ETH zürich | Carnegie Mellon University | Systems@ETH Zürich | SAFARI

MQSim:
A Framework for Enabling Realistic Studies of Modern Multi-Queue SSD Devices

Arash Tavakkol, Juan Gómez-Luna,
Mohammad Sadrosadati, **Saugata Ghose**, Onur Mutlu

February 13, 2018

Executive Summary

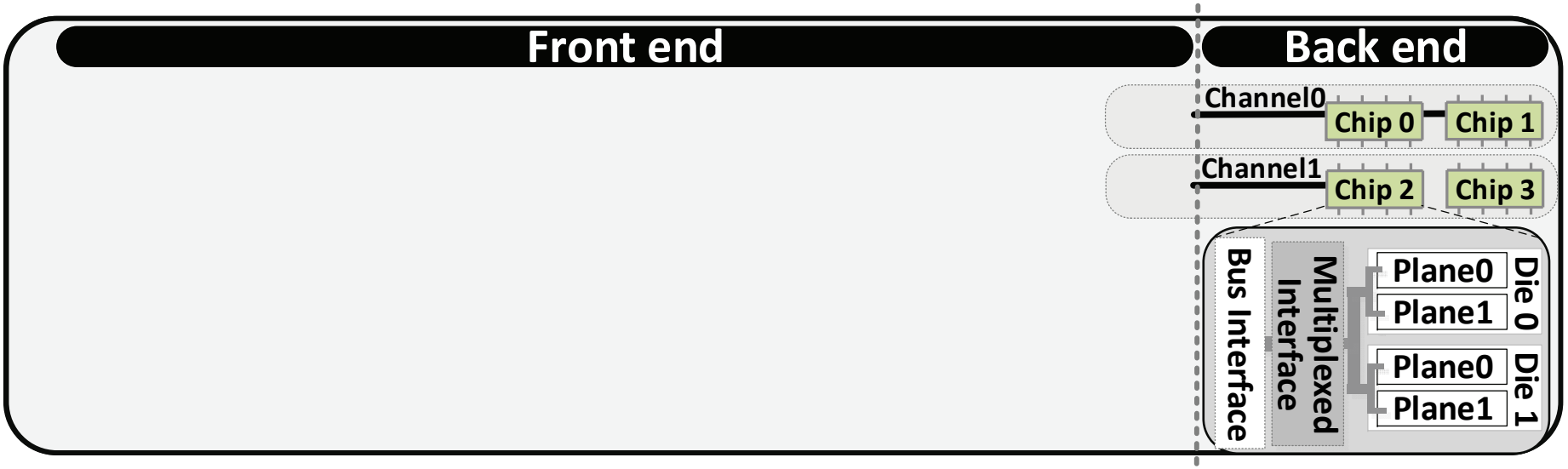
- **Solid-state drives (SSDs) are evolving to keep pace with performance, storage density demands**
  - Multi-queue SSDs (MQ-SSDs) use **new host protocols** (e.g., NVMe)
  - New SSDs make use of **emerging storage technologies** (e.g., PCM, 3D XPoint)
- **Existing simulators have not kept up with these changes**
  - They **do not support several major features**: multi-queue protocols, efficient steady-state SSD models, full end-to-end request latency
  - Compared to real MQ-SSDs, best existing simulator has **68–85% error rate**
- **We introduce MQSim, a new open-source simulator**
  - Models **all major features** of conventional SSDs *and* newer MQ-SSDs
  - Available with **full-system simulator integration**: accurate application modeling
  - Enables several new research directions
- **MQSim is highly accurate**
  - **Validated against four real MQ-SSDs**
  - Error rate for real workloads: **8–18%**

http://github.com/CMU-SAFARI/MQSim

Outline

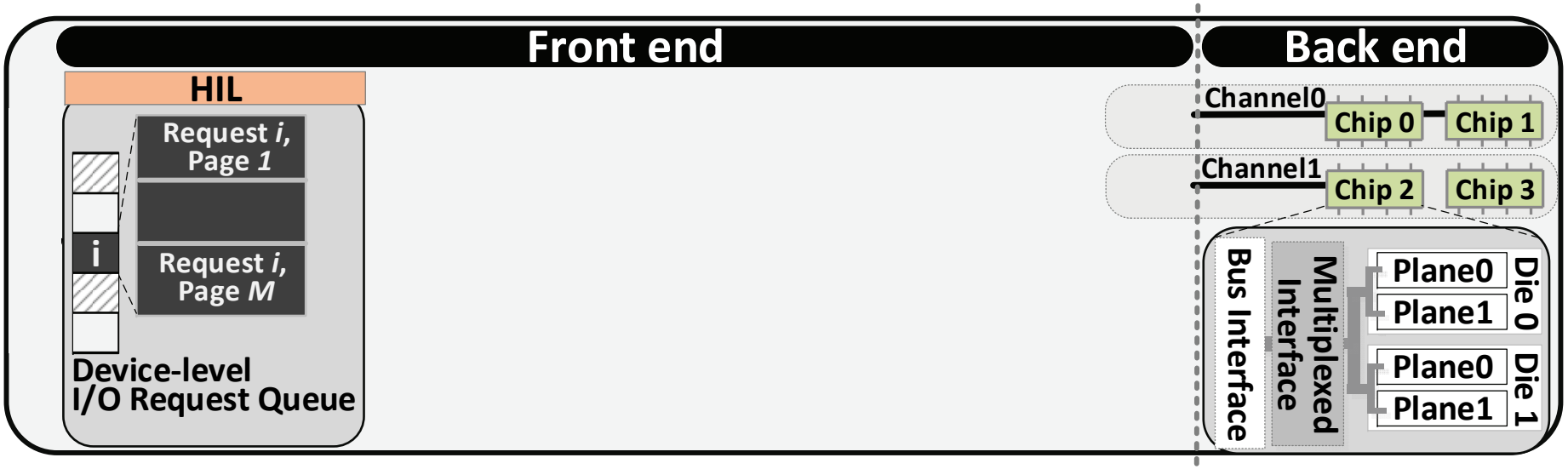
- Executive Summary
- **Looking Inside a Modern SSD**
- Challenges of Modeling Modern Multi-Queue SSDs
- MQSim: A New Simulator for Modern SSDs
- Evaluating MQSim Accuracy
- Research Directions Enabled by MQSim
- Conclusion

Internal Components of a Modern SSD



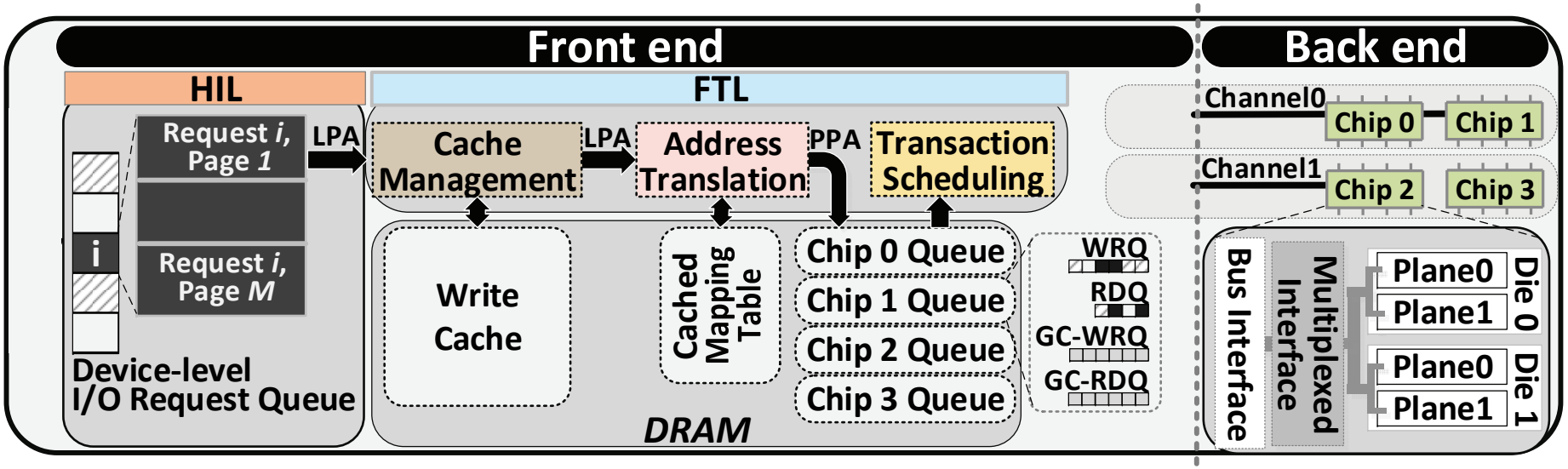
- **Back End:** data storage
  - Memory chips (e.g., NAND flash memory, PCM, MRAM, 3D XPoint)

Internal Components of a Modern SSD



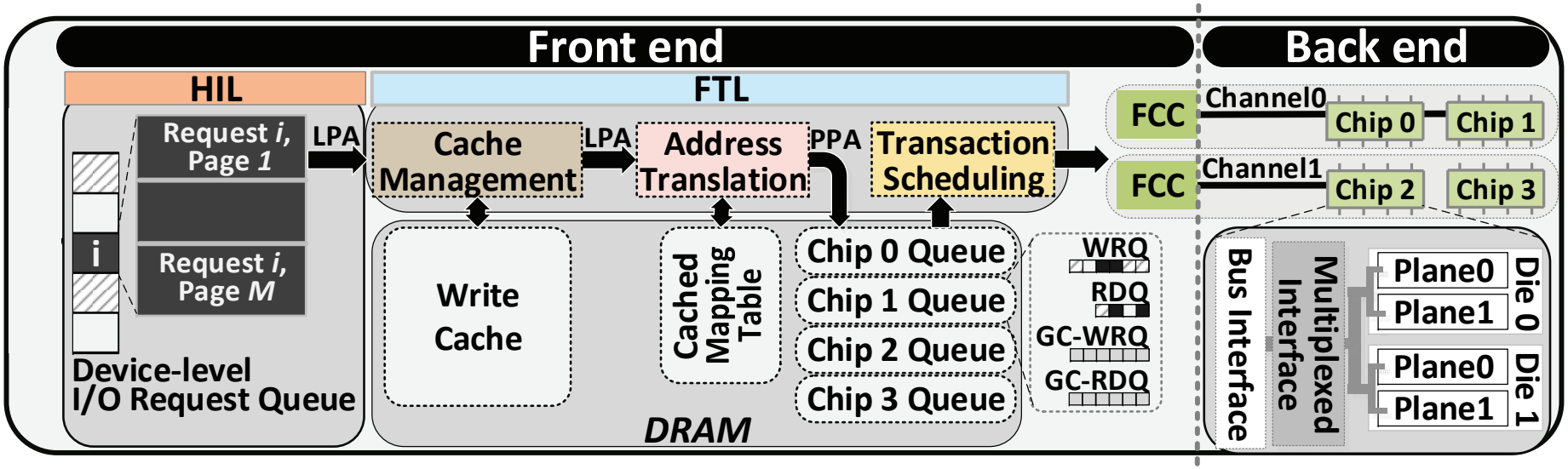
- Back End: data storage
  - Memory chips (e.g., NAND flash memory, PCM, MRAM, 3D XPoint)
- Front End: management and control units
  - Host–Interface Logic (HIL): protocol used to communicate with host

Internal Components of a Modern SSD



- Back End: data storage
  - Memory chips (e.g., NAND flash memory, PCM, MRAM, 3D XPoint)
- Front End: management and control units
  - Host–Interface Logic (HIL): protocol used to communicate with host
  - Flash Translation Layer (FTL): manages resources, processes I/O requests

Internal Components of a Modern SSD

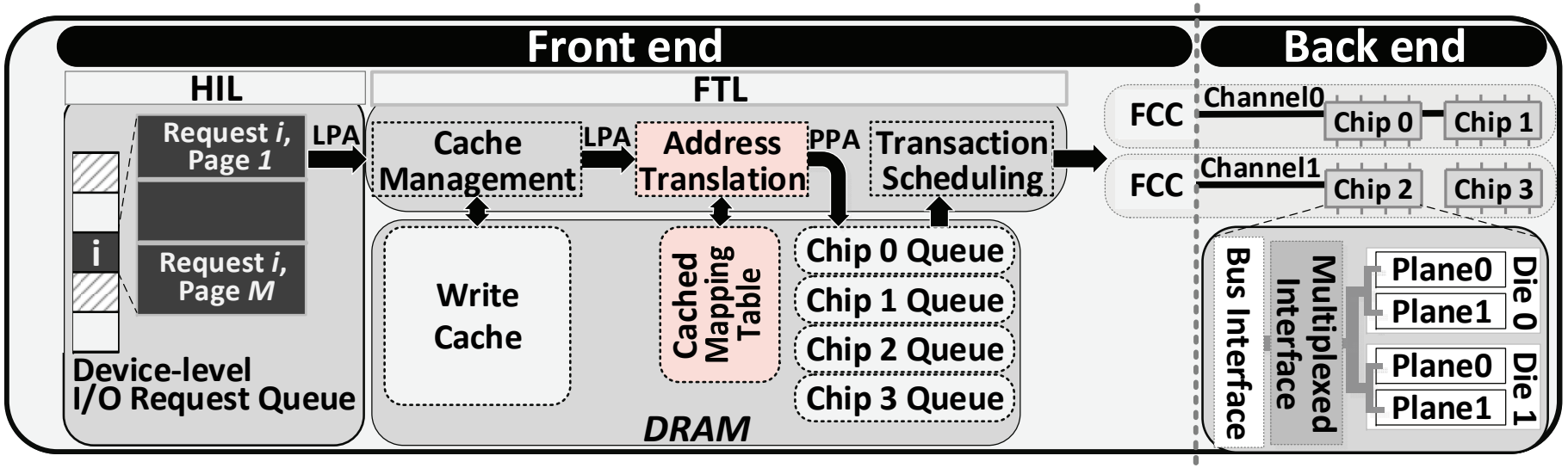


- **Back End:** data storage
  - Memory chips (e.g., NAND flash memory, PCM, MRAM, 3D XPoint)
- **Front End:** management and control units
  - Host–Interface Logic (HIL): protocol used to communicate with host
  - Flash Translation Layer (FTL): manages resources, processes I/O requests
  - **Flash Channel Controllers (FCCs):** sends commands to, transfers data with memory chips in back end

FTL: Managing the SSD's Resources

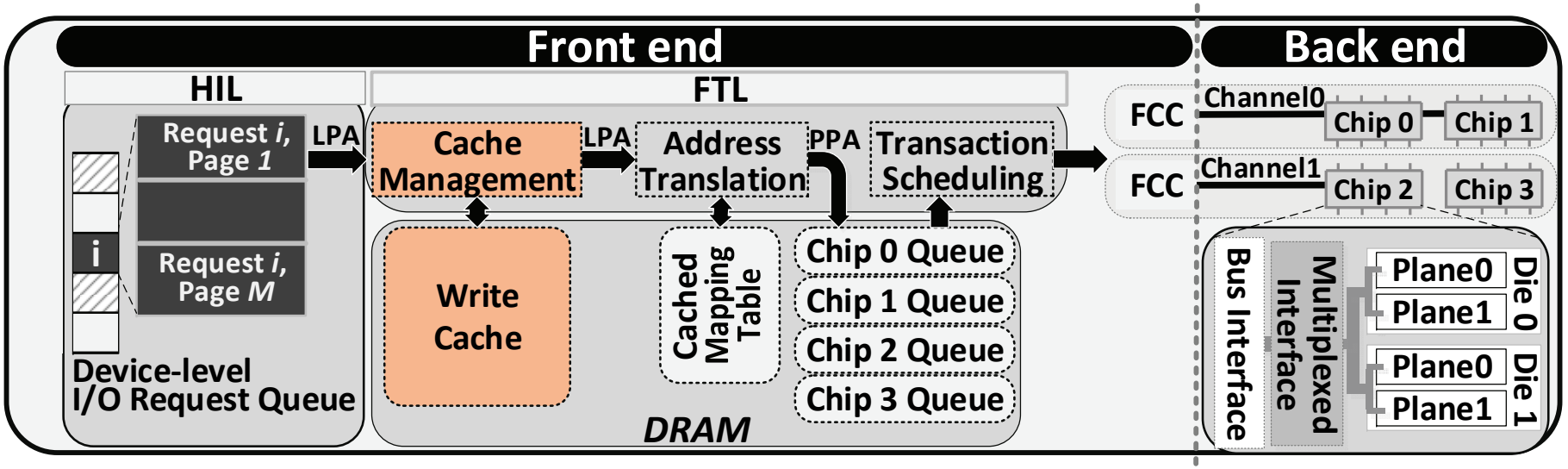
SAFARI

- Flash writes can take place only to pages that are erased
  - Perform **out-of-place updates** (i.e., write data to a different, free page), mark **old page as invalid**
  - **Update logical-to-physical mapping** (makes use of *cached mapping table*)
  - Some time later: **garbage collection** reclaims invalid physical pages *off the critical path of latency*



FTL: Managing the SSD's Resources

- Flash writes can take place only to pages that are erased
  - Perform **out-of-place updates** (i.e., write data to a different, free page), mark **old page as invalid**
  - **Update logical-to-physical mapping** (makes use of *cached mapping table*)
  - Some time later: **garbage collection** reclaims invalid physical pages *off the critical path of latency*
- **Write cache**: decreases resource contention, reduces latency



Outline

- Executive Summary
- Looking Inside a Modern SSD
- **Challenges of Modeling Modern Multi-Queue SSDs**
- MQSim: A New Simulator for Modern SSDs
- Evaluating MQSim Accuracy
- Research Directions Enabled by MQSim
- Conclusion

How Well Do Simulators Model Modern SSDs?

- State-of-the-art simulators designed for ***conventional*** SSDs
- We compare the performance of several simulators to four real modern ***multi-queue SSDs*** (MQ-SSDs)

Simulator	Error Rate vs. Real MQ-SSDs			
	SSD-A	SSD-B	SSD-C	SSD-D
SSDModel	91%	155%	196%	136%
FlashSim	99%	259%	310%	138%
SSDSim	70%	68%	74%	85%
WiscSim	95%	277%	324%	135%

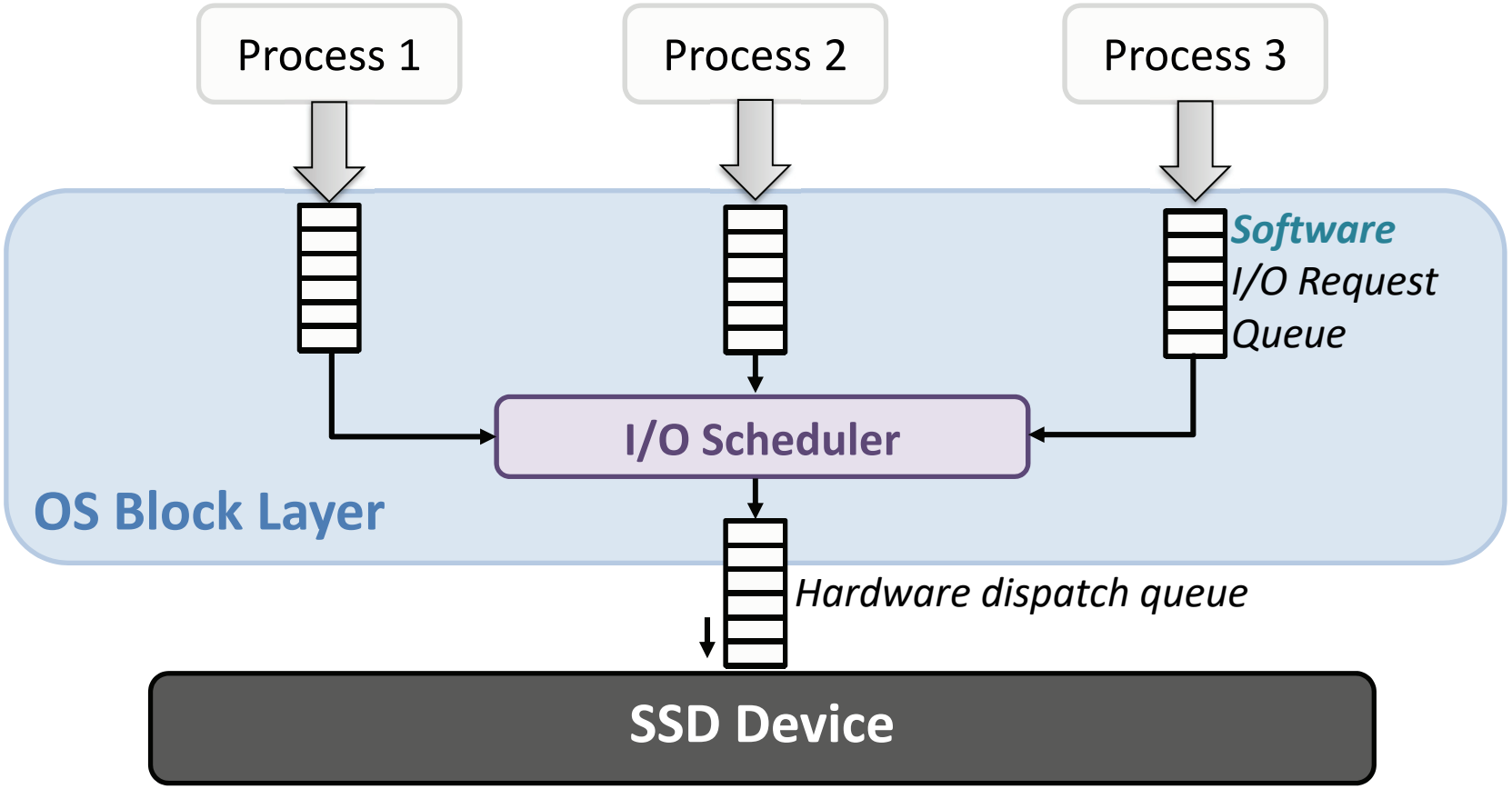
Why is the error rate so high?

- **Major features missing** in most existing simulators
  - **Multi-queue protocols** (e.g., NVMe)
  - Efficient, high-performance model of **steady-state behavior**
  - Full model of **end-to-end request latency**

Challenge 1: Support for Multi-Queue Protocols

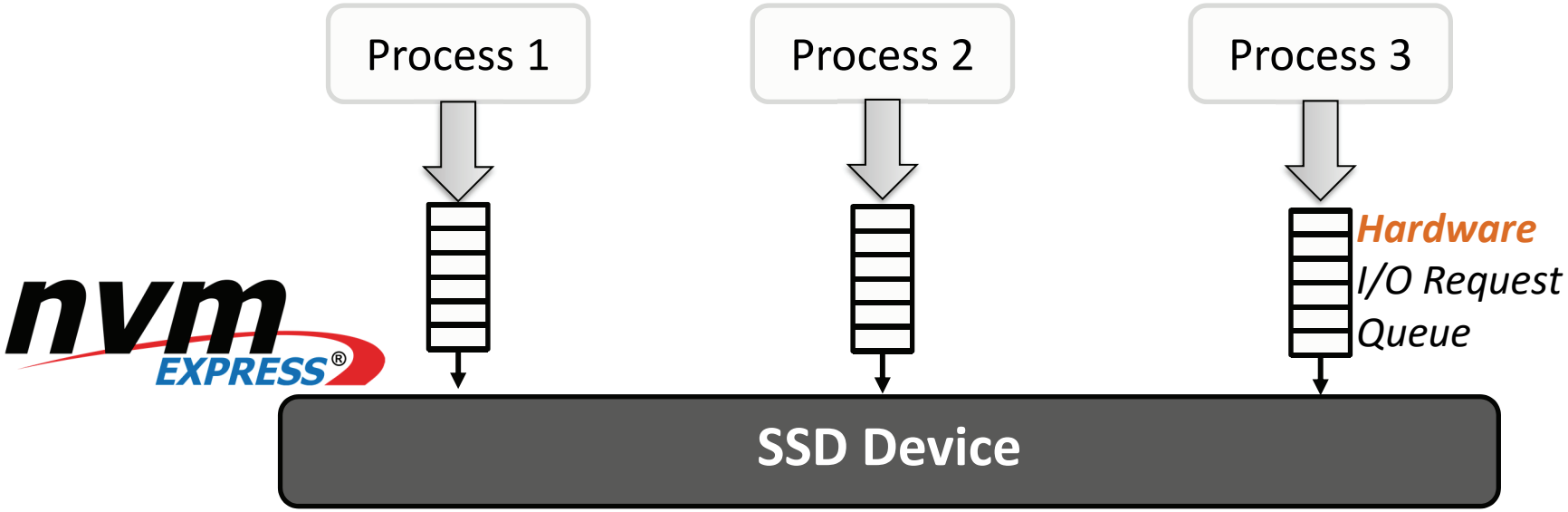
SAFARI

- Conventional host interface (e.g., SATA)
  - Designed for magnetic hard disk drives: only **thousands of IOPS** per device
  - **OS handles scheduling, fairness** control for I/O requests



Challenge 1: Support for Multi-Queue Protocols

- Modern host interface (e.g., NVMe)
  - Takes advantage of SSD throughput: enables **millions of IOPS** per device
  - Bypasses OS intervention: **SSD must perform scheduling, ensure fairness**



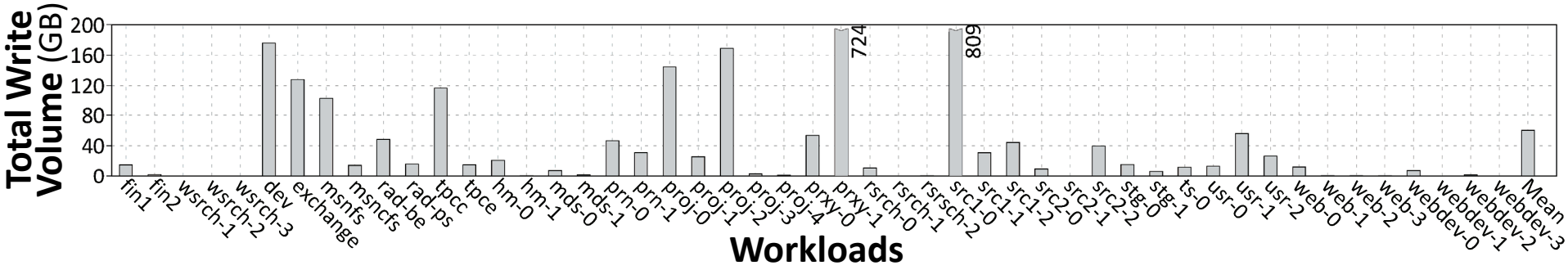
No existing SSD simulator models this today

In the paper: studies on how multiple queues in real MQ-SSDs affect performance, fairness

Challenge 2: High-Performance Steady-State Model

SAFARI

- SNIA: SSDs should be **evaluated in steady state**
  - *Fresh, out-of-the-box* (FOB) device **unlikely to perform garbage collection**
  - **Write cache not warmed up** for an FOB device
- Many previous SSD studies incorrectly simulate FOB devices
- Difficult to reach steady state in most simulators
  - **Very slow** (e.g., SSDSim execution time increases by up to 80x)
  - **Widely-used traces aren't large enough for proper warm-up**

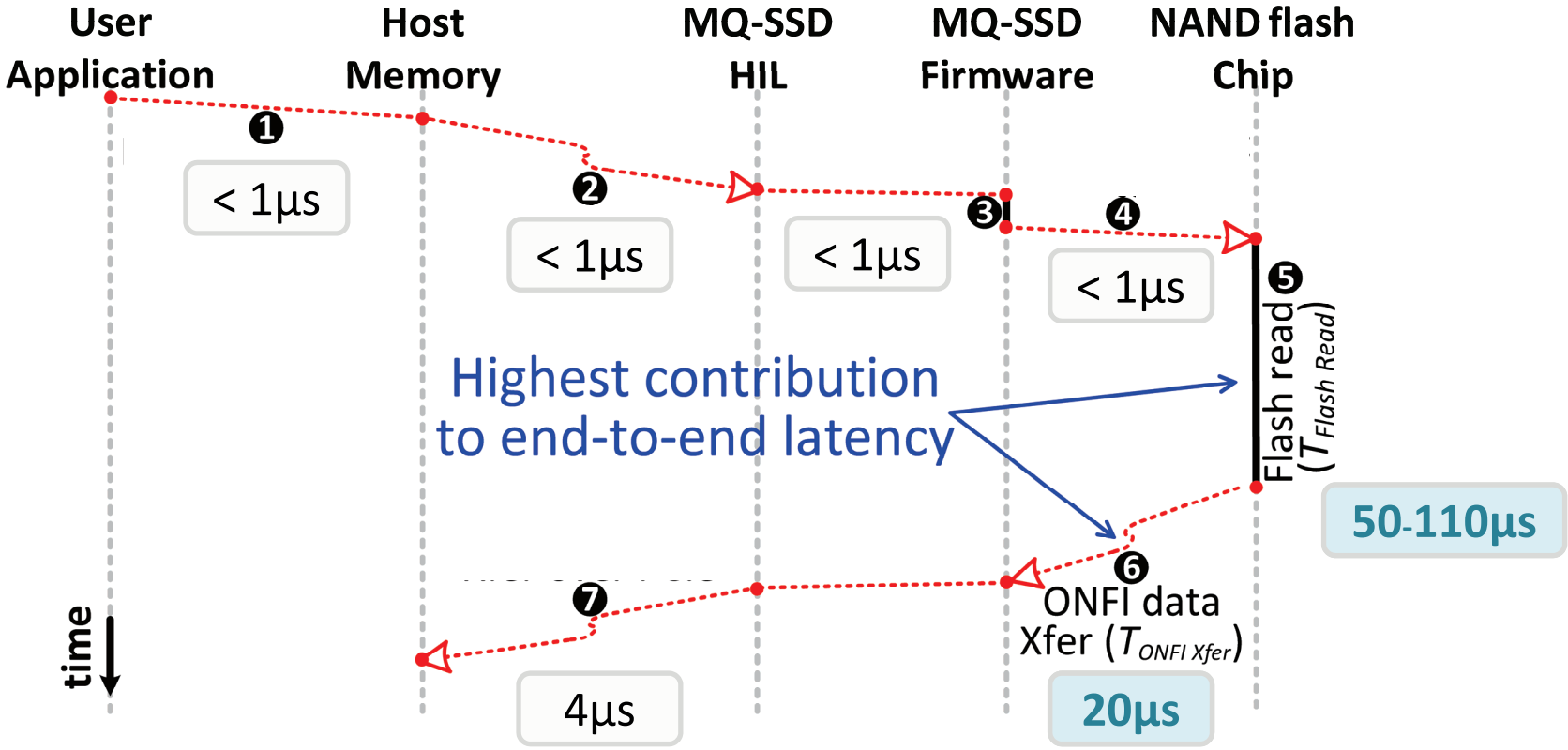


Existing simulators either don't model steady state or are slow at reaching steady state

Challenge 3: Complete Model of Request Latency

SAFARI

- Request to NAND flash based SSD

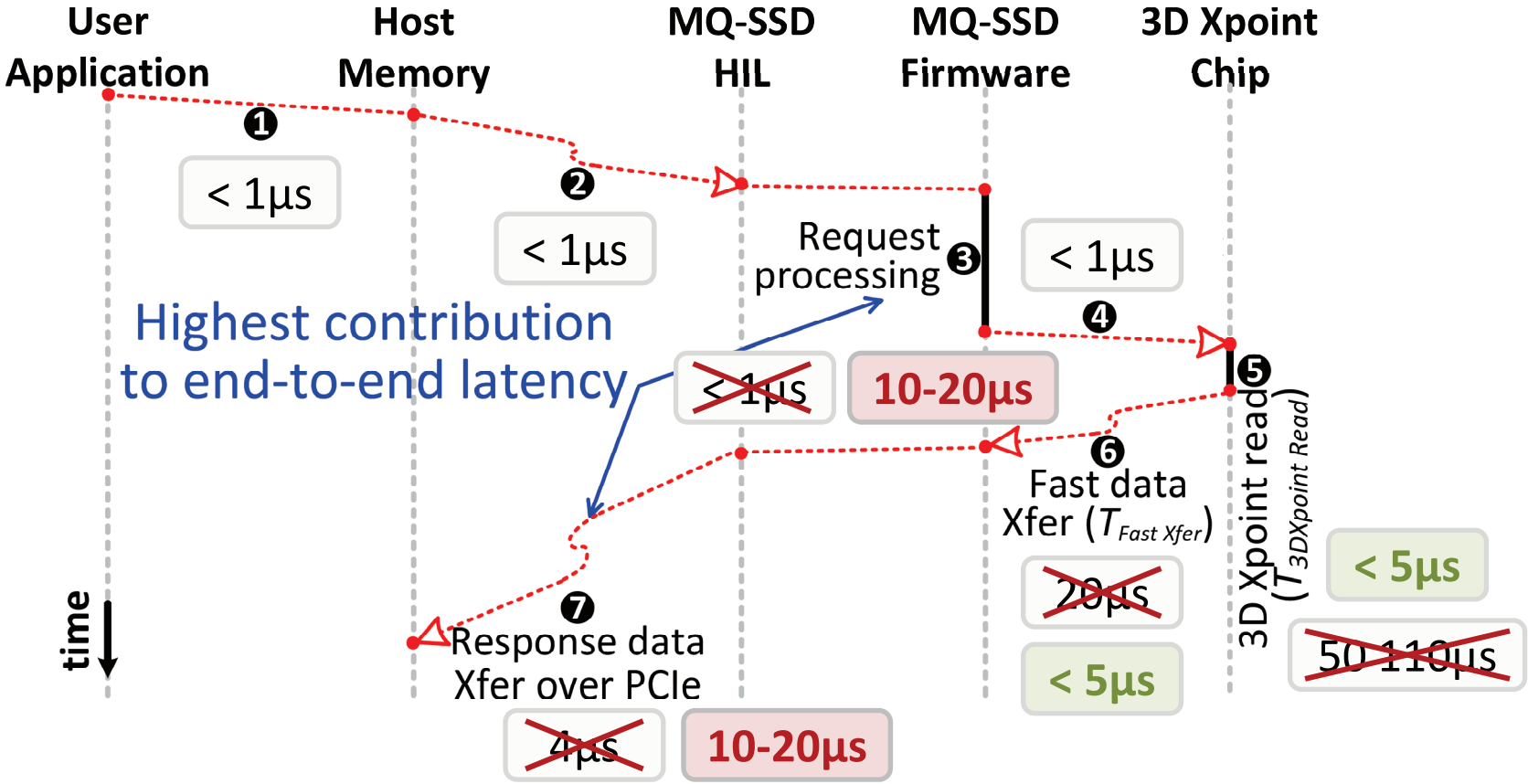


- Current simulators often model **only steps 5 and 6**
- What if we use a *different* non-volatile memory (NVM)?

Challenge 3: Complete Model of Request Latency

SAFARI

- Request to 3D XPoint based SSD



Existing simulators don't model all of the request latency, causing inaccuracies for new NVMs

Summary: Simulation Challenges of MQ-SSDs

- **Major features missing** in most existing simulators
  - **Multi-queue protocols** (e.g., NVMe)
  - Efficient, high-performance model of **steady-state behavior**
  - Full model of **end-to-end request latency**
- Missing features lead to **high inaccuracy**

OUR GOAL

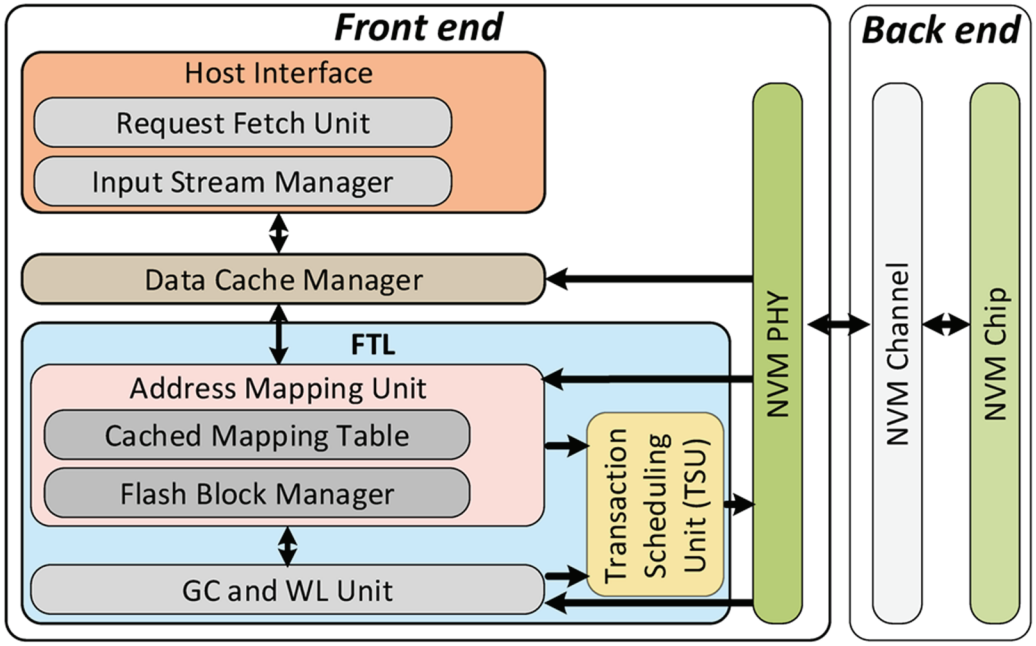
Develop a new simulator that *faithfully* models features of both modern MQ-SSDs and conventional SSDs

Outline

- Executive Summary
- Looking Inside a Modern SSD
- Challenges of Modeling Modern Multi-Queue SSDs
- **MQSim: A New Simulator for Modern SSDs**
- Evaluating MQSim Accuracy
- Research Directions Enabled by MQSim
- Conclusion

Major Features of MQSim

- **Accurately models conventional SATA-based SSDs *and* modern MQ-SSDs**
  - Multi-queue protocols
  - Support for efficient modeling of steady-state behavior
  - Full model of end-to-end I/O request latency



- **Flexible design**
  - Modular components
  - Integration with gem5 **full-system simulator**
  - Ability to support **emerging non-volatile memory (NVM) technologies**

- **Open-source release:** http://github.com/CMU-SAFARI/MQSim
  - Written in C++
  - MIT License

Experimental Methodology

- Compare MQSim to **four real, modern data center SSDs**

Code	Production Year	Capacity	Flash Technology
SSD-A	2016	800 GB	MLC
SSD-B	2016	256 GB	MLC
SSD-C	2017	1 TB	TLC
SSD-D	2016	512 GB	TLC

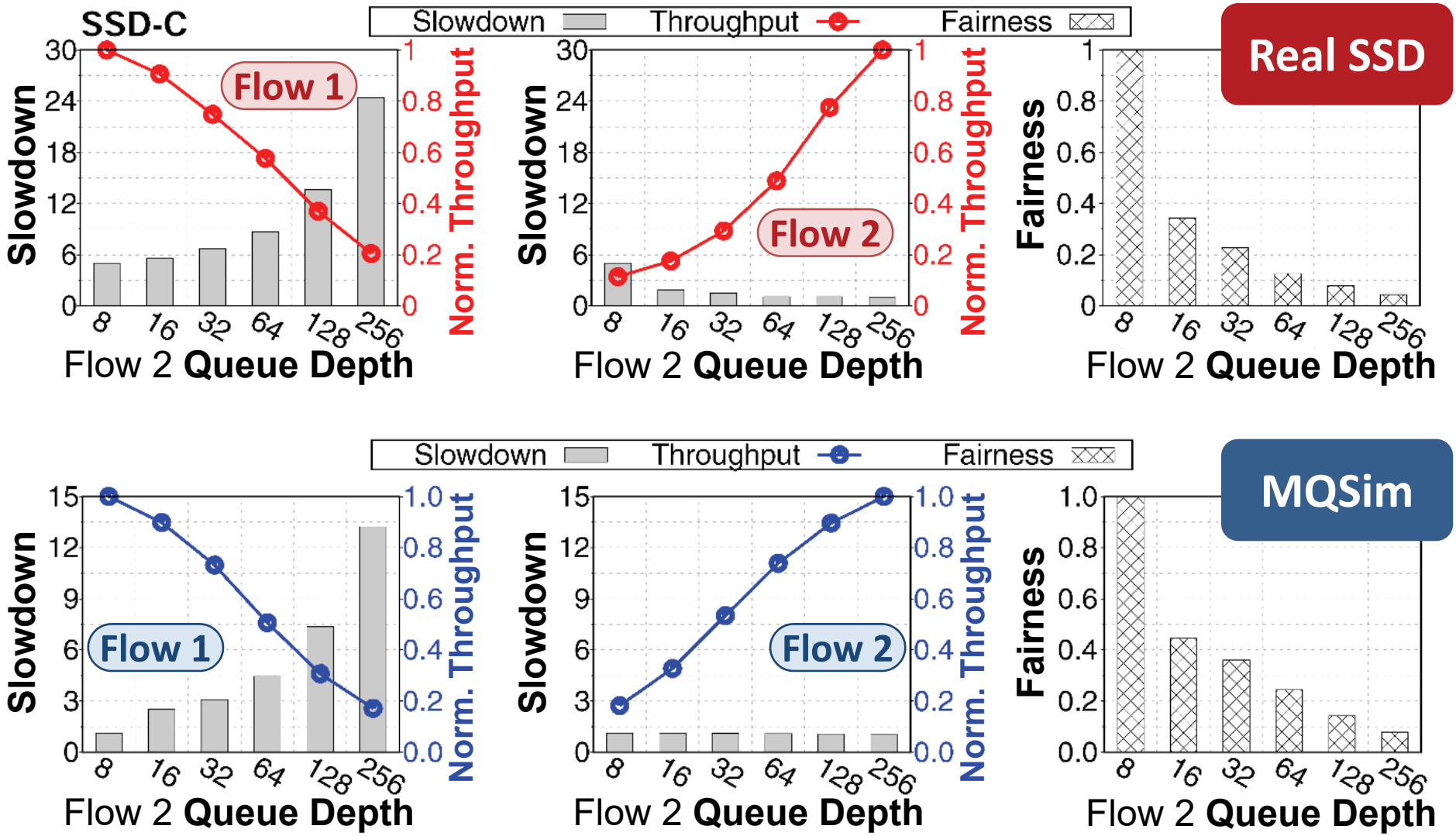
  - System config: Intel Xeon E3-1240, 32GB DDR4, Ubuntu 16.04.2
  - All SSDs use NVMe protocol over PCIe bus
  - Real SSDs preconditioned by writing to 70% of available logical space
  - **MQSim parameters set to model each SSD**

- We test using **multiple I/O flows** (1 flow = 1 application)
  - **Queue depth** controls flow **greediness** (number of simultaneous requests)
  - We show *representative results* here; full results in our paper

MQSim Multi-Queue Model

SAFARI

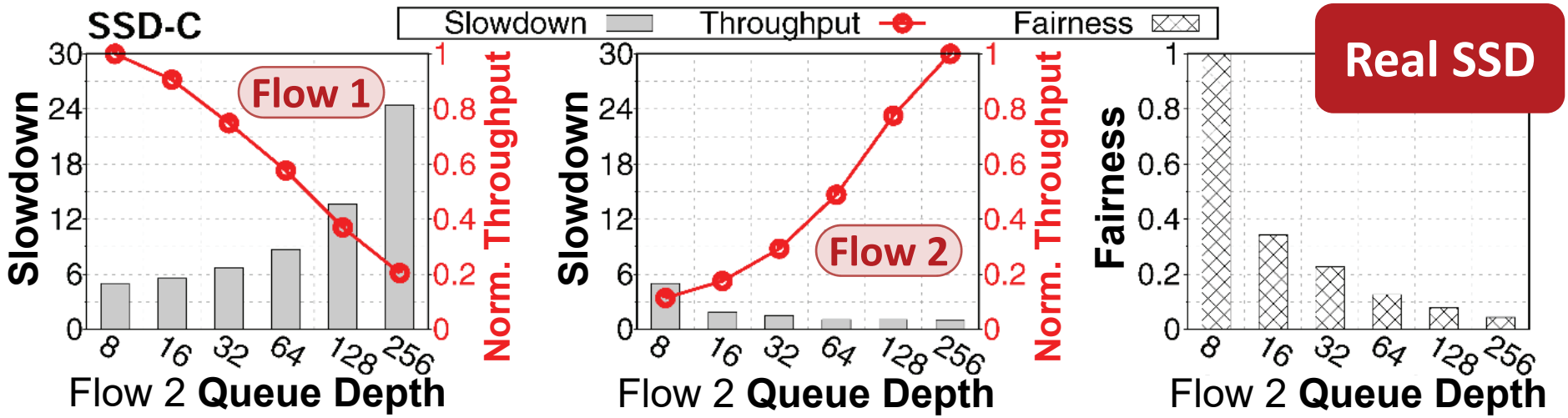
- **Inter-flow interference** exists in many state-of-the-art SSDs

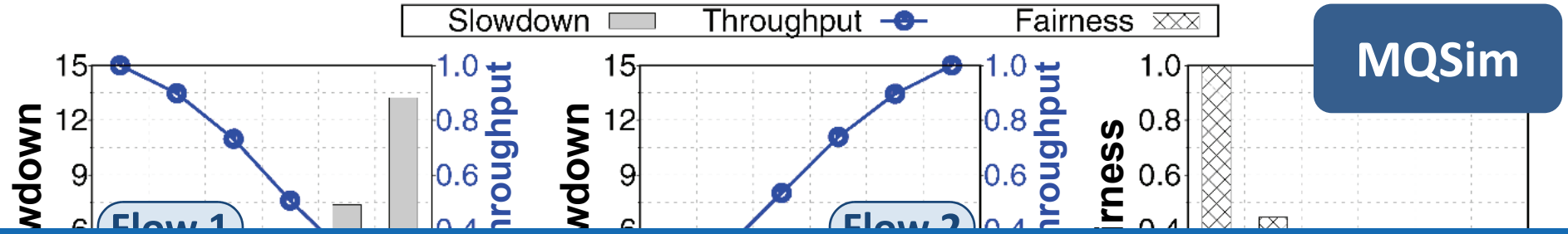


MQSim Multi-Queue Model

SAFARI

- **Inter-flow interference** exists in many state-of-the-art SSDs



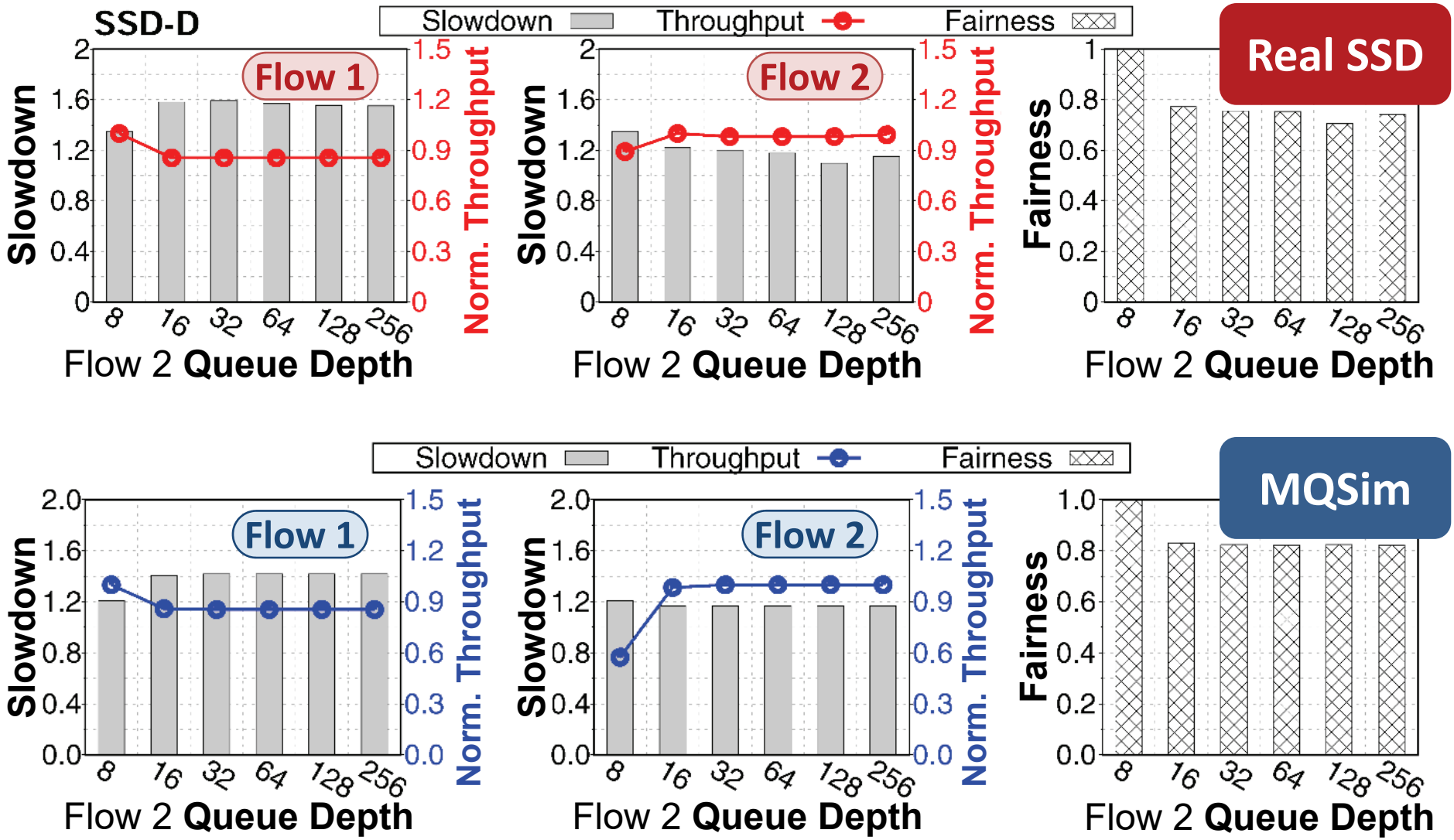


MQSim accurately captures inter-flow interference when one of two flows hogs I/O request bandwidth

MQSim Multi-Queue Model

SAFARI

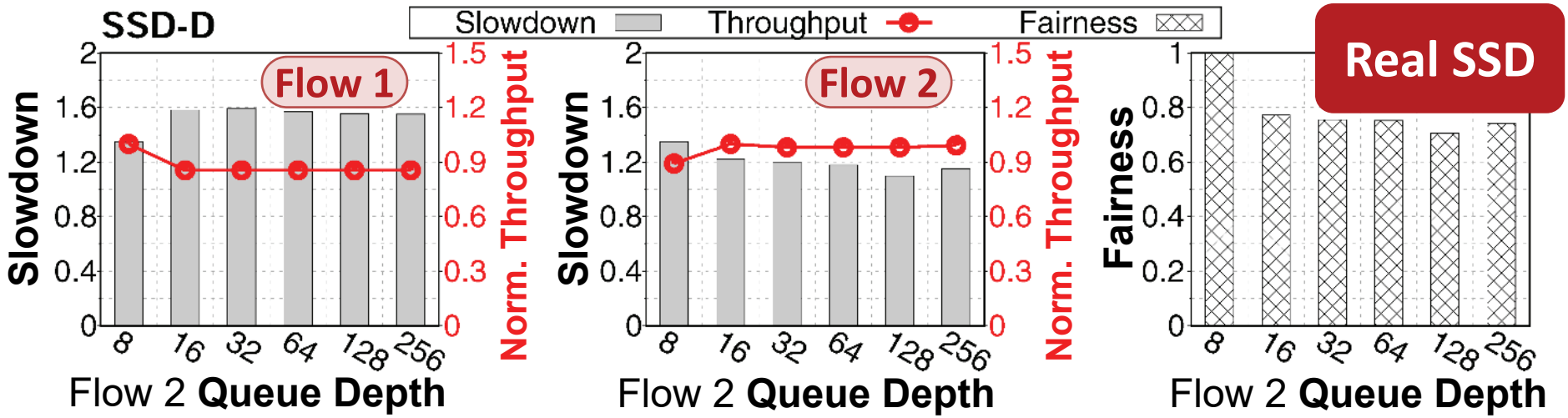
- Some SSDs have mechanisms to *control* inter-flow interference



MQSim Multi-Queue Model

SAFARI

- Some SSDs have mechanisms to *control* inter-flow interference

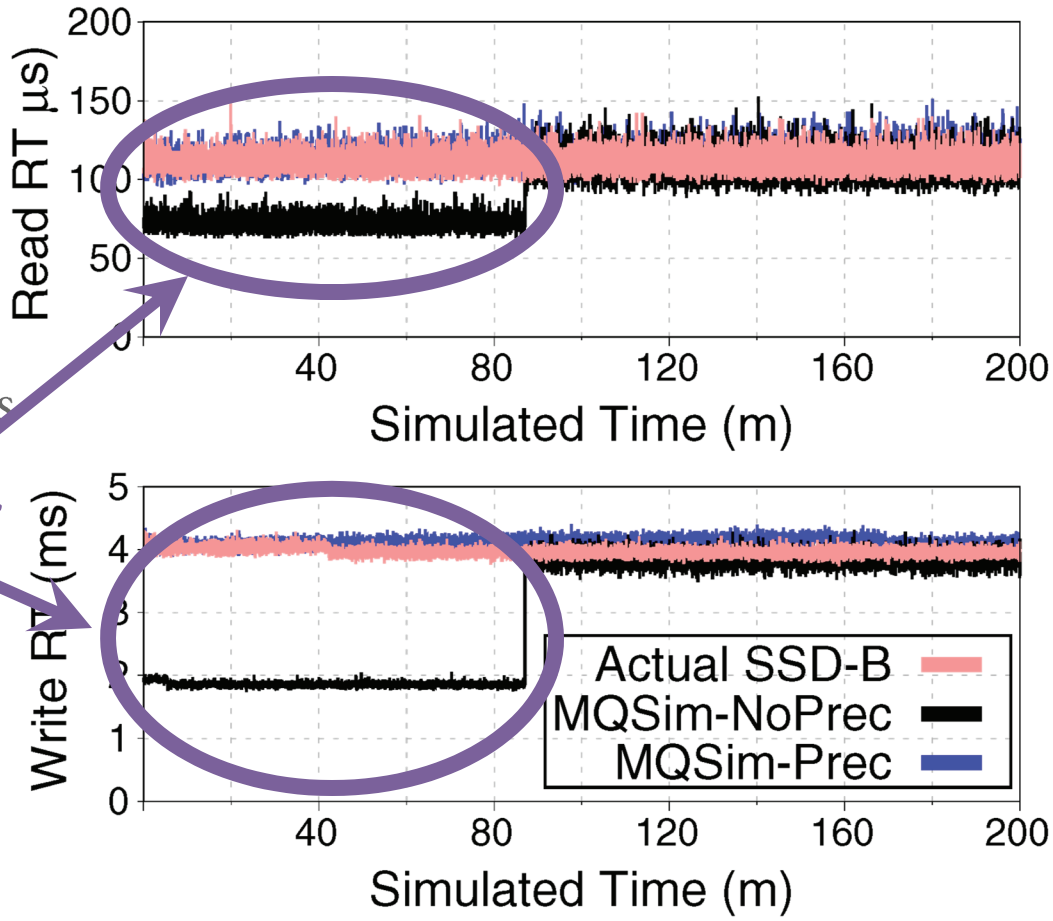


MQSim models the impact of control mechanisms for inter-flow interference in modern MQ-SSDs

Capturing Steady-State Behavior with MQSim

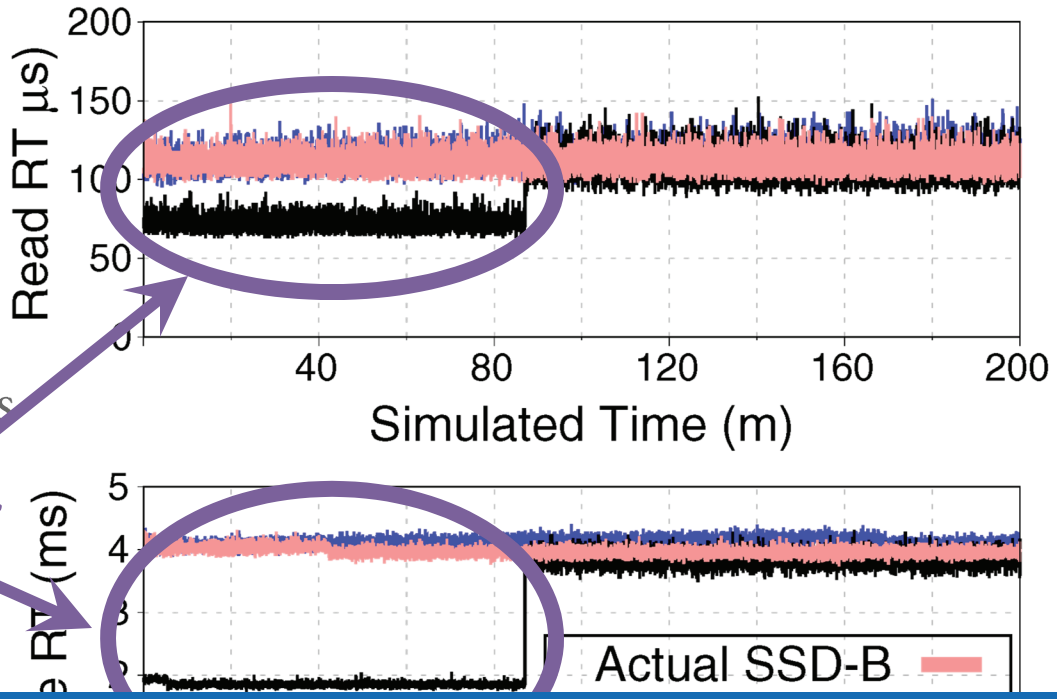
- MQSim includes an **efficient SSD preconditioning** mechanism
  - **Very fast**: does *not* need to execute actual requests
  - **Can be disabled** to simulate fresh, out-of-the-box (FOB) device
- Two-pass approach
  - Read input trace, perform statistical analysis
  - Scan entire storage space, change status of each physical page based on analysis

Response time (RT) differences between FOB SSDs, SSDs in steady state



Capturing Steady-State Behavior with MQSim

- MQSim includes an **efficient SSD preconditioning** mechanism
  - **Very fast**: does *not* need to execute actual requests
  - **Can be disabled** to simulate fresh, out-of-the-box (FOB) device
- Two-pass approach
  - Read input trace, perform statistical analysis
  - Scan entire storage space, change status of each physical page based on analysis



Response time (RT) differences between FOB SSDs, SSDs in steady state

With preconditioning, MQSim accurately captures the behavior of FOB SSDs *and* steady-state SSDs

Outline

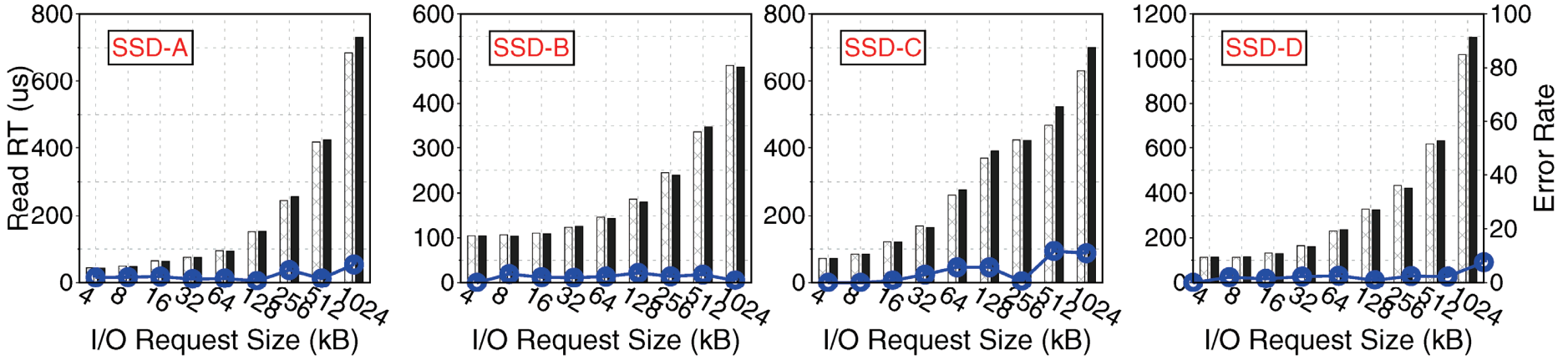
- Executive Summary
- Looking Inside a Modern SSD
- Challenges of Modeling Modern Multi-Queue SSDs
- MQSim: A New Simulator for Modern SSDs
- **Evaluating MQSim Accuracy**
- Research Directions Enabled by MQSim
- Conclusion

MQSim Accurately Captures Read/Write Latencies

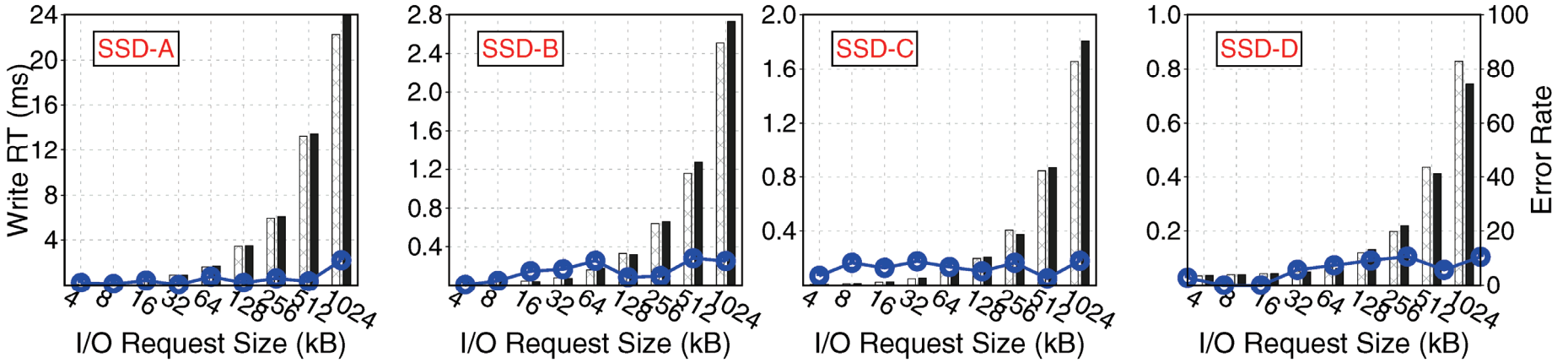
SAFARI

- Evaluations using two synthetic flows

Flow A: All Read Requests



Flow B: All Write Requests

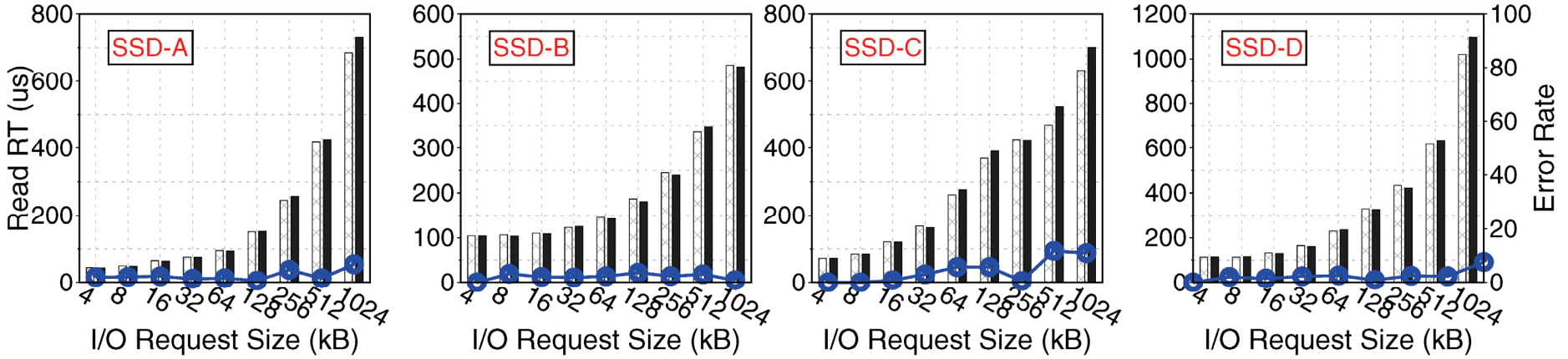


MQSim Accurately Captures Read/Write Latencies

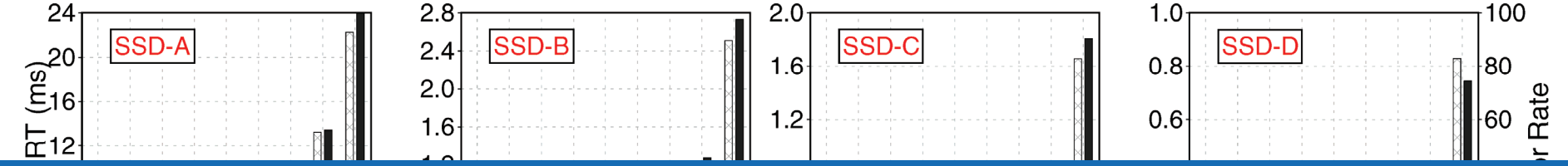
SAFARI

- Evaluations using two synthetic flows

Flow A: All Read Requests



Flow B: All Write Requests



Error rate vs. real SSDs, averaged across all four SSDs

Read latency: 2.9% **Write latency: 4.9%**

MQSim Is More Accurate Than Existing Simulators SAFARI

- Experiments using three real workload traces
  - Microsoft enterprise traces: TPC-C, TPC-E, Exchange Server
  - Comparison of total workload **execution time**

Simulator	Error Rate vs. Real MQ-SSDs			
	SSD-A	SSD-B	SSD-C	SSD-D
SSDModel	91%	155%	196%	136%
FlashSim	99%	259%	310%	138%
SSDSim	70%	68%	74%	85%
WiscSim	95%	277%	324%	135%
MQSim	**8%**	**6%**	**18%**	**14%**

  - MQSim has a **comparable execution time** to other simulators

MQSim is an order of magnitude more accurate at modeling MQ-SSDs than state-of-the-art simulators

Outline

- Executive Summary
- Looking Inside a Modern SSD
- Challenges of Modeling Modern Multi-Queue SSDs
- MQSim: A New Simulator for Modern SSDs
- Evaluating MQSim Accuracy
- **Research Directions Enabled by MQSim**
- Conclusion

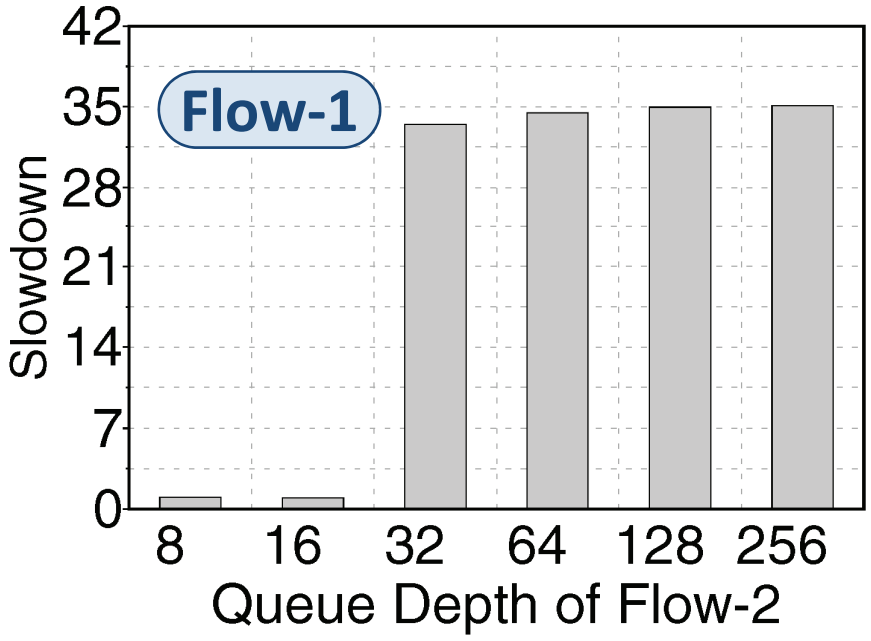
Research Directions Enabled by MQSim

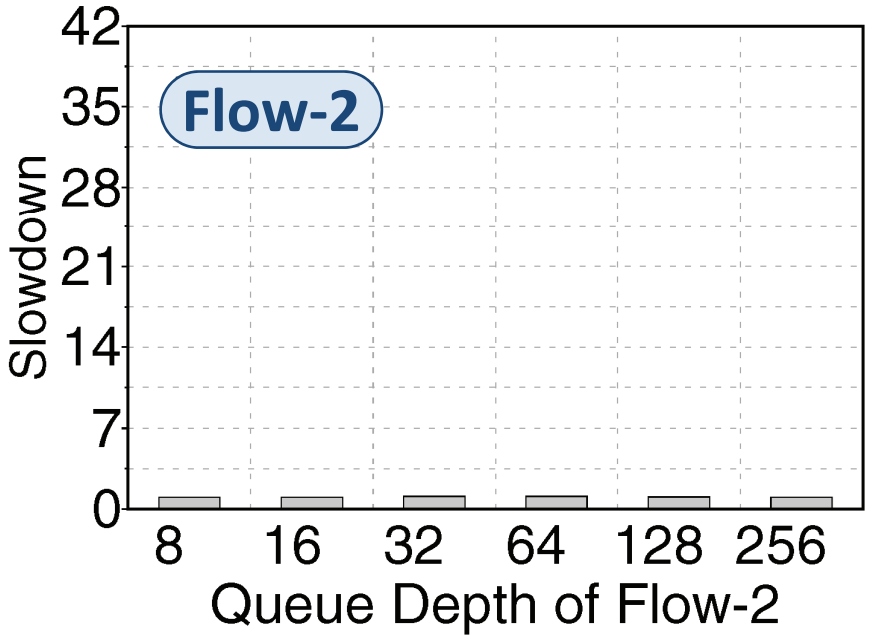
- MQSim's accurate models enable many new research studies not possible with existing simulators

- Fairness and performance effects of inter-flow interference within an SSD
  - We study three major sources of contention
    - » Write cache
    - » Cached mapping table
    - » Back end memory resources
  - Using I/O flows designed to isolate impact of contention at a single source

Interference at the Write Cache

- Two flows concurrently performing random-access writes
- Flow-1 has **high cache locality**, Flow-2 has **poor cache locality**
- We **increase Flow-2 greediness** (by adjusting queue depth)



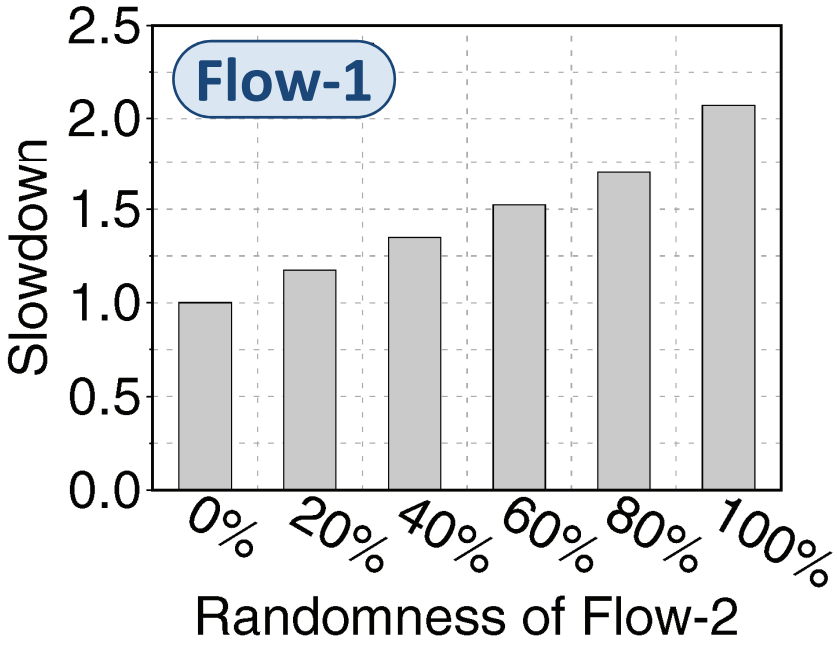


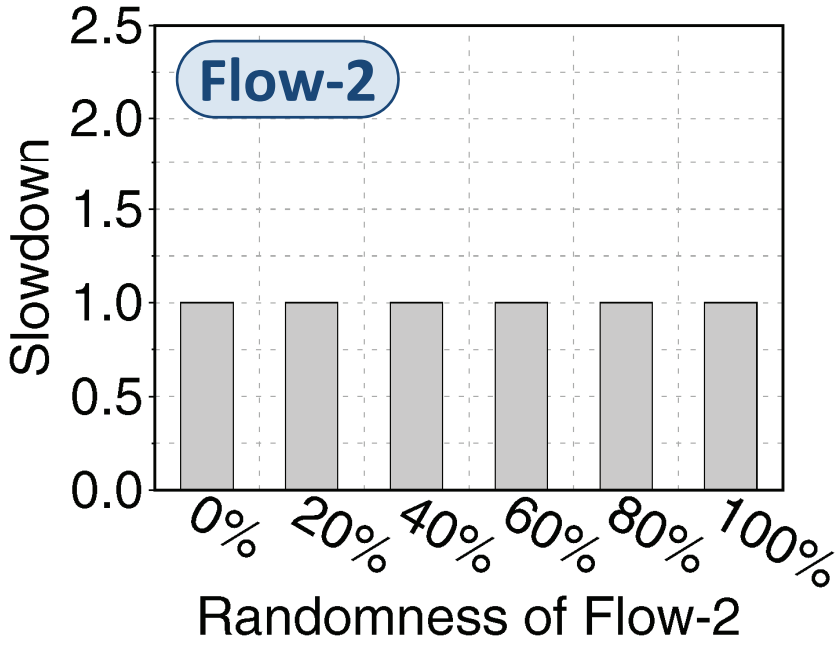
A greedier Flow-2 induces more write cache thrashing, **destroying Flow-1's cache locality**

Interference at the Cached Mapping Table (CMT)

SAFARI

- Two flows concurrently reading with different access patterns
- Flow-1 **sequential**; Flow-2 split between **sequential, random**
- We **increase randomness of Flow-2**, affecting misses to CMT

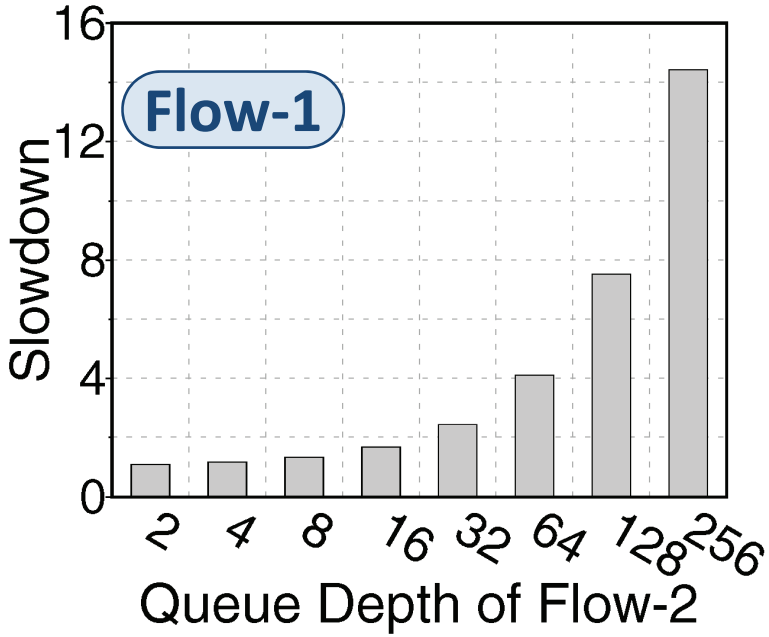


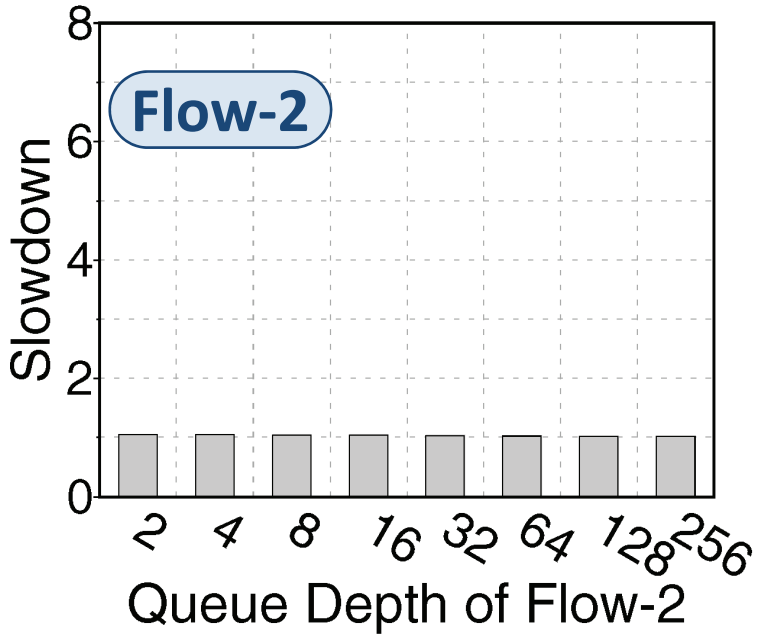


A more random Flow-2 increases CMT thrashing,
significantly lowering the CMT hit rate of Flow-1

Interference at the Back End

- Two flows concurrently performing random-access reads
- Flow-1 has low intensity, Flow-2 has higher intenity
- We increase Flow-2 greediness (by adjusting queue depth)





When Flow-2 is more intense, chip-level queues back up, **causing requests from Flow-1 to wait much longer**

Research Directions Enabled by MQSim

- MQSim's accurate models enable many new research studies not possible with existing simulators

- Fairness and performance effects of inter-flow interference within an SSD
  - We study three major sources of contention
    - » Write cache
    - » Cached mapping table
    - » Back end memory resources
  - Using I/O flows designed to isolate impact of contention at a single source

- In the paper: application-level studies
  - Application-level performance metrics (e.g., IPC, slowdown, fairness)
  - Using full applications running on MQSim integrated with gem5 full system simulator

Outline

- Executive Summary
- Looking Inside a Modern SSD
- Challenges of Modeling Modern Multi-Queue SSDs
- MQSim: A New Simulator for Modern SSDs
- Evaluating MQSim Accuracy
- Research Directions Enabled by MQSim
- **Conclusion**

Conclusion

- **Solid-state drives (SSDs) are evolving to keep pace with performance, storage density demands**
  - Multi-queue SSDs (MQ-SSDs) use **new host protocols** (e.g., NVMe)
  - New SSDs make use of **emerging storage technologies** (e.g., PCM, 3D XPoint)
- **Existing simulators have not kept up with these changes**
  - They **do not support several major features**: multi-queue protocols, efficient steady-state SSD models, full end-to-end request latency
  - Compared to real MQ-SSDs, best existing simulator has **68–85% error rate**
- **We introduce MQSim, a new open-source simulator**
  - Models **all major features** of conventional SSDs *and* newer MQ-SSDs
  - Available with **full-system simulator integration**: accurate application modeling
  - Enables several new research directions
- **MQSim is highly accurate**
  - **Validated against four real MQ-SSDs**
  - Error rate for real workloads: **8–18%**

http://github.com/CMU-SAFARI/MQSim

ETH zürich | Carnegie Mellon University | Systems@ETH Zürich | SAFARI

MQSim:

A Framework for Enabling Realistic Studies of Modern Multi-Queue SSD Devices

Arash Tavakkol, Juan Gómez-Luna,
Mohammad Sadrosadati, **Saugata Ghose**, Onur Mutlu

Download the Simulator:

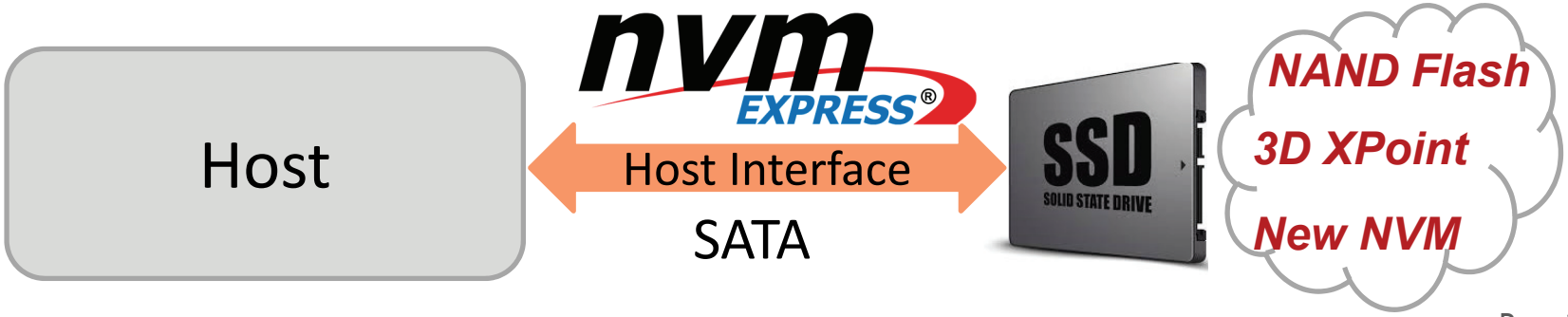
http://github.com/CMU-SAFARI/MQSim

Backup Slides

Enabling Higher SSD Performance and Capacity

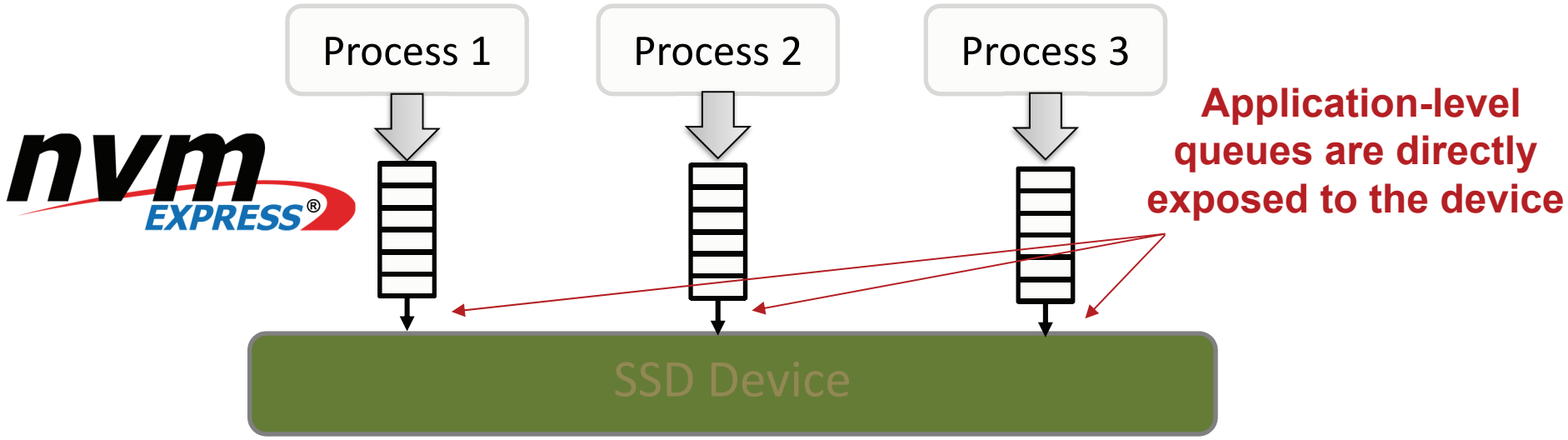
- Solid-state drives (SSDs) are widely used in today's computer systems
  - Data centers
  - Enterprise servers
  - Consumer devices

- I/O demand of both enterprise and consumer applications continues to grow
- SSDs are rapidly evolving to deliver improved performance



Simulation Challenges for Modern MQ-SSDs

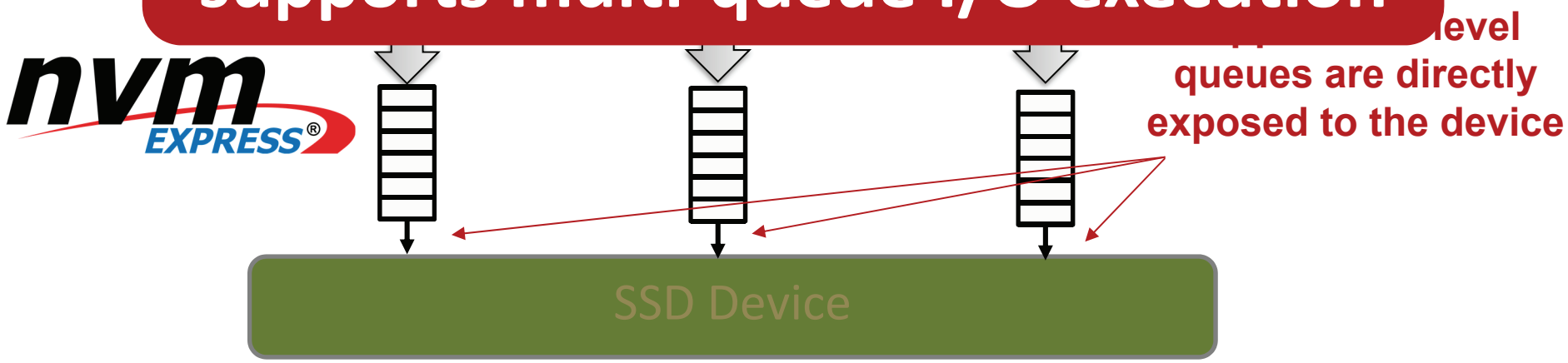
- 1. Multi-queue support in modern NVMe SSDs
  - The OS I/O scheduler is eliminated
  - The device itself is responsible for fairly servicing I/O requests from concurrently-running applications and guaranteeing high responsiveness



Simulation Challenges for Modern MQ-SSDs

- 1. Multi-queue support in modern NVMe SSDs
  - The OS I/O scheduler is eliminated
  - The device itself is responsible for fairly servicing I/O requests from concurrently-running applications and guaranteeing high responsiveness

No existing SSD simulation tool supports multi-queue I/O execution



Defining Slowdown and Fairness for I/O Flows

- RT_{fi}: response time of Flow f_i
- S_{fi}: slowdown of Flow f_i

$$S_{f_i} = RT_{f_i}^{shared} / RT_{f_i}^{alone}$$

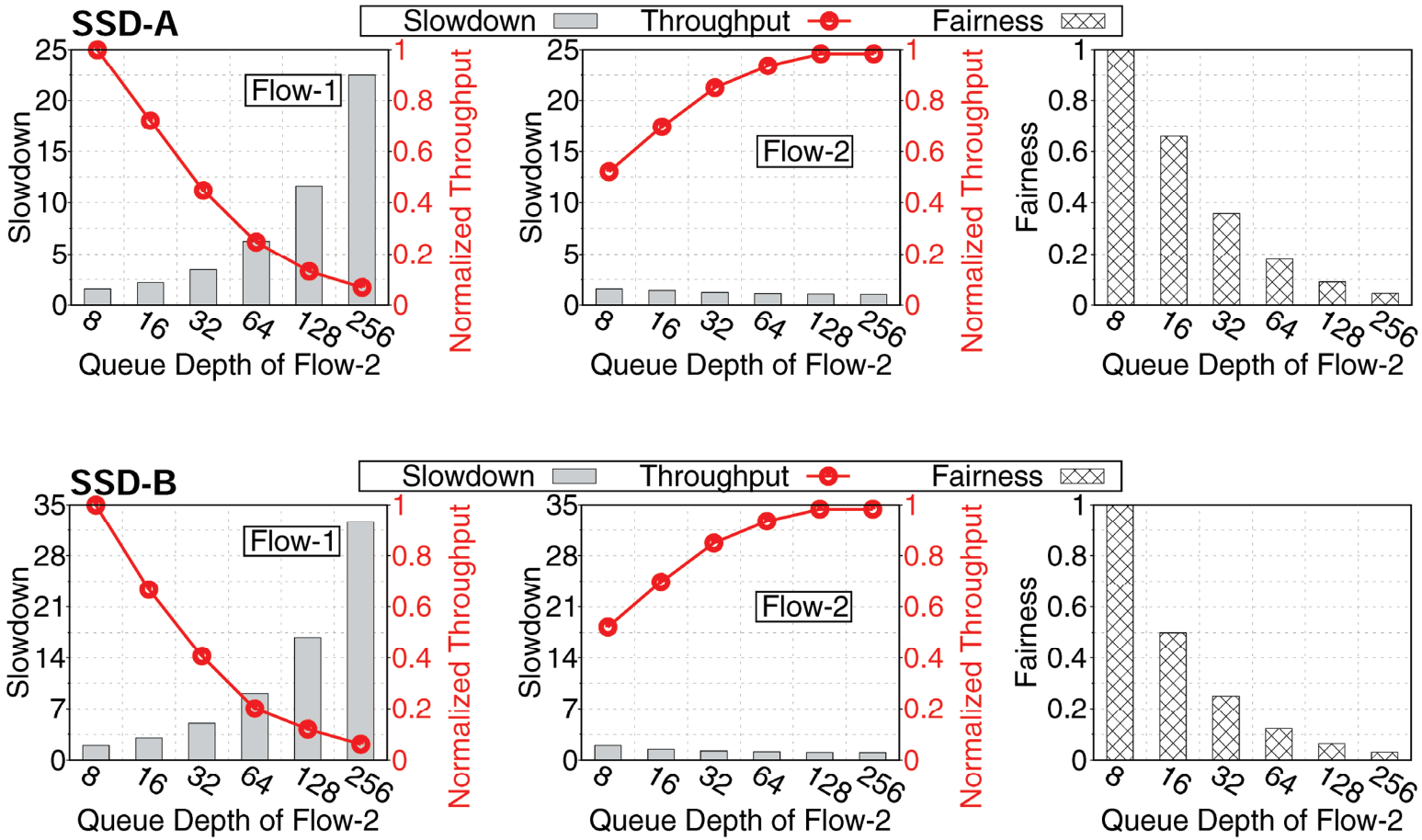
- F: fairness of slowdowns across multiple flows

$$F = \frac{\underset{i}{MIN}\{S_{f_i}\}}{\underset{i}{MAX}\{S_{f_i}\}}$$

  - $0 < F < 1$
  - Higher F means that system is *more* fair

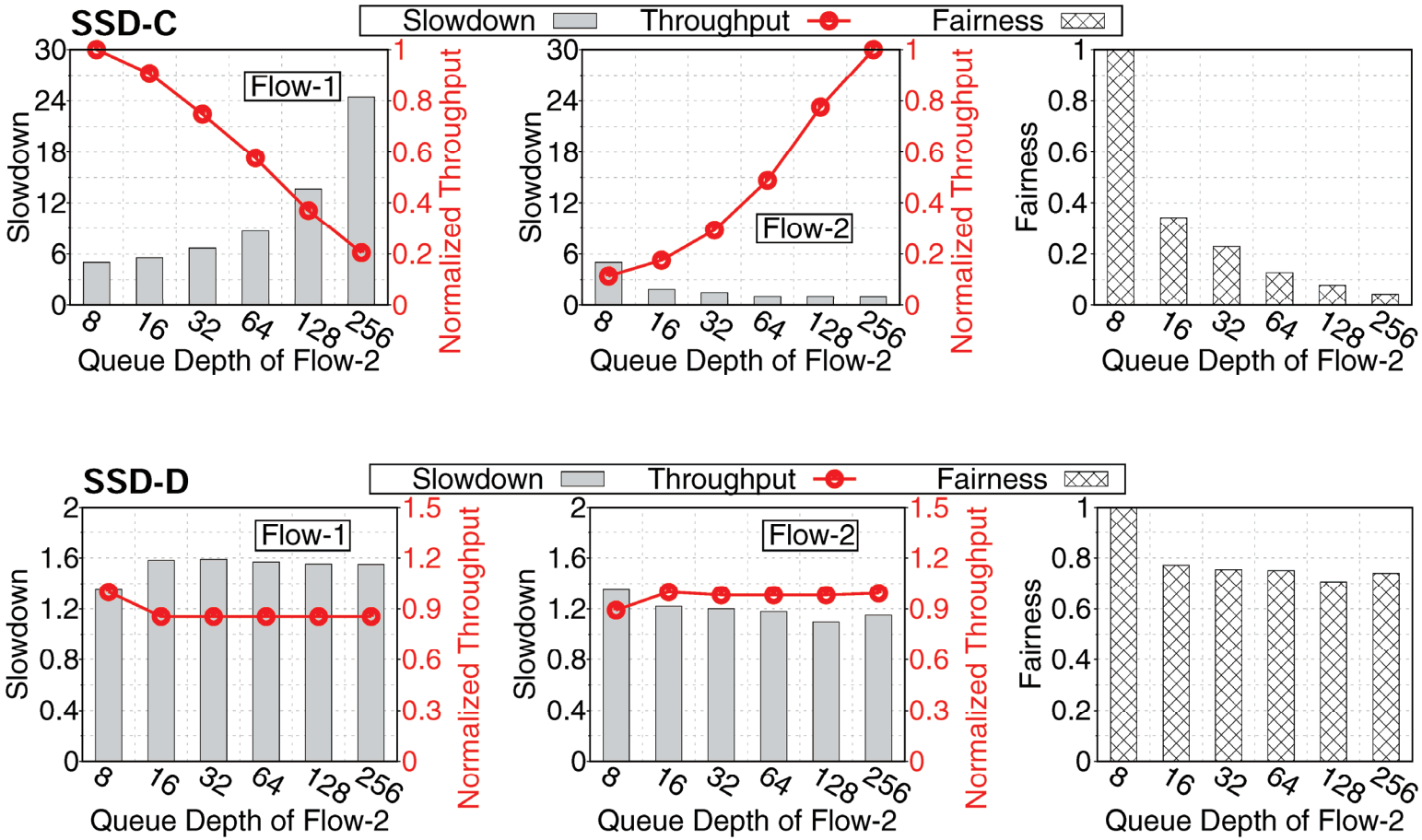
Concurrent Flow Execution in Real Devices

SAFARI



Concurrent Flow Execution in Real Devices

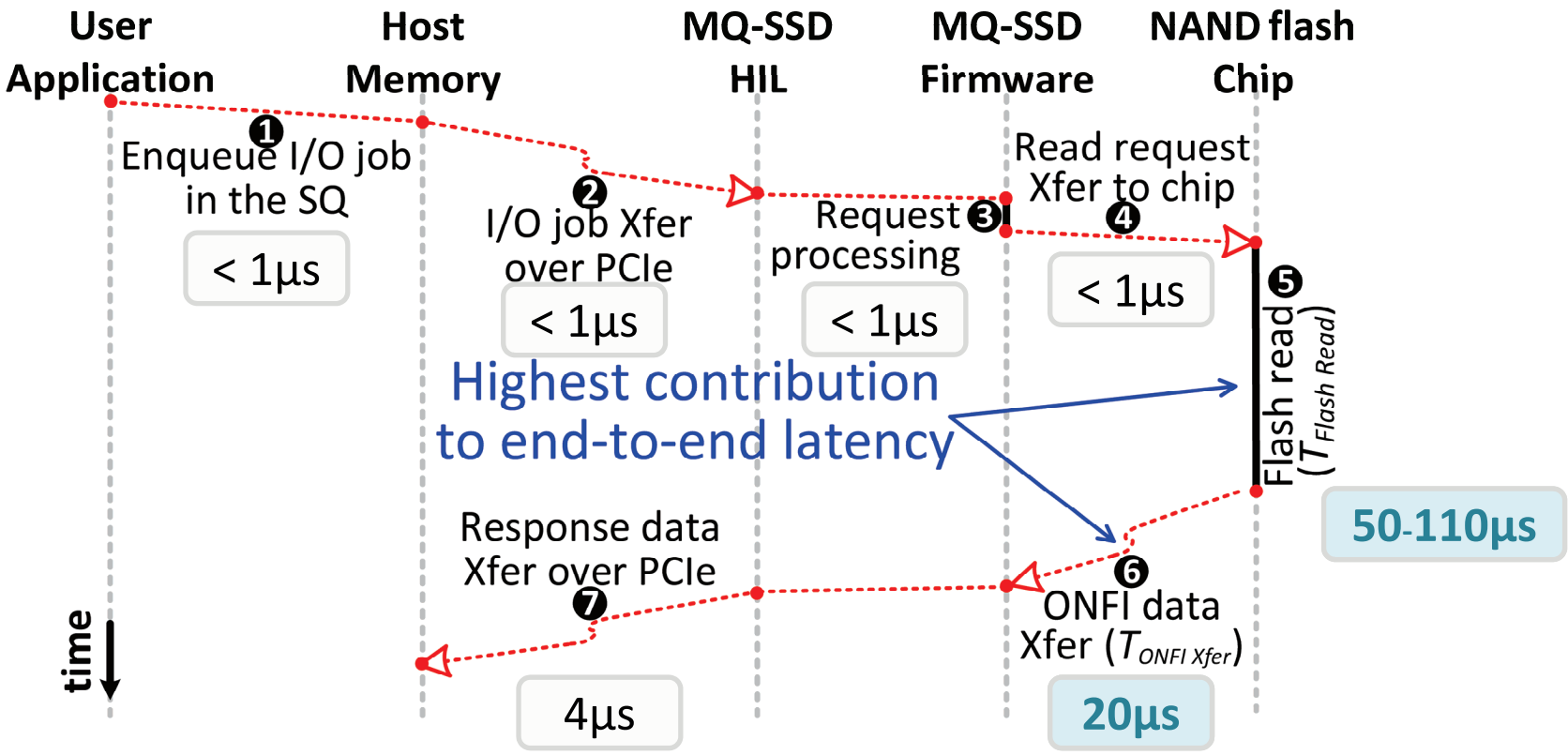
SAFARI



Challenge 3: Complete Model of Request Latency

SAFARI

- Request to NAND flash based SSD

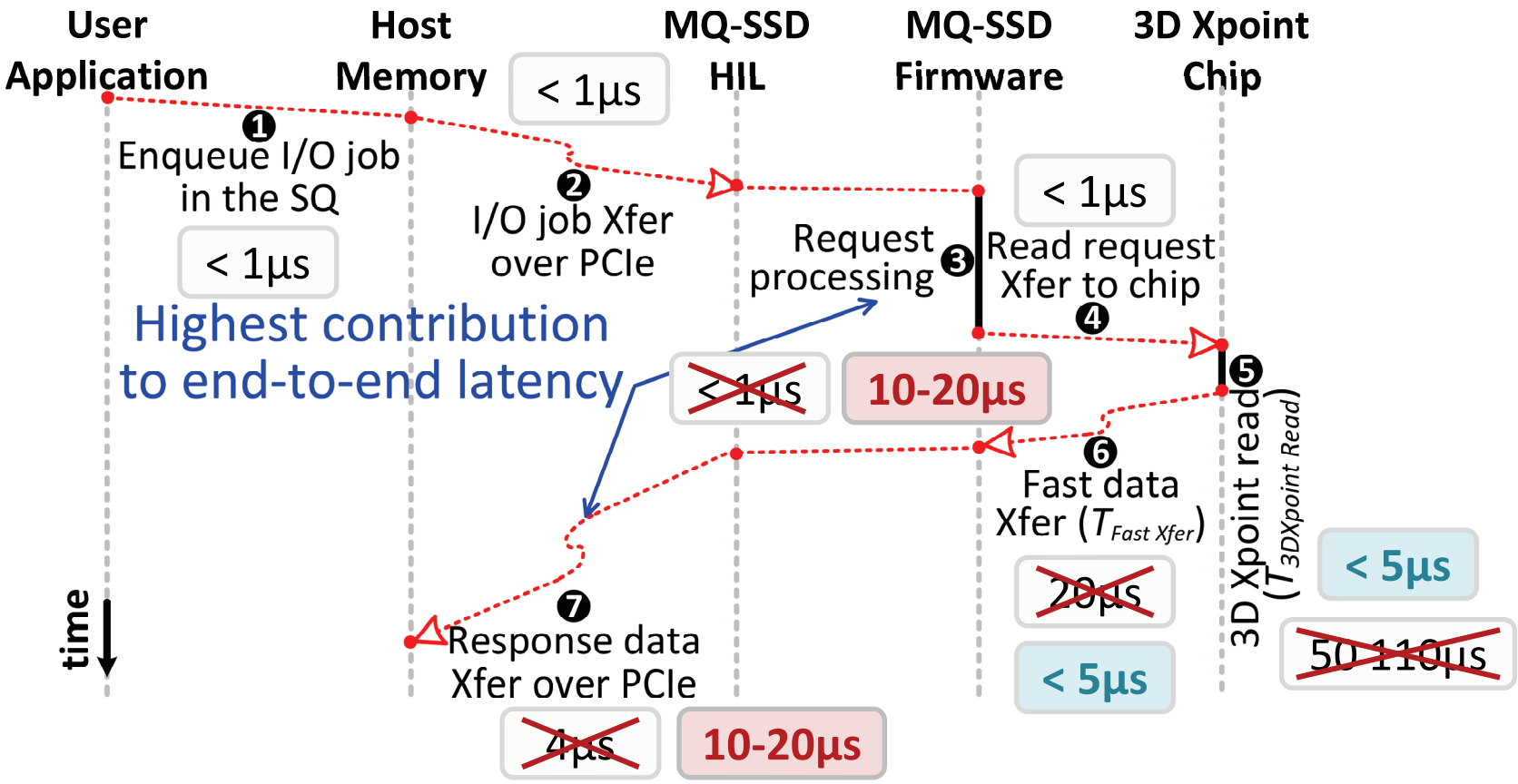


- Current simulators often model **only steps 5 and 6**
- What if we use a *different* non-volatile memory (NVM)?

Challenge 3: Complete Model of Request Latency

SAFARI

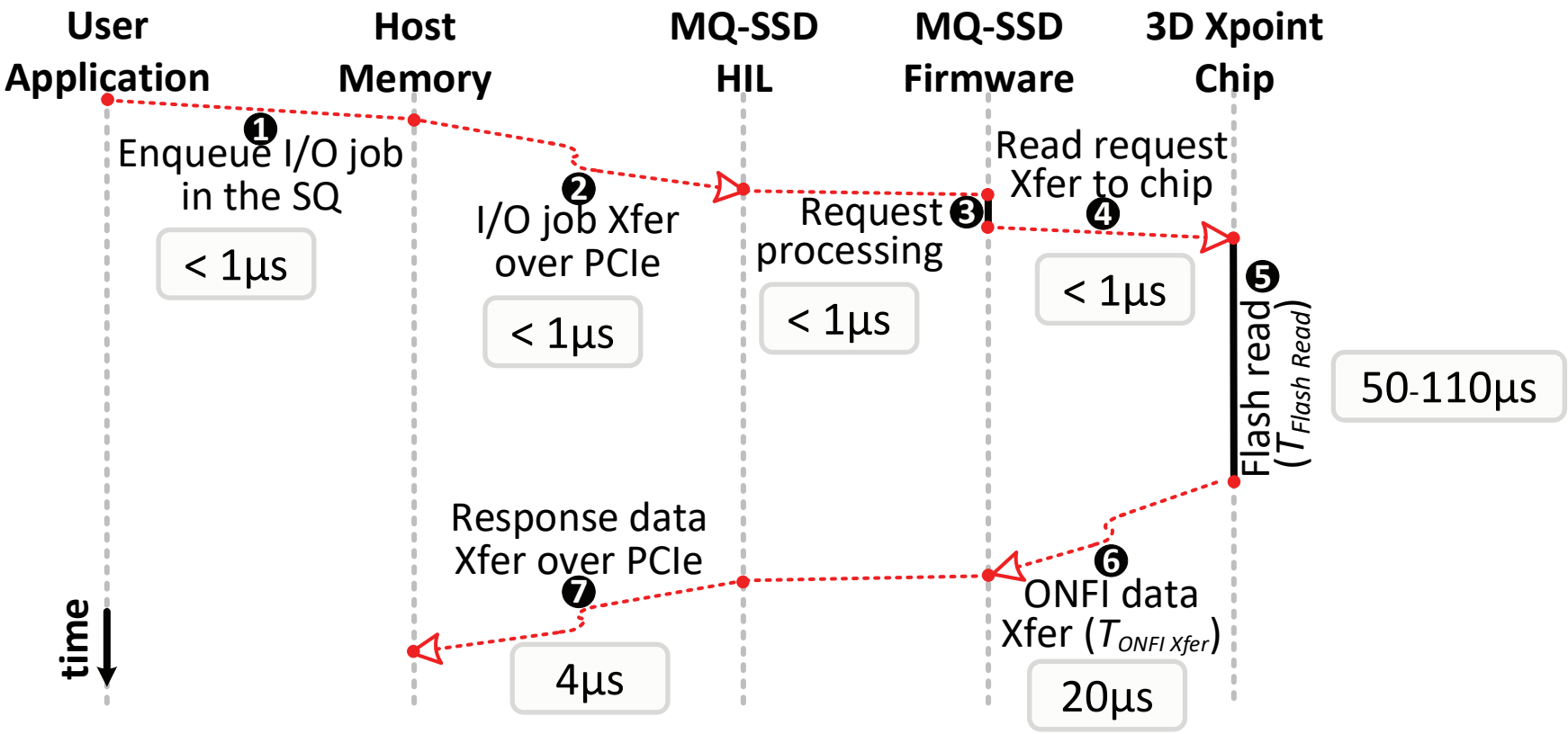
- Request to 3D XPoint based SSD



Challenge 3: Incomplete Models of Request Latency

SAFARI

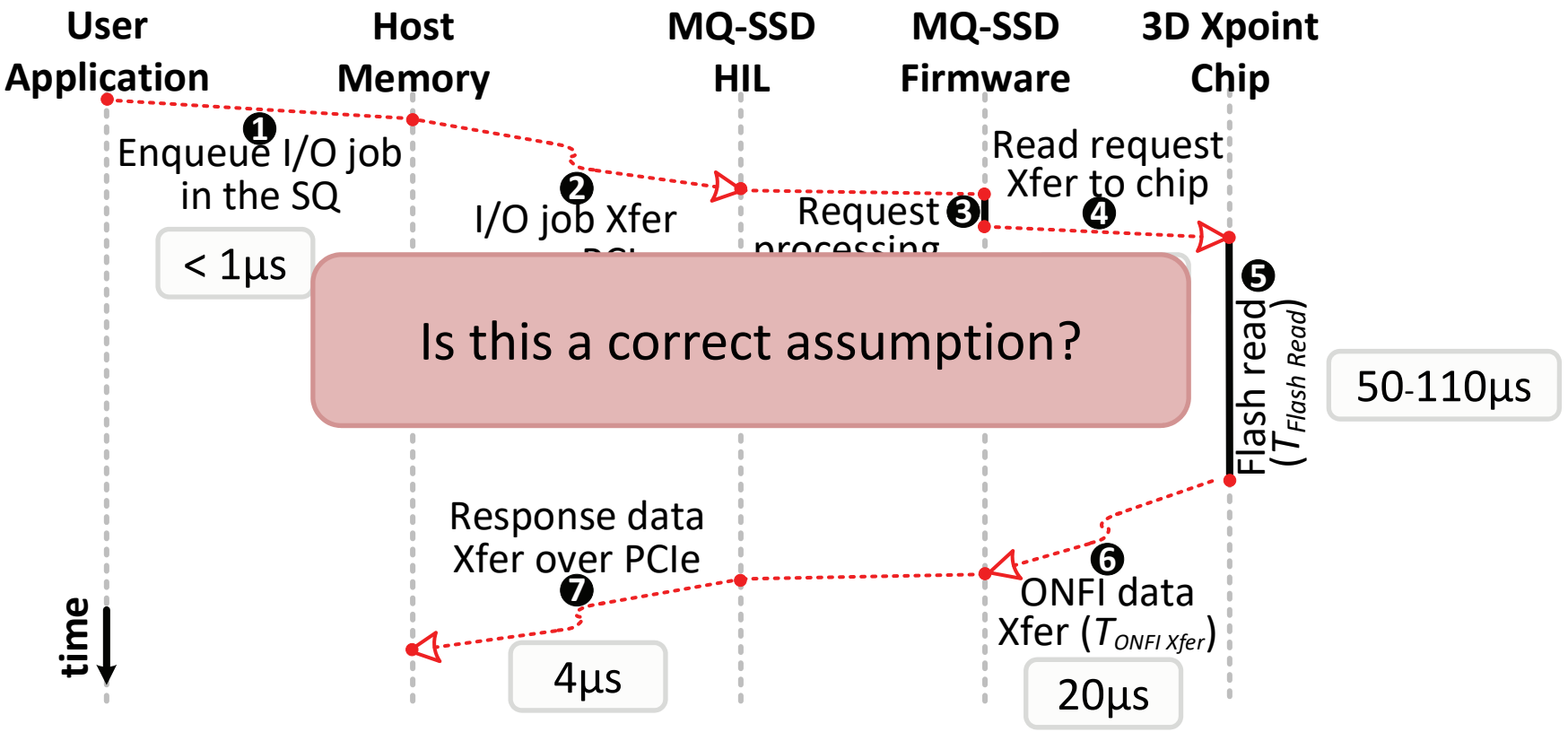
- 3. Real end-to-end request processing latency



Challenge 3: Incomplete Models of Request Latency

SAFARI

- 3. Real end-to-end request processing latency

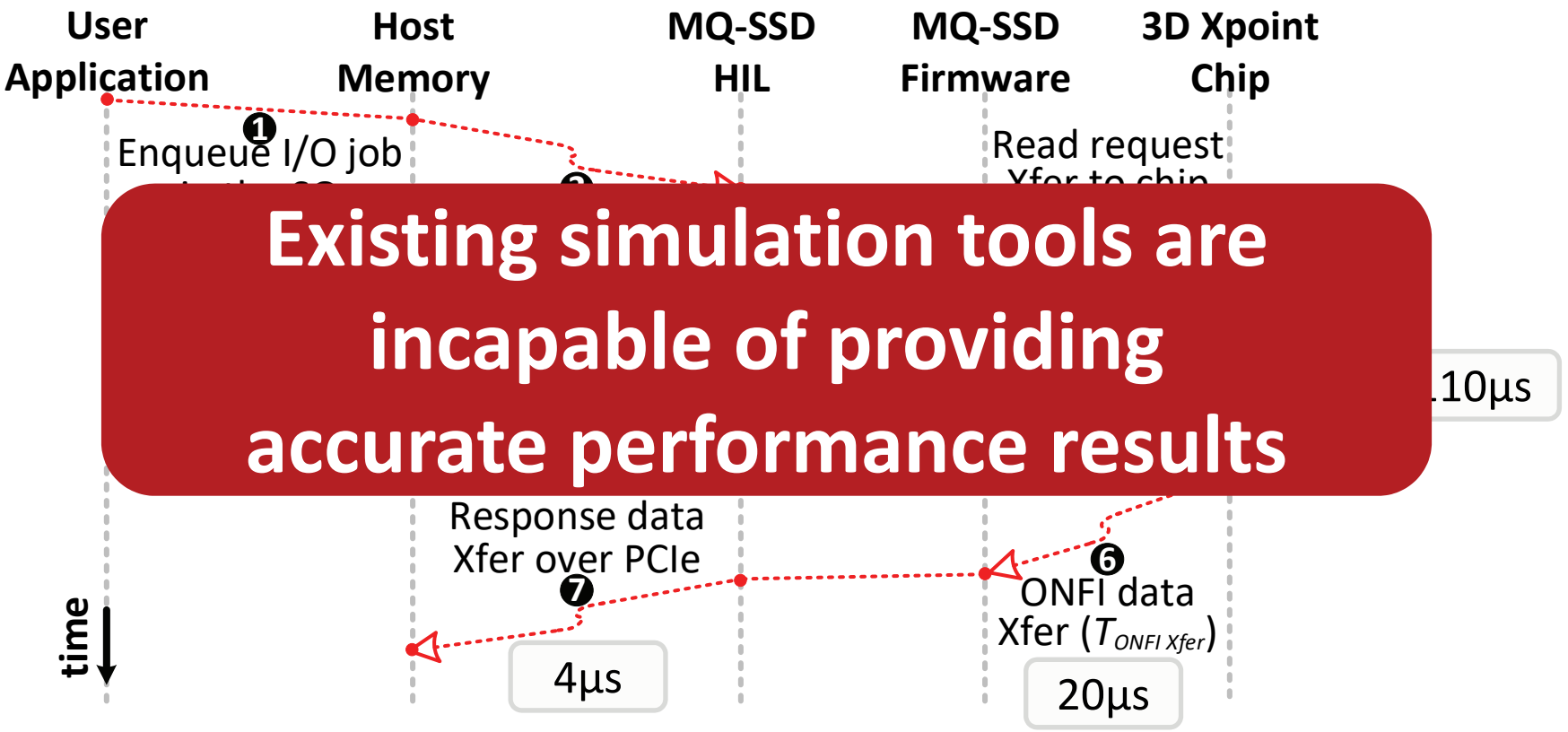


» The current simulators mainly model the latency of ❺ and ❻, assuming that they have the highest contribution in latency

Challenge 3: Incomplete Models of Request Latency

SAFARI

- 3. Real end-to-end request processing latency



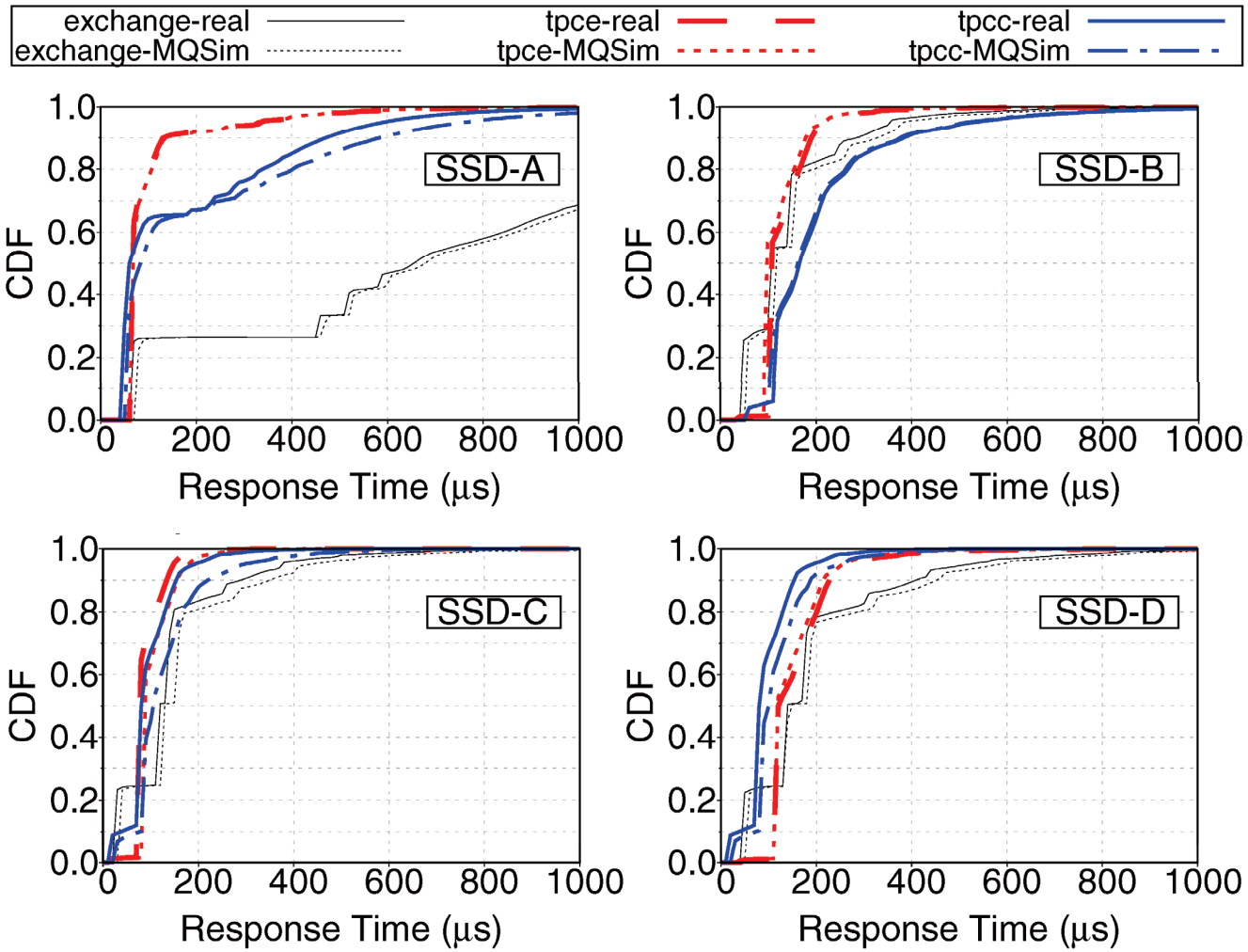
» The current simulators mainly model the latency of ❺ and ❻, assuming that they have the highest contribution in latency

Simulation Challenges for Modern MQ-SSDs

- The research community needs simulation tools that reliably model these features

- State-of-the-art SSD simulators fail to model a number of key properties of modern SSDs
  - They implement only the single queue approach
  - They do not adequately replicate steady-state behavior → the reported performance values may be far from the long-term results
  - They capture only the part of the request latency → do not capture the true request performance of SSDs

Real Trace Execution Results



Modeling a Modern MQ-SSD with MQSim

- MQSim provides a detailed end-to-end latency model
  - » A model for all of the request processing steps in modern SSDs
  - » All of the SSD components are modeled and contributed to the end-to-end latency

Comparison with Existing Simulators

SAFARI

Simulator	HIL Protocol		Execution Mode				End-to-End Latency					Front-End Components								Simulation Error (%)			
	NVMe MQ	SATA NCQ	Alone[1]	Full[2]	Emul[3]	Prec[4]	NVM R/W[5]	NVM Xfer	FTL Proc[6]	Cache Acc.[7]	Host Xfer[8]	Map P[9]	Map H[10]	GC	Write Cache	TSU[11]	WRL[12]	MQ FTL[13]	LOC[14]	SSD-A	SSD-B	SSD-C	SSD-D
MQSim	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	13K	8	6	18	14
SSDModel [3]			✓				✓	✓			✓	✓		✓			✓		1K	91	155	196	136
FlashSim [38]			✓				✓	✓			✓	✓	✓	✓			✓		8K	99	259	310	138
SSDSim [27]			✓				✓	✓				✓		✓	✓				5K	70	68	74	85
NANDFlashSim [32]							✓	✓											7K	–	–	–	–
VSSIM [92]					✓		✓	✓				✓	✓	✓	✓		✓		6K	–	–	–	–
WiscSim [26]		✓	✓				✓	✓	✓			✓	✓	✓			✓		7K	95	277	324	135
SimpleSSD [35]			✓	✓			✓	✓					✓	✓		✓	✓		7K	–	–	–	–

[1] Standalone execution [2] Integrated execution with full-system simulator [3] SSD emulation for real system
[4] Fast and accurate preconditioning of the modeled SSD to enable accurate steady-state results
[5] Flash (NVM) read/write timing [6] FTL request processing overhead [7] Accurate modeling of write cache access latency
[8] Host-to-device and device-to-host data transfer delay [9] Page-level address mapping [10] Hybrid address mapping
[11] FTL transaction scheduling unit [12] FTL wear-leveling unit [13] Built-in support for multi-queue-aware request processing in FTL
[14] Lines of source code

Fairness Study Methodology

SAFARI

SSD Organization	Host interface: PCIe 3.0 (NVMe 1.2) User capacity: 480 GB Write cache: 256 MB, CMT: 4 MB 8 channels, 4 chips per channel `QueueFetchSize` = 512
Flash Communication Interface	ONFI 3.1 (NV-DDR2) Width: 8 bit, Rate: 333 MT/s
Flash Microarchitecture	8 KiB page, 448 B metadata per page, 256 pages per block, 2048 blocks per plane, 2 planes per die
Flash Access Parameters	Read latency: 75 µs, Program latency: 750 µs, Erase latency: 3.8 ms

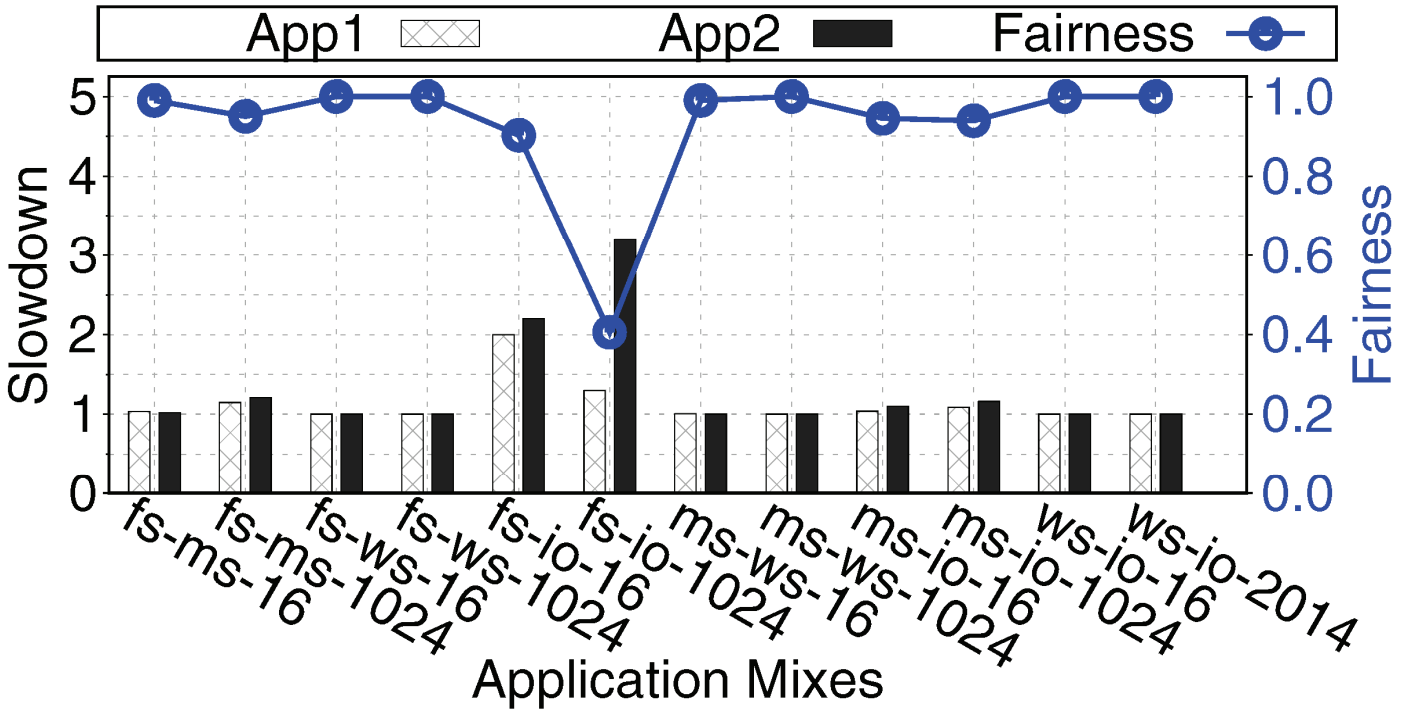
Questions

- **Do you have any plan to support near data processing?**
  - Yes, we have plans to boost MQSim with support for near data processing
  - In six months

- **Major companies are interested in the performance-cost ratio; How can MQSim help them?**
  - MQSim can be augmented with a cost model
  - Based on the input workload characteristics, MQSim can determine those designs with best performance cost ratio

Measuring Impact of Interference on Applications

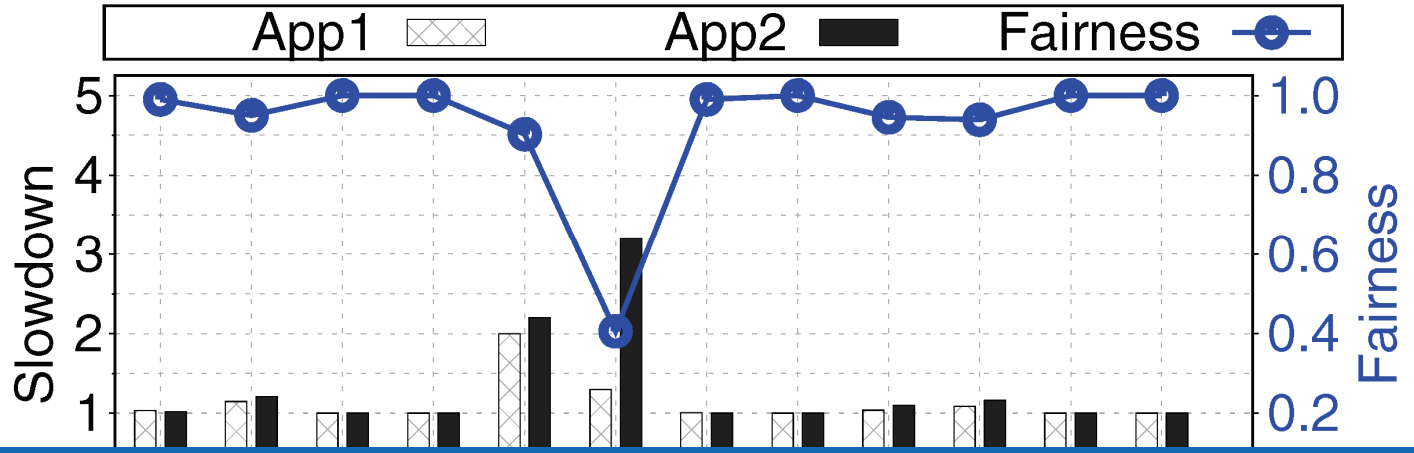
- Execute two-application workload bundles
  - Applications: file-server (fs), mail-server (ms), web-server (ws), IOzone large file access (io)
  - Vary queue fetch size (16 vs. 1024) to model fairness control vs. no control
  - Full methodology in the paper
- Compare application slowdown, fairness across entire system



Measuring Impact of Interference on Applications

SAFARI

- Execute two-application workload bundles
  - Applications: file-server (fs), mail-server (ms), web-server (ws), IOzone large file access (io)
  - Vary queue fetch size (16 vs. 1024) to model **fairness control vs. no control**
  - Full methodology in the paper
- Compare application slowdown, fairness across entire system



Application-level studies with MQSim capture **inter-application interference across the entire stack**

Conclusion & Future Work

- MQSim: A new simulator that accurately captures the behavior of both modern multi-queue and conventional SATA-based SSDs
- MQSim faithfully models a number of critical features absent in existing SSD simulators
- MQSim enables a wide range of studies
- Future work
  - Explore the design space of fairness and QoS techniques in MQ-SSDs
  - Design new management mechanism for modern SSDs based on emerging memory technologies