Harmonia

Enhancing Data Placement and Migration in Hybrid Storage Systems via Multi-Agent Reinforcement Learning

Rakesh Nadig

6 August 2025

FMS: the Future of Memory and Storage

Harmonia

Enhancing Data Placement and Migration in Hybrid Storage Systems via Multi-Agent Reinforcement Learning

Rakesh Nadig, Vamanan Arulchelvan, Rahul Bera, Taha Shahroodi, Gagandeep Singh, Andreas Kakolyris, Mohammad Sadrosadati, Jisung Park and Onur Mutlu

the Future of Memory and Storage

SAFARI Research Group
safari.ethz.ch

ETH zürich

Executive Summary

- **Background**:
  - A hybrid storage system (HSS) uses multiple different storage devices to provide scalable storage capacity at high performance.
  - The performance of an HSS highly depends on the effectiveness of its (1) data-placement, and (2) data-migration policies.
- **Problem**: Two key shortcomings of prior HSS data-management techniques:
  - Prior techniques focus on improving only data-placement or only data-migration policies, but do not optimize both.
  - Naively combining prior data-placement and data-migration techniques provide sub-optimal performance due to lack of coordination between the two policies.
- **Goal**: Design a holistic data management technique that (1) optimizes both data-placement and data-migration policies, and (2) achieves coordination between the two policies.
- **Contribution**: Harmonia, the first multi-agent online reinforcement learning-based HSS data-management technique that:
  - Performs **combined optimization** of both data placement and data migration policies
  - Achieves **coordination** between data placement and data migration to avoid conflicting decisions
  - Provides **adaptivity** to changing workload demands and underlying device characteristics
  - Can **easily extend** to any number of storage devices
- **Key Results**: Evaluate on **real systems** using seventeen data-intensive workloads
  - Harmonia **improves performance by 32%/33%** compared to the best previous data placement technique in performance-optimized/cost-optimized dual-HSS configuration.
  - In a tri-/quad-HSS configuration, Harmonia outperforms the state-of-the-art policy by **37%/42%.**
  - Harmonia's performance benefits come with low latency (**240ns** for inference) and storage overheads (**206 KiB** in DRAM for both RL agents together).

Talk Outline

Key Shortcomings of Prior Techniques

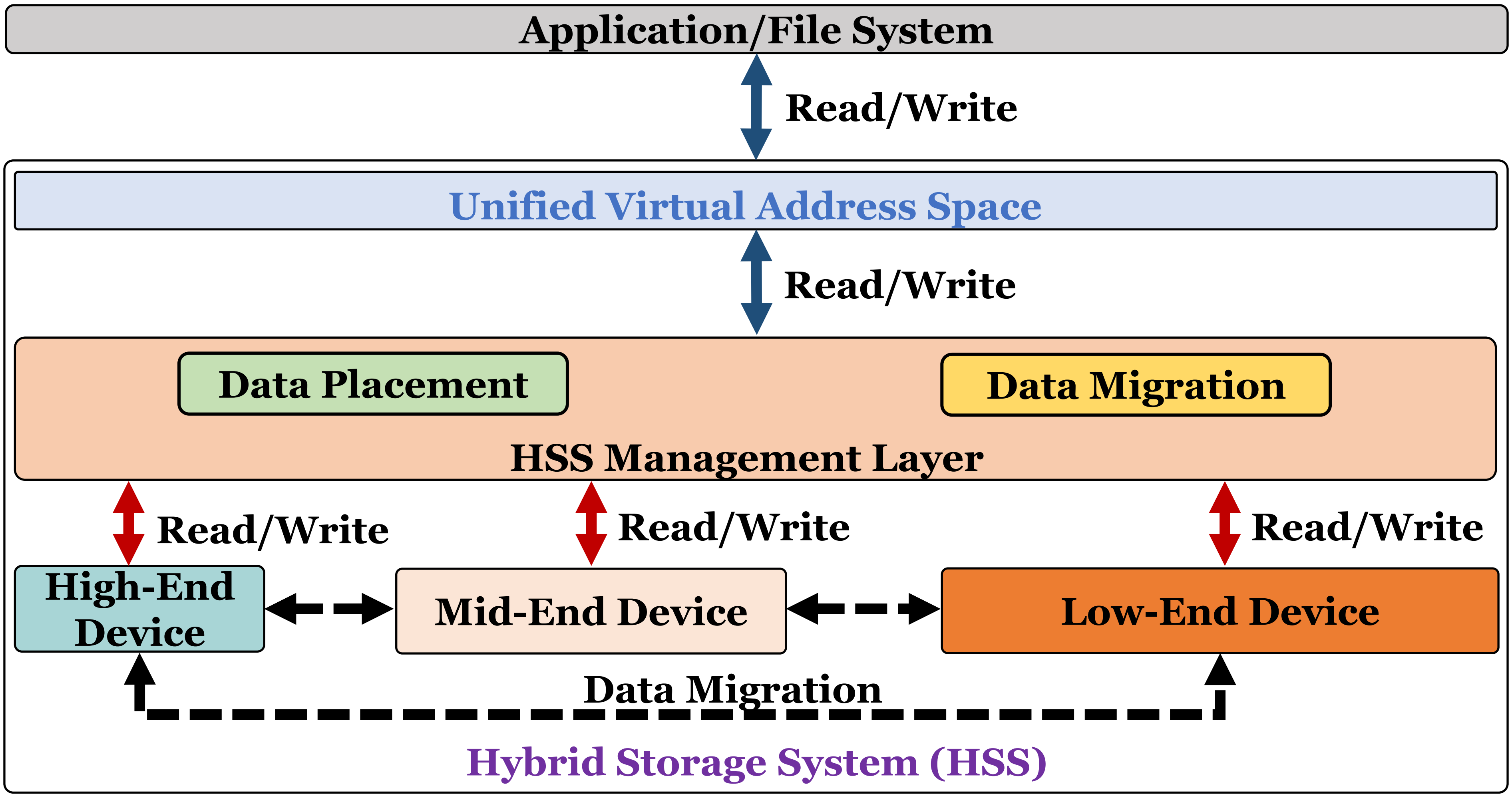
HSS Data Management using Reinforcement Learning

Harmonia: Overview

Evaluation and Key Results

Conclusion

Hybrid Storage System



Hybrid Storage System

Application/File System

Performance of a hybrid storage system **highly depends** on the ability of the **HSS management layer** to effectively perform **data placement** and **data migration**

Data Migration

Hybrid Storage System (HSS)

Key Shortcomings in Prior Techniques

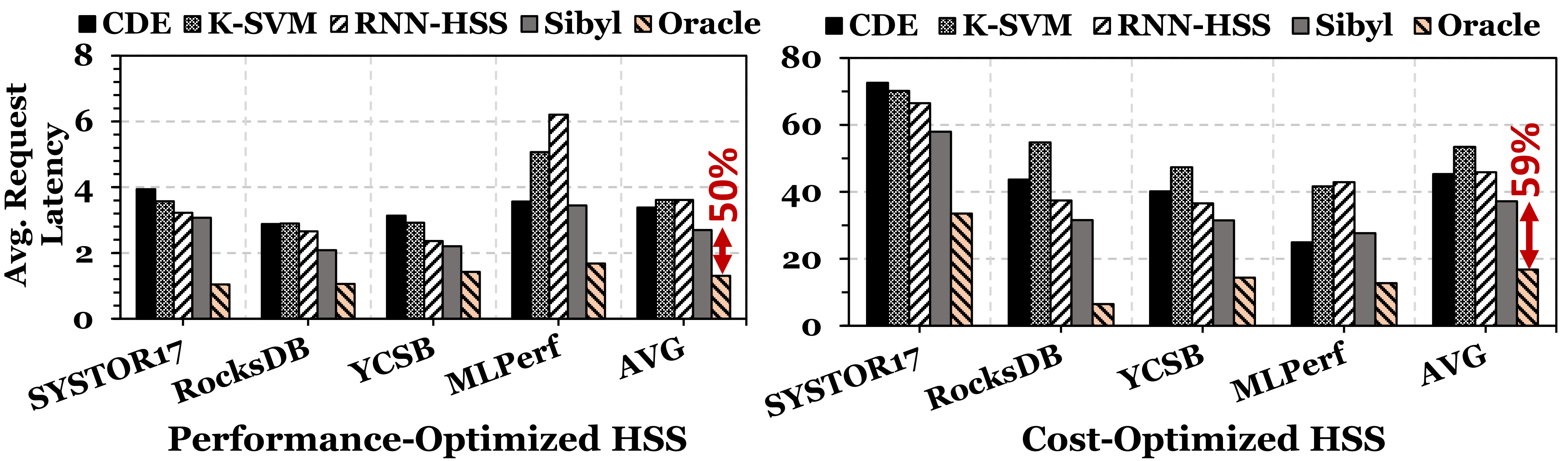
We observe two key shortcomings that significantly limit the performance benefits of prior data-management techniques:

Prior techniques do not optimize both data-placement and data-migration policies together

Naïve combination of prior data-placement and data-migration techniques achieves sub-optimal performance due to lack of coordination

Lack of a Holistic Data Management in HSS

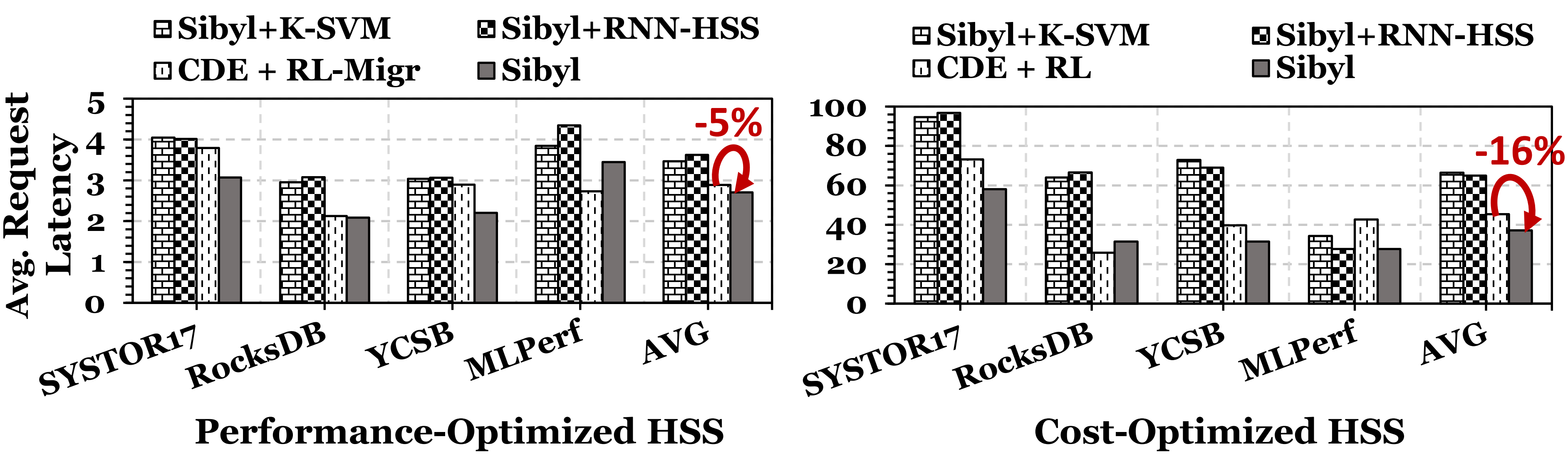
- Prior techniques
  - do not optimize both data placement and data migration policies together
  - use heuristic approaches, which do not adapt to changes in workload access patterns and HSS conditions



The large performance gap between the best-performing prior approach, Sibyl, and Oracle is due to its heuristic migration policy

Lack of Coordination in Extended Techniques

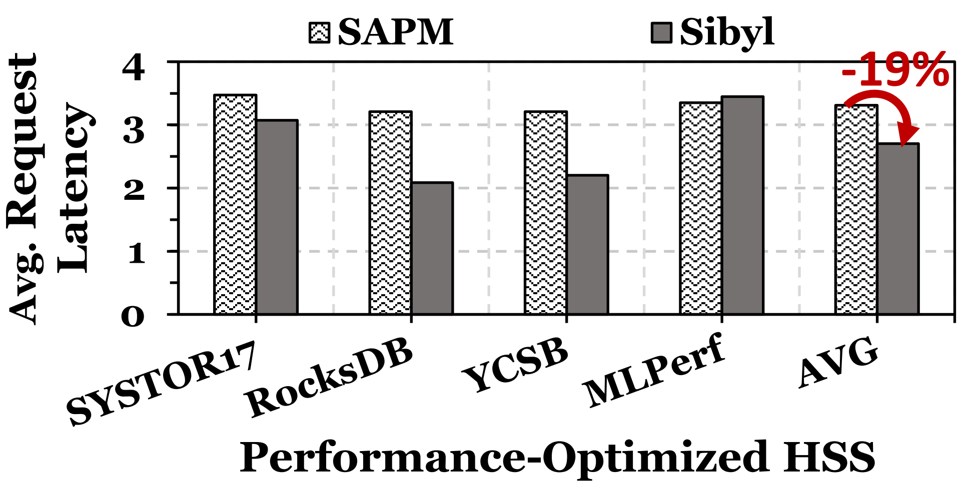
- Naïve combination of prior data-placement and data-migration techniques shows sub-optimal performance
- Lack of coordination between data-placement and data-migration techniques result in conflicting decisions

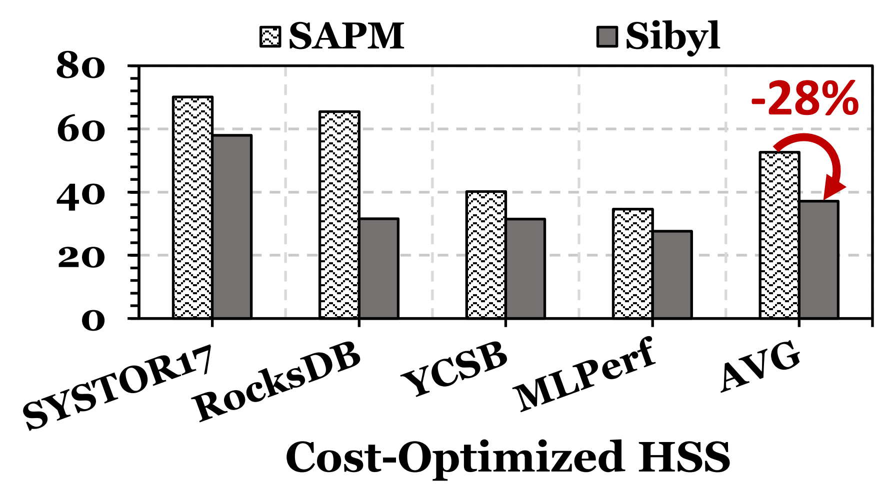


Naïve combinations of prior techniques underperform even compared to the best-performing standalone data-placement policy

Limitations of a Single-Agent RL Technique

- An RL-based technique can adapt to changes in workload and HSS configurations
  - Sibyl is the best-performing prior HSS data-management technique
- Data placement and data migration are two different tasks in HSS with different objectives
- A single RL agent based technique cannot optimize different tasks concurrently because it relies on task similarity to learn multiple tasks





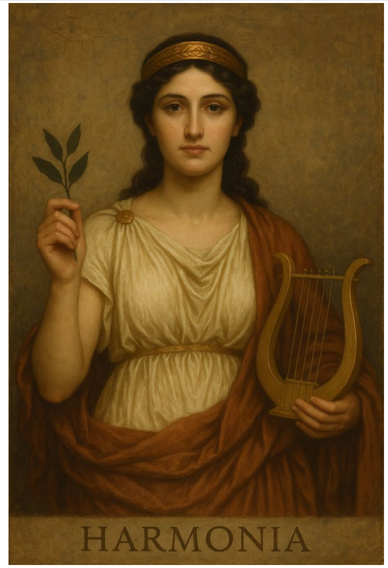
SAPM achieves lower performance than Sibyl because it cannot learn optimal policies for two different tasks concurrently

Our Goal

A **holistic data management mechanism** that :

1. **Performs combined optimization** of both data-placement and data-migration policies
2. **Achieves coordination** between data-placement and data-migration policies

Our Proposal



Harmonia

A holistic HSS data-management technique that uses *multi-agent online reinforcement learning* to achieve
(1) combined optimization of data-placement and data-migration policies
(2) coordination between the two policies

Harmonia is the Greek goddess of harmony and balance
https://en.wikipedia.org/wiki/Harmonia

Talk Outline

Key Shortcomings of Prior Techniques

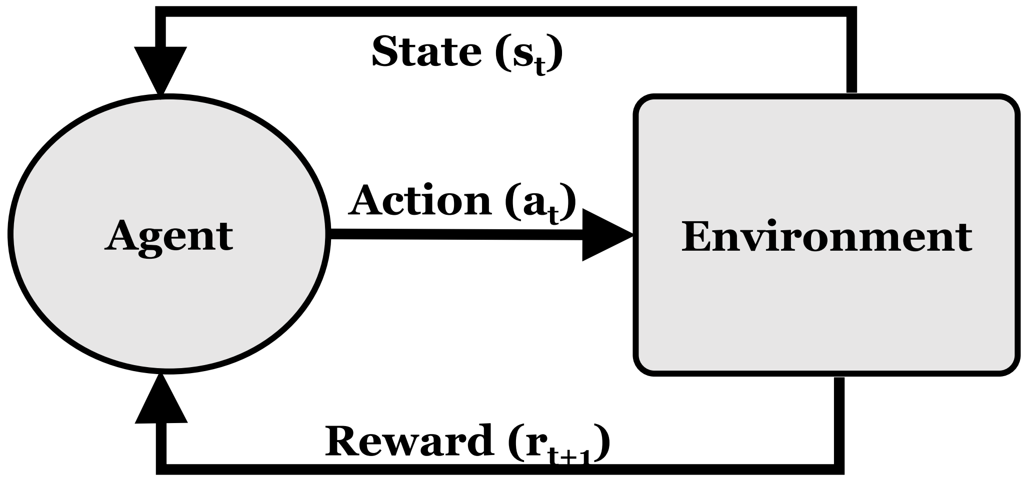
HSS Data Management using Reinforcement Learning

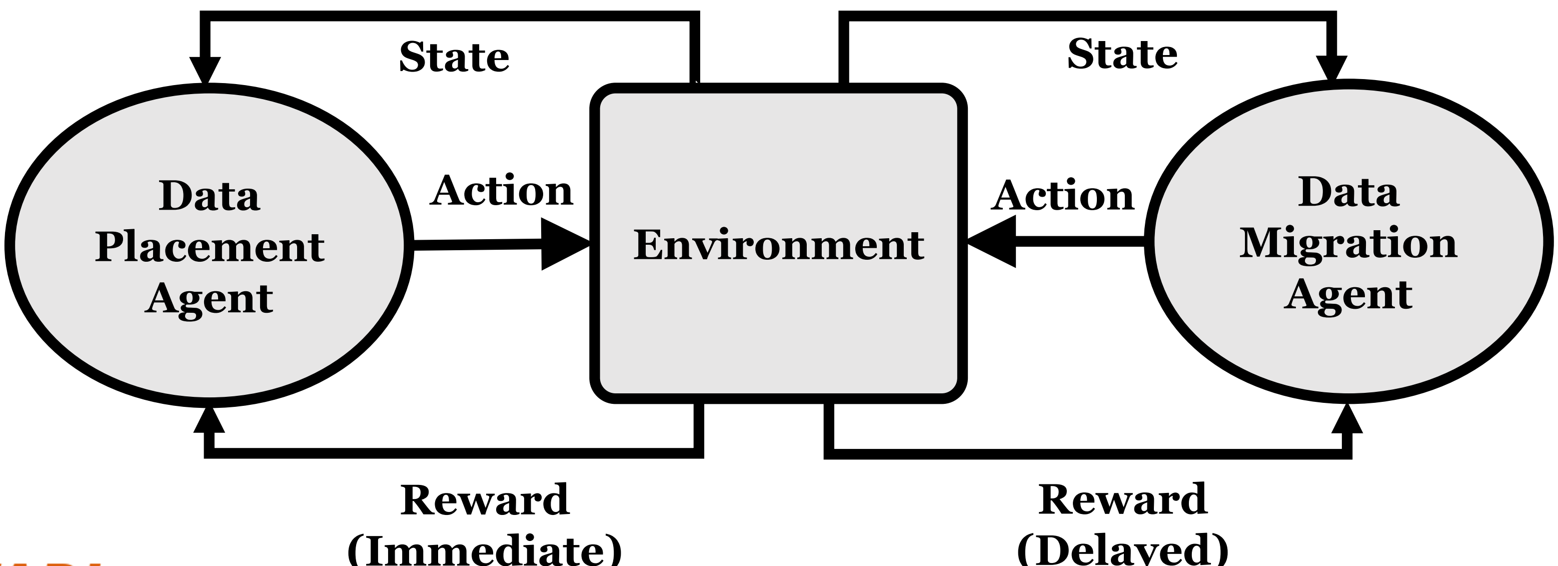
Harmonia: Overview

Evaluation and Key Results

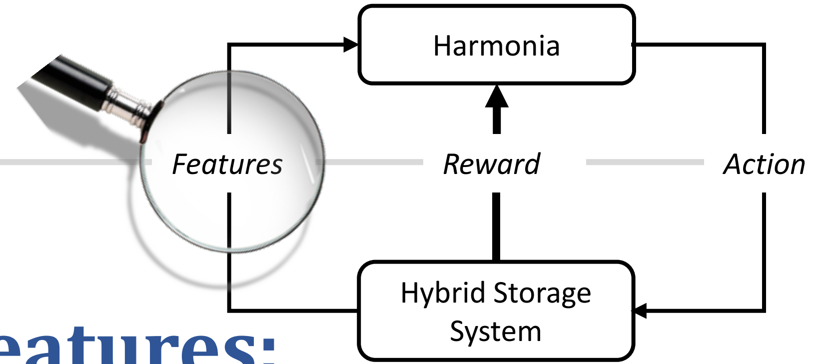
Conclusion

Formulating Data Management as RL





What is State?



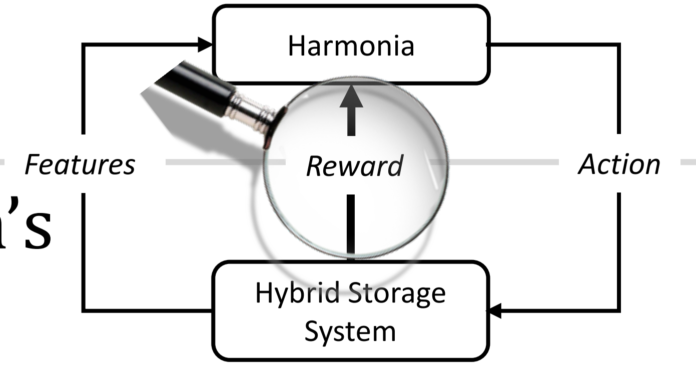
- **Limited number of state features:**
  - Reduce the implementation overhead
  - RL agent is more sensitive to reward

- ***7*-dimensional** vector of state features

  $$O_t = (size_t, type_t, intr_t, cnt_t, cap_t, curr_t, migr_t)$$

- We **quantize the state representation** into bins to reduce storage overhead

What is Reward?



- Defines the **objective** of Harmonia's agents
- We formulate the reward as

$$R_{placement} = (1/Latency_{current\_request})$$

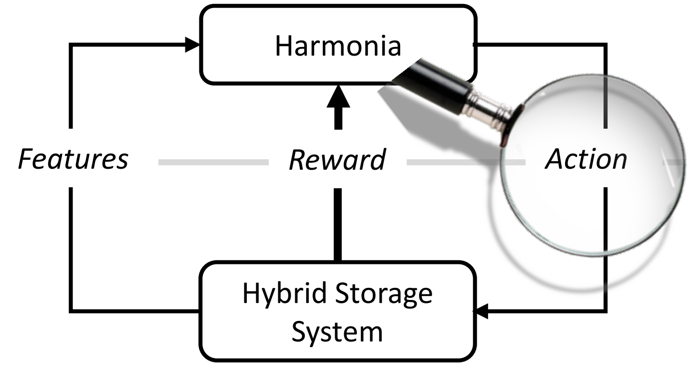
- Step reward for data placement

$$R_{migr\_delayed} = \begin{cases} \dfrac{n}{\sum_{i=t+1}^{t+n} L_i} - P_{migr_t} & \textit{after migrating x pages} \\ 0 & \textit{otherwise} \end{cases}$$

- Deferred reward for data migration
- Reward is issued based on the placement latencies of 'n' requests
- P_{migrt} : penalty based on migration interval to avoid ping-pong migration

- Latency encapsulates two key aspects:
  - **Internal state of the device** (e.g., read/write latencies, the latency of garbage collection, queuing delays, ...)
  - **Throughput**

What is Action?



- Data Placement
  - At every new page request, the action is to **select a storage device to place** the I/O request data

- Data Migration
  - For every page already placed in HSS, **select the storage device to migrate** the data

- Action can be **easily extended** to any number of storage devices

Talk Outline

Key Shortcomings of Prior Techniques

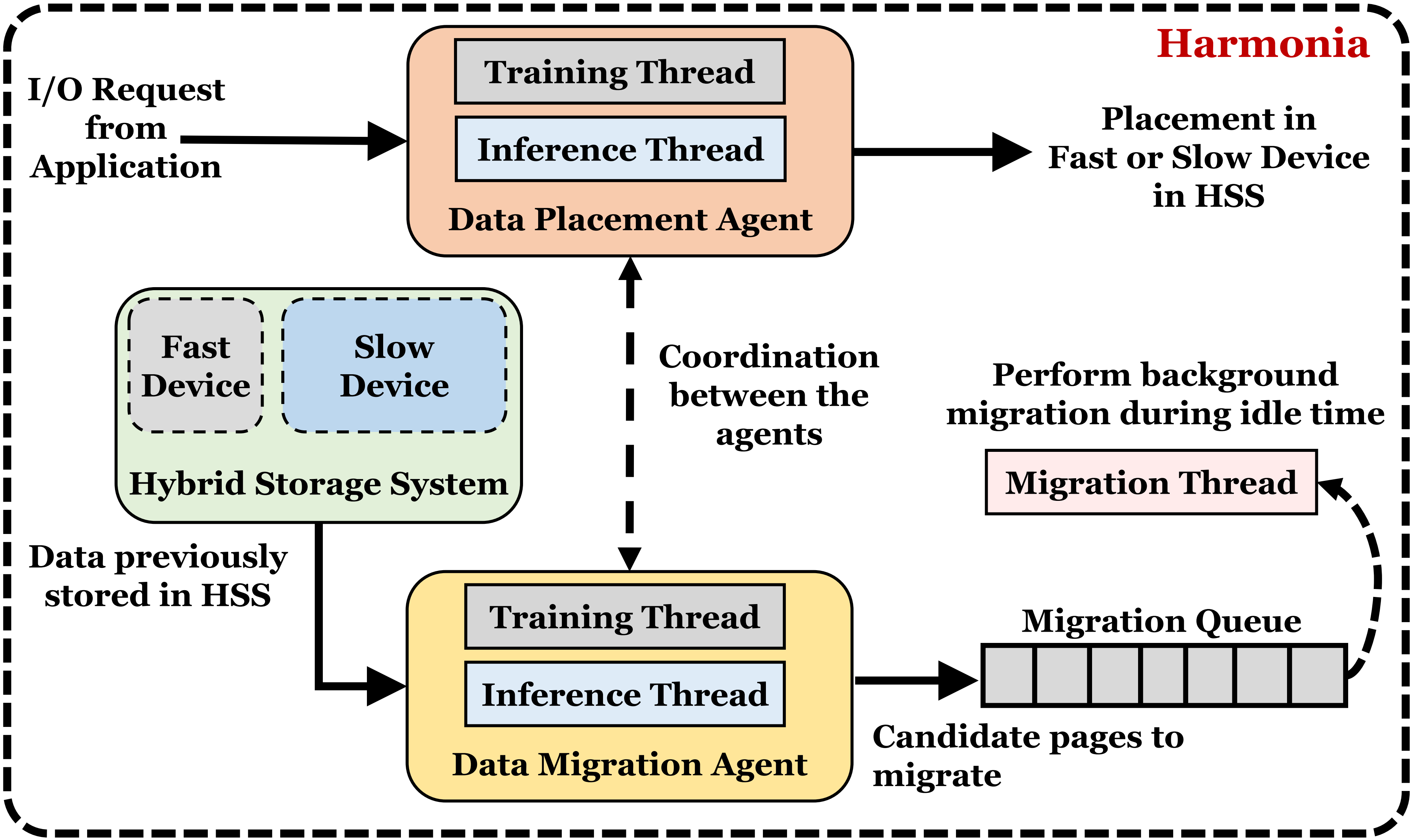
HSS Data Management using Reinforcement Learning

Harmonia: Overview

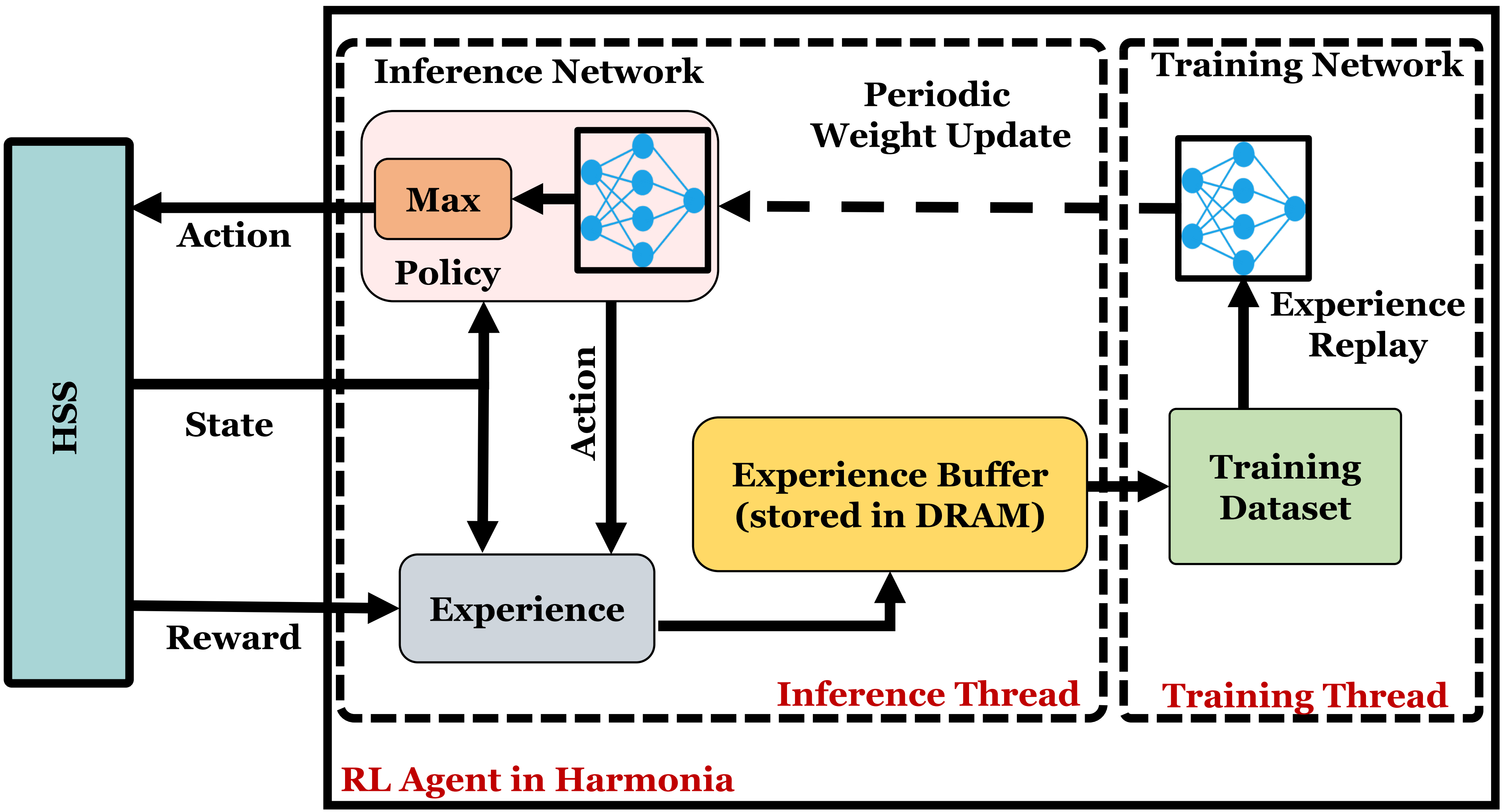
Evaluation and Key Results

Conclusion

Harmonia: Design



Harmonia: RL Agent Design



Talk Outline

Key Shortcomings of Prior Techniques

HSS Data Management using Reinforcement Learning

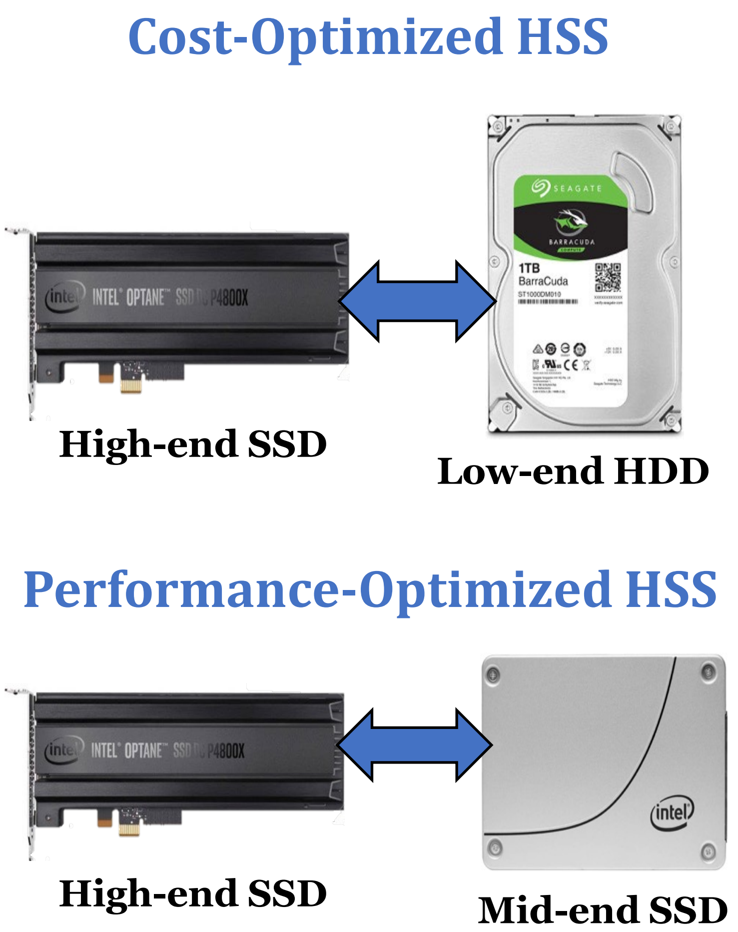
Harmonia: Overview

Evaluation and Key Results

Conclusion

Evaluation Methodology (1/2)

- **Real system** with various HSS configurations
  - Dual-hybrid, tri-hybrid and quad-hybrid storage systems



Evaluation Methodology (2/2)

- **Eight data-management** baselines:

Data Placement

- **CDE** [Matsui+, Proc. IEEE'17]
- **Sibyl** [Singh+, ISCA 2022]

Data Migration

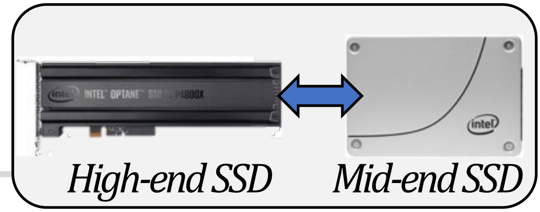
- **RNN-HSS** [Doudali+, HPDC'19]
- **K-SVM** [Shetti+, NAS 2022]

Extended Techniques

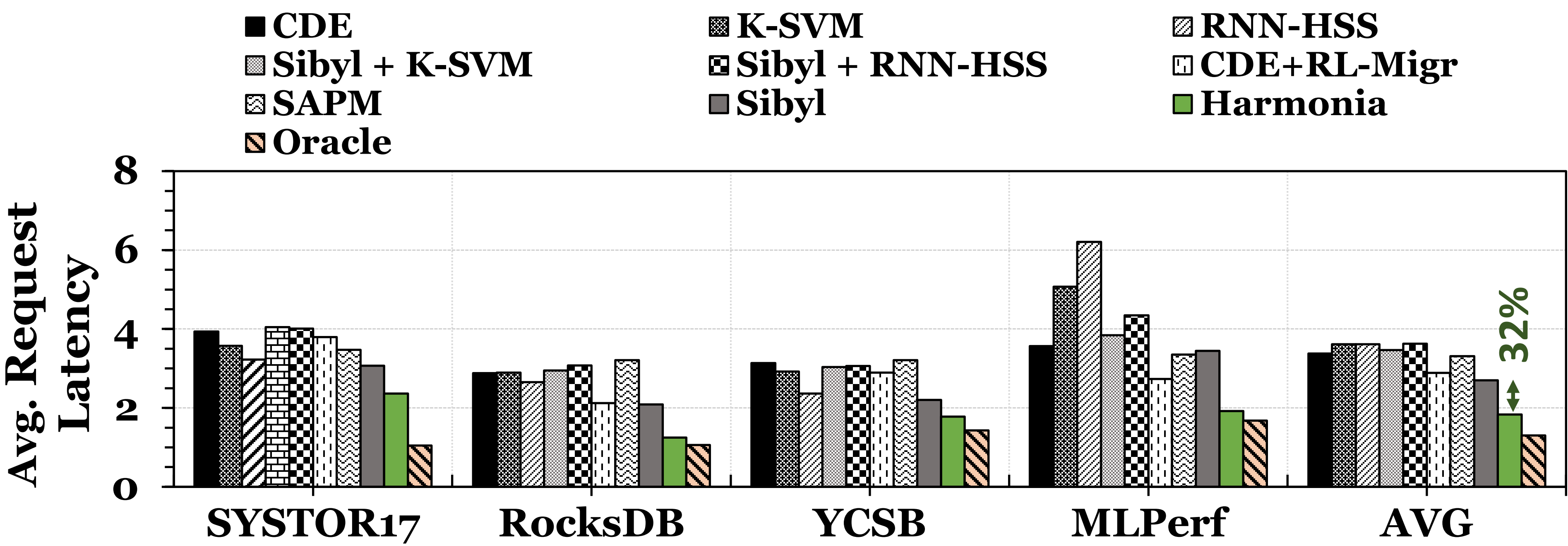
- **Sibyl + K-SVM**
- **Sibyl + RNN-HSS**
- **CDE + RL-Migr**
- **SAPM**

- **17 data-intensive workloads** from:
  - Yahoo! Cloud Serving Benchmark (YCSB): Cloud Database workloads (e.g., data collected from insert, update, read, scan)
  - SYSTOR '17: Traffic on enterprise virtual desktop infrastructure (VDI)
  - YCSB RocksDB: Key-value store
  - MLPerf Storage: Image Classification and Cosmology Parameter Prediction

Performance Analysis (1/2)



Performance-Optimized HSS

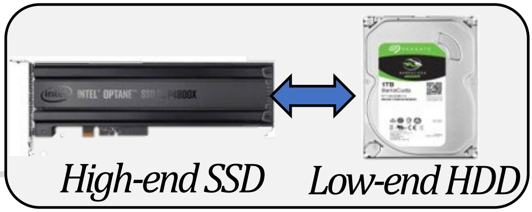


Harmonia

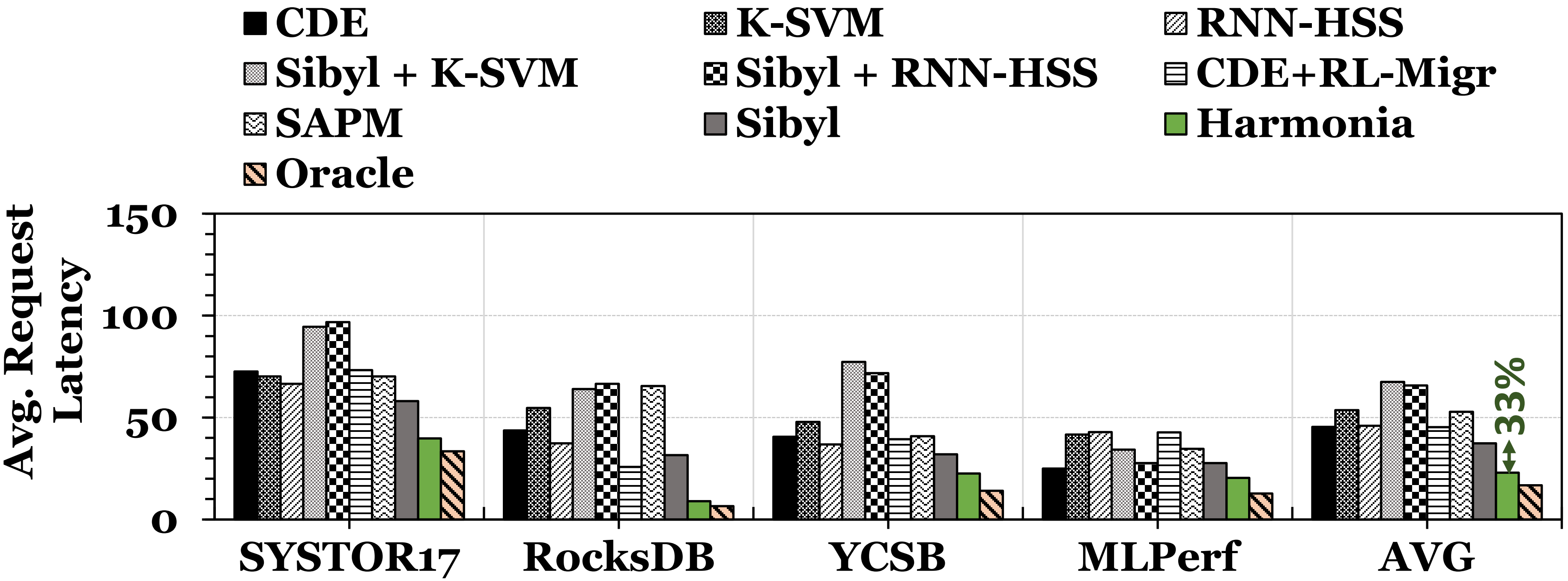
(1) improves average performance by 32% over Sibyl

(2) bridges the performance gap between Sibyl and Oracle by 64%

Performance Analysis (2/2)



Cost-Optimized HSS



Harmonia

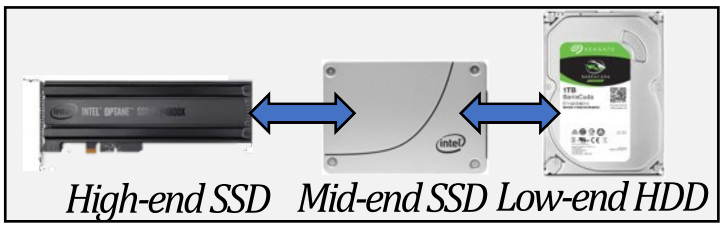
(1) improves average performance by 33% over Sibyl

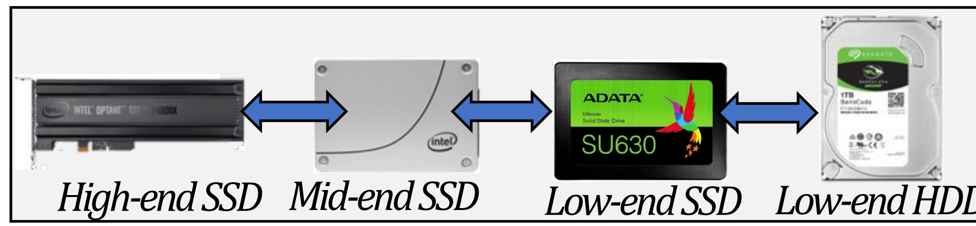
(2) bridges the performance gap between Sibyl and Oracle by 64%

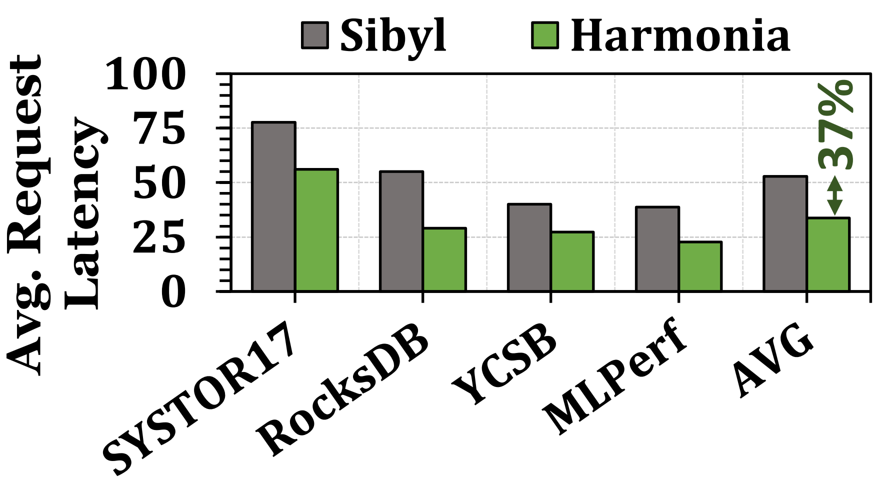
Extensibility Analysis

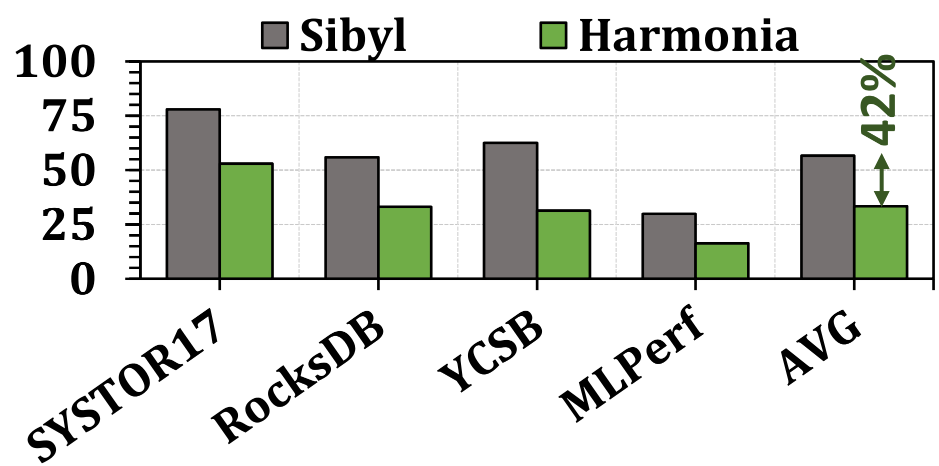
Extending Harmonia for **more devices**:

1. **Add a new action** in each agent
2. **Add the remaining capacity** of the new device as a state feature for each agent







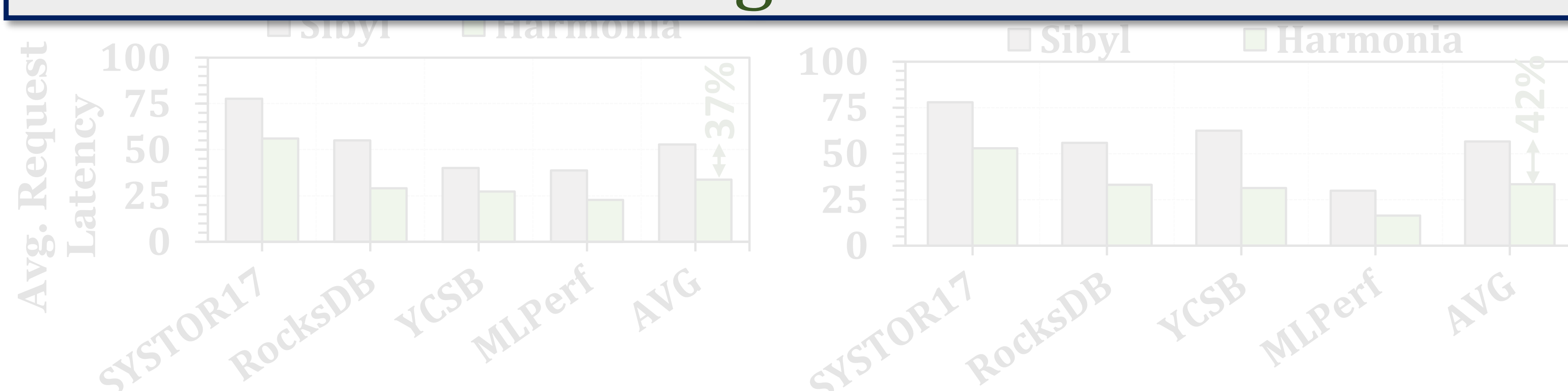


Extensibility Analysis

Extending Harmonia for **more devices:**

1. **Add a new action** in each agent
2. **Add the remaining capacity** of the new device as a state feature for each agent

Harmonia achieves higher performance improvement over prior approaches as the number of storage devices increases



Harmonia: Overhead Analysis

- **206 KiB** of total storage cost
  - Experience buffer, inference and training network
- **32-bit** metadata overhead per page for state features
- Inference Latency **- 240ns**
- Training Latency **- 53µs (background)**

Small storage overhead

Small inference overhead

More in the Paper (1/2)

- **Throughput (IOPS) evaluation**
  - Harmonia provides high end-to-end throughput compared to baseline policies because it indirectly captures throughput (size/latency)
- Evaluation on **mixed workloads**
  - Harmonia provides higher performance benefits when multiple workloads are executed concurrently
- **Tail latency** analysis
  - Harmonia significantly improves tail latency by performing data migration operations during system idle times
- **Performance sensitivity** to
  - Migration queue size
  - Number of incoming requests considered in migration agent's reward
- **Convergence** of Harmonia's policies
  - Both data-placement and data-migration policies converge in less than 8K I/O requests

More in the Paper (2/2)

Harmonia: A Multi-Agent Reinforcement Learning Approach to Data Placement and Migration in Hybrid Storage Systems

Rakesh Nadig[§] Vamanan Arulchelvan[§] Rahul Bera[§] Taha Shahroodi[§] Gagandeep Singh[†]
Andreas Kakolyris[§] Mohammad Sadrosadati[§] Jisung Park[∇] Onur Mutlu[§]

[§]*ETH Zürich* [†]*AMD Research* [∇]*POSTECH*

Abstract

Hybrid storage systems (HSS) combine multiple storage devices with diverse characteristics to achieve high performance and capacity at low cost. The performance of an HSS highly depends on the effectiveness of two key policies: (1) the data-placement policy, which determines the best-fit storage device for incoming data, and (2) the data-migration policy, which rearranges stored data (i.e., prefetches hot data and evicts cold data) across the devices to sustain high HSS performance. Prior works focus on improving only data placement or only data migration in HSS, which leads to relatively low HSS performance. Unfortunately, no prior work tries to optimize both policies together.

Our goal is to design a holistic data-management technique that optimizes both data-placement and data-migration policies to fully exploit the potential of an HSS, and thus significantly improve system performance. We demonstrate the need for multiple reinforcement learning (RL) agents to accomplish our goal. We propose **Harmonia**, a multi-agent reinforcement learning (RL)-based data-management technique that employs two lightweight autonomous RL agents, a data-placement agent and a data-migration agent, which adapt their policies for the current workload and HSS con-

policy determines the best-fit storage device in the HSS for incoming I/O requests. The data-migration policy rearranges data across the storage devices (i.e., prefetches frequently-accessed data to the fast device and evicts cold data to the slow device) to sustain high HSS performance, which can degrade over time due to (1) misplacement of data, and (2) changes in workload access patterns.

We identify four key challenges in designing efficient data-placement and data-migration policies. First, workload access patterns and HSS conditions (e.g., access latencies, device capacity utilization) can change frequently in data-intensive environments, which makes it hard to optimize the policies. Second, the data-placement policy should have a low performance overhead as it operates on the critical path of I/O request handling. Third, the data-migration policy needs to migrate data across storage devices in a timely manner without impacting the latency of incoming I/O requests. Fourth, the two policies should *not* make conflicting decisions, which can adversely impact HSS performance and device lifetimes.

Limitations of prior works. Prior works propose techniques for either data placement (e.g., [5–7, 9, 13–33, 52–68]) or data migration (e.g., [7, 11, 30, 33, 69–73]), but they provide relatively low performance when employed together or alone. Our motivational

https://arxiv.org/pdf/2503.20507.pdf

Talk Outline

Key Shortcomings of Prior Techniques

HSS Data Management using Reinforcement Learning

Harmonia: Overview

Evaluation and Key Results

Conclusion

Harmonia: Summary

First work to use multi-agent online reinforcement learning to optimize data management in HSS

Improves performance
by 32%/33% over the best-performing prior work
on performance-optimized/cost-optimized dual HSS

Achieves higher performance improvement over prior approaches when more storage devices are added

Low inference latency and storage overheads the Future of Memory and Storage

Harmonia

Enhancing Data Placement and Migration in Hybrid Storage Systems via Multi-Agent Reinforcement Learning

Rakesh Nadig, Vamanan Arulchelvan, Rahul Bera, Taha Shahroodi, Gagandeep Singh, Andreas Kakolyris, Mohammad Sadrosadati, Jisung Park and Onur Mutlu

SAFARI
SAFARI Research Group
safari.ethz.ch

ETH zürich

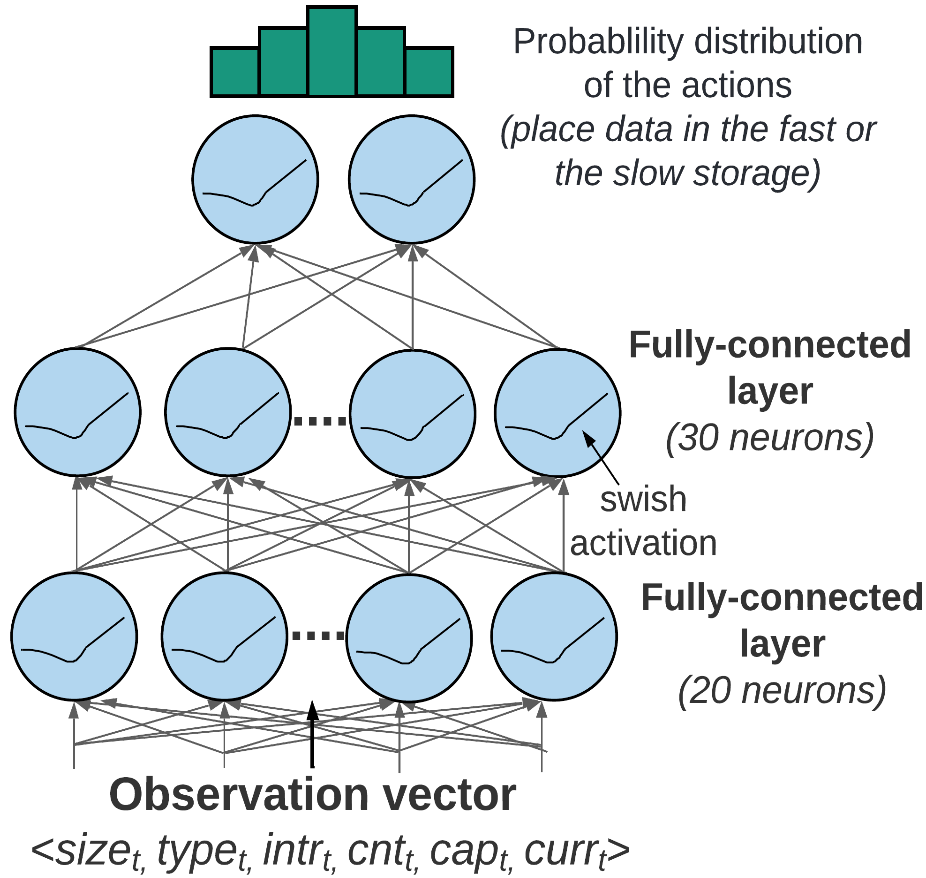
Backup Slides

State Features in Harmonia

Feature	Description	# of bins	Encoding (bits)
req_type	Request type (read/write)	2	1
req_size	Request size (in pages)	8	3
acc_intr	Access interval of the requested page	64	8
acc_freq	Access frequency of the requested page	64	8
fast_cap	Free space in the fast storage device	8	3
curr_dev	Storage device where the requested page currently resides	2	1
migr_intr	Migration interval of a page	64	8

Training and Inference Network

- Training and inference network **allow parallel execution**
- **Observation vector as the input**
- **Produces probability distribution of Q-values**



Hyper-parameter Tuning

- Different hyper-parameter configurations were chosen using the design of experiments (DoE) technique

Hyper Parameter	Design Space	Placement Agent	Migration Agent
Discount Factor (γ)	0-1	0.9	0.1
Learning Rate (α)	$1e^{-5} - 1e^{0}$	$1e^{-3}$	$1e^{-2}$
Exploration Rate (ϵ)	0-1	0.001	0.001
Batch Size	64-256	128	256
Experience Buffer Size	10-10000	1000	1000

Evaluation Methodology

Host System	AMD Ryzen 7 2700G[110], 8-cores@3.5 GHz, 8×64/32 KiB L1-I/D, 4 MiB L2, 8 MiB L3, 16 GiB RDIMM DDR4 2666 MHz
Storage Devices	**Characteristics**
H: Intel Optane SSD P4800X [47]	375 GB, PCIe 3.0 NVMe, SLC, R/W: 2.4/2 GB/s, random R/W: 550000/500000 IOPS
M: Intel SSD D3-S4510 [48]	1.92 TB, SATA TLC (3D), R/W: 560/510 MB/s, random R/W: 895000/21000 IOPS
L: Seagate HDD ST1000DM010 [108]	1 TB, SATA 6Gb/s 7200 RPM Max. Sustained Transfer Rate: 210 MB/s
$\mathbf{L}_{\mathbf{SSD}}$: ADATA SU630 SSD [111]	960 GB, SATA, TLC, R/W: 520/450 MB/s
PMEM (Emulated): Intel Optane Persistent Memory 200 Series [112]	128 GB, Memory Mode R/W: 7.45/2.25 GB/s (256B)
HSS Configurations	**Devices**
Performance-Optimized	high-end (H) & middle-end (M)
Cost-Optimized	high-end (H) & low-end (L)
HSS with PMEM	Emulated PMEM (PMEM) & high-end SSD (H)
Tri-HSS	high-end (H) & middle-end (M) & low-end (L)
Quad-HSS	high-end (H) & middle-end (M) & low-end SSD (L_{SSD}) & low-end HDD (L)

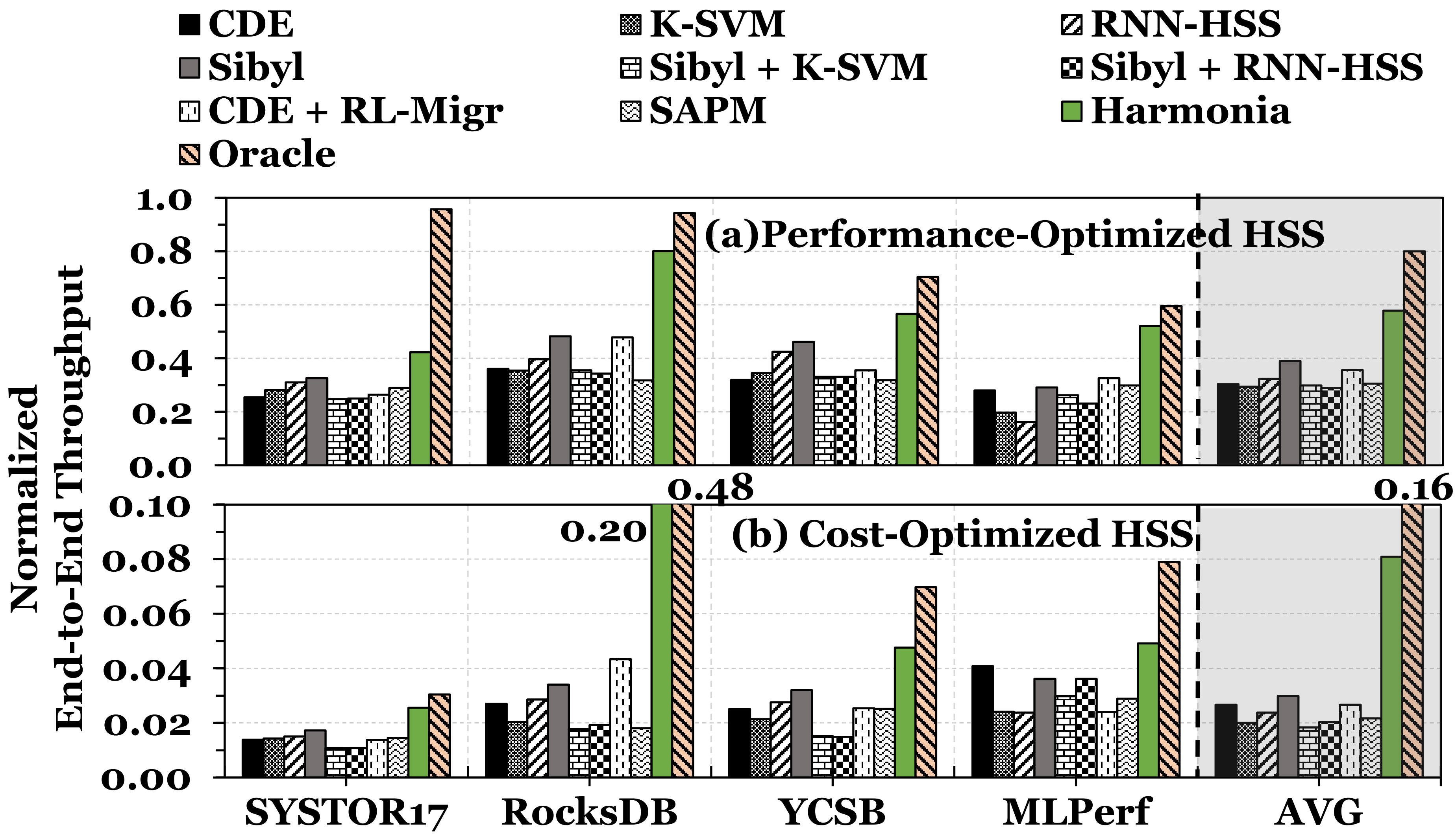
Workload Characteristics

Benchmark Suite	Traces	Read %	Avg. Request Size (KB)	Avg. Inter-Request Time (μs)
SYSTOR17 [113]	LUN0	0.2	31.7	1163.9
	LUN1	0.3	34.2	1864.1
	LUN2	7.6	31.1	1418.9
	LUN3	3.5	42.7	734.5
	LUN4	0.5	26.3	823.1
RocksDB [114]	ssd-00	79.9	108.9	66.4
	ssd-01	73.5	75.1	40.7
	ssd-02	79.9	7.5	3.3
	ssd-03	79.9	9.5	3.5
	ssd-04	79.9	7.8	3.6
YCSB [115]	YCSB-B	51.3	45.9	9.3
	YCSB-C	47.6	54.6	6.5
	YCSB-D	55.9	36.1	8.5
	YCSB-E	52.1	46.6	9.6
	YCSB-F	49.5	53.1	6.6
MLPerf Storage [116]	ResNet50	80.0	172.6	500.1
	CosmoFlow	83.4	180.1	1023.8

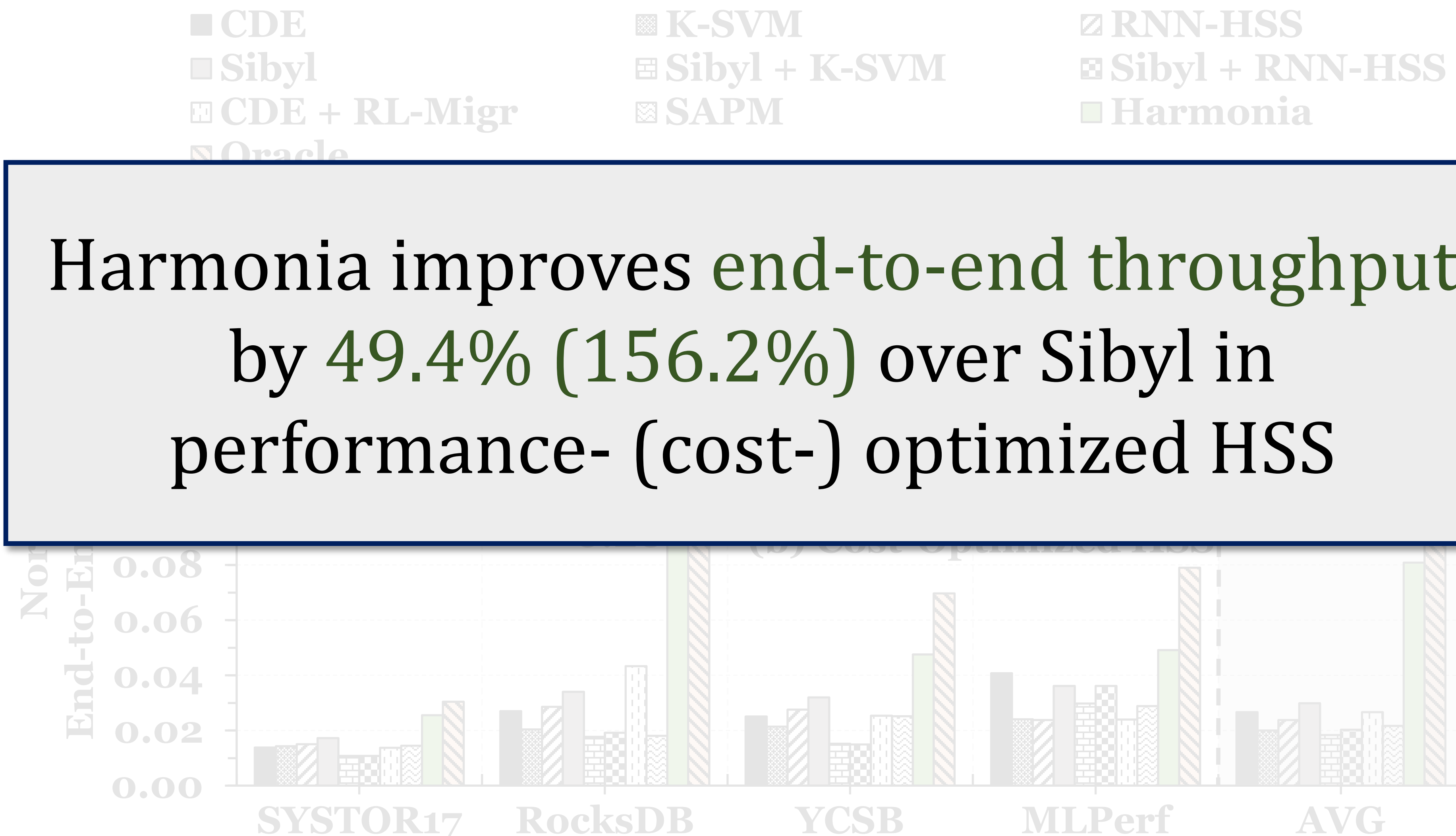
Multi -Programmed Workloads

Mix	Constituent Workloads [113–115]	Description
mix1	ssd-02 and LUN4	ssd-02 is read-intensive and LUN4 is write-intensive
mix2	LUN1 and ssd-04	LUN0 is write-intensive and ssd-04 is read-intensive
mix3	YCSB-C and YCSB-F	Both have near-equal read-write ratio
mix4	ssd-00, ssd-04, YCSB-A and LUN0	Two read-intensive and two write-intensive workloads
mix5	ssd-00, LUN0, YCSB-C and YCSB-F	Read-intensive, write-intensive and two workloads with a near-equal read-write ratio
mix6	YCSB-B, YCSB-D, LUN0, LUN1, LUN4, ssd-00, ssd-02, ssd03	Two with near-equal read-write ratio, three write-intensive and three read-intensive

Throughput Analysis



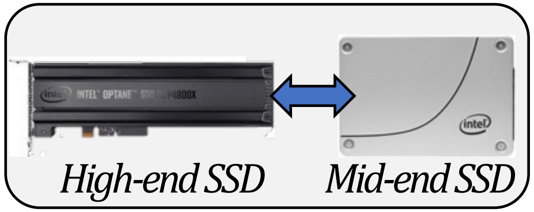
Throughput Analysis

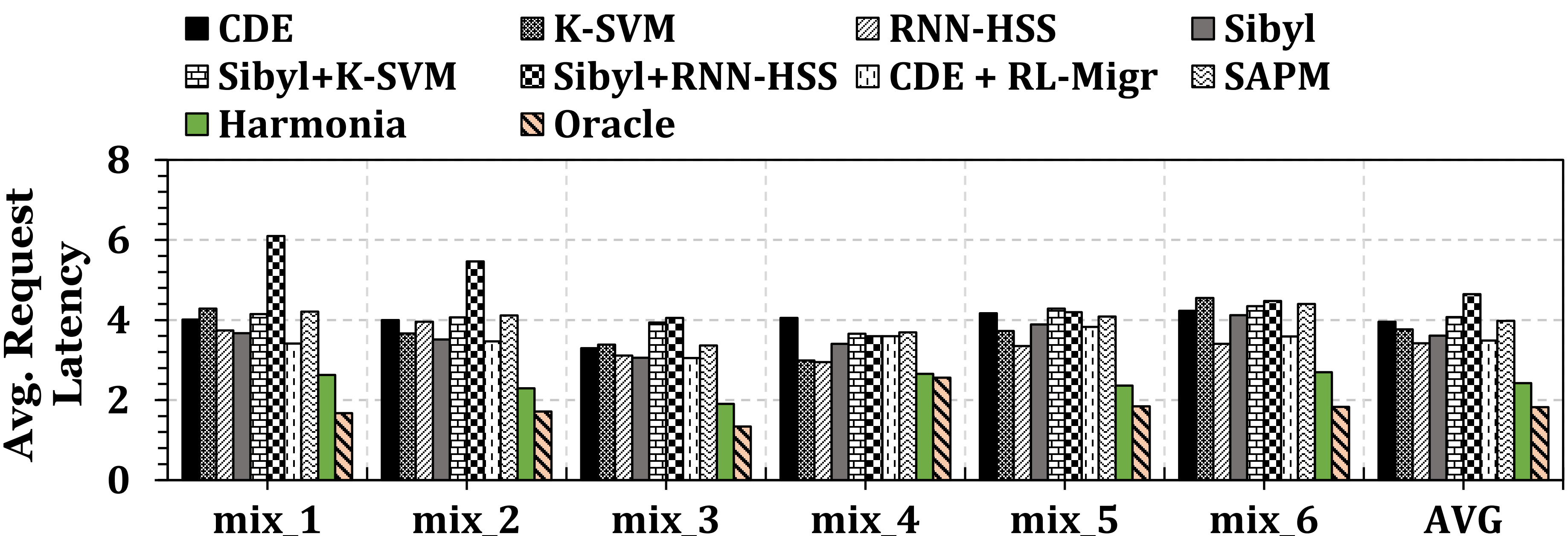


Harmonia improves end-to-end throughput by 49.4% (156.2%) over Sibyl in performance- (cost-) optimized HSS

Multi-Programmed Workloads (1/2)

Performance-Optimized HSS

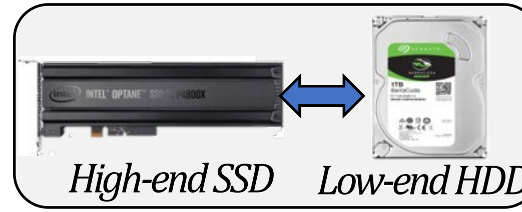


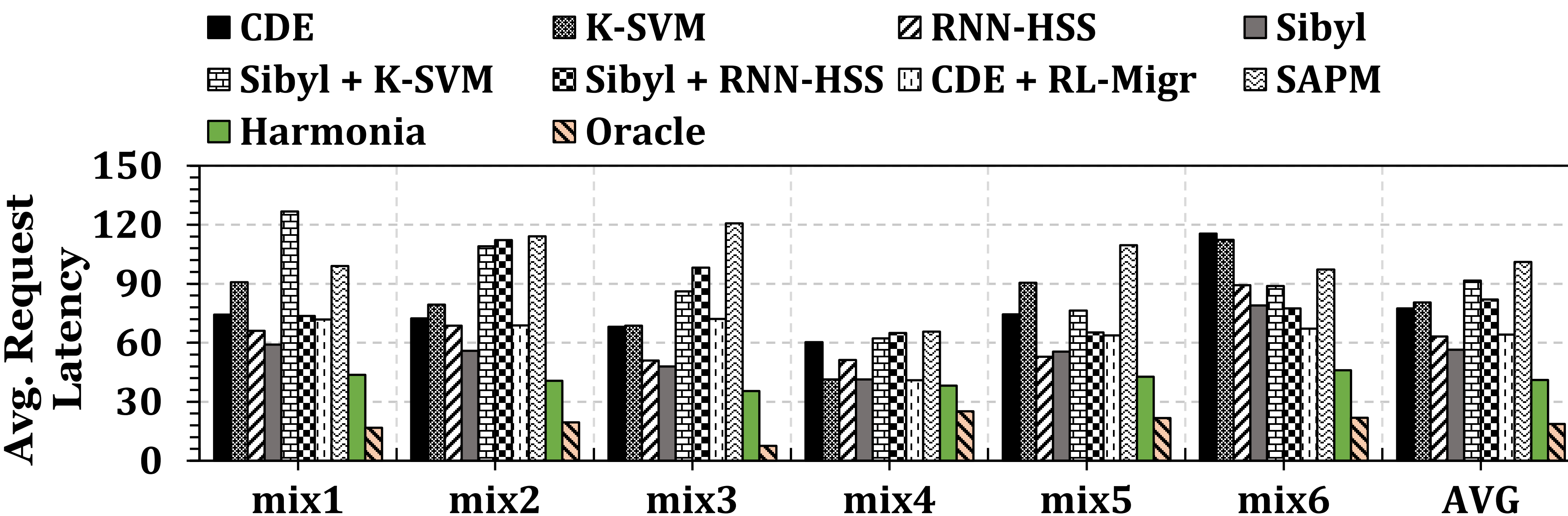


Harmonia improves performance by 33% on average over best-performing prior work, Sibyl

Multi-Programmed Workloads (2/2)

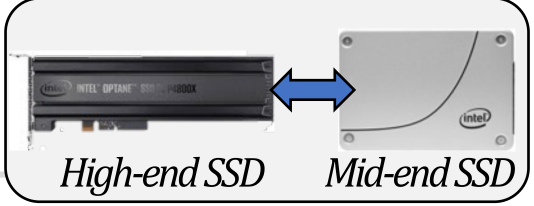
Cost-Optimized HSS

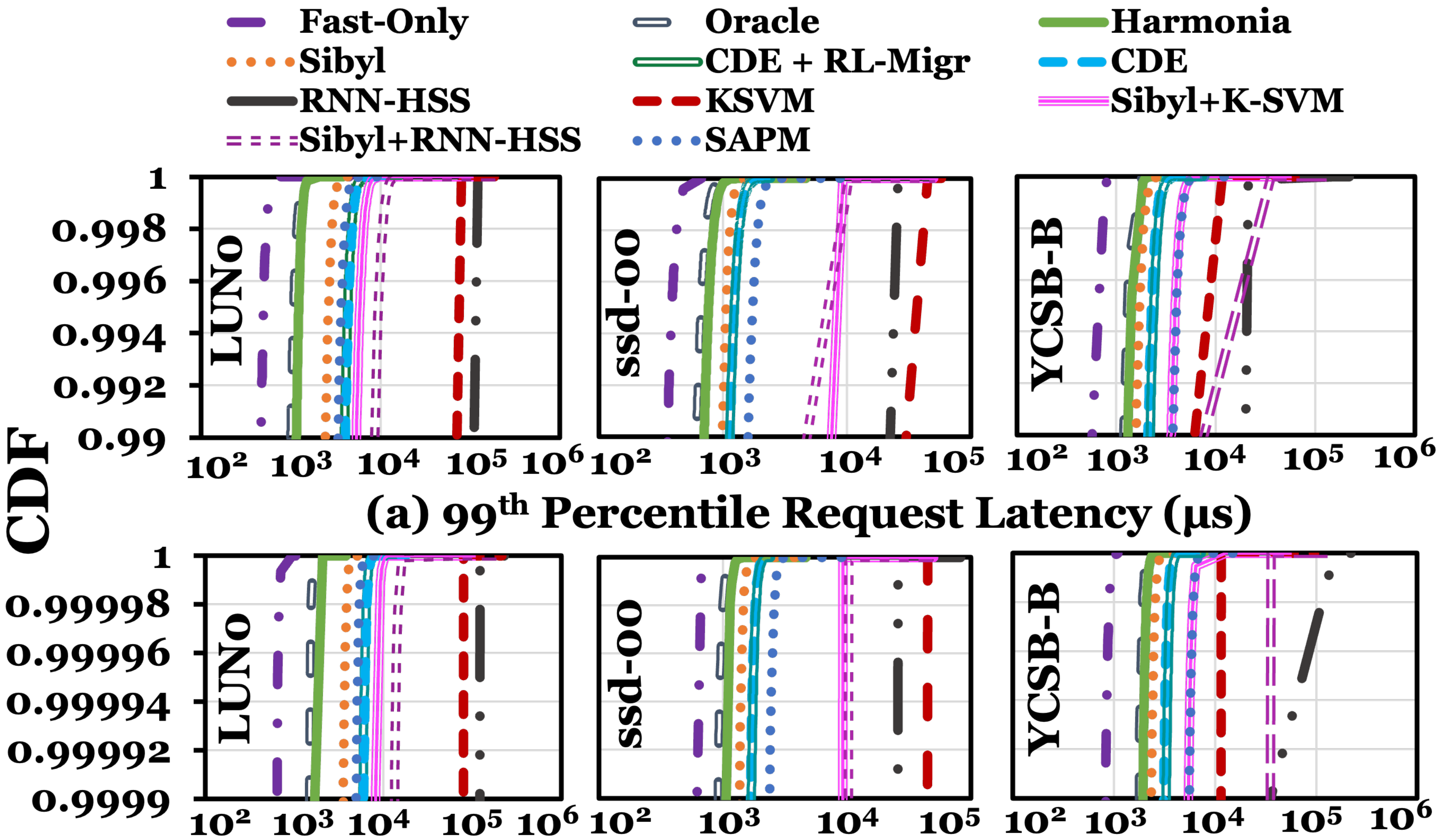




Harmonia improves performance by 25% over Sibyl due to faster learning of access pattern variations compared to prior approaches

Tail Latency

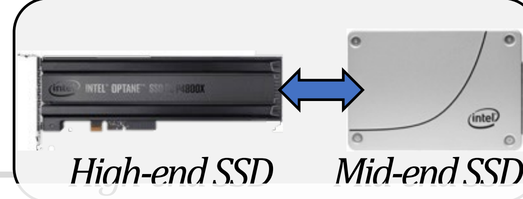


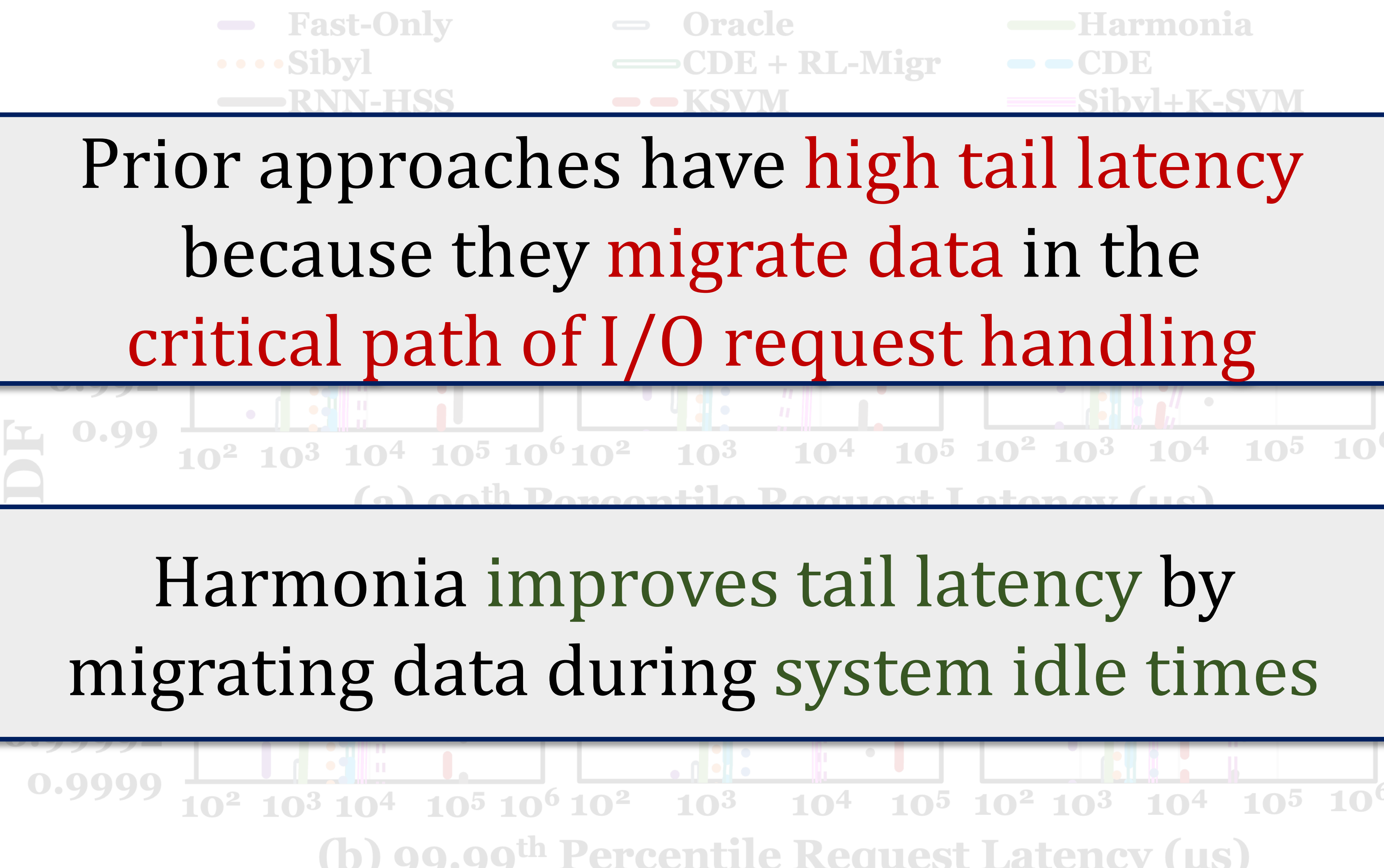


(a) 99th Percentile Request Latency (µs)

(b) 99.99th Percentile Request Latency (µs)

Tail Latency

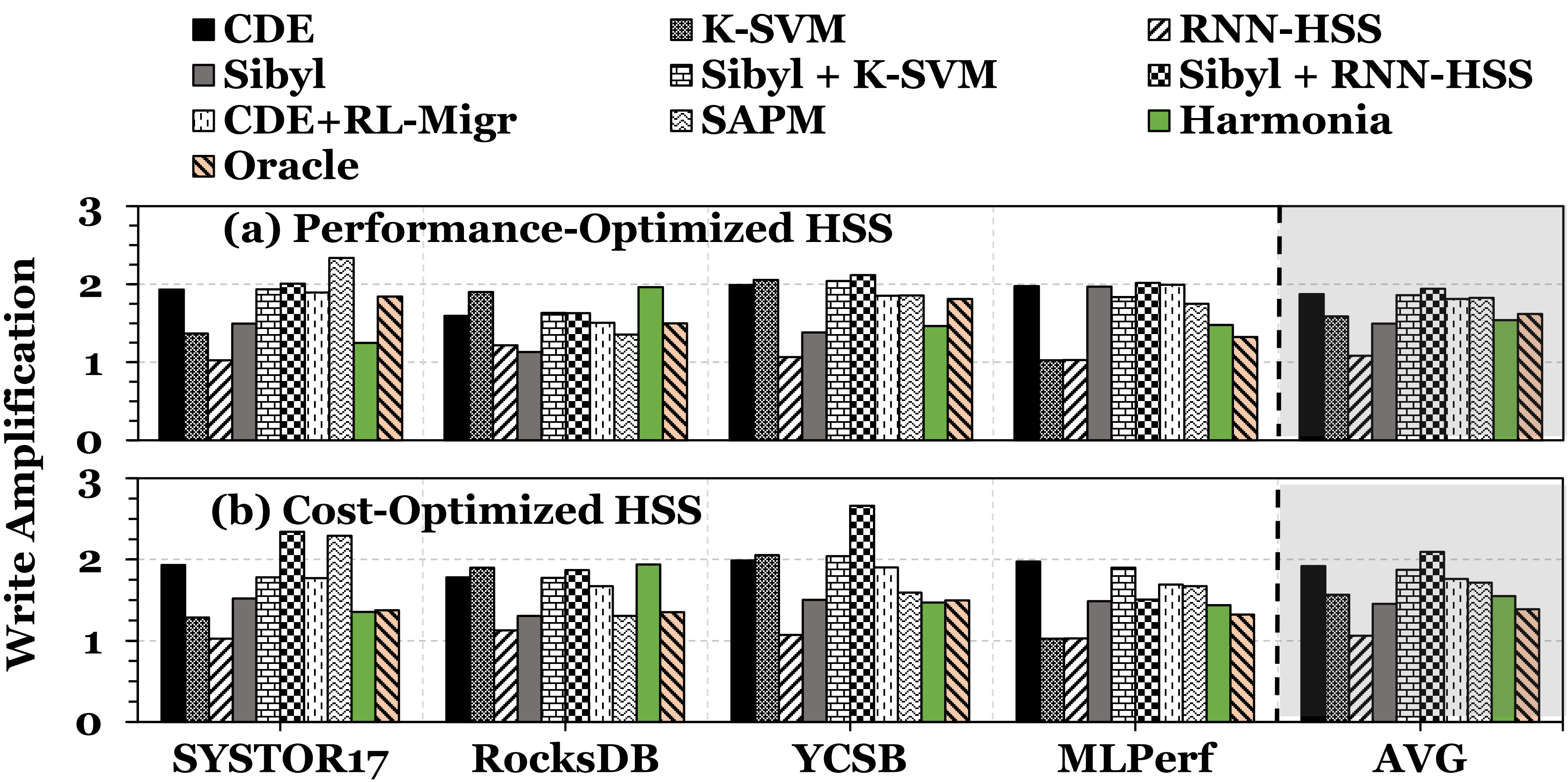




Prior approaches have high tail latency because they migrate data in the critical path of I/O request handling

Harmonia improves tail latency by migrating data during system idle times

Impact on Device Lifetimes

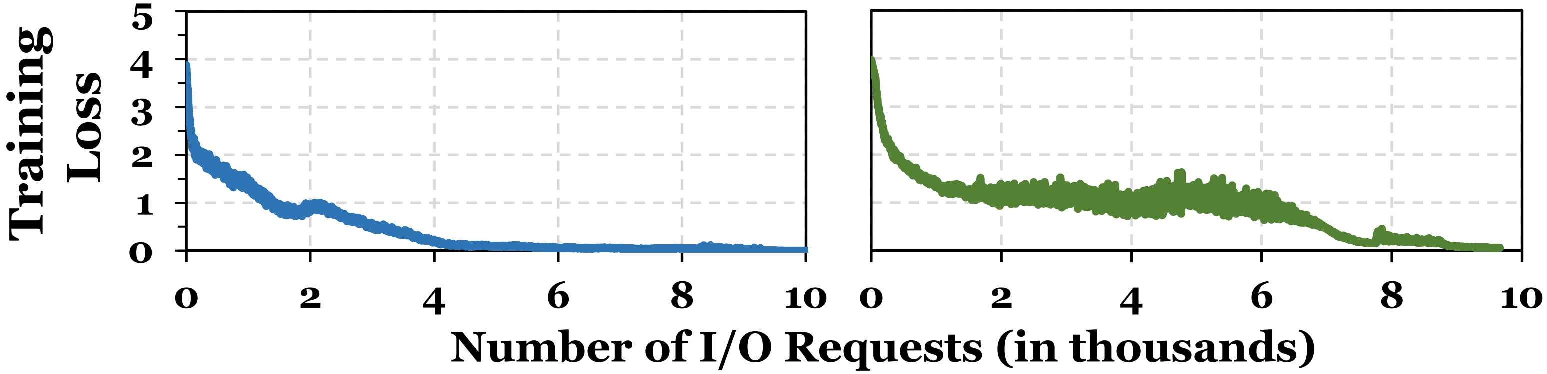


Impact on Device Lifetimes

Harmonia learns to migrate more data in read-intensive workloads, since there are infrequent updates from the application

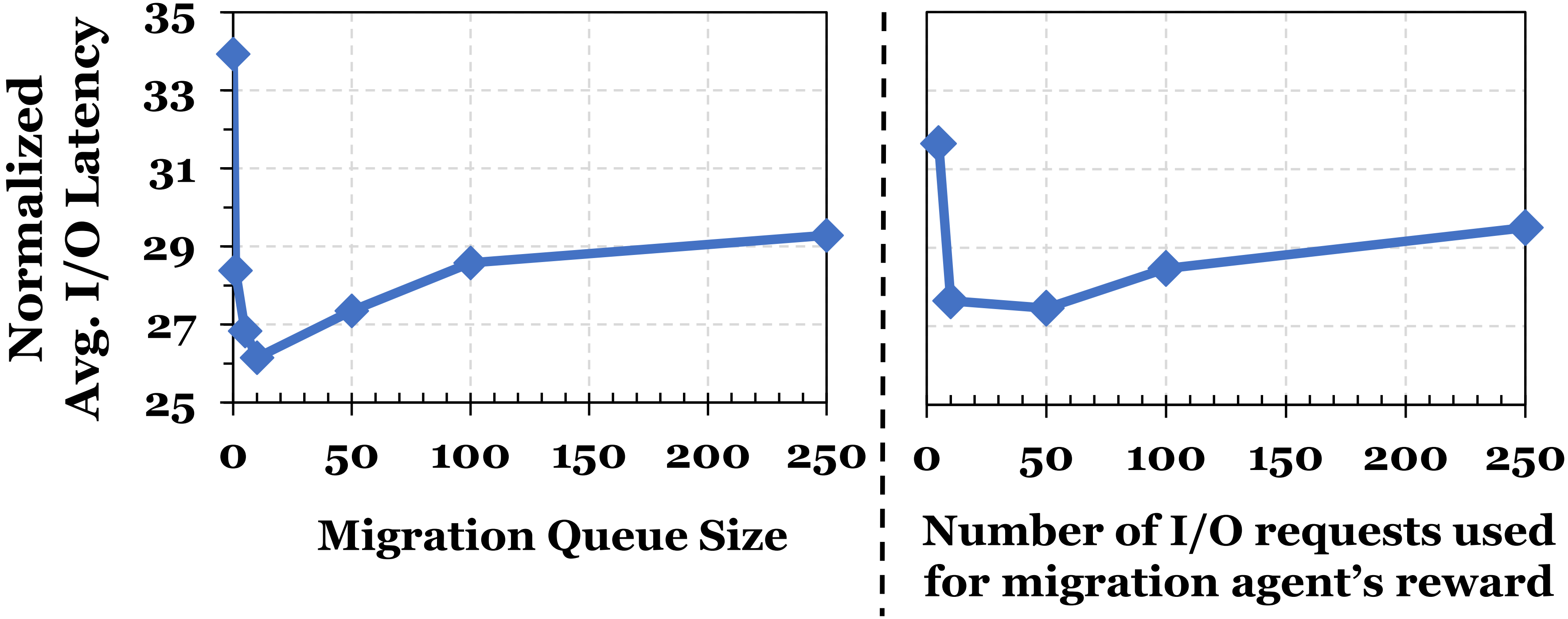
Harmonia migrates less data in write-intensive workloads as data is moved during updates from application , which causes lower write amplification

Convergence of Harmonia's Policies

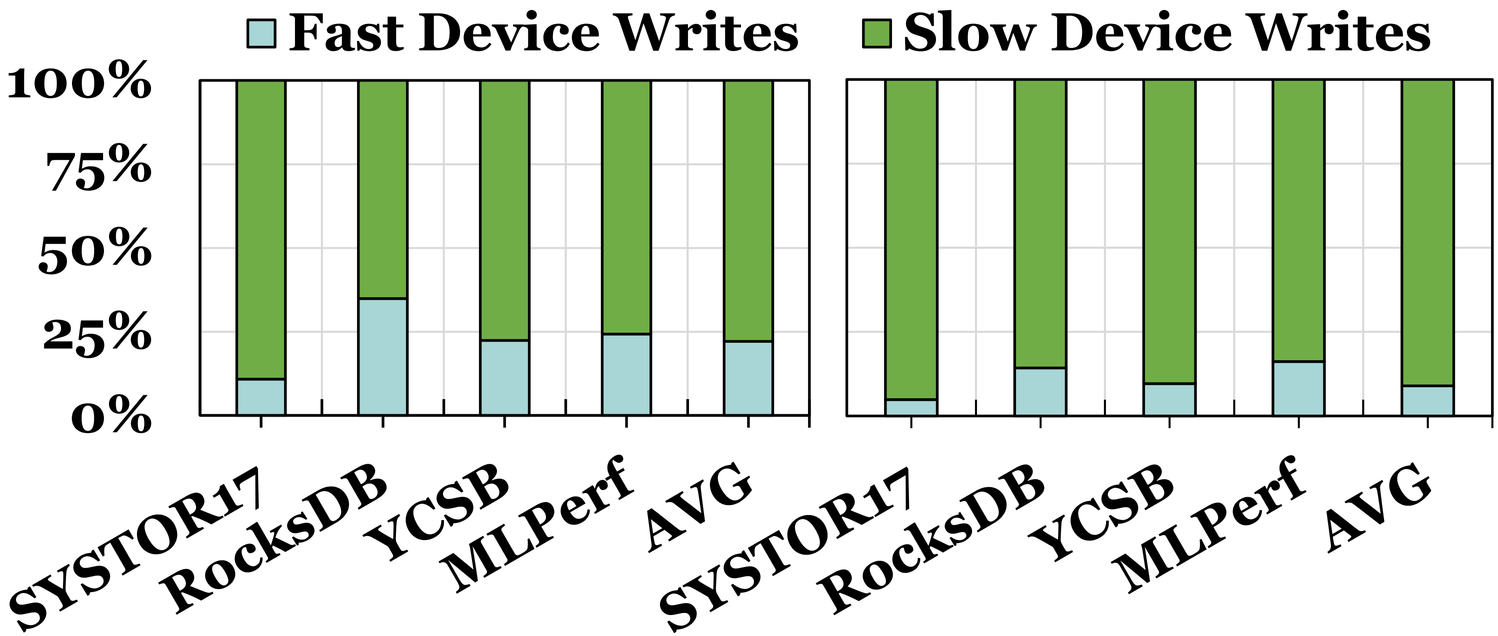


(a) Data-Placement Agent (b) Data-Migration Agent

Sensitivity Analysis



Write Traffic Distribution



(a) Performance-Optimized HSS

(b) Cost-Optimized HSS