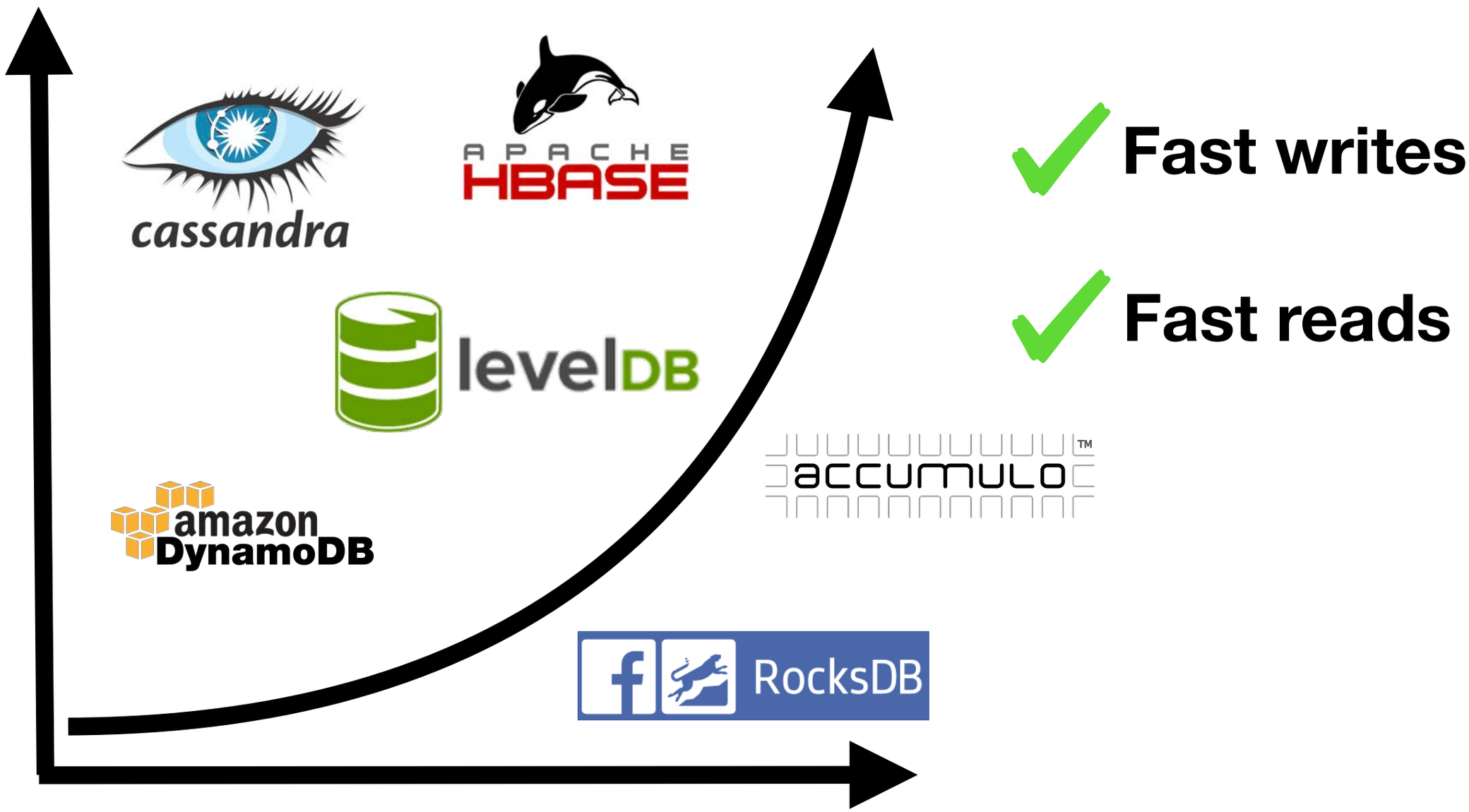
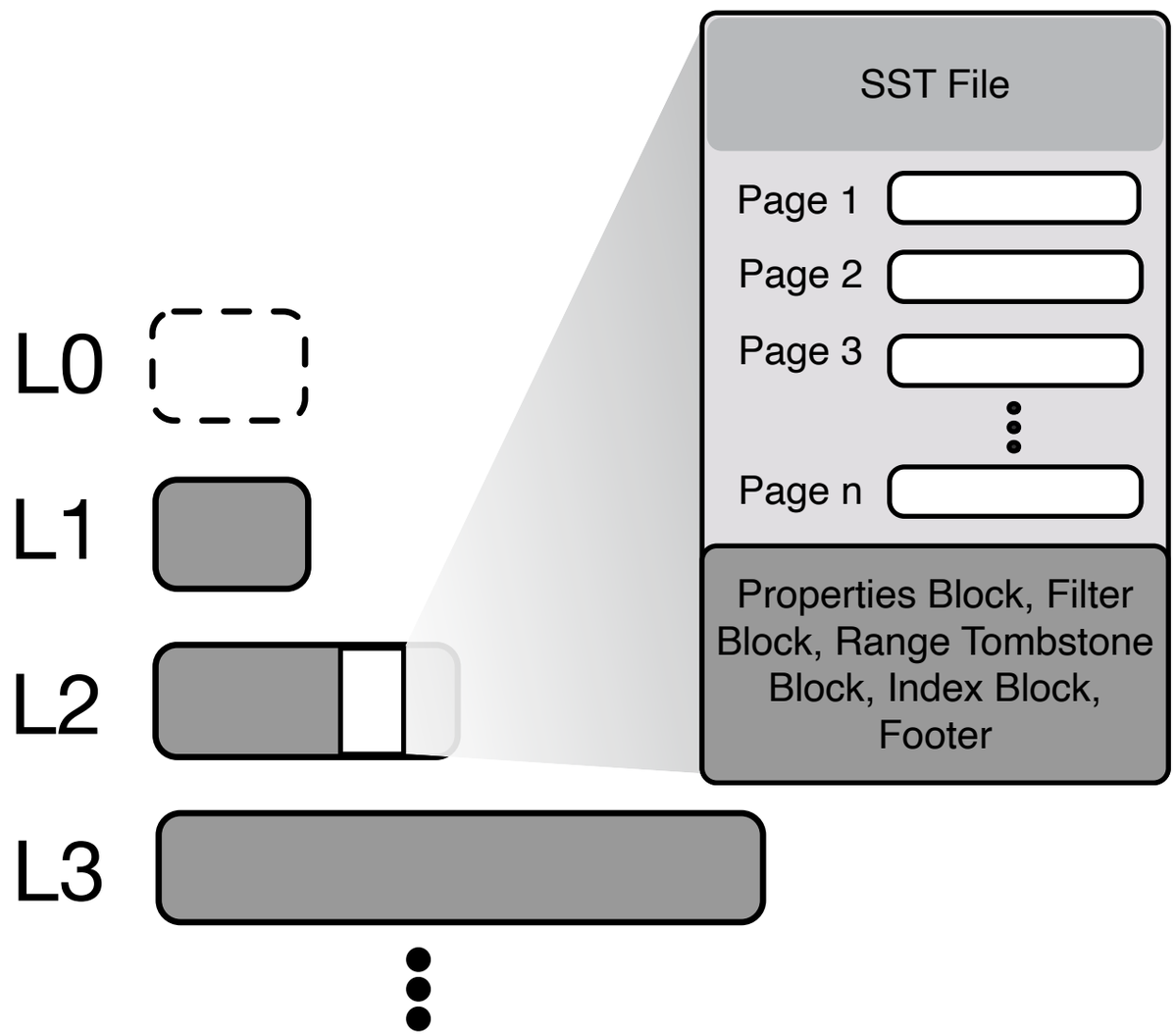
Efficient Deletes in LSM Engines

Dimitris Staratzis

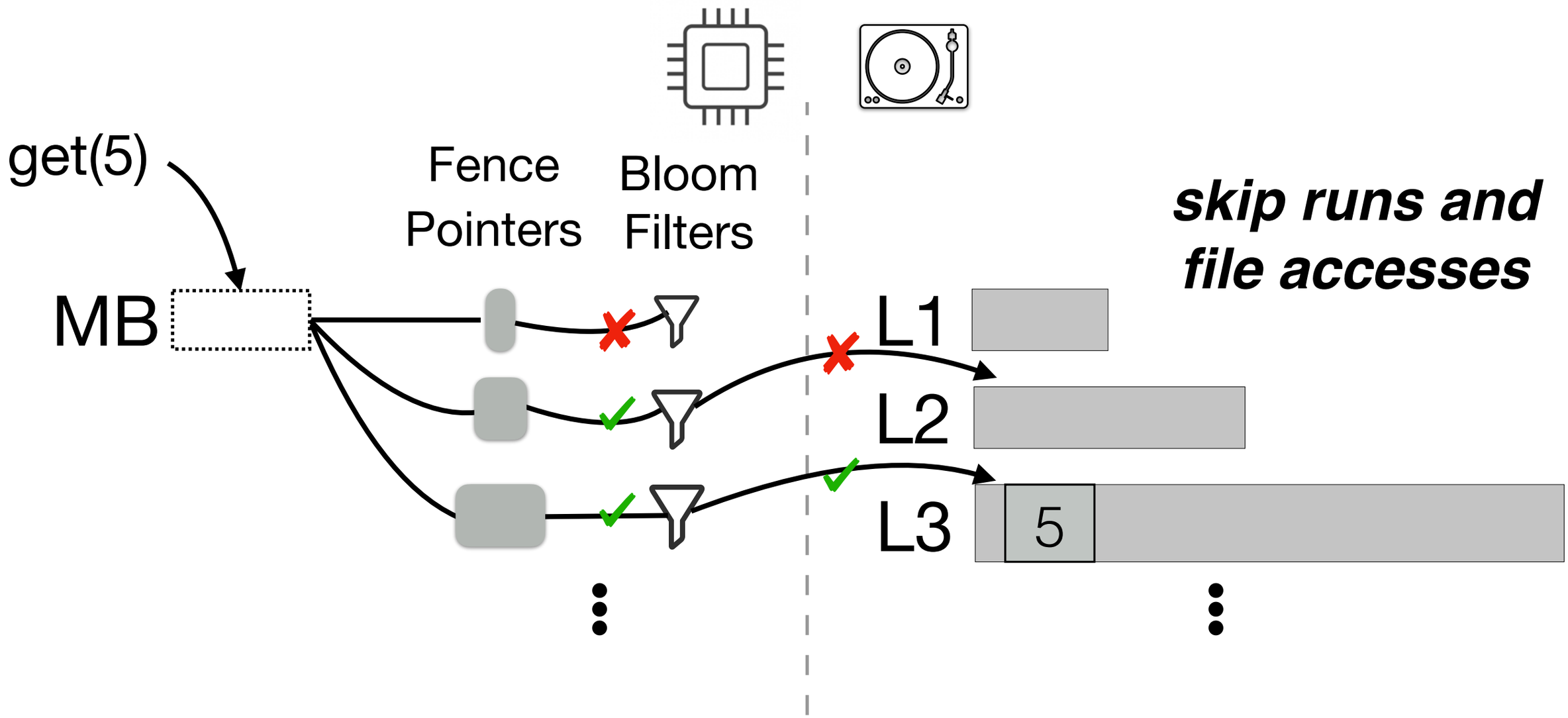
LSM - trees are everywhere



RocksDB structure



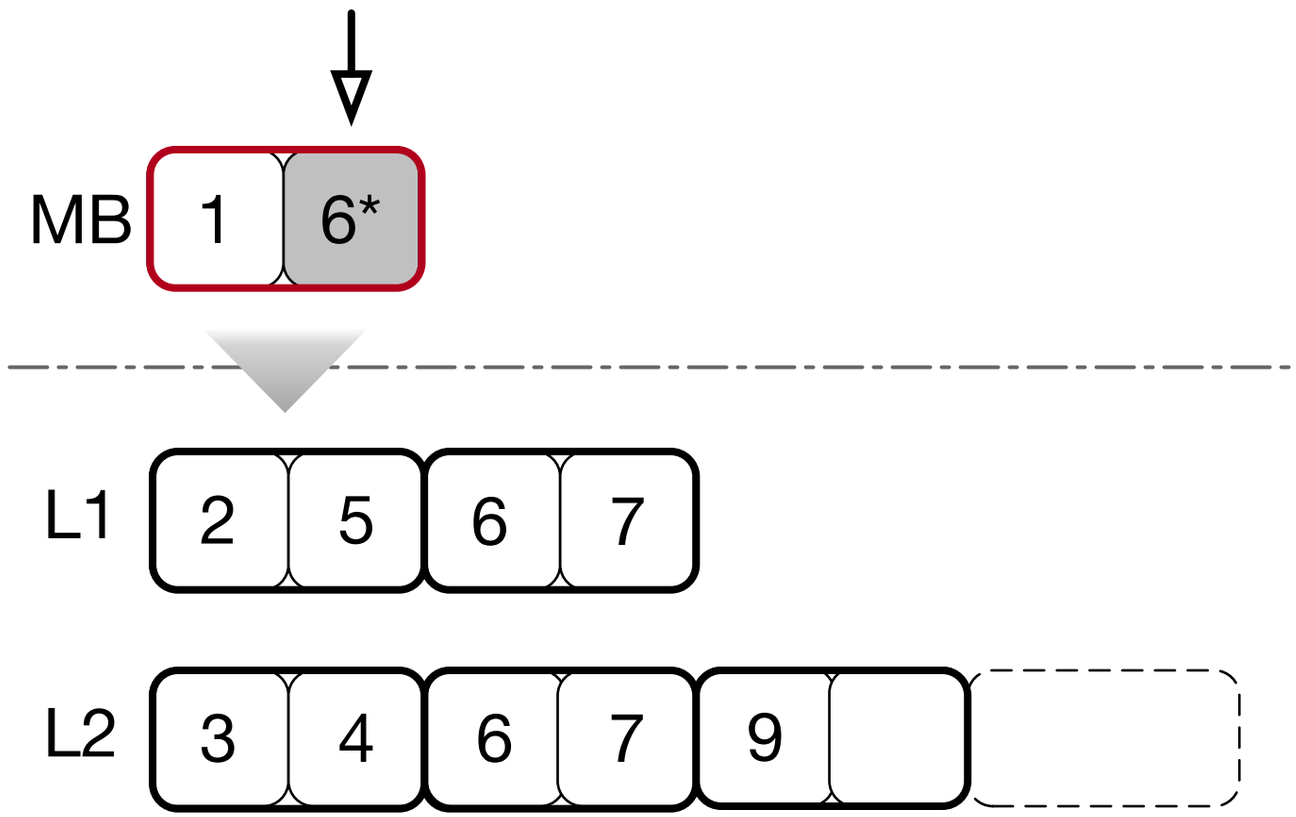
Read-Path


get(5)
MB
Fence Pointers
Bloom Filters
L1
L2
L3
5
skip runs and file accesses

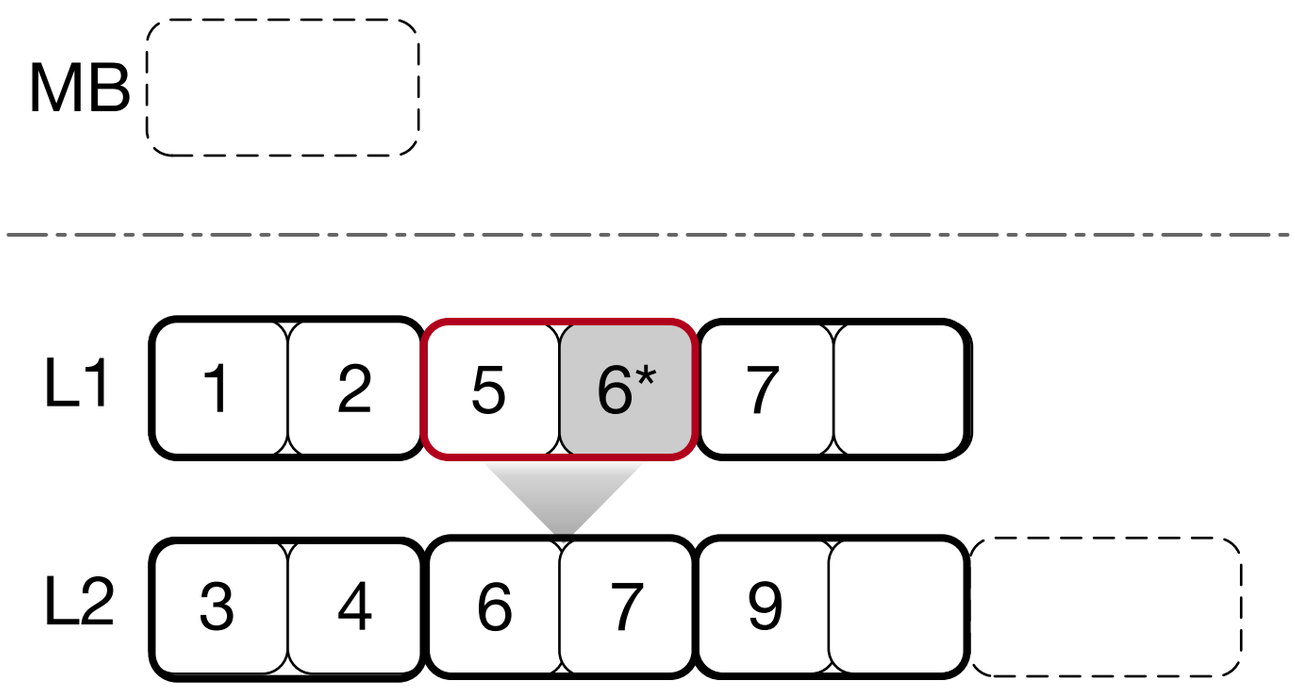
How to achieve fast deletes?

Out-of place!

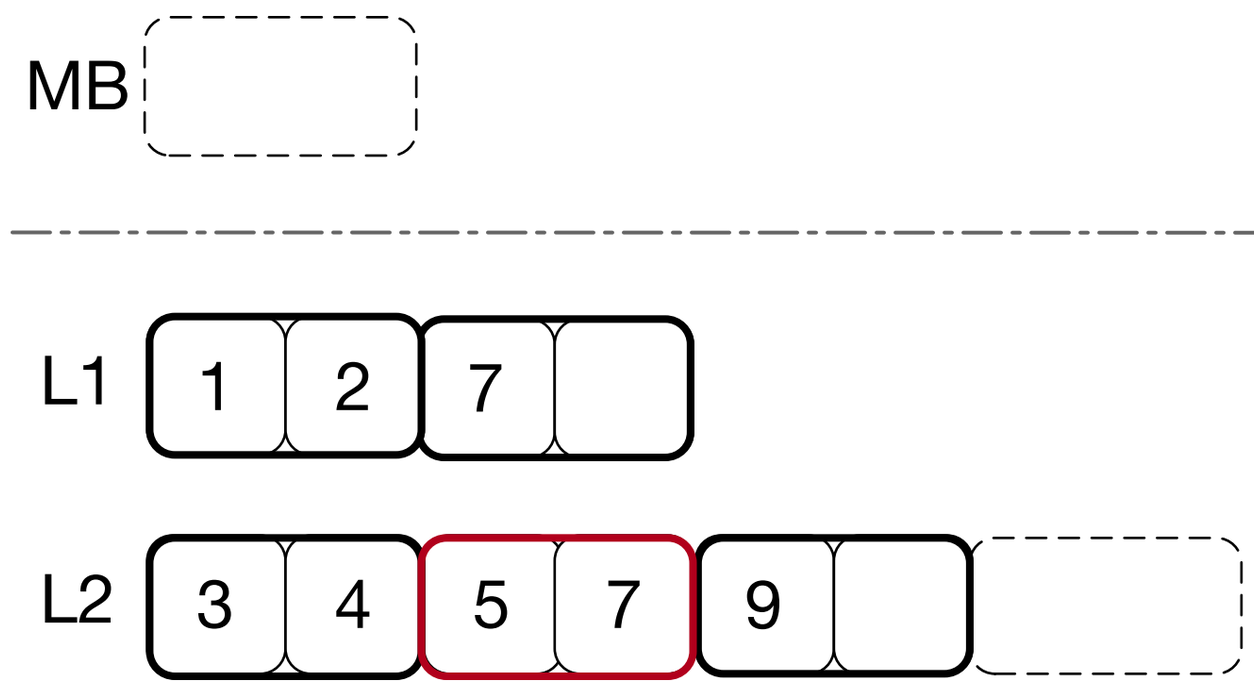
Deletes



Deletes

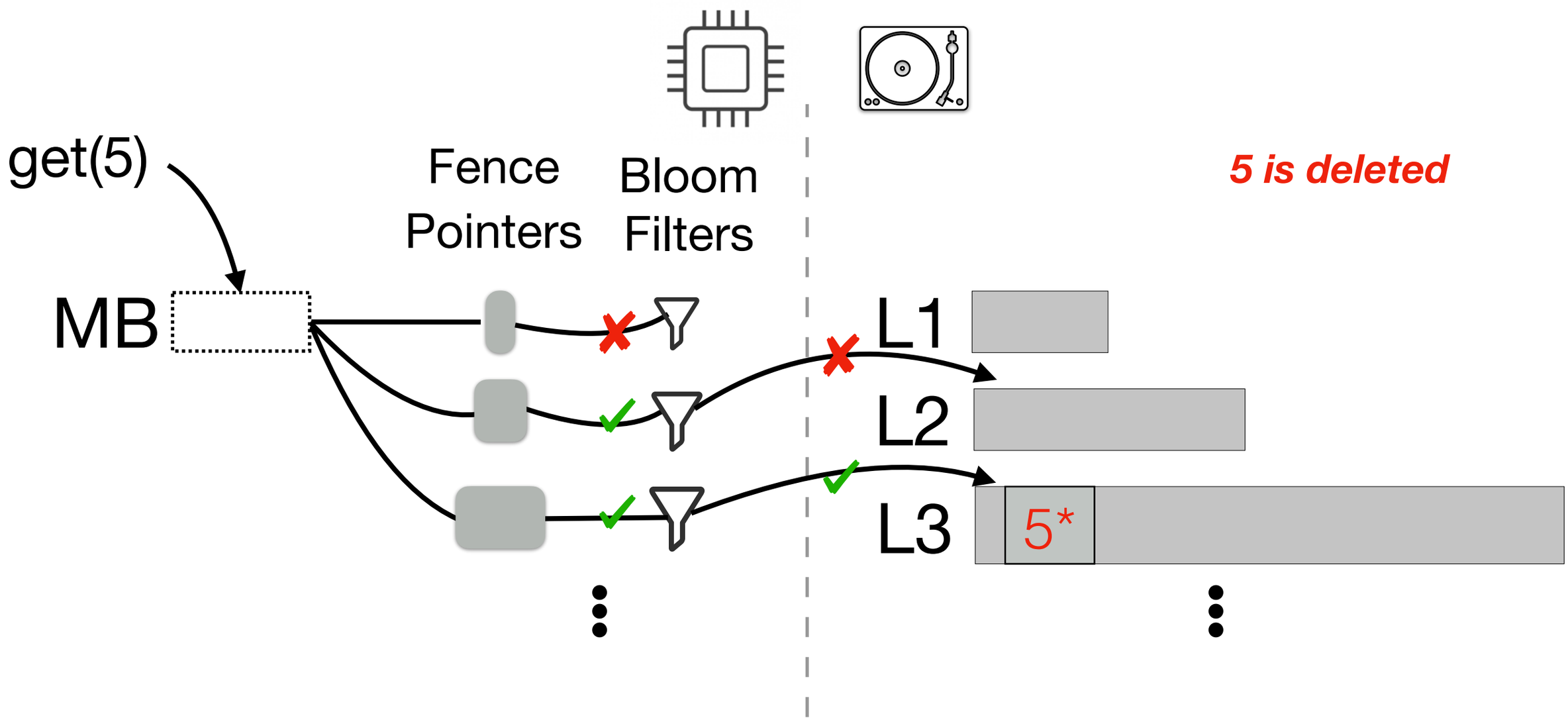


Deletes

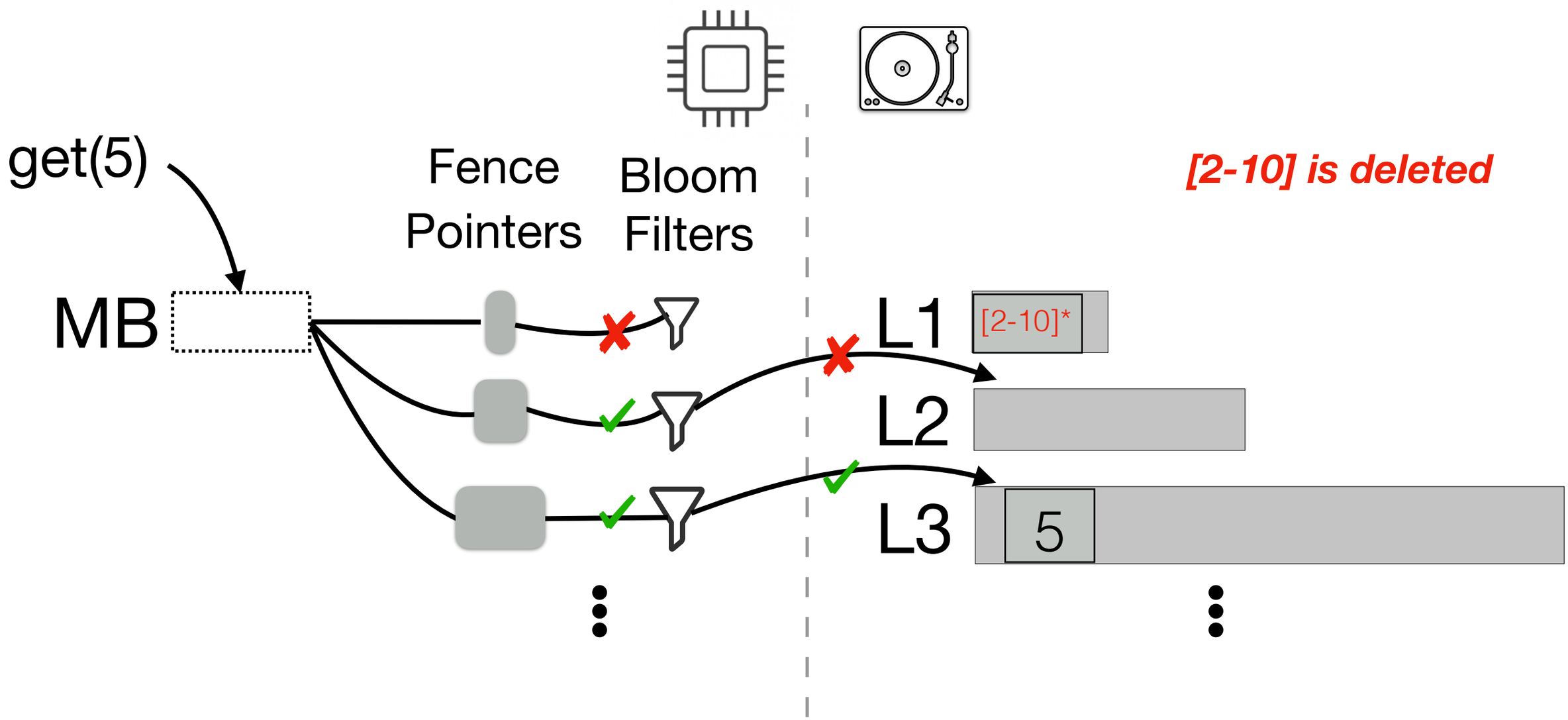


Same process for both range and point deletes

Read-Path for Point Deletes

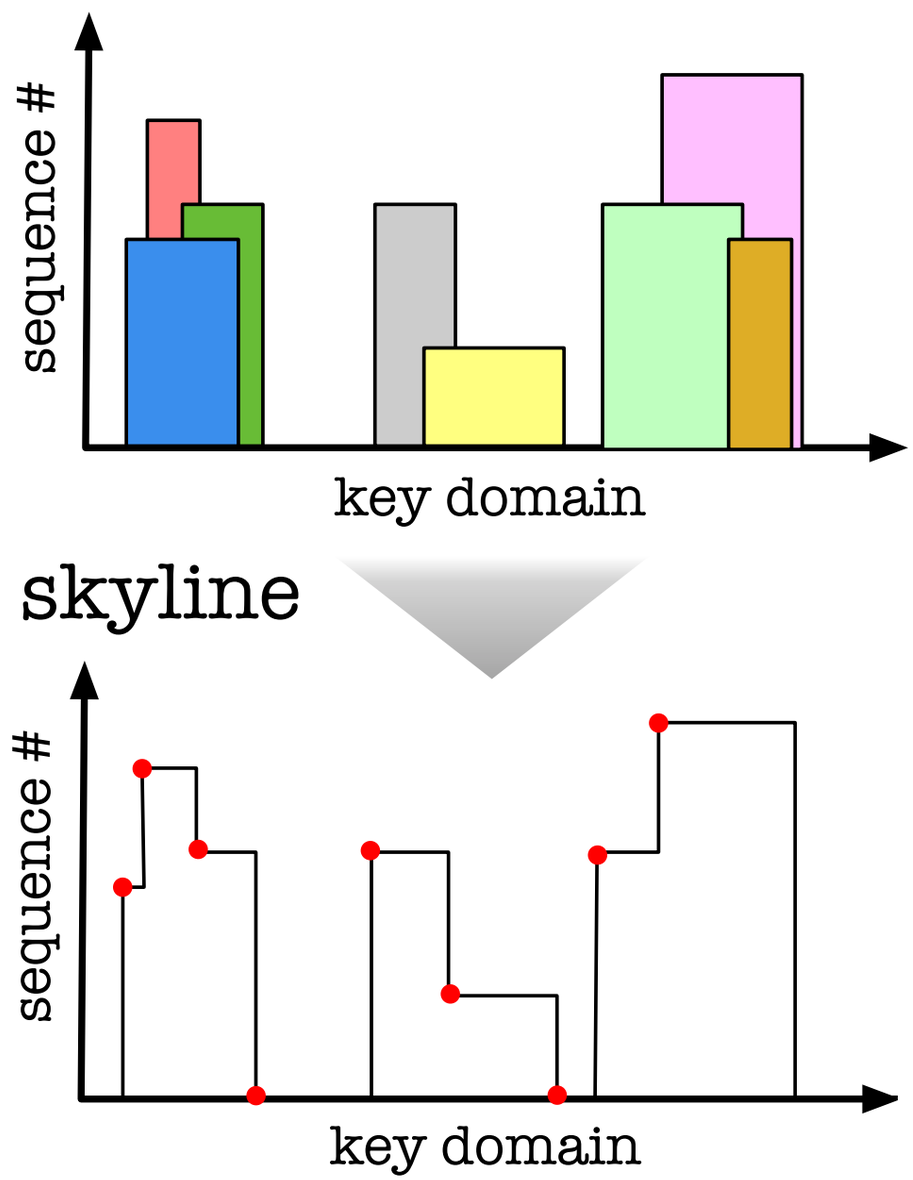


Read-Path for Range Deletes

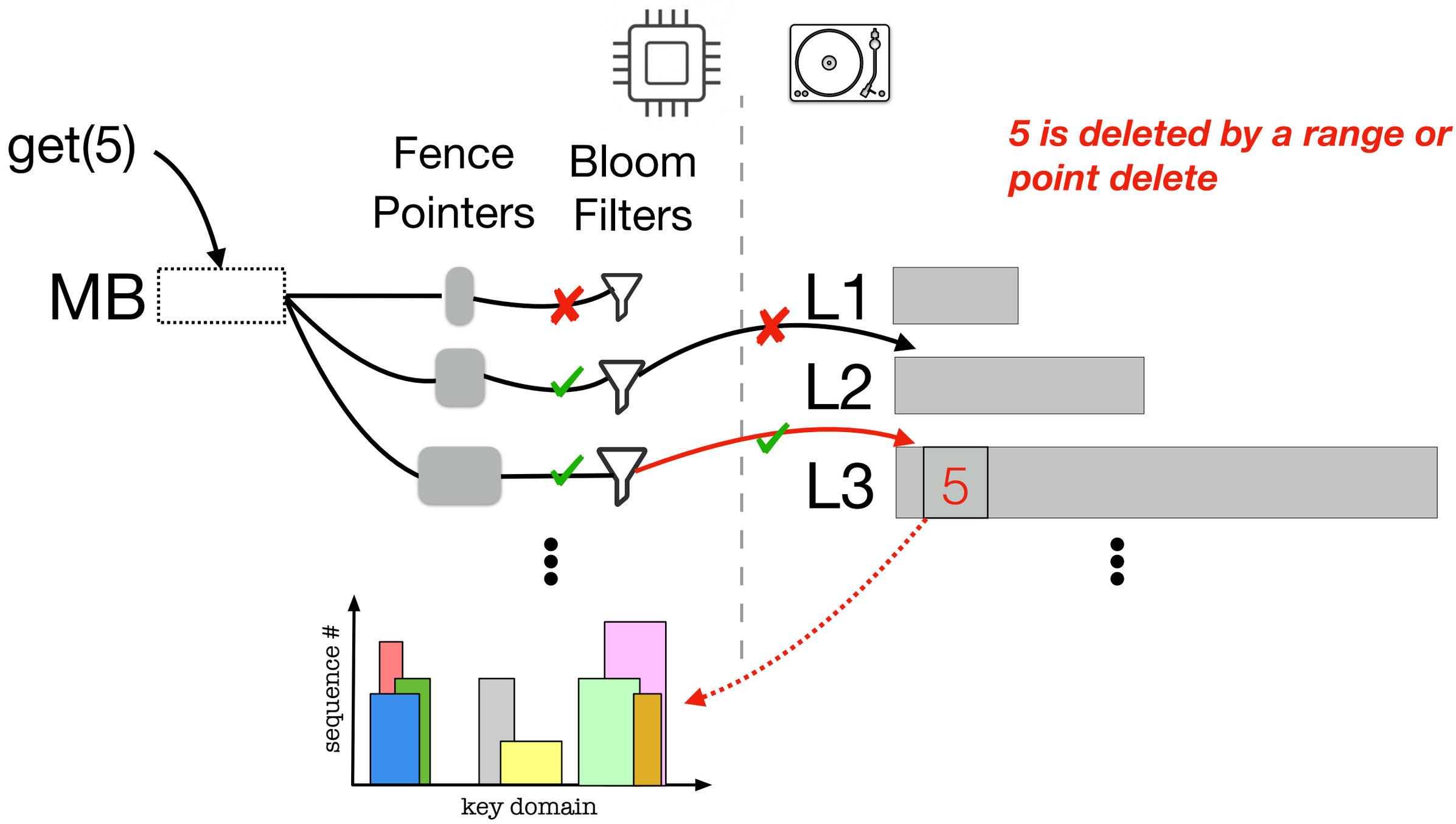


Need for an auxiliary structure!

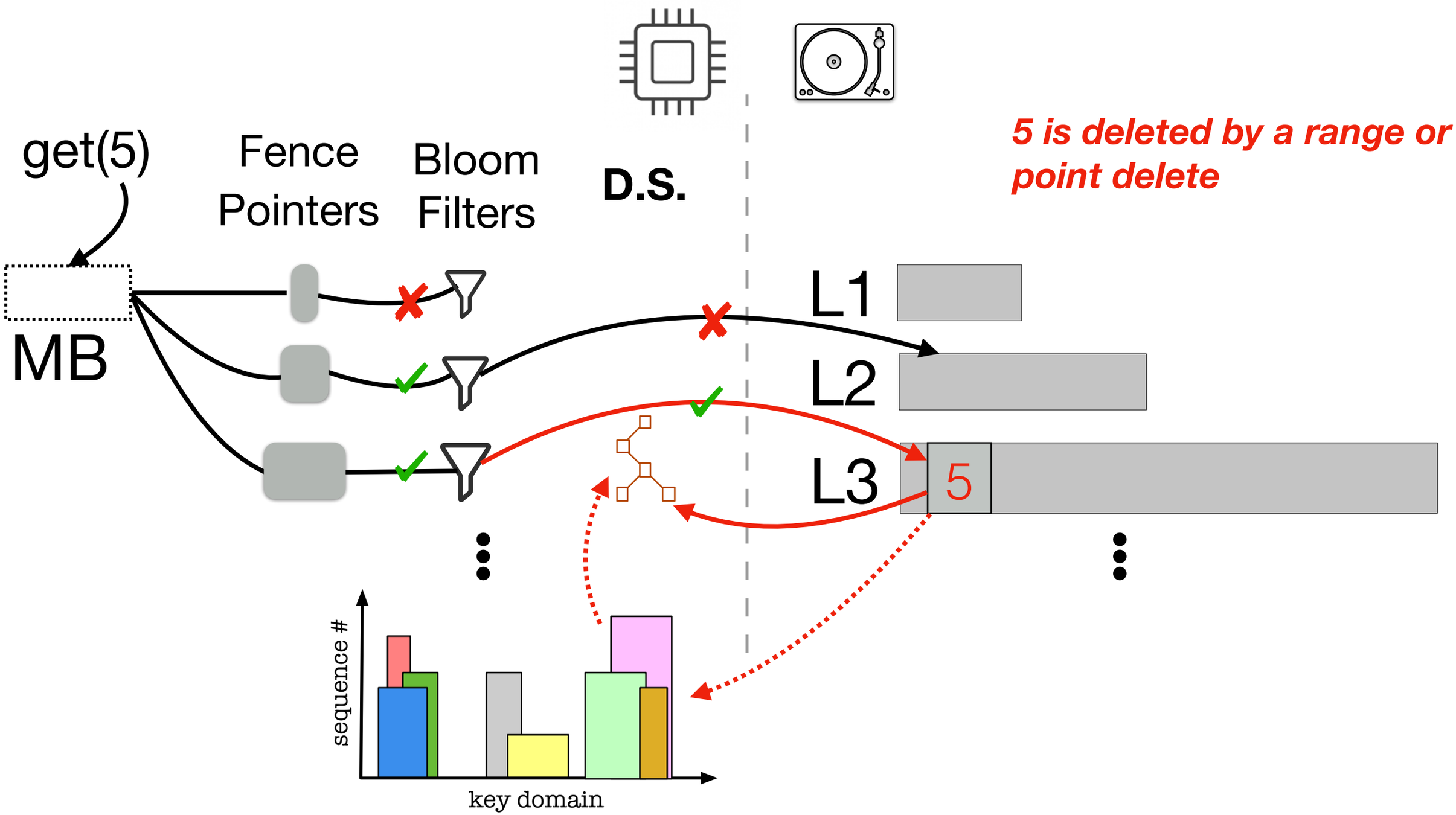
The Skyline



Read - Path Overview

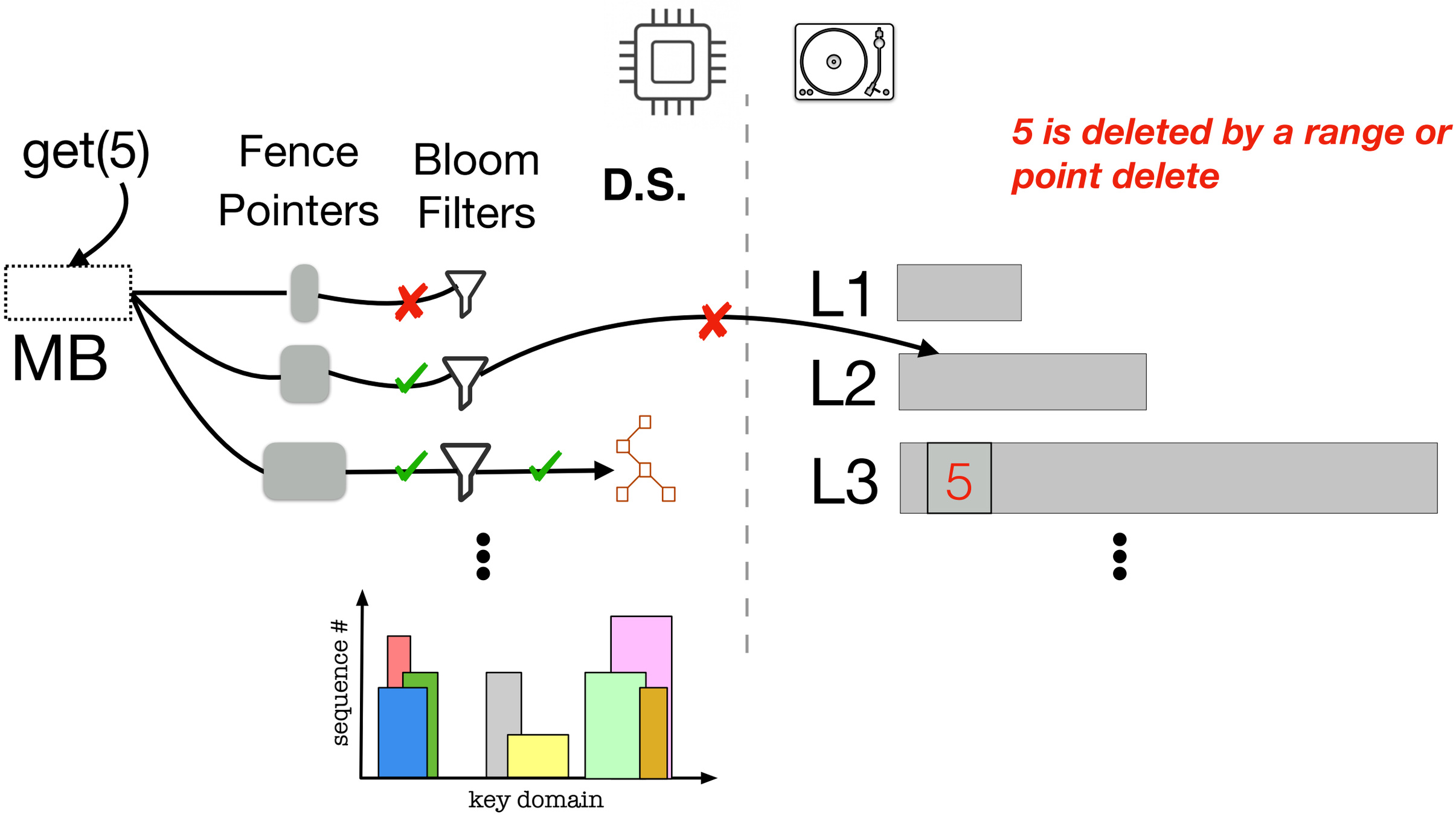


Proposed solution



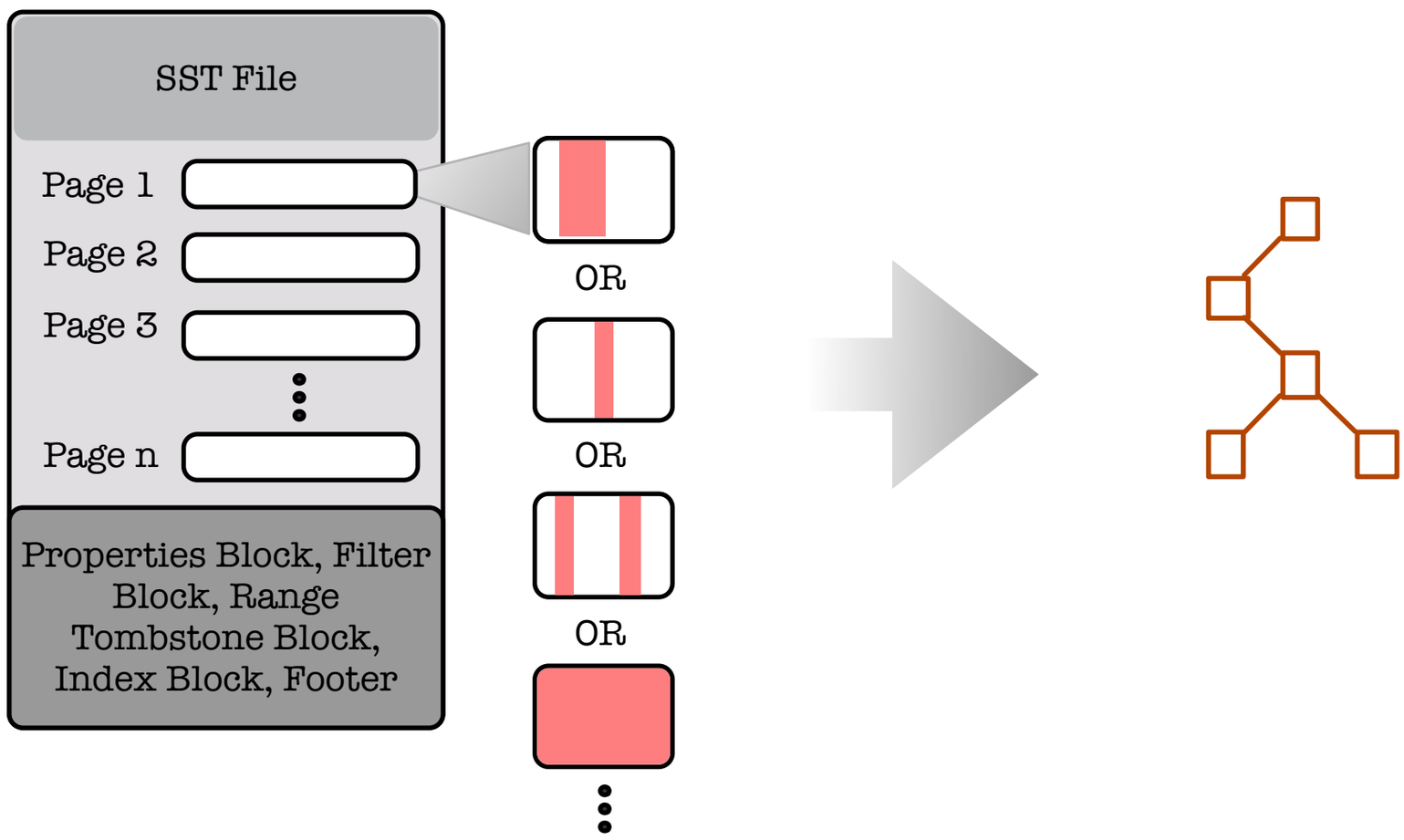
5 is deleted by a range or point delete

Proposed solution in action

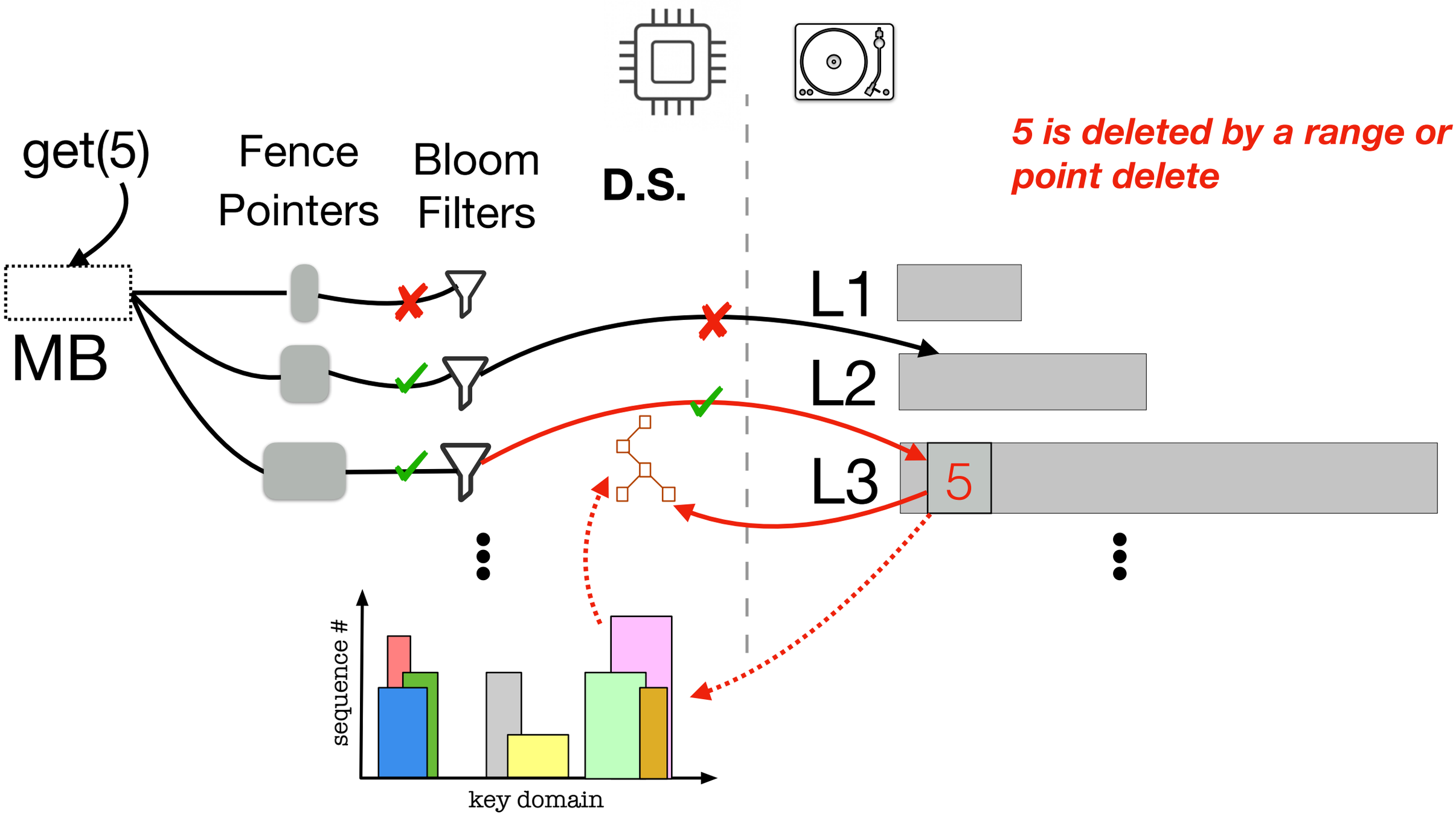


5 is deleted by a range or point delete

Optimization

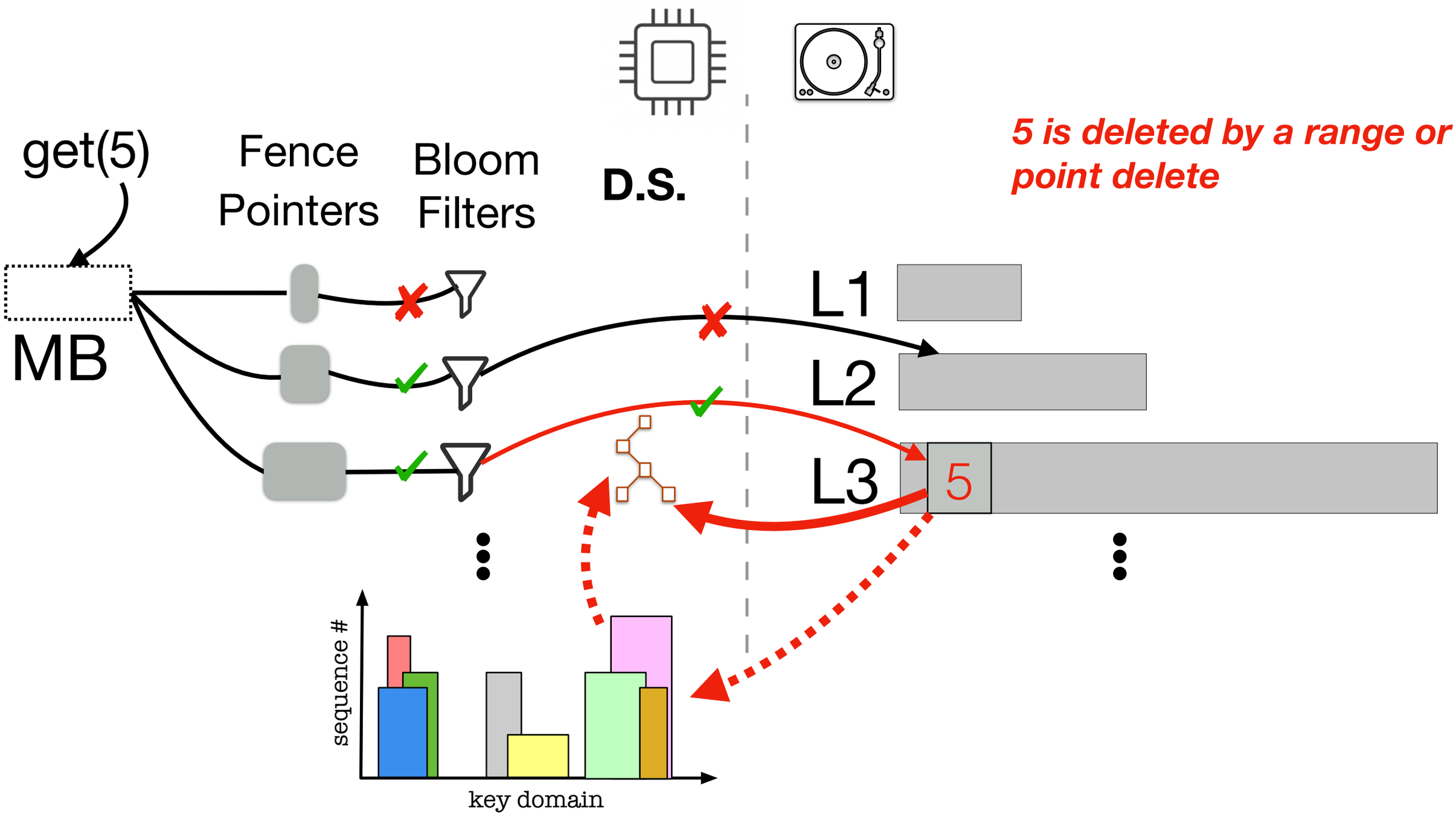


Optimization benefit



5 is deleted by a range or point delete

Optimization benefit



D.S. desired properties

- **No false positives**
- **Small memory footprint**
- **Support for fast inserts and reads**

Our method vs caching

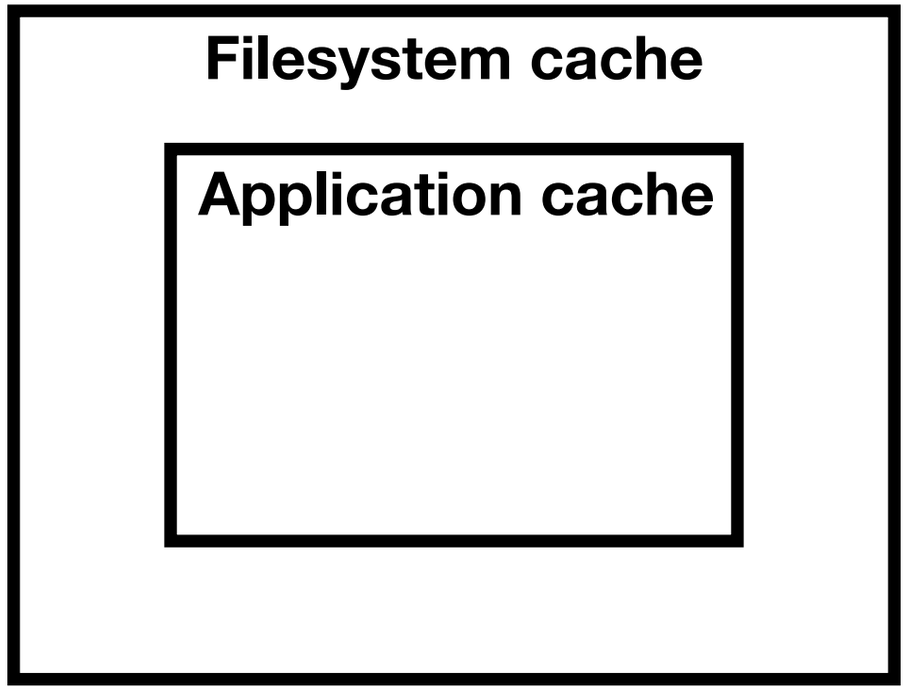
- **Better control of what we store**
- **Tunable memory footprint**
- **Both could work together**

Experimentation

Required Tools

- **Program that uses the RocksDB API**
- **Workload generator with multiple options**
- **Automated shell script to run experiments**

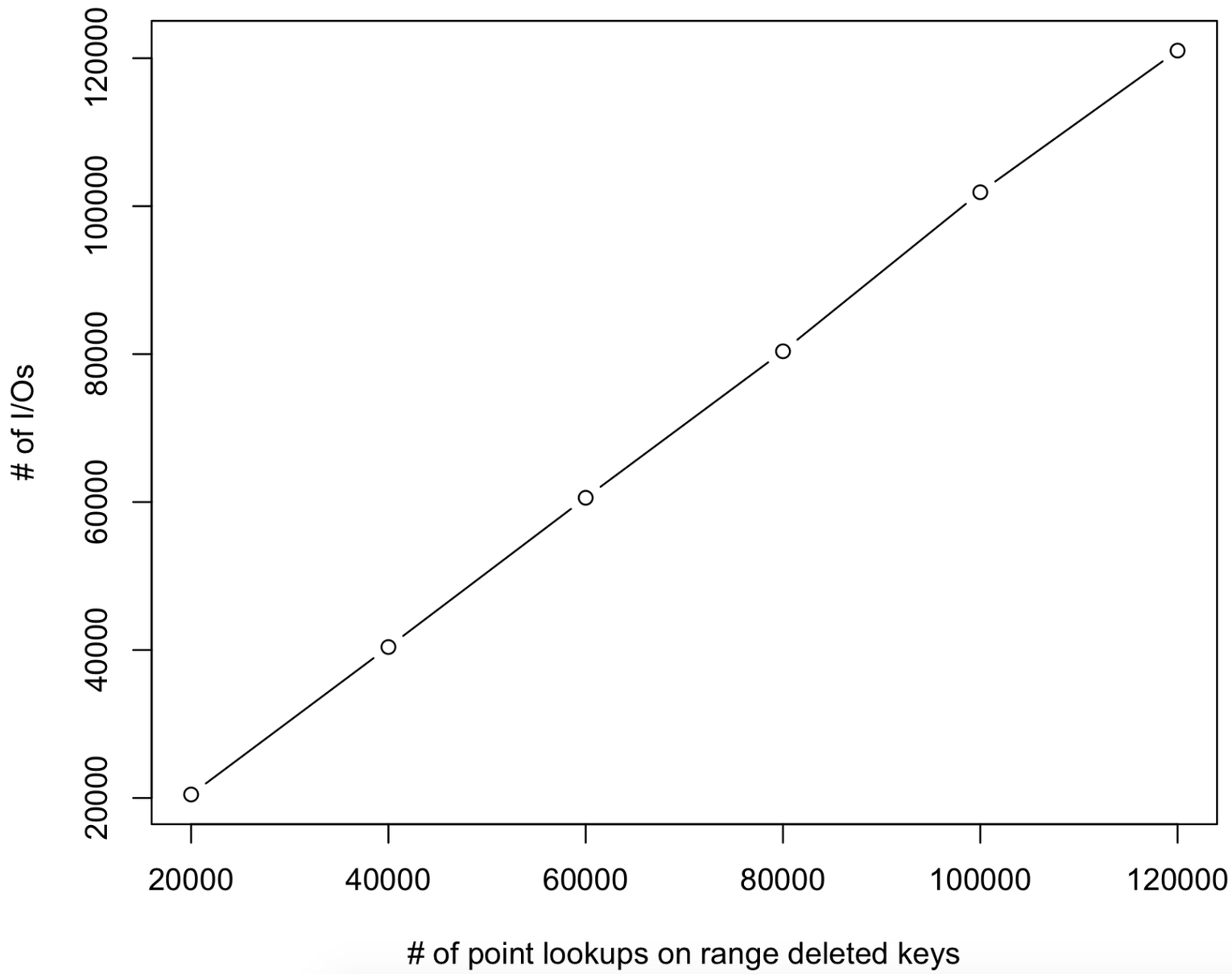
Application vs Filesystem cache



Disabled filesystem cache -> no OS interference

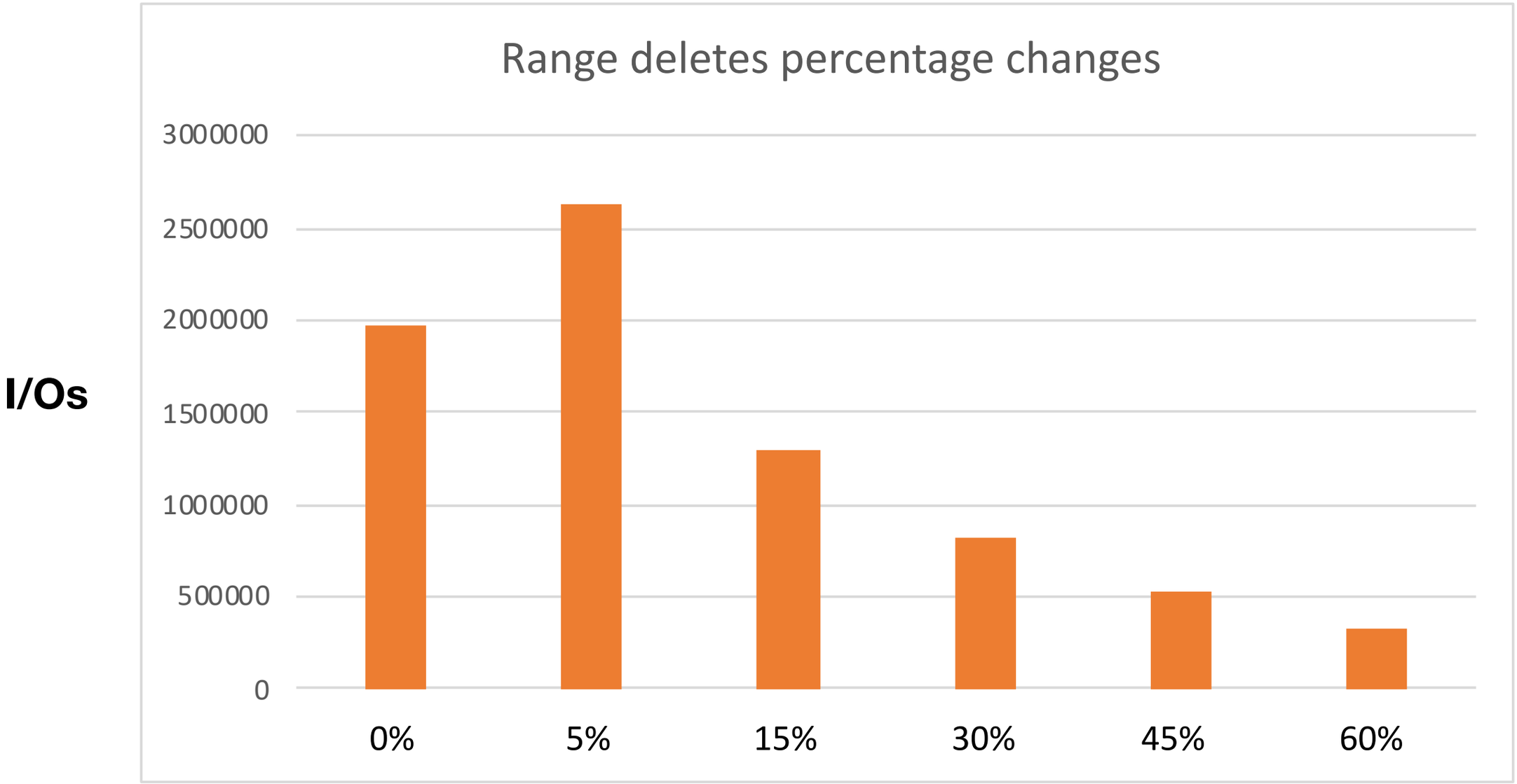
Repetitive queries on range deleted key

Both Filesystem and Application cache disabled



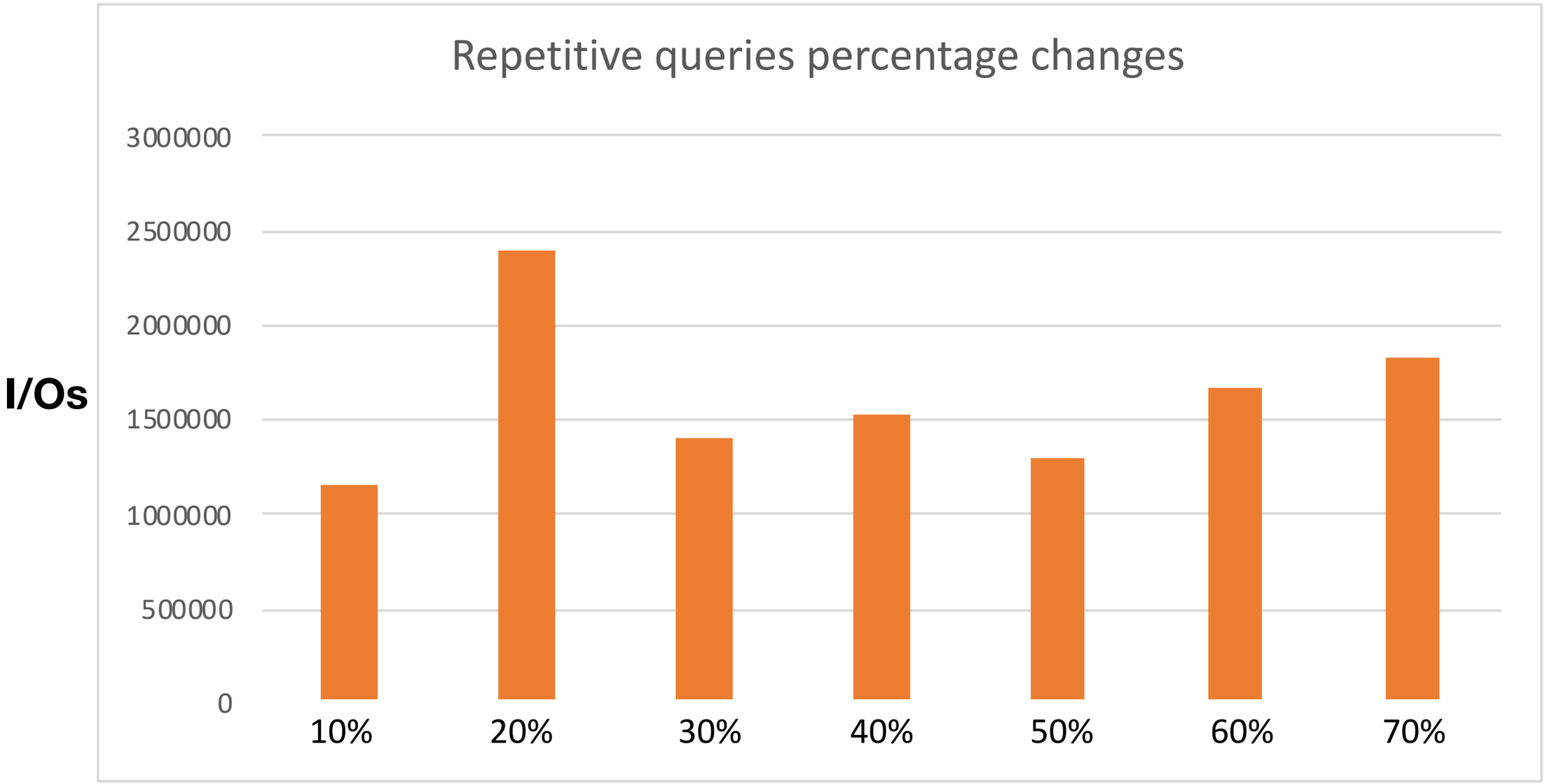
Measuring range deletes I/Os

Inserts: 1.000.000
Queries: 500.000
Repetitive Queries: 40%
Entry size: 1MB
Application Cache (Block Cache) size: 8MB
Variable: range deleted portion of the DB



Measuring range deletes I/Os

Inserts: 1.000.000
Queries: 500.000
Range deleted portion of the DB: 15%
Entry size: 1MB
Application Cache (Block Cache) size: 8MB
Variable: percentage of repetitive queries



Thank you!