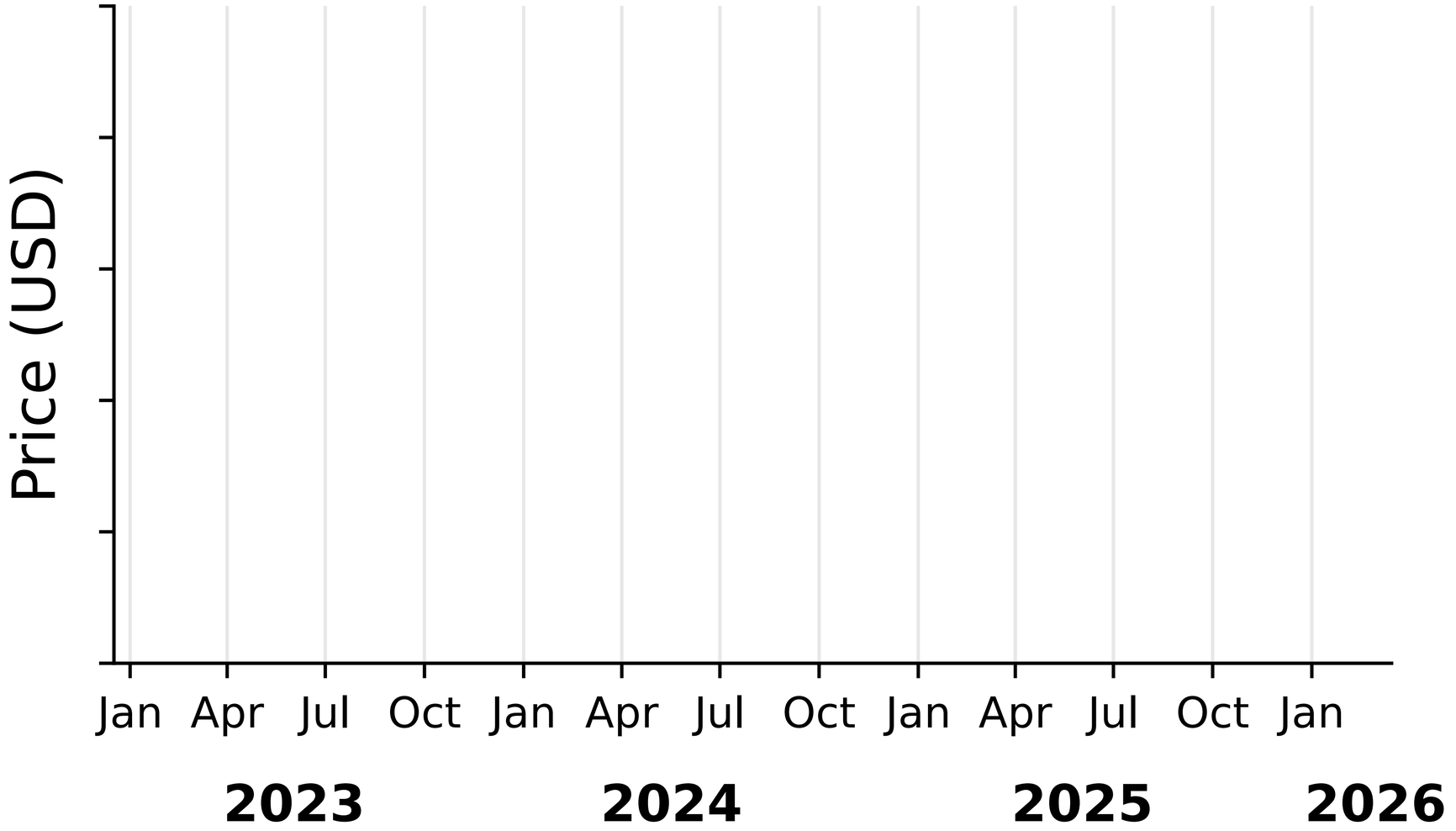
Unleashing The Potential of Datacenter SSDs by Taming Performance Variability

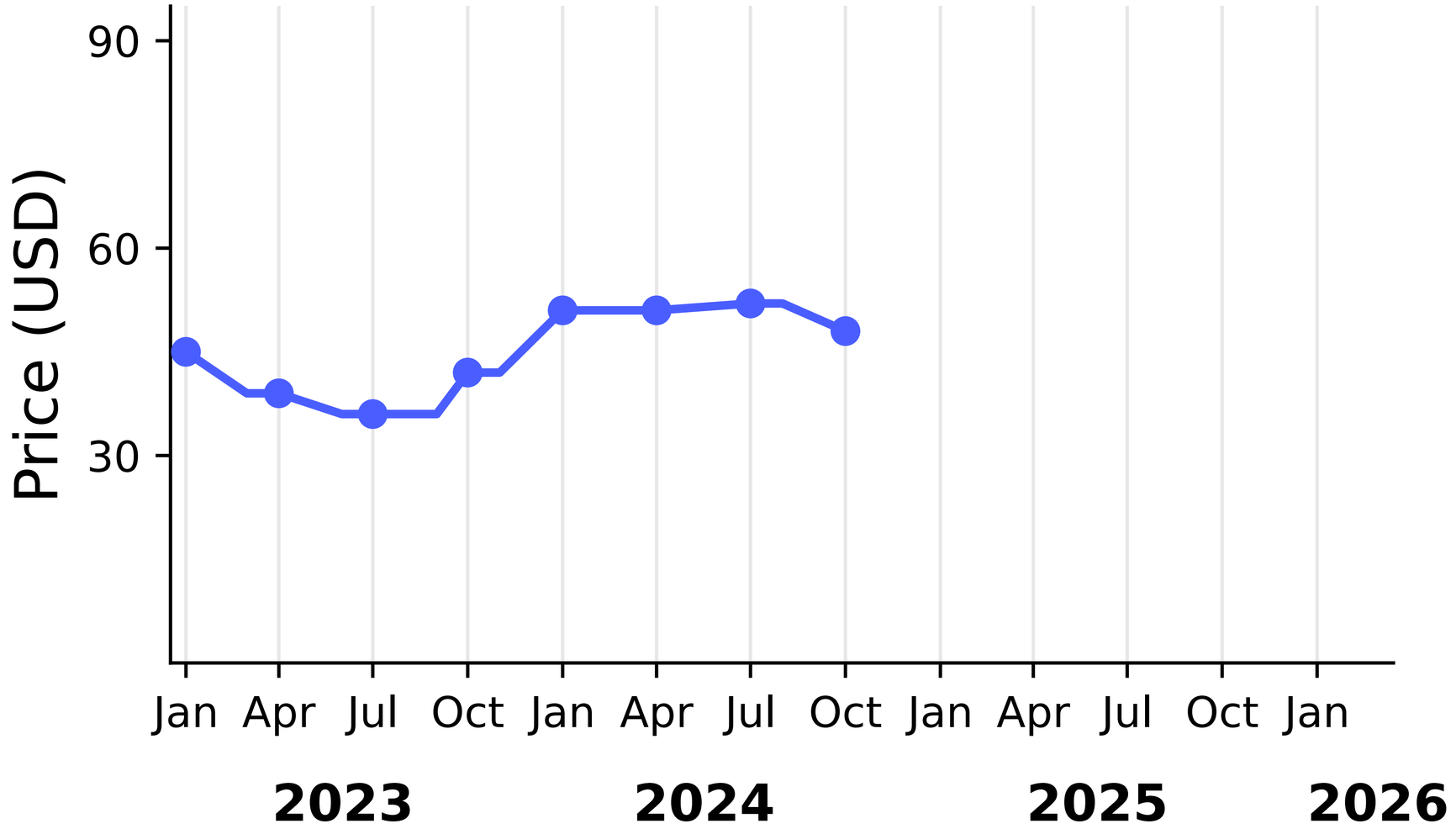
Gohar Irfan Chaudhry, Ankit Bhardwaj, Zhenyuan Ruan, Adam Belay

1TB datacenter-grade SSD (PCIe NVMe)



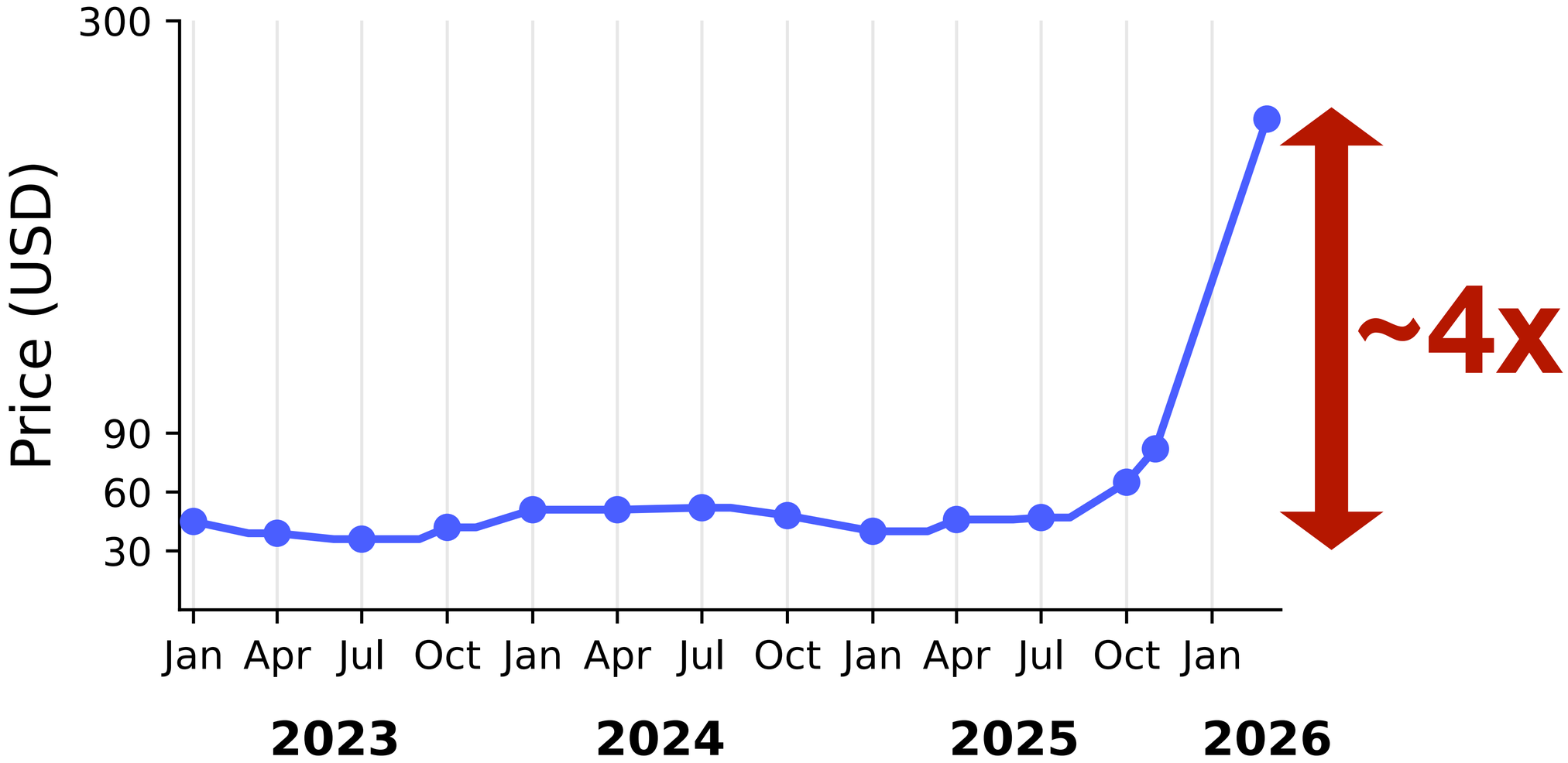
Source: Tom's Hardware *Source: Wilk Elektronik SA*

1TB datacenter-grade SSD (PCIe NVMe)



Source: Tom's Hardware Source: Wilk Elektronik SA

1TB datacenter-grade SSD (PCIe NVMe)

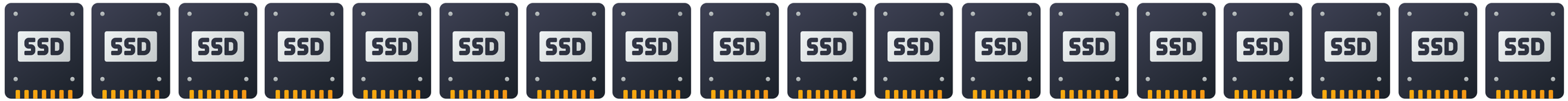


Source: Tom's Hardware Source: Wilk Elektronik SA

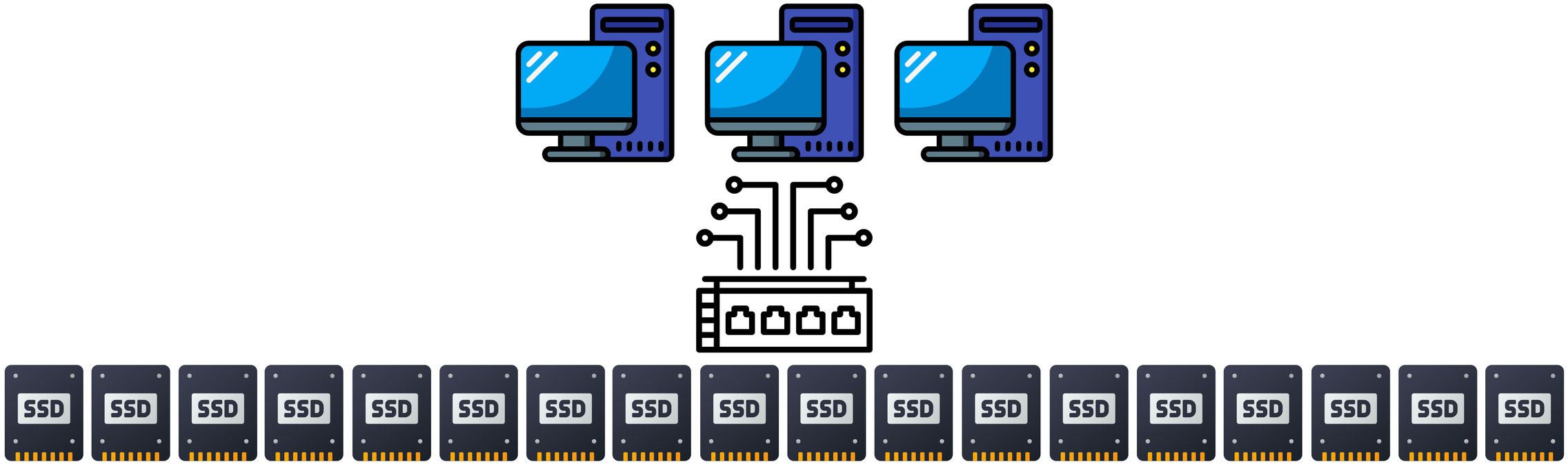
Background: Flash storage in the datacenter

Background: Flash storage in the datacenter

Background: Flash storage in the datacenter



Background: Flash storage in the datacenter



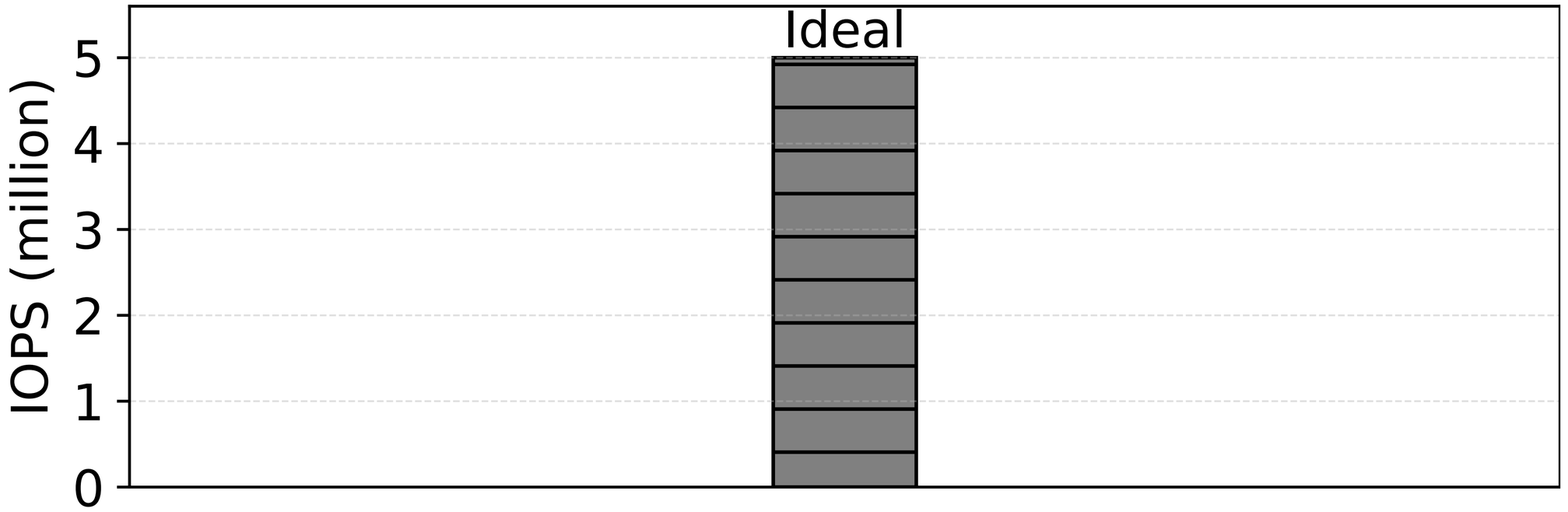
But, are we getting all the performance we paid for?

Benchmarking our SSD testbed

H/W = 12 SSDs**; Workload =** 90% reads**; Latency SLO =** P90 < 1ms

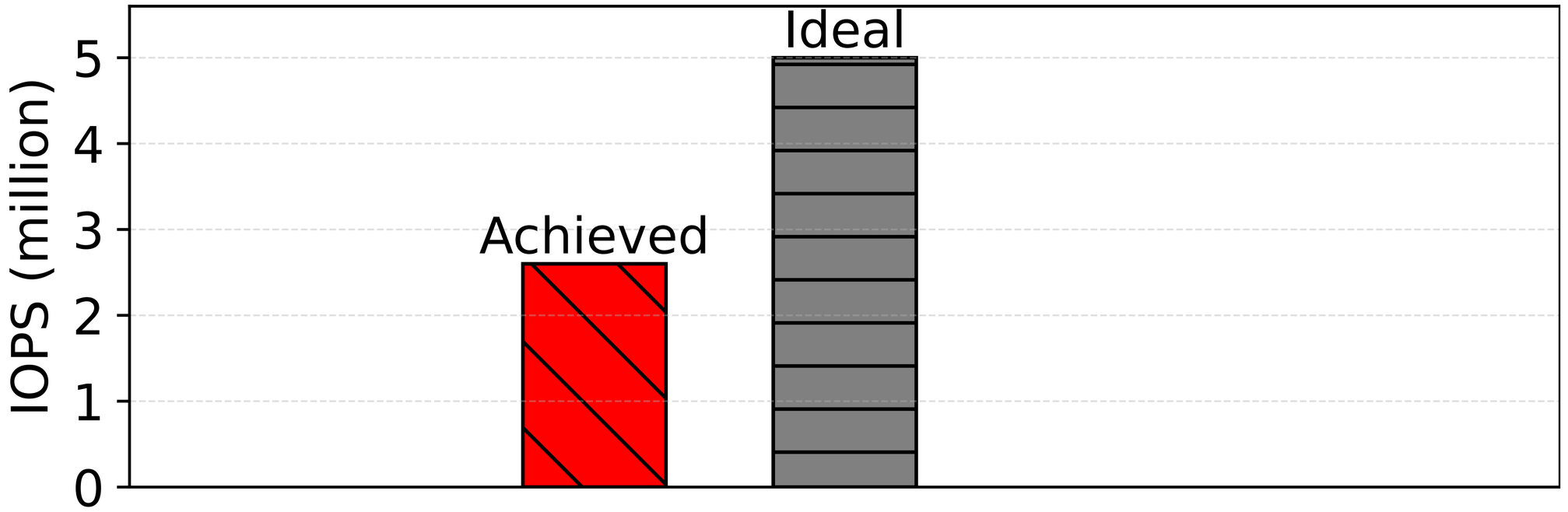
Benchmarking our SSD testbed

H/W = 12 SSDs**; Workload =** 90% reads**; Latency SLO =** P90 < 1ms



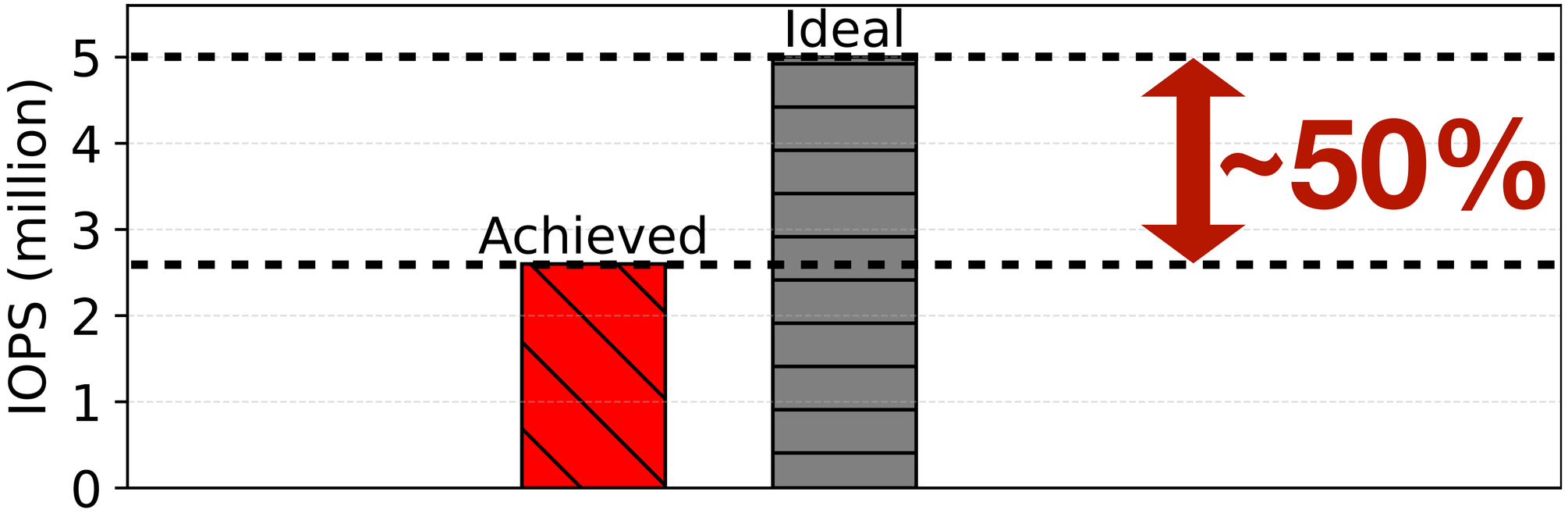
Benchmarking our SSD testbed

H/W = 12 SSDs**; Workload =** 90% reads**; Latency SLO =** P90 < 1ms



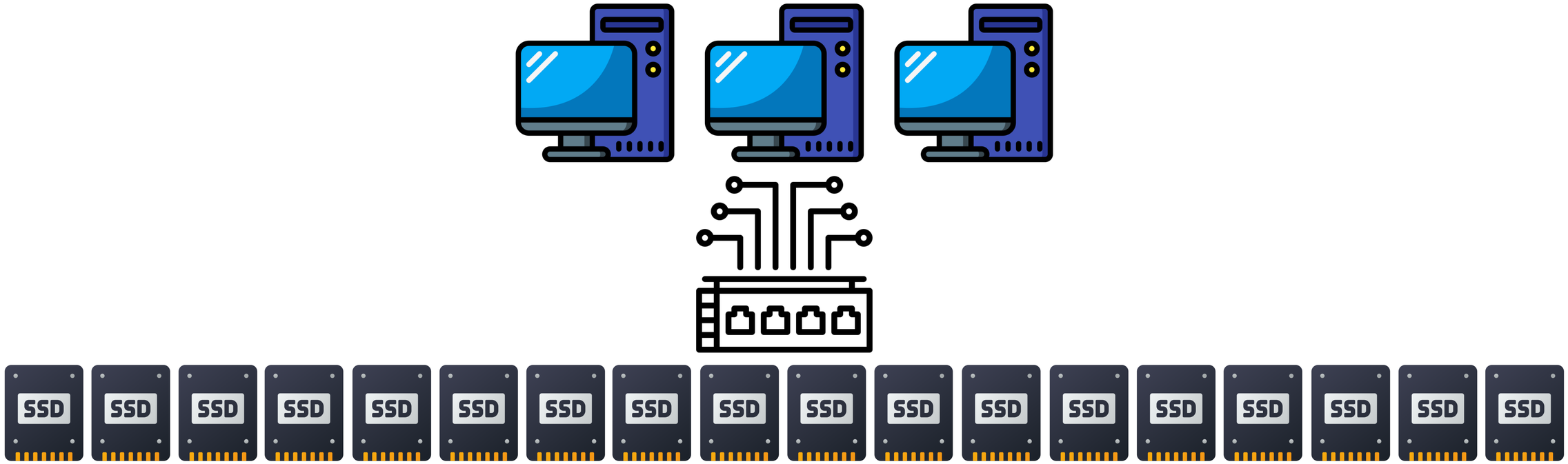
Benchmarking our SSD testbed

Significant performance (IOPS) left on the table…



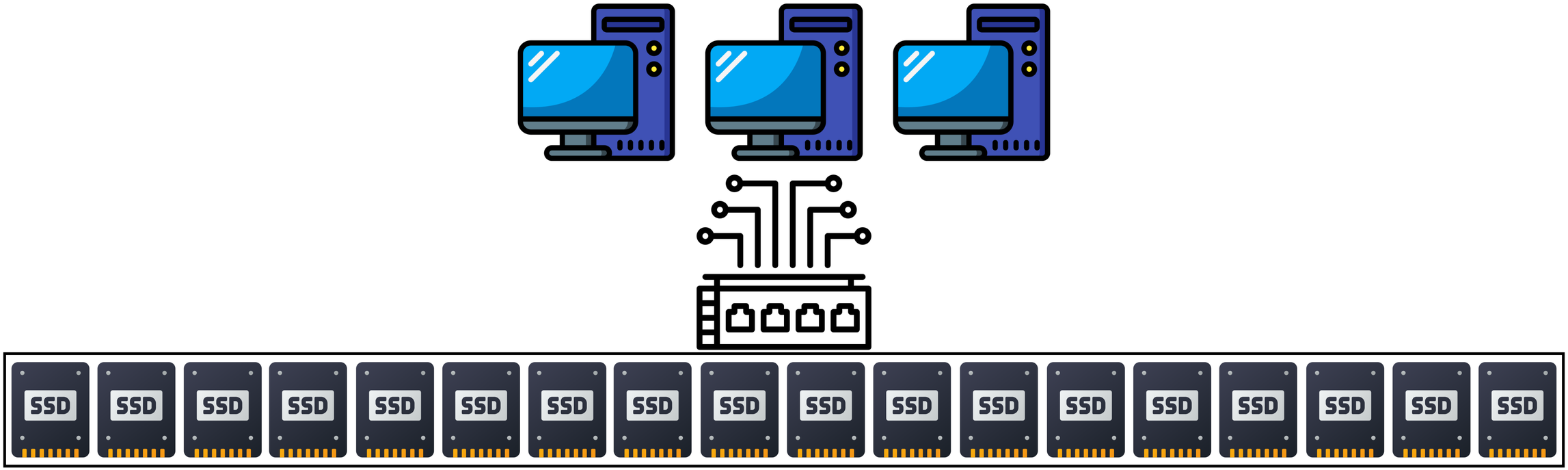
Where did all the performance go?

Flash storage in the datacenter



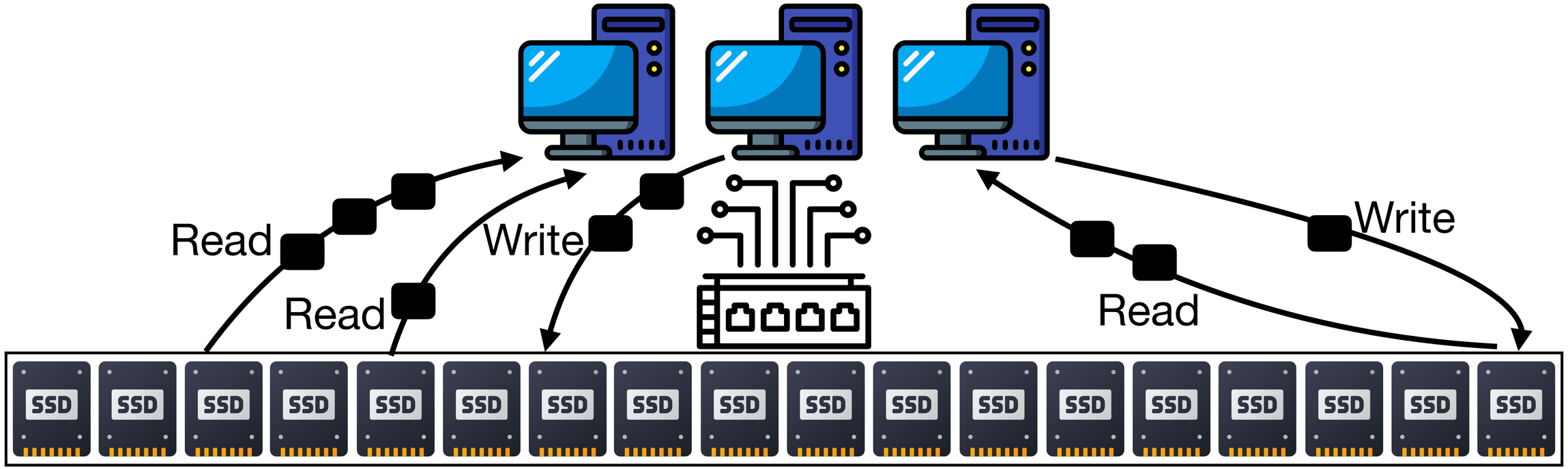
Flash storage in the datacenter

Existing systems give equal weight to all SSDs when load balancing…



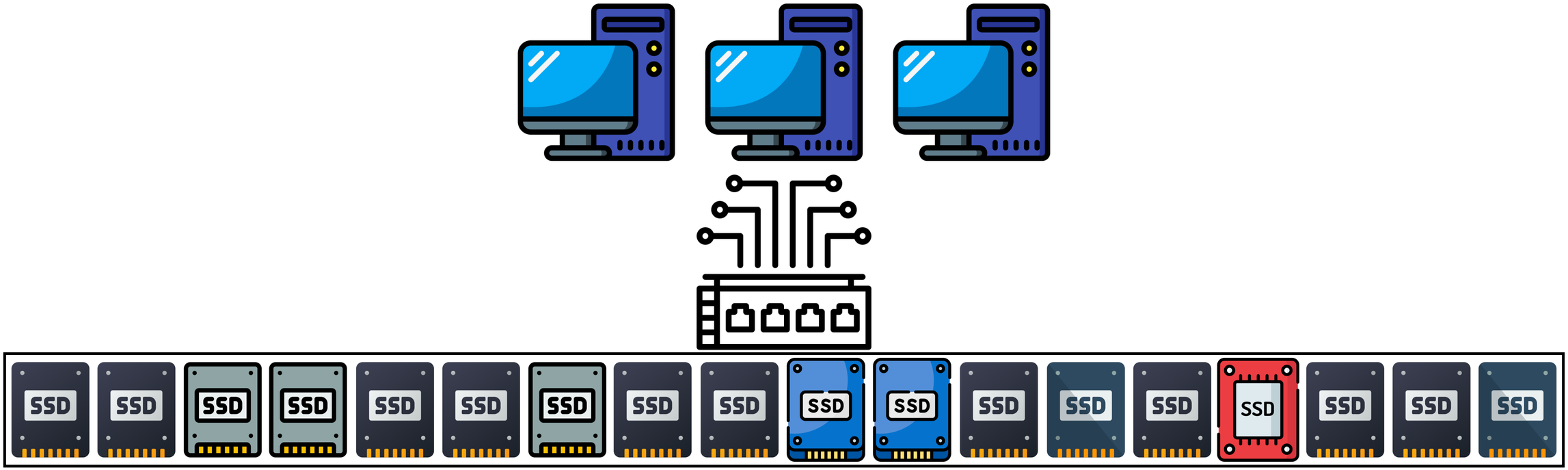
Flash storage in the datacenter

… and treat each I/O request homogeneously across the fleet!



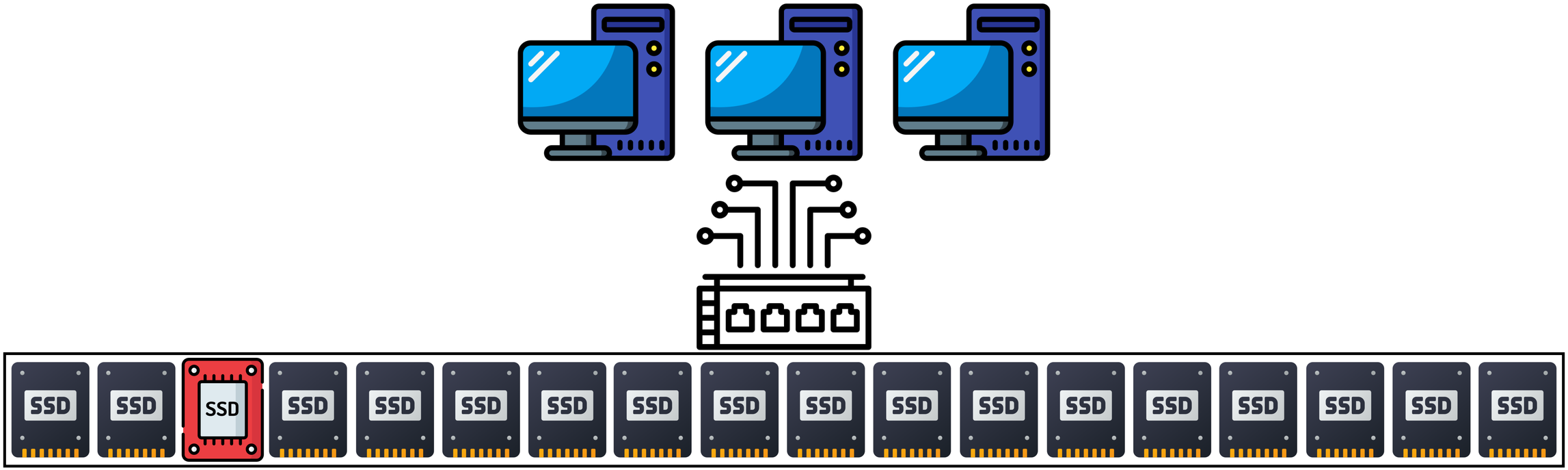
Flash storage in the datacenter

All SSDs are NOT the same…



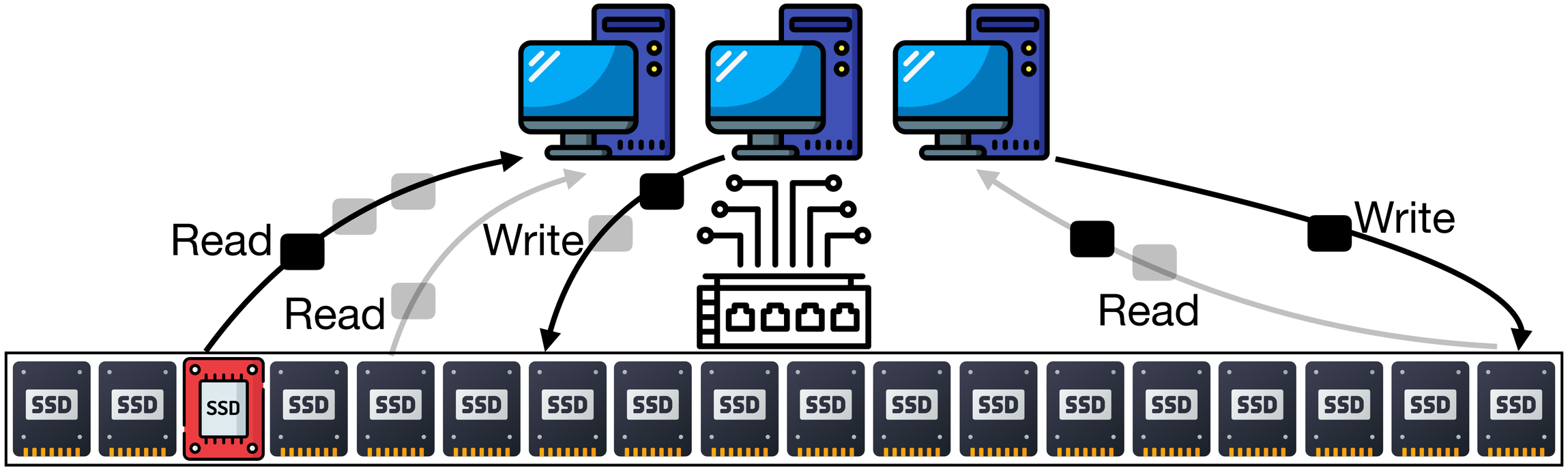
Flash storage in the datacenter

… even a single slow SSD can limit the system's aggregate performance!



Flash storage in the datacenter

… even a single slow SSD can limit the system's aggregate performance!



SSDs suffer from significant performance variability!

Performance Variability

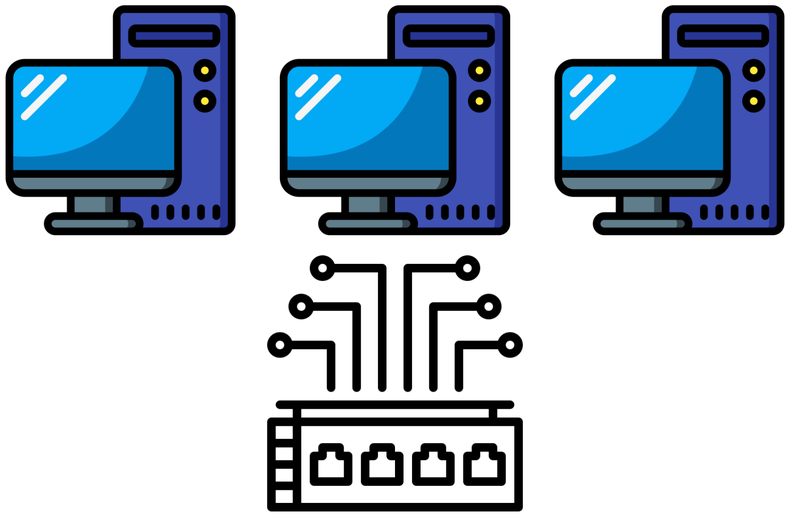
1. Device heterogeneity

2. Read/write interference

3. Garbage collection

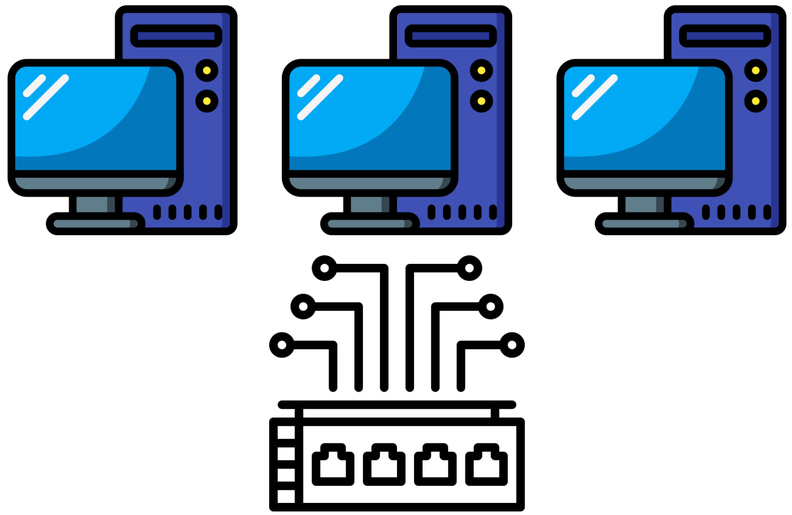
Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.

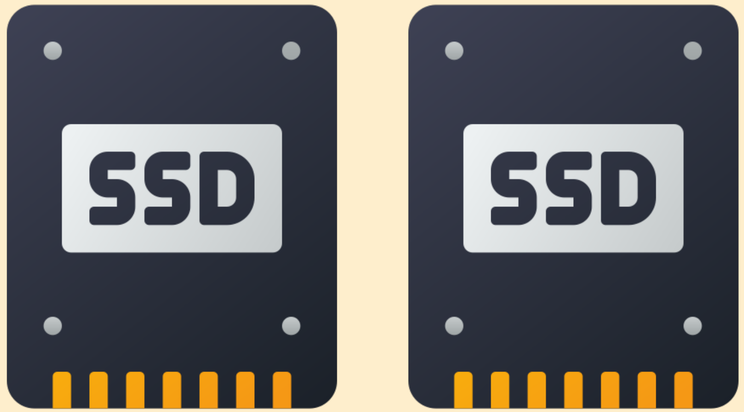


Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.

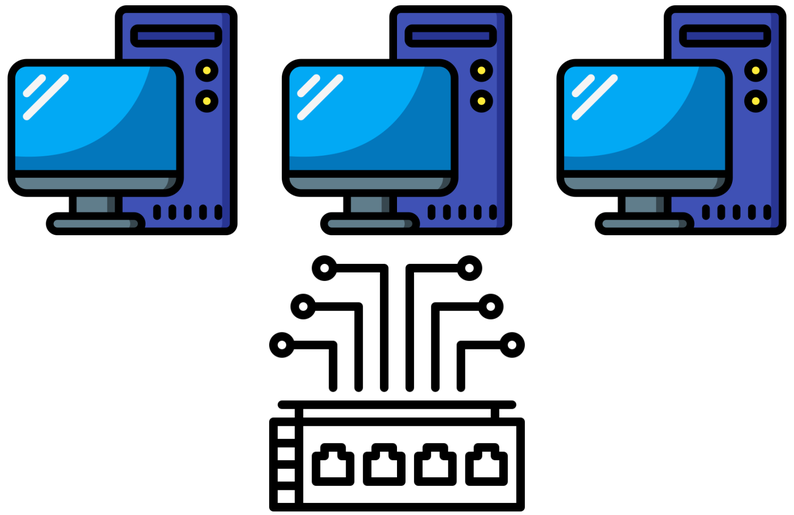


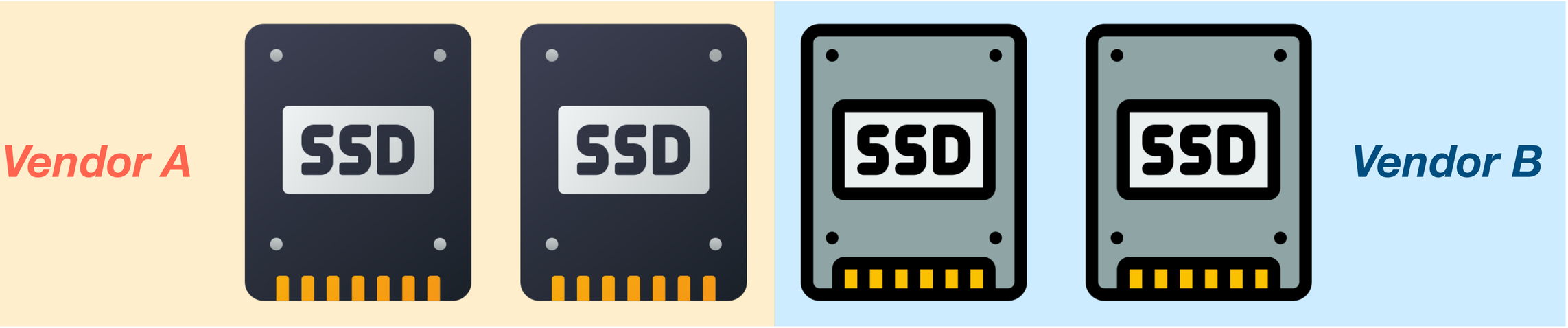
Vendor A



Variability #1: Device heterogeneity

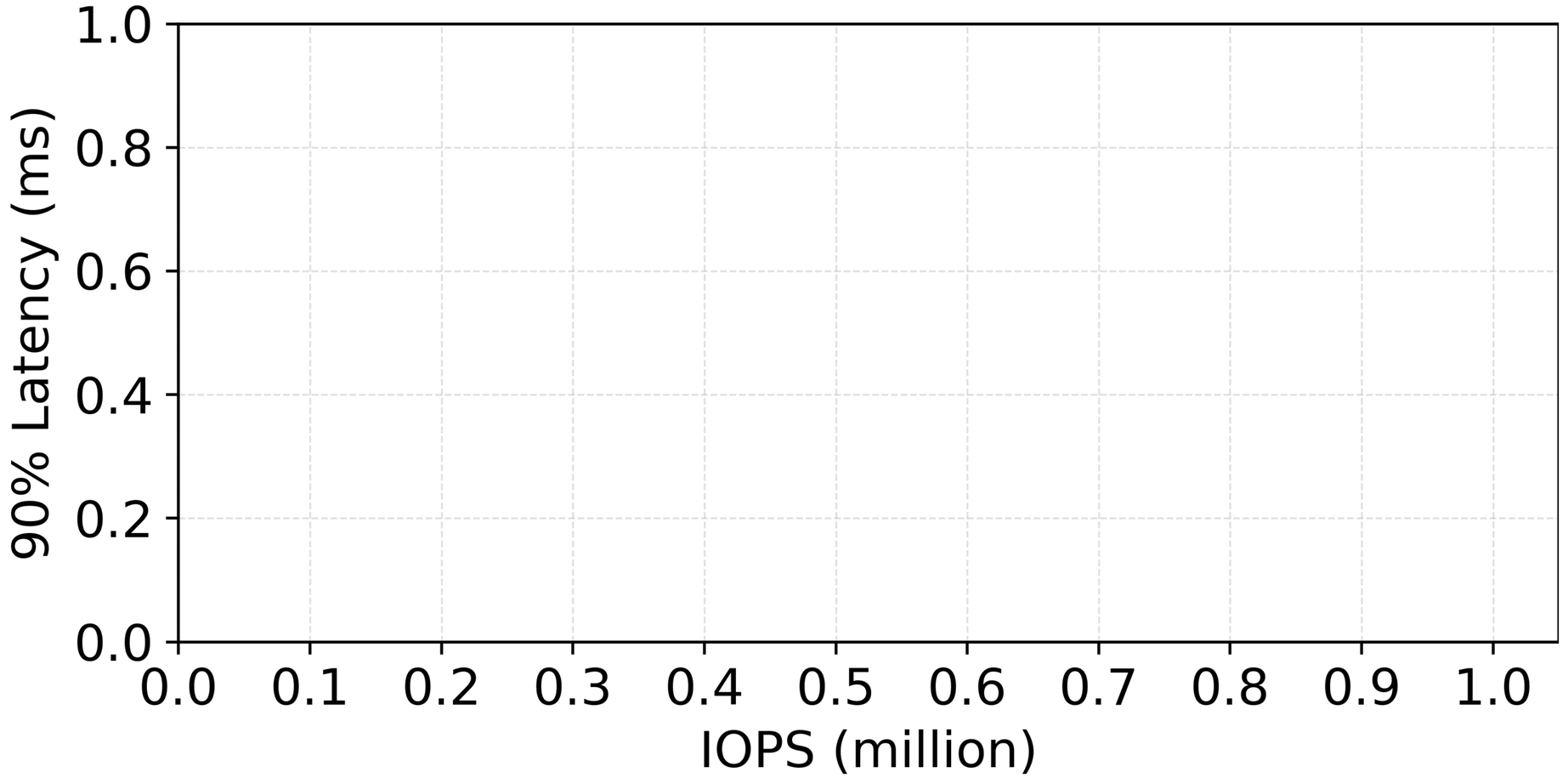
Different vendors, models, versions, wear/tear etc.





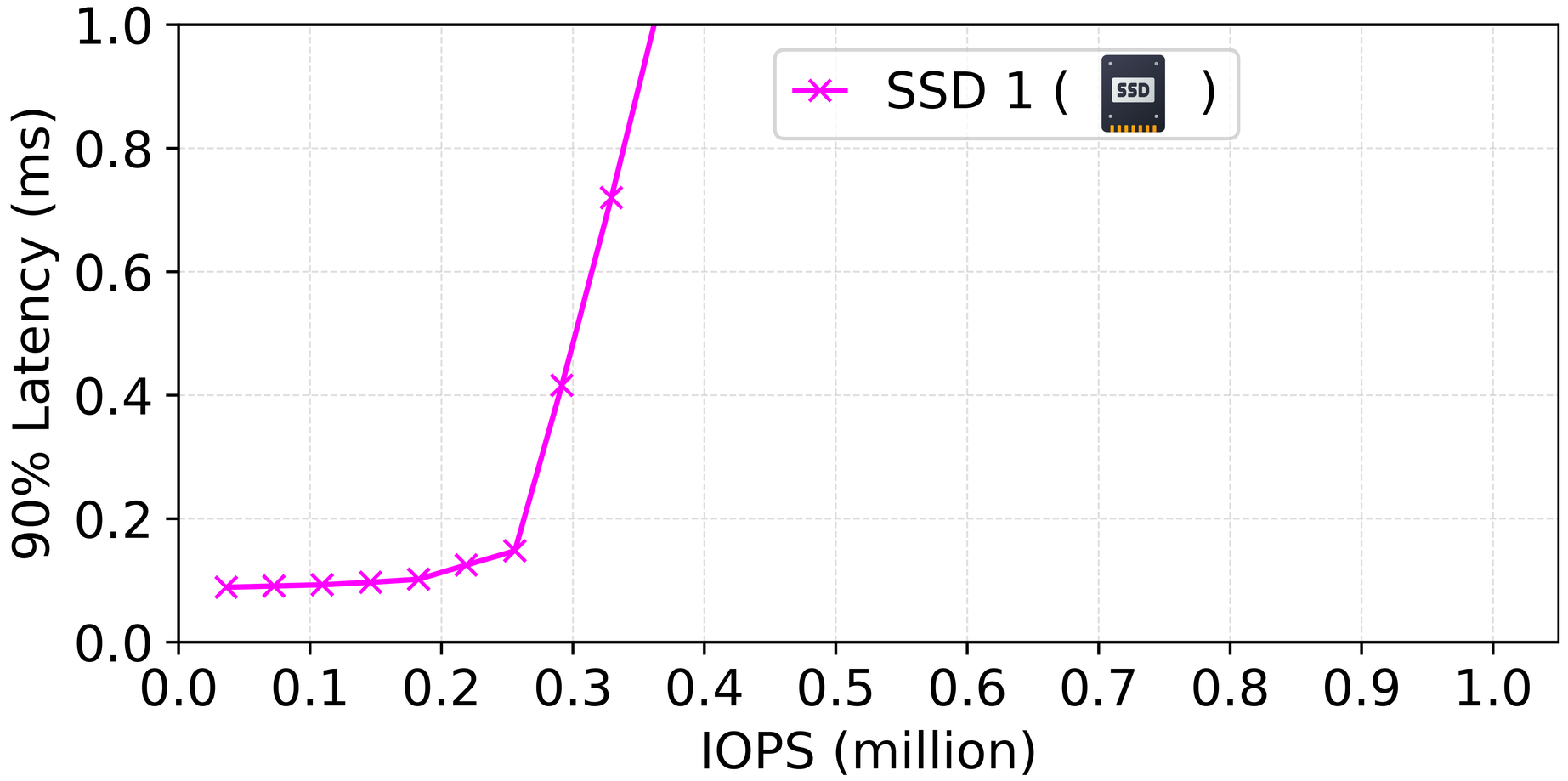
Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.



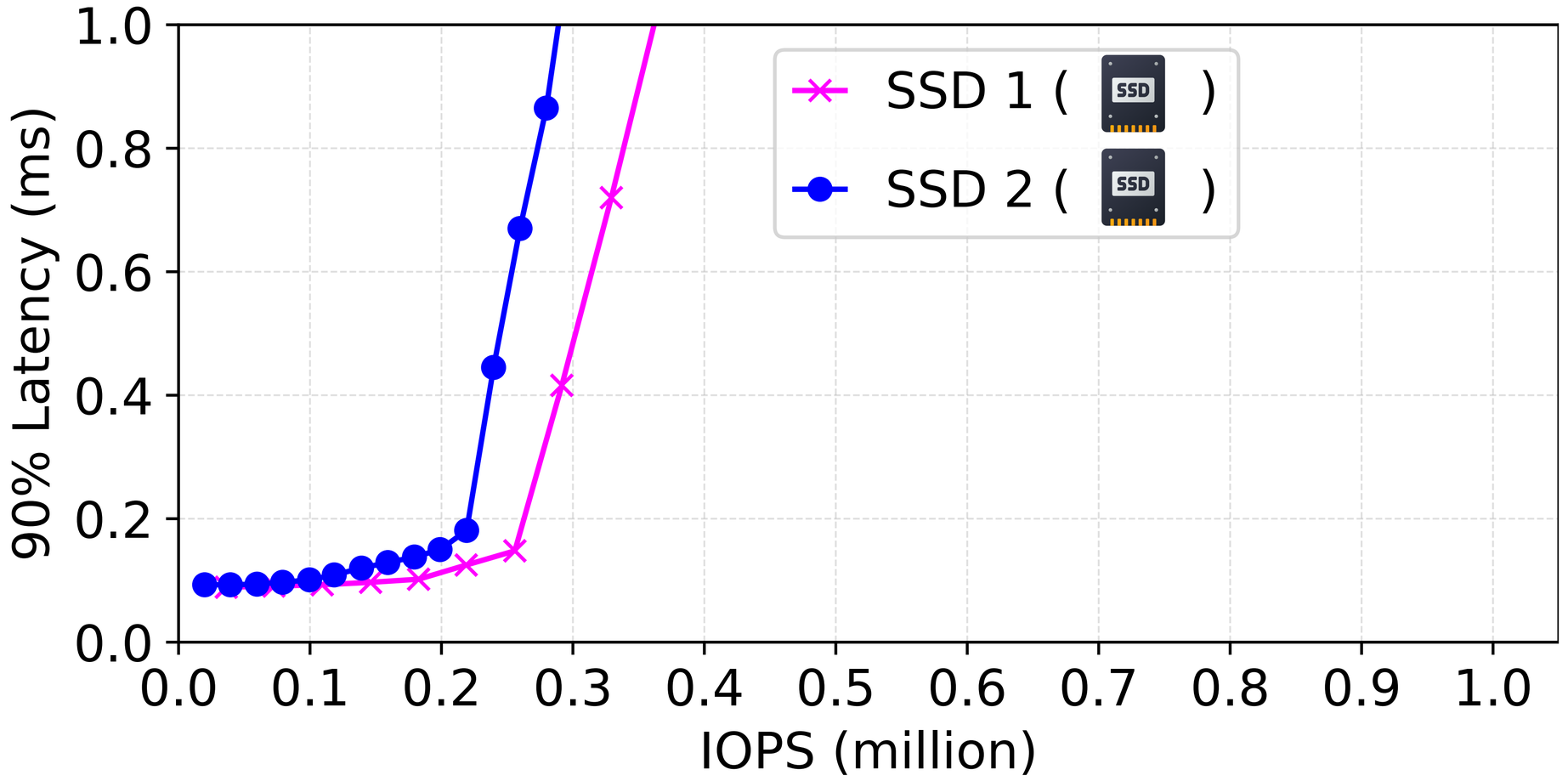
Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.



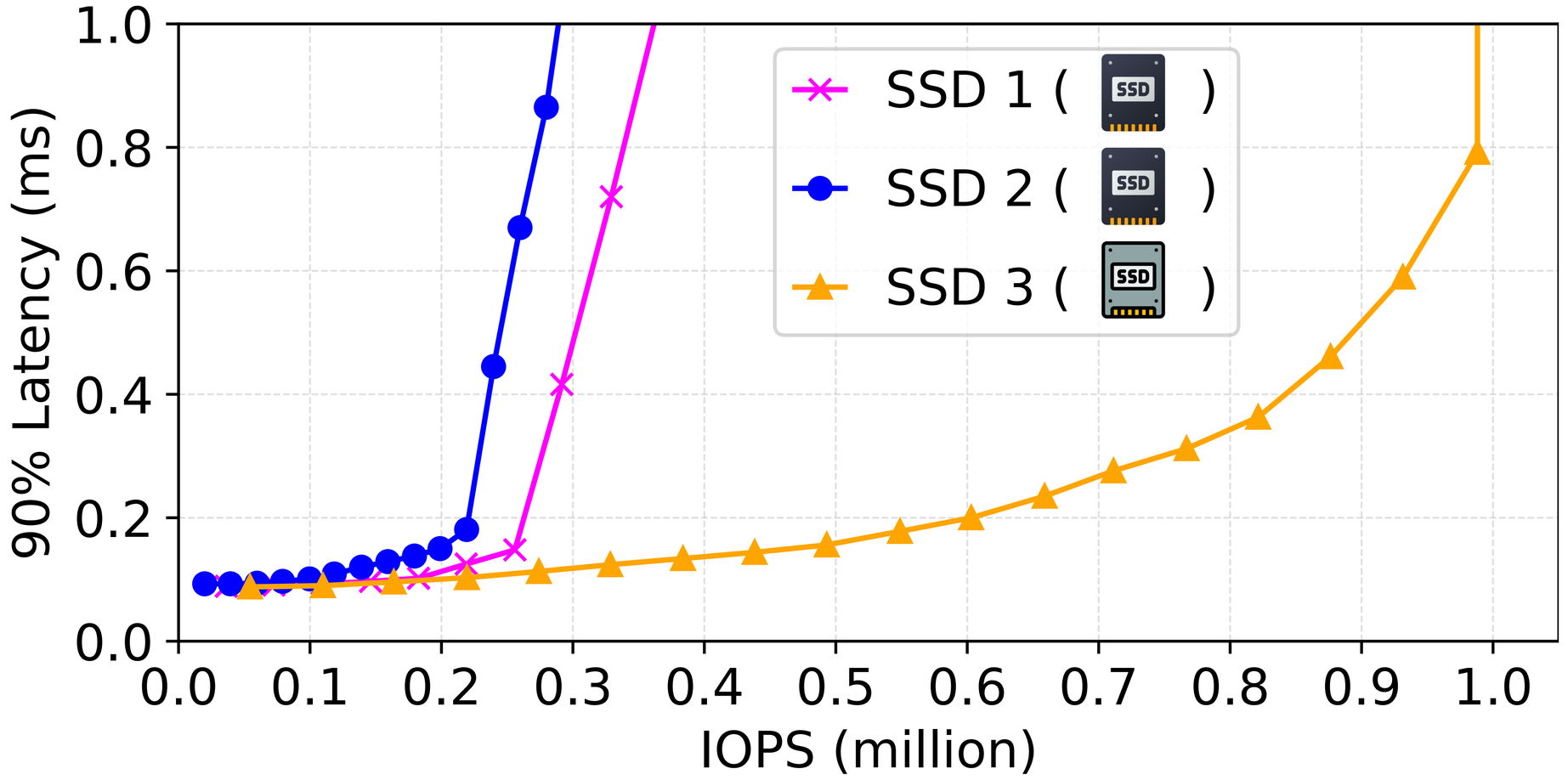
Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.



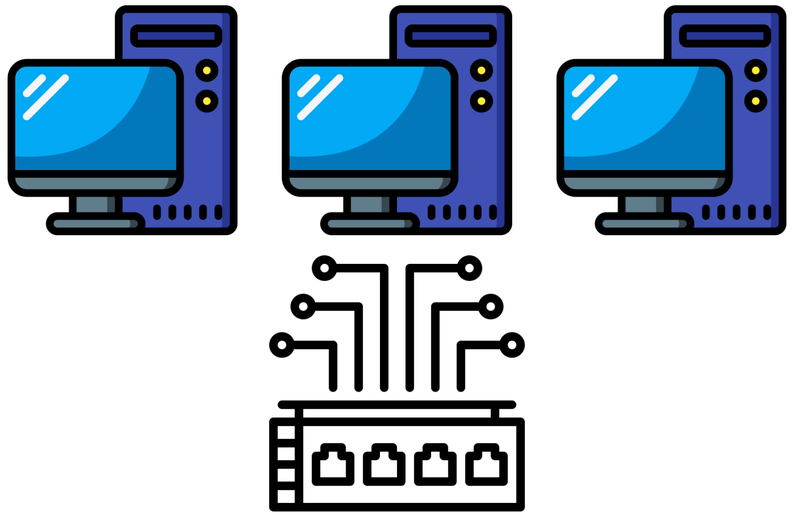
Variability #1: Device heterogeneity

Different vendors, models, versions, wear/tear etc.



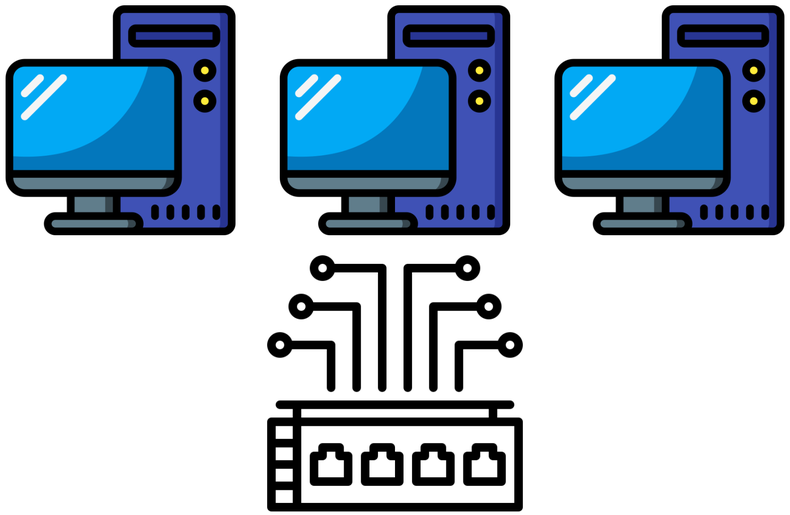
Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



Variability #2: Read/write interference

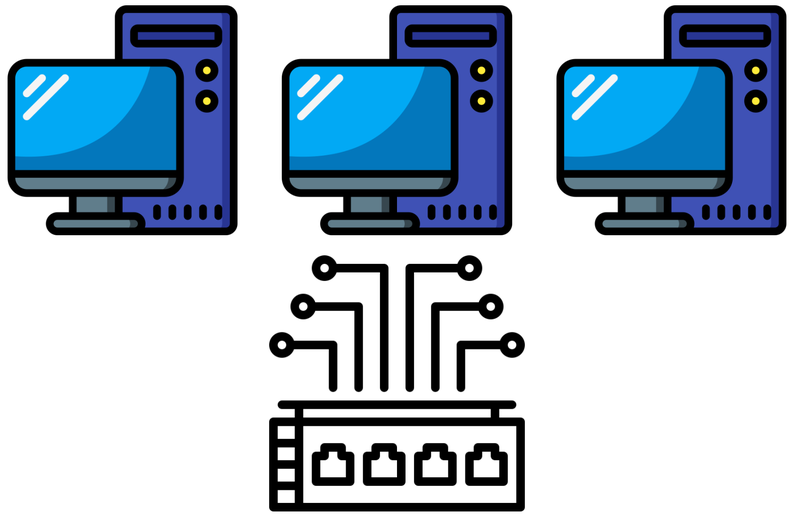
Different sensitivity to mixing reads and writes



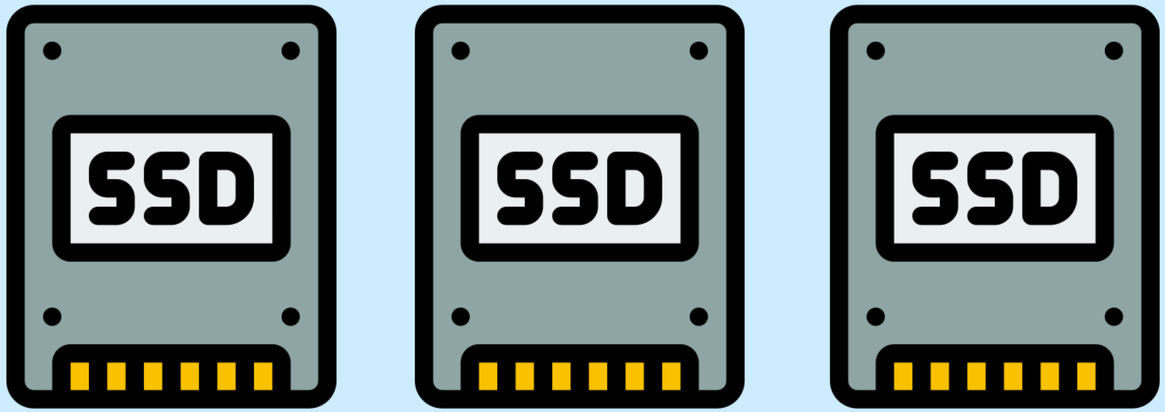
Write Optimized

Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



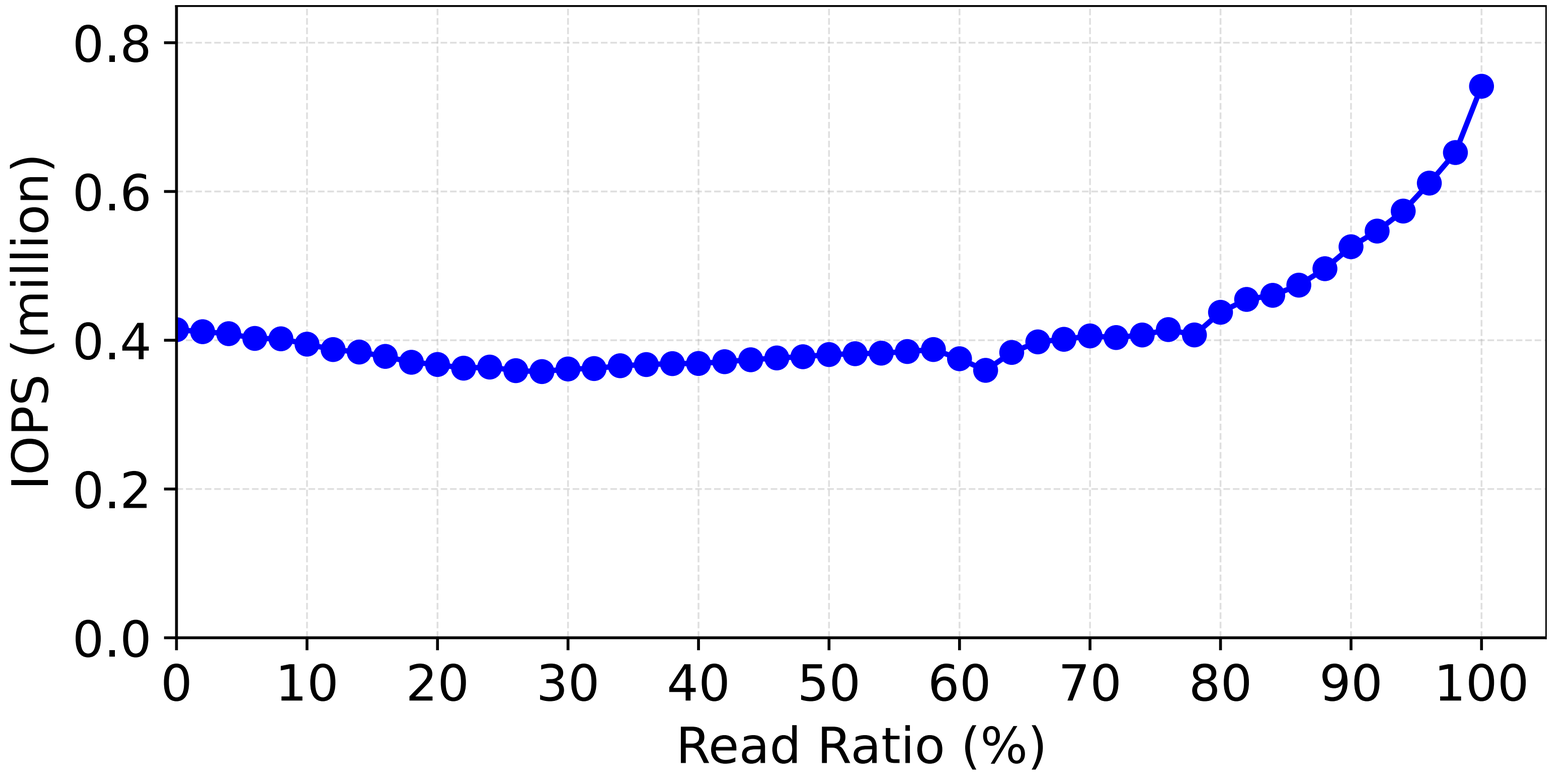
Write Optimized



Read Optimized

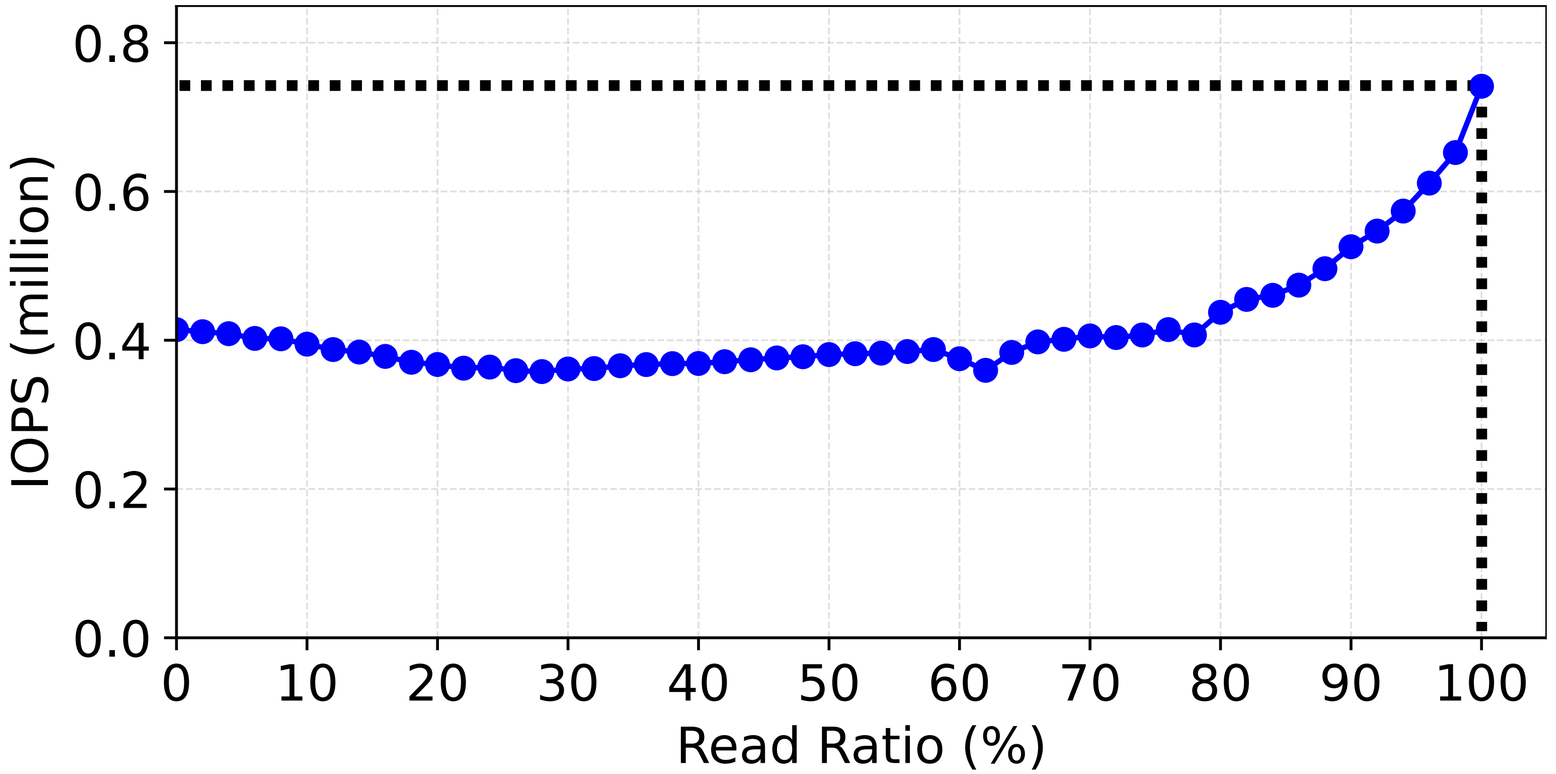
Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



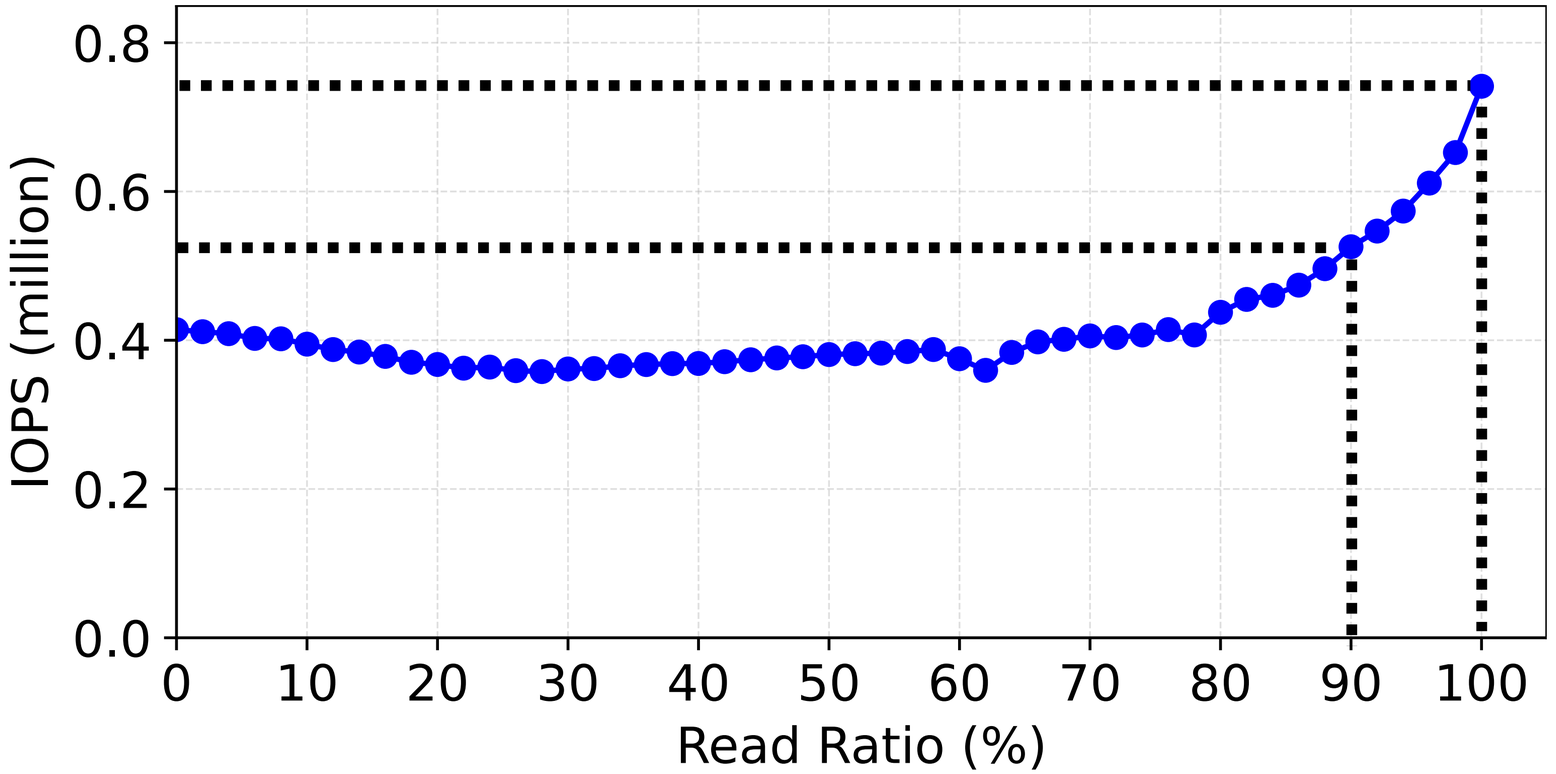
Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



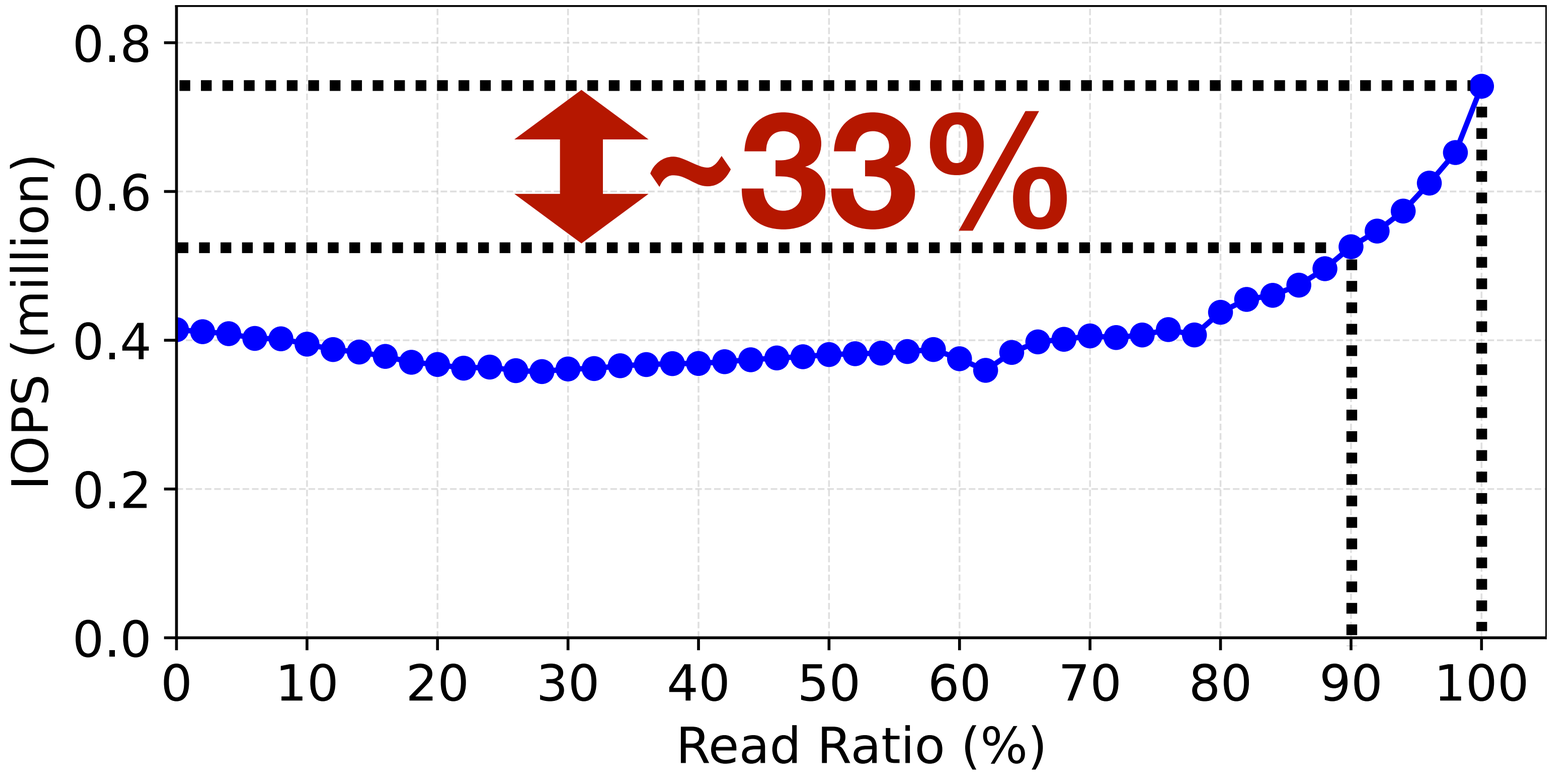
Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



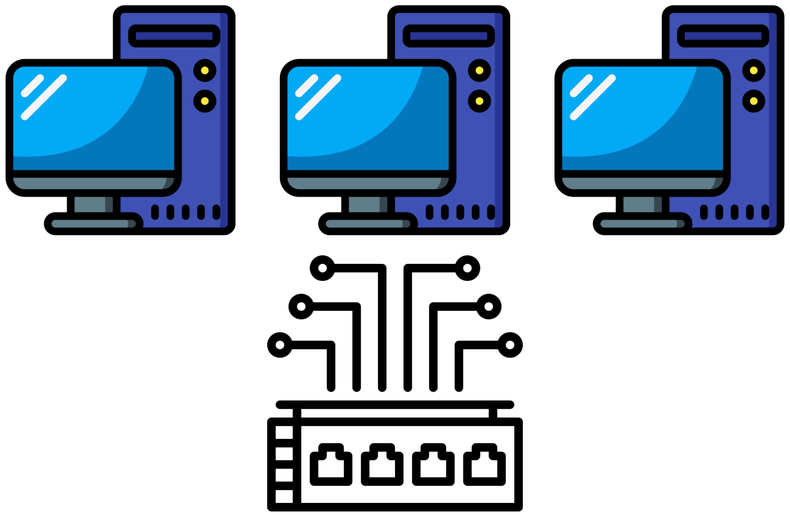
Variability #2: Read/write interference

Different sensitivity to mixing reads and writes



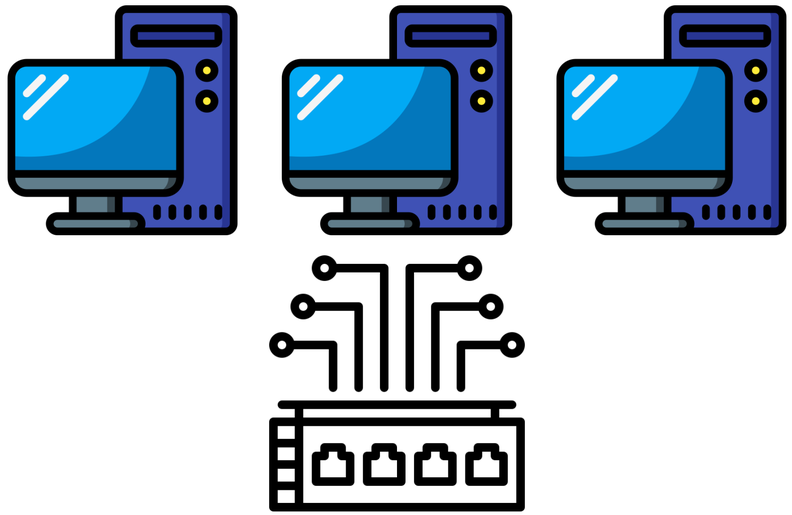
Variability #3: Garbage collection

SSD-internal events can happen anytime



Variability #3: Garbage collection

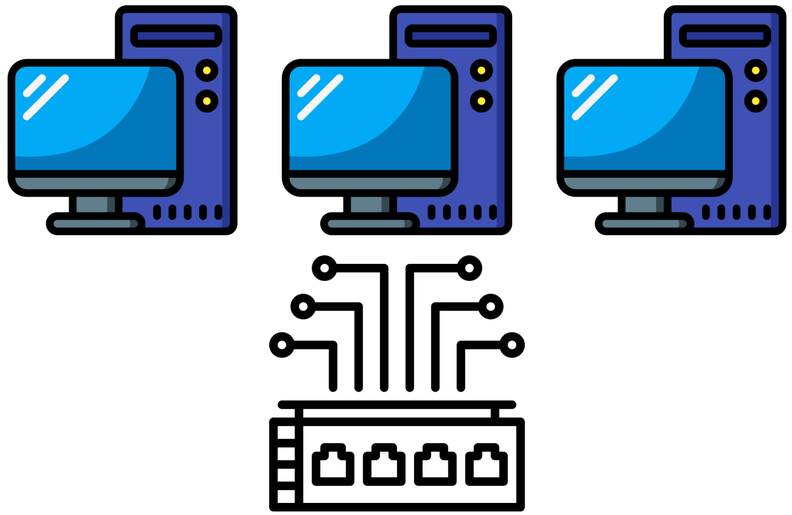
SSD-internal events can happen anytime



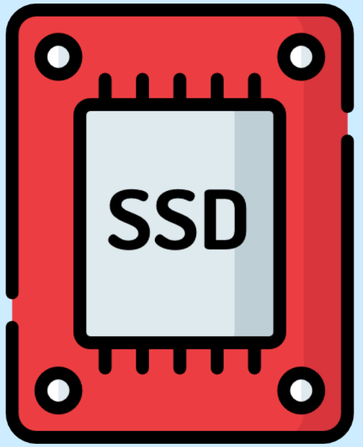
Normal Operation

Variability #3: Garbage collection

SSD-internal events can happen anytime



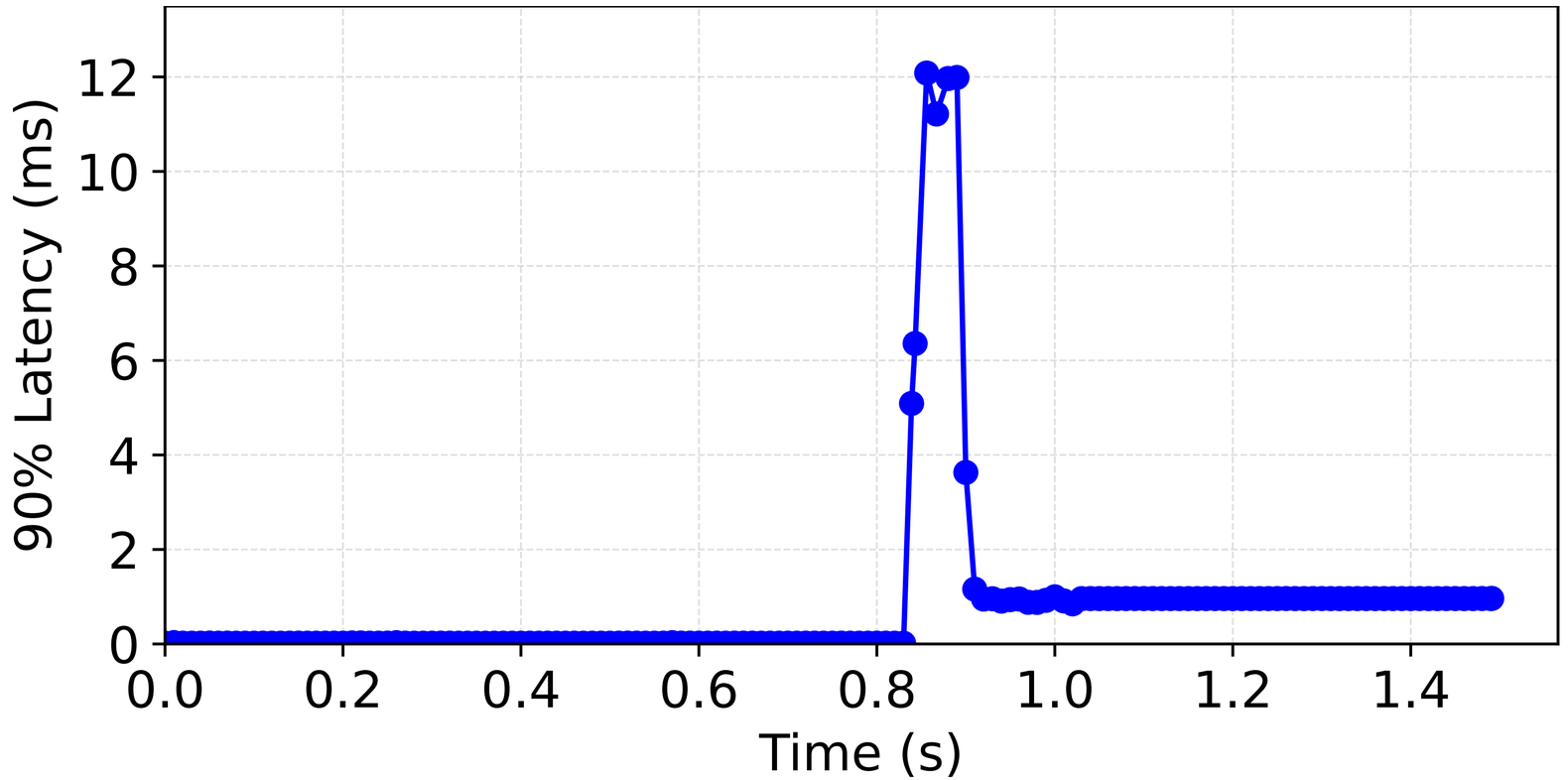
Normal Operation



Garbage Collection

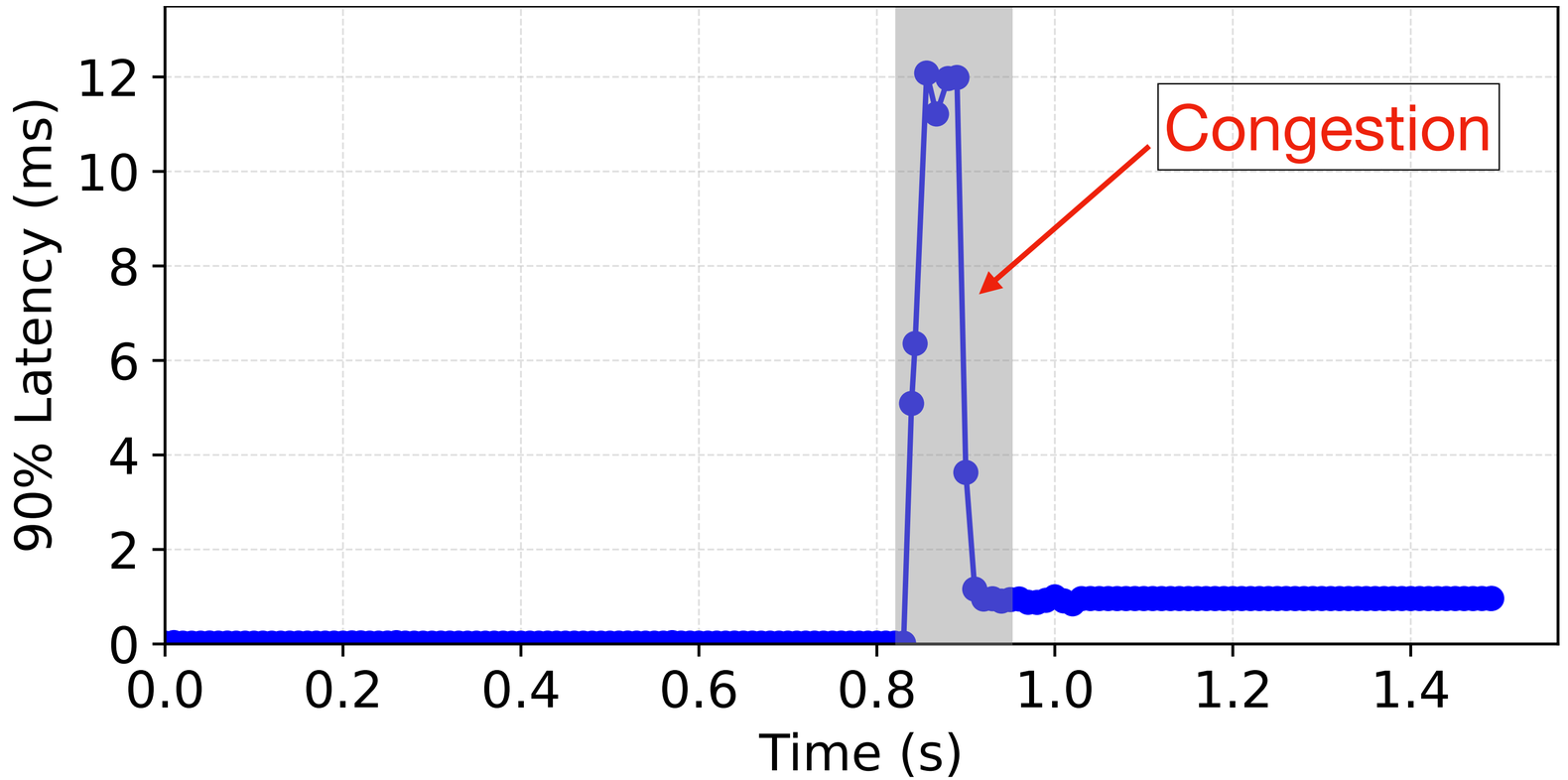
Variability #3: Garbage collection

SSD-internal events can happen anytime



Variability #3: Garbage collection

SSD-internal events can happen anytime



Performance Variability

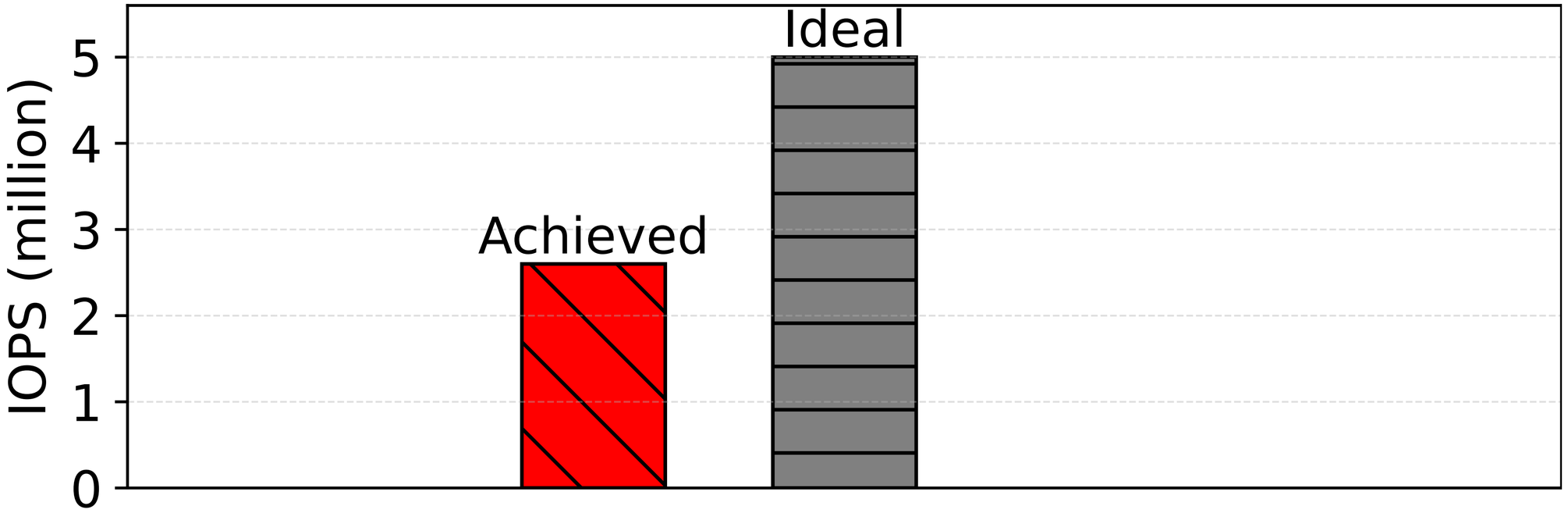
1. Device heterogeneity

2. Read/write interference

3. Garbage collection

Benchmarking our SSD testbed

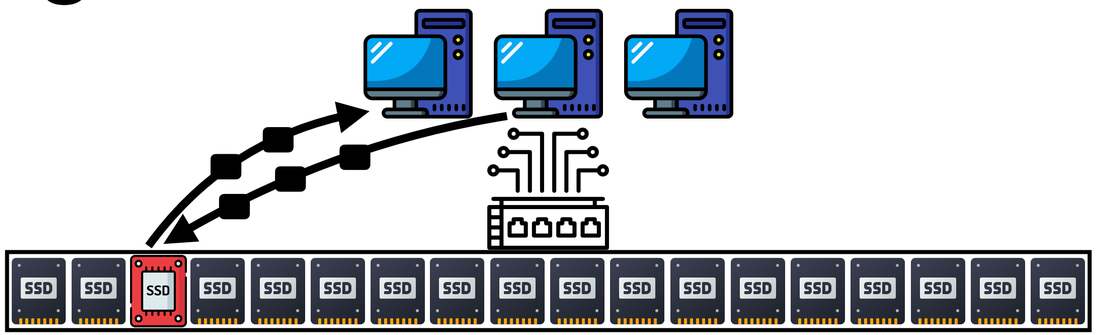
Failing to cohesively manage performance variability leads to lost performance



Challenges

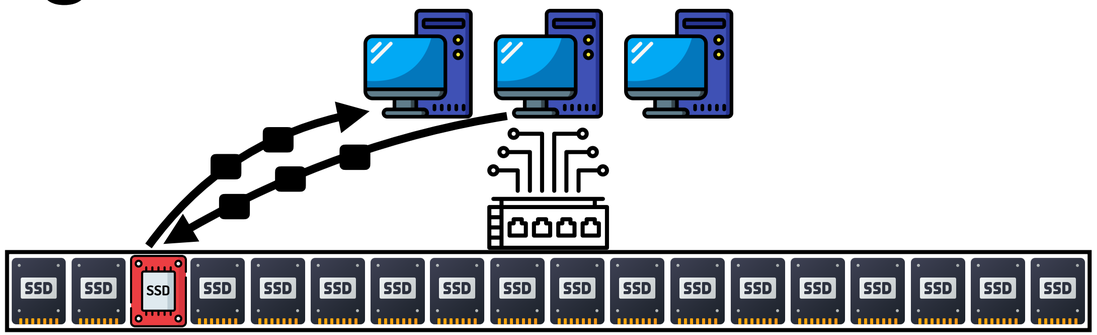
Challenges

1. Inflexible routing

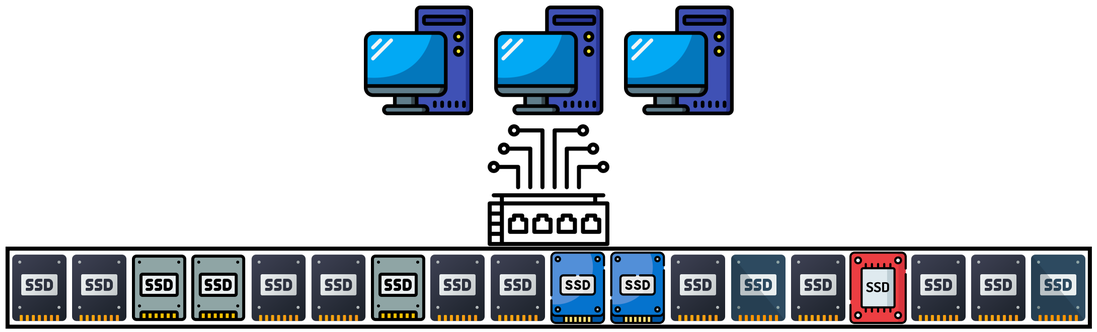


Challenges

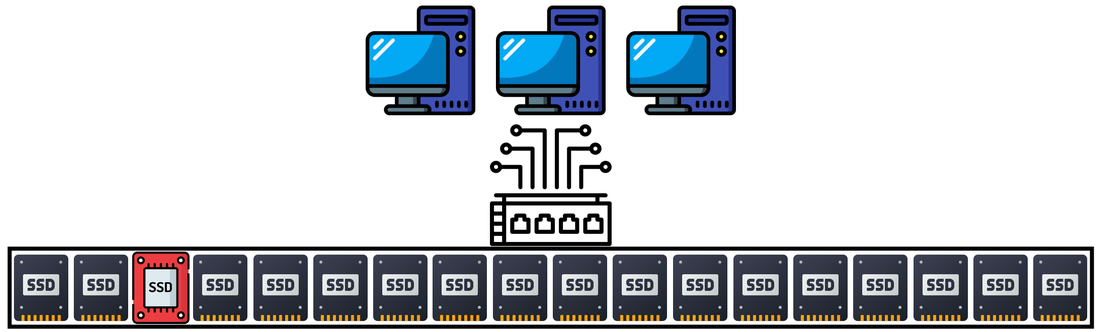
1. Inflexible routing



2. Conflicting timescales



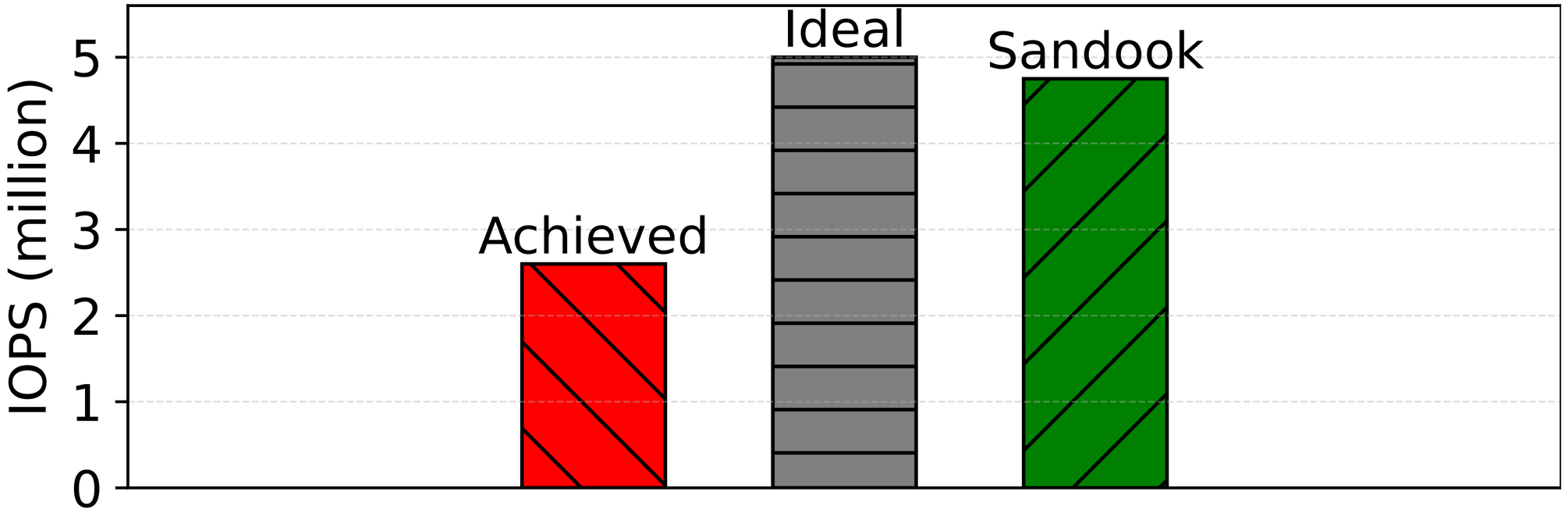
Slower change in load/profiles



Immediate occurrence of GC

Sandook

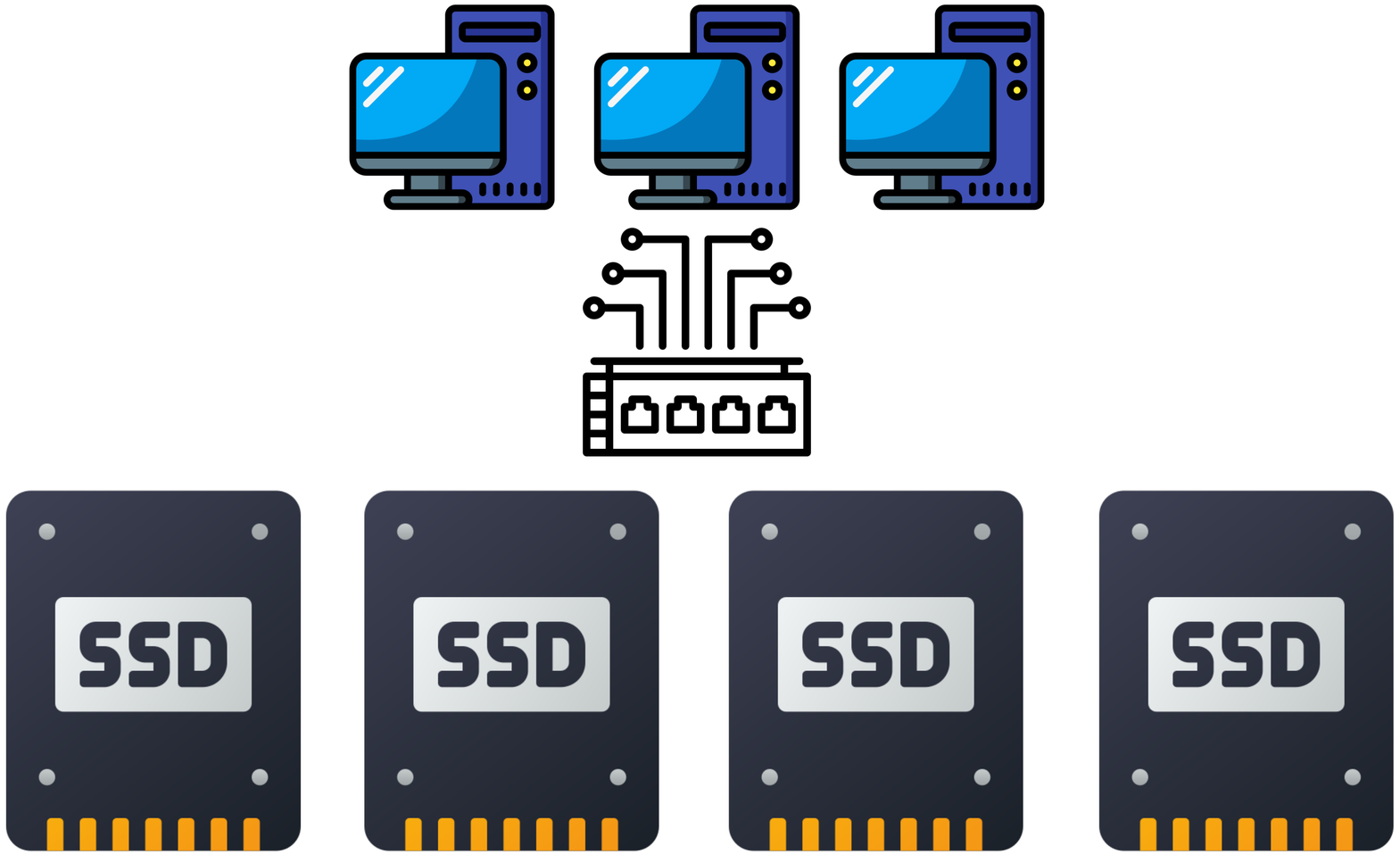
Unleashing the potential of datacenter SSDs by taming performance variability



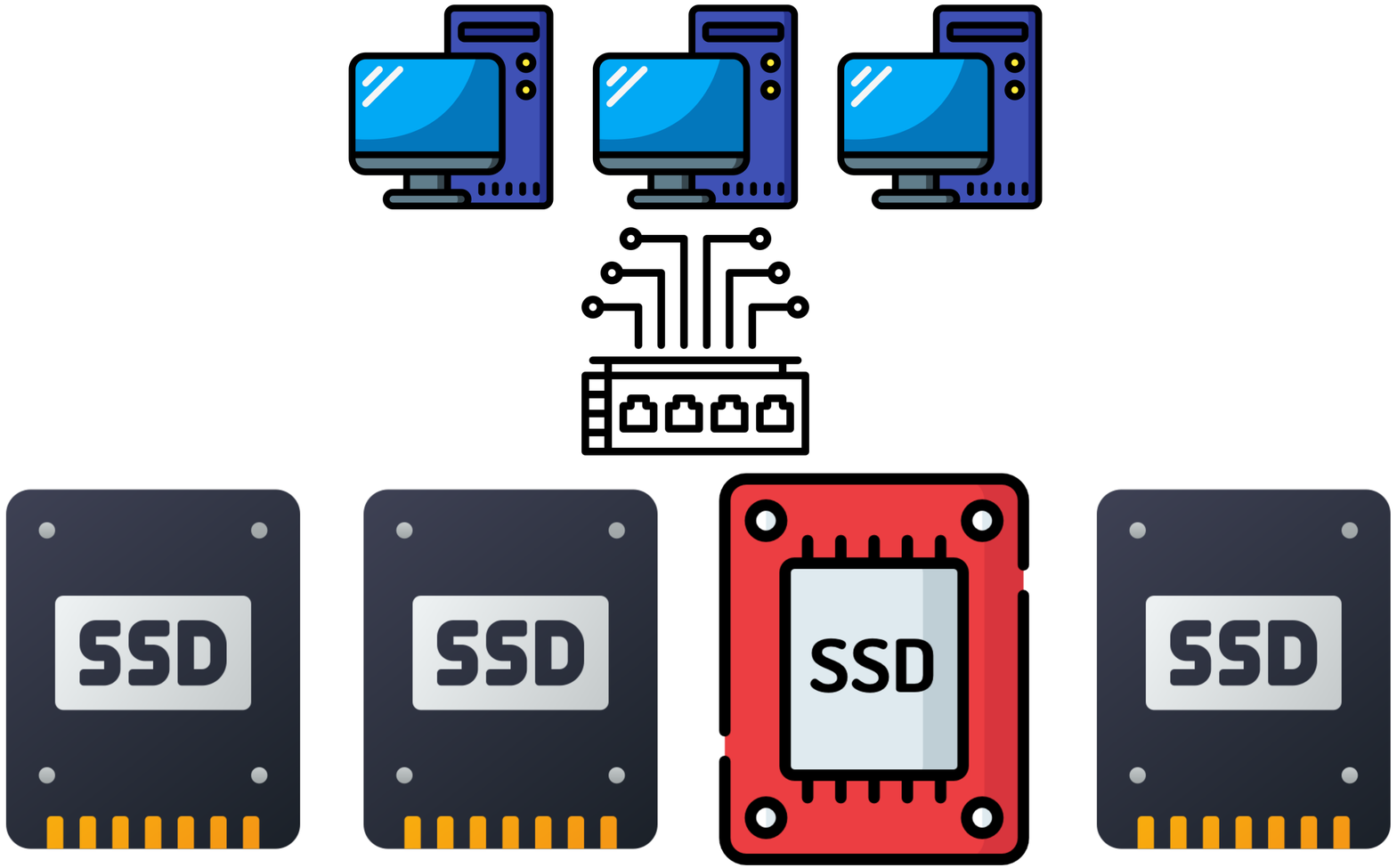
1. Routing flexibility
2. Timescale separation

1. Routing flexibility

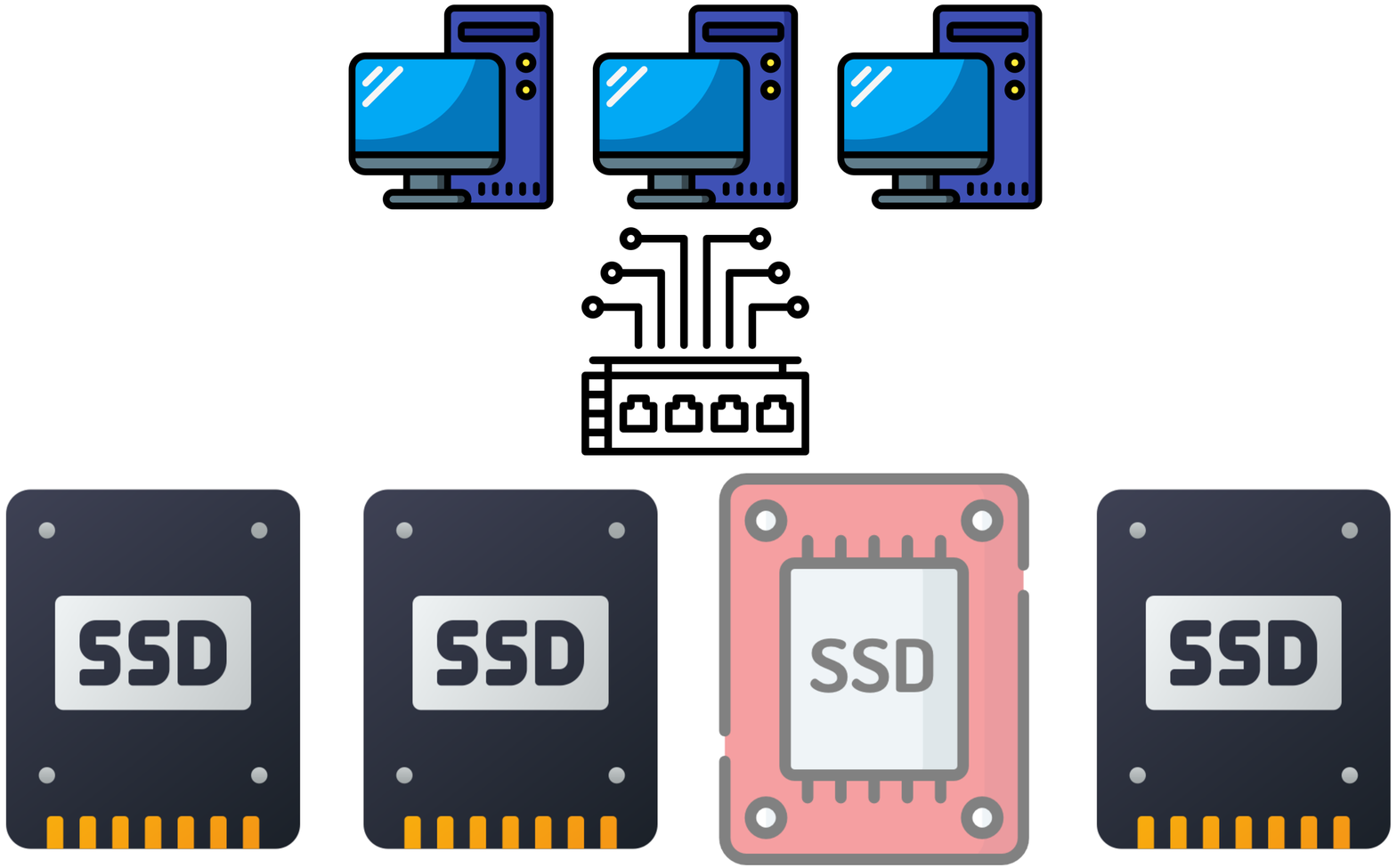
1. Routing flexibility



1. Routing flexibility

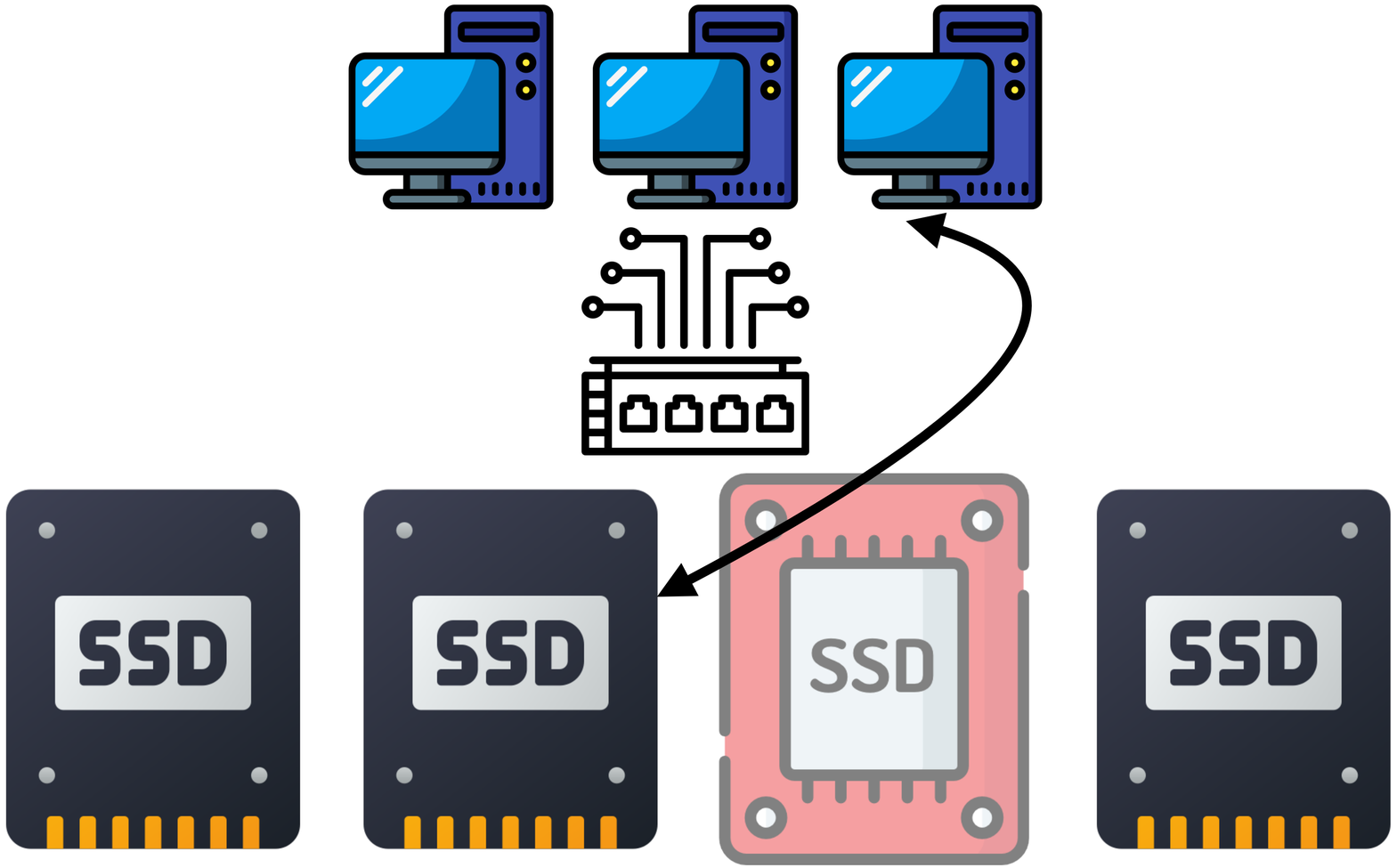


1. Routing flexibility



1. Routing flexibility

Multiple choices to read/write from at runtime



1. Routing flexibility

Multiple choices to read/write from at runtime

1. Routing flexibility

Multiple choices to read/write from at runtime

- **Log-structured** placement layer for writes
  - Can append anywhere

1. Routing flexibility

Multiple choices to read/write from at runtime

- **Log-structured** placement layer for writes
  - Can append anywhere
- **Replication** for ensuring multiple choices per-read
  - Already common in datacenters for fault-tolerance

1. Routing flexibility

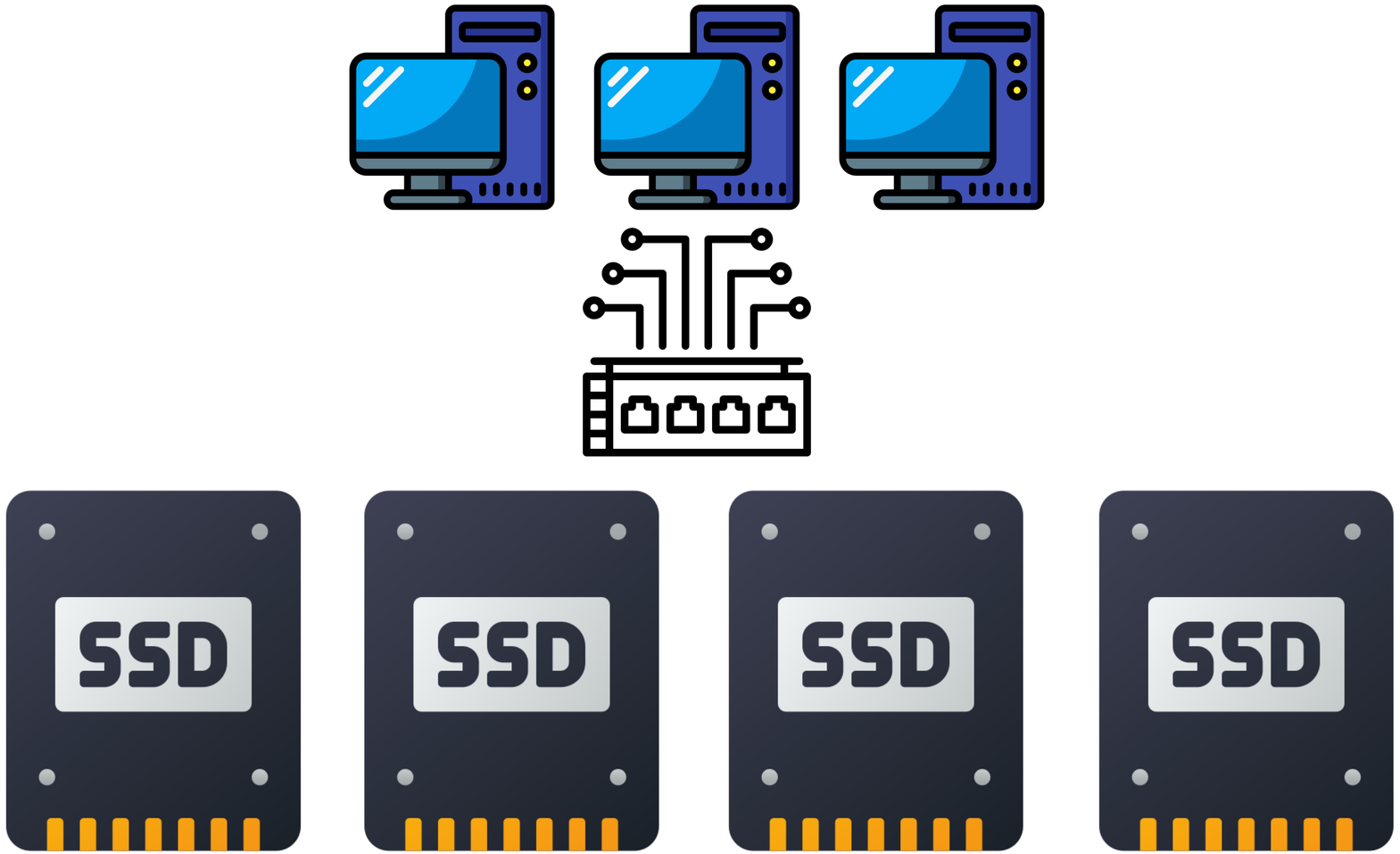
Multiple choices to read/write from at runtime

- **Log-structured** placement layer for writes
  - Can append anywhere
- **Replication** for ensuring multiple choices per-read
  - Already common in datacenters for fault-tolerance

Can run diverse scheduling policies without tripping over one another

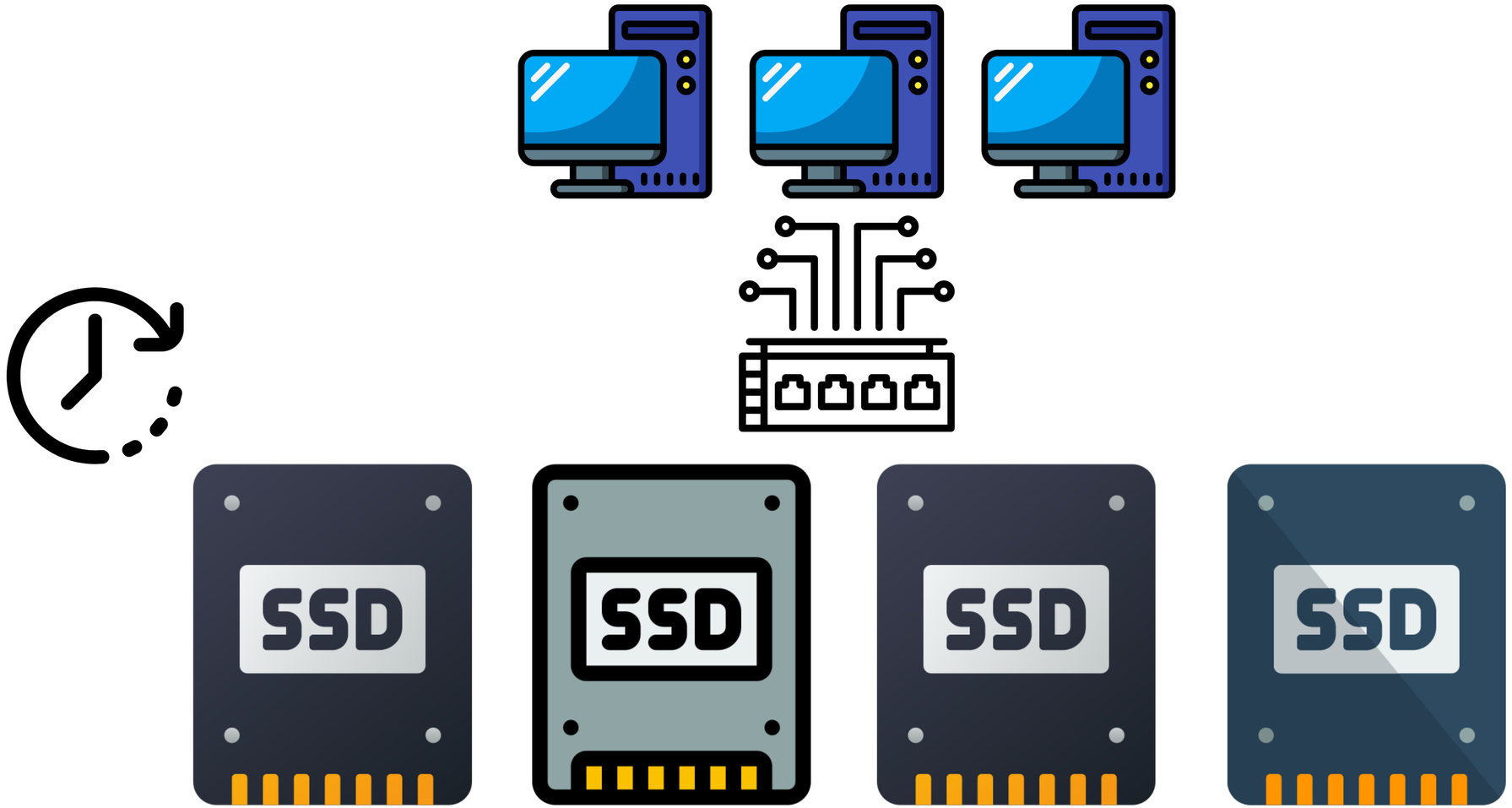
2. Timescale separation

2. Timescale separation



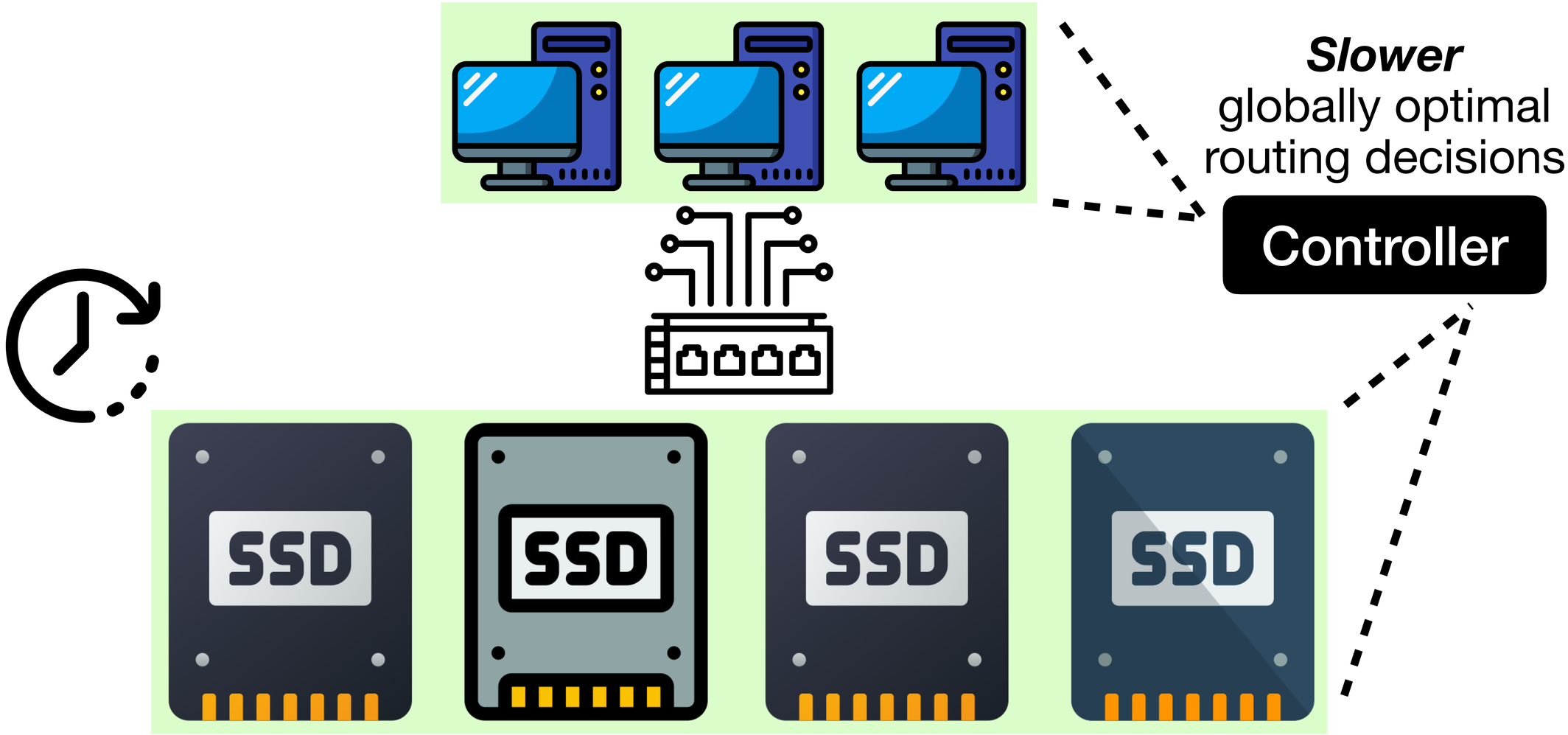
2. Timescale separation

SSD characteristics may shift *gradually*



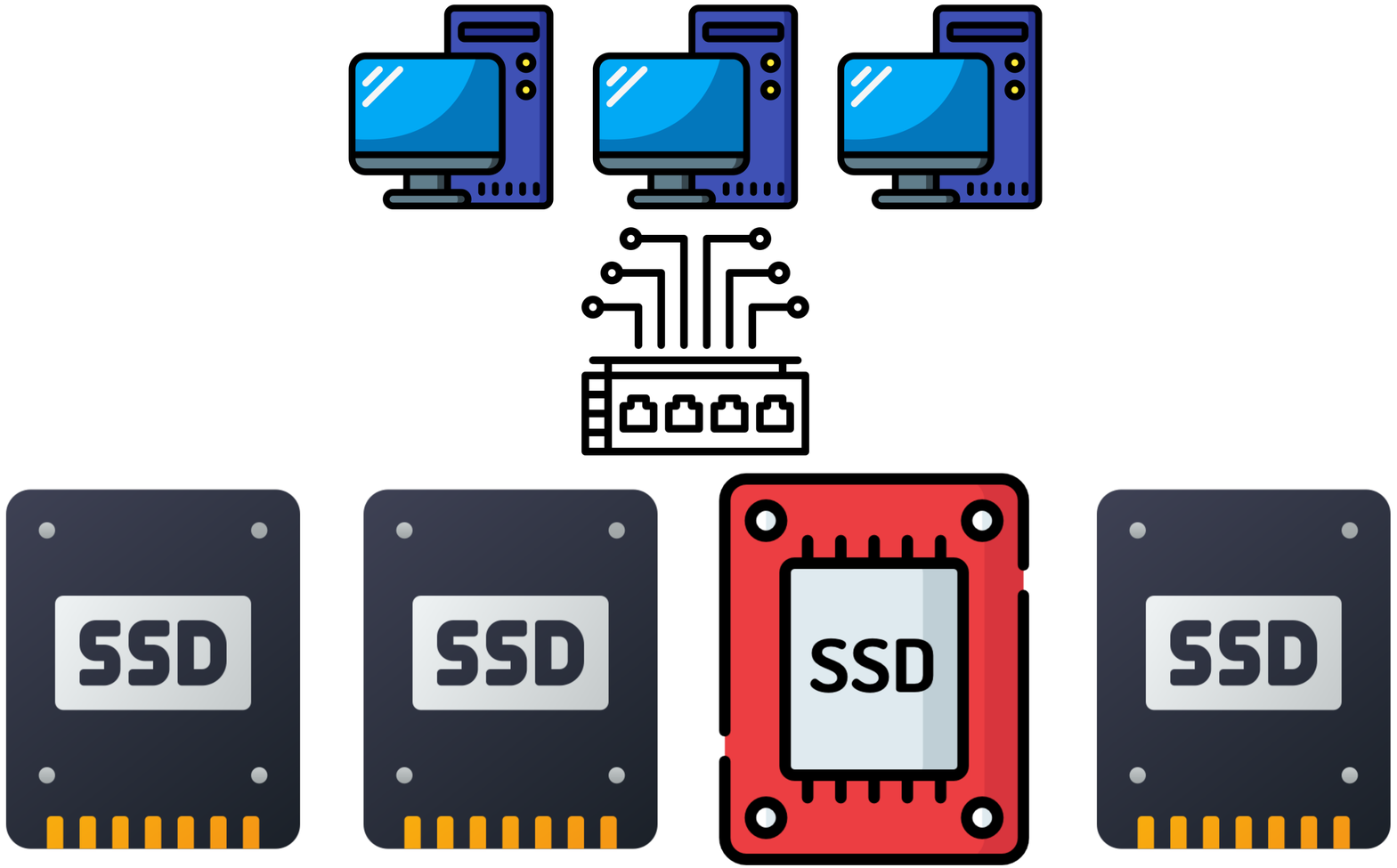
2. Timescale separation

SSD characteristics may shift *gradually*



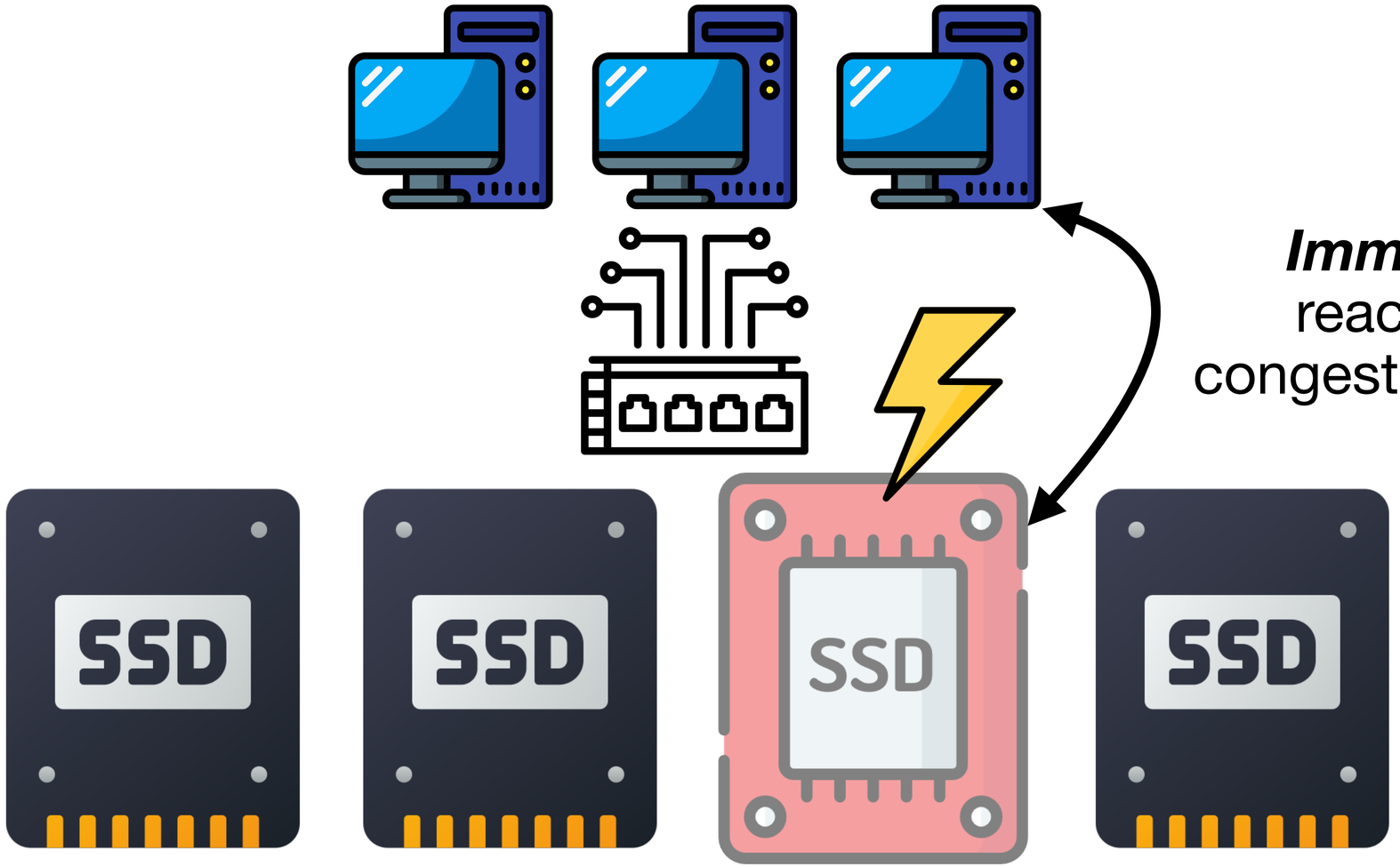
2. Timescale separation

SSD-internal events can happen *anytime*, at *sub-millisecond* scales



2. Timescale separation

SSD-internal events can happen *anytime*, at *sub-millisecond* scales



Immediate
reaction to
congestion signals

2. Timescale separation

Detect short-term as well as long-horizon variability

2. Timescale separation

Detect short-term as well as long-horizon variability

- **Centralized controller** for global scheduling decisions that evolve **slowly**
  - Per-SSD routing rules from global demand and SSD performance

2. Timescale separation

Detect short-term as well as long-horizon variability

- **Centralized controller** for global scheduling decisions that evolve **slowly**
  - Per-SSD routing rules from global demand and SSD performance
- **Clients** may temporarily disobey routing rules in **immediate** response to congestion signals from the SSDs

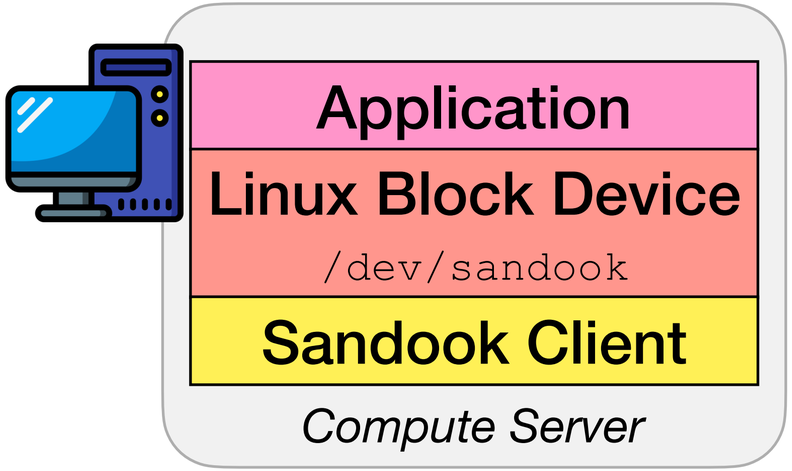
2. Timescale separation

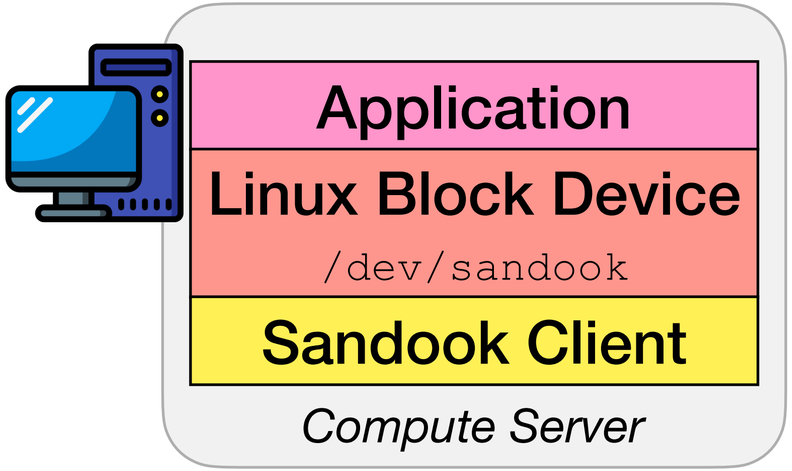
Detect short-term as well as long-horizon variability

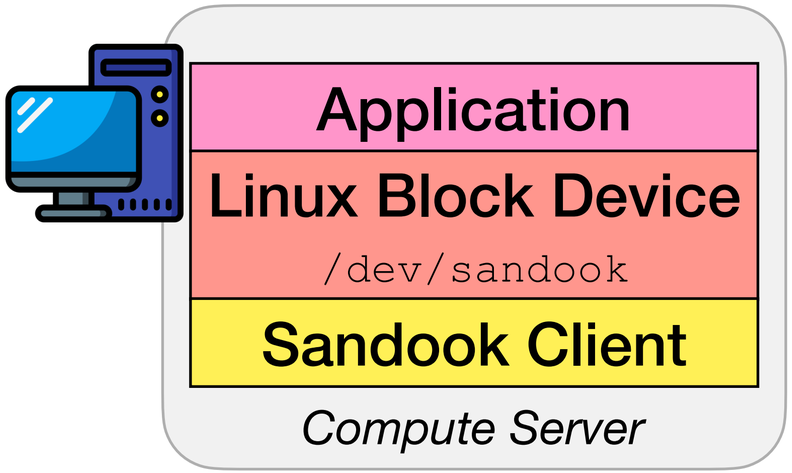
- **Centralized controller** for global scheduling decisions that evolve **slowly**
  - Per-SSD routing rules from global demand and SSD performance
- **Clients** may temporarily disobey routing rules in **immediate** response to congestion signals from the SSDs

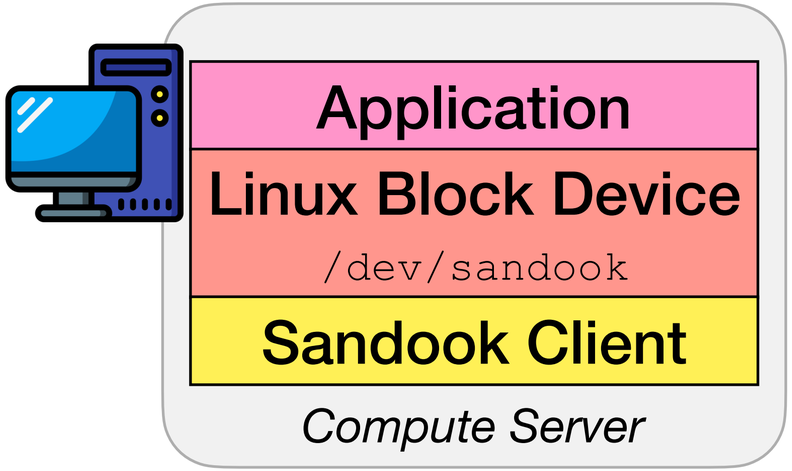
Fast reaction times while maintaining close to optimal routing, without requiring a controller in the critical path

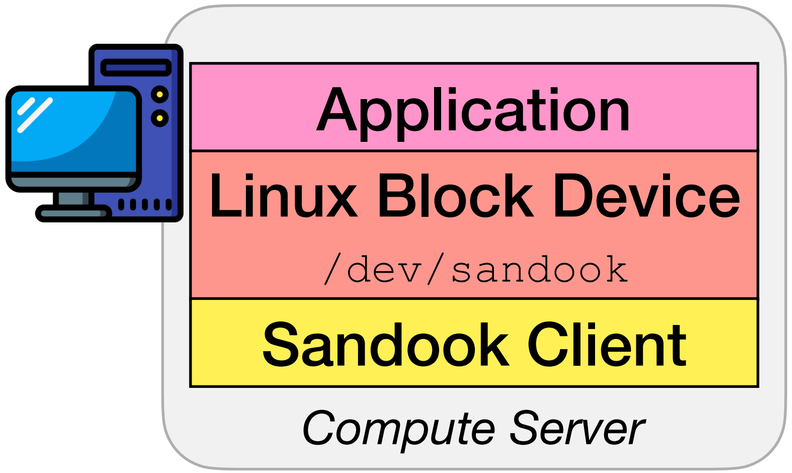
Sandook’s Design

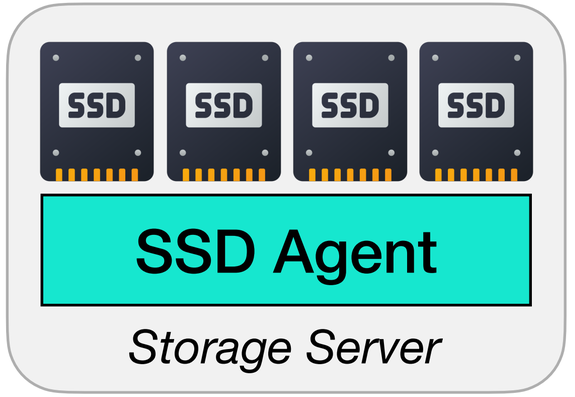

Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server

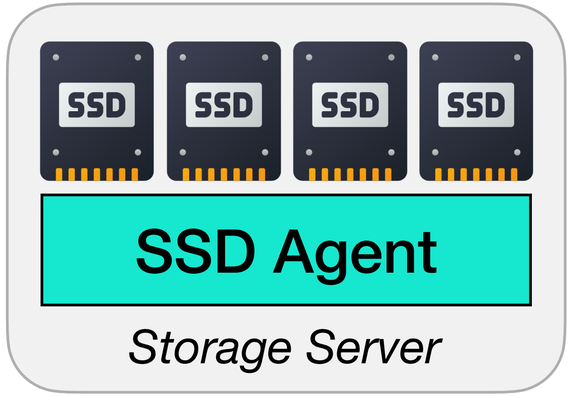

Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server

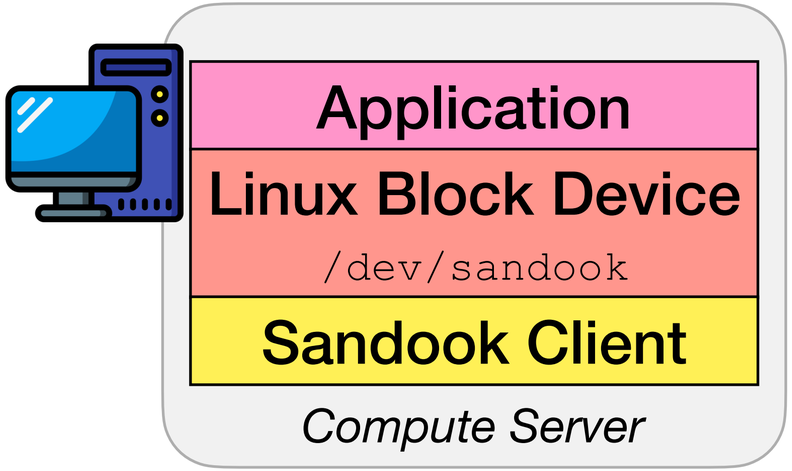

Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server

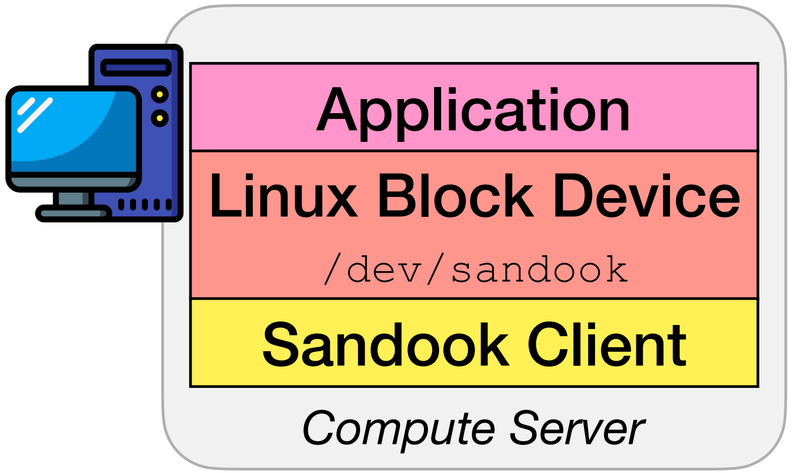

Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server

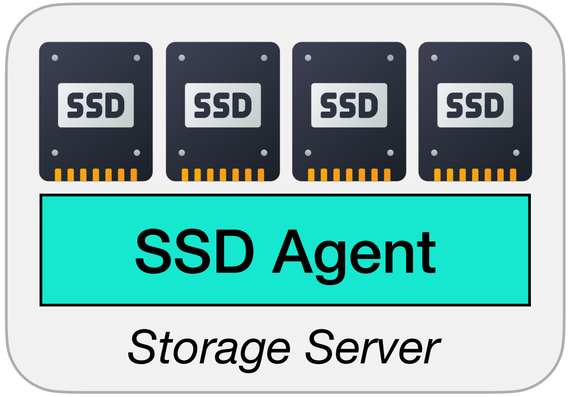

Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server

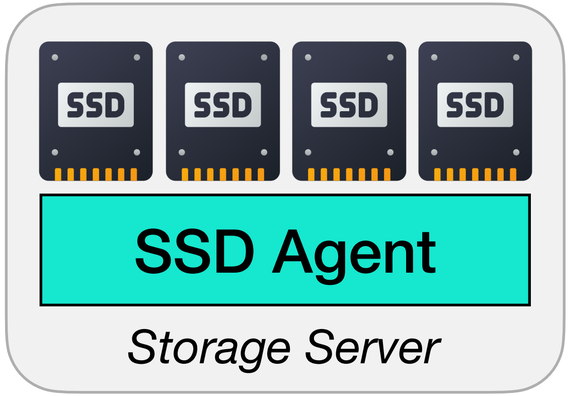

SSD
SSD
SSD
SSD
SSD Agent
Storage Server


SSD
SSD
SSD
SSD
SSD Agent
Storage Server

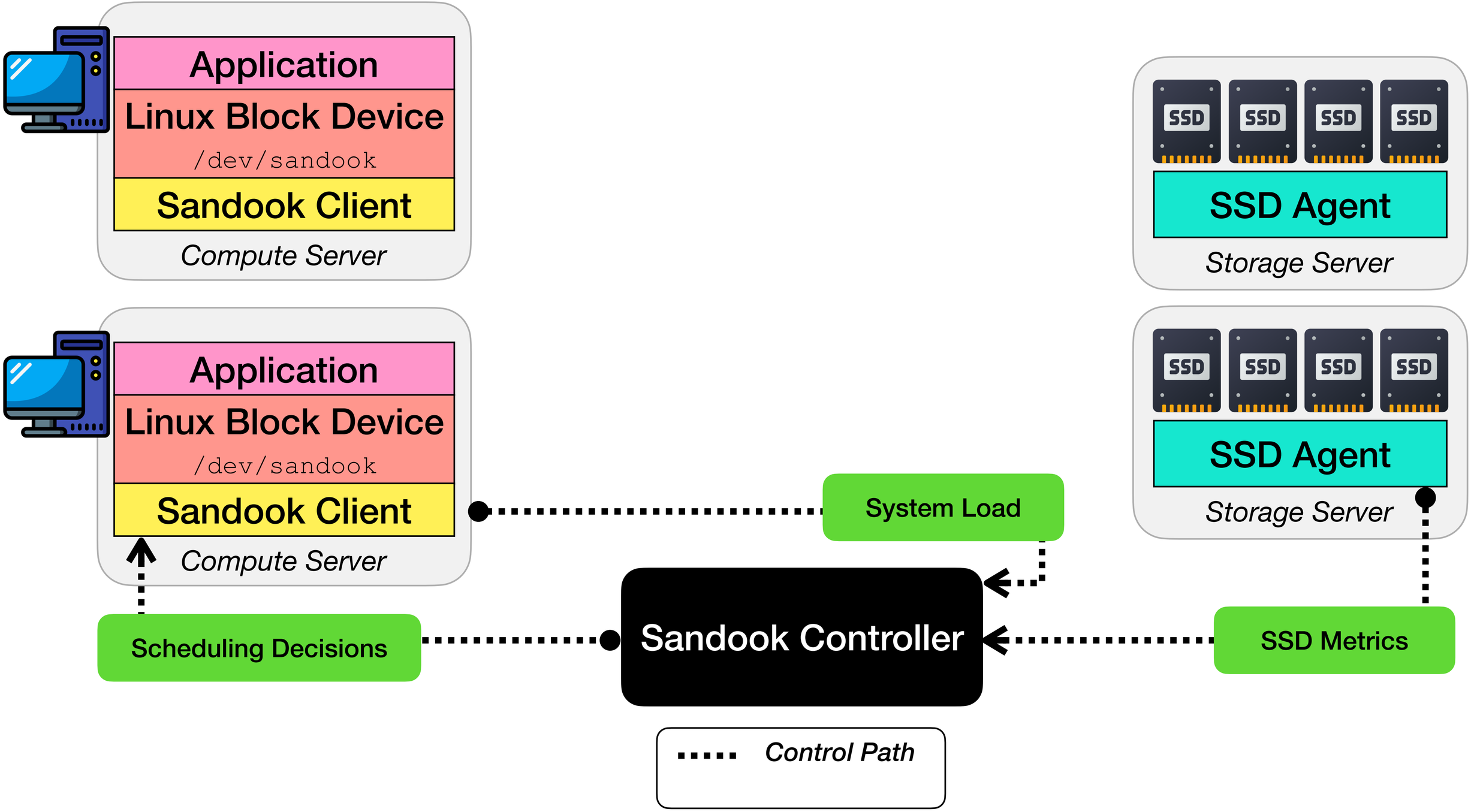


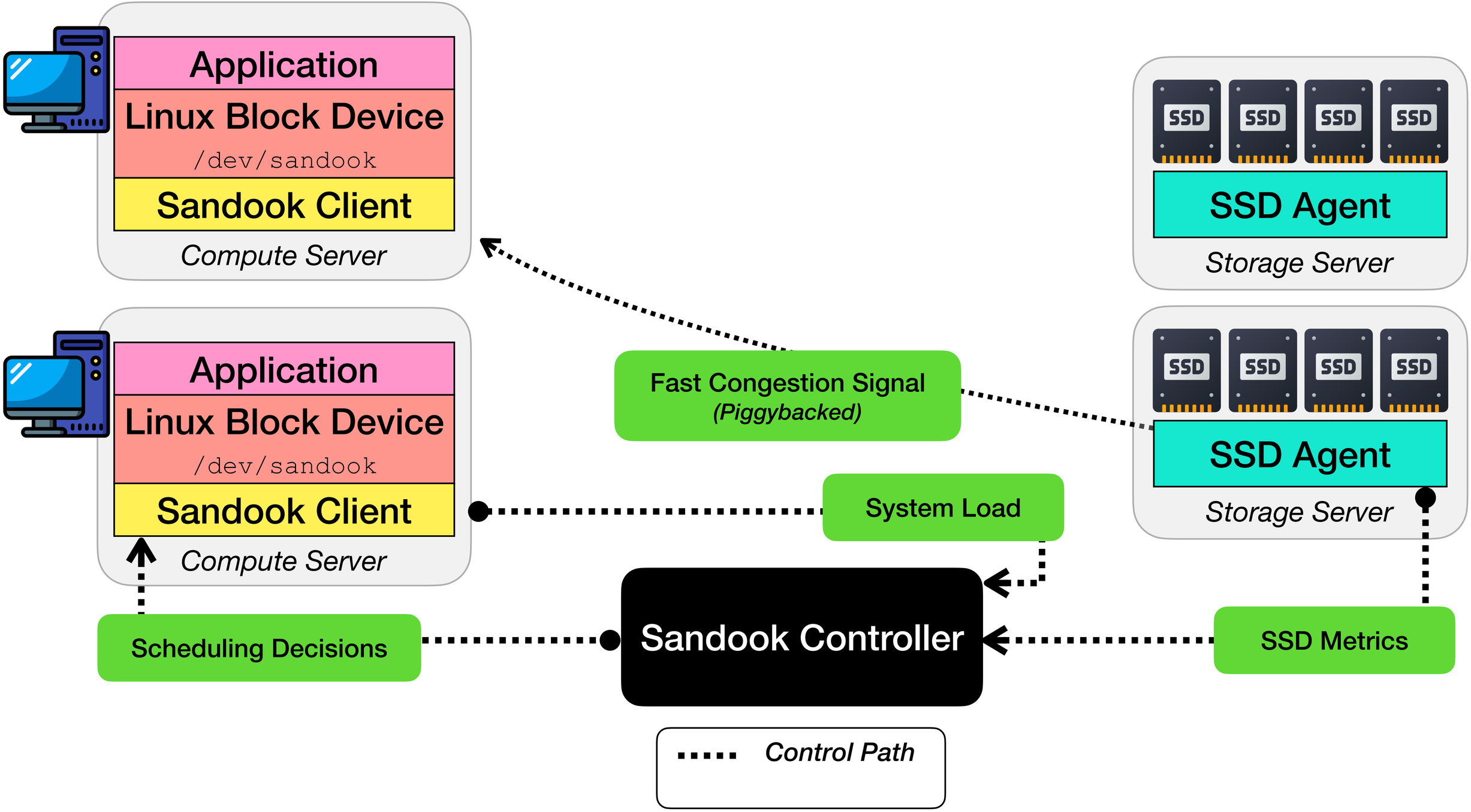


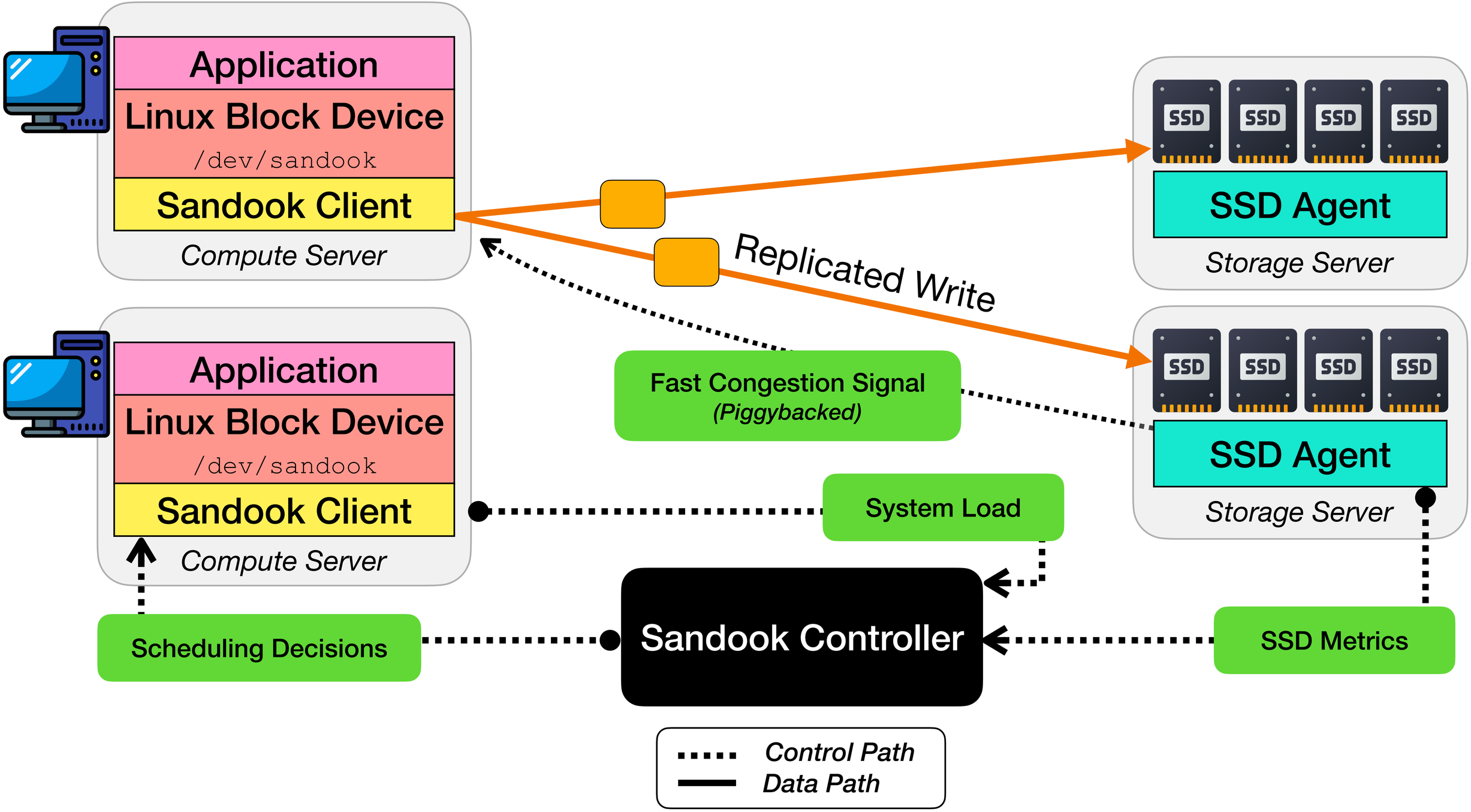


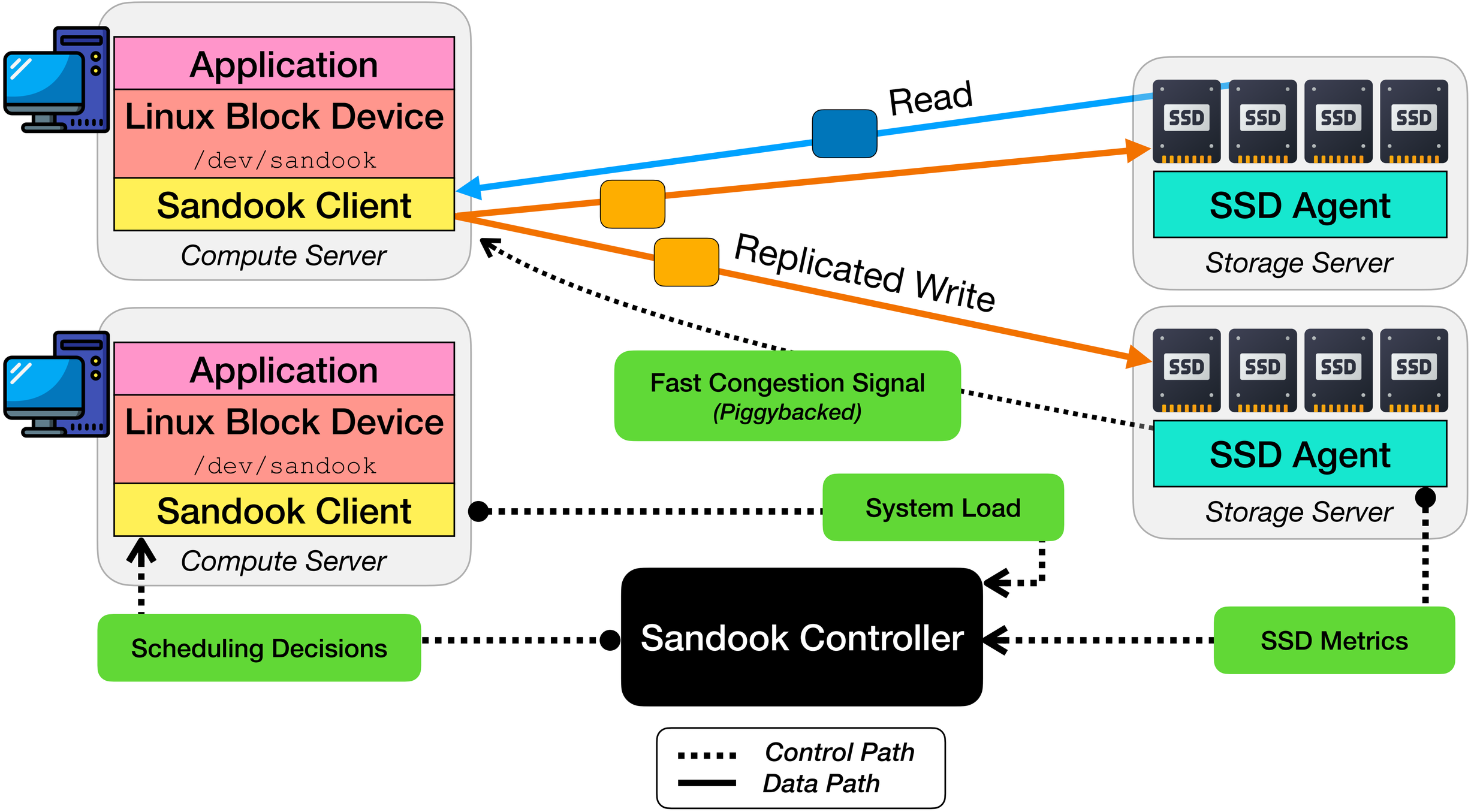


Sandook Controller


Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
System Load
Scheduling Decisions
Sandook Controller
SSD Metrics
Control Path


Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
Fast Congestion Signal
(Piggybacked)
System Load
Scheduling Decisions
Sandook Controller
SSD Metrics
Control Path


Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
Replicated Write
Fast Congestion Signal
(Piggybacked)
System Load
Sandook Controller
Scheduling Decisions
SSD Metrics
Control Path
Data Path


Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
Application
Linux Block Device
/dev/sandook
Sandook Client
Compute Server
Read
Replicated Write
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
SSD
SSD
SSD
SSD
SSD Agent
Storage Server
Fast Congestion Signal
(Piggybacked)
System Load
Sandook Controller
Scheduling Decisions
SSD Metrics
Control Path
Data Path

How does this design help us address performance variability?

Performance Variability	Policy	Design Aspect

Performance Variability	Policy	Design Aspect
1. Device heterogeneity	Profile-driven load steering	SSD metrics + system load *(slower timescale)*

Performance Variability	Policy	Design Aspect
1. Device heterogeneity	Profile-driven load steering	SSD metrics + system load *(slower timescale)*
2. Read/write interference	Read/write segregation	Replication/choices *(routing flexibility)*

Performance Variability	Policy	Design Aspect
1. Device heterogeneity	Profile-driven load steering	SSD metrics + system load *(slower timescale)*
2. Read/write interference	Read/write segregation	Replication/choices *(routing flexibility)*
3. Garbage collection	Storage congestion control	Congestion signal *(faster timescale)*

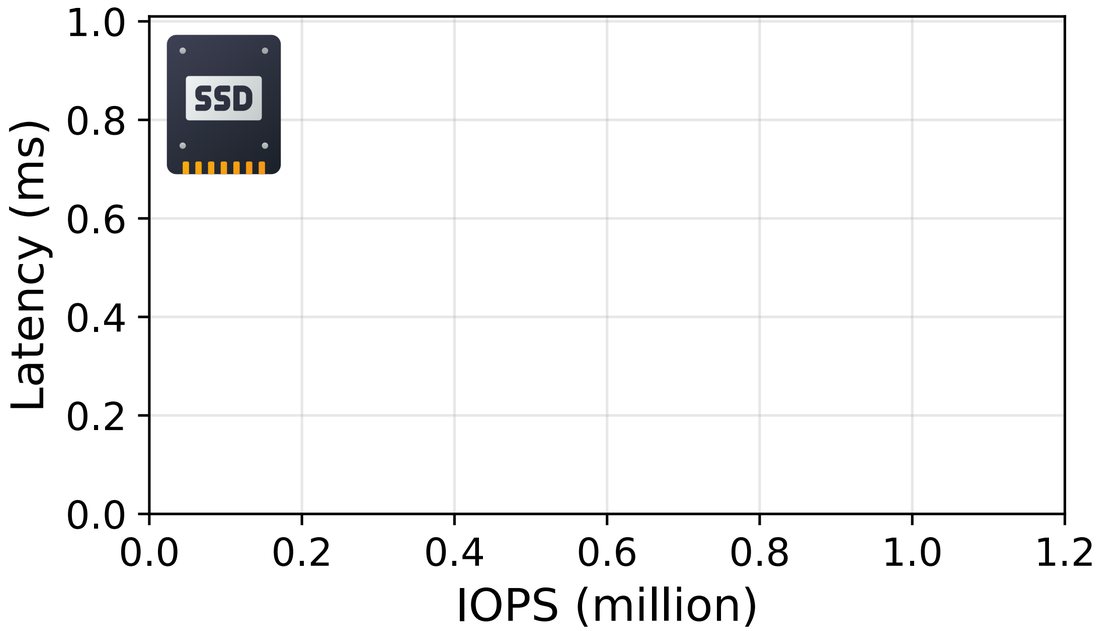
Performance Variability	Policy	Design Aspect
1. Device heterogeneity	Profile-driven load steering	SSD metrics + system load *(slower timescale)*
2. Read/write interference	Read/write segregation	Replication/choices *(routing flexibility)*
3. Garbage collection	Storage congestion control	Congestion signal *(faster timescale)*

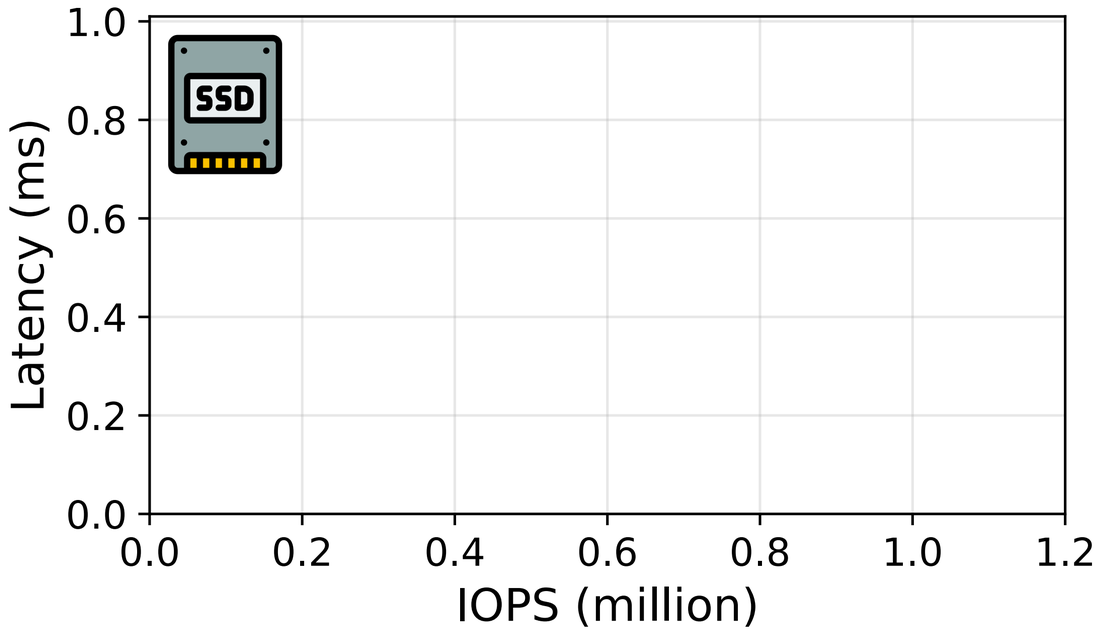
1. Profile-driven load steering

Incorporating performance capabilities of each SSD

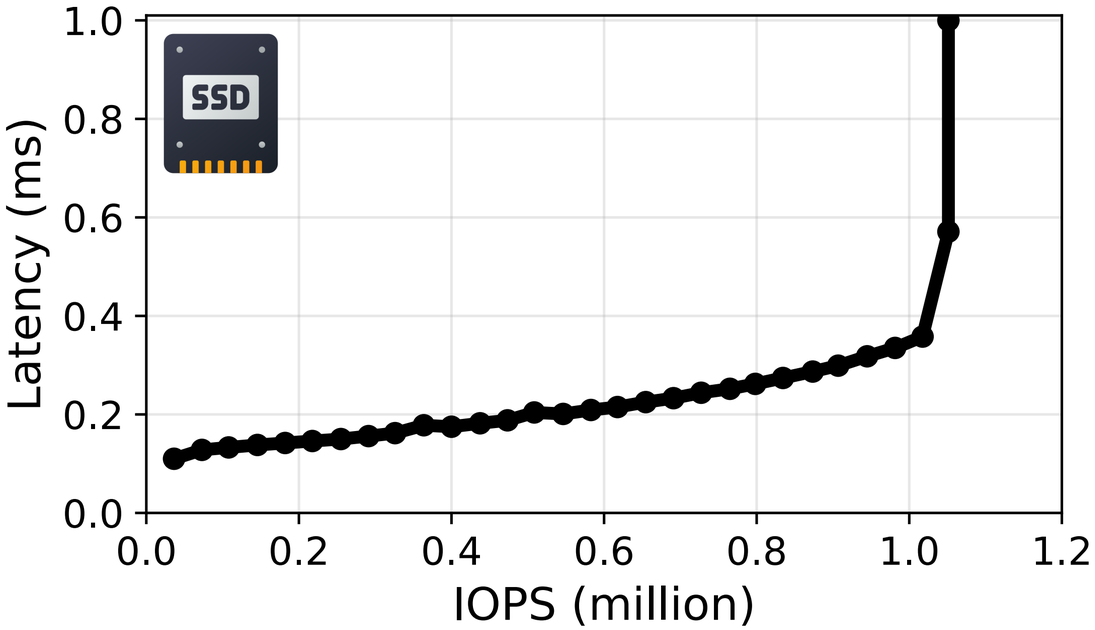
- Two objectives (using Linear Programming):
  - Satisfy system-wide IOPS demand
  - Minimize overall latency
- Consider load-latency profiles of each SSD to distribute load

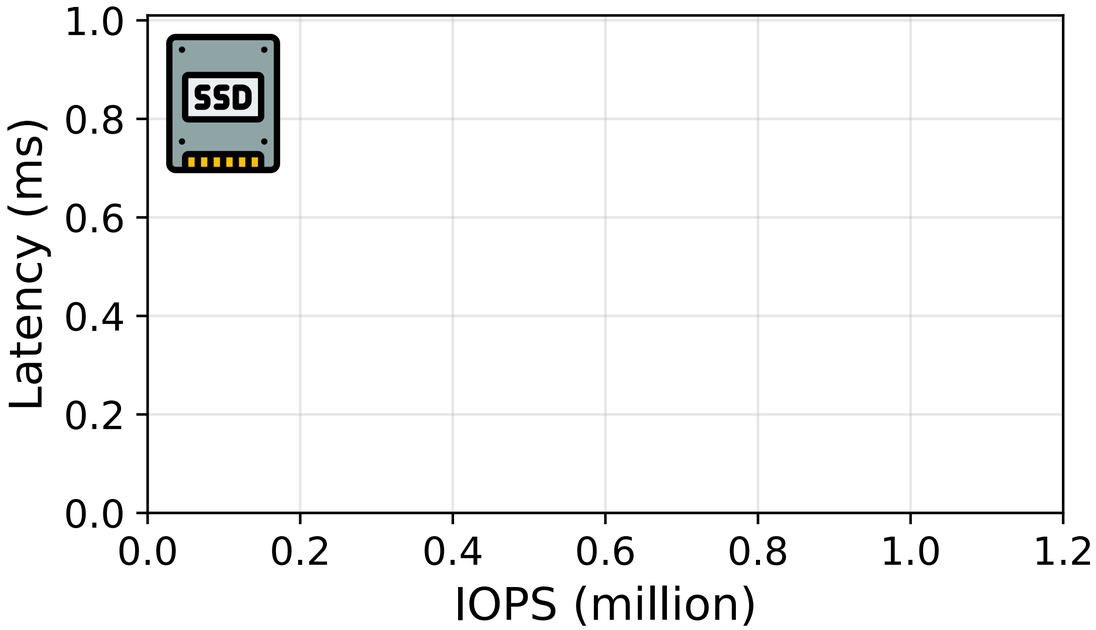
1. Profile-driven load steering



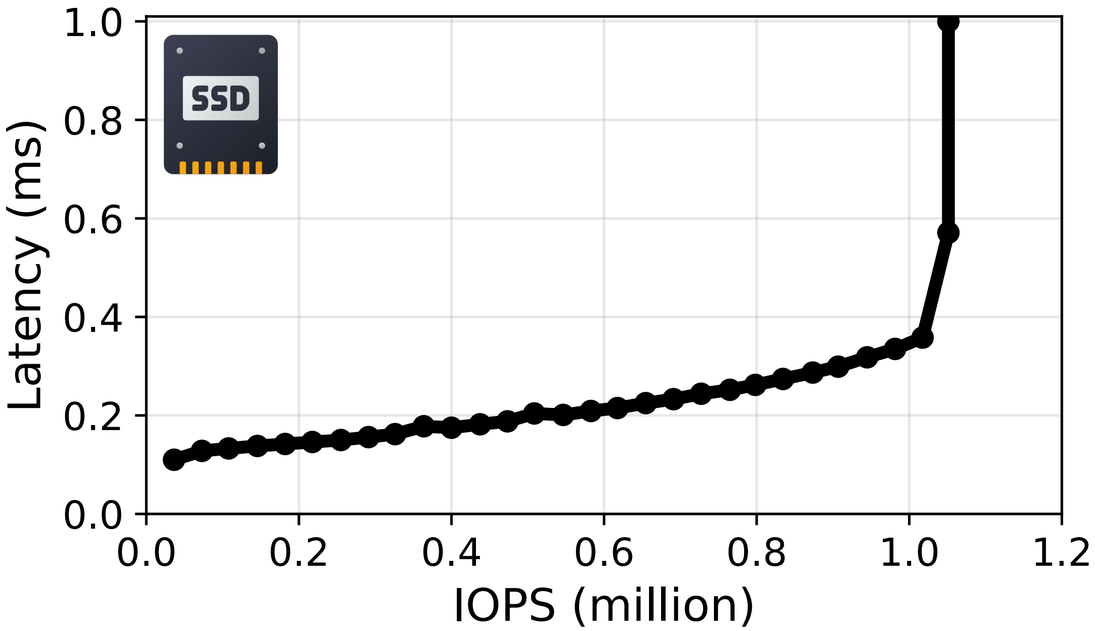


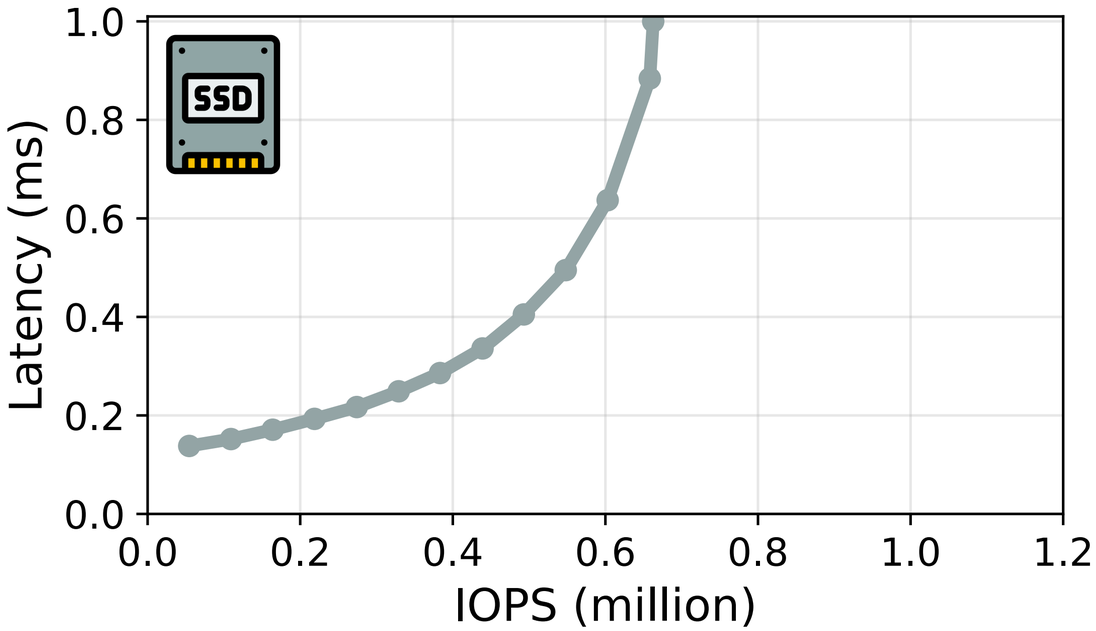
1. Profile-driven load steering



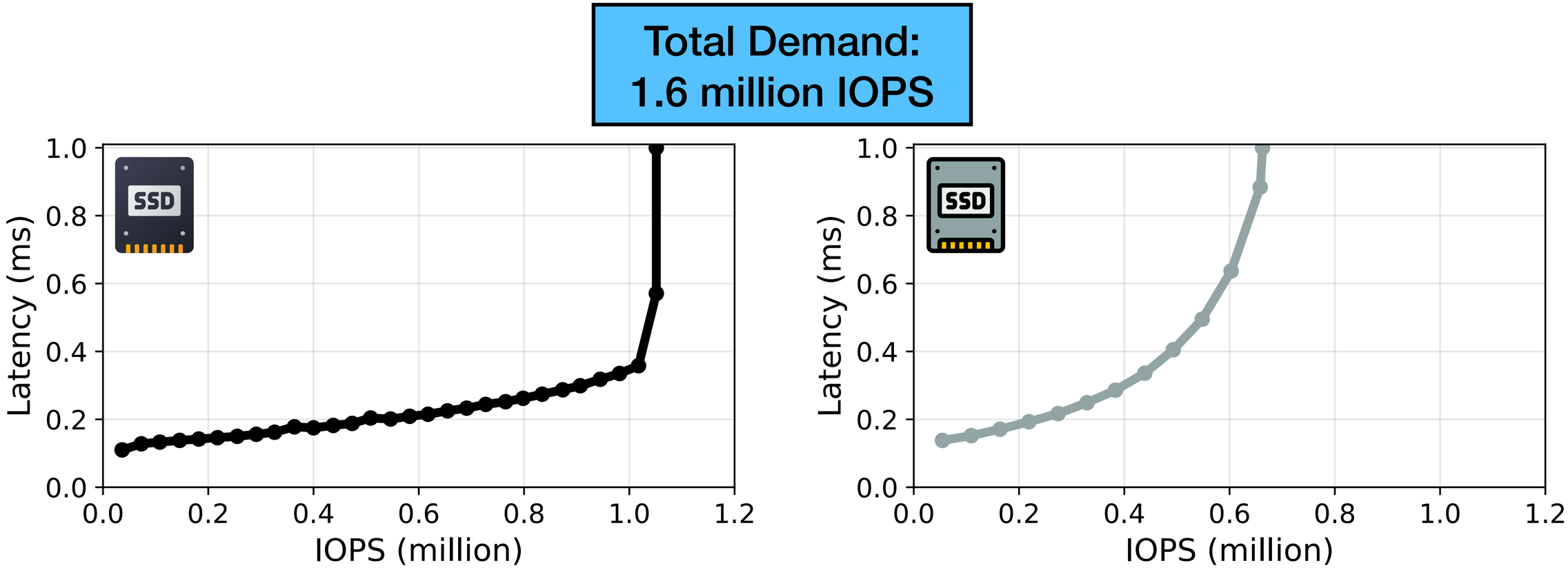


1. Profile-driven load steering



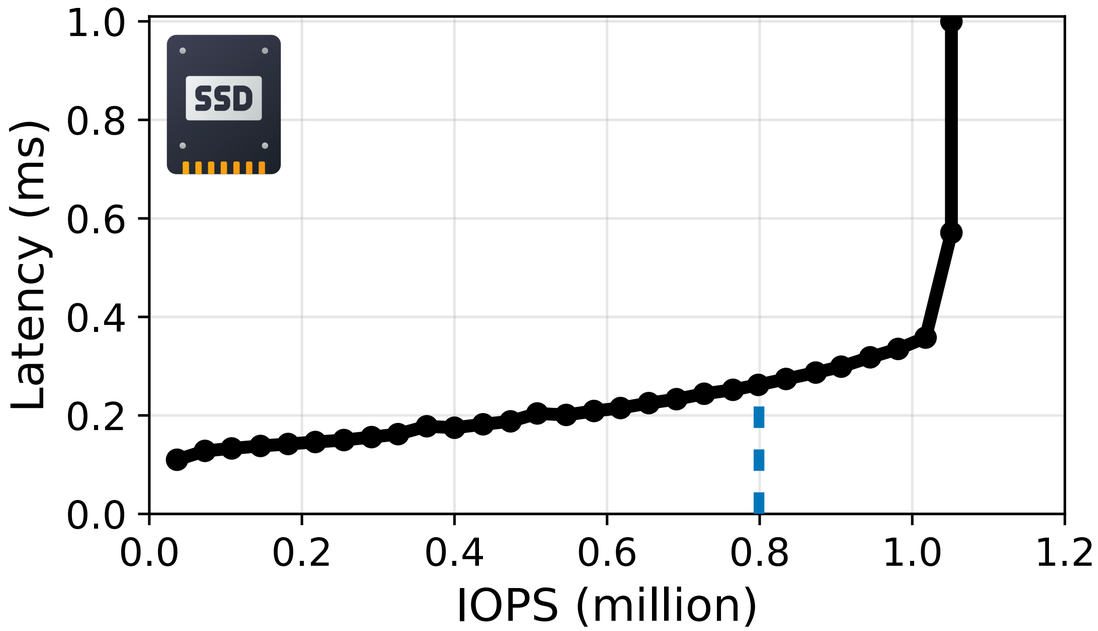


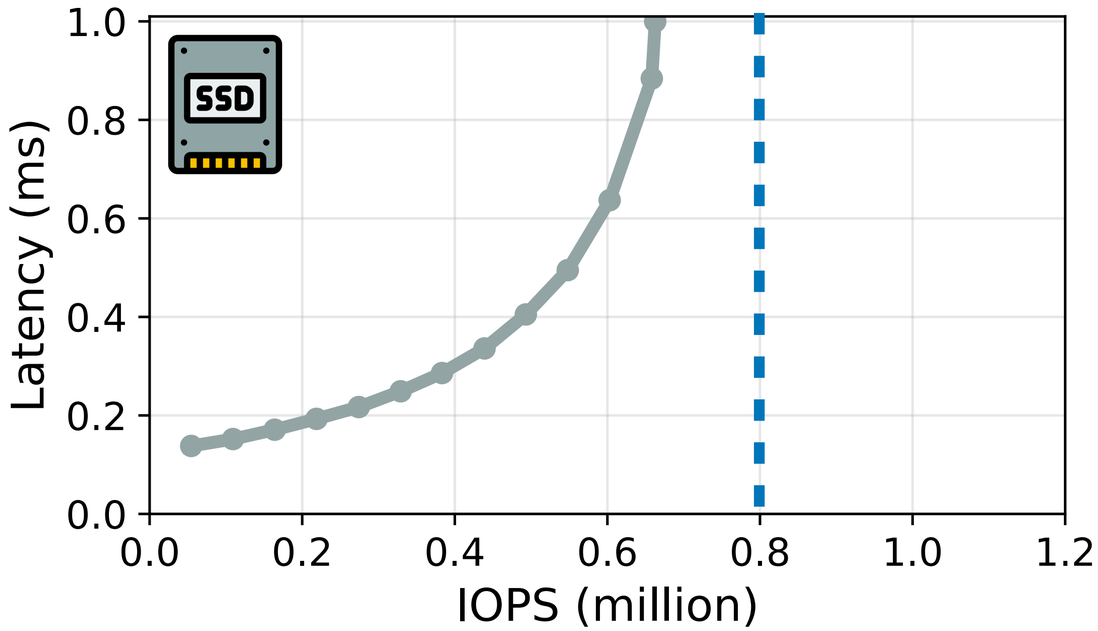
1. Profile-driven load steering



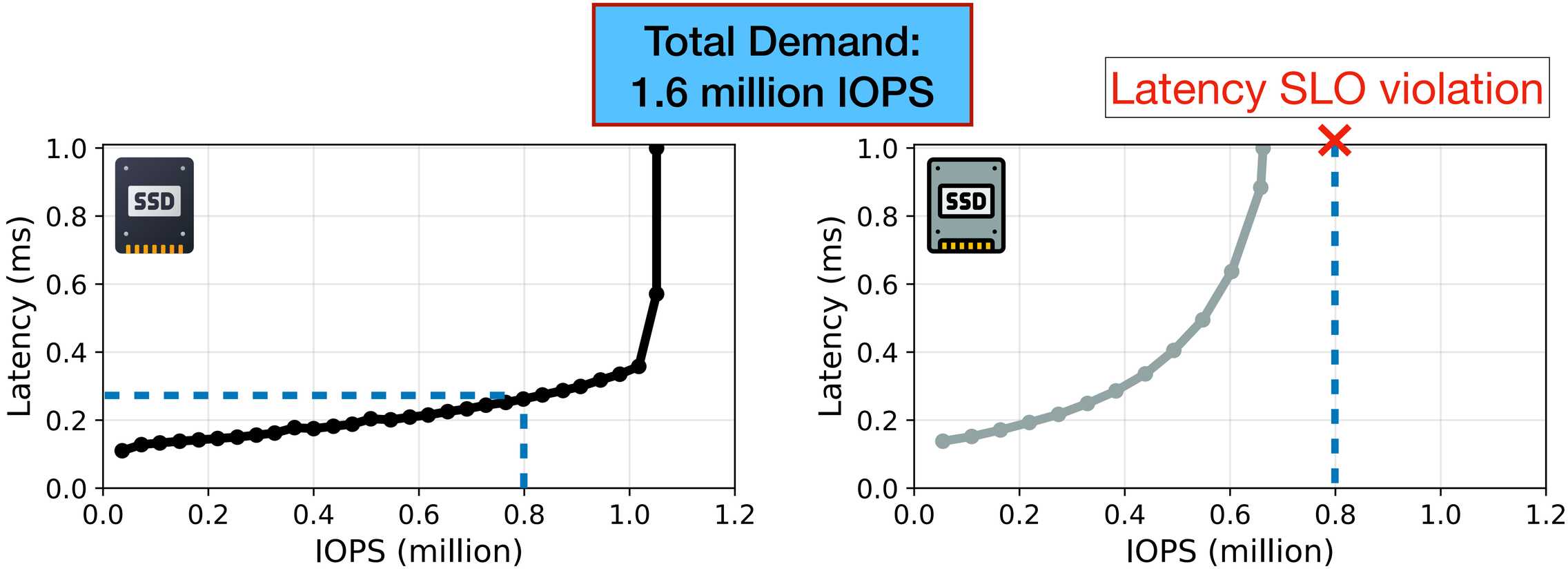
1. Profile-driven load steering

Total Demand:
1.6 million IOPS



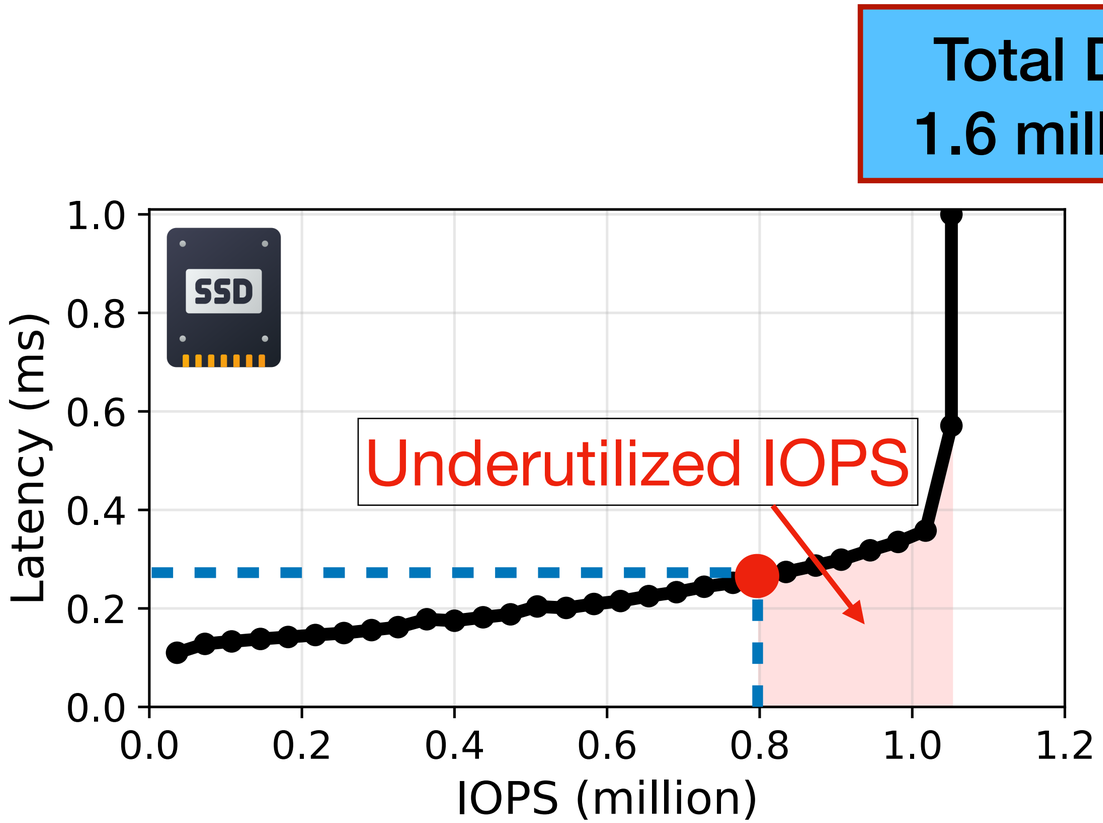


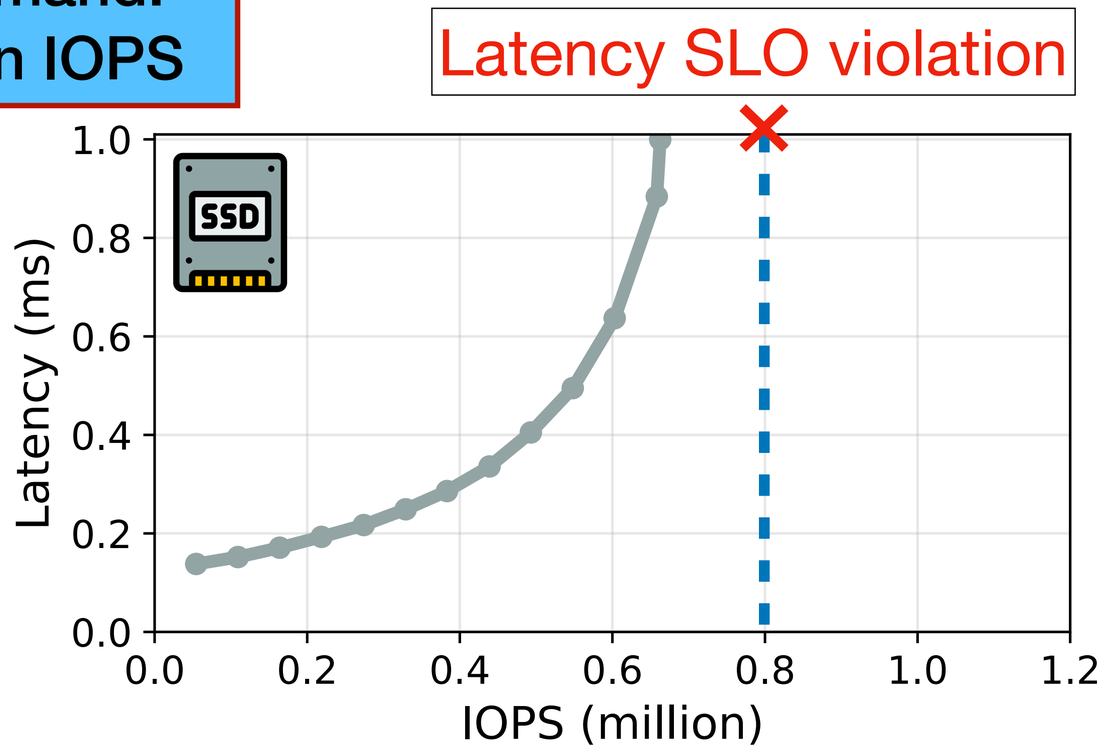
1. Profile-driven load steering



1. Profile-driven load steering

SLO violations (+ underutilization) with profile-agnostic routing

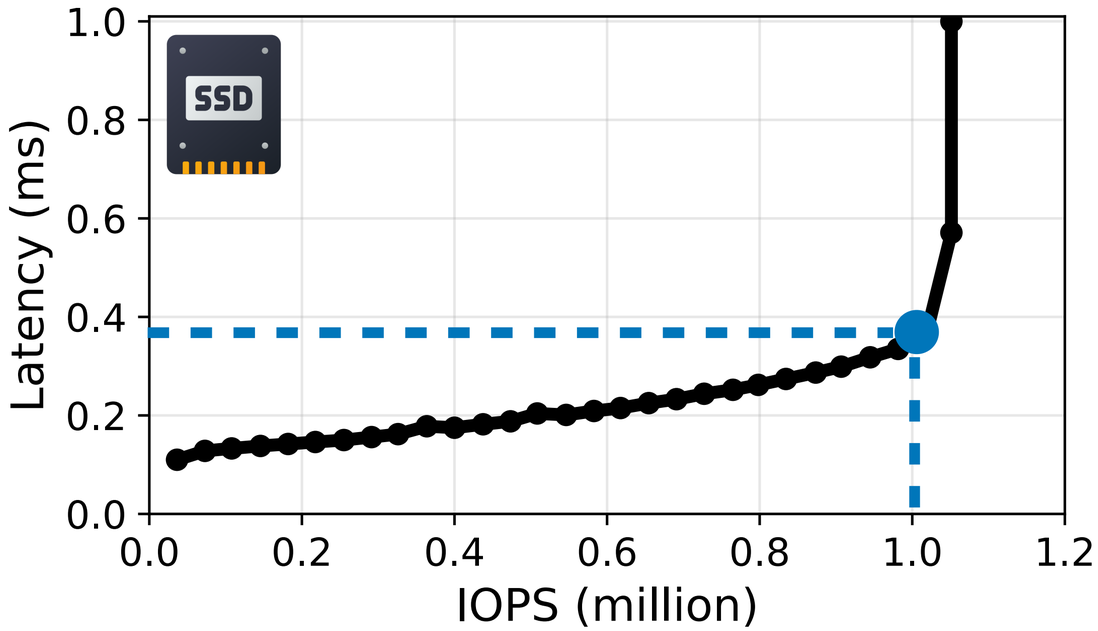


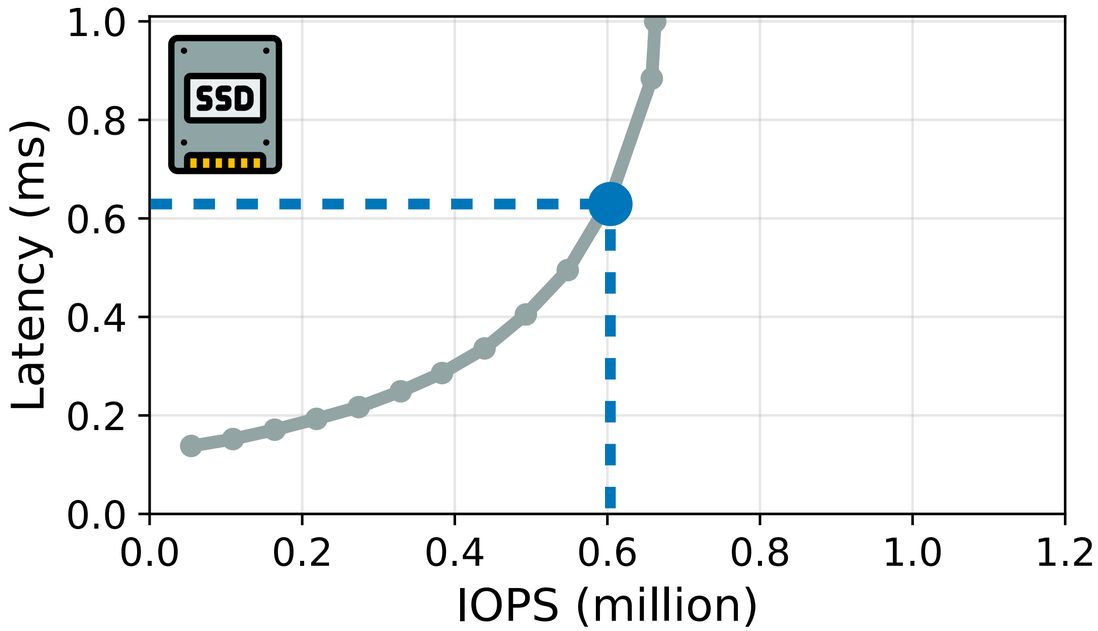


1. Profile-driven load steering

Tap into most of the available IOPS while meeting latency SLOs

Total Demand:
1.6 million IOPS





Sandook Controller

SSD Weights

SSD = *63%* SSD = *37%*

2. Read/write segregation

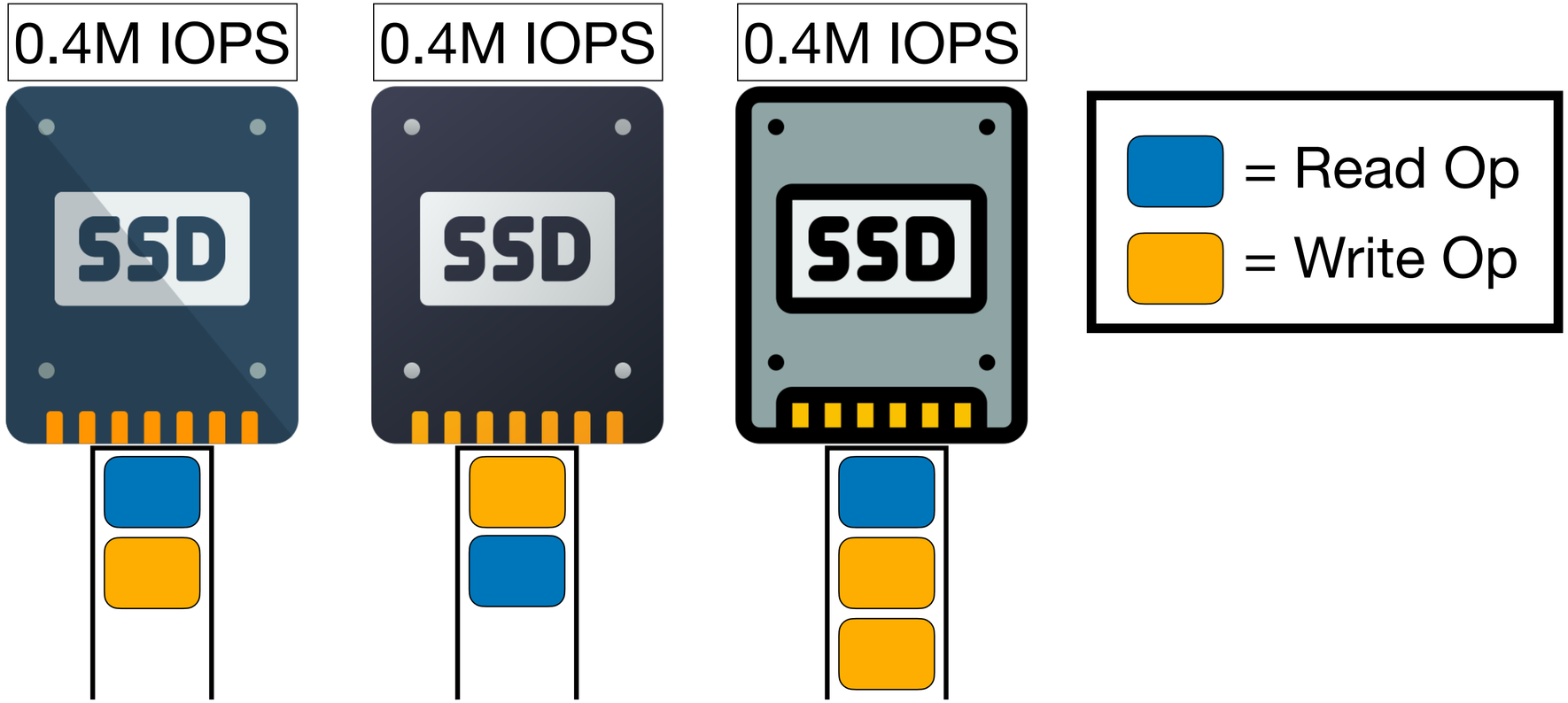
Physically isolating read and write operations

- Assign modes to SSDs: either **read-only-** or **write-**mode
  - Minimize mixing of reads with writes
  - Periodically determine the size of read/write sets based on global demand
  - Switch modes to spread load

2. Read/write segregation

Read/write interference leads to sub-optimal performance

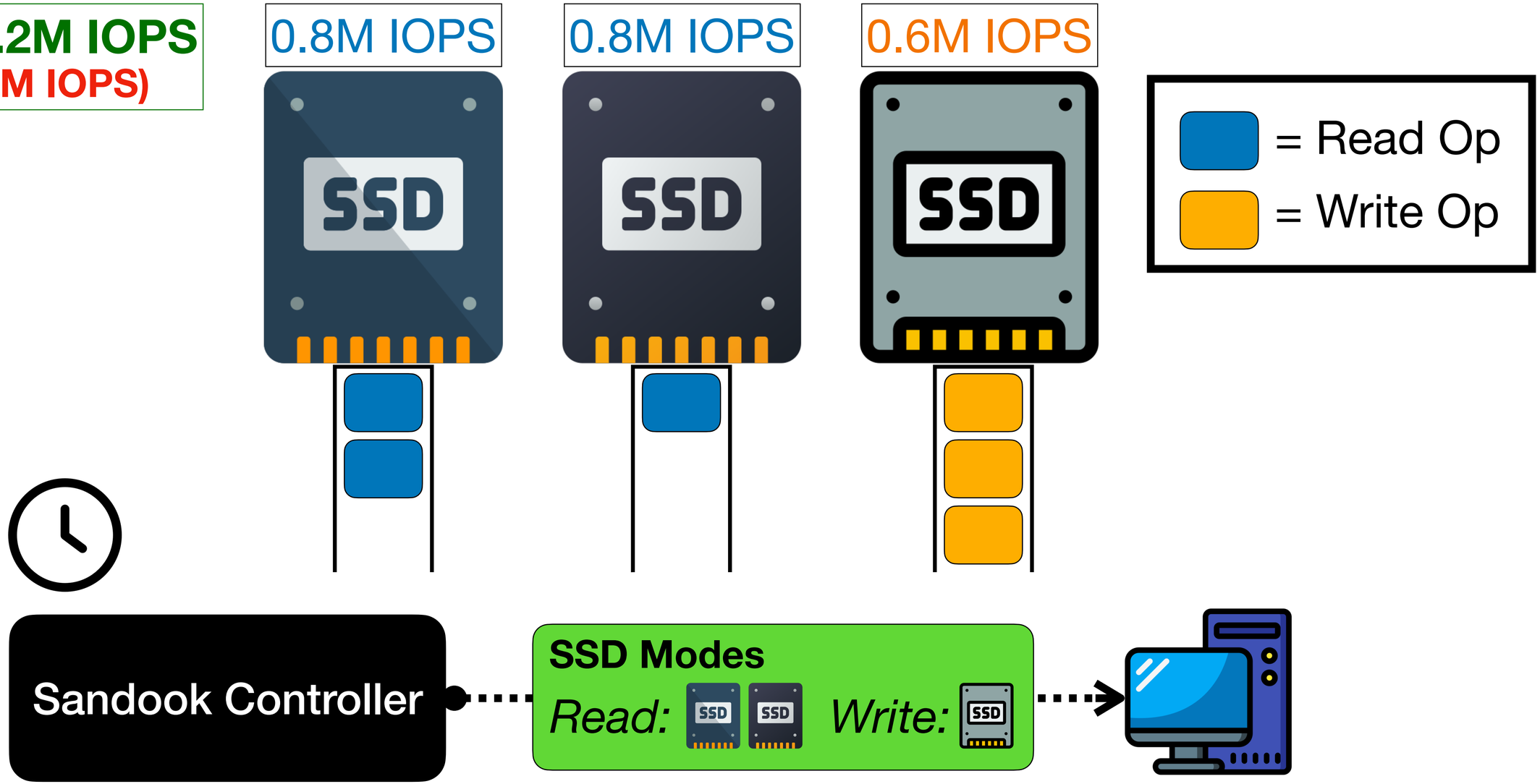
Total: 1.2M IOPS



2. Read/write segregation

Physically isolating read/write operations unlocks higher aggregate performance

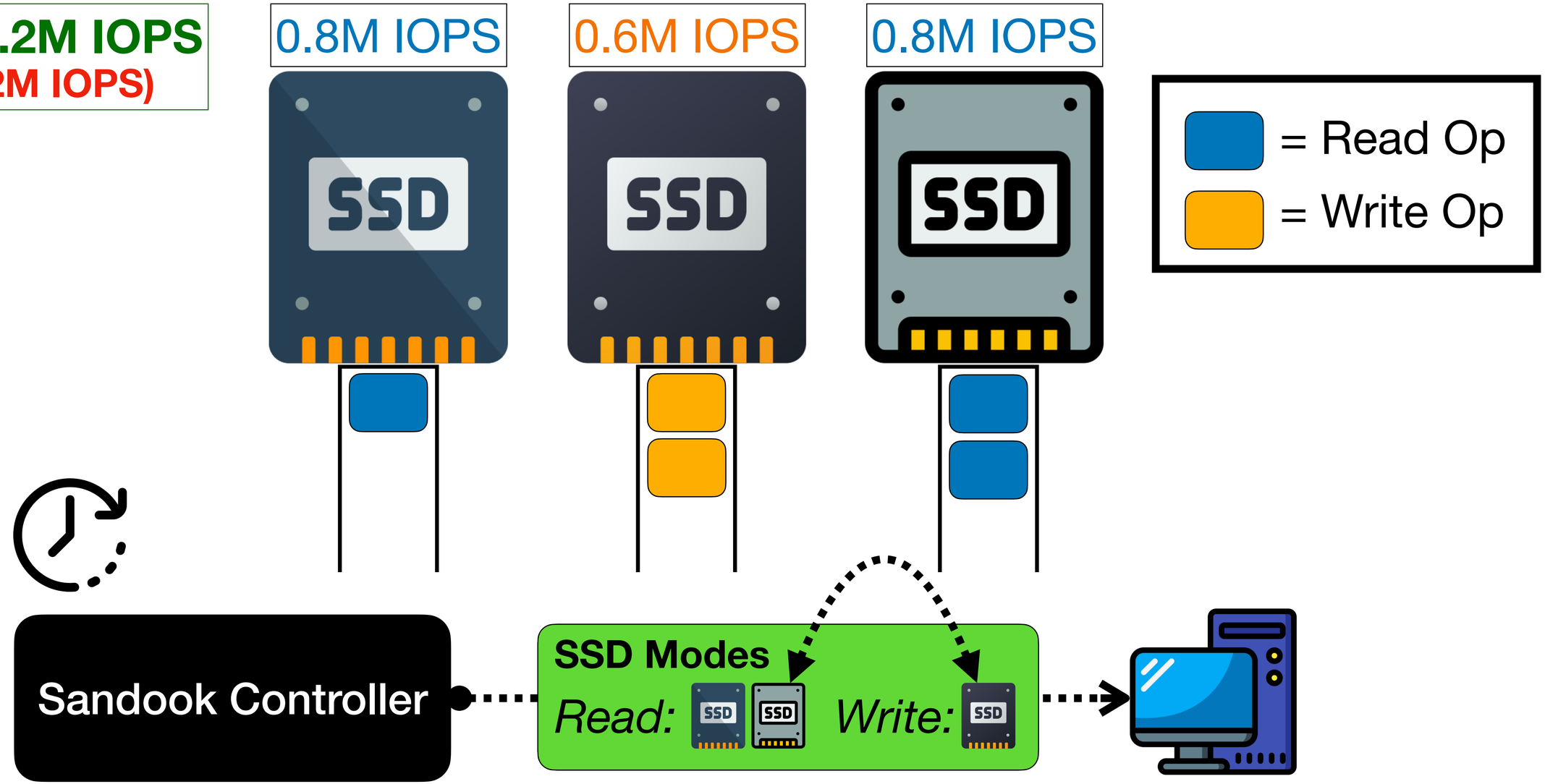
Total: 2.2M IOPS
(vs 1.2M IOPS)



2. Read/write segregation

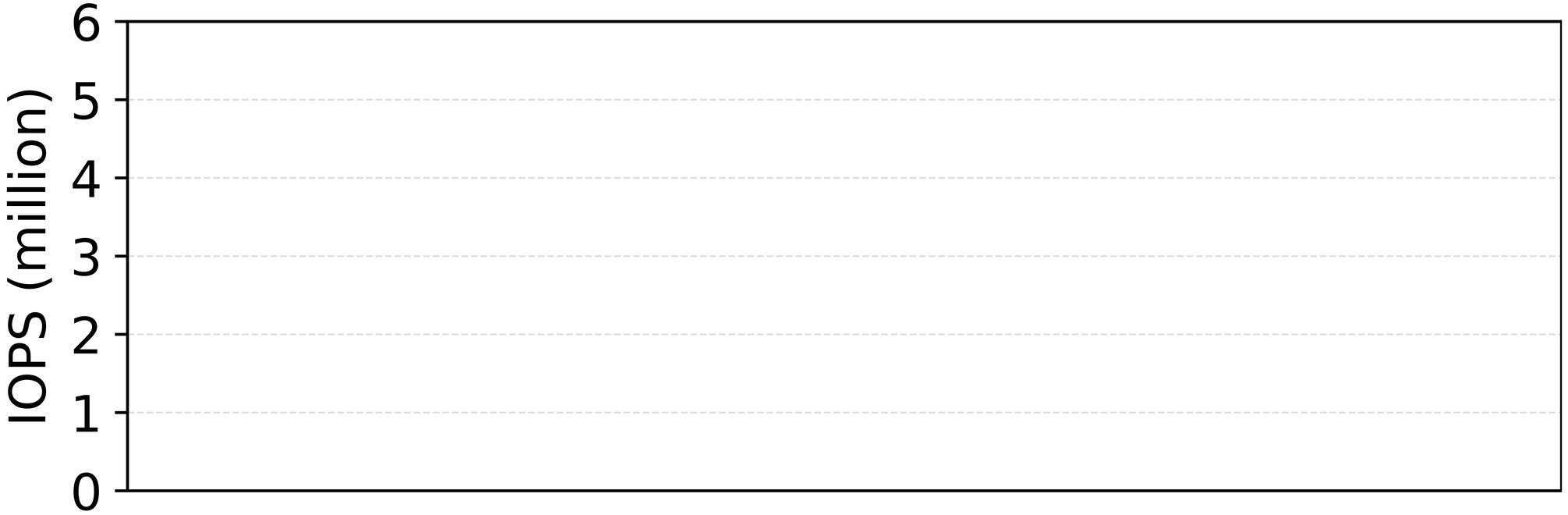
Switch modes periodically for spreading new blocks evenly

Total: 2.2M IOPS
(vs 1.2M IOPS)

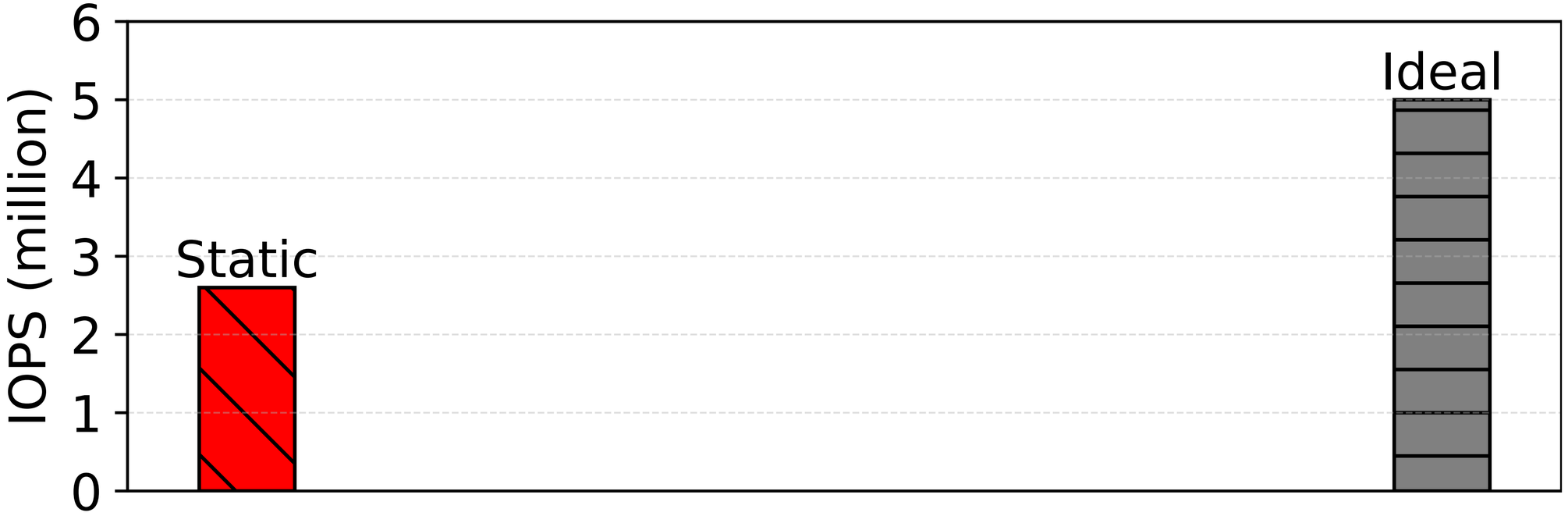


Zooming into the performance…

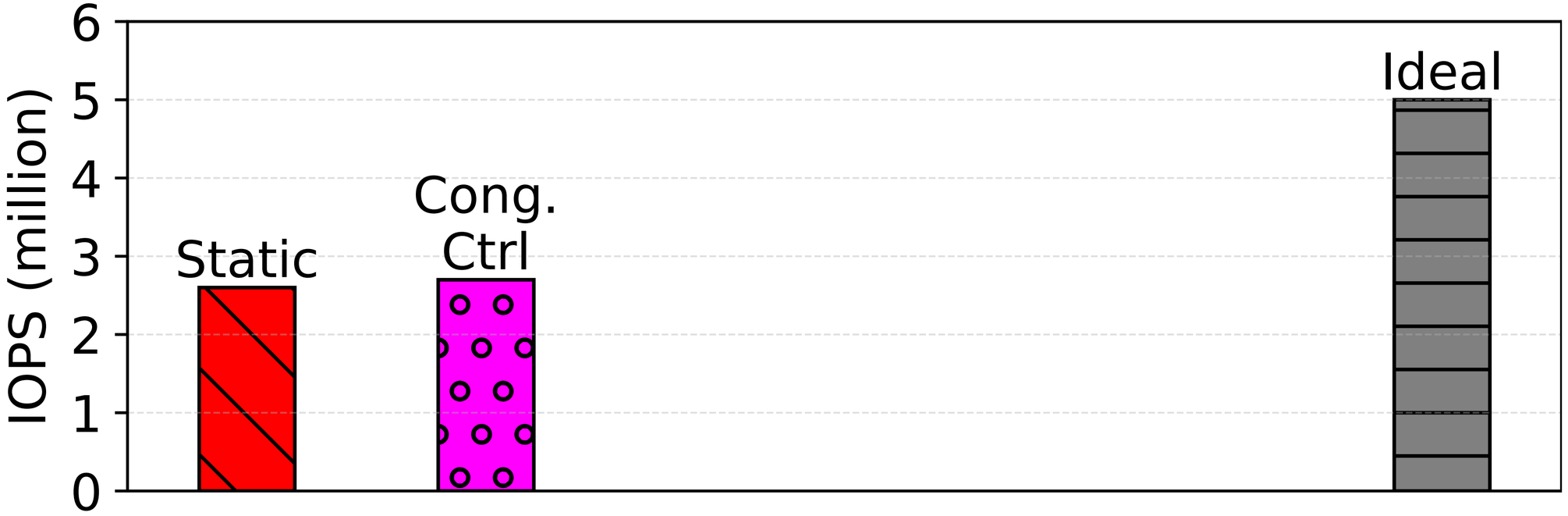
Multiplexing SSDs over mixed read/write workloads



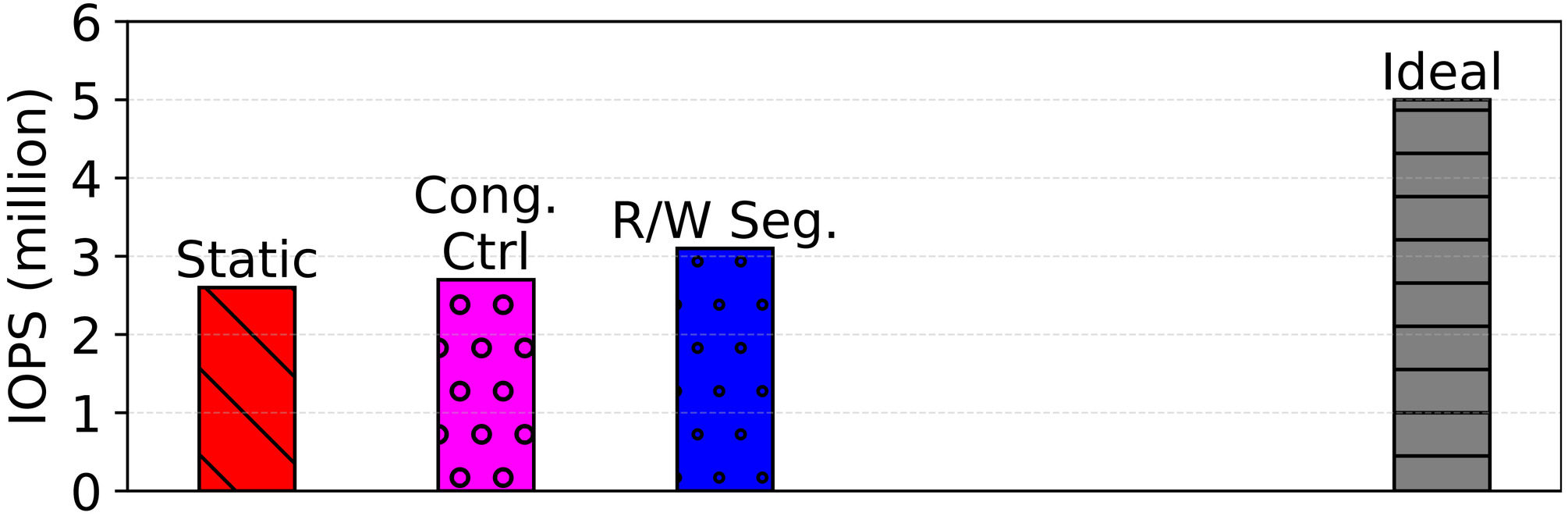
Multiplexing SSDs over mixed read/write workloads



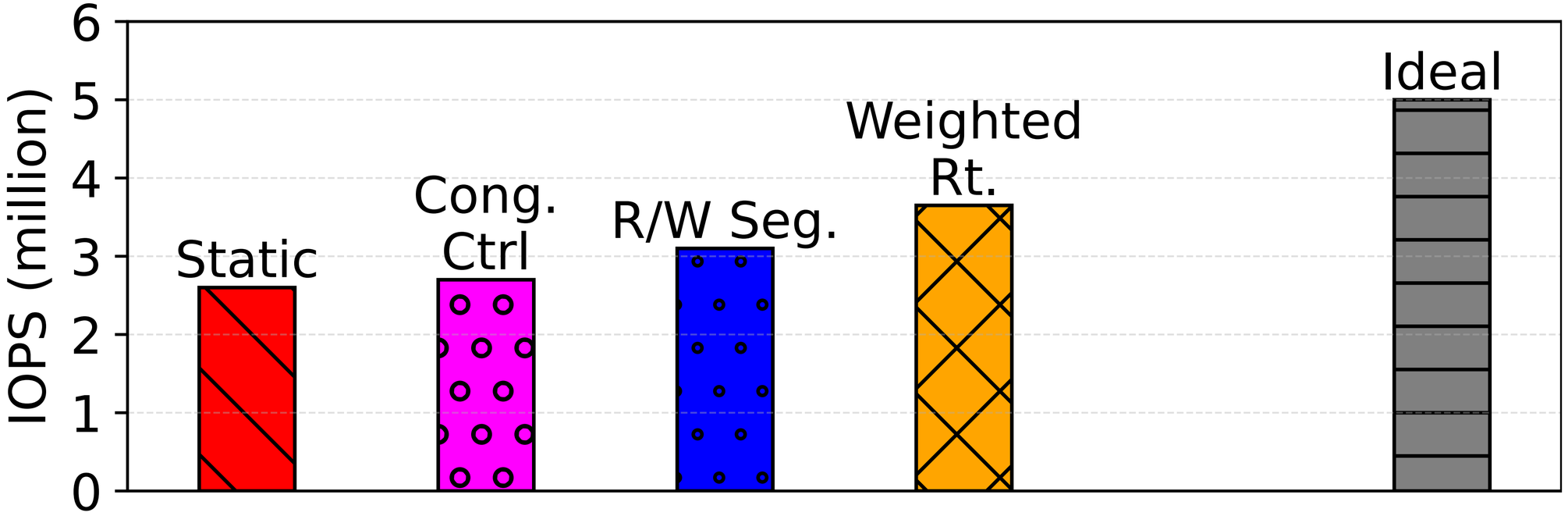
Multiplexing SSDs over mixed read/write workloads



Multiplexing SSDs over mixed read/write workloads

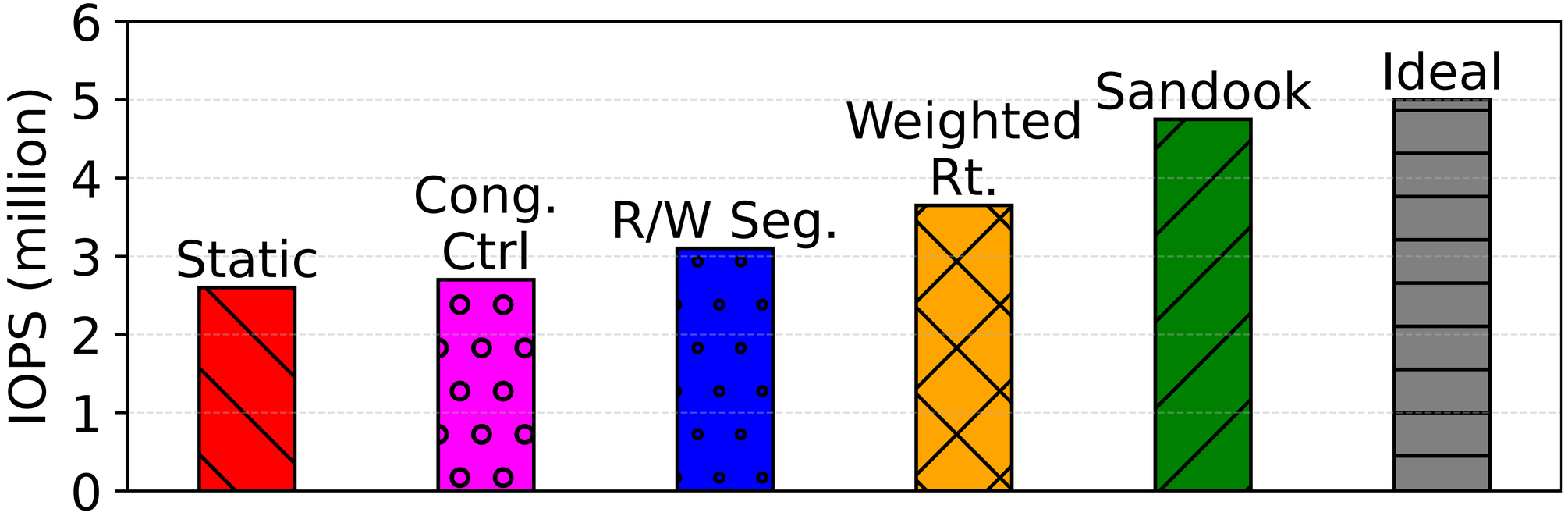


Multiplexing SSDs over mixed read/write workloads



Multiplexing SSDs over mixed read/write workloads

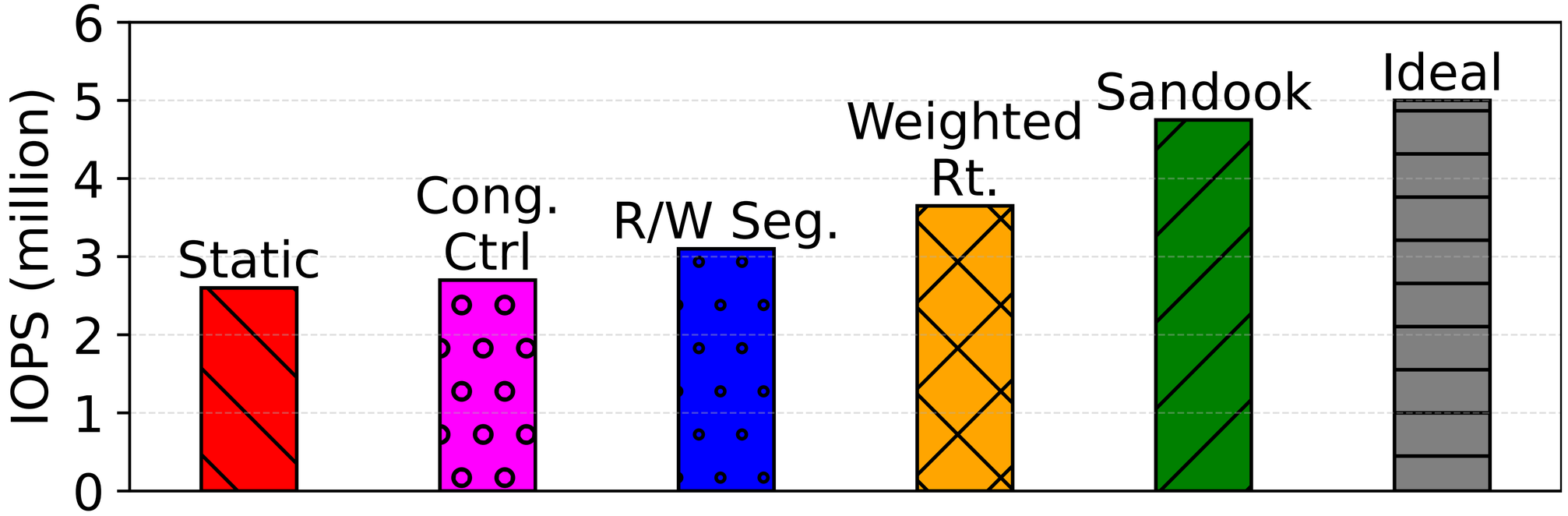
Sandook's holistic approach unlocks 95% of ideal throughput!



Multiplexing SSDs over mixed read/write workloads

Sandook's holistic approach unlocks 95% of ideal throughput!

Under the same cost budget ($), Sandook can deliver ~2x performance.



Conclusion

Sandook can unlock high performance + high utilization of a shared pool of SSDs

Conclusion

Sandook can unlock high performance + high utilization of a shared pool of SSDs

- Manages various sources of performance variability in datacenter SSDs
  - Routing flexibility + timescale separation

Conclusion

Sandook can unlock high performance + high utilization of a shared pool of SSDs

- Manages various sources of performance variability in datacenter SSDs
  - Routing flexibility + timescale separation
- 1.7x raw storage throughput, 1.9x application throughput, 88% lower latency

Conclusion

Sandook can unlock high performance + high utilization of a shared pool of SSDs

- Manages various sources of performance variability in datacenter SSDs
  - Routing flexibility + timescale separation
- 1.7x raw storage throughput, 1.9x application throughput, 88% lower latency
- Linux compatible without requiring any application code-changes

Conclusion

Sandook can unlock high performance + high utilization of a shared pool of SSDs

- Manages various sources of performance variability in datacenter SSDs
  - Routing flexibility + timescale separation
- 1.7x raw storage throughput, 1.9x application throughput, 88% lower latency
- Linux compatible without requiring any application code-changes
- Available as open-source software at **https://bit.ly/mit-sandook**