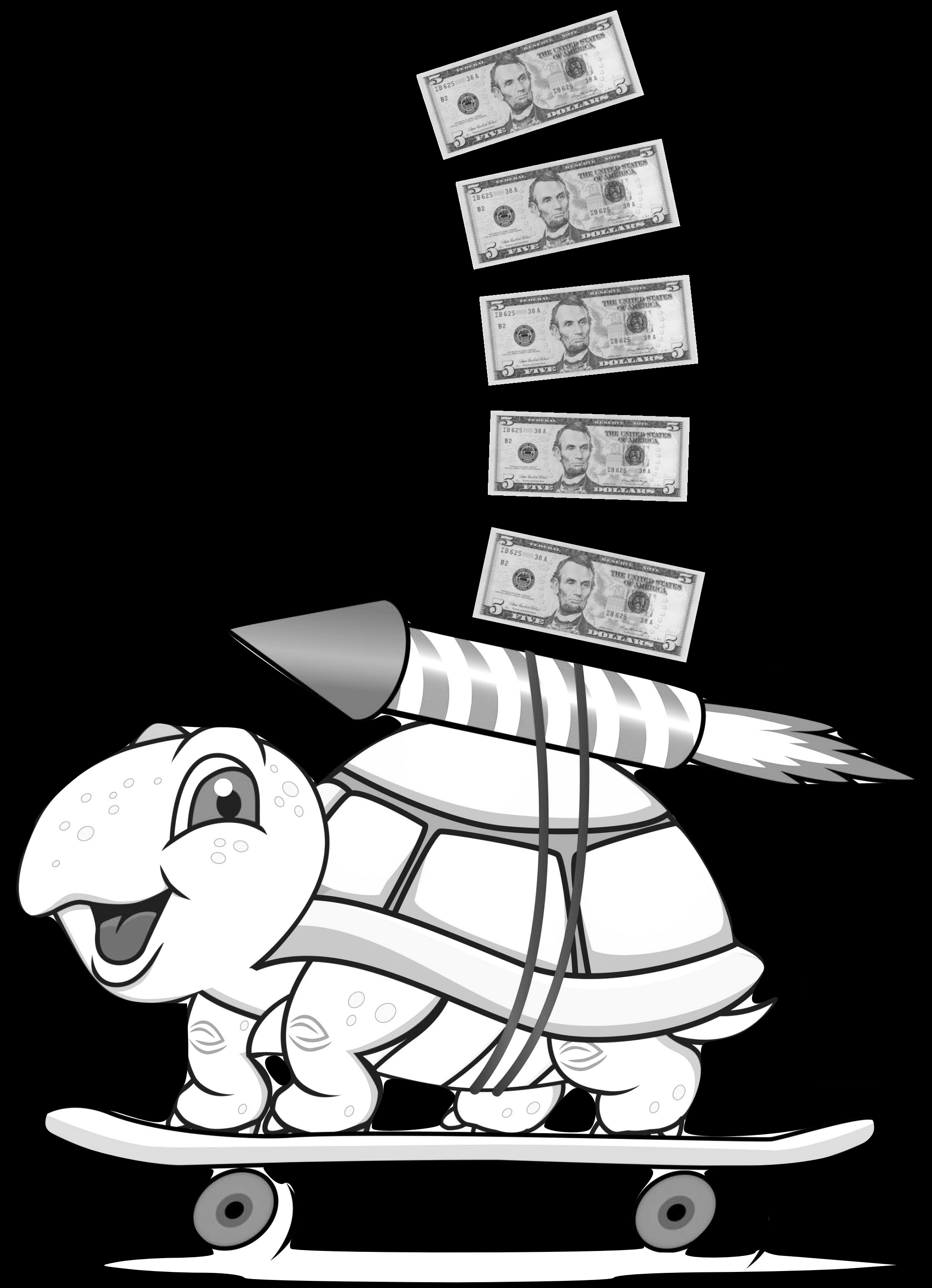
phd phd phd phd phd

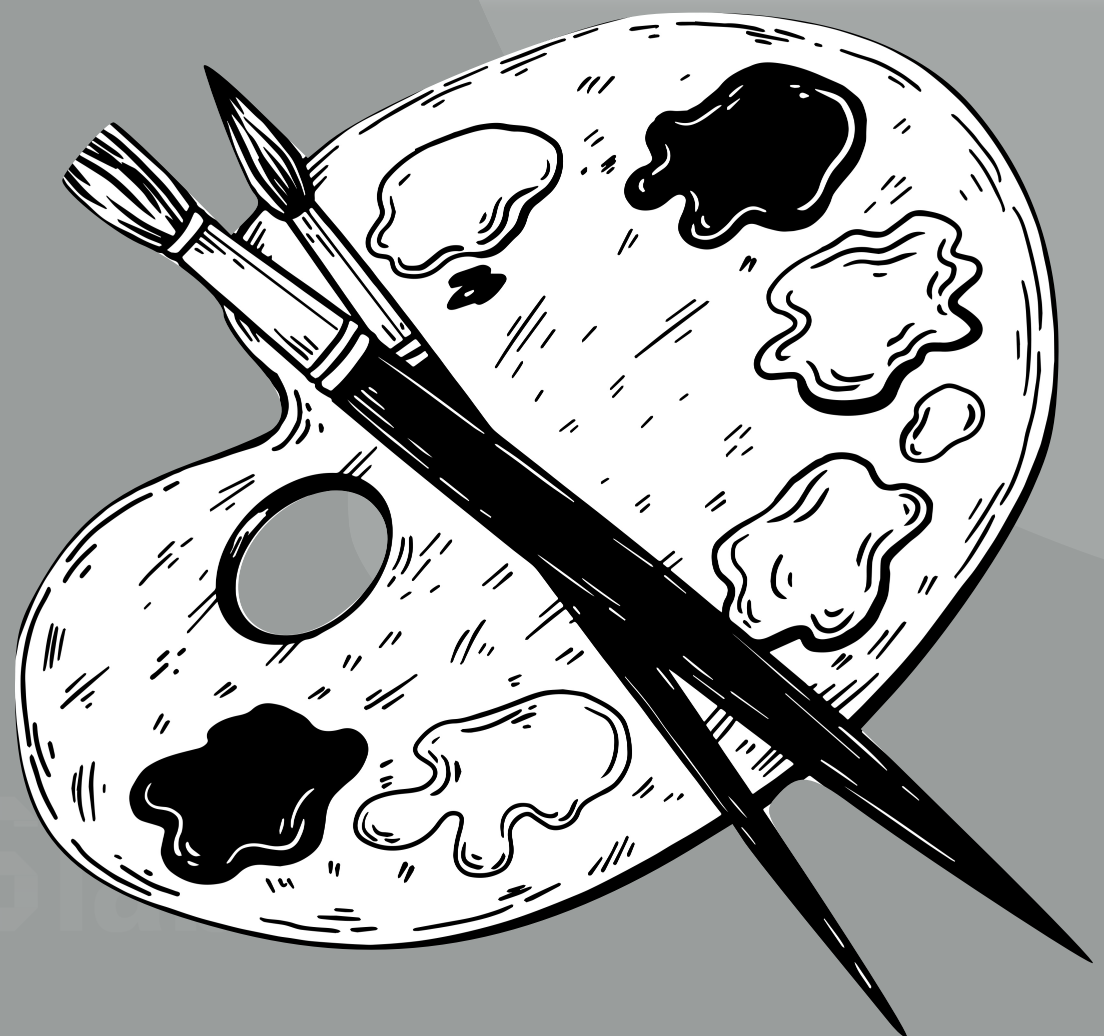
BUILD

GET *N* EXPERT DESIGNERS

GIVE THEM *T* TIME

HOPE FOR THE BEST





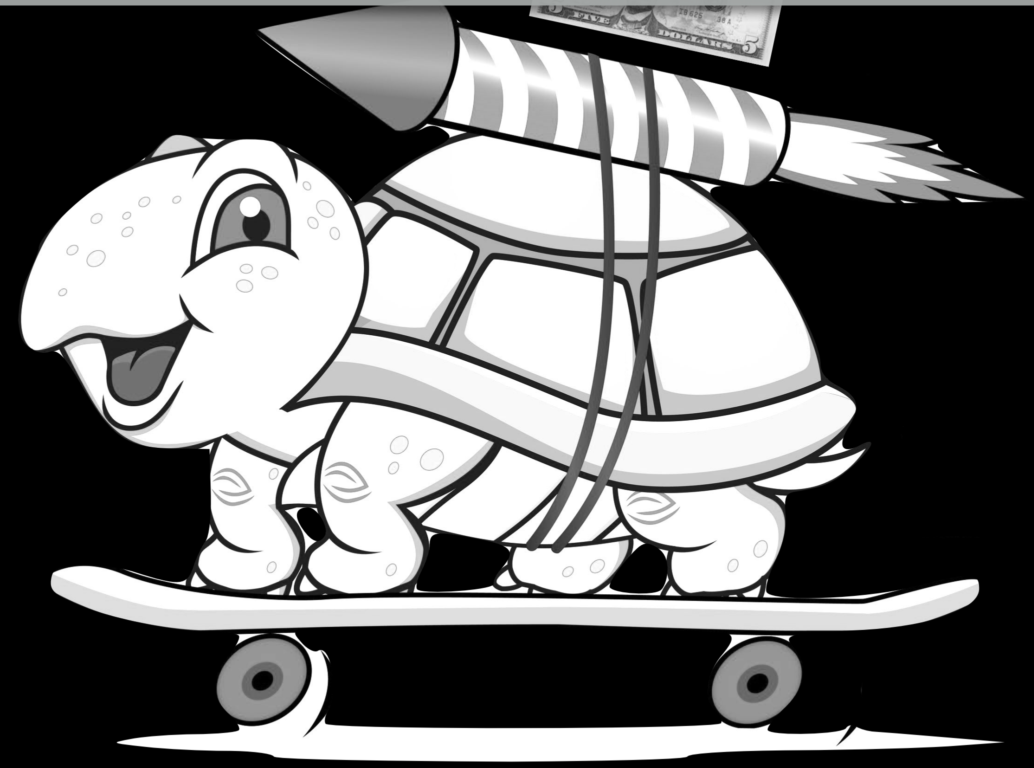
design is an art

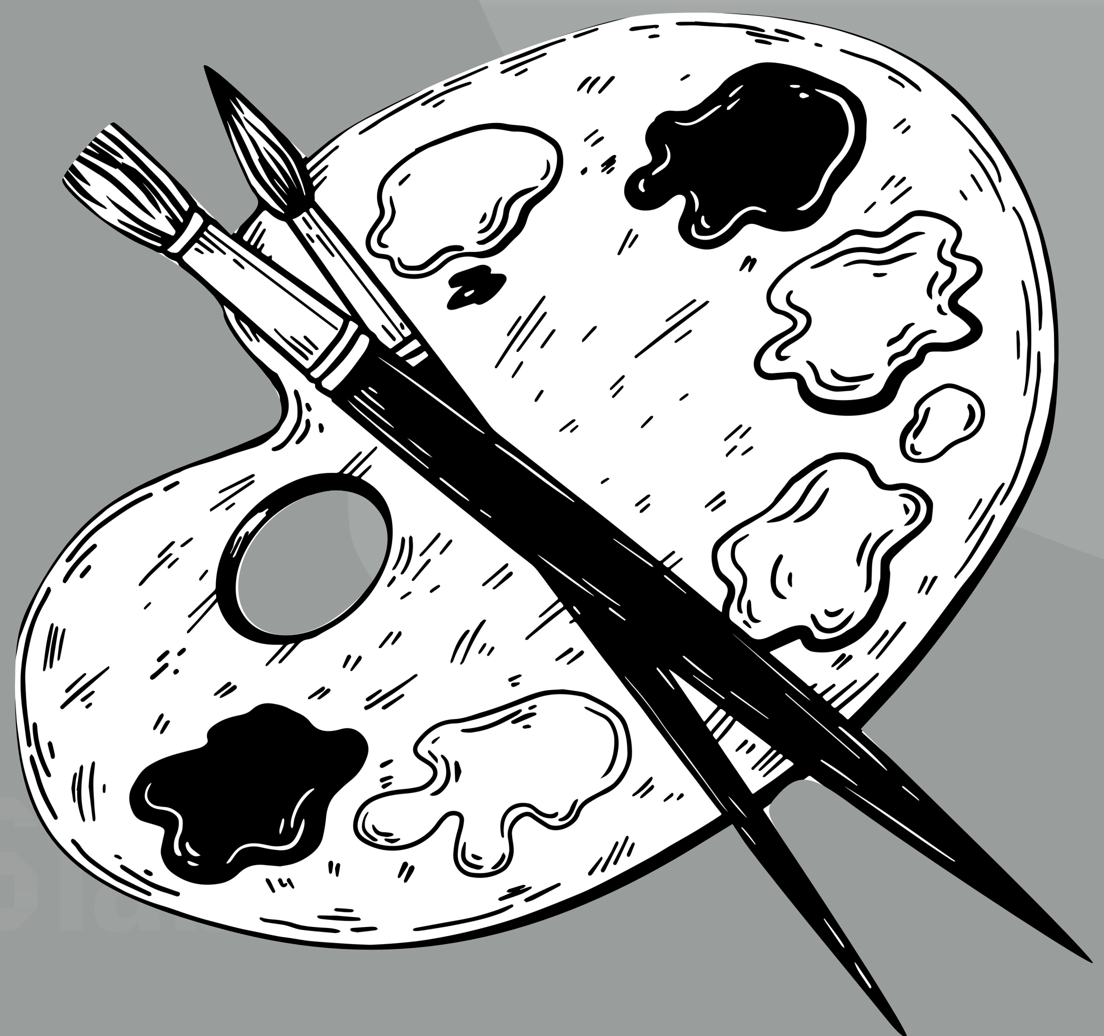
Design: 6-7 years
Reasoning: months/impossible

GET *N* EXPERT DESIGNERS

GIVE THEM *T* TIME

HOPE FOR THE BEST



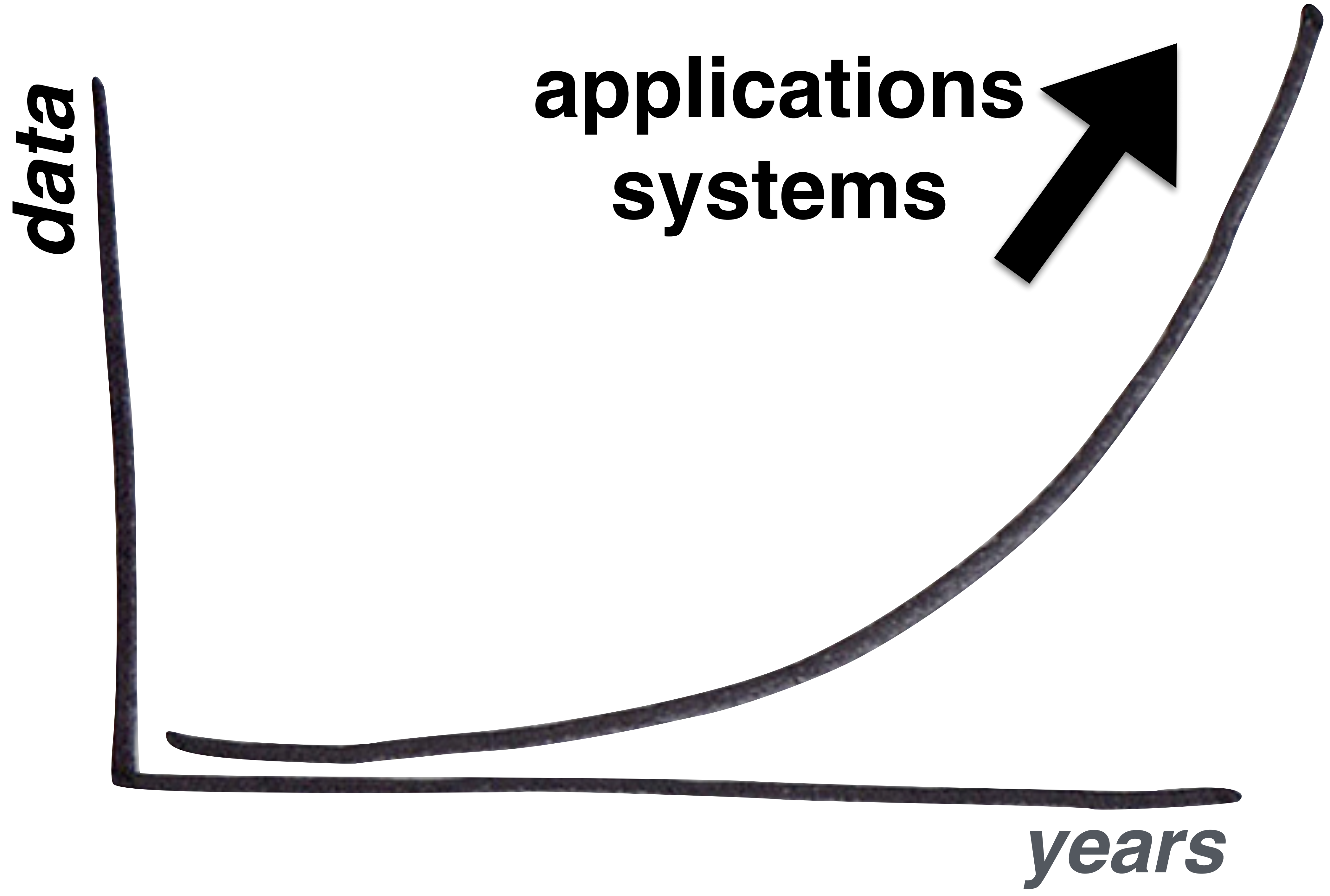


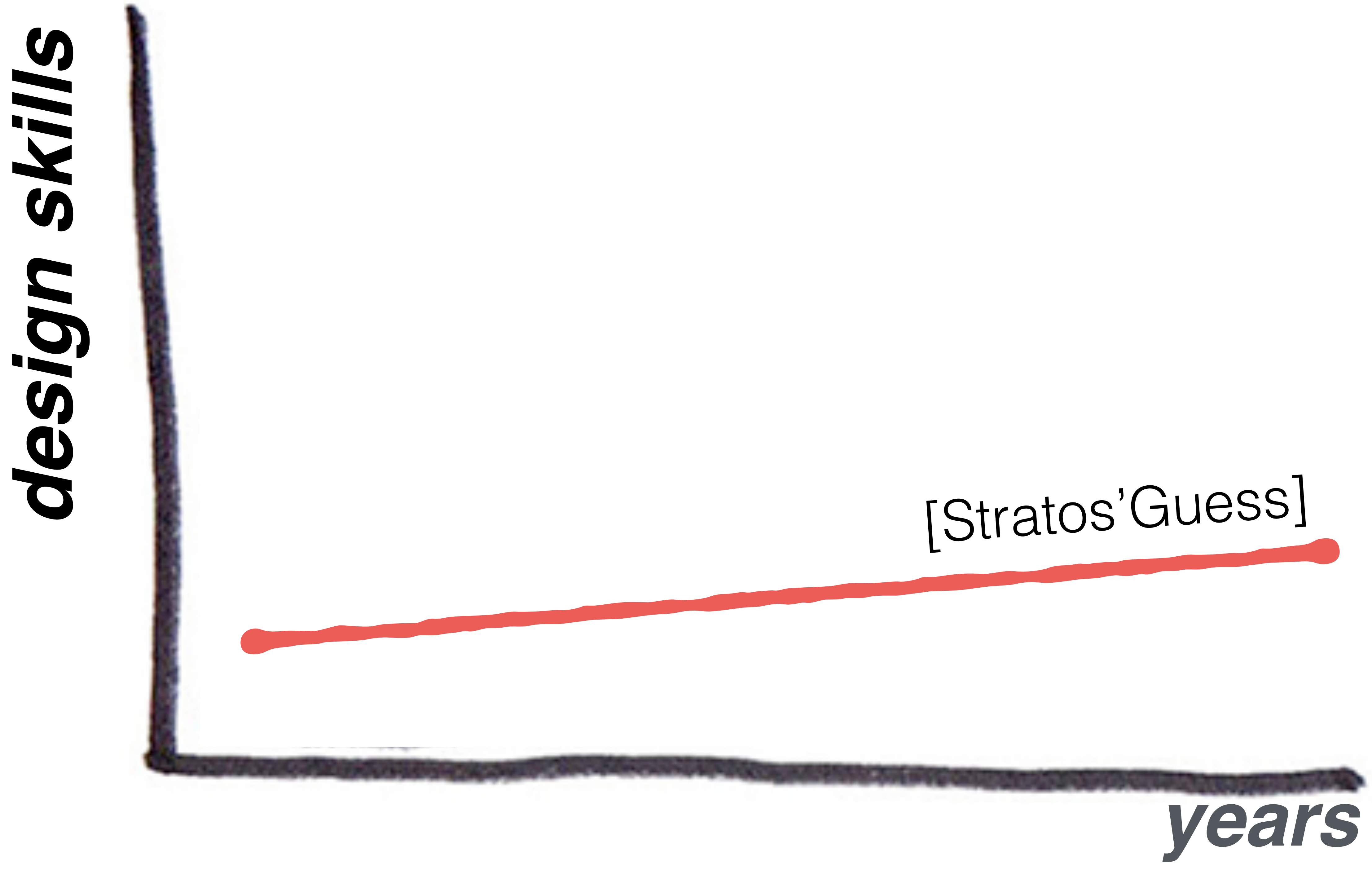
design is an art

sign:
Re
mpossible
phd
phd
is an art

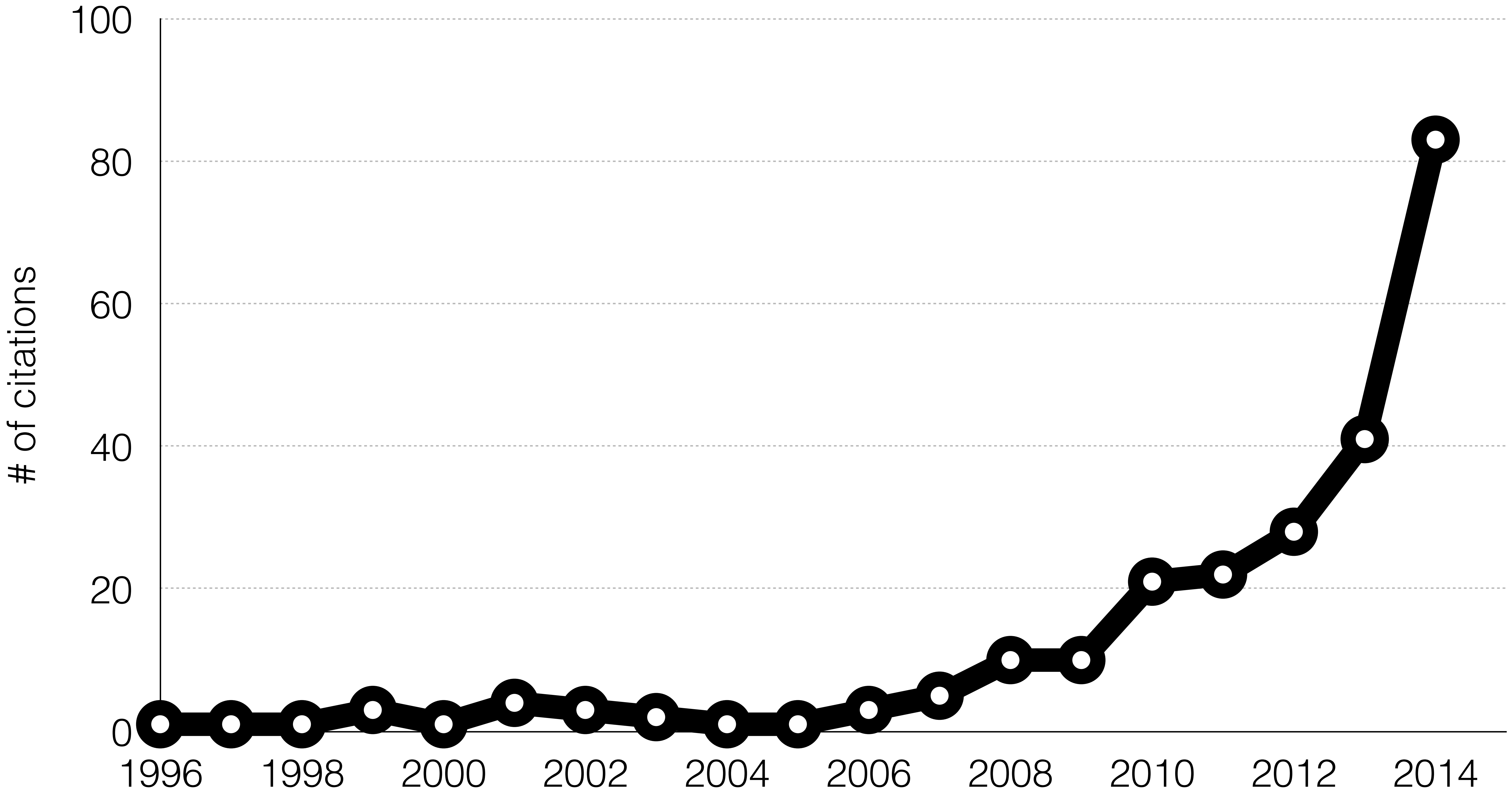

data
hardware
applications
years
phd
phd
is an art

1 design/research skills do not scale





2 no one knows everything out there

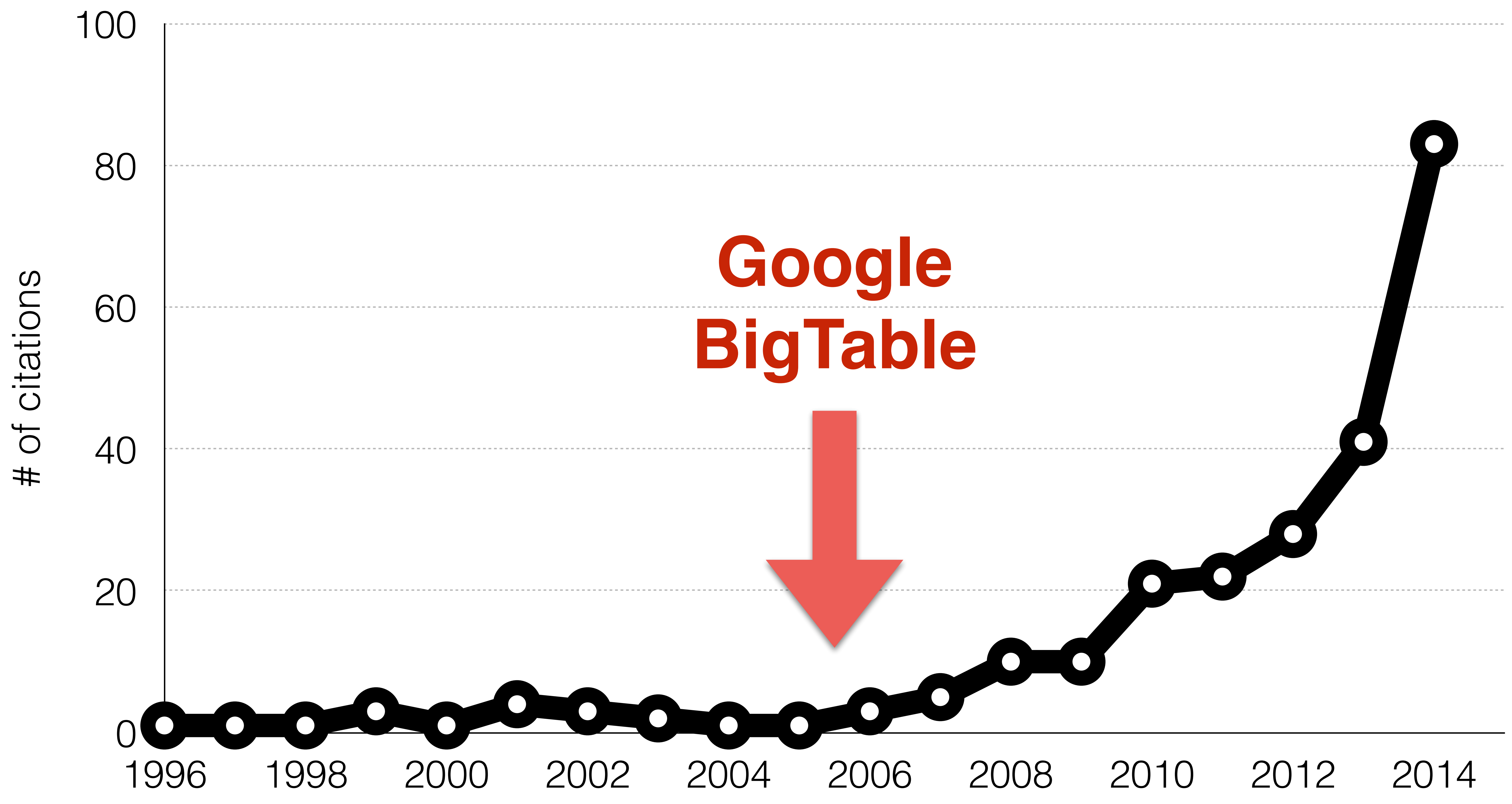


NoSQL storage

P. O'Neil, E. Cheng, D. Gawlick, E, O'Neil
The log-structured merge-tree (LSM-tree)
Acta Informatica 33 (4): 351–385, 1996

DASlab
@ Harvard SEAS

2 no one knows everything out there

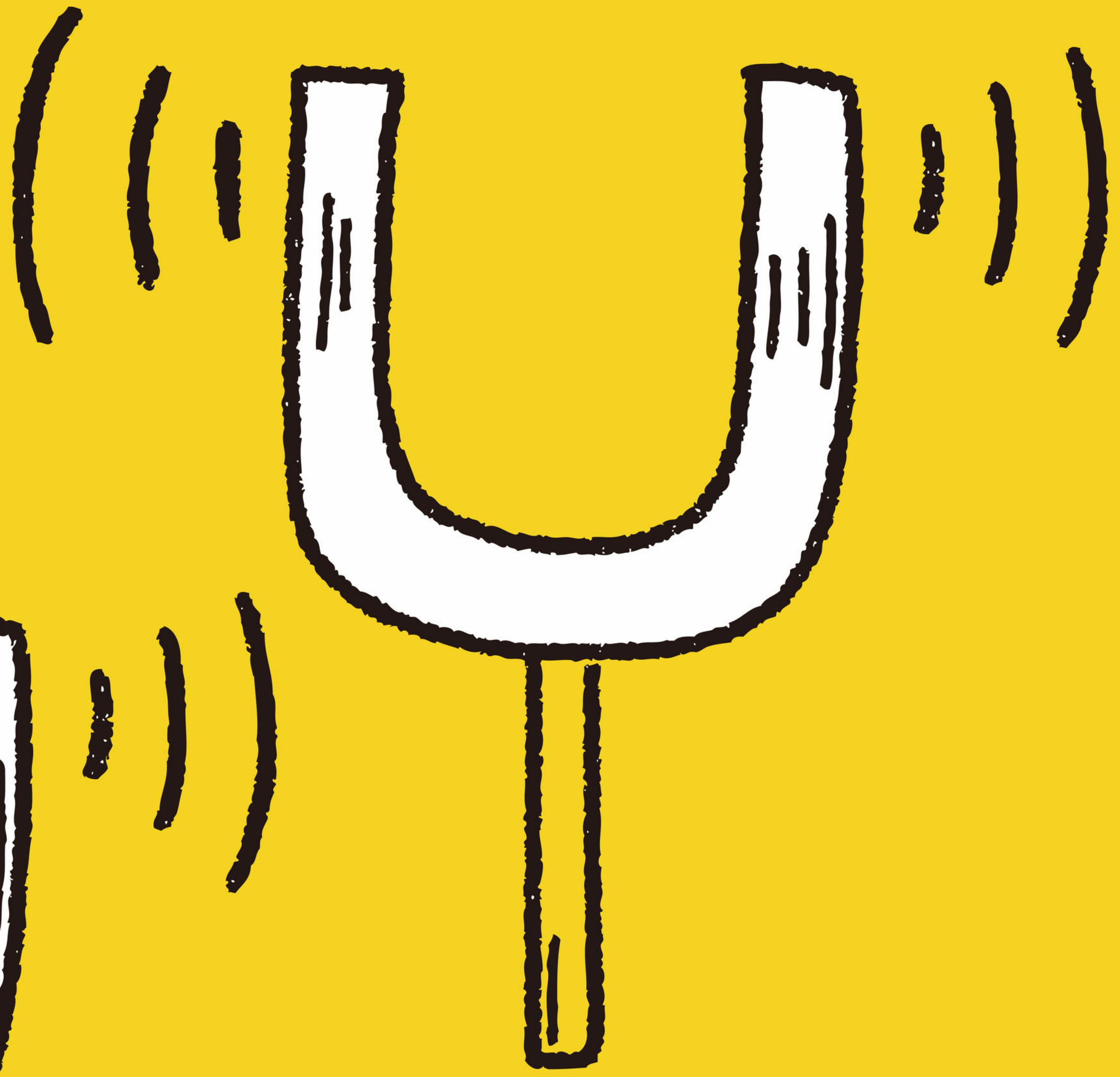


NoSQL storage

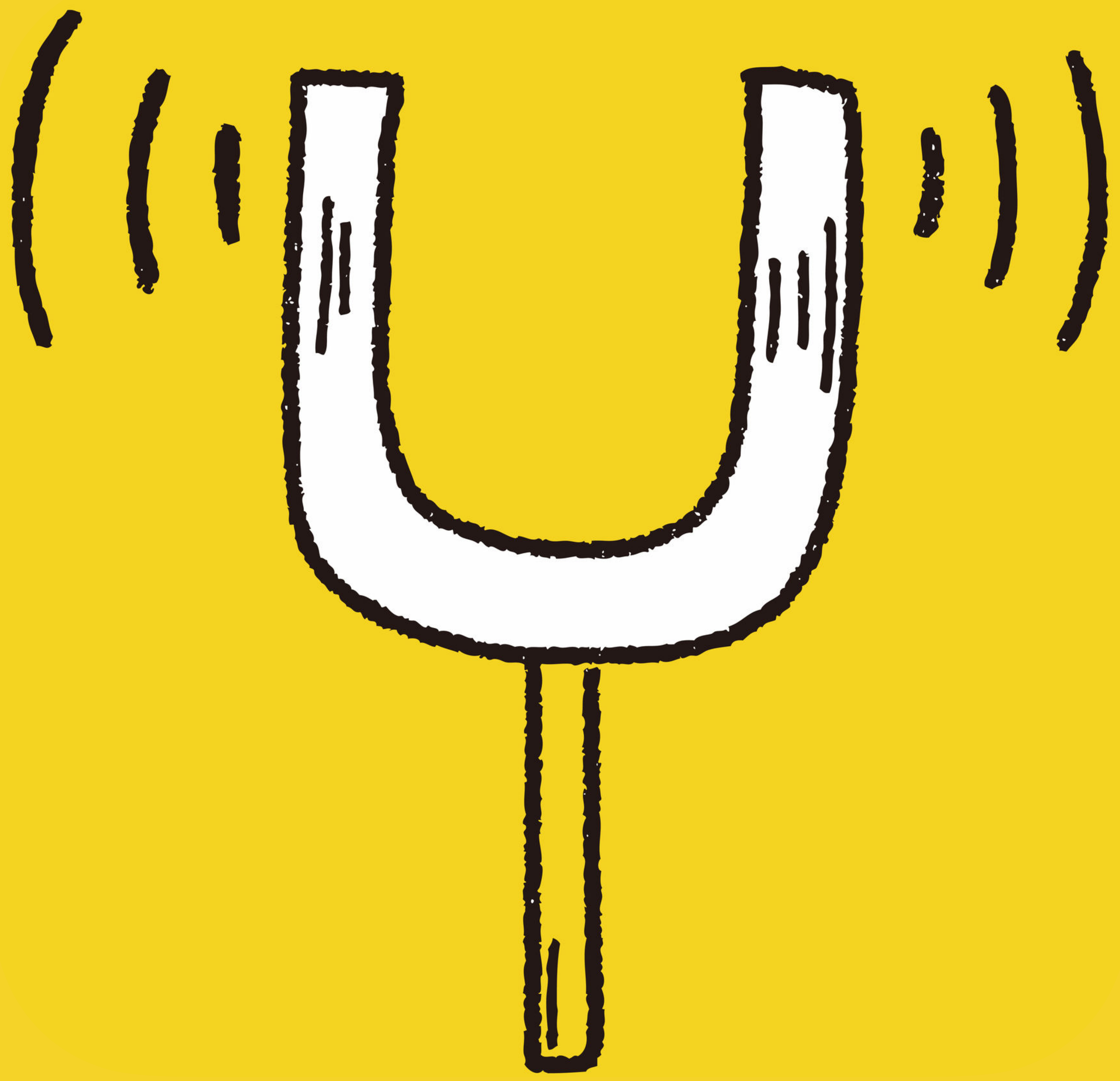
P. O'Neil, E. Cheng, D. Gawlick, E, O'Neil
The log-structured merge-tree (LSM-tree)
Acta Informatica 33 (4): 351–385, 1996

DASlab @ Harvard SEAS

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?**

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.

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.

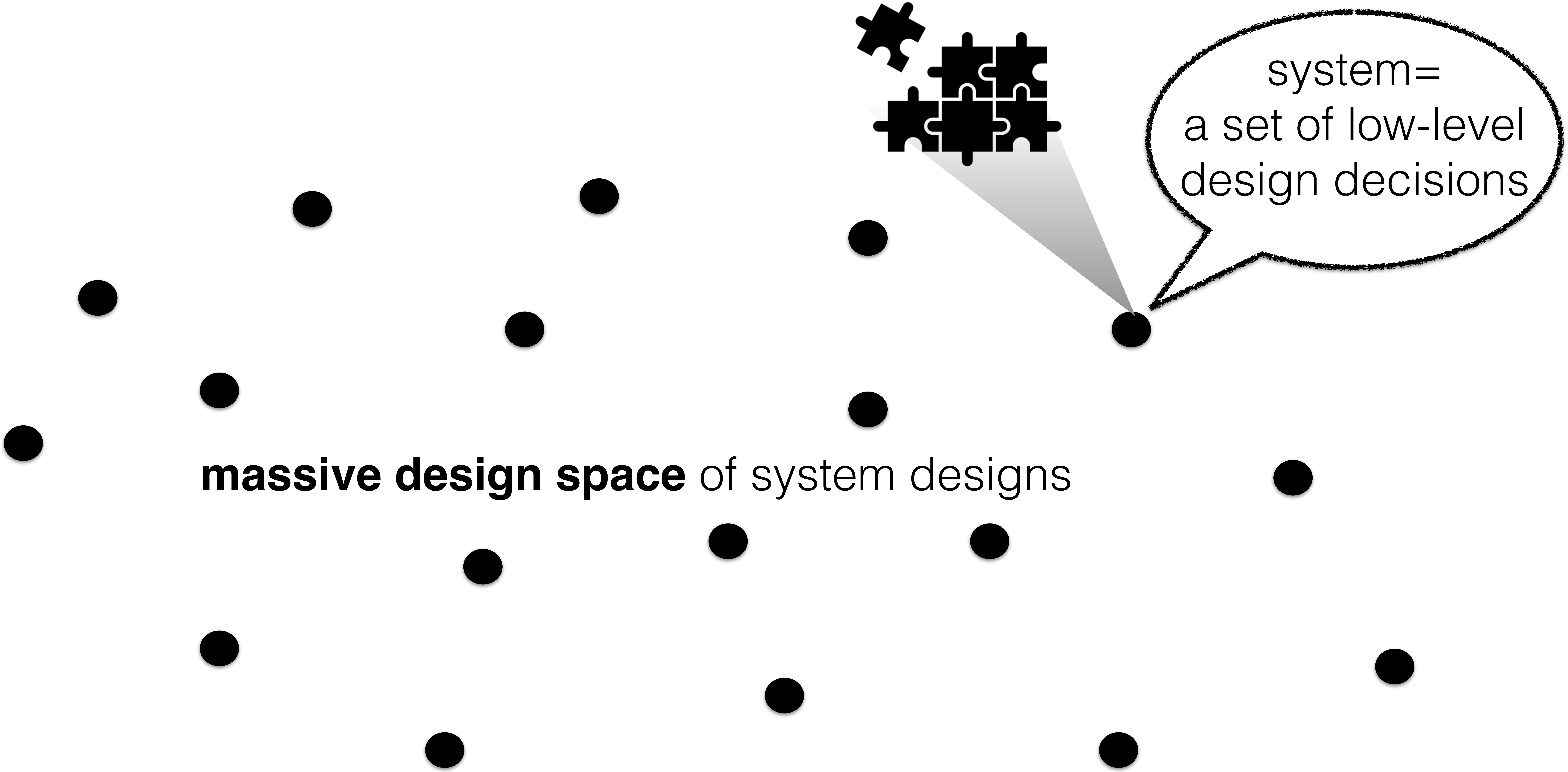
These ideas can lead to better systems but we need something more to

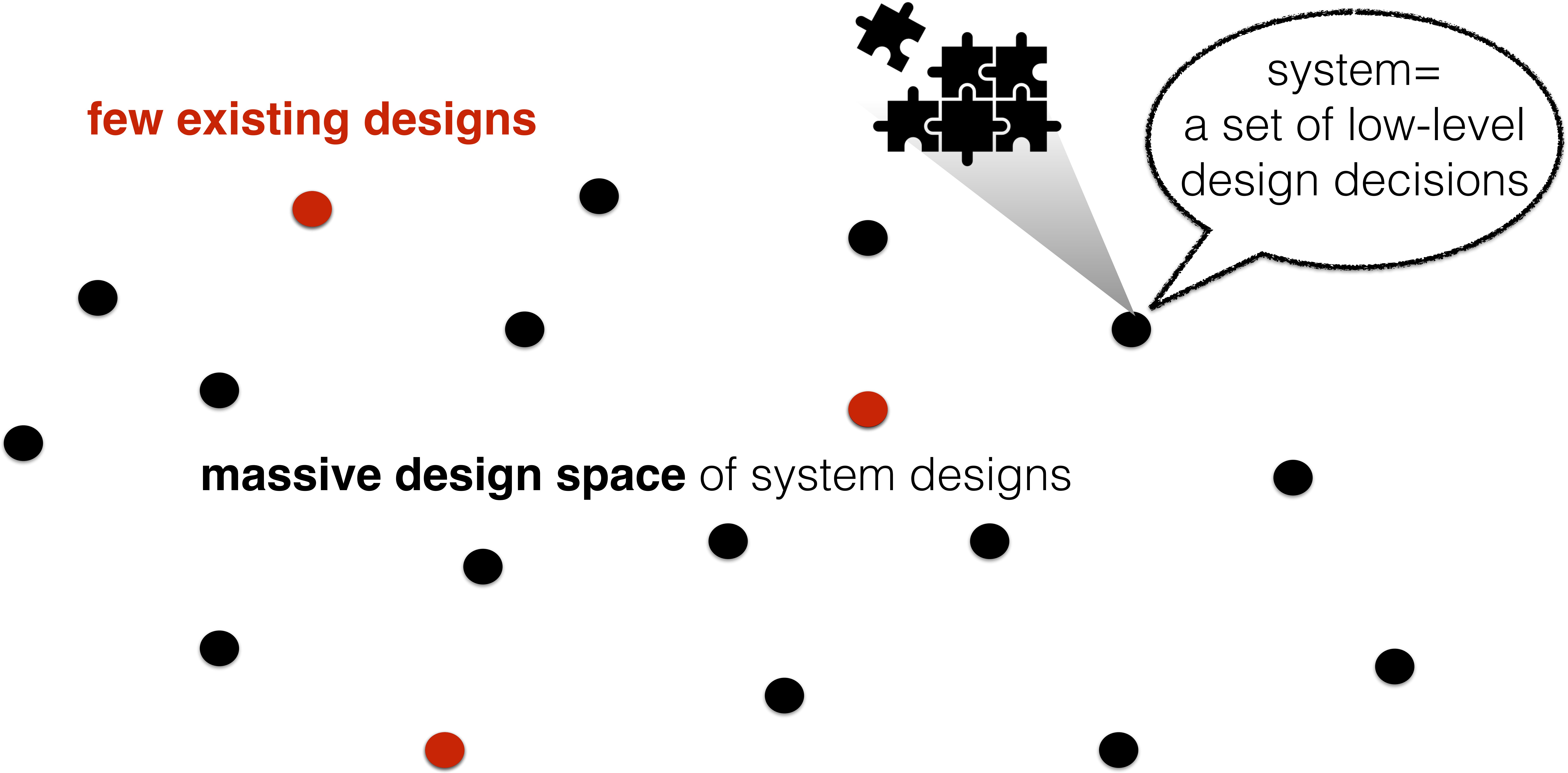
FIND FAST THE BEST POSSIBLE DESIGN

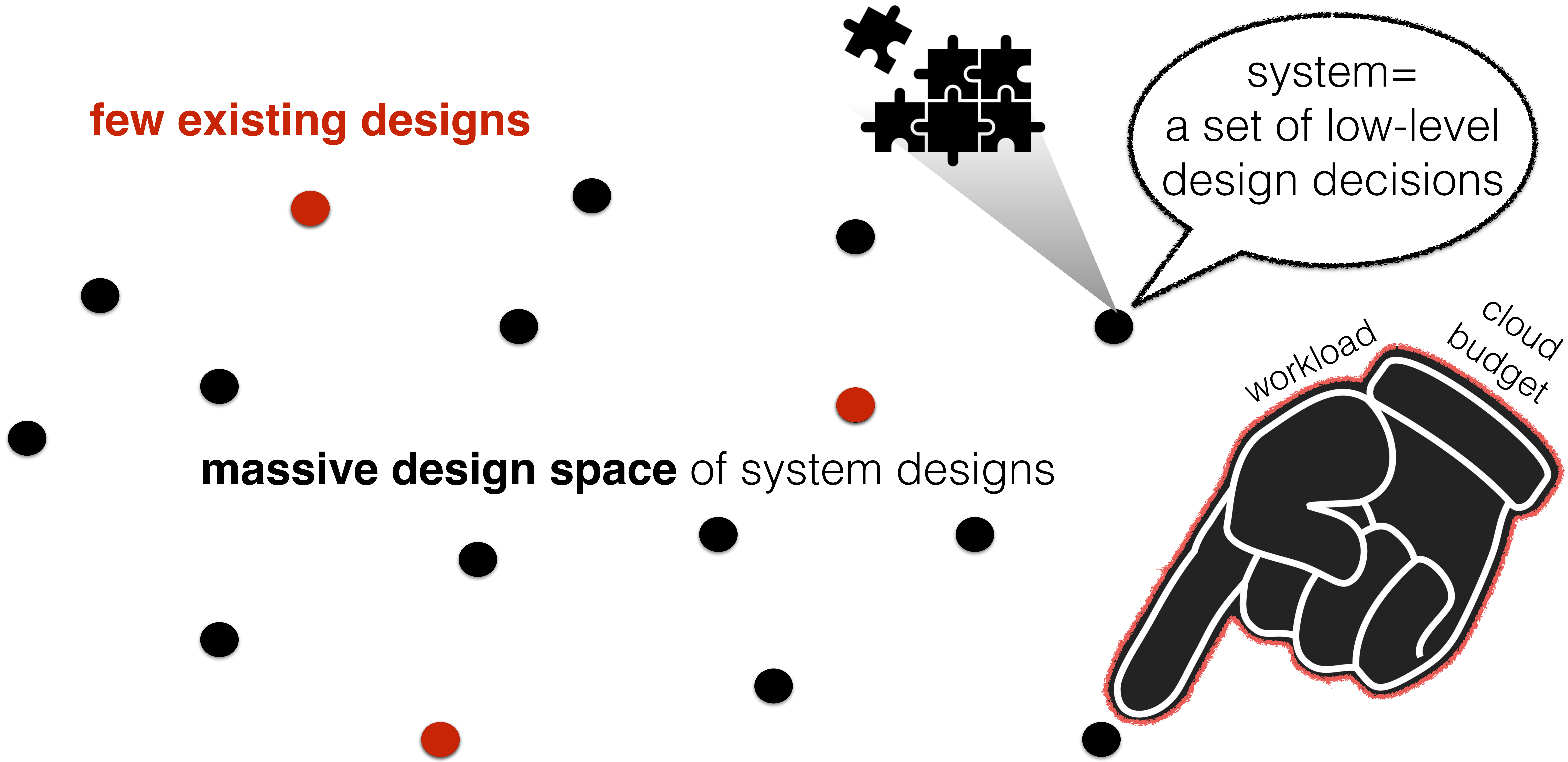
SELF-DESIGNING SYSTEMS

Automatically invent & build the perfect system for any new application

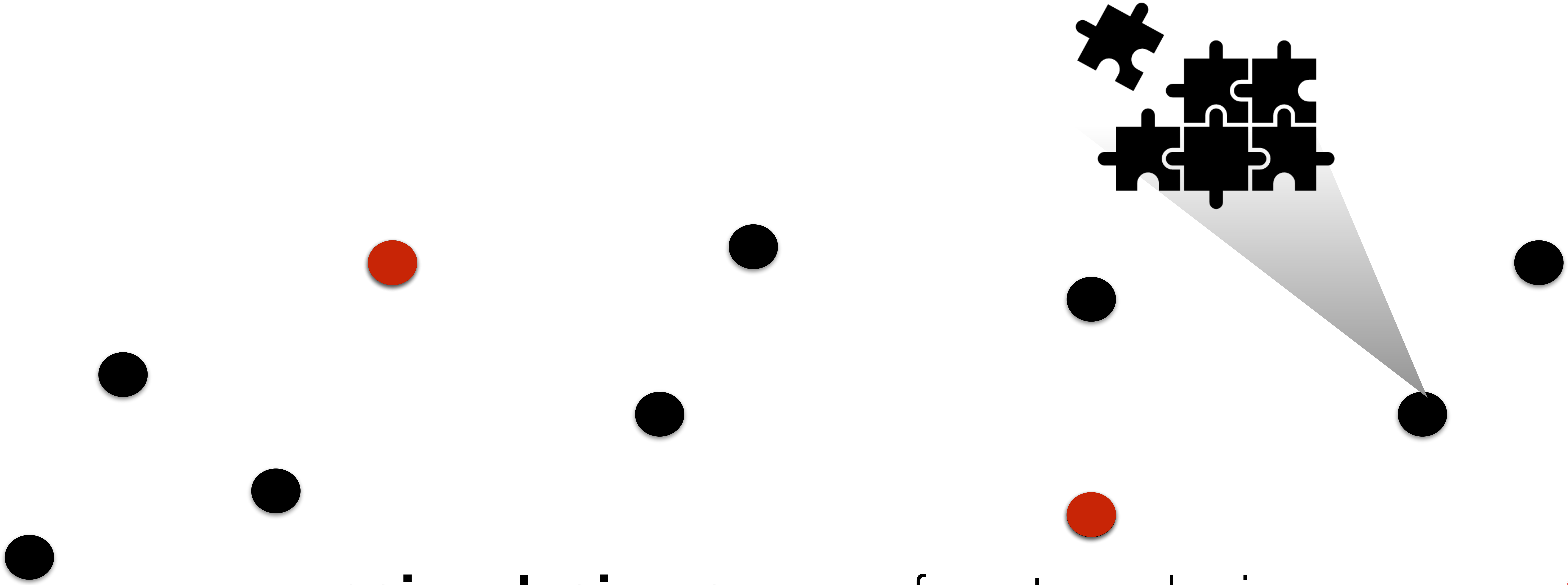
massive design space of system designs


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


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



DASlab
@ Harvard SEAS



massive design space of system designs

reasoning: understand all the design decisions & their impact

DASlab @ Harvard SEAS

HOW
DO WE
START

SLAs

robustness

concurrency

multi-tenancy

data types

HOW DO WE START

indexing

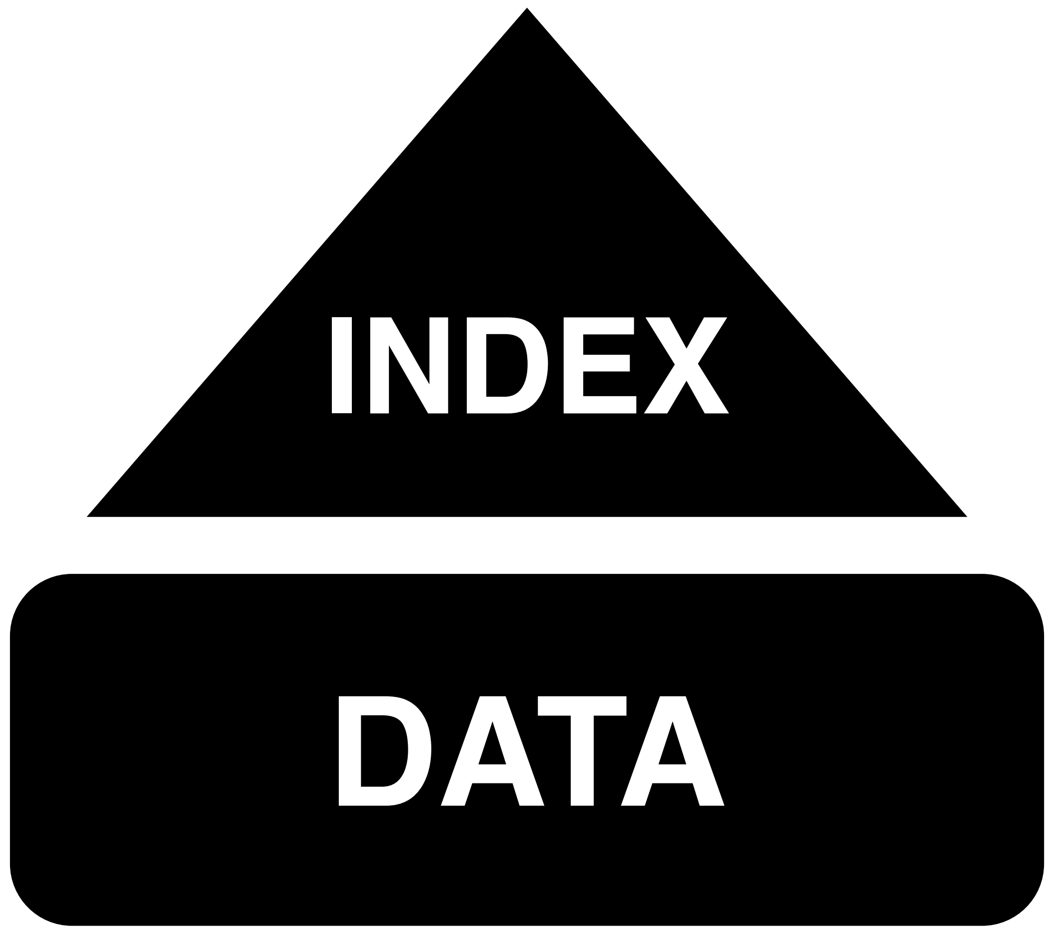
complex operations

optimizer

cloud

hardware

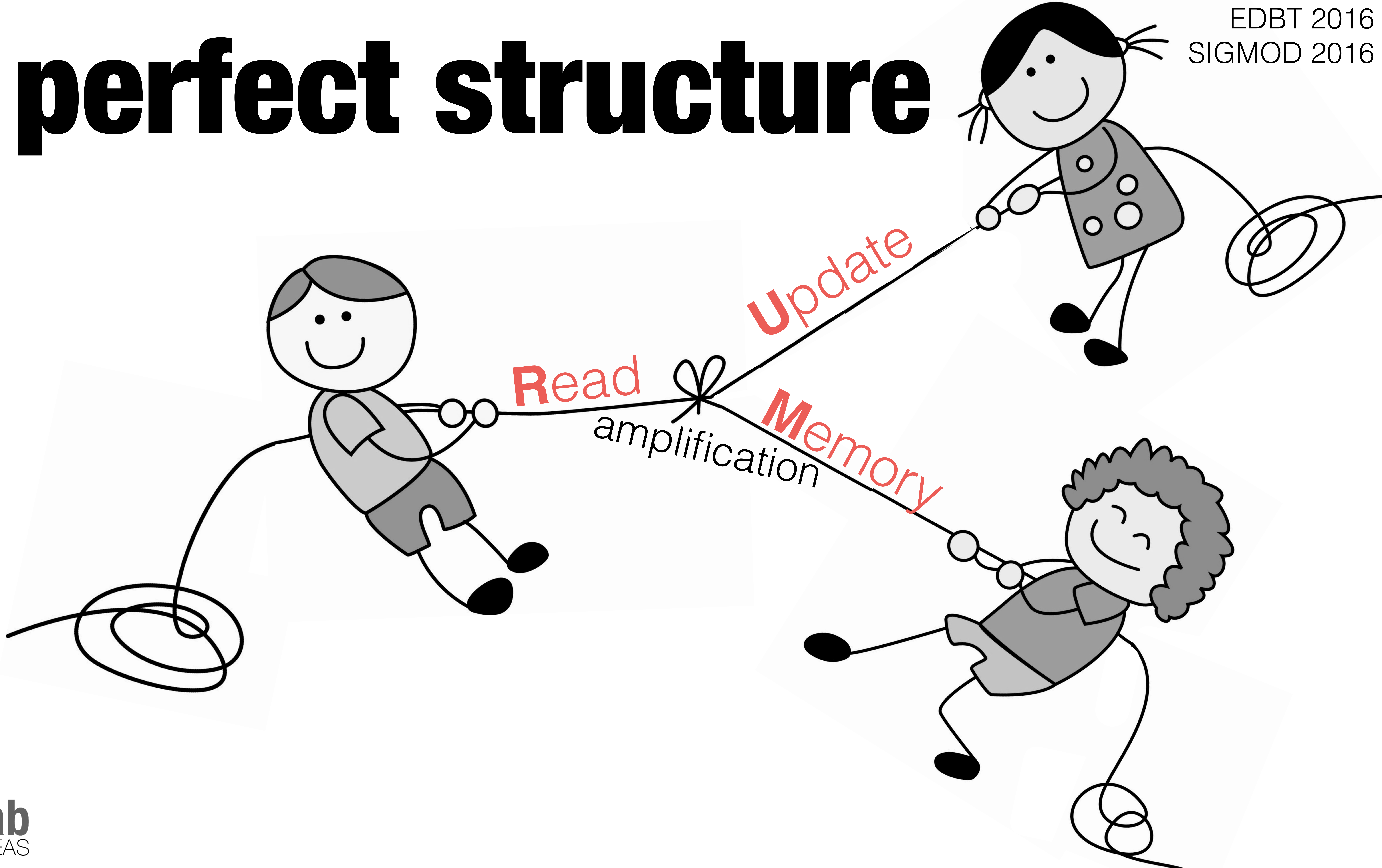
DASlab @ Harvard SEAS



HOW
DO WE
START

no perfect structure

EDBT 2016
SIGMOD 2016



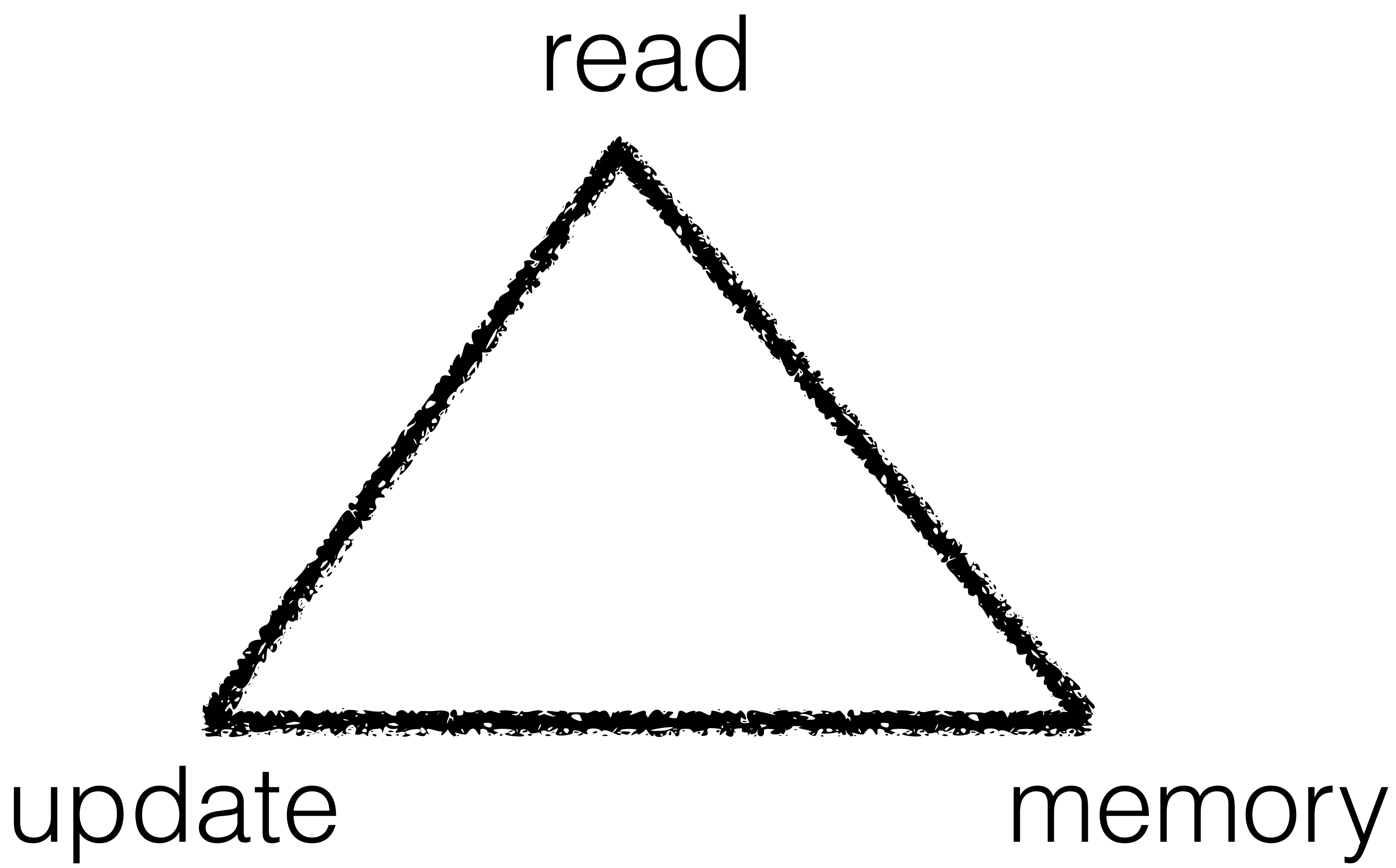
no perfect structure

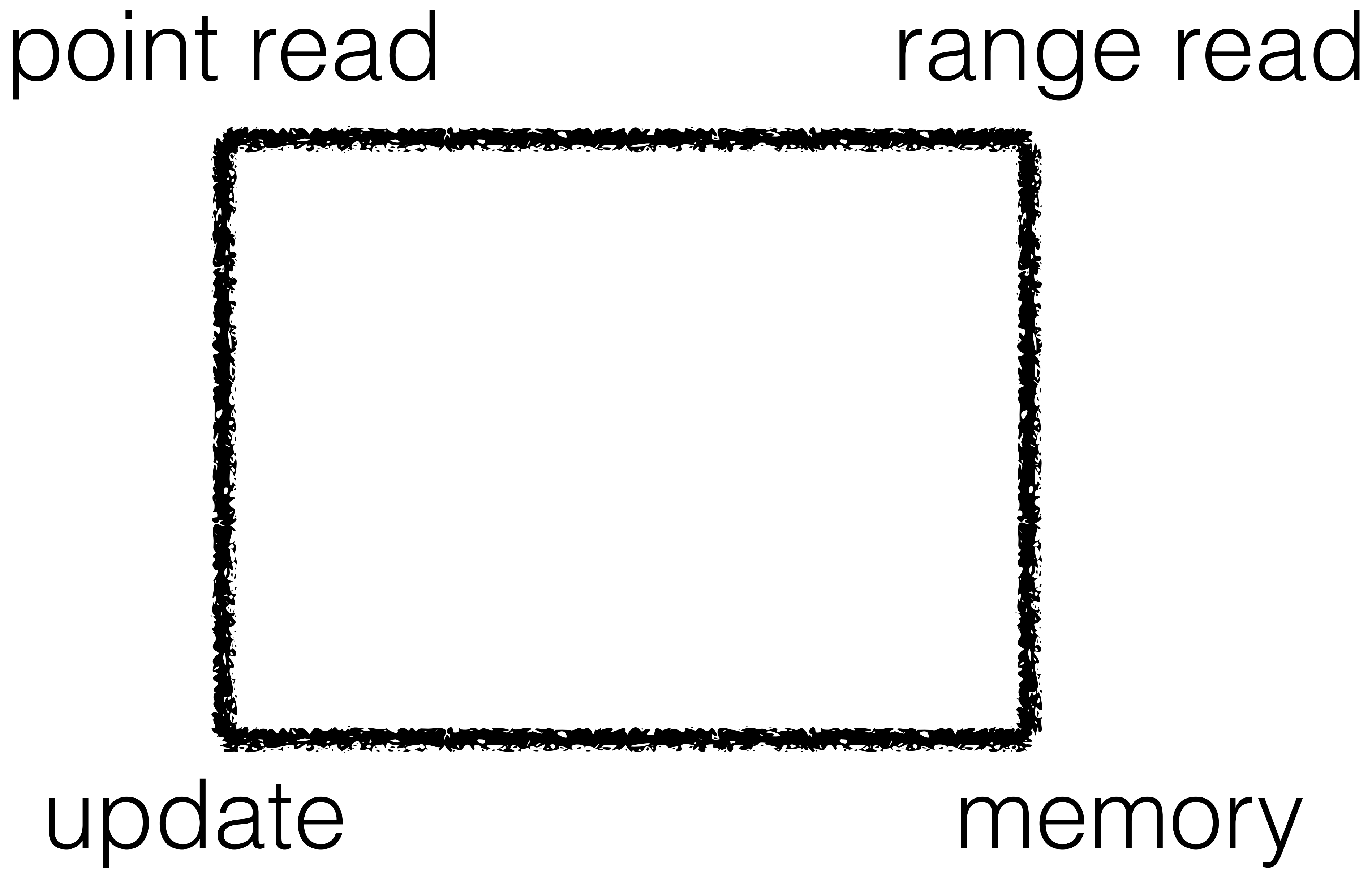
EDBT 2016
SIGMOD 2016

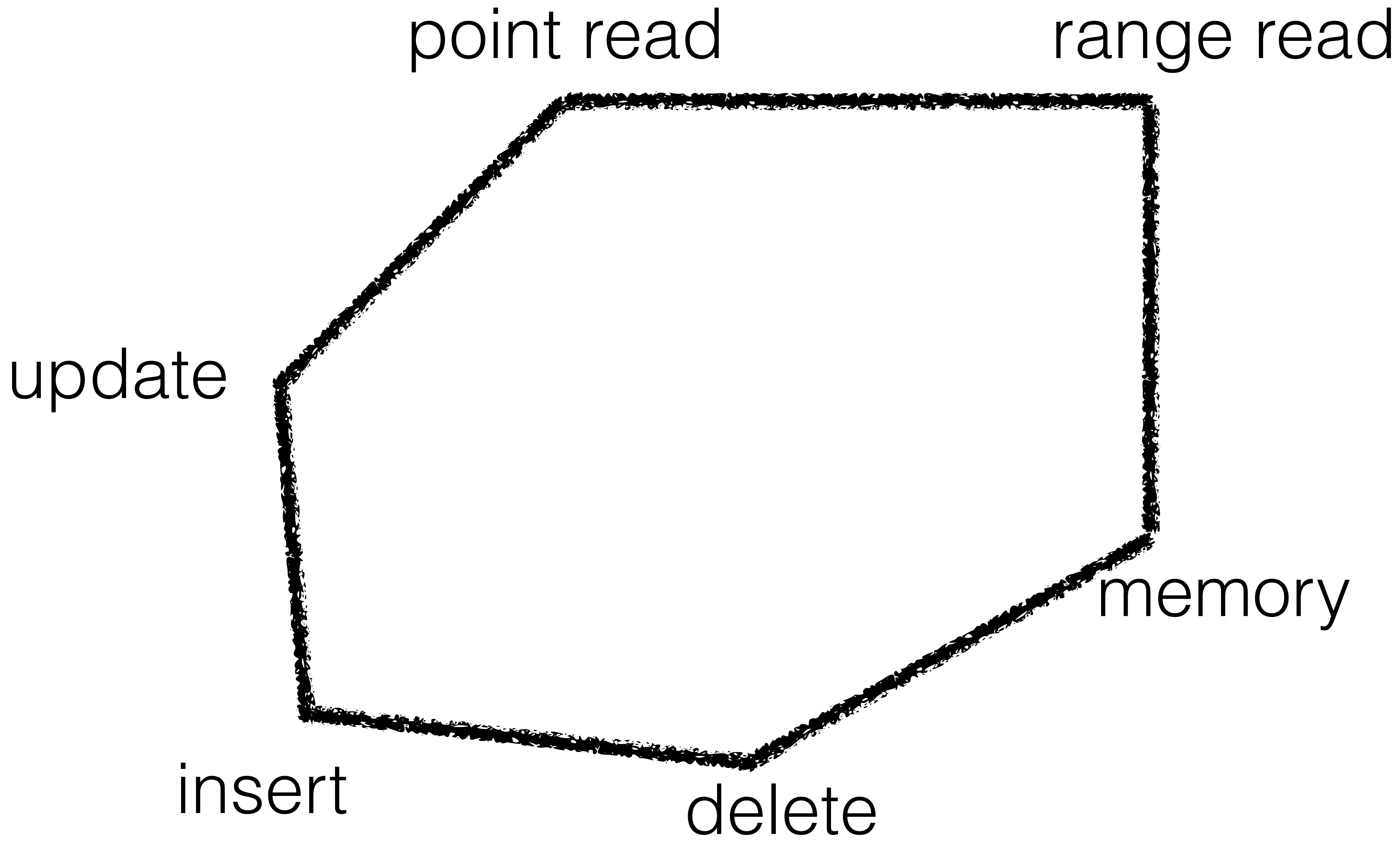
DASlab
@ Harvard SEAS

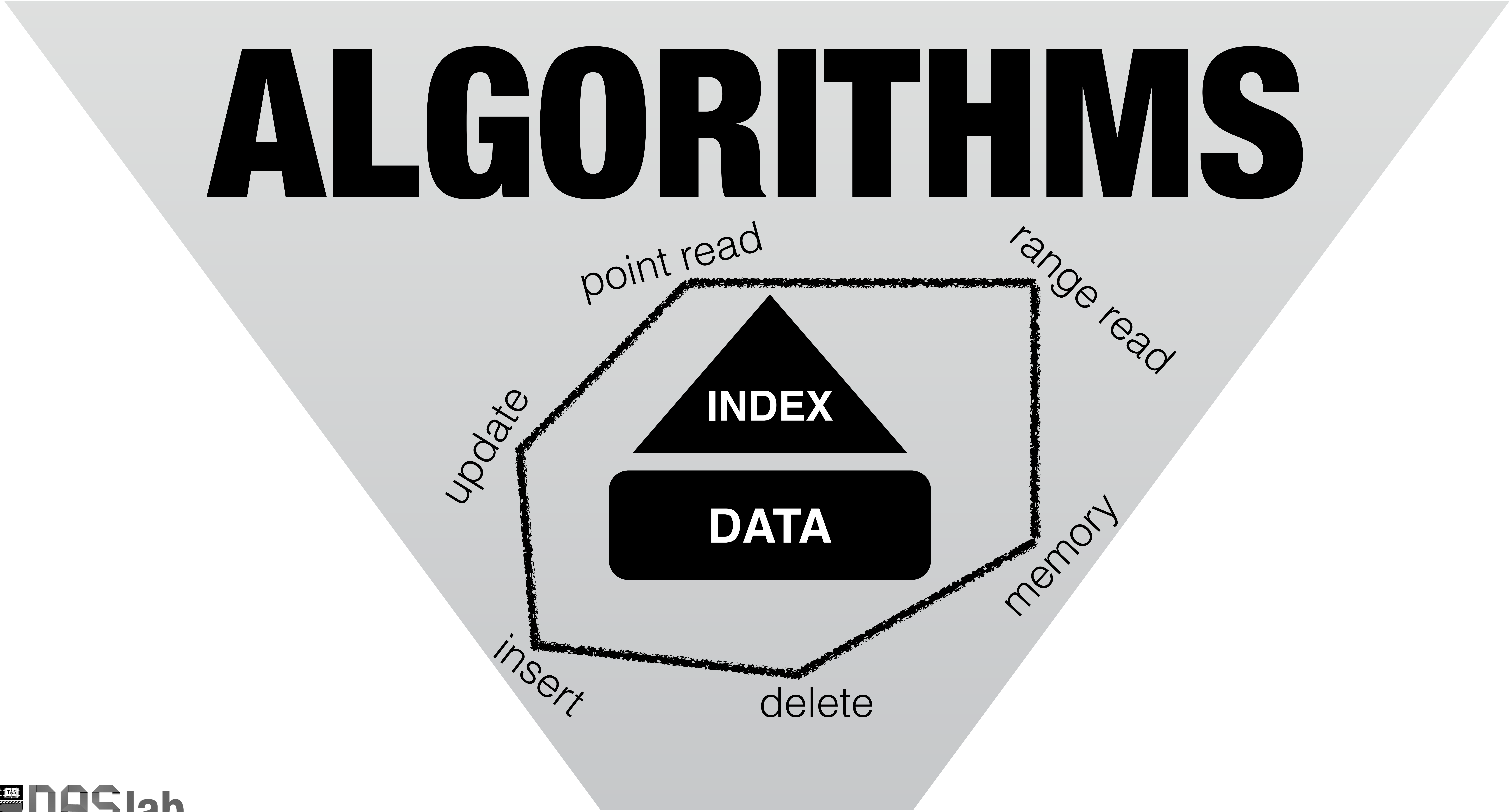
no perfect structure

EDBT 2016
SIGMOD 2016

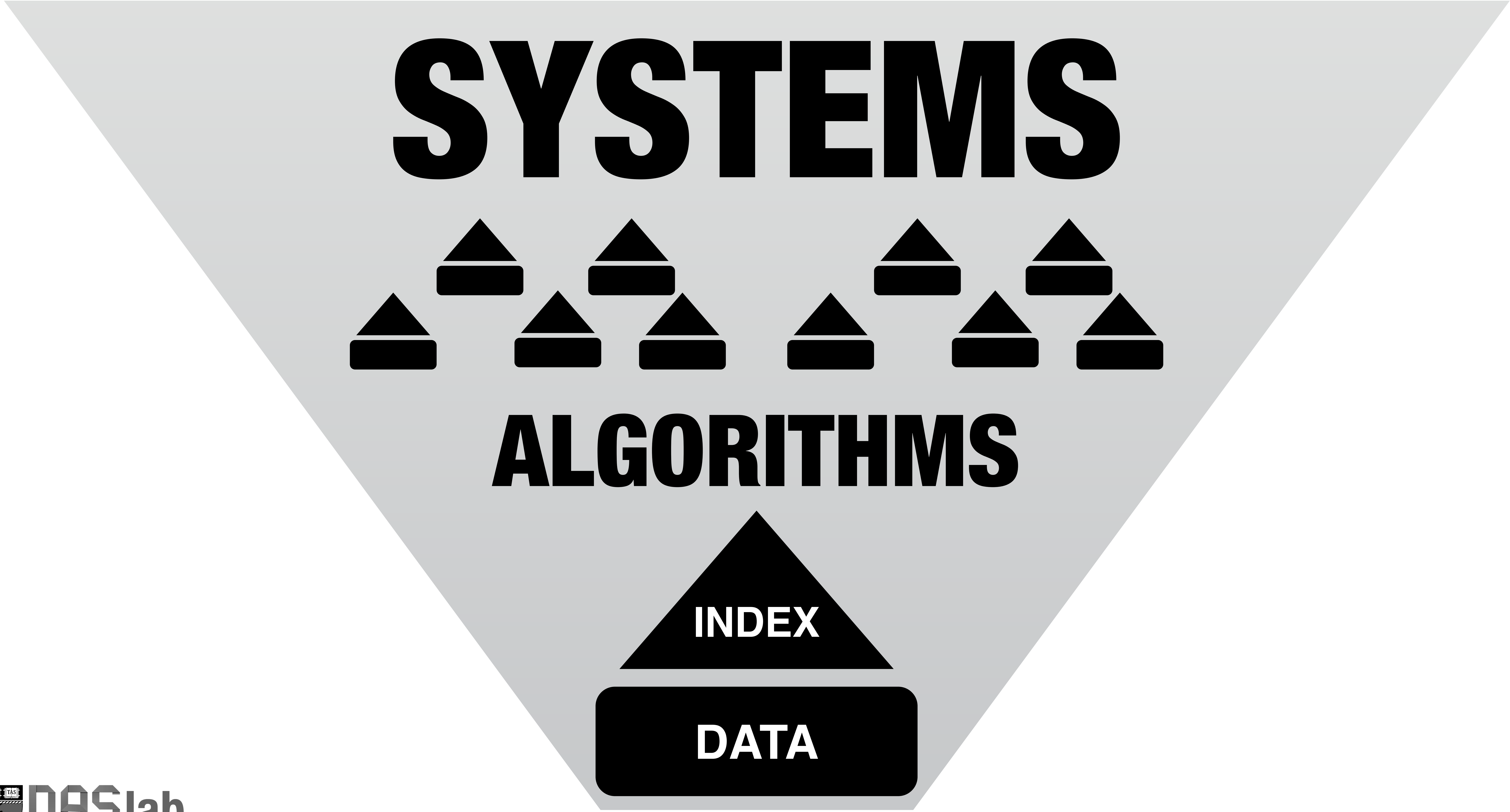

read
update
memory


point read
range read
update
memory


point read
range read
update
memory
insert
delete



DASlab
@ Harvard SEAS



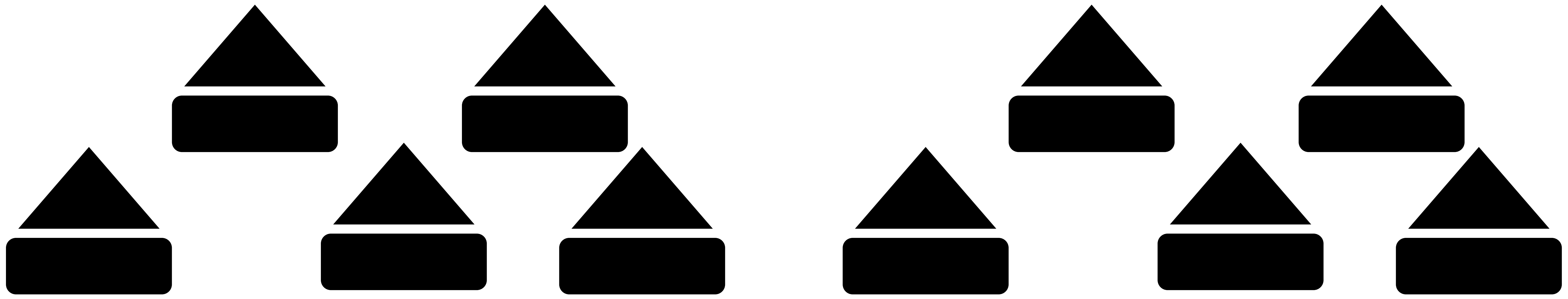
DASlab
@ Harvard SEAS

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

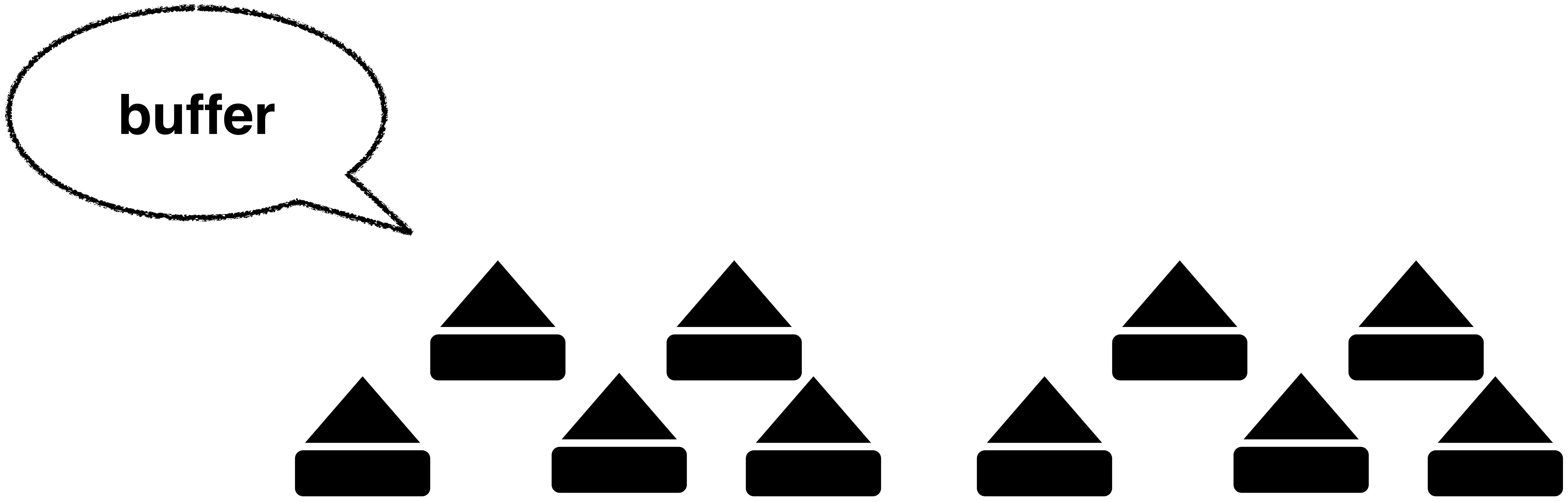


NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE

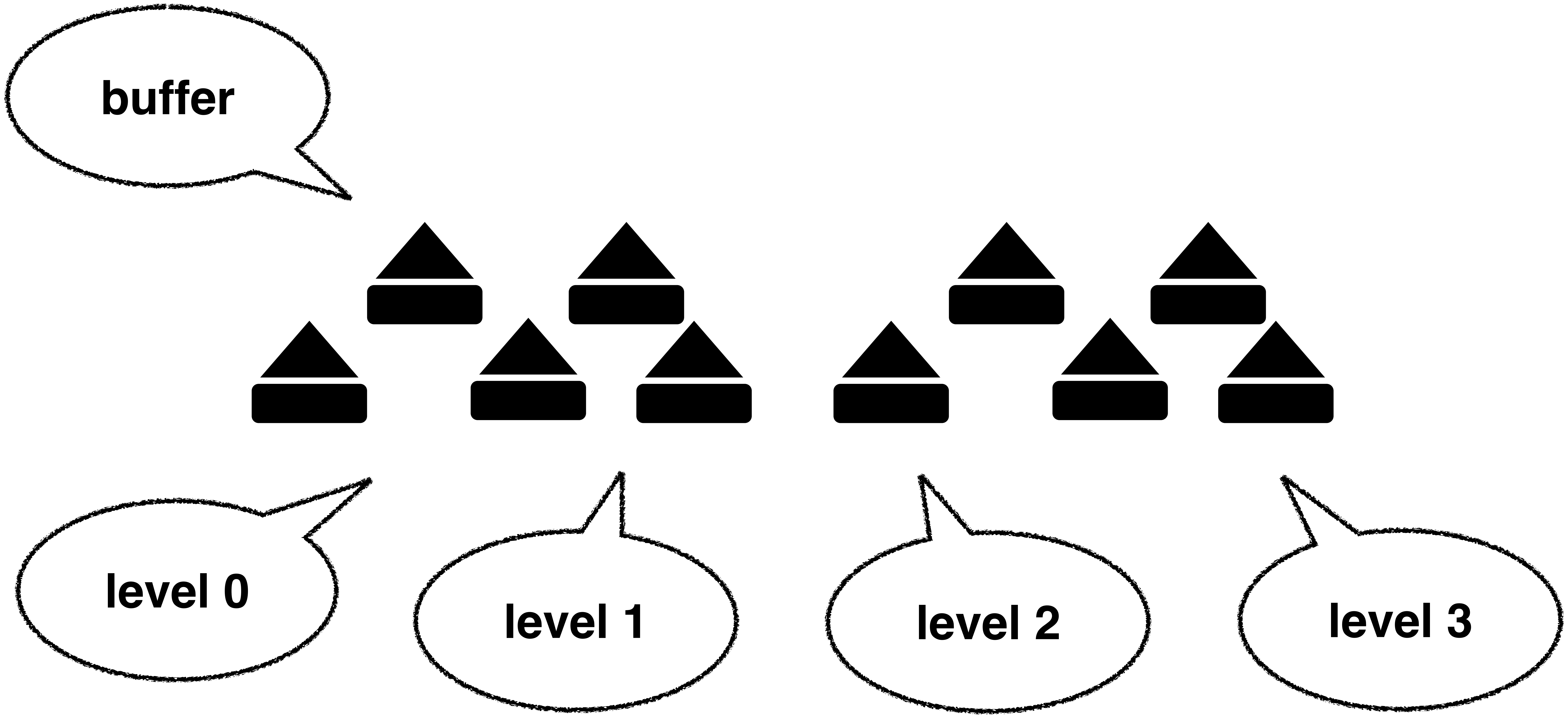


NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE

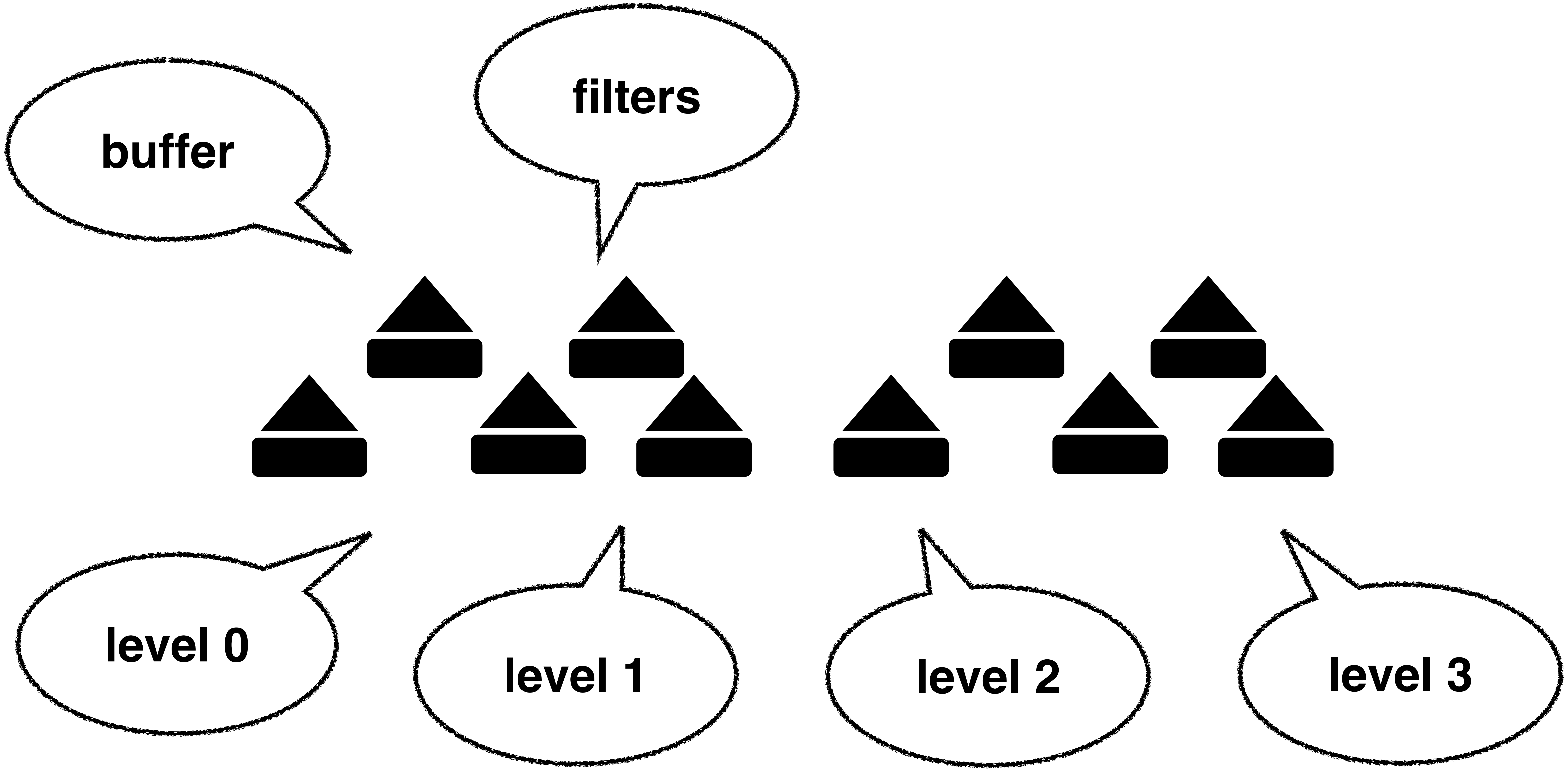


NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE

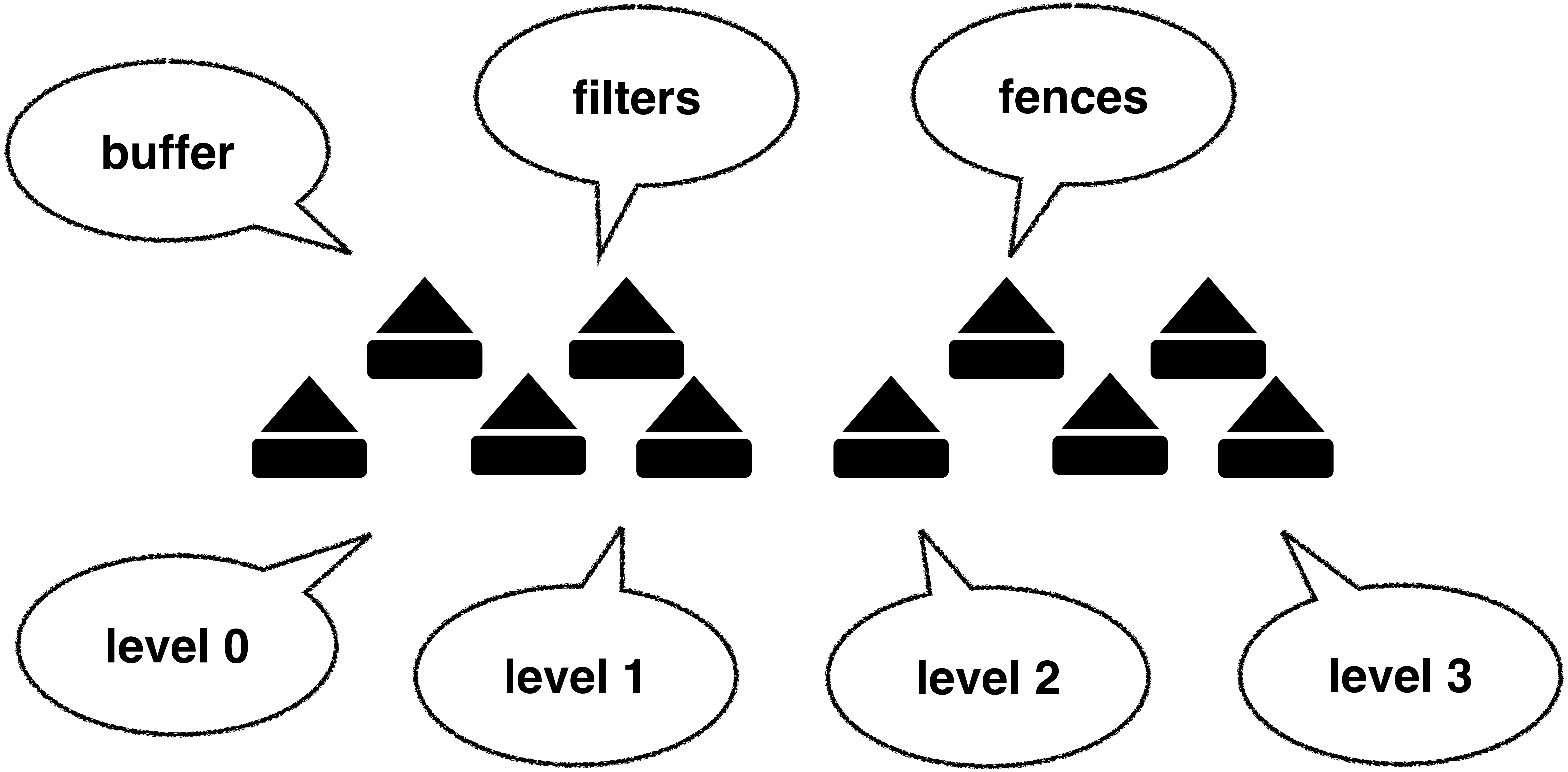


NoSQL systems are the backbone of the BigData and AI era

LSM-tree
KV-stores

FACEBOOK, AMAZON, GOOGLE, TWITTER, LINKEDIN

MACHINE LEARNING, SQL, CRYPTO, SCIENCE



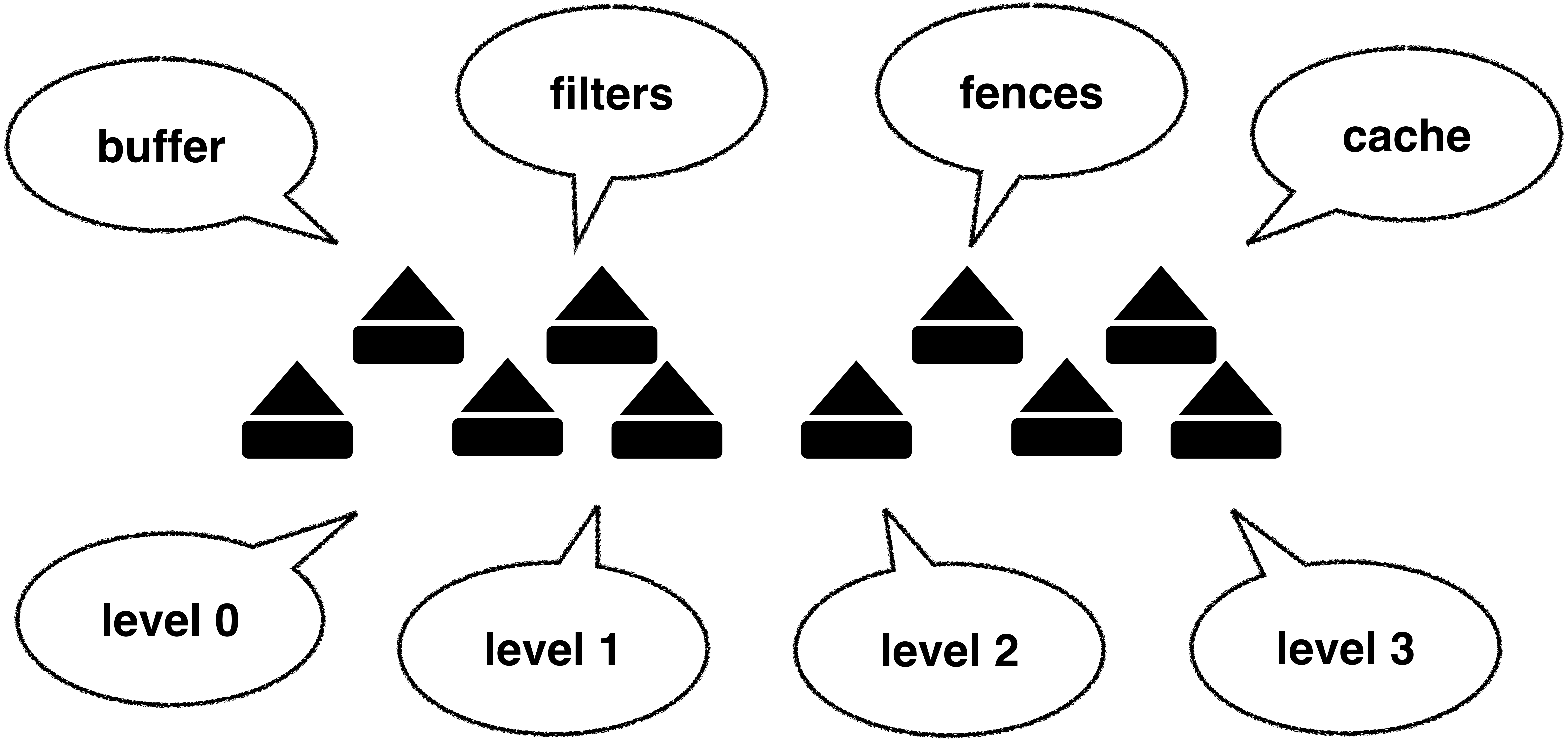
DASlab @ Harvard SEAS

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



DASlab @ Harvard SEAS

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

@ Harvard SEAS

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

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

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

DASlab @ Harvard SEAS

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
filters
fences
cache

diverse data structures	interactions	hardware	parallelism	robustness cloud cost SLAs

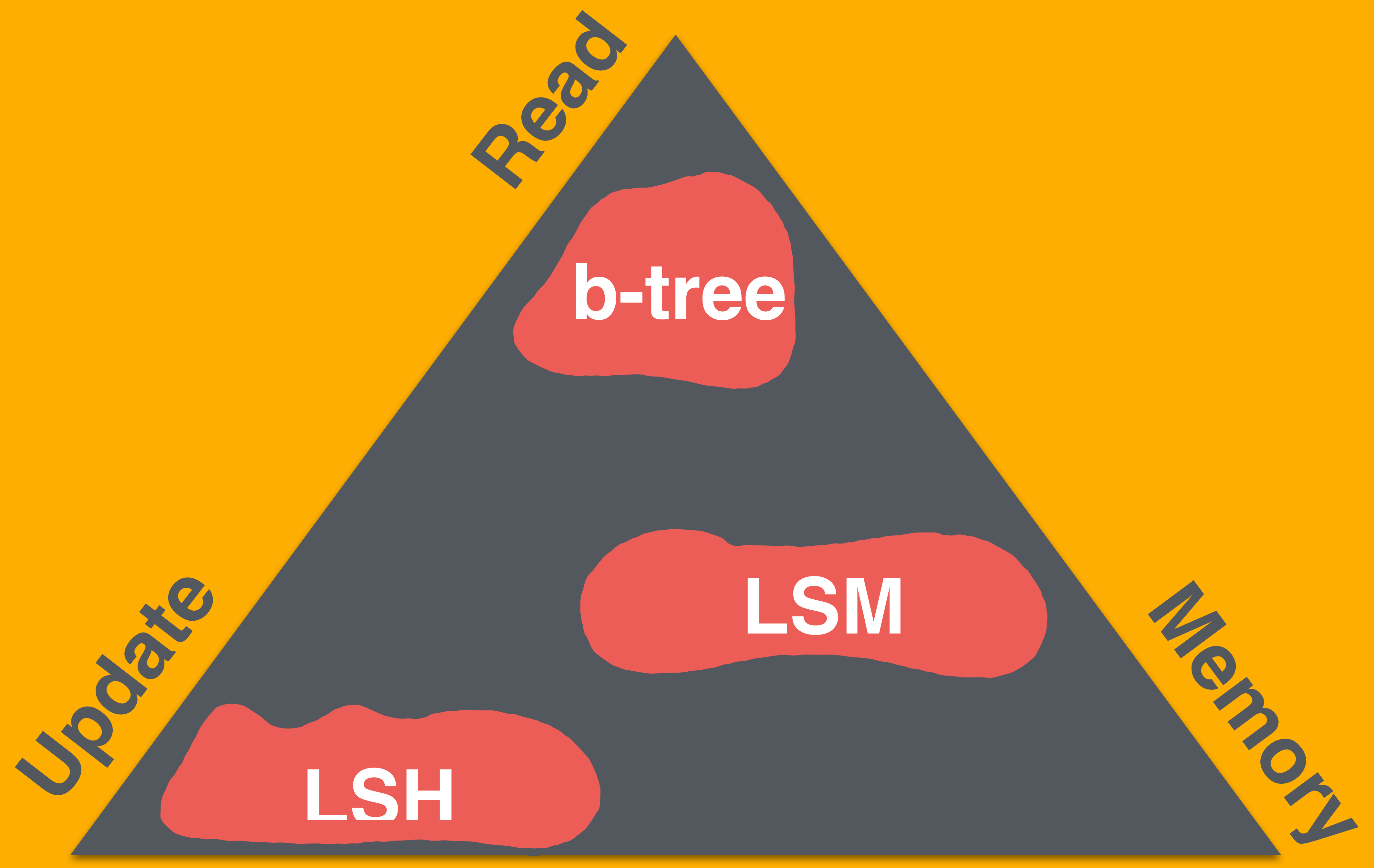
There exist numerous variations of NoSQL KV-stores

LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft)

diverse data structures	interactions	hardware	parallelism	robustness cloud cost SLAs

There exist numerous variations of NoSQL KV-stores

LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft)



diverse data structures

interactions

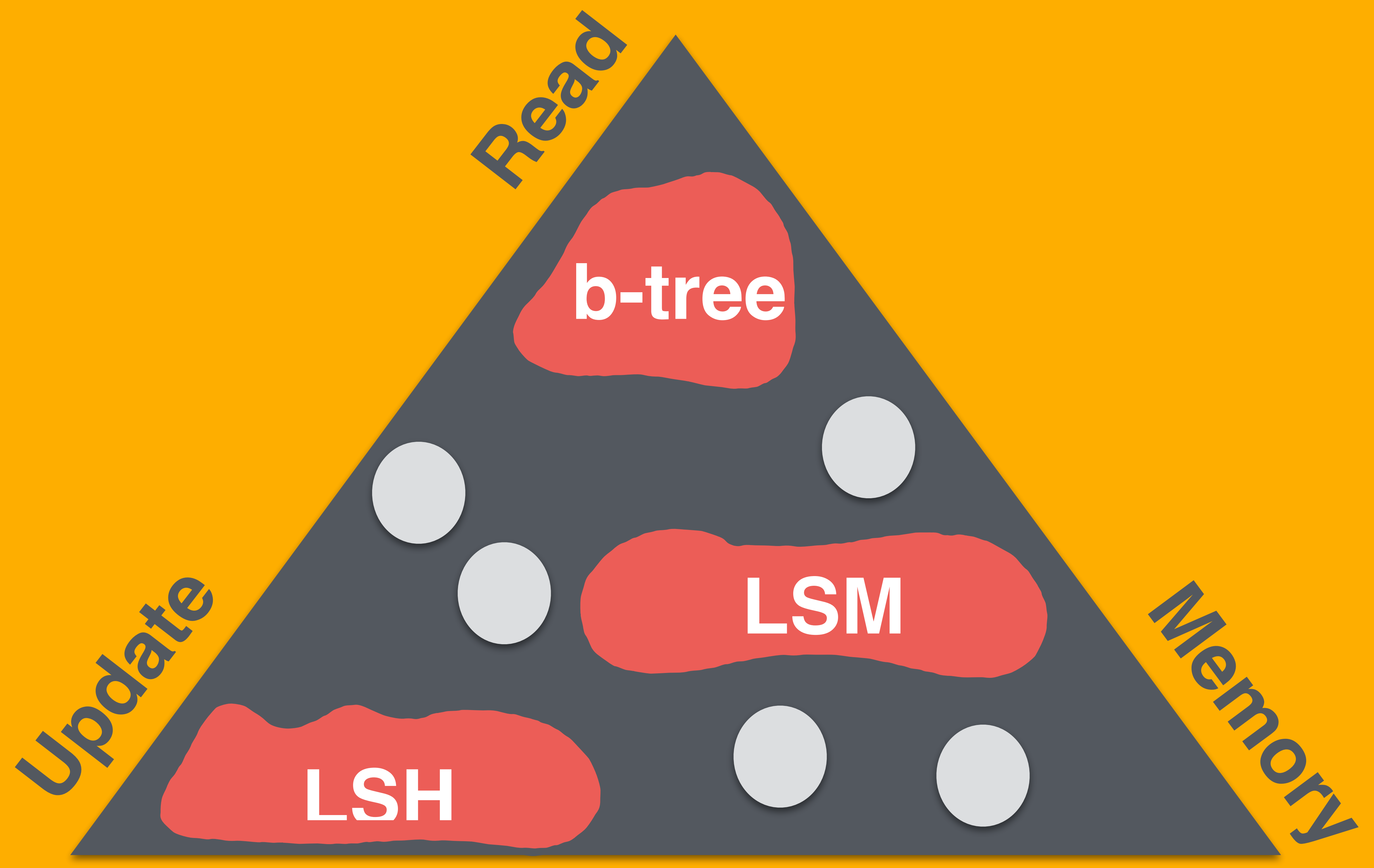
hardware

parallelism

robustness
cloud cost
SLAs

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

diverse data structures	interactions	hardware	parallelism	robustness cloud cost SLAs

There exist numerous variations of NoSQL KV-stores

LSM-tree variants, B-trees (MongoDB), Hash-index (Microsoft)

diverse
data structures

interactions

hardware

parallelism

robustness
cloud cost
SLAs

Requirements/Goals

diverse data structures

interactions

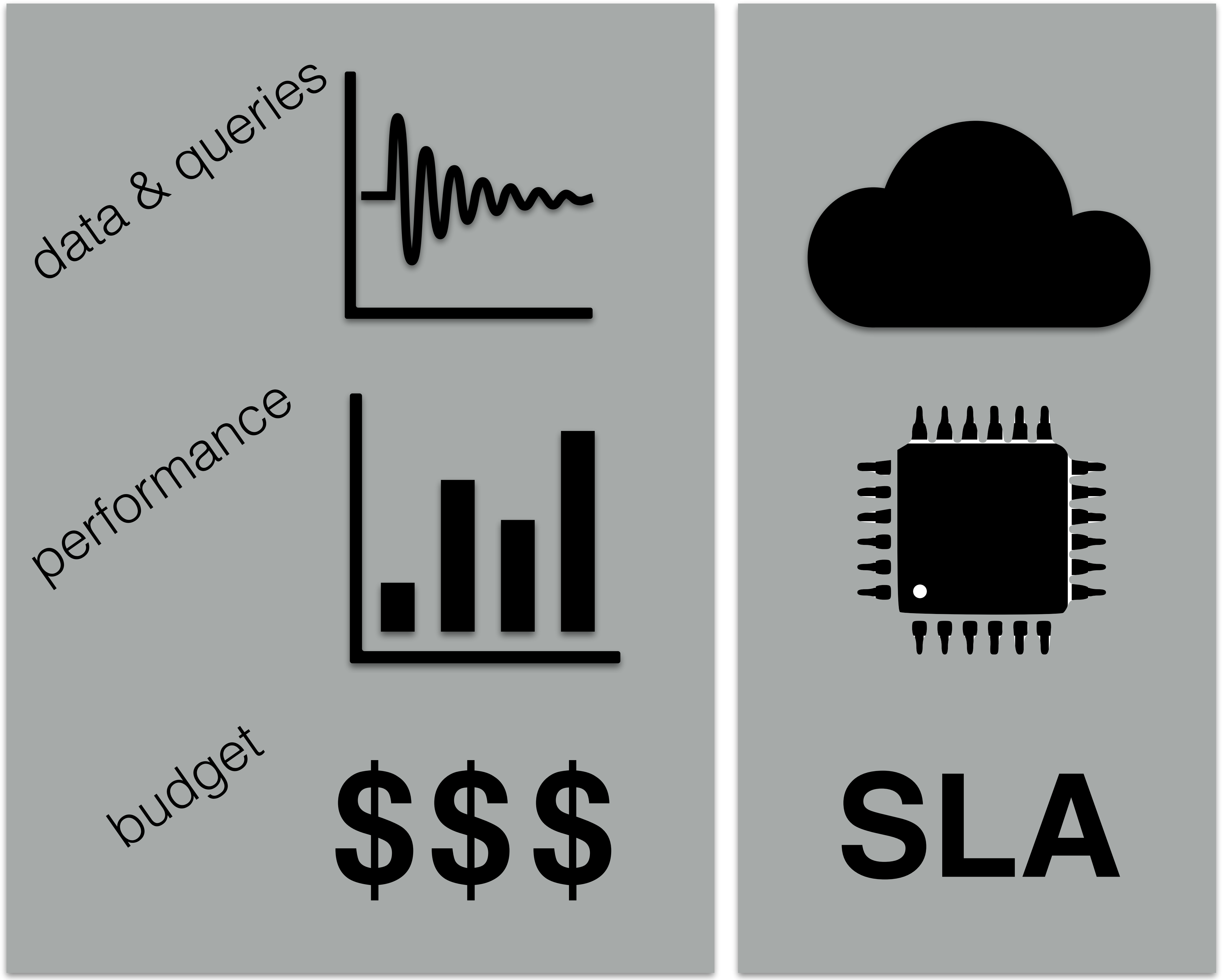
hardware

parallelism

robustness
cloud cost
SLAs

Requirements/Goals

Context



diverse data structures

interactions

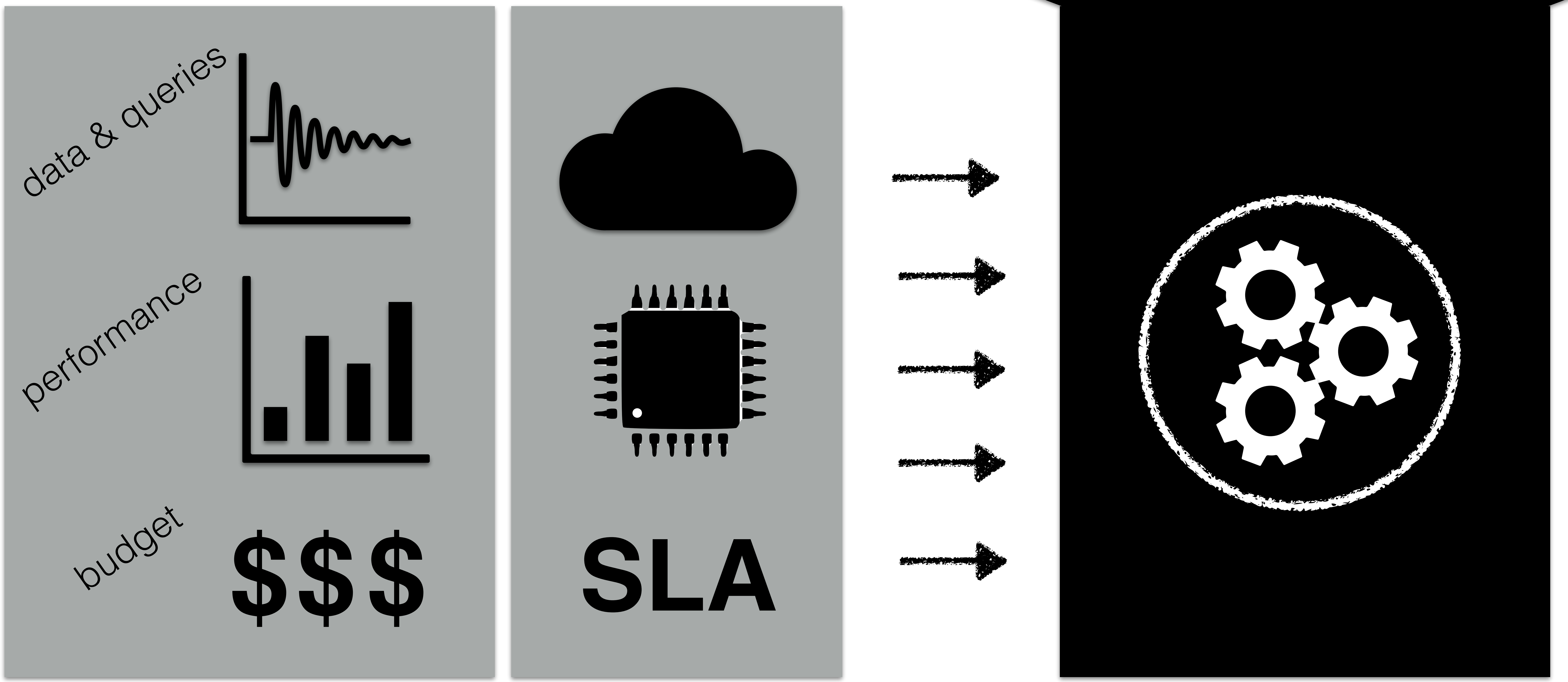
hardware

parallelism

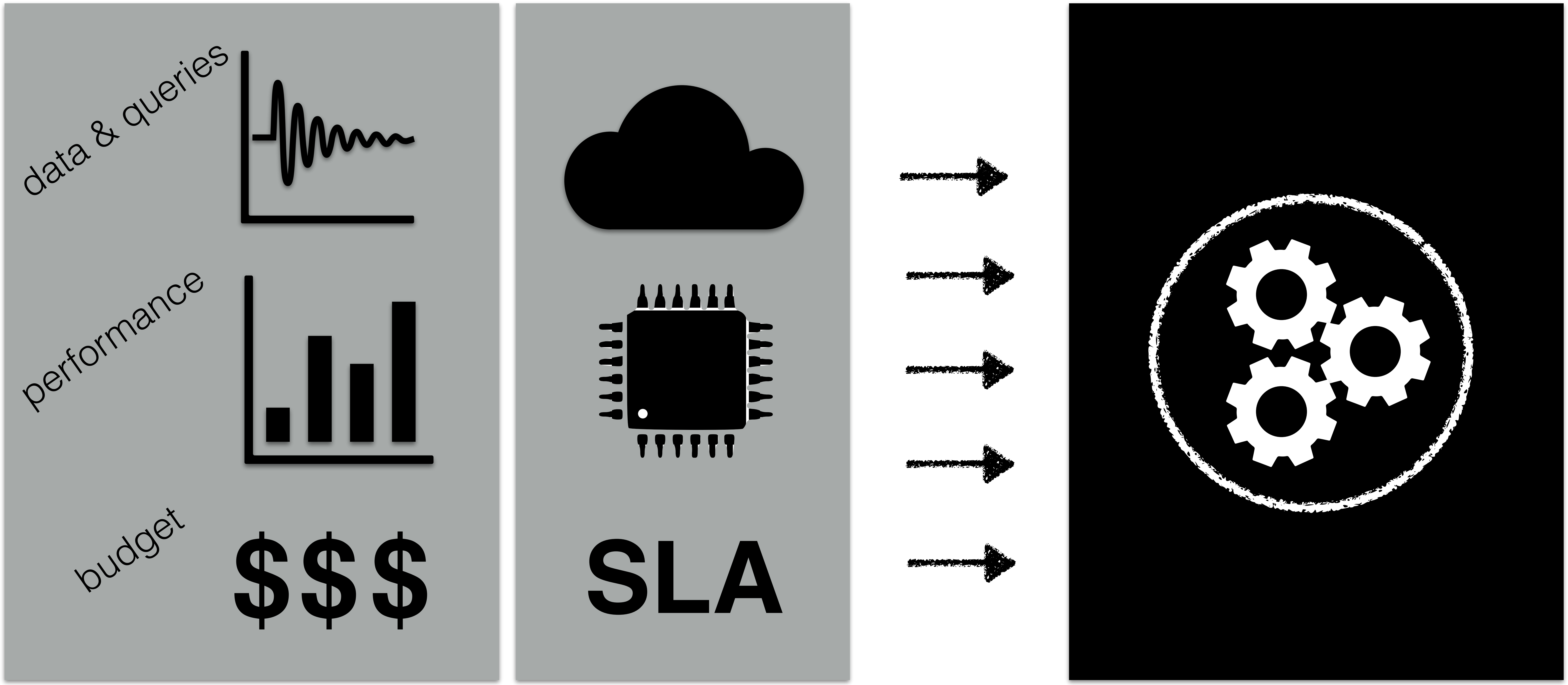
robustness
cloud cost
SLAs

Requirements/Goals

Context

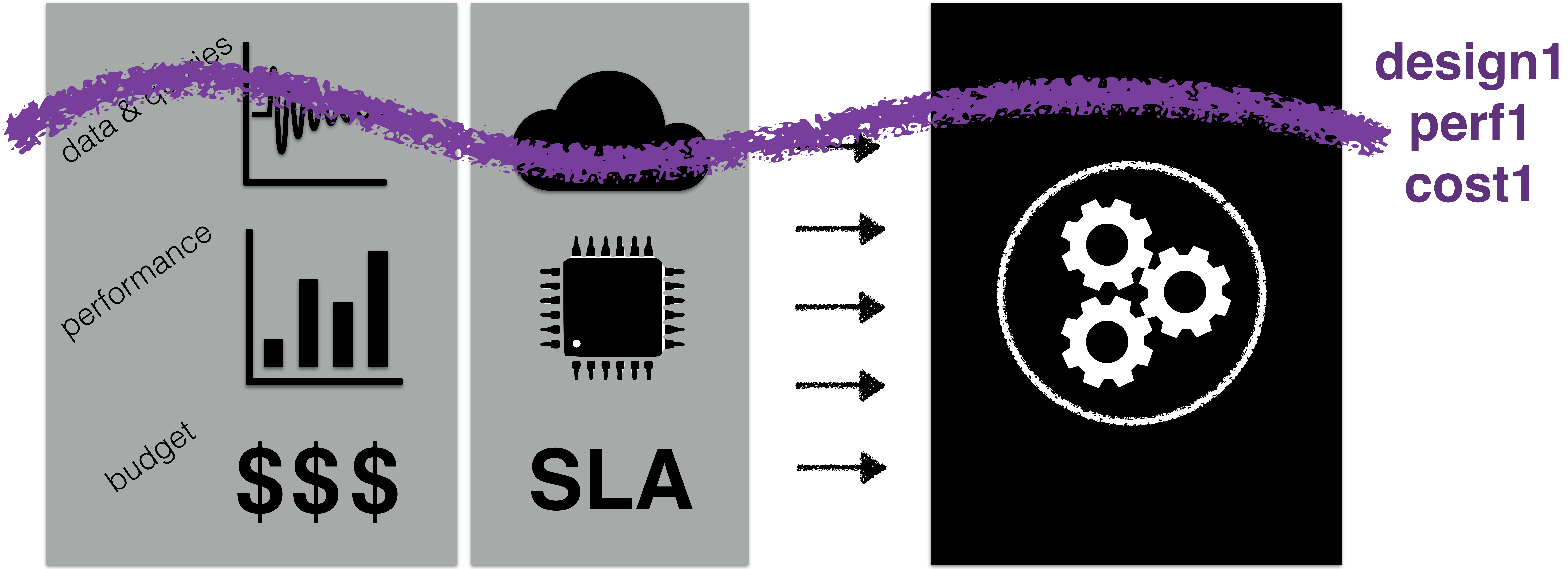


DASlab
@ Harvard SEAS

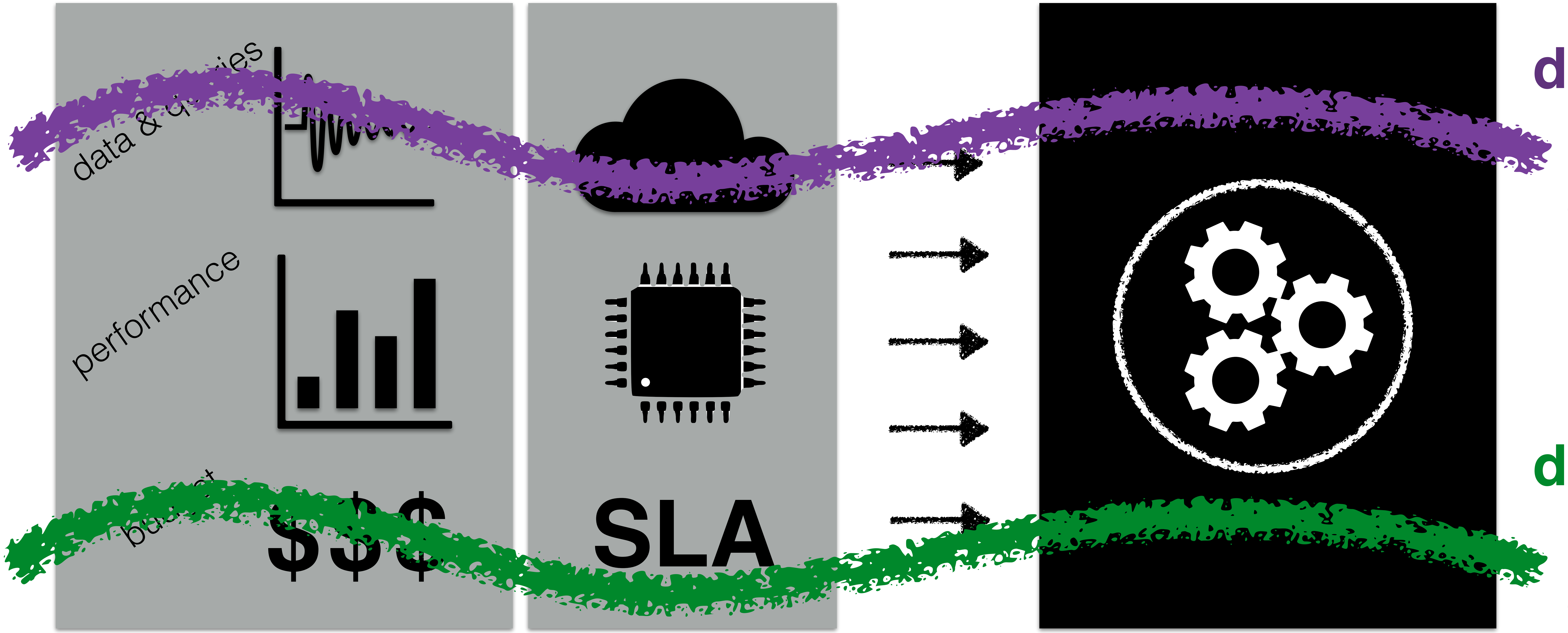


what-if reasoning

DASlab @ Harvard SEAS



what-if reasoning



design1
perf1
cost1

design2
perf2
cost2

what-if reasoning

AUTO DESIGN

Rob Tarjan, Turing Award 1986

"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 SYSTEMS?

Rob Tarjan, Turing Award 1986

"**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*] the **grammar** of systems design

Nikos Kazantzakis, philosopher

the **grammar** of systems design

action *is* *for* *nothing* *holy*
hope *the* *most* *of*
fear *form*
am *free*
ultimate *I* *theory*

the **grammar** of systems design

Nikos Kazantzakis, philosopher

action is

holy

the most

of

form

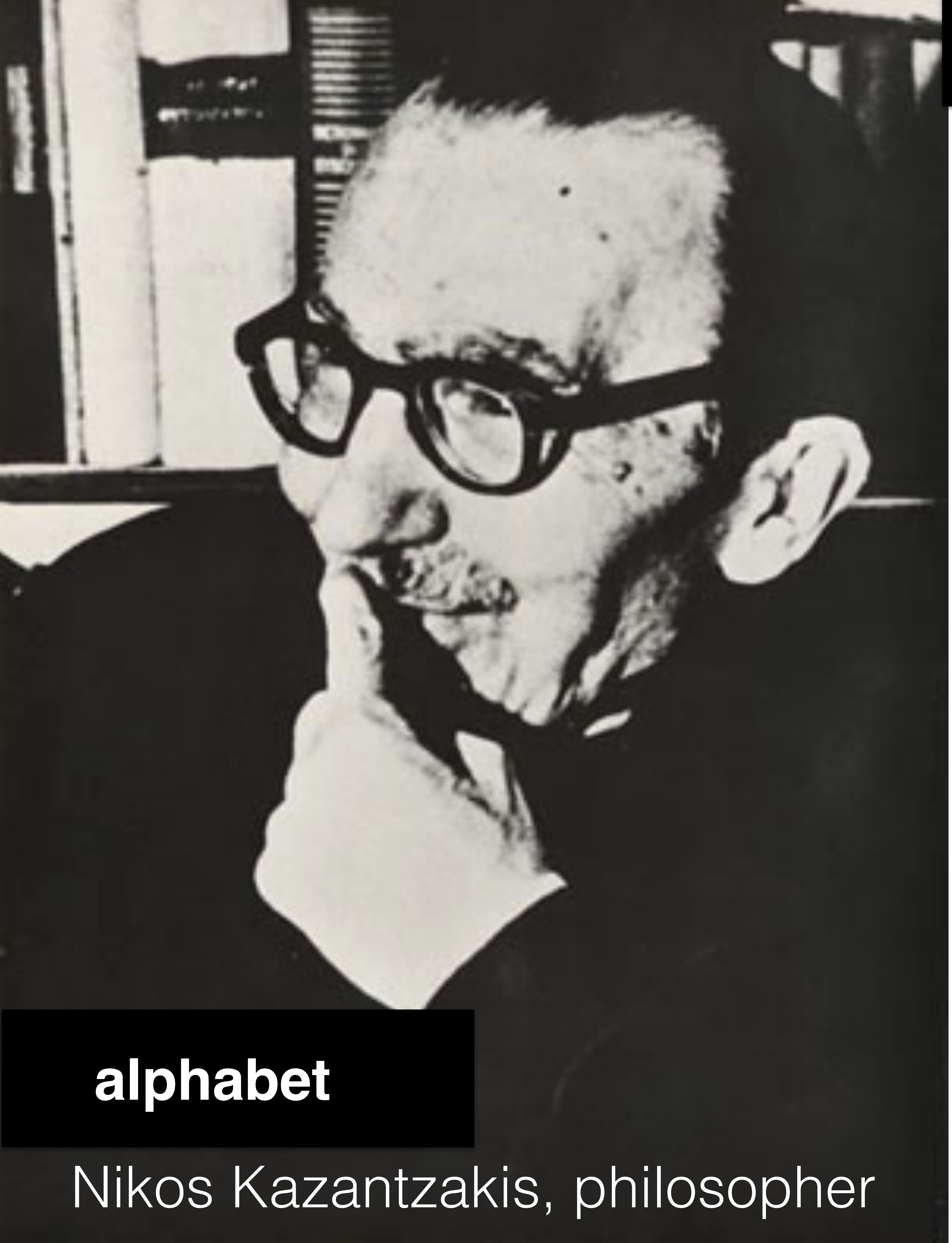
ultimate theory

I hope for nothing

I fear nothing

I am free

the **grammar** of systems design



Nikos Kazantzakis, philosopher

action is

holy

the most of

form

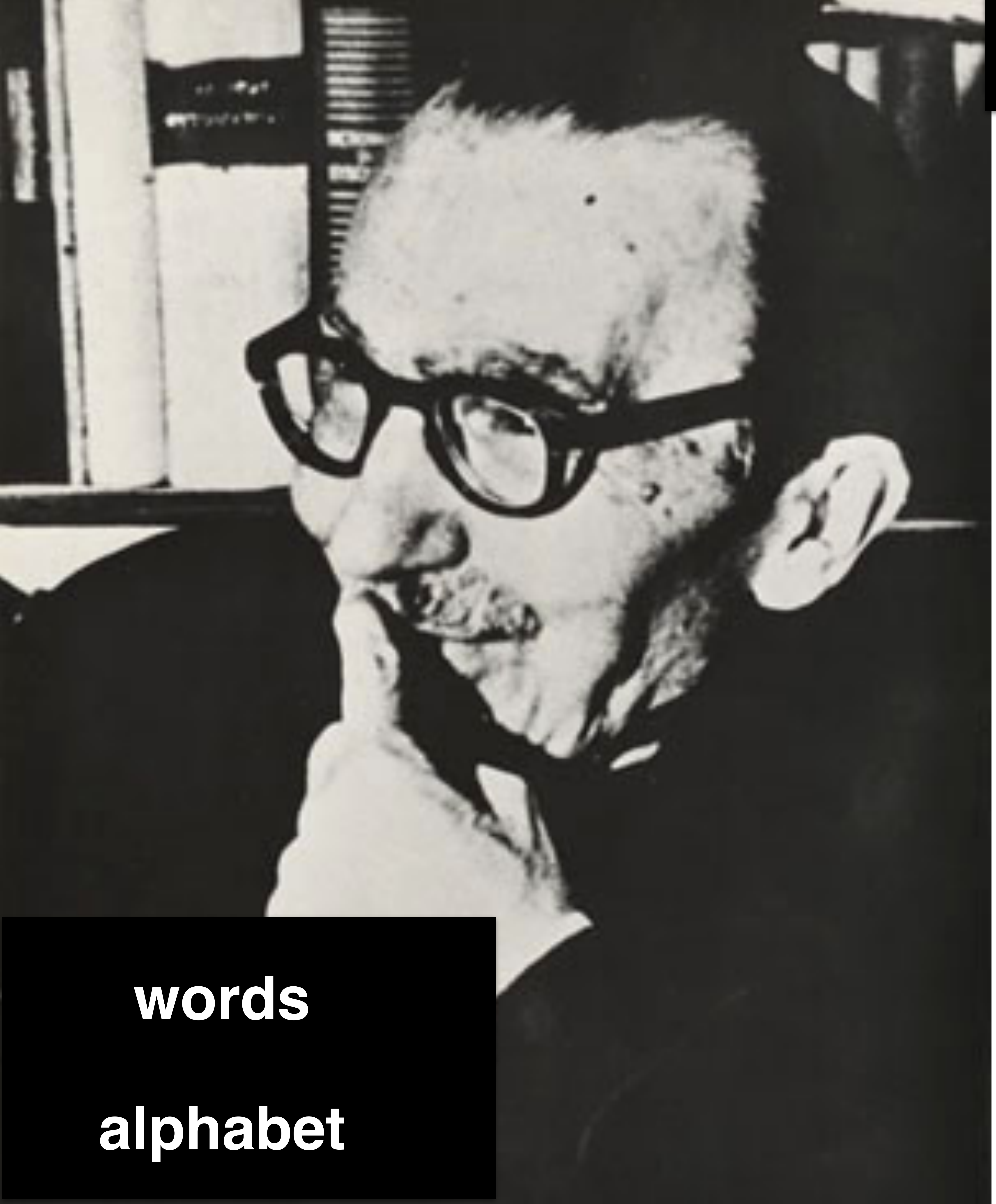
ultimate theory

I hope for nothing

I fear nothing

I am free

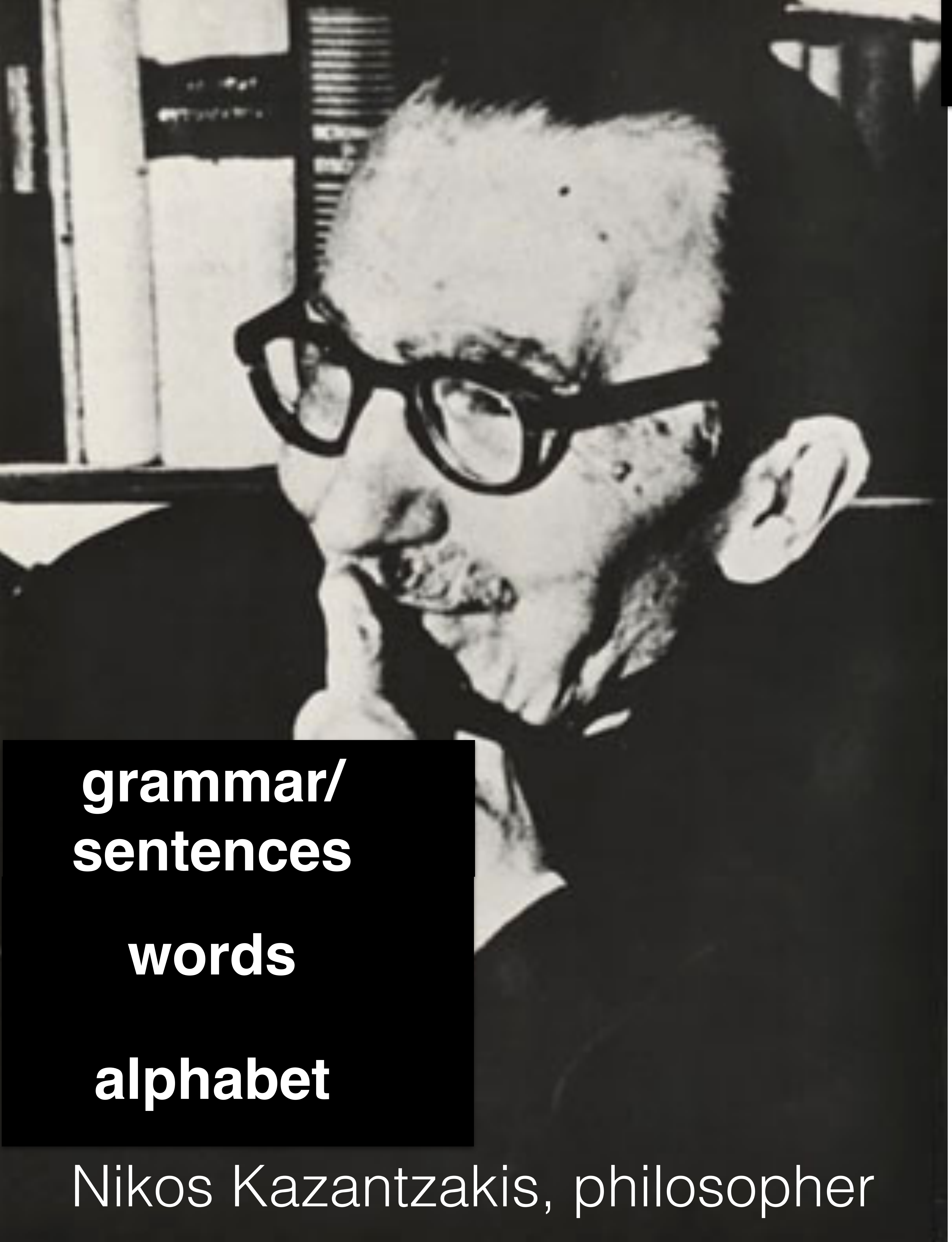
the **grammar** of systems design



Nikos Kazantzakis, philosopher

action is
the most holy of
form
ultimate theory

I hope for nothing
I fear nothing
I am free



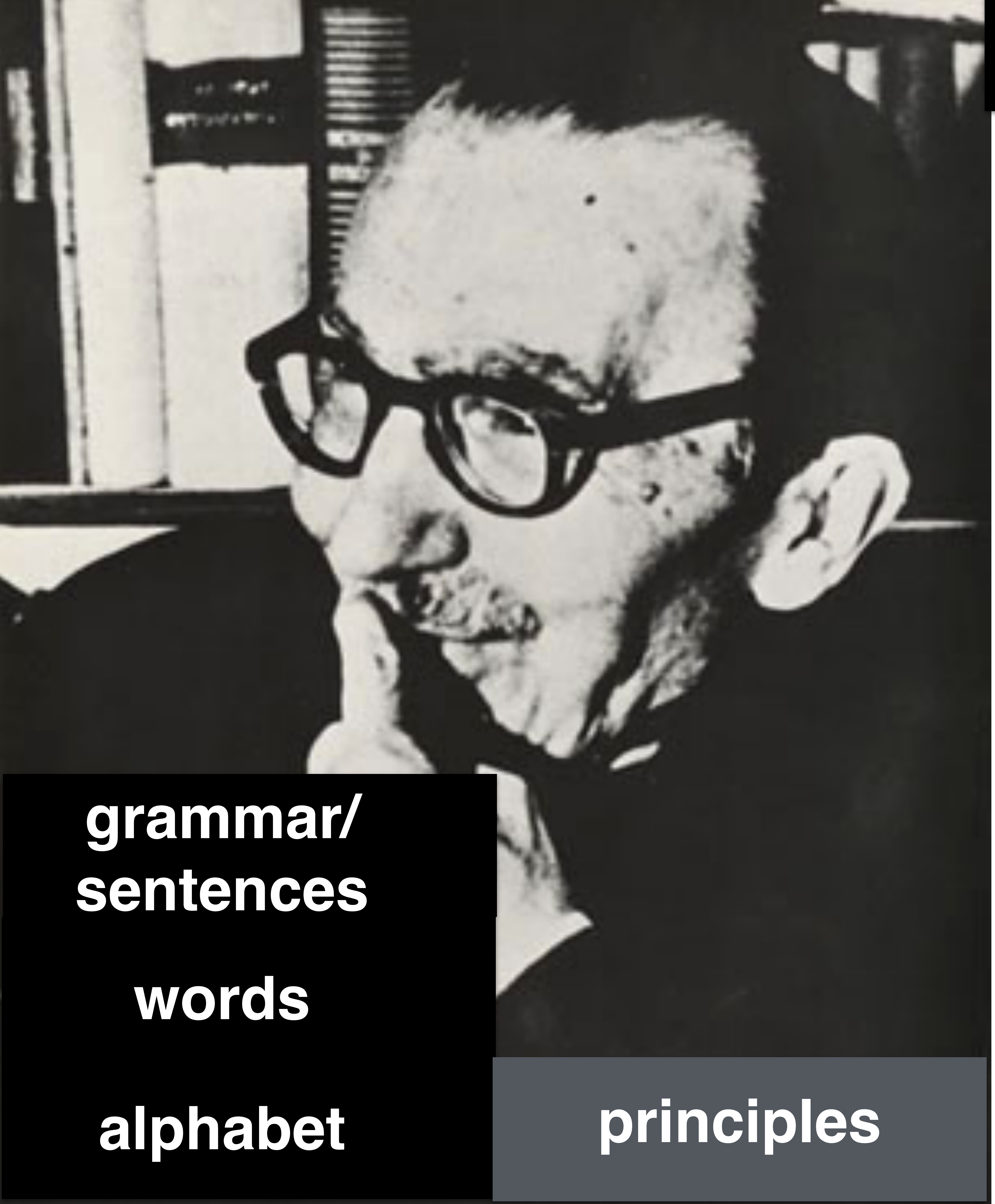
Nikos Kazantzakis, philosopher

the **grammar** of systems design

action is
the most holy of
form
ultimate theory

I hope for nothing
I fear nothing
I am free

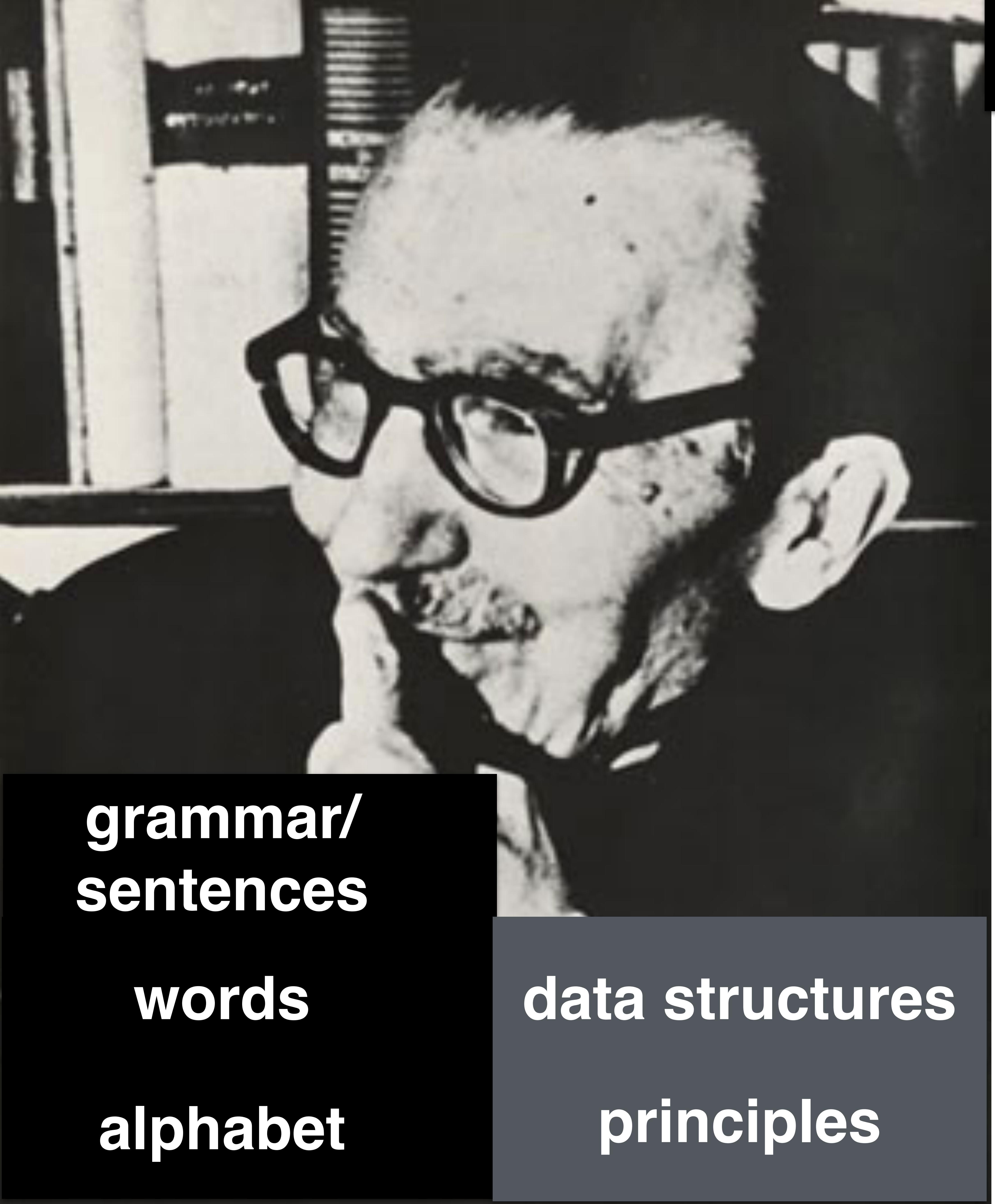
the **grammar** of systems design



Nikos Kazantzakis, philosopher

action is
the most holy
form of
ultimate theory

I hope for nothing
I fear nothing
I am free



grammar/
sentences

words

alphabet

data structures

principles

Nikos Kazantzakis, philosopher

action is

the most holy

form of

ultimate theory

I hope for nothing

I fear nothing

I am free

the **grammar** of systems design

grammar/ sentences	**interactions**
words	**data structures**
alphabet	**principles**

Nikos Kazantzakis, philosopher

action is the most holy form of theory

ultimate

I hope for nothing
I fear nothing
I am free

the **grammar** of systems design

grammar/ sentences	**interactions**
words	**data structures**
alphabet	**principles**

Nikos Kazantzakis, philosopher

action is
the most holy
of
form
ultimate theory

NEW

I hope for nothing
I fear nothing
I am free

the **grammar** of systems design

grammar/ sentences	**interactions**
words	**data structures**
alphabet	**principles**

Nikos Kazantzakis, philosopher

action is the most holy form of theory

which are "all" possible *systems* **we may ever invent?**

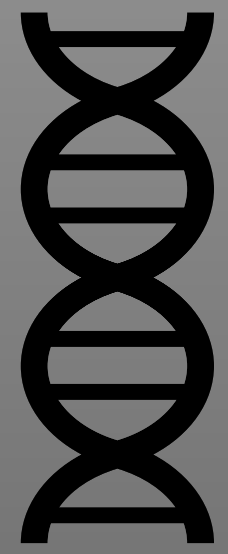
I hope for nothing
I fear nothing
I am free

DESIGN SPACE

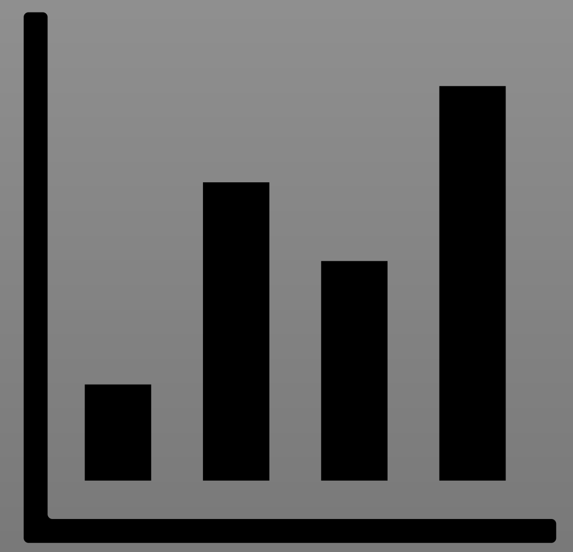
PERFORMANCE ESTIMATION

FIND BEST DESIGN

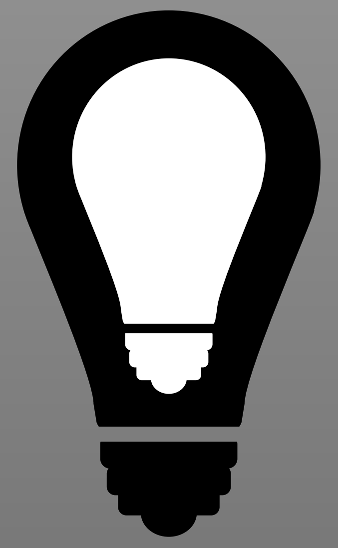
PERPETUAL LEARNING POSSIBLE



DESIGN SPACE



PERFORMANCE ESTIMATION



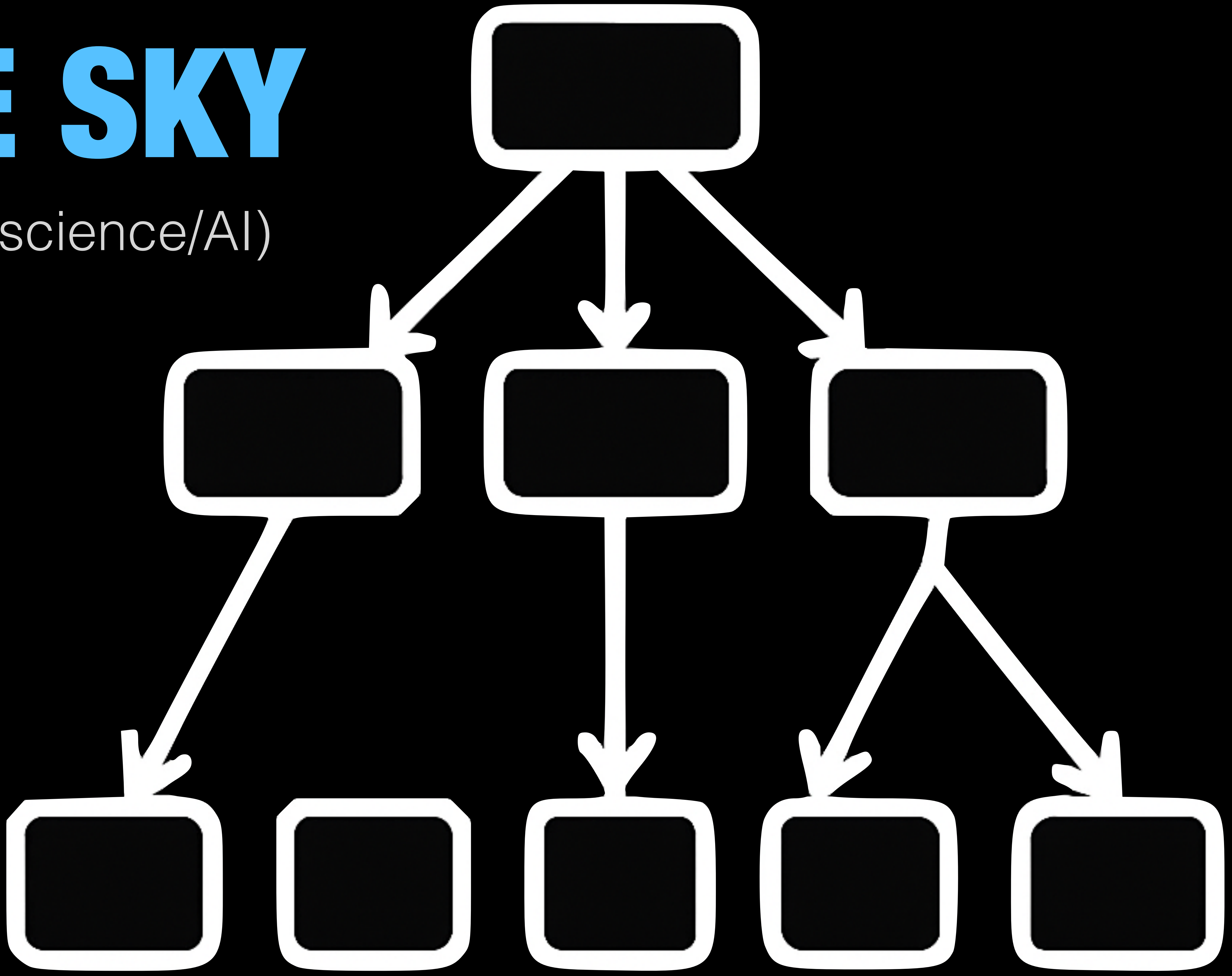
FIND BEST DESIGN

SIGMOD'18

MORE DATA STRUCTURES THAN STARS IN THE SKY

(The most fundamental component of computer science/AI)

$>10^{48}$



10^{24}

5×10^{3}

Literature

Stars

Possibilities We Discovered

10-100X FASTER SYSTEMS

Limousine: NoSQL KV-Store

Agents' context management, but also all kinds of big data infra

SIGMOD'24, VLDB'22

Image Calculator: Image AI

Storage for Training and Infenence

SIGMOD'24, CIDR'25

TorchTitan with PyTorch@META

Large Model Training Algorithms

MLsys 2023, ICLR'25

Now doing the same with RAG, Agents, LLMs, …

CS 265

Stratos Idreos

BIG DATA SYSTEMS

NoSQL | Neural Networks | Image AI | LLMs | Data Science