Isolating Namespace and Performance in Key-Value SSDs for Multi-tenant Environments

Donghyun Min and Youngjae Kim

Sogang University, South Korea

DISCOS LABORATORY

The 13th ACM Workshop on Hot Topics In Storage and File Systems **(HotStorage' 21, July 27-28)**

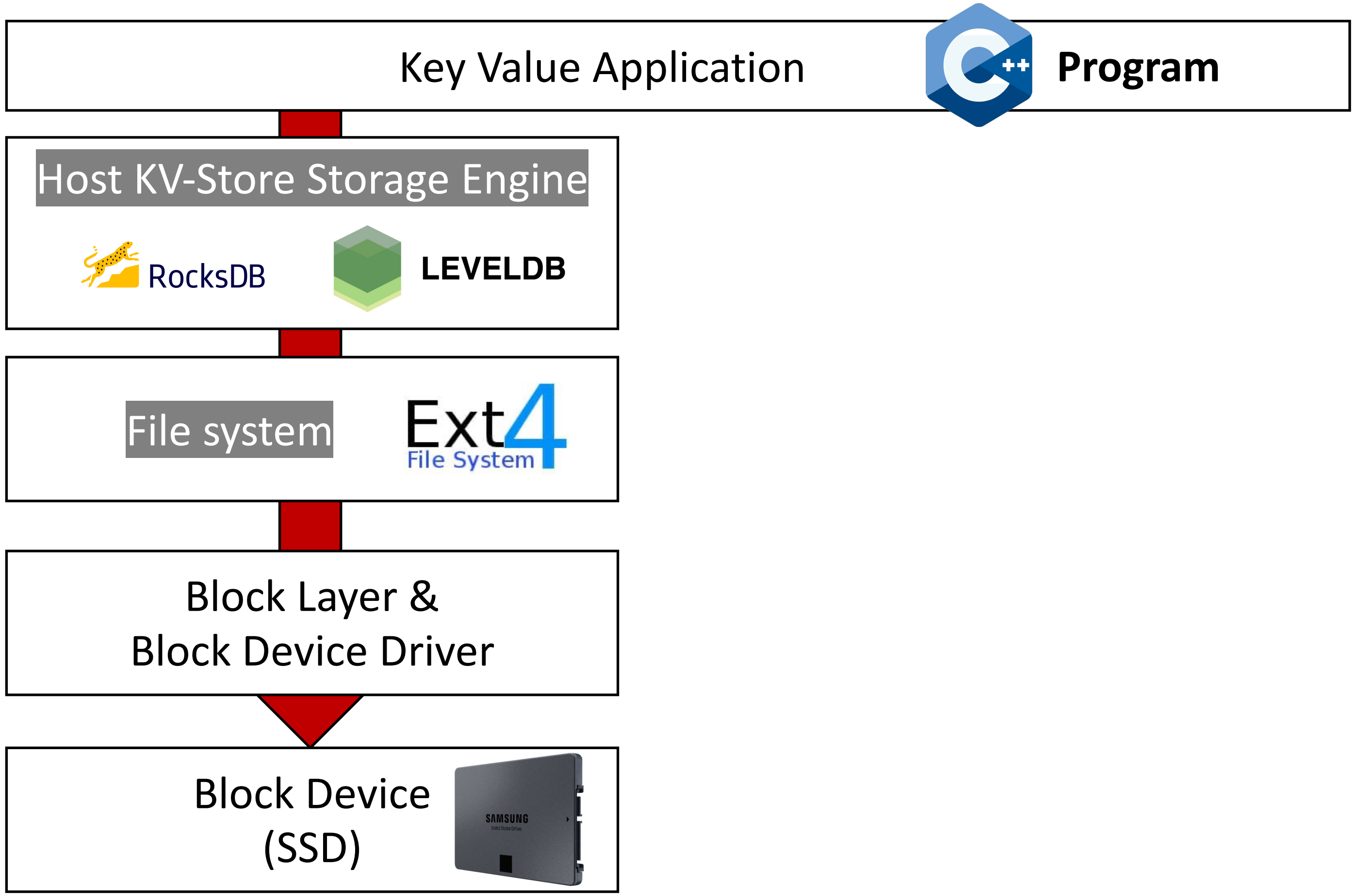
SOGANG UNIVERSITY

Key-Value Store (KV-Store)

- Key-Value Store (KV-Store) is a type of NoSQL database.
  - KV-Store uses simple Key-Value (KV) interface to store/retrieve data.
  - Host-side KV-Store
    - E.g., RocksDB, LevelDB, …

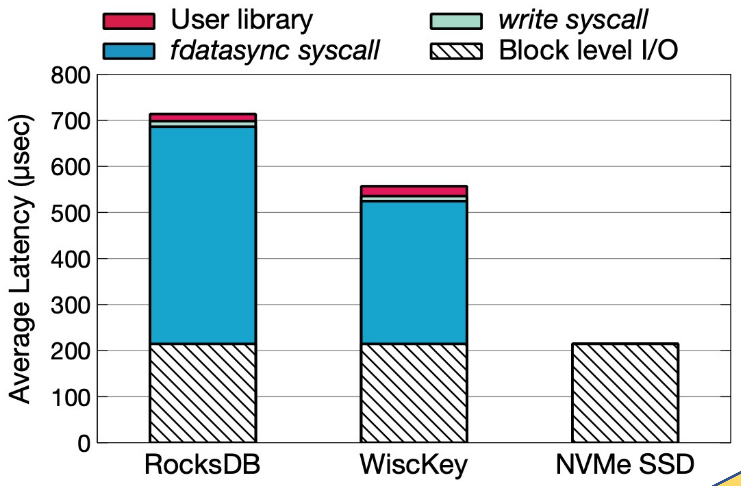
Key-Value Solid-State Drive (KVSSD)

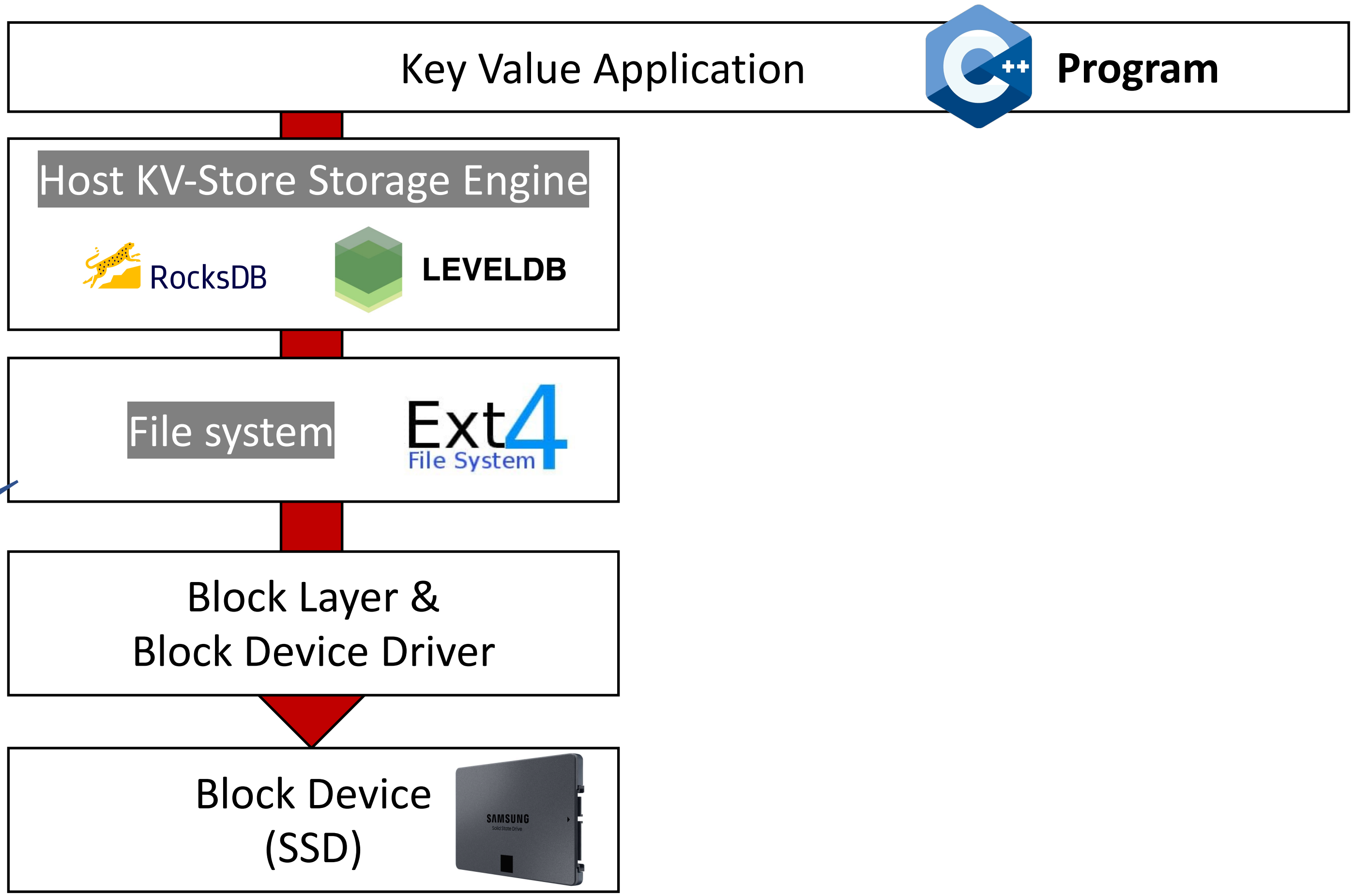
- KVSSD runs storage engine of KV-Store on the SSD.



Key-Value Solid-State Drive (KVSSD)

- KVSSD runs storage engine of KV-Store on the SSD.



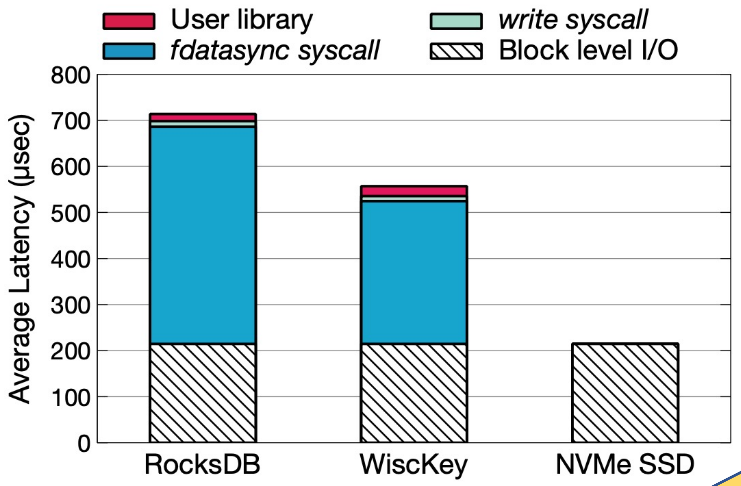


Host file system overhead has been reported[1].

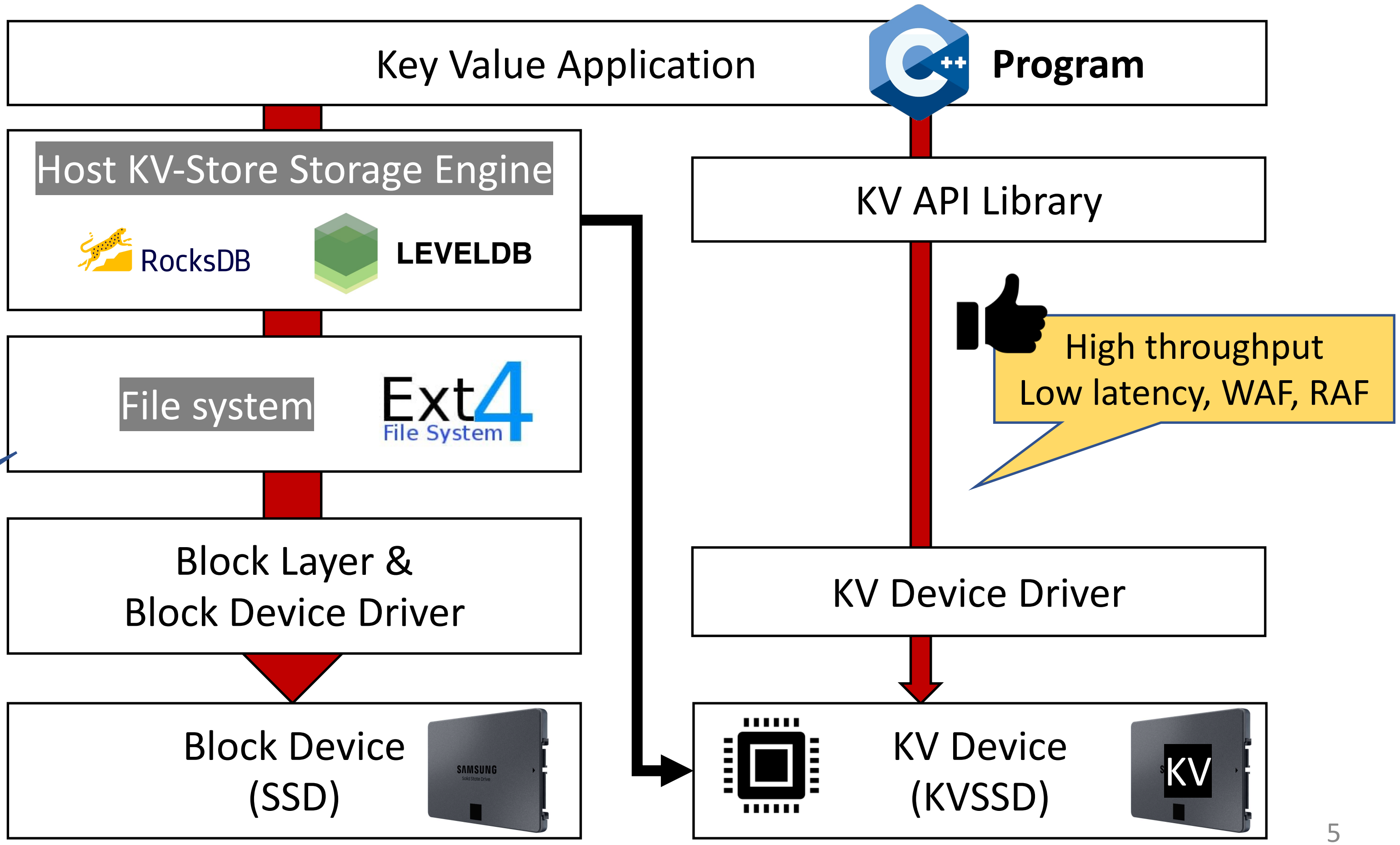
[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

Key-Value Solid-State Drive (KVSSD)

- KVSSD runs storage engine of KV-Store on the SSD.



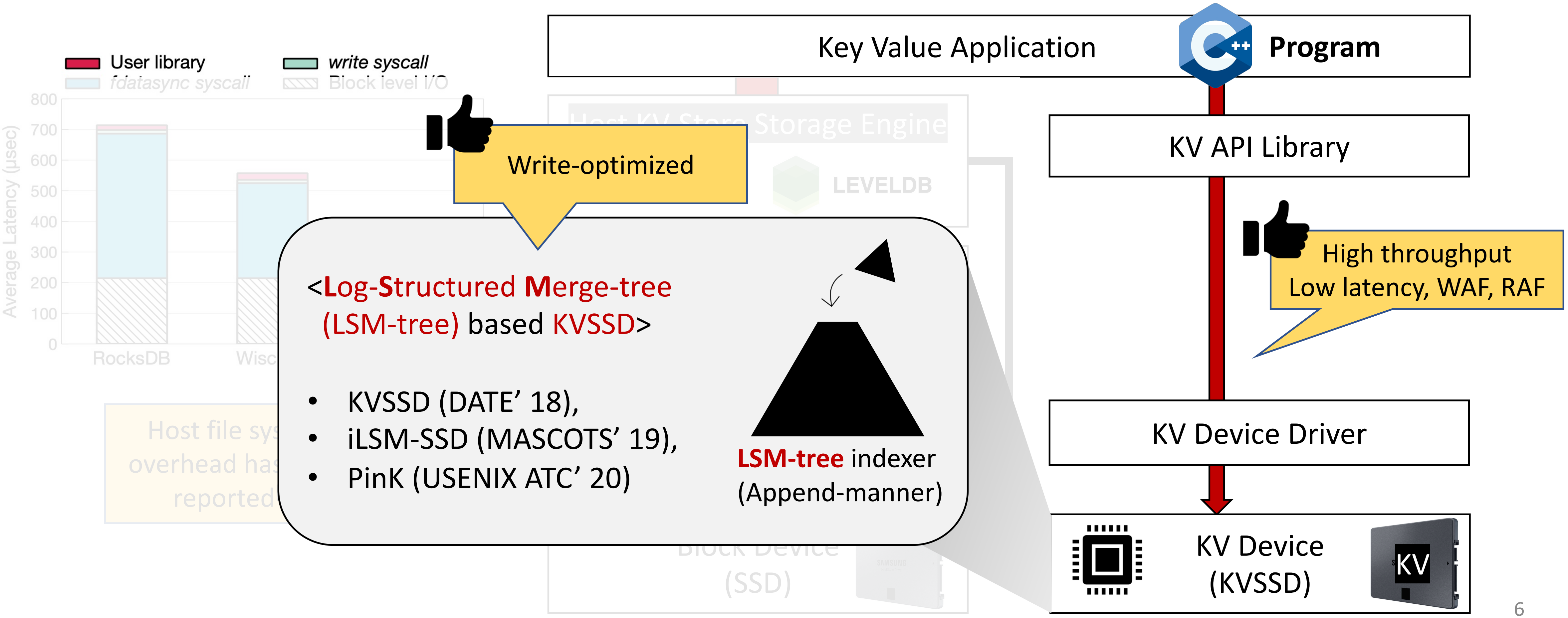
Host file system overhead has been reported[1].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

Key-Value Solid-State Drive (KVSSD)

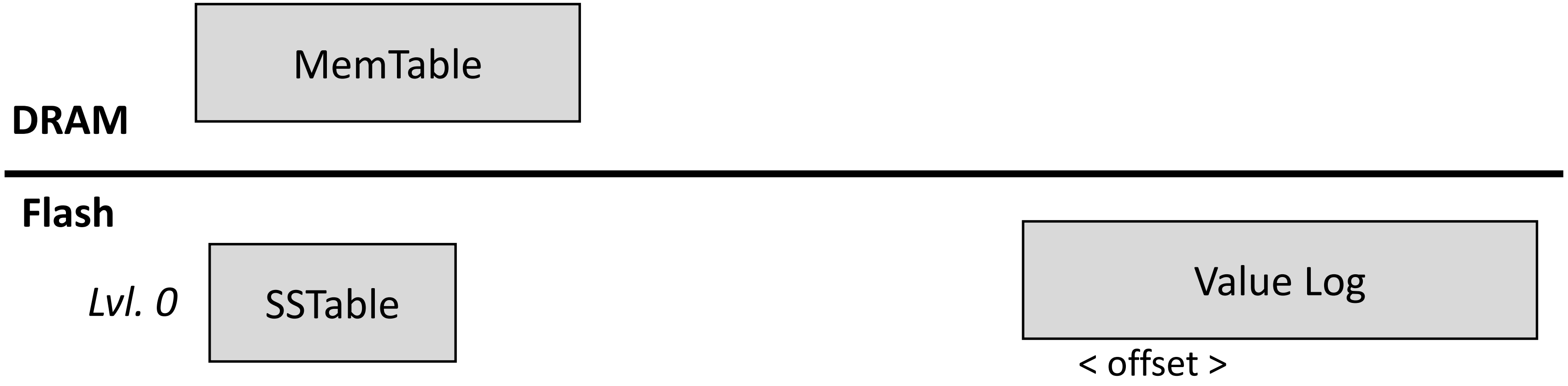
- KVSSD runs storage engine of KV-Store on the SSD.



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

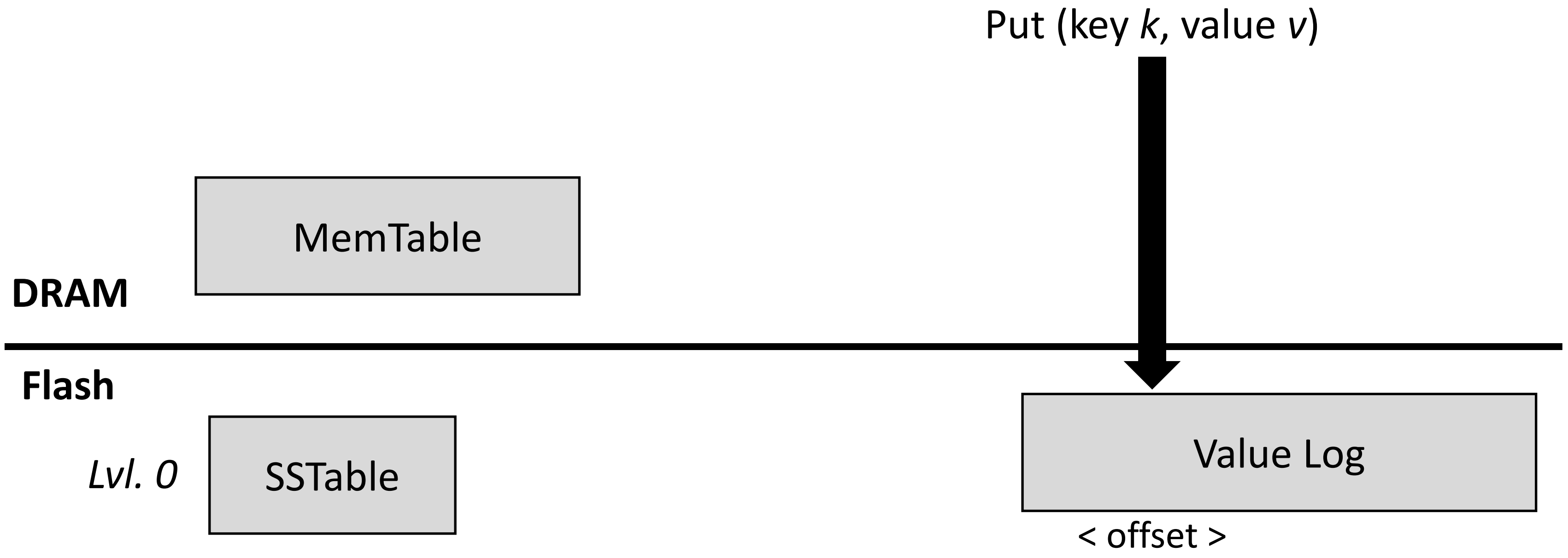


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

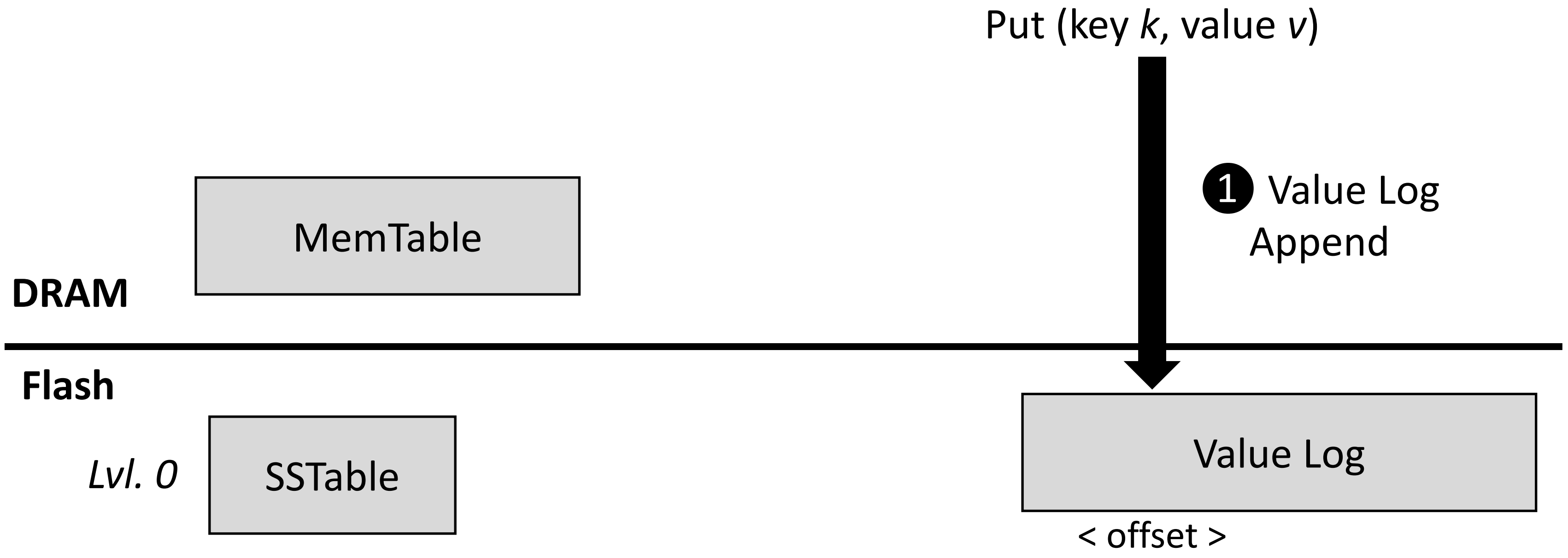


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Sogang University

Log-Structured Merge-tree (LSM-tree) in KVSSD

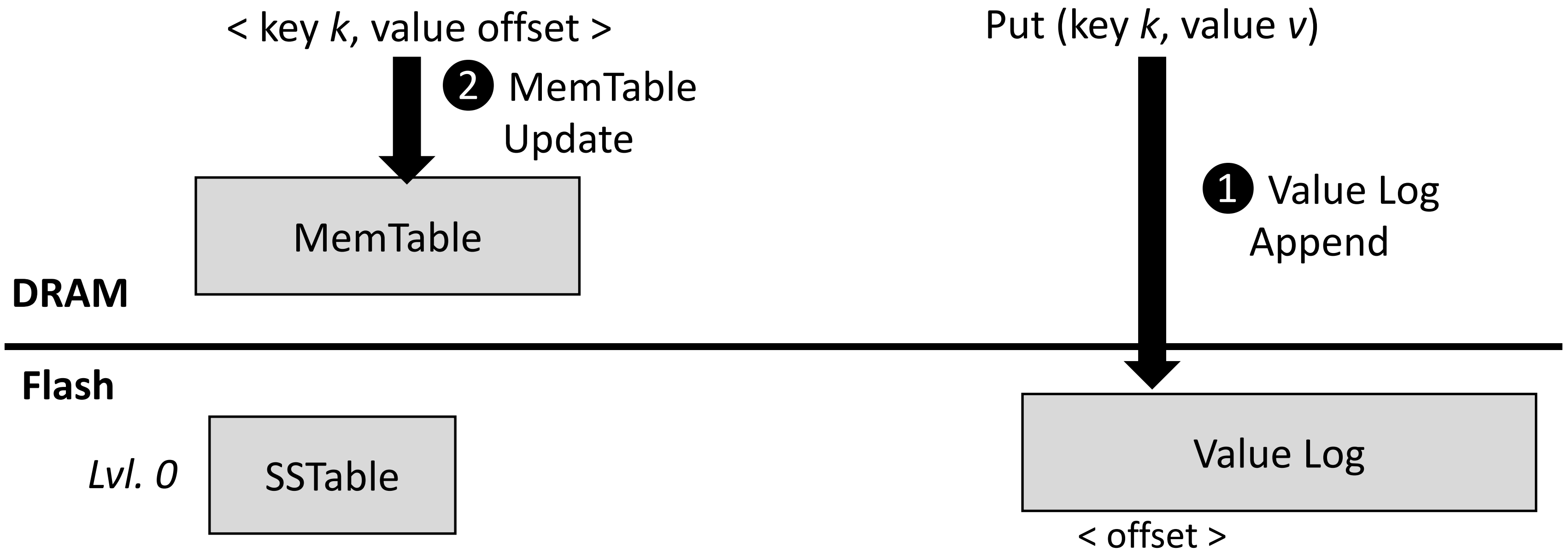
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

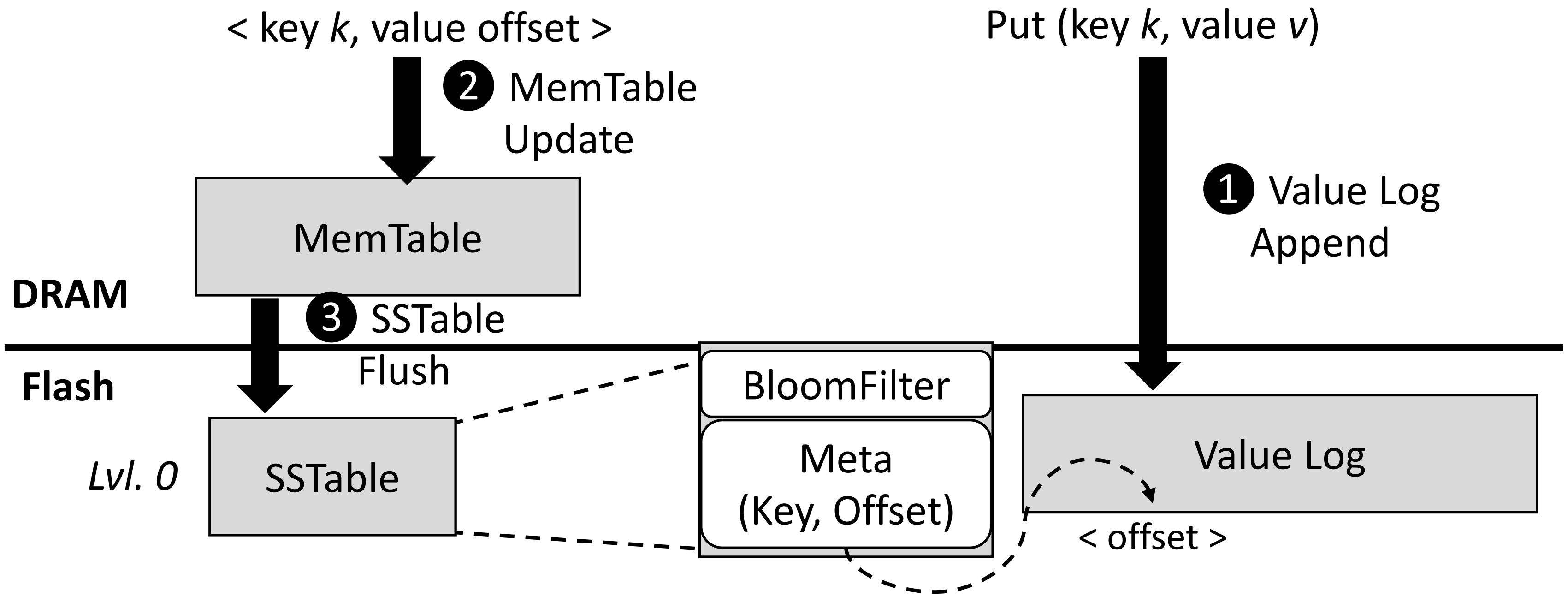
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

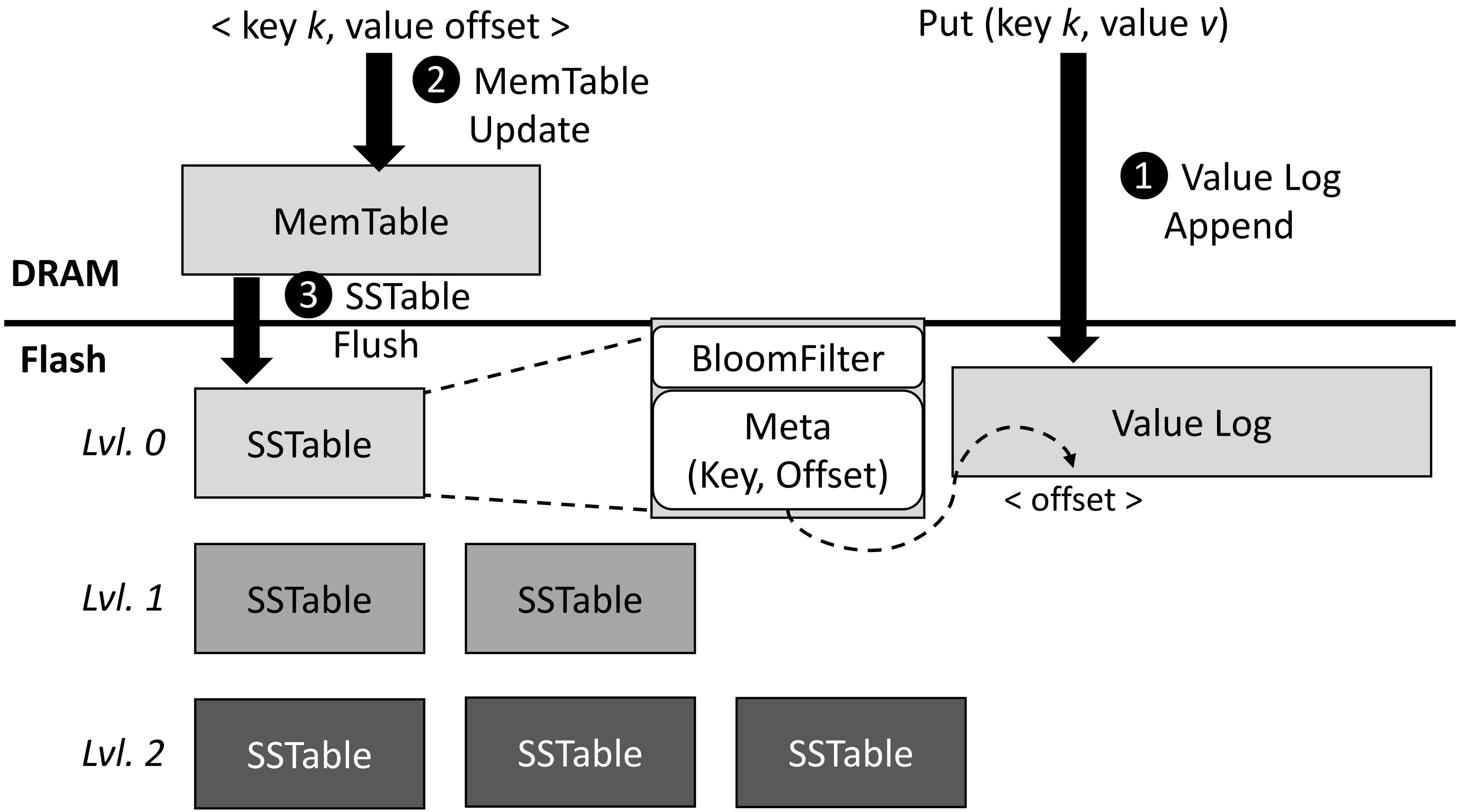


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

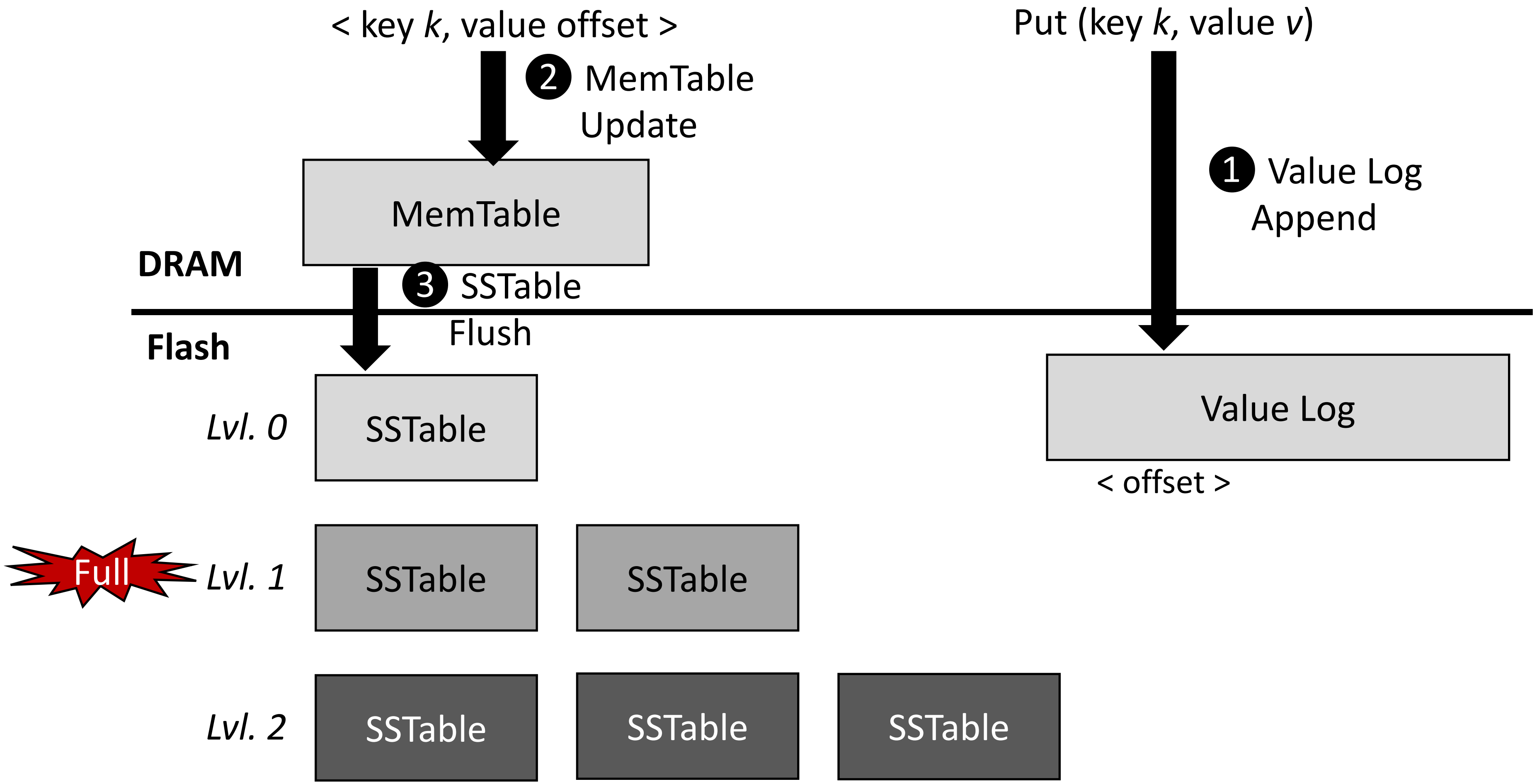
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

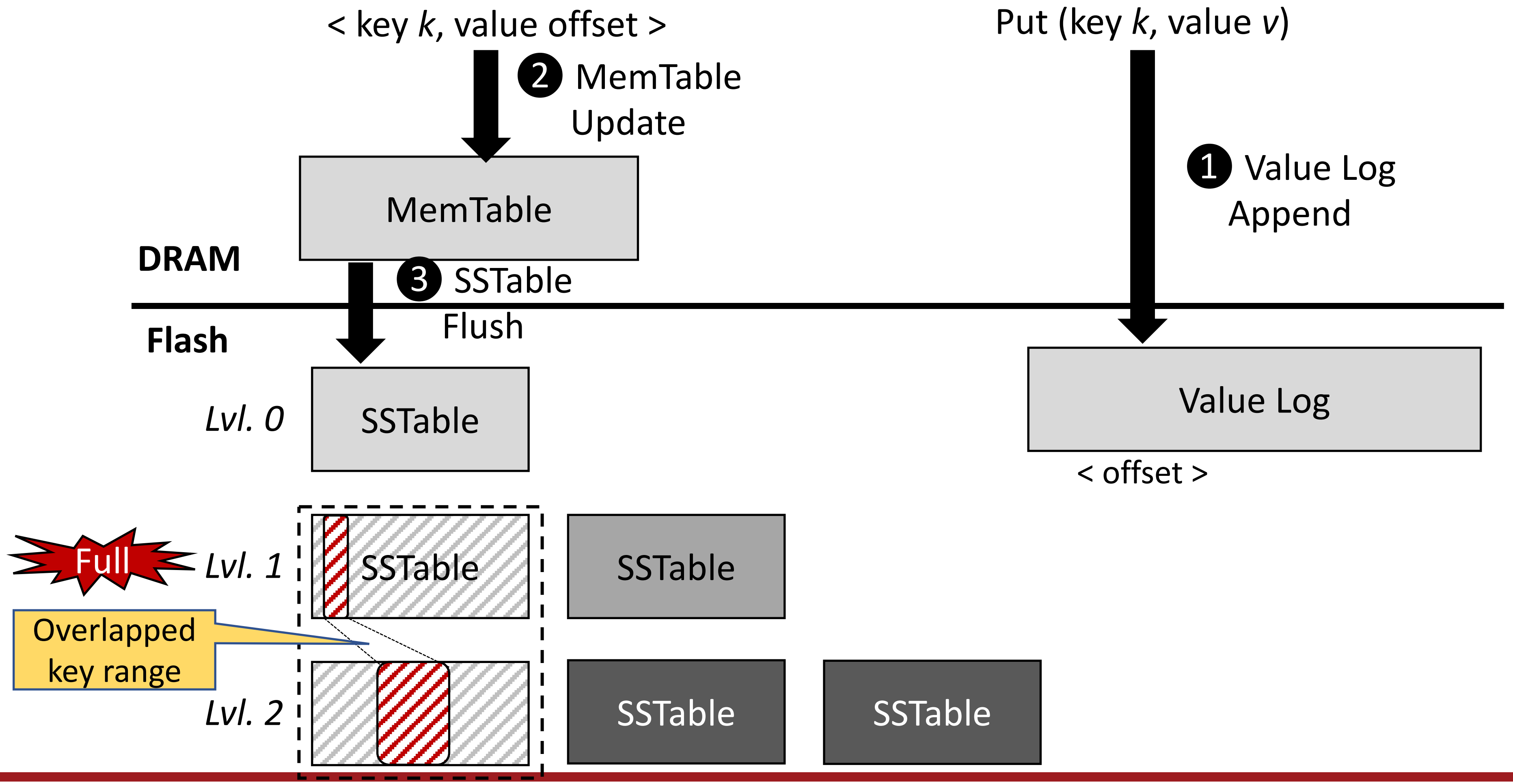
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

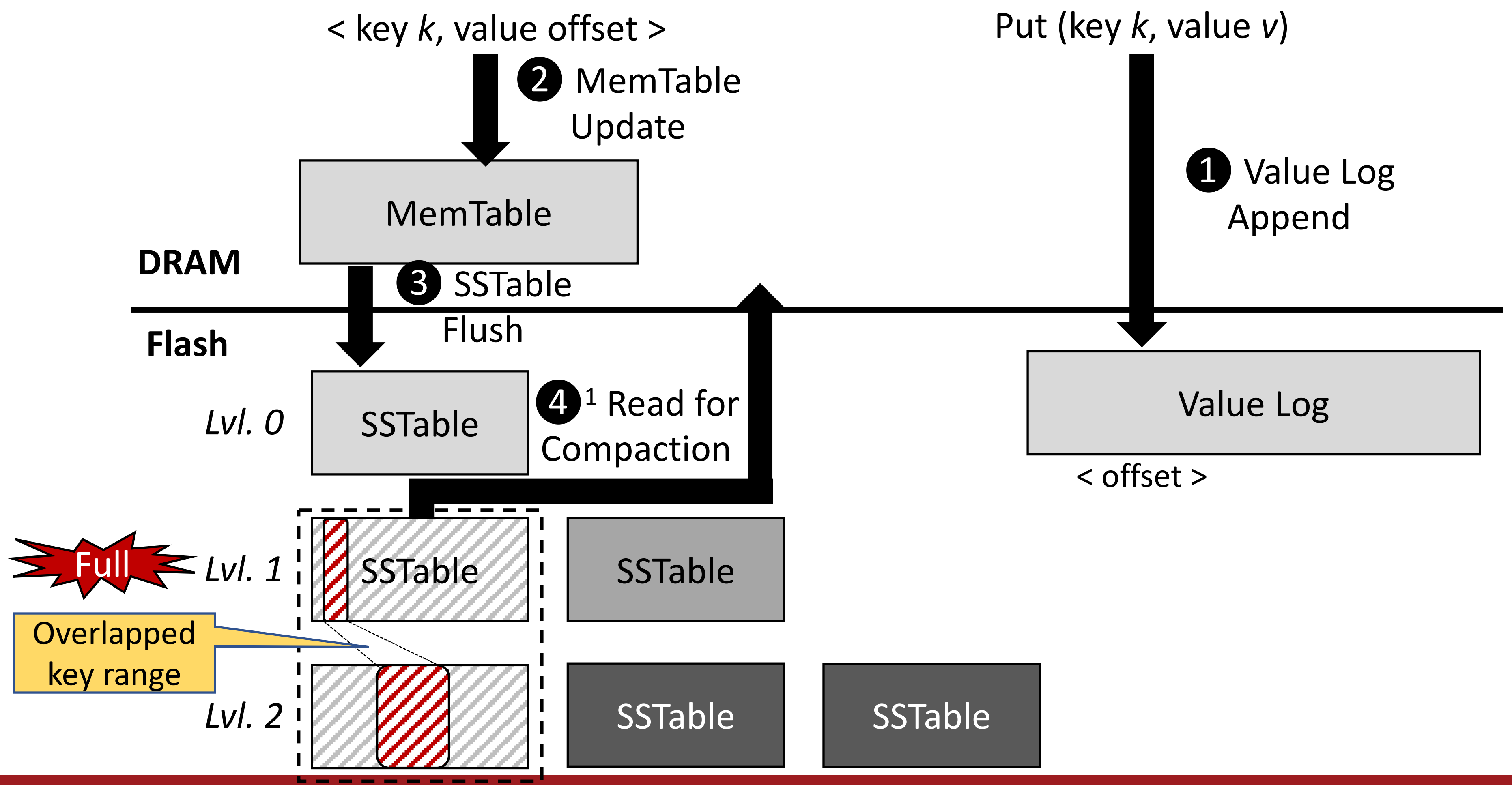


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

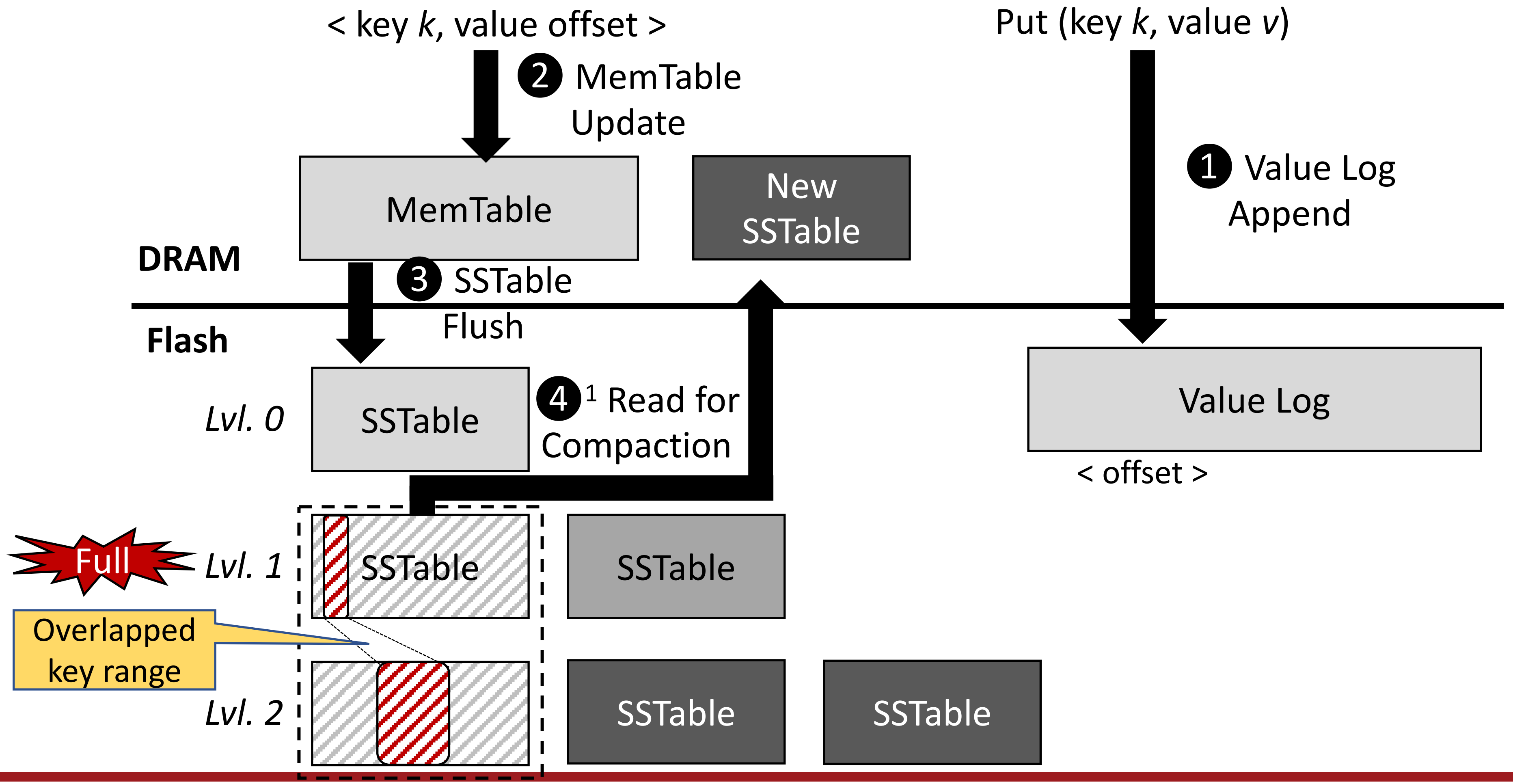


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Sogang University

Log-Structured Merge-tree (LSM-tree) in KVSSD

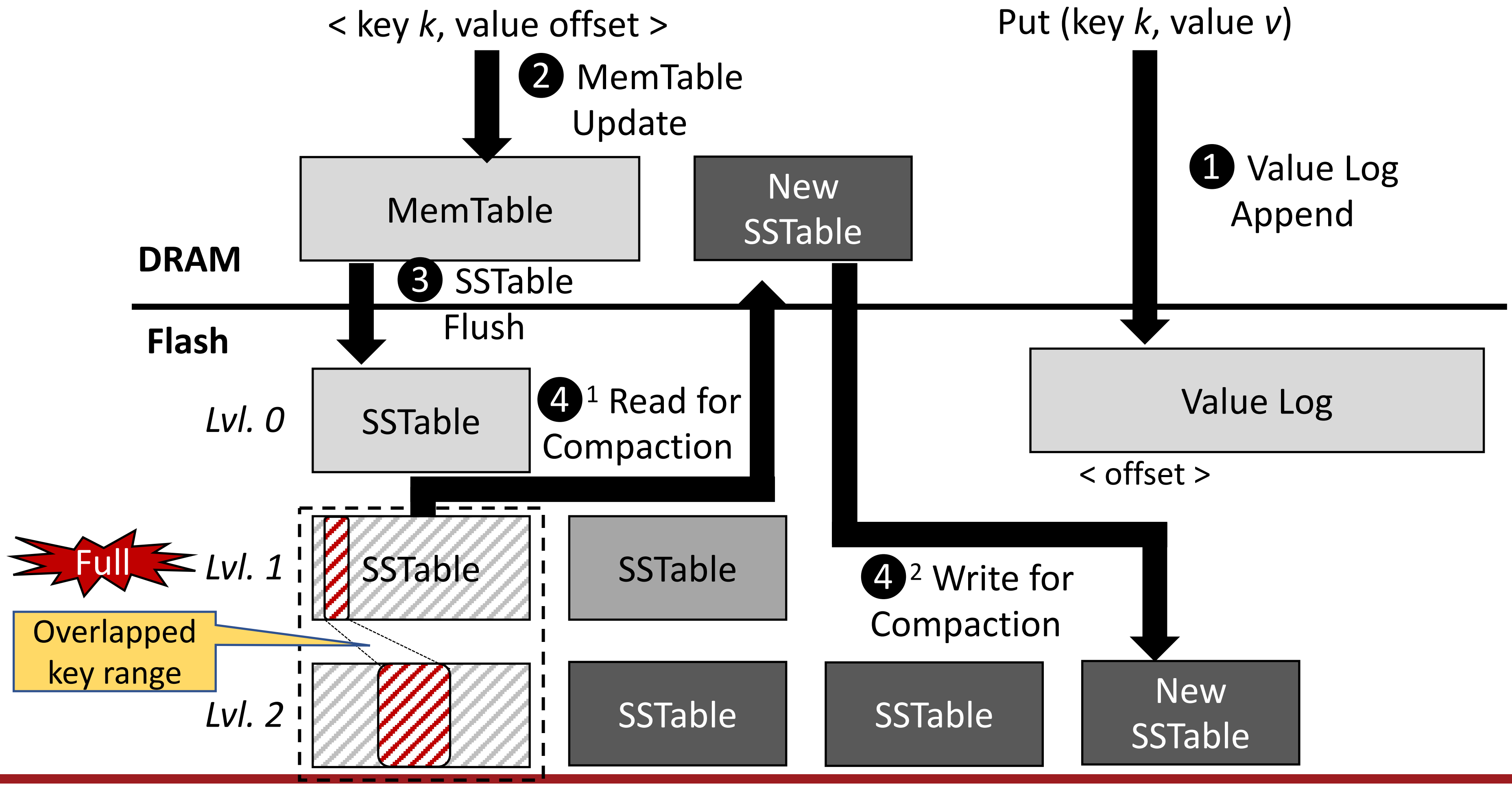
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

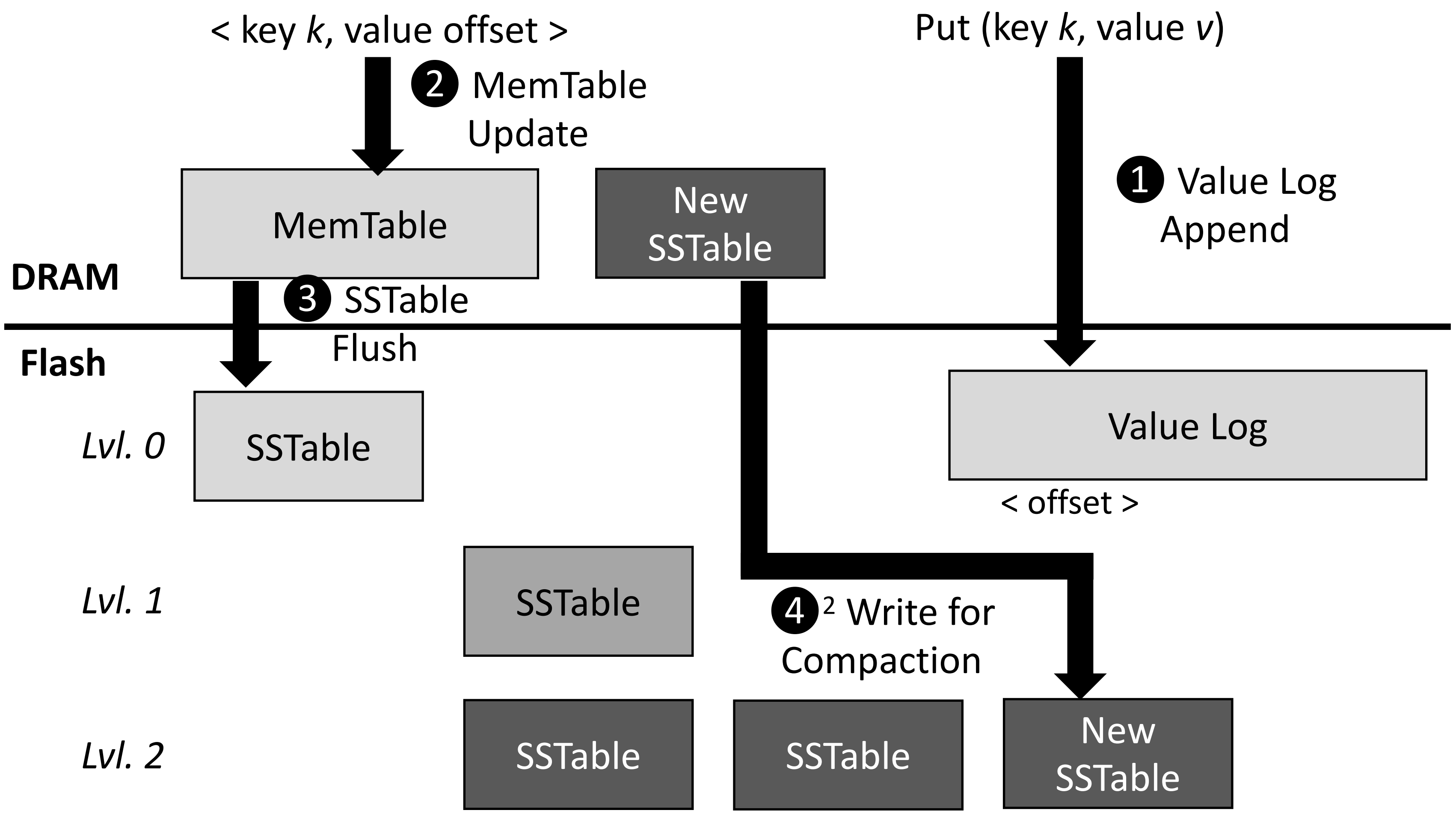


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

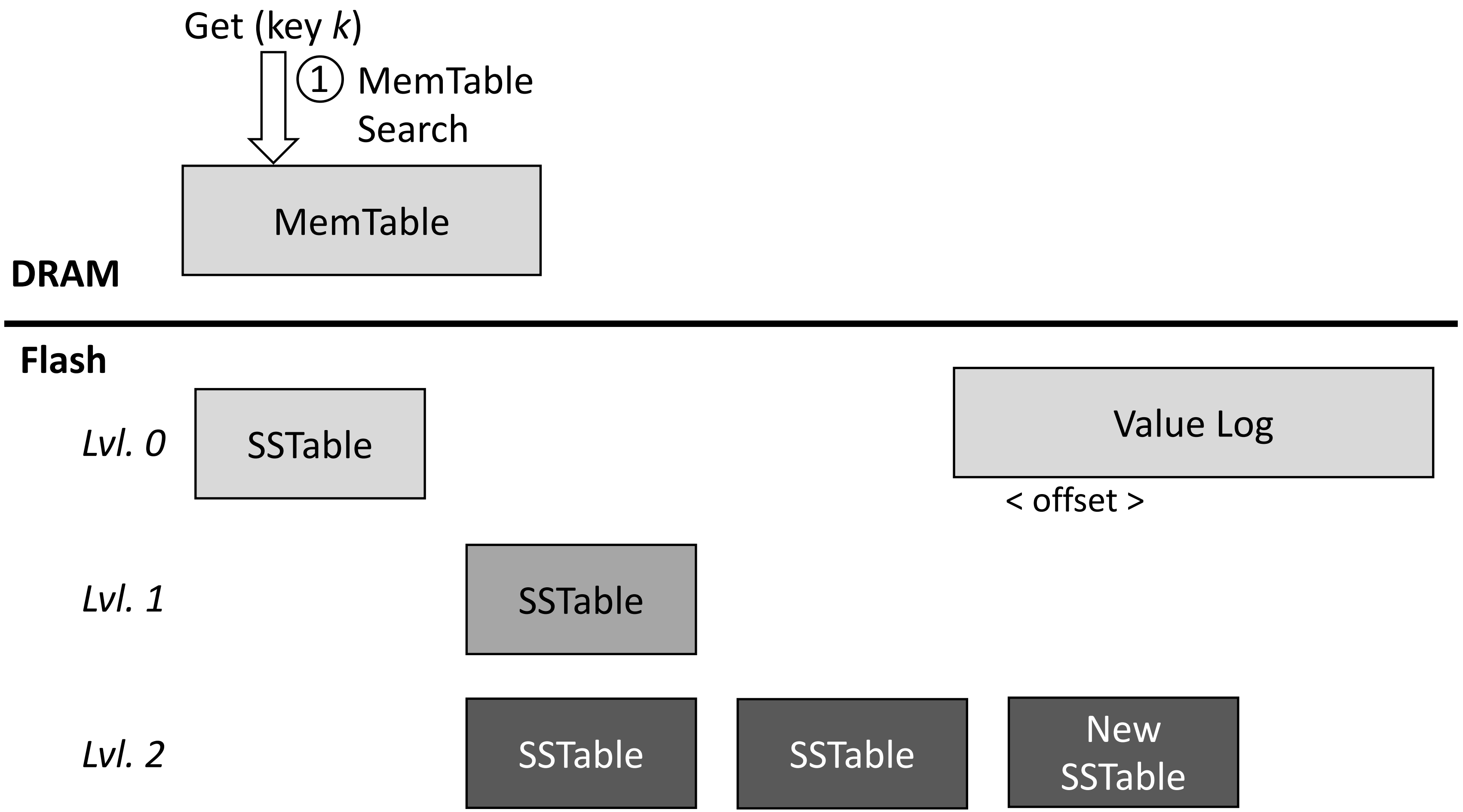


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

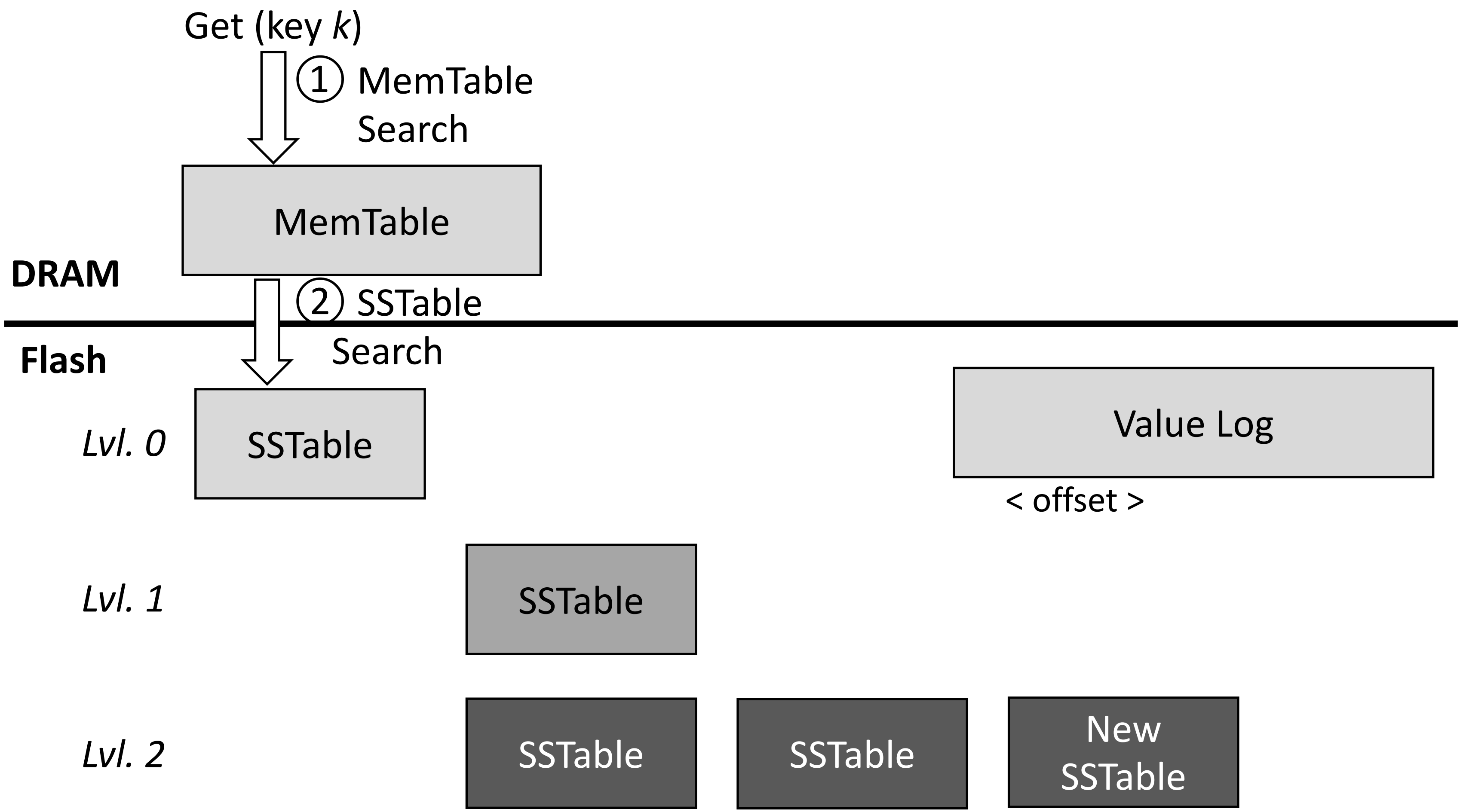


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

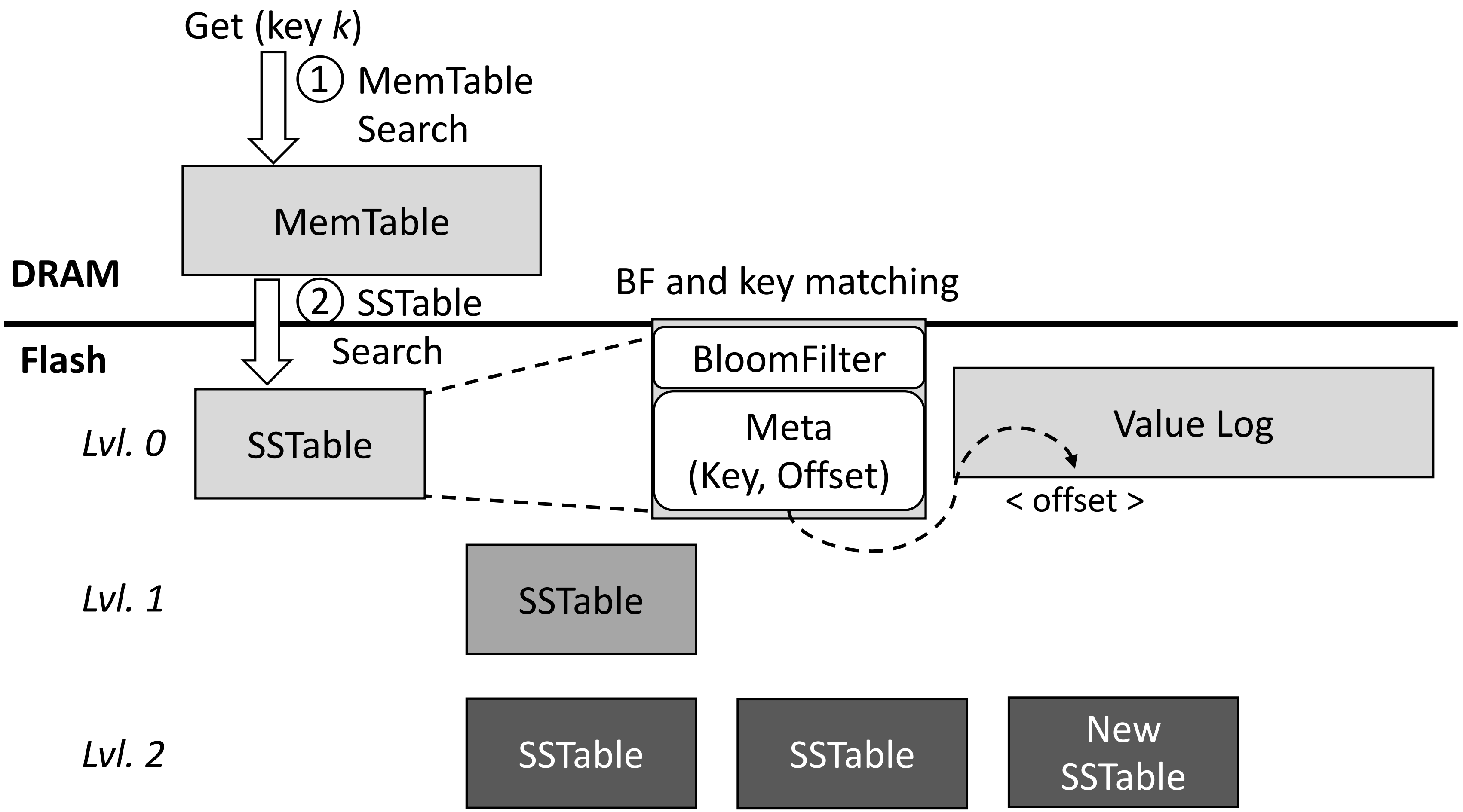
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Log-Structured Merge-tree (LSM-tree) in KVSSD

- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].

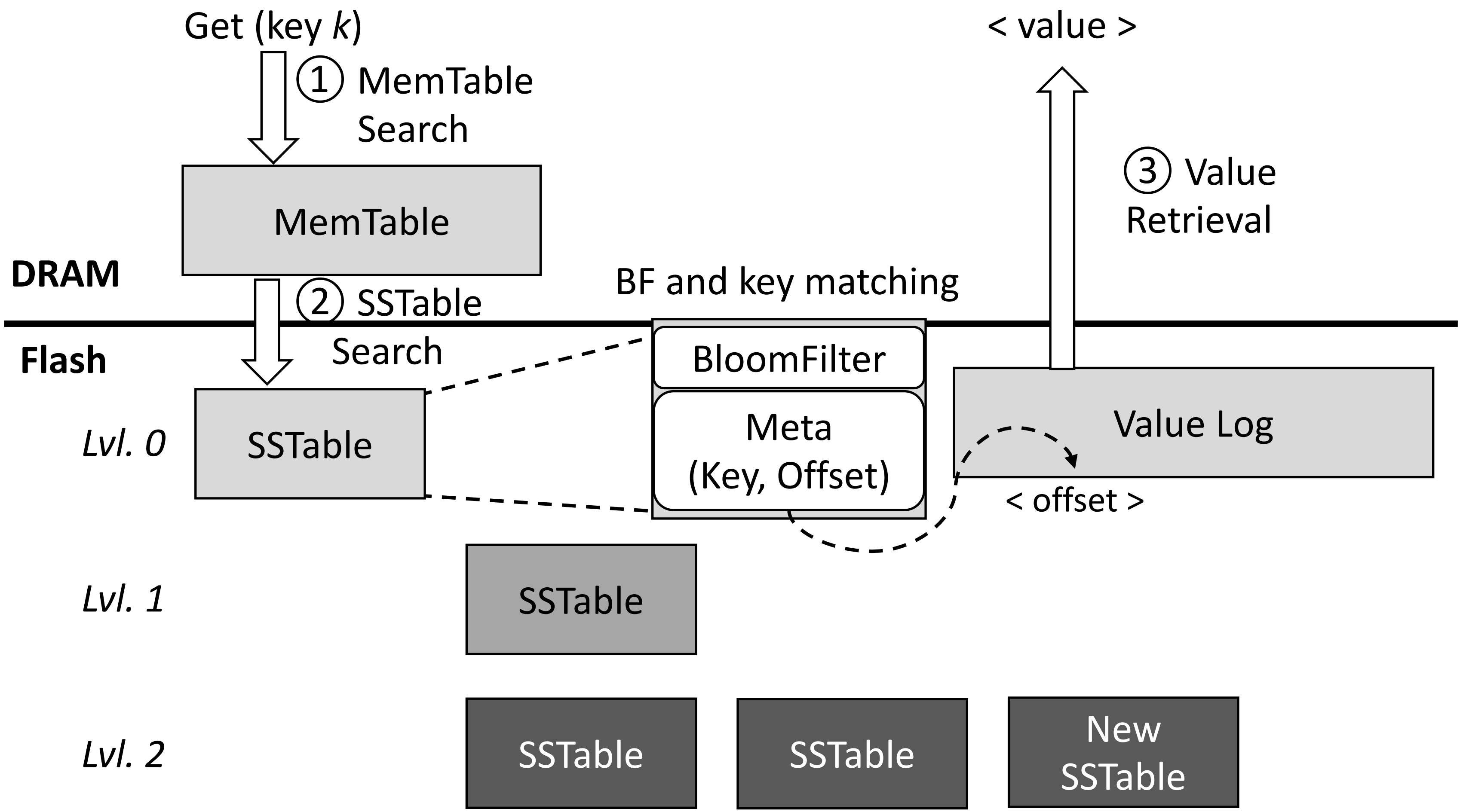


[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

SOGANG UNIVERSITY

Log-Structured Merge-tree (LSM-tree) in KVSSD

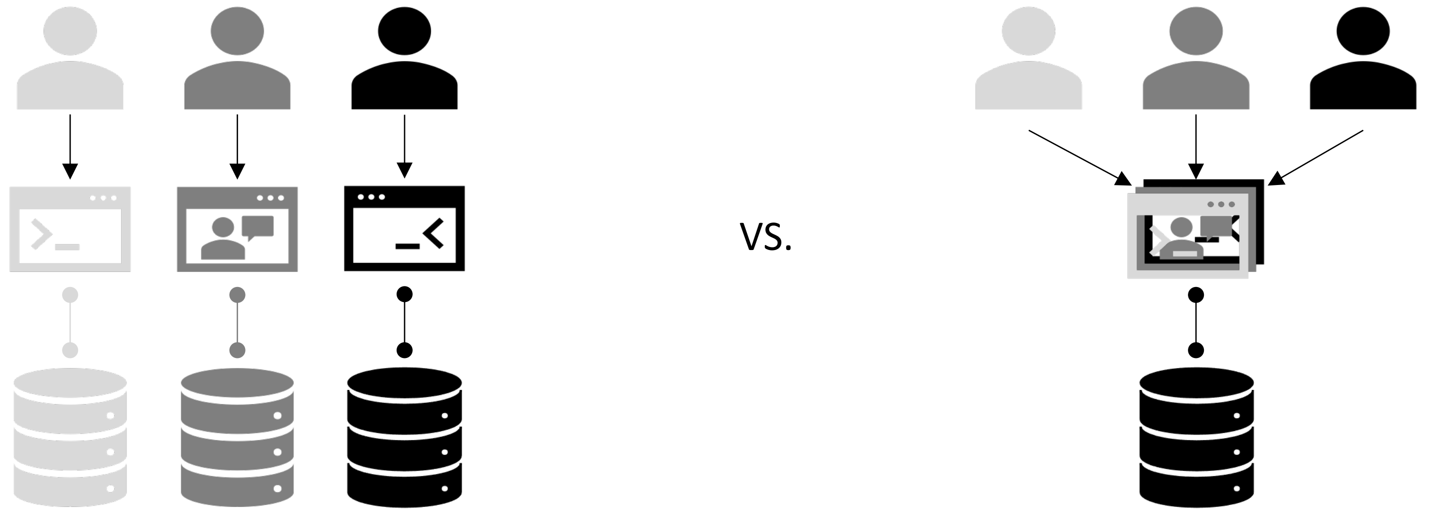
- Several KVSSDs[1, 2] are implemented based on key-value separated LSM-tree indexing structure[3].



[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.
[2] PinK: High-speed In-storage Key-Value Store with Bounded Tails, USENIX ATC, 2020.
[3] WiscKey: Separating Keys from Values in SSD-Conscious Storage, USENIX FAST, 2016.

Problems of the current LSM-Tree based KVSSD

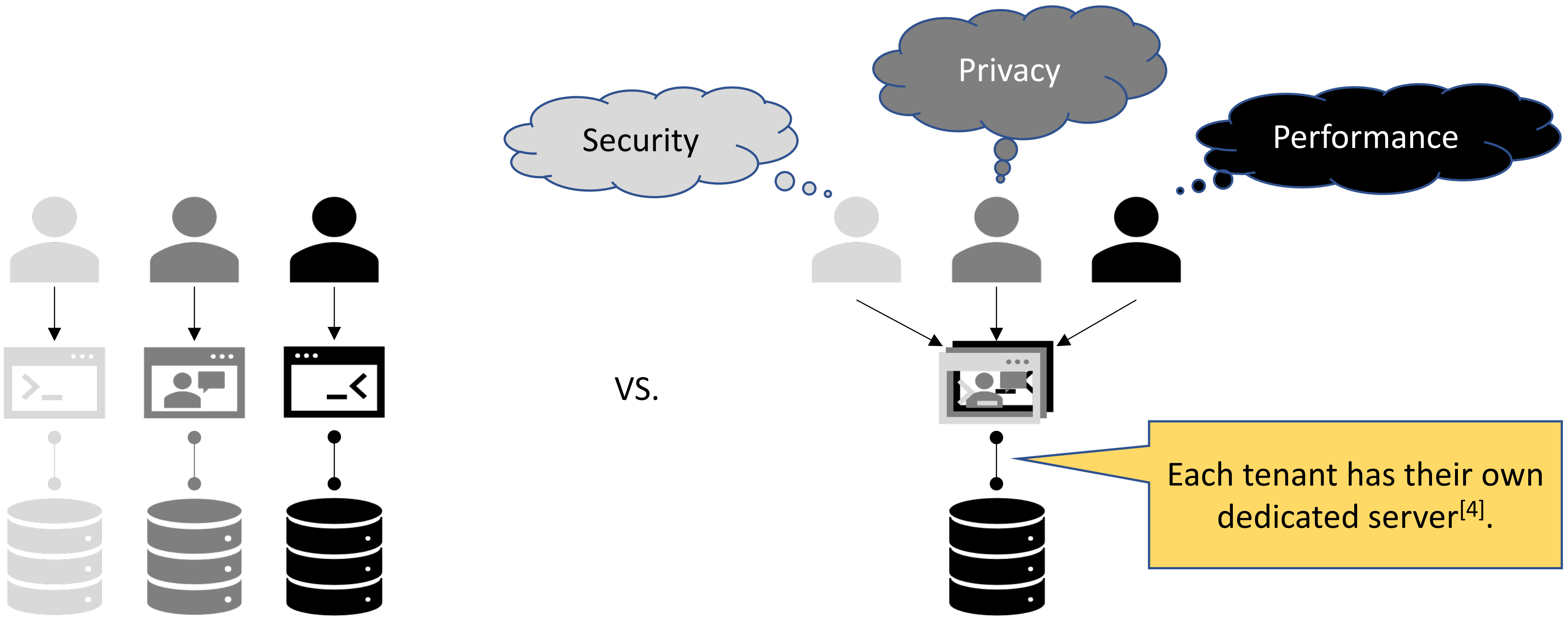
- Problem 1: Lack multi-tenancy and namespace isolation support.
  - Multi-tenancy is an architecture that can host multiple DB instances of tenants on a server.



<Single-tenant vs. Multi-tenant>

Problems of the current LSM-Tree based KVSSD

- Problem 1: Lack multi-tenancy and namespace isolation support.
  - Multi-tenancy is an architecture that can host multiple DB instances of tenants on a server.

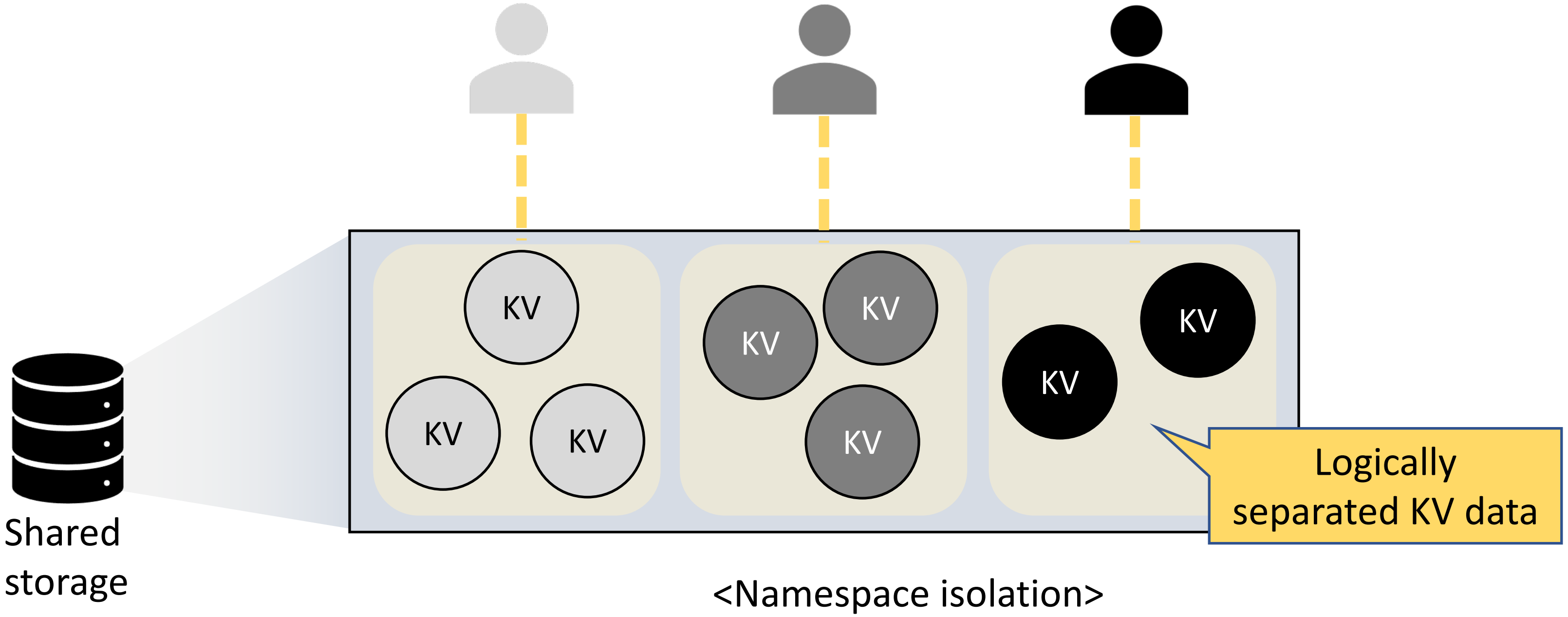


<Single-tenant vs. Multi-tenant>

[4] pClock: An Arrival Curve Based Approach for QoS Guarantees in Shared Storage Systems, ACM SIGMETRICS, 2007.

Problems of the current LSM-Tree based KVSSD

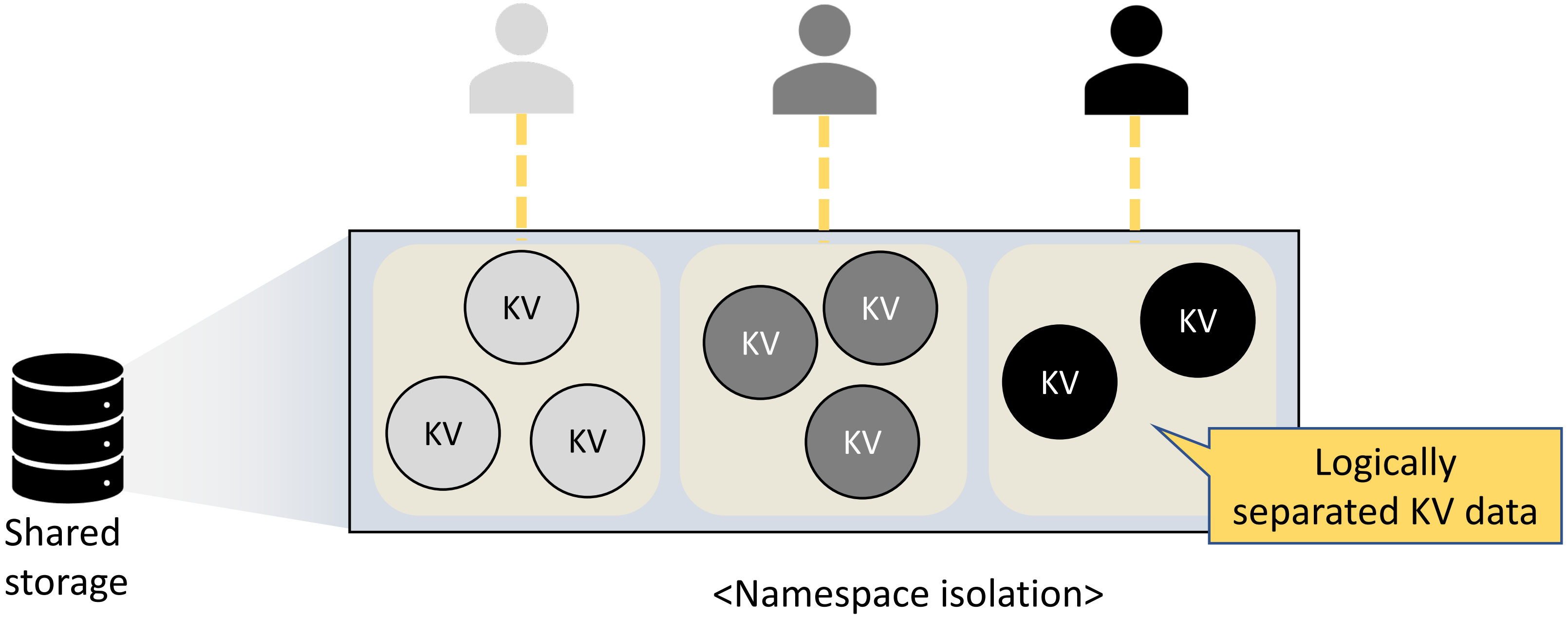
- Problem 1: Lack multi-tenancy and namespace isolation support.
  - To this end, namespace isolation is supported.



<Namespace isolation>

Problems of the current LSM-Tree based KVSSD

- Problem 1: Lack multi-tenancy and namespace isolation support.
  - To this end, namespace isolation is supported.

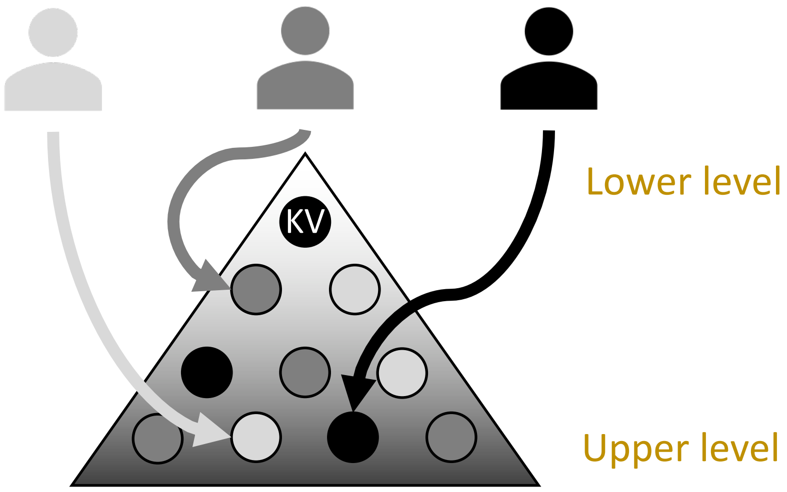


<Namespace isolation>

However, current LSM-tree based KVSSDs lack design and implementation for namespace isolation.

Problems of the current LSM-Tree based KVSSD (Cont.)

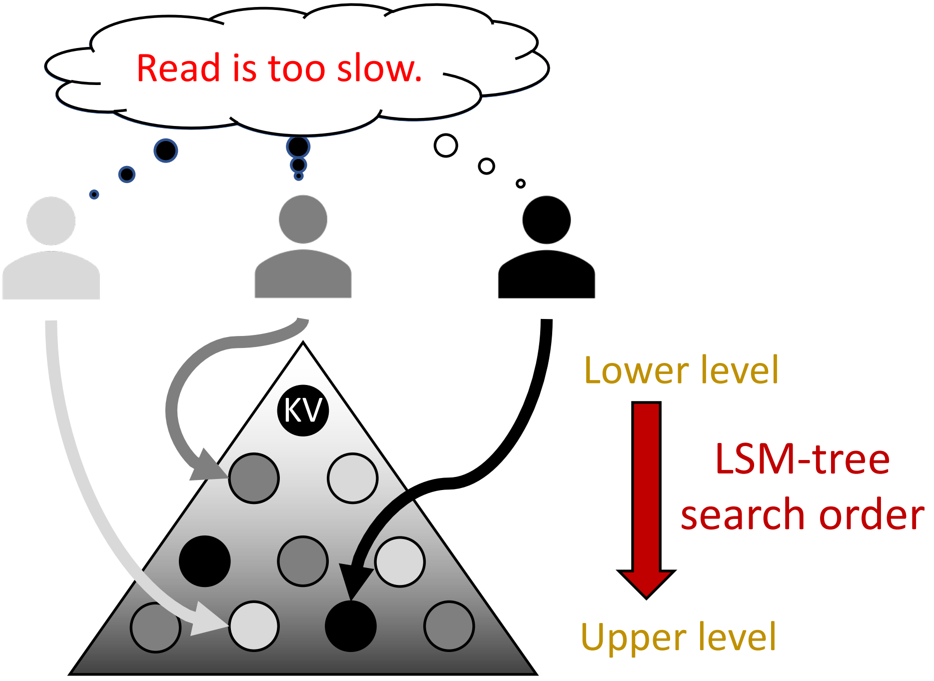
- Problem 2: Limited per-tenant read performance
  - Multiple KV data of tenants are still managed by a global-shared single LSM-tree.



<Global-shared LSM-tree>

Problems of the current LSM-Tree based KVSSD (Cont.)

- Problem 2: Limited per-tenant read performance
  - Multiple KV data of tenants are still managed by <u>a global-shared single LSM-tree</u>.

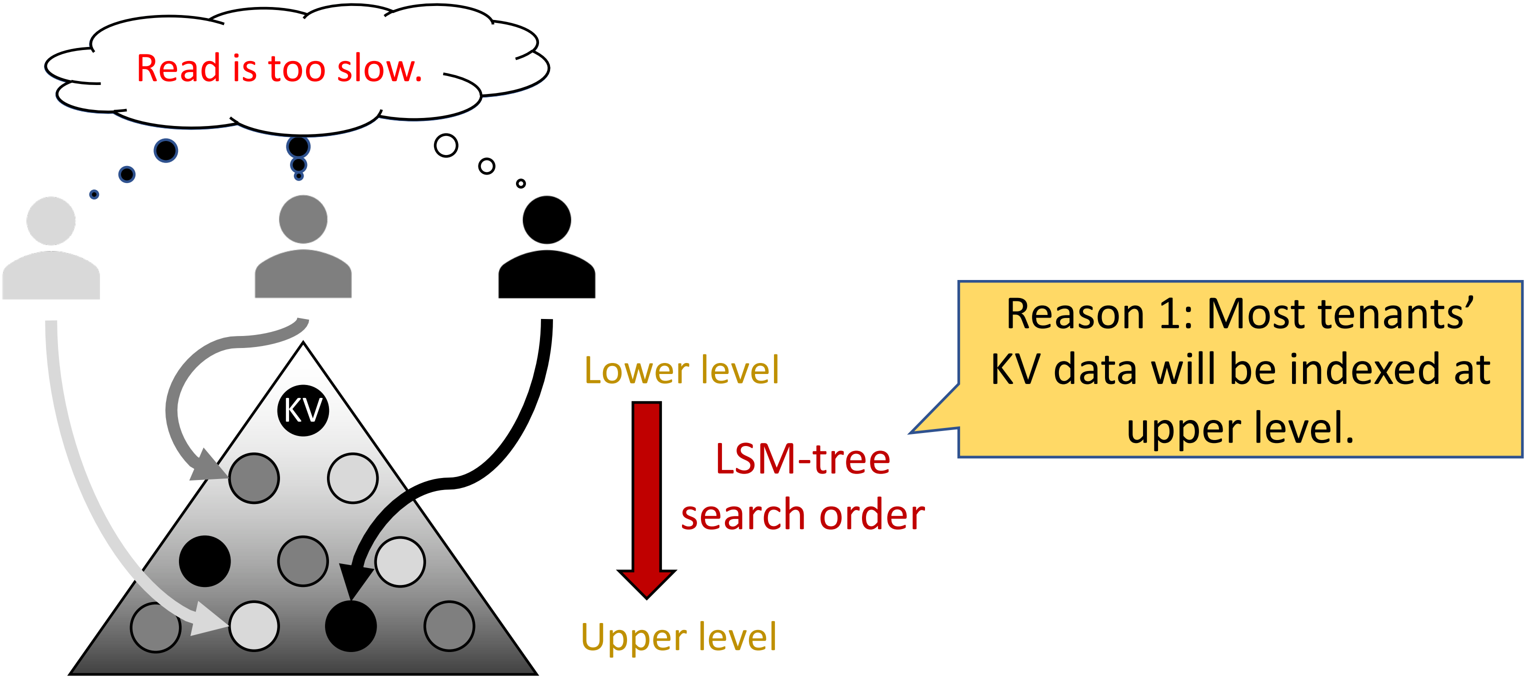


<Global-shared LSM-tree>

SOGANG UNIVERSITY

Problems of the current LSM-Tree based KVSSD (Cont.)

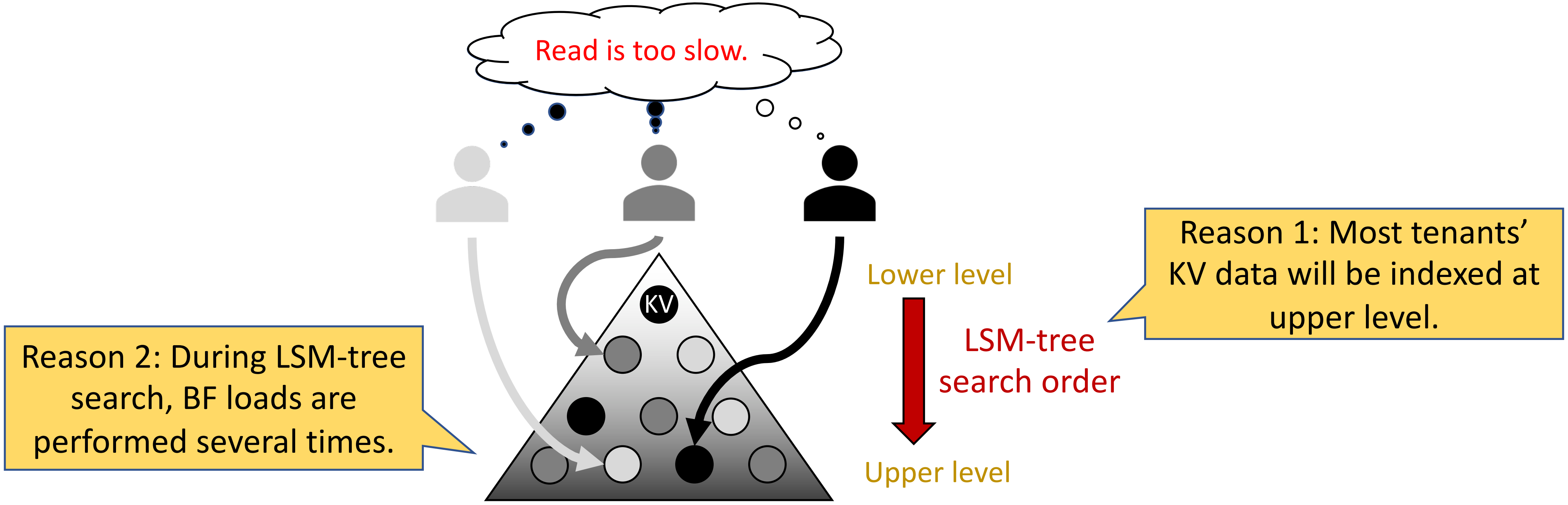
- Problem 2: Limited per-tenant read performance
  - Multiple KV data of tenants are still managed by a global-shared single LSM-tree.



<Global-shared LSM-tree>

Problems of the current LSM-Tree based KVSSD (Cont.)

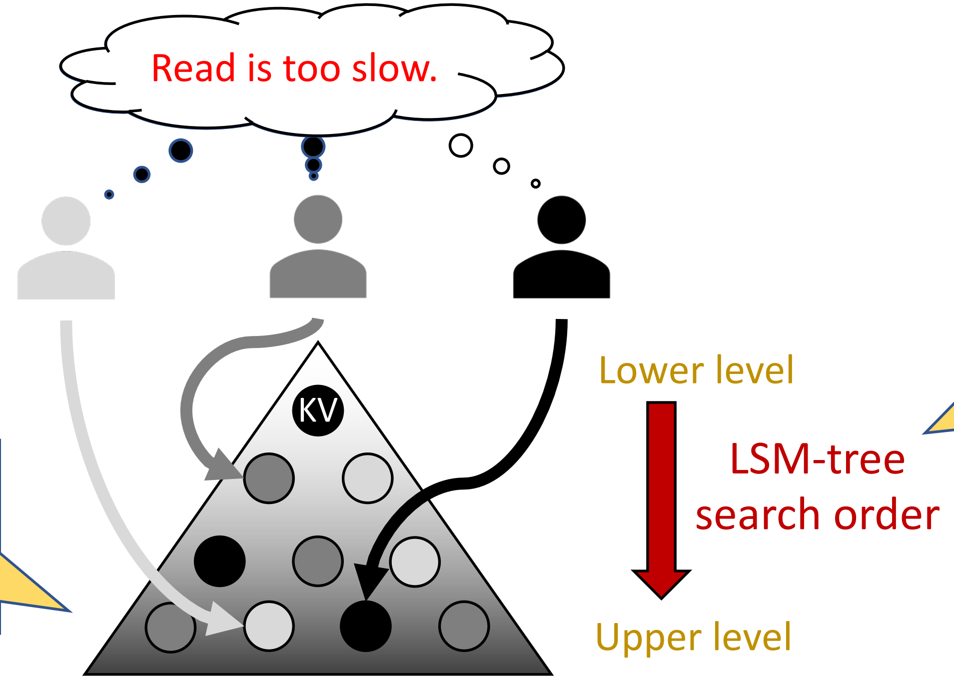
- Problem 2: Limited per-tenant read performance
  - Multiple KV data of tenants are still managed by a global-shared single LSM-tree.



<Global-shared LSM-tree>

Problems of the current LSM-Tree based KVSSD (Cont.)

- Problem 2: Limited per-tenant read performance
  - Multiple KV data of tenants are still managed by a global-shared single LSM-tree.



Reason 1: Most tenants' KV data will be indexed at upper level.

Reason 2: During LSM-tree search, BF loads are performed several times.

Current LSM-tree based KVSSDs have difficulty in providing the promised read performance that storage device can provide.

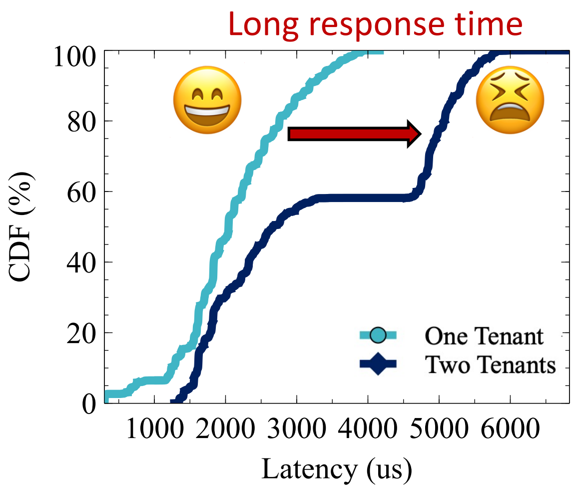
Motivation Experiment

- Configuration
  - iLSM-SSD[1], recent LSM-tree based KVSSD.
  - Key size: 8B, Value size: 1KB.
  - # of KV requests issued (per tenant): 1M.
- KV tenant <u>Read</u> Scenarios
  - (1): When only tenant *x*'s KV data occupies a LSM-tree.
  - (2): When LSM-tree is shared by tenant *x*'s and *y*'s own KV data at the same time.

[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

Motivation Experiment

- Configuration
  - iLSM-SSD[1], recent LSM-tree based KVSSD.
  - Key size: 8B, Value size: 1KB.
  - # of KV requests issued (per tenant): 1M.
- KV tenant Read Scenarios
  - (1): When only tenant *x*'s KV data occupies a LSM-tree.
  - (2): When LSM-tree is shared by tenant *x*'s and *y*'s own KV data at the same time.
- Result & Analysis (from the tenant *x*'s perspective)
  - Response time: (1) << (2).

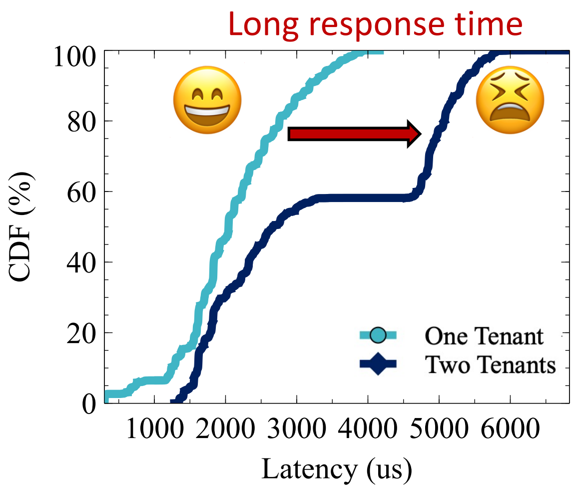


<Response time for read>

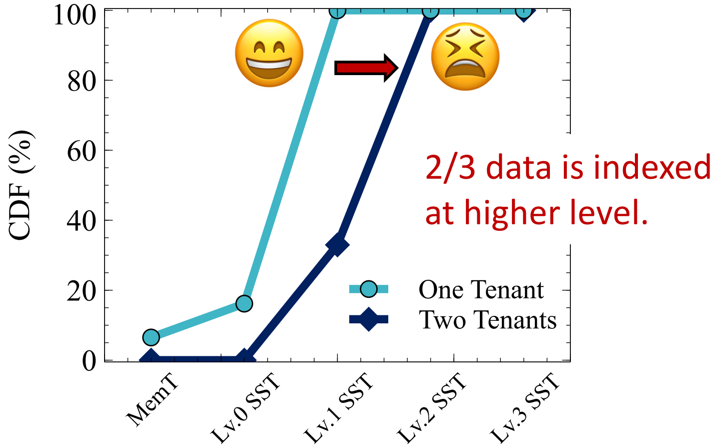
[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

Motivation Experiment

- Configuration
  - iLSM-SSD[1], recent LSM-tree based KVSSD.
  - Key size: 8B, Value size: 1KB.
  - # of KV requests issued (per tenant): 1M.
- KV tenant Read Scenarios
  - (1): When only tenant *x*'s KV data occupies a LSM-tree.
  - (2): When LSM-tree is shared by tenant *x*'s and *y*'s own KV data at the same time.
- Result & Analysis (from the tenant *x*'s perspective)
  - Response time: (1) << (2).
  - Reason 1: 67% of tenant *x*'s KV data are indexed at L_2.
  - Reason 2: # of BF loads are increased by 77%.



<Response time for read>



<Level of accessed data>

[1] iLSM-SSD: An Intelligent LSM-tree Based Key-Value SSD for Data Analytics, MASCOTS, 2019.

SOGANG UNIVERSITY

Design Goals

- Therefore, we have the following design goals for multi-tenant KVSSD.

  (1) Multi-tenant KVSSD supports namespace isolation.

  (2) Multi-tenant KVSSD minimizes read performance overhead for performance isolation.

Design Goals

- Therefore, we have the following design goals for multi-tenant KVSSD.

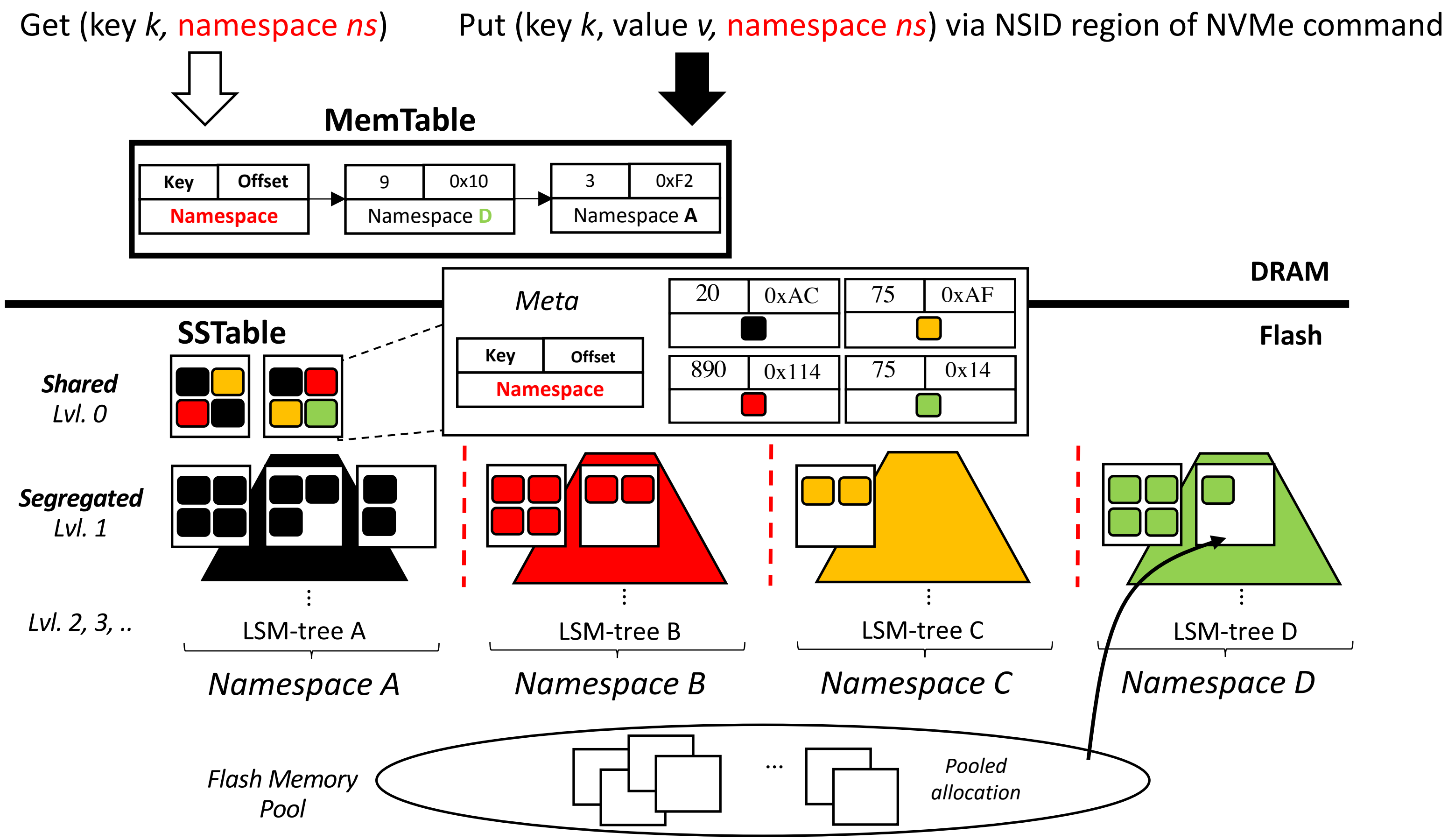
(1) Multi-tenant KVSSD supports namespace isolation.

(2) Multi-tenant KVSSD minimizes read performance overhead for performance isolation.

We propose a multi-tenant Iso-KVSSD, satisfying these two goals.

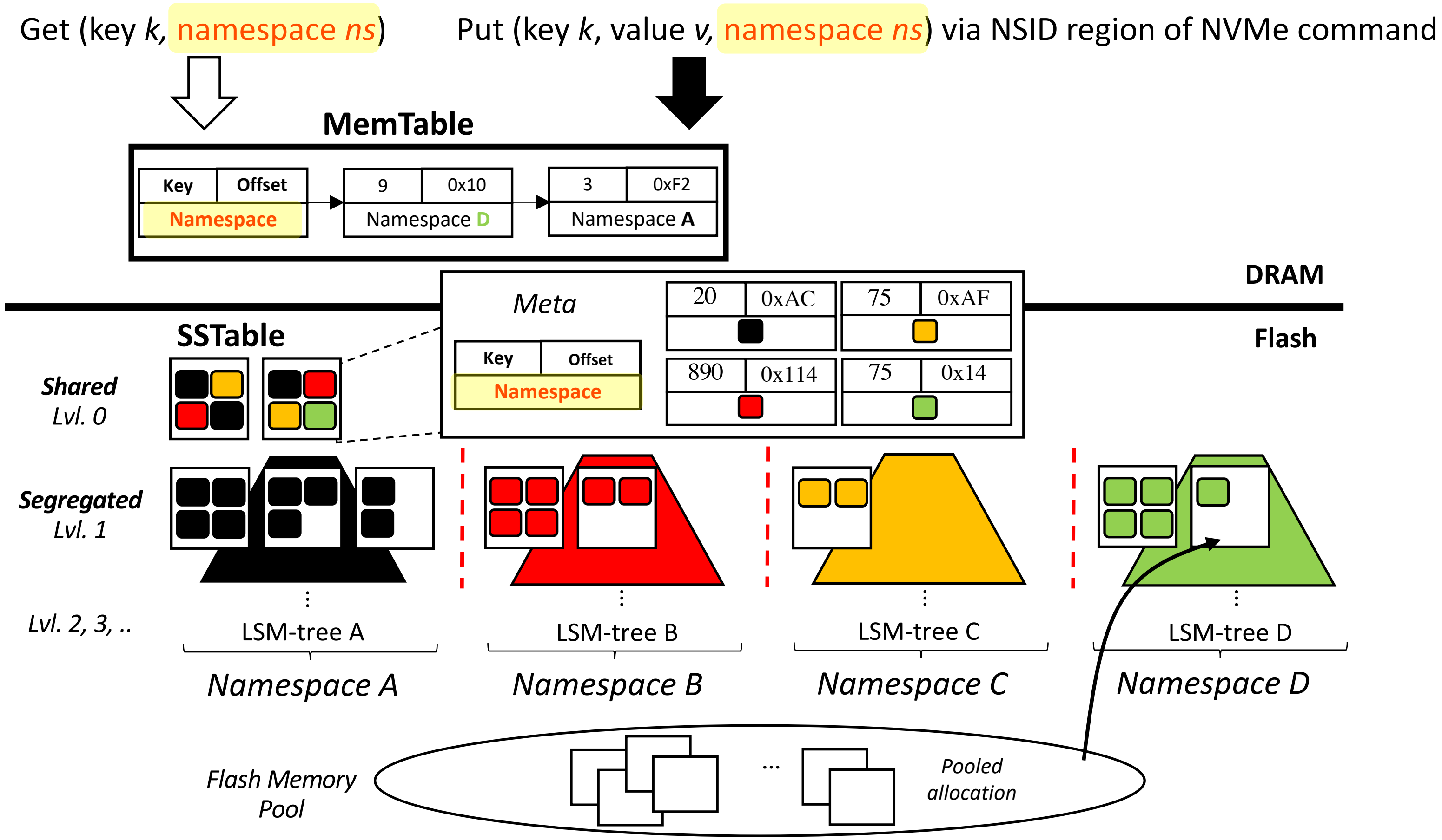
Per-namespace dedicated LSM-tree

- Iso-KVSSD employs per-namespace dedicated LSM-tree design.



Per-namespace dedicated LSM-tree

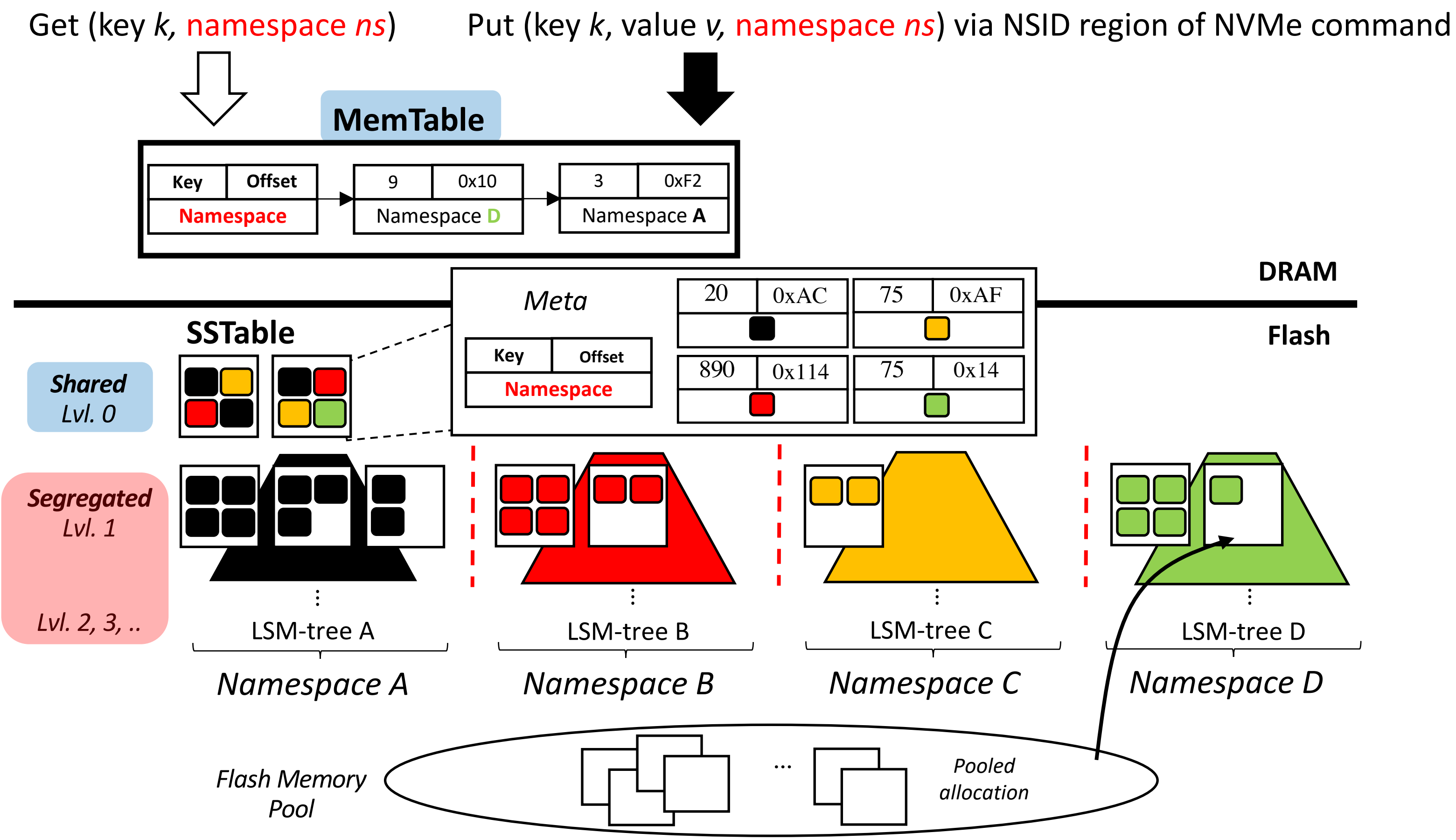
- Iso-KVSSD employs per-namespace dedicated LSM-tree design.



SOGANG UNIVERSITY

Per-namespace dedicated LSM-tree

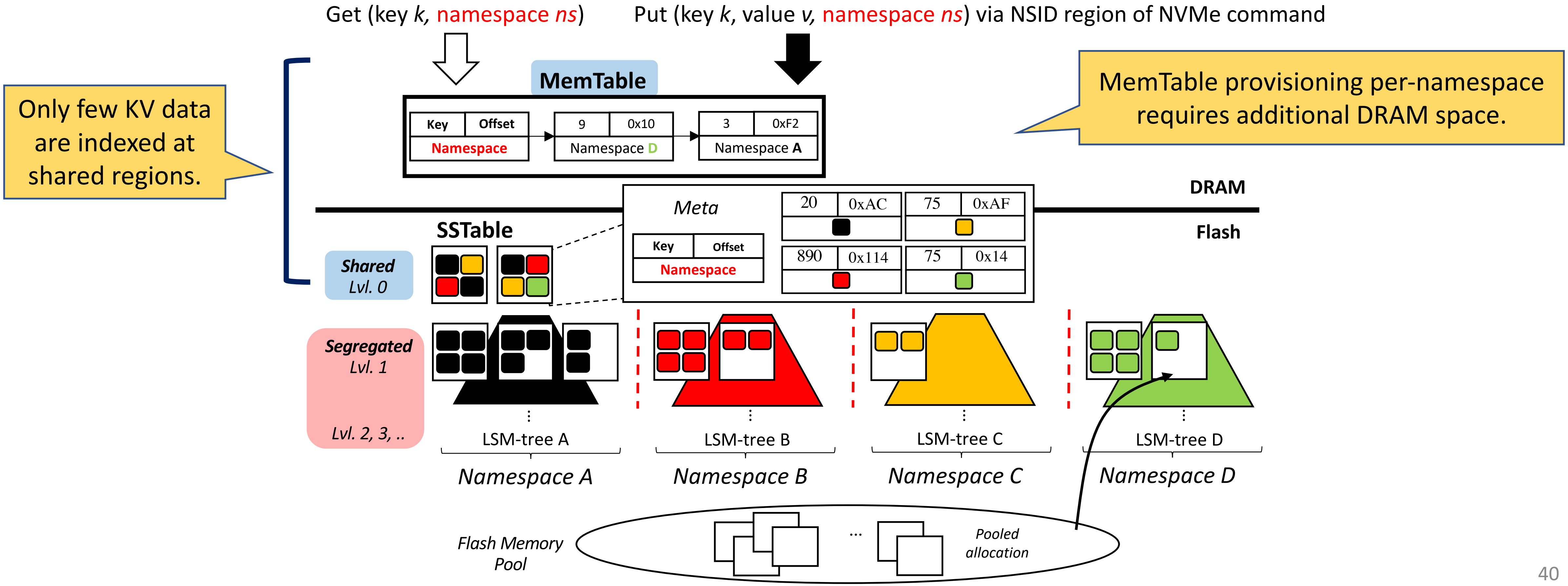
- Iso-KVSSD employs per-namespace dedicated LSM-tree design.



SOGANG UNIVERSITY

Per-namespace dedicated LSM-tree

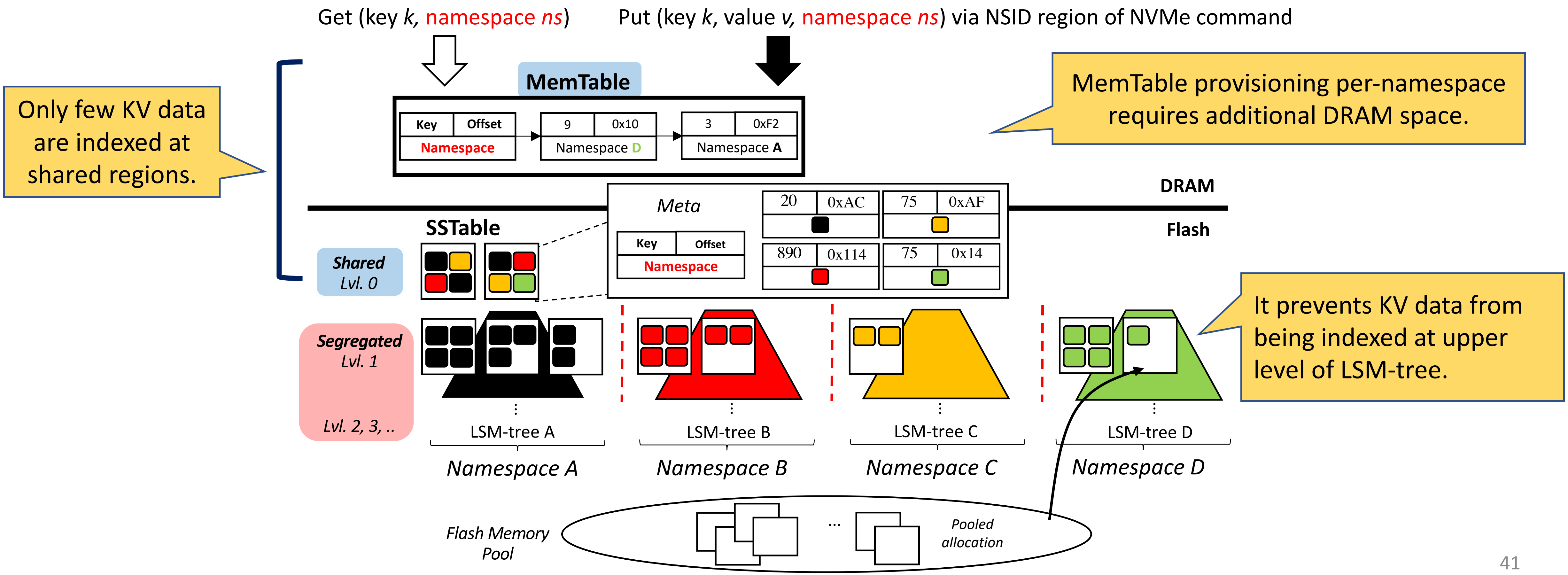
- Iso-KVSSD employs per-namespace dedicated LSM-tree design.



SOGANG UNIVERSITY

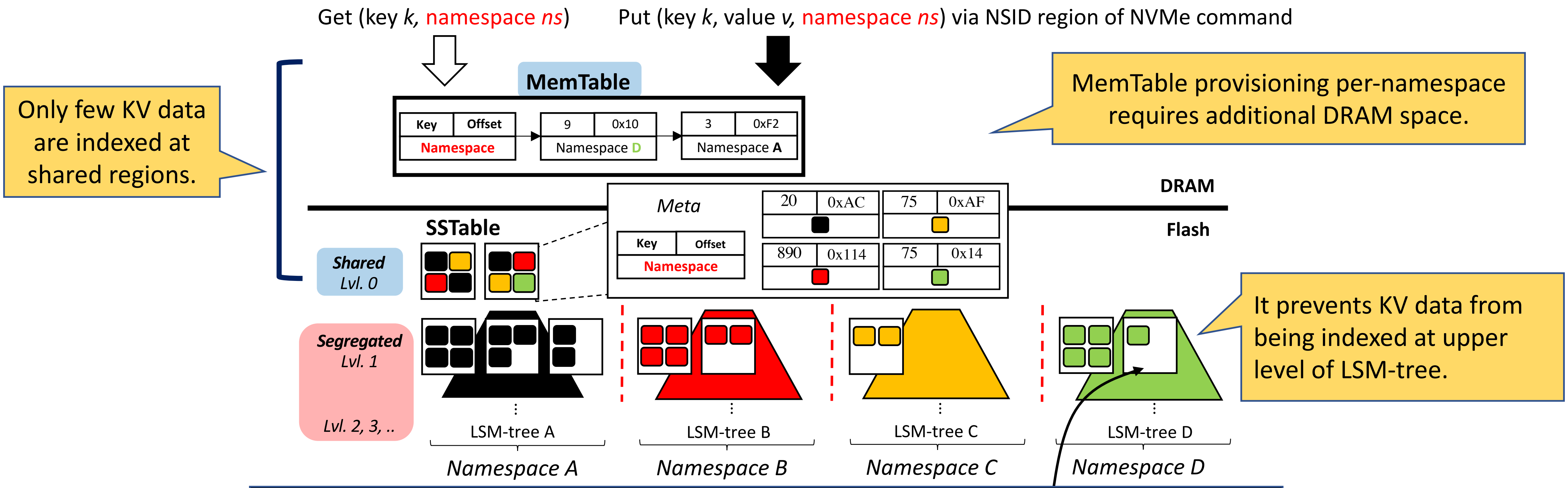
Per-namespace dedicated LSM-tree

- Iso-KVSSD employs per-namespace dedicated LSM-tree design.



Per-namespace dedicated LSM-tree

- Iso-KVSSD employs per-namespace dedicated LSM-tree design.

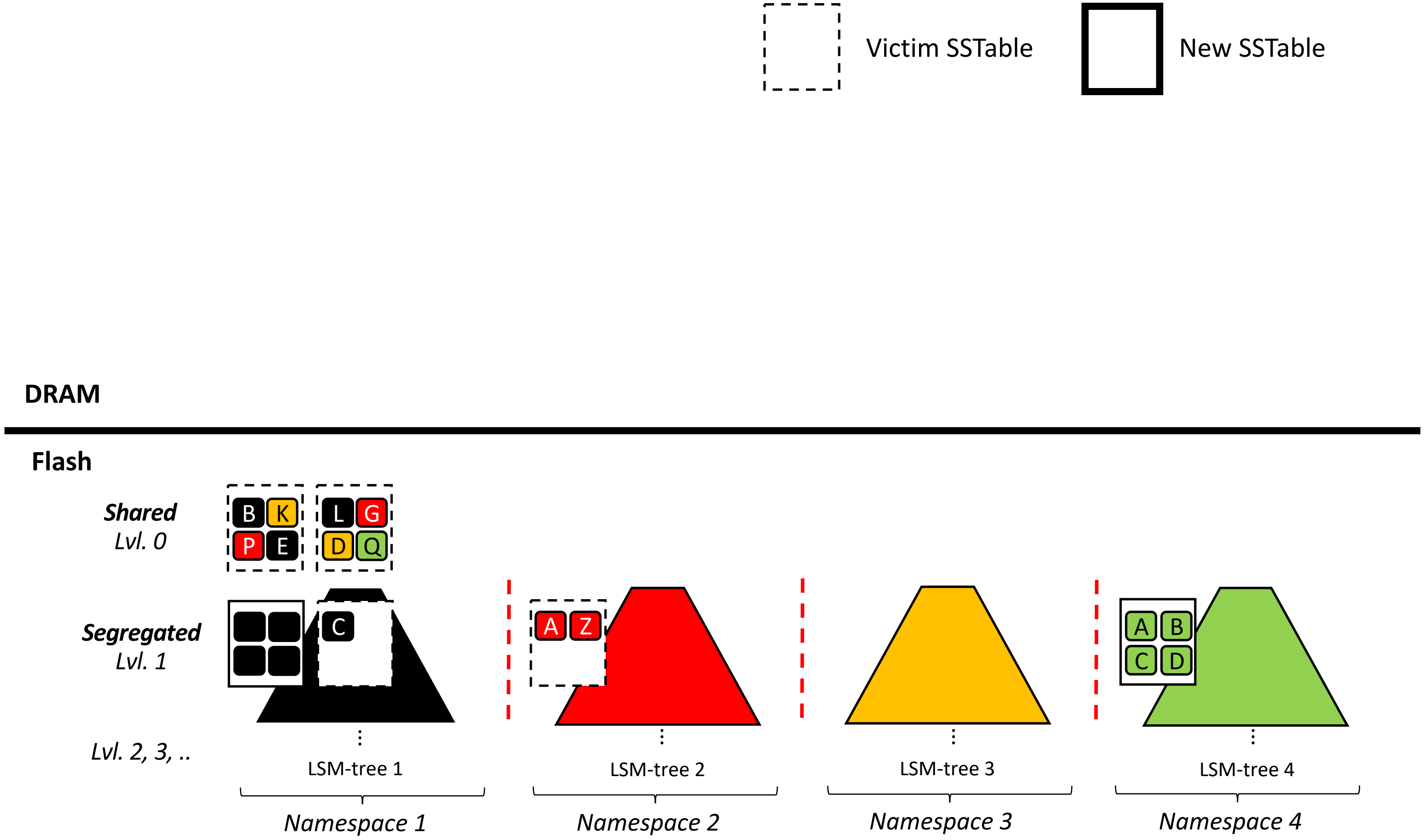


- Iso-KVSSD controls access based on user's namespace information.
- Per-namespace LSM-tree design reduces KV data's access latency.

SOGANG UNIVERSITY

Namespace Isolation Mechanism

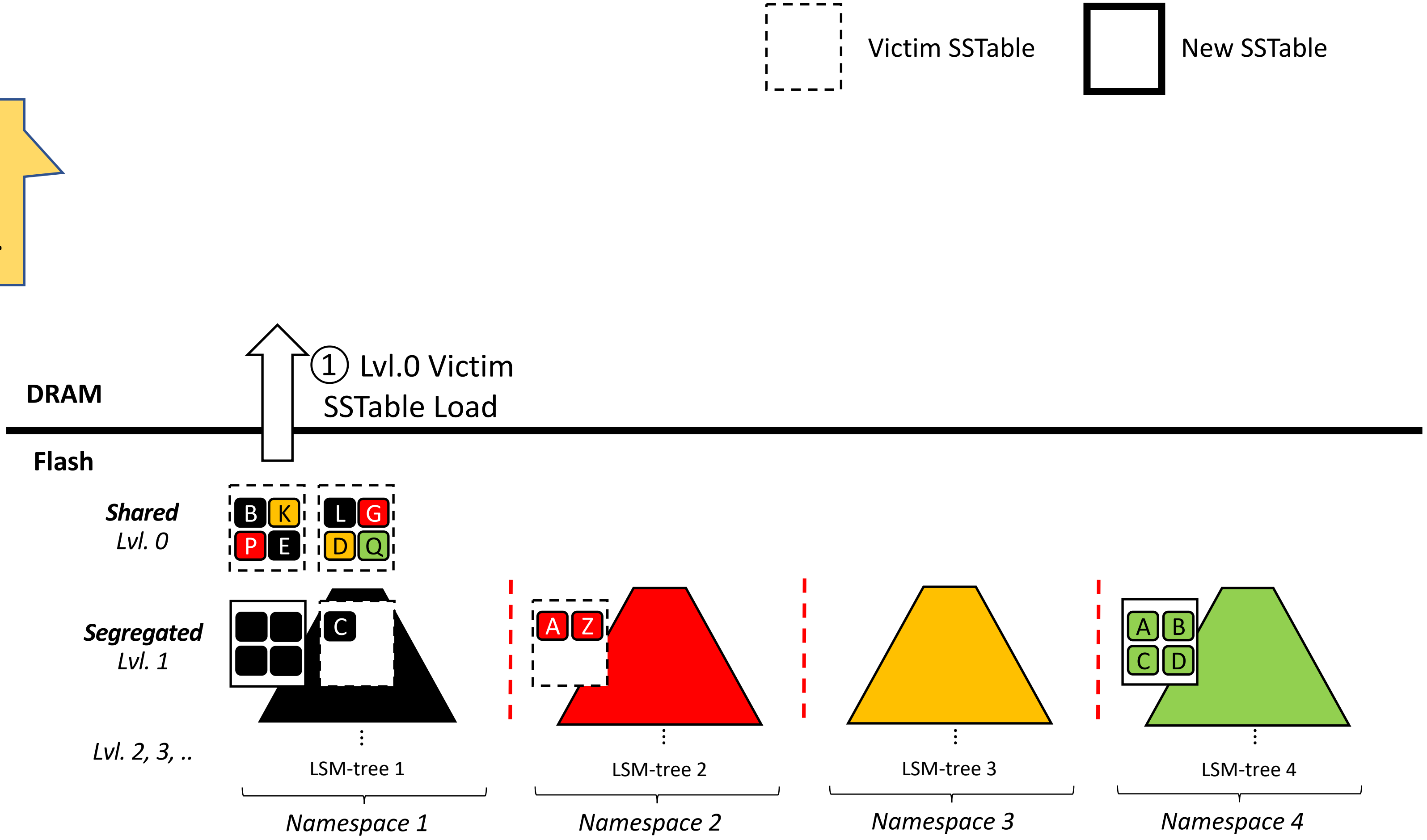
- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.



Namespace Isolation Mechanism

- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.

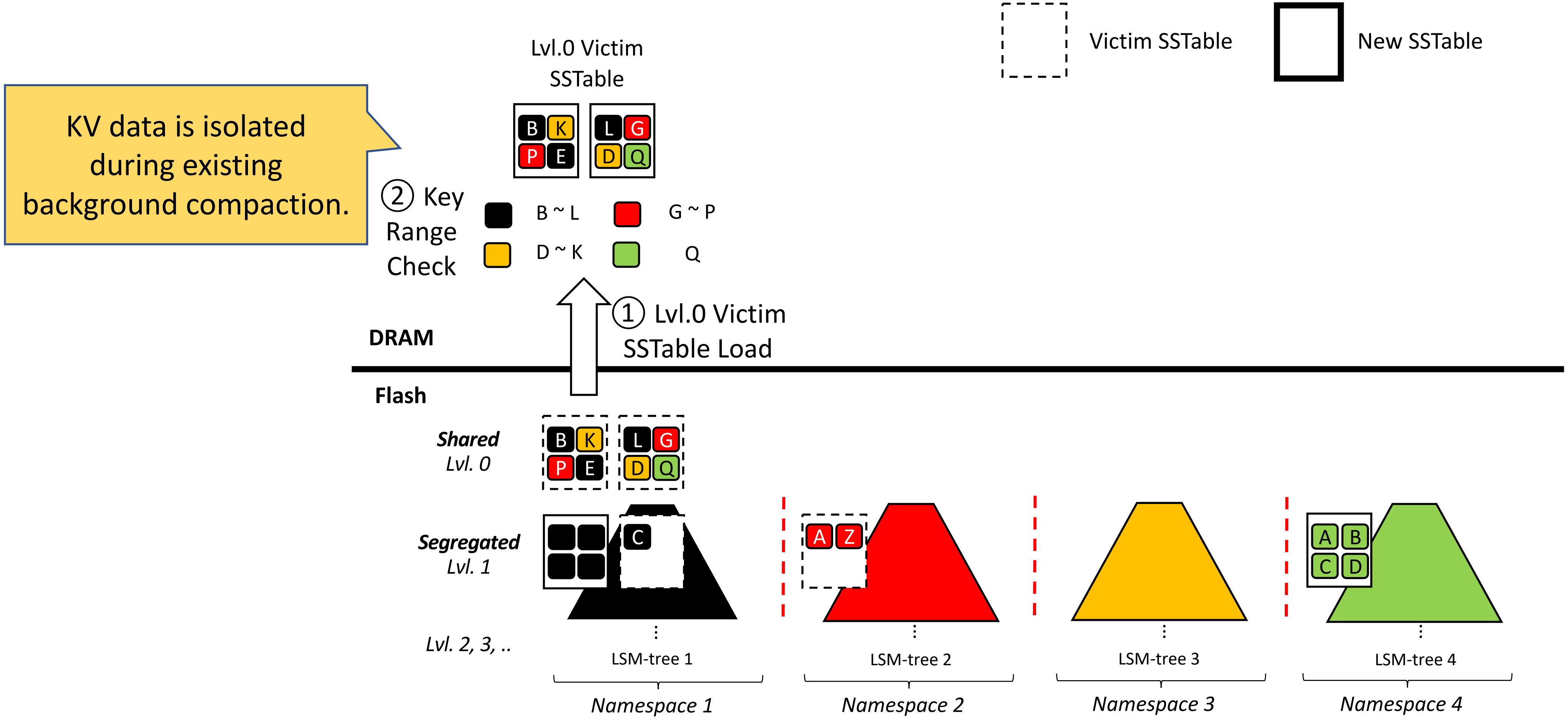
KV data is isolated during existing background compaction.



SOGANG UNIVERSITY

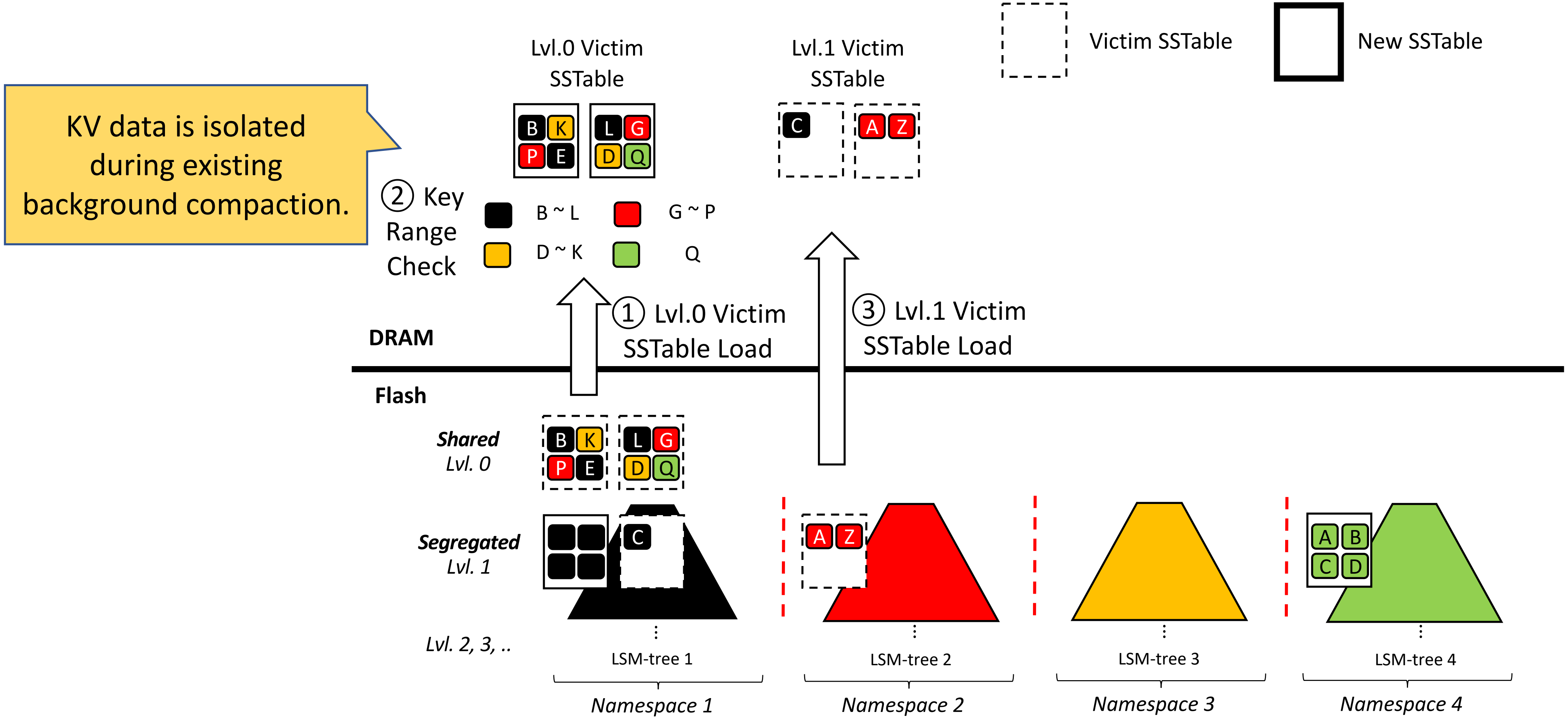
Namespace Isolation Mechanism

- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.



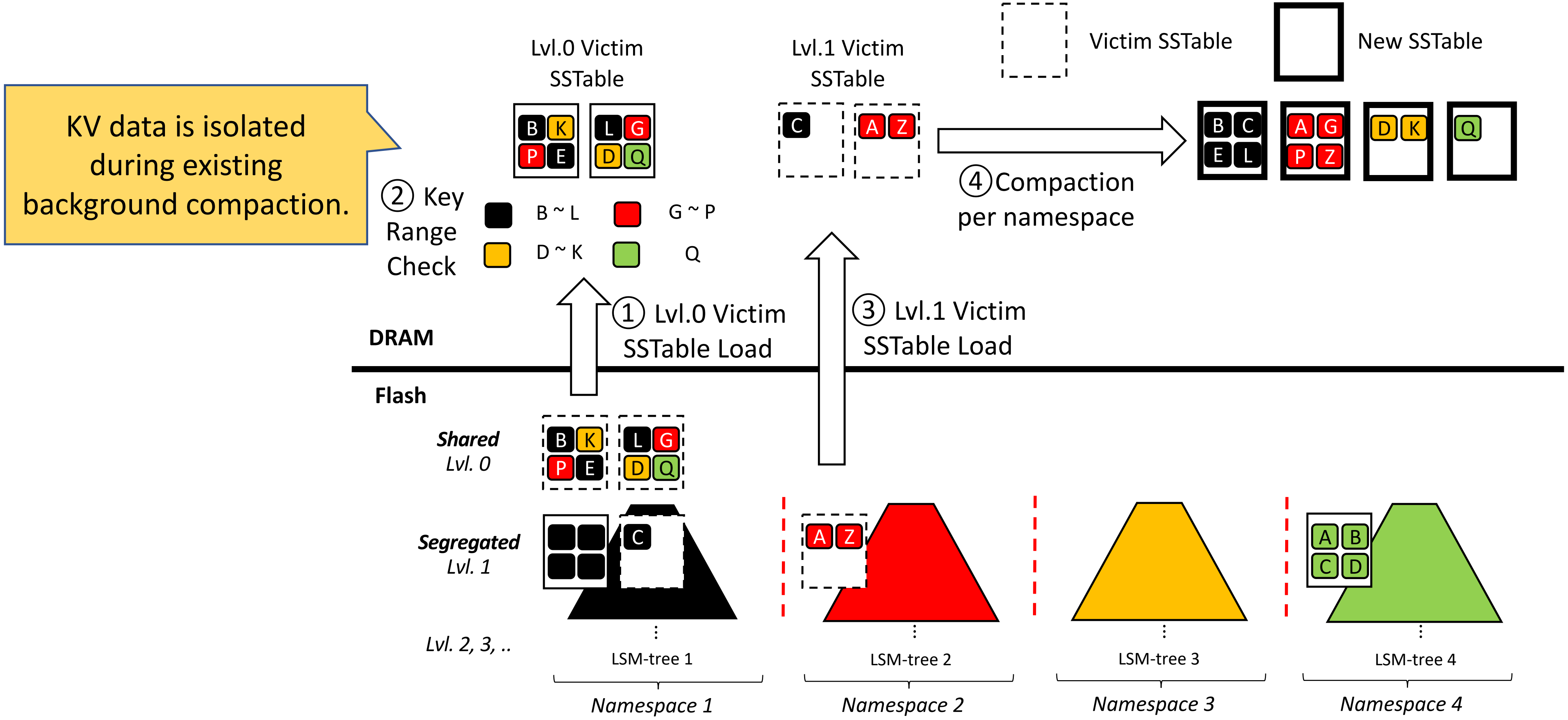
Namespace Isolation Mechanism

- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.



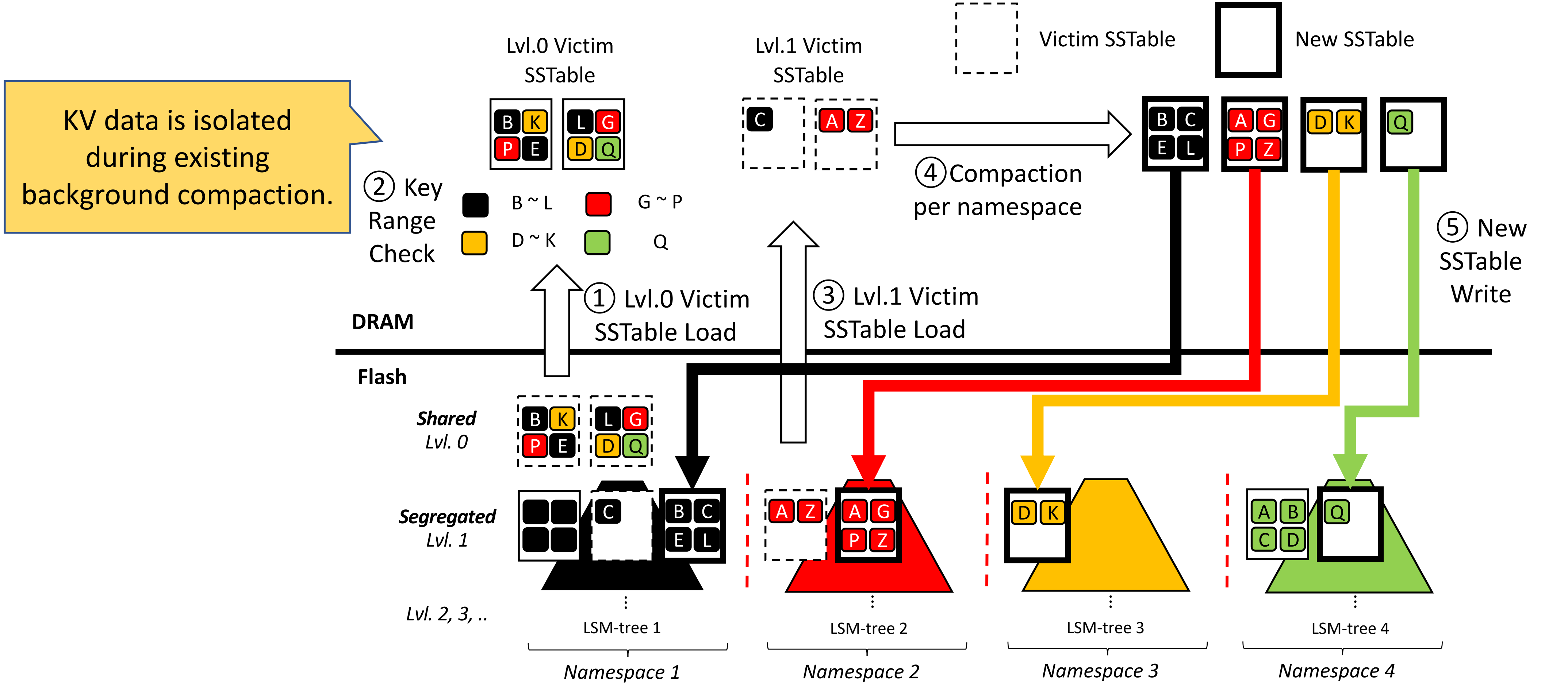
Namespace Isolation Mechanism

- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.



Namespace Isolation Mechanism

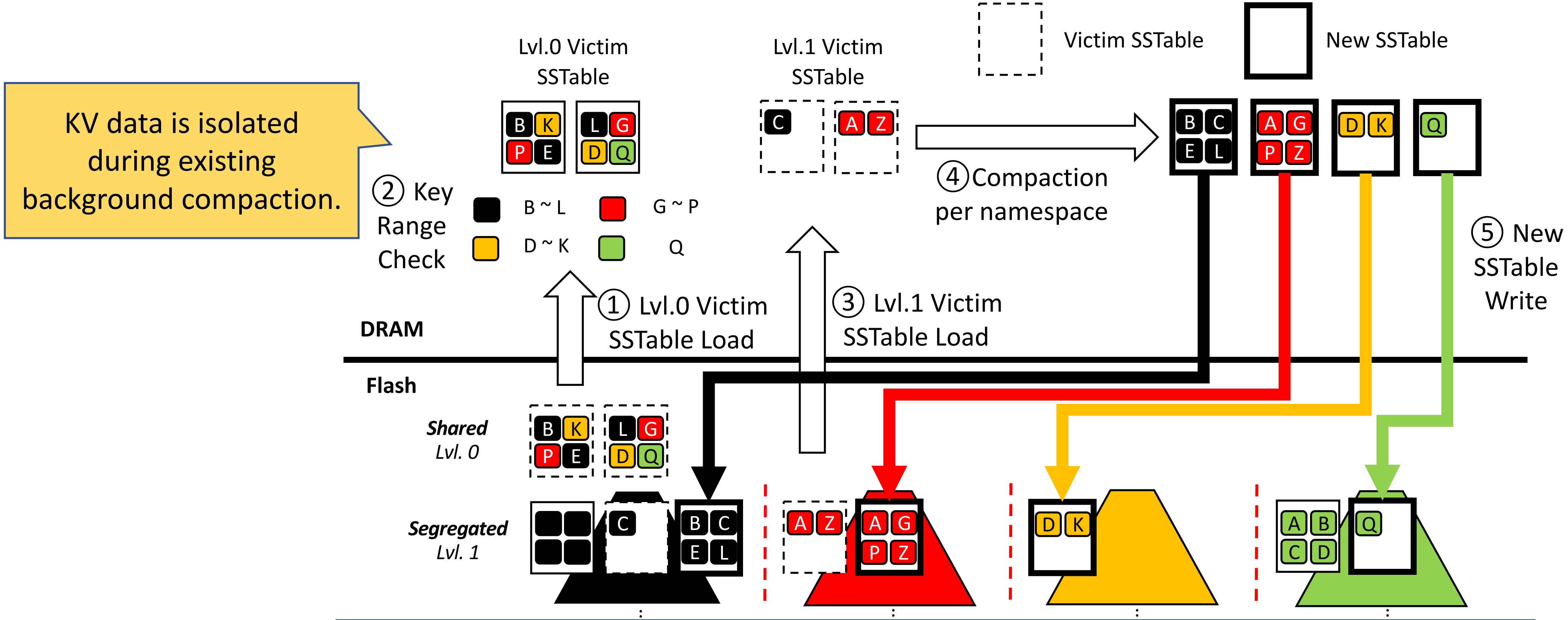
- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.



SOGANG UNIVERSITY

Namespace Isolation Mechanism

- Namespace Isolation Mechanism segregates KV data into per-namespace LSM-trees.

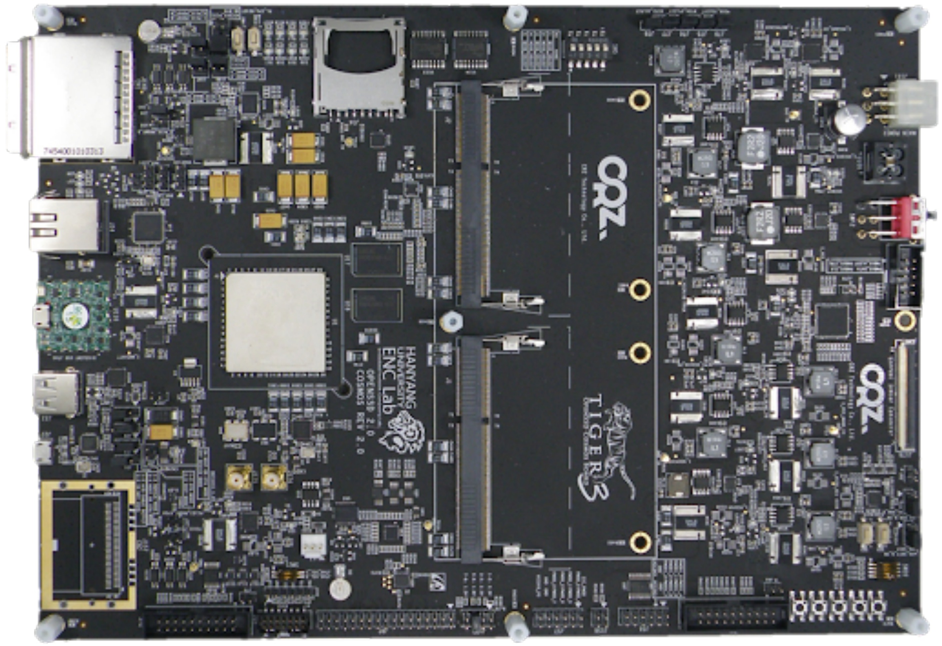


Namespace Isolation mechanism substantially segregates KV data without perceivable overhead.

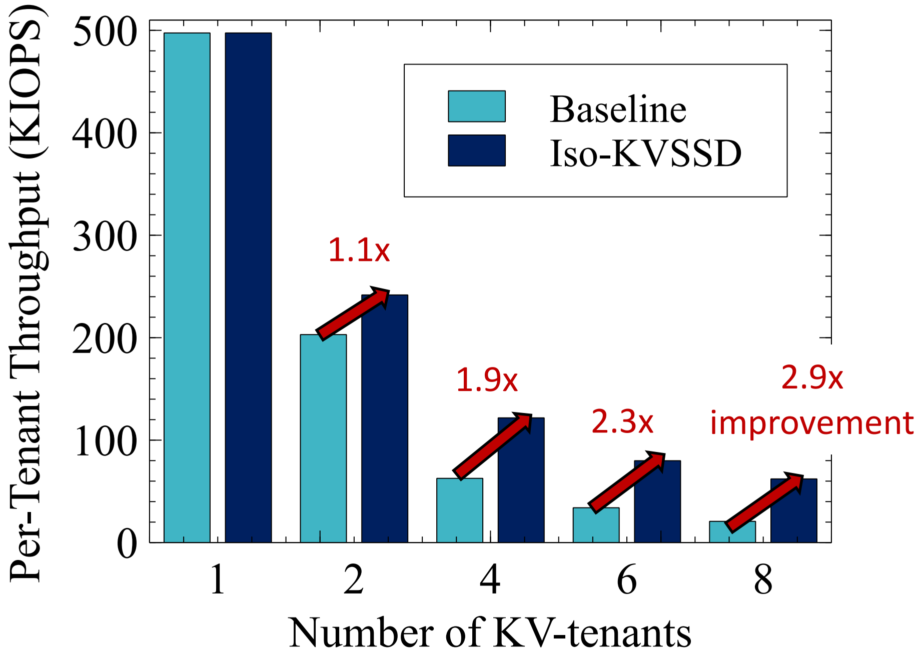
SOGANG UNIVERSITY

Experimental Setup

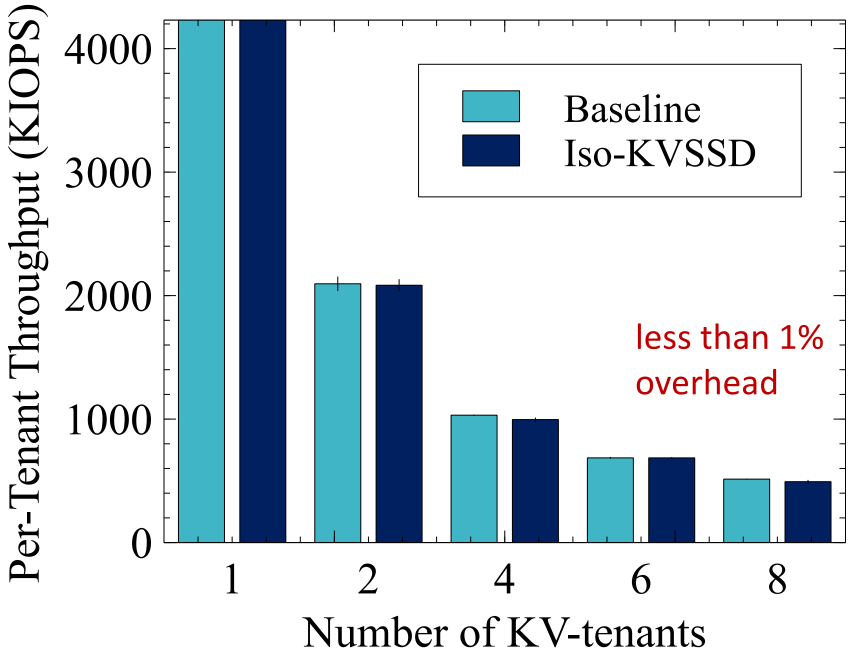
- Prototyped Iso-KVSSD on FPGA-based Cosmos+ OpenSSD.
  - 1TB NAND memory, 1GB DDR3 DRAM, ARM Cortex-A9 processors.
- Configuration
  - Key size: 8B, Value size: 1KB.
  - # of KV requests issued (per tenant): 1M.
- Workloads
  - Put() or Get() only synthetic workloads.
- Comparison
  - Baseline: iLSM-SSD with **global-shared** LSM-tree.
  - Iso-KVSSD: iLSM-SSD with **per-namespace** LSM-tree.



Throughput Comparison



<Throughput Get() only >

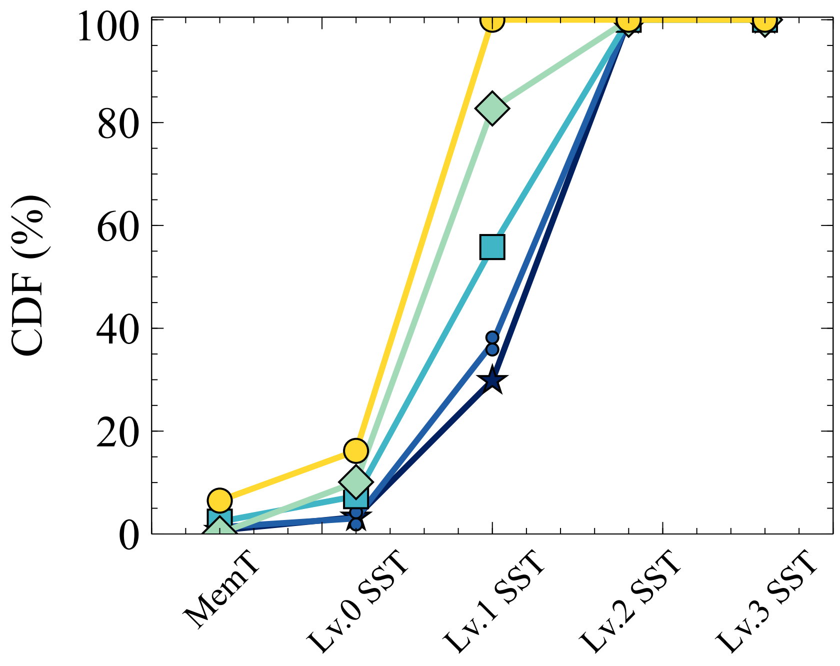


<Throughput Put() only >

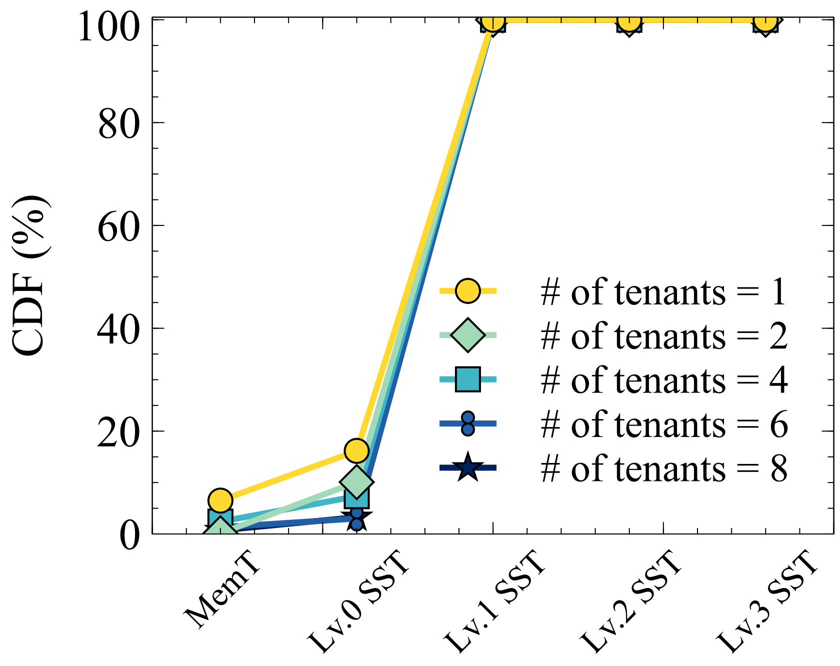
Iso-KVSSD has an average 2.9x higher read throughput than the baseline with negligible write performance overhead.

Impact of Per-namespace LSM-tree: Level Distribution

- Level distribution of where KV data is indexed in the LSM-trees.



<Baseline >



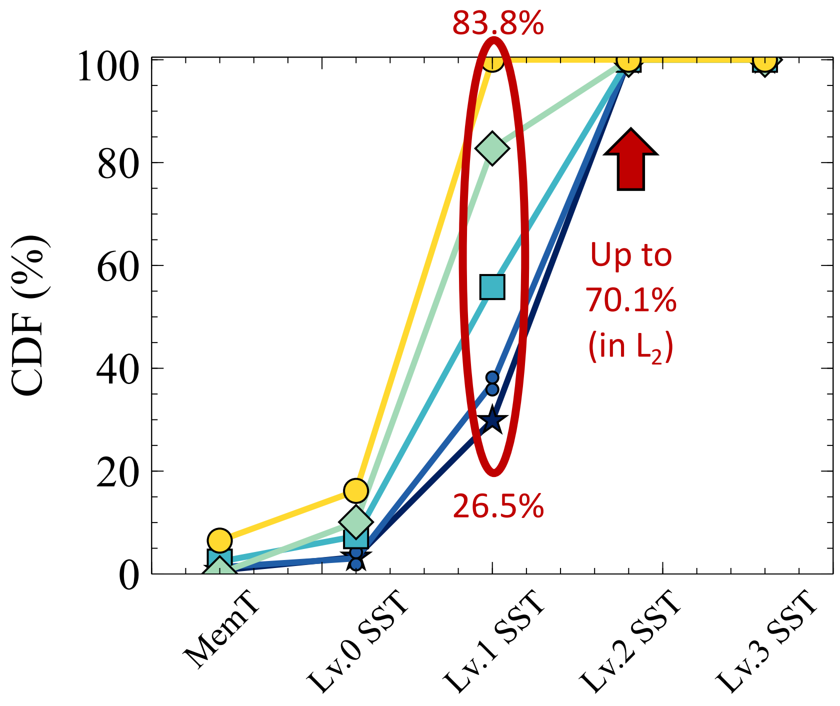
<Iso-KVSSD>

Per-namespace LSM-tree significantly reduces level depth at which KV data is indexed in the LSM-tree.

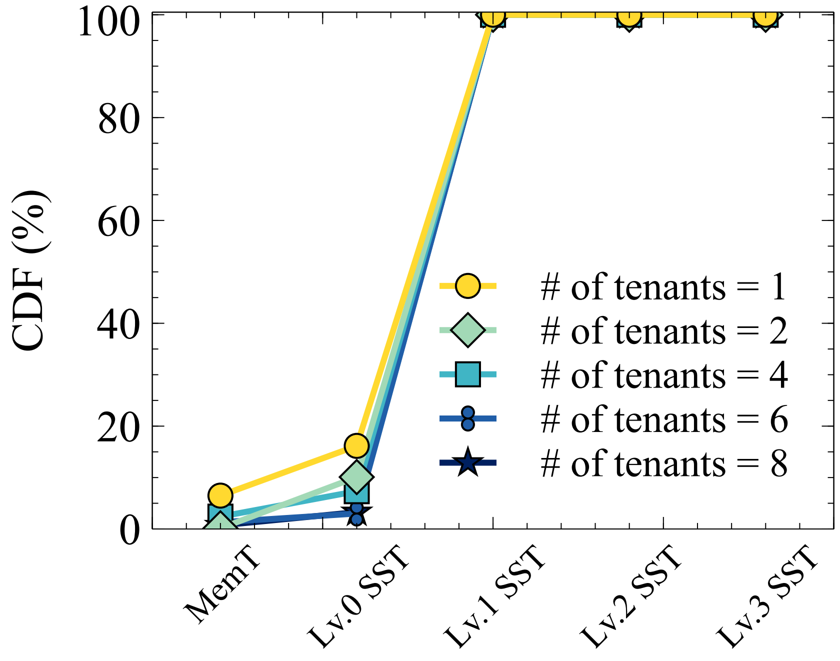
SOGANG UNIVERSITY

Impact of Per-namespace LSM-tree: Level Distribution

- Level distribution of where KV data is indexed in the LSM-trees.



<Baseline >

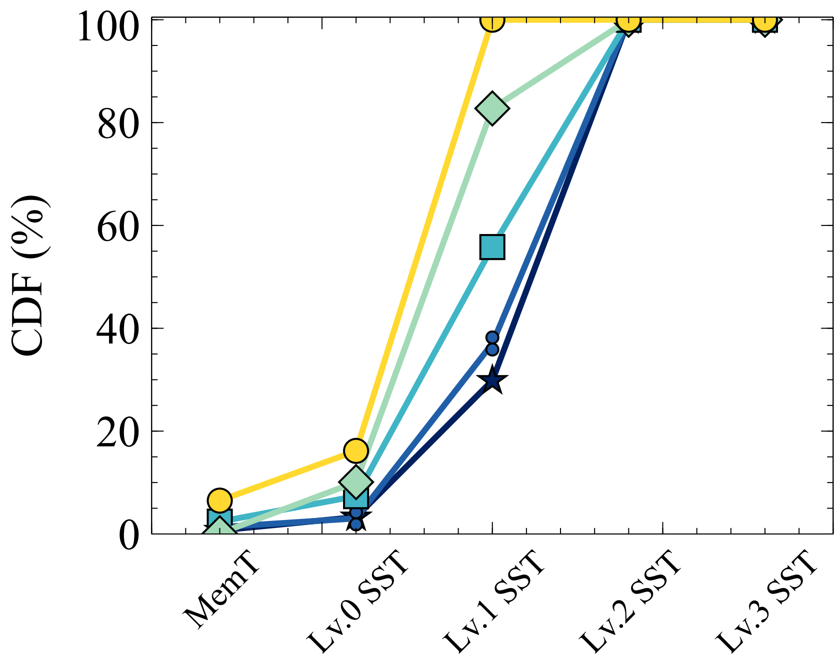


<Iso-KVSSD>

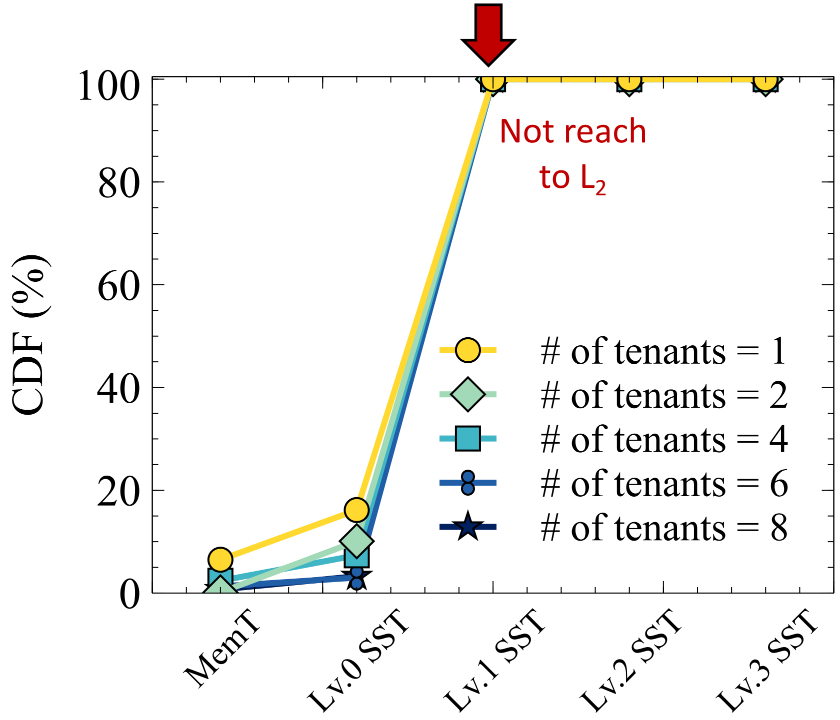
Per-namespace LSM-tree significantly reduces level depth at which KV data is indexed in the LSM-tree.

Impact of Per-namespace LSM-tree: Level Distribution

- Level distribution of where KV data is indexed in the LSM-trees.



<Baseline >

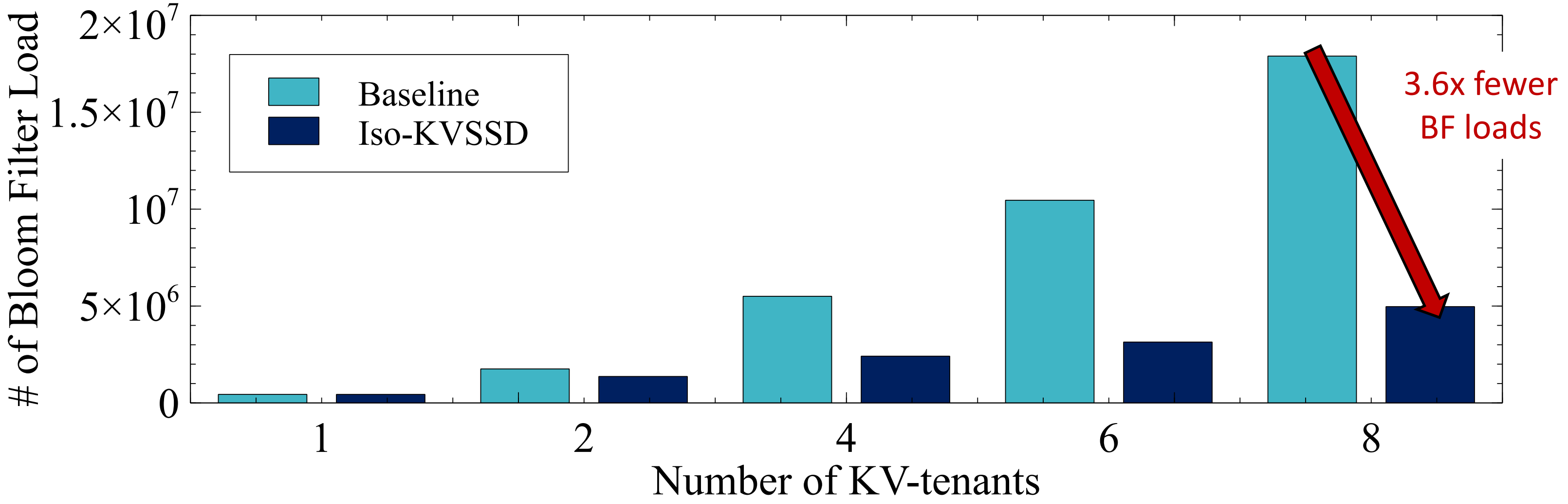


<Iso-KVSSD>

Per-namespace LSM-tree significantly reduces level depth at which KV data is indexed in the LSM-tree.

Impact of Per-namespace LSM-tree: # of Bloom Filter Loads

- The number of Bloom filter (BF) loads during LSM-tree search



Per-namespace LSM-tree significantly reduces the number of BF loads during KV data search process.

Sogang University

Conclusion

- Iso-KVSSD with per-namespace LSM-tree design
  - Identifies the user's namespace information for namespace isolation.
  - Manages the KV data using per-namespace LSM-tree design for performance isolation.

  - Provides strict view showing only the KV data corresponding to each user's namespace.
  - Offers 2.9x higher per-tenant read throughput and 2.8x lower per-tenant read response time than the baseline with a global-shared LSM-tree.

Isolating Namespace and Performance in Key-Value SSDs for Multi-tenant Environments

Donghyun Min

mdh38112@sogang.ac.kr

DISCOS LABORATORY

SOGANG UNIVERSITY