# Range Delete Filter for LSM Tree

Ming-Han Hsieh
Yu-Cheng Huang
Jingyi Li

# Outline

1. Intro to Range Delete
2. RDF Design
   i) Skyline RDF
   ii) Perlevel RDF
3. Our implementation
4. Experiment Results
5. Future work

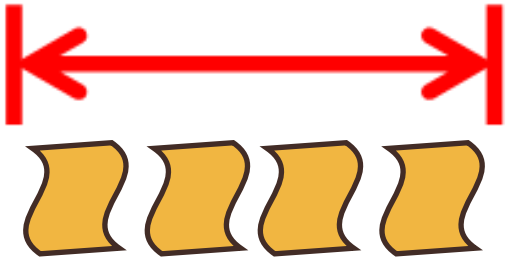
Target :

Utilize range delete information to decrease the unnecessary disk IO

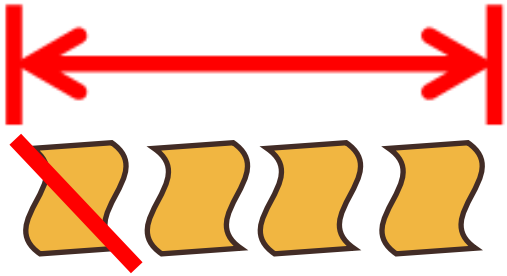
# What is a range delete?

# What is a range delete?

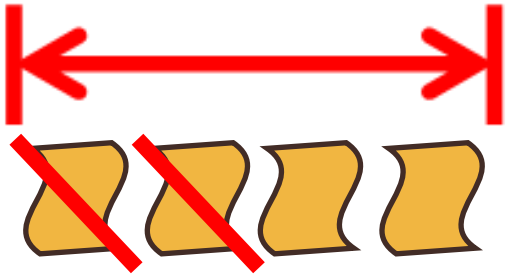
# What is a range delete?



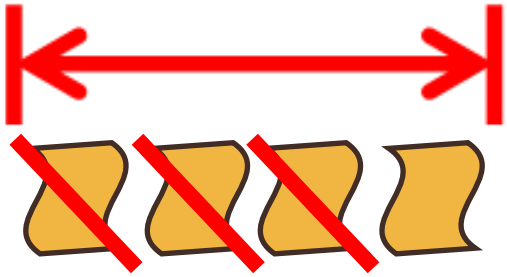
# What is a range delete?



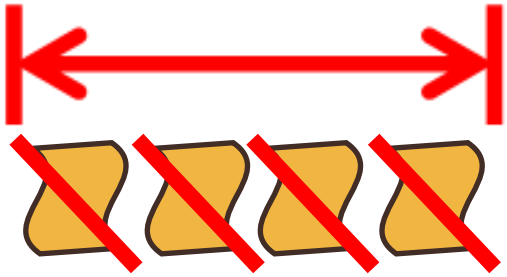
# What is a range delete?



# What is a range delete?



# What is a range delete?



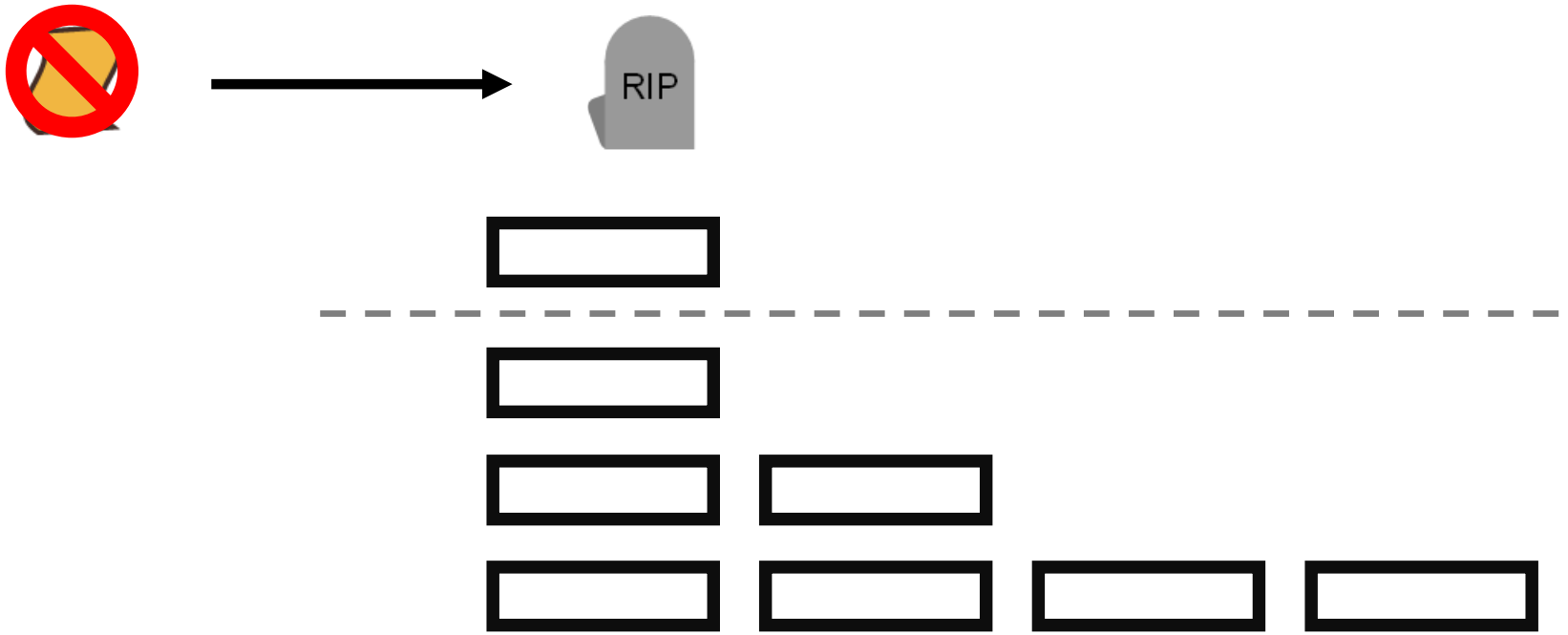
# How do we represent it?

# Point Delete

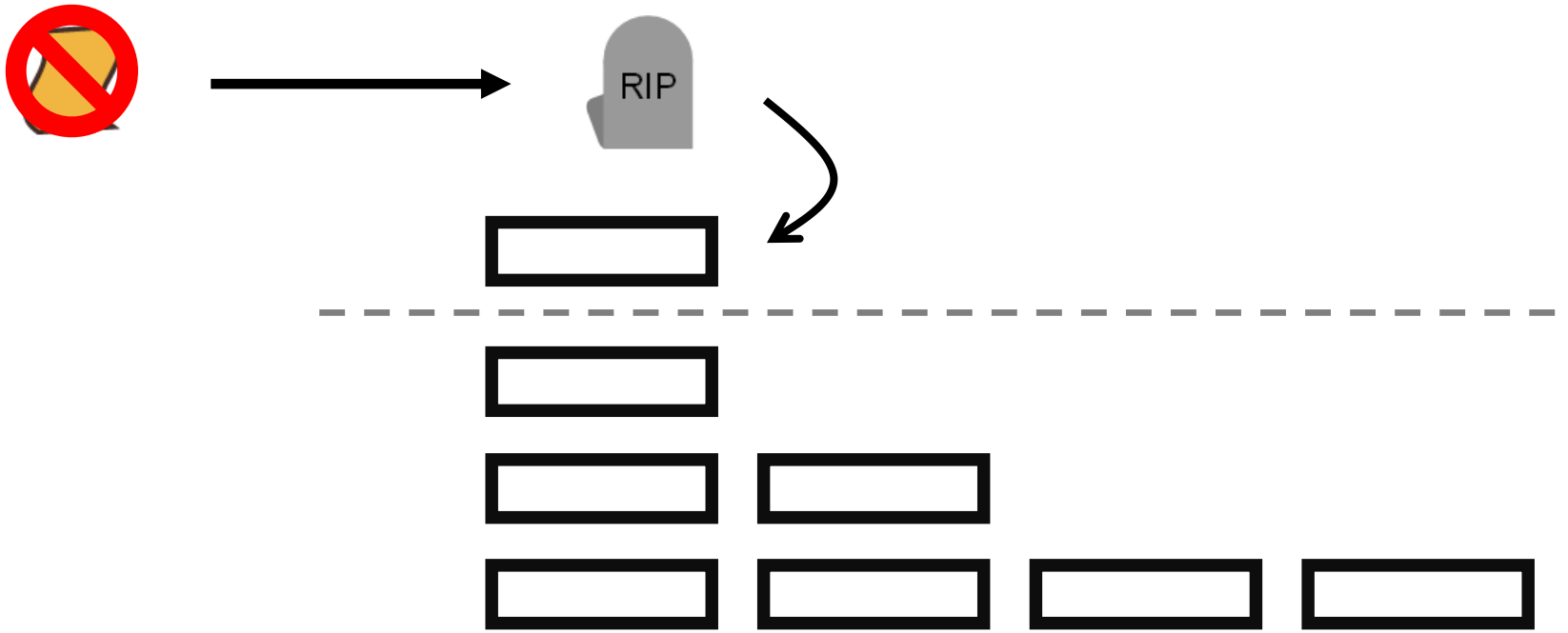
# Point Delete

# Point Delete

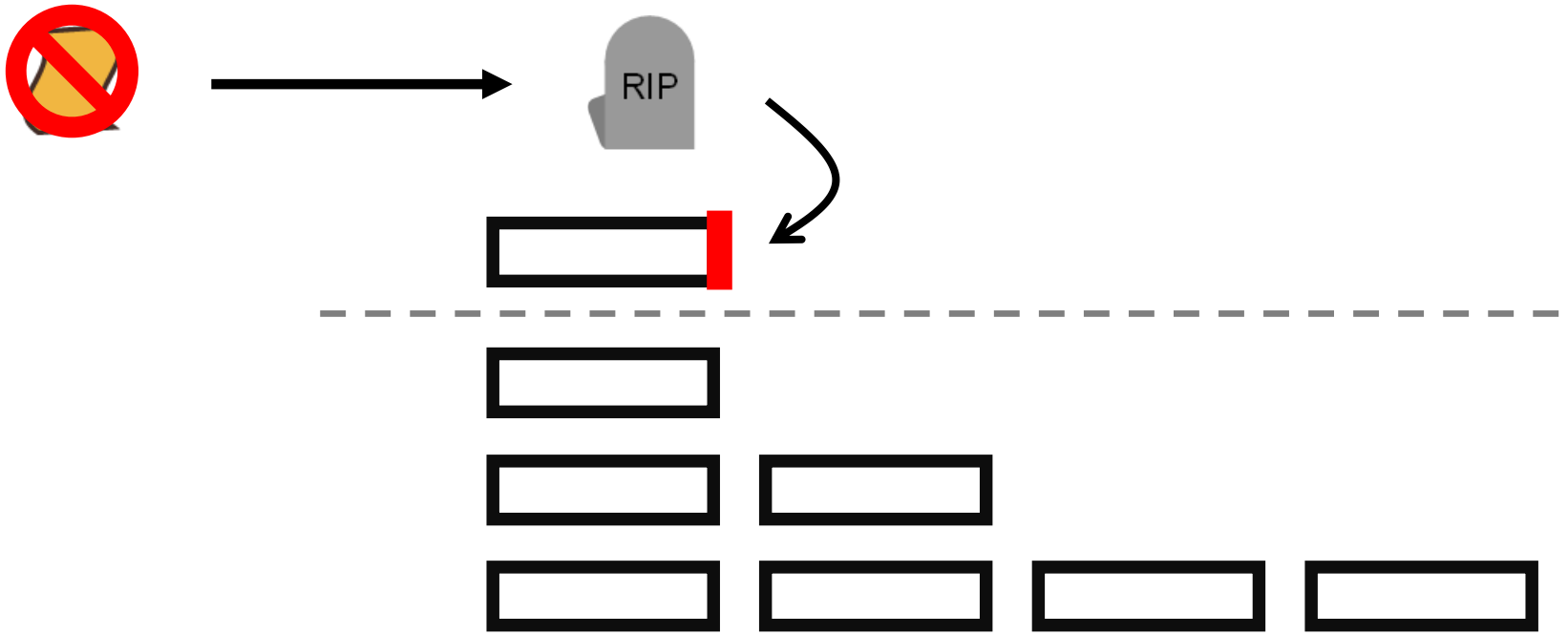
# Point Delete



# Point Delete



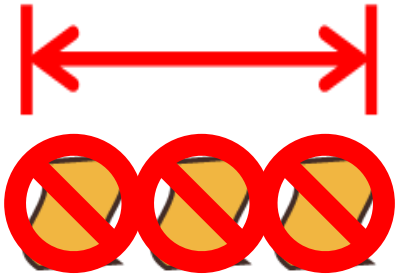
# Point Delete



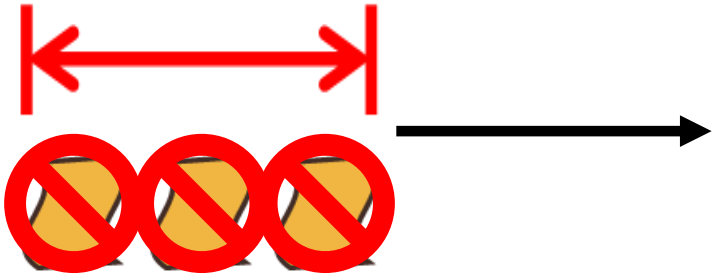
# Range Delete?

# Range Delete

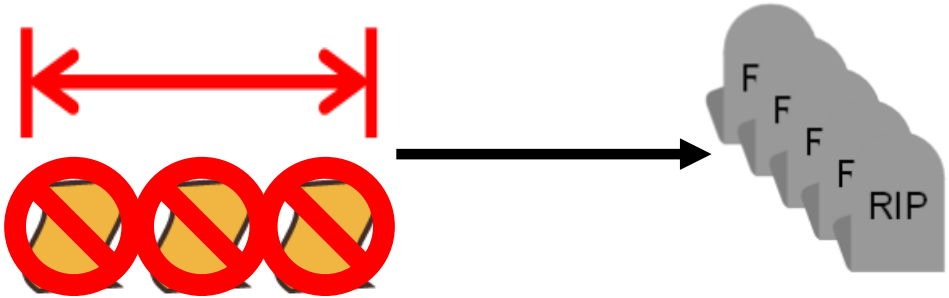
# Range Delete



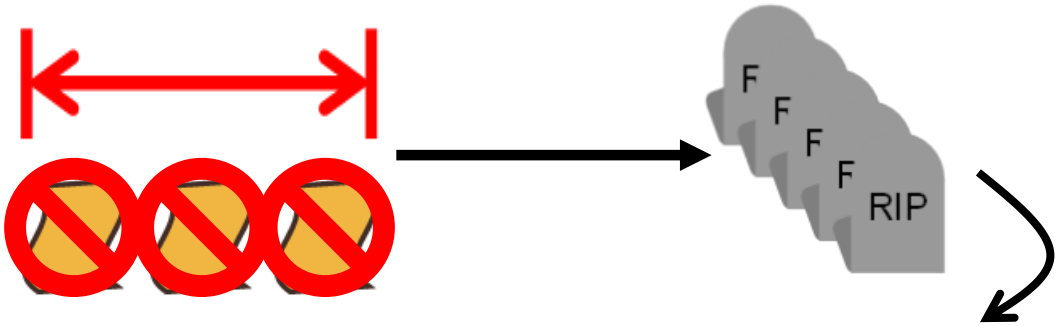
# Range Delete



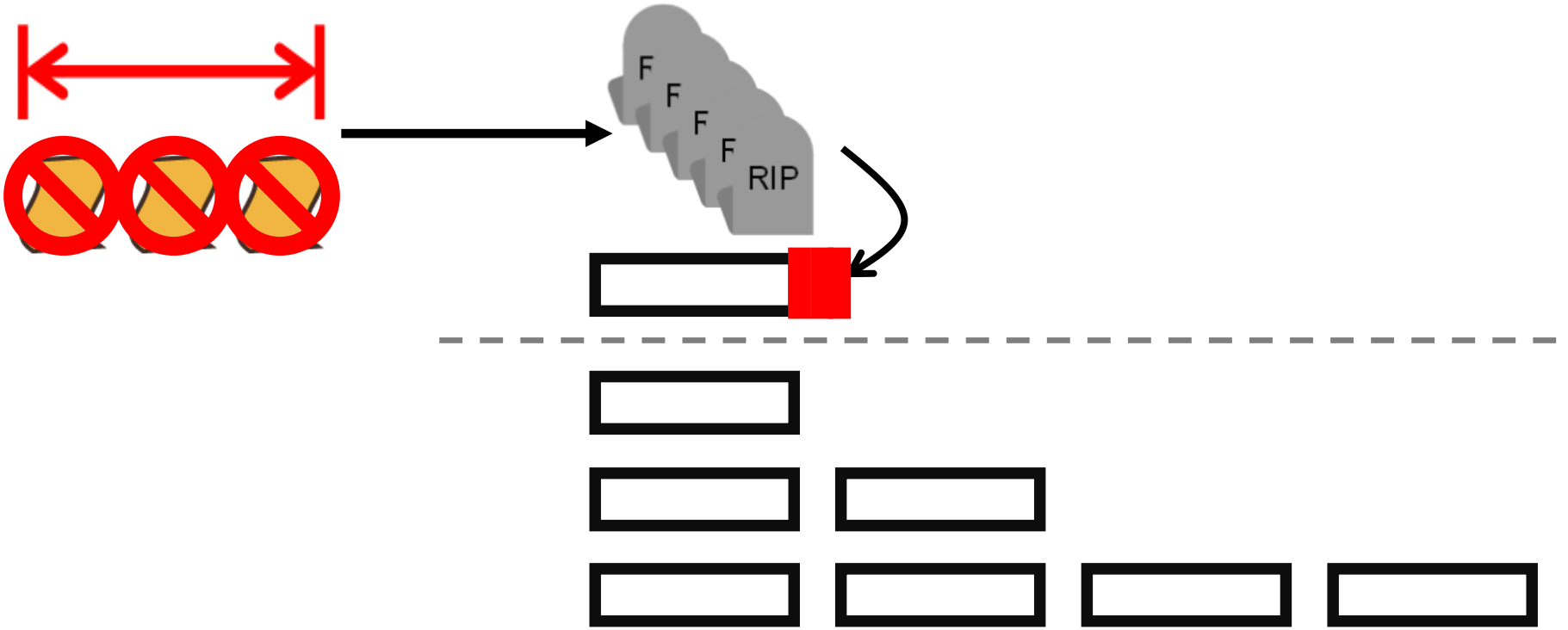
# Range Delete



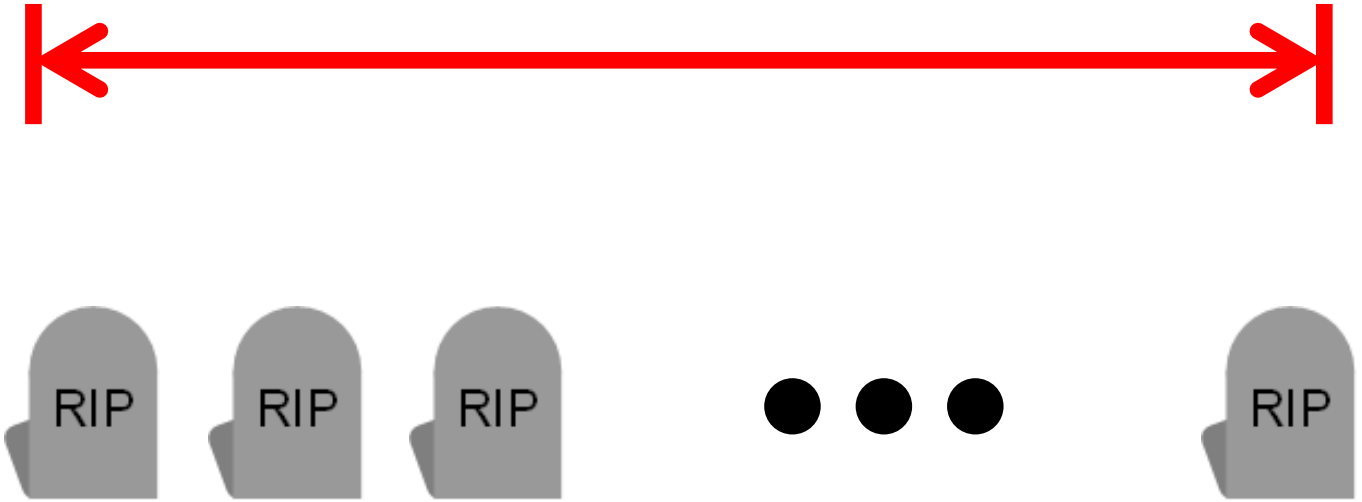
# Range Delete



# Range Delete



# Very long range?


RIP
RIP
RIP
RIP

1

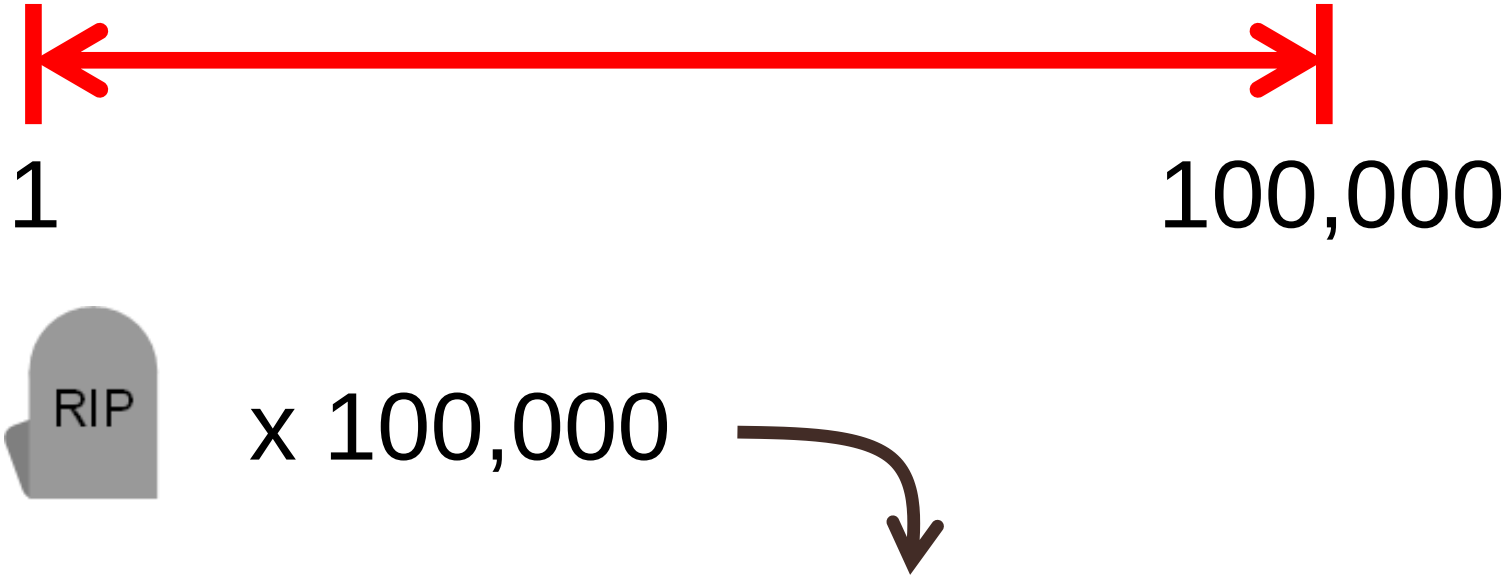

1
100,000


1
100,000
RIP

1

100,000


x 100,000


1
100,000
RIP
x 100,000


1
100,000

1 100,000

(start, end) = (1, 100000)

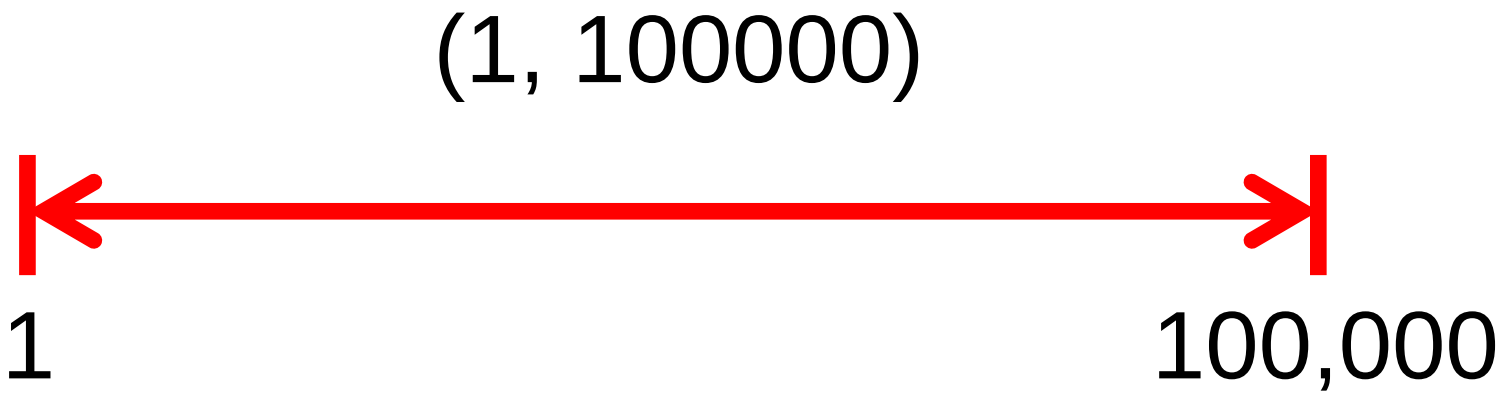
1 100,000

(start, end) = (1, 100000) A tuple

1 100,000

(start, end) = (1, 100000)

Range Tombstone

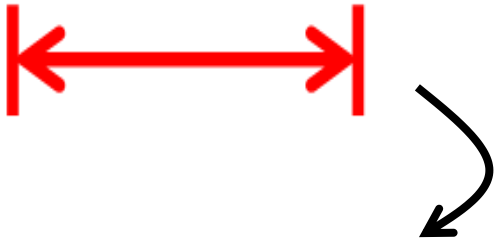



Range Tombstone

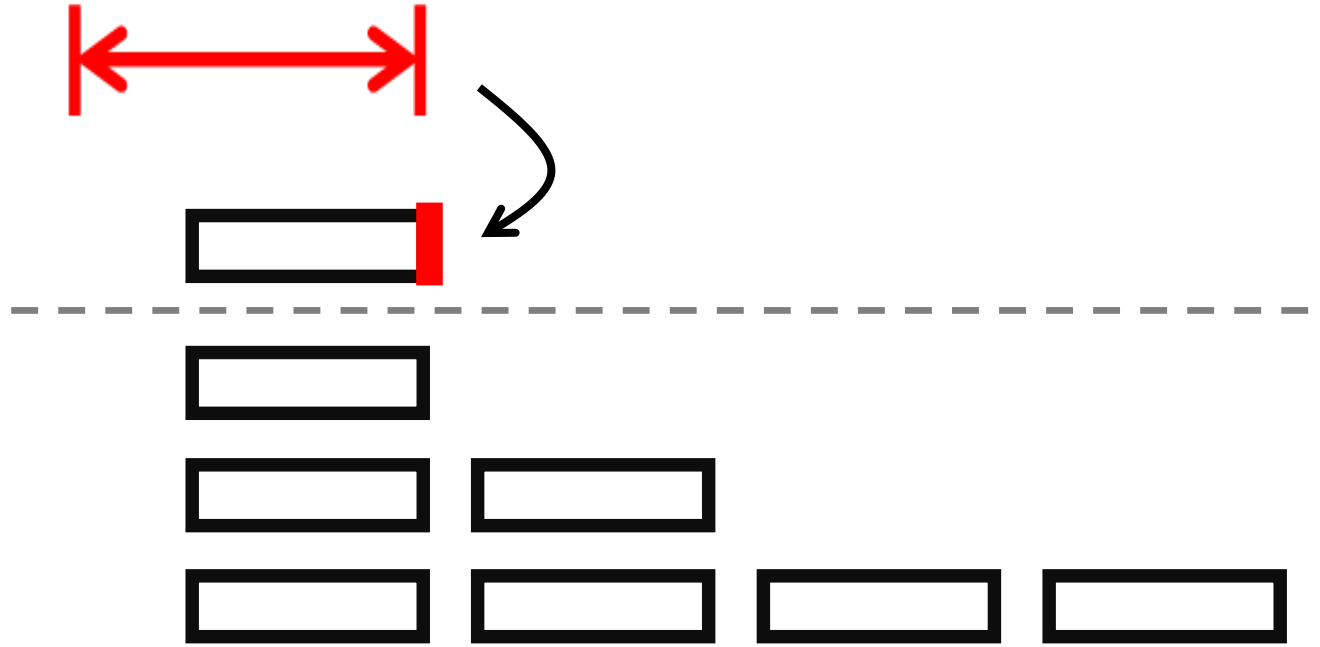
# Range Delete

# Directly append

# Directly append



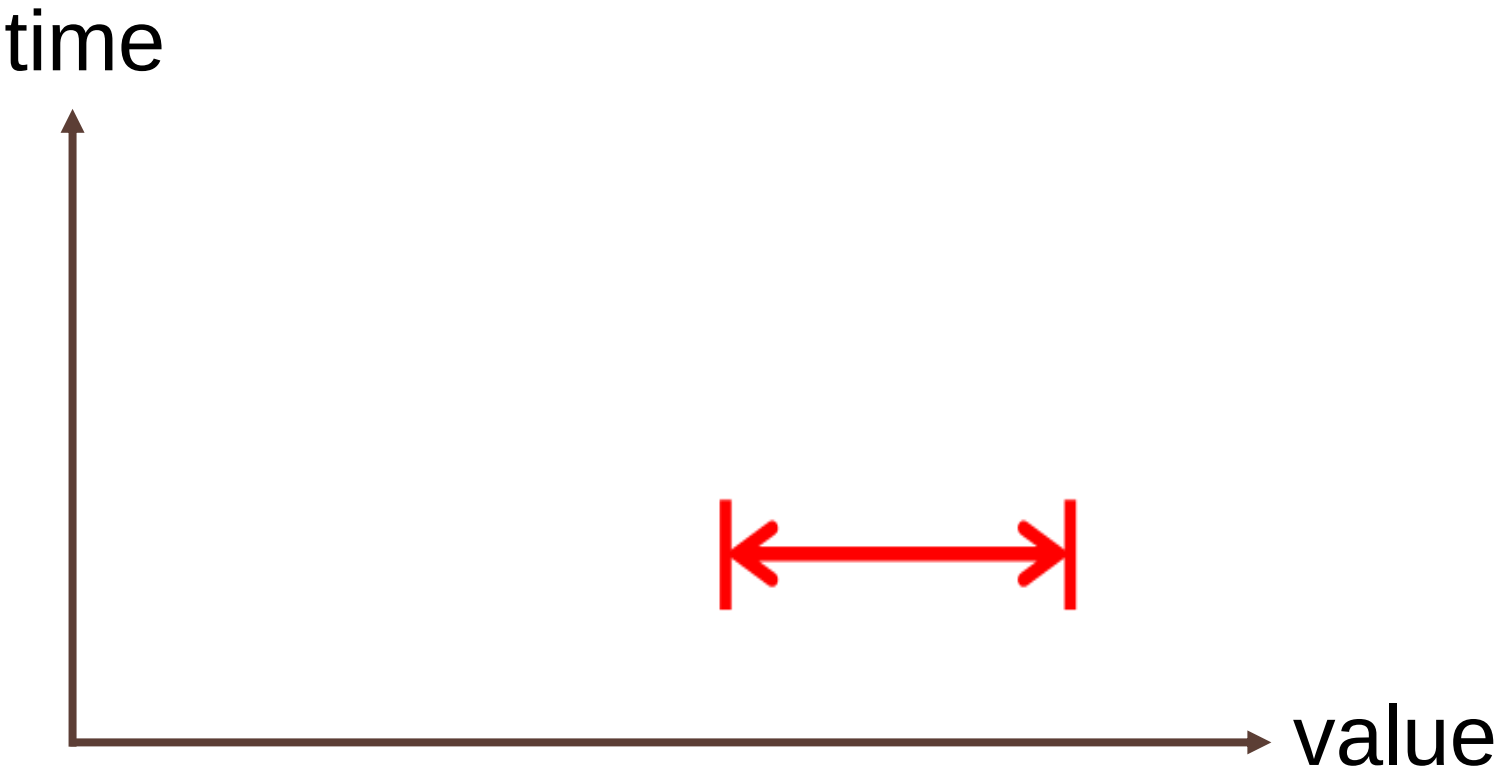
# Directly append

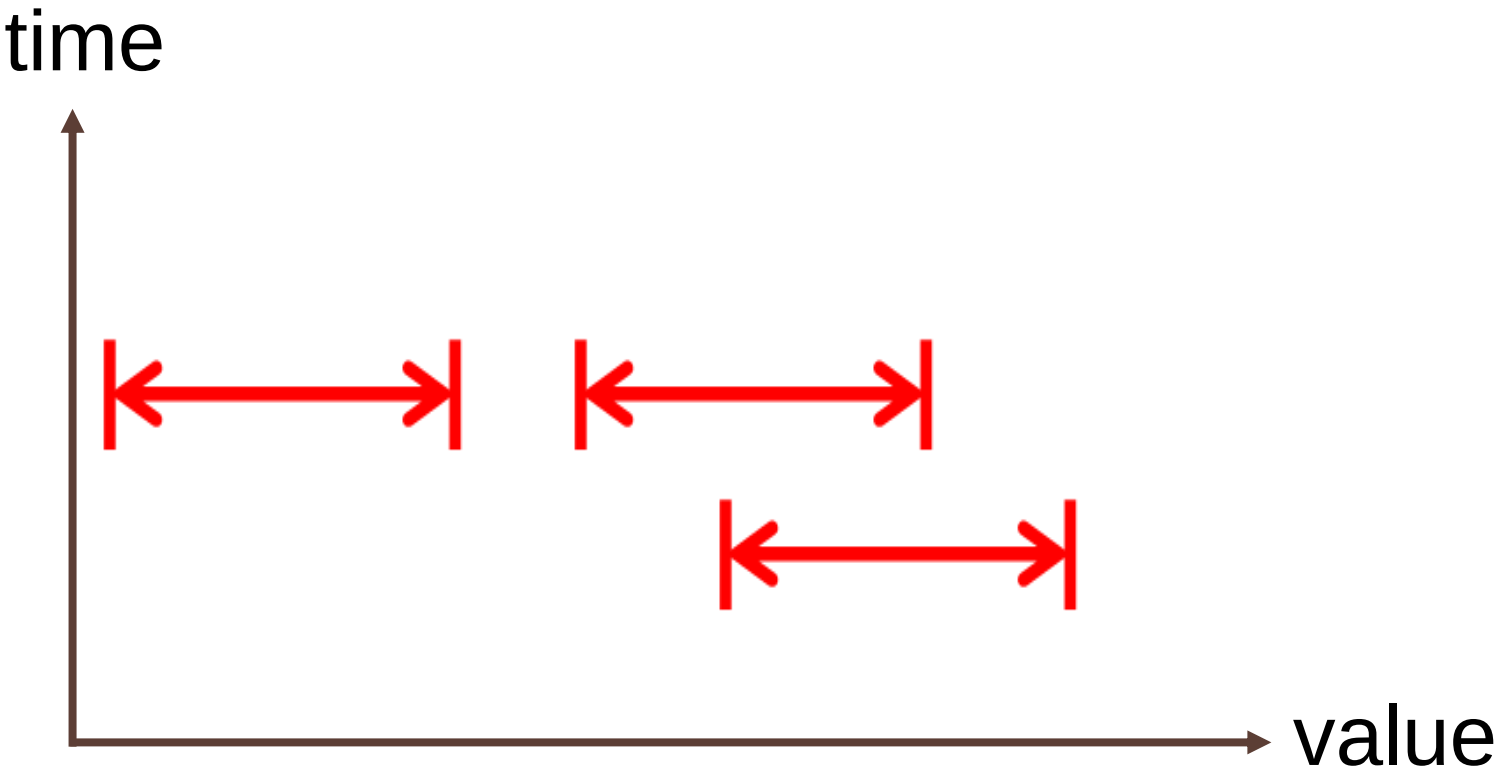


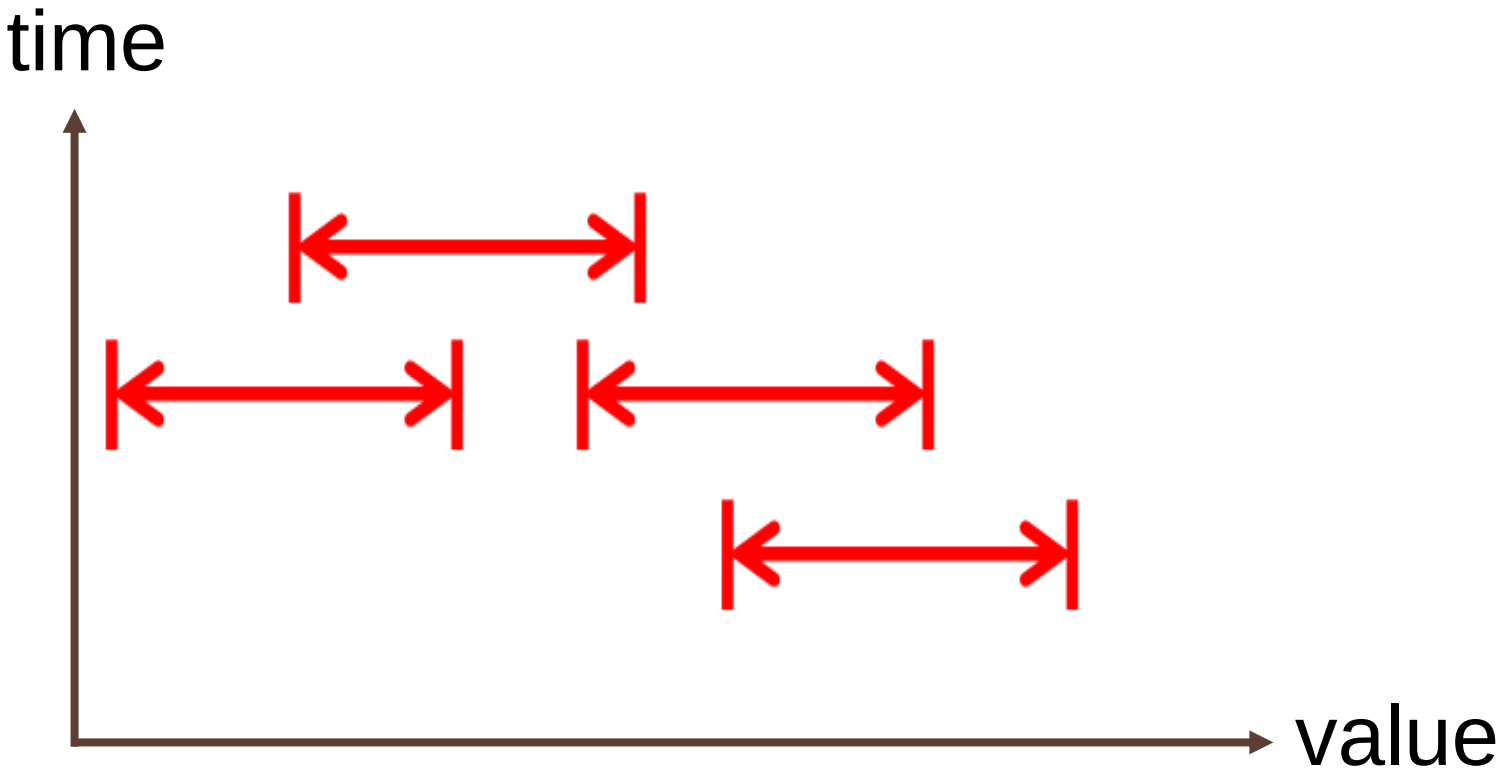
OR

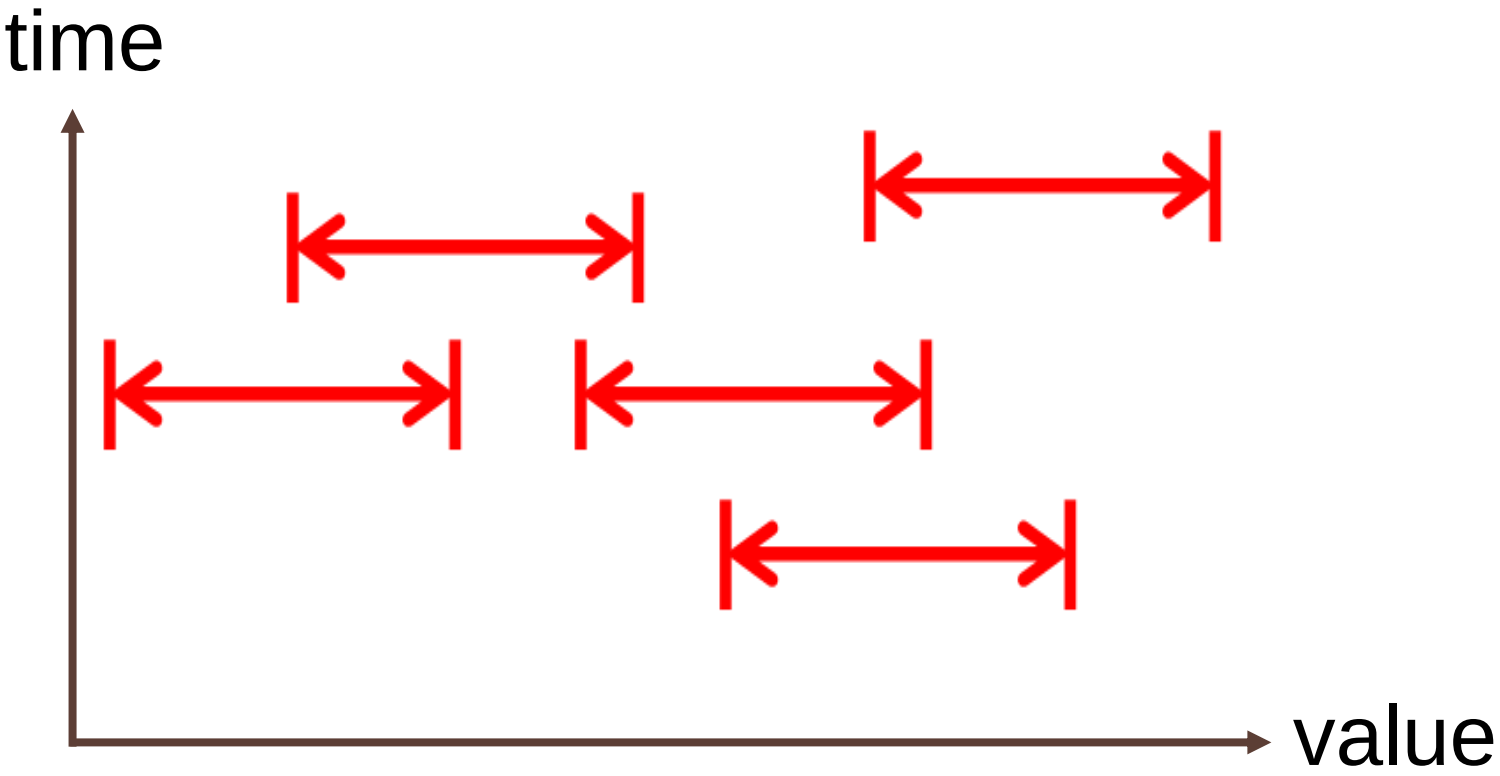
# RocksDB

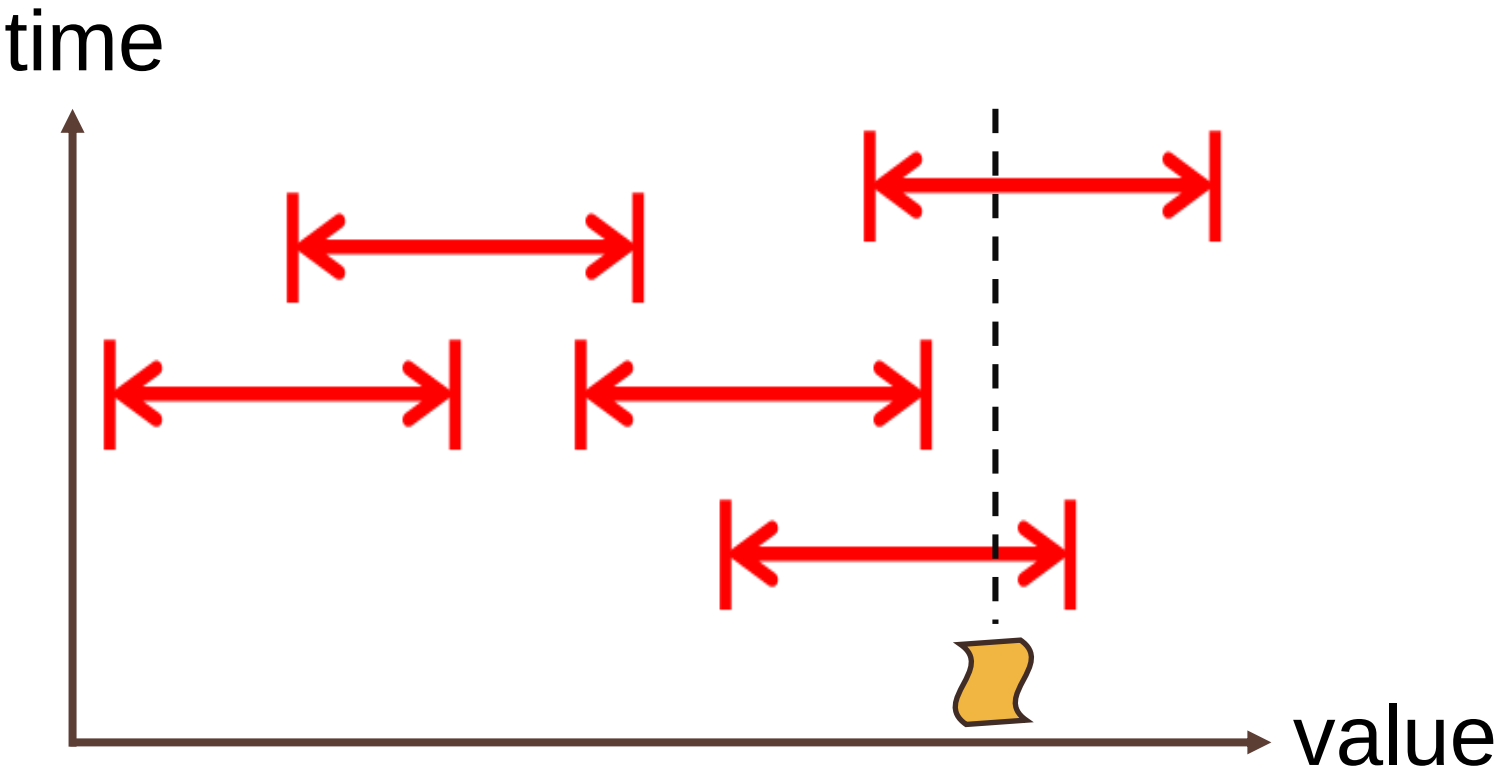

time
value

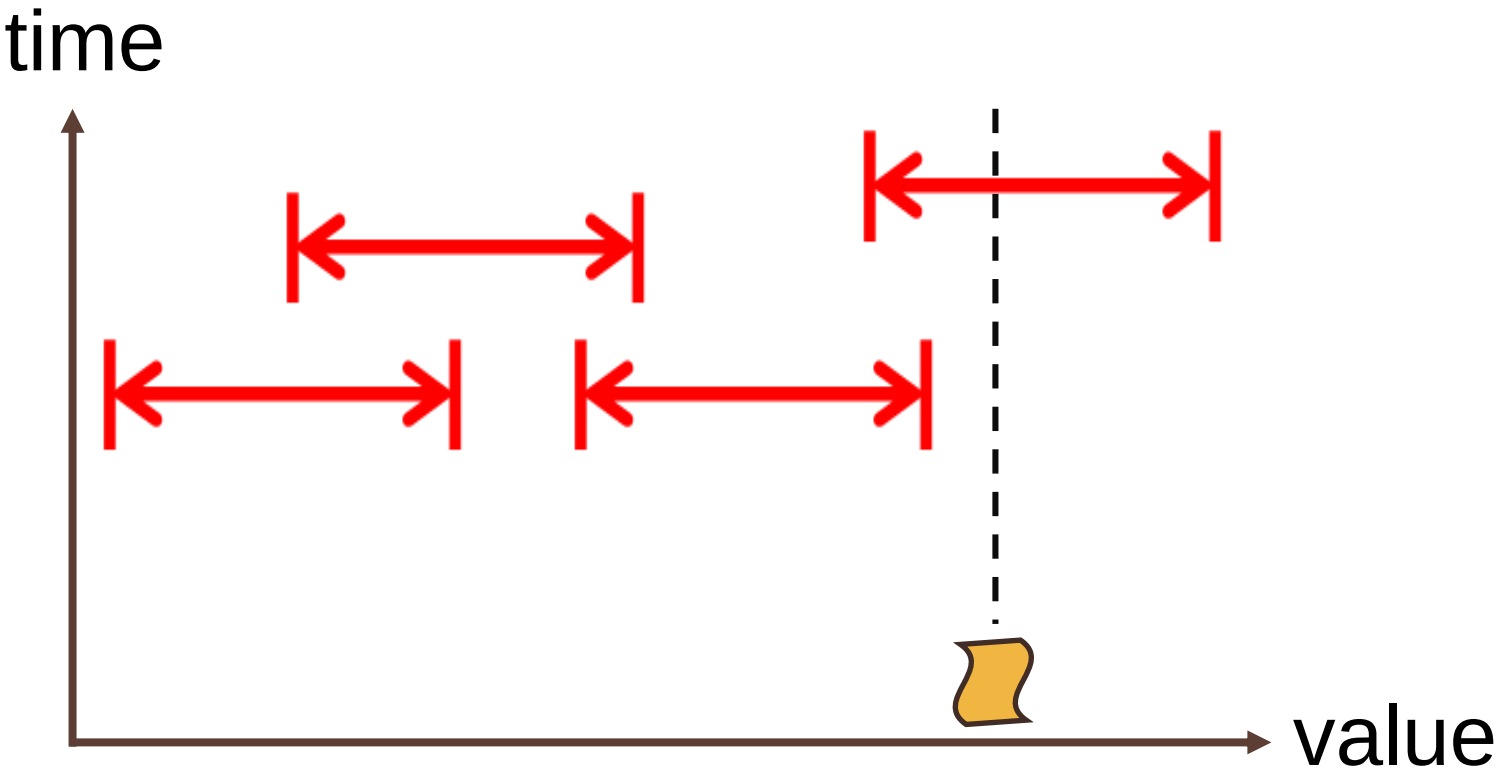

time
value

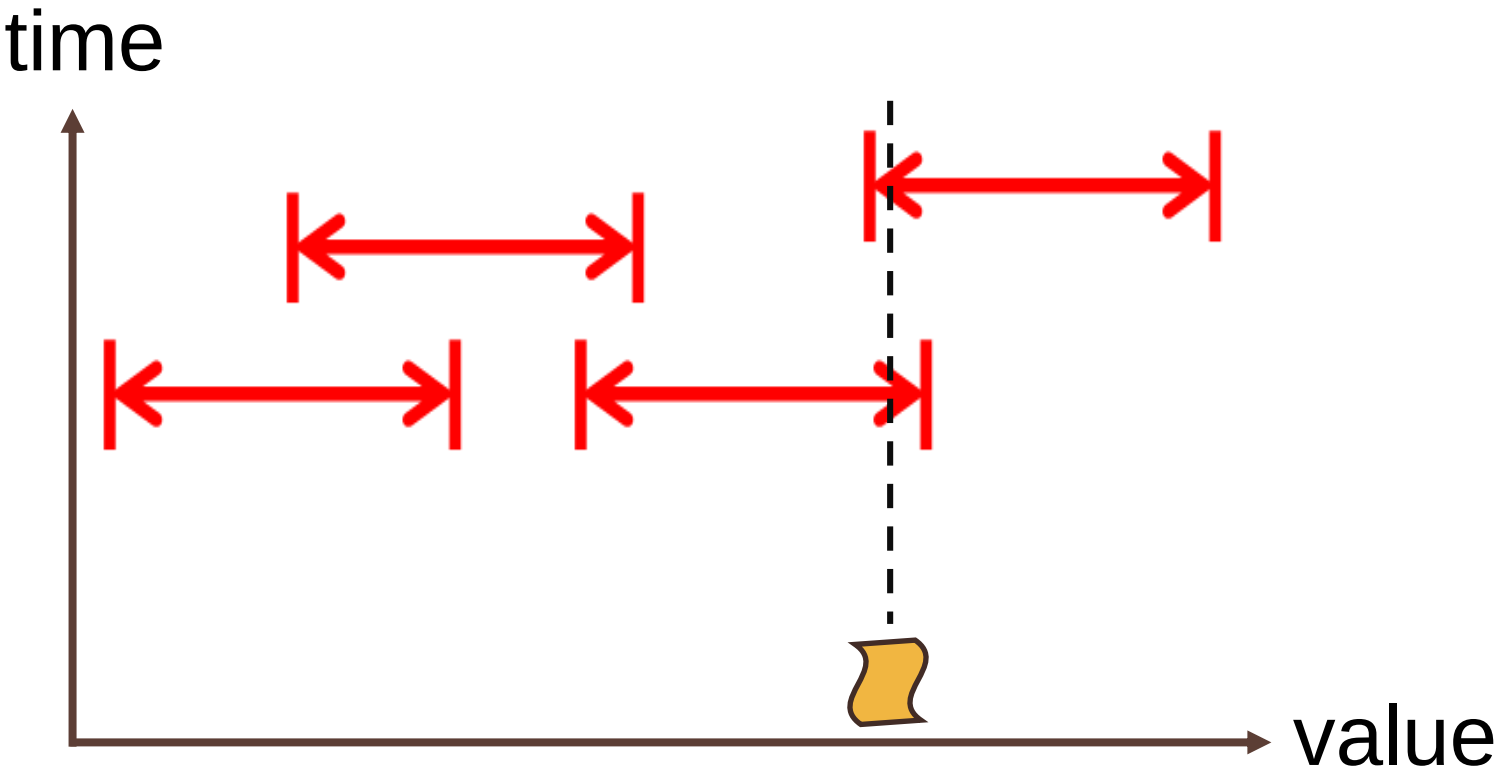

time
value

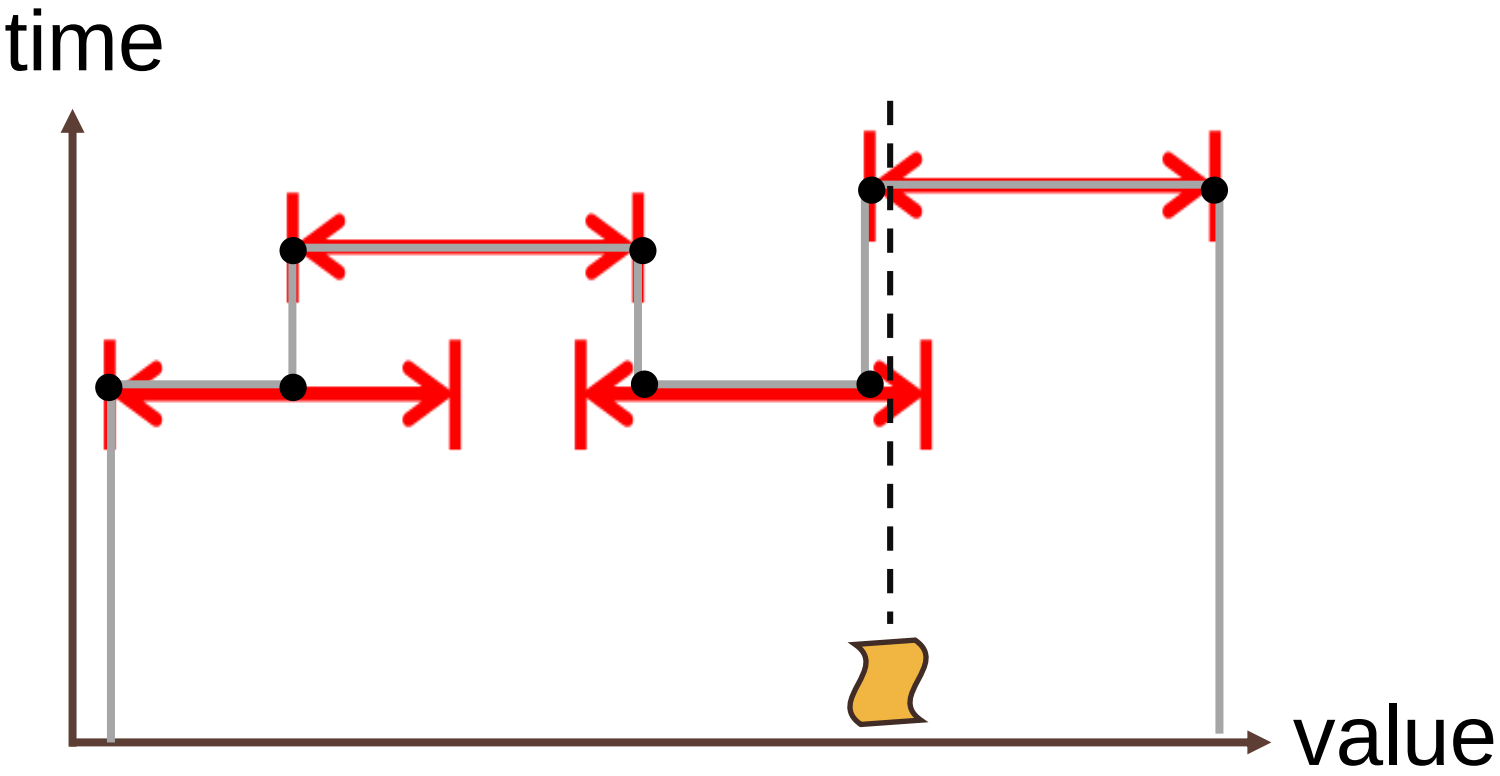

time
value

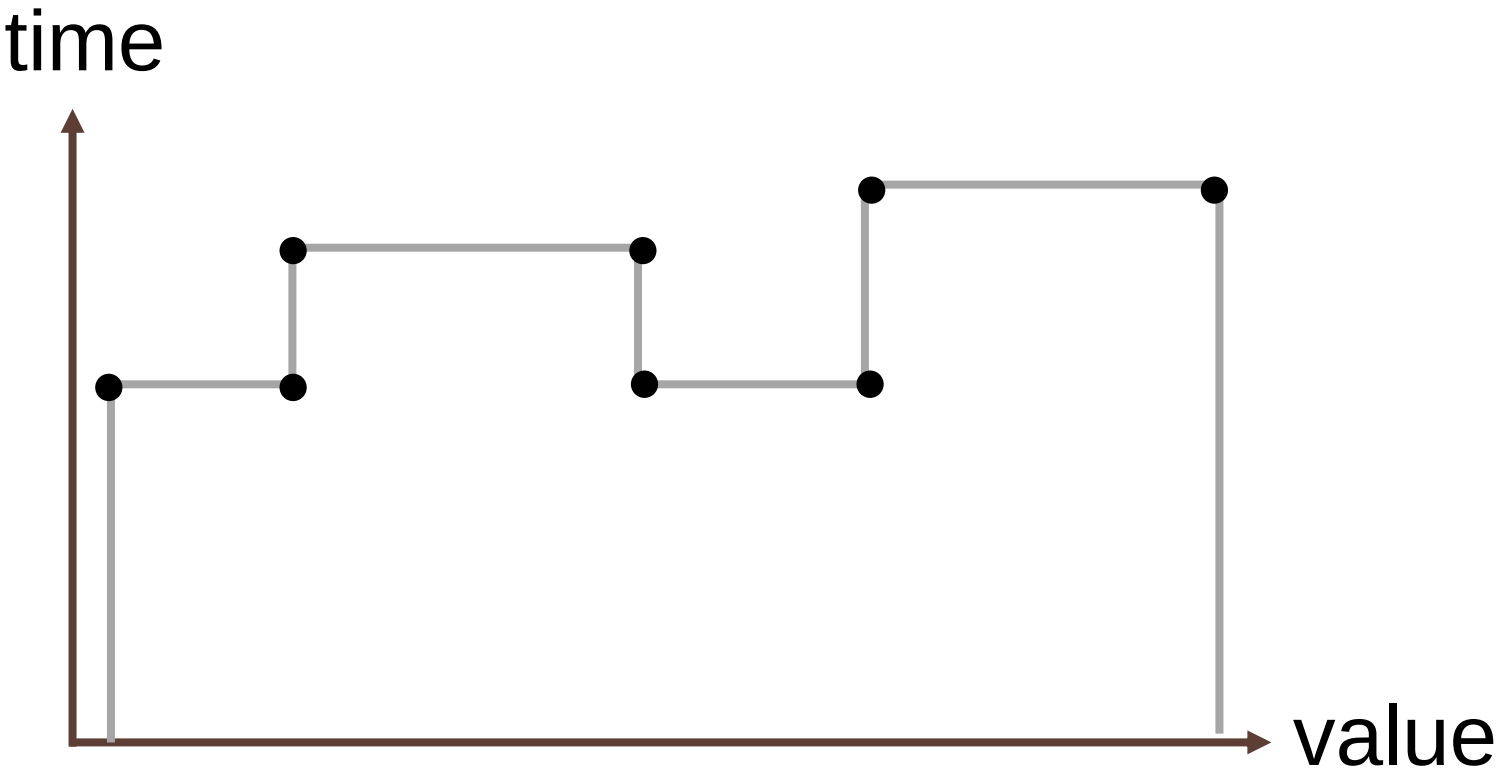

time
value

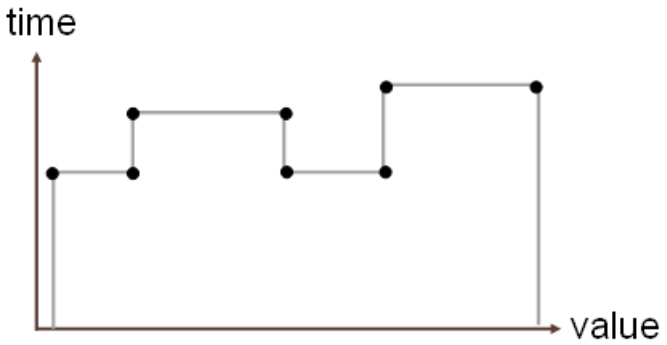

time
value


time
value


time
value

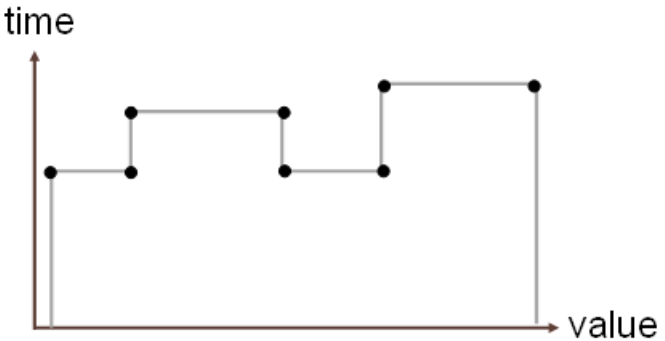

time
value


time
value




# Skyline RDF

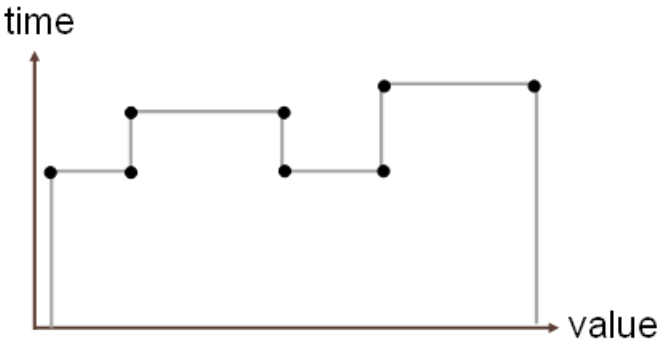
(start, end, **time**)



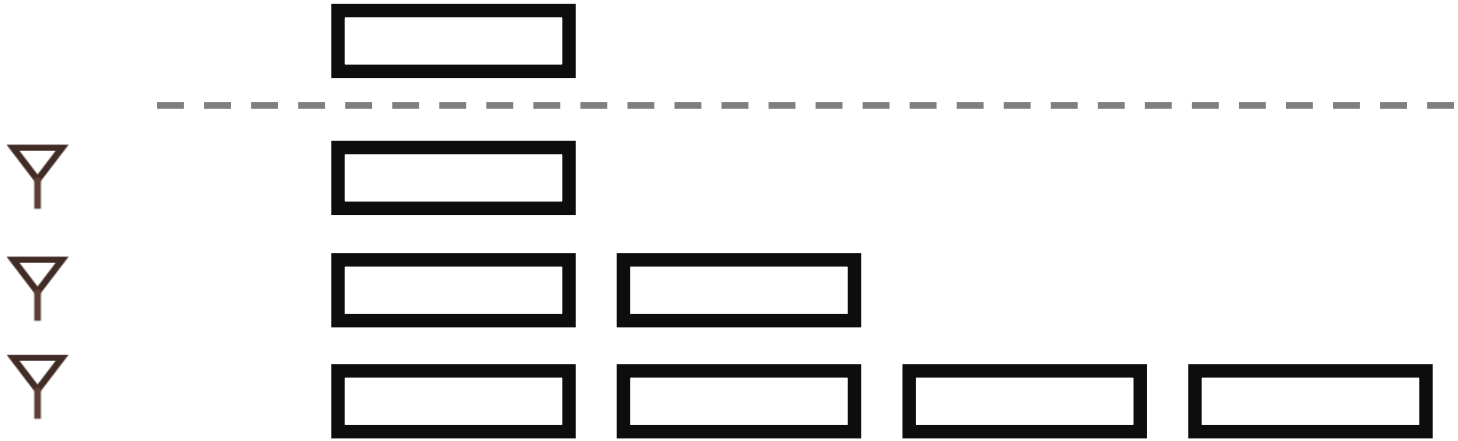

Skyline RDF

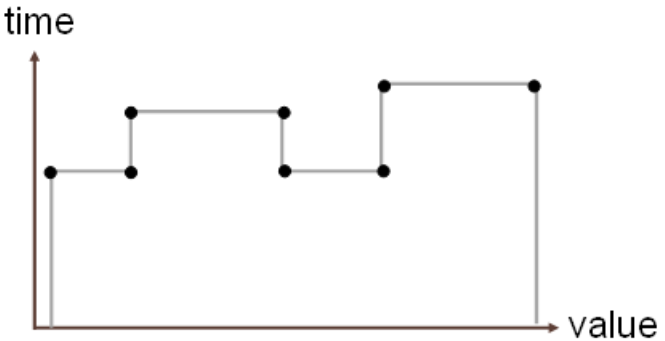
# Point Query (PQ)

- PQ



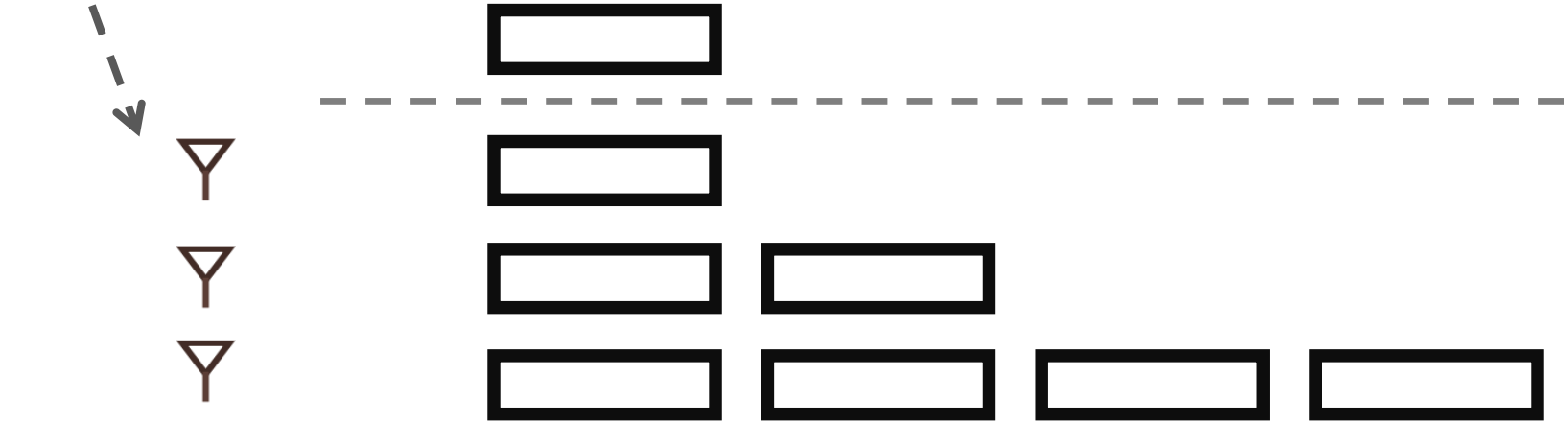

Skyline RDF

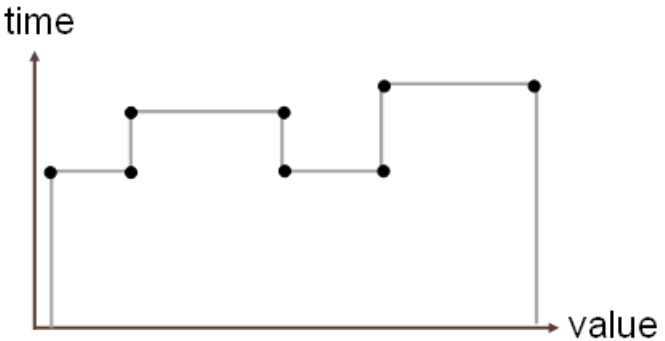





PQ

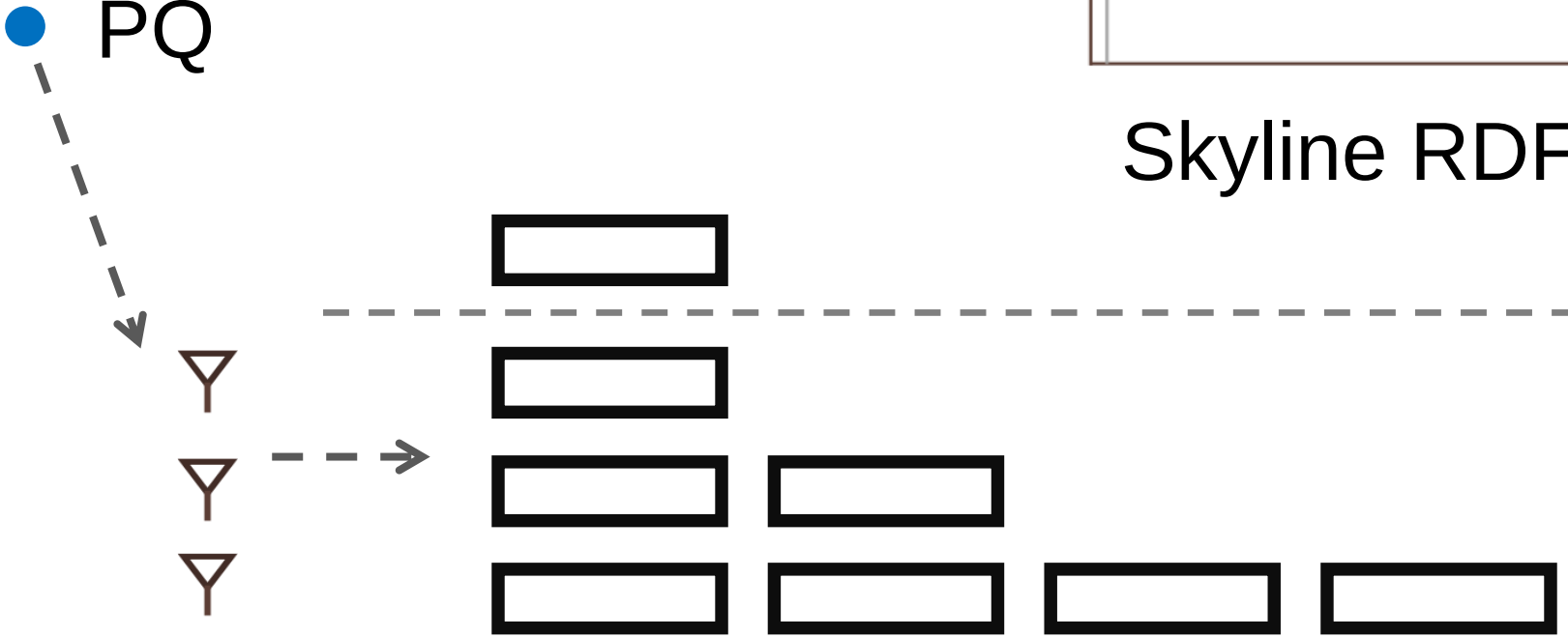
Skyline RDF






PQ

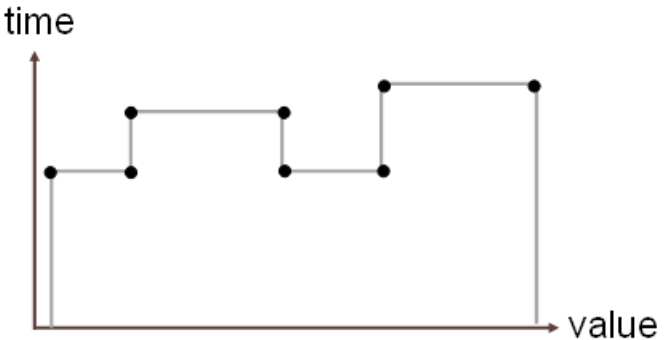
Skyline RDF



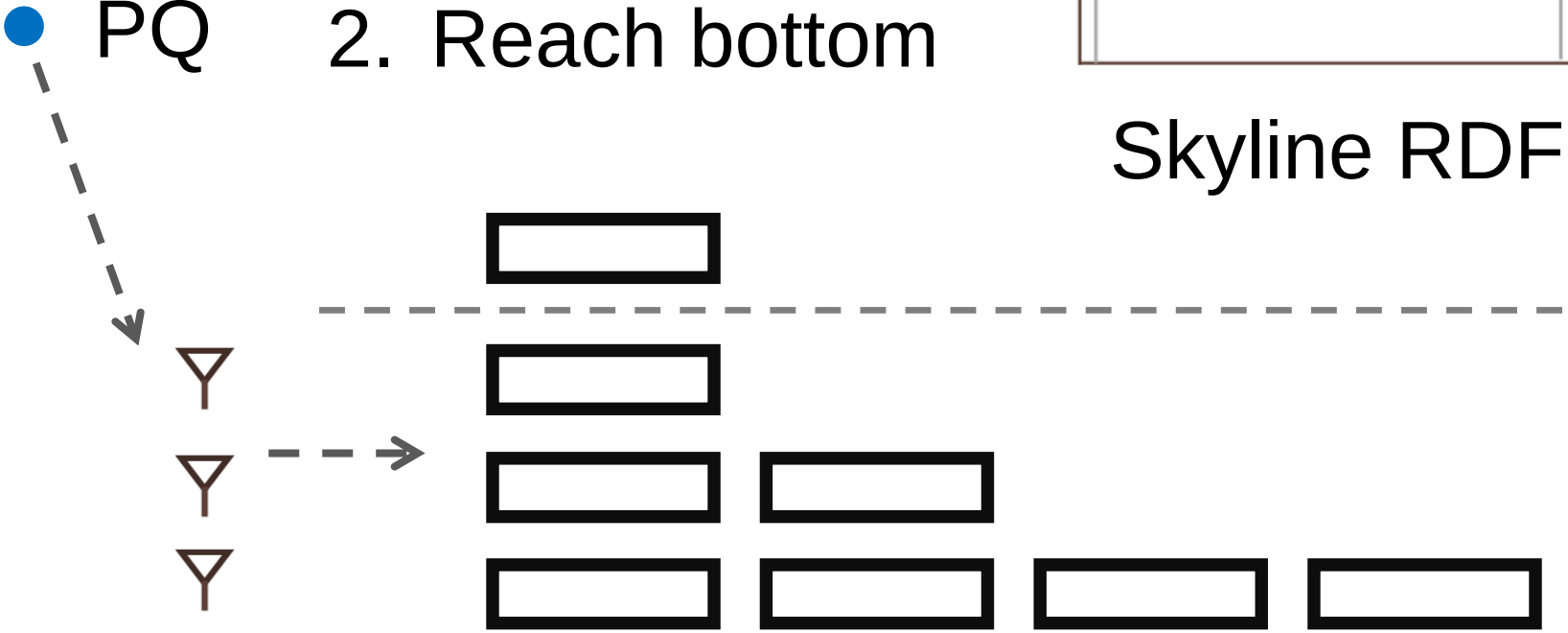
Stop @

1. Entry found
2. Reach bottom

PQ



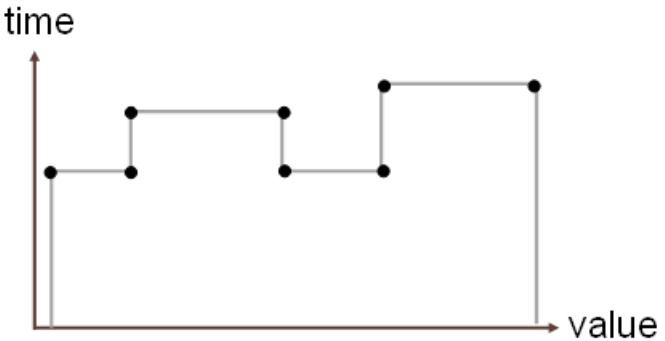

Skyline RDF



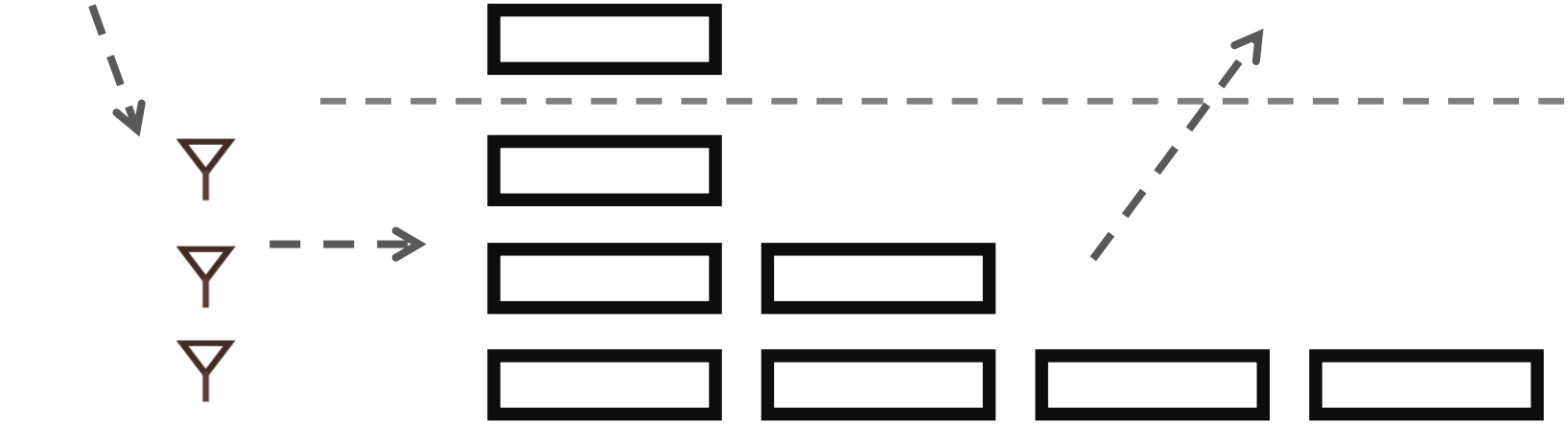
Stop @

1. Entry found
2. Reach buttom

PQ



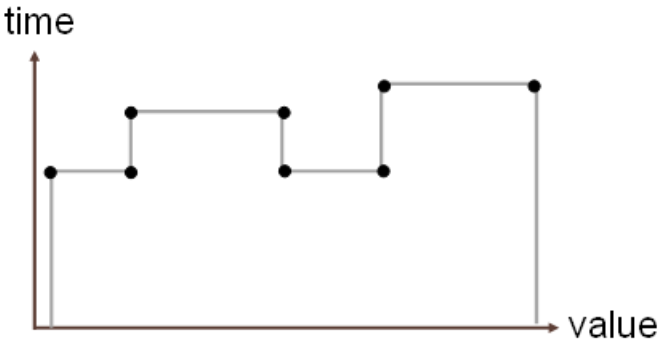

Skyline RDF



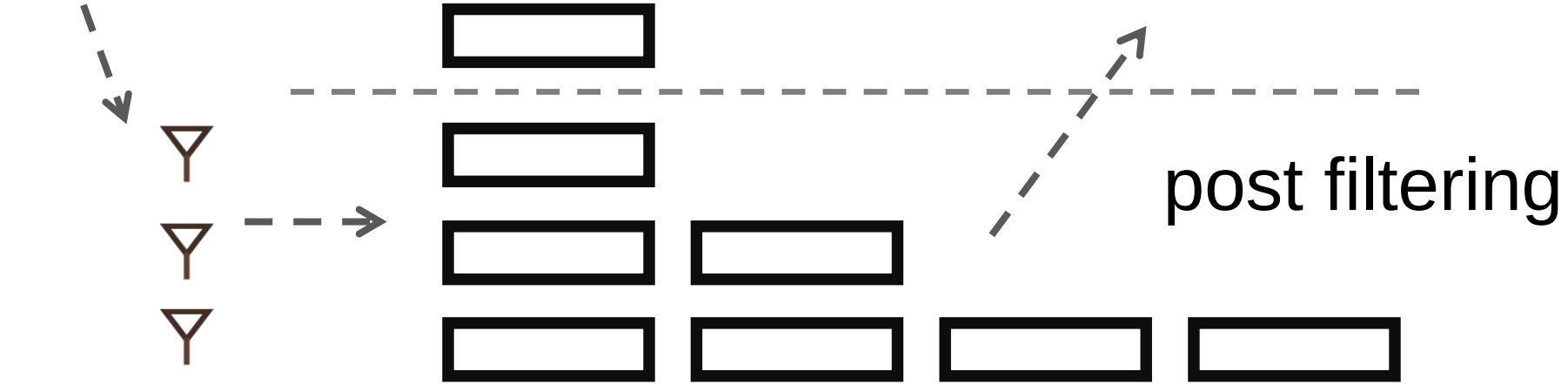
Stop @

1. Entry found
2. Reach bottom

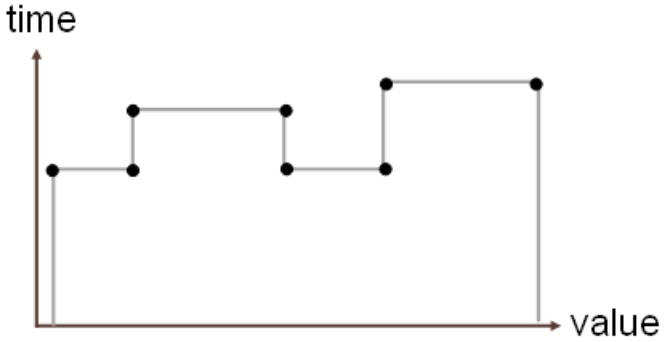
PQ




Skyline RDF

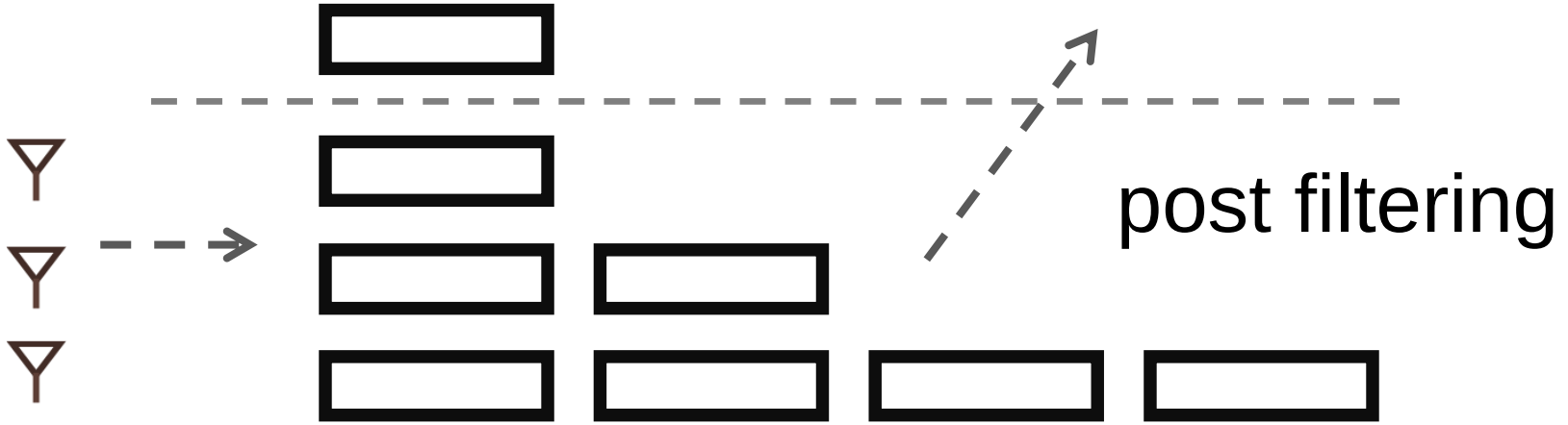


Extra slow disk access



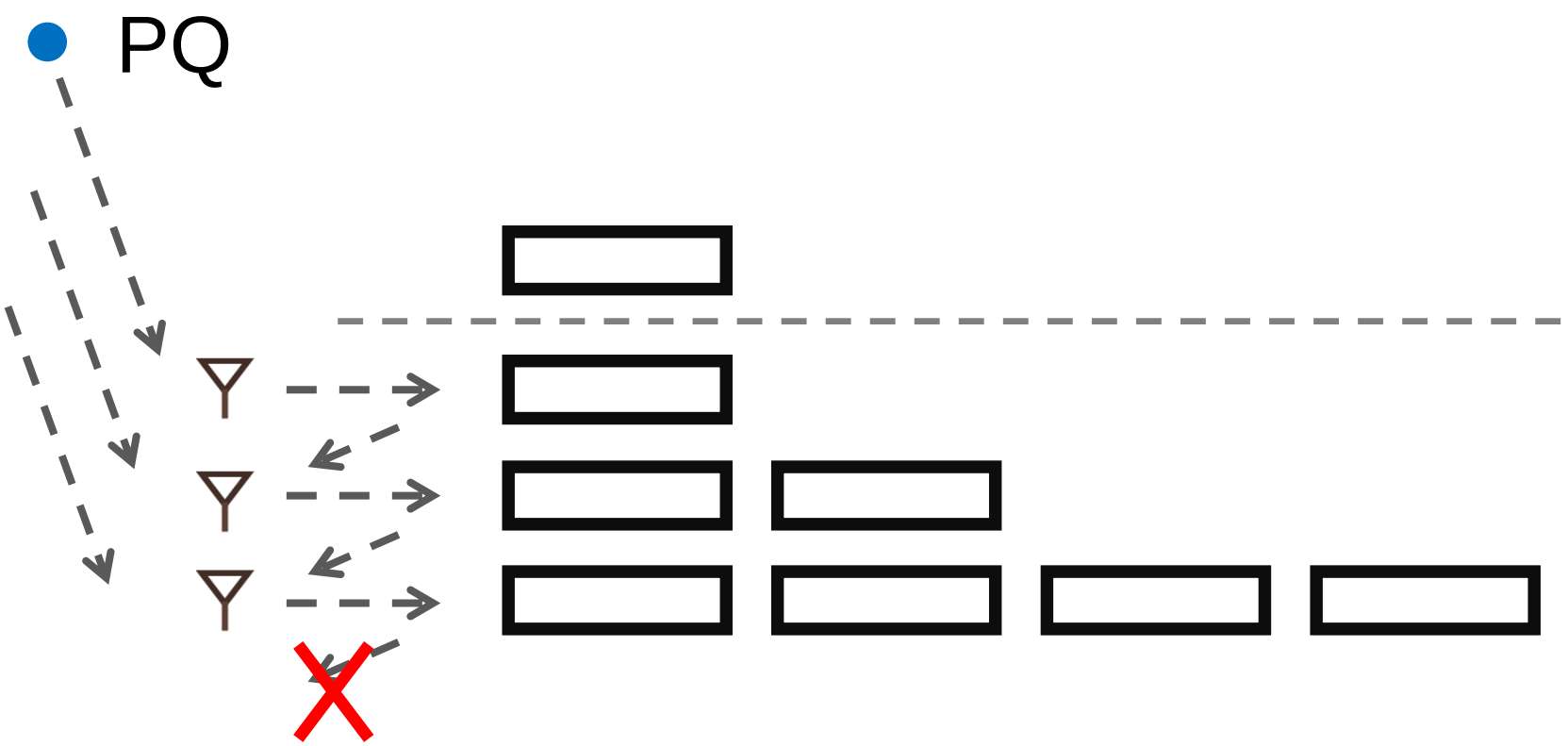

- PQ

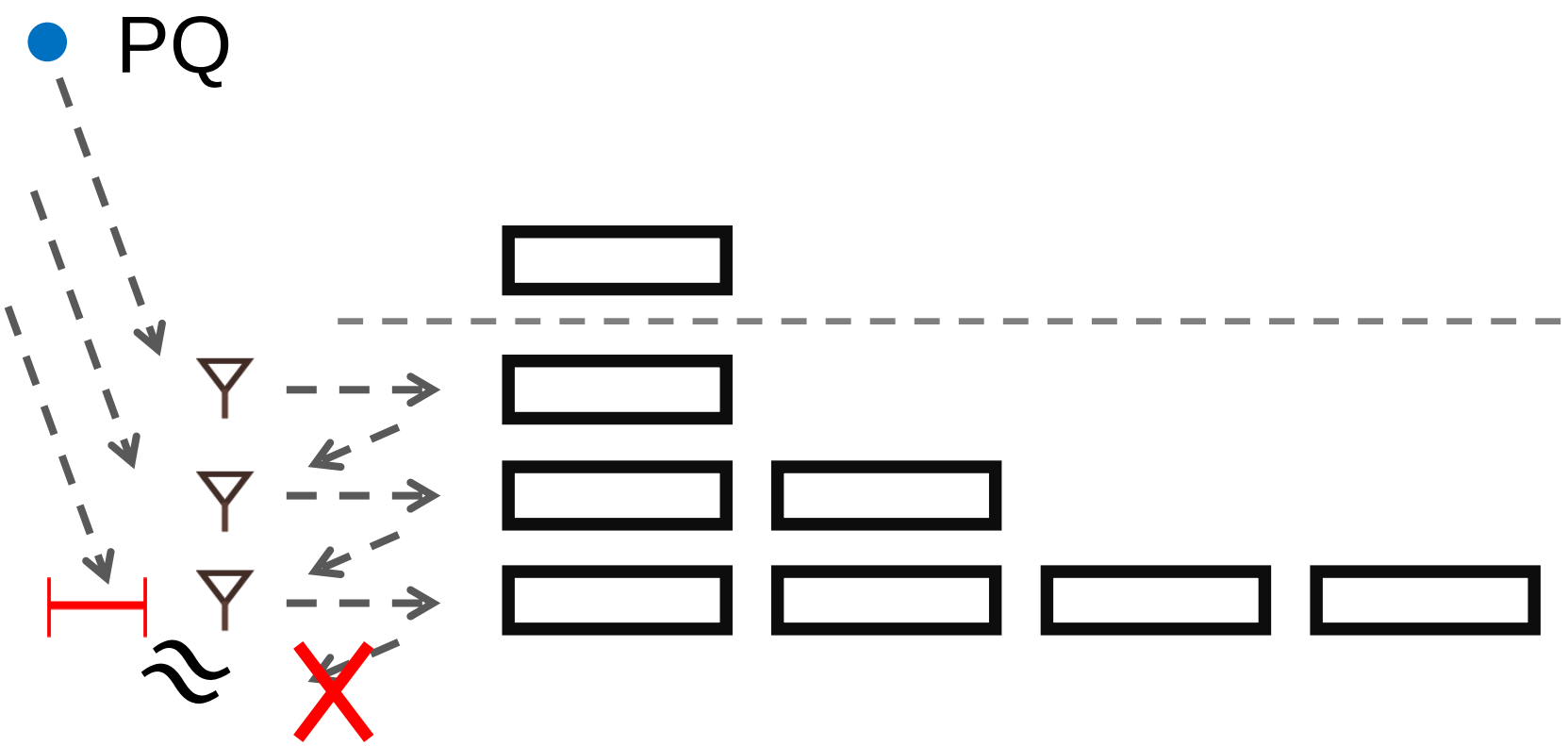
Skyline RDF



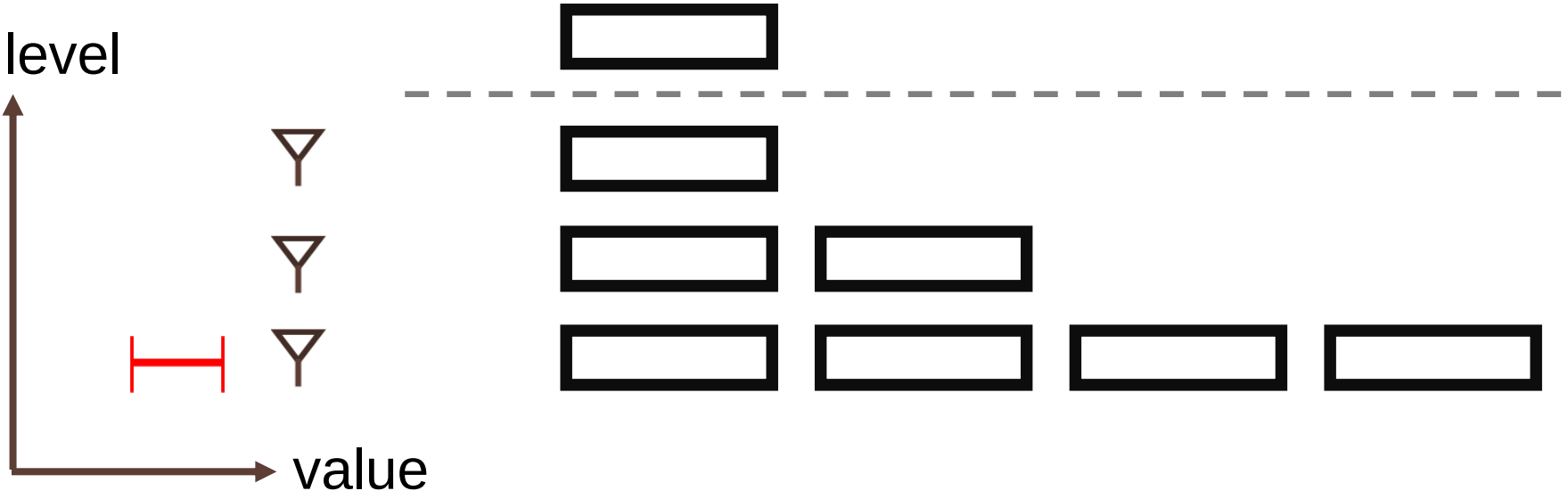
post filtering

# What can we do?

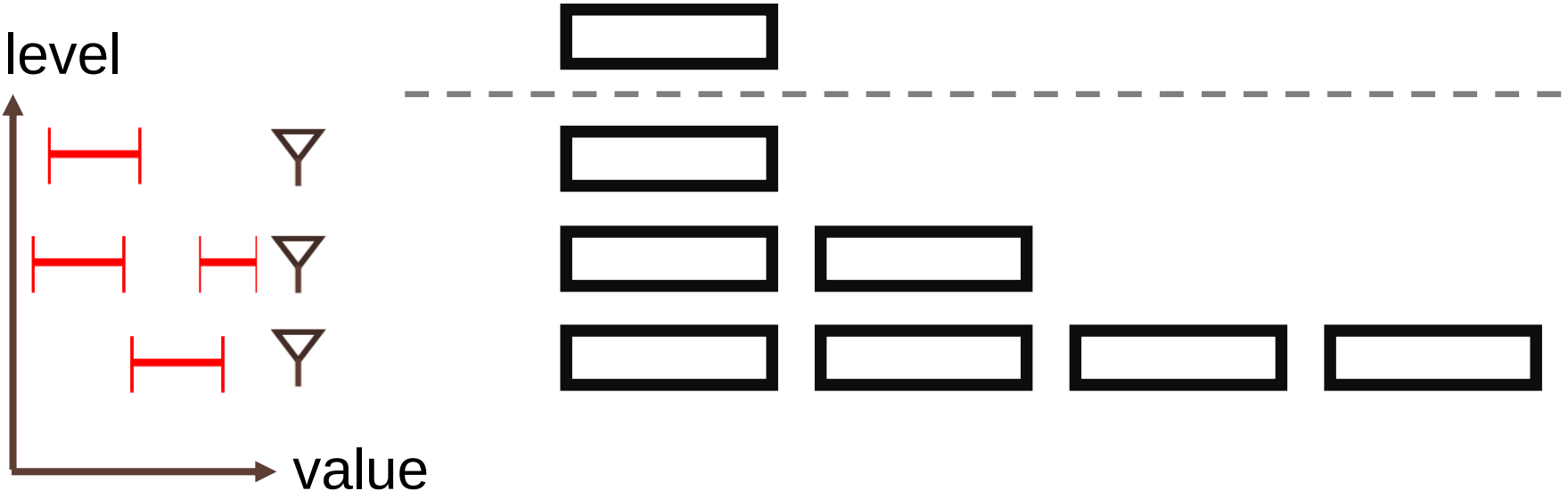

PQ

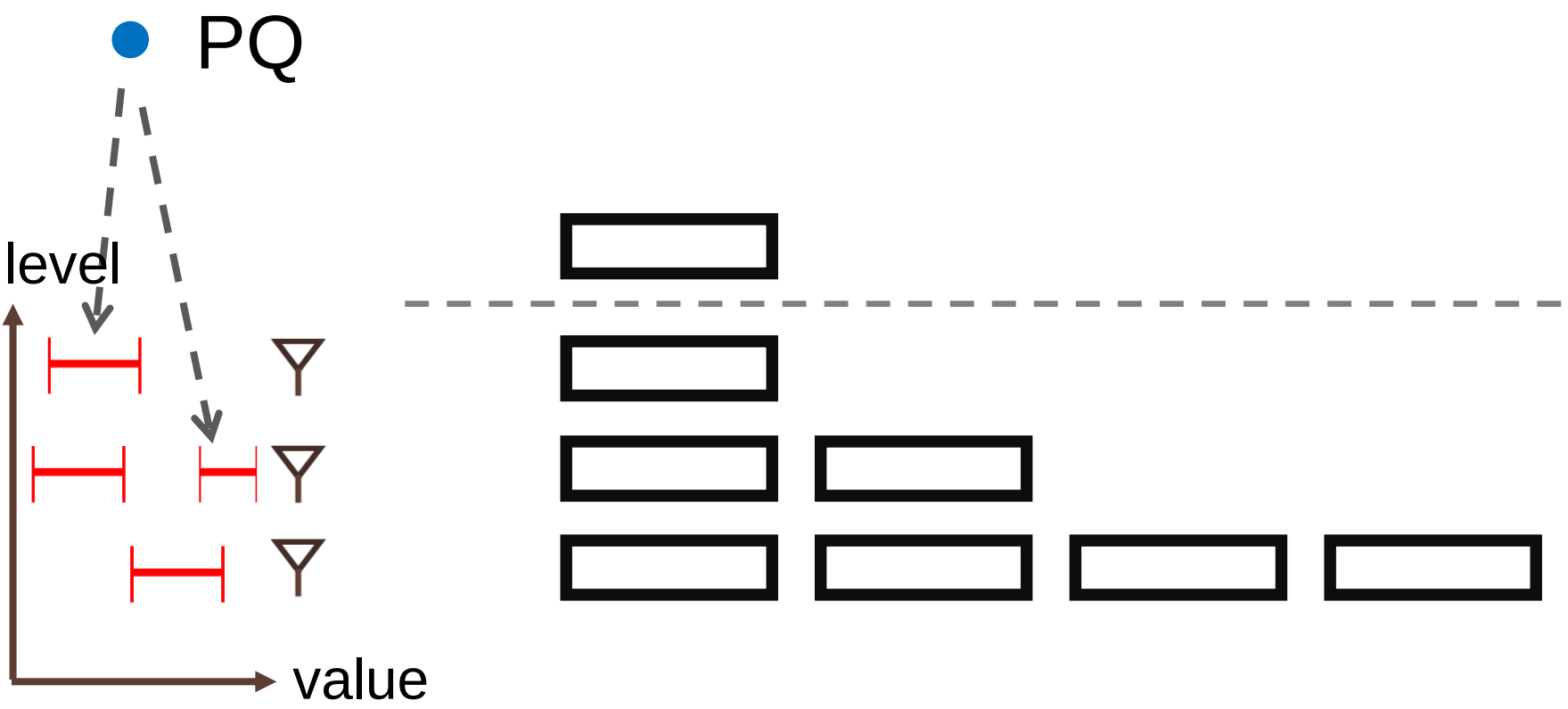

PQ

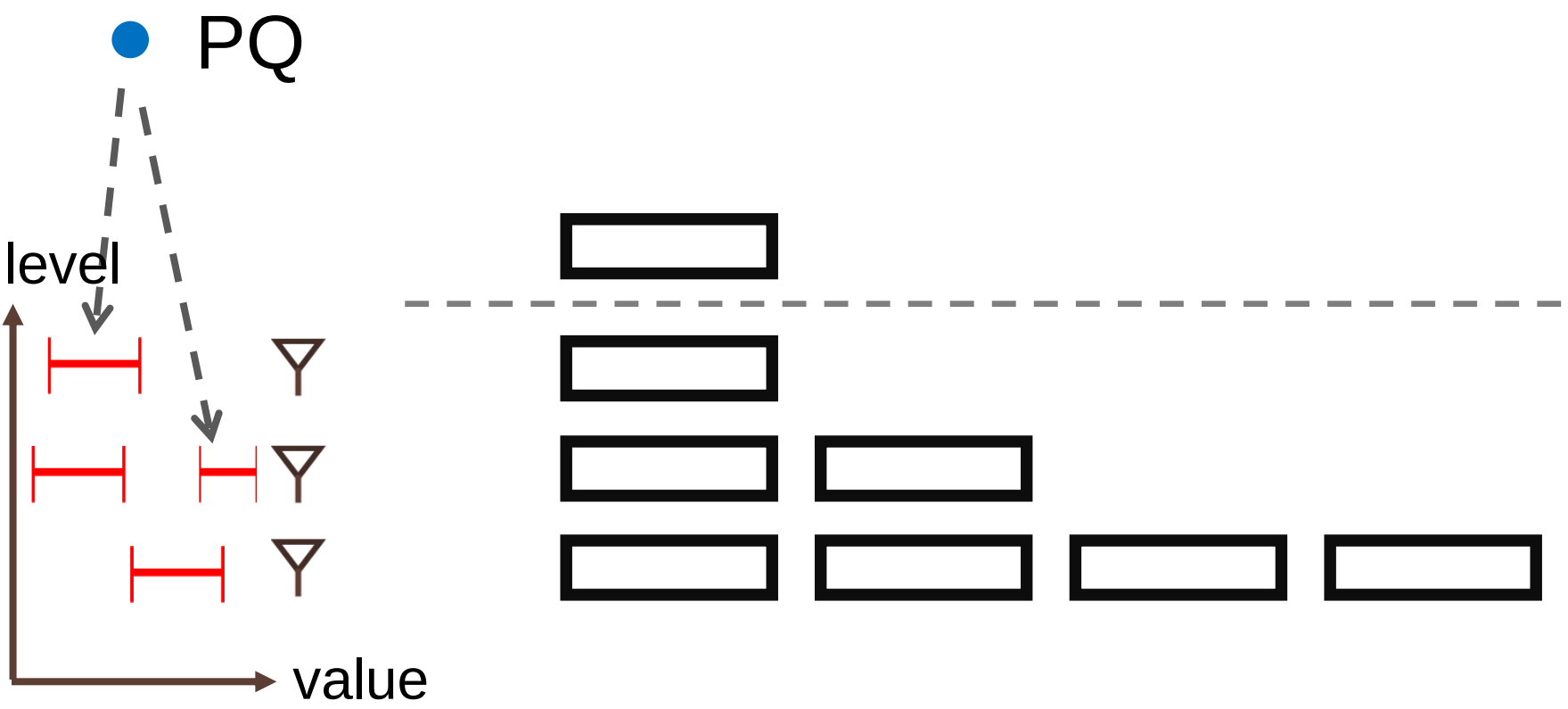
- PQ



- PQ




PQ
level
value




pre filtering
Perlevel RDF

# Our method

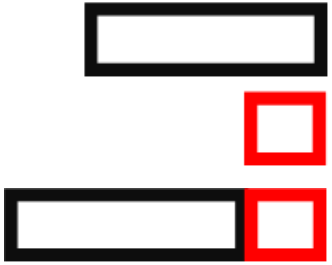
# Perlevel RDF

@ compaction

$level_i$

$level_{i+1}$

# Perlevel RDF



point inserts

range deletes

a SST file

@ compaction

$level_i$

$level_{i+1}$

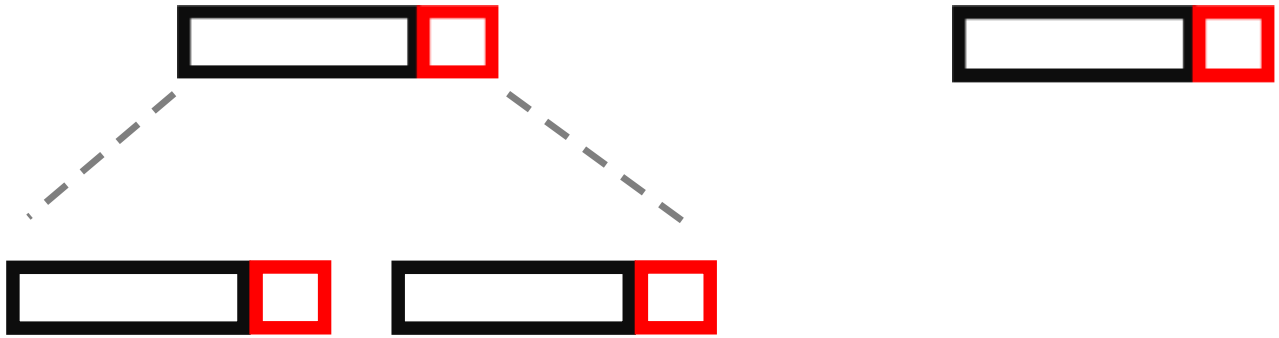
# Perlevel RDF

1. Find compaction target

@ compaction

$level_i$

$level_{i+1}$



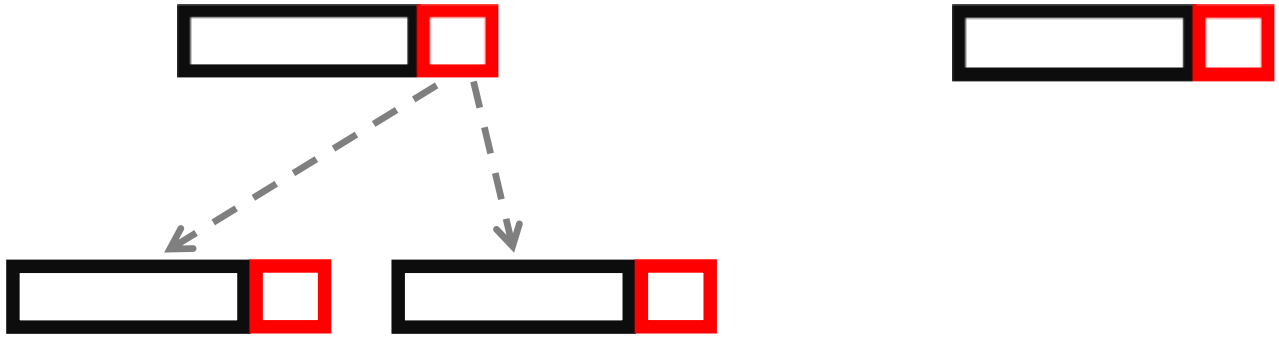
# Perlevel RDF

2. Propagate delete information

@ compaction

$level_i$

$level_{i+1}$



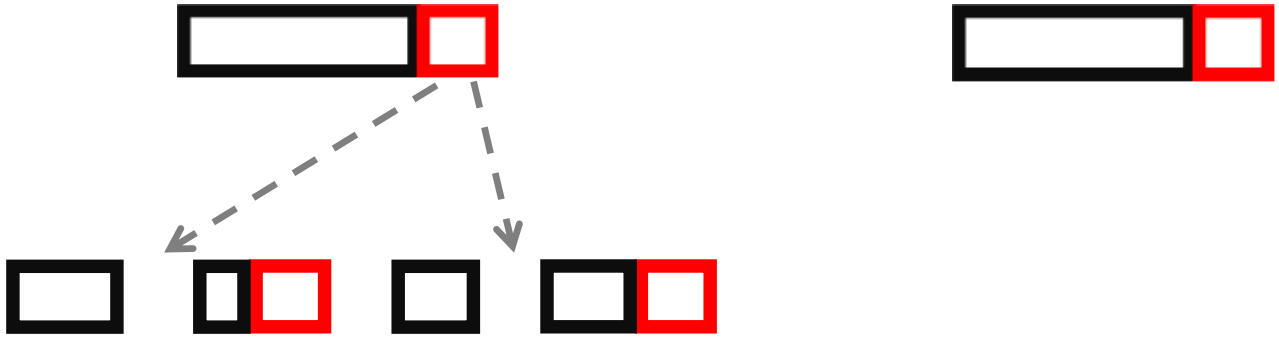
# Perlevel RDF

2. Propagate delete information

@ compaction

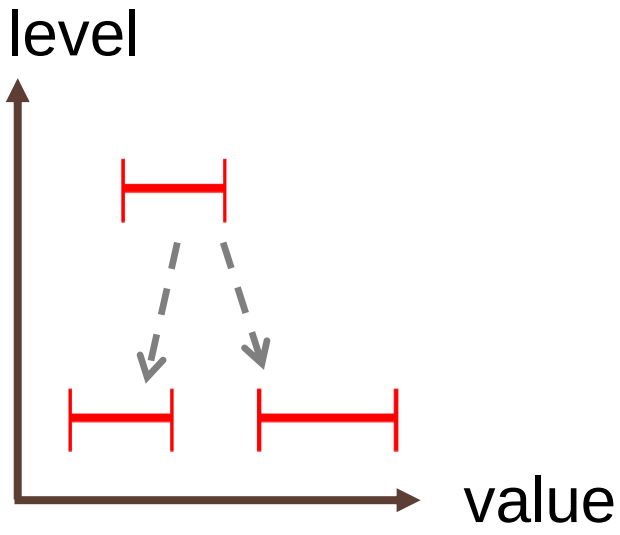
$level_i$

$level_{i+1}$



# Perlevel RDF

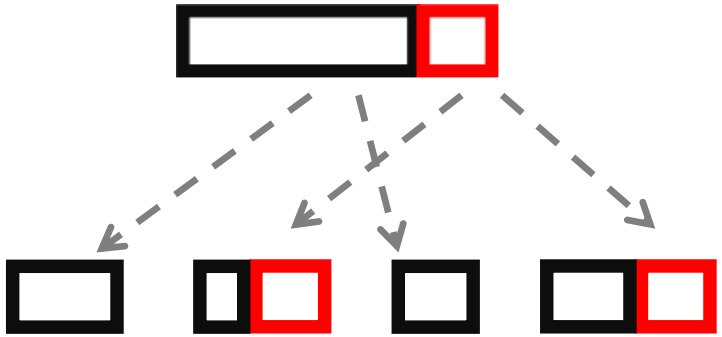
3. Merge




@ compaction

$level_i$

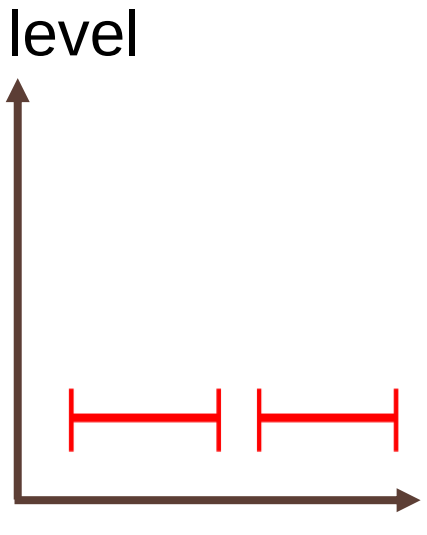
$level_{i+1}$



# Perlevel RDF

3. Merge

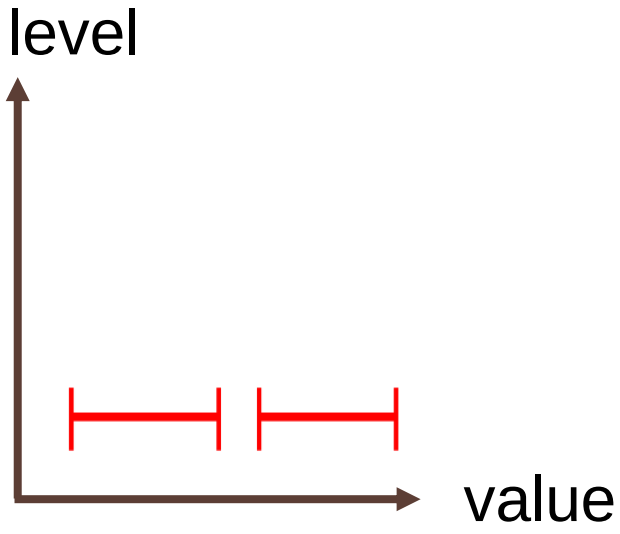
@ compaction




$level_i$

$level_{i+1}$

# Perlevel RDF

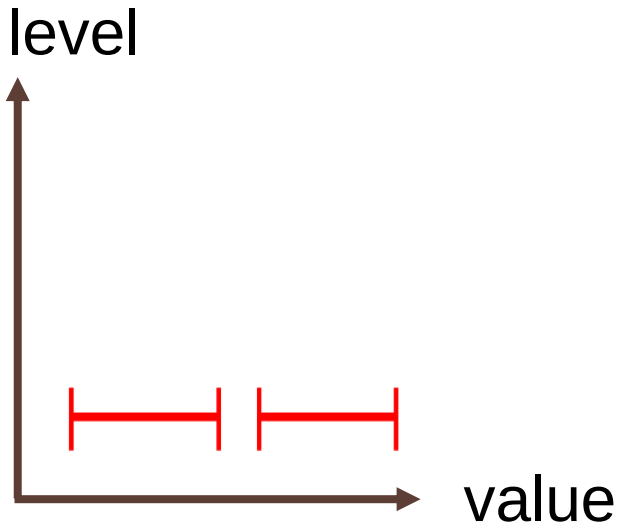



$level_i$

$level_{i+1}$

(a, b, **time**)

# Perlevel RDF




$level_i$

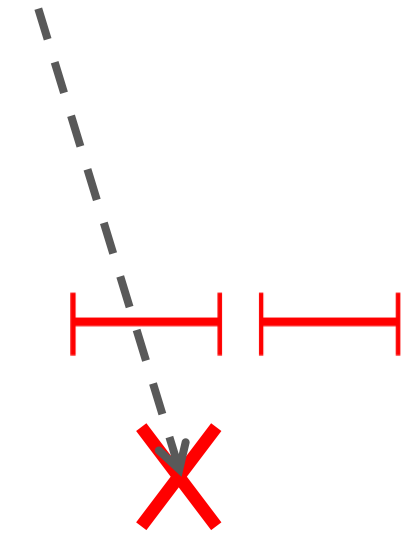
$level_{i+1}$

(a, b, ~~time~~)

But

# Perlevel RDF

- PQ

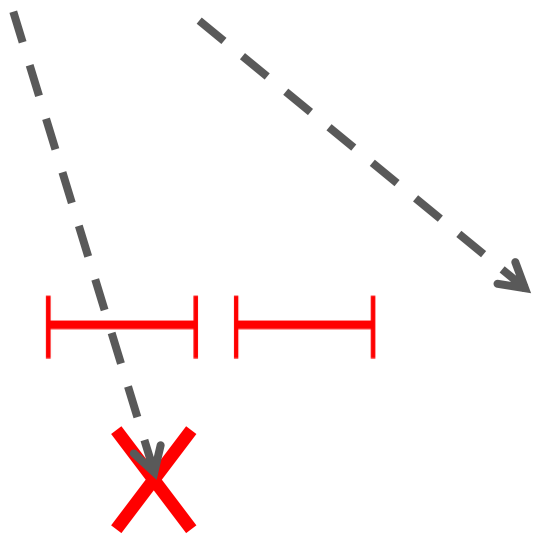


$level_i$

$level_{i+1}$

# Perlevel RDF

- PQ



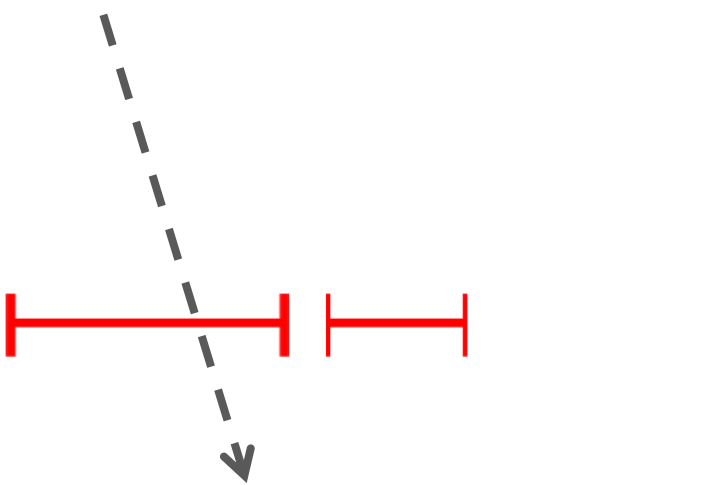
$level_i$

$level_{i+1}$

# Enhancement

# Perlevel RDF split

- Insert

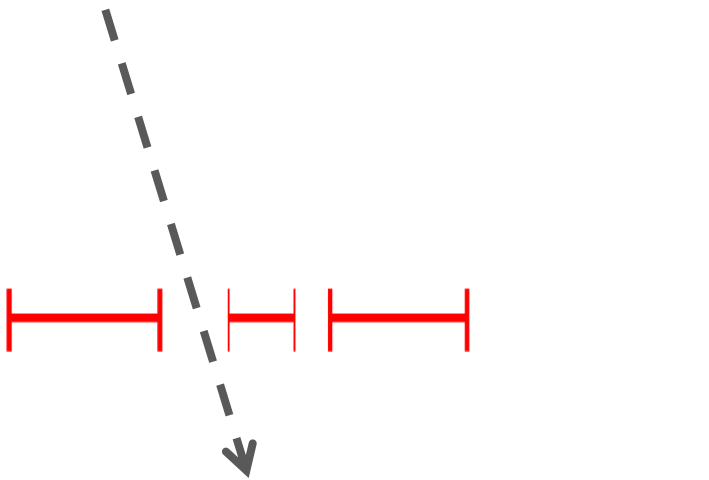


$level_i$

$level_{i+1}$

# Perlevel RDF split

- Insert

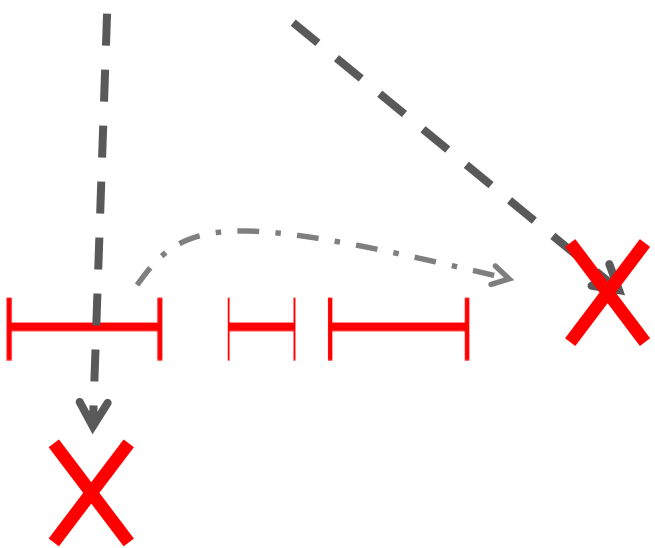


$level_i$

$level_{i+1}$

# Perlevel RDF split

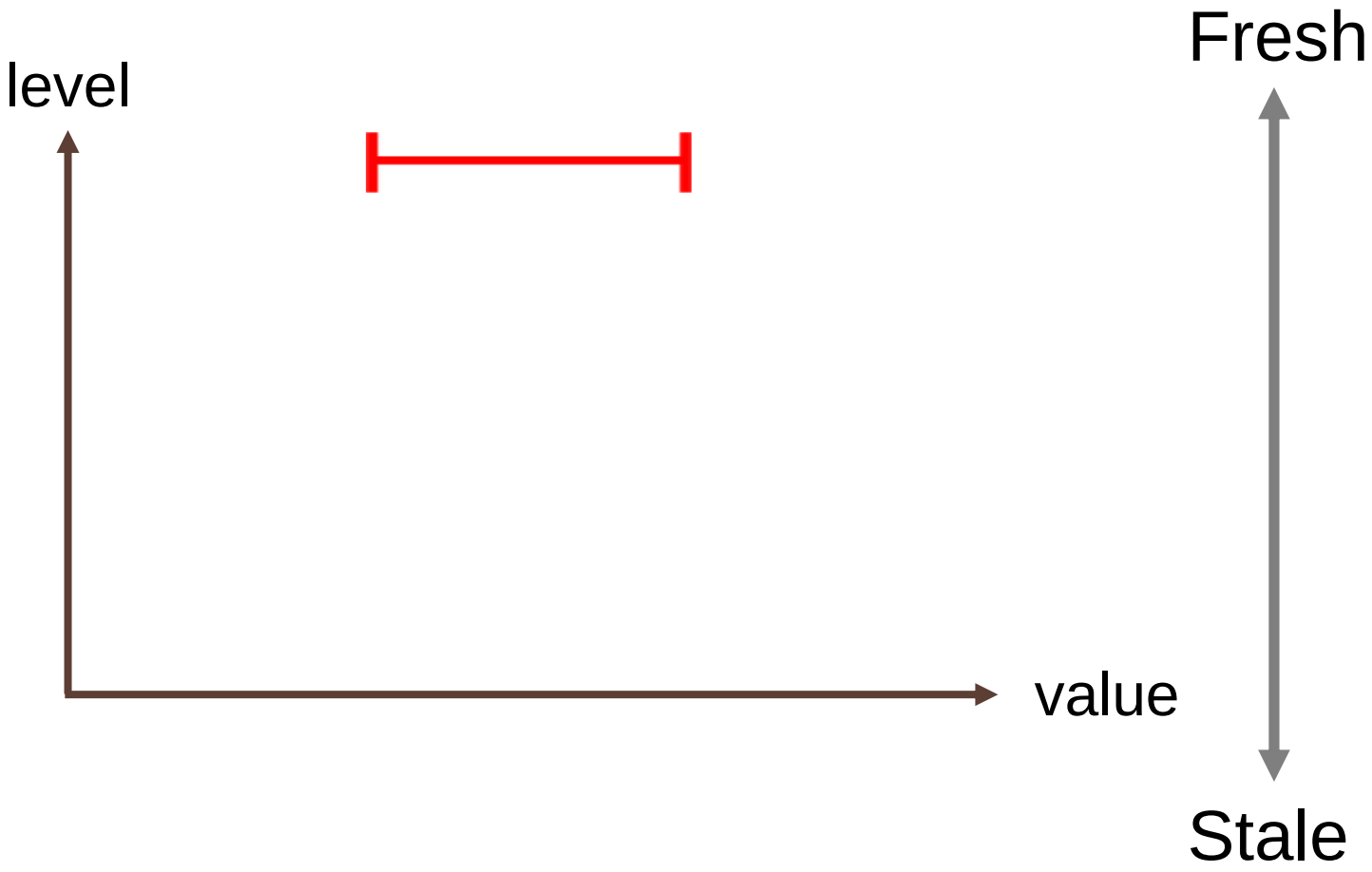
- PQ

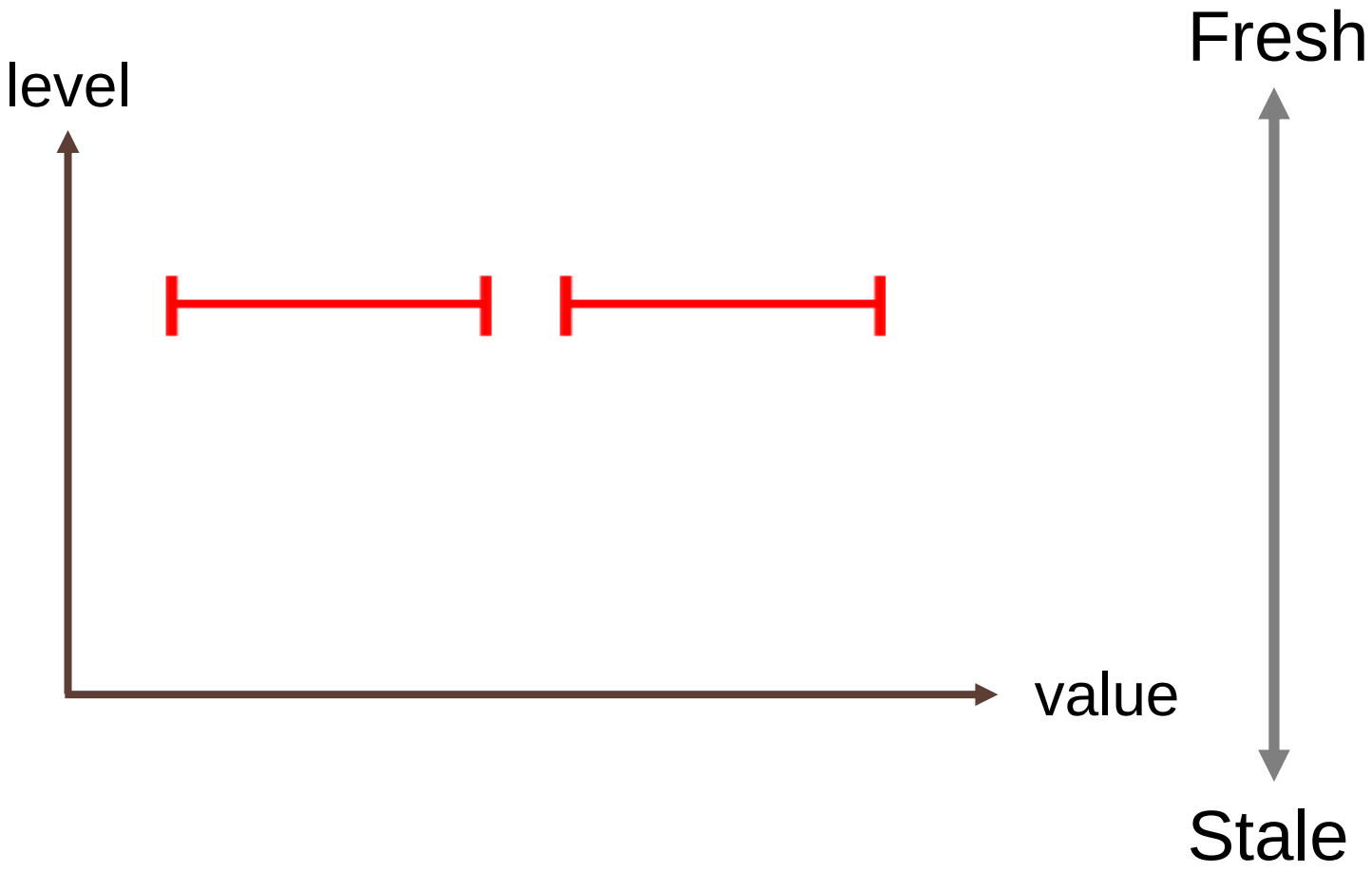


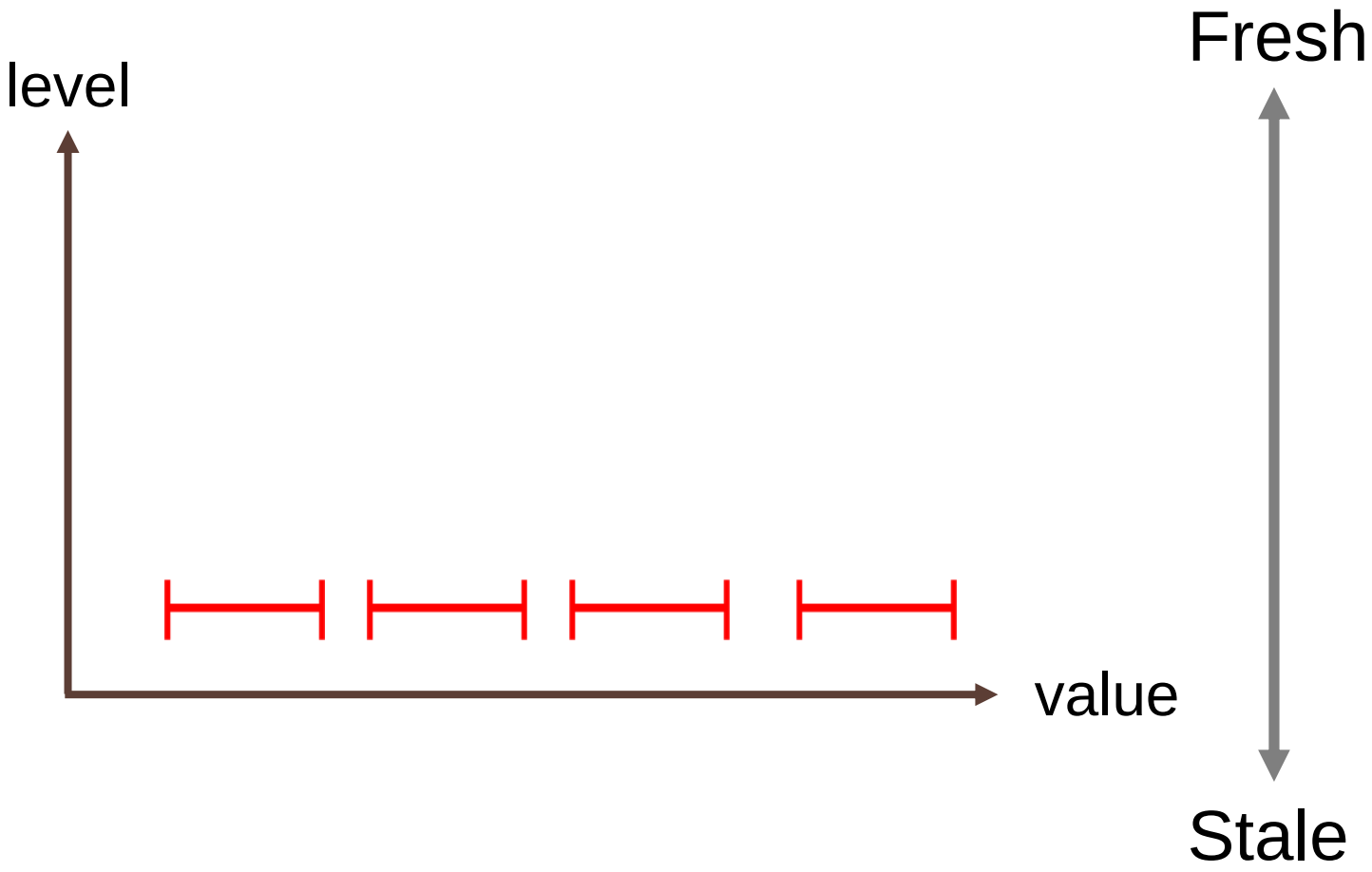
$level_i$

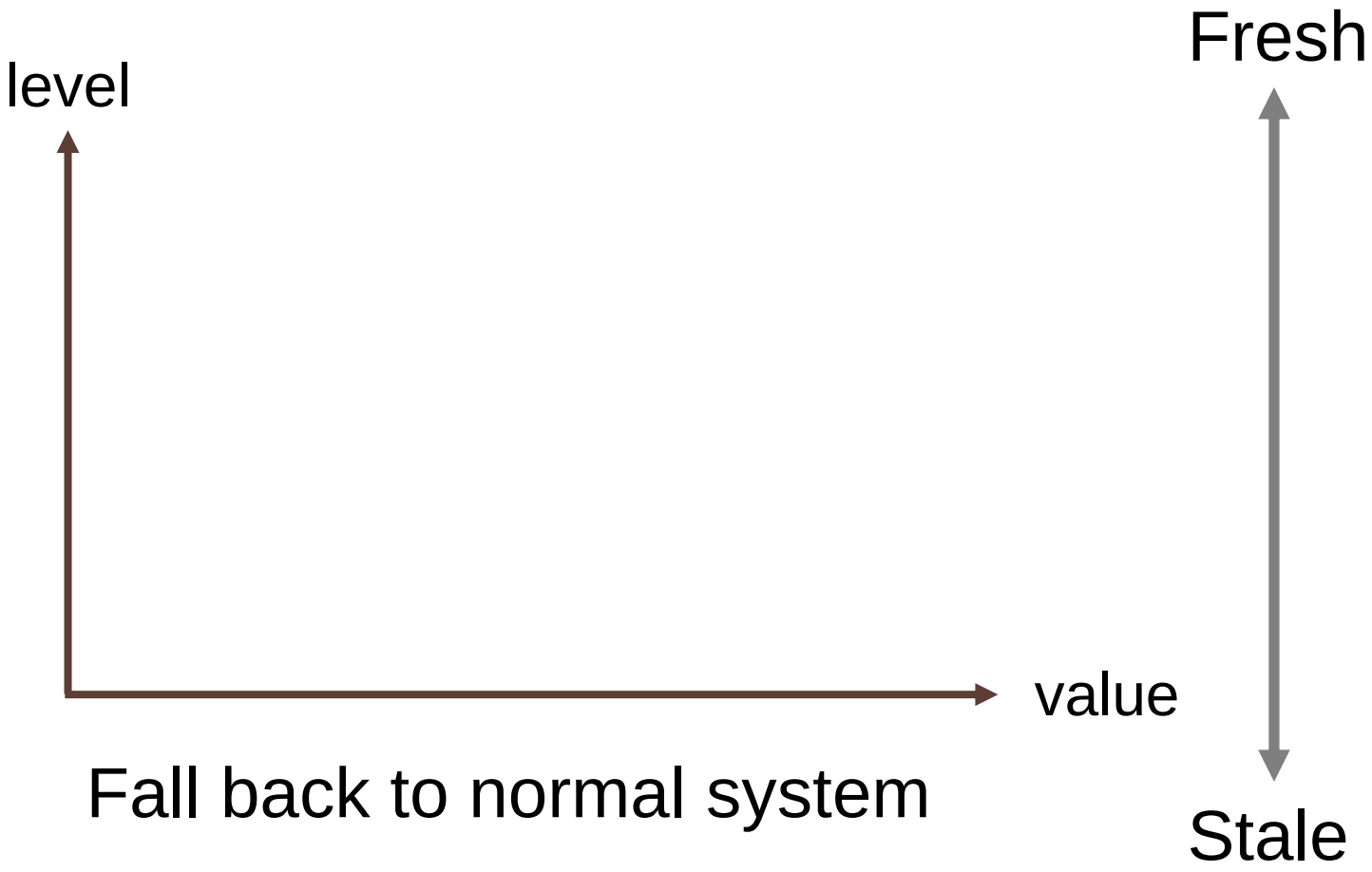
$level_{i+1}$

# As time passed by

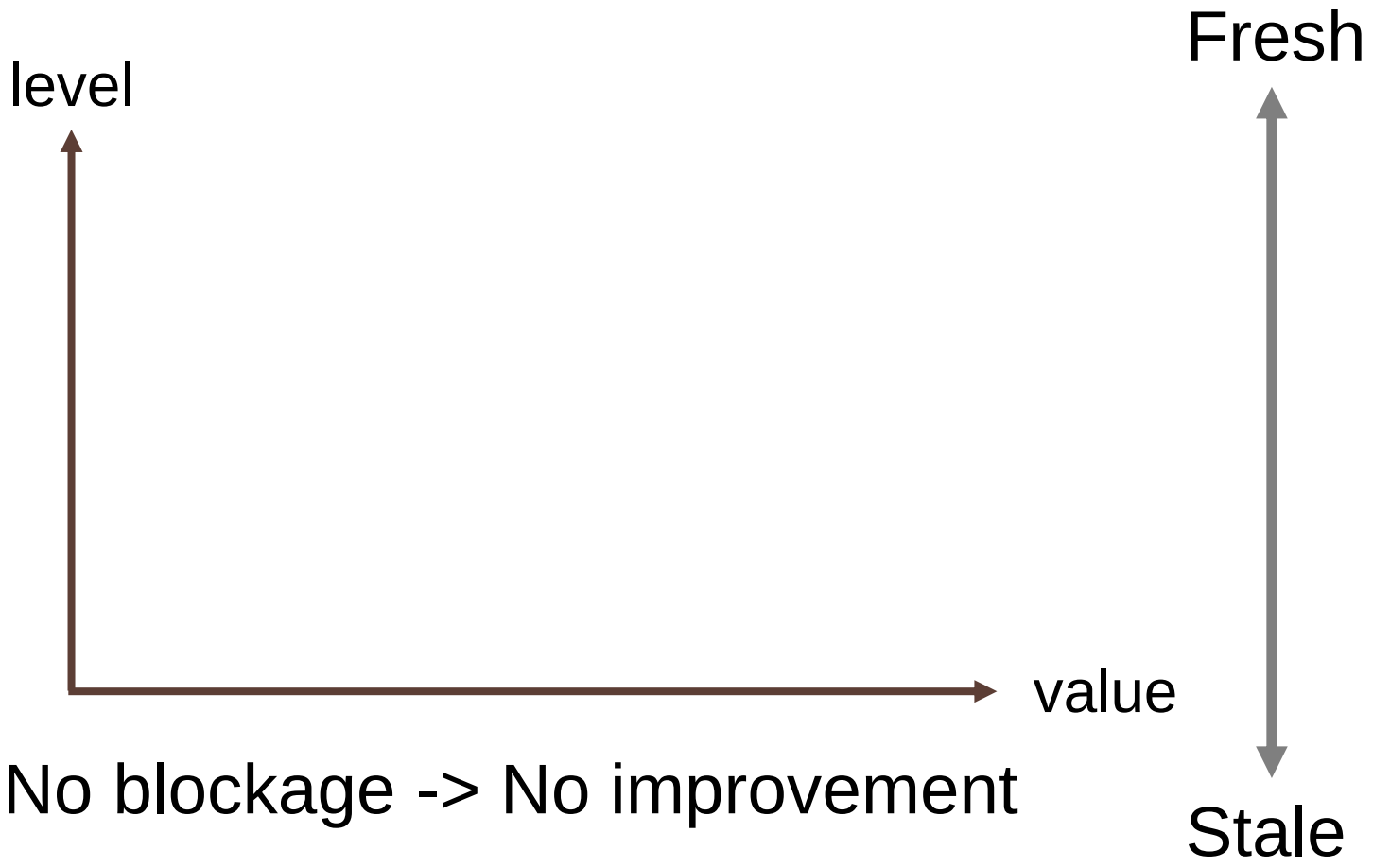

level
value
Fresh
Stale


level
value
Fresh
Stale


level
value
Fresh
Stale



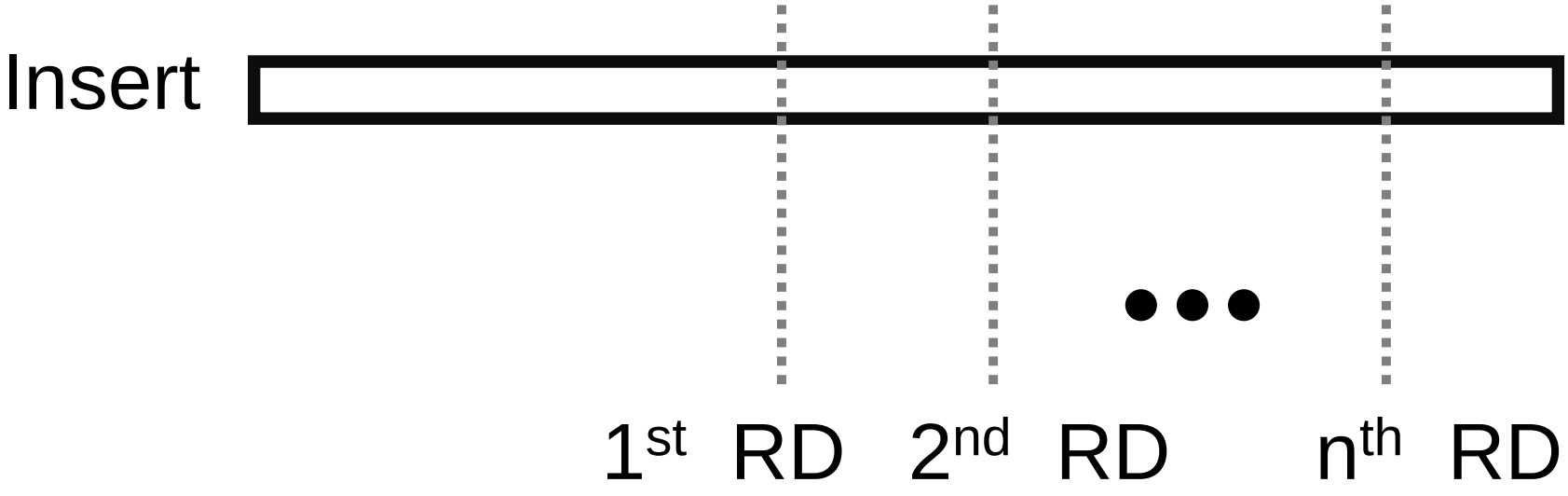

Fall back to normal system



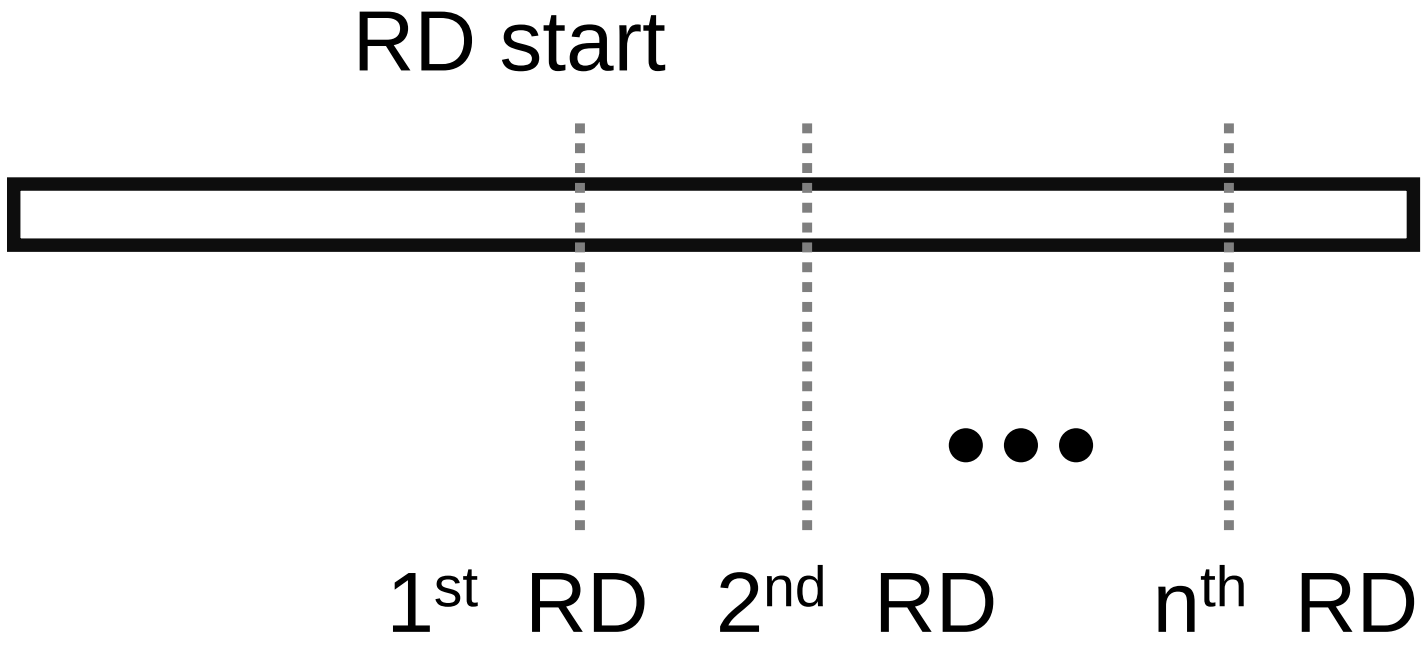

No blockage -> No improvement

# Workload

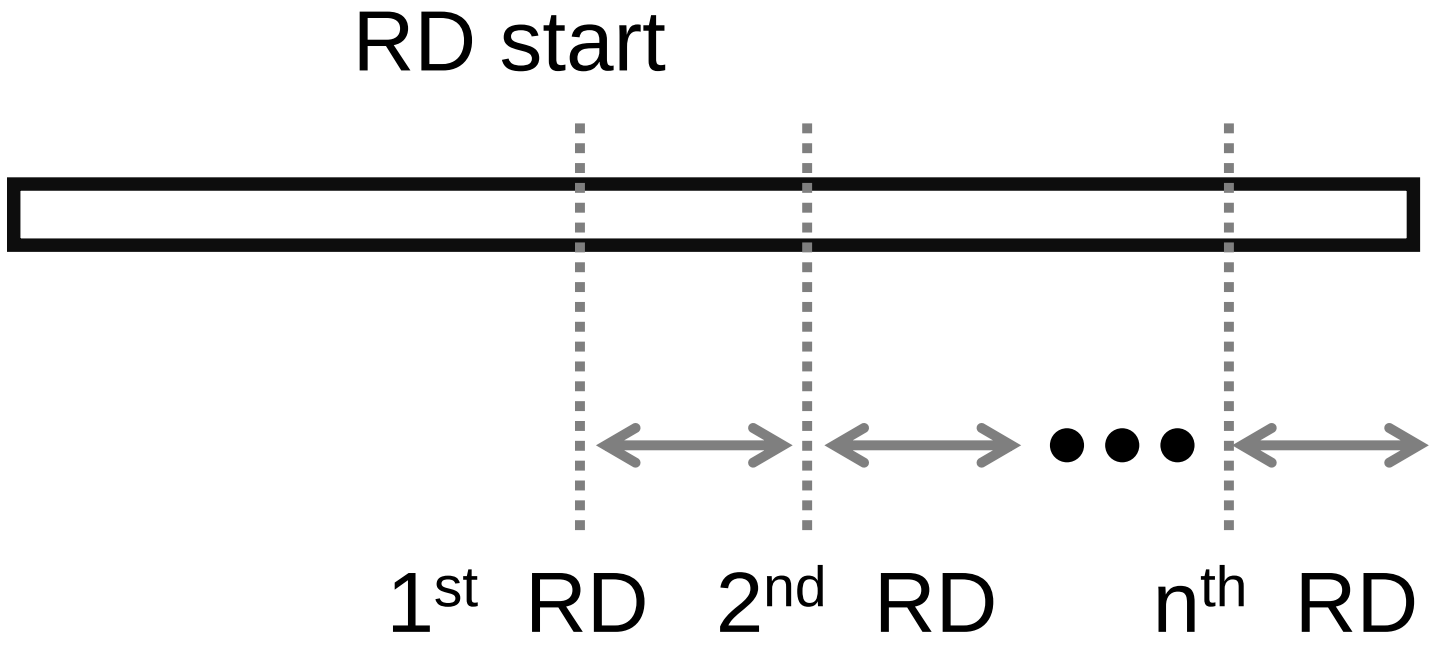
Workload:



Workload:

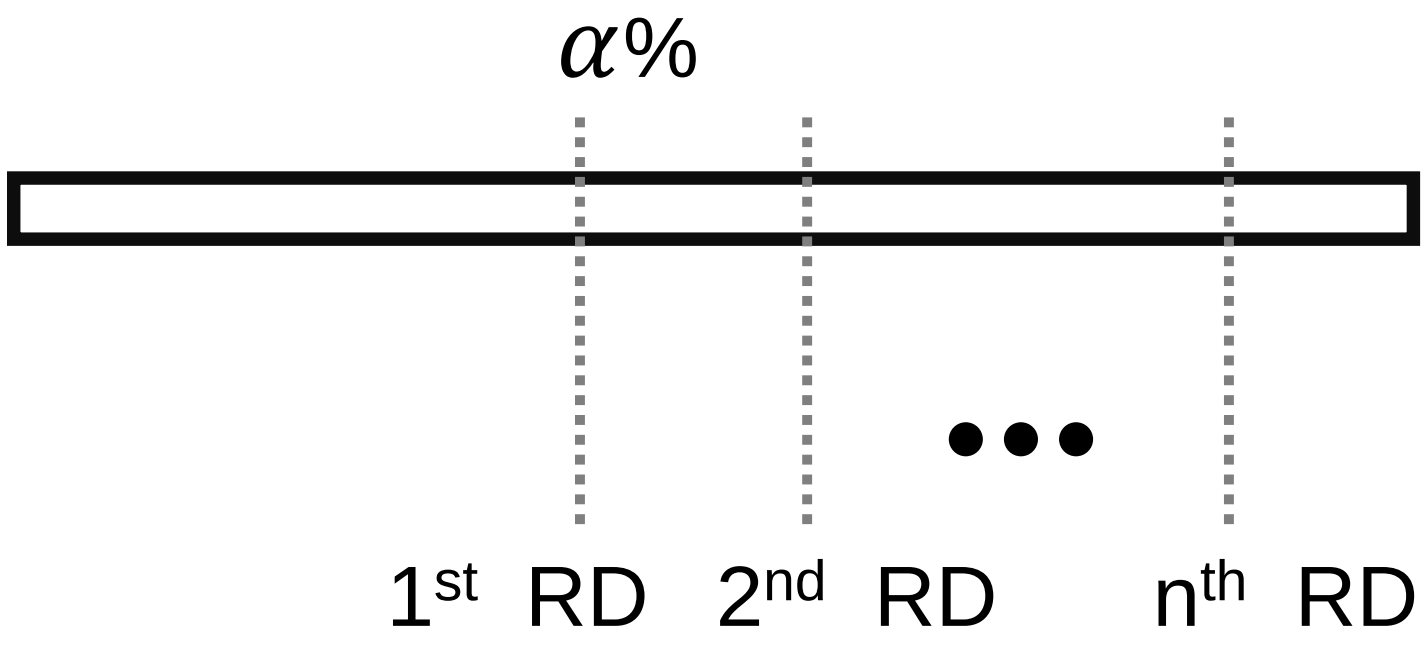


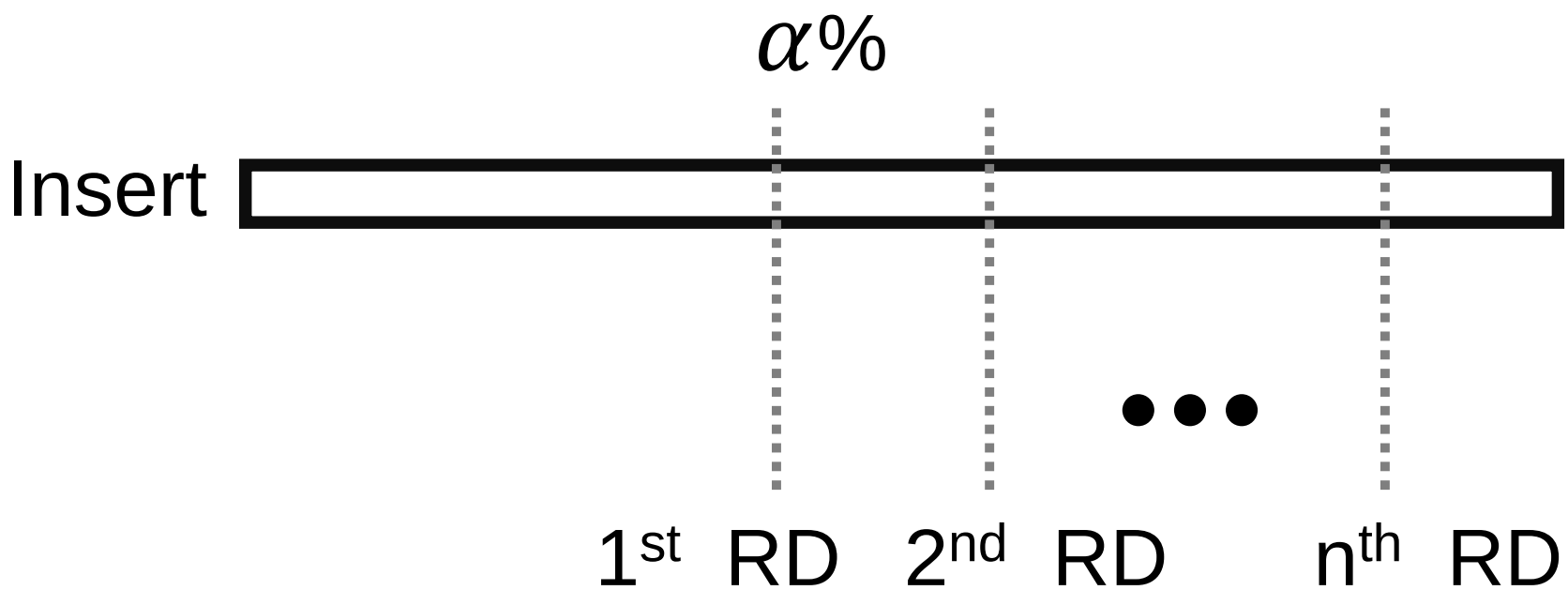
Workload:



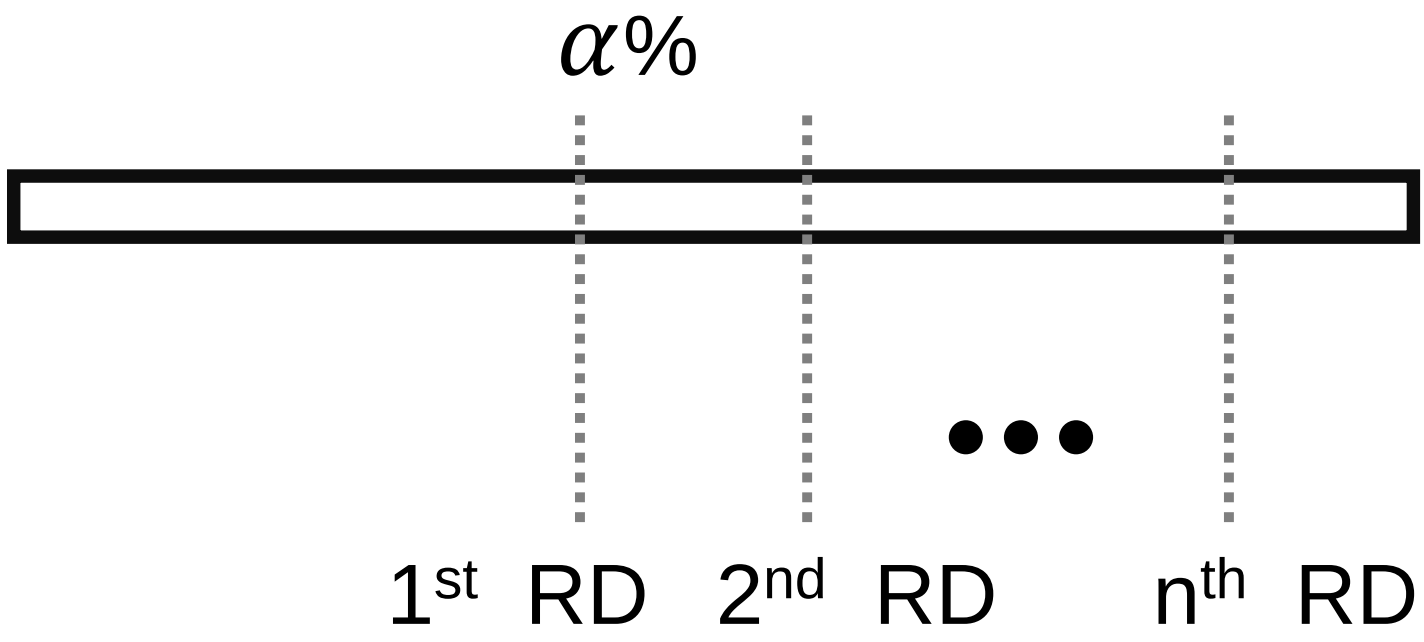
Workload:

Insert




α%
Insert
1st RD
2nd RD
nth RD

Workload:



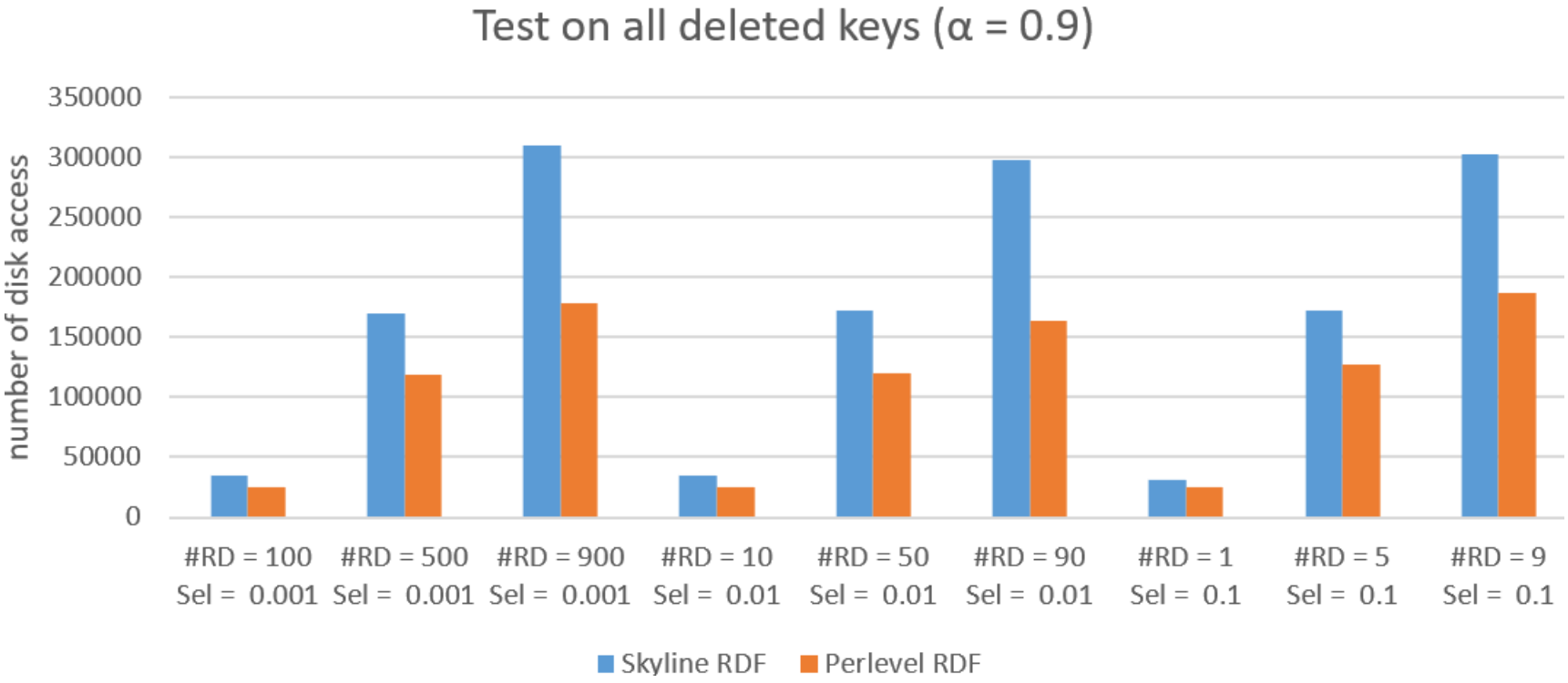

RD selectivity: ratio of deleted Insert

# Experiment settings

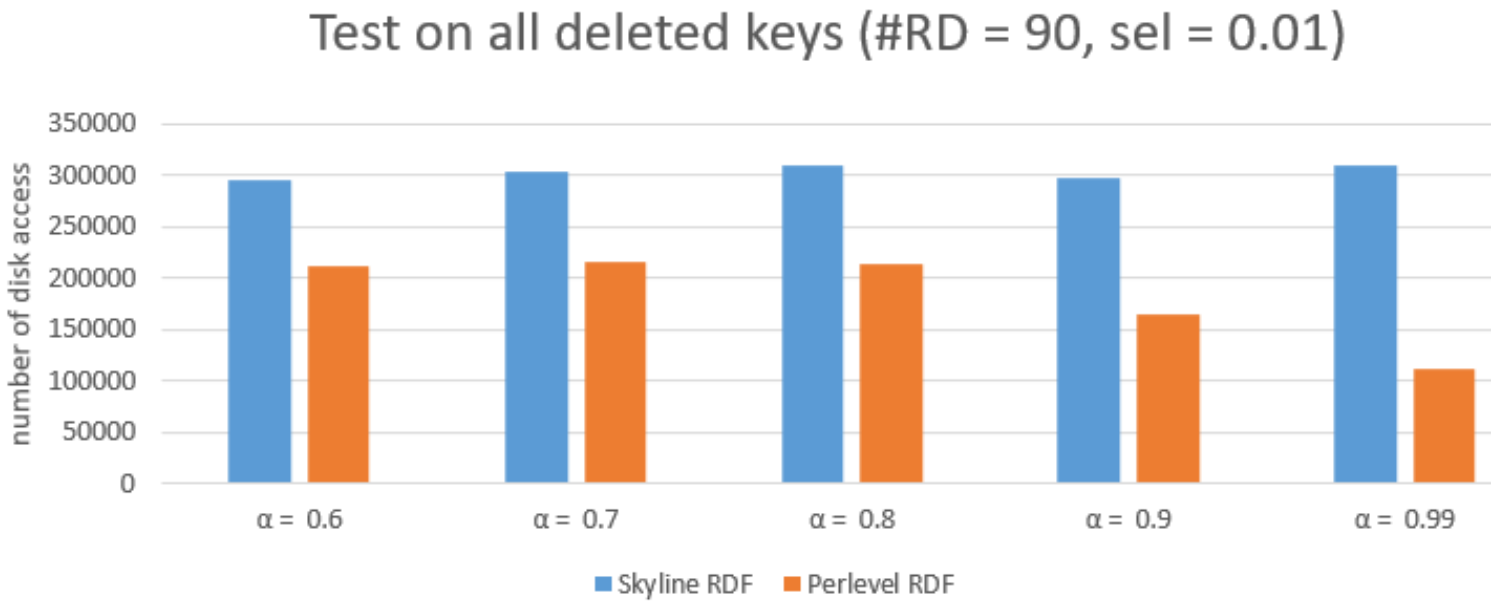
CPU: Intel Core 17-10750H 2.6GHz
Total cores: 6
Total Threads: 12
RAM: 24 GB

# Results

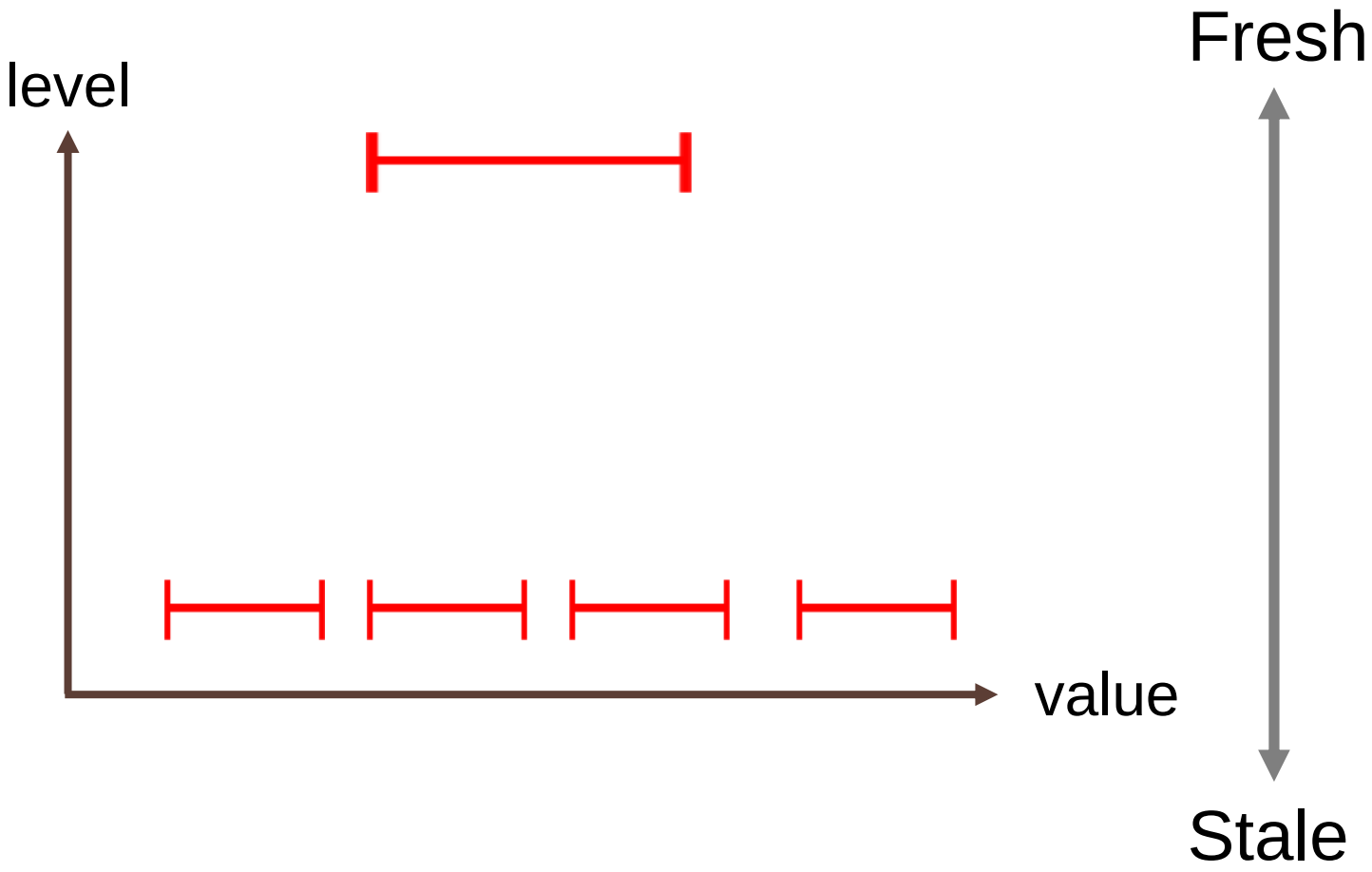
# Query on all deleted keys



# Range delete freshness



# Future work


level
value
Fresh
Stale