Scaling Log-Structured KV-Stores

featuring

Monkey, Dostoevsky and **Wacky**

SIGMOD17/18/19

DASlab @ Harvard SEAS

Niv Dayan

Log-Structured KV-Stores

RocksDB

BigTable

Amazon DynamoDB

riak

cassandra

WT

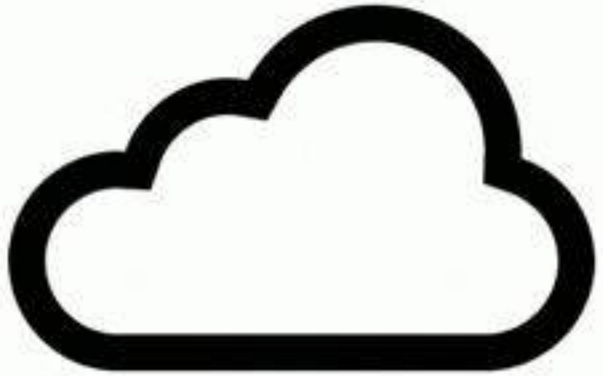
accumulo

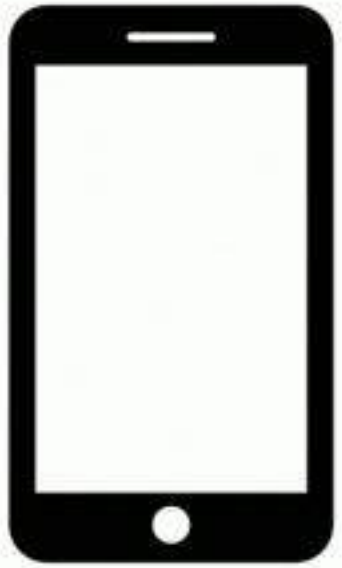
HBASE

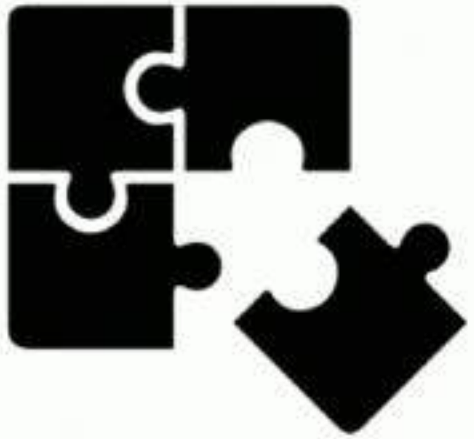
Couchbase

levelDB

Log-Structured KV-Stores







Why Log-Structured KV-Stores?

Why Log-Structured KV-Stores?

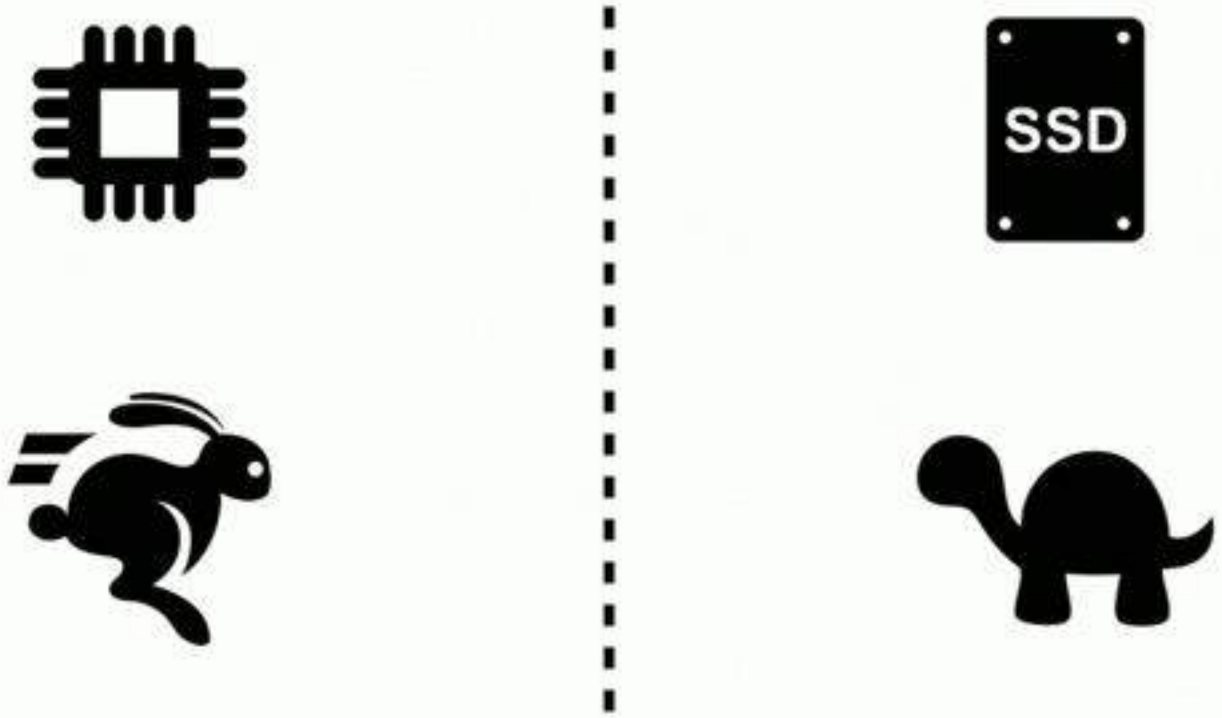
fast writes

Why Log-Structured KV-Stores?

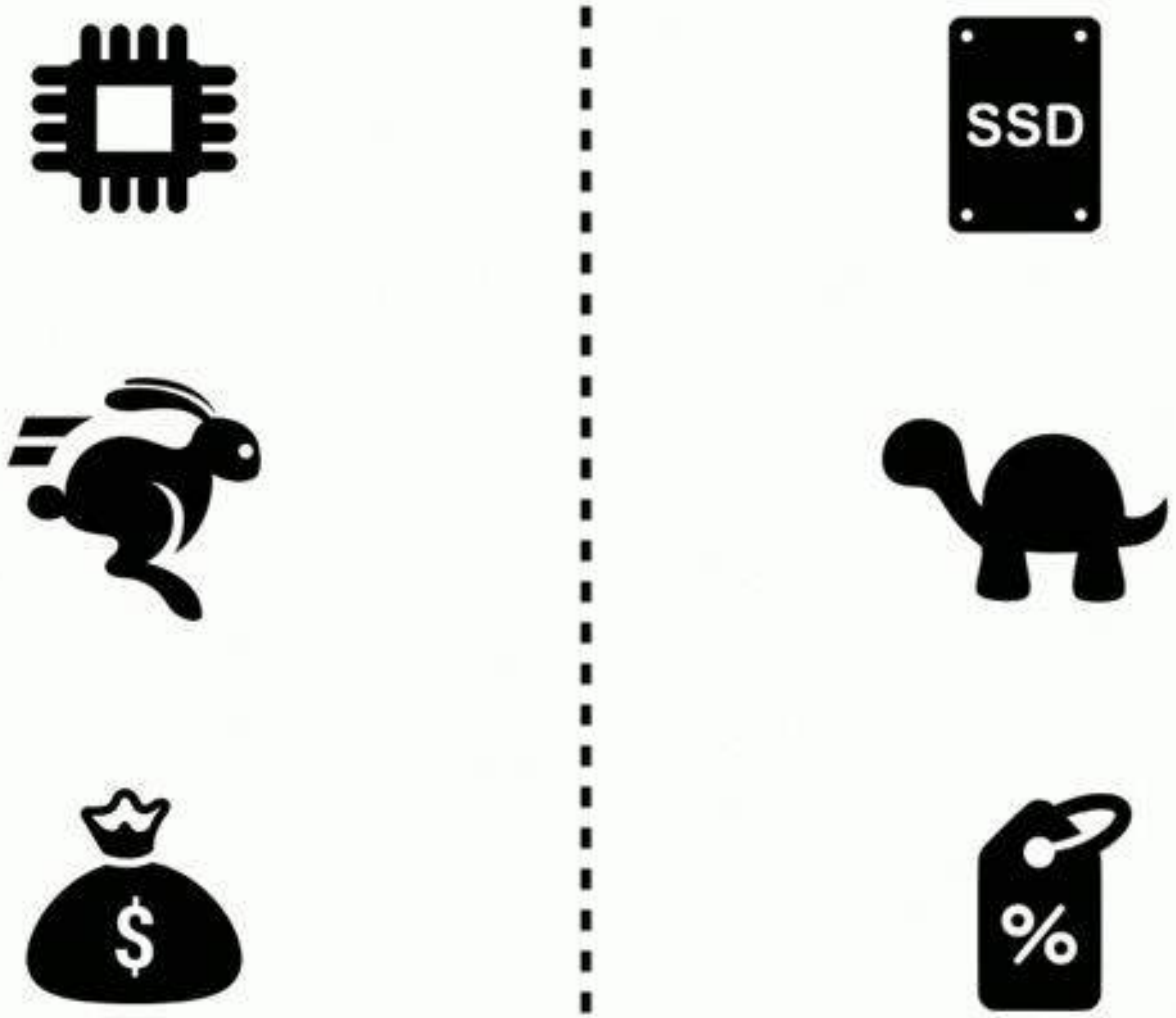
memory

storage

Why Log-Structured KV-Stores?



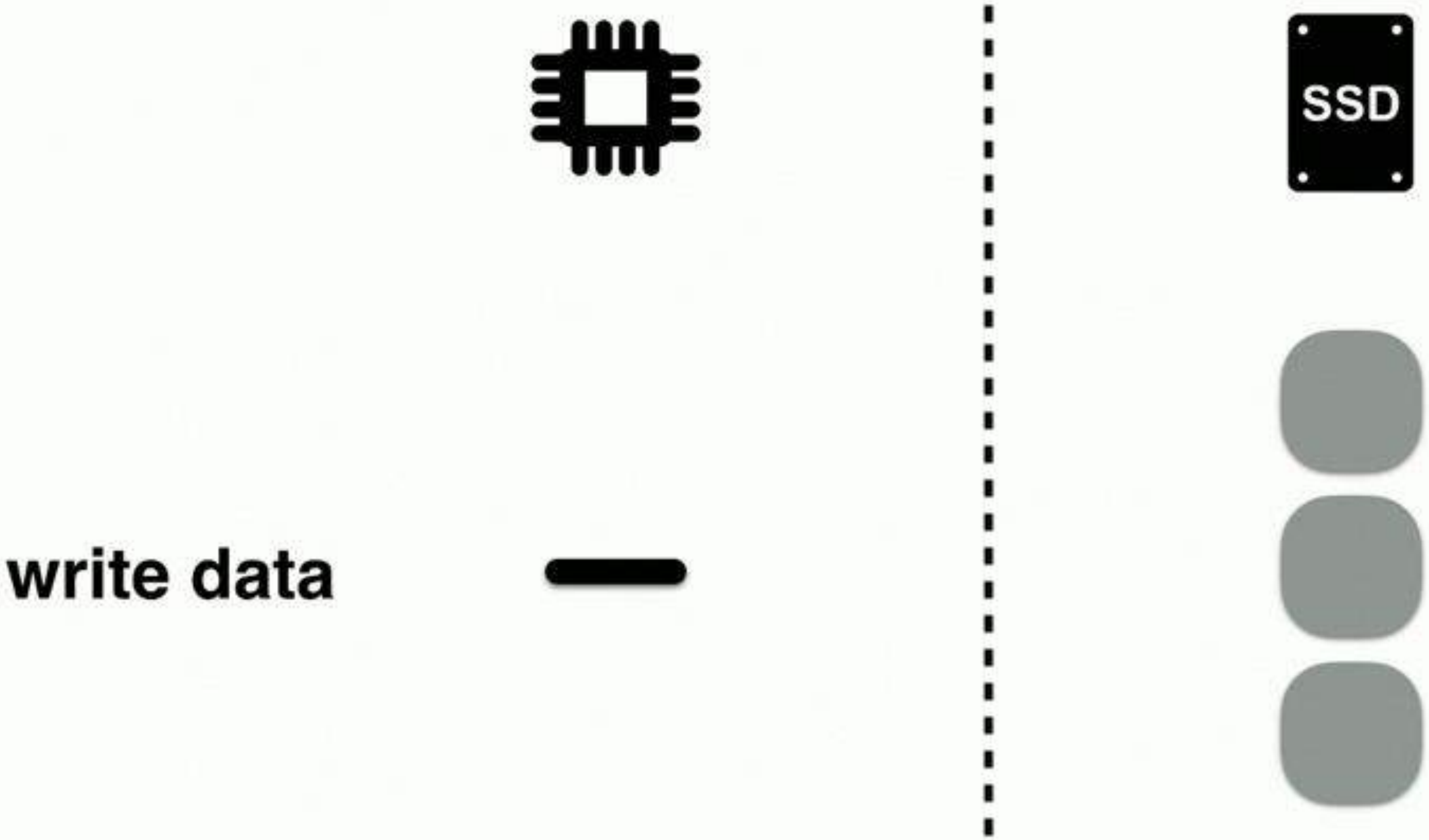
Why Log-Structured KV-Stores?

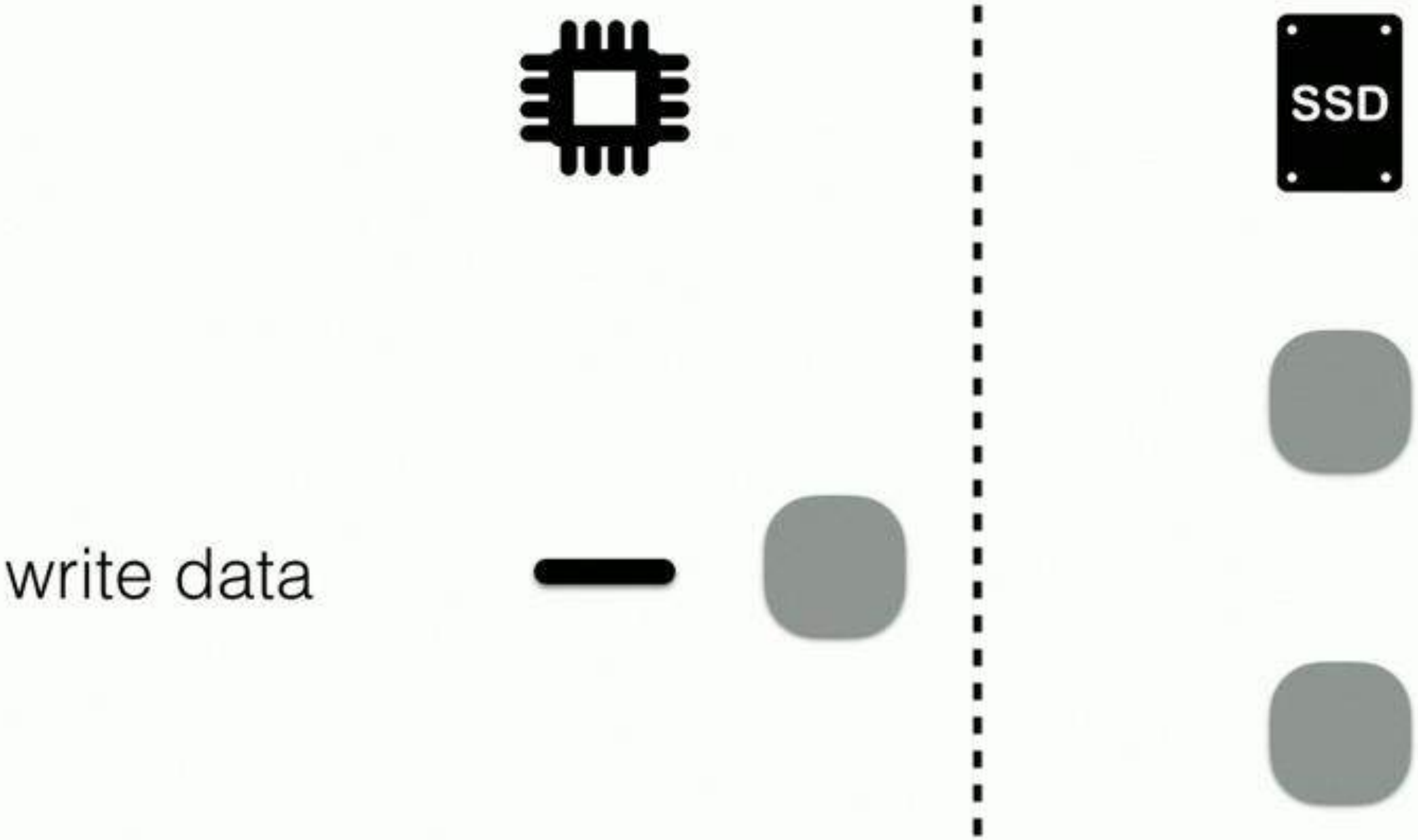


Why Log-Structured KV-Stores?

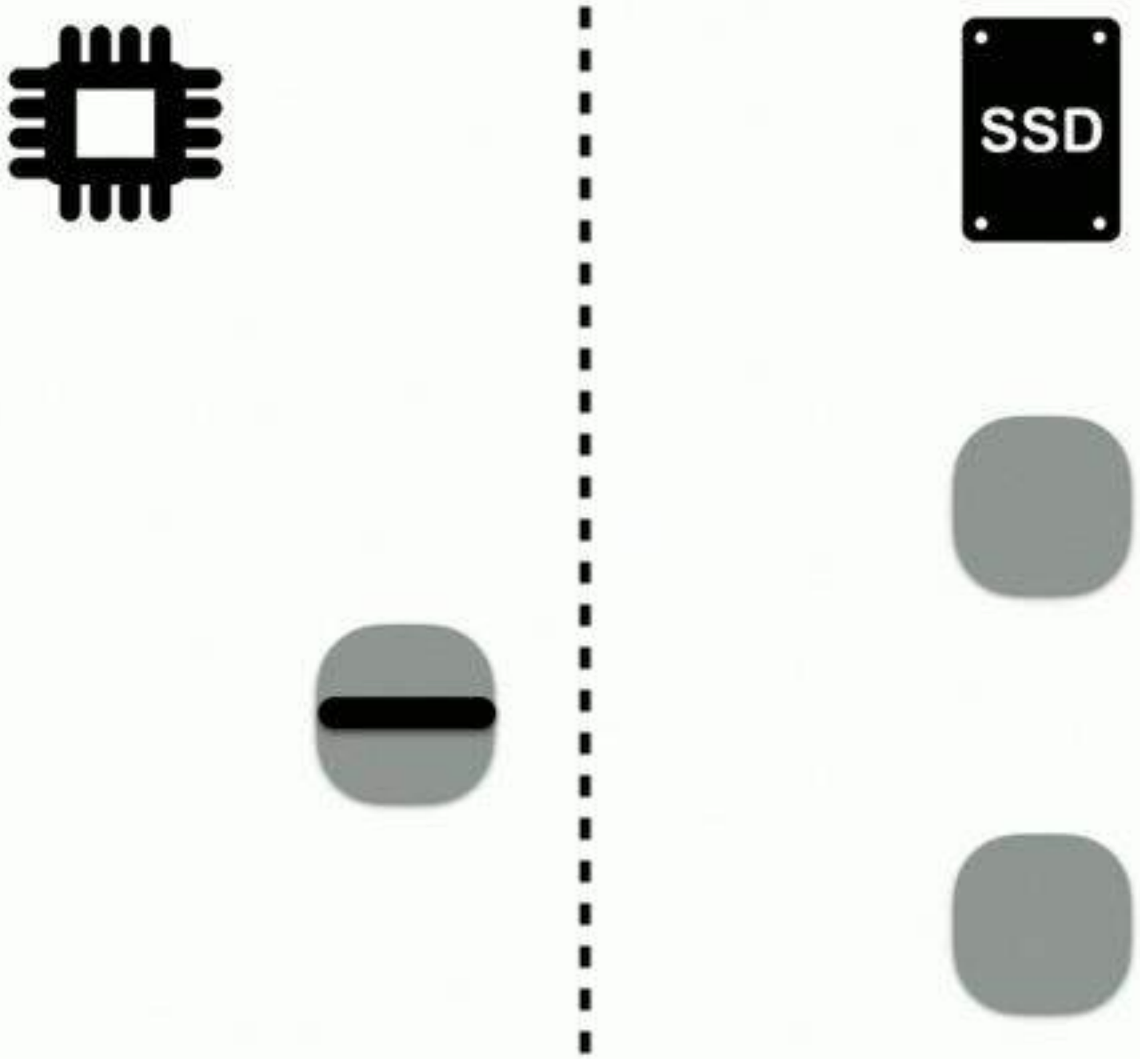
byte-addressable

block-addressable

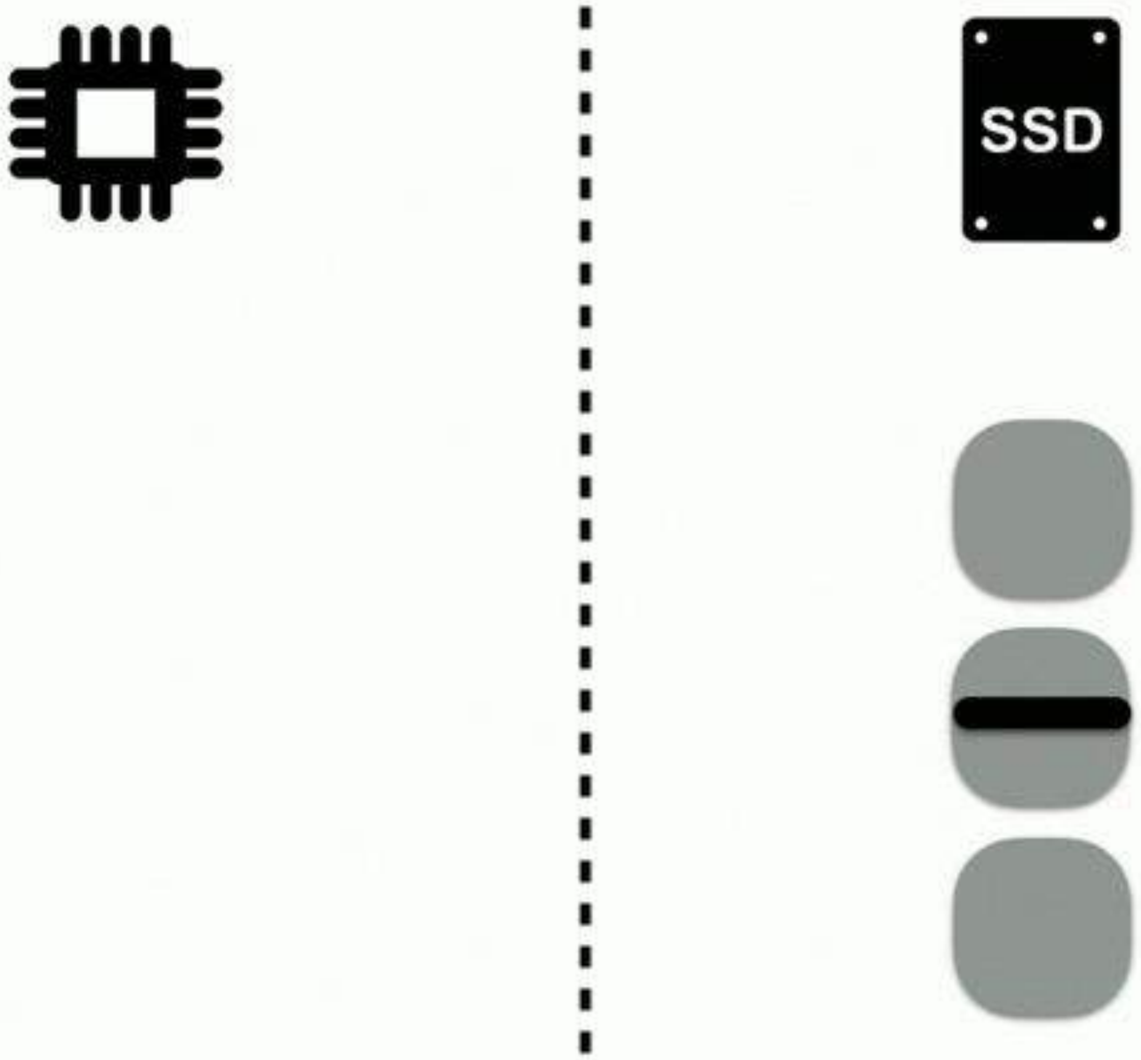

write data
SSD


SSD
write data

write data



In-Place Writes



write data

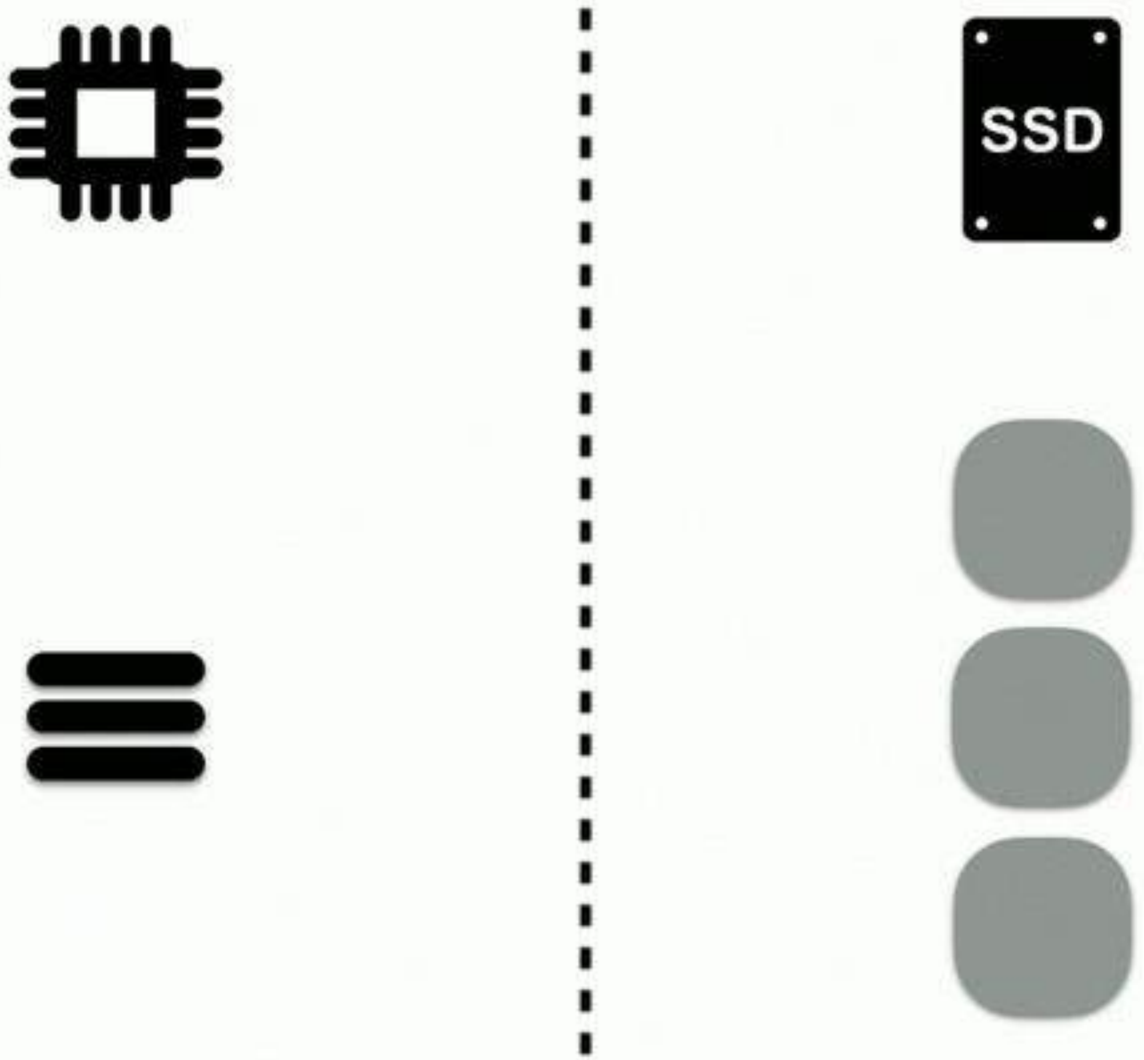
In-Place Writes

write data

B-trees

Log-Structured Writes

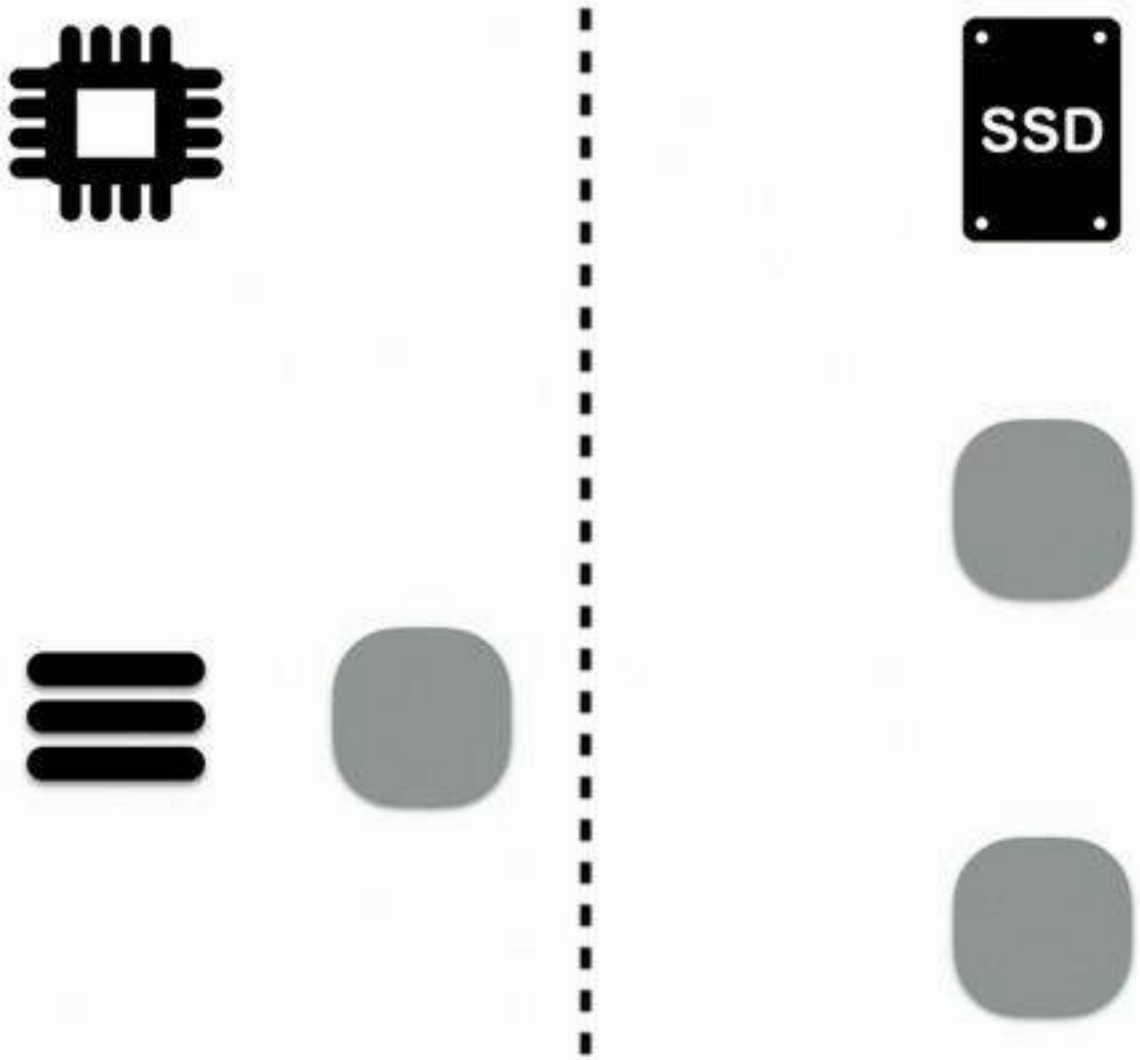
buffer writes



DASlab @ Harvard SEAS

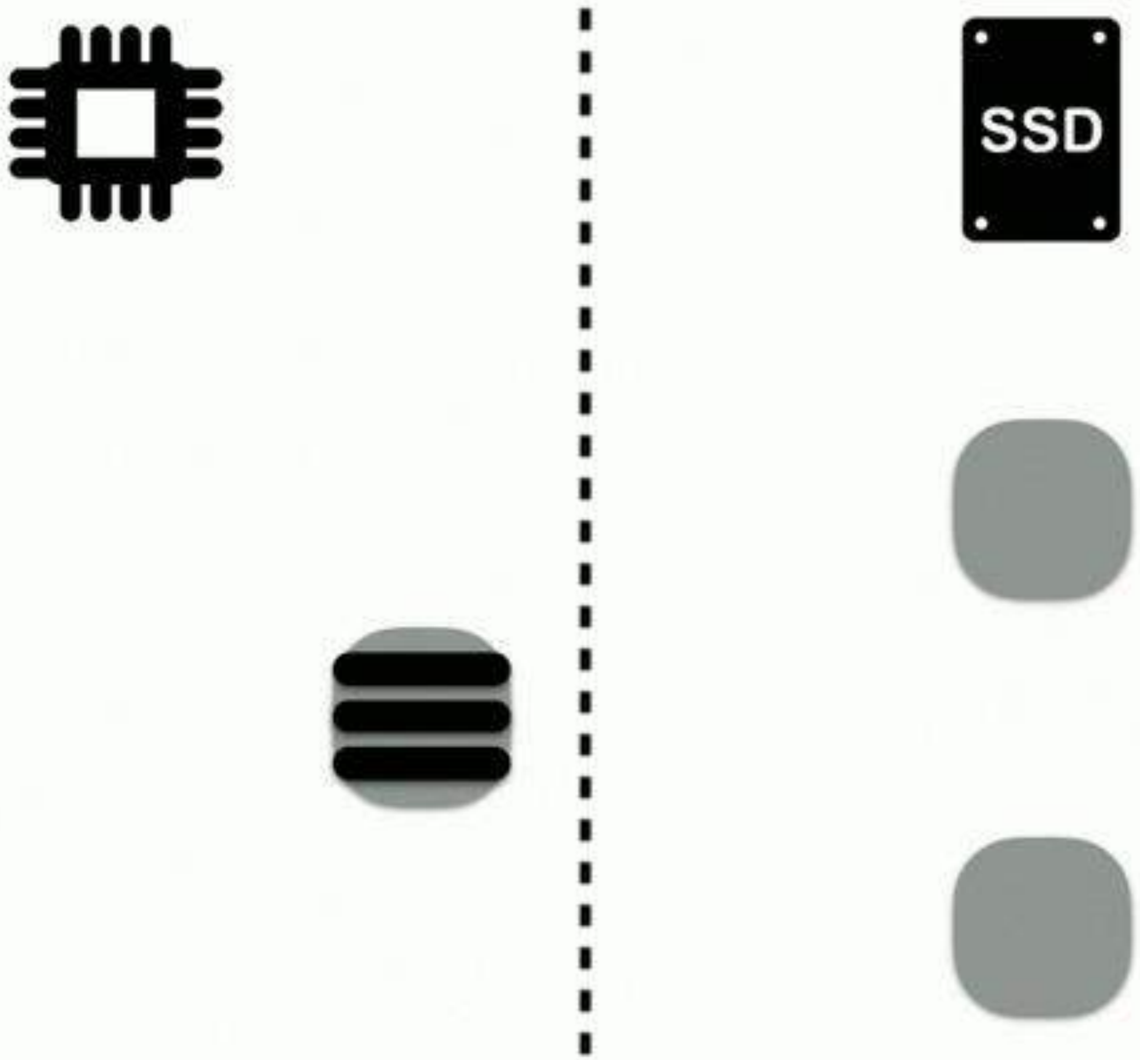
Log-Structured Writes

buffer writes

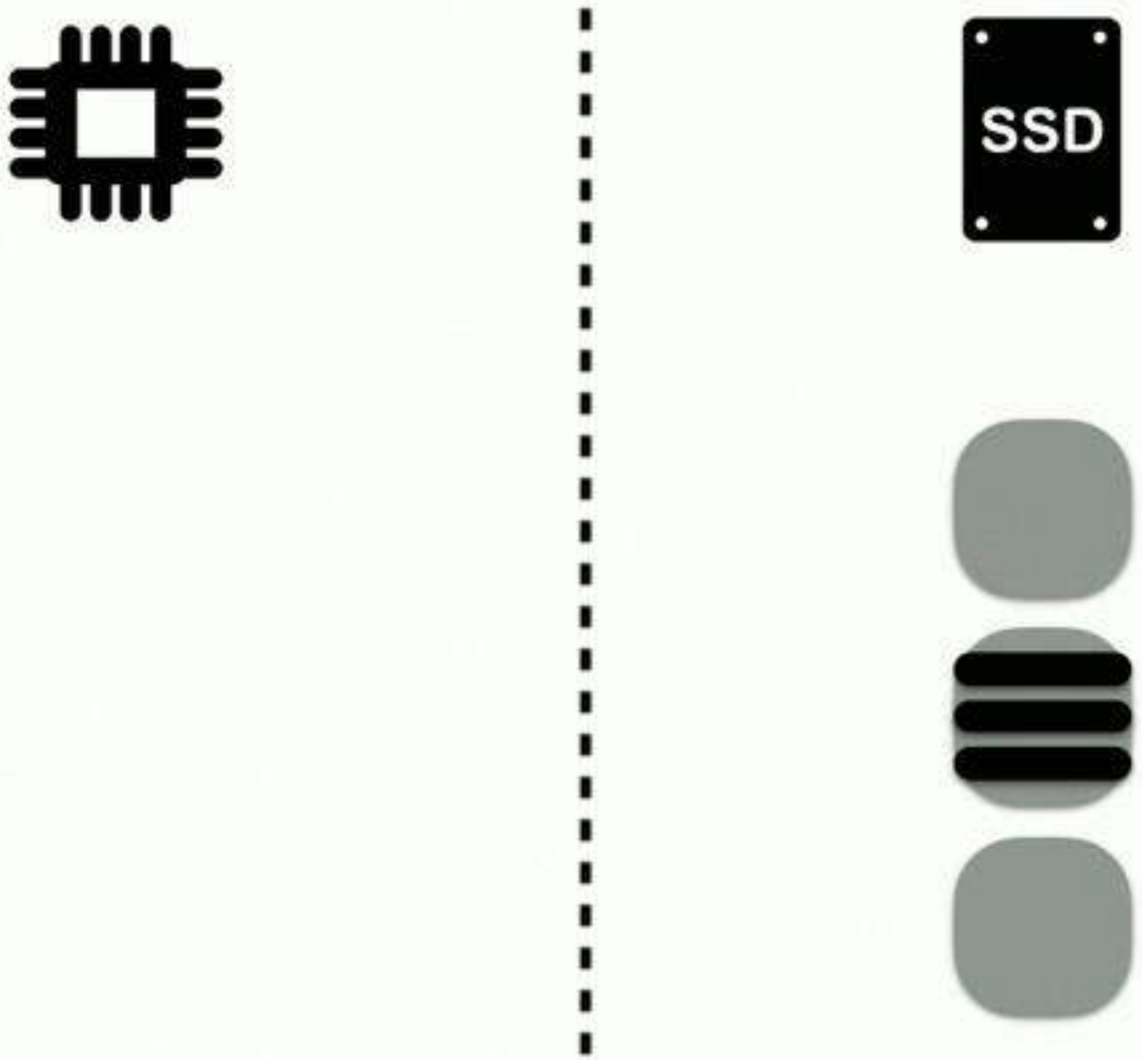


Log-Structured Writes

buffer writes



Log-Structured Writes



buffer writes

Log-Structured KV-Stores

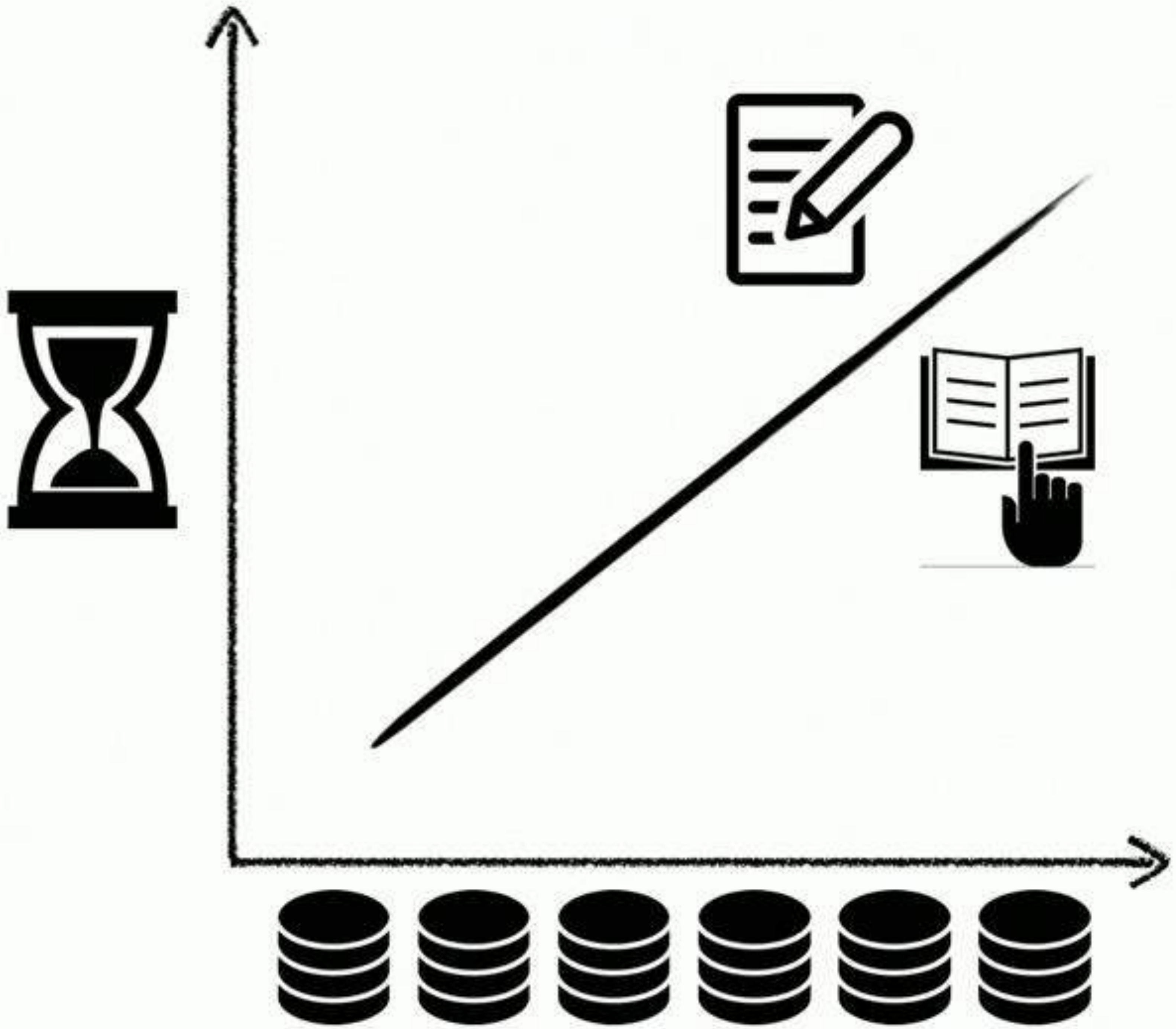
fast writes

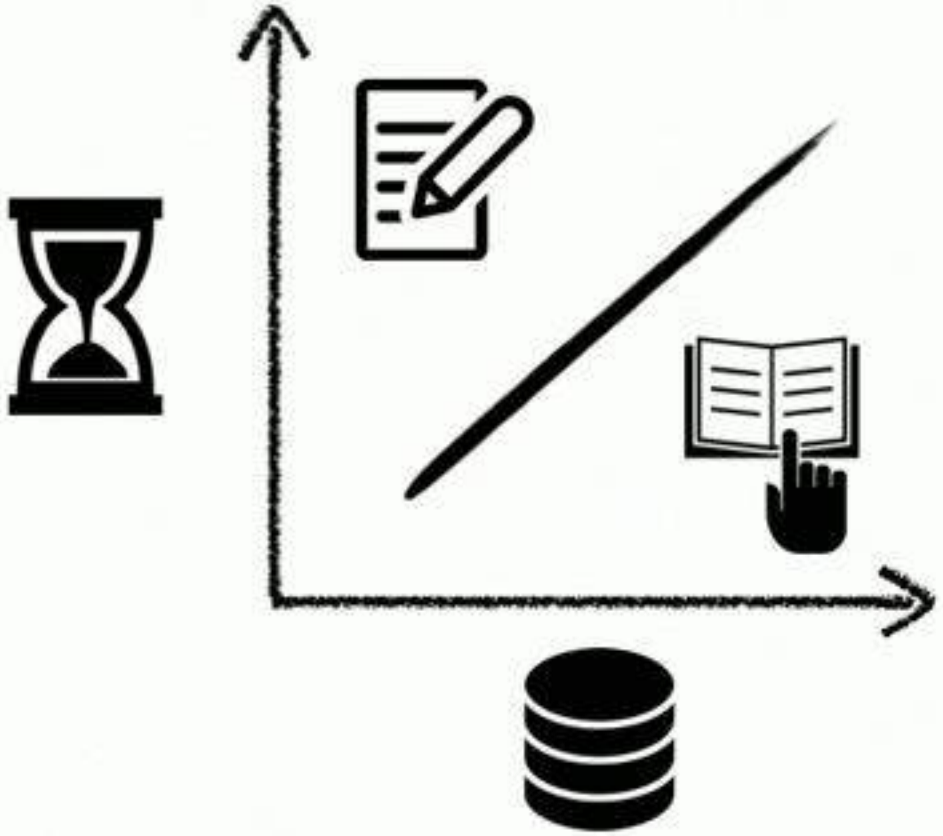
Log-Structured KV-Stores

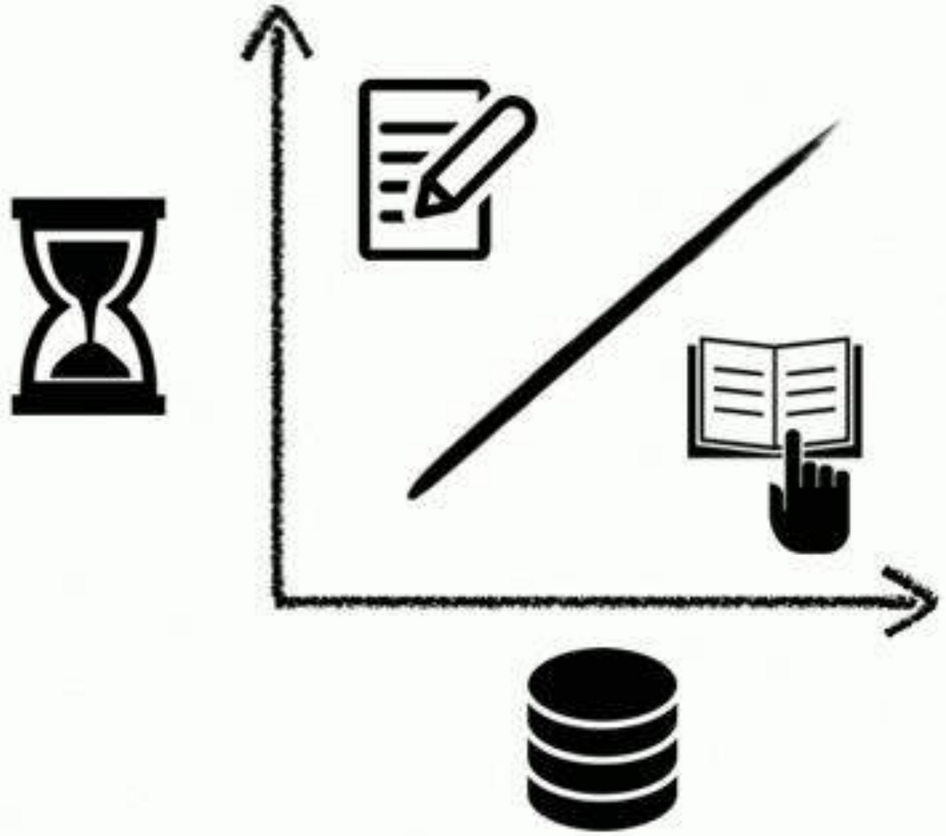
fast writes

fast reads

massive data





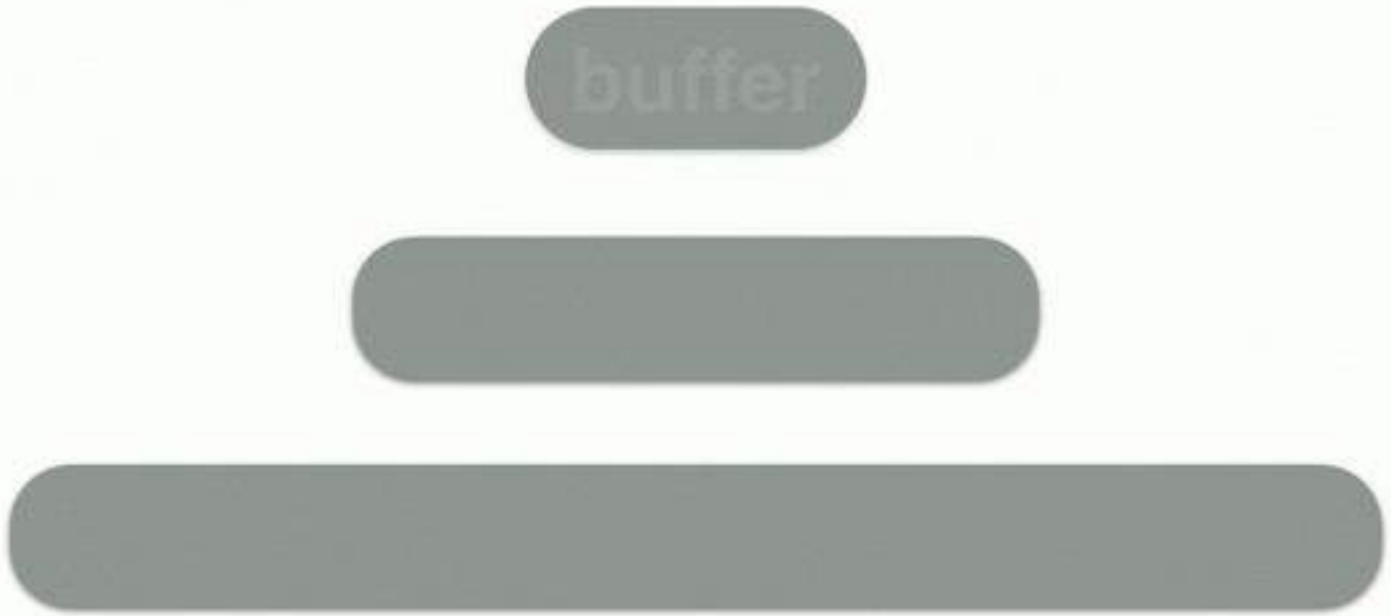


Background

Background

The Log-Structured Merge-Tree

Background



LSM-tree

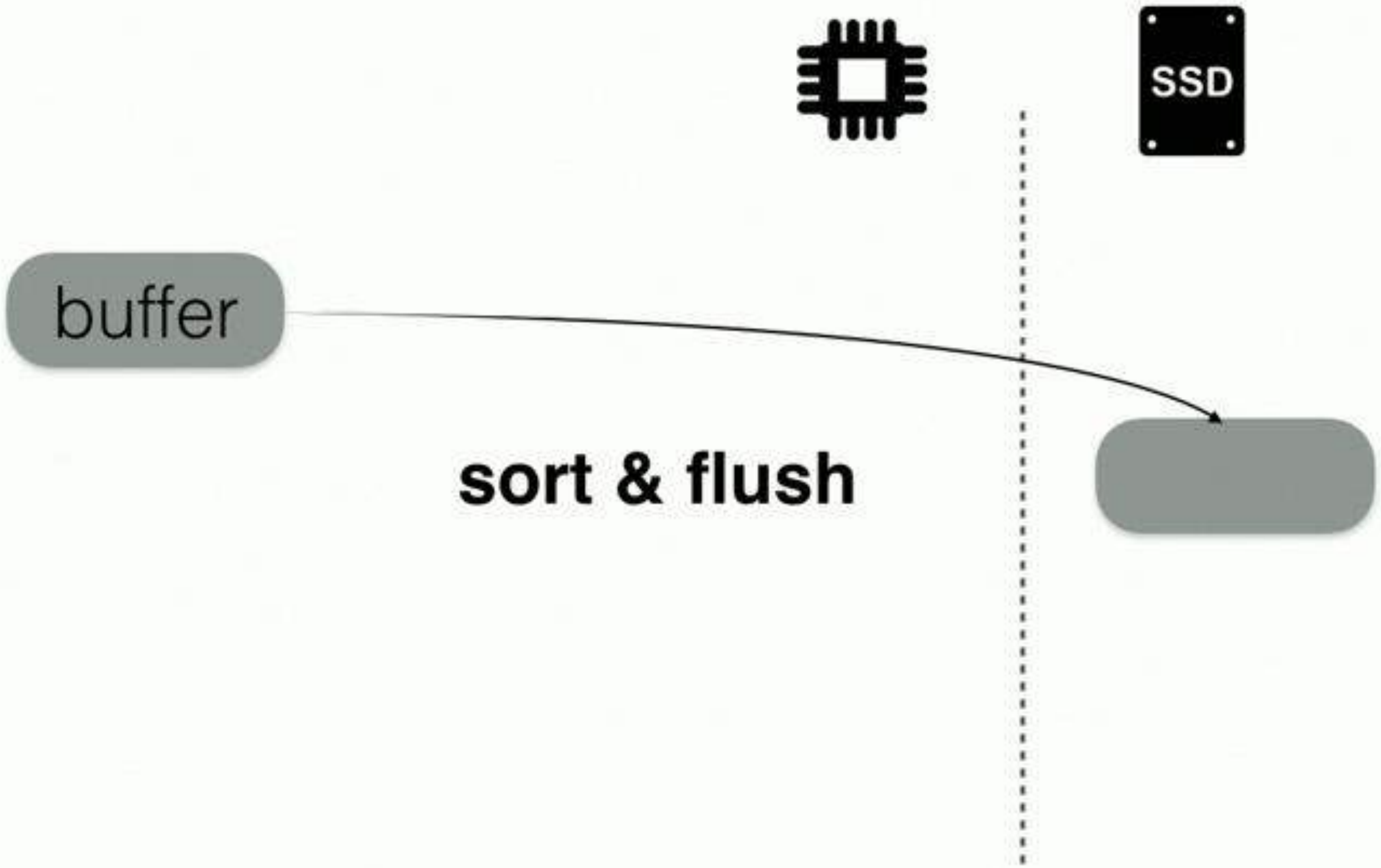
buffer

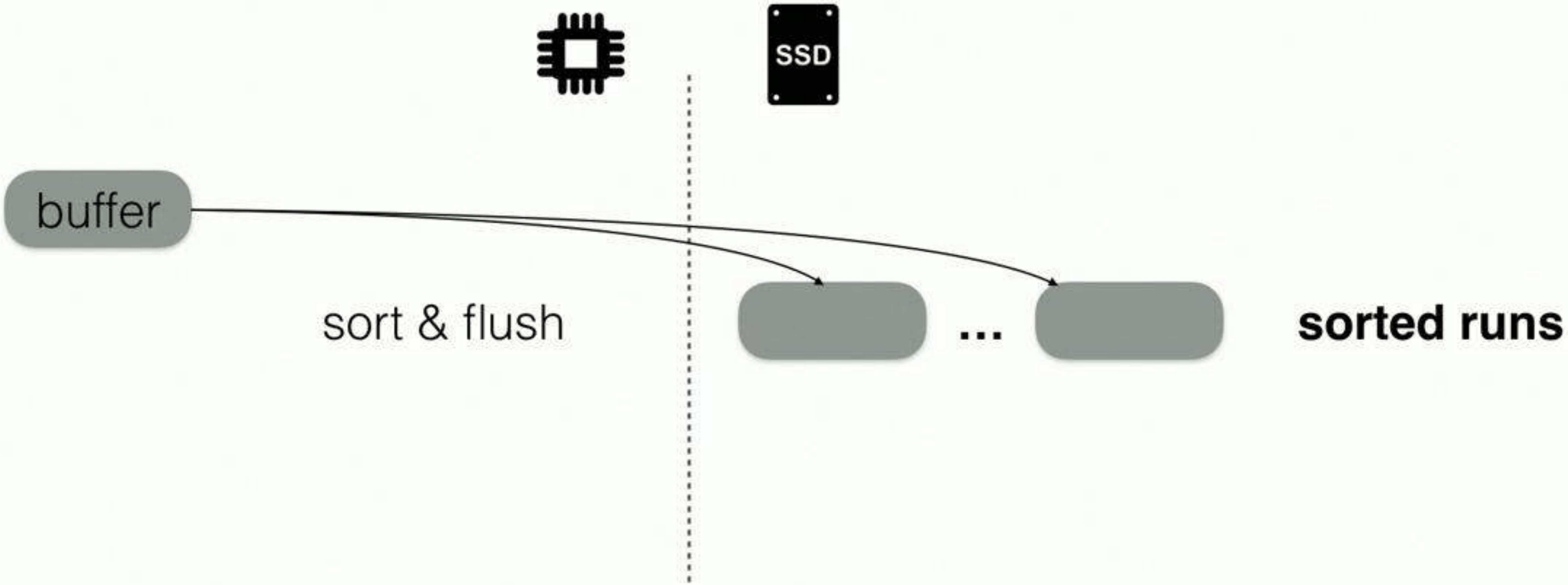
writes

↓

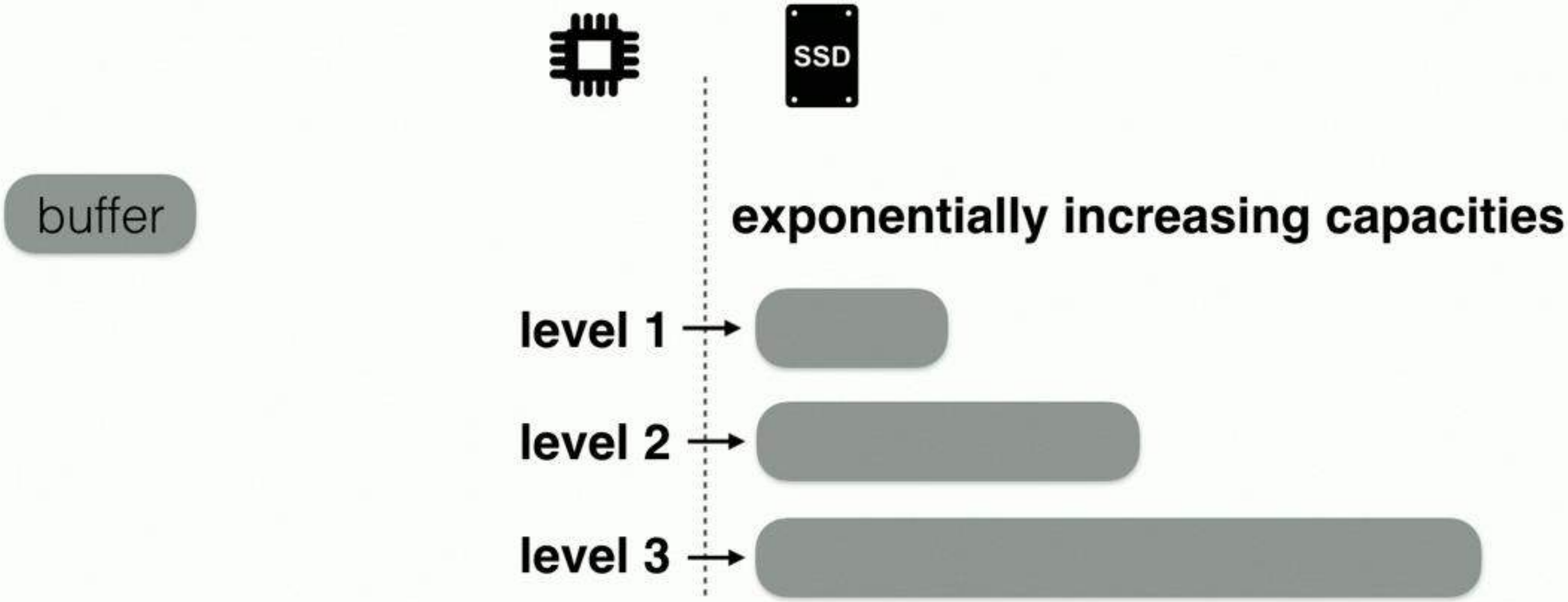
buffer

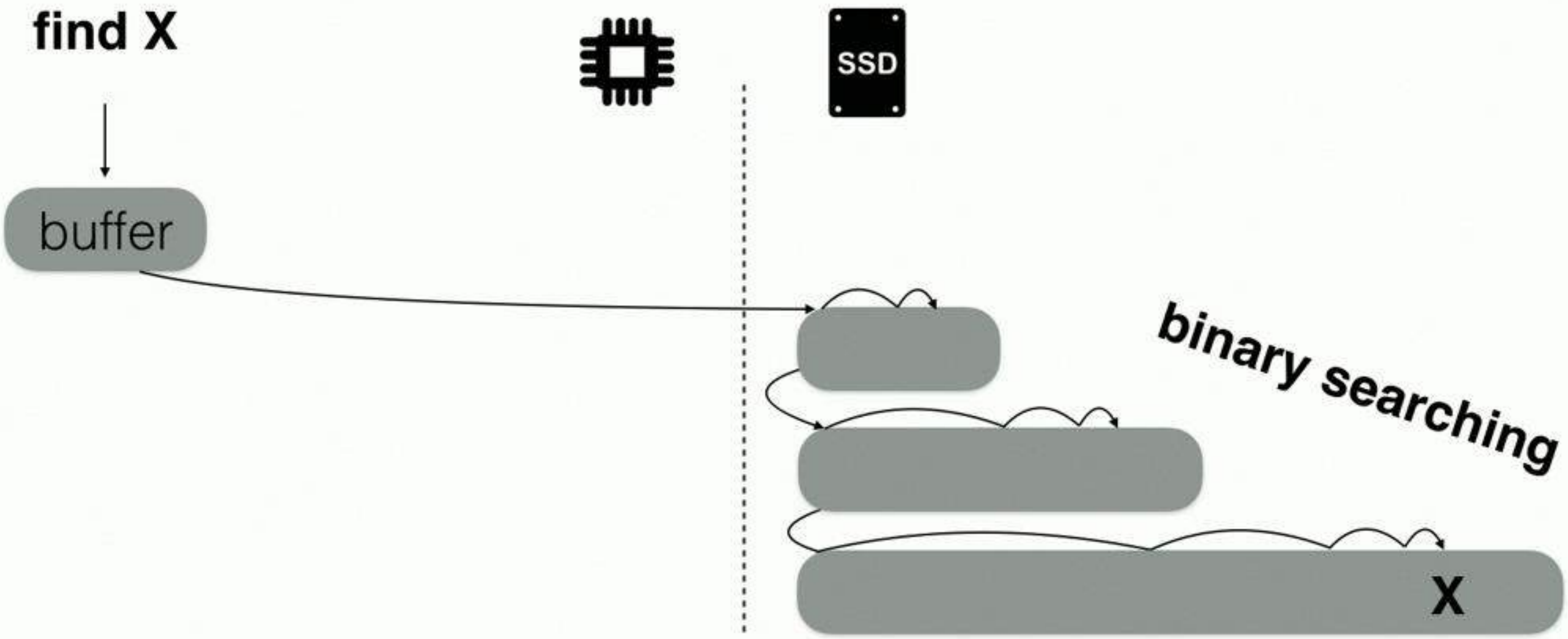
buffer gets full

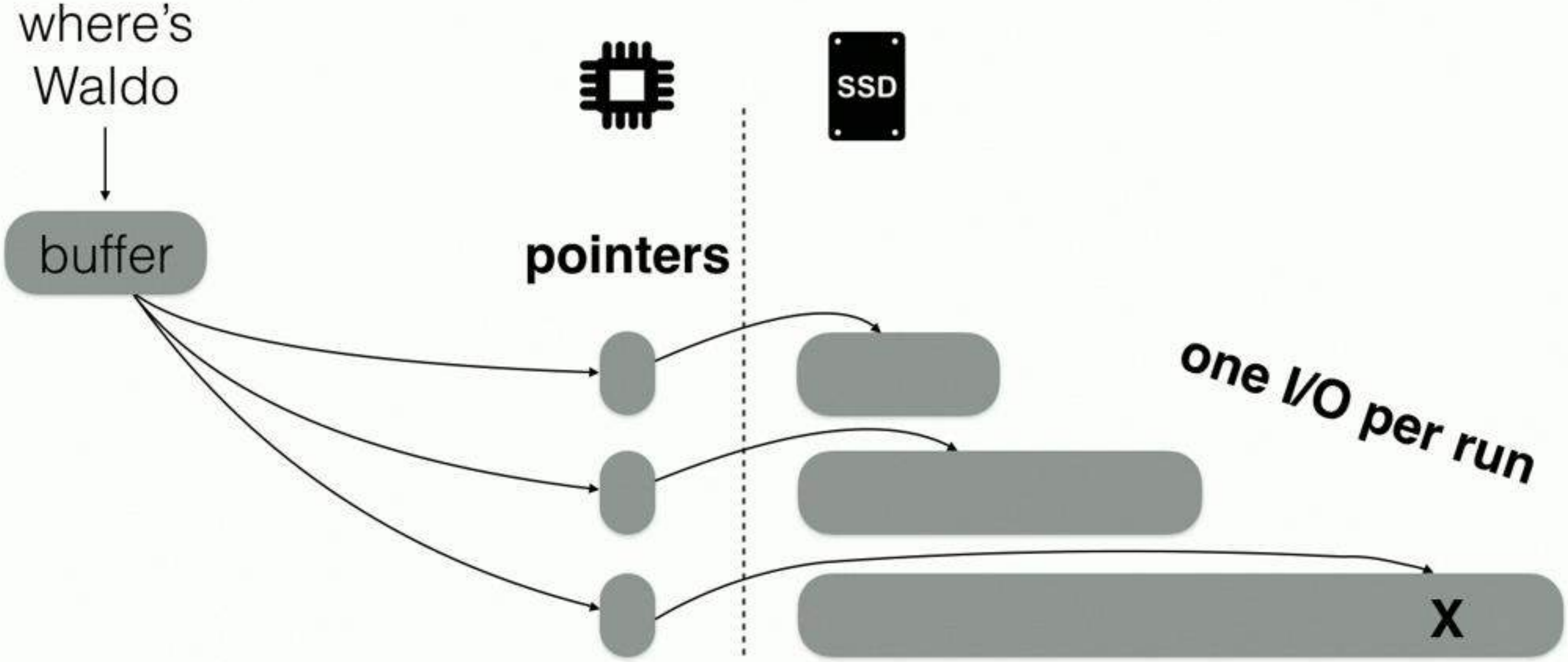

SSD
buffer
sort & flush



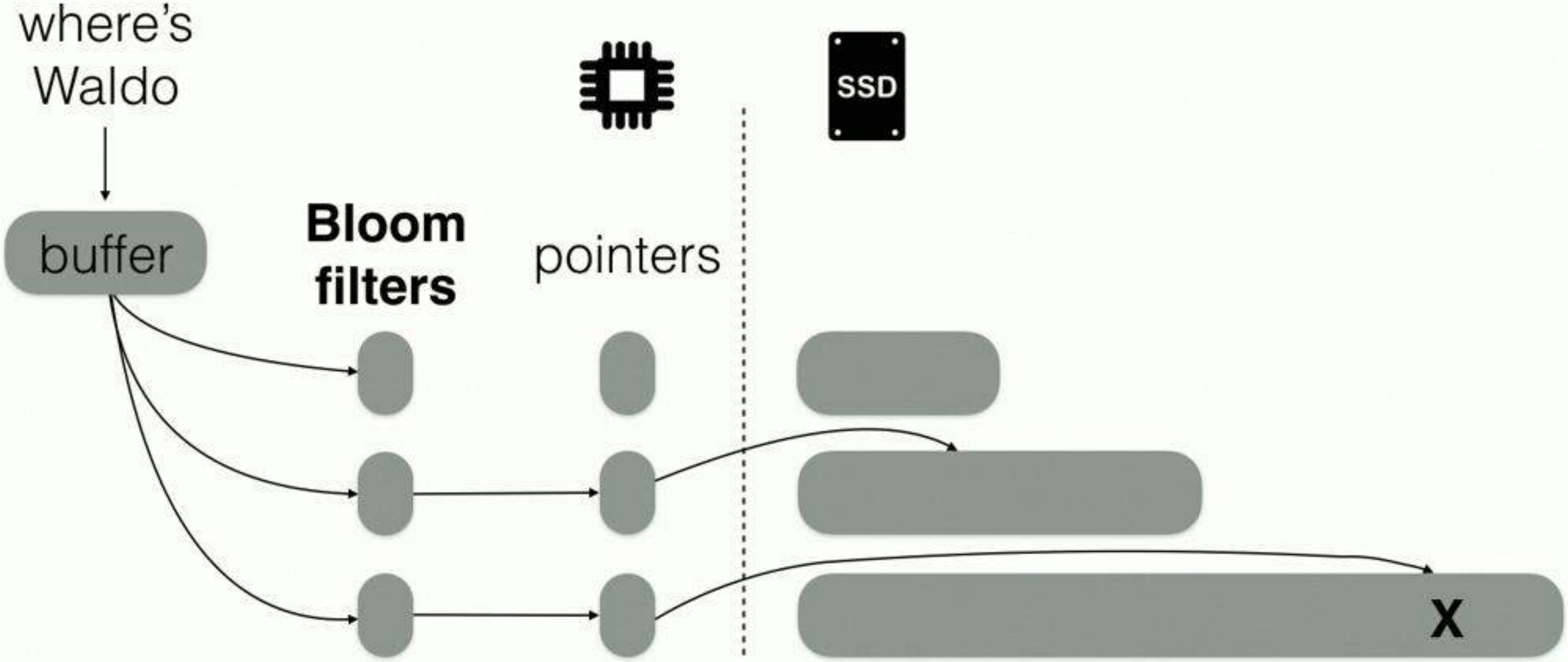
SSD
buffer
sort & flush
...
sorted runs


buffer
SSD
exponentially increasing capacities
level 1
level 2
level 3

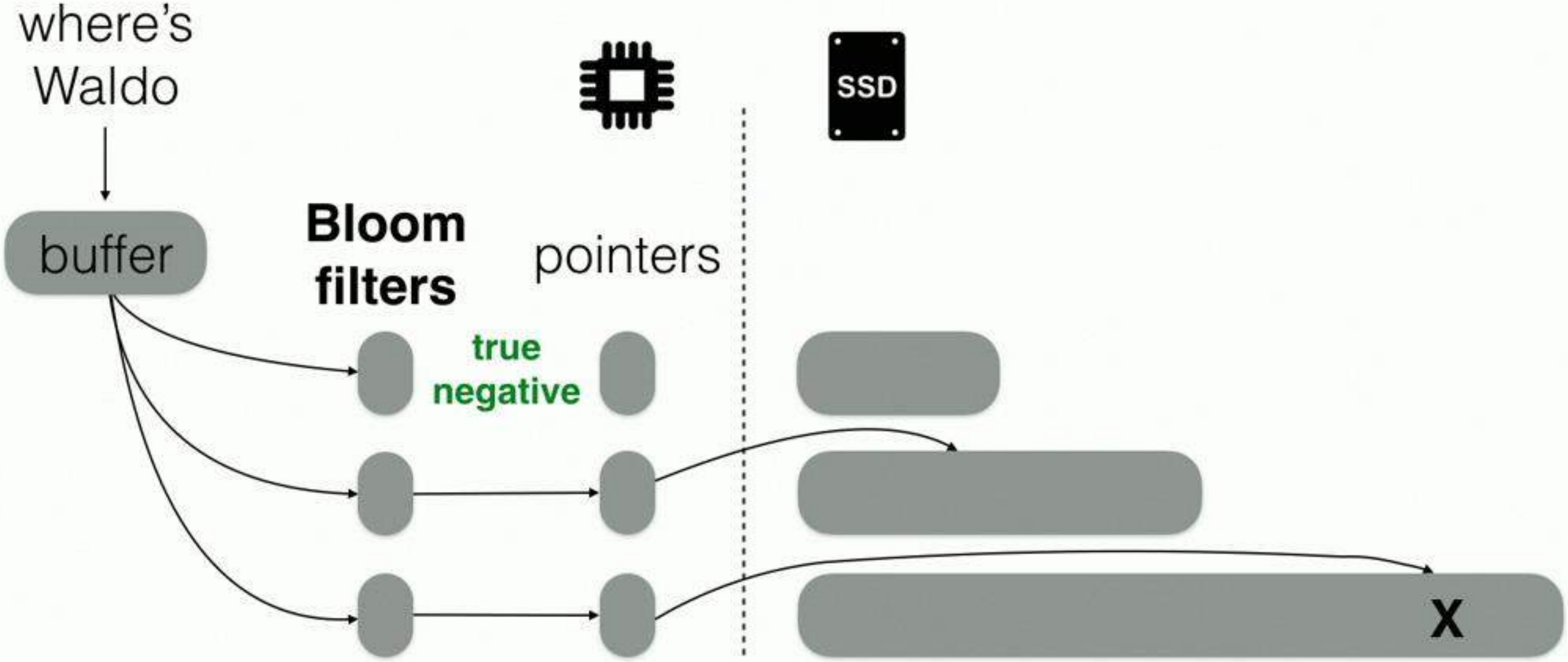

find X
SSD
buffer
binary searching
X

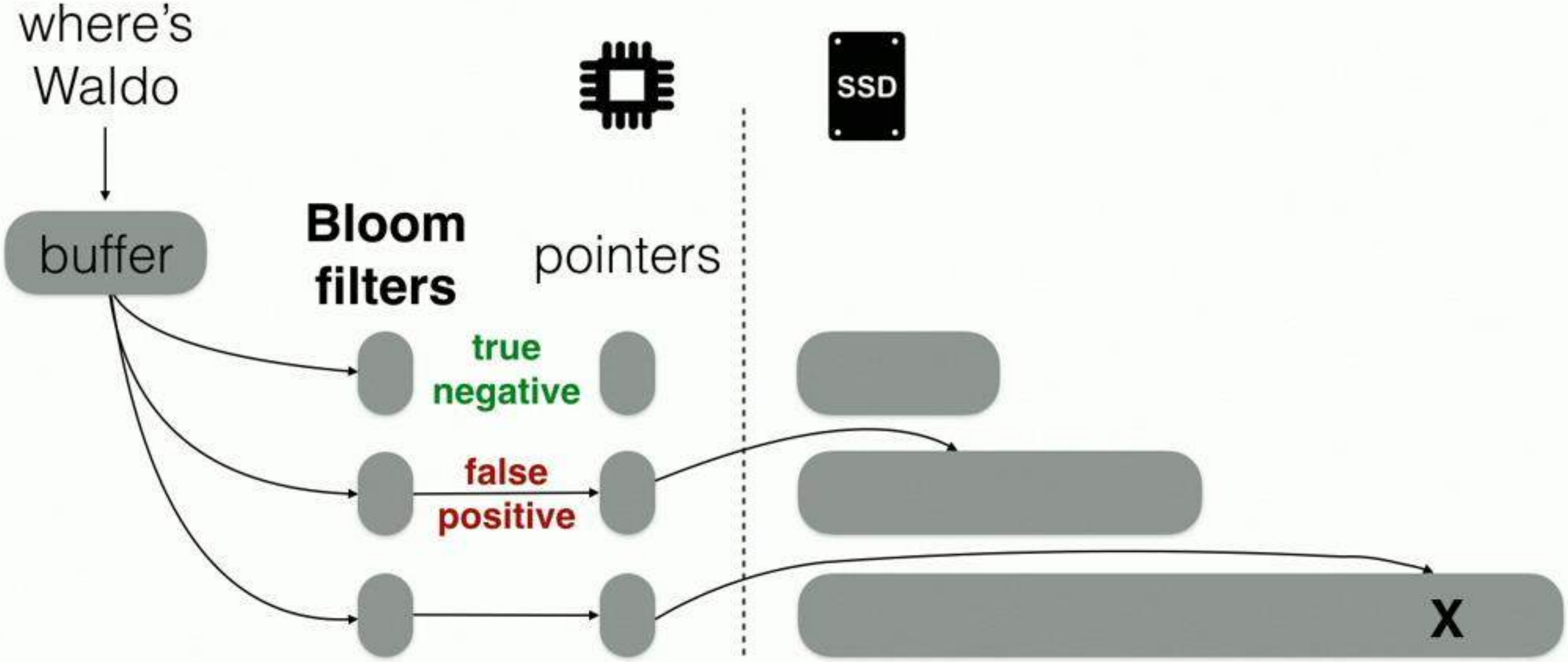


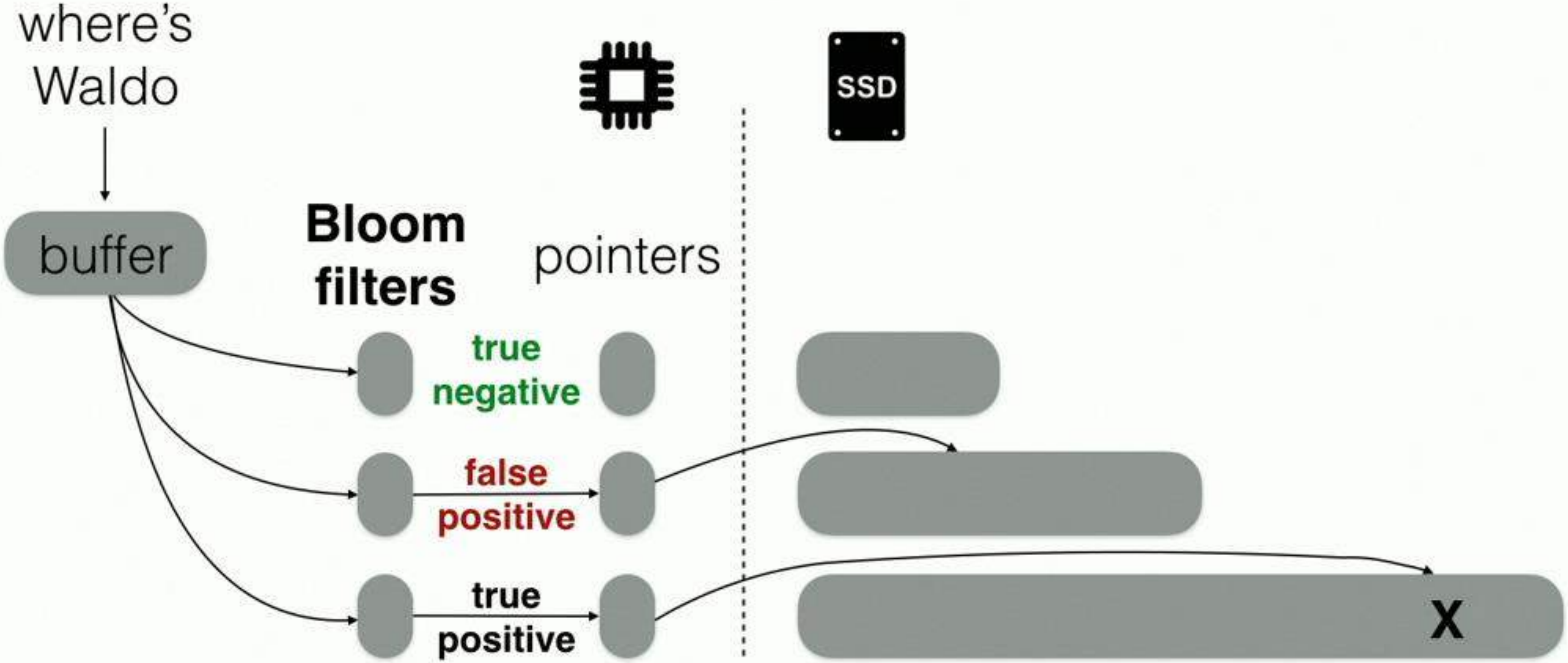
DASlab @ Harvard SEAS



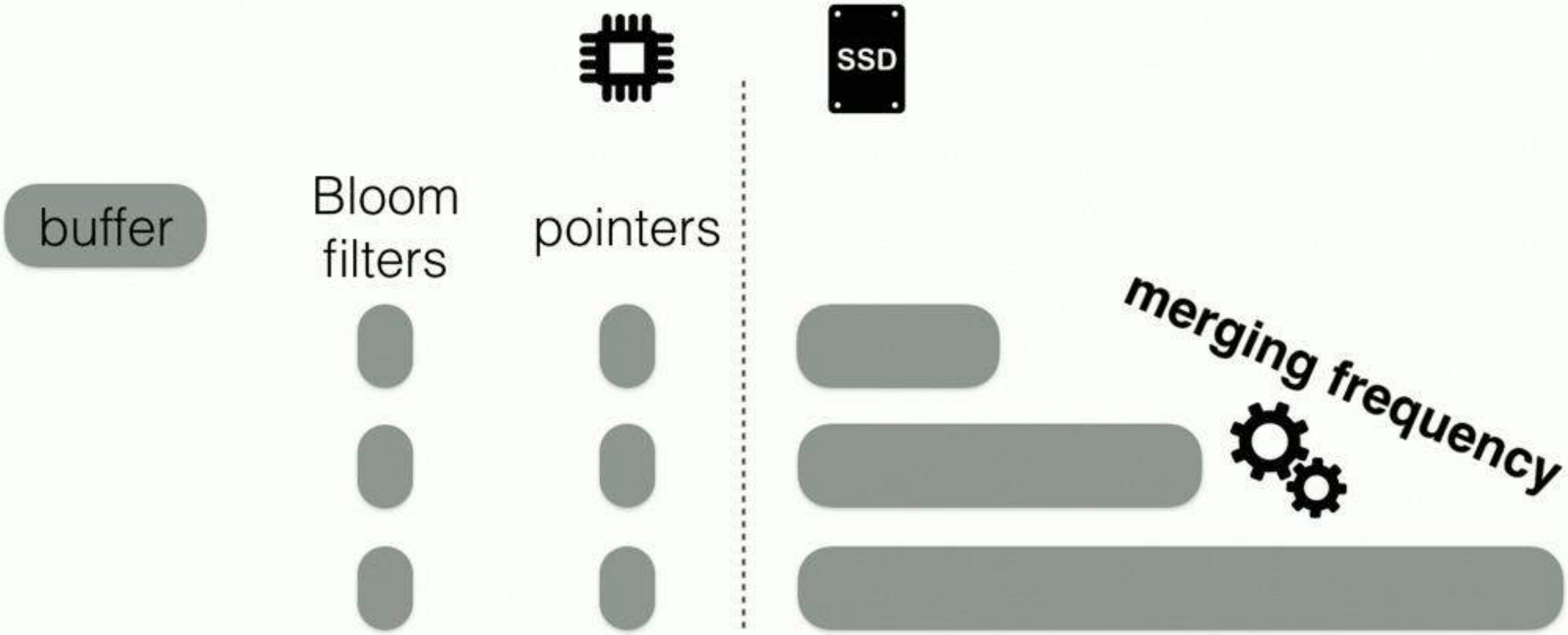
DASlab
@ Harvard SEAS


where's
Waldo
buffer
Bloom
filters
pointers
SSD
true
negative
X


where's Waldo
buffer
Bloom filters
pointers
SSD
true negative
false positive
X



DASlab @ Harvard SEAS



DASlab
@ Harvard SEAS

merging

writes

reads

merging ↑

writes ↑

↓ reads

merging

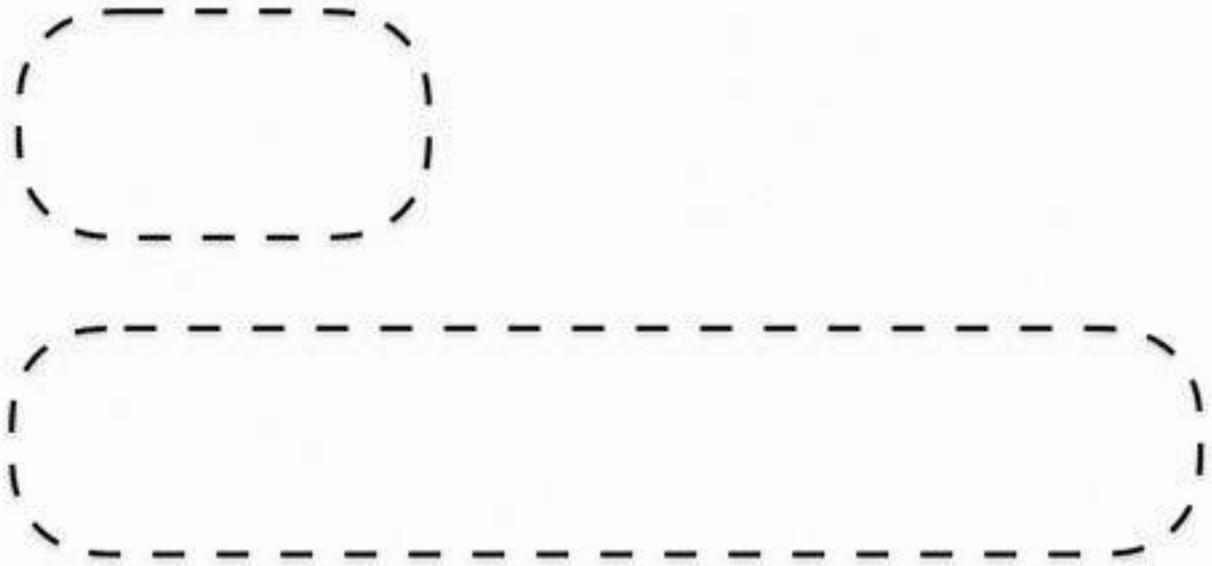
Tiering
write-optimized

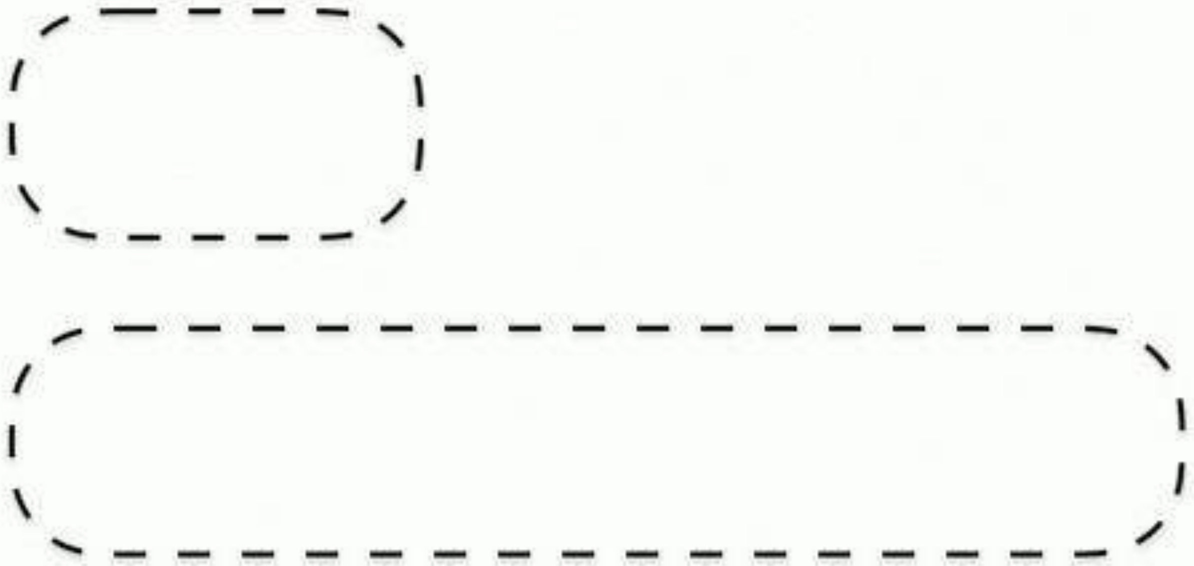
cassandra

Leveling
read-optimized

RocksDB

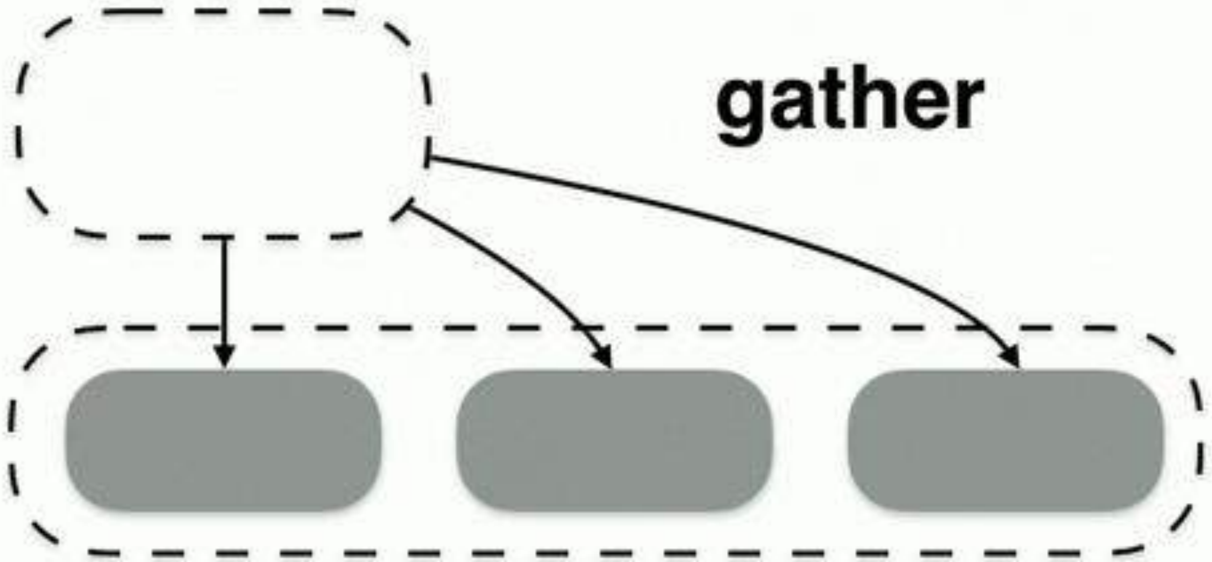
Tiering	Leveling
write-optimized	read-optimized





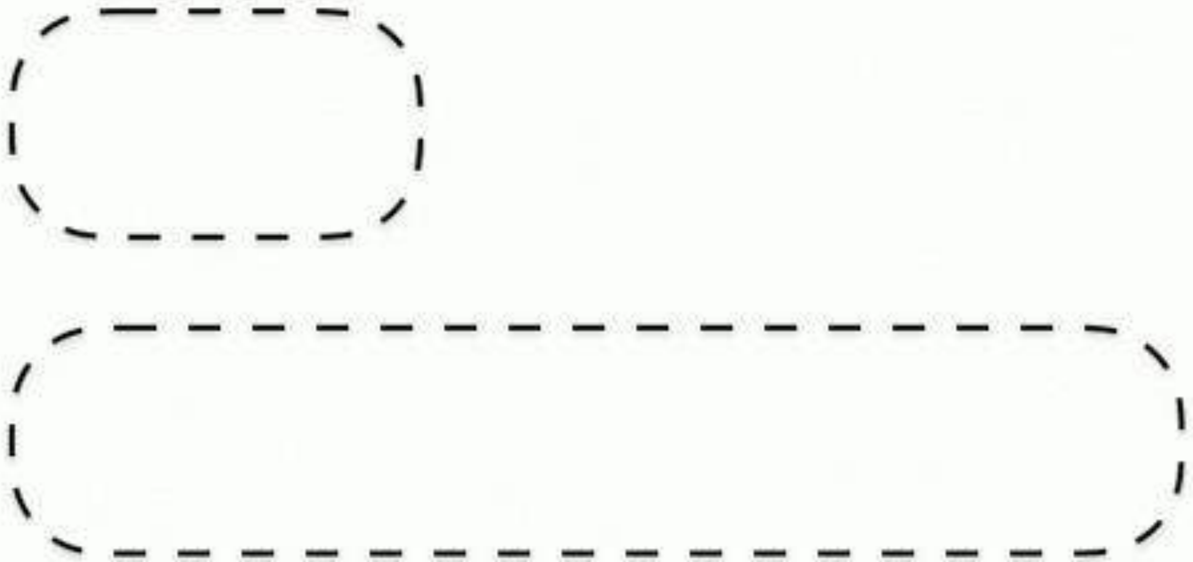
Tiering

write-optimized



Leveling

read-optimized

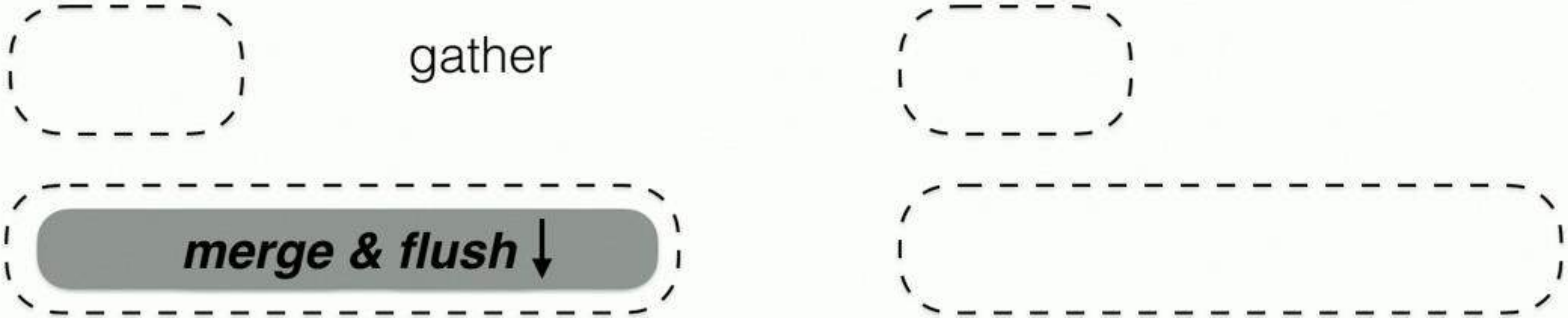


Tiering

write-optimized

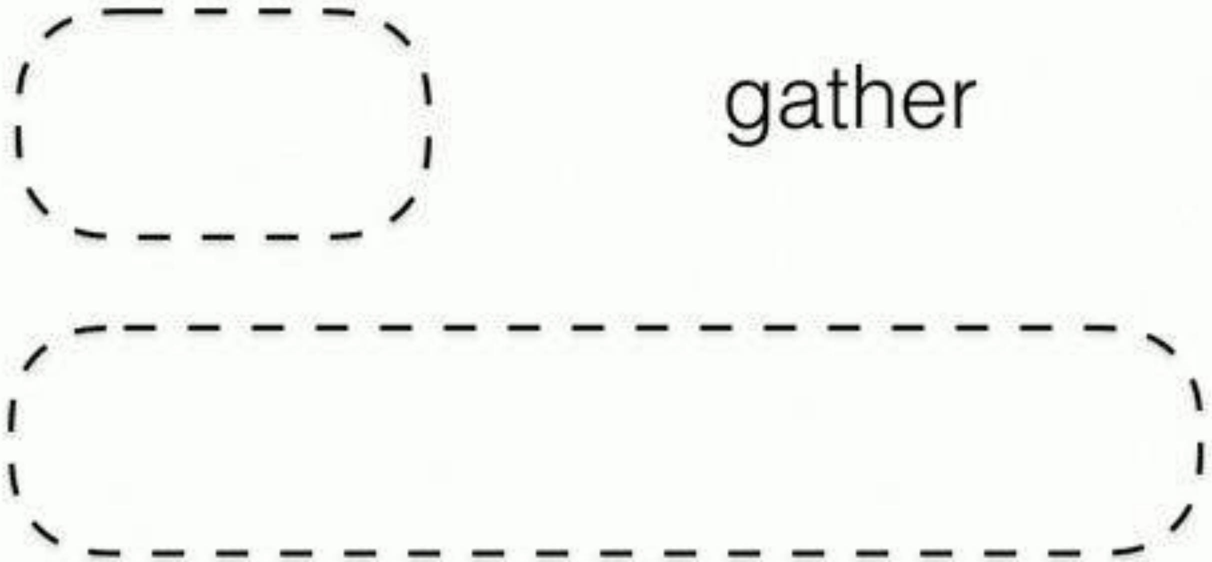
Leveling

read-optimized



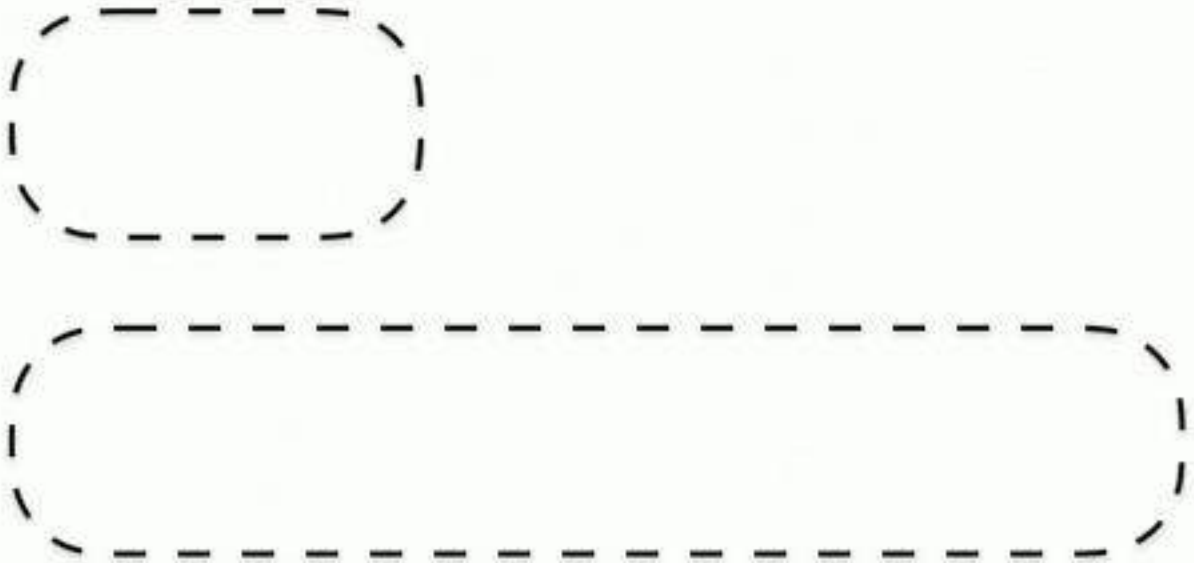
Tiering

write-optimized



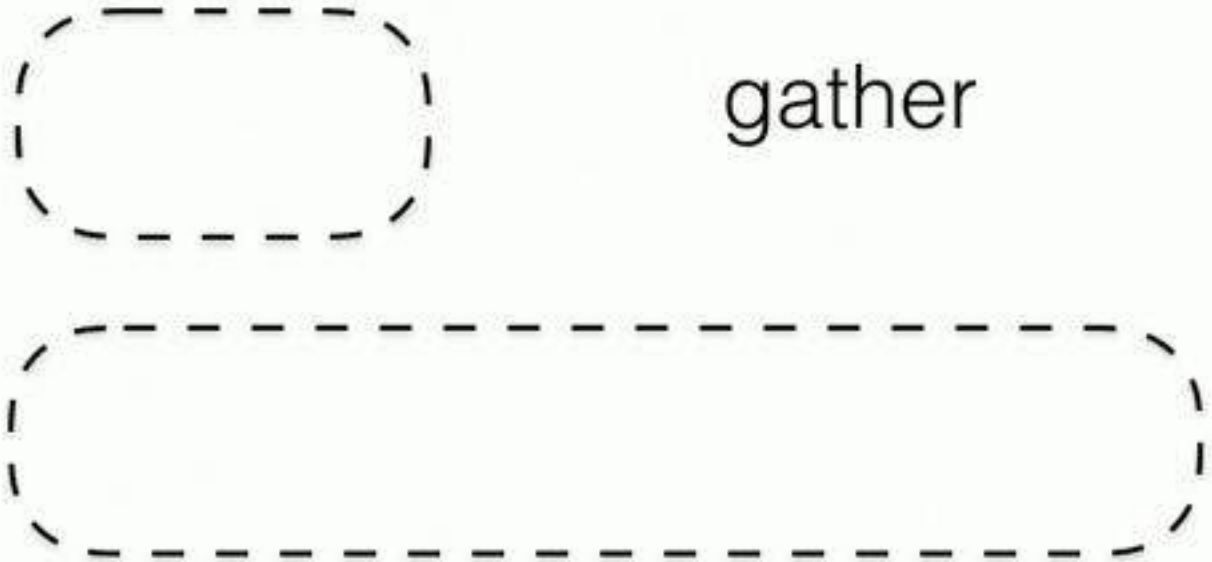
Leveling

read-optimized



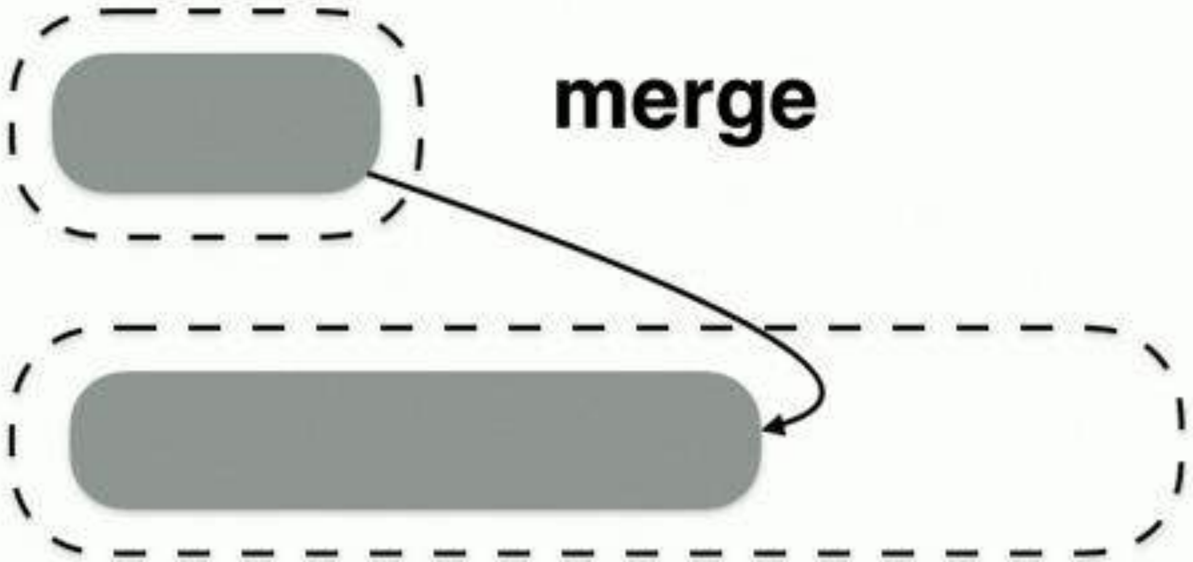
Tiering

write-optimized



Leveling

read-optimized

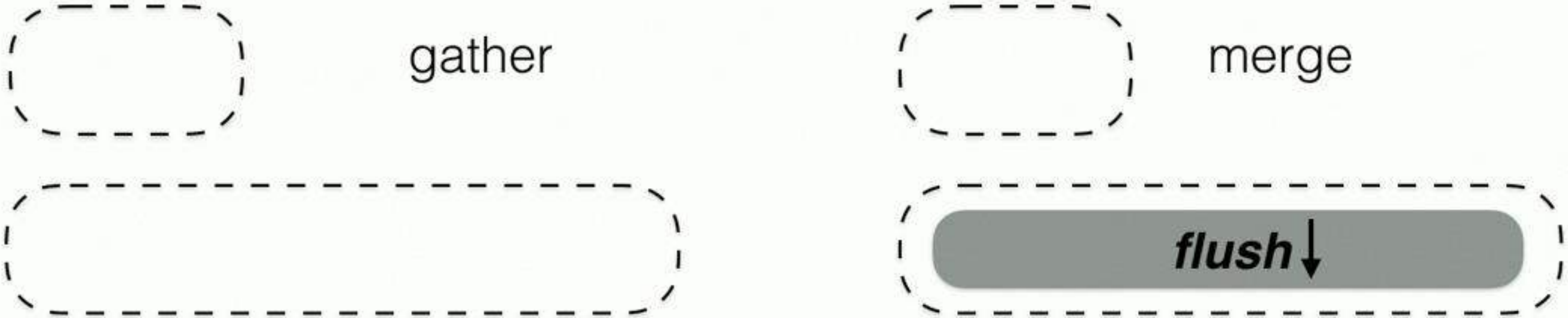


Tiering

write-optimized

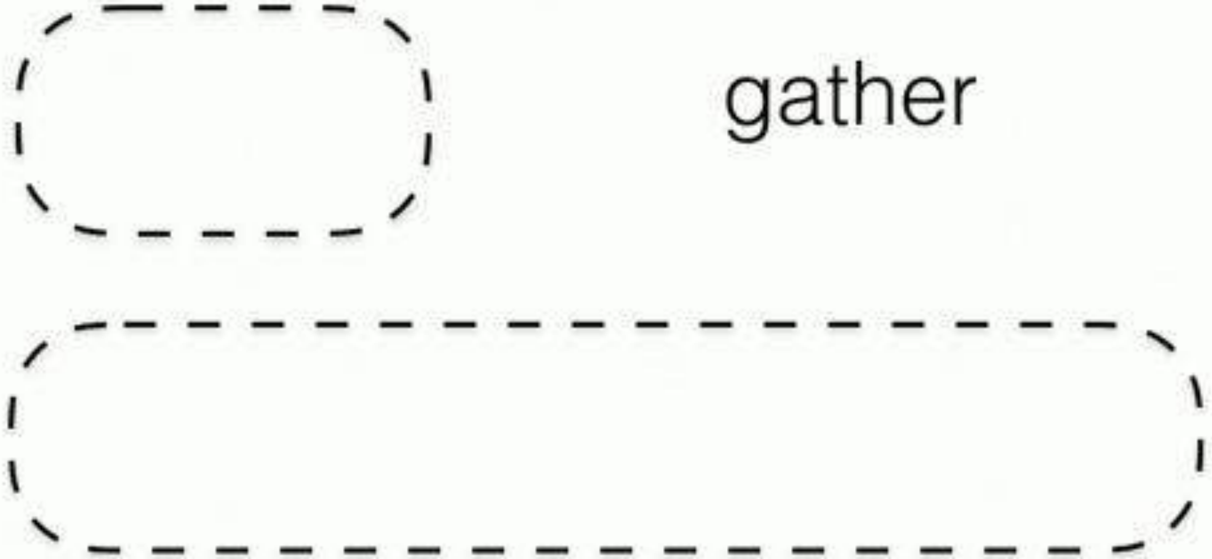
Leveling

read-optimized



Tiering

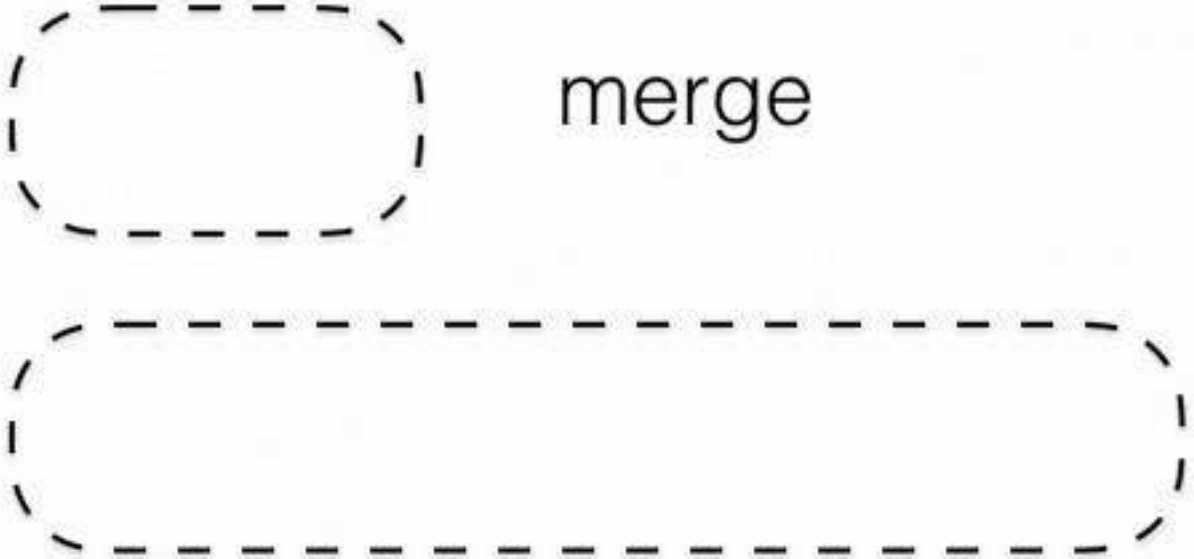
write-optimized



gather

Leveling

read-optimized



merge

Tiering

write-optimized

$\log_R(N)$

Leveling

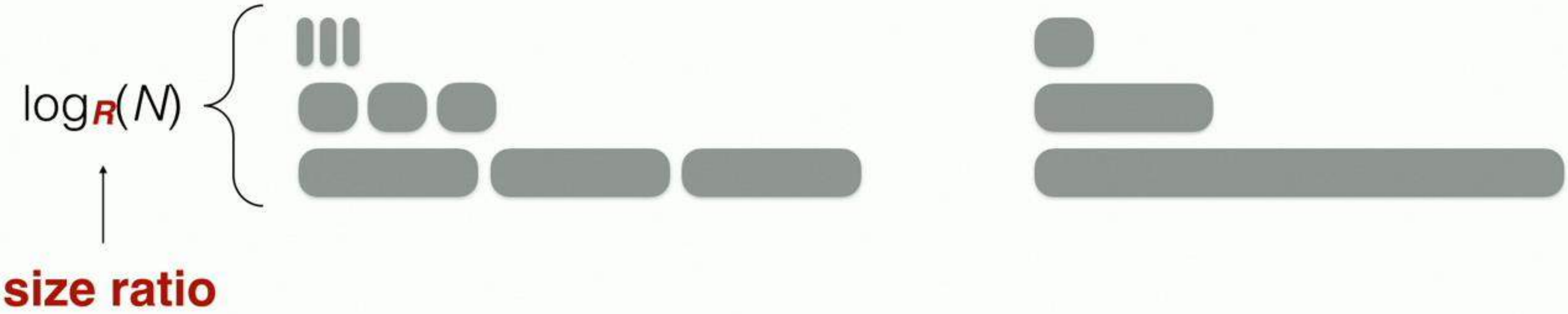
read-optimized

Tiering

write-optimized

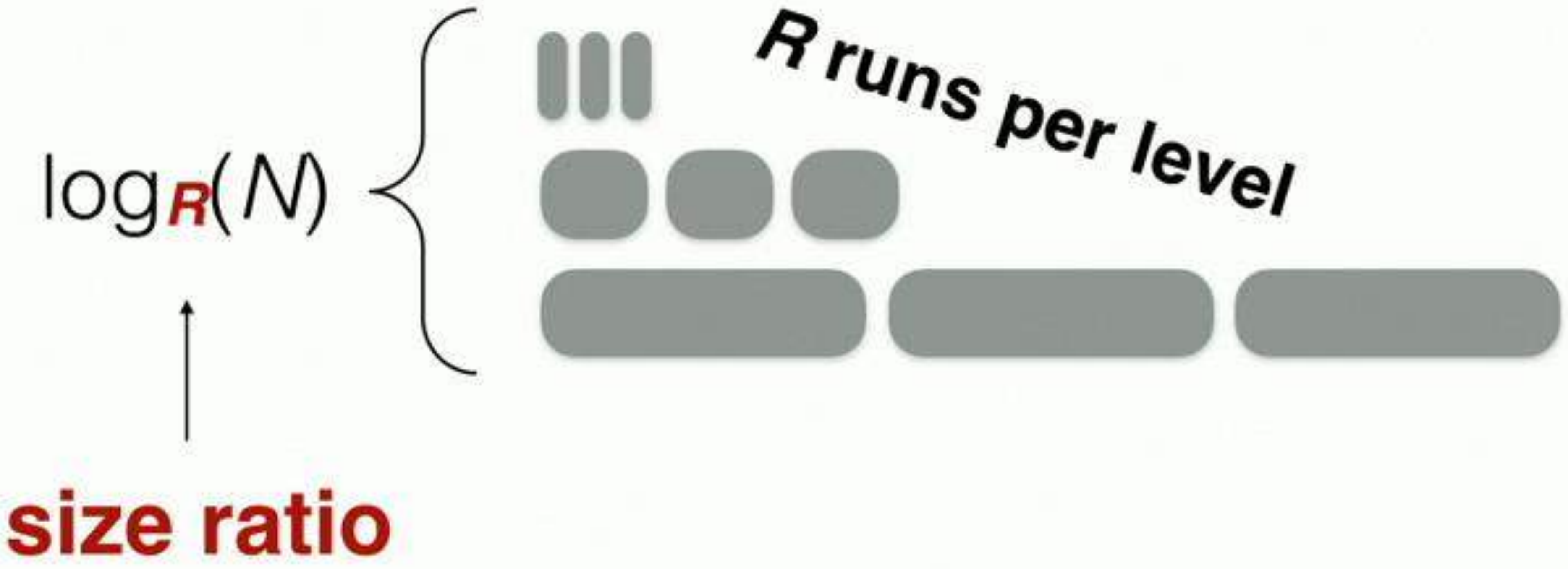
Leveling

read-optimized



Tiering

write-optimized



Leveling

read-optimized

Tiering

write-optimized

Leveling

read-optimized

size ratio *R*

Tiering

write-optimized

Leveling

read-optimized

size ratio *R*

Tiering

write-optimized

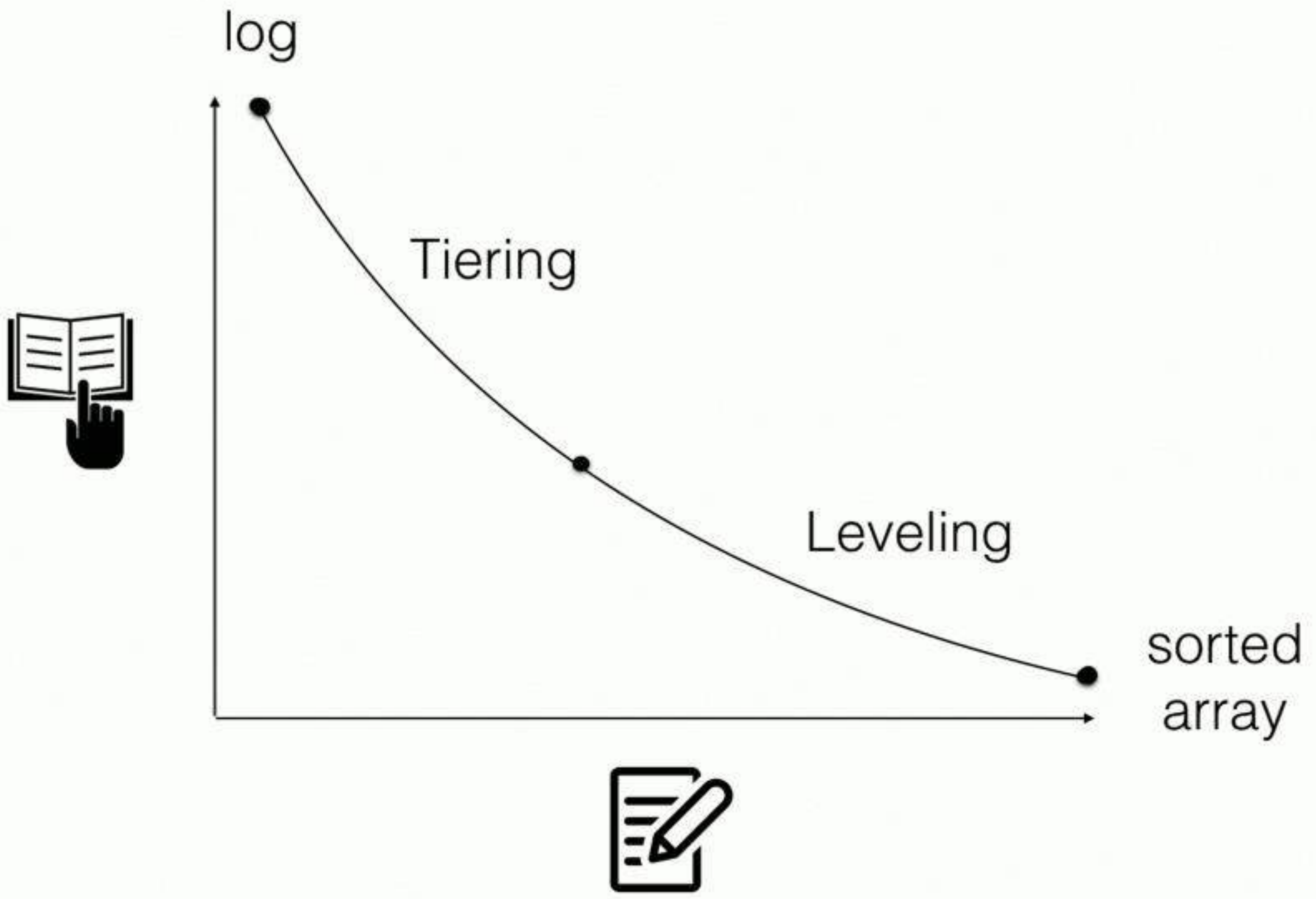
Leveling

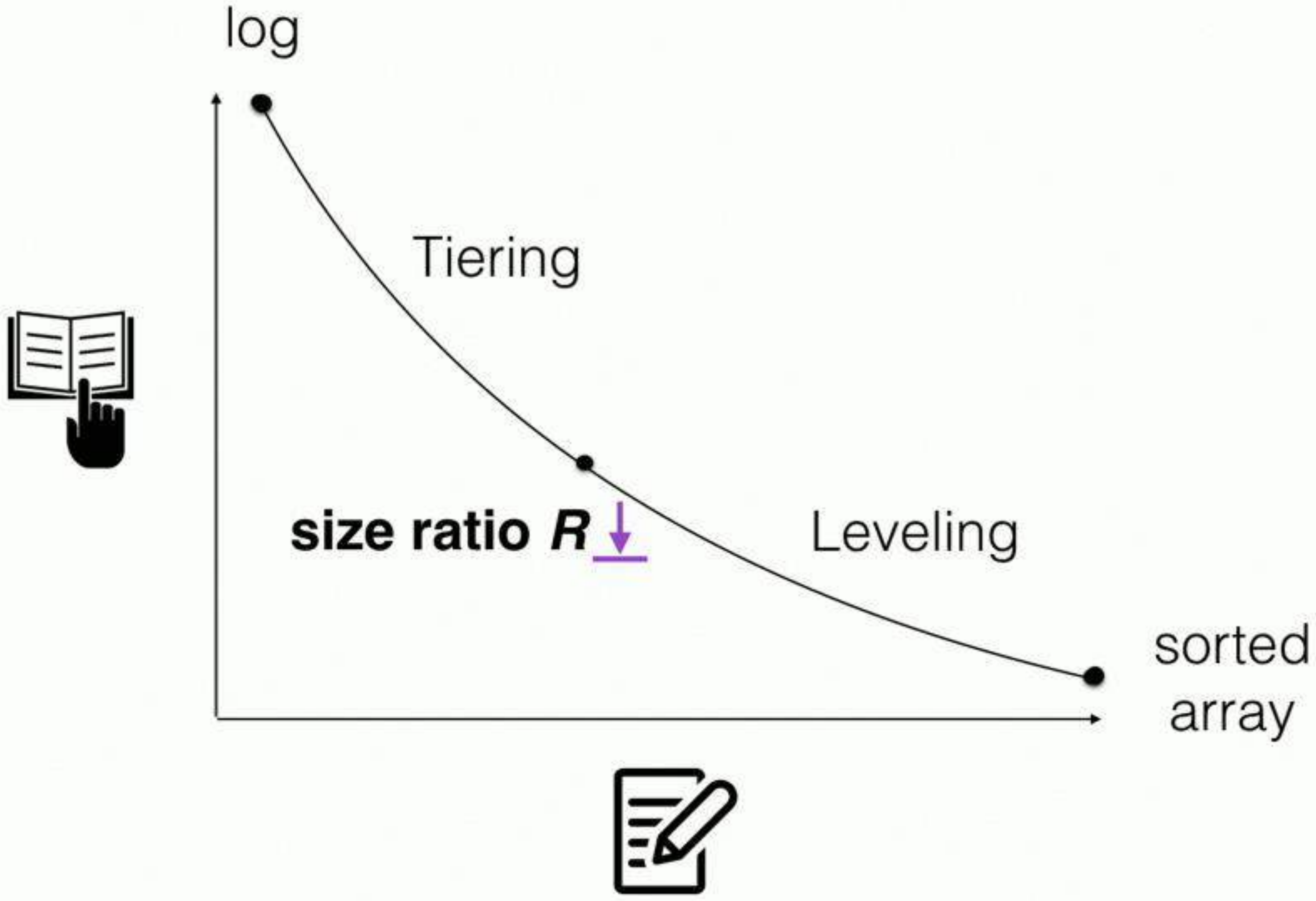
read-optimized

size ratio *R*

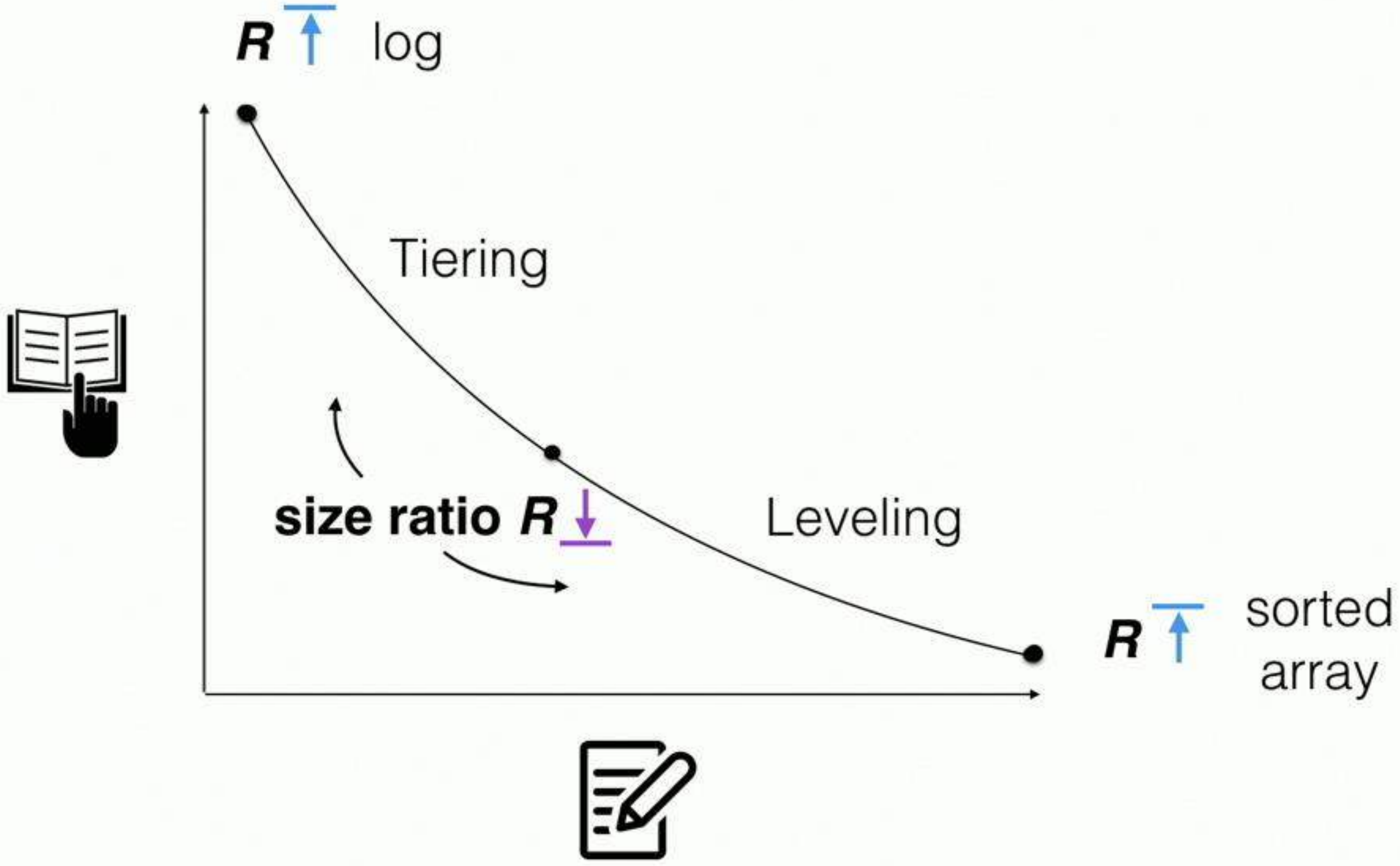
Tiering	Leveling
write-optimized	read-optimized
O(*N*) runs per level	1 run per level
log	**sorted array**

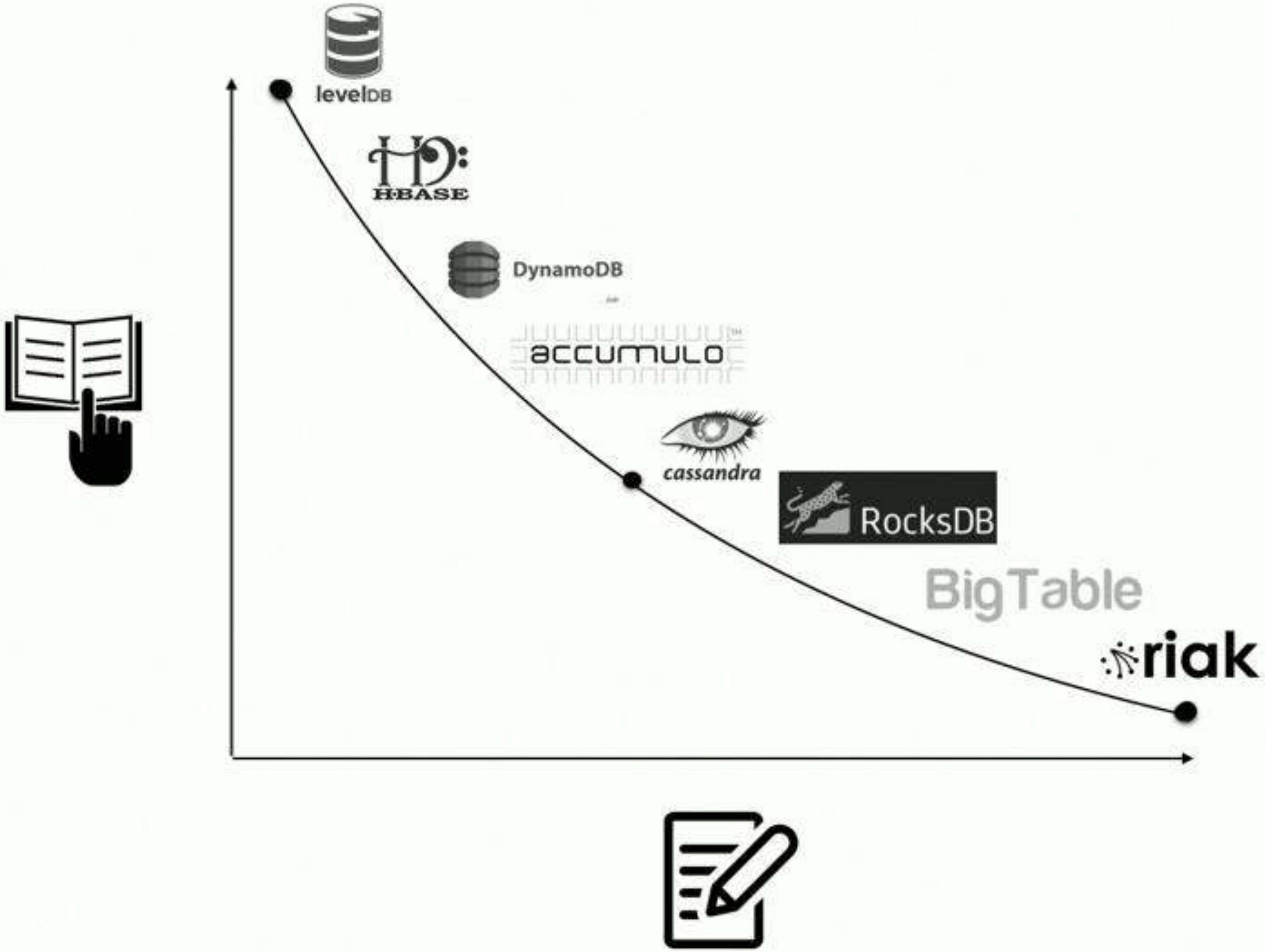
size ratio *R* ⇑


log
Tiering
Leveling
sorted
array

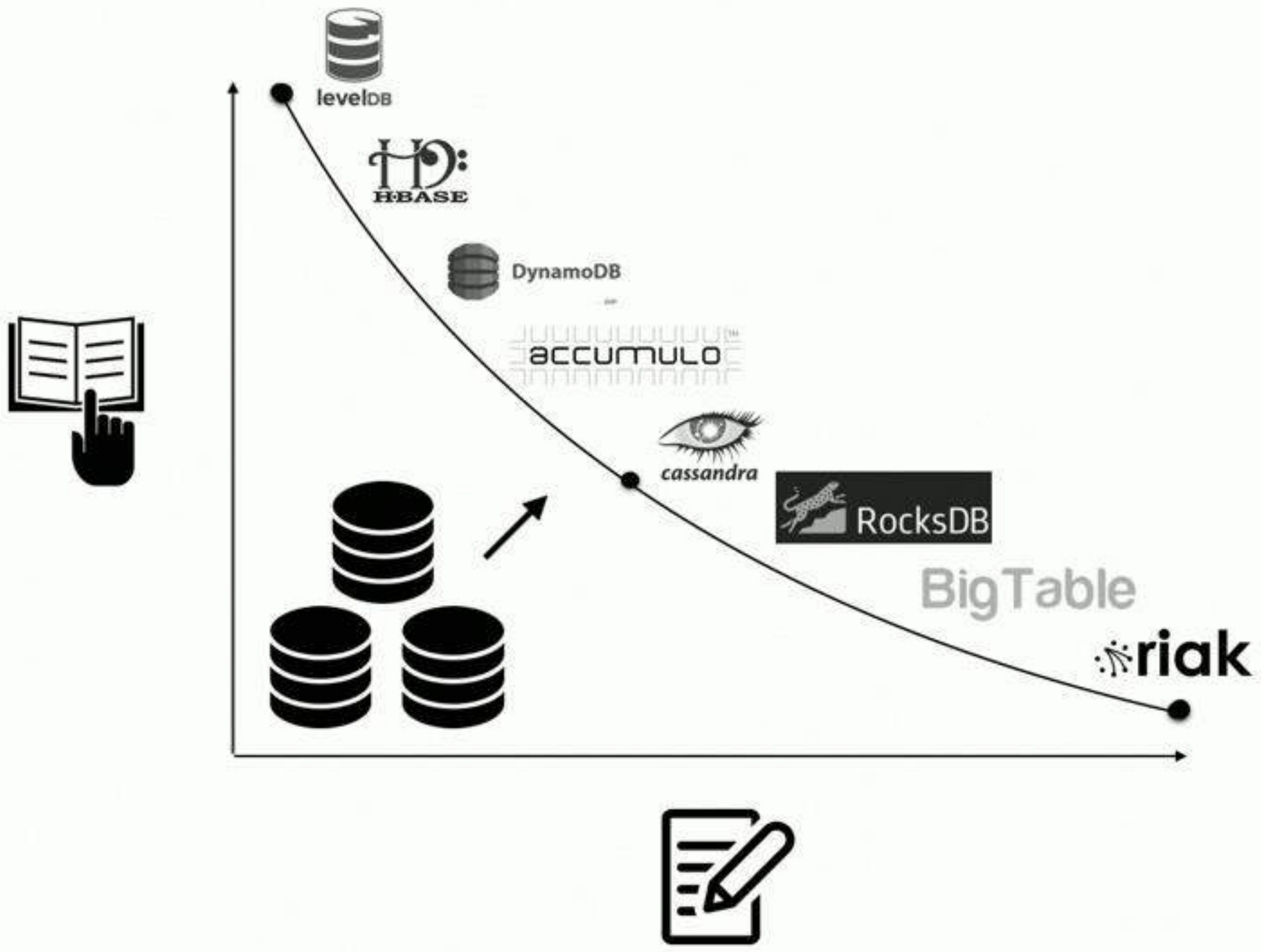


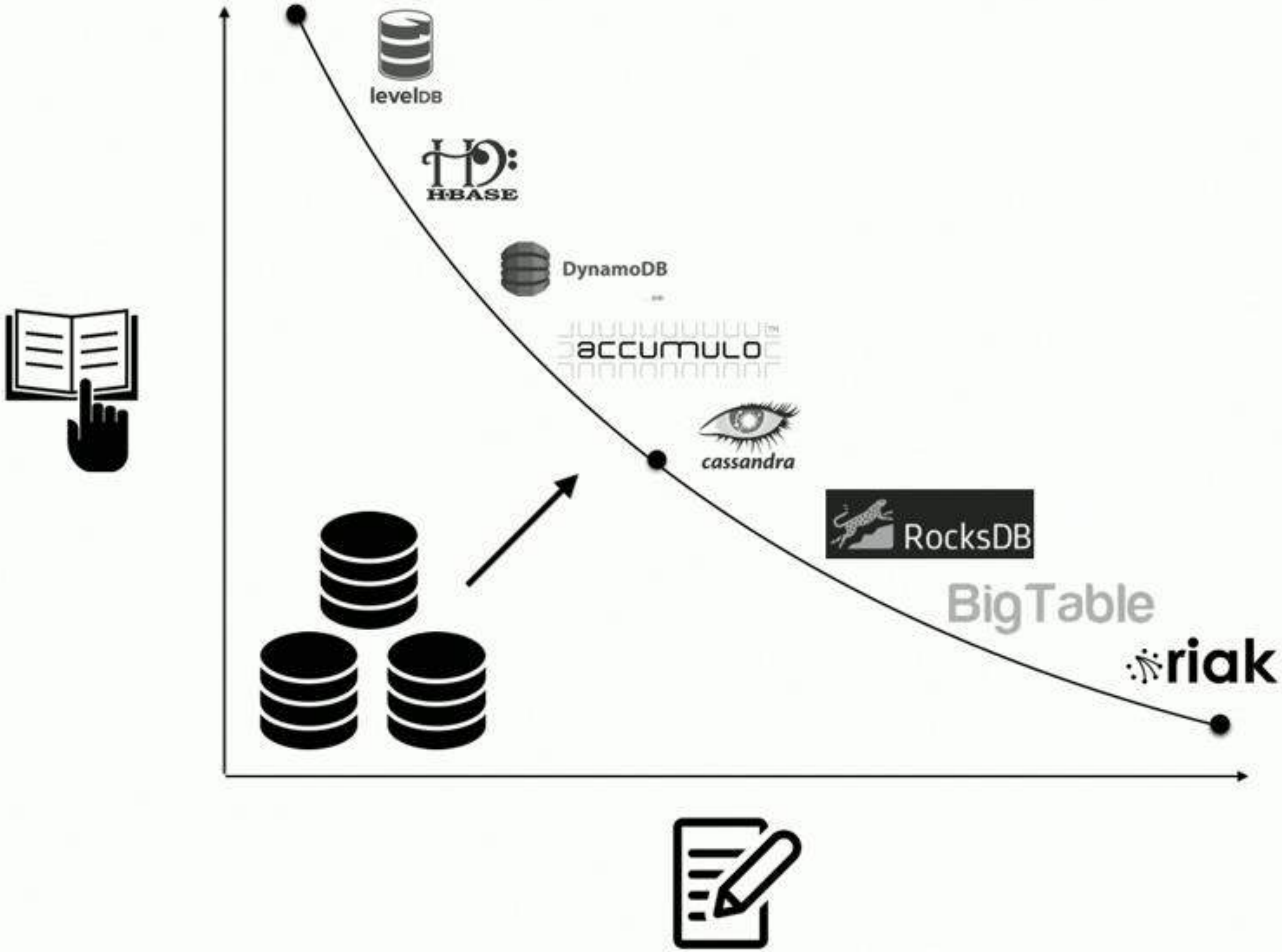
DASlab
@ Harvard SEAS

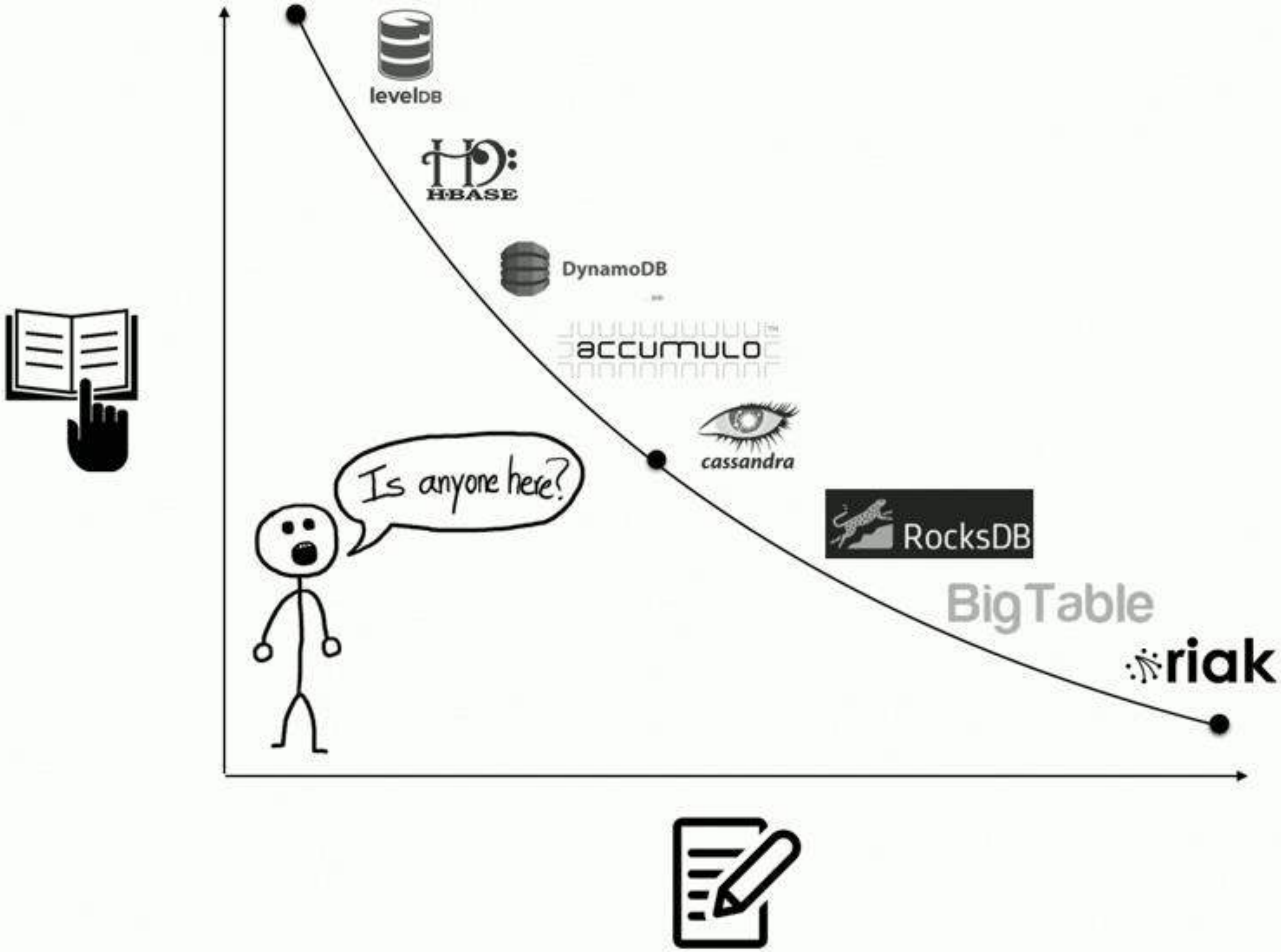

R log
Tiering
size ratio R
Leveling
R sorted array

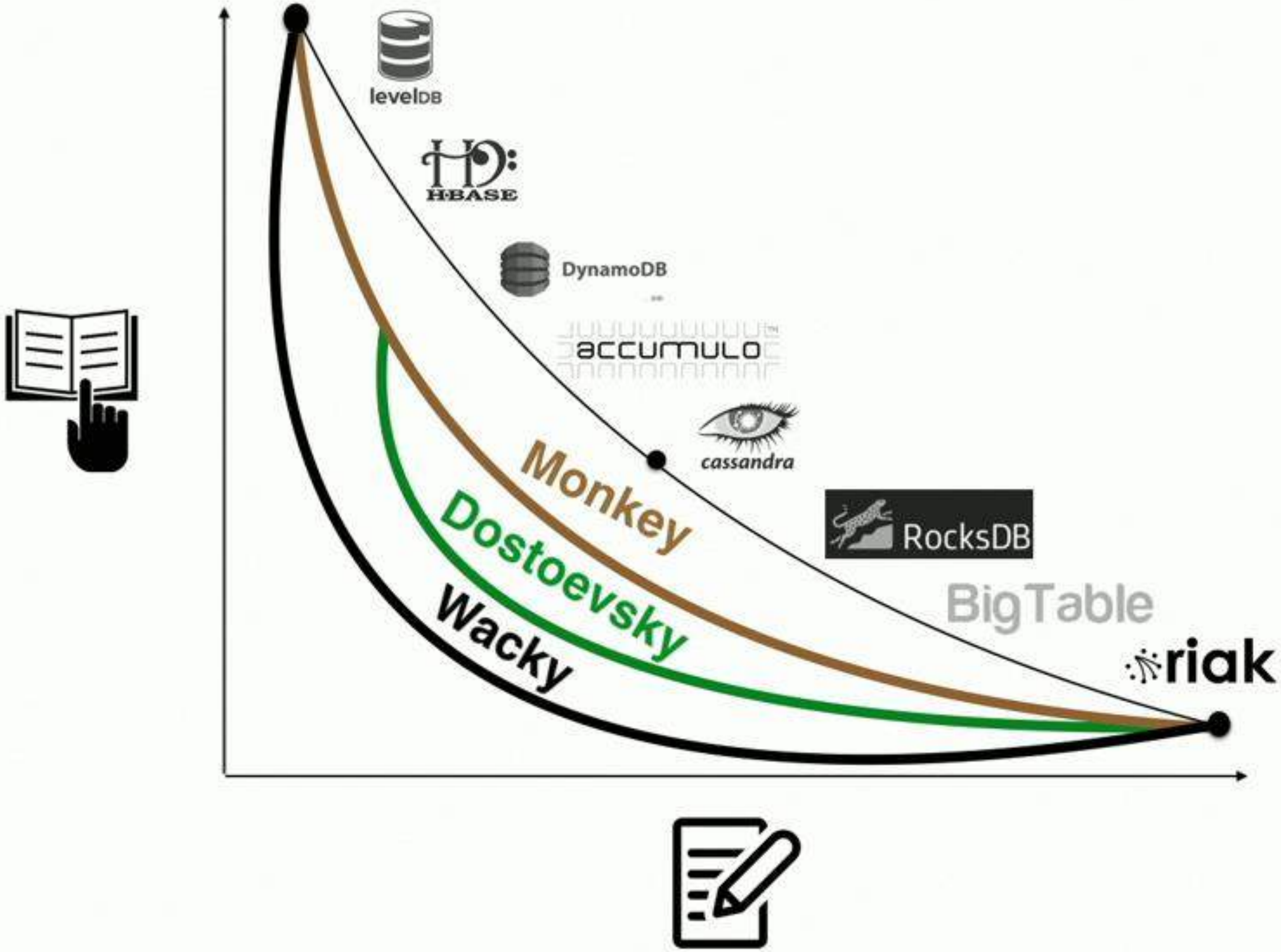


DASlab
@ Harvard SEAS


levelDB
HBASE
DynamoDB
accumulo
cassandra
RocksDB
BigTable
riak


levelDB
HBASE
DynamoDB
accumulo
cassandra
RocksDB
BigTable
riak


levelDB
HBASE
DynamoDB
accumulo
cassandra
RocksDB
BigTable
riak
Is anyone here?



DASlab @ Harvard SEAS

Monkey: **O**ptimal **N**avigable **Key**-Value Store SIGMOD17

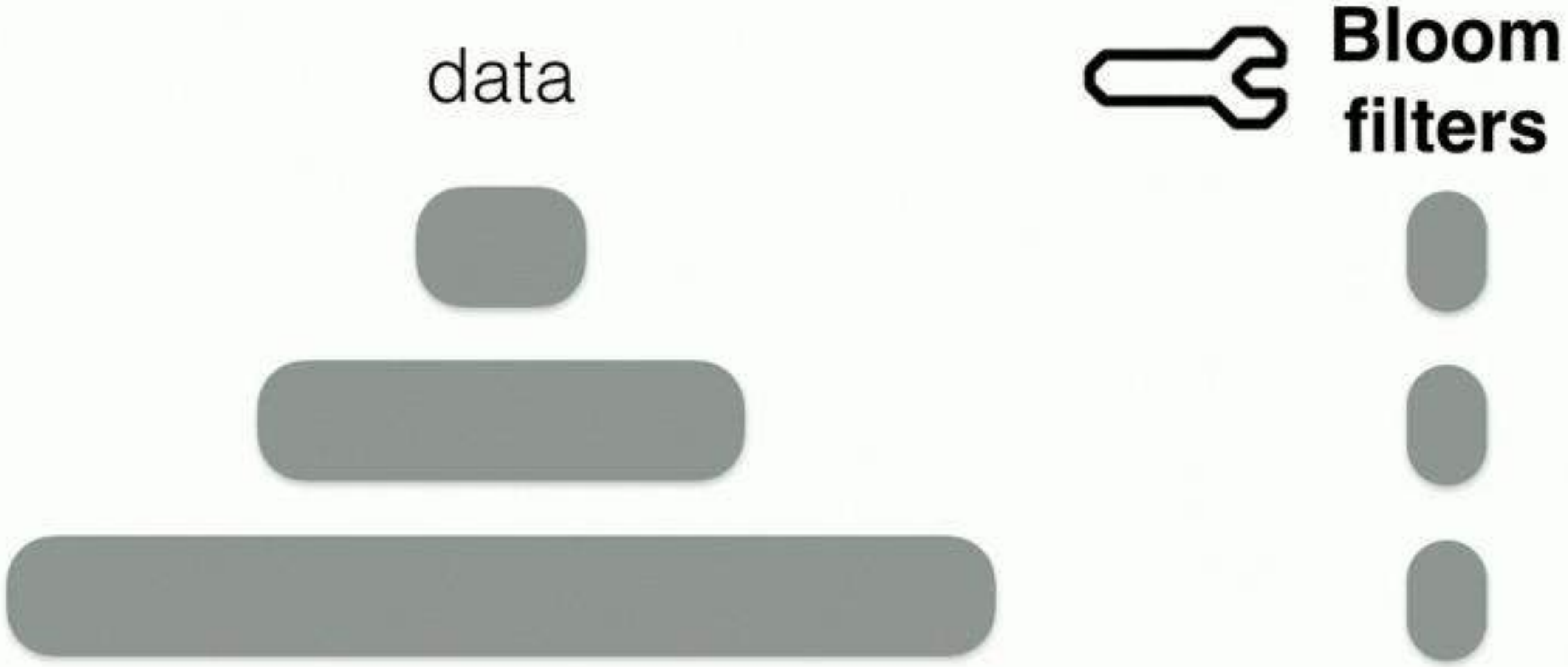
DASlab @ Harvard SEAS

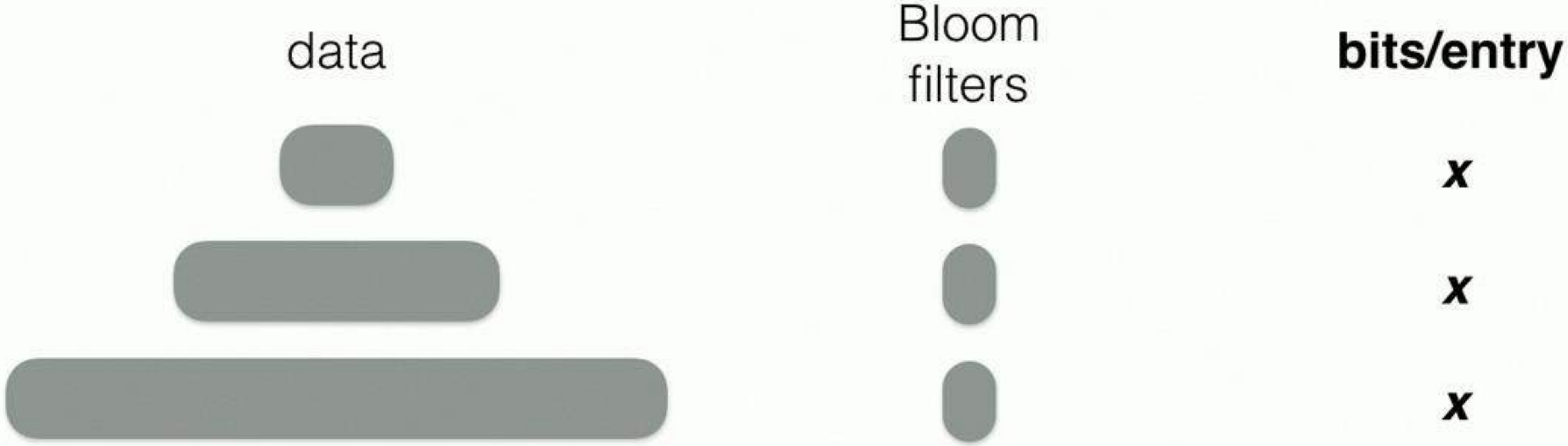
Monkey: **O**ptimal **N**avigable **Key**-Value Store SIGMOD17

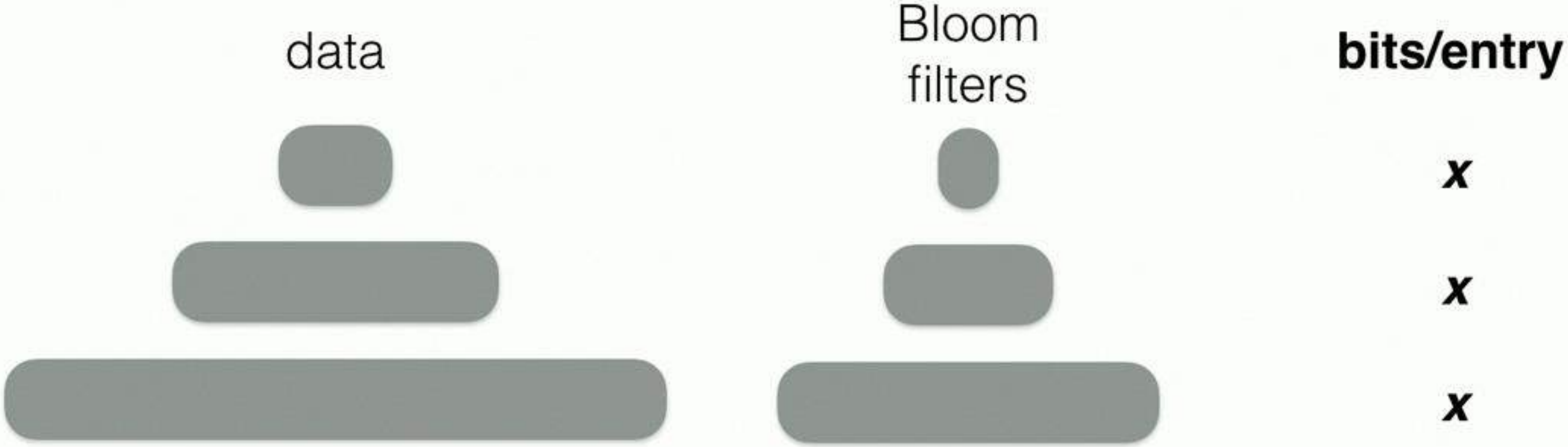
Niv Dayan
Manos Athanassoulis
Stratos Idreos

DASlab @ Harvard SEAS

Monkey: **O**ptimal **N**avigable **Key**-Value Store SIGMOD17




data
Bloom filters
bits/entry
x
x
x


data
Bloom filters
bits/entry
x
x
x

data

Bloom filters

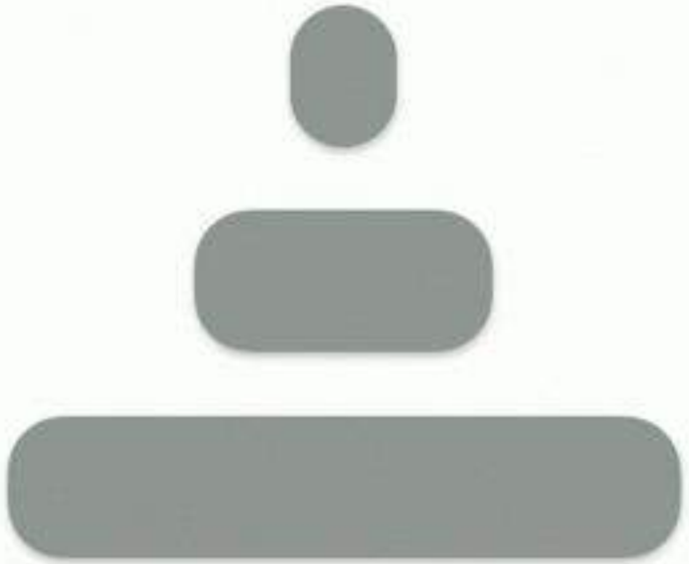
false positive rate

$O(e^{-x})$

$O(e^{-x})$

$O(e^{-x})$

Bloom filters



false positive rate

$$\left.\begin{array}{c} O(e^{-x}) \\ O(e^{-x}) \\ O(e^{-x}) \end{array}\right\} = O(\boldsymbol{e^{-x}} \cdot \mathbf{log}_{\boldsymbol{R}}(\boldsymbol{N}))$$

Bloom filters	false positive rate
	$O(e^{-x})$
	$O(e^{-x})$
	$O(e^{-x})$

$$\left.\begin{array}{c} O(e^{-x}) \\ O(e^{-x}) \\ O(e^{-x}) \end{array}\right\} = O(\boldsymbol{e^{-x}} \cdot \mathbf{log}_{\boldsymbol{R}}(\boldsymbol{N}))$$

Bloom filters

most memory

false positive rate

$O(e^{-x})$

$O(e^{-x})$

$O(e^{-x})$

Bloom filters

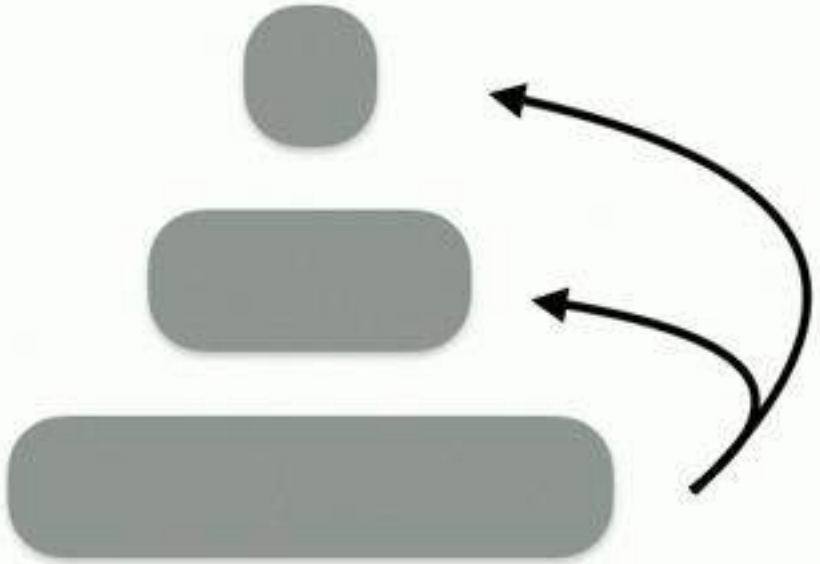
false positive rate

$O(e^{-x})$

$O(e^{-x})$

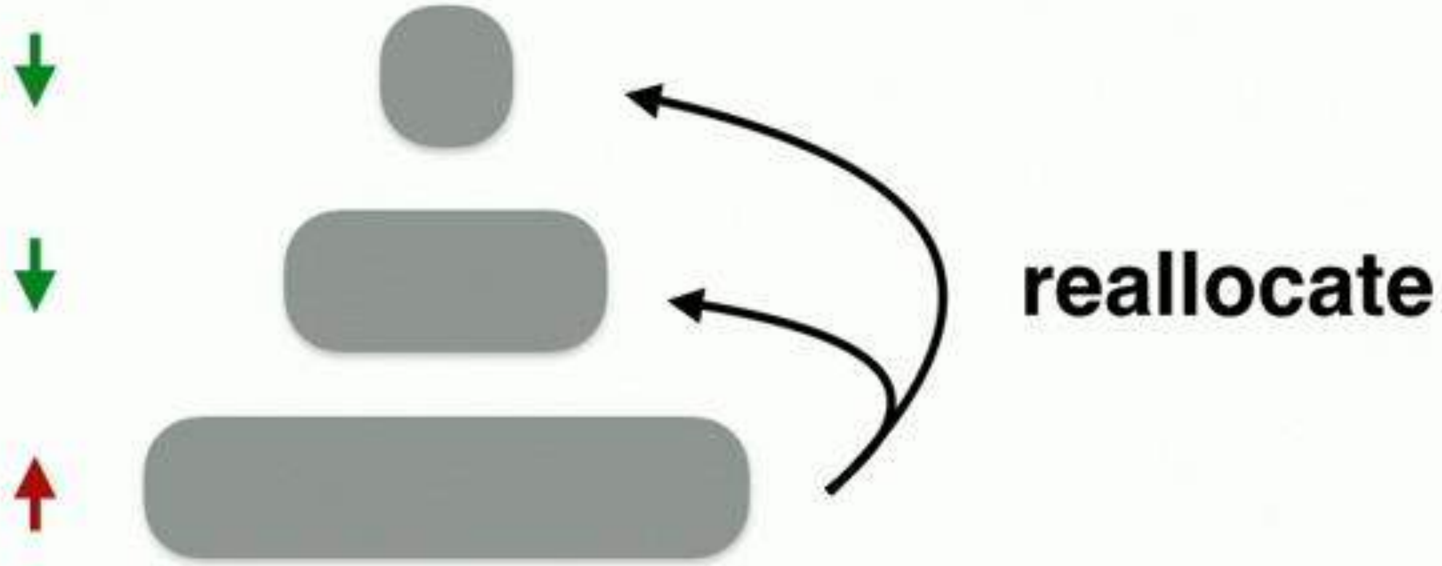
most memory $O(e^{-x})$

saves at most 1 I/O!



reallocate

same memory - fewer false positives



relax

false positive rates

$0 < p_0 < 1$

$0 < p_1 < 1$

$0 < p_2 < 1$

relax

false positive rates

$0 < p_0 < 1$

$0 < p_1 < 1$

$0 < p_2 < 1$

model

read cost $= f(\boldsymbol{p_0}, \boldsymbol{p_1} \ldots)$

memory footprint $= f(\boldsymbol{p_0}, \boldsymbol{p_1} \ldots)$

relax

false positive rates

$0 < p_0 < 1$

$0 < p_1 < 1$

$0 < p_2 < 1$

model

read cost $= \sum_{1}^{L} p_i$

memory footprint $= -\sum_{i}^{L} \frac{N}{T^{L-i}} \cdot \frac{\ln(p_i)}{\ln(2)^2}$

DASlab @ Harvard SEAS

relax

false positive rates

$0 < p_0 < 1$

$0 < p_1 < 1$

$0 < p_2 < 1$

model

read cost $= \sum_{1}^{L} p_i$

memory footprint $= -\sum_{i}^{L} \frac{N}{T^{L-i}} \cdot \frac{\ln(p_i)}{\ln(2)^2}$

optimize

in terms of p_0**,** p_1 ...

DASlab @ Harvard SEAS

false
positive rate

$p_0 \approx O(e^{-x}/\boldsymbol{R^2})$

$p_1 \approx O(e^{-x}/\boldsymbol{R^1})$

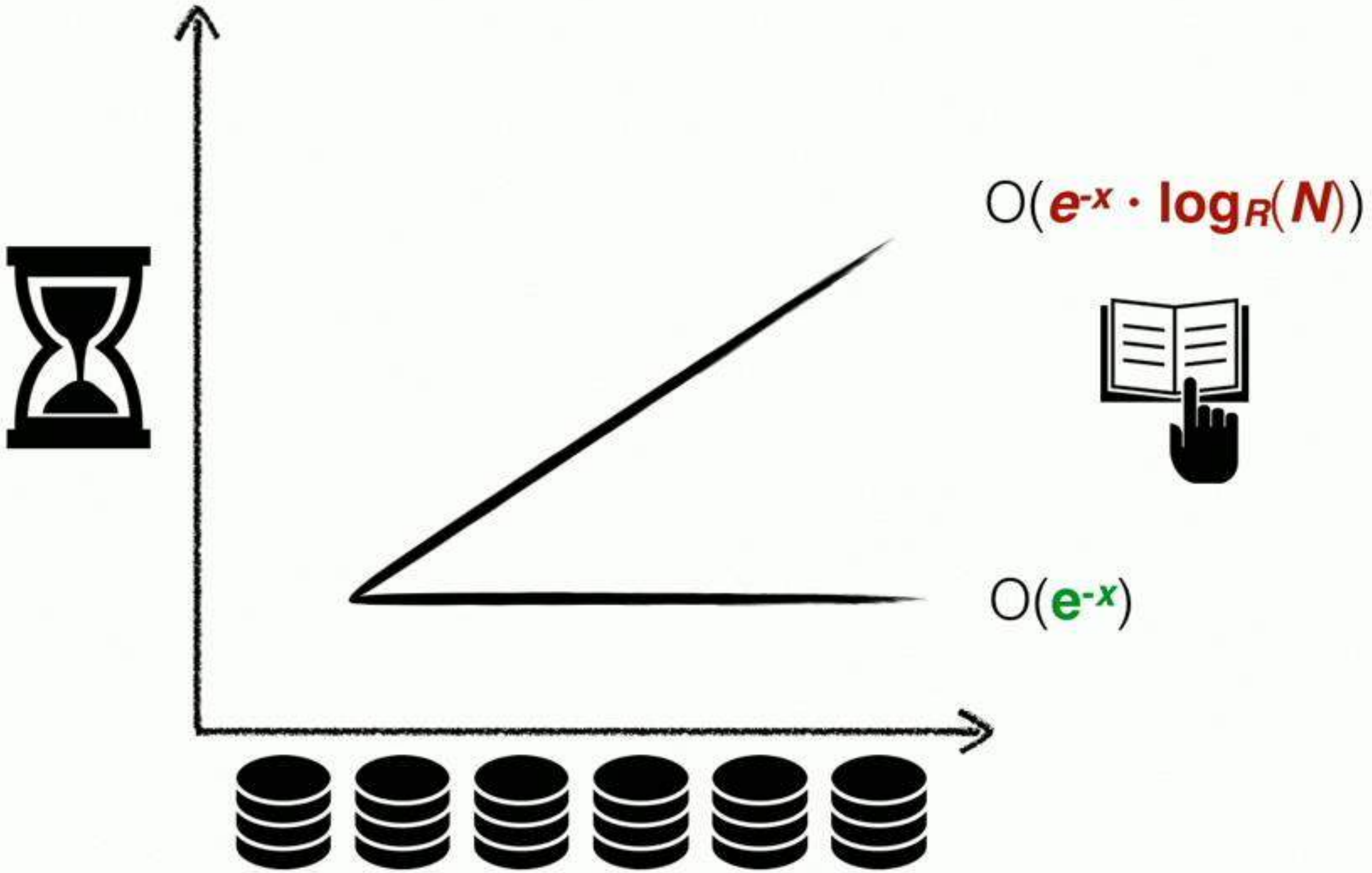
$p_2 \approx O(e^{-x}/\boldsymbol{R^0})$

false positive rate

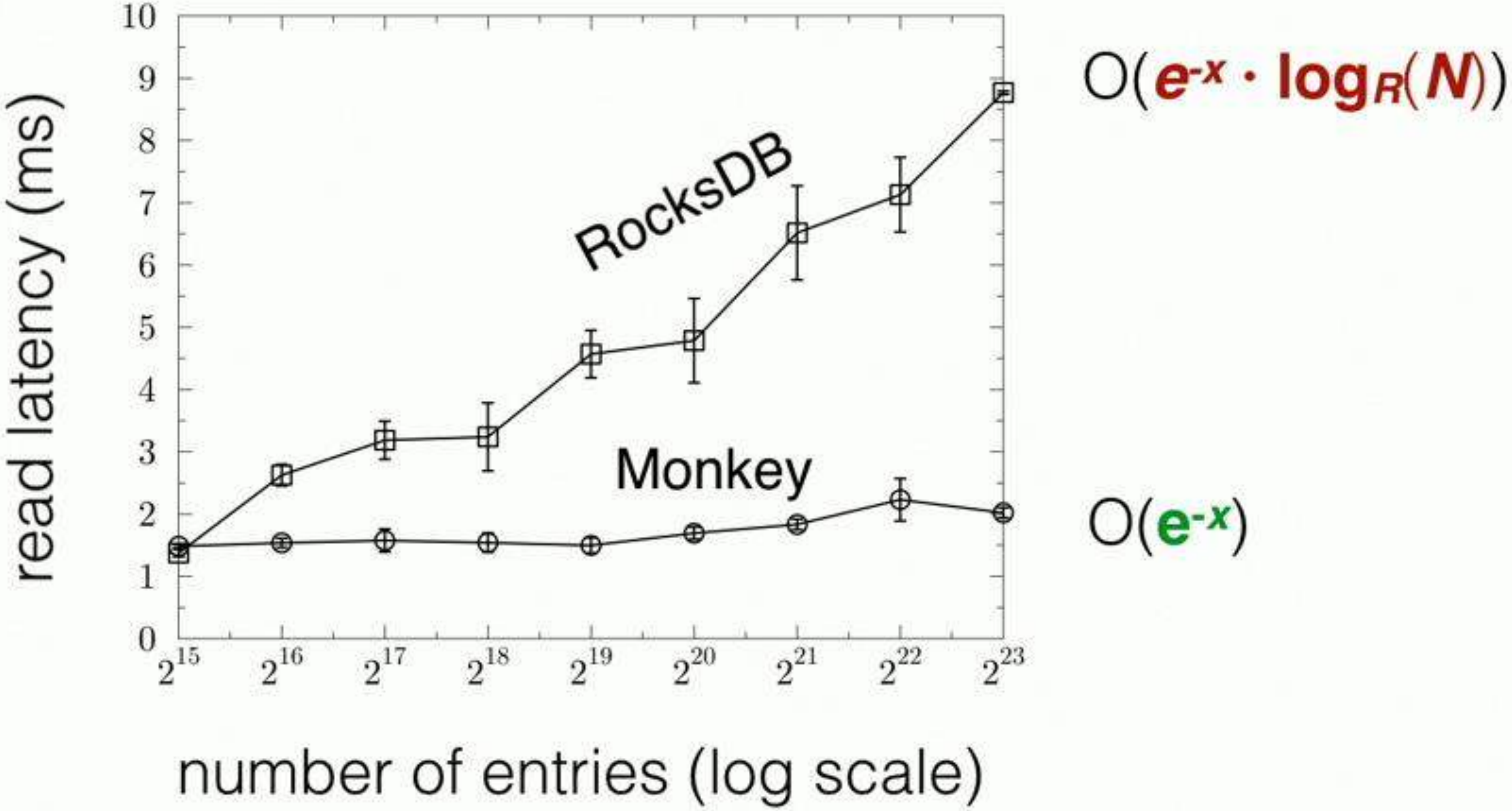
geometric progression

$$\left.\begin{array}{c} O(e^{-x}/R^{2}) \\ \\ O(e^{-x}/R^{1}) \\ \\ O(e^{-x}/R^{0}) \end{array}\right\} = \quad O(e^{-x})$$

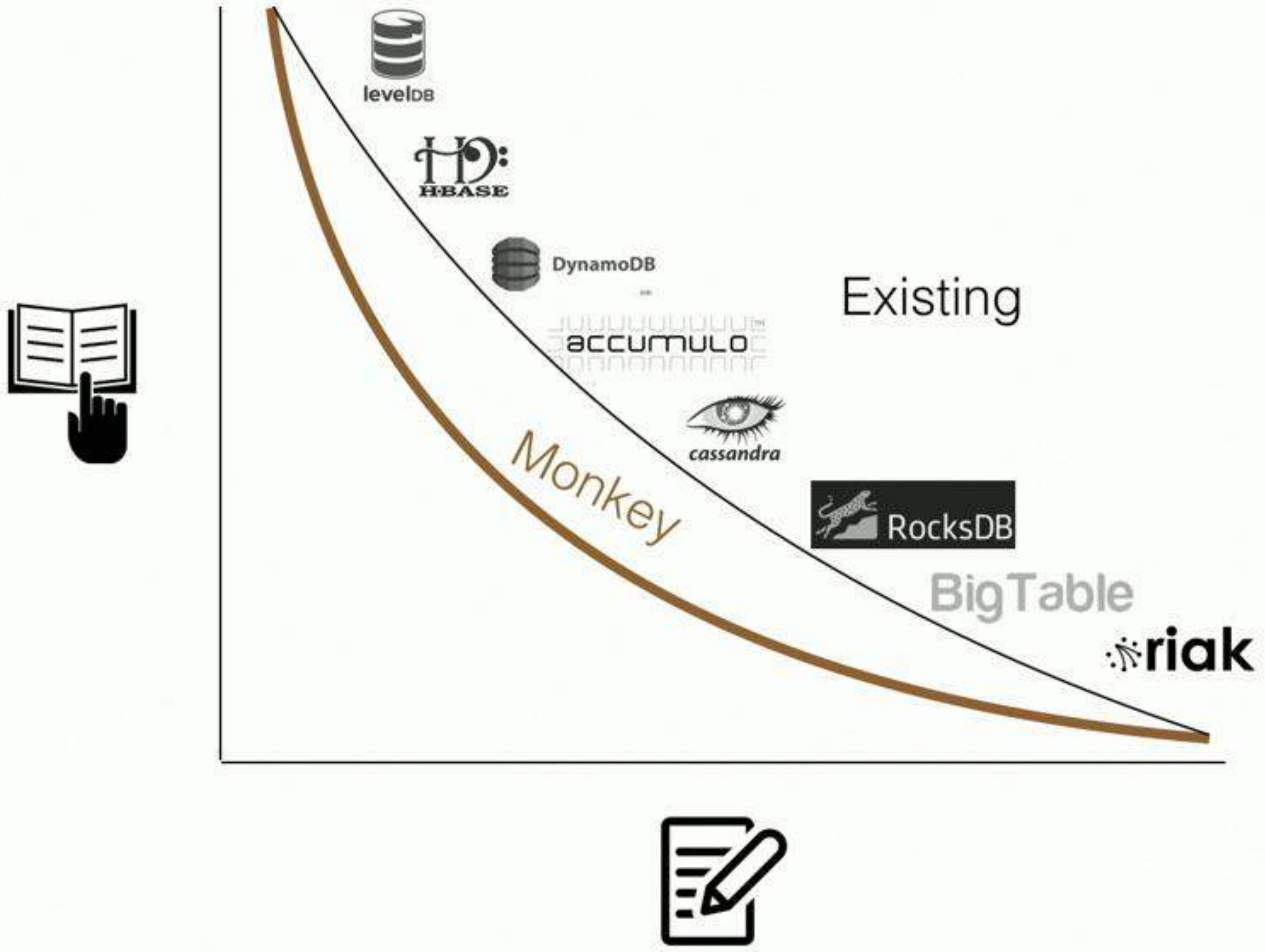
$$O(e^{-x} \cdot \log_R(N)) \quad > \quad O(e^{-x})$$

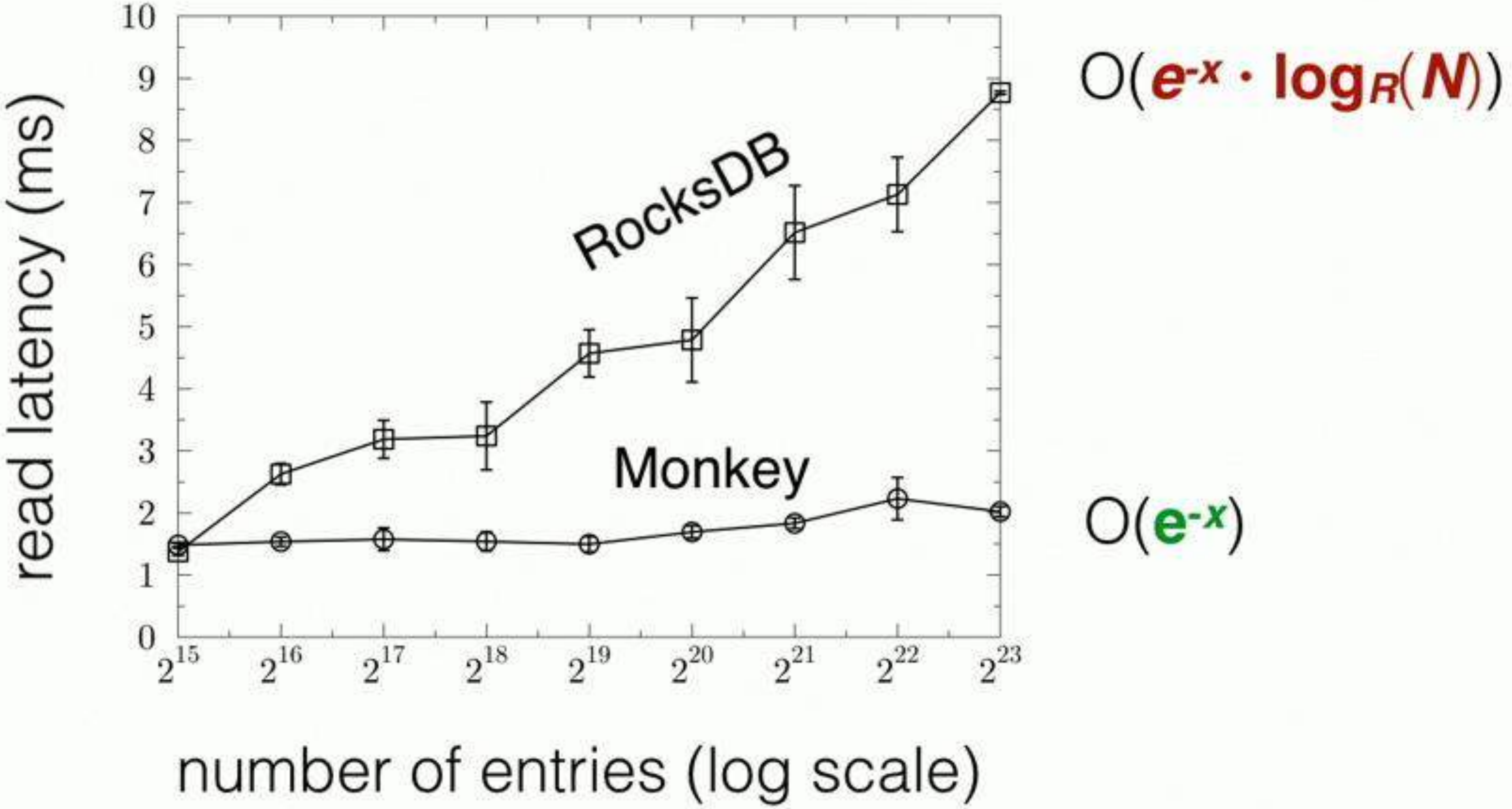


DASlab @ Harvard SEAS

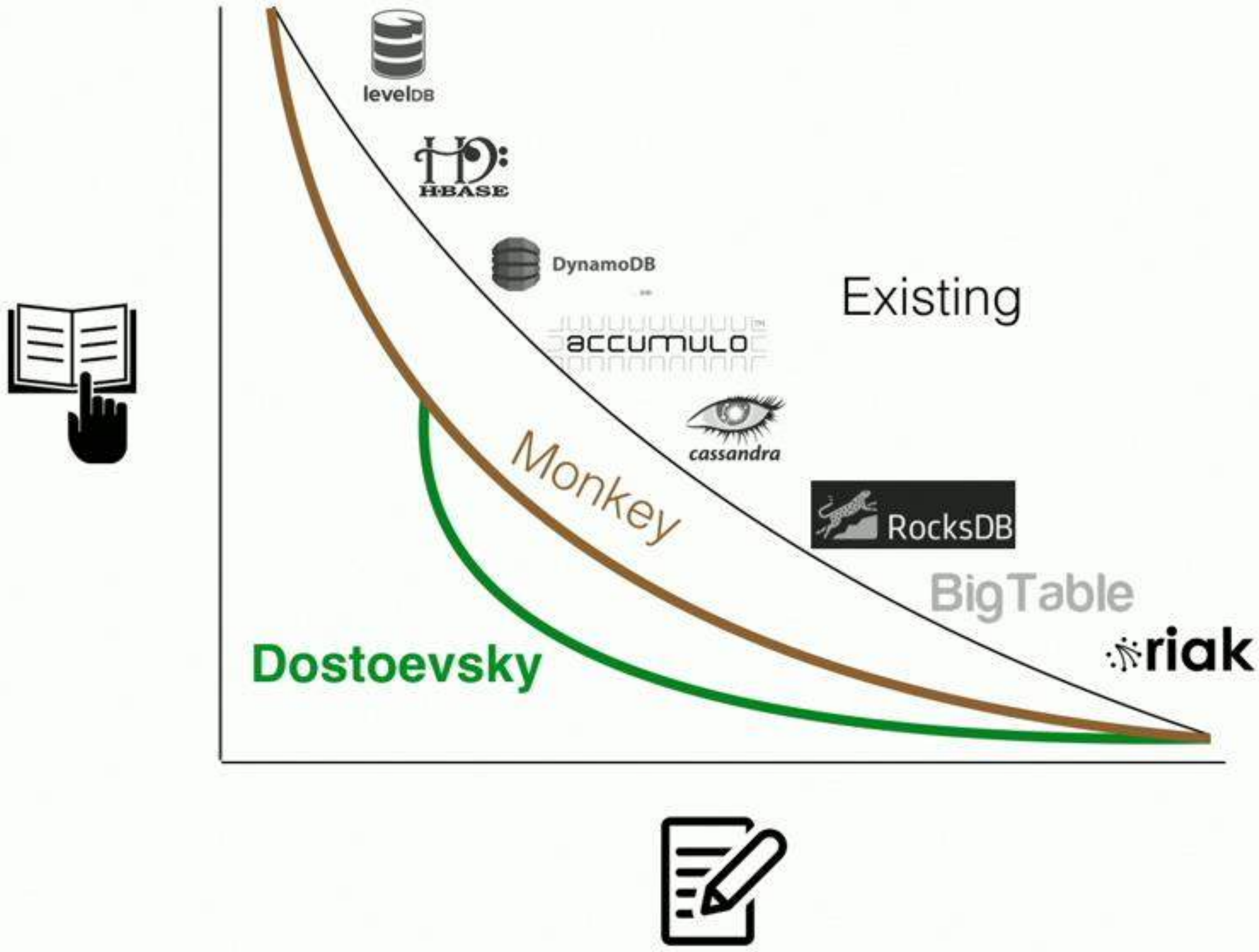


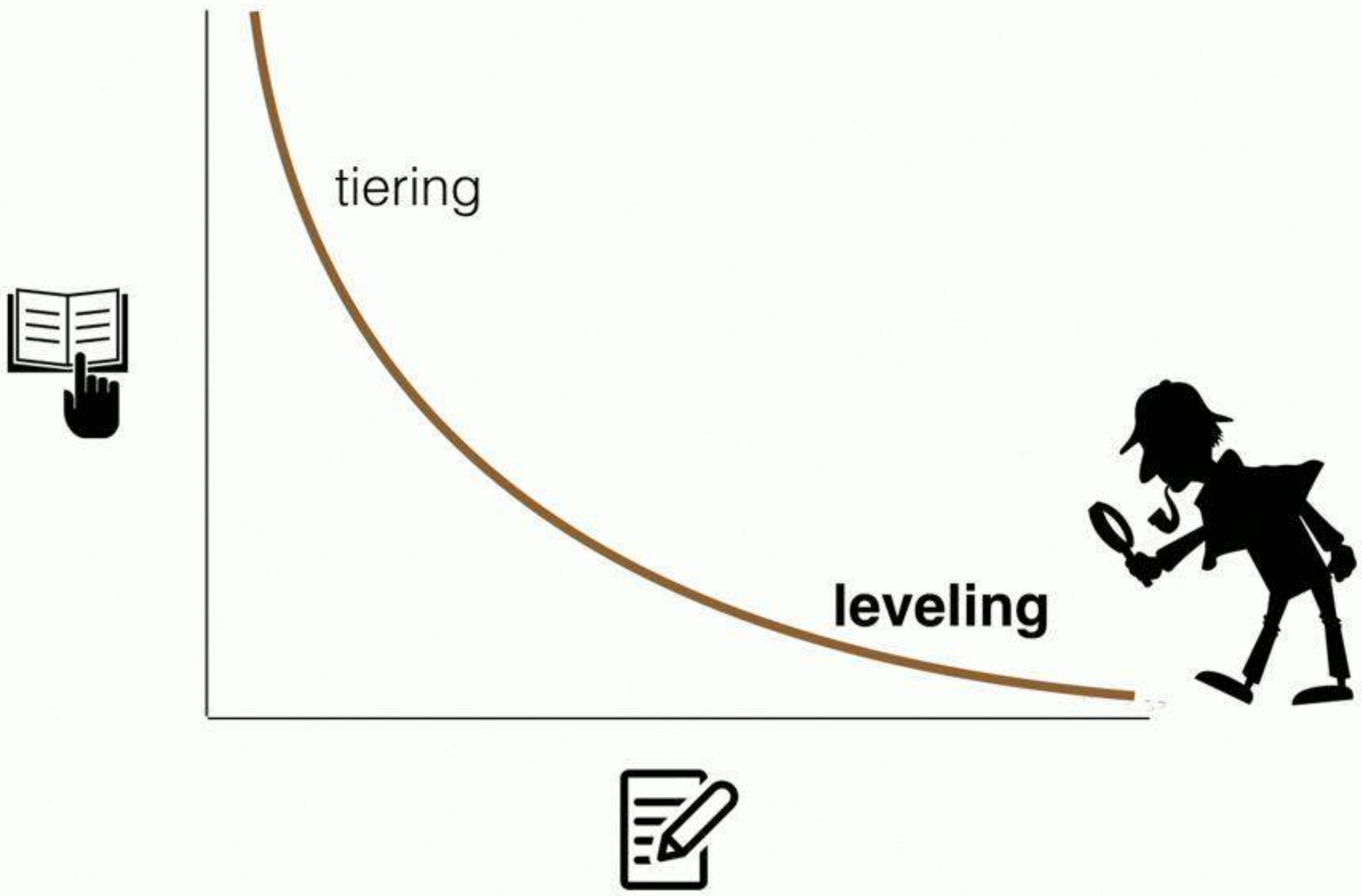
DASlab
@ Harvard SEAS


levelDB
HBASE
DynamoDB
accumulo
cassandra
RocksDB
BigTable
riak
Existing
Monkey



DASlab
@ Harvard SEAS


levelDB
HBASE
DynamoDB
accumulo
cassandra
RocksDB
BigTable
riak
Existing
Monkey
Dostoevsky


tiering
leveling

I/O overheads with leveling

point

long range

short range

writes

point

false positive rates

exponentially decreasing

$O(\mathbf{e^{-x}}/\boldsymbol{R^2})$

$O(\mathbf{e^{-x}}/\boldsymbol{R})$

$O(\mathbf{e^{-x}})$

false positive rates

$O(e^{-x}/R^2)$

$O(e^{-x}/R)$

point $\longrightarrow$ $O(e^{-x})$

point

long range

short range

writes

largest level

$O(e^{-x})$

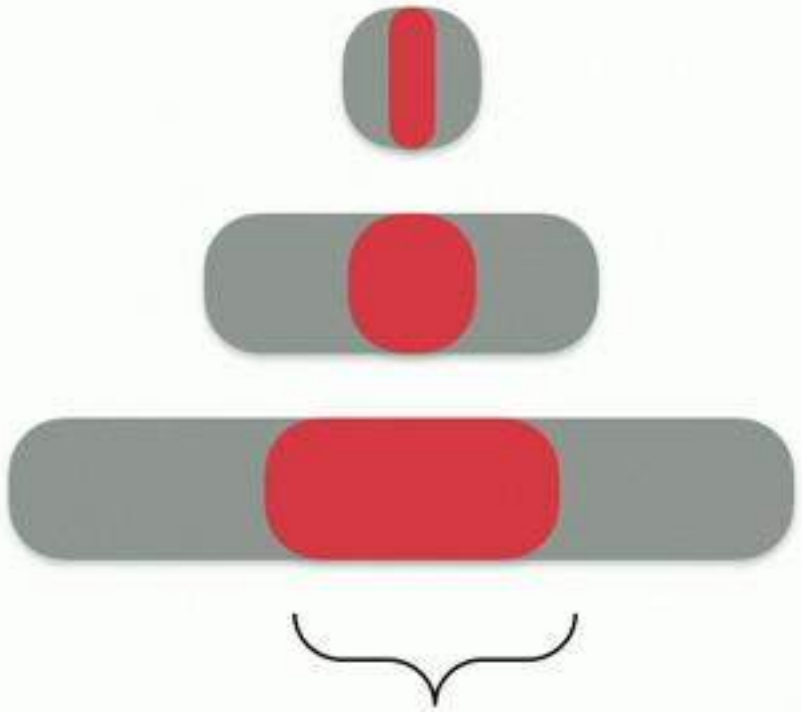
long range

target range

O(s/R^2)

O(s/R)

O(s)



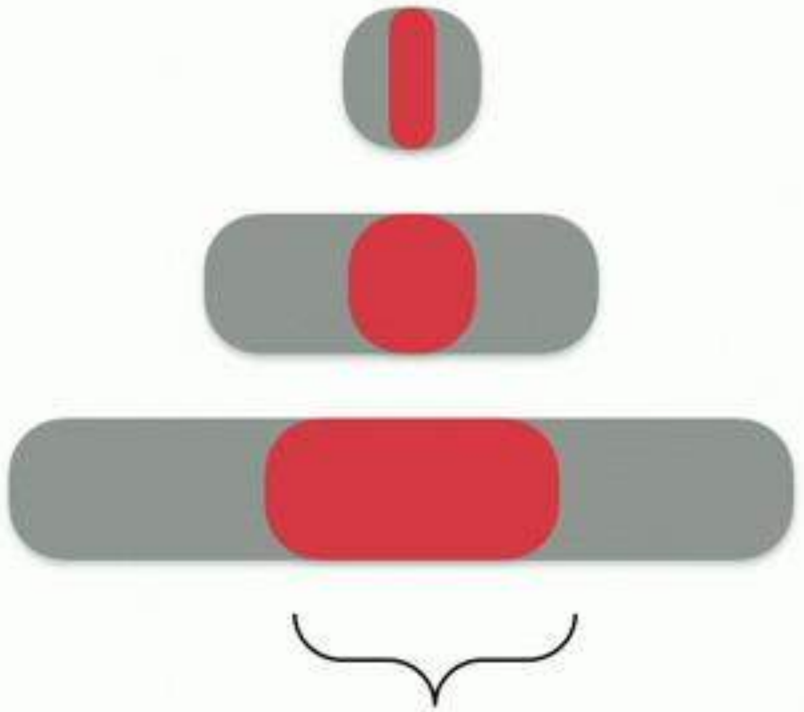
target key range

target range

$O(s/R^2)$

$O(s/R)$

long range $O(s)$ **largest level**

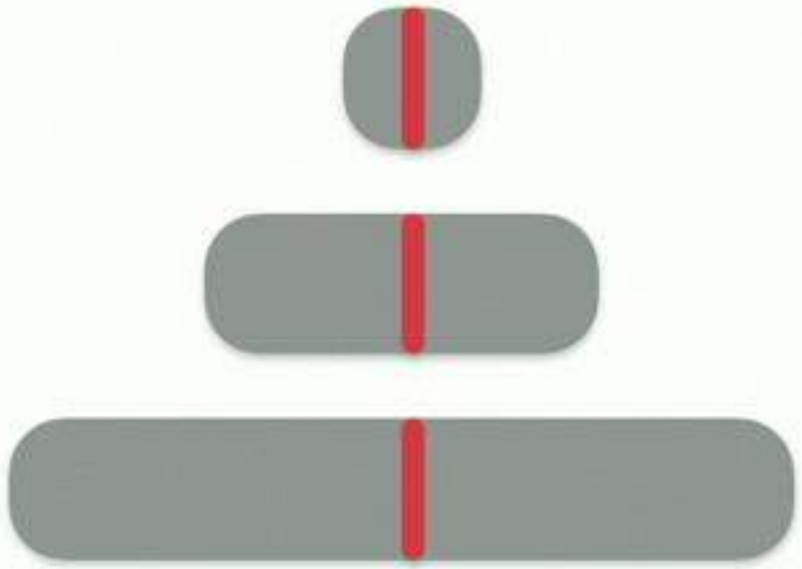


target key range

DASlab @ Harvard SEAS

point	long range	**short range**	writes
largest level	largest level		
$O(e^{-x})$	$O(s)$		

short range



target range

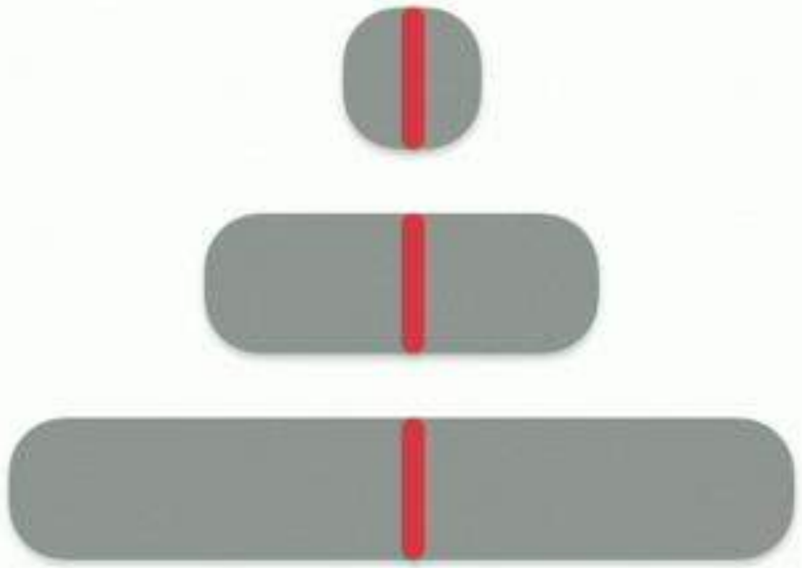
1

1

1

short range

target range



1

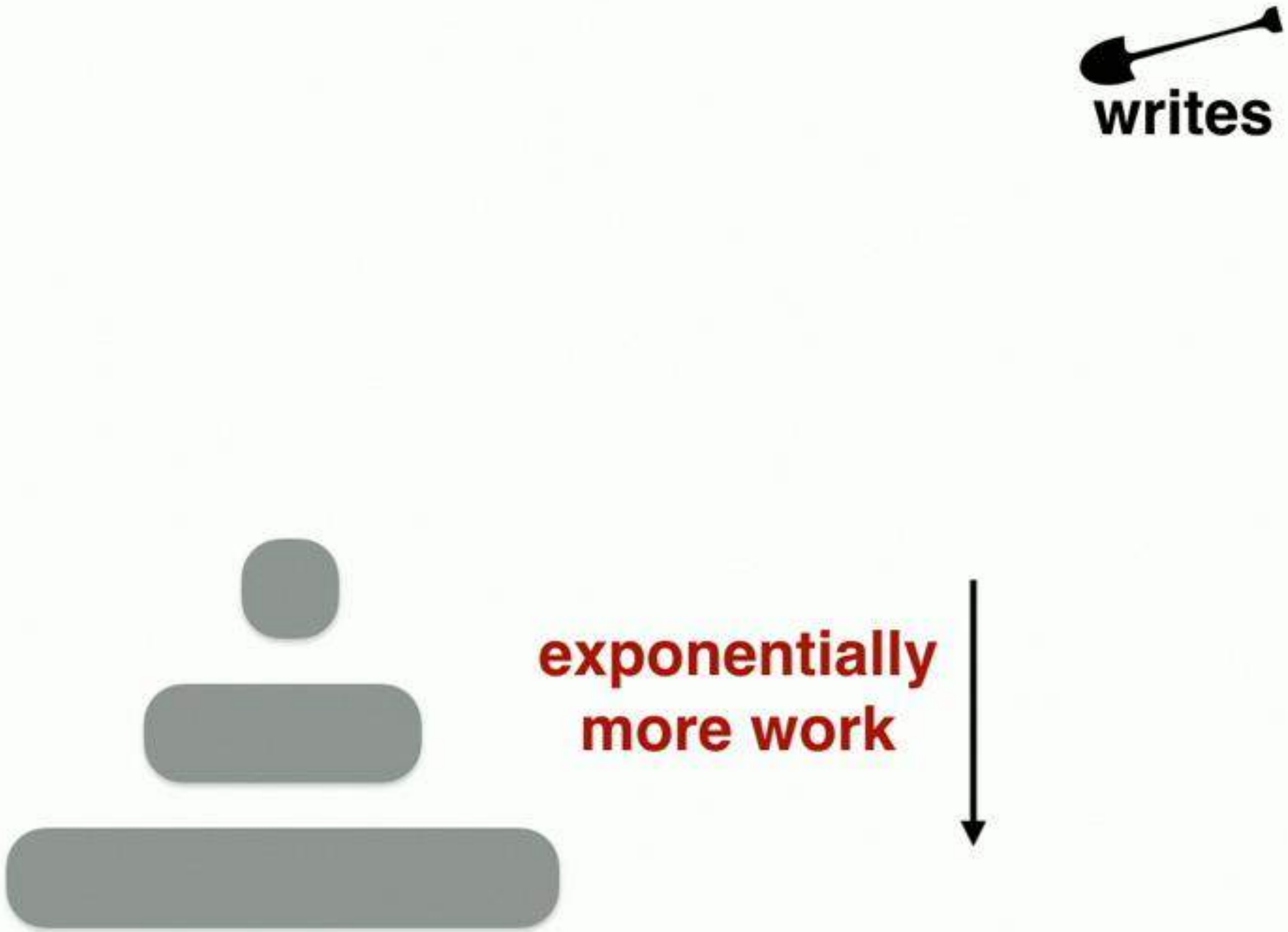
1

1

all levels

$O(\mathbf{\log_R(N)})$

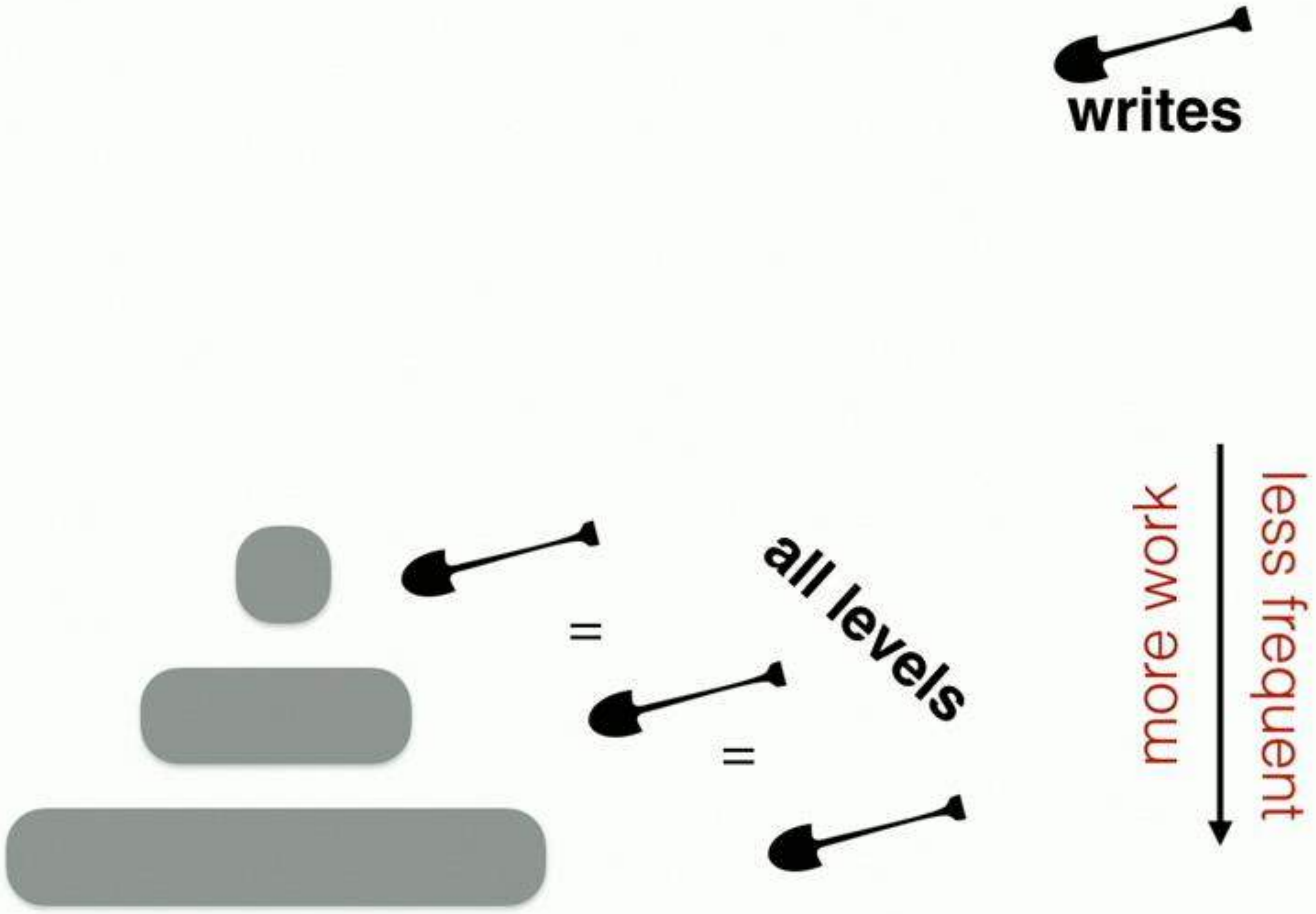
point	long range	short range	**writes**
largest level	largest level	all levels	
$O(e^{-x})$	$O(s)$	$O(\log_{R}(N))$	

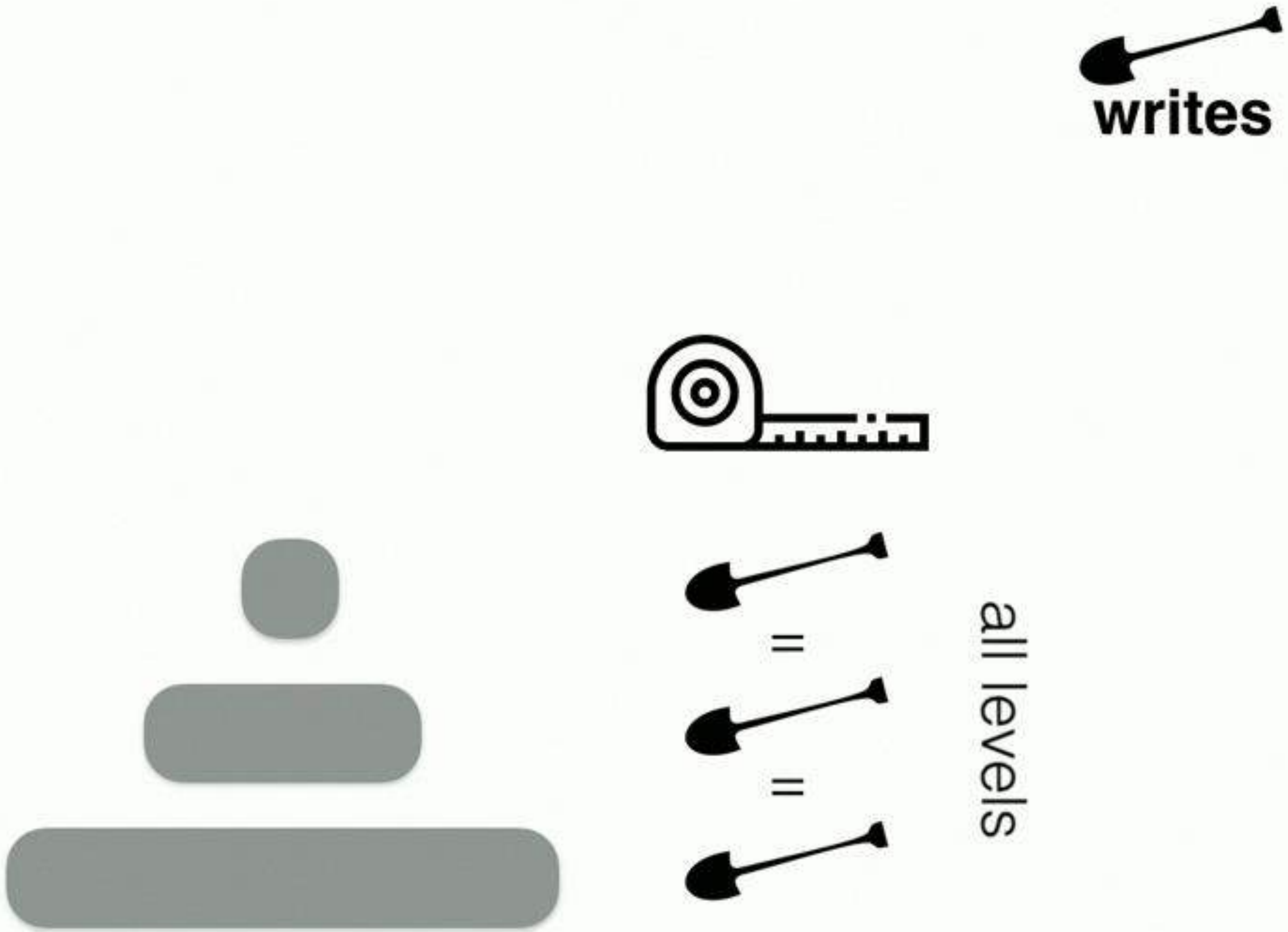

writes
exponentially
more work

writes

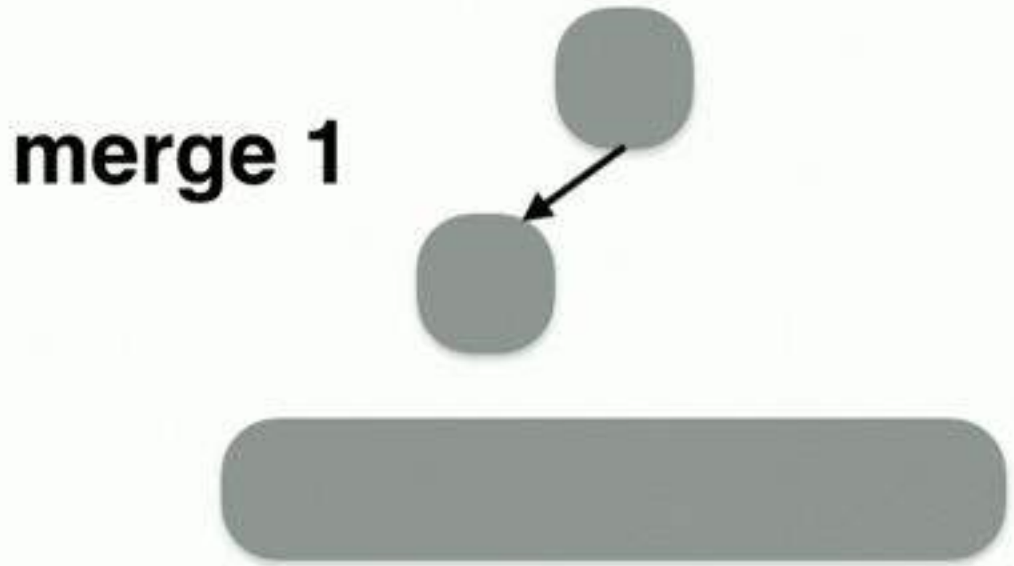
exponentially
more work

exponentially
less frequent

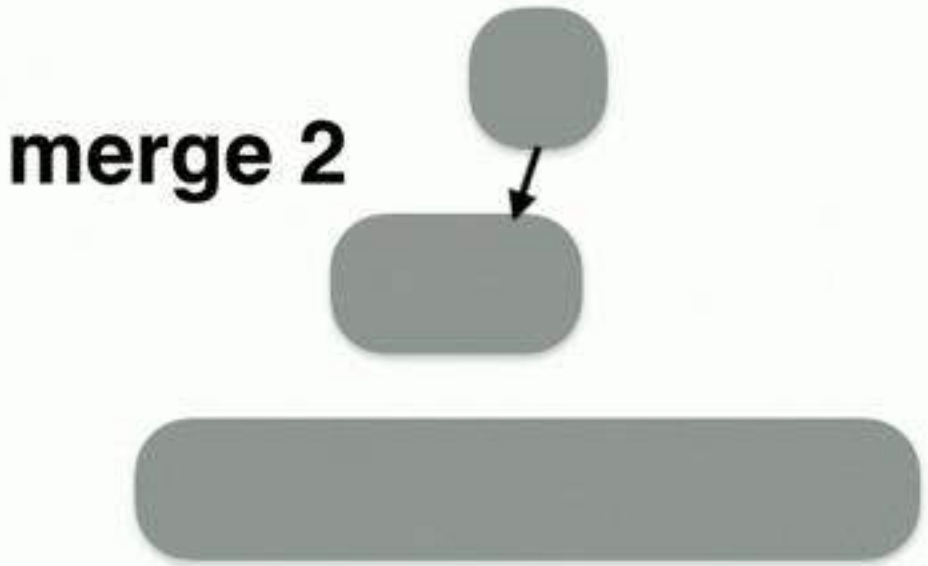

writes
=
=
all levels
more work
less frequent


writes
=
=
all levels

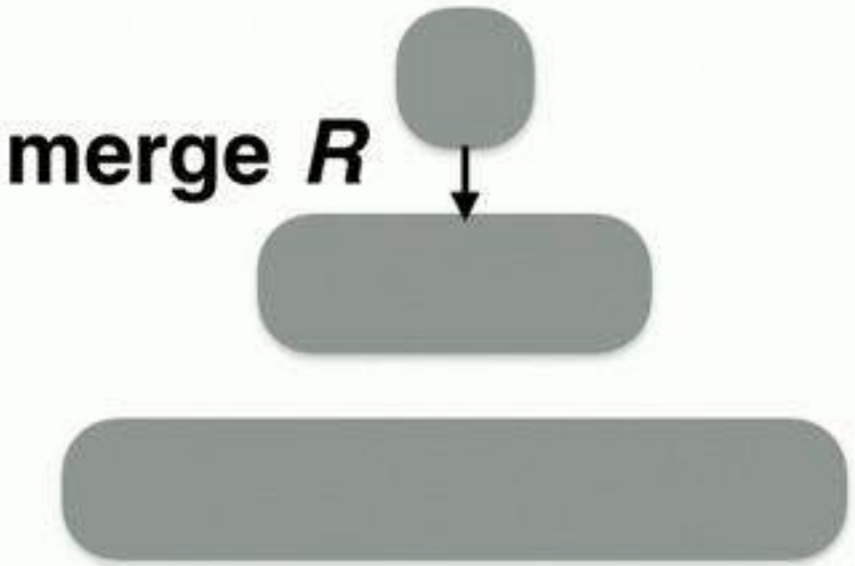
writes



writes



writes



writes

write-amplification

O(***R***)

O(***R***)

O(***R***)

writes

$$\left.\begin{array}{l} O(R) \\ O(R) \\ O(R) \end{array}\right\} \mathbf{O}(\boldsymbol{R} \cdot \log_{\boldsymbol{R}}(\boldsymbol{N}))$$

point	long range	short range	writes
largest level	largest level	all levels	all levels
$O(e^{-x})$	$O(s)$	$O(\log_R(N))$	$O(R \cdot \log_R(N))$
=	=	=	=
$O(e^{-x}/R^2)$	$O(s/R^2)$	1	$O(R)$
+	+	+	+
$O(e^{-x}/R)$	$O(s/R)$	1	$O(R)$
+	+	+	+
$O(e^{-x})$	$O(s)$	1	$O(R)$

point	long range	writes

point	long range	writes
largest level	**largest level**	**all levels**

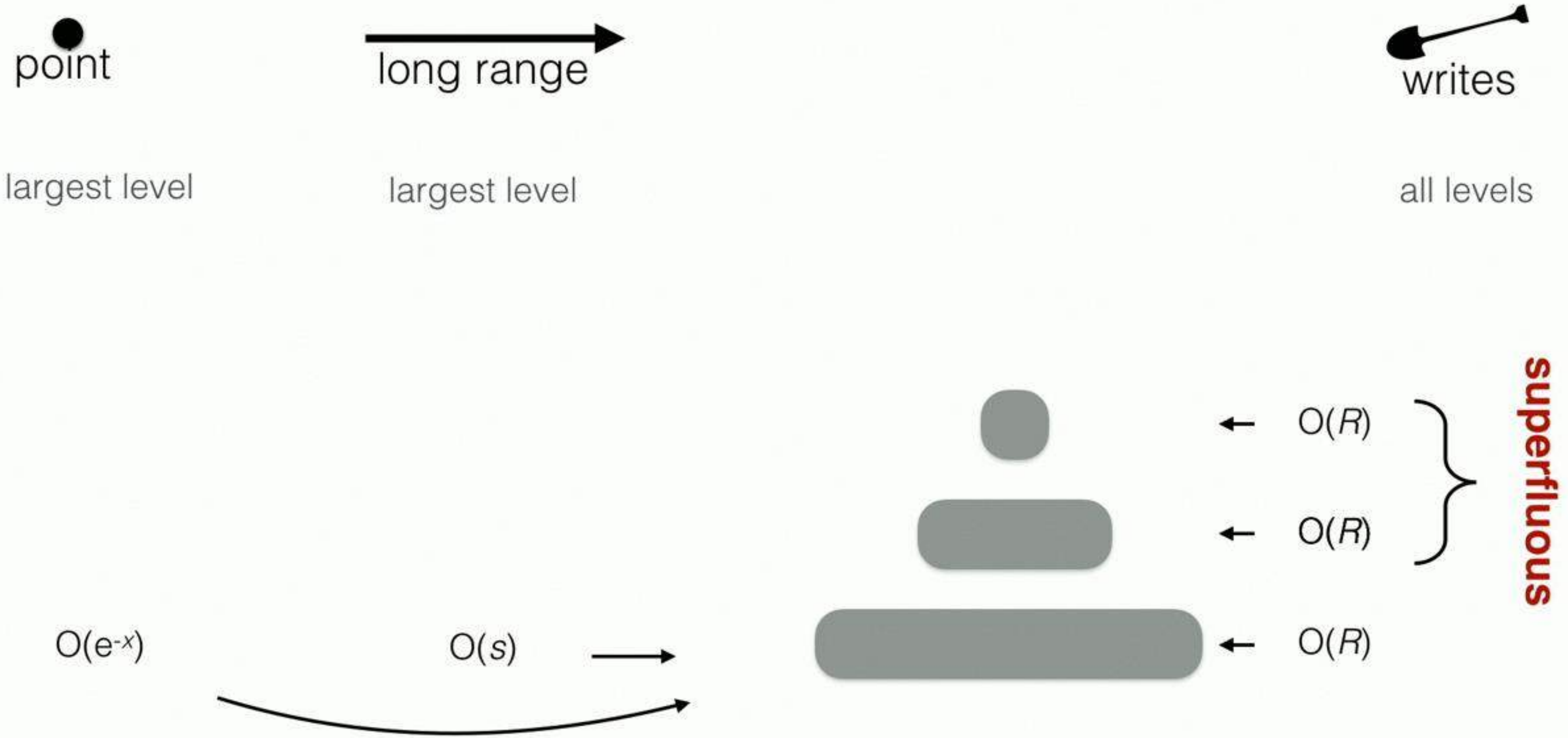
$O(R)$

$O(R)$

$O(R)$

$O(e^{-x})$

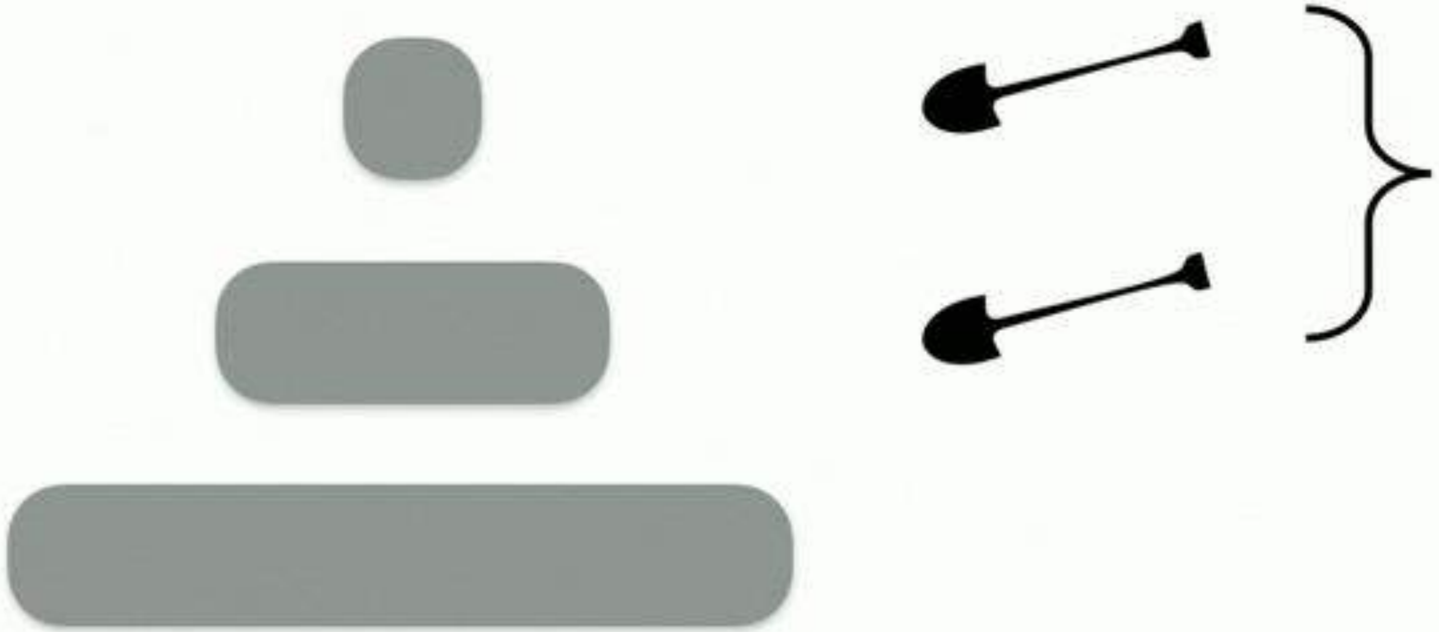
$O(s)$



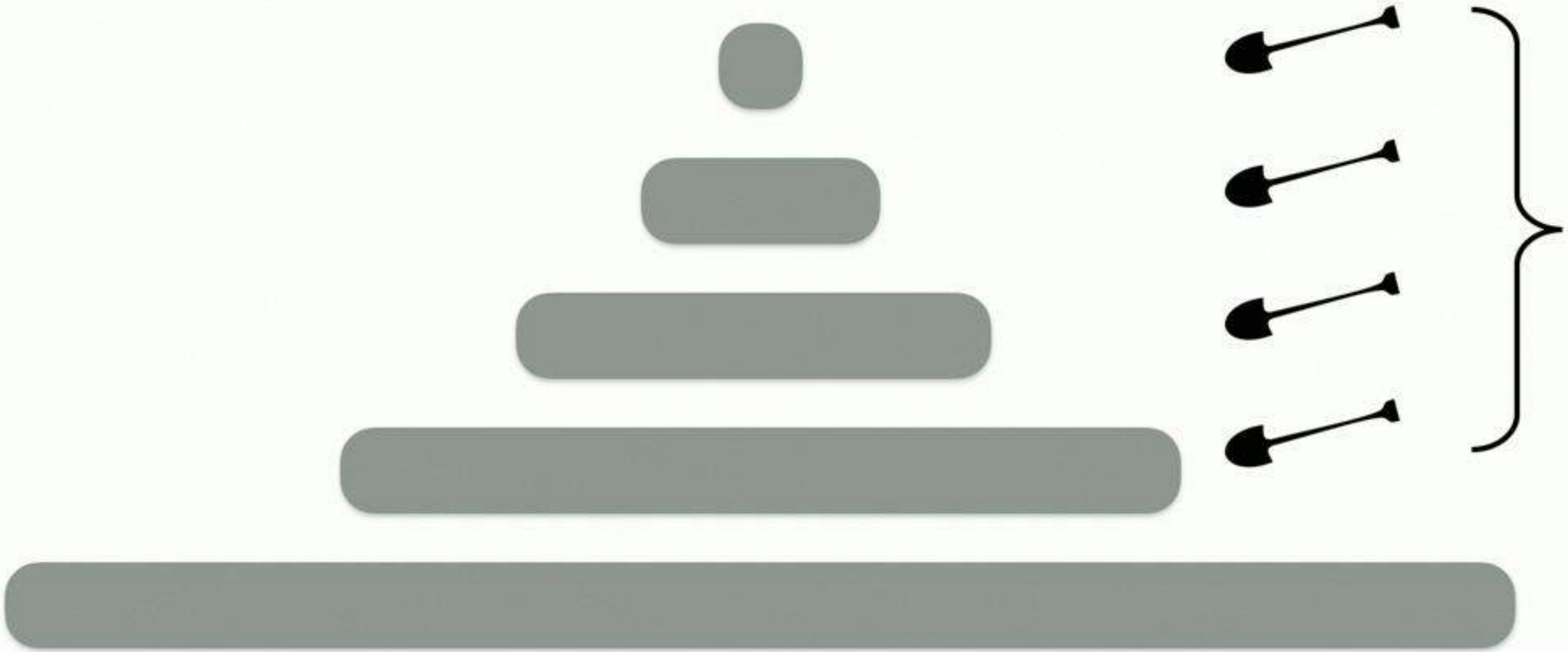
DASlab @ Harvard SEAS

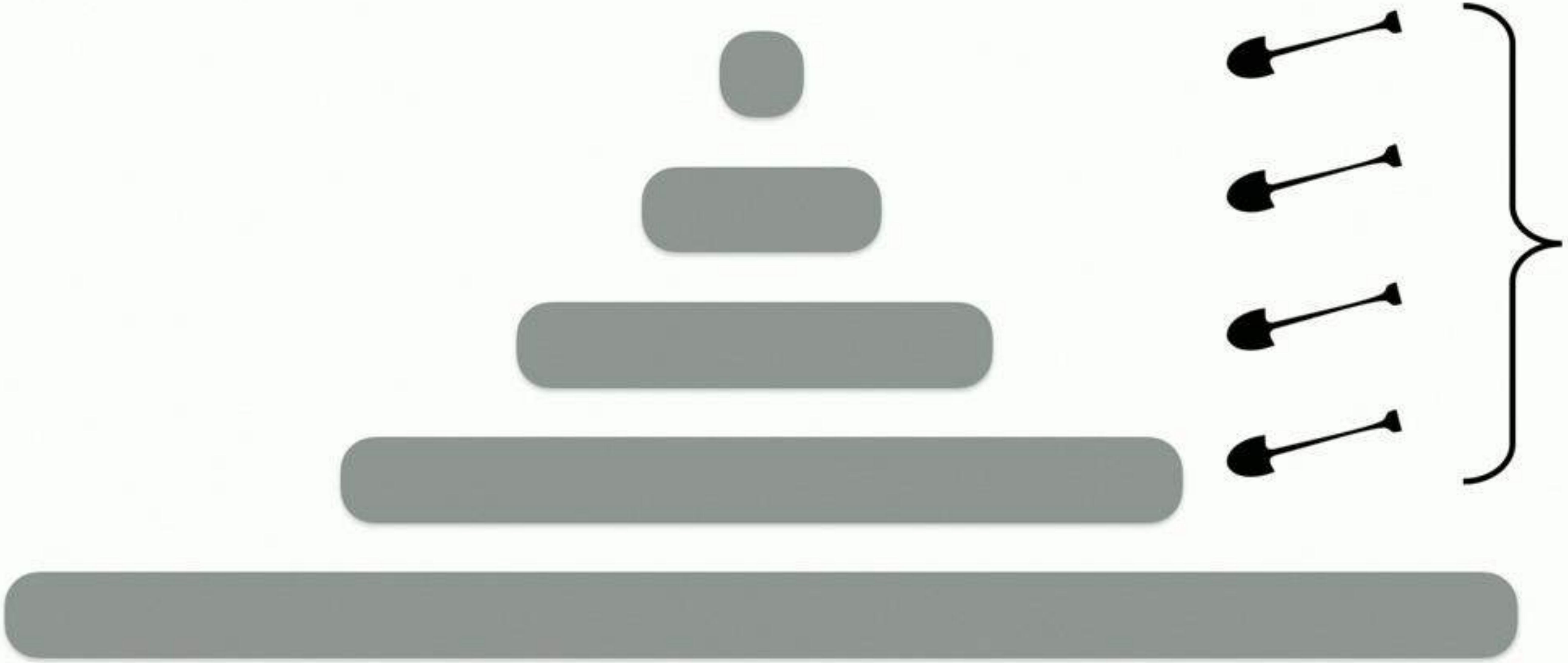
for point lookups and long range lookups

merging at **smaller levels** is **superfluous**

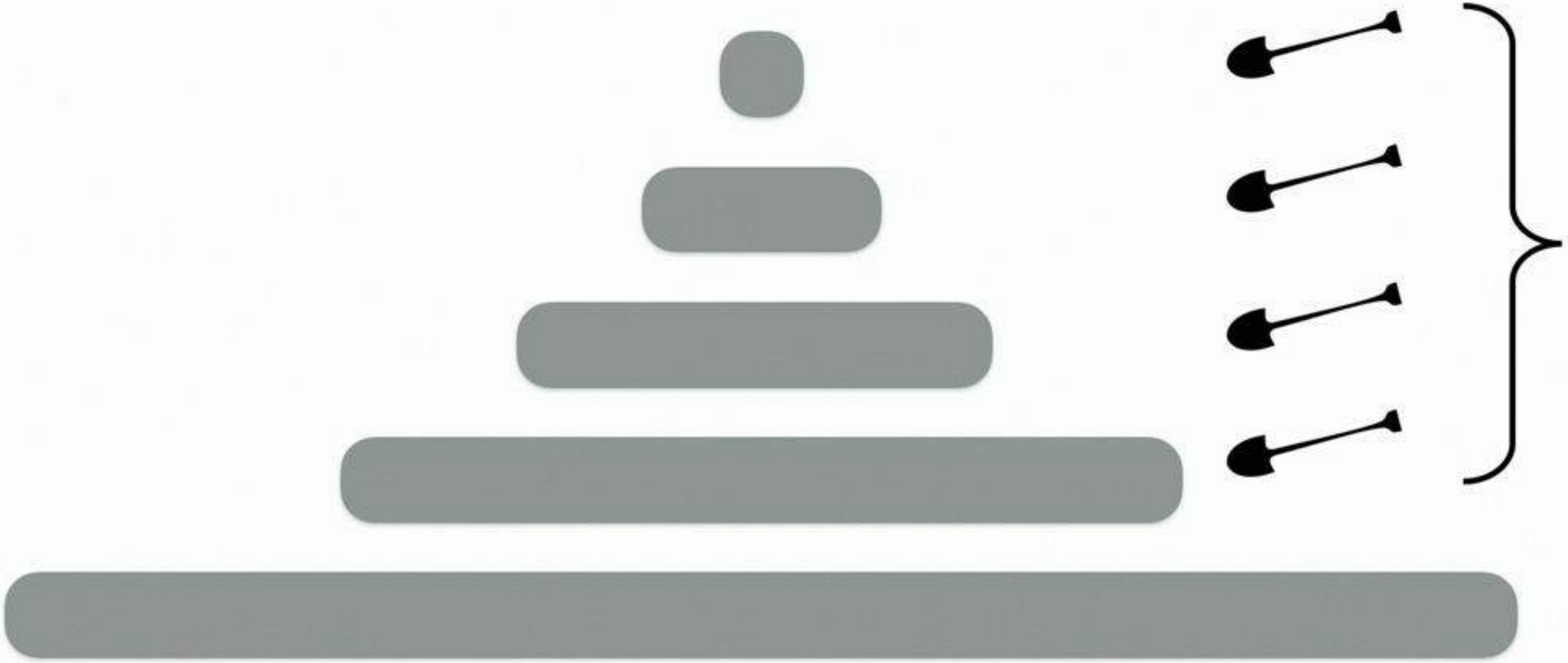


worse as data grows!





poor performance



poor performance

lower device lifetime (on SSD)

Dostoevsky

SIGMOD18

Dostoevsky: **S**pace-**T**ime **O**ptimized **Ev**olvable **S**calable **Key**-Value Store

very write-optimized

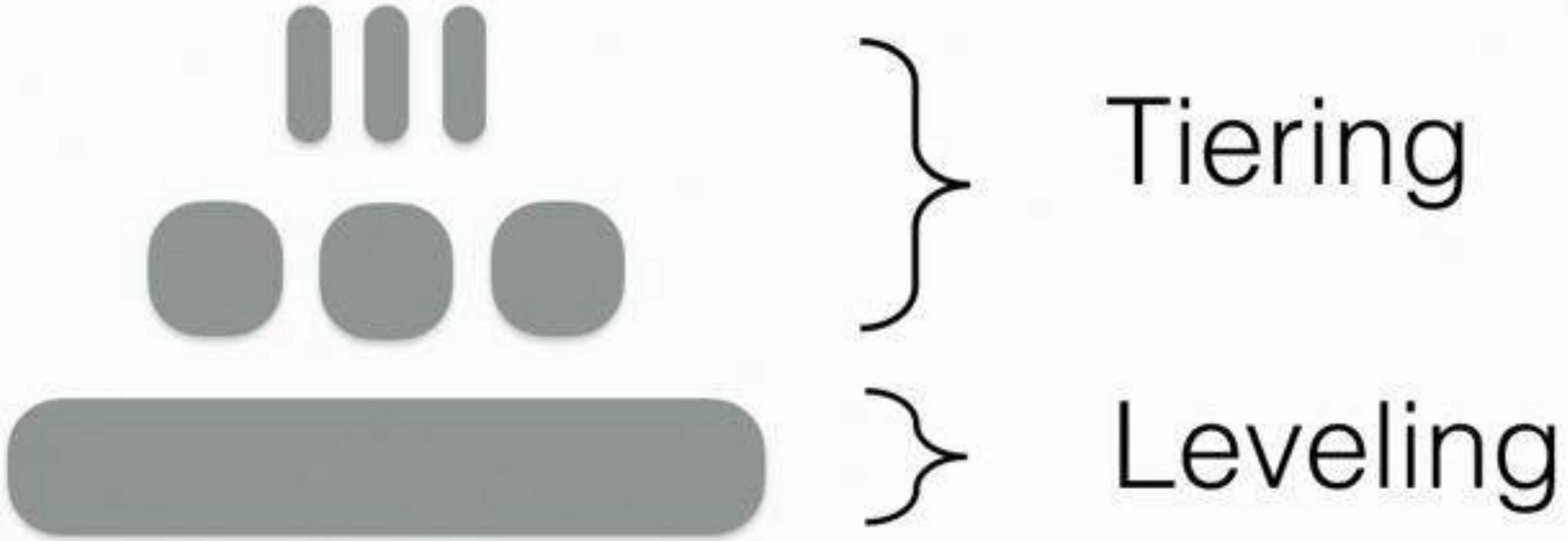
Dostoevsky: **S**pace-**T**ime **O**ptimized **Ev**olvable **S**calable **K**ey-Value Store

Tiering
write-optimized

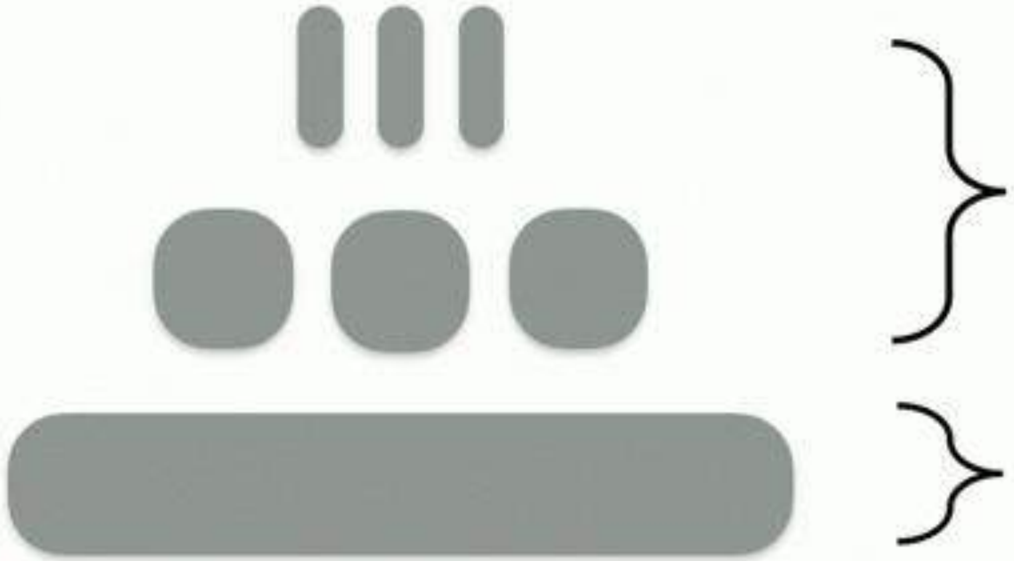
Leveling
read-optimized

Tiering	**Lazy Leveling**	Leveling
write-optimized	mixed-optimized	read-optimized

Lazy Leveling



Lazy Leveling



merge when level fills

merge to have at most 1 run

point

long range

short range

writes

point

false positive rates

$O(e^{-x}/R^3)$

$O(e^{-x}/R^2)$

$O(e^{-x})$

point

false positive rates

$O(\mathbf{e}^{-x}/R^3)$

exponentially decreasing

$O(\mathbf{e}^{-x}/R^2)$

$O(\mathbf{e}^{-x})$

false positive rates

$O(e^{-x}/R^3)$

$O(e^{-x}/R^2)$

point $\longrightarrow$ $O(e^{-x})$

point

$O(e^{-x})$

with uniform FPRs

$$O(\log_R(N) \cdot R \cdot e^{-x}) \left\{ \begin{array}{l} O(e^{-x}) \\ O(e^{-x}) \\ O(e^{-x}) \end{array} \right.$$

point

long range

short range

writes

$O(e^{-x})$

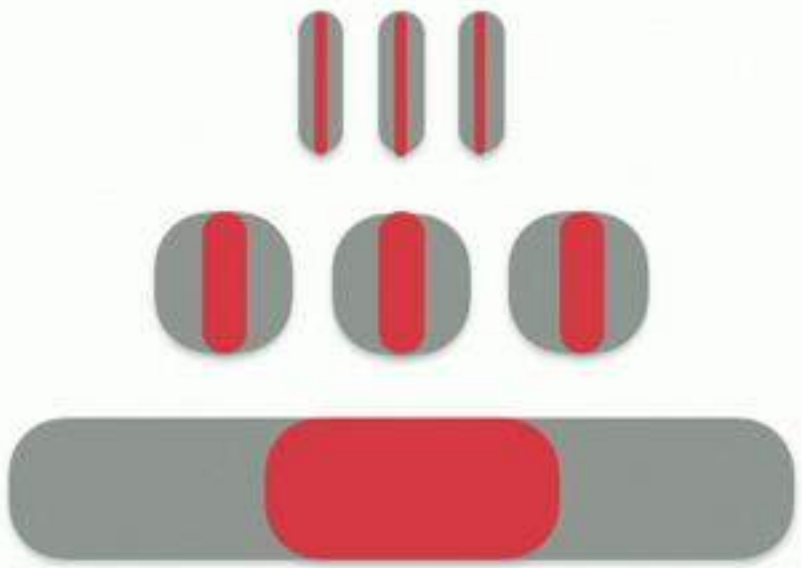
long range

target range

$O(s/R^2)$

$O(s/R)$

$O(s)$



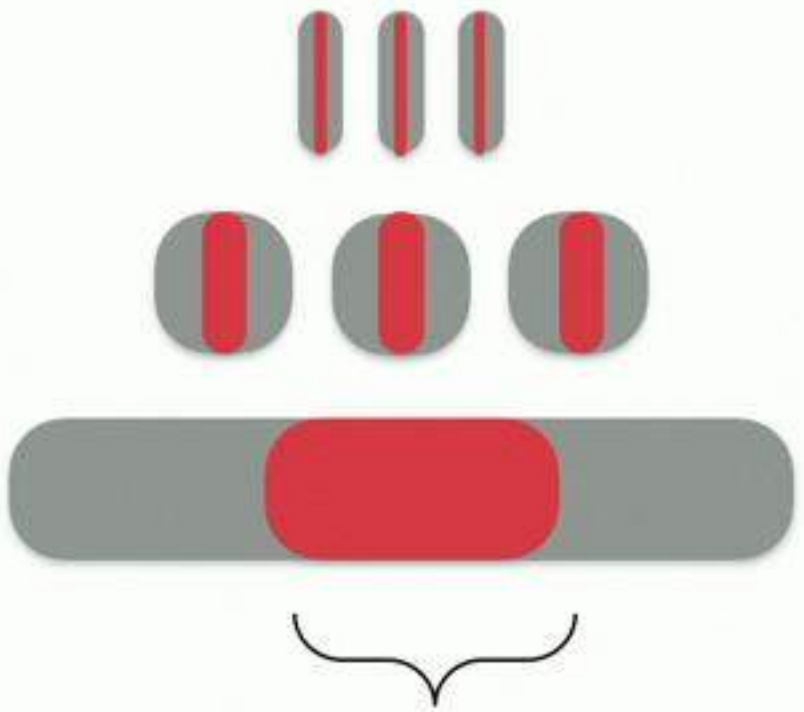
target key range

target range

$O(s/R^2)$

$O(s/R)$

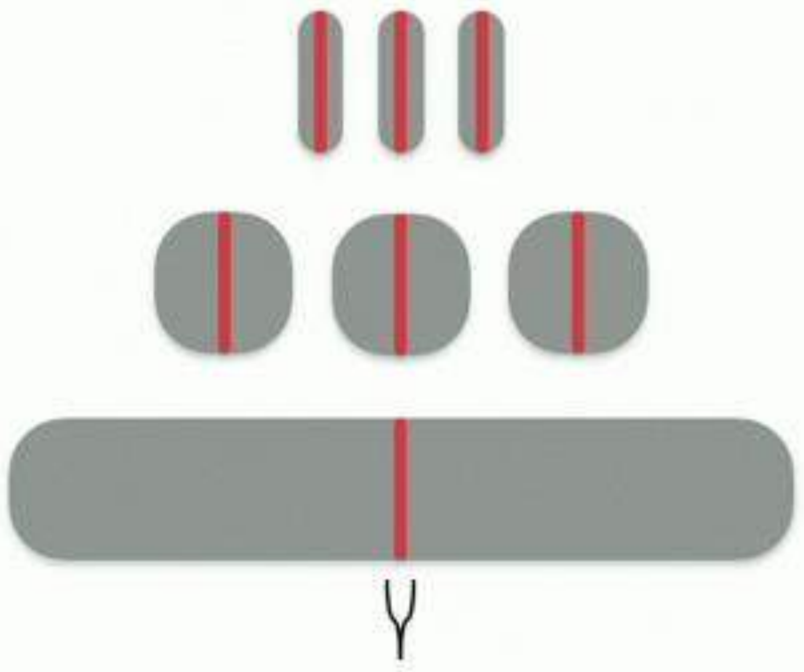
long range $O(s)$ largest level



target key range

point	long range	**short range**	writes
$O(e^{-x})$	$O(s)$		

short range



$O(R)$

$O(R)$

1

$$O(1+R\cdot(\log_R(N)-1))$$

target key range

DASlab @ Harvard SEAS

point $O(e^{-x})$

long range $O(s)$

short range $O(1+R\cdot(\log_R(N)-1))$

writes

writes

write-amplification

$O(\mathbf{1})$

$O(\mathbf{1})$

$O(\boldsymbol{R})$

writes

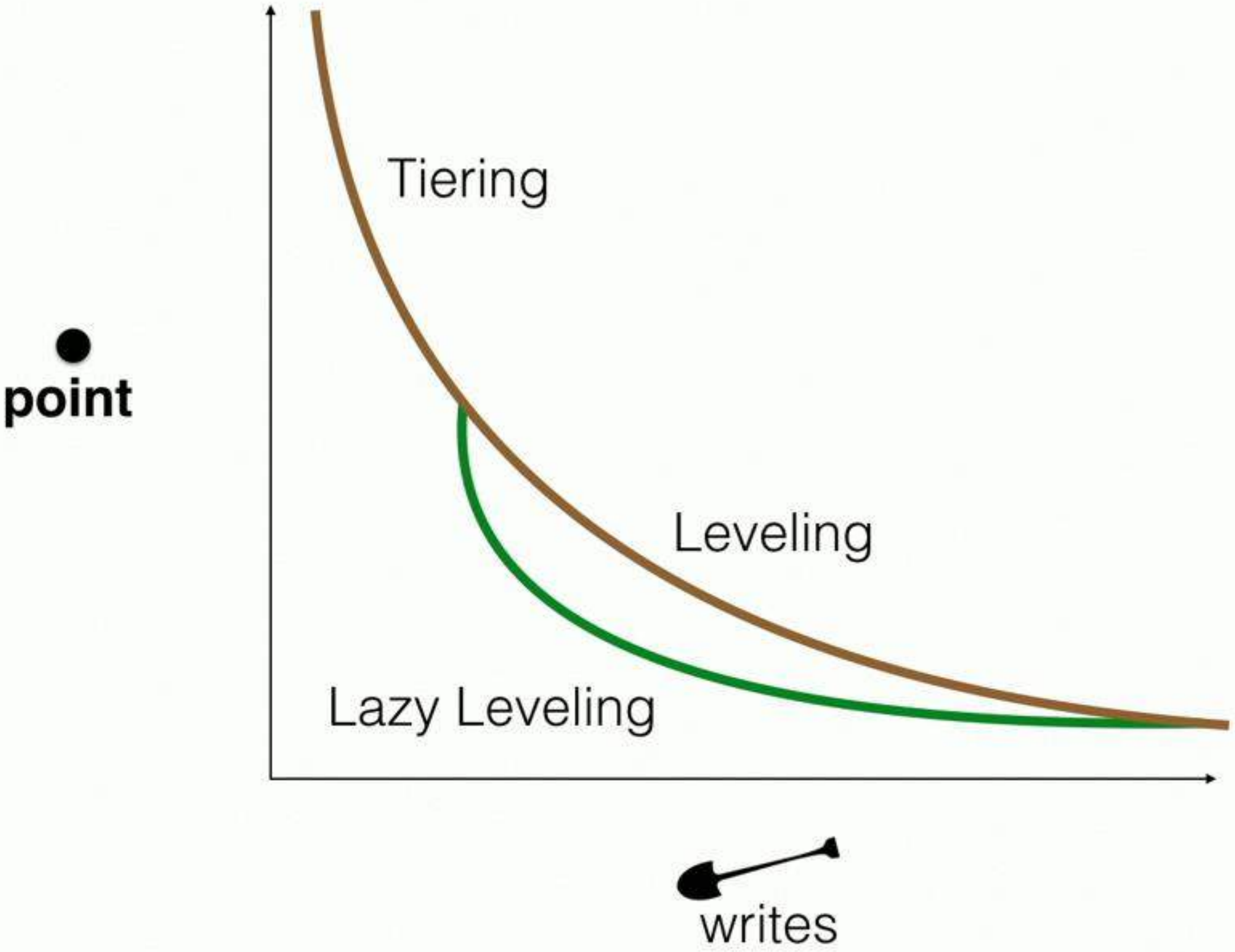
write-amplification

$$\left.\begin{array}{l} O(1) \\ O(1) \\ O(R) \end{array}\right\} O(\,\boldsymbol{R} + \mathbf{log}_R(\boldsymbol{N})\,)$$

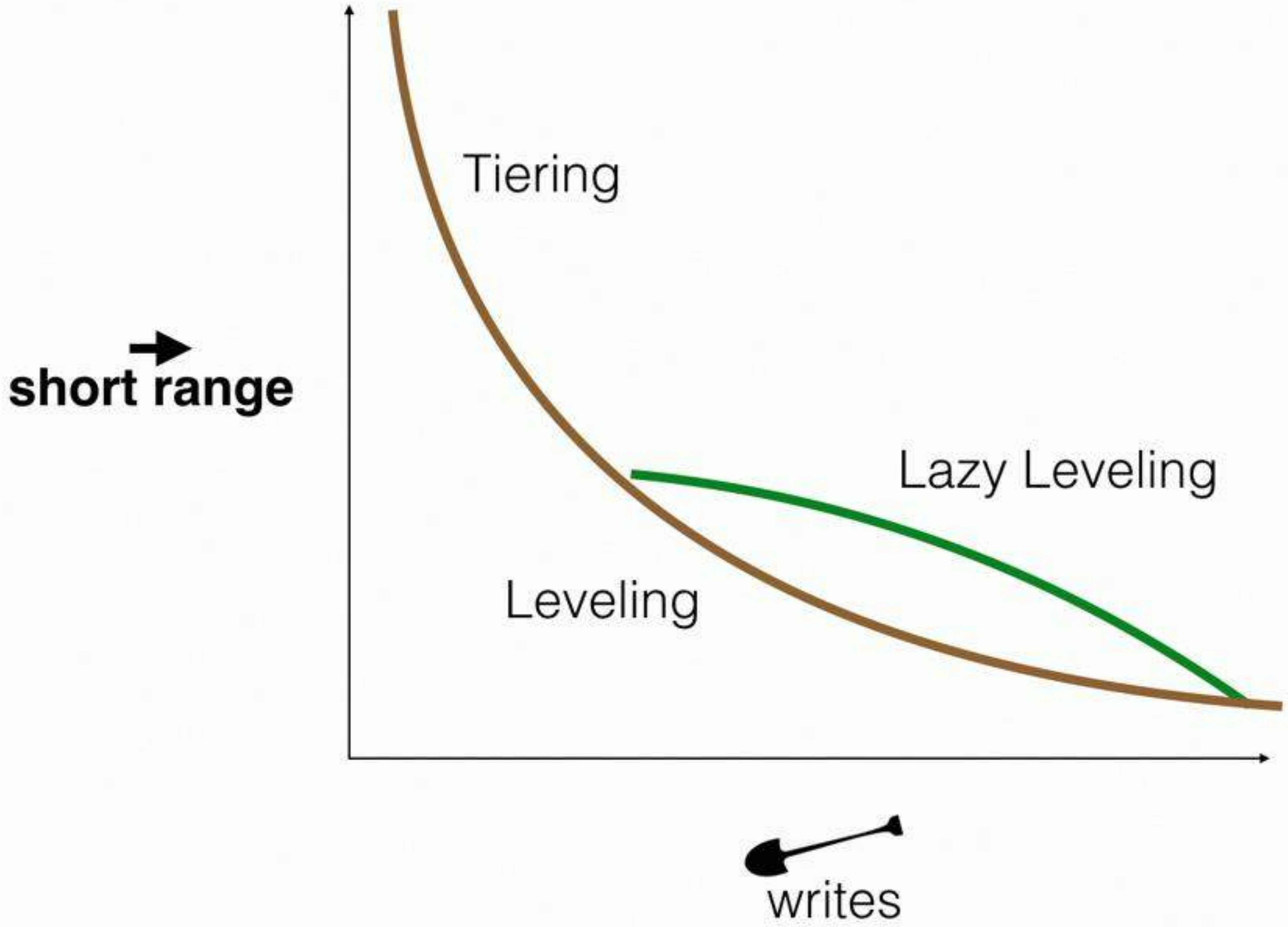
point	long range	short range	writes
$O(e^{-x})$	$O(s)$	$O(1+R \cdot (\log_R(N)-1))$	$O(R + \log_R(N))$

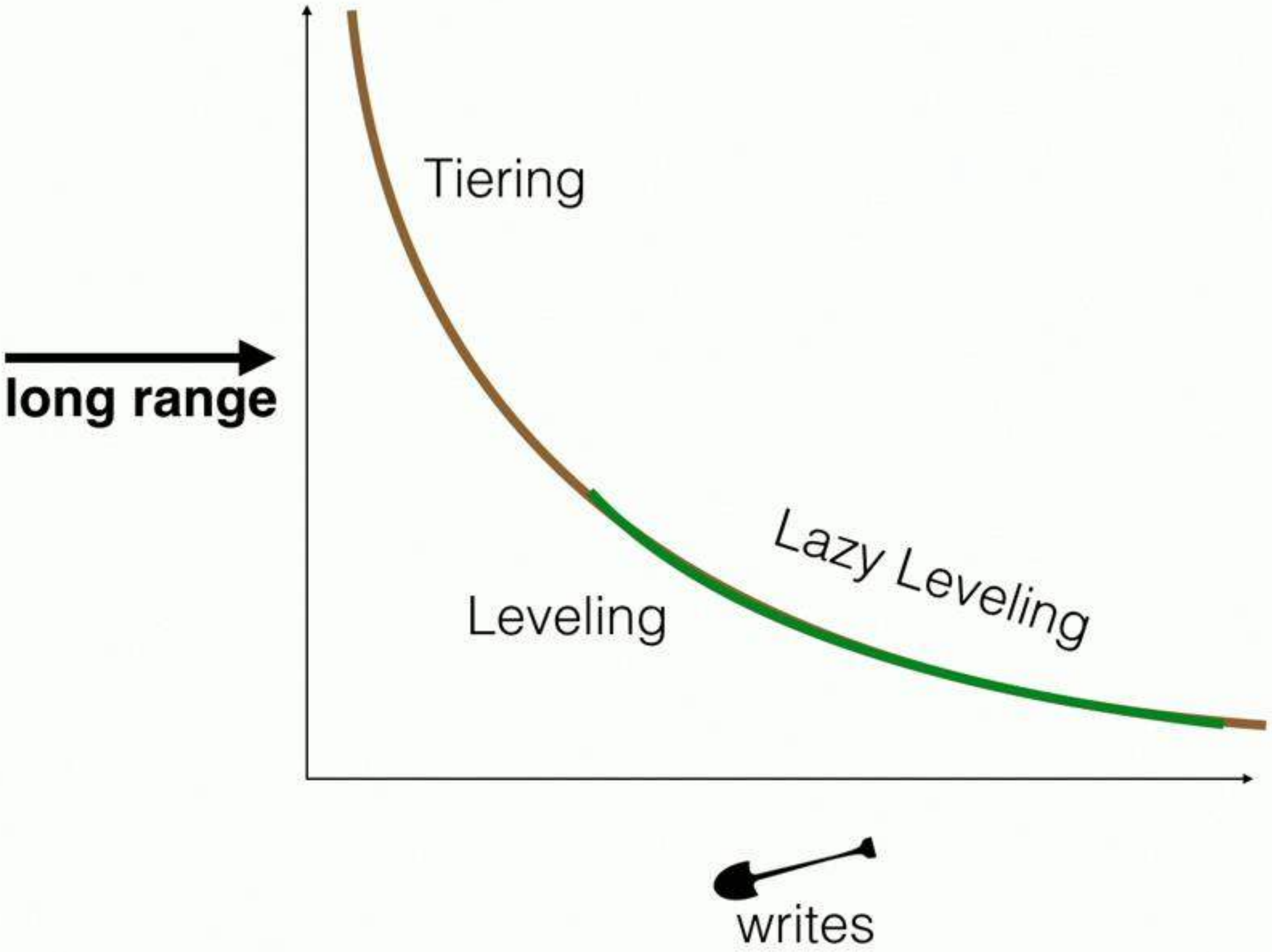
	point	long range	short range	writes
Lazy Leveling	$O(e^{-x})$	$O(s)$	$O(1+R \cdot (\log_R(N)-1))$	$O(R + \log_R(N))$
	=	=	$\vee$	$\wedge$
Leveling	$O(e^{-x})$	$O(s)$	$O(\log_R(N))$	$O(R \cdot \log_R(N))$

	point	long range	short range	writes
Tiering	$O(R \cdot e^{-x})$	$O(R \cdot s)$	$O(R \cdot \log_R(N))$	$O(\log_R(N))$
	$\vee$	$\vee$	$\vee$	$\wedge$
Lazy Leveling	$O(e^{-x})$	$O(s)$	$O(1+R \cdot (\log_R(N)-1))$	$O(R + \log_R(N))$
Leveling	$O(e^{-x})$	$O(s)$	$O(\log_R(N))$	$O(R \cdot \log_R(N))$



DASlab @ Harvard SEAS


short range
Tiering
Lazy Leveling
Leveling
writes
DASlab
@ Harvard SEAS



long range
Tiering
Lazy Leveling
Leveling
writes

Tiering

Lazy Leveling

Leveling

Tiering

writes

Lazy Leveling

Leveling

Tiering

writes

Lazy Leveling

Leveling

short range

Tiering

writes

Lazy Leveling

writes & point

Leveling

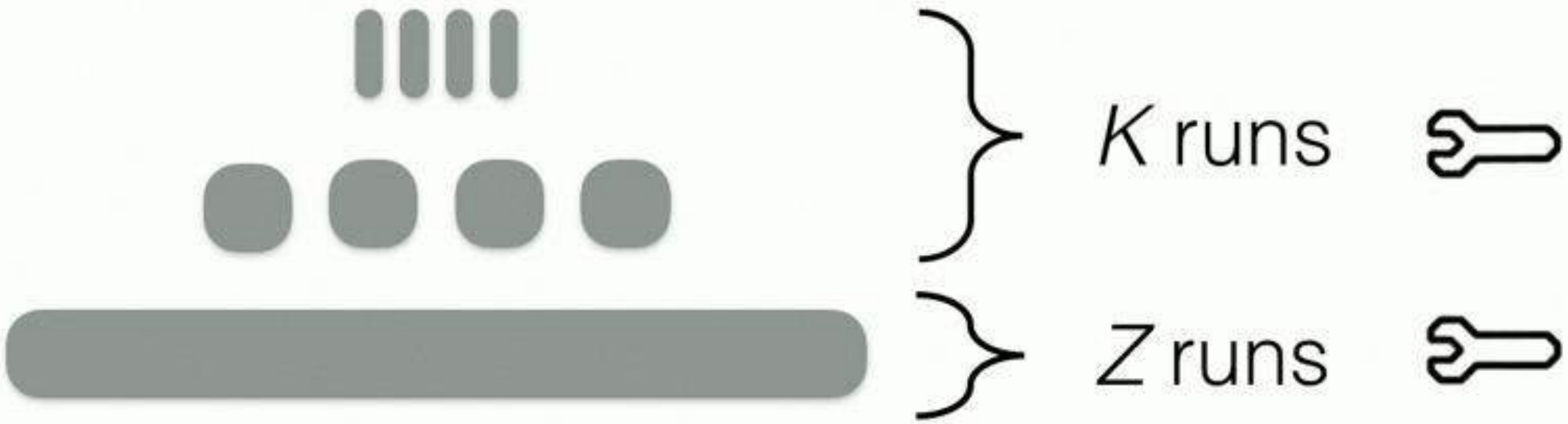
short range

Lazy Leveling

Tiering

Leveling

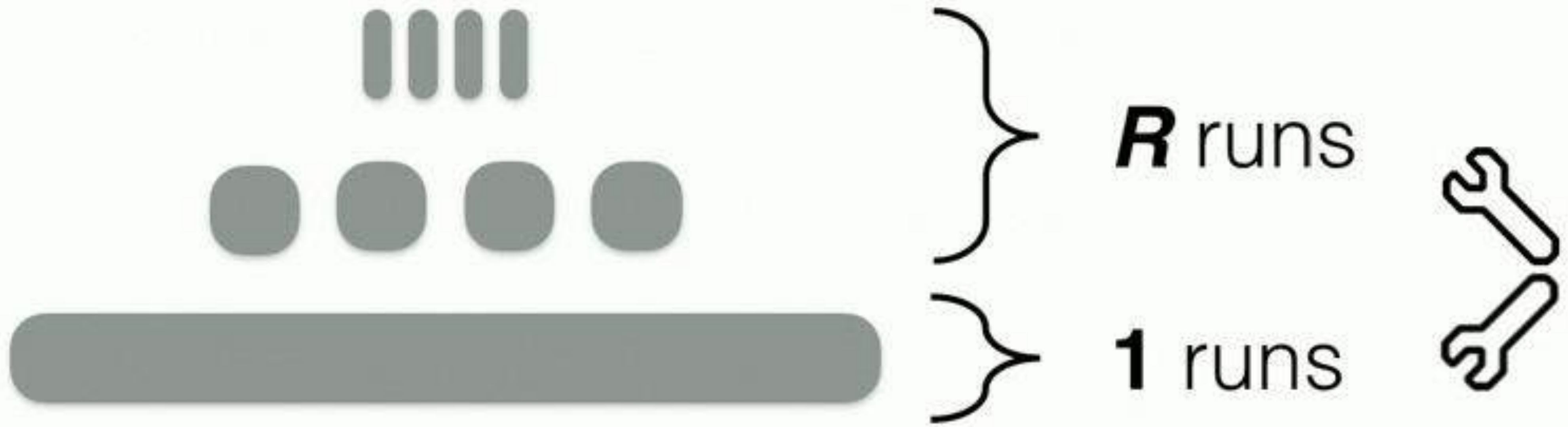
Fluid LSM-Tree



DASlab @ Harvard SEAS

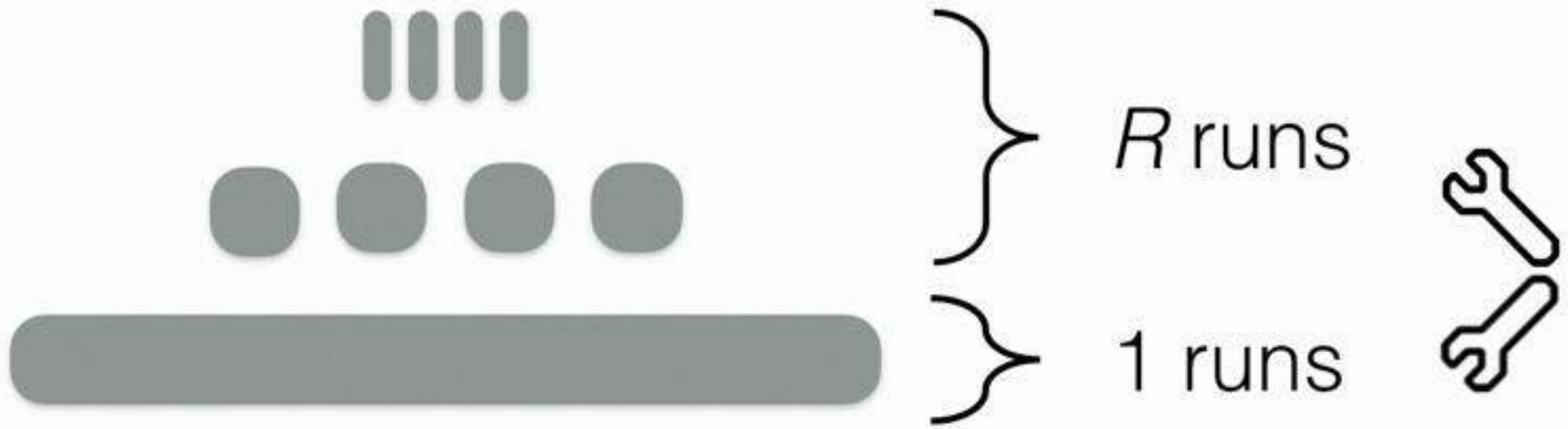
Fluid LSM-Tree

Lazy Leveling



point **long range** **short range** **writes**

Lazy Leveling



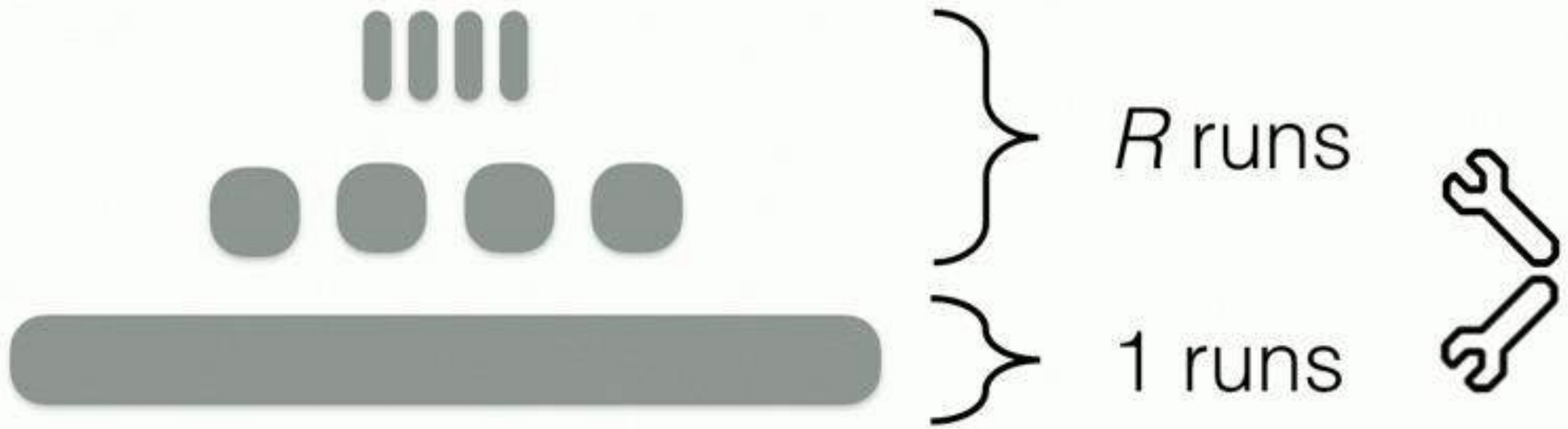
point

long range

optimize
short range

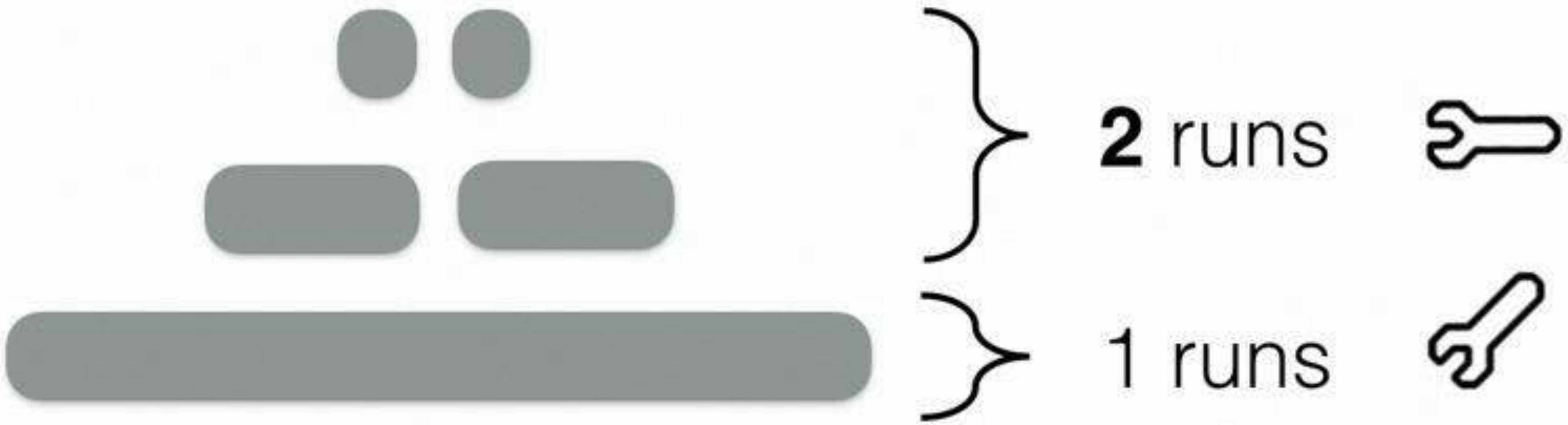
writes

Lazy Leveling



point | long range | optimize **short range** | **writes**

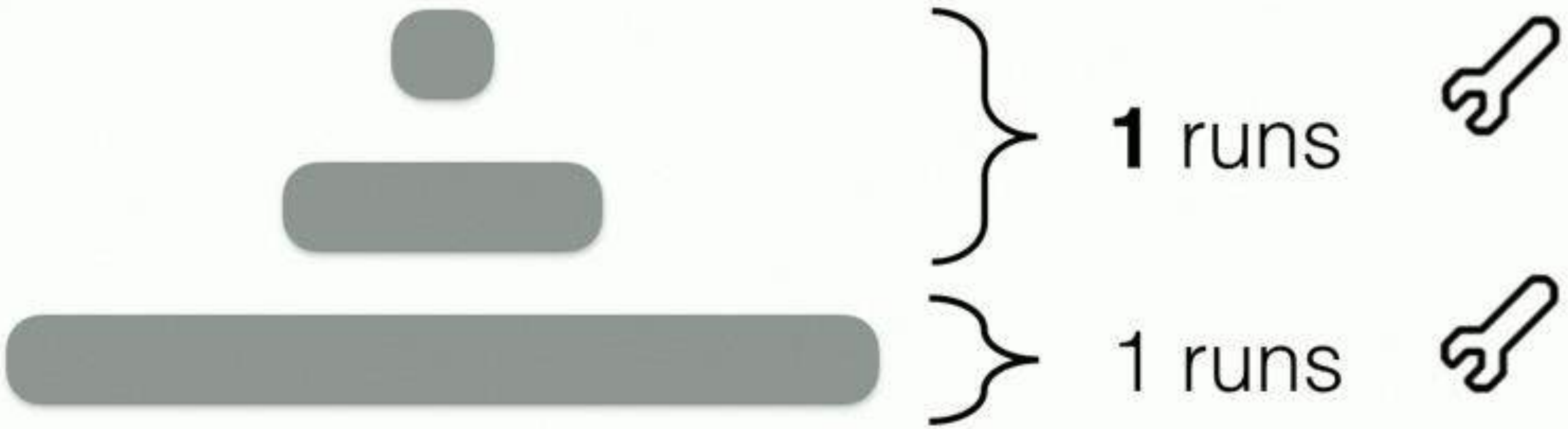
Lazy Leveling



optimize

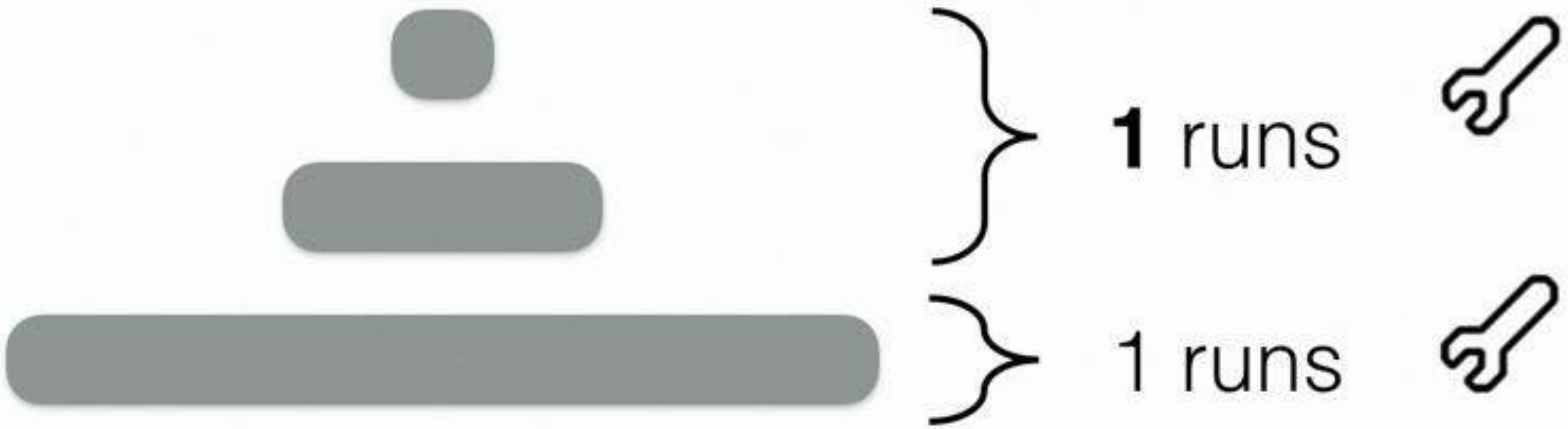
point | long range | **short range** | **writes**

Leveling



point | long range | short range | optimize **writes**

Leveling



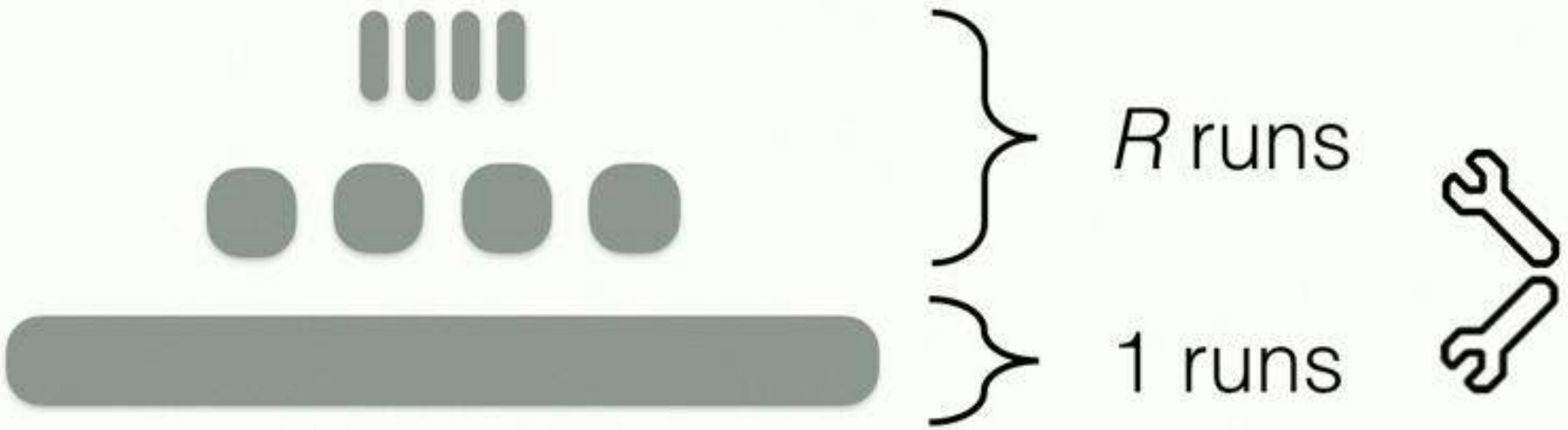
point

long range

short range

optimize
writes

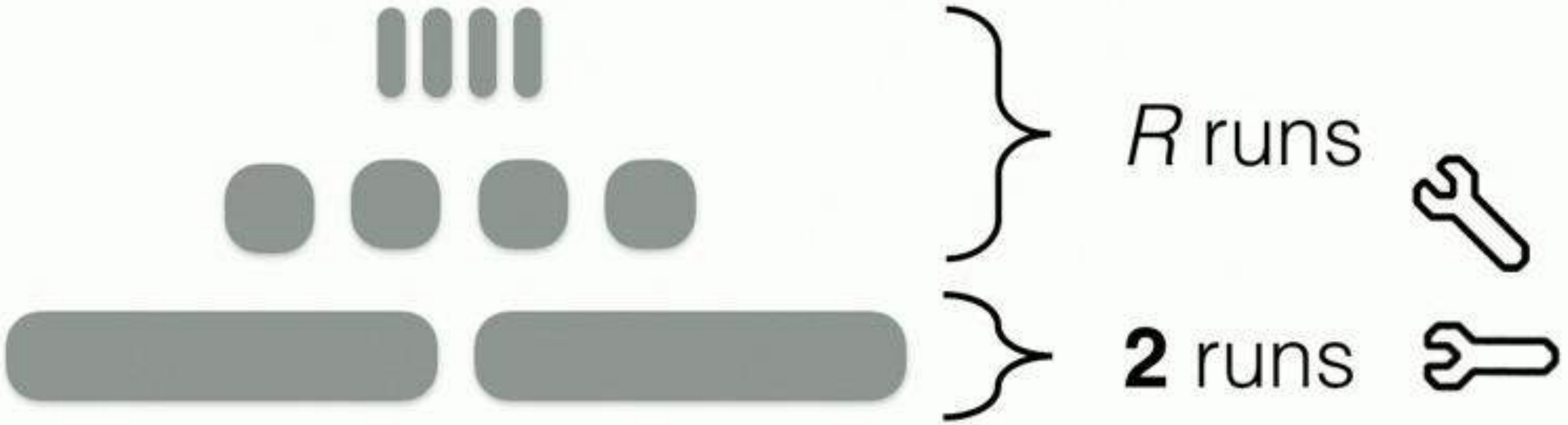
Lazy Leveling



optimize

point | long range | short range | writes

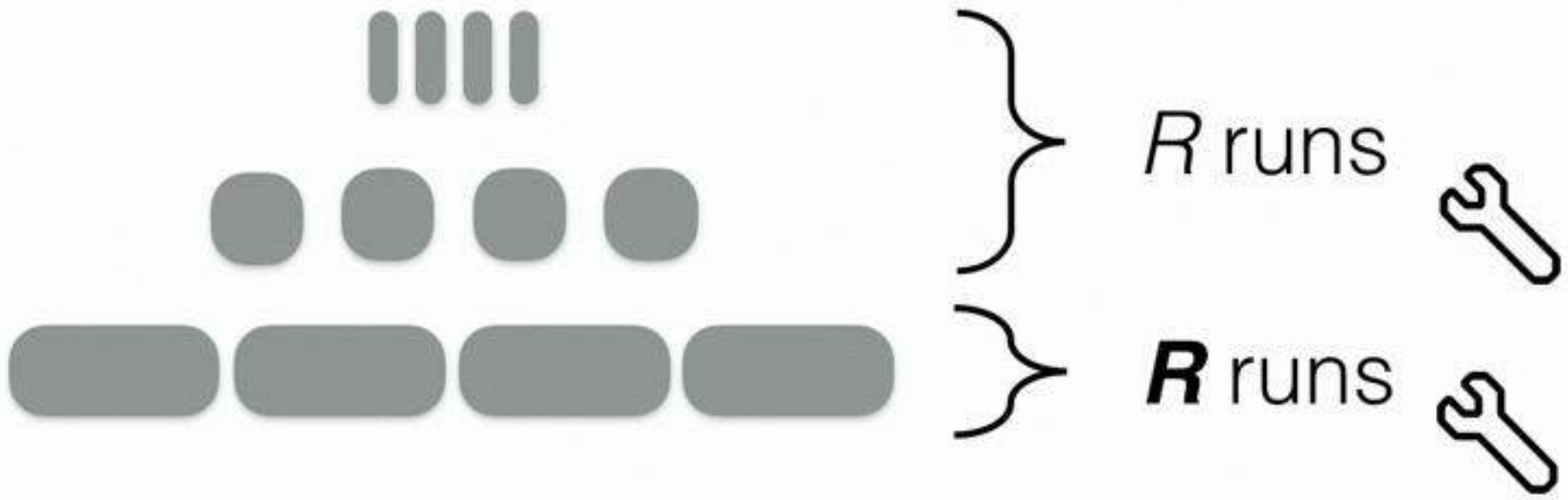
Lazy Leveling



DASlab @ Harvard SEAS

point | long range | short range | optimize **writes**

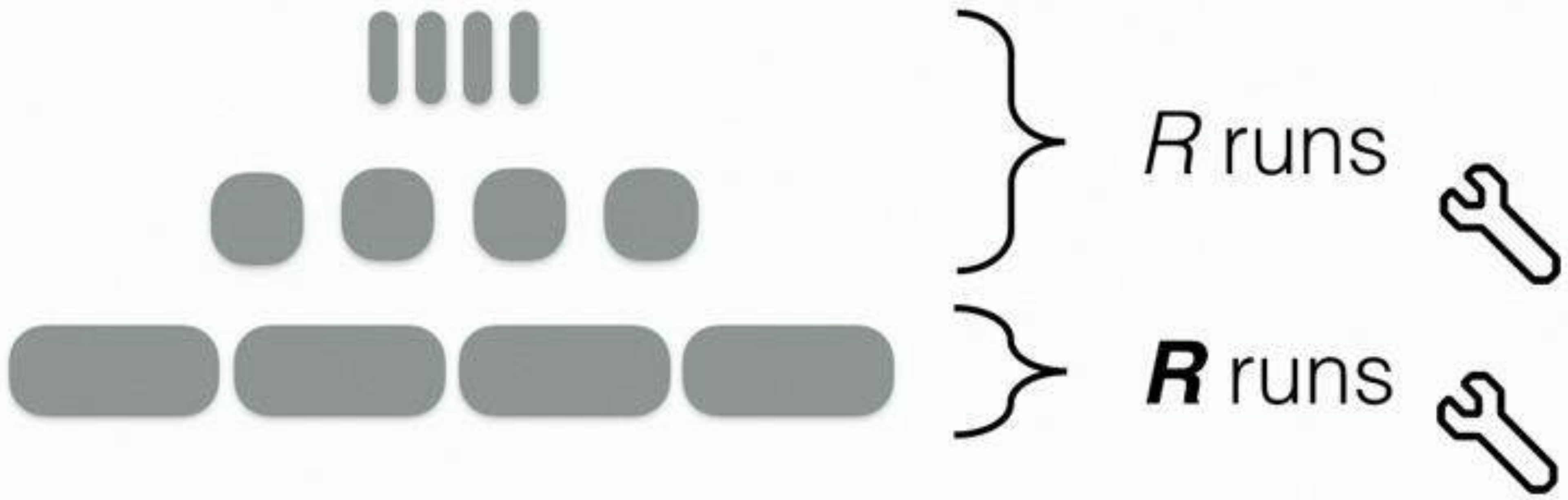
Tiering



optimize

point long range short range writes

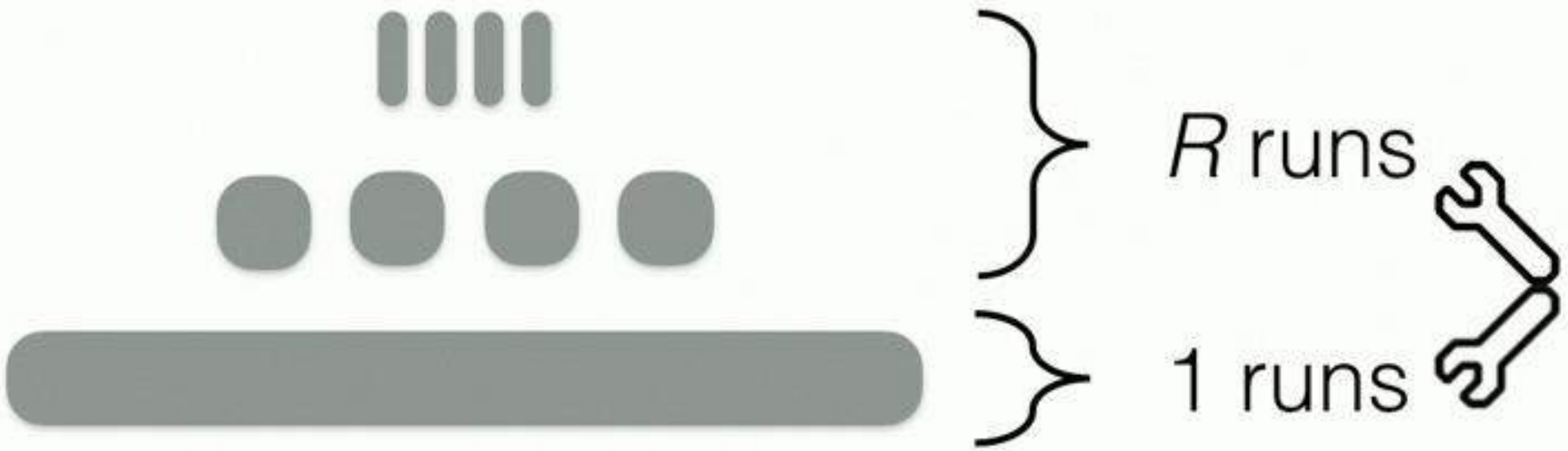
Tiering



optimize

point long range short range writes

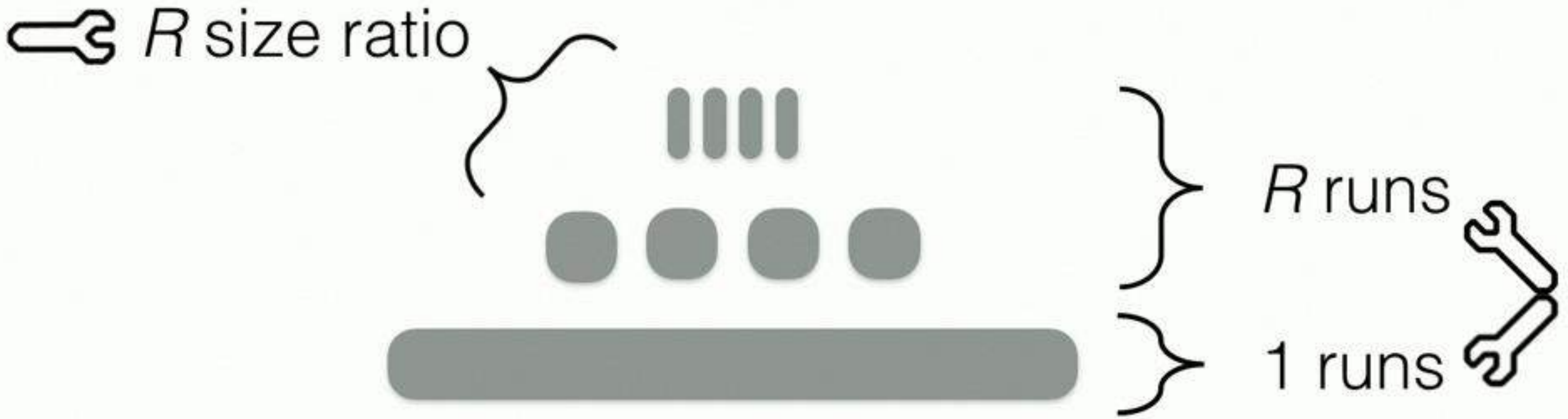
Lazy Leveling



optimize

point long range short range writes

Lazy Leveling



optimize

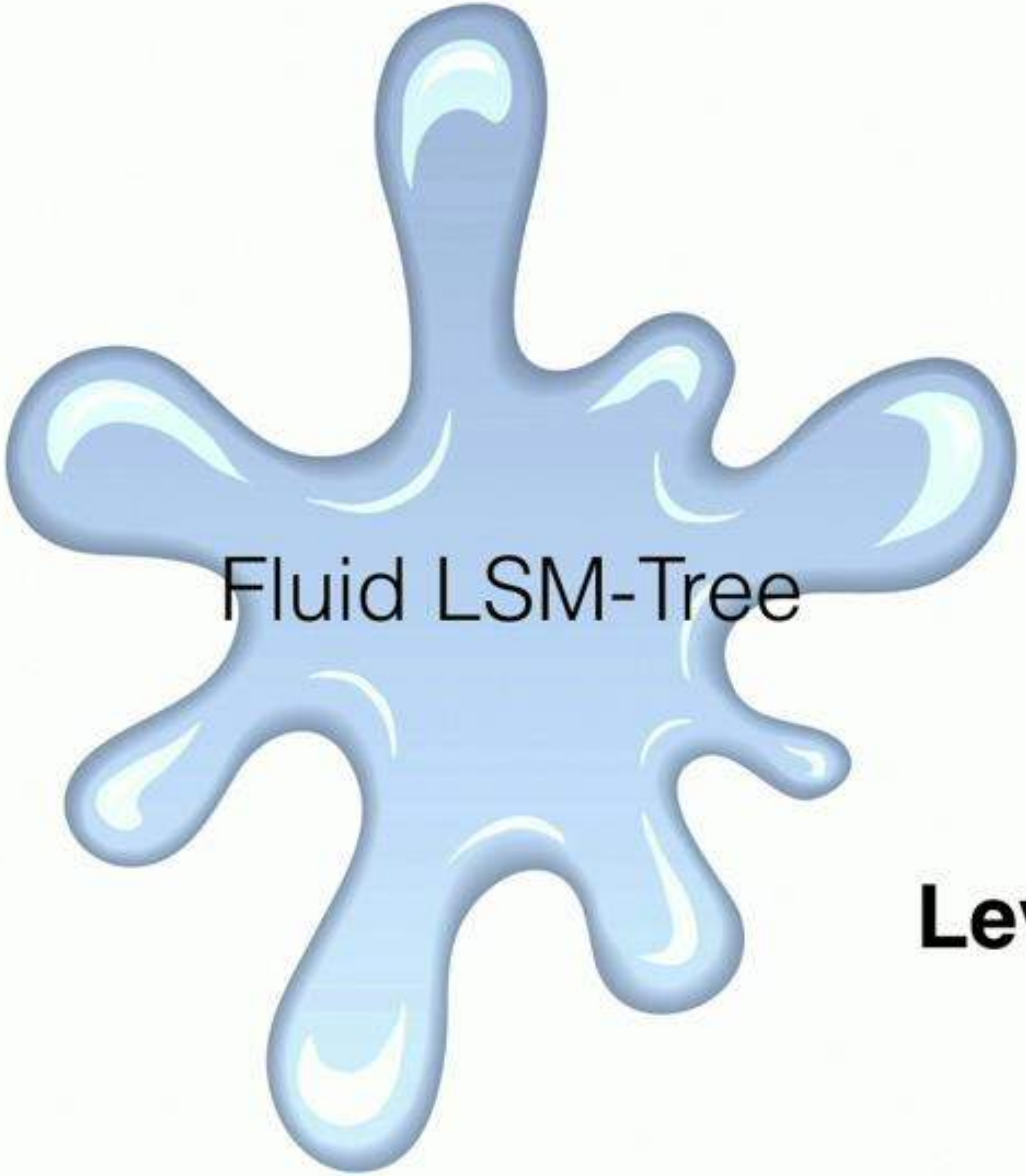
point long range short range writes

Lazy Leveling

Fluid LSM-Tree

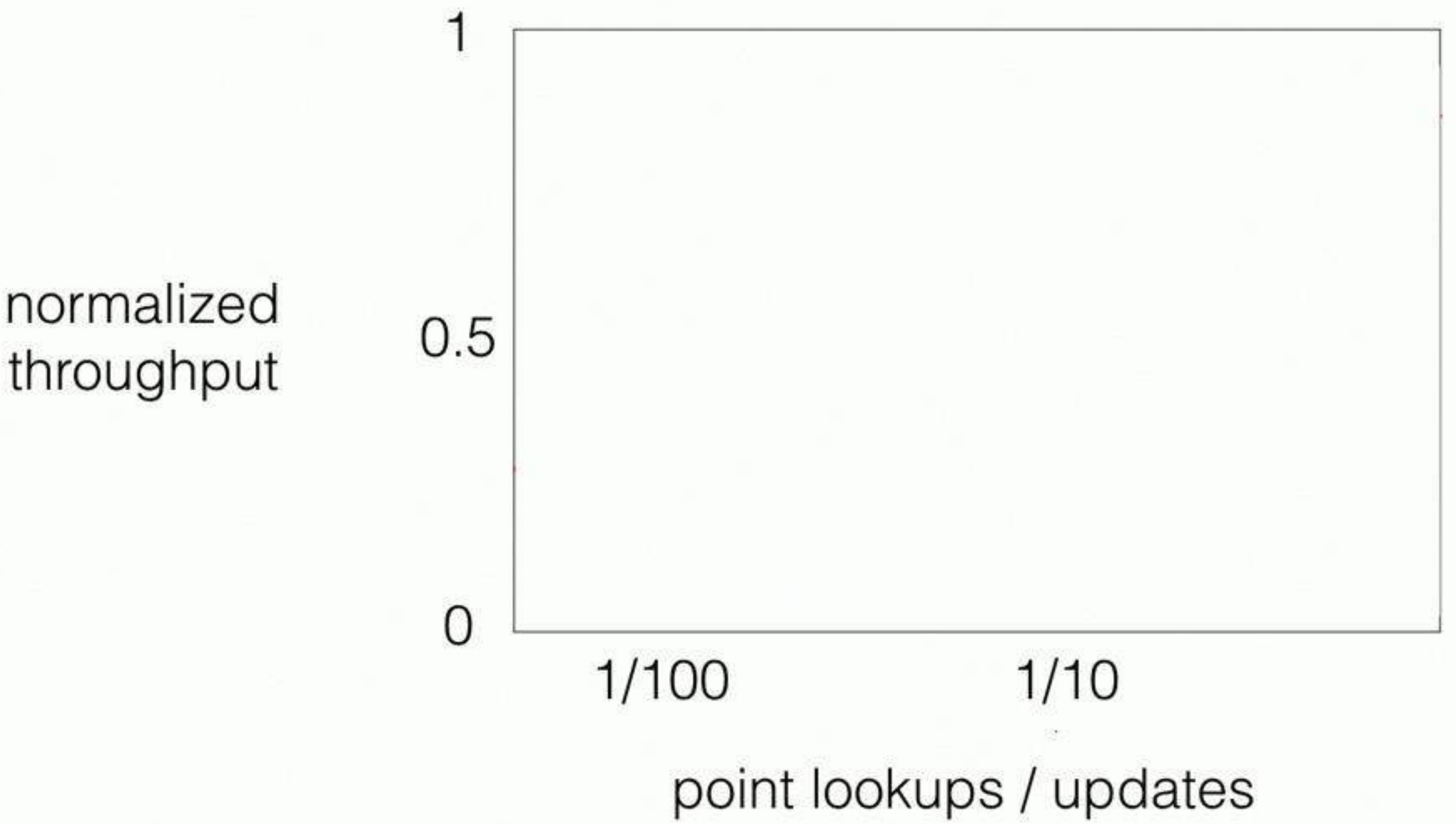
- *R* size ratio
- *K* runs at smaller levels
- *Z* runs at largest level

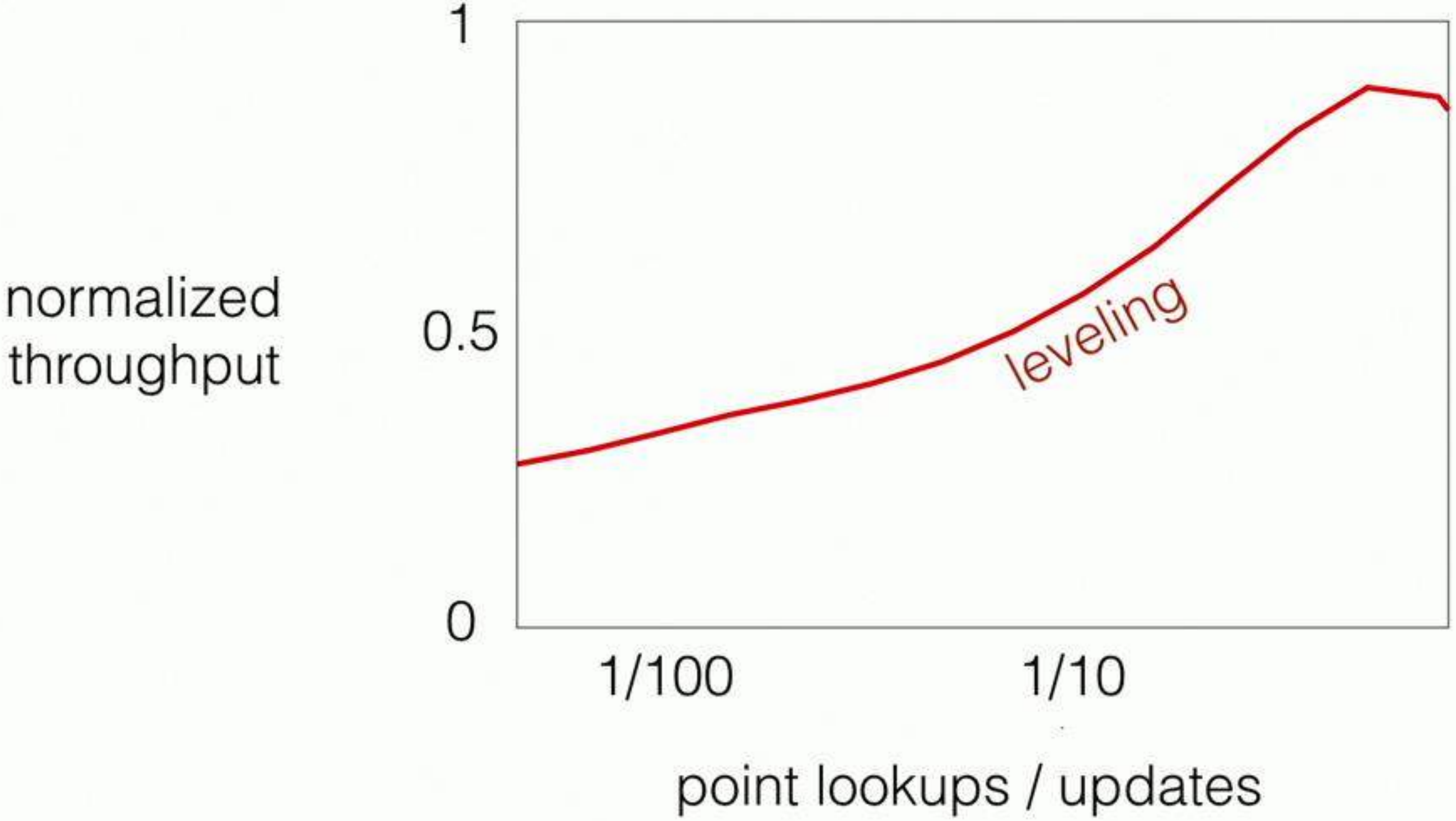
Lazy Leveling

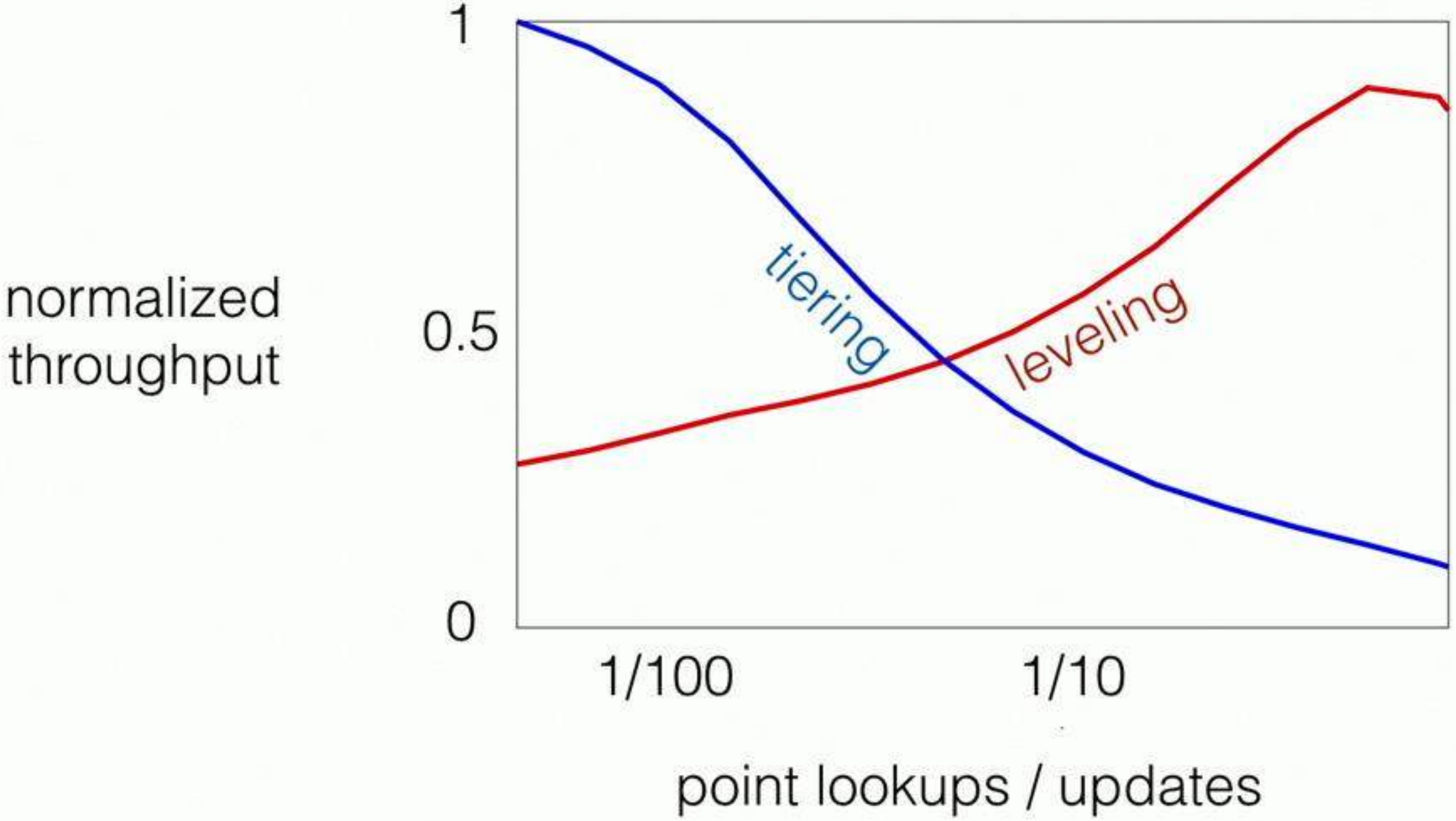


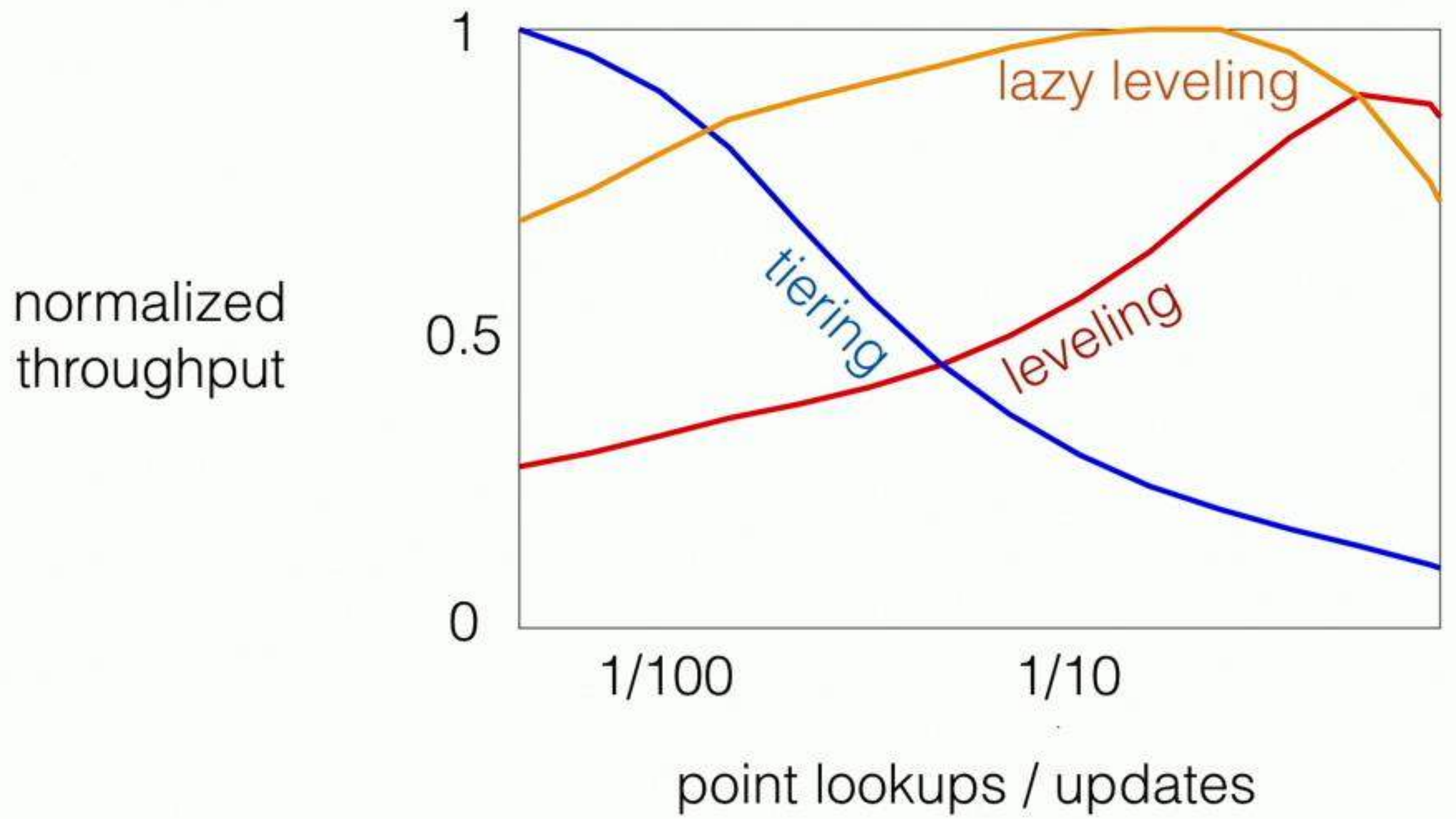
Tiering

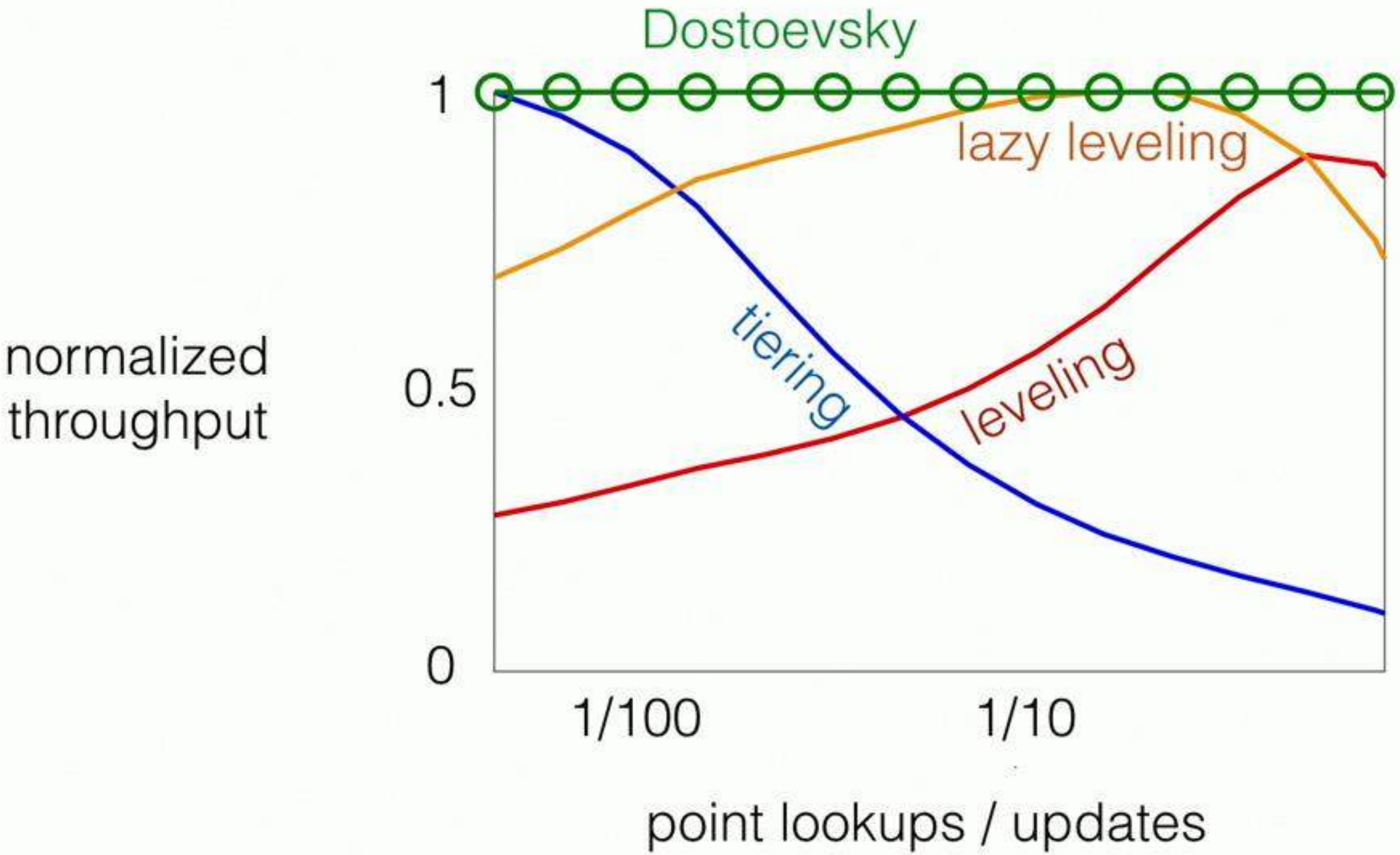
Leveling

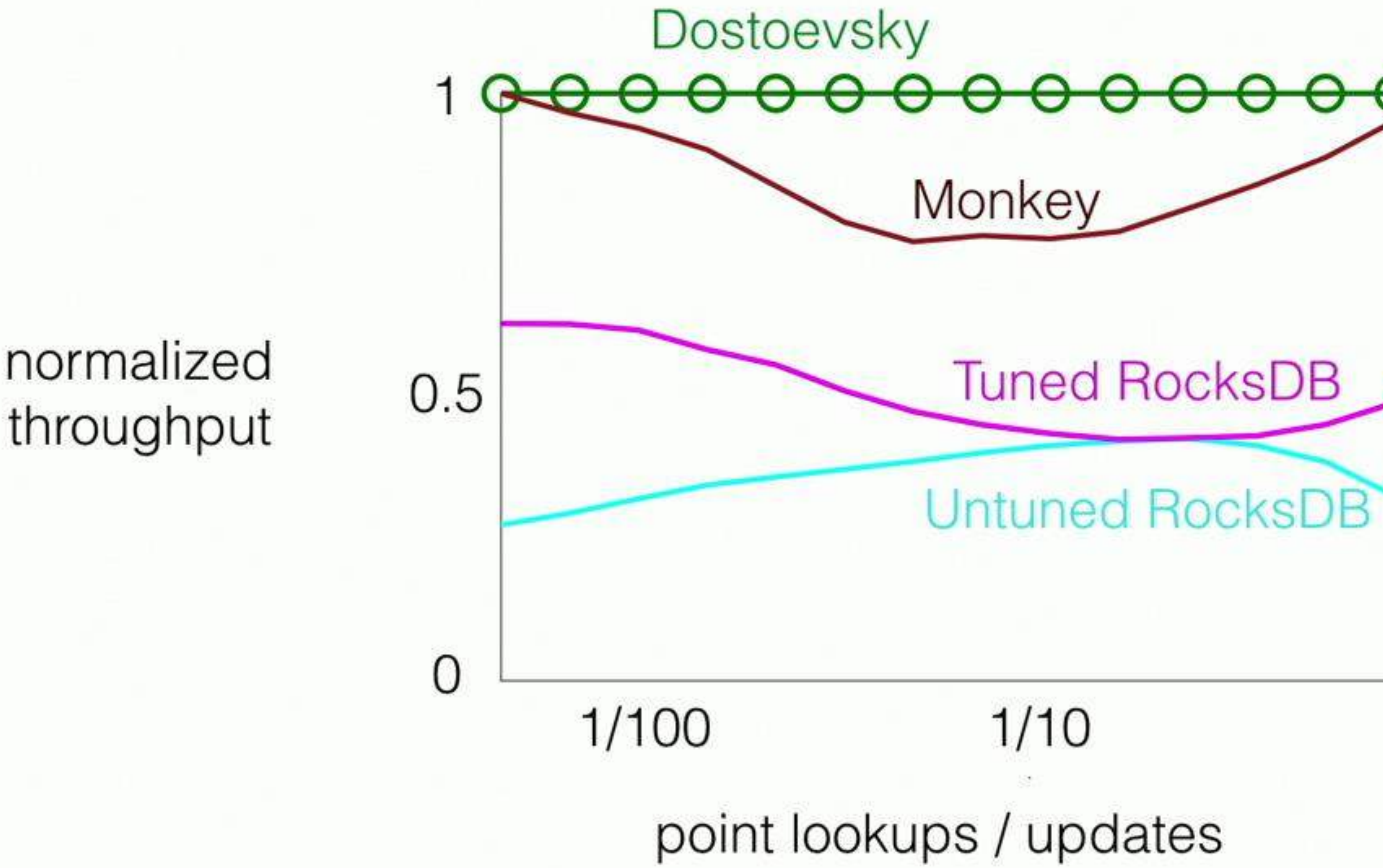

normalized throughput
1
0.5
0
1/100
1/10
point lookups / updates


normalized
throughput
1
0.5
0
leveling
1/100
1/10
point lookups / updates


normalized
throughput
1
0.5
0
tiering
leveling
1/100
1/10
point lookups / updates


normalized
throughput
1
0.5
0
lazy leveling
tiering
leveling
1/100
1/10
point lookups / updates


Dostoevsky
1
0.5
0
lazy leveling
tiering
leveling
normalized
throughput
1/100
1/10
point lookups / updates


Dostoevsky
1
Monkey
normalized
throughput
0.5
Tuned RocksDB
Untuned RocksDB
0
1/100
1/10
point lookups / updates

New Work

Lazy Leveling

writes

$O(\ \boldsymbol{R} + \mathbf{log}_{\boldsymbol{R}}(\boldsymbol{N})\ )$

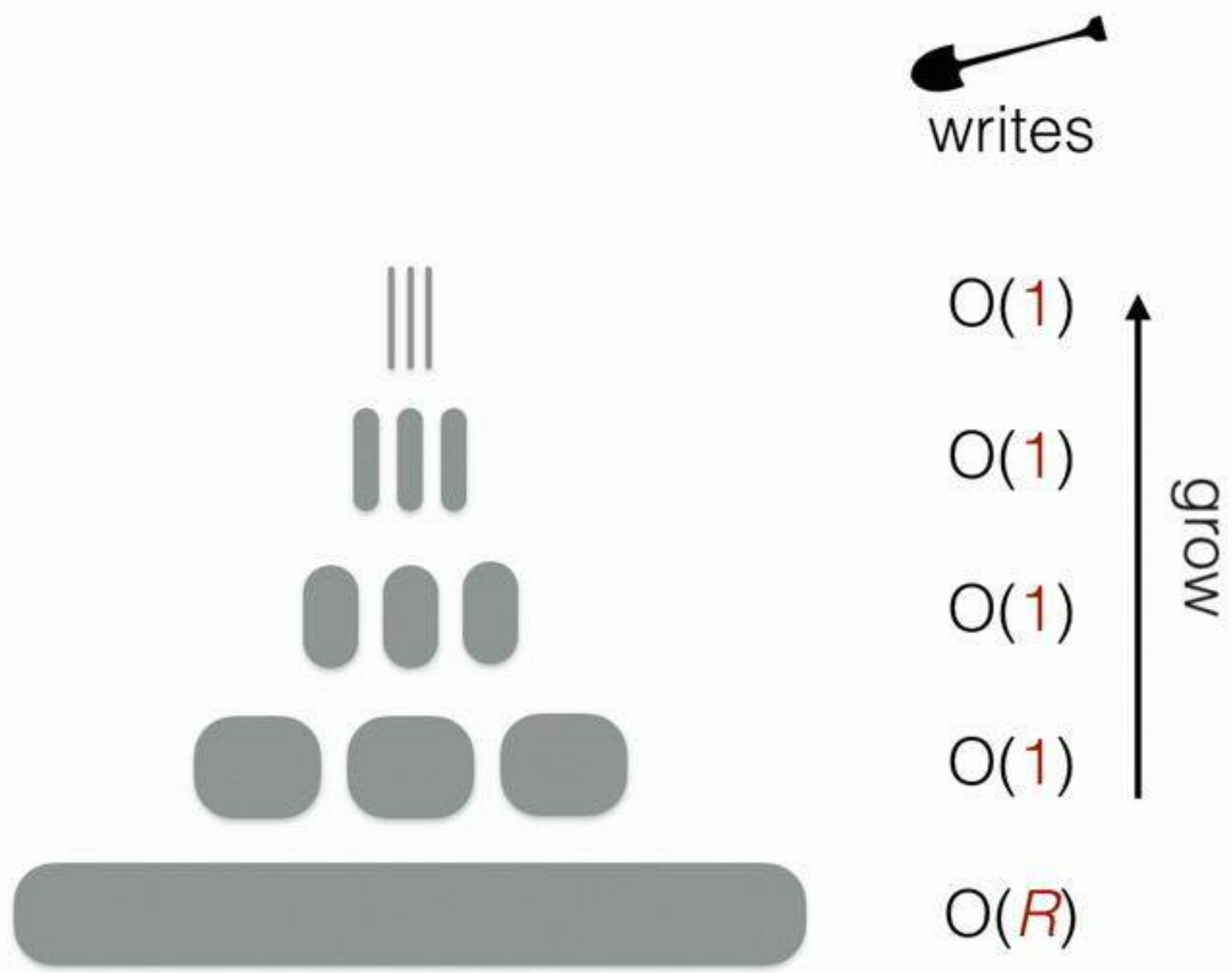
Lazy Leveling

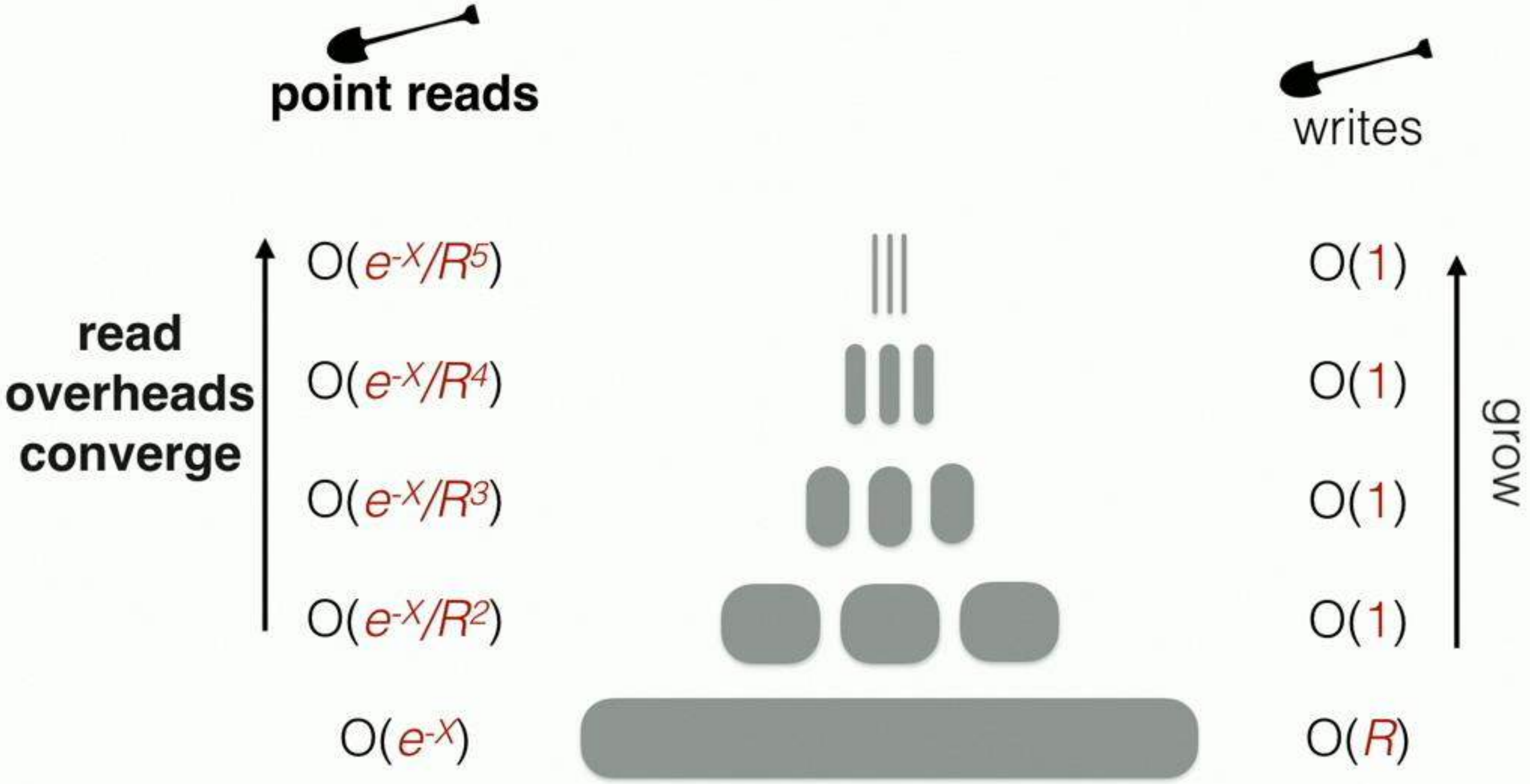
writes

O(1)

O(1)

O(R)

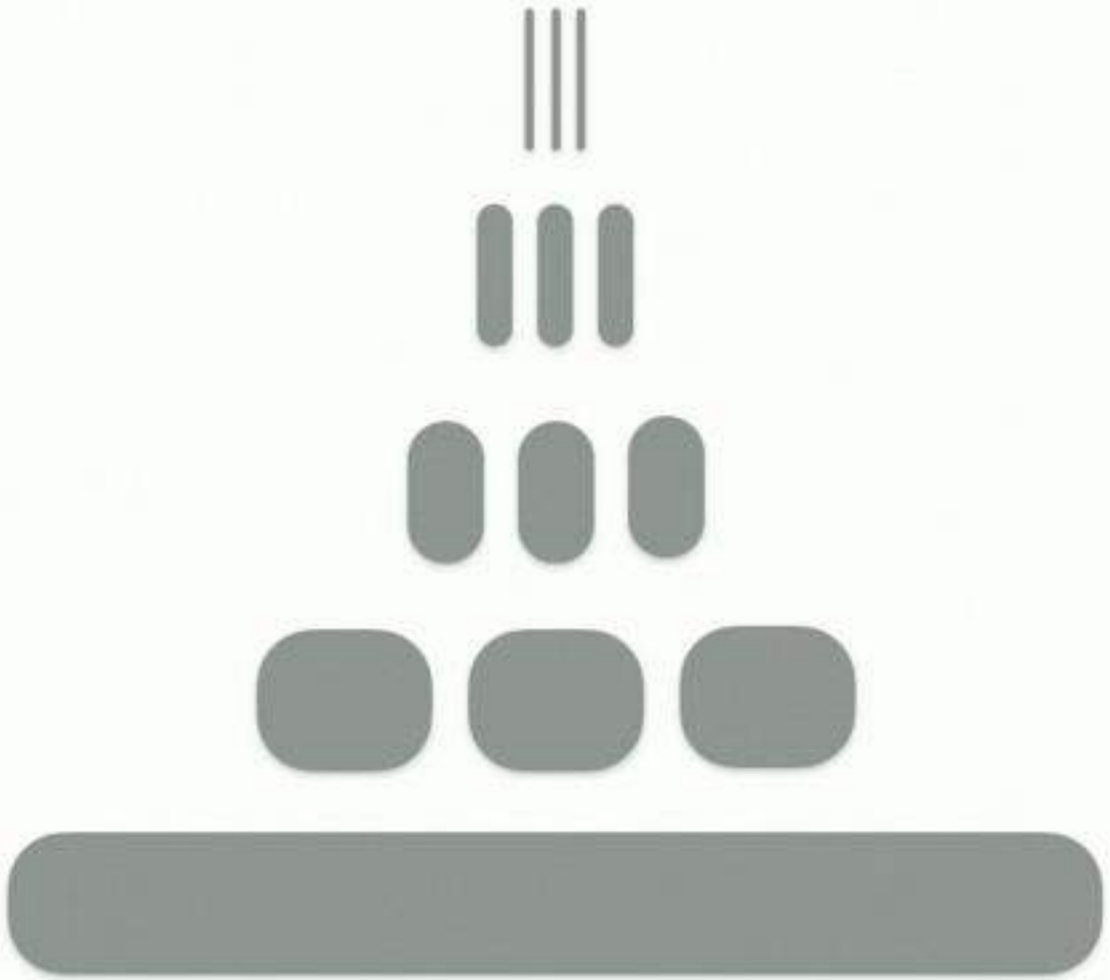

writes
O(1)
O(1)
O(1)
O(1)
O(R)
grow


point reads
writes
read overheads converge
grow
$O(e^{-X}/R^5)$
$O(e^{-X}/R^4)$
$O(e^{-X}/R^3)$
$O(e^{-X}/R^2)$
$O(e^{-X})$
$O(1)$
$O(1)$
$O(1)$
$O(1)$
$O(R)$

memory	point reads	writes
$O(X/R^4)$	$O(e^{-X}/R^5)$	$O(1)$
$O(X/R^3)$	$O(e^{-X}/R^4)$	$O(1)$
$O(X/R^2)$	$O(e^{-X}/R^3)$	$O(1)$
$O(X/R^1)$	$O(e^{-X}/R^2)$	$O(1)$
$O(X)$	$O(e^{-X})$	$O(R)$

converge

grow



DASlab
@ Harvard SEAS

memory	point reads	writes
$O(X/R^4)$	$O(e^{-X}/R^5)$	$O(1)$
$O(X/R^3)$	$O(e^{-X}/R^4)$	$O(1)$
$O(X/R^2)$	$O(e^{-X}/R^3)$	$O(1)$
$O(X/R^1)$	$O(e^{-X}/R^2)$	$O(1)$
$O(X)$	$O(e^{-X})$	$O(R)$

growing as $O(\log_R(N))$

exponentially diminishing returns

memory	point reads	writes
$O(X/R^4)$	$O(e^{-X}/R^5)$	$O(1)$
$O(X/R^3)$	$O(e^{-X}/R^4)$	$O(1)$
$O(X/R^2)$	$O(e^{-X}/R^3)$	$O(1)$
$O(X/R^1)$	$O(e^{-X}/R^2)$	$O(1)$
$O(X)$	$O(e^{-X})$	$O(R)$

growing as $O(\log_R(N)$

DASlab @ Harvard SEAS

Wacky SIGMOD19

Wacky: **A**morphous **C**alculable **K**e**y**-Value Store

DASlab @ Harvard SEAS

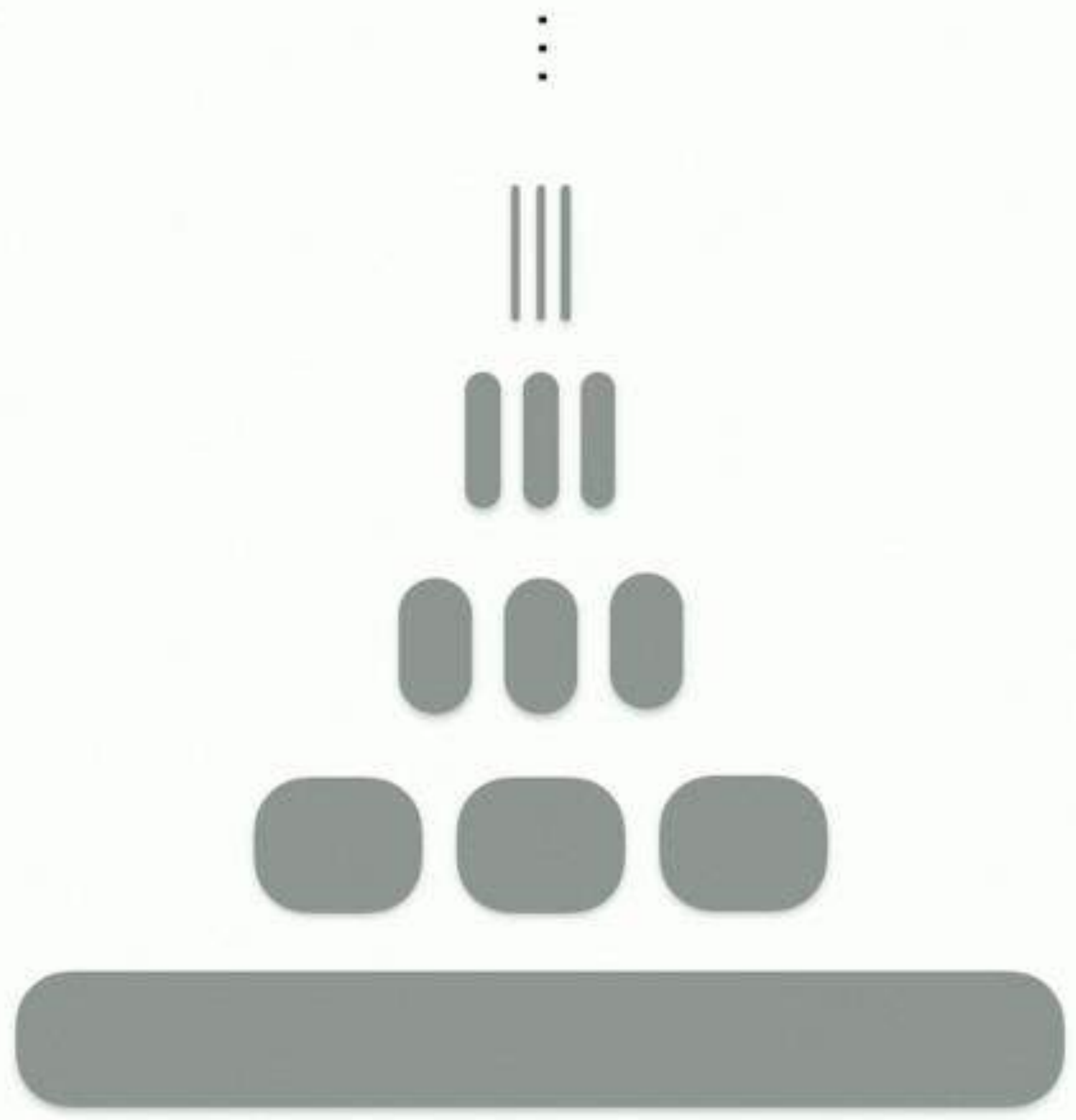
Wacky: **A**morphous **C**alculable **K**ey-Value Store

introducing the **Log-Structured Merge-Bush**

Wacky: **A**morphous **C**alculable **K**ey-Value Store

LSM-bush

DASlab @ Harvard SEAS



runs

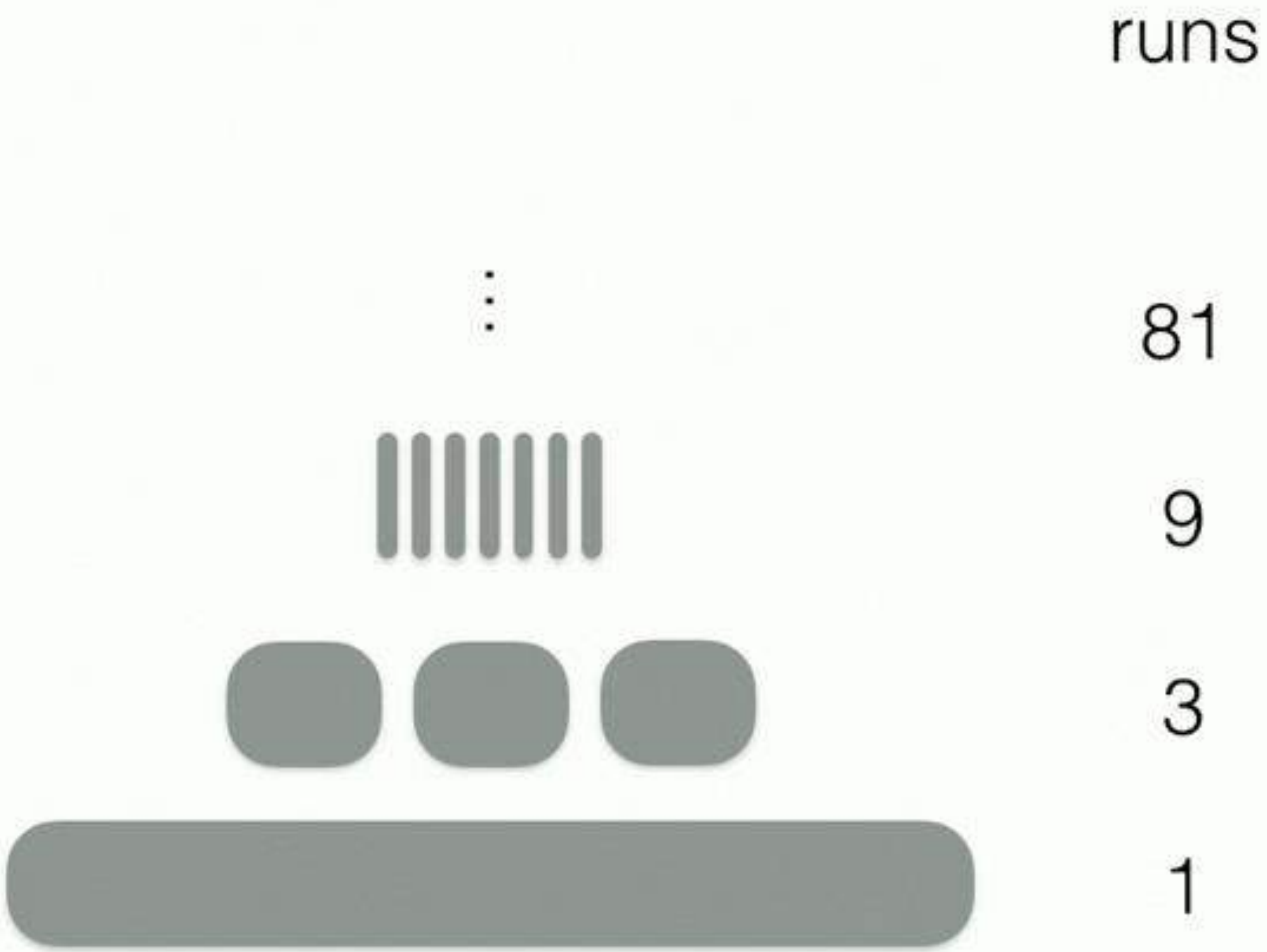
3

3

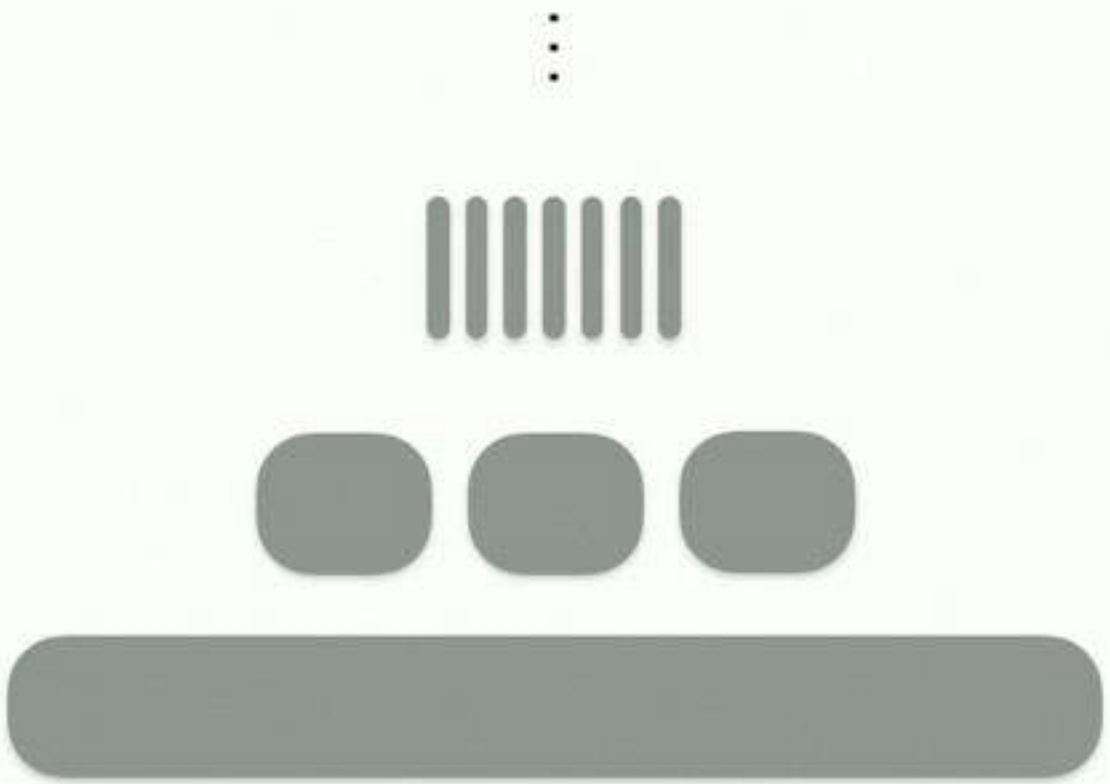
3

3

1


runs
81
9
3
1

runs

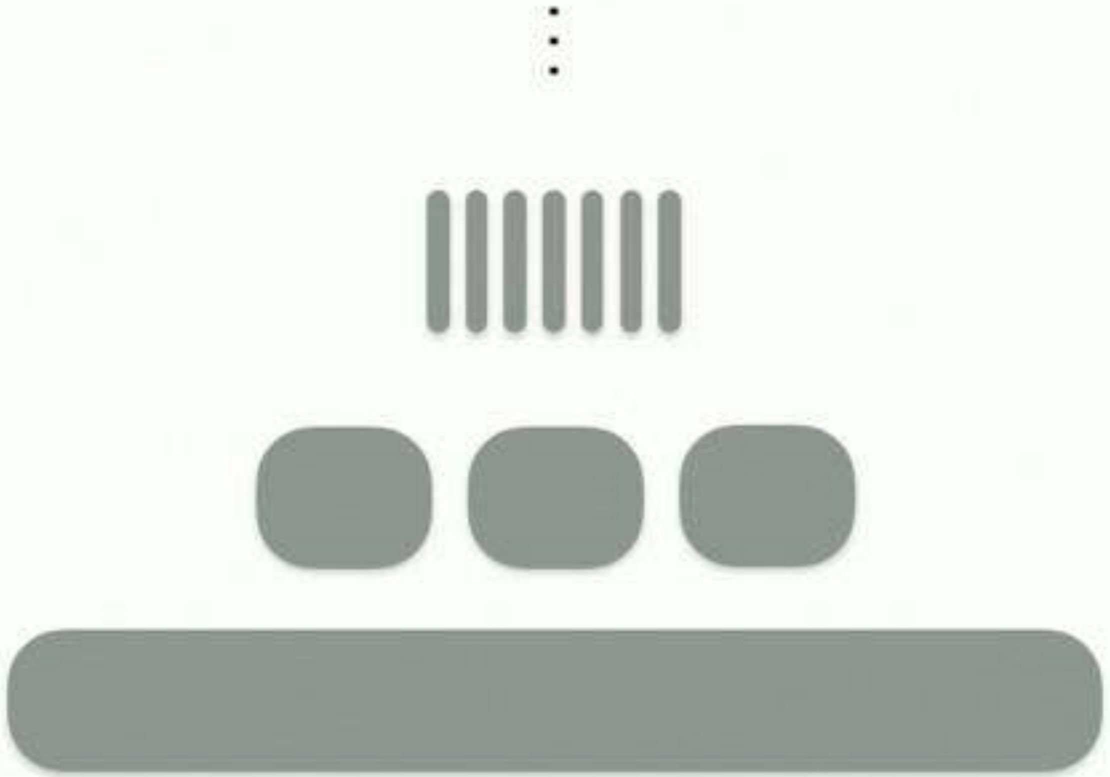


$O(R^{2^2})$

$O(R^{2^1})$

$O(R^{2^0})$

$O(1)$



$O(R^{2^2})$

$O(R^{2^1})$

$O(R^{2^0})$

$O(1)$

writes

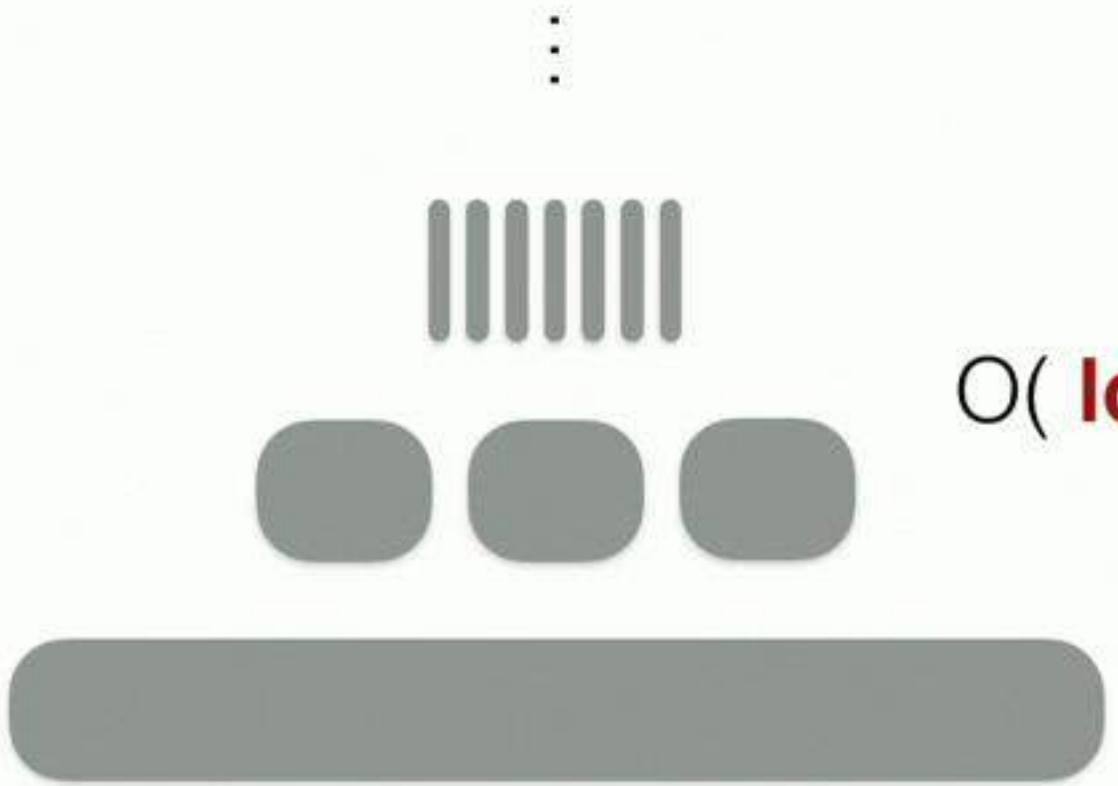
O(**$\log_2 \log_R(N)$**)

memory

$O(X)$

point reads

$O(e^{-X})$

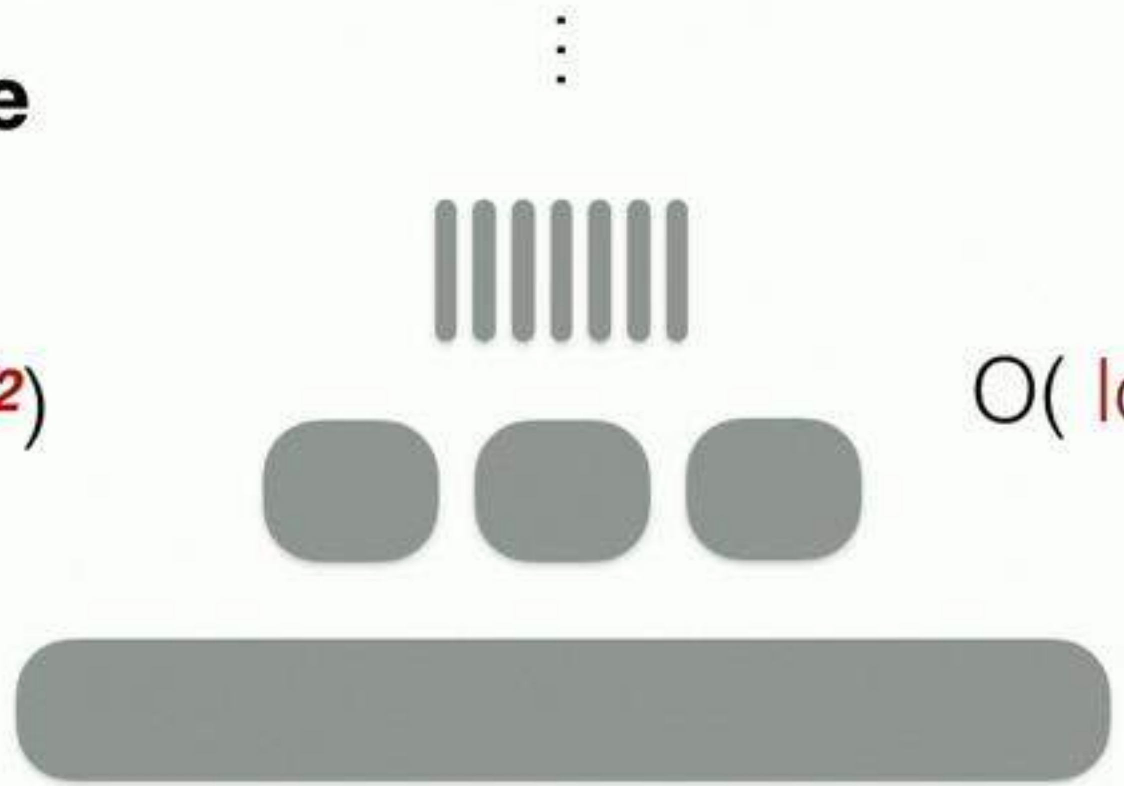


writes

$O(\,\mathbf{\log_2 \log_R(N)}\,)$

DASlab @ Harvard SEAS

memory	point reads	**range**	writes
$O(X)$	$O(e^{-X})$	$O(\boldsymbol{N^{1/2}})$	$O(\log_2 \log_R(N))$



Tiering

Lazy Leveling

LSM-bush

Leveling

Wacky generalizes

Tiering

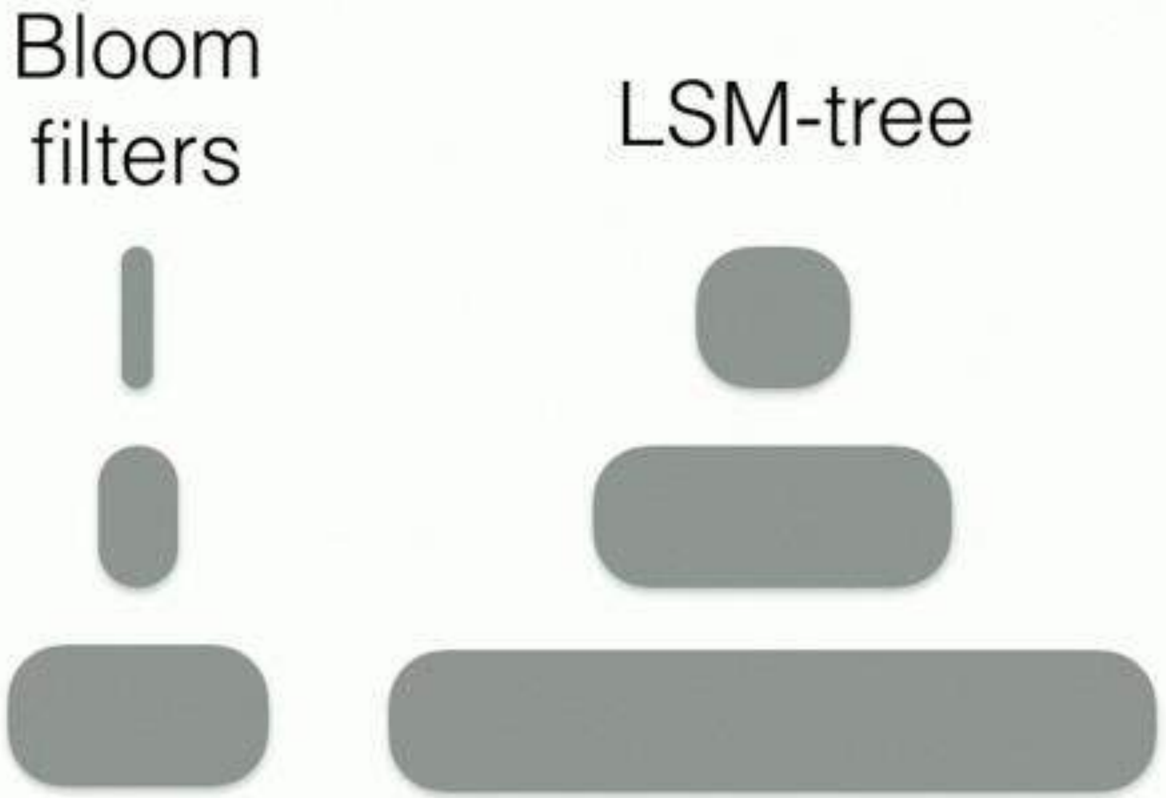
Lazy Leveling

LSM-bush

Leveling

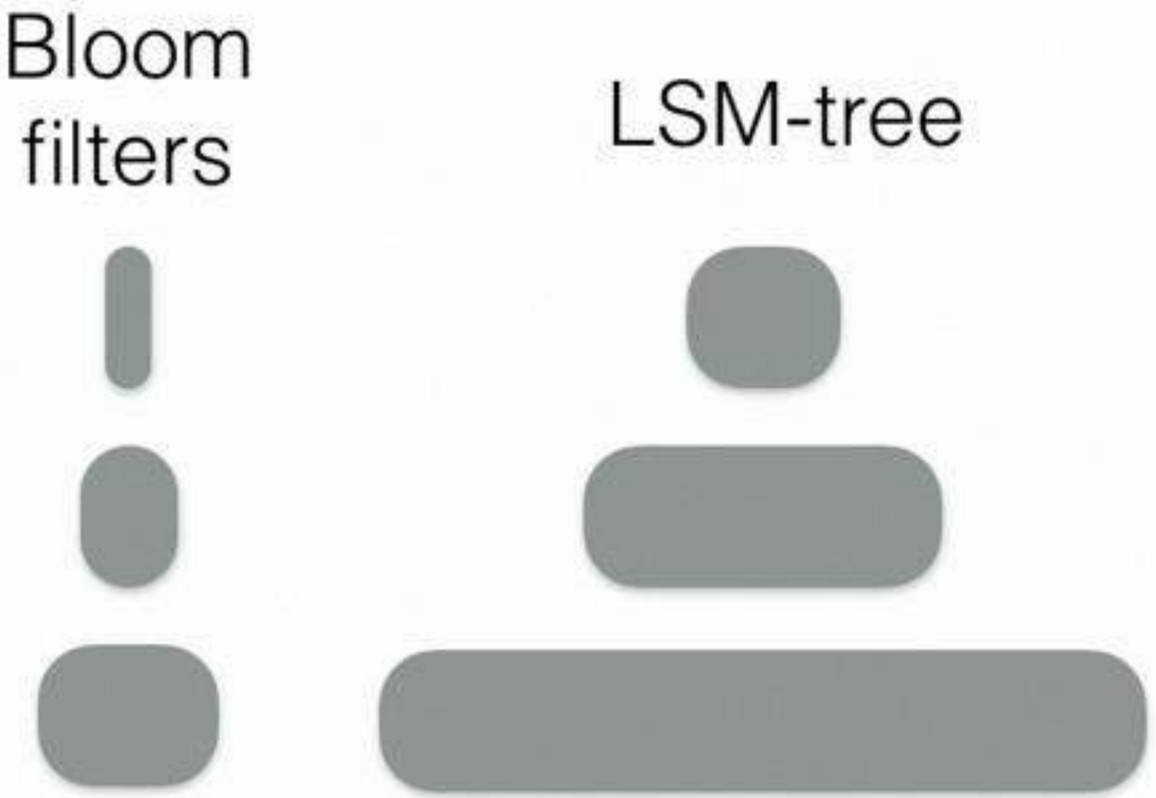
Conclusion

Conclusion



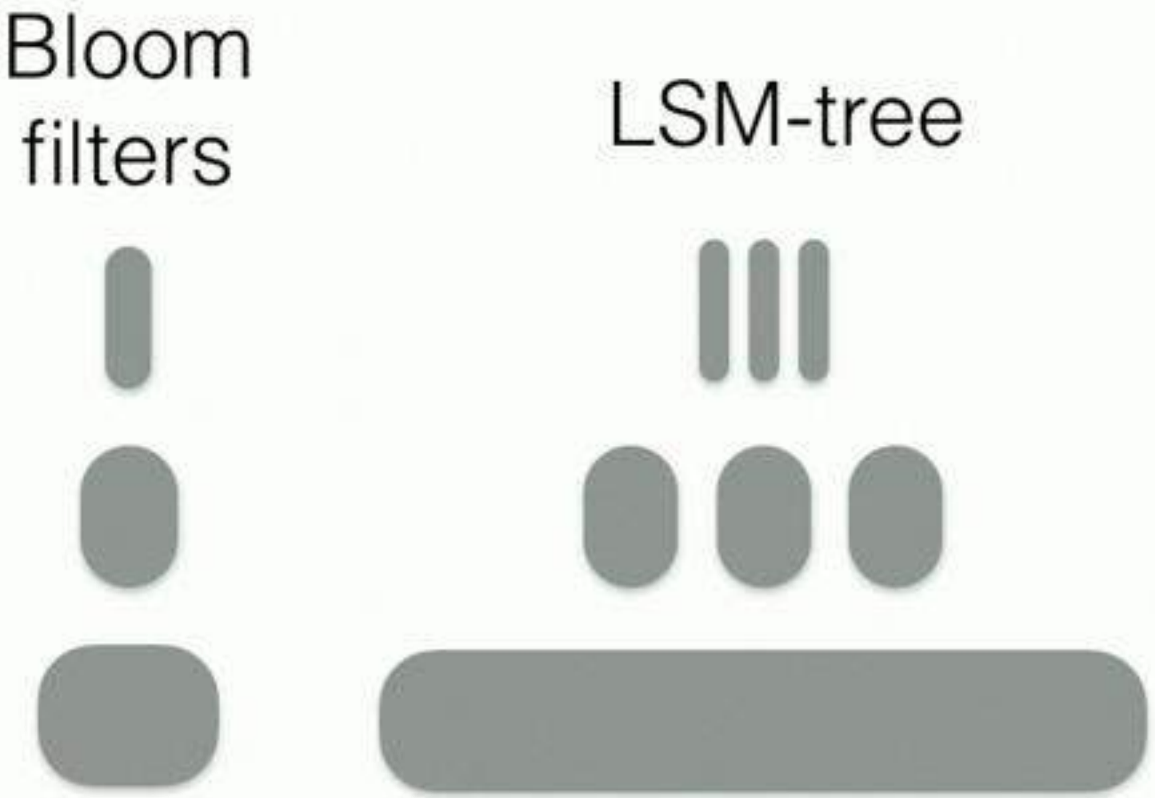
Conclusion

optimizes memory allocation



Conclusion

optimizes
memory
allocation



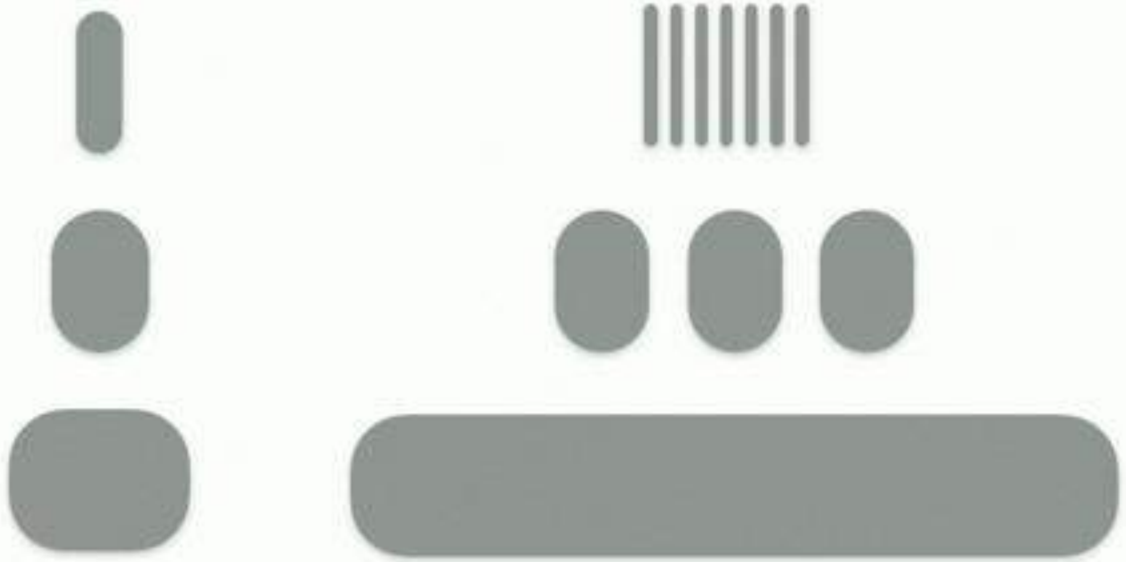
**lazy smaller
levels**

Conclusion

optimizes memory allocation

Bloom filters

LSM-tree

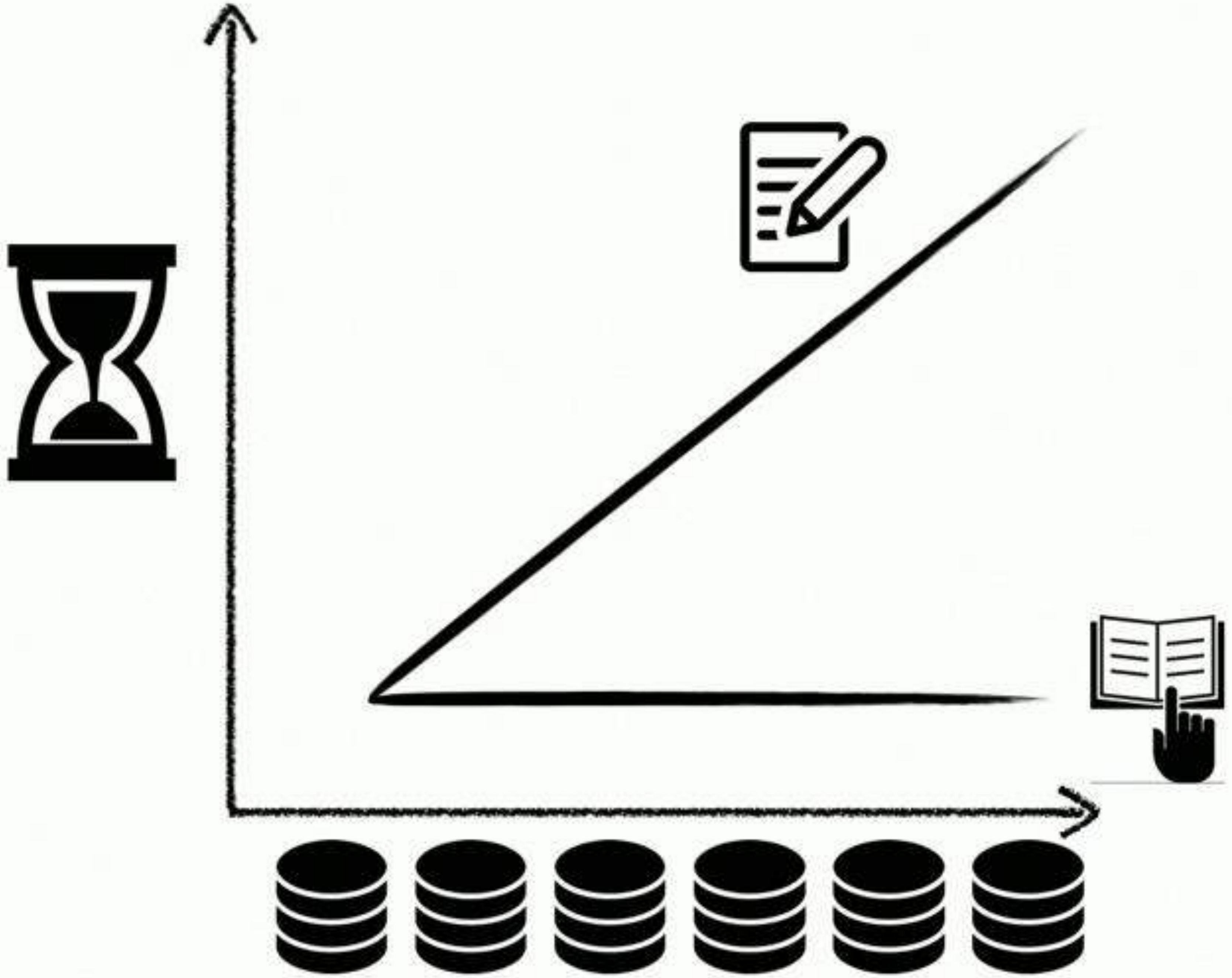


lazy smaller levels

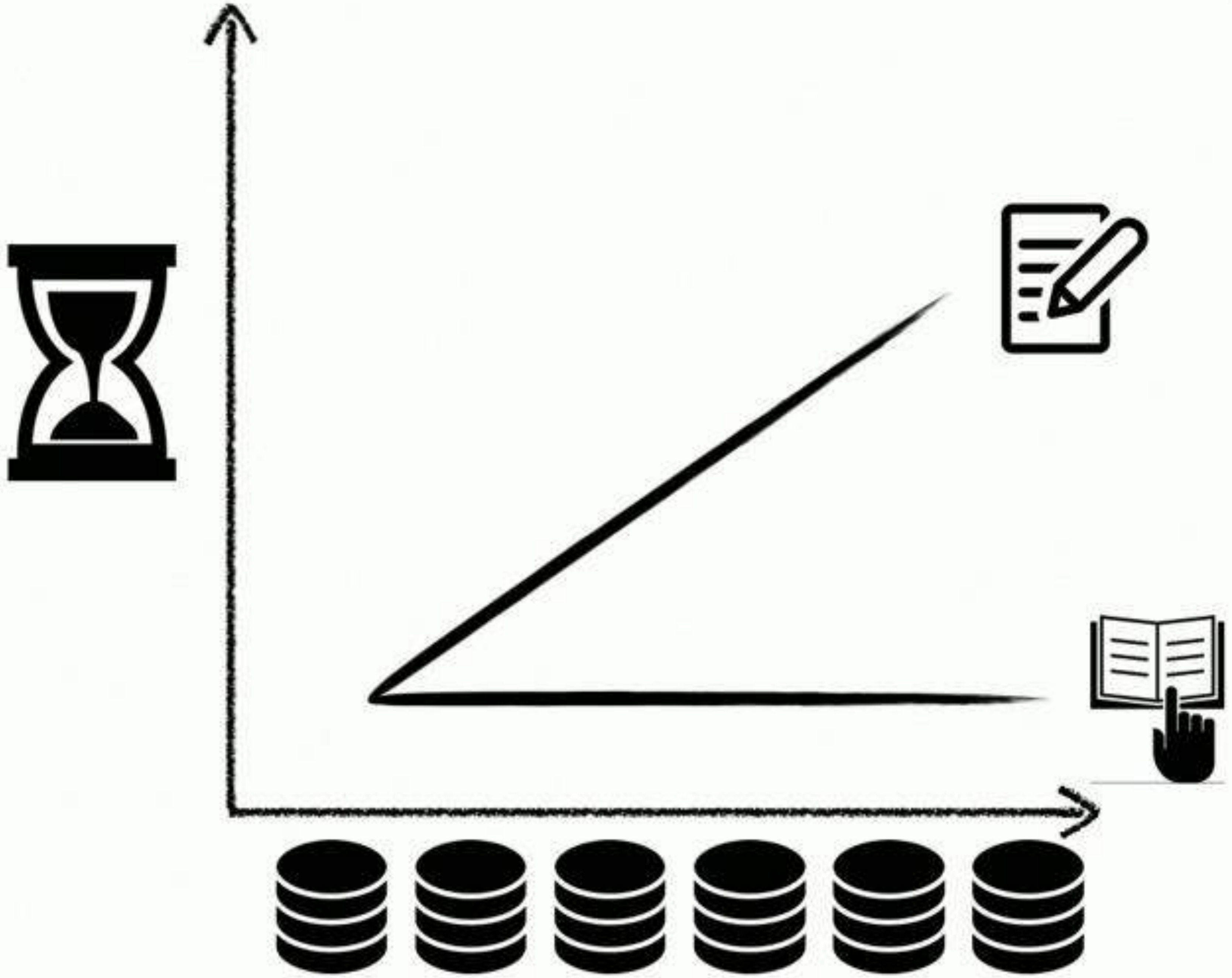
increasing ratios

DASlab @ Harvard SEAS

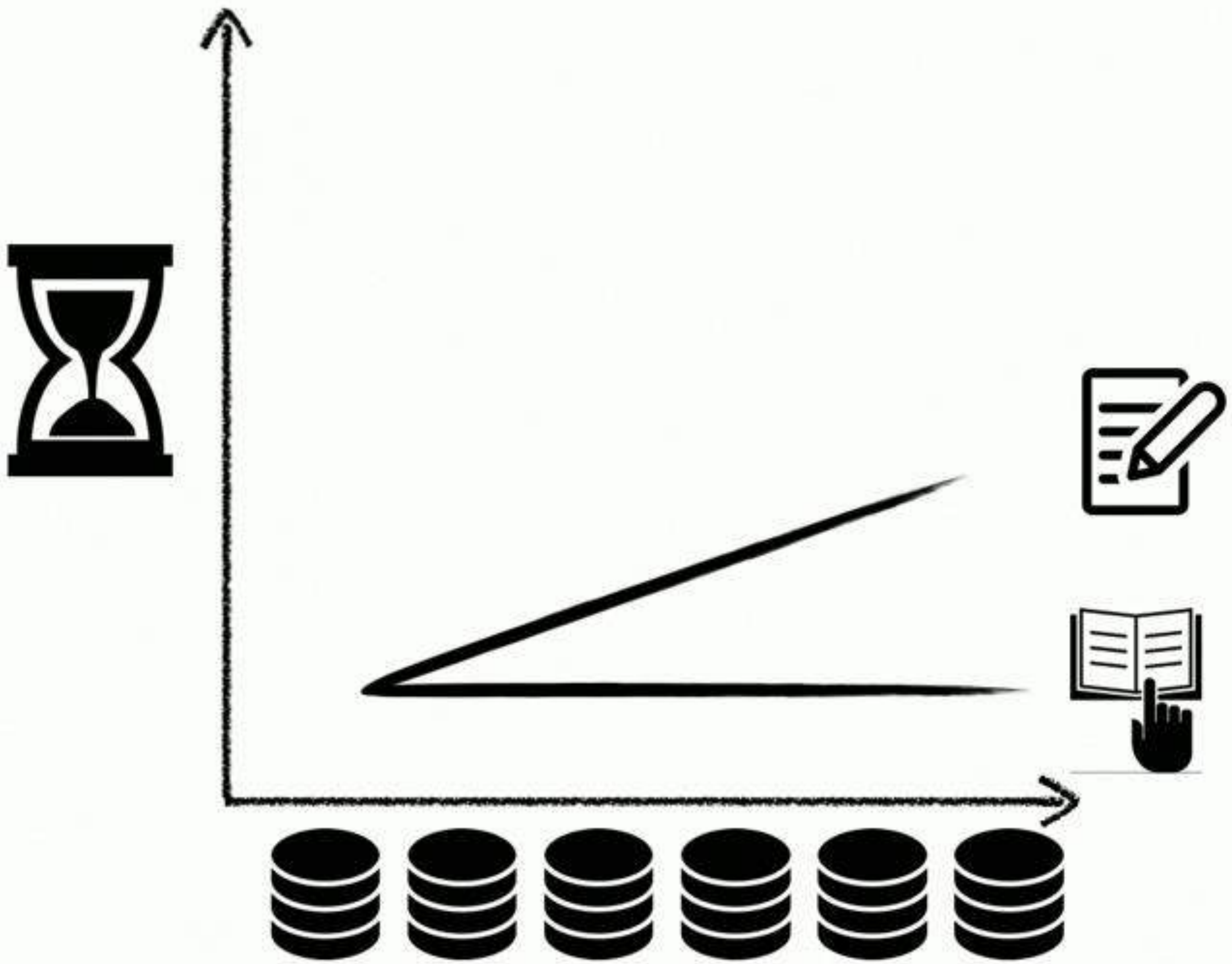
Conclusion



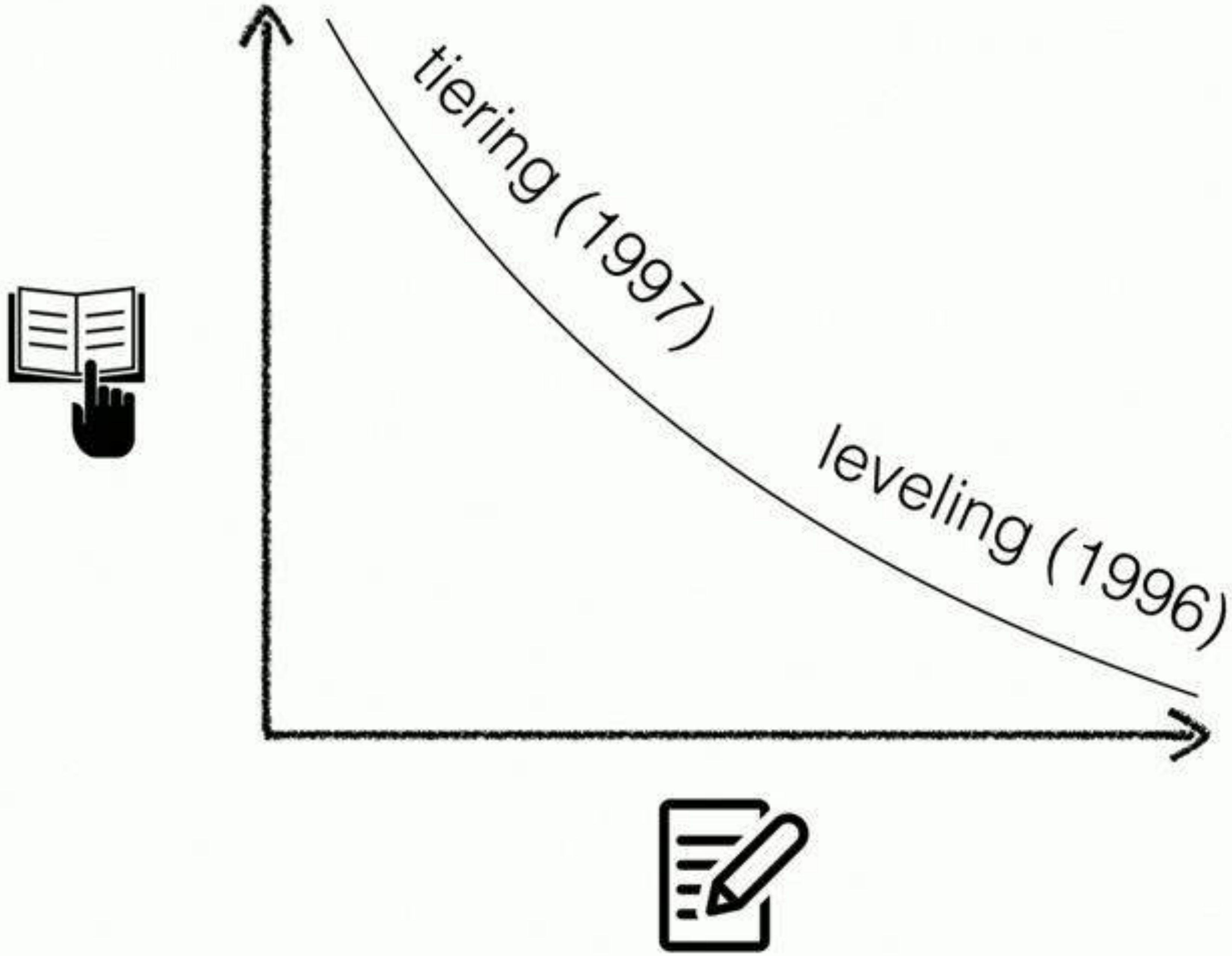
Conclusion



Conclusion

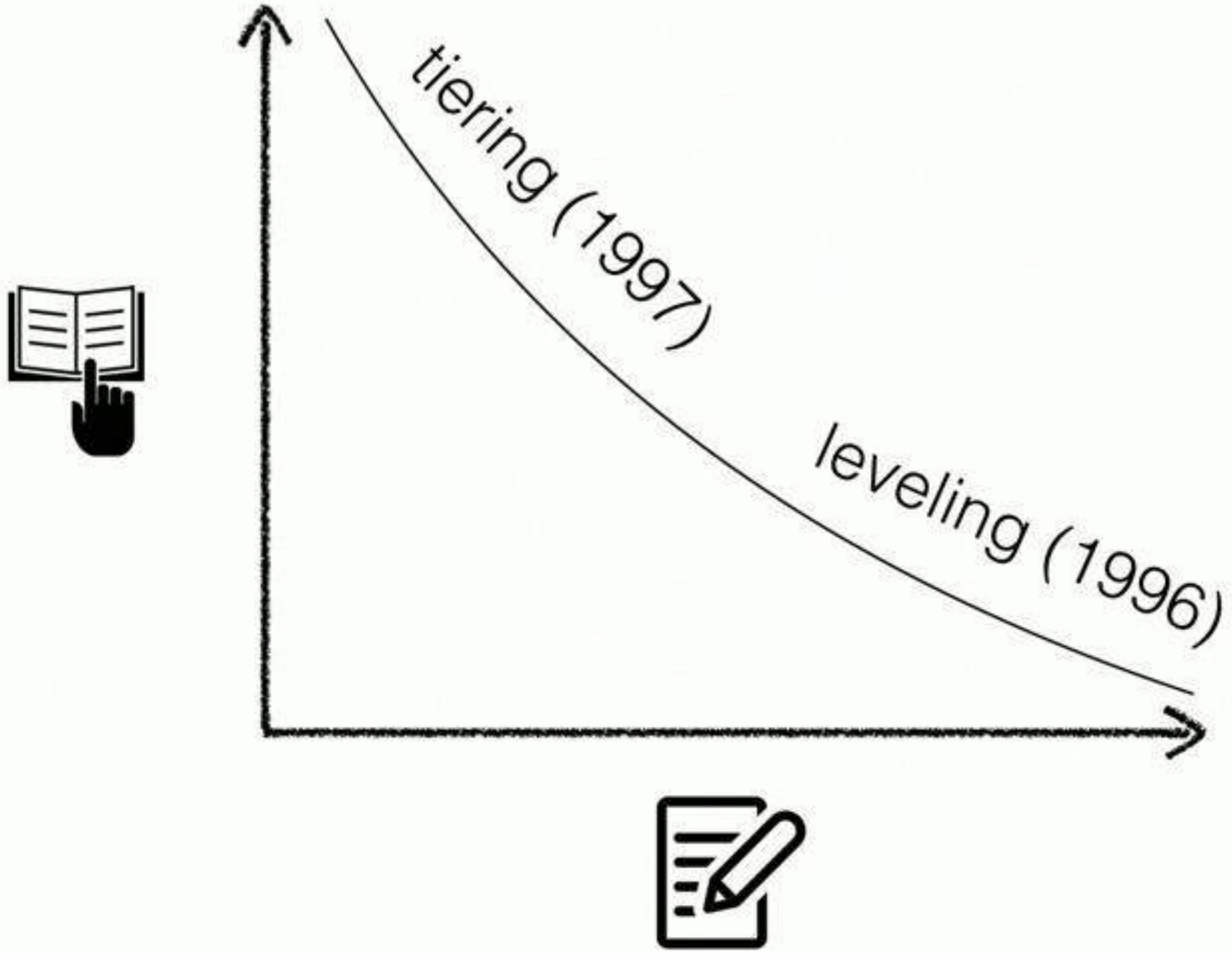


Conclusion



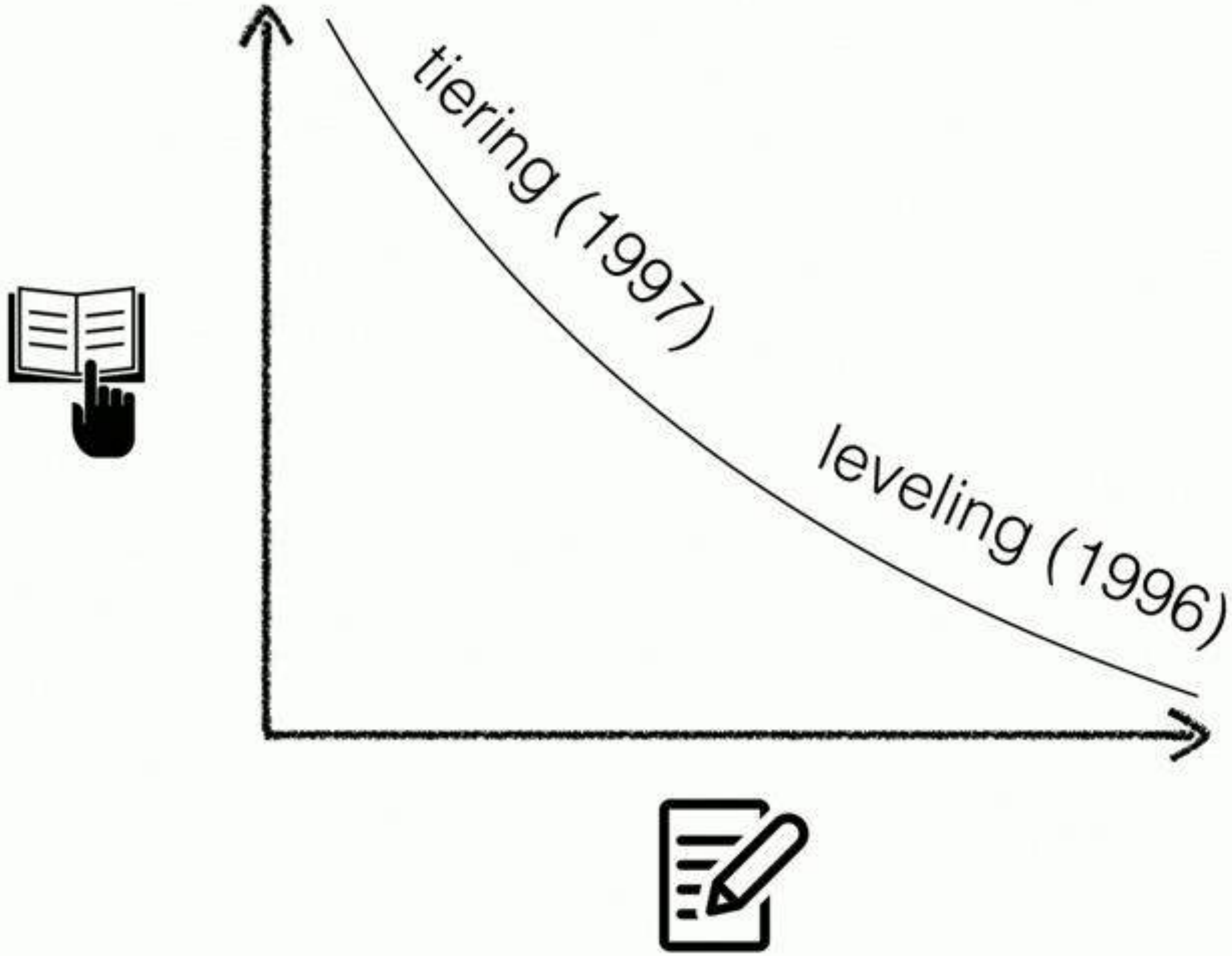
DASlab
@ Harvard SEAS

Conclusion



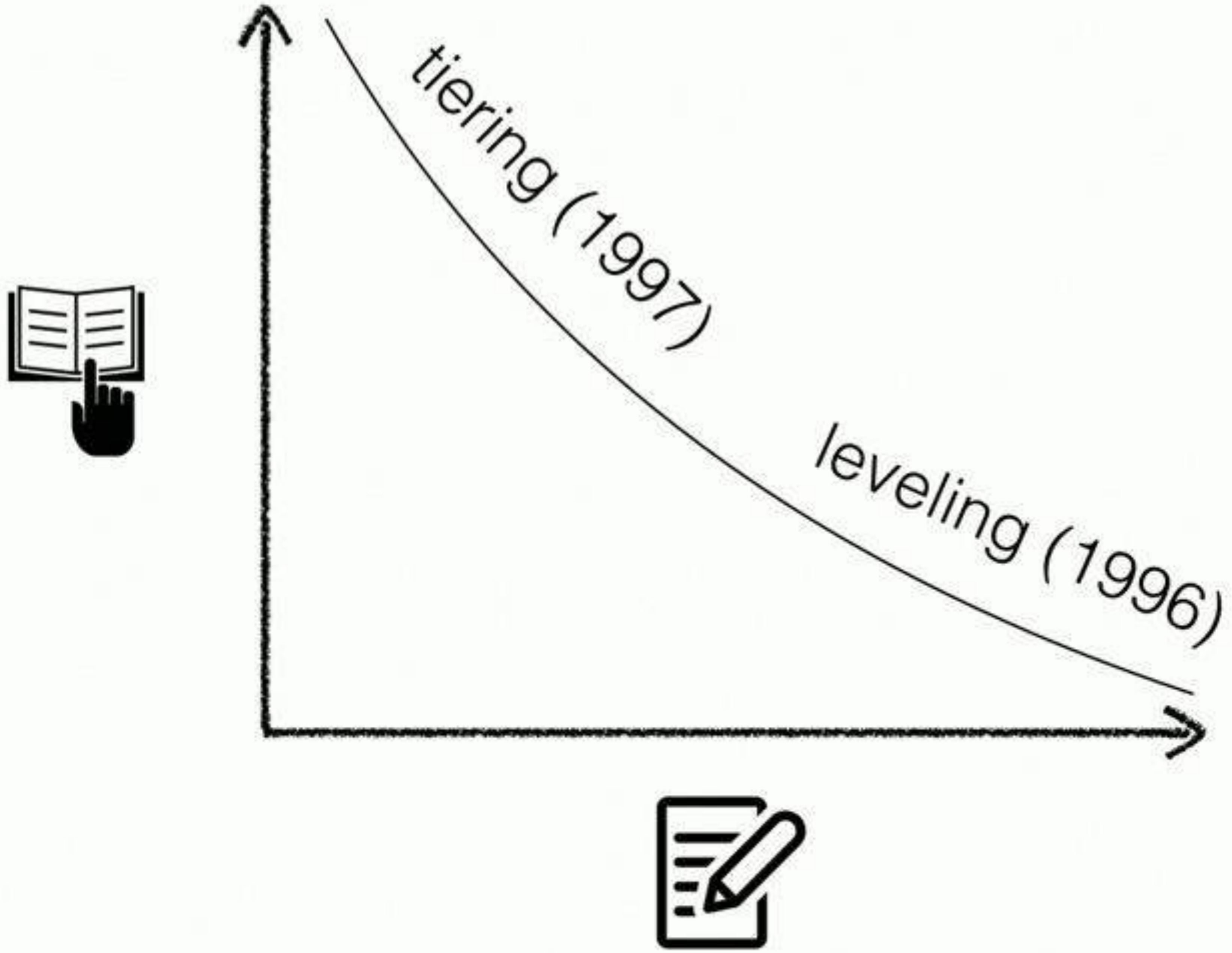
- little memory

Conclusion



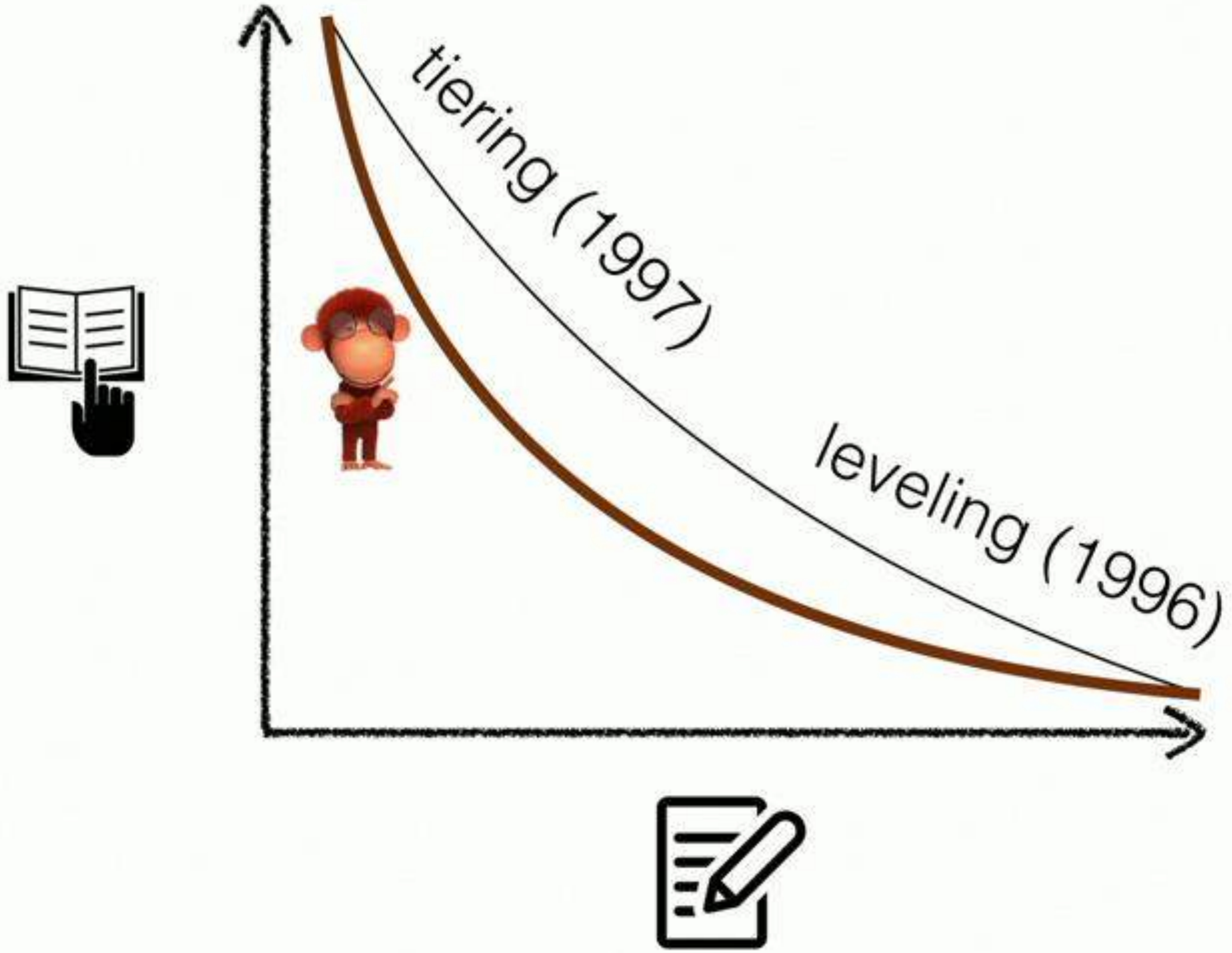
DASlab @ Harvard SEAS

Conclusion



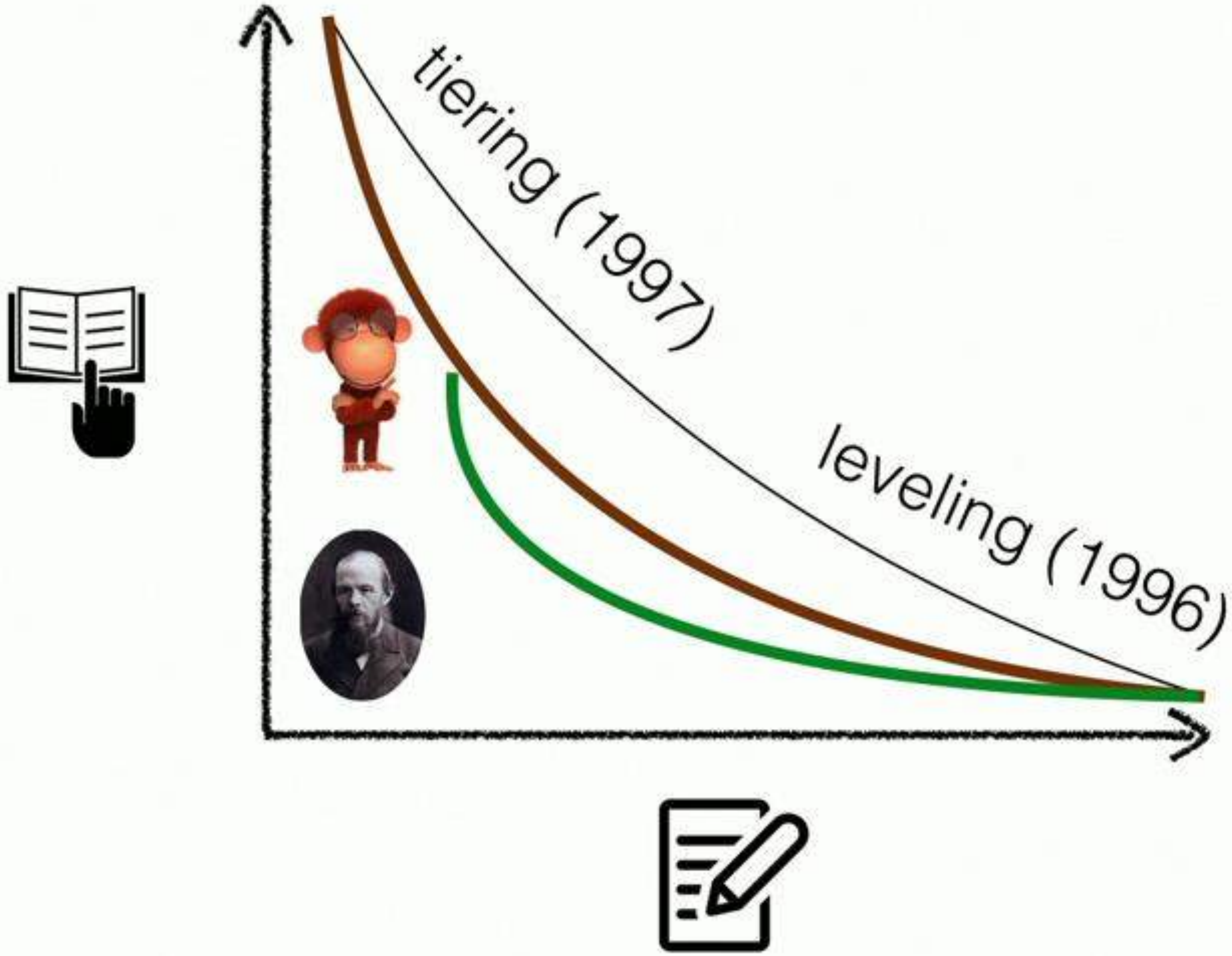
ample memory

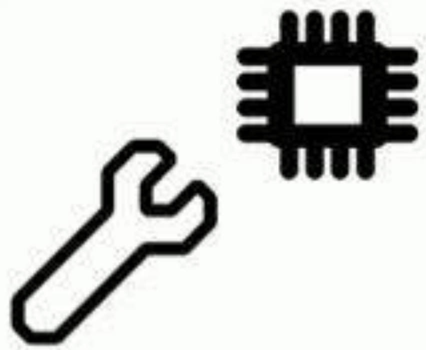
Conclusion



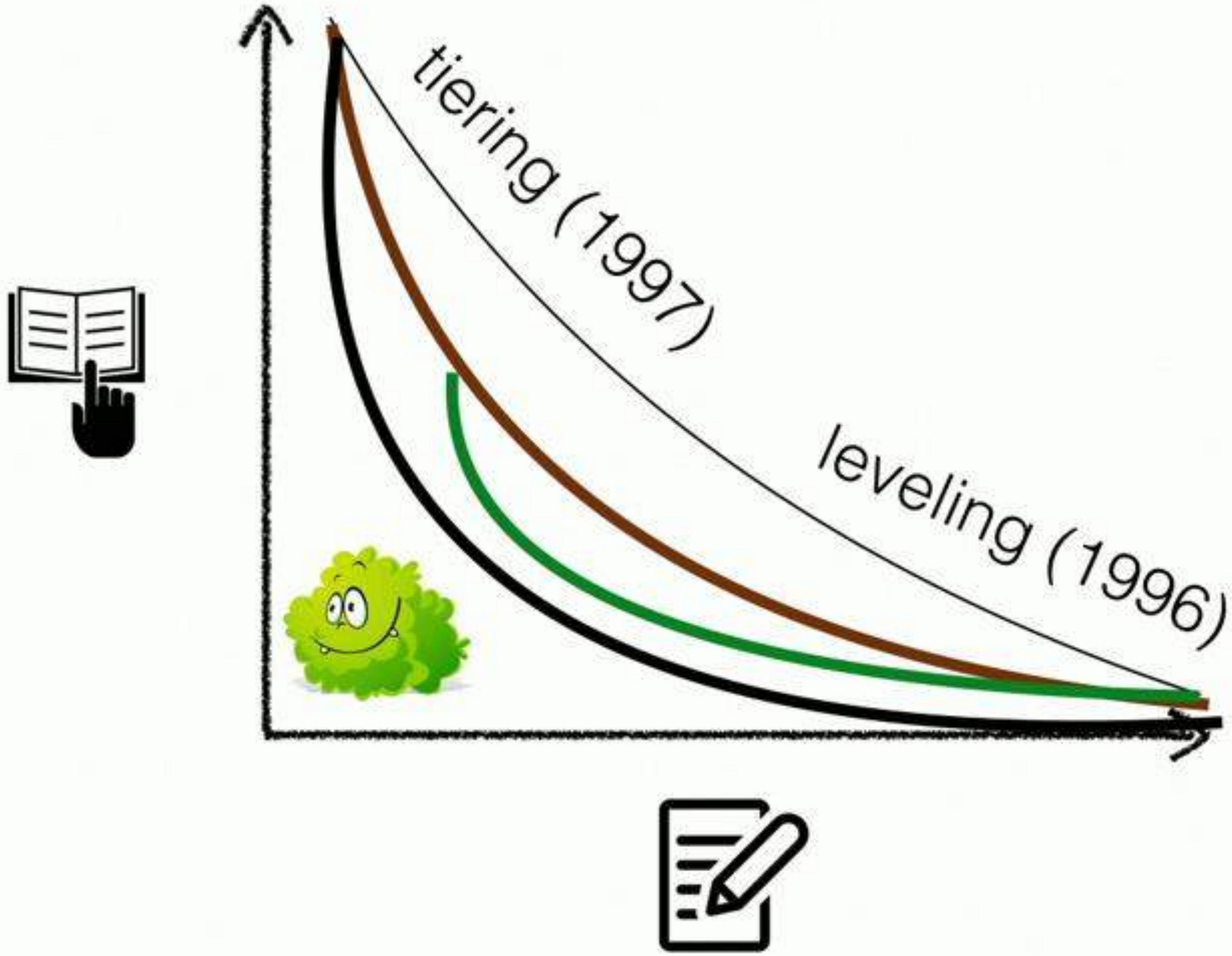
ample memory

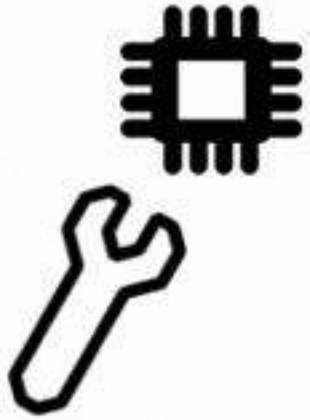
Conclusion





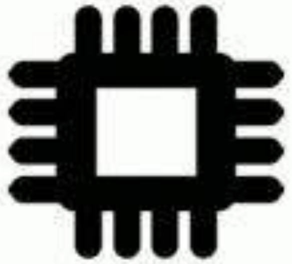
Conclusion



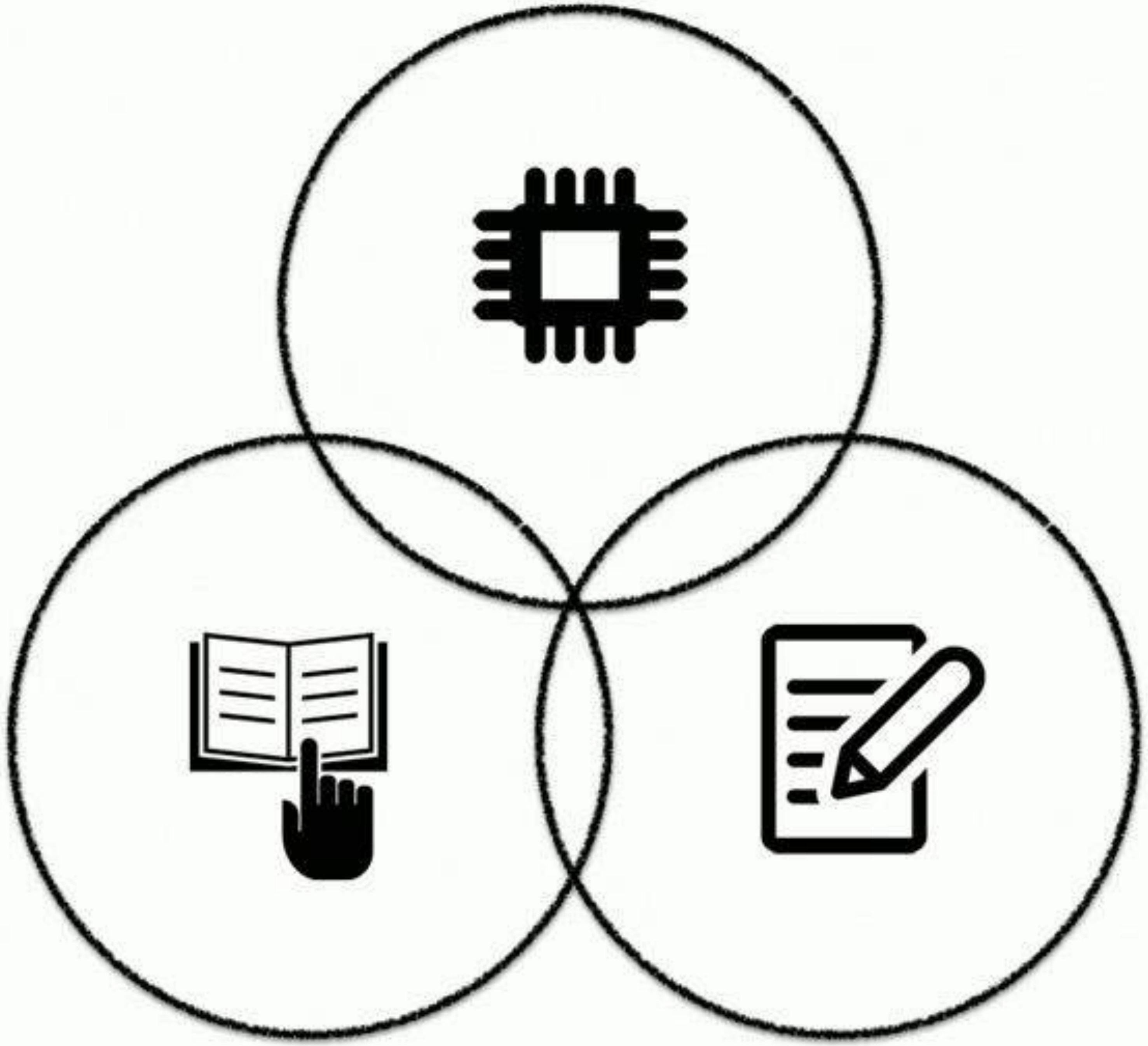


DASlab @ Harvard SEAS

Conclusion

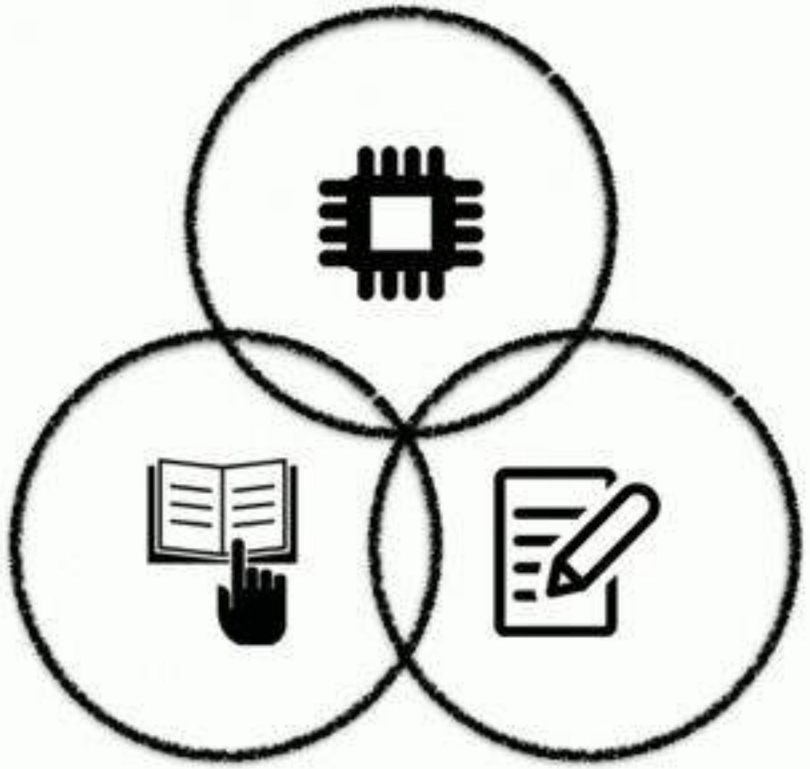


Conclusion

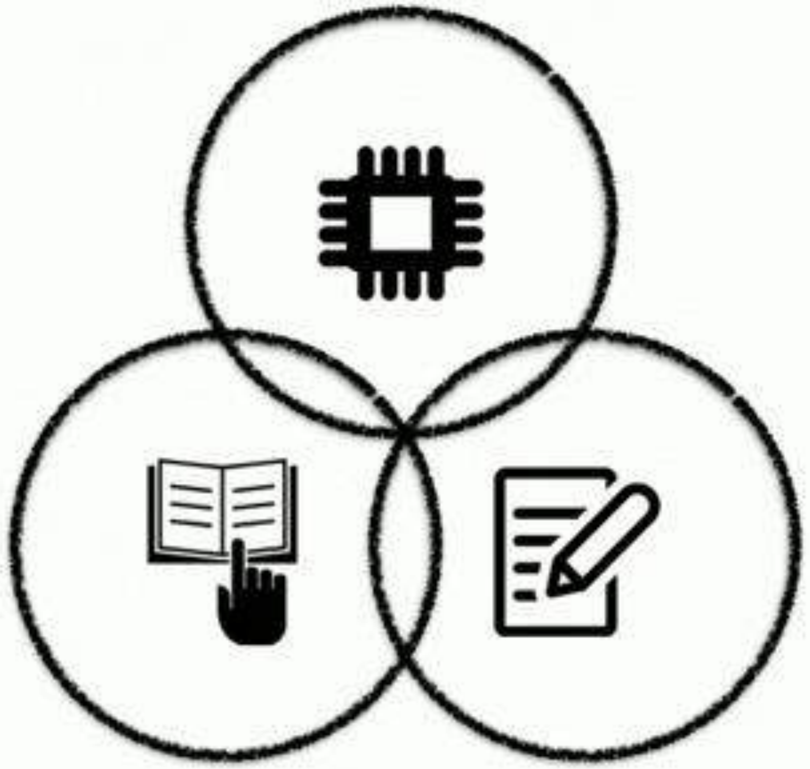


DASlab
@ Harvard SEAS

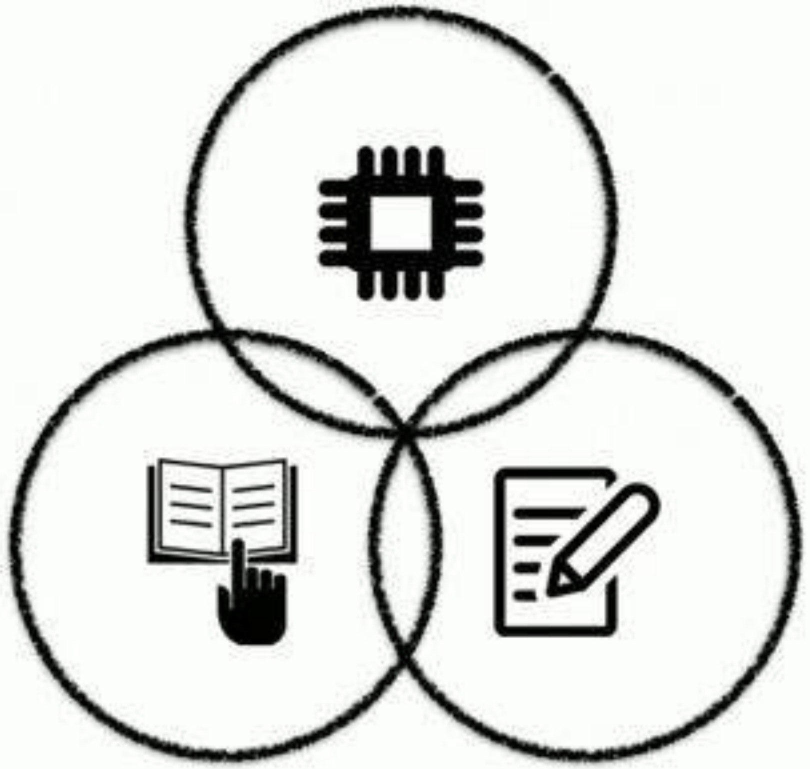
Conclusion



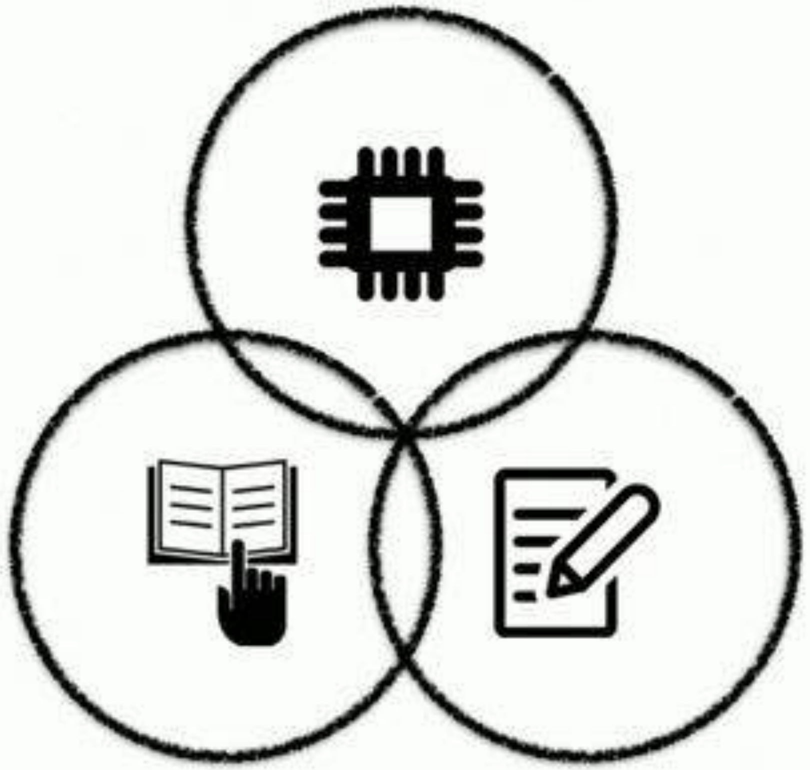
Conclusion



CrimsonDB



CrimsonDB



Thanks!