Toward Privacy-Aware Data Systems

navigating the **privacy-performance tradeoff**

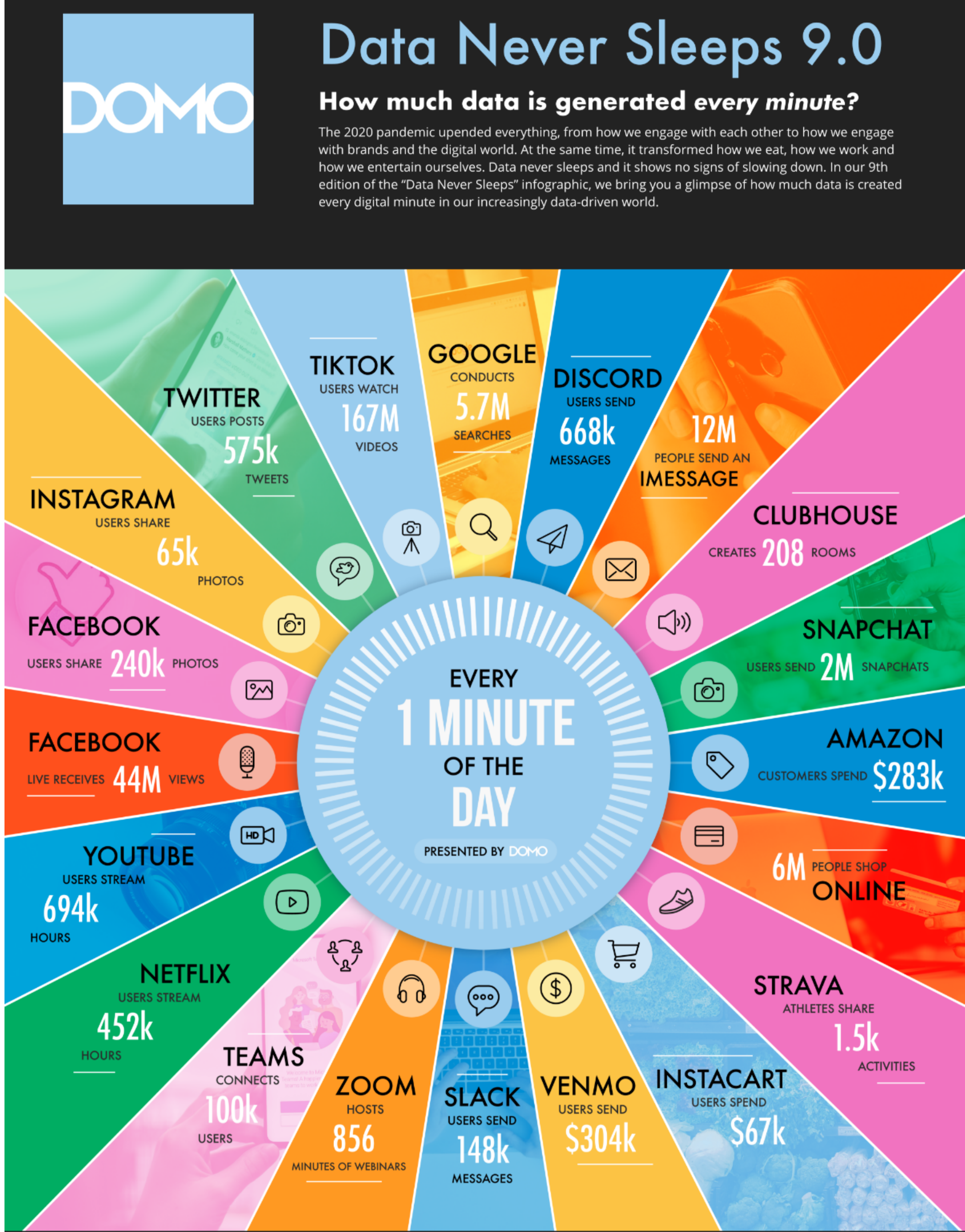
Subhadeep Sarkar

https://subhadeep.net/

Brandeis UNIVERSITY

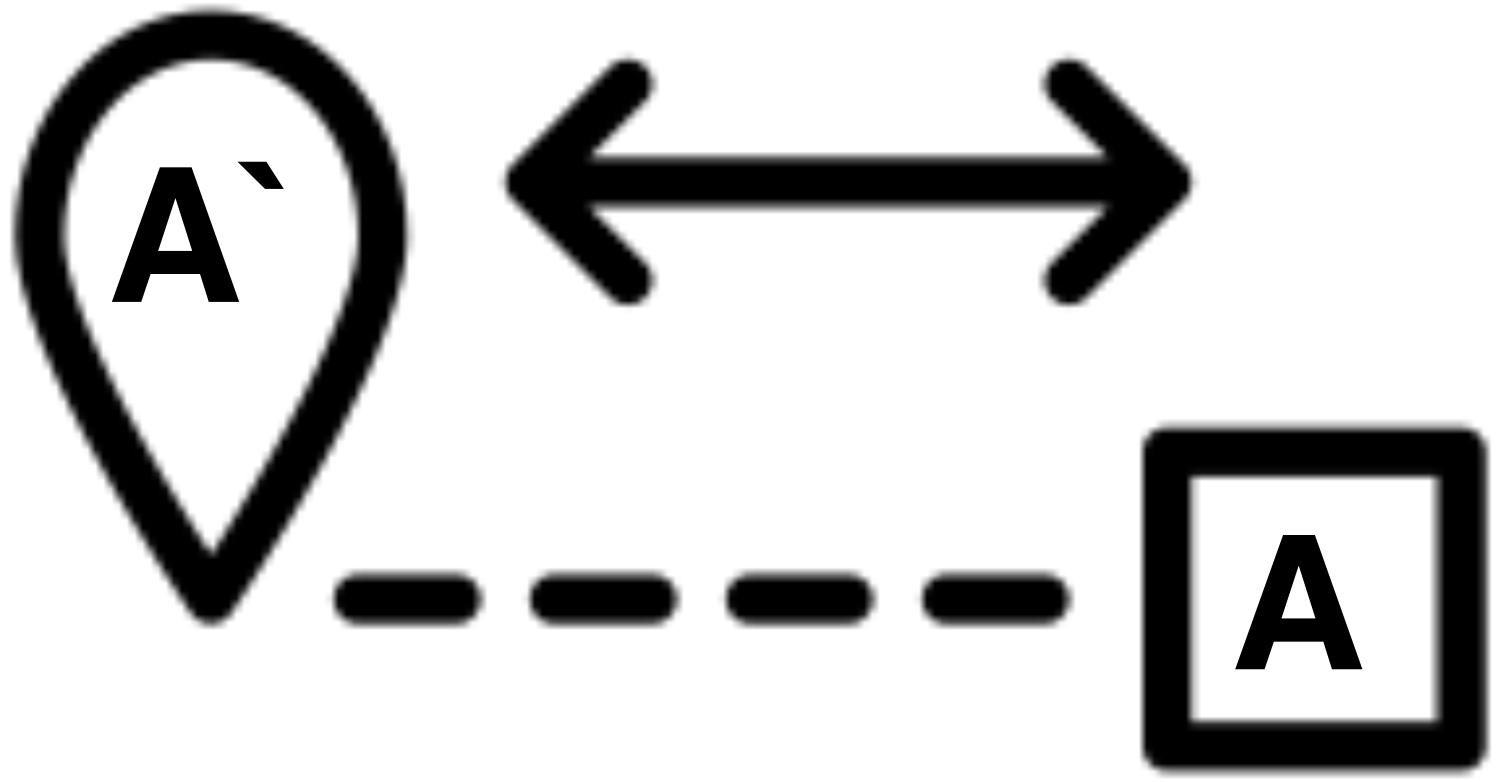
Every two days we generate as much data as we did since the dawn of humanity until 2003.

— **Eric Schmidt** (CEO, Google), 2010



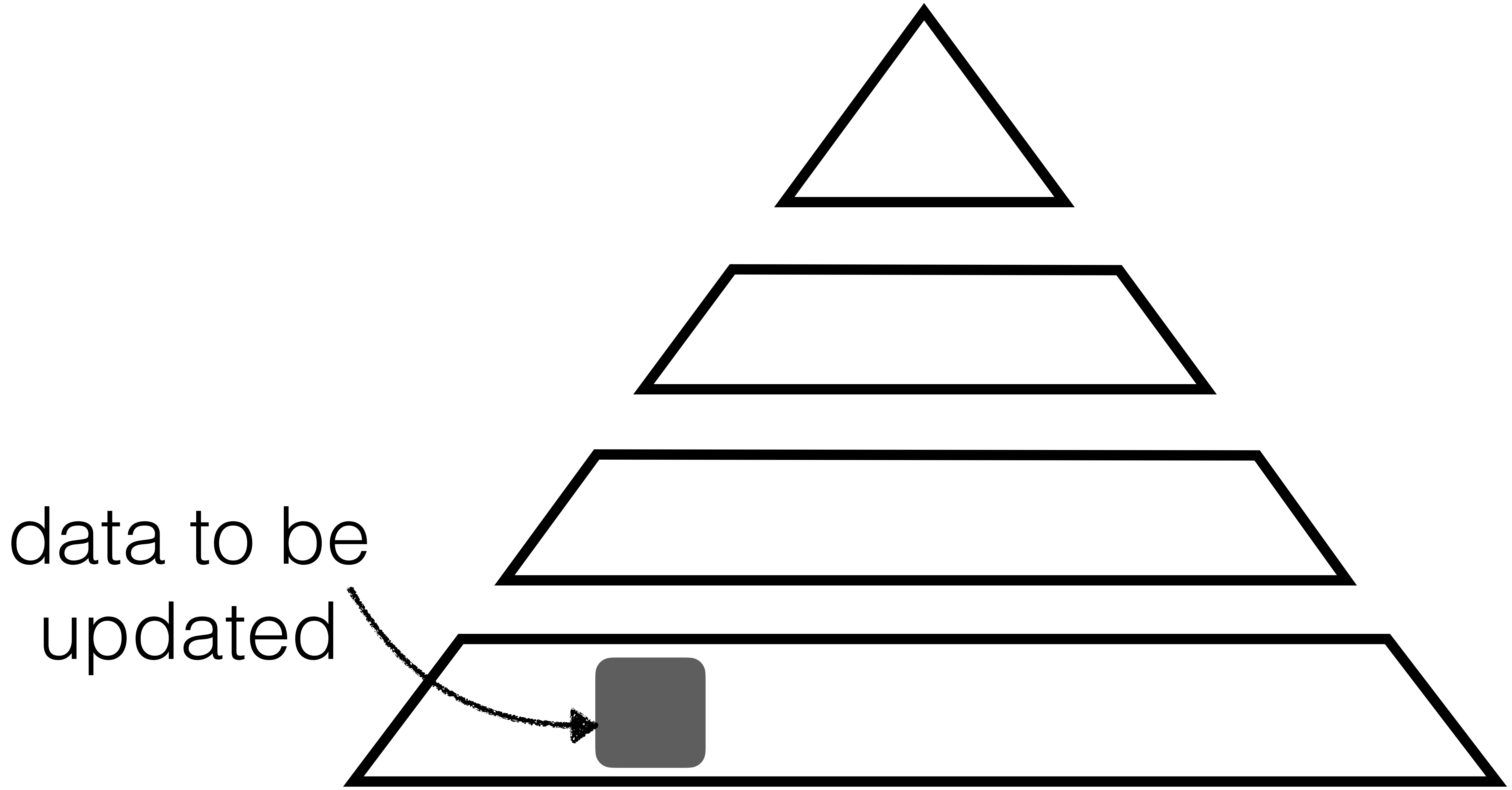
Ingestion-Optimized Systems

batched inserts

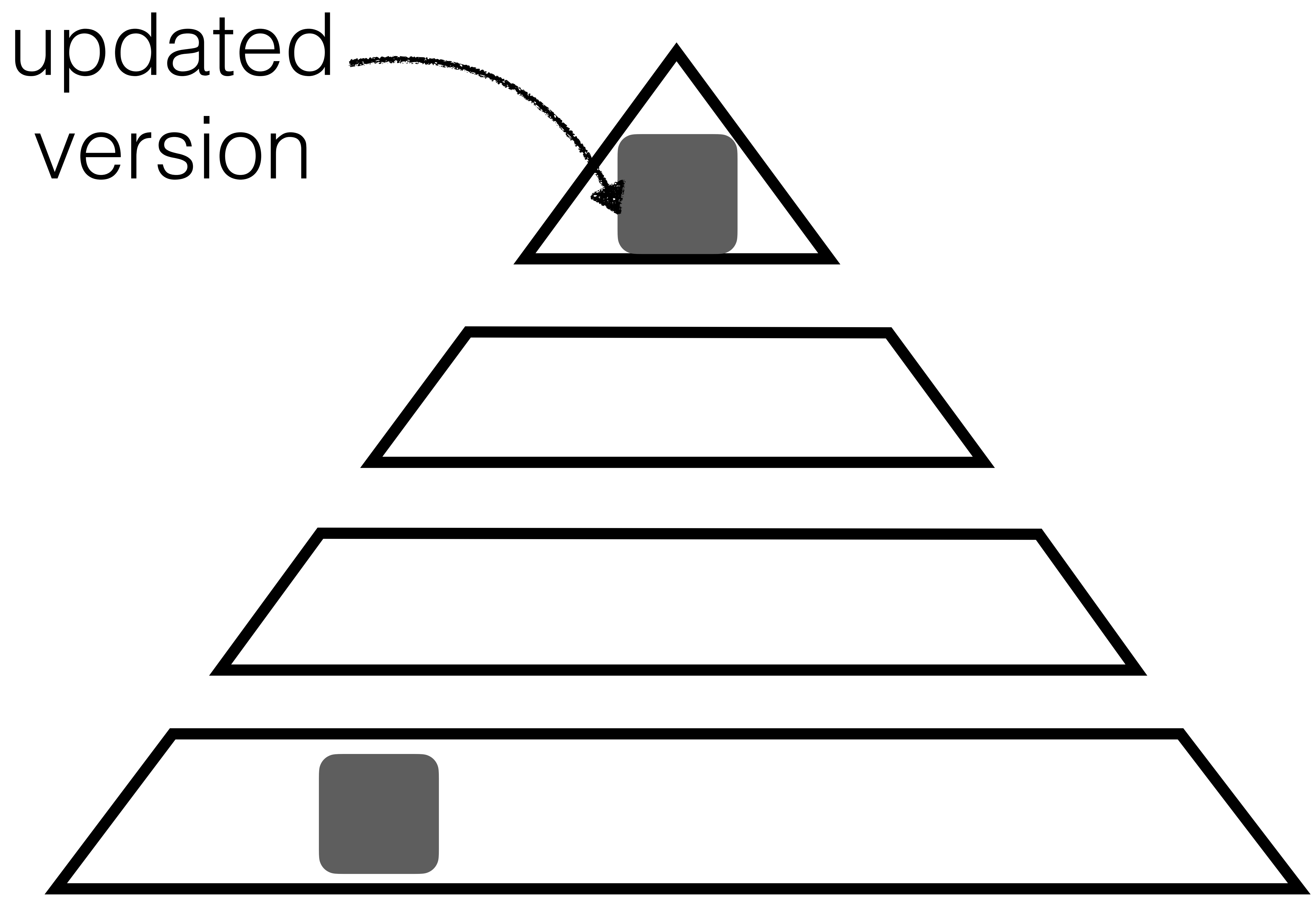


out-of-place
deletes/updates

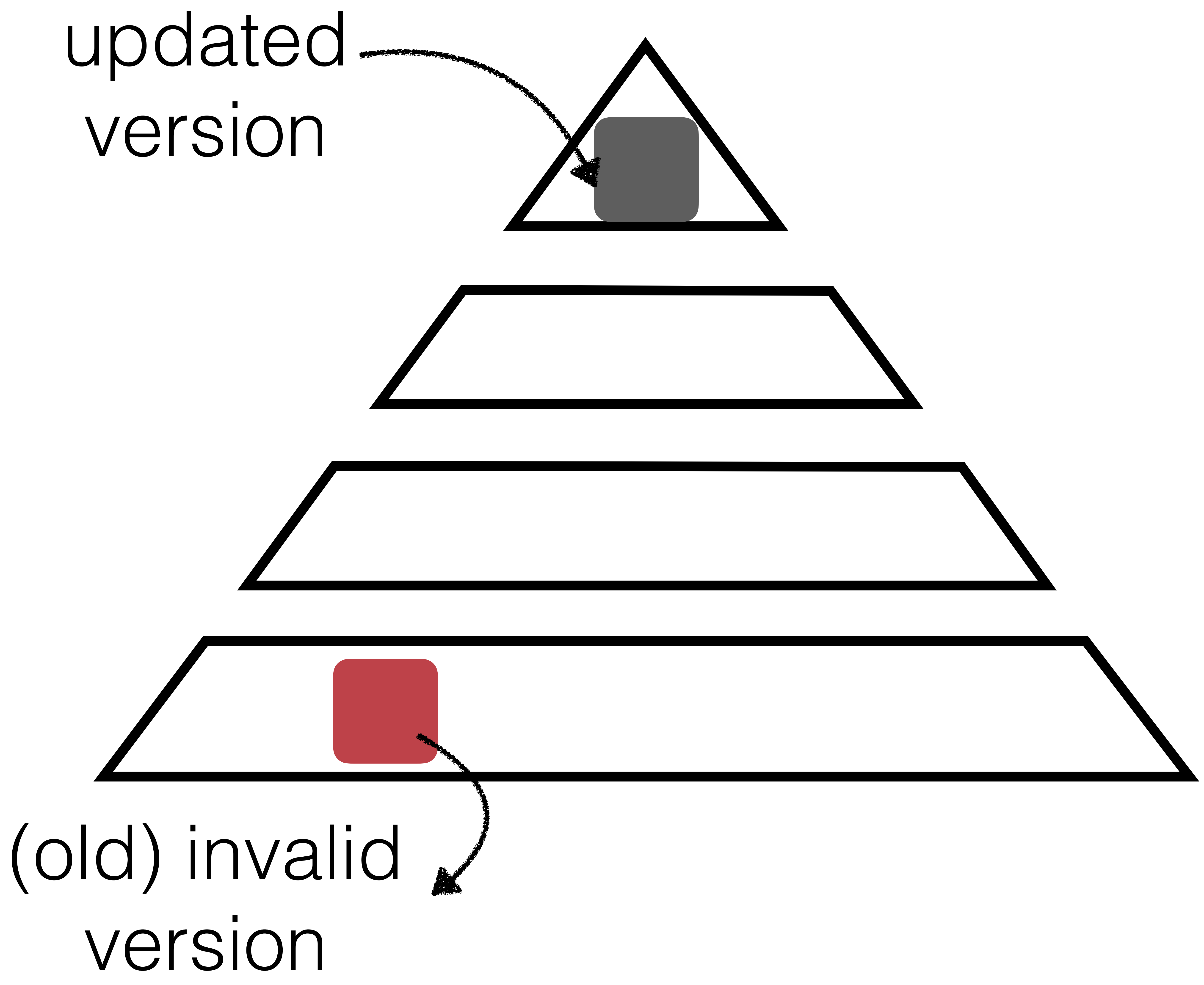
Out-of-place Deletes/Updates



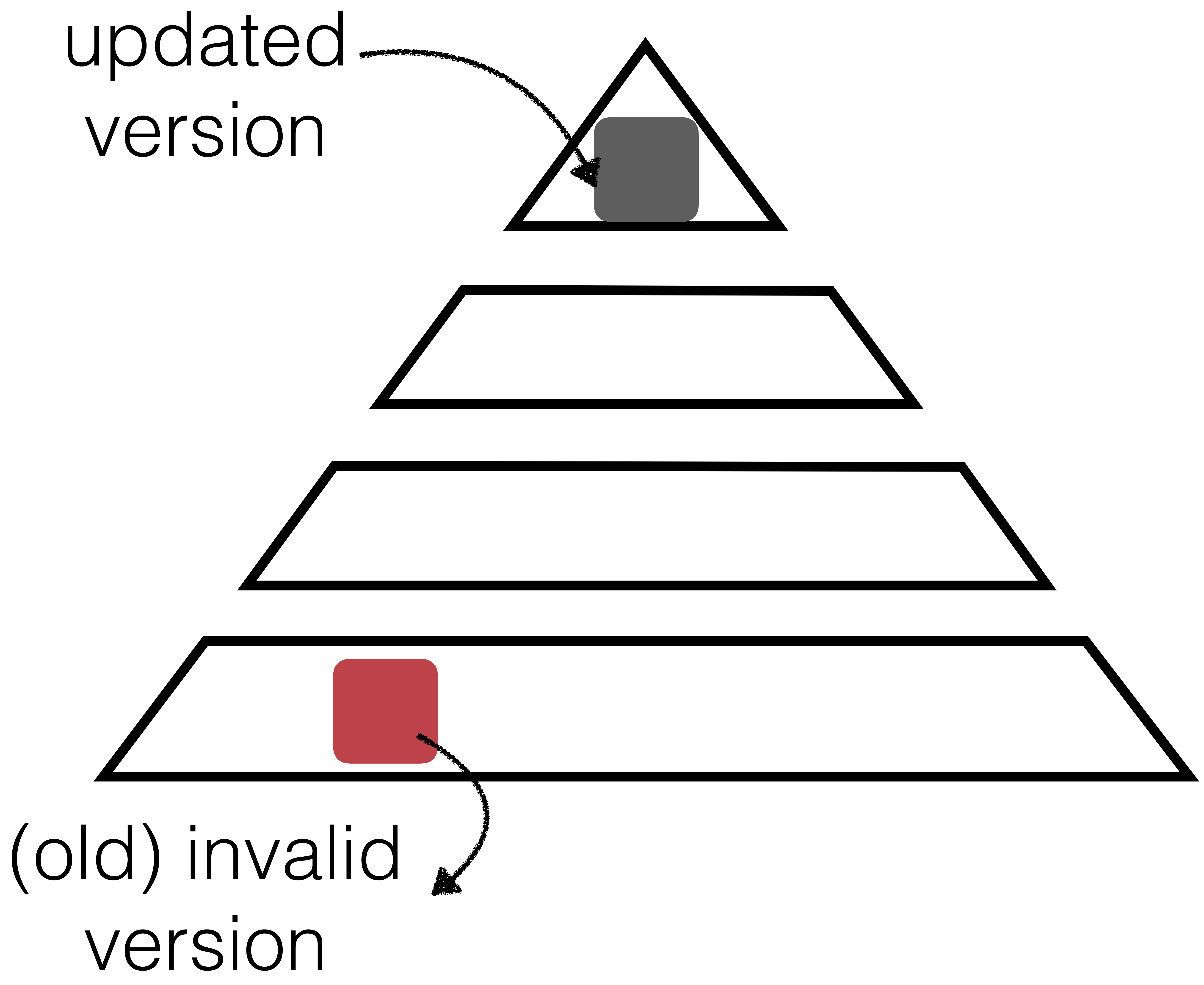
Out-of-place Deletes/Updates



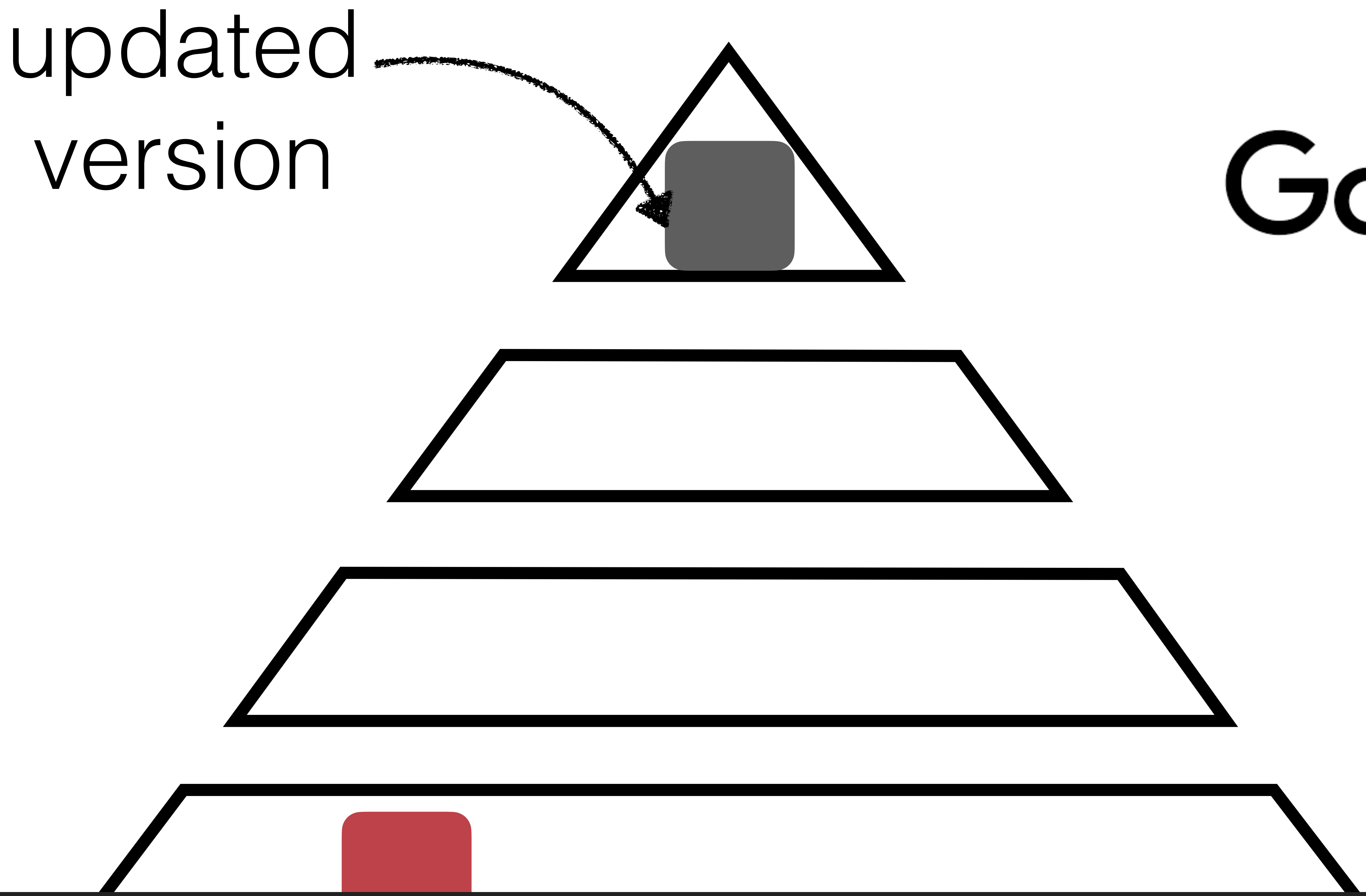
Out-of-place Deletes/Updates



Out-of-place Deletes/Updates



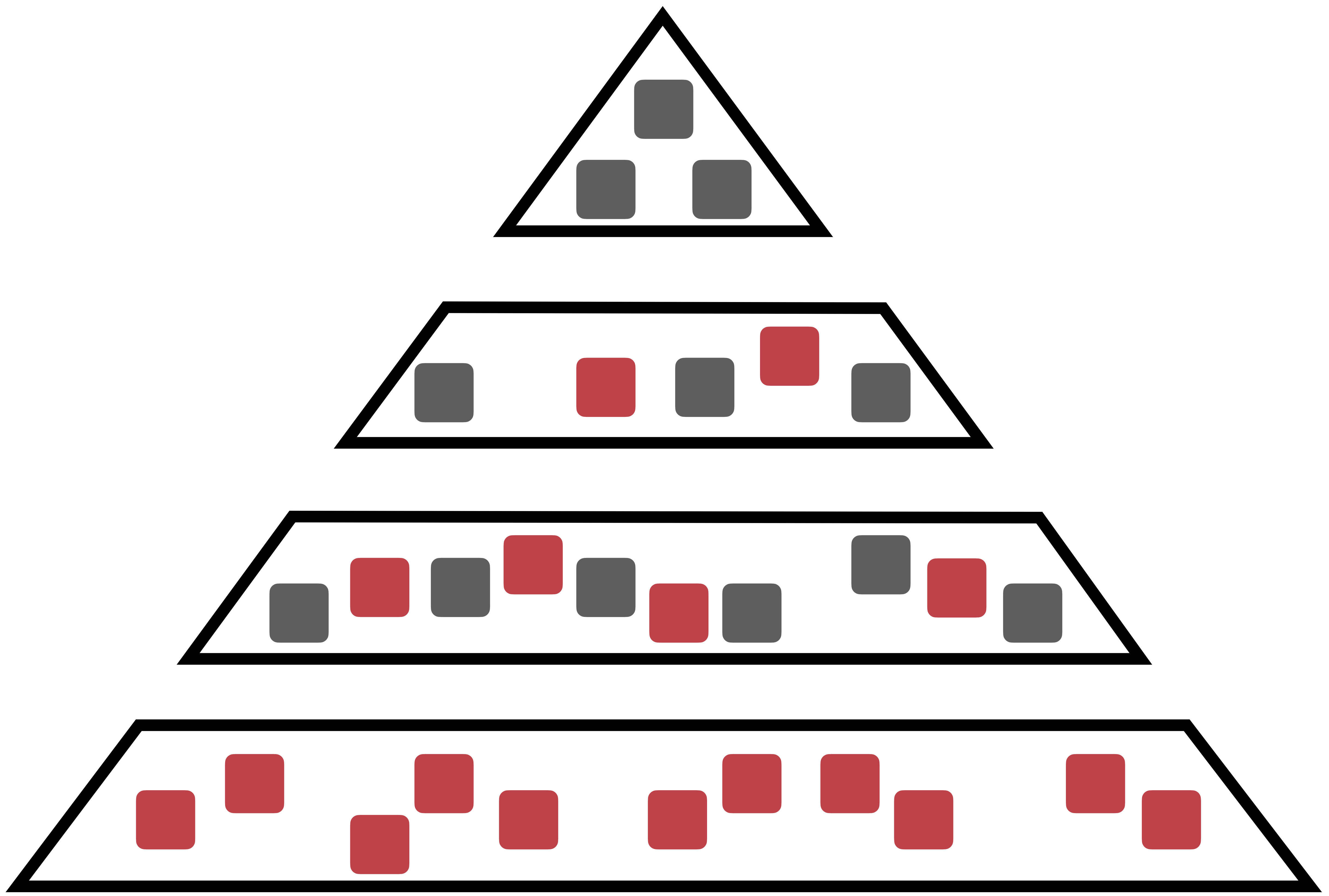
Out-of-place Data Systems



Google Meta amazon THE APACHE SOFTWARE FOUNDATION SAP mongoDB

Hidden Cost: does not scale with deletes!

Hidden Cost of Logical Deletes



⬛ delete-markers / tombstones

🟥 logically invalidated data

up to 2x

high space amplification

1.5x - 5x

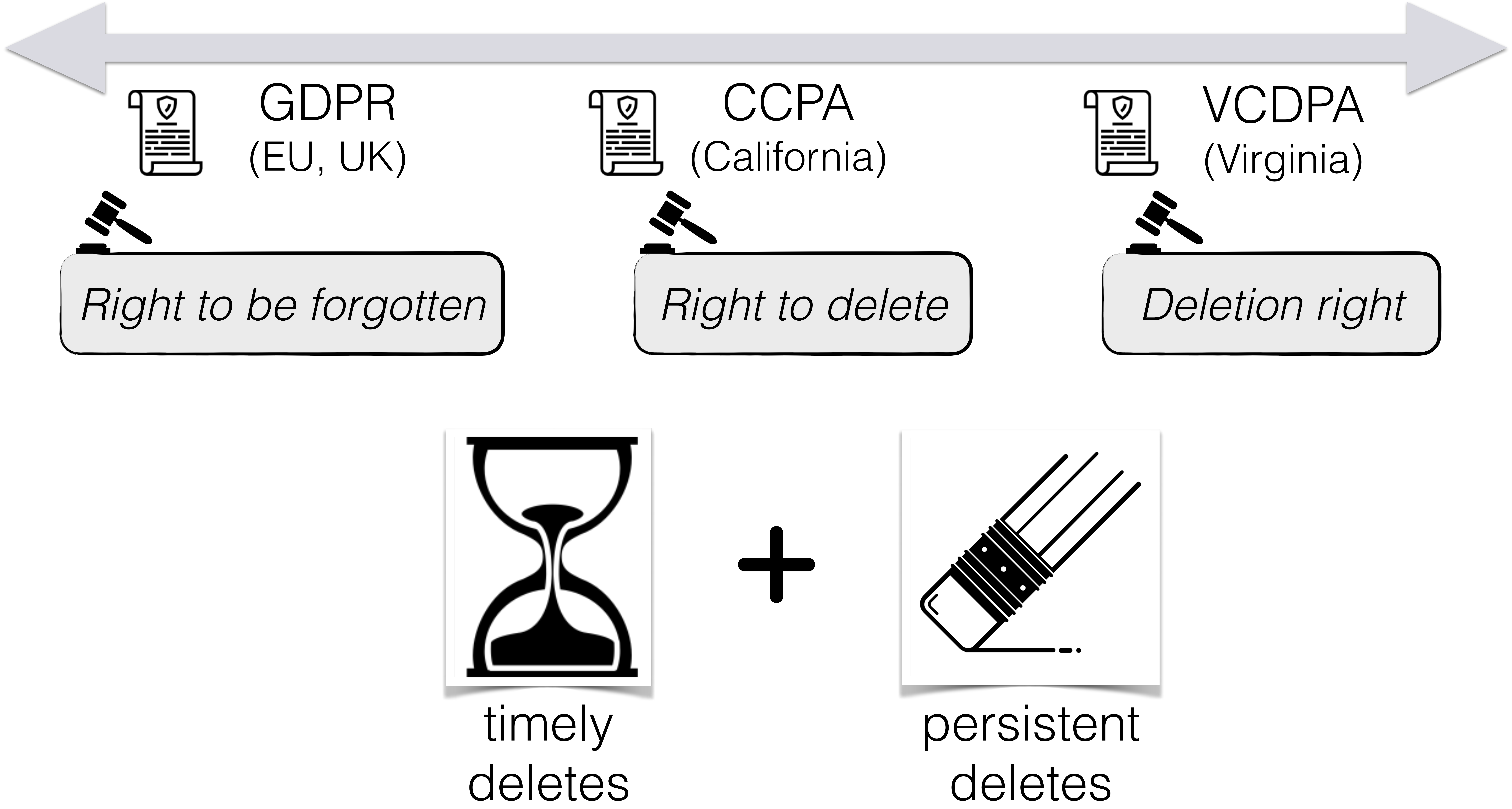
poor read performance

increased operational cost

Logical Deletes & **Data Privacy**

CCPA
VCDPA
GDPR
HEAVY
ROBUST
MODERATE
LIMITED
REGULATION & ENFORCEMENT

[DLA Piper's Data Protection Laws of the World Handbook, 2022]


GDPR
(EU, UK)
Right to be forgotten
CCPA
(California)
Right to delete
VCDPA
(Virginia)
Deletion right
timely deletes
+
persistent deletes

Right to be forgotten

Right to delete

Deletion right

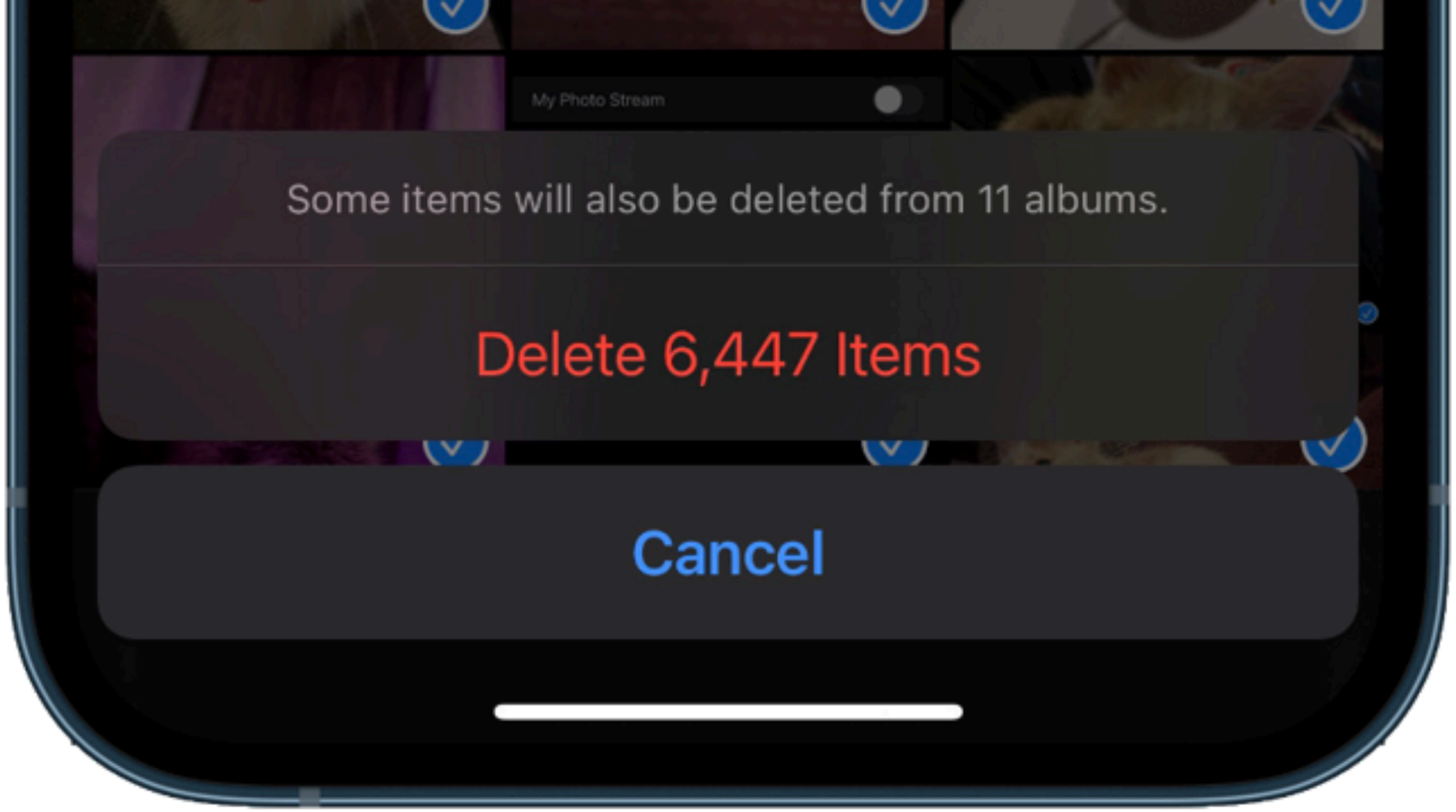
Even years later, Twitter doesn't delete your direct messages

Zack Whittaker, Natasha Lomas / 1:57 PM EST • February 15, 2019

timely deletes

+

persistent deletes

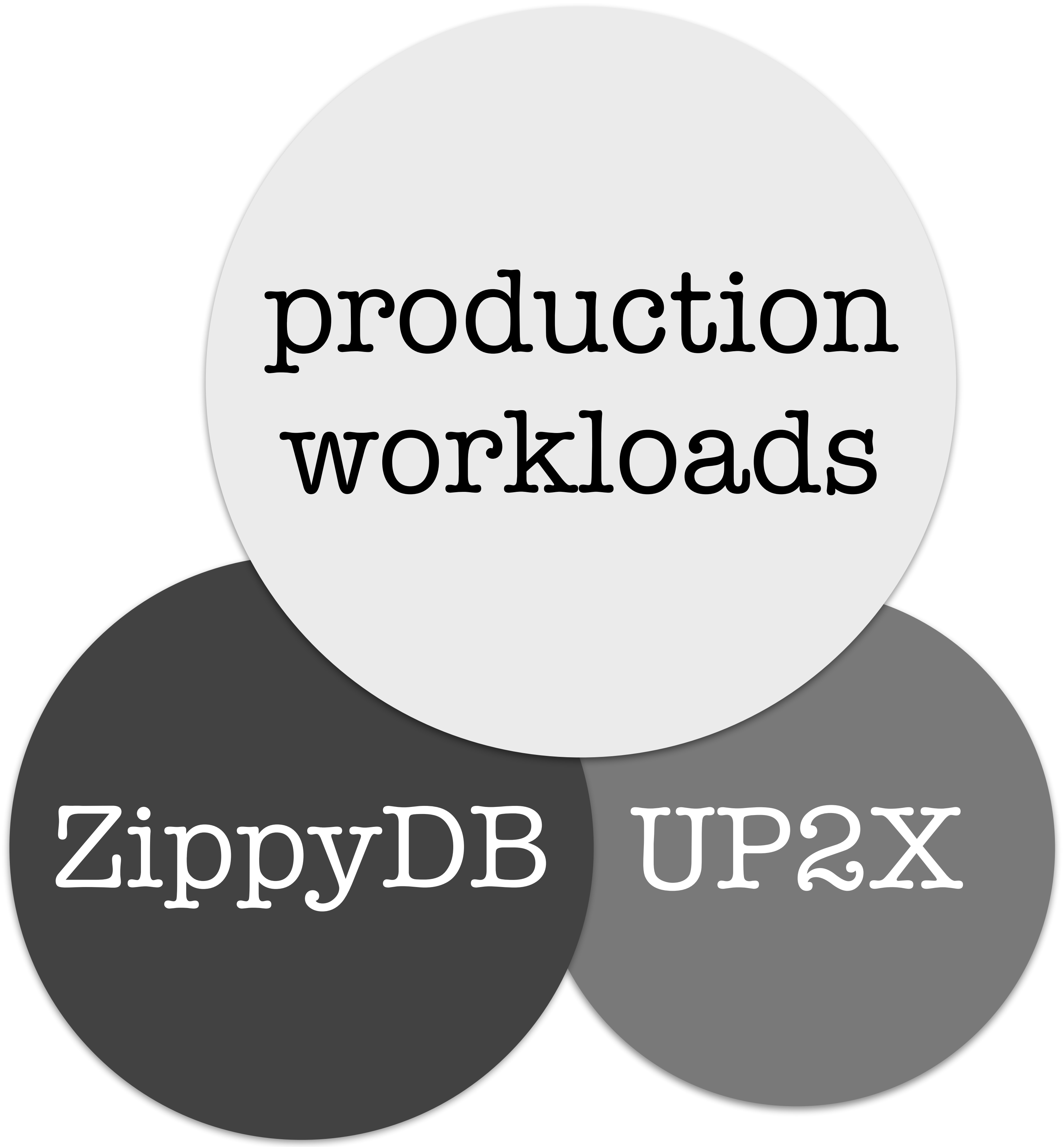


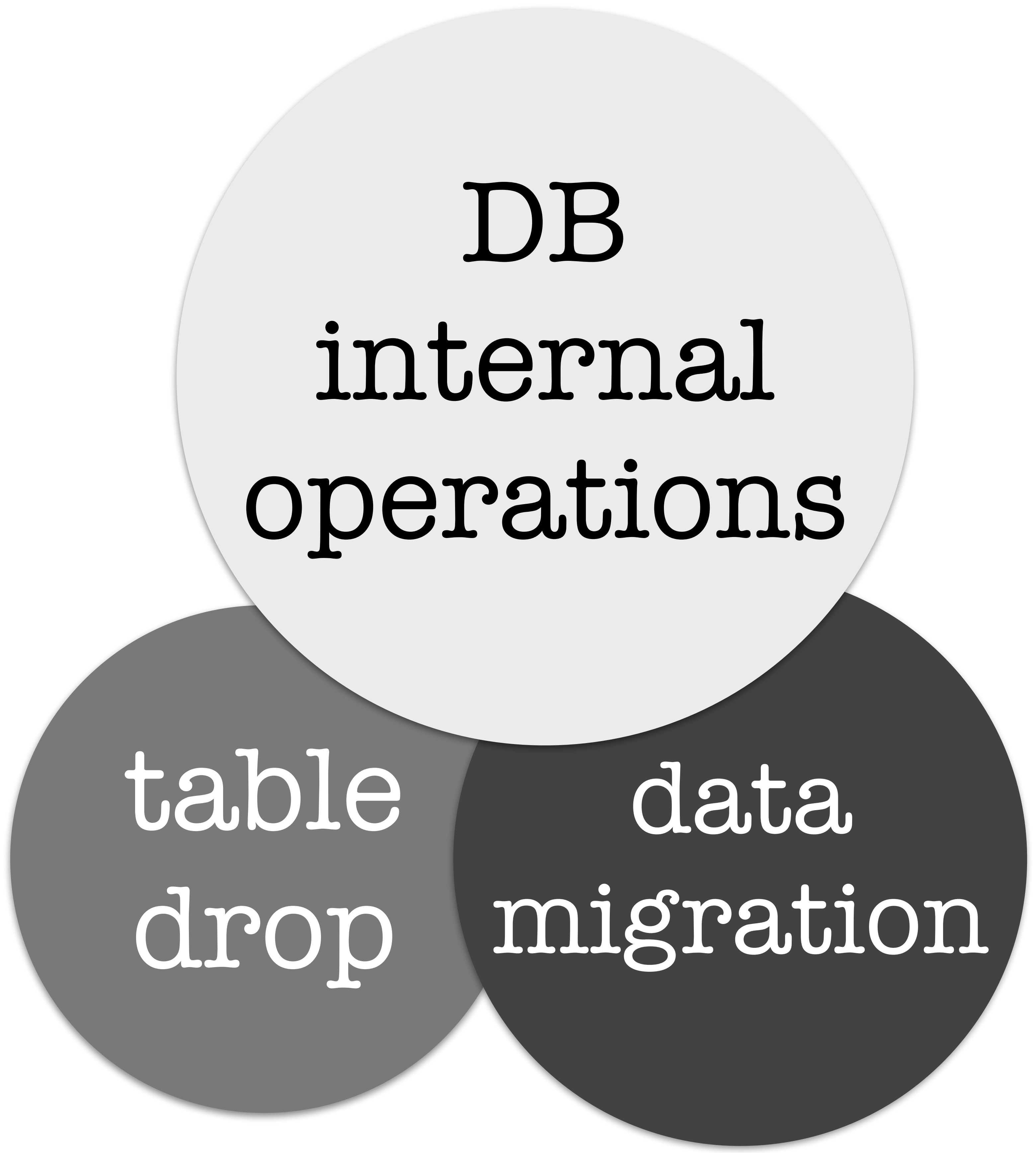
Some items will also be deleted from 11 albums.

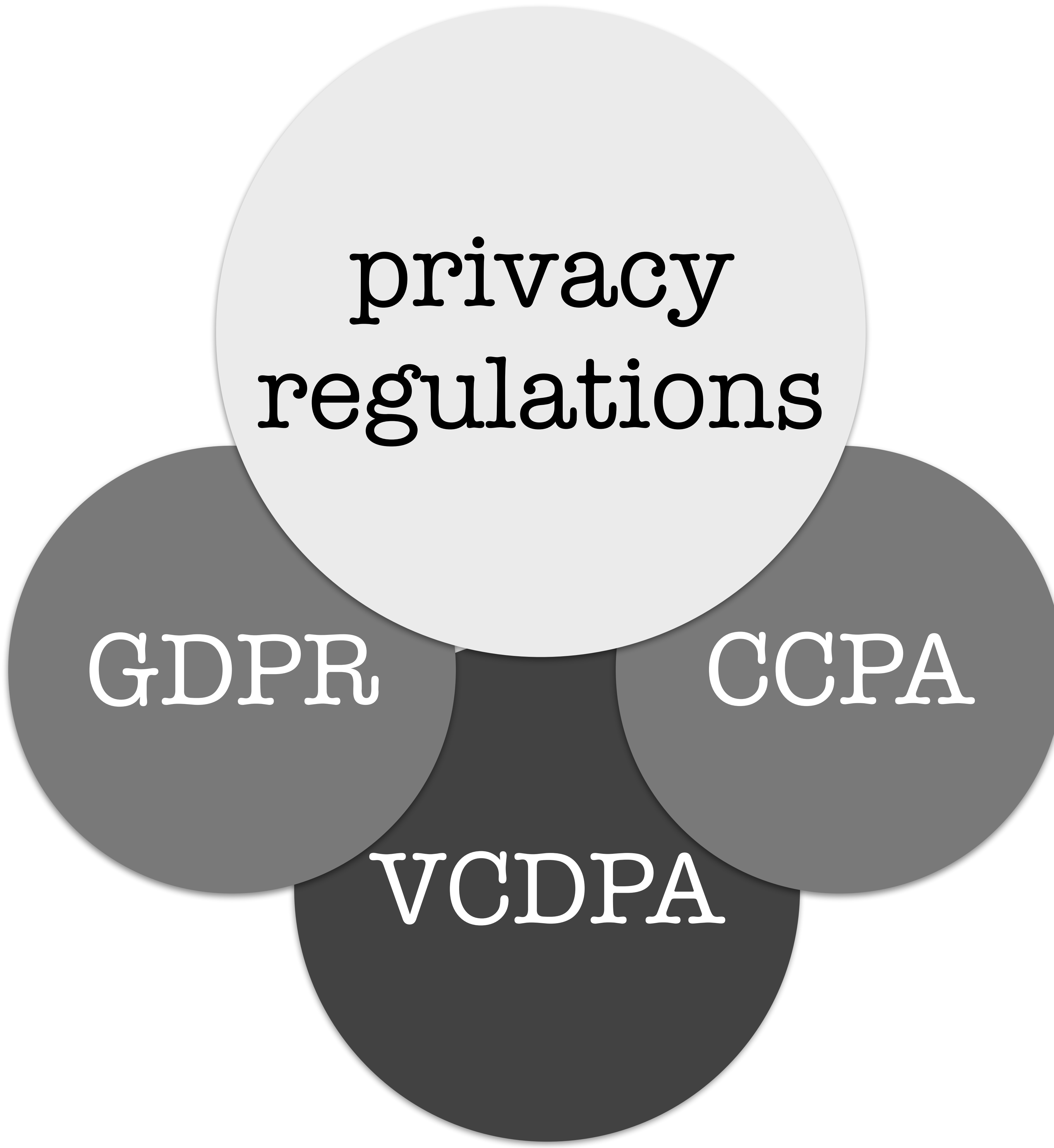
100M+ deletes/day

deletes in **batches**

user-generated deletes







Goal: Enabling **Privacy through Deletion**

dissecting privacy policies and regulations

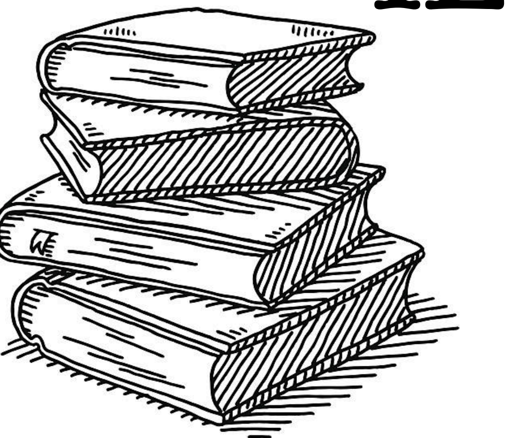
- IEEE DEBull 2022
- EDBT 2022
- IEEE iThings 2018

fundamental design changes in data layouts and access methods

- ICDE 2023-a
- TPCTC 2022
- SIGMOD 2022-b
- VLDB 2021

navigating the privacy-performance tradeoff

- ICDE 2023-c
- ACM TODS 2023-b
- SIGMOD 2022-a
- SIGMOD 2020



Goal: Enabling **Privacy through Deletion**

dissecting privacy policies and regulations

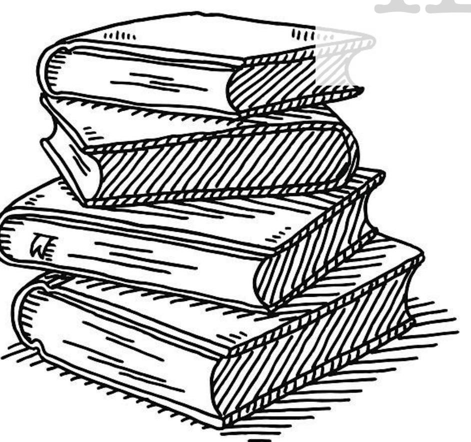
- IEEE DEBull 2022
- EDBT 2022
- IEEE iThings 2018

fundamental design changes in data layouts and access methods

- ICDE 2023-a
- TPCTC 2022
- SIGMOD 2022-b
- VLDB 2021

navigating the privacy-performance tradeoff

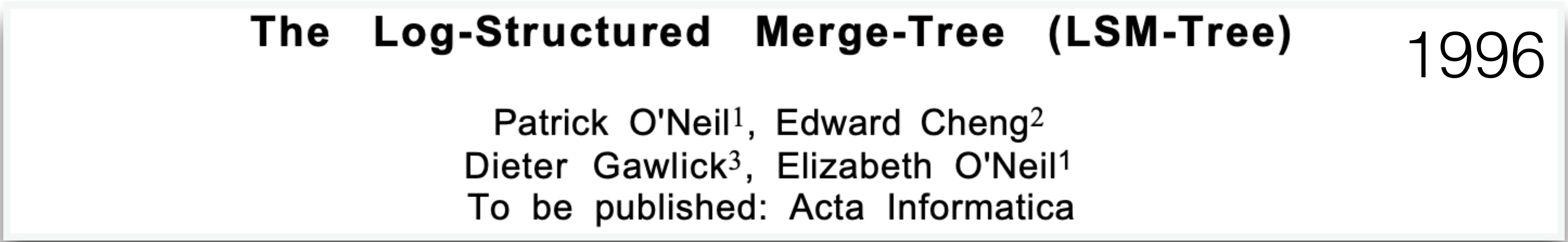
- ICDE 2023-b
- SIGMOD 2022-a
- SIGMOD 2020
- ACM TODS (under review)



Log-**S**tructured **M**erge-tree

LSM-tree

LSM-tree



The Log-Structured Merge-Tree (LSM-Tree)

Patrick O'Neil[1], Edward Cheng[2]
Dieter Gawlick[3], Elizabeth O'Neil[1]
To be published: Acta Informatica

1996

LSM-tree

NoSQL

RocksDB
WT
levelDB
SCYLLA
DynamoDB
cassandra
tarantool
Bigtable
APACHE HBASE
riak
accumulo™

SQLite

relational

influxdb
QuasarDB

time-series

2023

LSM-tree

NoSQL

RocksDB

WT

levelDB

SCYLLA

DynamoDB

cassandra

tarantool

Bigtable

APACHE HBASE

riak

accumulo™

SQLite

influxdb

QuasarDB

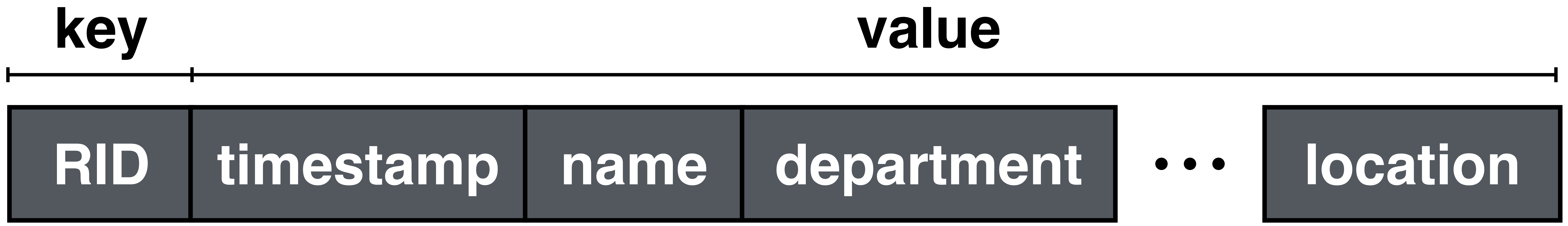
relational

time-series

2023

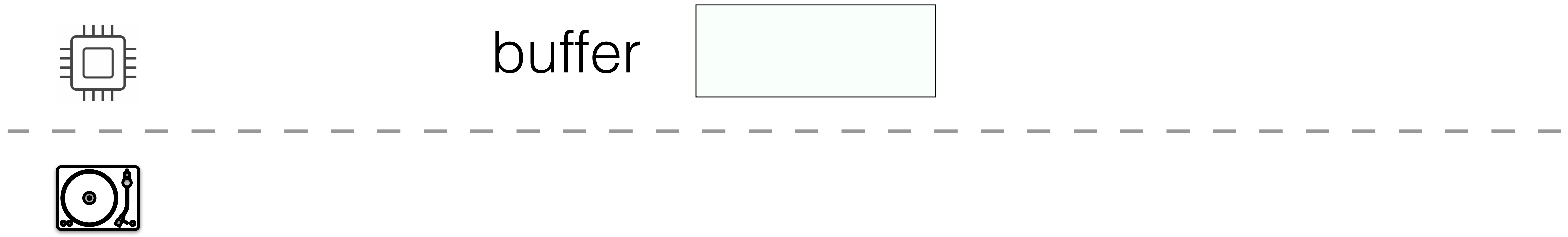
LSM **Basics**

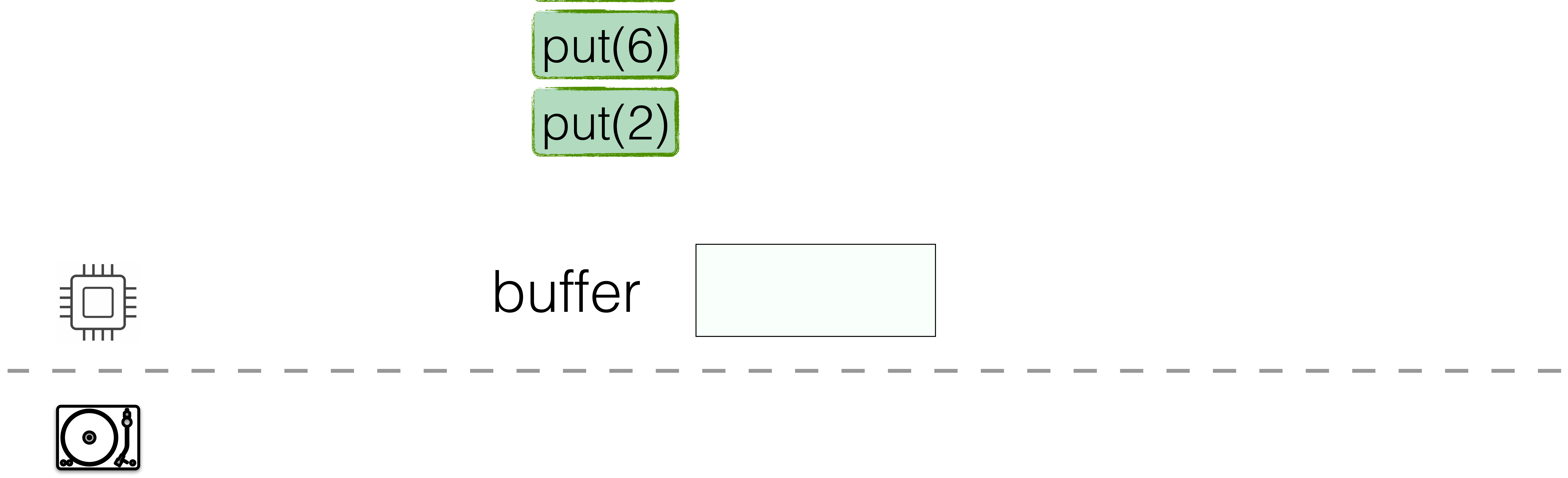
key-value pairs

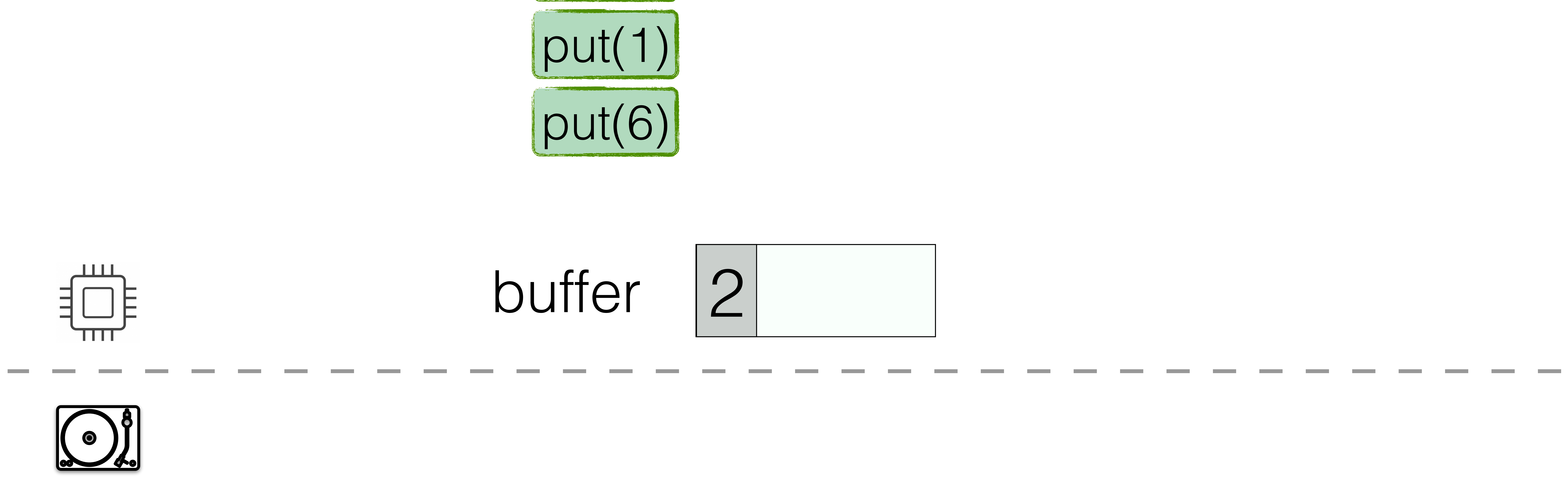


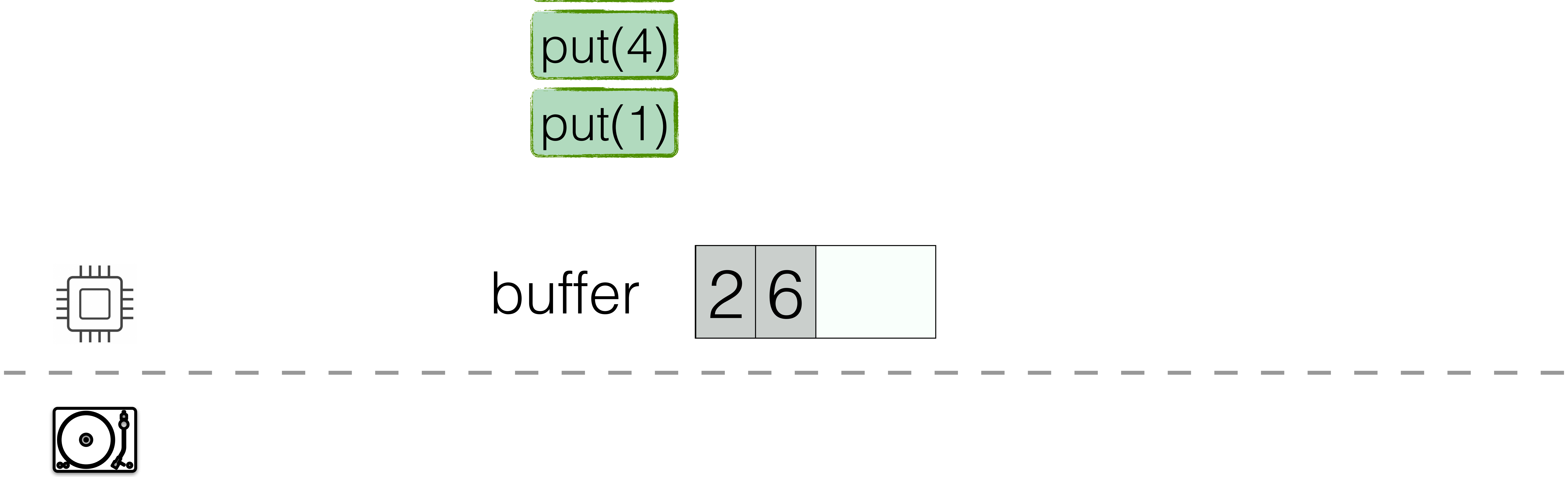
LSM **Basics**

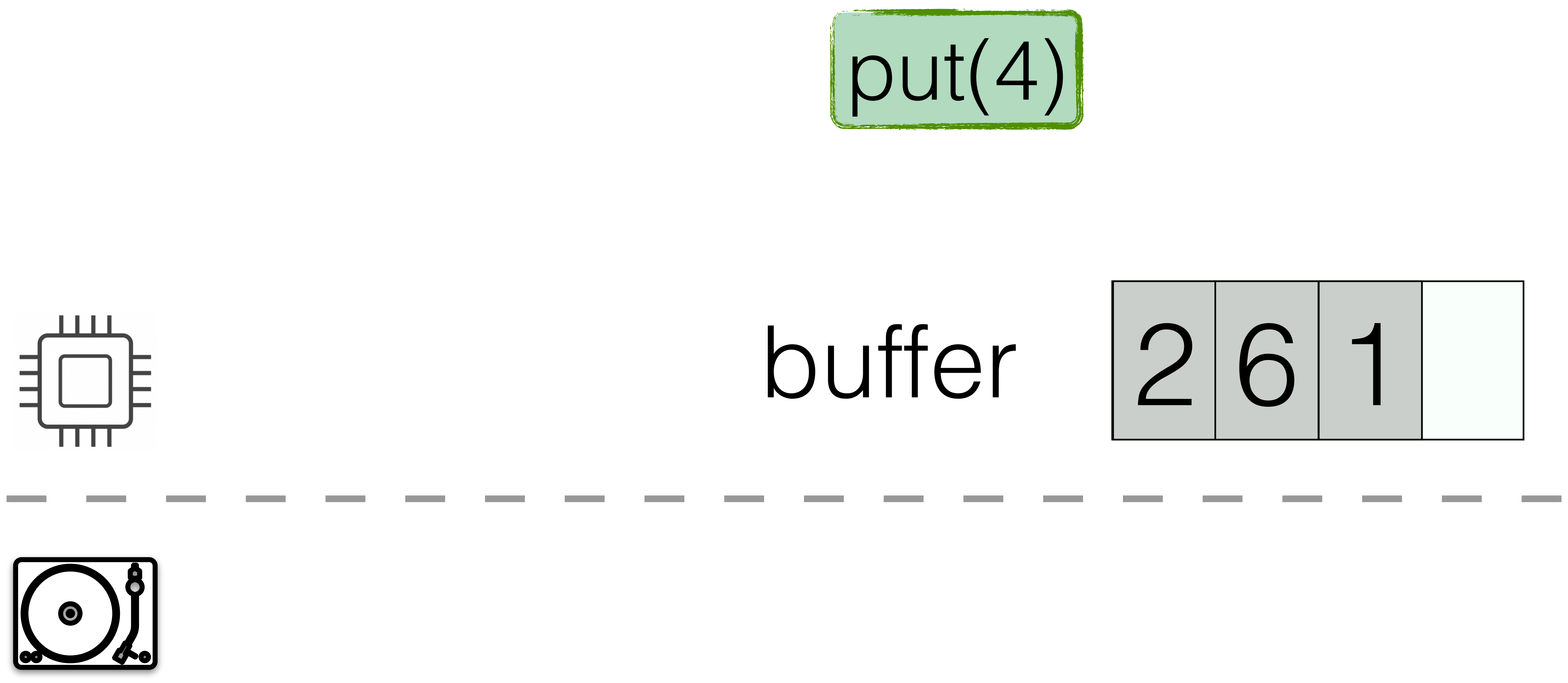
key-value pairs

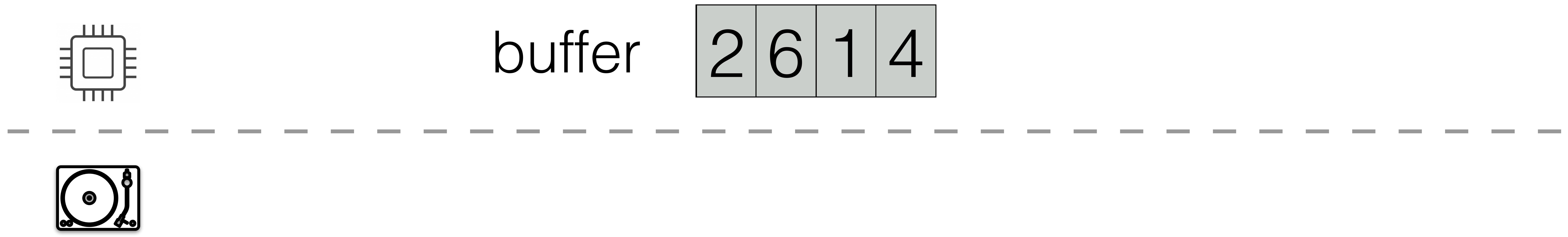

buffer

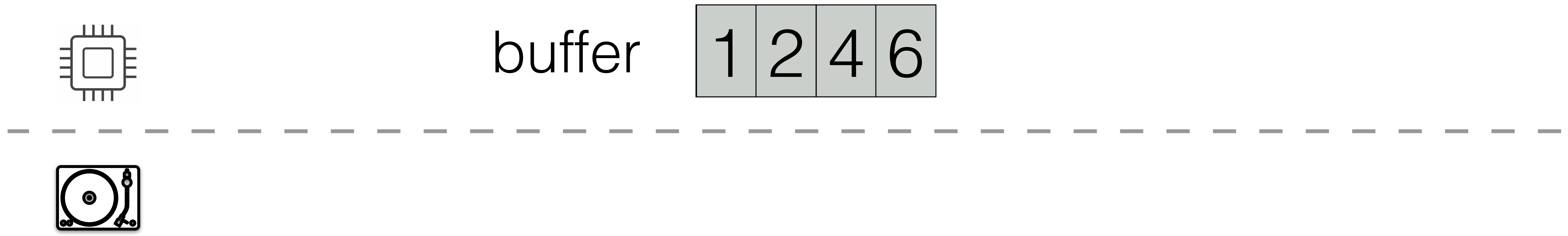

put(6)
put(2)
buffer

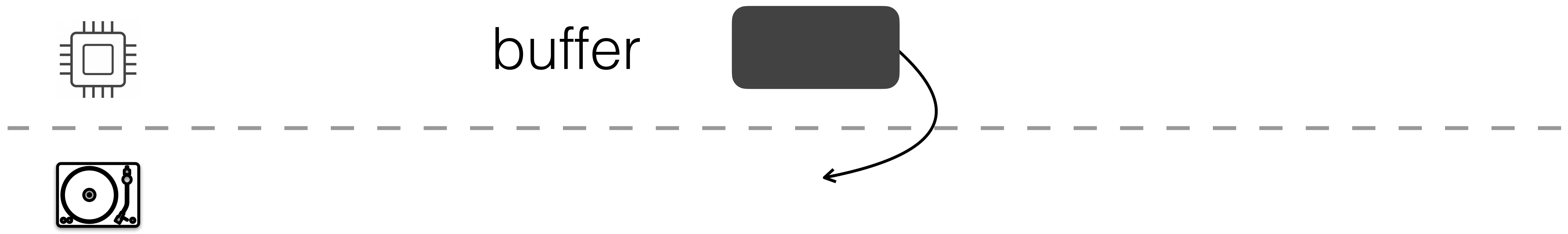

put(1)
put(6)
buffer
2

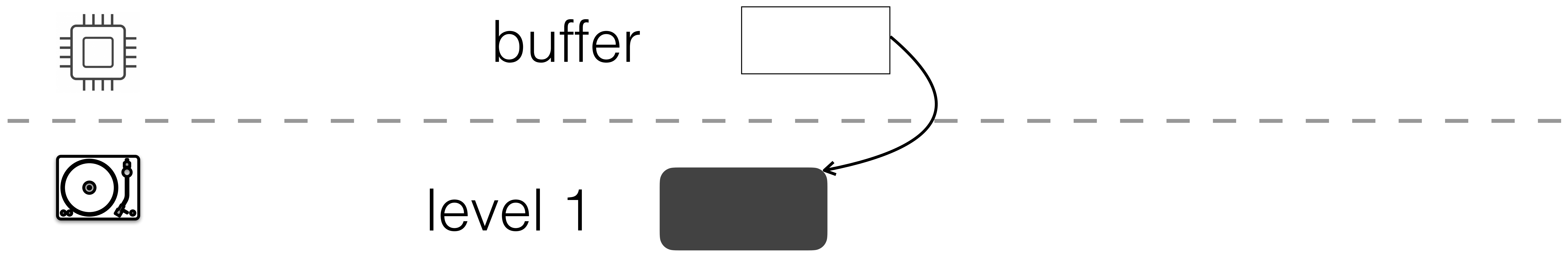

put(4)
put(1)
buffer
2
6


put(4)
buffer
2
6
1

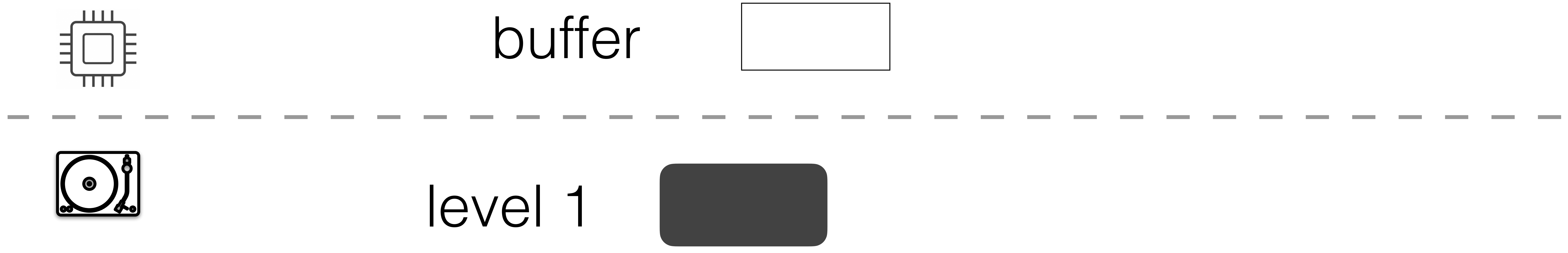

buffer
2
6
1
4

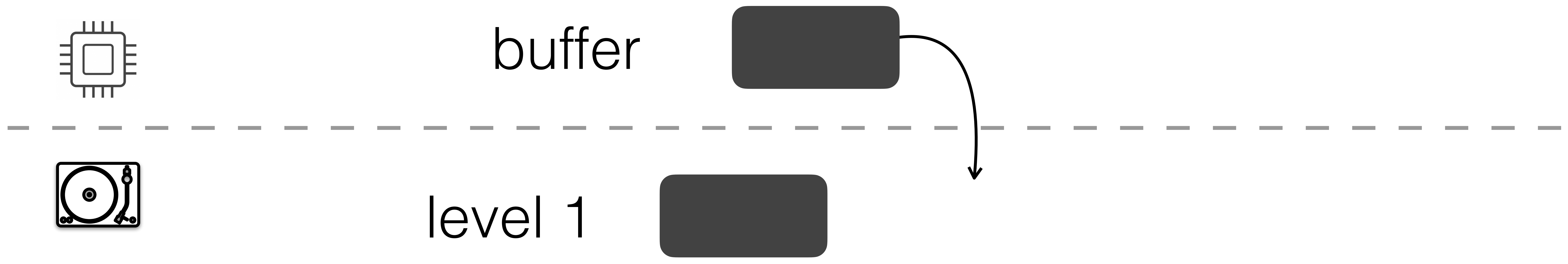

buffer
1
2
4
6

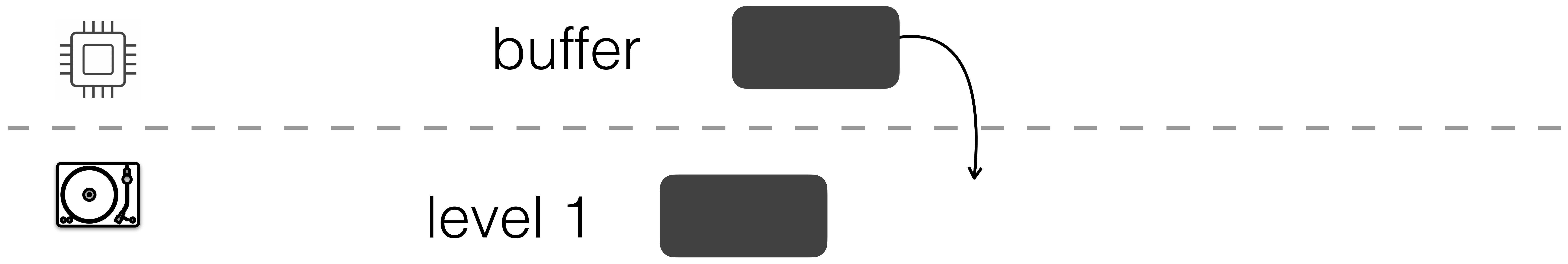

buffer

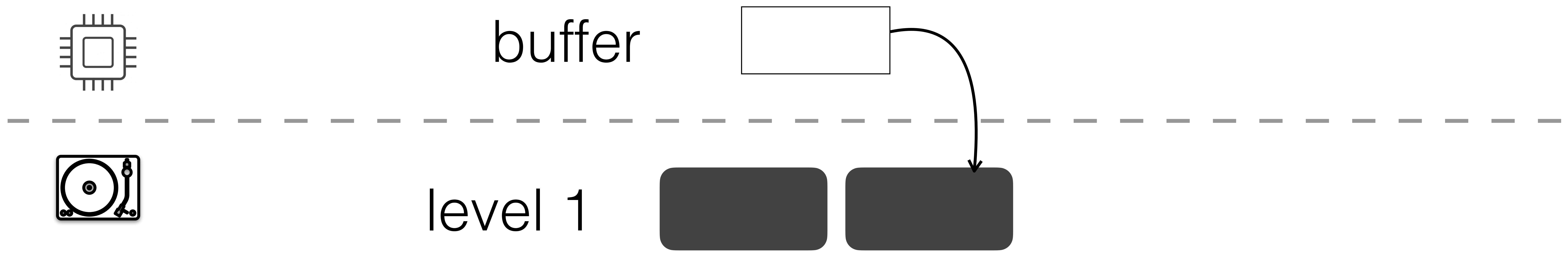


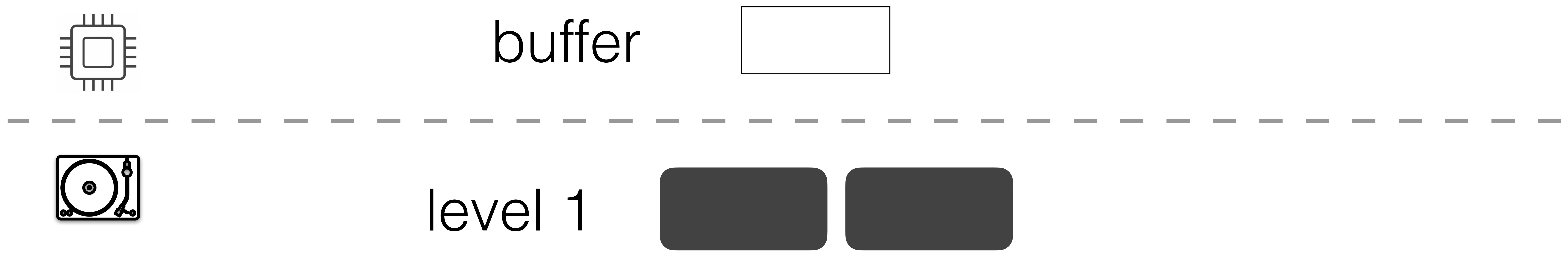
immutable sorted run

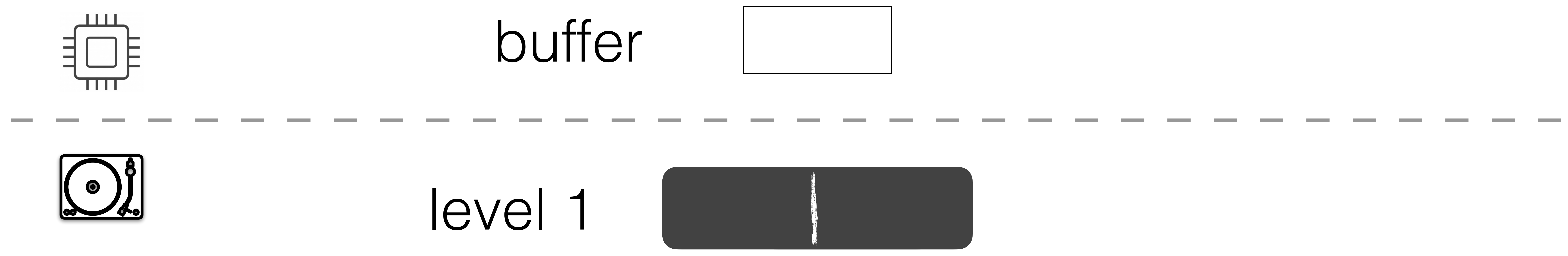

buffer
level 1

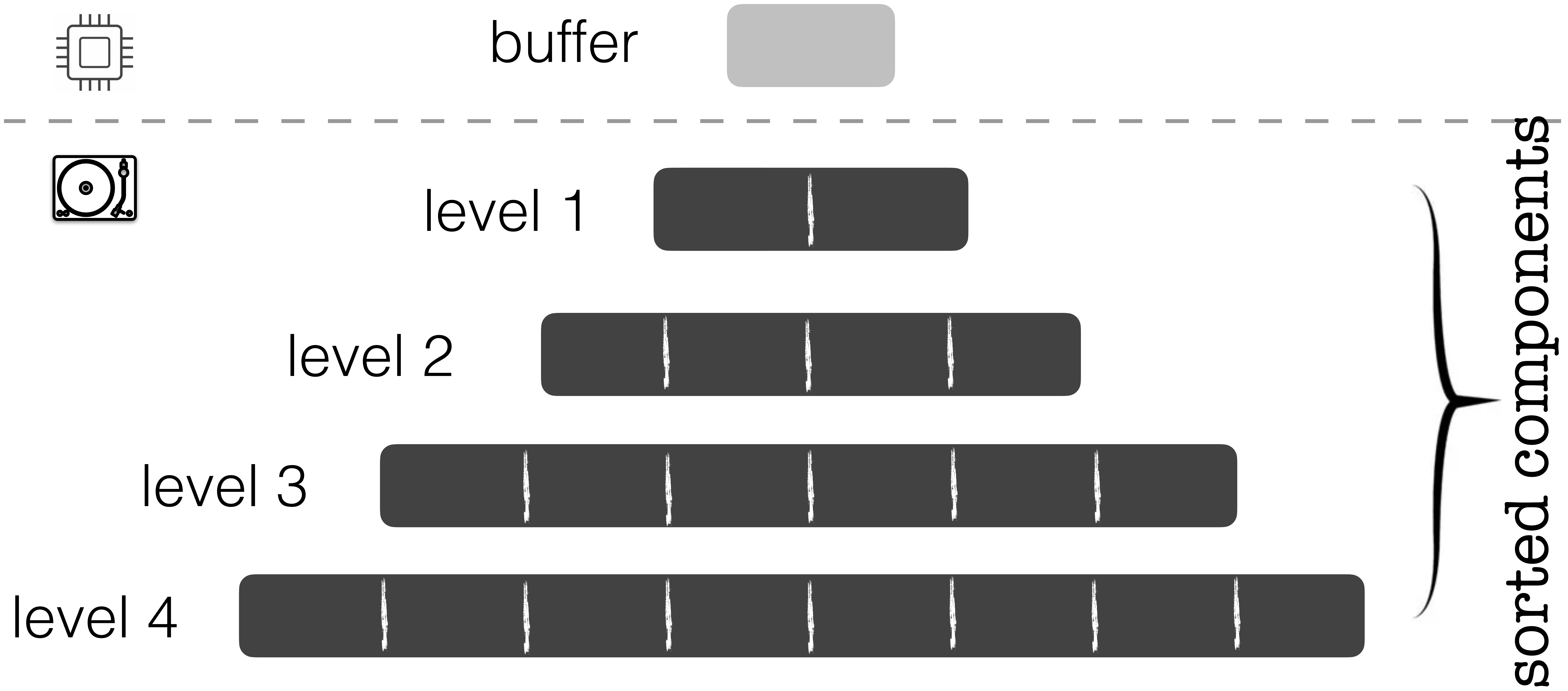

buffer
level 1

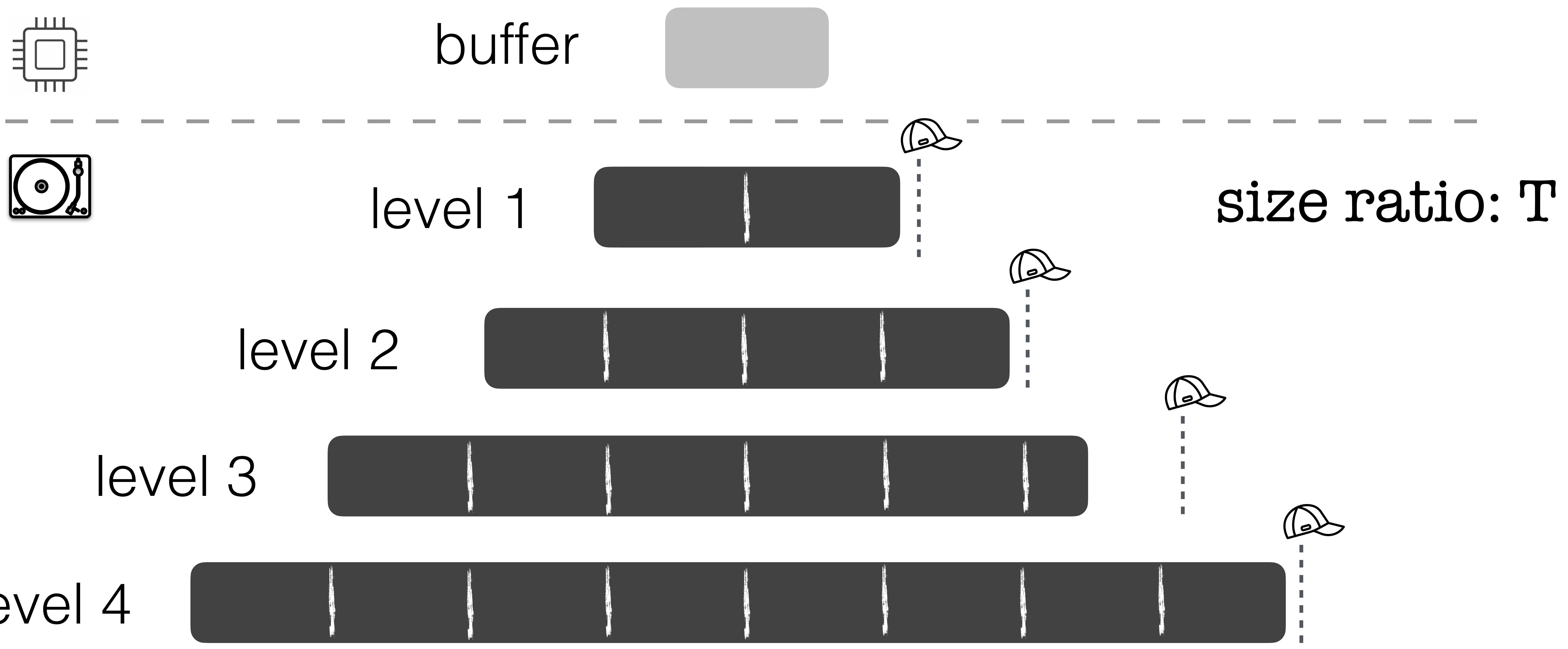

buffer
level 1

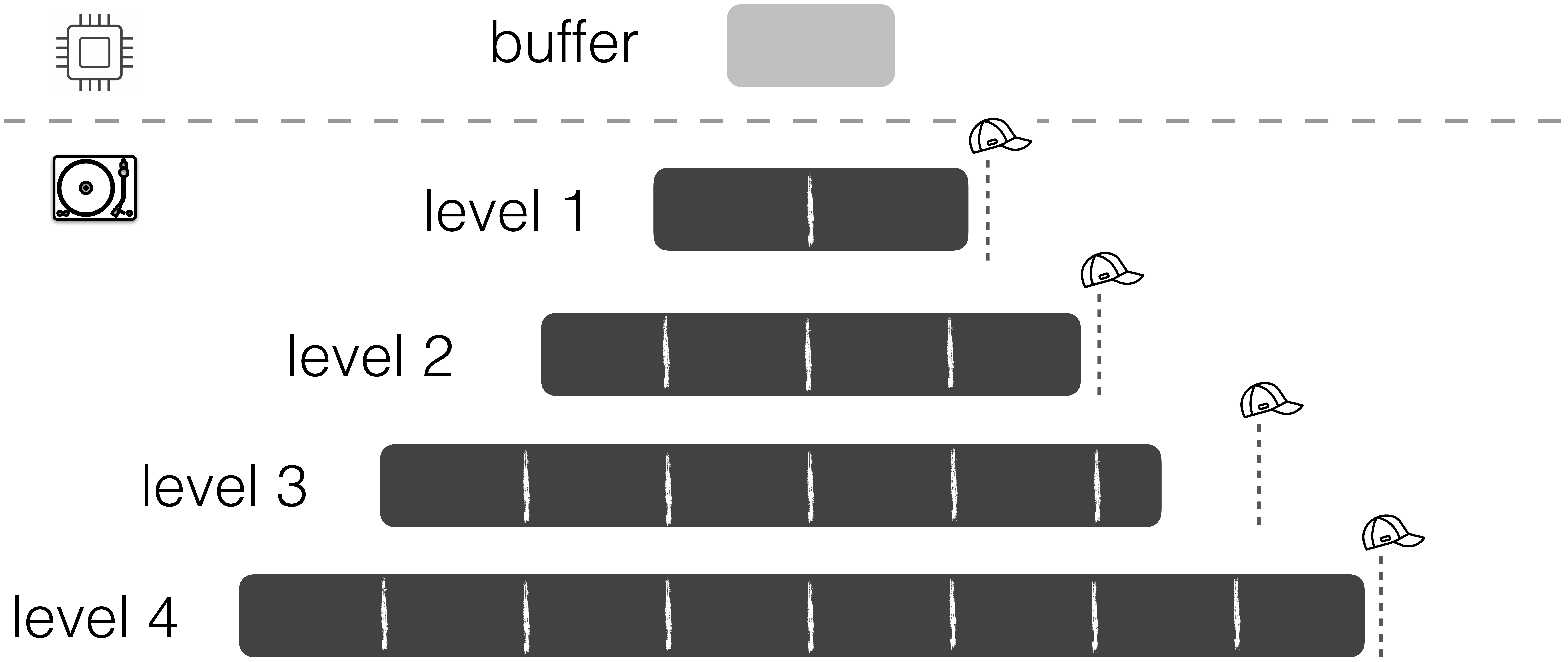

buffer
level 1

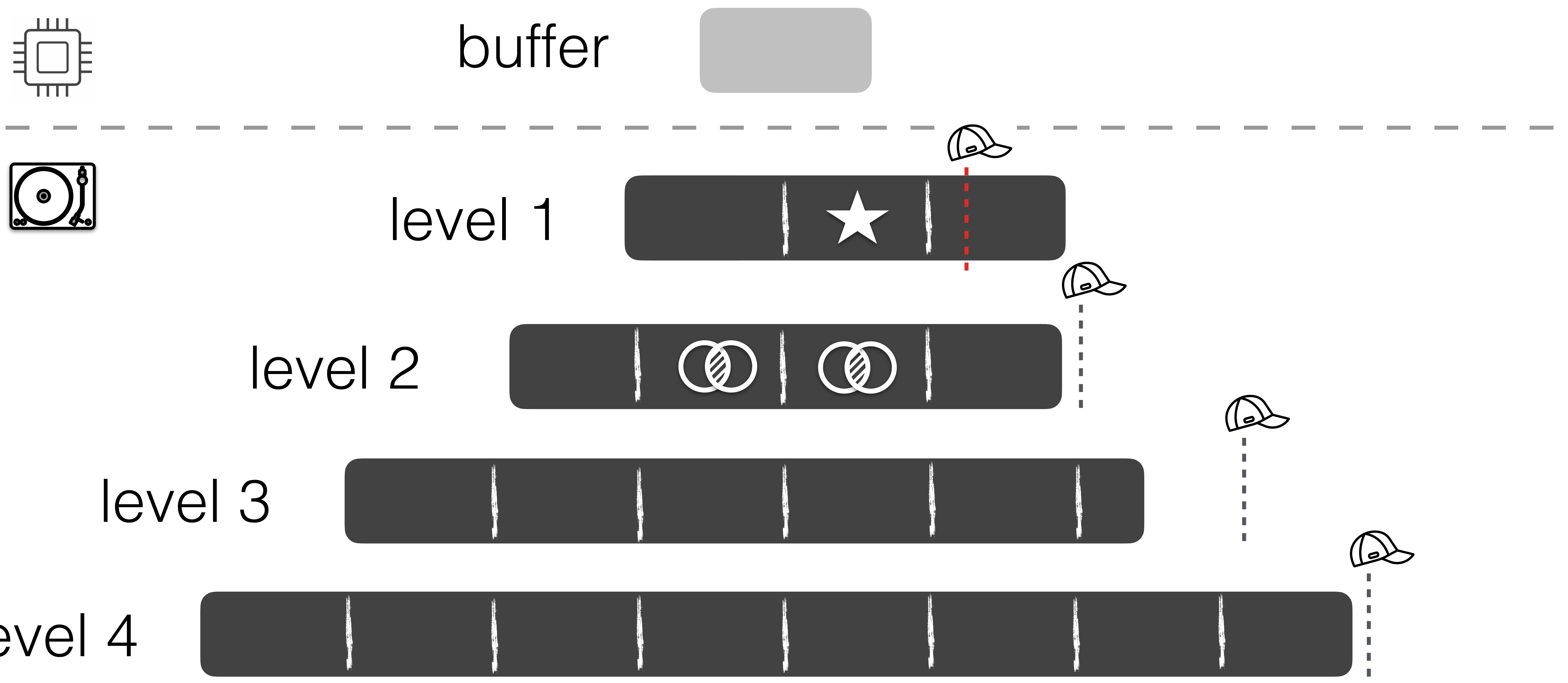

buffer
level 1

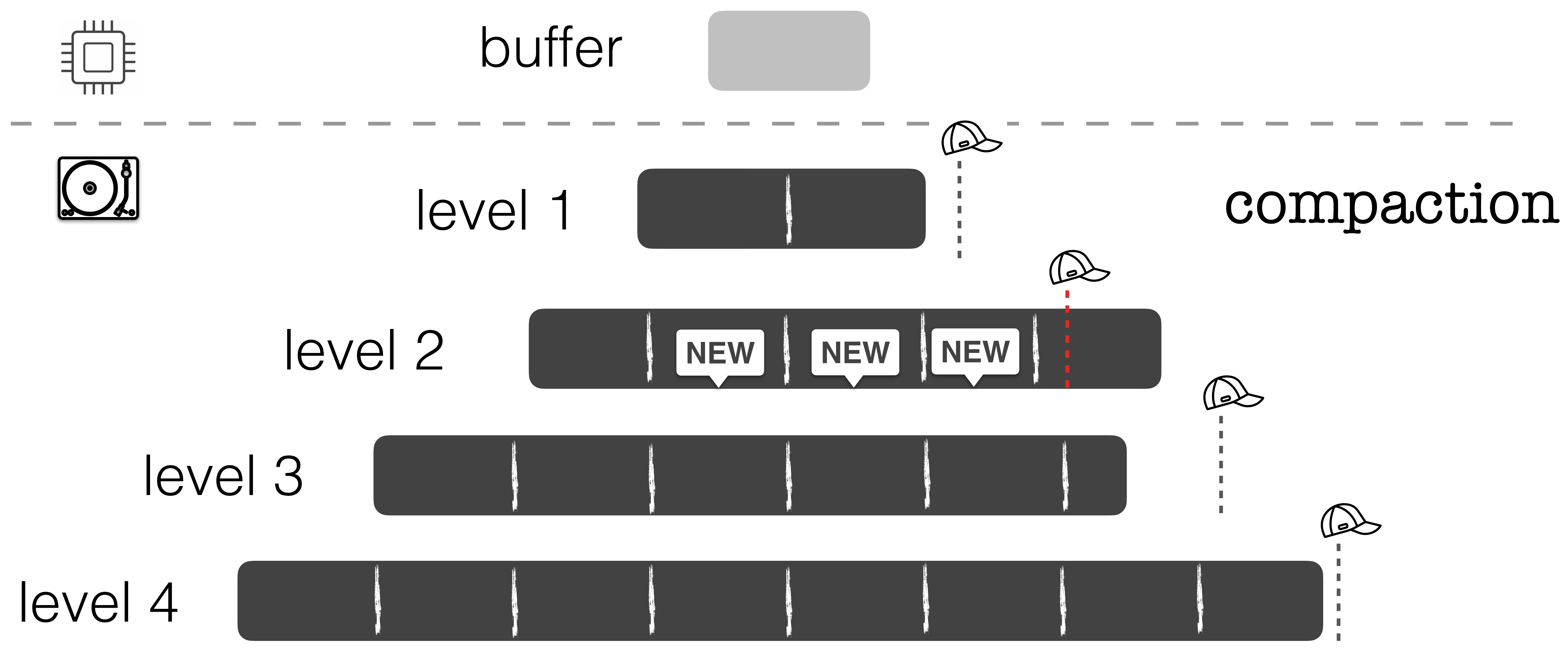

buffer
level 1

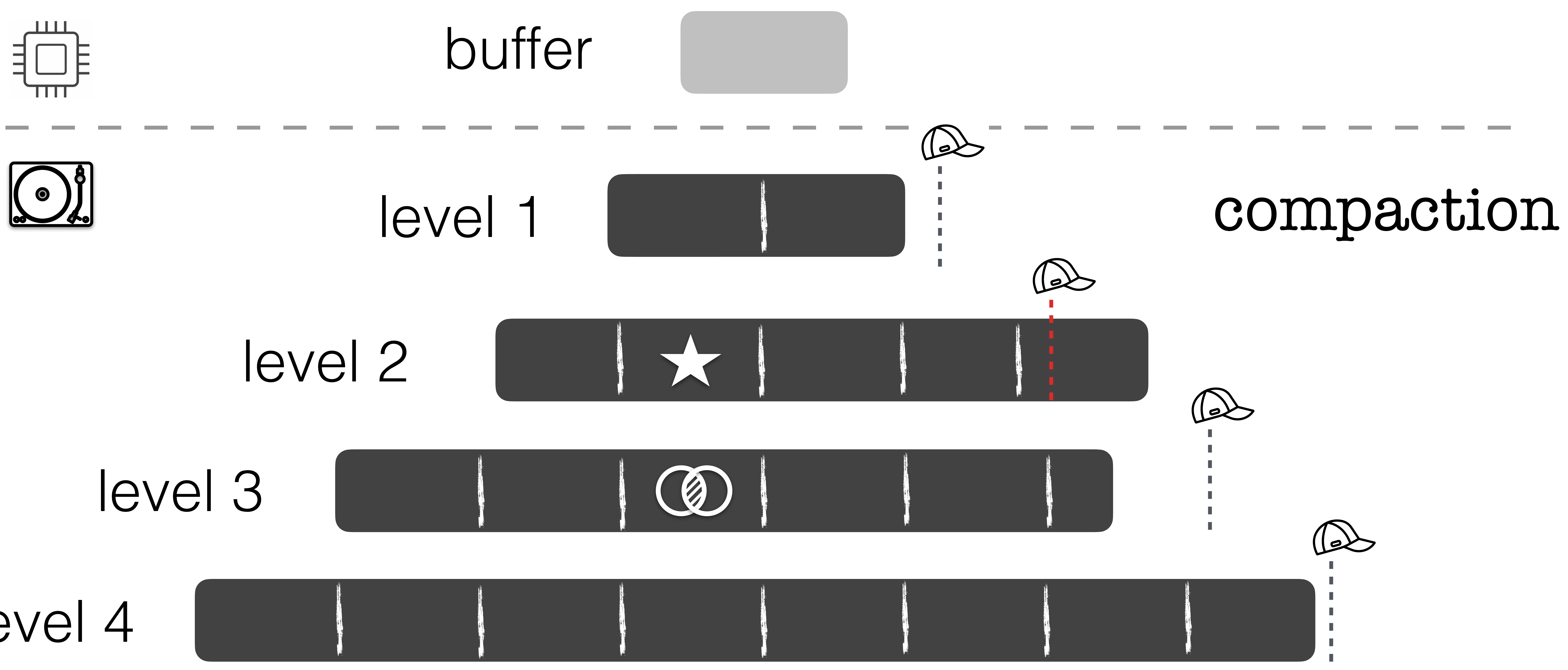

buffer
level 1
level 2
level 3
level 4
sorted components

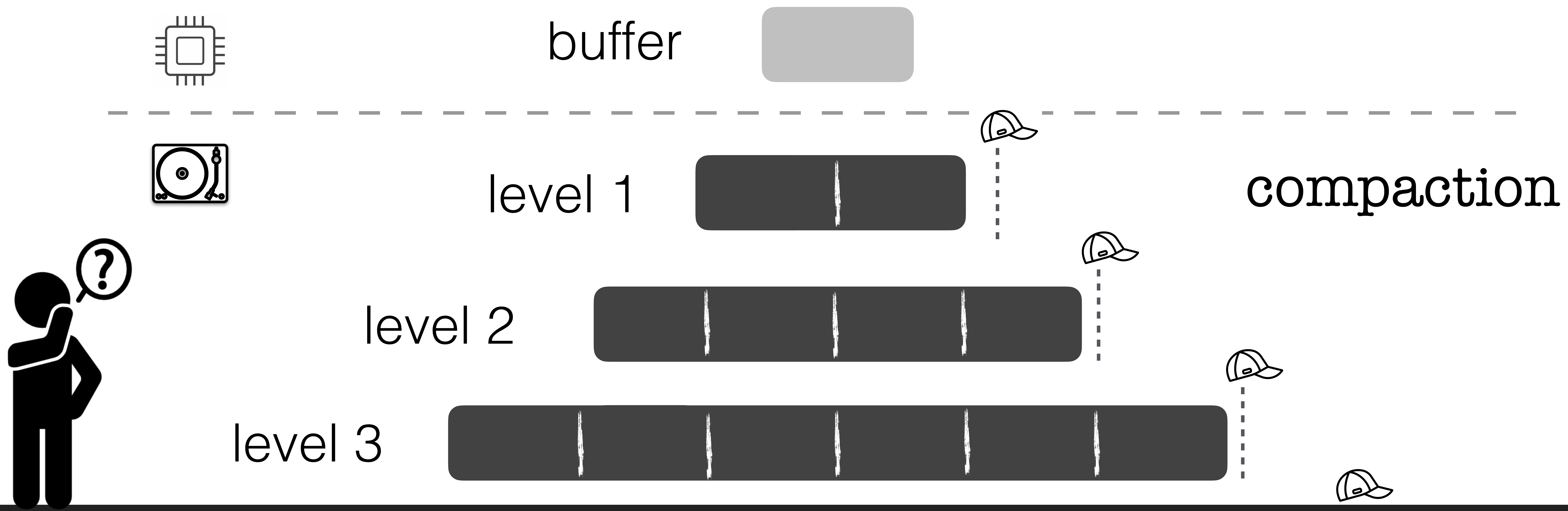

buffer
level 1
level 2
level 3
level 4
size ratio: T


buffer
level 1
level 2
level 3
level 4


buffer
level 1
level 2
level 3
level 4


buffer
level 1
level 2
NEW
NEW
NEW
level 3
level 4
compaction


buffer
level 1
level 2
level 3
level 4
compaction



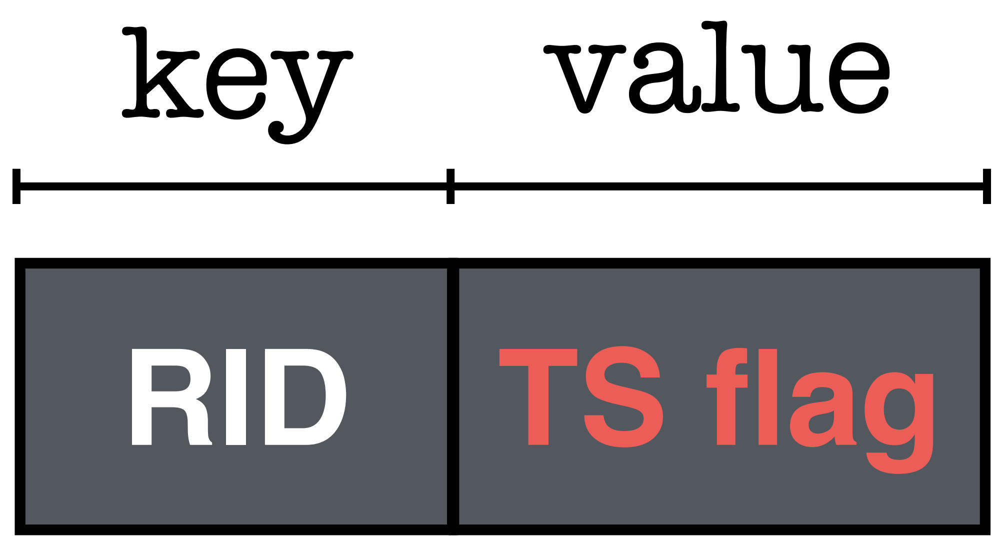
So, what about deletes?

Deletes in LSMs

delete := insert tombstone

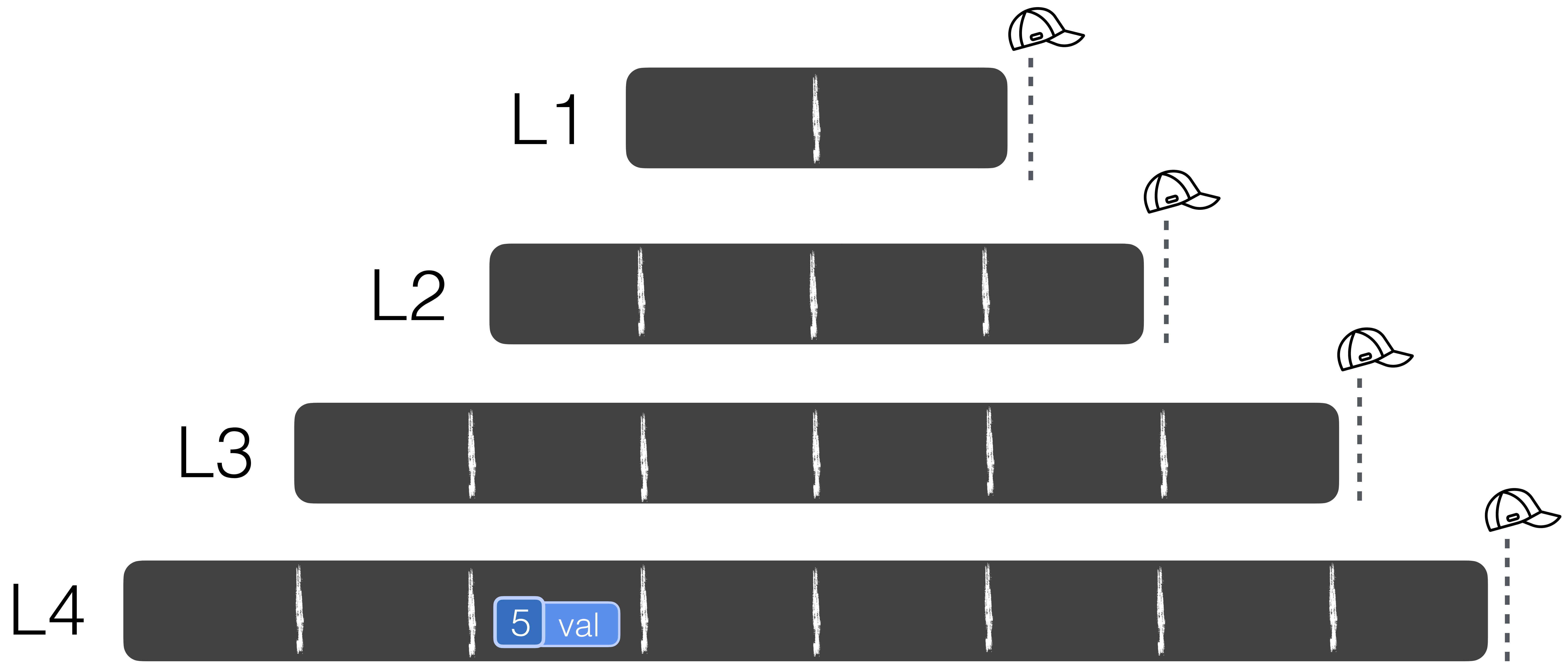
Deletes in LSMs

delete := insert tombstone



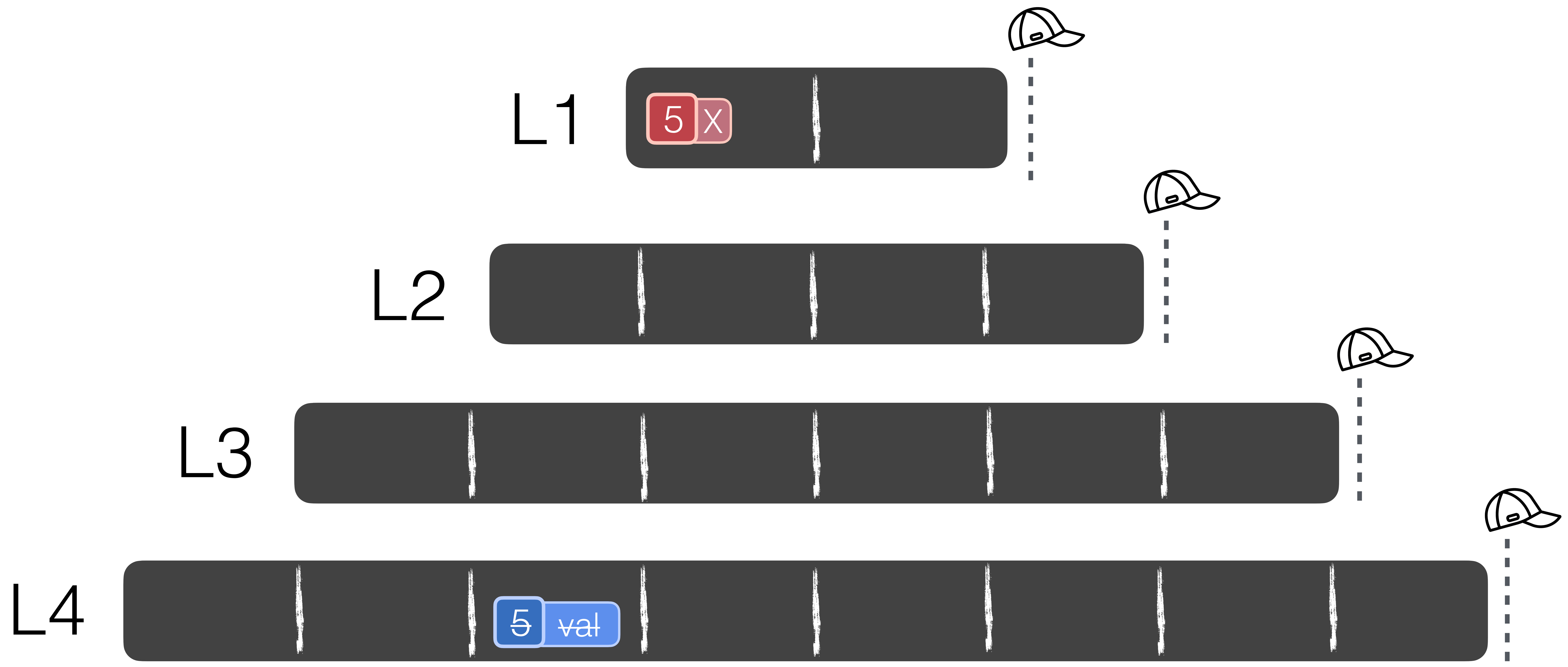
Deletes in LSMs

delete(5)



Deletes in LSMs

delete(5)

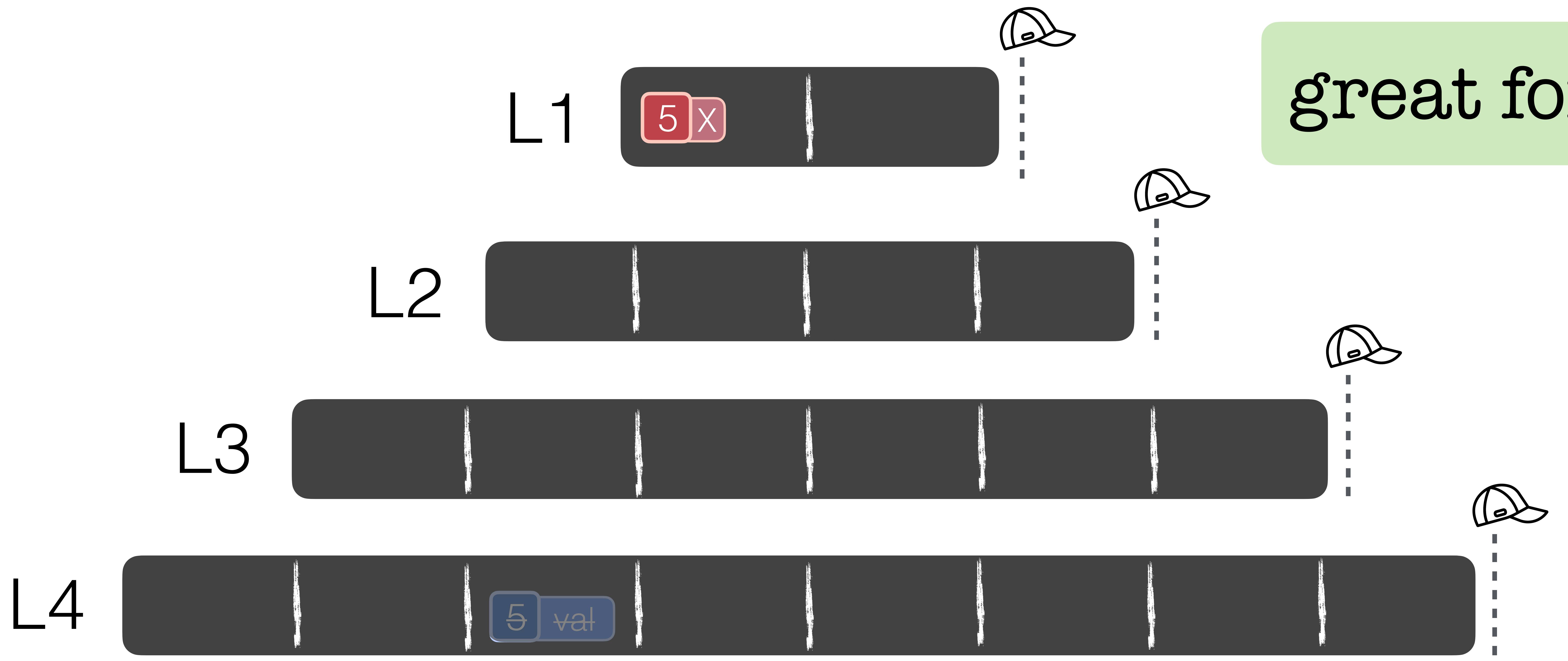


Deletes in LSMs

delete(5)

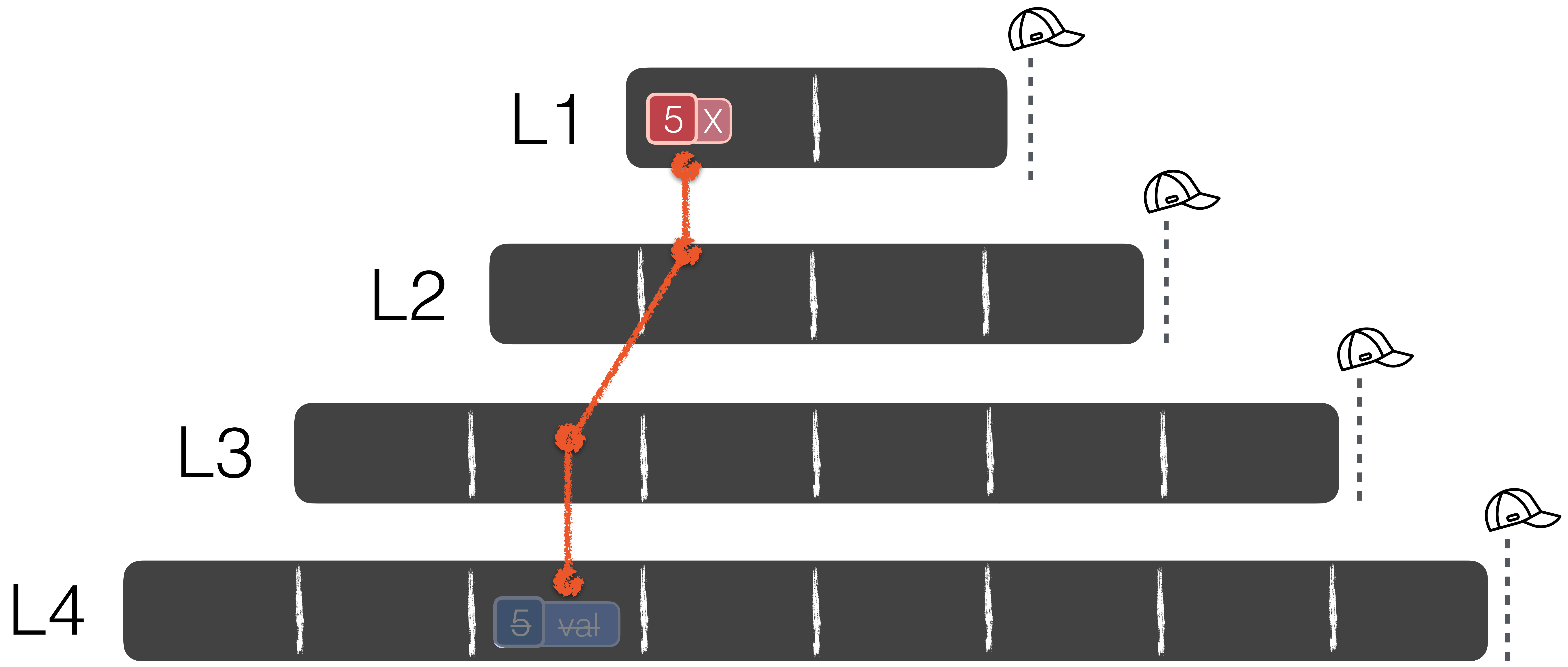
no support for timely delete persistence

great for inserts



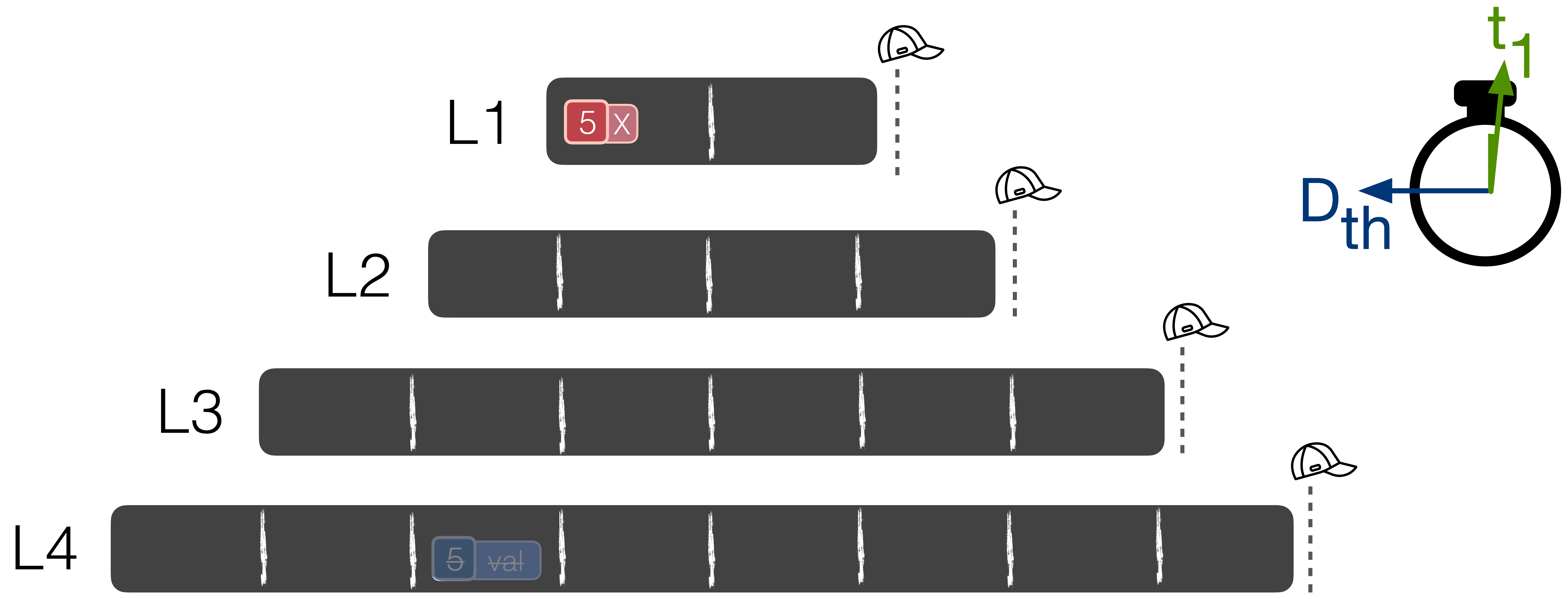
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



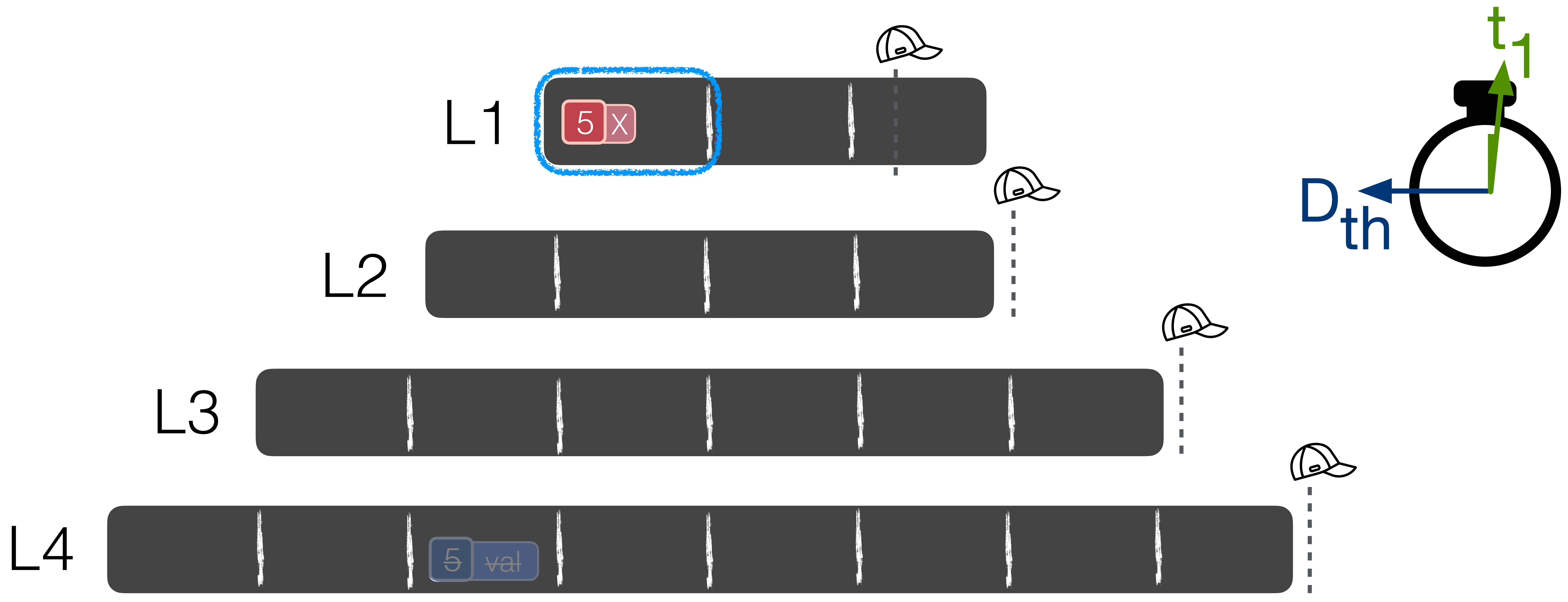
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



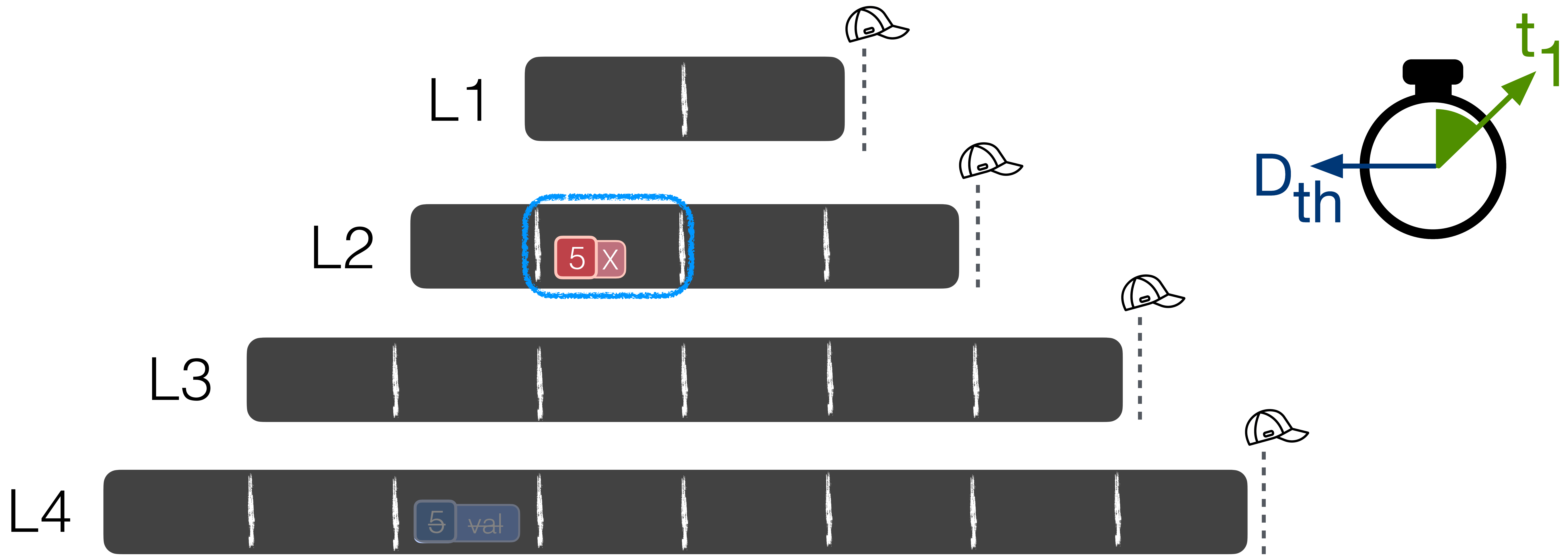
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



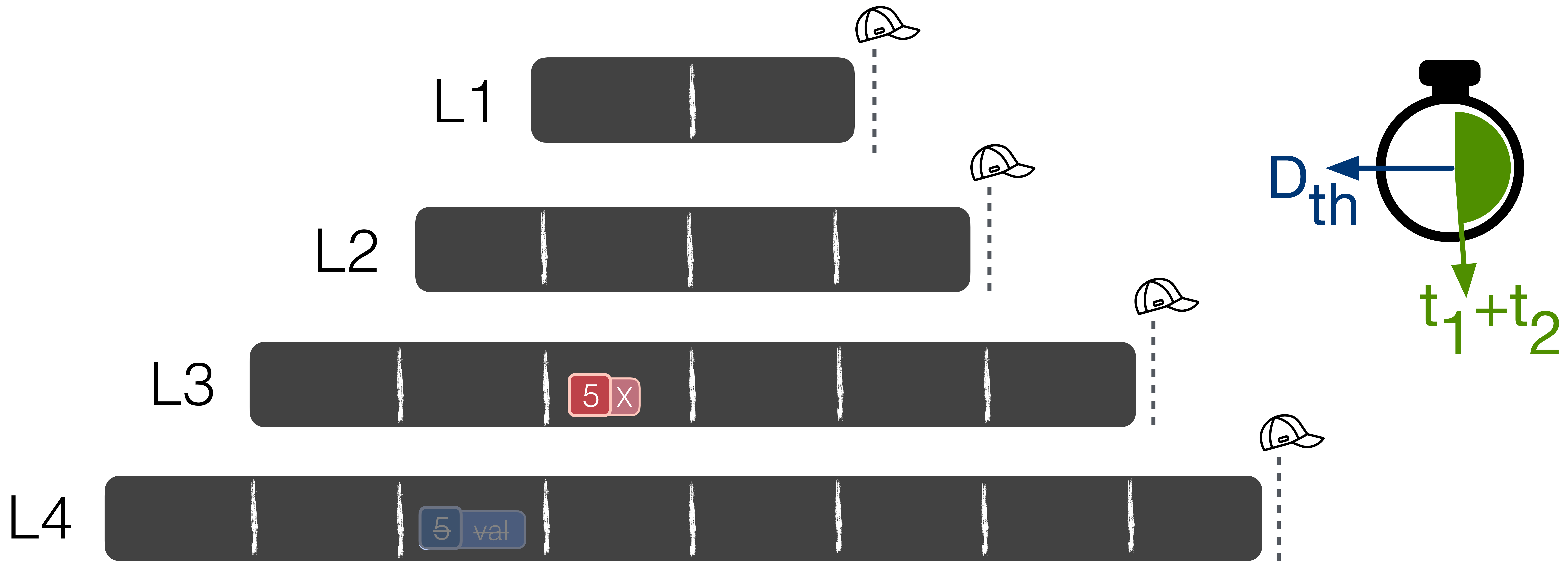
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



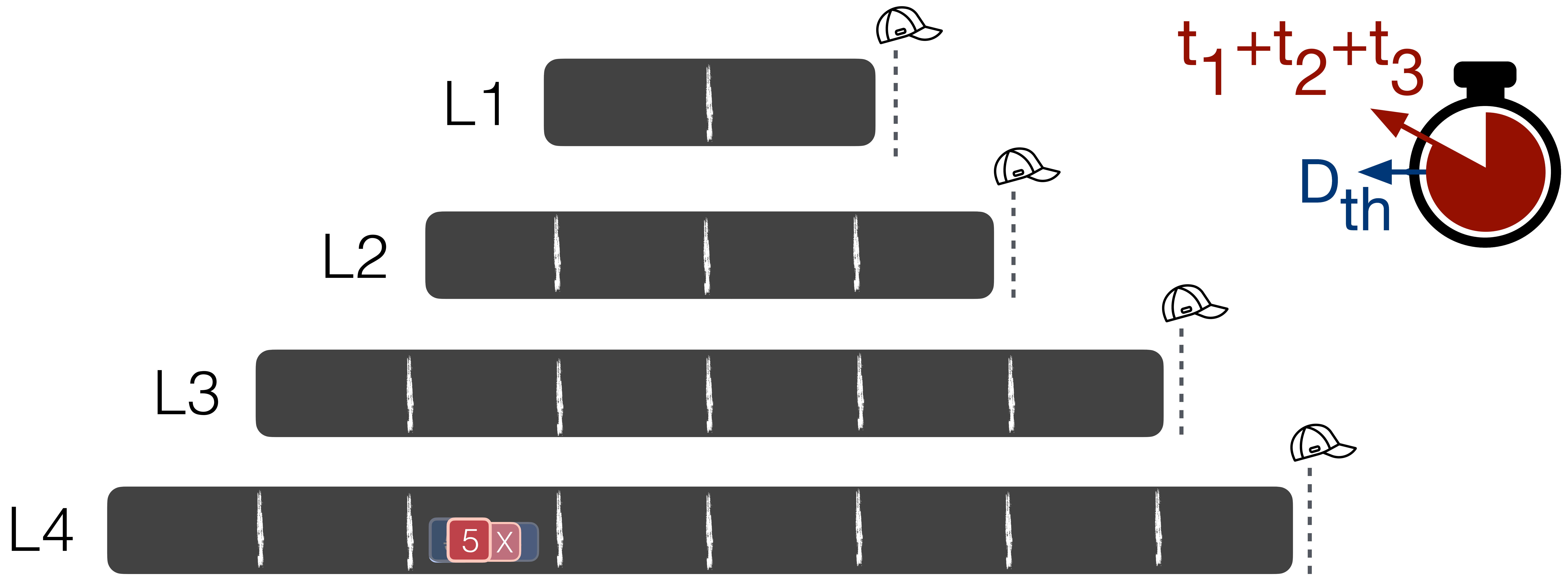
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



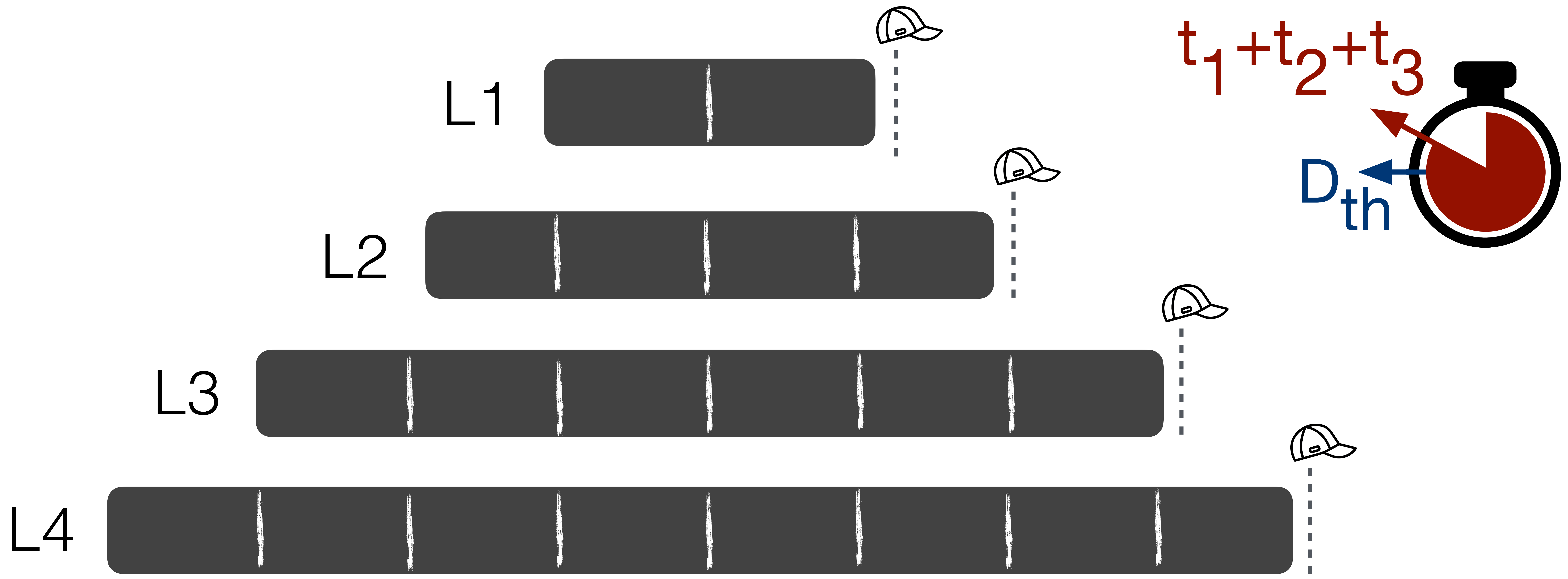
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}



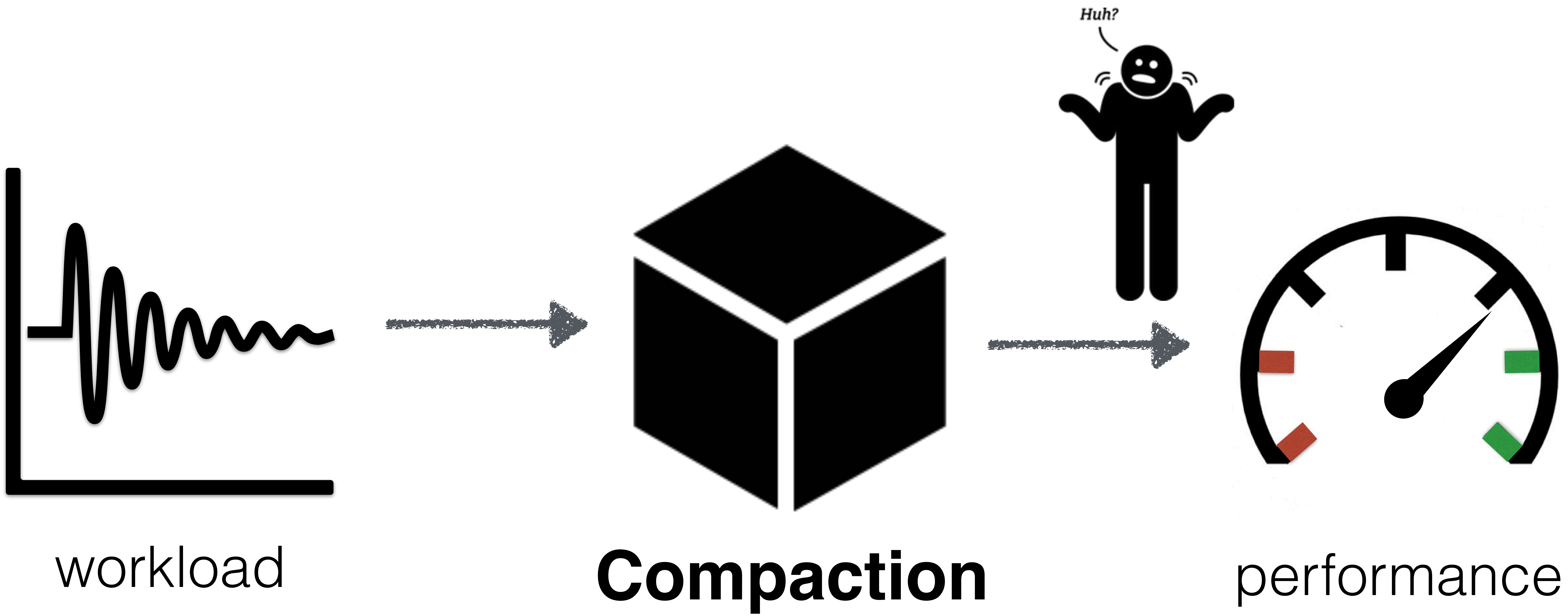
Problem: Persisting Deletes Timely

delete(5) within threshold: D_{th}

unbounded delete persistence latency

- File picking policy
- Tree shape
- Ingestion rate

Intuition: Compaction holds the key!


Huh?
workload
Compaction
performance

What are the **design choices**?

How does a choice **affect performance**?

Compaction

VLDB 2021

SIGMOD 2022-a

VLDB 2021
SIGMOD 2022-a

1 **How** to organize the data on device?

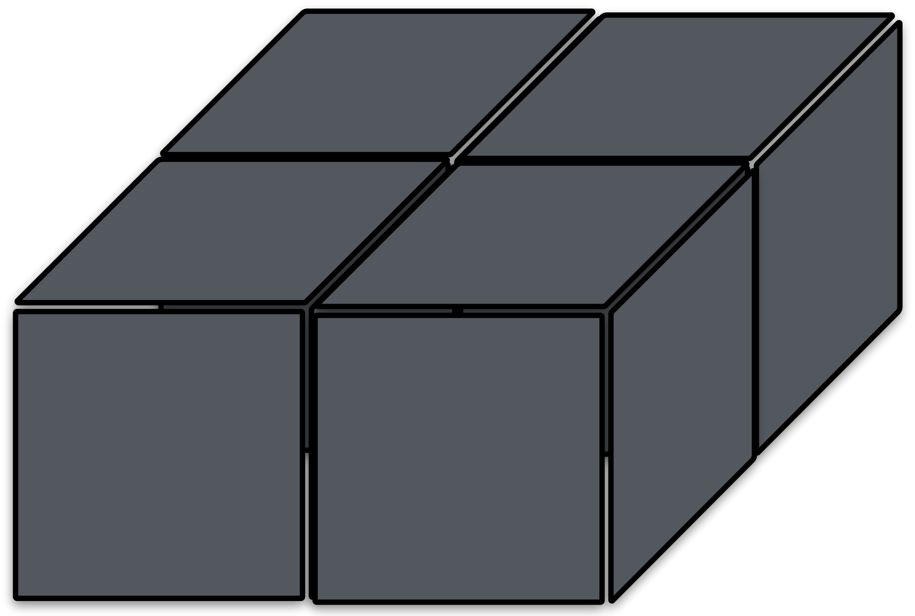
2 **How much** data to move at-a-time?

3 **Which** block of data to be moved?

4 **When** to re-organize the data layout?

VLDB 2021
SIGMOD 2022-a

Data Layout	1	**How** to organize the data on device?
Compaction granularity	2	**How much** data to move at-a-time?
Data movement policy	3	**Which** block of data to be moved?
Trigger	4	**When** to re-organize the data layout?



any existing + completely new

compaction strategies

Database	Data layout	Compaction Trigger					Compaction Granularity				Data Movement Policy							
		Level saturation	#Sorted runs	File staleness	Space amp.	Tombstone-TTL	Level	Sorted run	File (single)	File (multiple)	Round-robin	Least overlap (+1)	Least overlap (+2)	Coldest file	Oldest file	Tombstone density	Expired TS-TTL	N/A (entire level)
RocksDB [30], Monkey [22]	Leveling / 1-Leveling	✓		✓					✓	✓		✓		✓	✓	✓		
	Tiering		✓		✓	✓		✓										✓
LevelDB [32], Monkey (J.) [21]	Leveling	✓							✓		✓	✓	✓					
SlimDB [47]	Tiering	✓							✓	✓								✓
Dostoevsky [23]	*L*-leveling	✓^{L}	✓^{T}				✓^{L}	✓^{T}				✓^{L}						✓^{T}
LSM-Bush [24]	Hybrid leveling	✓^{L}	✓^{T}				✓^{L}	✓^{T}				✓^{L}						✓^{T}
Silk [11], Silk+ [12]	Leveling	✓							✓	✓	✓							
HyperLevelDB [35]	Leveling	✓							✓		✓	✓	✓					
PebblesDB [46]	Hybrid leveling	✓							✓	✓								✓
Cassandra [8]	Tiering		✓	✓		✓		✓										✓
	Leveling	✓				✓			✓	✓		✓				✓	✓	
WiredTiger [62]	Leveling	✓					✓											✓
X-Engine [34], Leaper [63]	Hybrid leveling	✓							✓	✓		✓				✓		
HBase [7]	Tiering		✓					✓										✓
AsterixDB [3]	Leveling	✓					✓											✓
	Tiering		✓					✓										✓
Tarantool [57]	*L*-leveling	✓^{L}	✓^{T}				✓^{L}	✓^{T}										✓

FADE

FAst **DE**lete

family of
compaction strategies

FAst **DE**lete

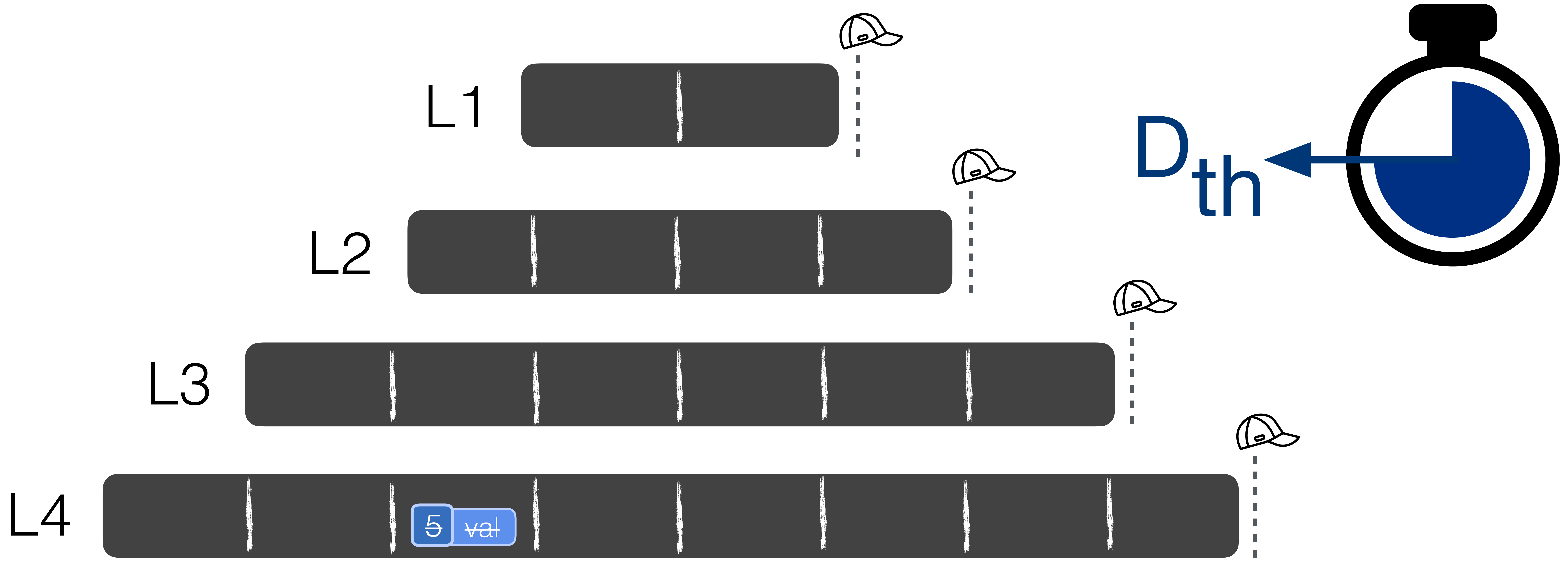
compaction trigger	compaction file picking policy

decompose tasks

piggyback

FAst **DE**lete

delete(5) within threshold: D_{th}

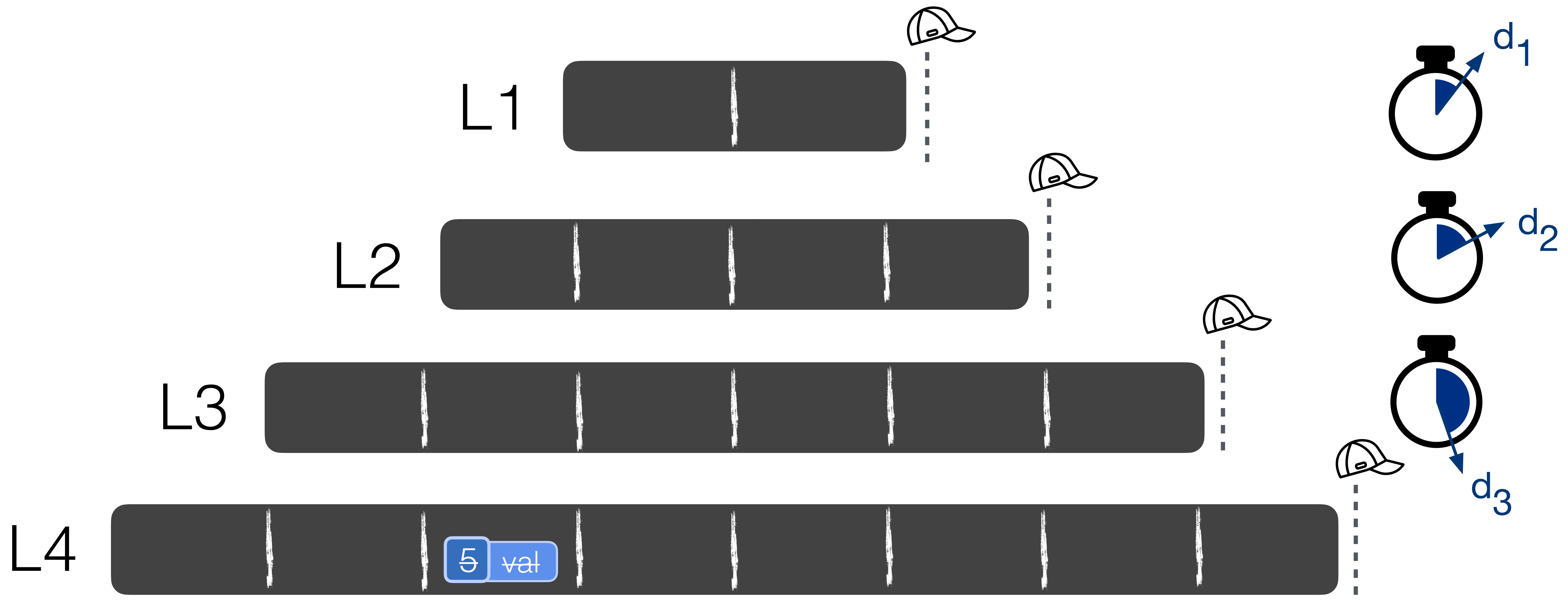


FAst DElete

delete(5) within threshold: D_{th}

$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

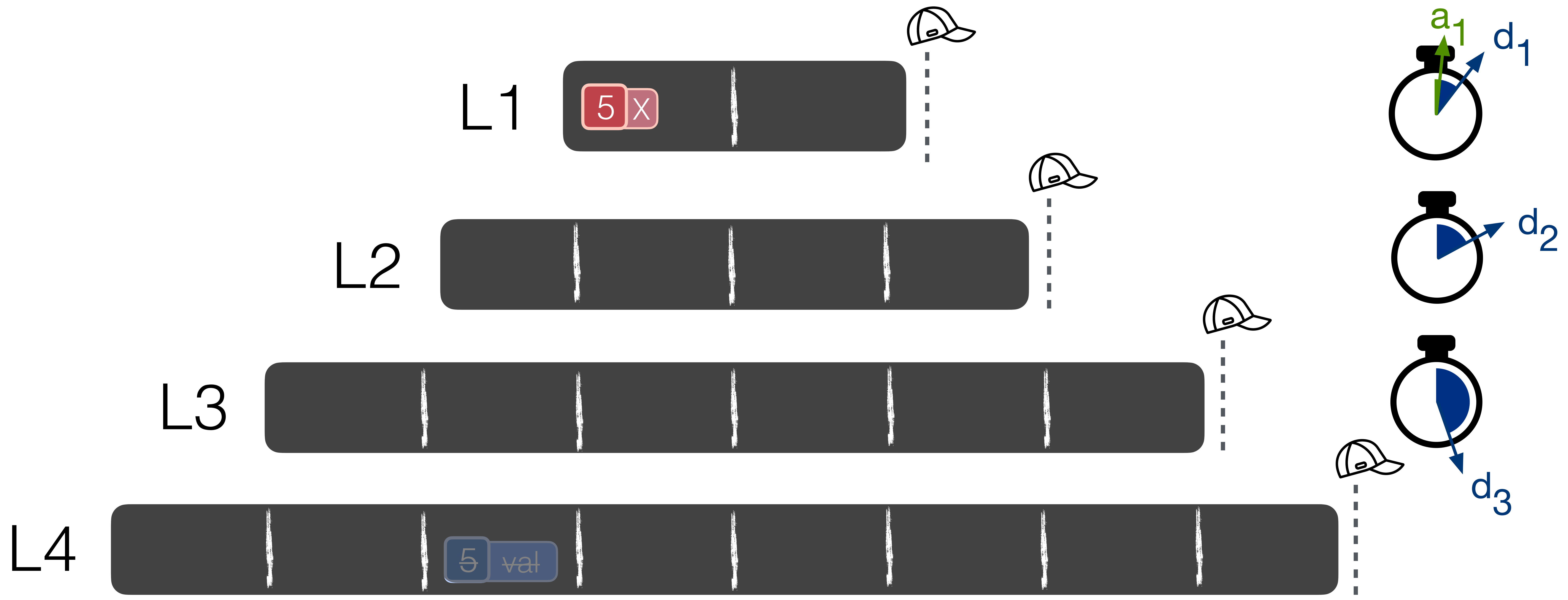


FAst **DE**lete

$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

delete(5) within threshold: D_{th}

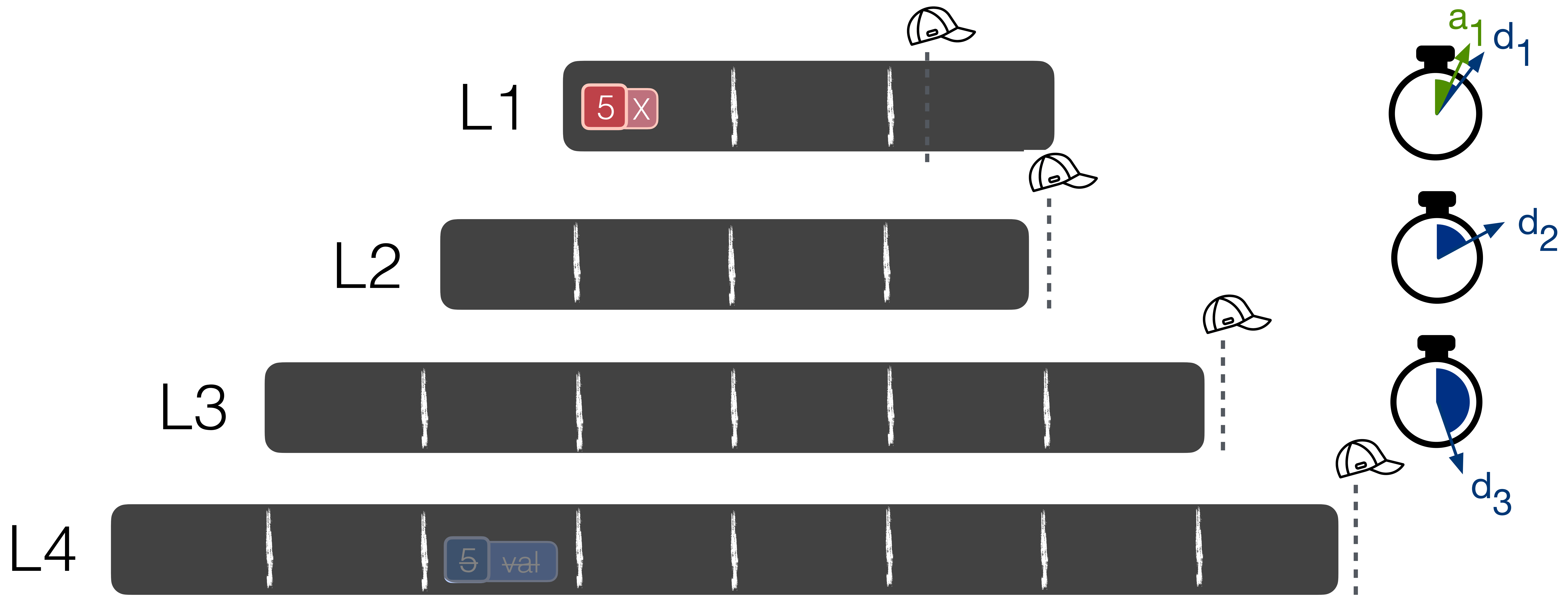


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

FAst **DE**lete

delete(5) within threshold: D_{th}

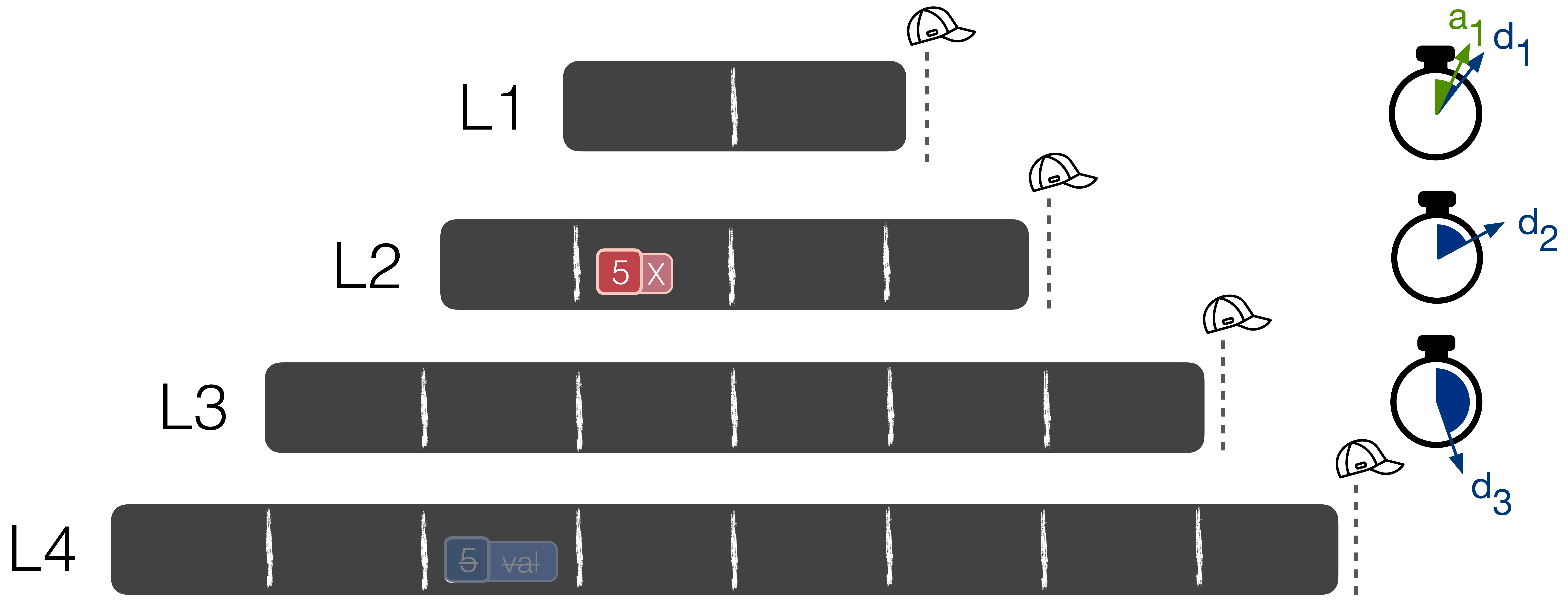


FAst **DE**lete

$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

delete(5) within threshold: D_{th}

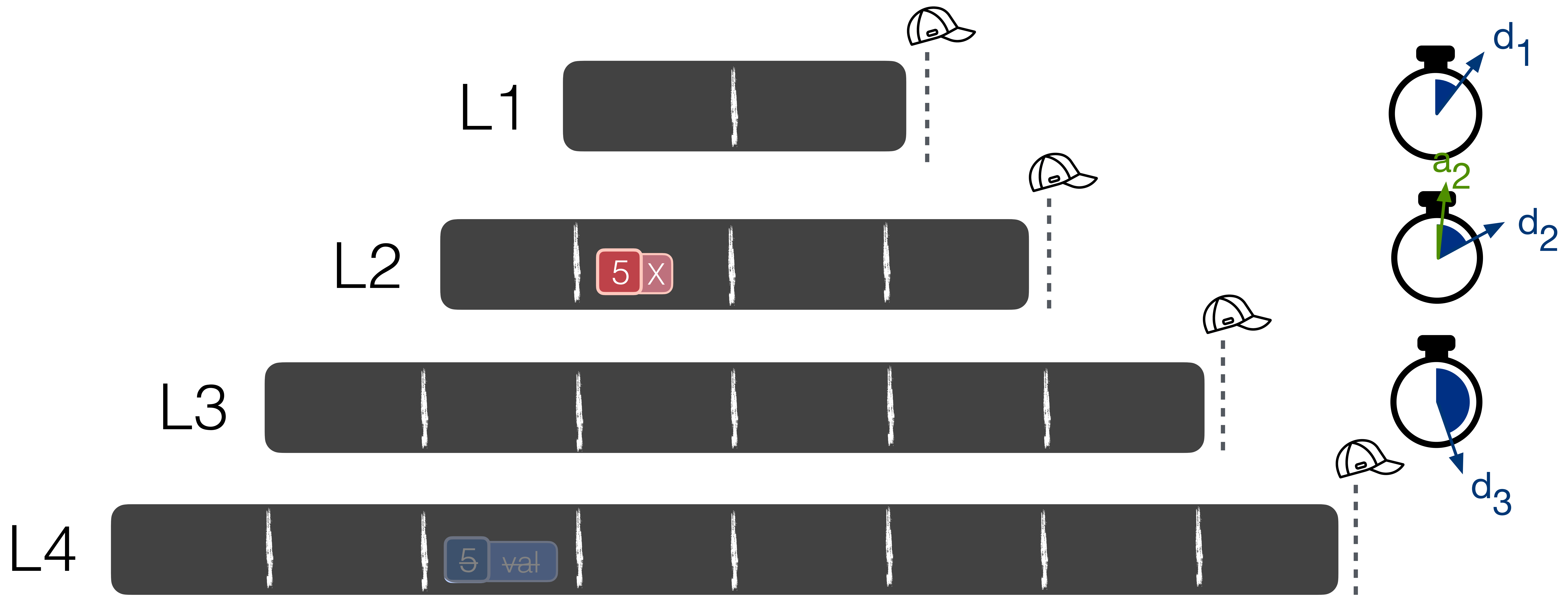


FAst **DE**lete

$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

delete(5) within threshold: D_{th}

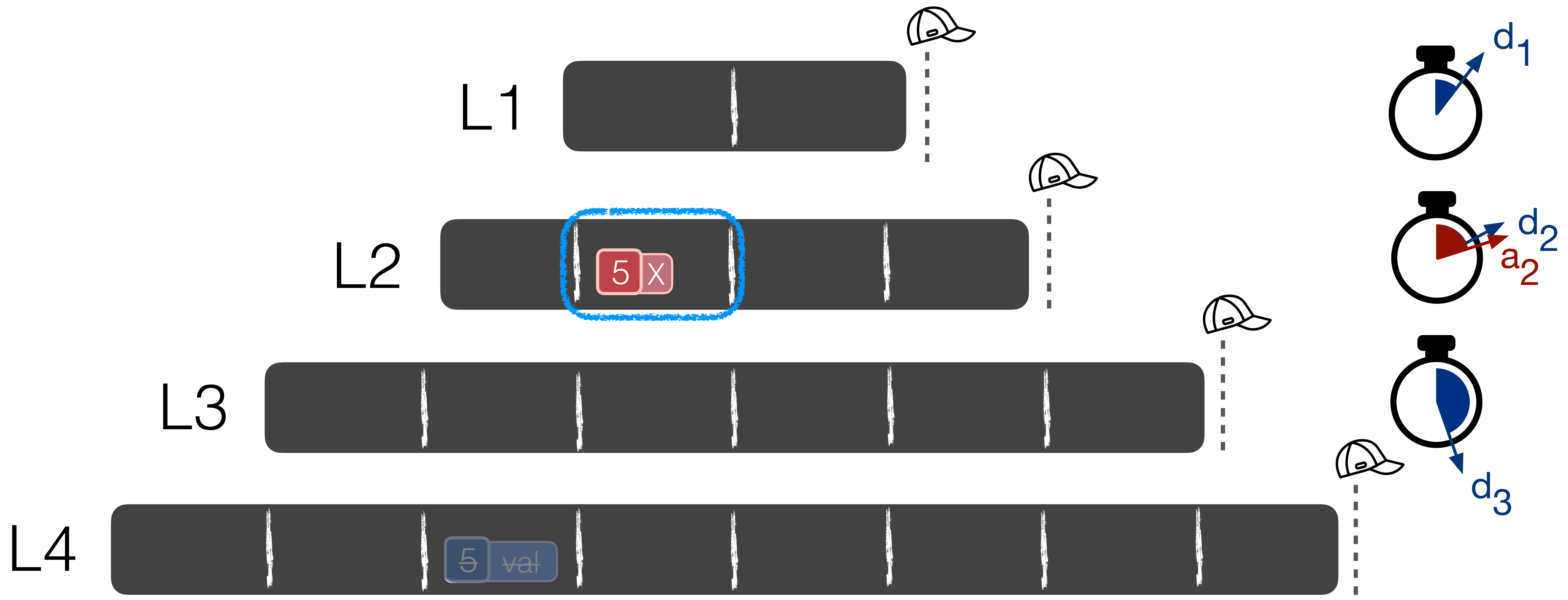


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

FAst **DE**lete

delete(5) within threshold: D_{th}

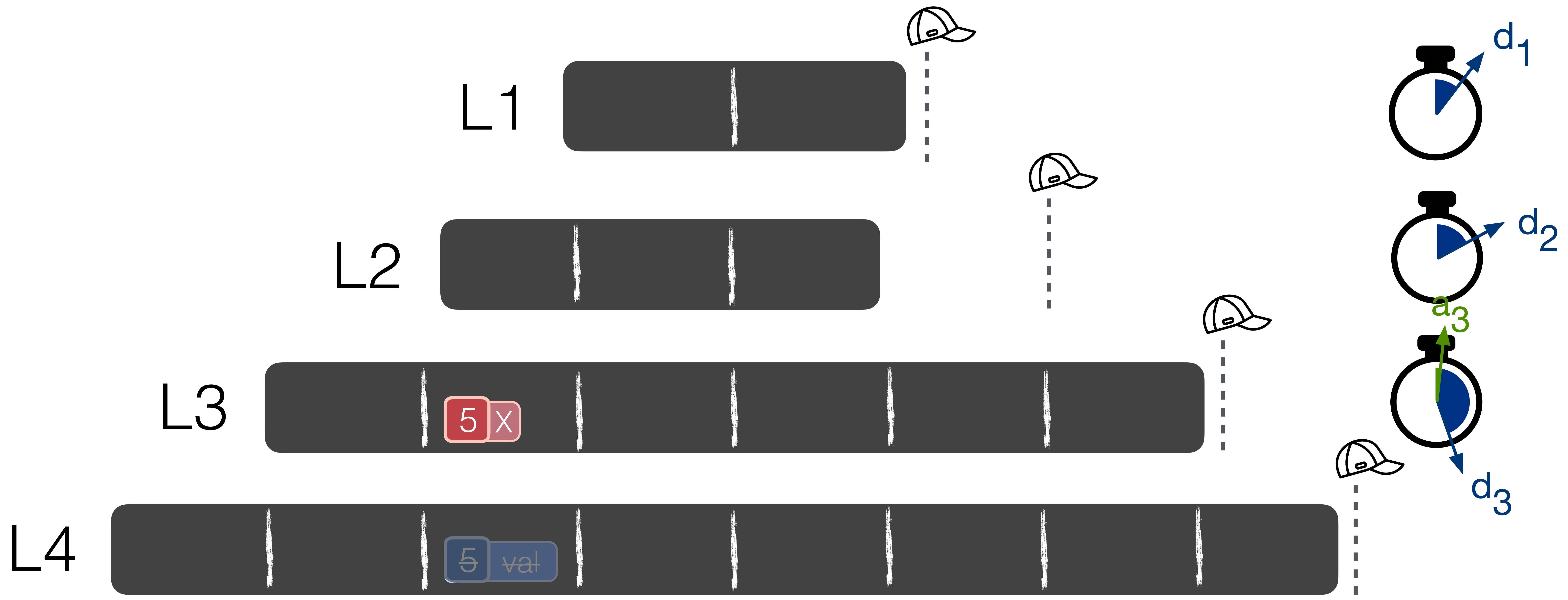


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

FAst DElete

delete(5) within threshold: D_{th}

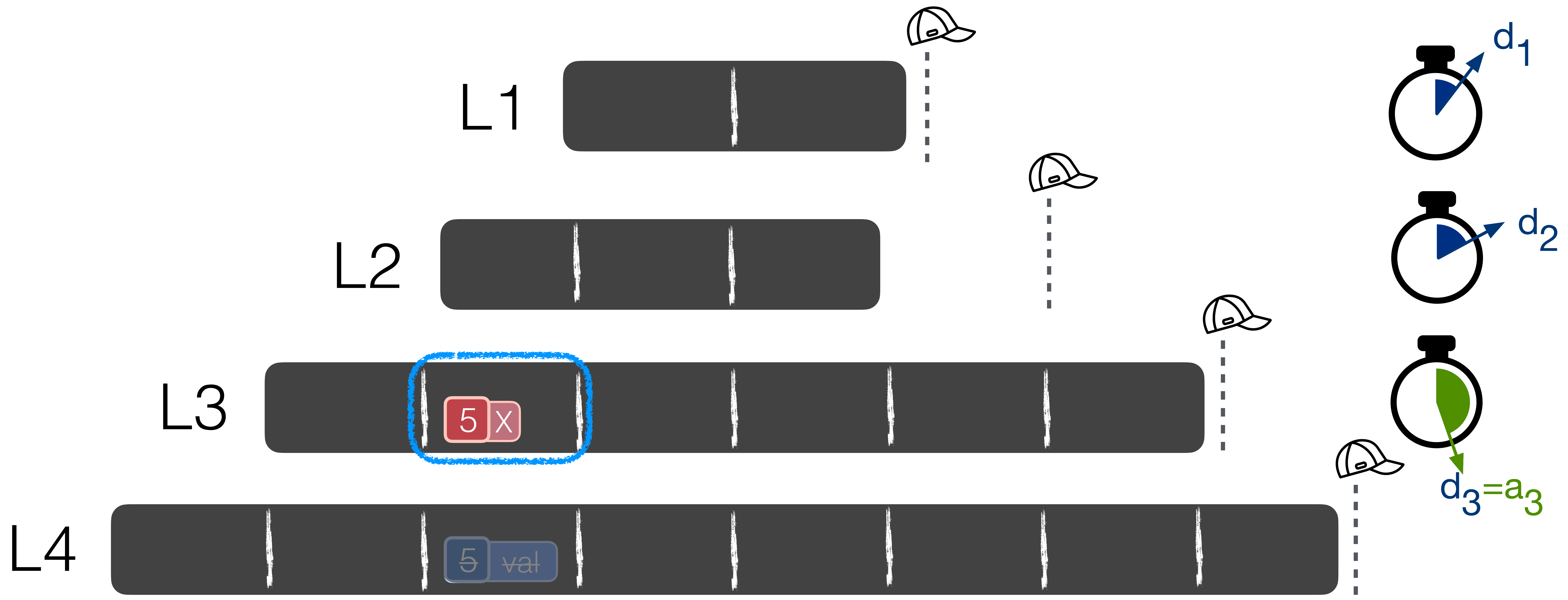


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

FAst **DE**lete

delete(5) within threshold: D_{th}

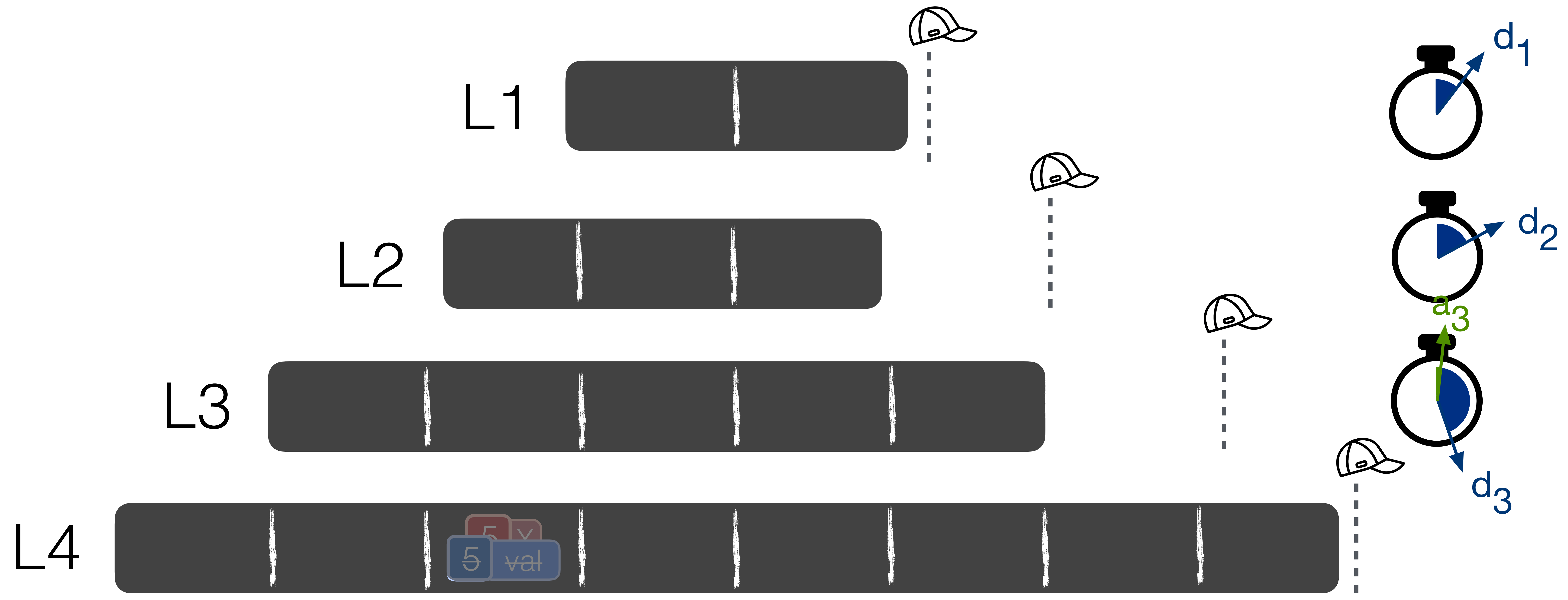


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

$$d_i = T \cdot d_{i-1}$$

FAst **DE**lete

delete(5) within threshold: D_{th}

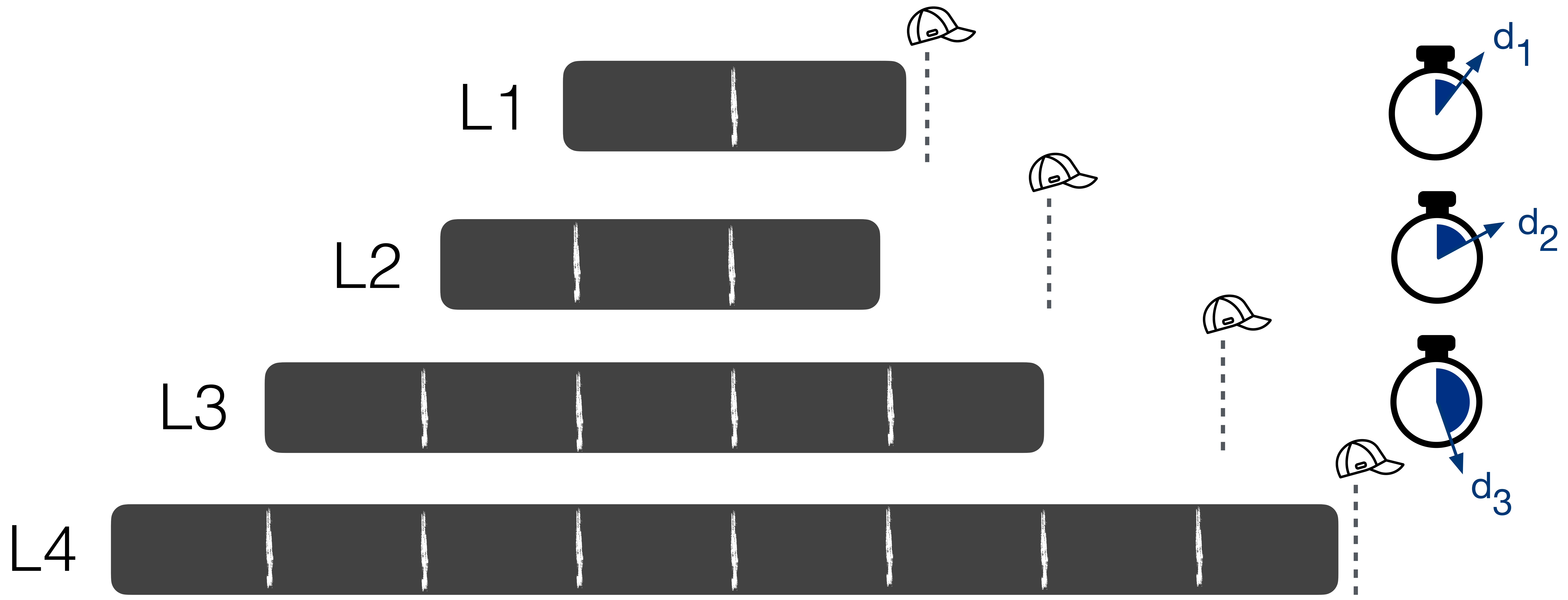


$$\sum_{i=1}^{L-1} d_i \leq D_{th}$$

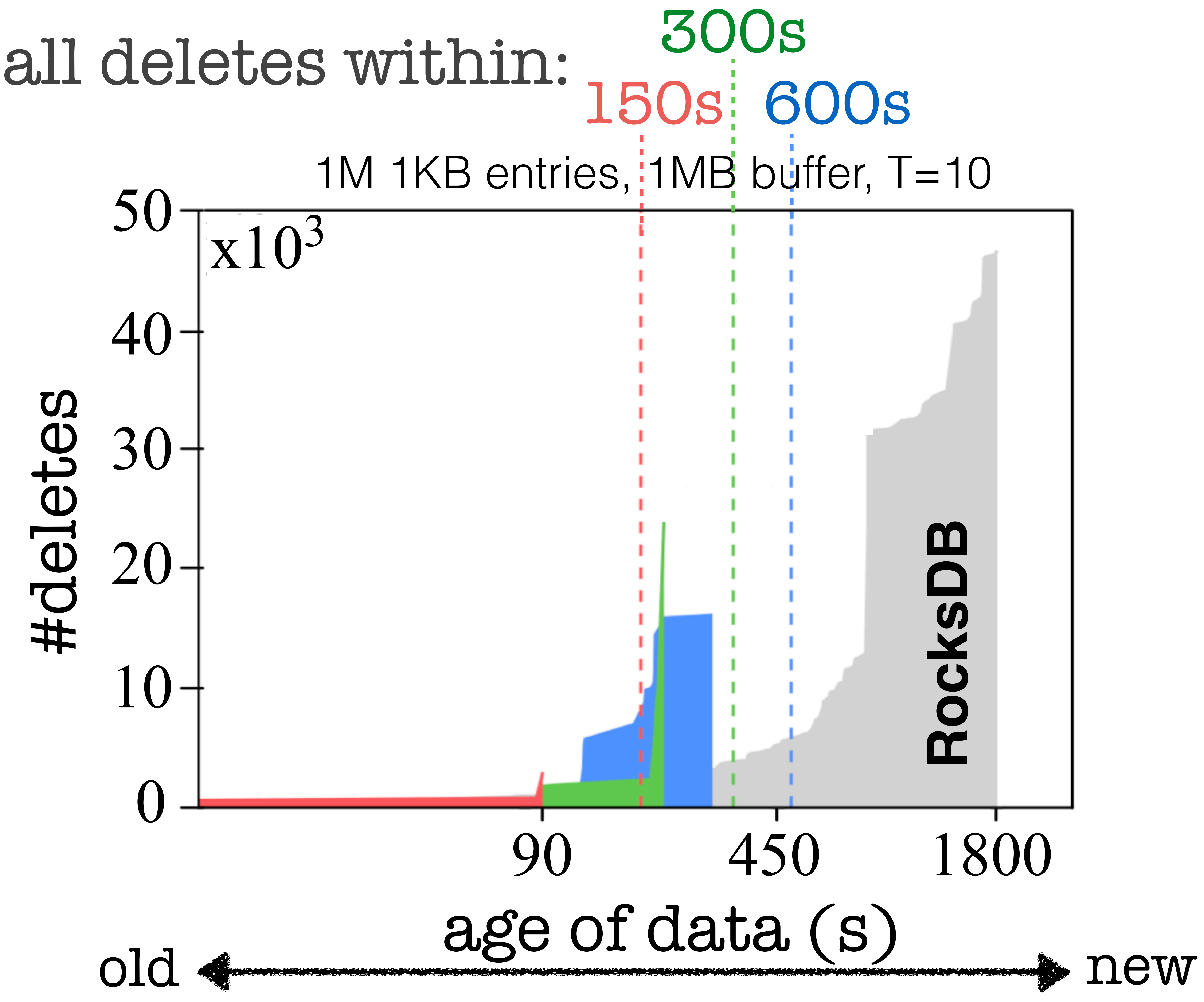
$$d_i = T \cdot d_{i-1}$$

FAst **DE**lete

delete(5) within threshold: D_{th}



persist all deletes within:



persists deletes timely

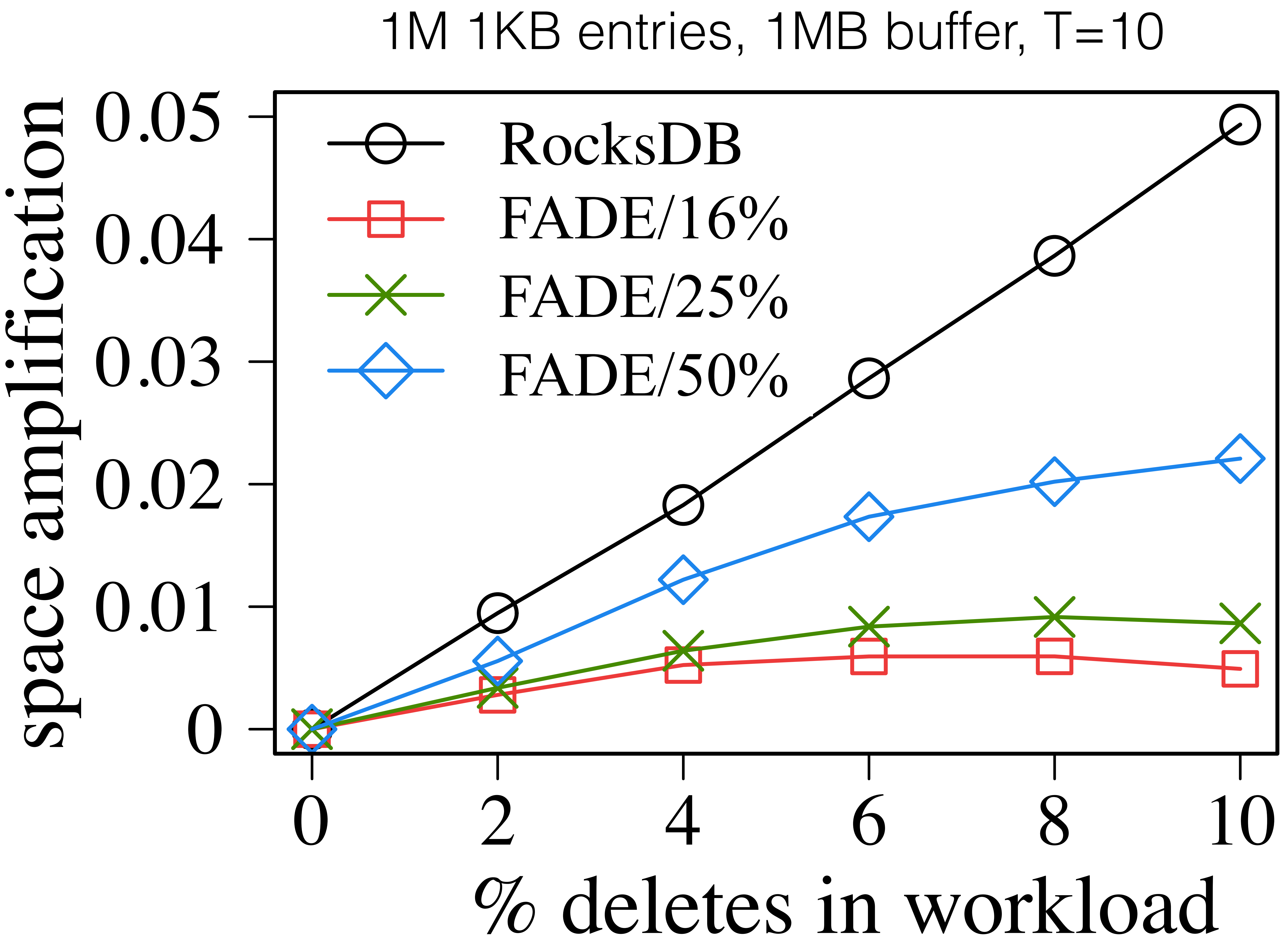
within threshold

reduced space amplification

2.1x - 9.8x

persists deletes timely

within threshold



improved read performance

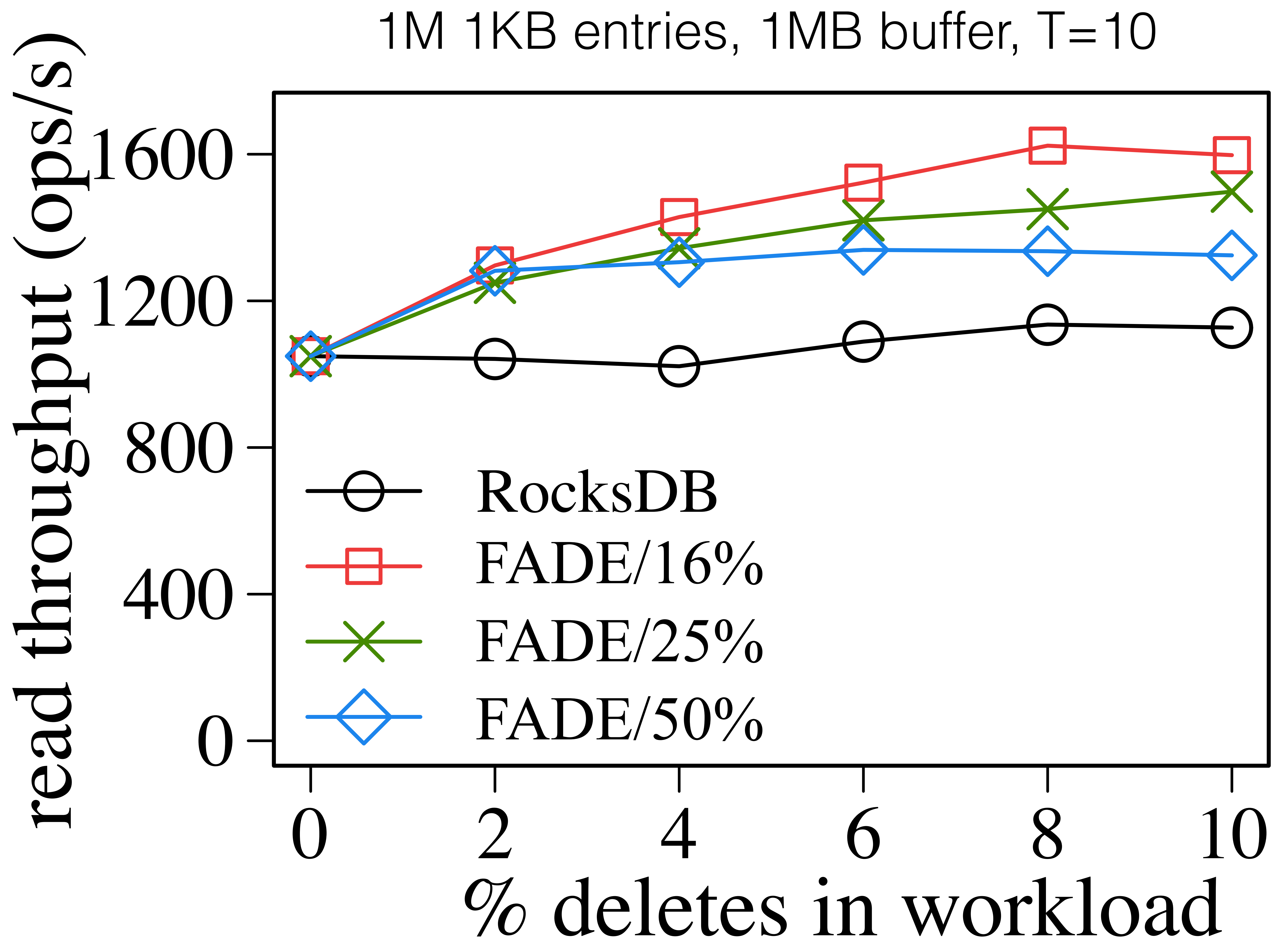
1.2x - 1.4x

reduced space amplification

2.1x - 9.8x

persists deletes timely

within threshold



higher write amplification

4% - 25%

improved read performance

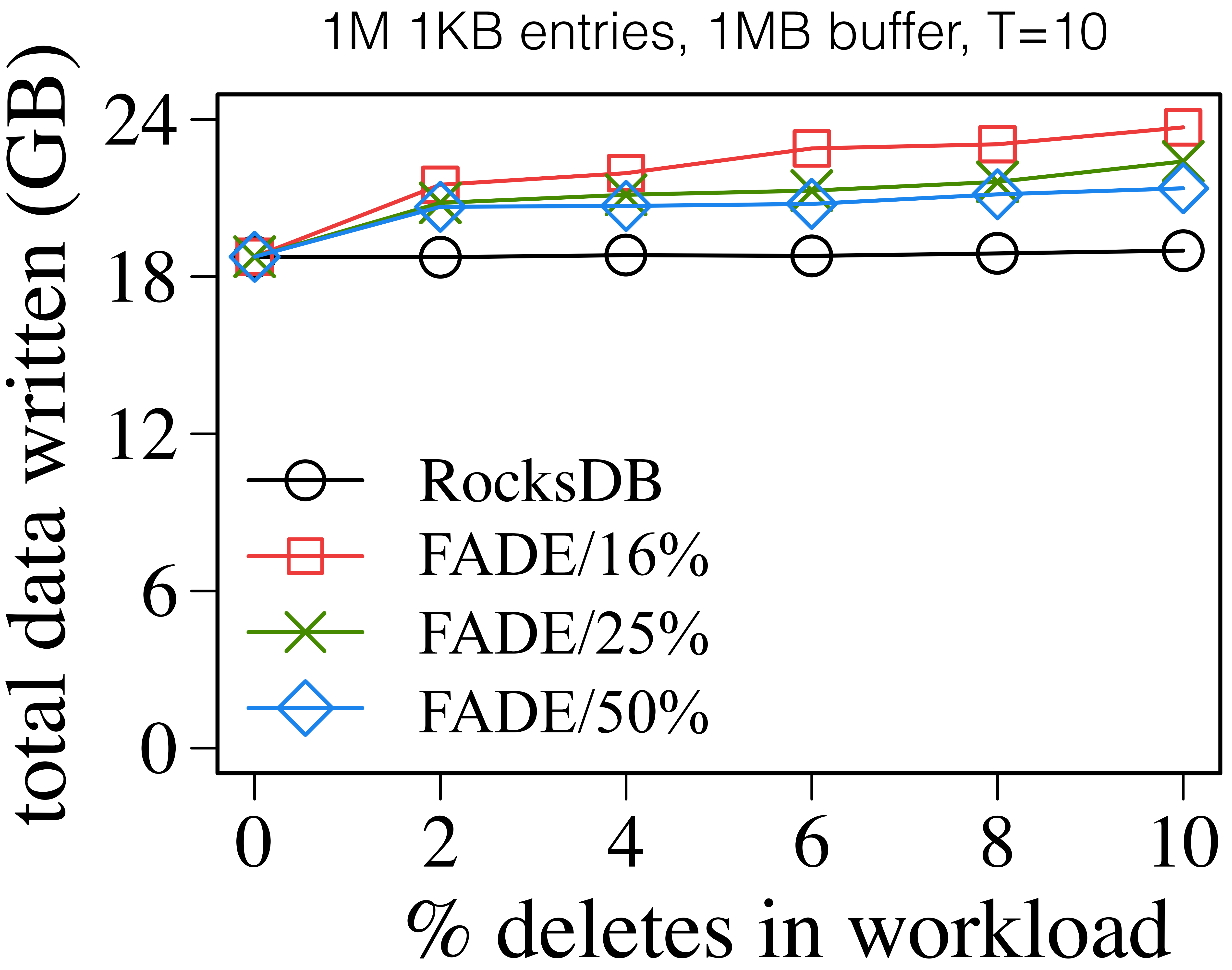
1.2x - 1.4x

reduced space amplification

2.1x - 9.8x

persists deletes timely

within threshold



higher write amplification

4% - 25%

improved read performance

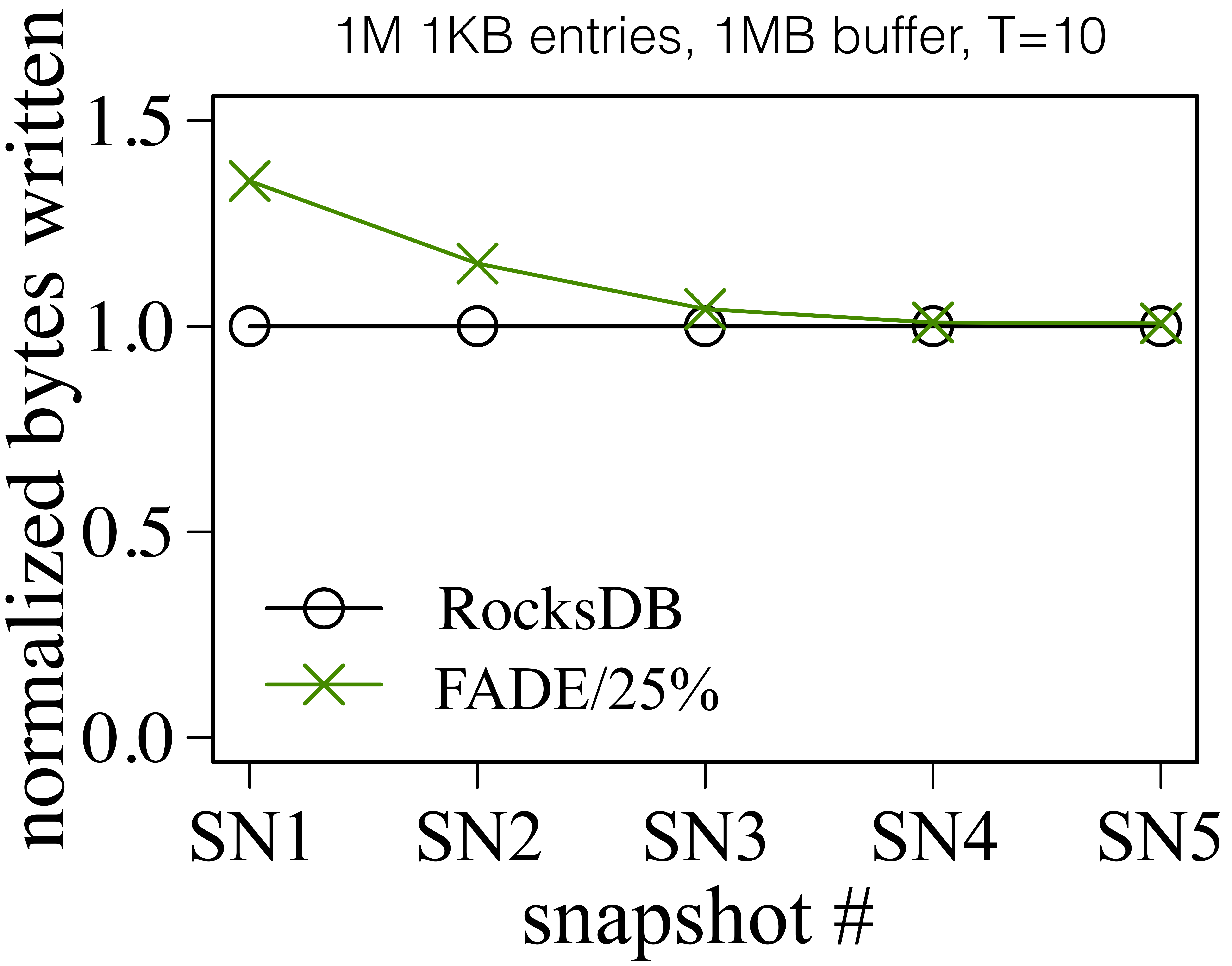
1.2x - 1.4x

reduced space amplification

2.1x - 9.8x

persists deletes timely

within threshold



higher write amplification

0.7%

improved read performance

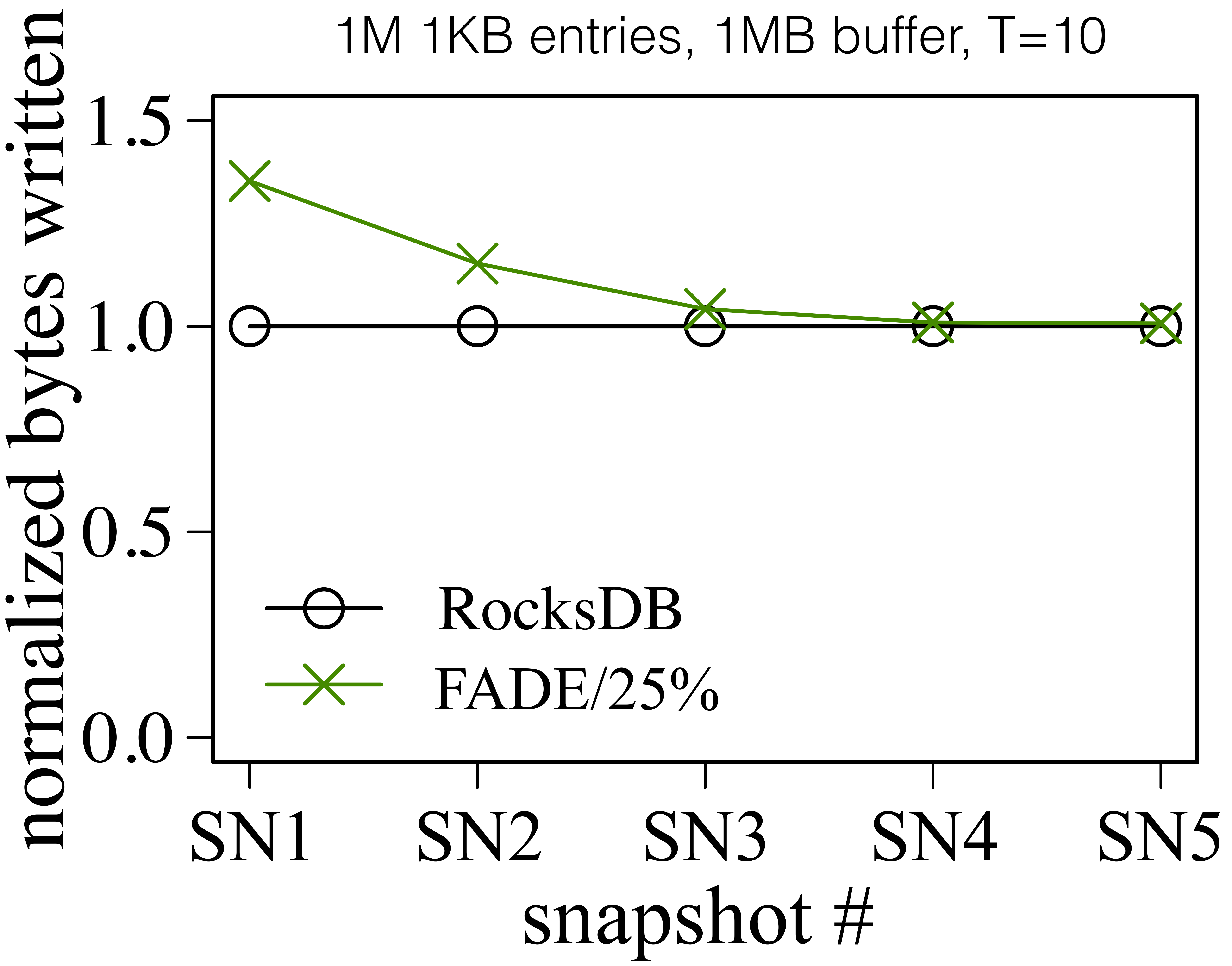
1.2x - 1.4x

reduced space amplification

2.1x - 9.8x

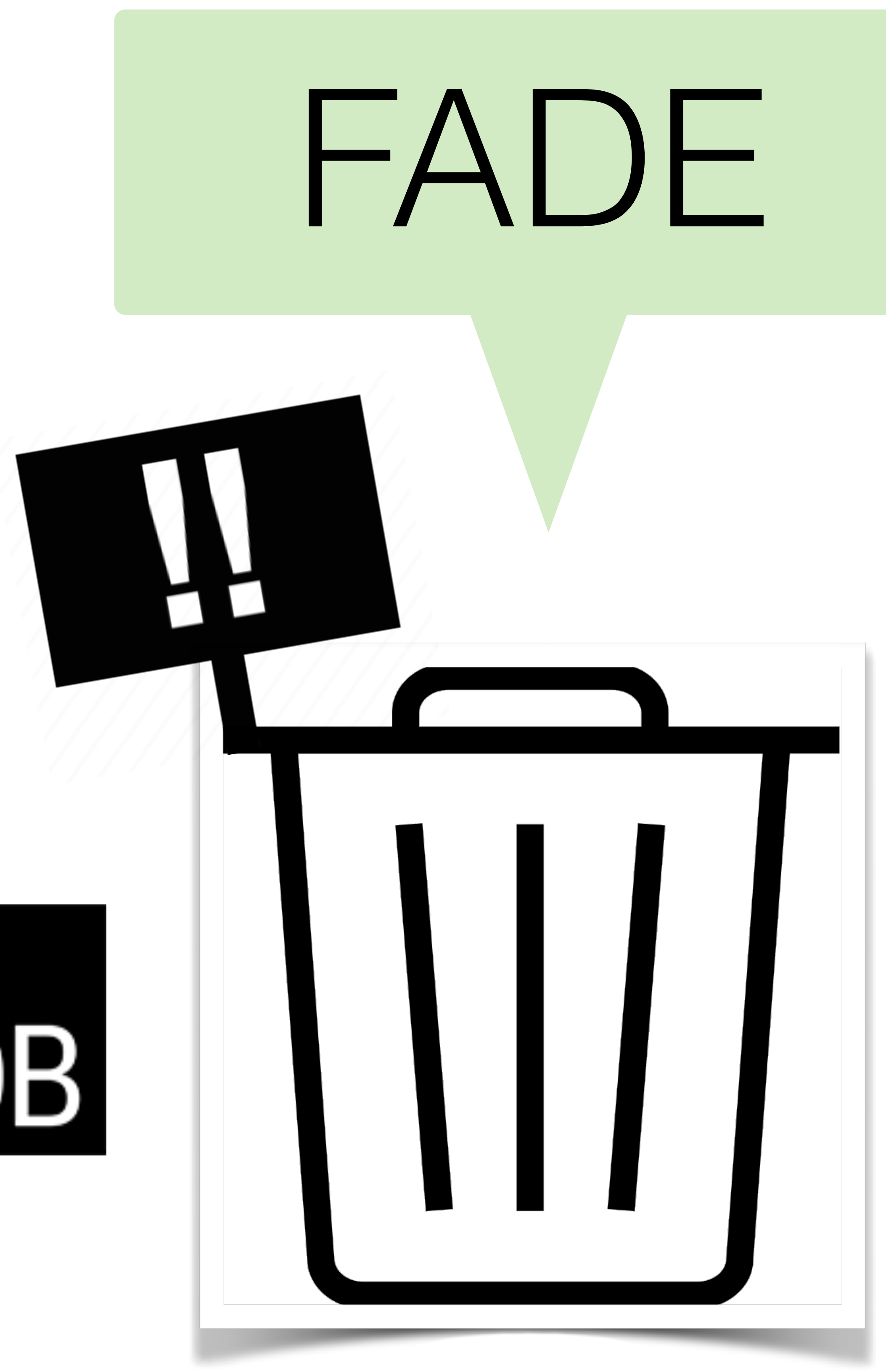
persists deletes timely

within threshold



EDBT 2022

DEBull 2022



RocksDB

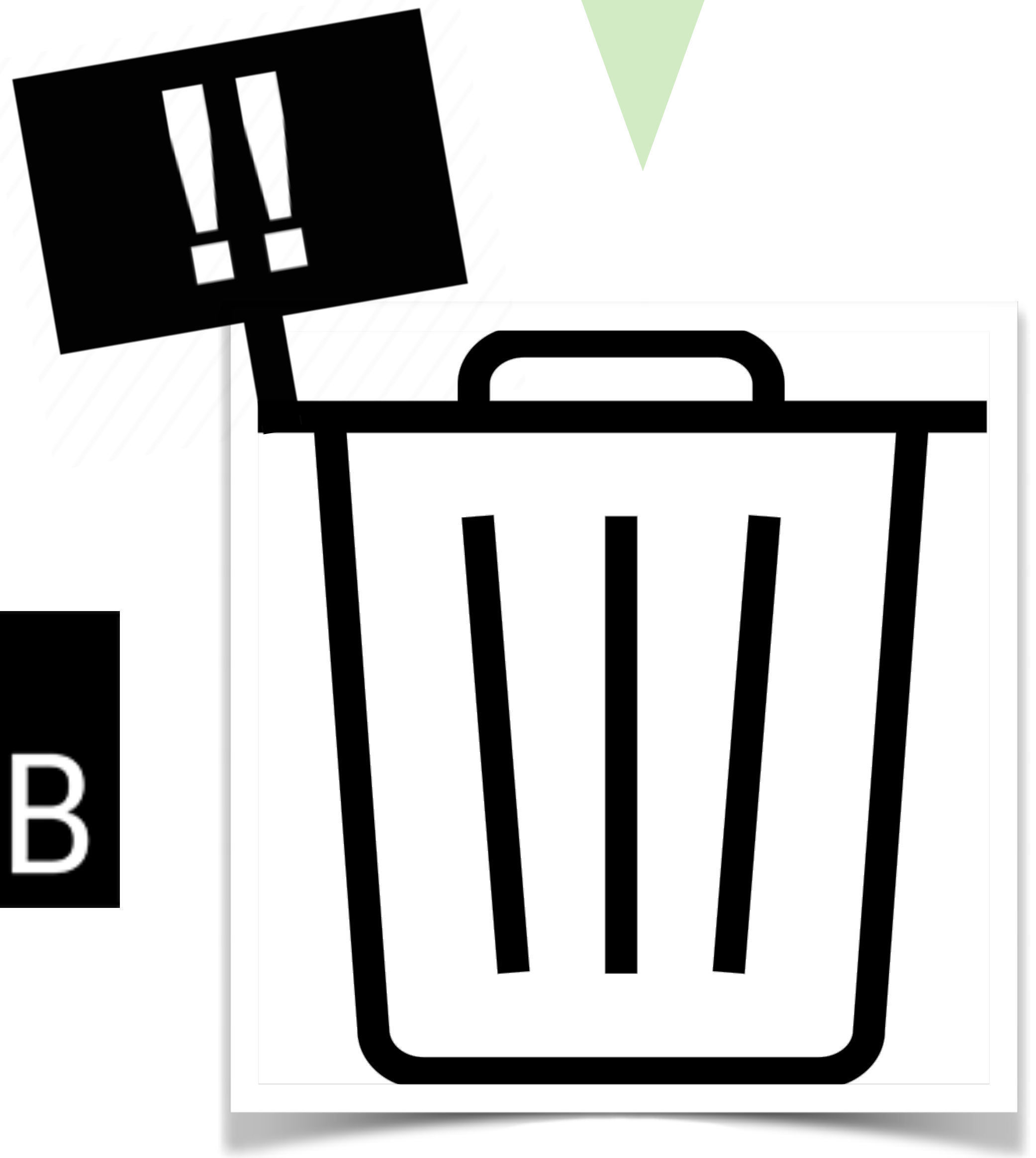
on-demand

persist all logical
deletes within D days

EDBT 2022

DEBull 2022

FADE



RocksDB

on-demand

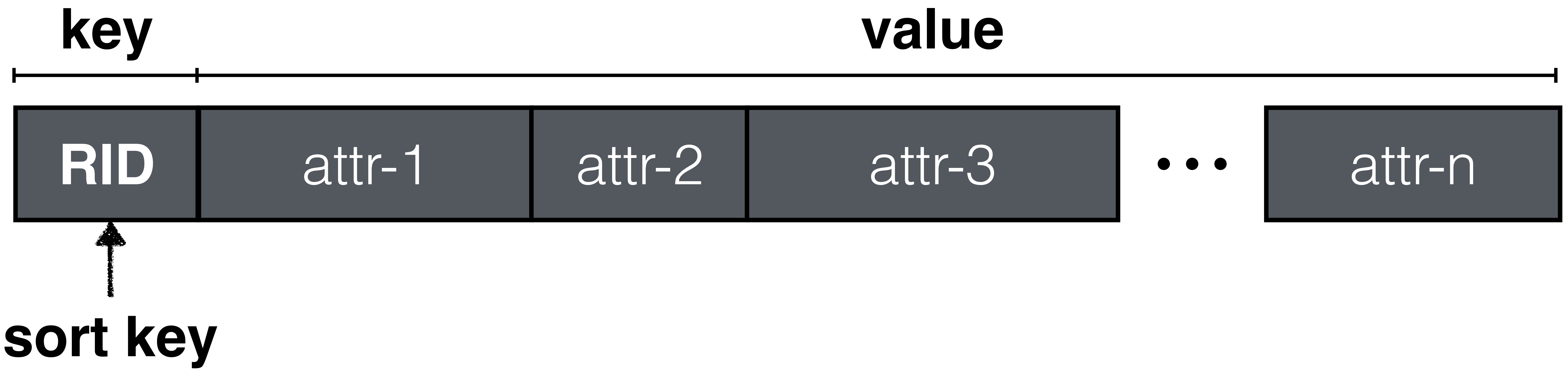
persist all logical **deletes within D days**

retention-based

delete **all data older than T days**

Realizing **Retention-Based Deletes**

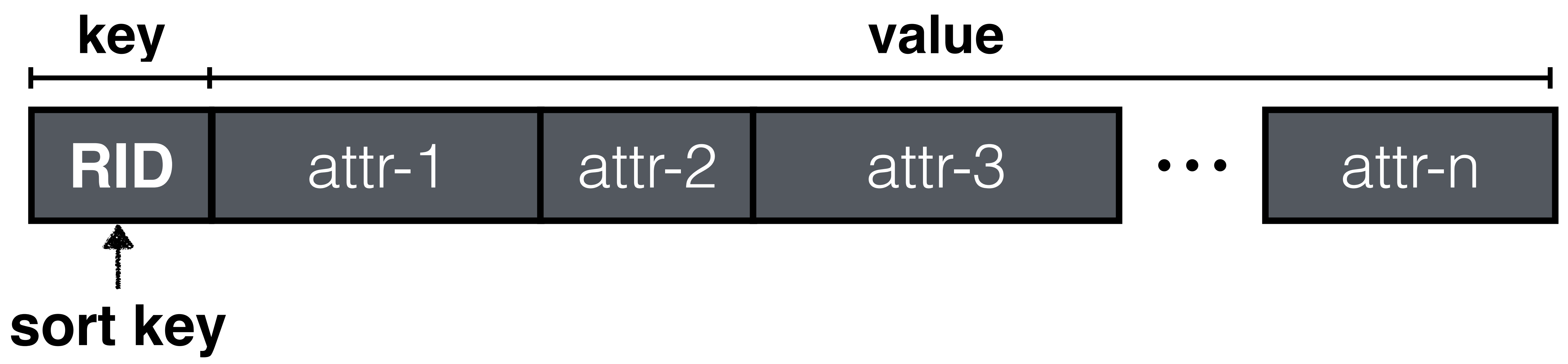
delete all entries older than: TS_x



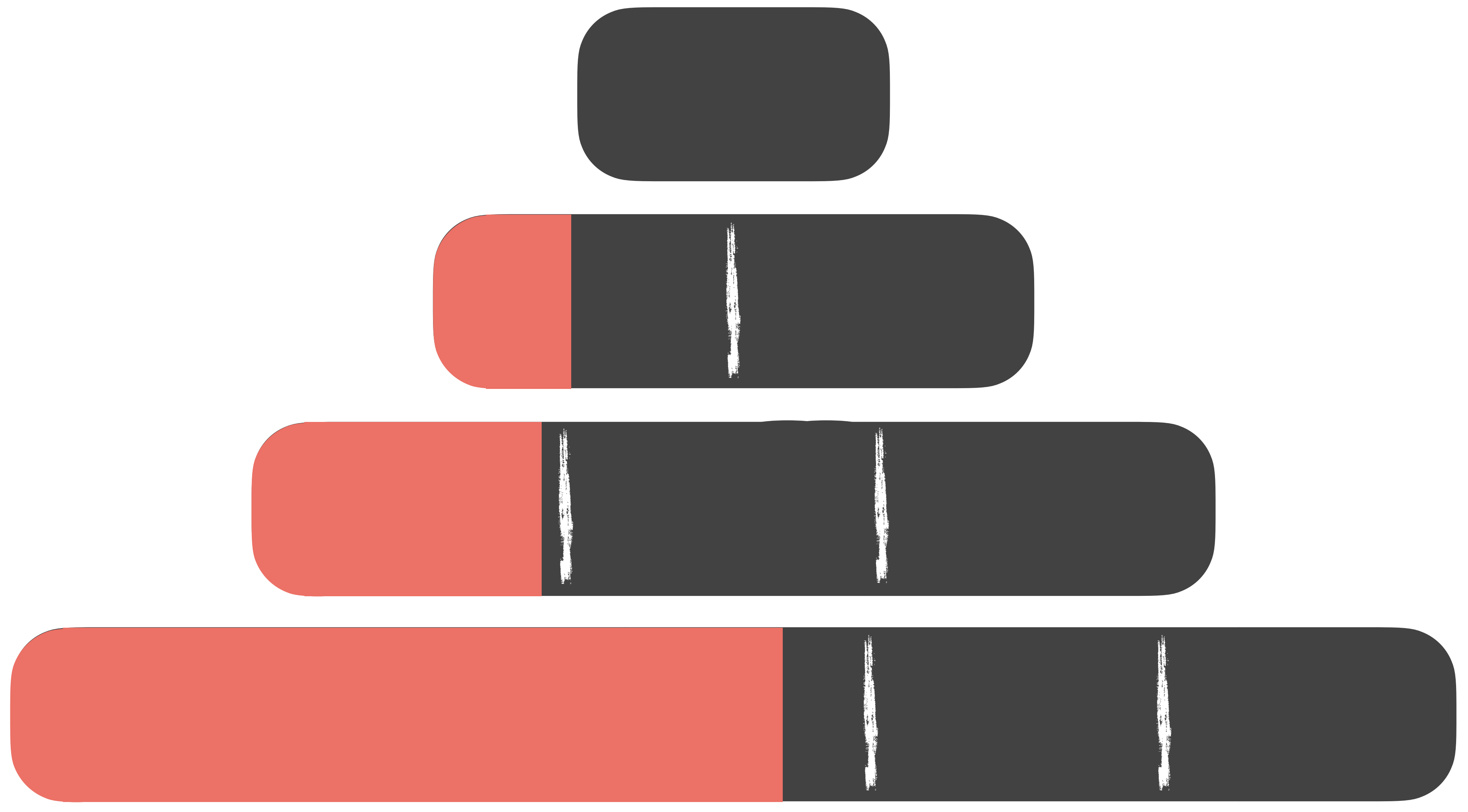
sort key = delete key

Realizing **Retention-Based Deletes**

delete all entries older than: TS_x

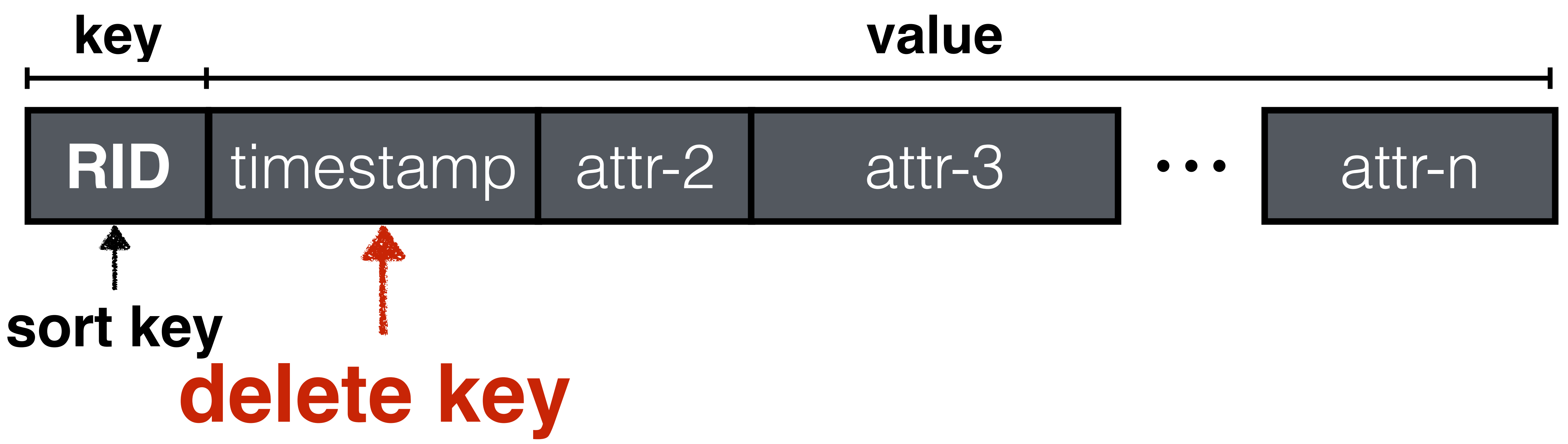


sort key = delete key

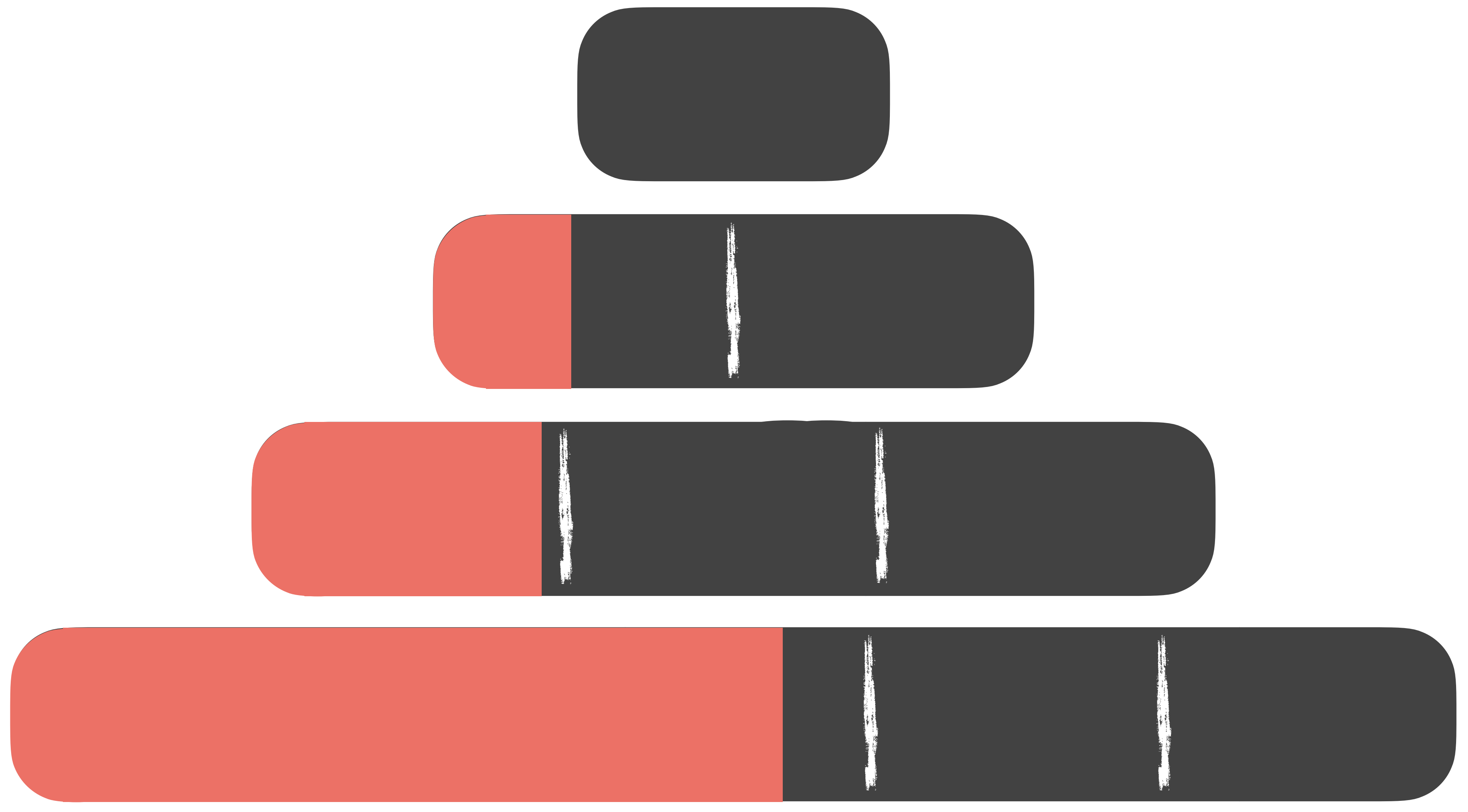


Realizing **Retention-Based Deletes**

delete all entries older than: TS_x

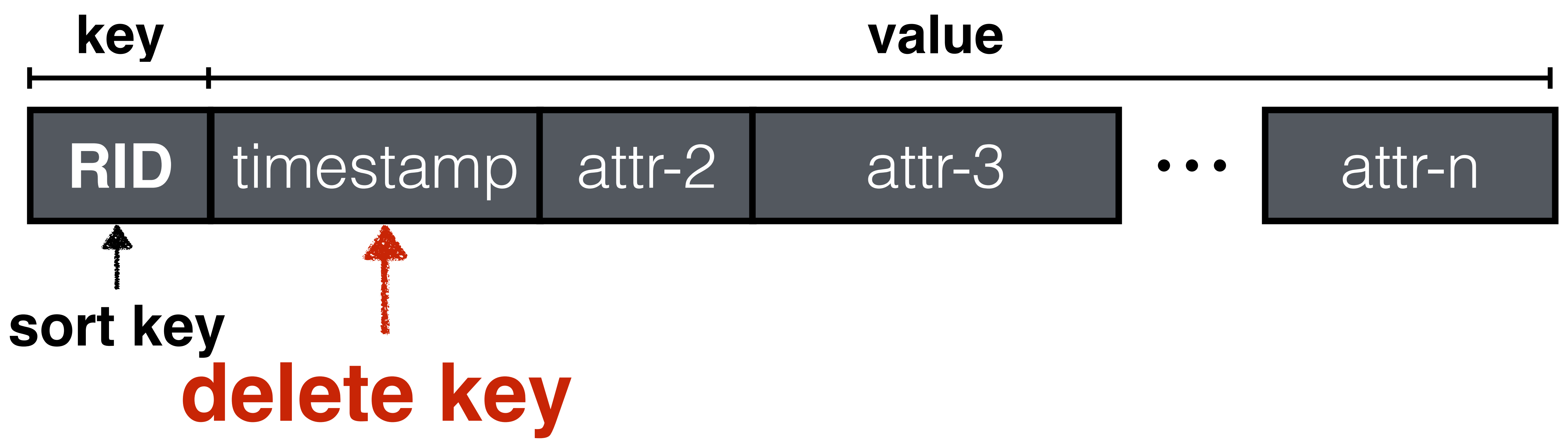


sort key ≠ delete key

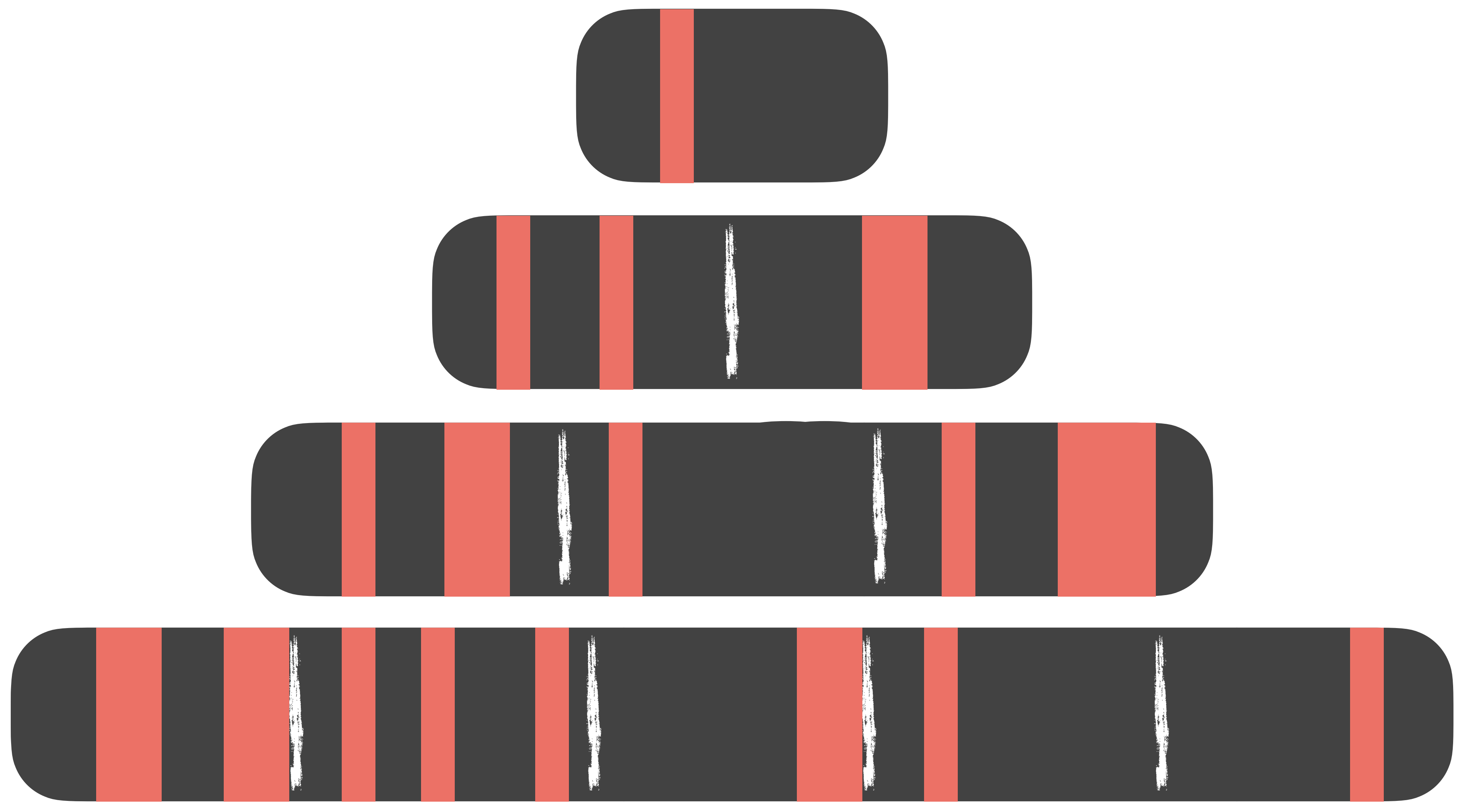


Realizing **Retention-Based Deletes**

delete all entries older than: TS_x



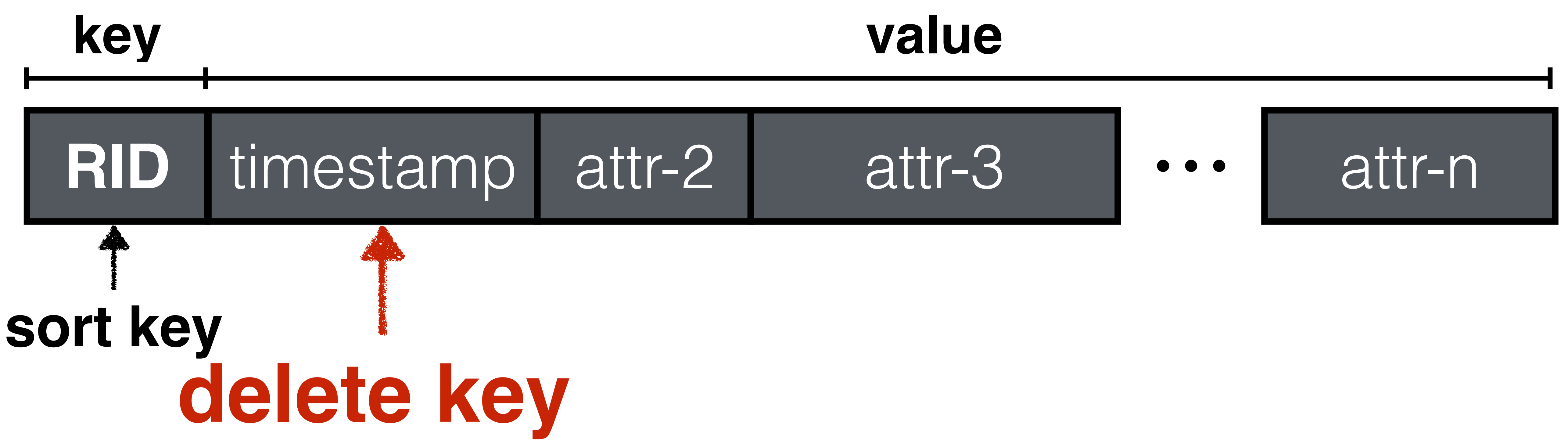
sort key ≠ delete key



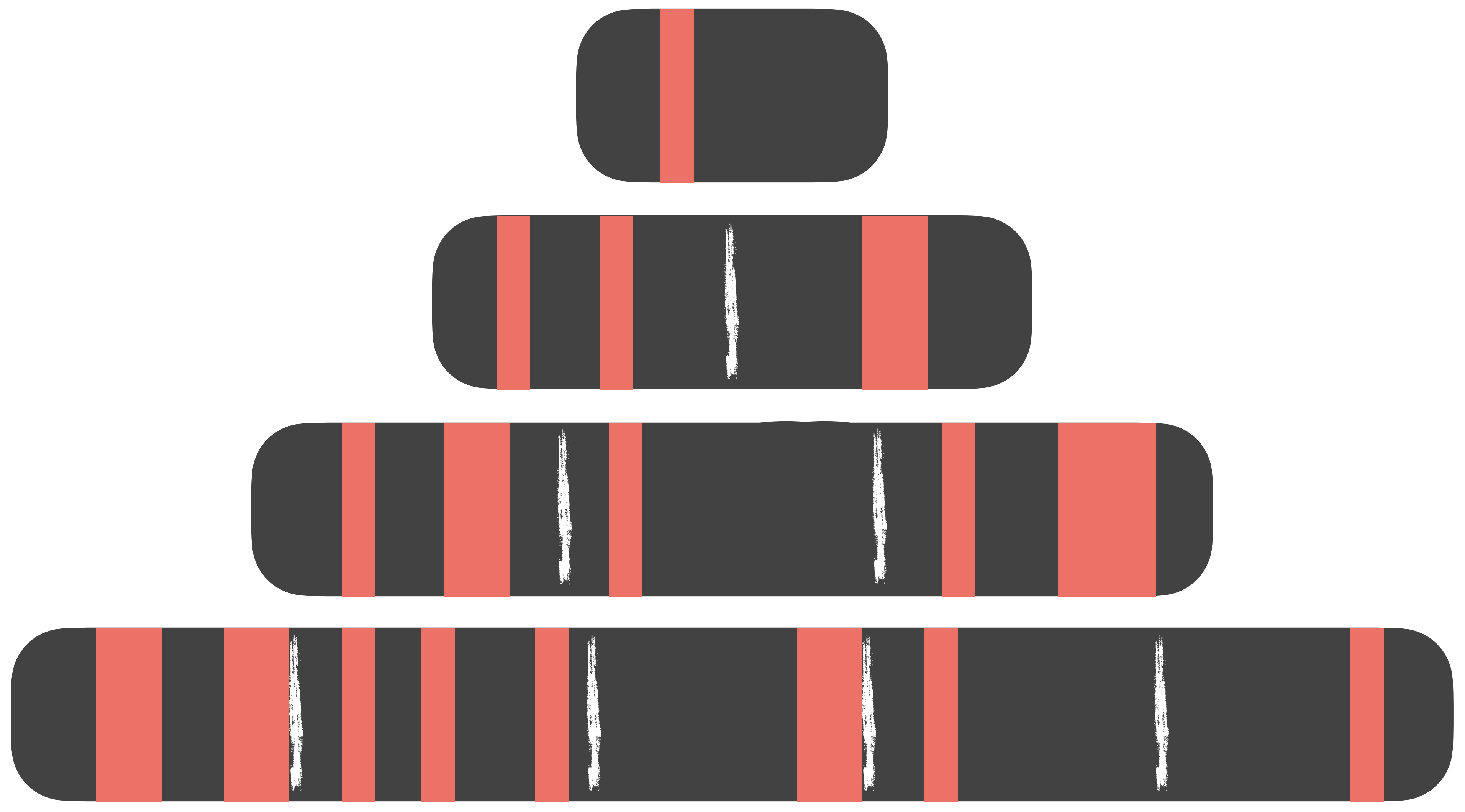
scattered occurrences

Realizing **Retention-Based Deletes**

delete all entries older than: TS_x



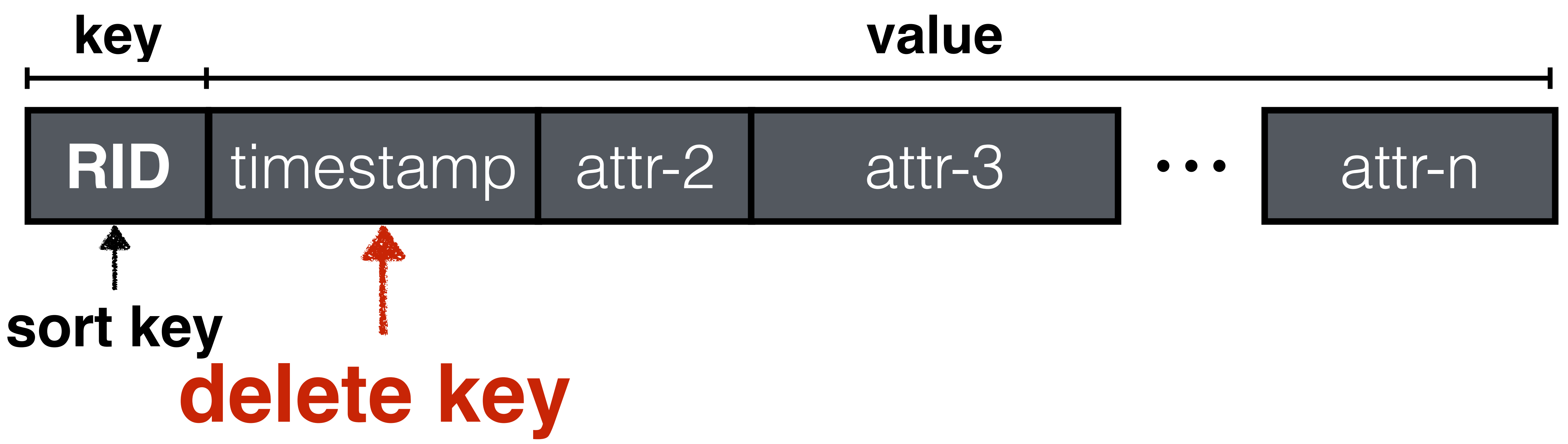
sort key ≠ delete key



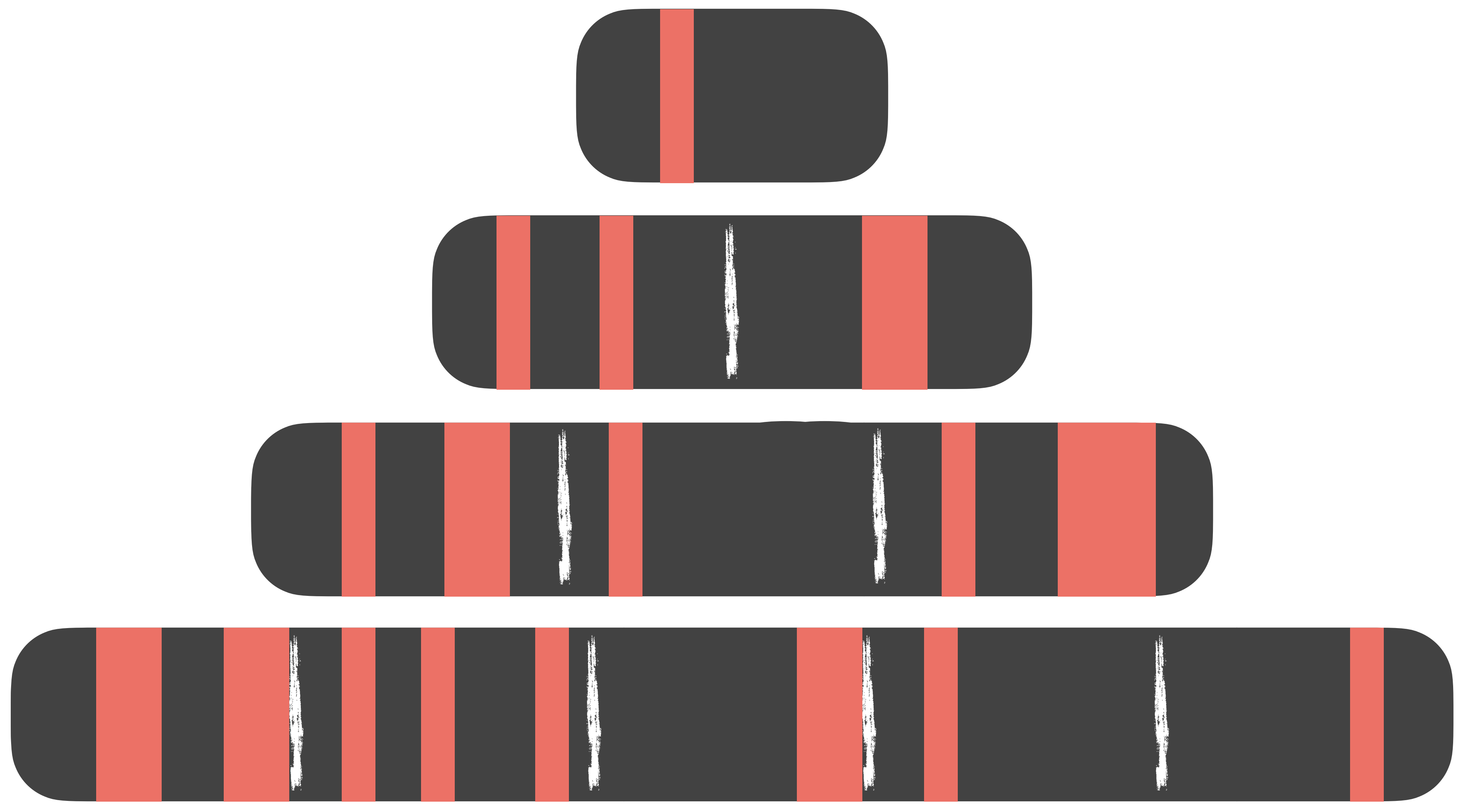
scattered occurrences

Realizing **Retention-Based Deletes**

delete all entries older than: TS_x



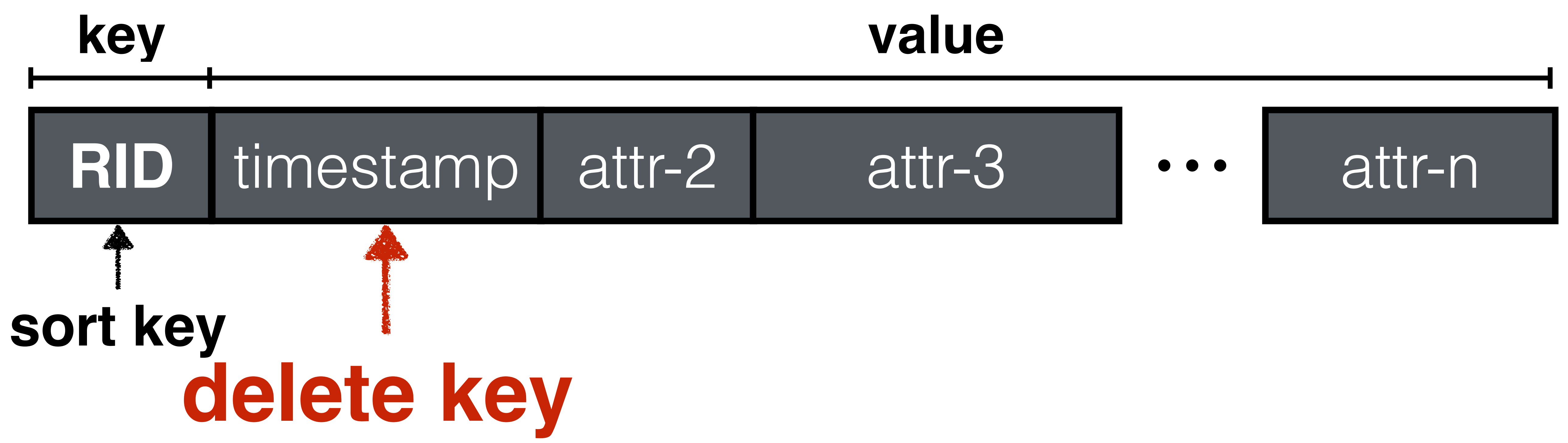
sort key ≠ delete key



scattered occurrences

Realizing **Retention-Based Deletes**

delete all entries older than: TS_x



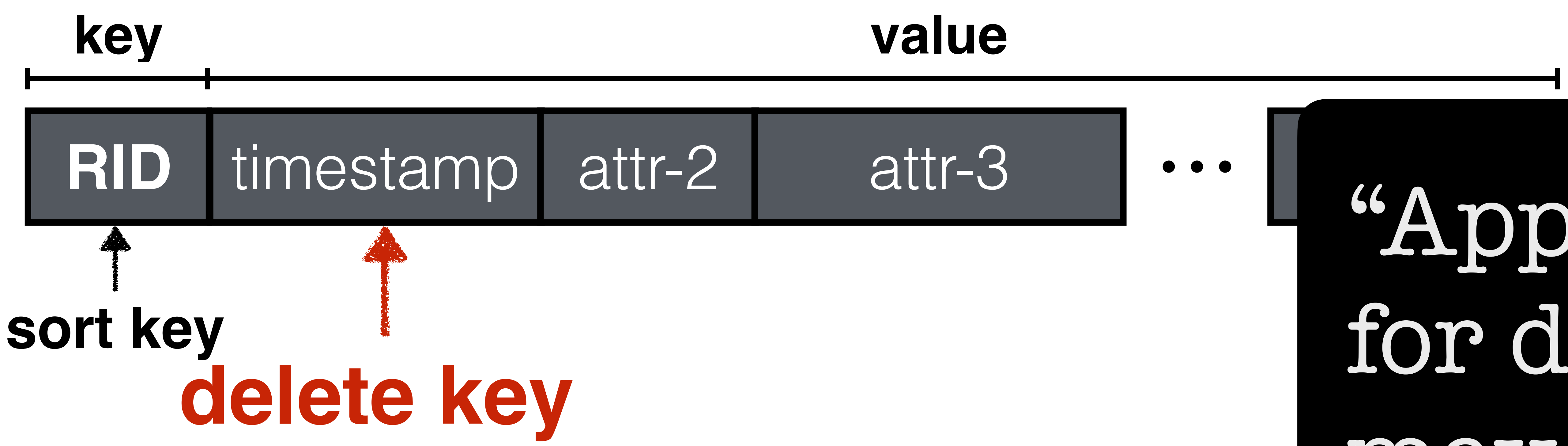
latency spikes

superfluous I/Os

sort key ≠ delete key

Realizing **Retention-Based Deletes**

delete all entries older than: TS_x



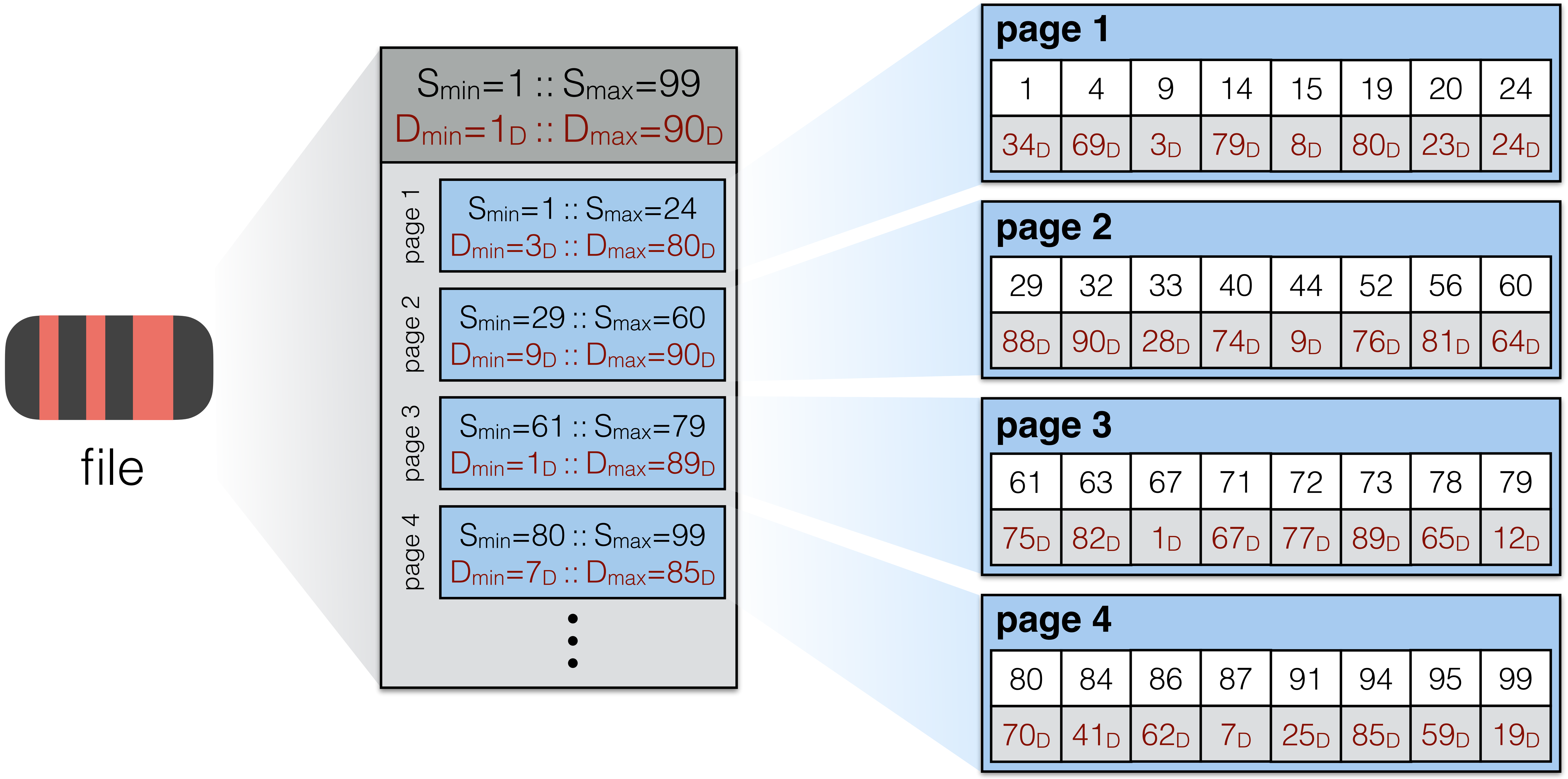
sort key ≠ delete ke

"Applications have requirements for deletes every day. E.g., they may keep data for 30 days, … effectively purging 1/30 of the database every day.

This induces performance pains!"

Realizing **Retention-Based Deletes**

delete all entries older than <= 65_D



$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$

page 1: $S_{min}=1$ :: $S_{max}=24$, $D_{min}=3_D$ :: $D_{max}=80_D$

page 2: $S_{min}=29$ :: $S_{max}=60$, $D_{min}=9_D$ :: $D_{max}=90_D$

page 3: $S_{min}=61$ :: $S_{max}=79$, $D_{min}=1_D$ :: $D_{max}=89_D$

page 4: $S_{min}=80$ :: $S_{max}=99$, $D_{min}=7_D$ :: $D_{max}=85_D$

page 1

1	4	9	14	15	19	20	24
34_D	69_D	3_D	79_D	8_D	80_D	23_D	24_D

page 2

29	32	33	40	44	52	56	60
88_D	90_D	28_D	74_D	9_D	76_D	81_D	64_D

page 3

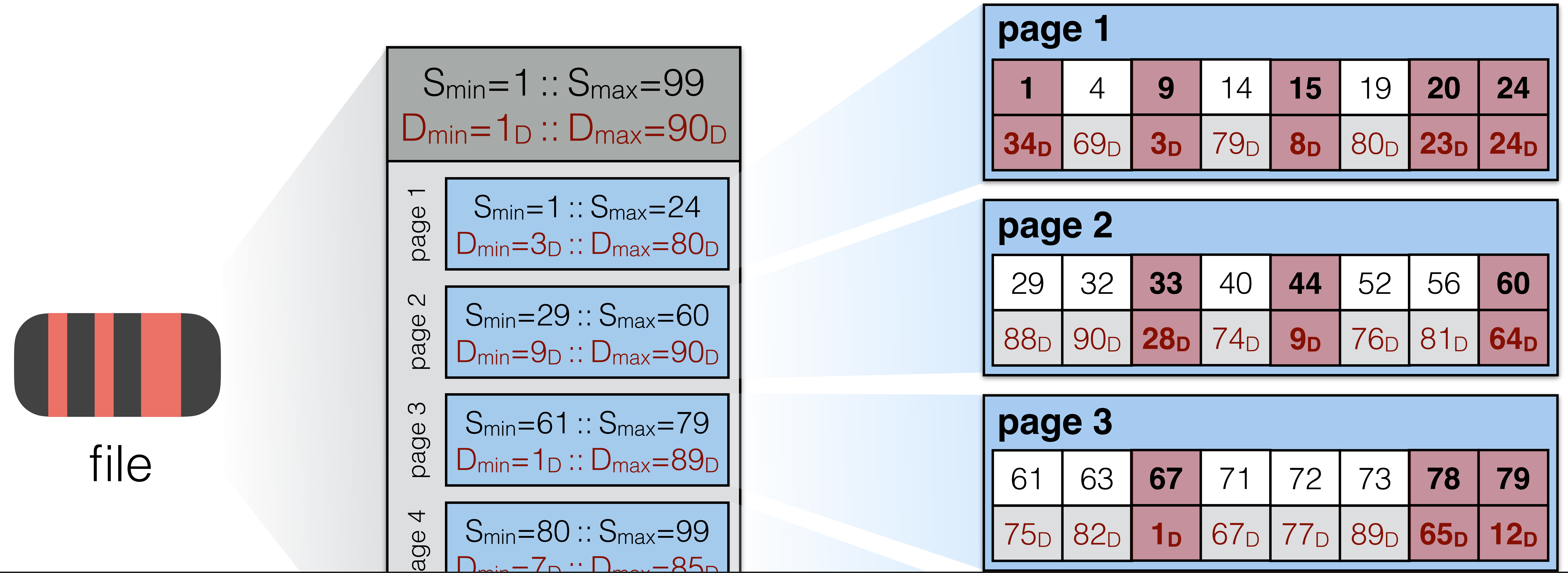
61	63	67	71	72	73	78	79
75_D	82_D	1_D	67_D	77_D	89_D	65_D	12_D

page 4

80	84	86	87	91	94	95	99
70_D	41_D	62_D	7_D	25_D	85_D	59_D	19_D

Realizing **Retention-Based Deletes**

delete all entries older than <= 65_D


file
$S_{min}=1 :: S_{max}=99$
$D_{min}=1_D :: D_{max}=90_D$
page 1
$S_{min}=1 :: S_{max}=24$
$D_{min}=3_D :: D_{max}=80_D$
page 2
$S_{min}=29 :: S_{max}=60$
$D_{min}=9_D :: D_{max}=90_D$
page 3
$S_{min}=61 :: S_{max}=79$
$D_{min}=1_D :: D_{max}=89_D$
age 4
$S_{min}=80 :: S_{max}=99$
$D_{min}=7_D :: D_{max}=85_D$

page 1

1	4	**9**	14	**15**	19	**20**	**24**
34_D	69_D	**3_D**	79_D	**8_D**	80_D	**23_D**	**24_D**

1 I/O

page 2

29	32	**33**	40	**44**	52	56	**60**
88_D	90_D	**28_D**	74_D	**9_D**	76_D	81_D	**64_D**

1 I/O

page 3

61	63	**67**	71	72	73	**78**	**79**
75_D	82_D	**1_D**	67_D	77_D	89_D	**65_D**	**12_D**

1 I/O

Intuition: Data Layout holds the key!

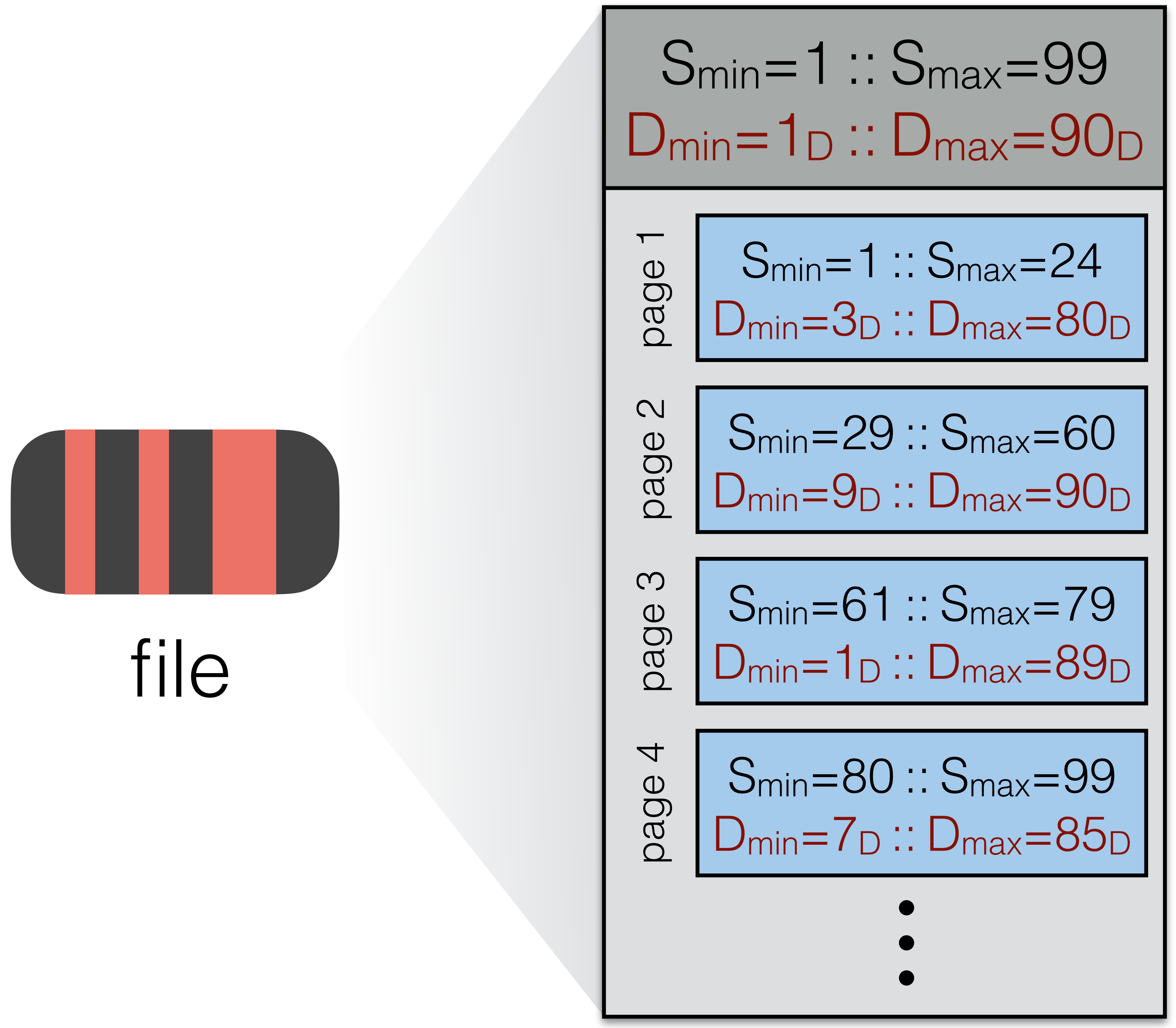
Realizing **Retention-Based Deletes**

KiWi

Key Weaving storage layout

Key Weaving storage layout

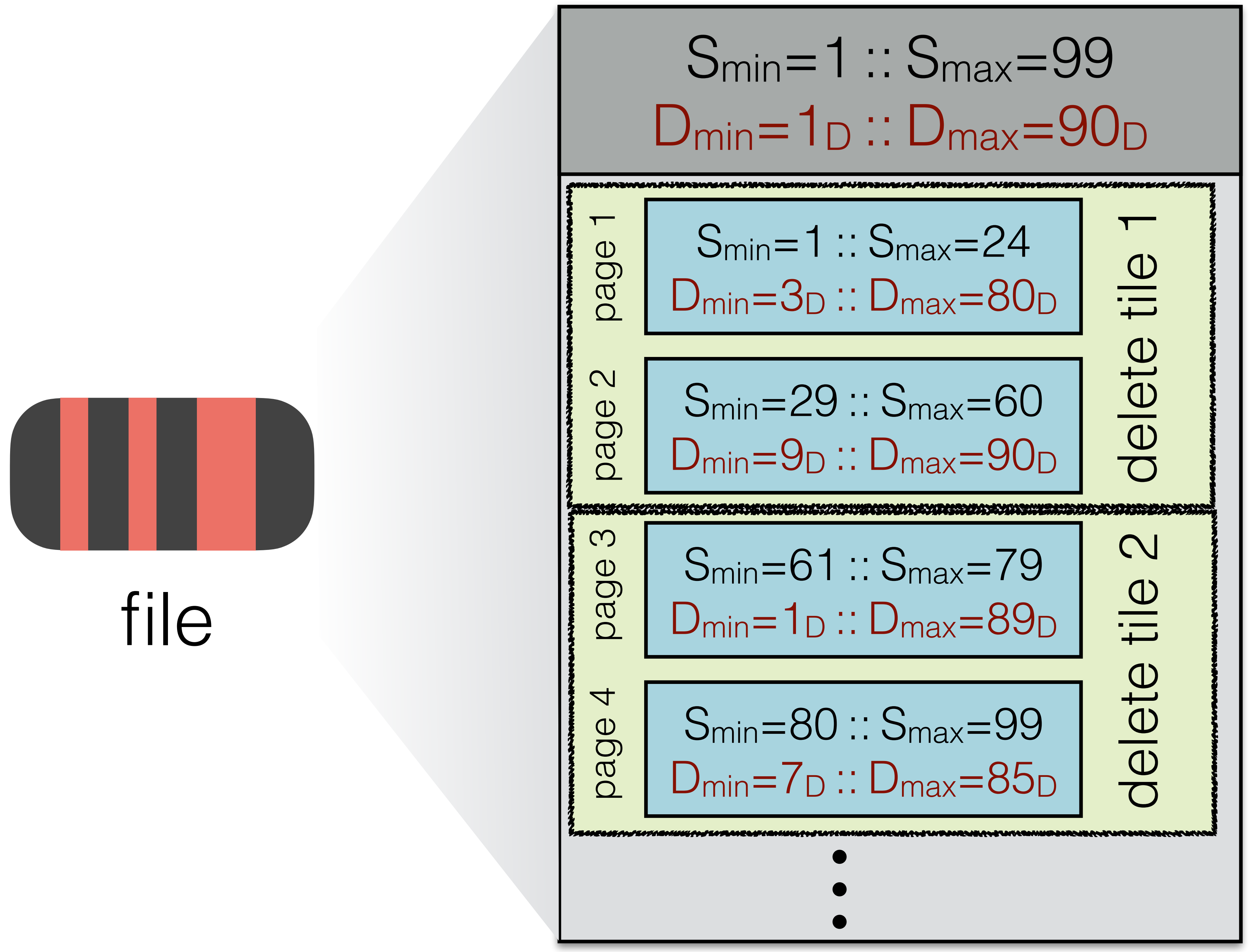
delete all entries older than <= 65_D



file
$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$
page 1
$S_{min}=1$ :: $S_{max}=24$
$D_{min}=3_D$ :: $D_{max}=80_D$
page 2
$S_{min}=29$ :: $S_{max}=60$
$D_{min}=9_D$ :: $D_{max}=90_D$
page 3
$S_{min}=61$ :: $S_{max}=79$
$D_{min}=1_D$ :: $D_{max}=89_D$
page 4
$S_{min}=80$ :: $S_{max}=99$
$D_{min}=7_D$ :: $D_{max}=85_D$

Key Weaving storage layout

delete all entries older than <= 65_D

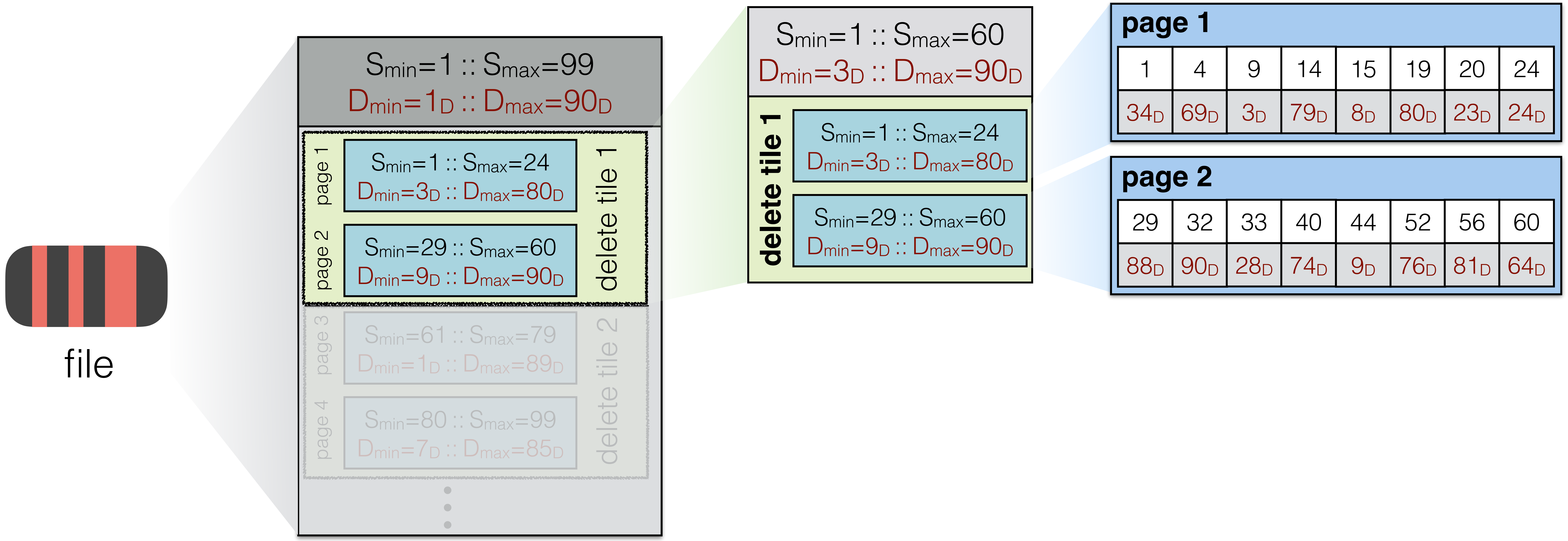


file
$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$
page 1
$S_{min}=1$ :: $S_{max}=24$
$D_{min}=3_D$ :: $D_{max}=80_D$
page 2
$S_{min}=29$ :: $S_{max}=60$
$D_{min}=9_D$ :: $D_{max}=90_D$
delete tile 1
page 3
$S_{min}=61$ :: $S_{max}=79$
$D_{min}=1_D$ :: $D_{max}=89_D$
page 4
$S_{min}=80$ :: $S_{max}=99$
$D_{min}=7_D$ :: $D_{max}=85_D$
delete tile 2

partitioned on S

Key Weaving storage layout

delete all entries older than <= 65_D



file
$S_{min}=1 :: S_{max}=99$
$D_{min}=1_D :: D_{max}=90_D$
page 1
$S_{min}=1 :: S_{max}=24$
$D_{min}=3_D :: D_{max}=80_D$
page 2
$S_{min}=29 :: S_{max}=60$
$D_{min}=9_D :: D_{max}=90_D$
delete tile 1
page 3
$S_{min}=61 :: S_{max}=79$
$D_{min}=1_D :: D_{max}=89_D$
page 4
$S_{min}=80 :: S_{max}=99$
$D_{min}=7_D :: D_{max}=85_D$
delete tile 2
$S_{min}=1 :: S_{max}=60$
$D_{min}=3_D :: D_{max}=90_D$
delete tile 1
$S_{min}=1 :: S_{max}=24$
$D_{min}=3_D :: D_{max}=80_D$
$S_{min}=29 :: S_{max}=60$
$D_{min}=9_D :: D_{max}=90_D$

page 1

1	4	9	14	15	19	20	24
34_D	69_D	3_D	79_D	8_D	80_D	23_D	24_D

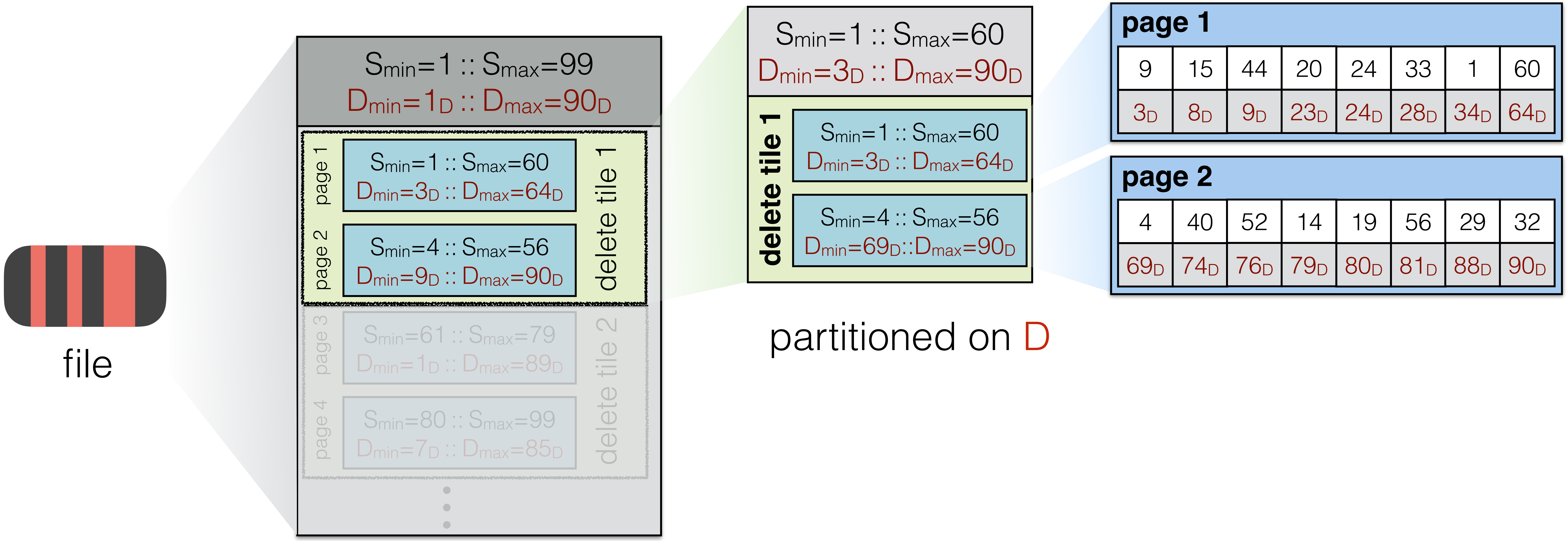
page 2

29	32	33	40	44	52	56	60
88_D	90_D	28_D	74_D	9_D	76_D	81_D	64_D

partitioned on S

Key Weaving storage layout

delete all entries older than <= 65$_D$


file
S_{min}=1 :: S_{max}=99
D_{min}=1$_D$ :: D_{max}=90$_D$
page 1
S_{min}=1 :: S_{max}=60
D_{min}=3$_D$ :: D_{max}=64$_D$
page 2
S_{min}=4 :: S_{max}=56
D_{min}=9$_D$ :: D_{max}=90$_D$
delete tile 1
page 3
S_{min}=61 :: S_{max}=79
D_{min}=1$_D$ :: D_{max}=89$_D$
page 4
S_{min}=80 :: S_{max}=99
D_{min}=7$_D$ :: D_{max}=85$_D$
delete tile 2
S_{min}=1 :: S_{max}=60
D_{min}=3$_D$ :: D_{max}=90$_D$
delete tile 1
S_{min}=1 :: S_{max}=60
D_{min}=3$_D$ :: D_{max}=64$_D$
S_{min}=4 :: S_{max}=56
D_{min}=69$_D$::D_{max}=90$_D$

page 1

9	15	44	20	24	33	1	60
3$_D$	8$_D$	9$_D$	23$_D$	24$_D$	28$_D$	34$_D$	64$_D$

page 2

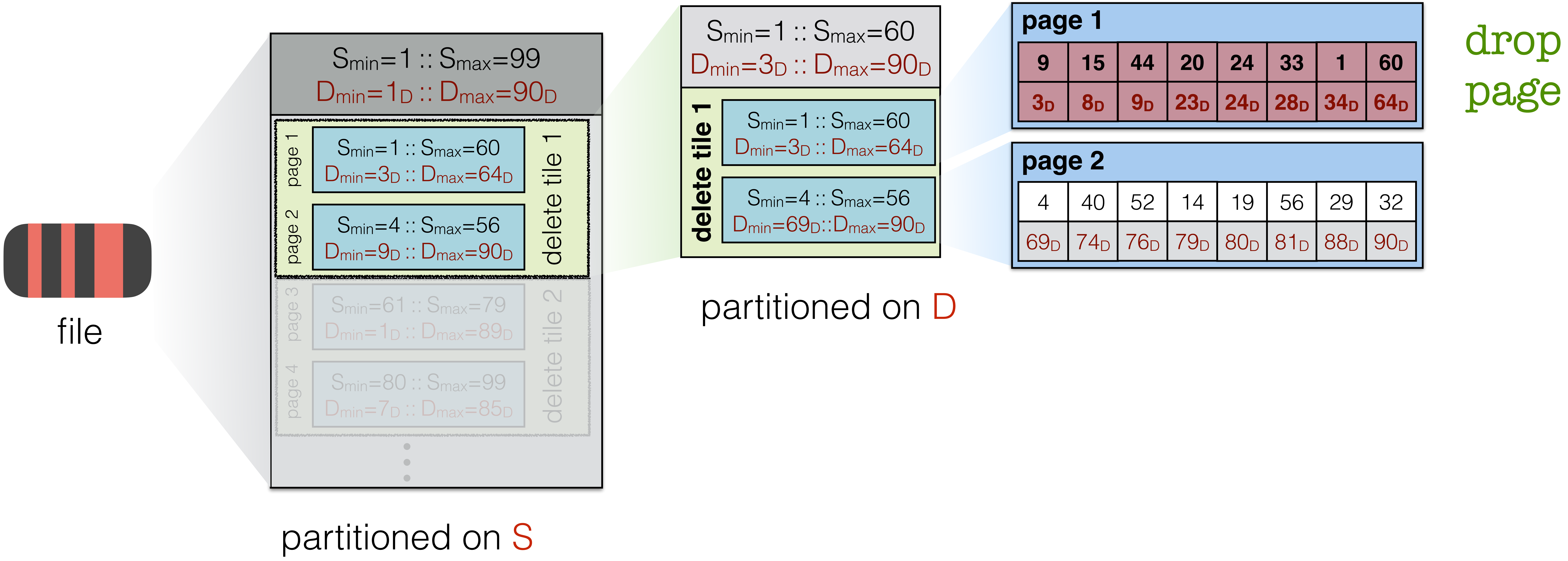
4	40	52	14	19	56	29	32
69$_D$	74$_D$	76$_D$	79$_D$	80$_D$	81$_D$	88$_D$	90$_D$

partitioned on D

partitioned on S

Key Weaving storage layout

delete all entries older than <= 65_D


file
$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$
page 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
page 2
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=9_D$ :: $D_{max}=90_D$
delete tile 1
page 3
$S_{min}=61$ :: $S_{max}=79$
$D_{min}=1_D$ :: $D_{max}=89_D$
page 4
$S_{min}=80$ :: $S_{max}=99$
$D_{min}=7_D$ :: $D_{max}=85_D$
delete tile 2
partitioned on S
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=90_D$
delete tile 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=69_D$::$D_{max}=90_D$
partitioned on D
page 1
page 2
drop page

page 1

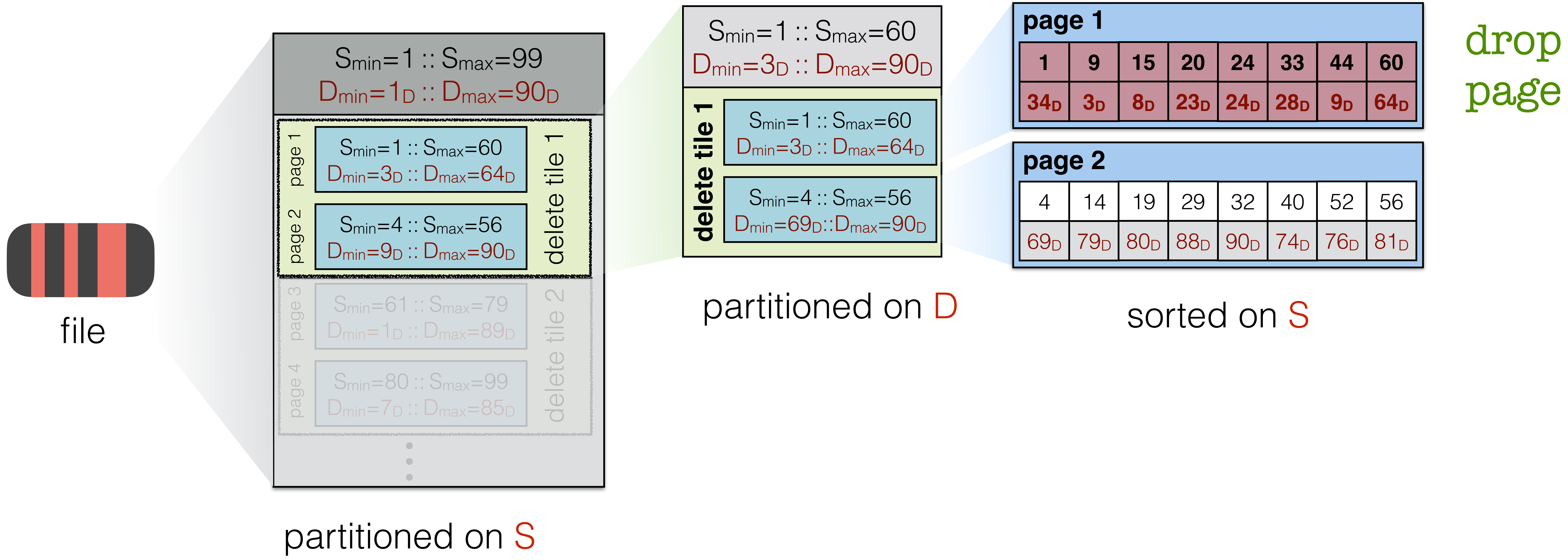
9	15	44	20	24	33	1	60
3_D	8_D	9_D	23_D	24_D	28_D	34_D	64_D

page 2

4	40	52	14	19	56	29	32
69_D	74_D	76_D	79_D	80_D	81_D	88_D	90_D

Key Weaving storage layout

delete all entries older than <= 65_D


file
$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$
page 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
page 2
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=9_D$ :: $D_{max}=90_D$
delete tile 1
page 3
$S_{min}=61$ :: $S_{max}=79$
$D_{min}=1_D$ :: $D_{max}=89_D$
page 4
$S_{min}=80$ :: $S_{max}=99$
$D_{min}=7_D$ :: $D_{max}=85_D$
delete tile 2
partitioned on S
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=90_D$
delete tile 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=69_D$::$D_{max}=90_D$
partitioned on D
page 1
page 2
sorted on S
drop page

page 1

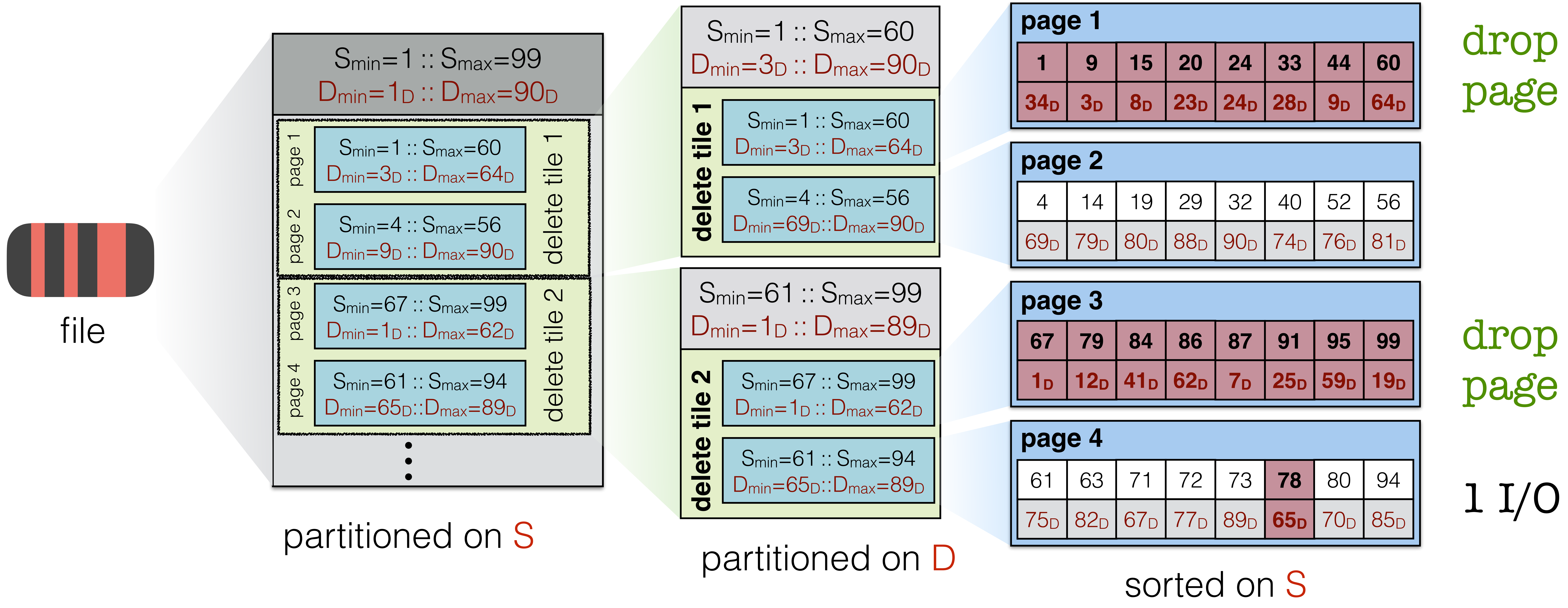
1	9	15	20	24	33	44	60
34_D	3_D	8_D	23_D	24_D	28_D	9_D	64_D

page 2

4	14	19	29	32	40	52	56
69_D	79_D	80_D	88_D	90_D	74_D	76_D	81_D

Key Weaving storage layout

delete all entries older than <= 65_D



file
$S_{min}=1$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=90_D$
page 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
page 2
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=9_D$ :: $D_{max}=90_D$
delete tile 1
page 3
$S_{min}=67$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=62_D$
page 4
$S_{min}=61$ :: $S_{max}=94$
$D_{min}=65_D$::$D_{max}=89_D$
delete tile 2
partitioned on S
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=90_D$
delete tile 1
$S_{min}=1$ :: $S_{max}=60$
$D_{min}=3_D$ :: $D_{max}=64_D$
$S_{min}=4$ :: $S_{max}=56$
$D_{min}=69_D$::$D_{max}=90_D$
$S_{min}=61$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=89_D$
delete tile 2
$S_{min}=67$ :: $S_{max}=99$
$D_{min}=1_D$ :: $D_{max}=62_D$
$S_{min}=61$ :: $S_{max}=94$
$D_{min}=65_D$::$D_{max}=89_D$
partitioned on D
drop page
drop page
1 I/O
sorted on S

page 1

1	9	15	20	24	33	44	60
34_D	3_D	8_D	23_D	24_D	28_D	9_D	64_D

page 2

4	14	19	29	32	40	52	56
69_D	79_D	80_D	88_D	90_D	74_D	76_D	81_D

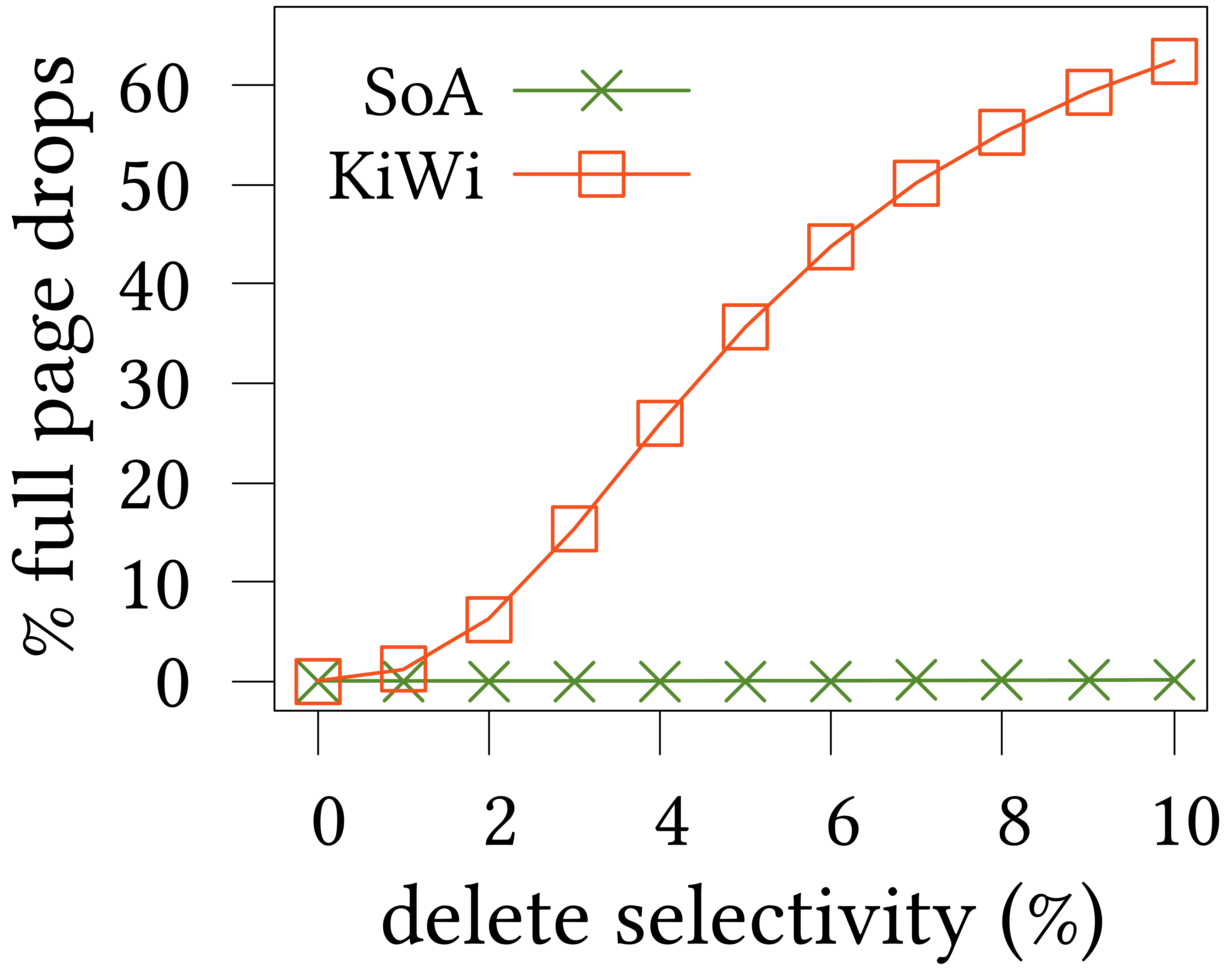
page 3

67	79	84	86	87	91	95	99
1_D	12_D	41_D	62_D	7_D	25_D	59_D	19_D

page 4

61	63	71	72	73	**78**	80	94
75_D	82_D	67_D	77_D	89_D	**65_D**	70_D	85_D

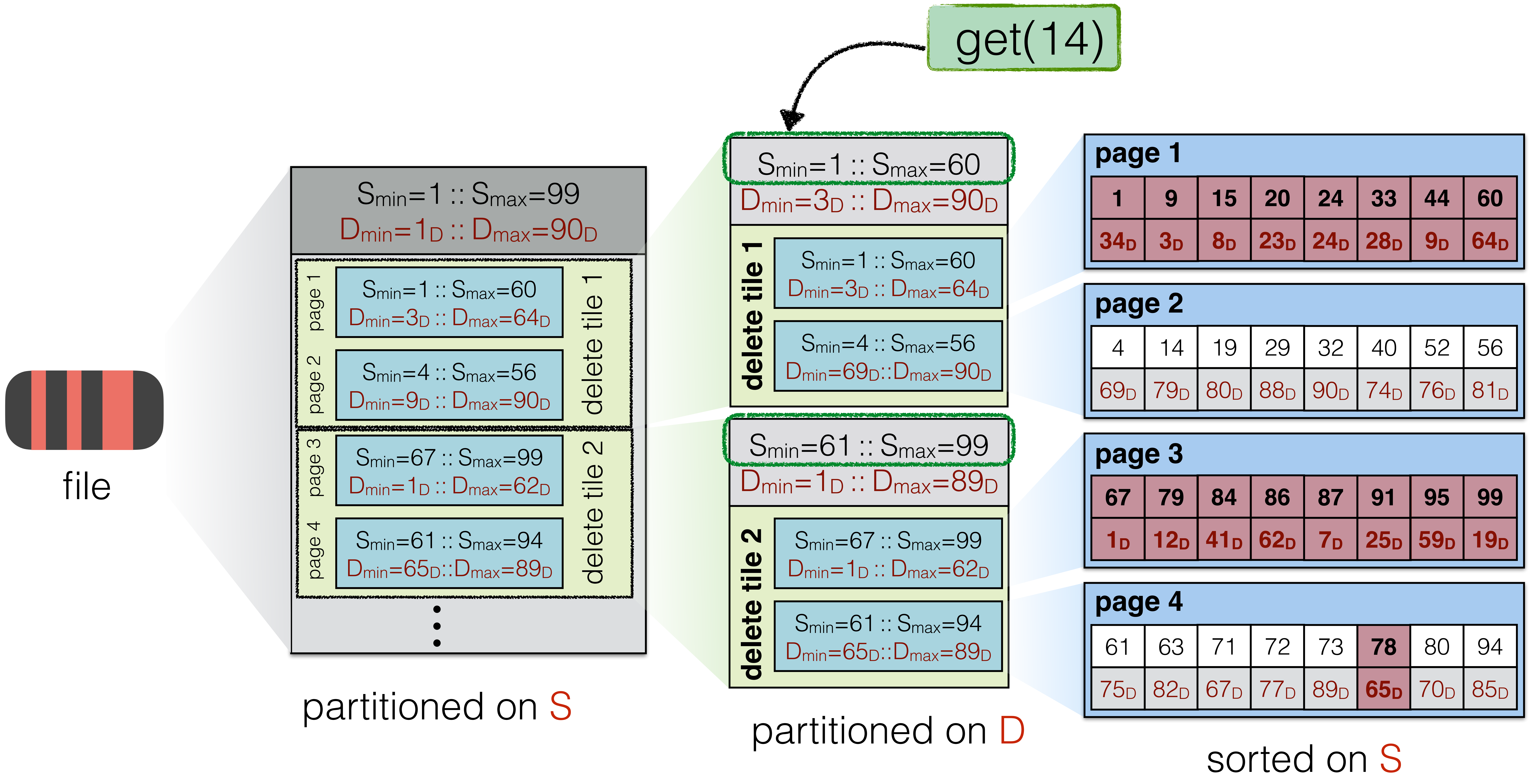
5M entries, buffer = file = 256 pages, T=10



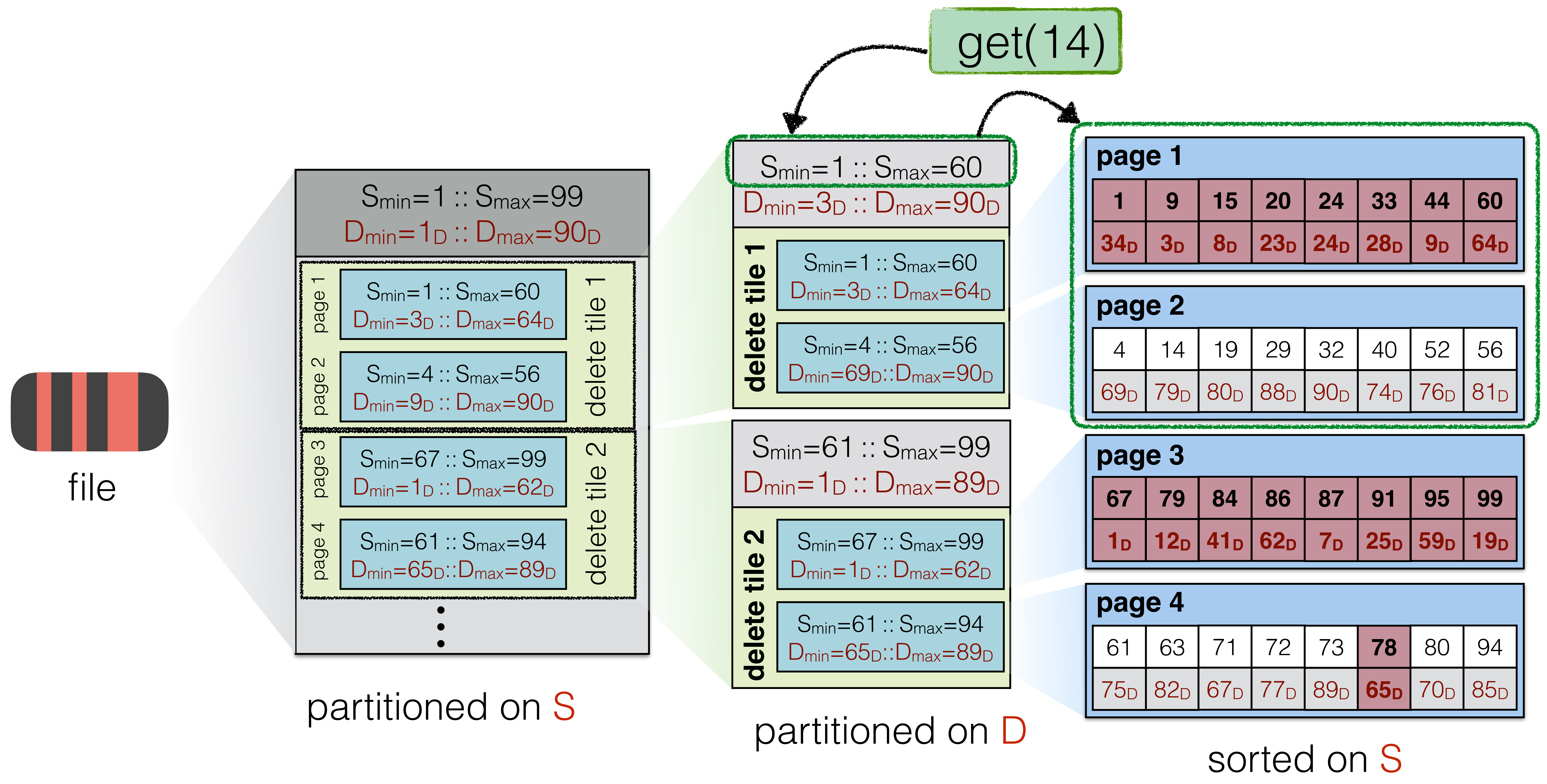
superior delete performance

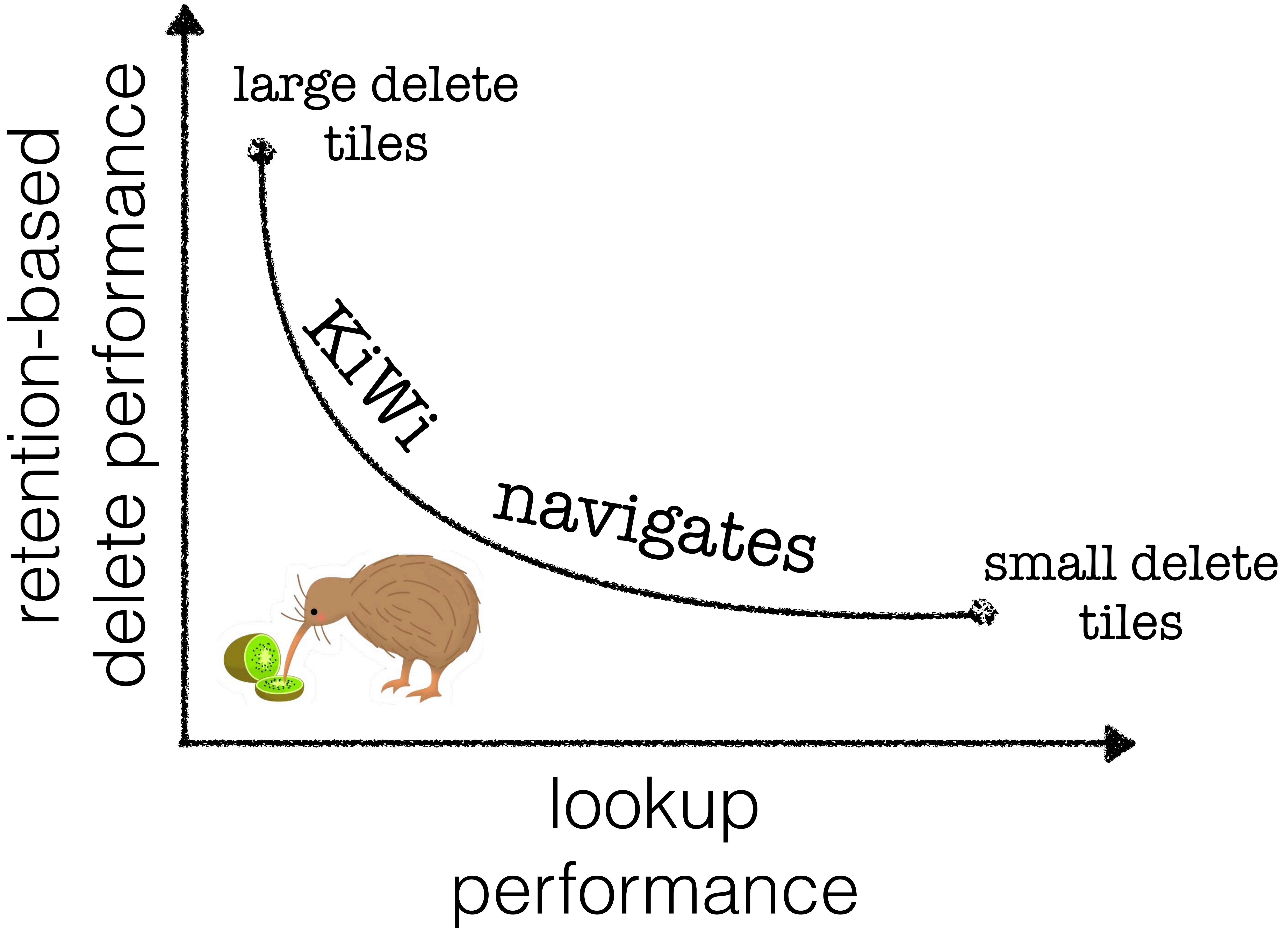
up to 2.5x

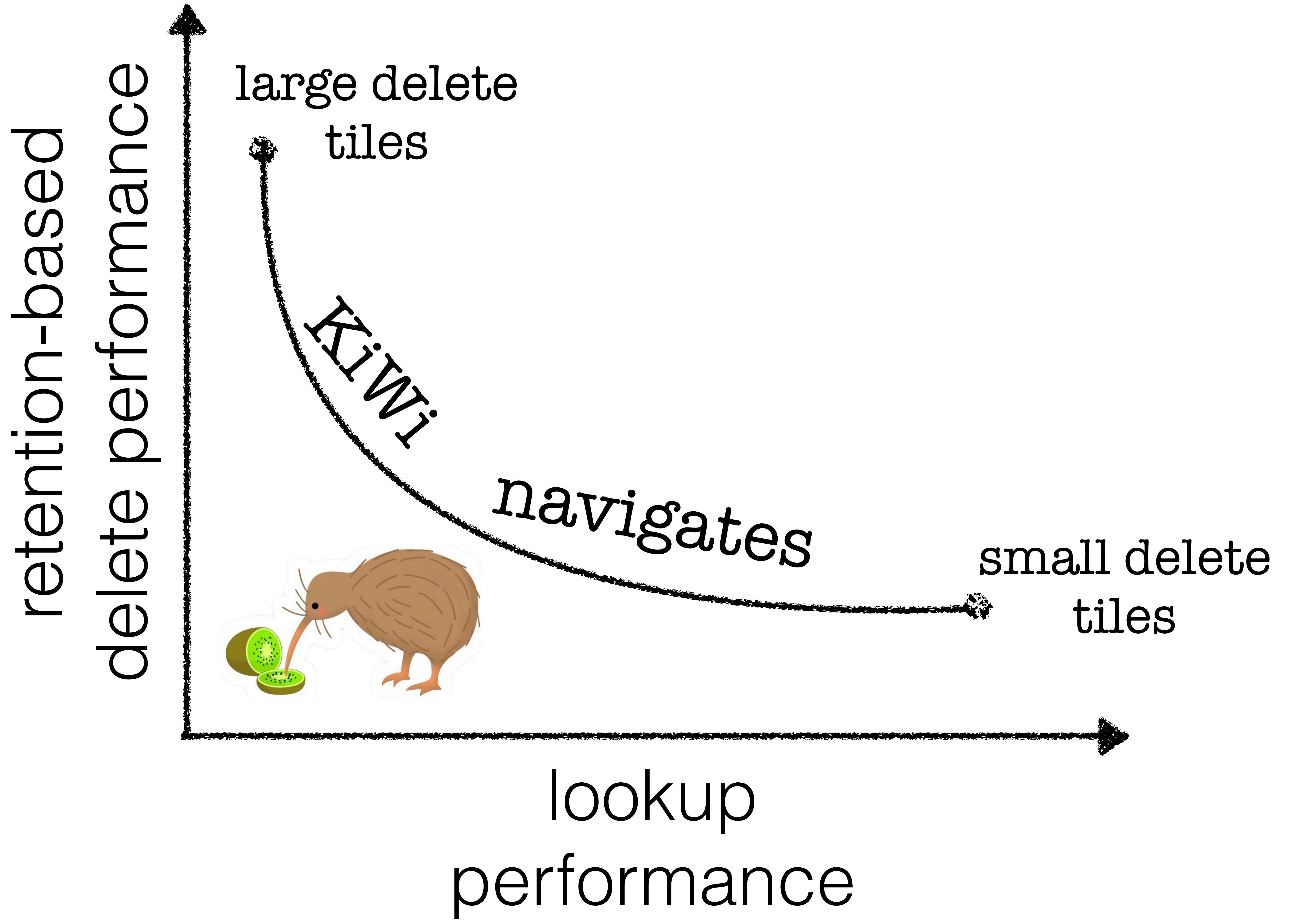
Key Weaving storage layout

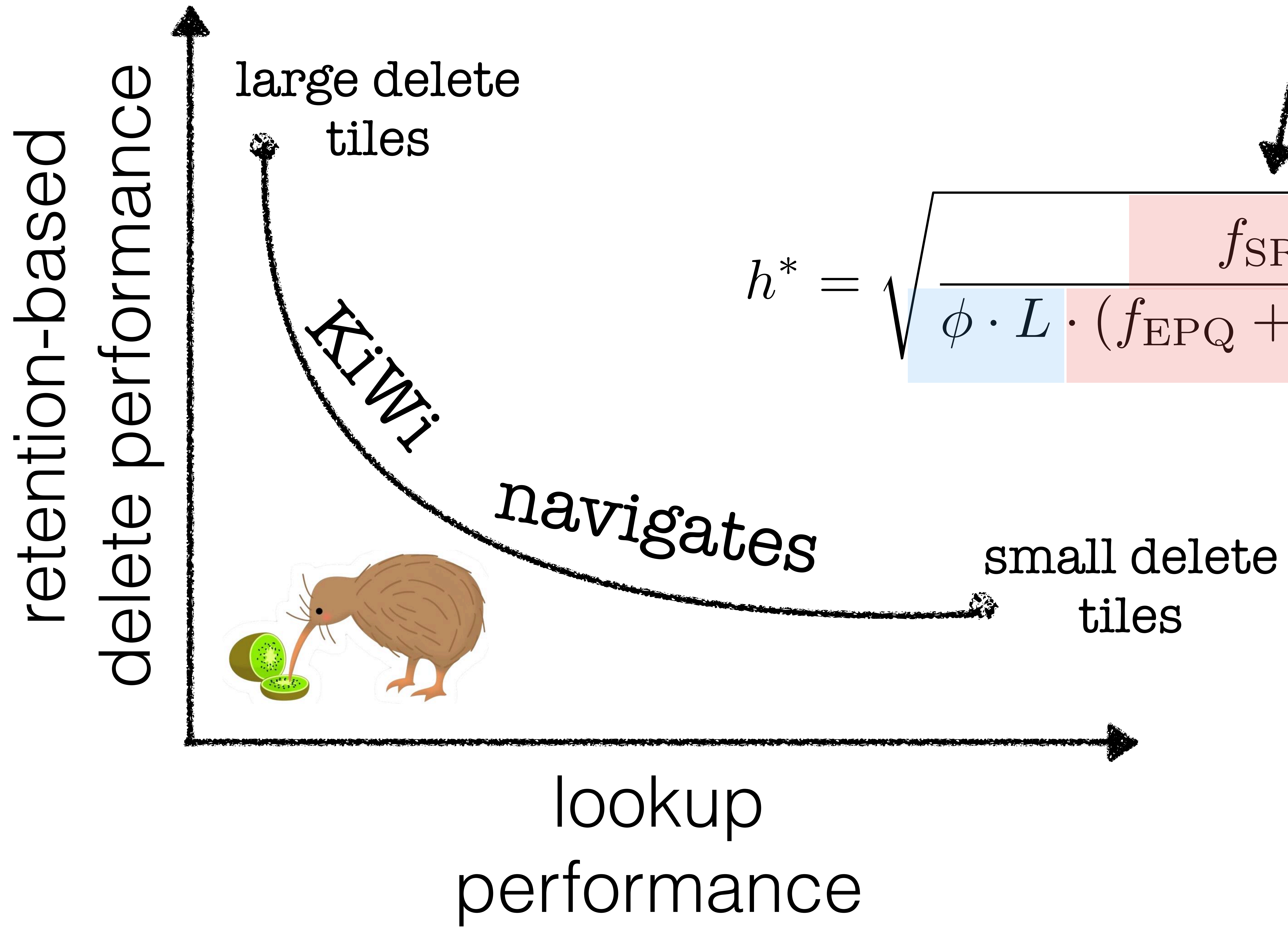


Key Weaving storage layout




retention-based delete performance
lookup performance
large delete tiles
small delete tiles
KiWi
navigates


retention-based
delete performance
large delete
tiles
KiWi
navigates
small delete
tiles
lookup
performance



workload

data structure

$$h^* = \sqrt{\frac{f_{\text{SRD}} \cdot \frac{N}{B}}{\phi \cdot L \cdot (f_{\text{EPQ}} + f_{\text{PQ}}) \cdot + L \cdot f_{\text{SRQ}}}}$$

h^*	= optimal delete tile size
f_{SRD}	= prop. of retention-based deletes
f_{EPQ}	= prop. of empty point queries
f_{PQ}	= prop. of non-empty point queries
f_{SRQ}	= prop. of short range queries
L	= levels in tree
N	= entries in tree
B	= entries in a page
ϕ	= false positive rate of query filter

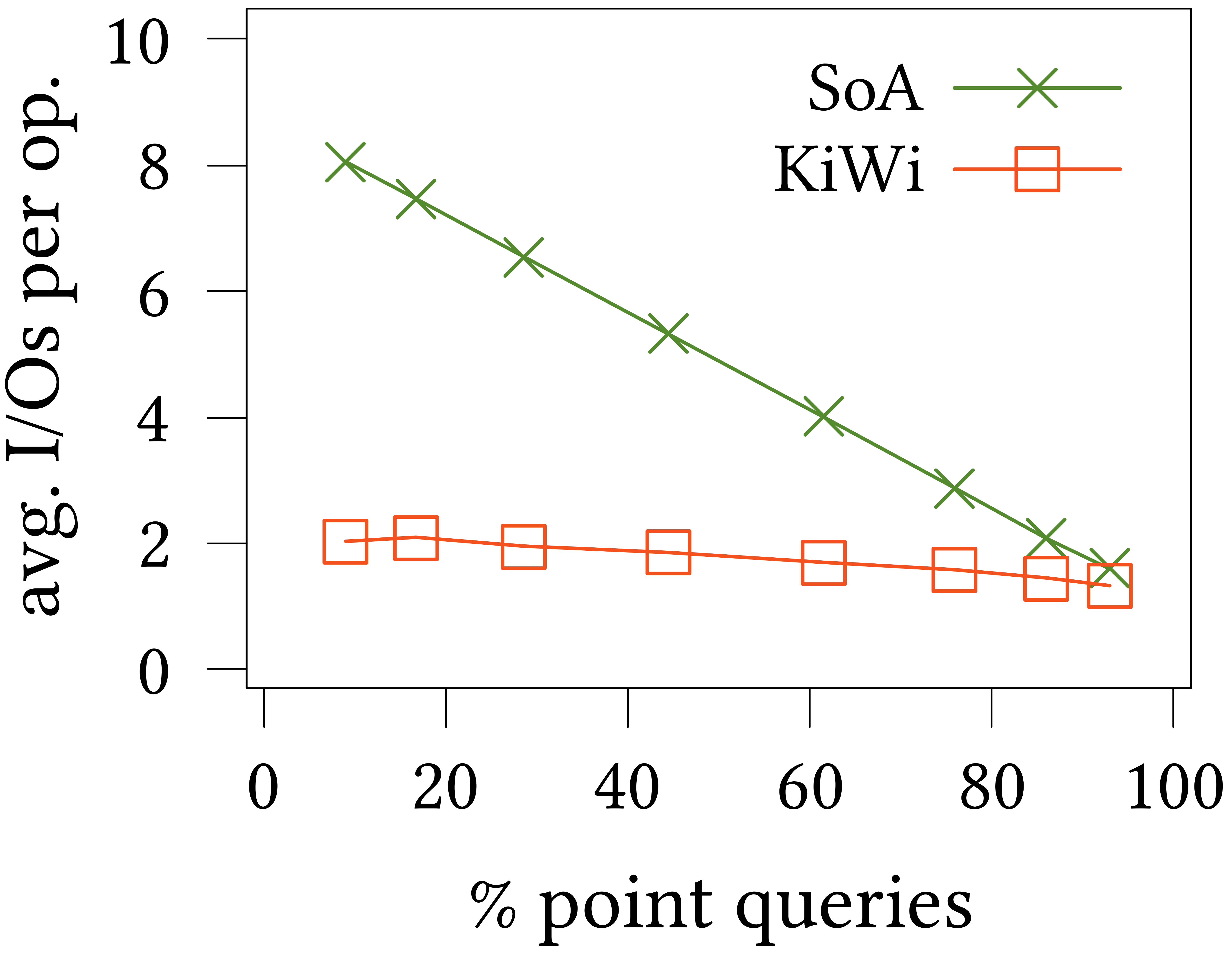
better overall performance

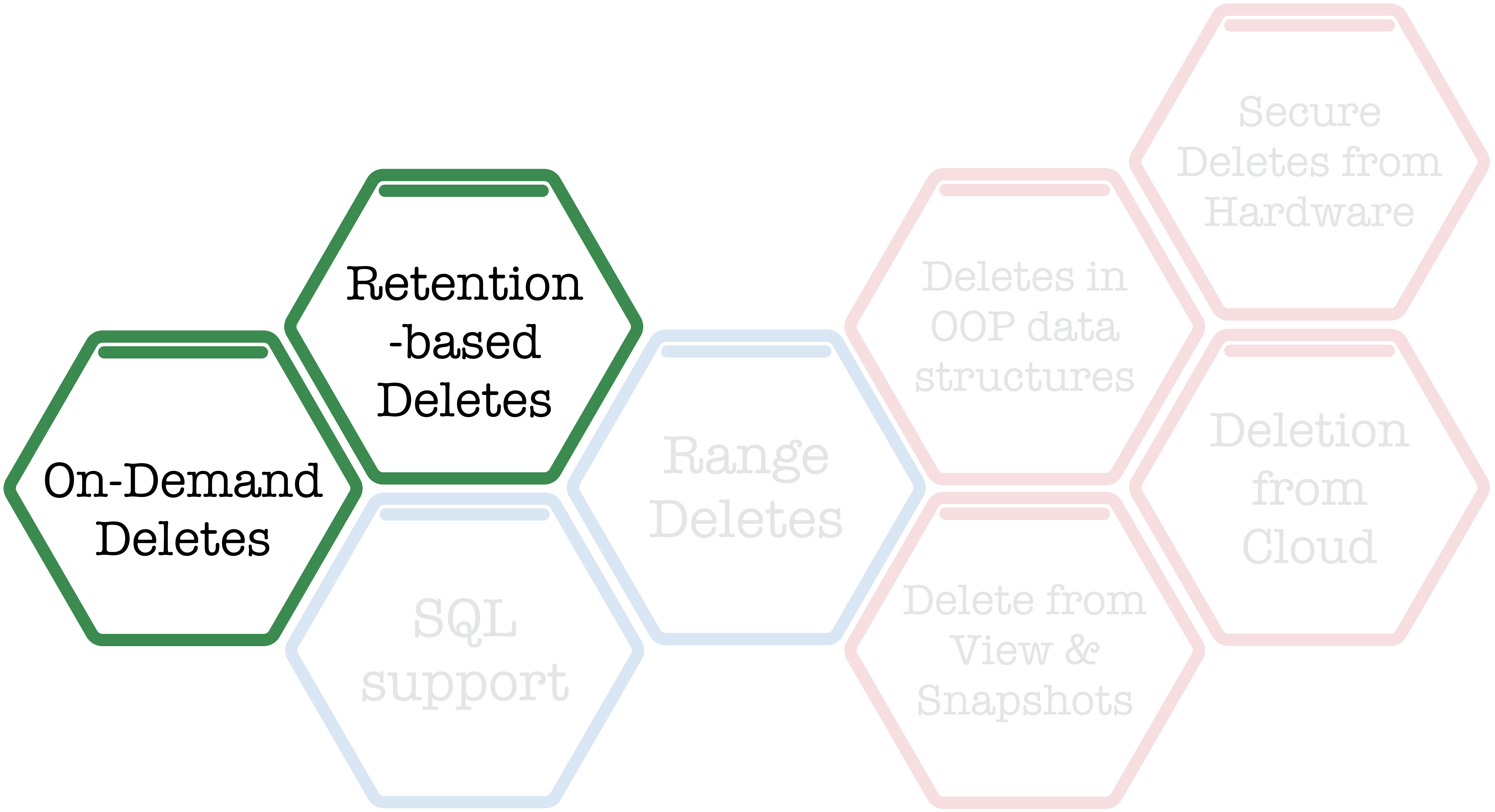
up to 4x

superior delete performance

up to 2.5x

5M entries, buffer = file = 256 pages, T=10




On-Demand Deletes
Retention-based Deletes
SQL support
Range Deletes
Deletes in OOP data structures
Delete from View & Snapshots
Secure Deletes from Hardware
Deletion from Cloud

FADE

SQL support

Range Deletes

Deletes in OOP data structures

Delete from View & Snapshots

Secure Deletes from Hardware

Deletion from Cloud

FADE

SQL support

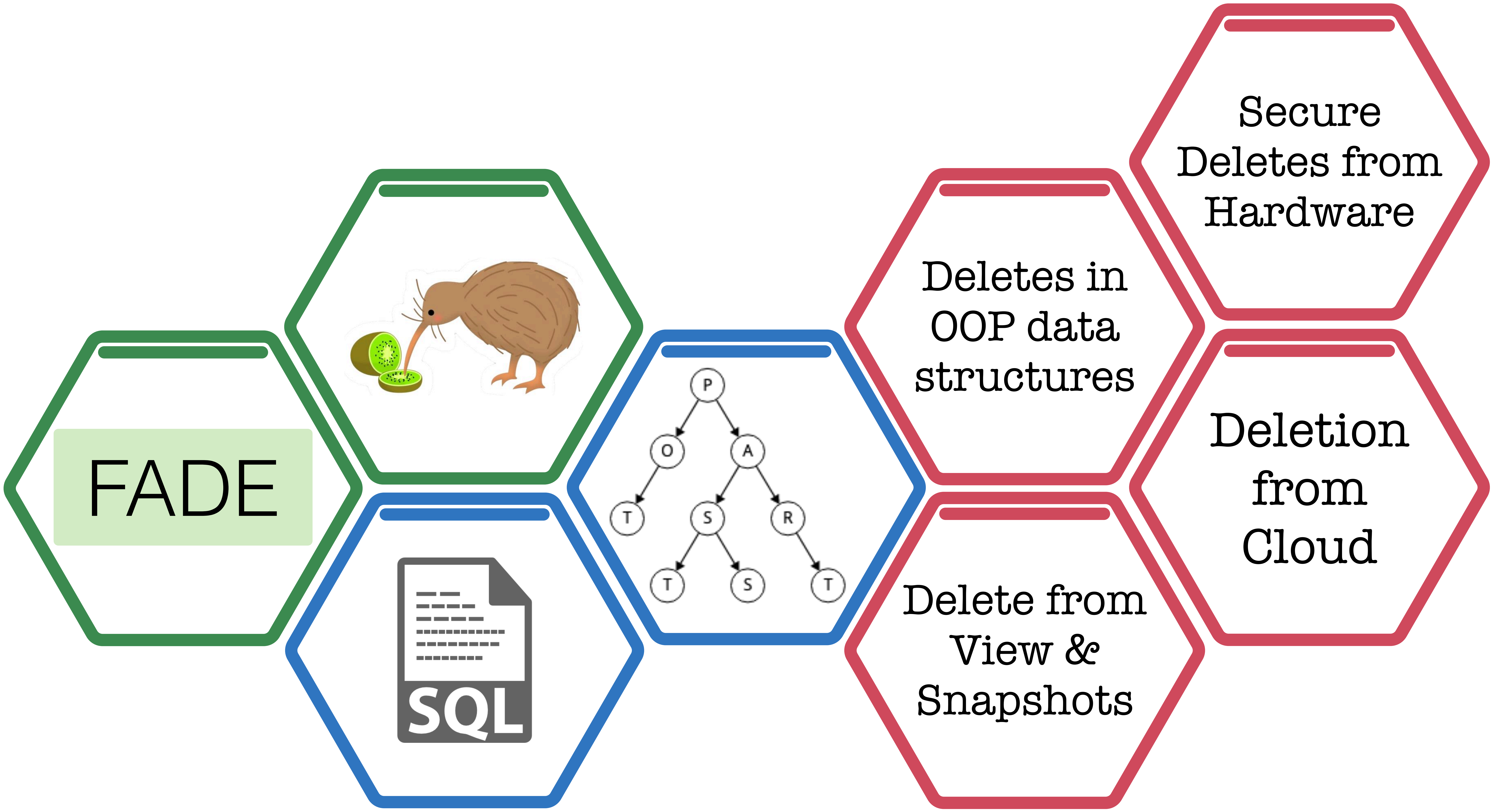
Range Deletes

Deletes in OOP data structures

Delete from View & Snapshots

Secure Deletes from Hardware

Deletion from Cloud


FADE
SQL
P
O
A
T
S
R
T
S
T
Deletes in OOP data structures
Delete from View & Snapshots
Secure Deletes from Hardware
Deletion from Cloud

THANK YOU!

Actively seeking students and collaborators!

Subhadeep Sarkar

https://subhadeep.net/

Brandeis UNIVERSITY