

Mitigate HDD Fail-Slow by Pro-actively Utilizing System-level Data Redundancy with Enhanced HDD Controllability and Observability

Jingpeng Hao, Yin Li, Xubin Chen, Tong Zhang

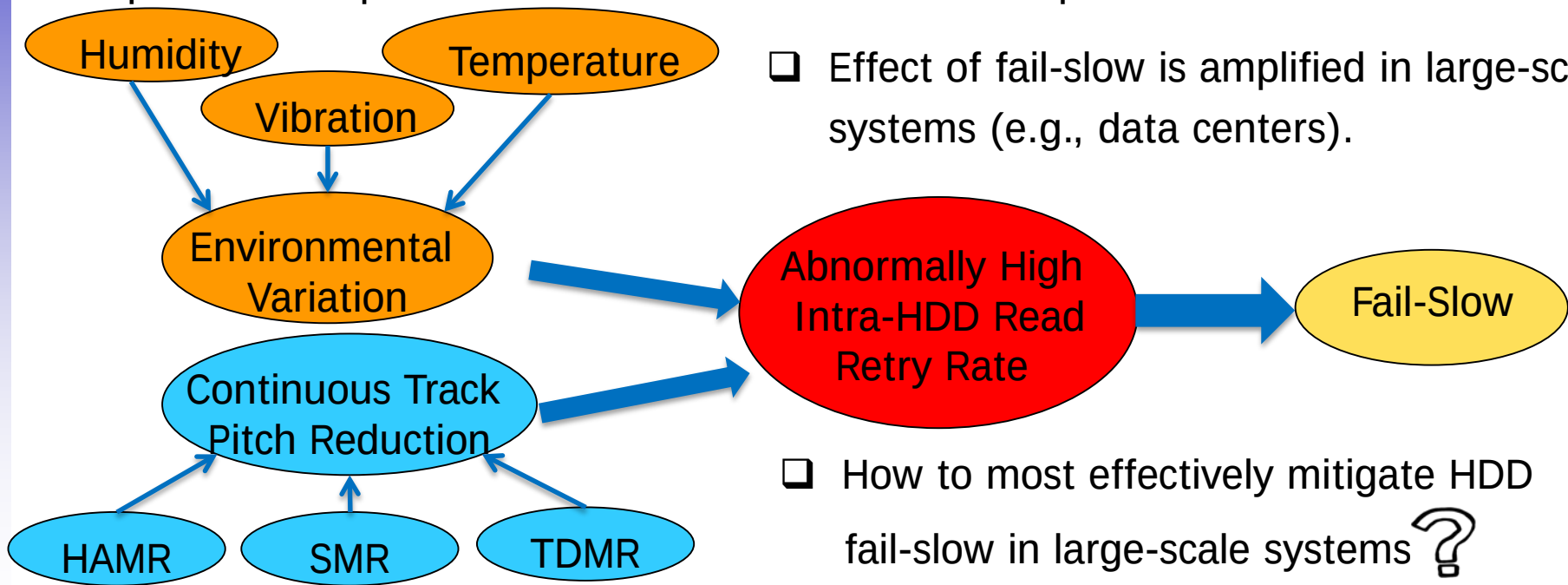
Electrical, Computer and Systems Engineering Department

Rensselaer Polytechnic Institute



HDD Fail-Slow

- ❑ The well-documented “fail-slow at the scale” problem: HDDs can occasionally operate at a speed much slower than their normal specs.

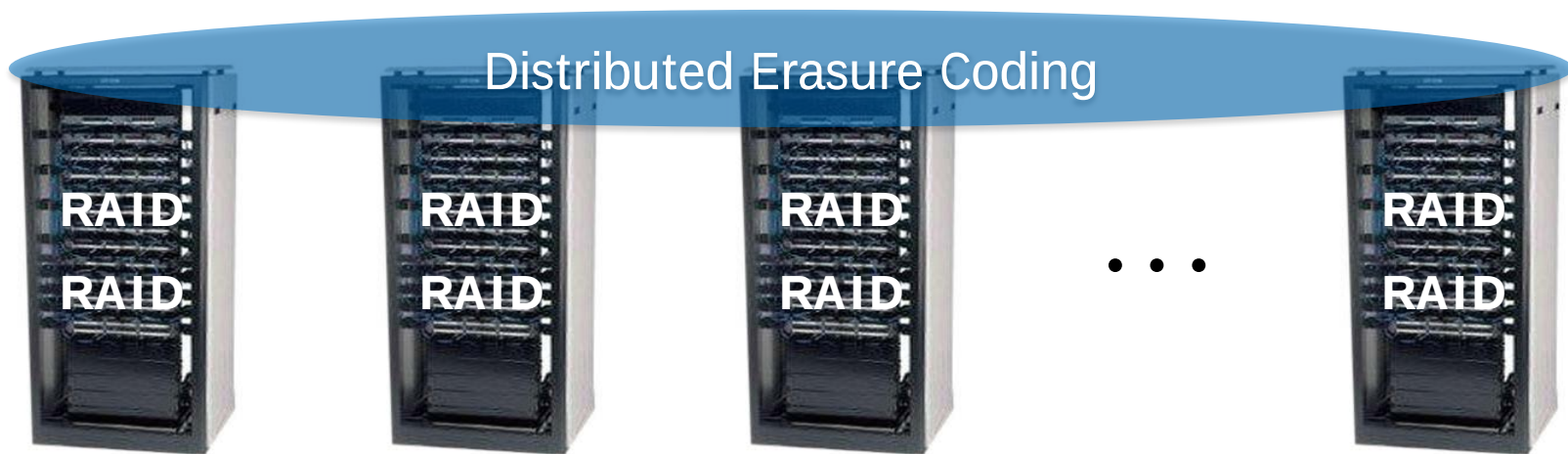


HDD Read Retry

- ❑ In case of sector read failure, repeat reading this sector with additional disk rotations until success (long delay) or time-out (data loss)

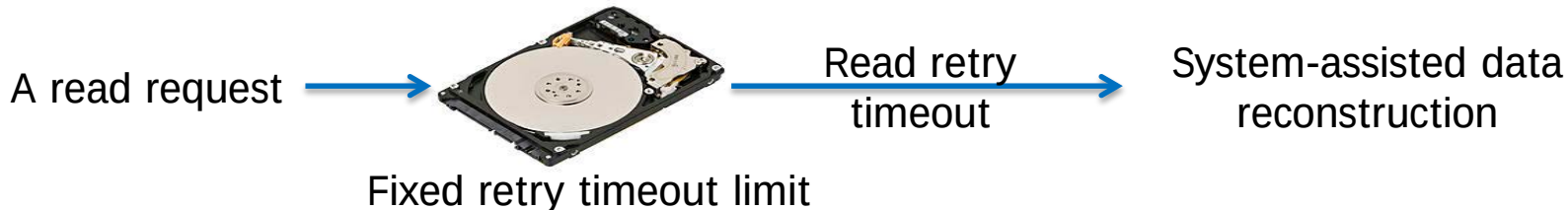


Abundant system-level data redundancy in large-scale systems

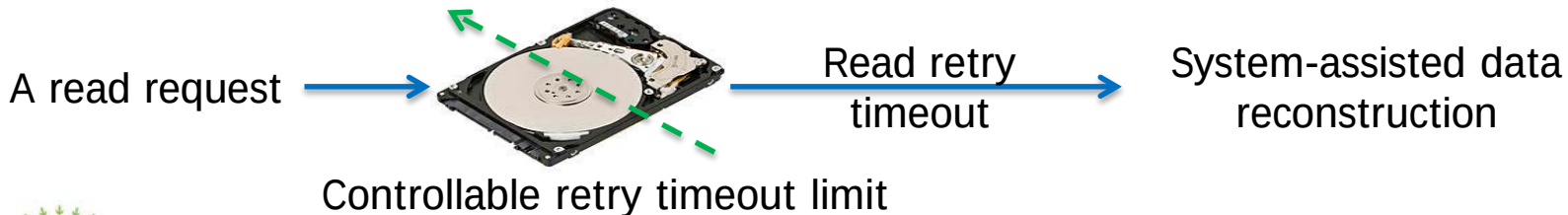


Mitigate HDD Fail-Slow

- ❑ Complement HDD read retry with system-assisted data reconstruction



- ❑ Enhance the controllability of HDDs in terms of read retry



OCP (open compute project) proposal: fail-fast read of data center HDDs

➡ Per-request controllable read retry timeout limit

Mitigate HDD Fail-Slow

- ❑ Enhance the controllability of HDDs in terms of read retry



...



Controllable retry timeout limit

Intra-HDD retry



- x Longer per-HDD read latency
- ✓ Less cross-HDD read traffic



System-assisted data reconstruction



- ✓ Shorter per-HDD read latency
- x More cross-HDD read traffic

Pro-active Design Approach

1. Normal mode: solely rely on intra-HDD read retry

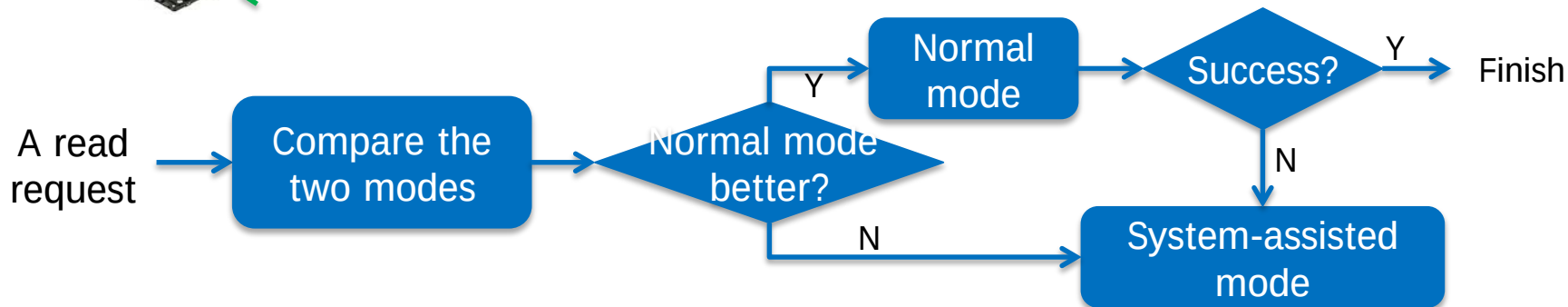


Fixed retry timeout limit

2. System-assisted mode: leverage system-assisted data reconstruction by reducing retry timeout limit or even eliminating retry



Controllable retry timeout limit



Pro-active Design Approach

- ❑ To maximize practical feasibility, we assume
 - ✓ The simplest host-side HDD controllability: host can only turn-on/off HDD read retry on the per-request basis
 - ✓ The simplest host-side HDD observability: host can only inquiry HDDs regarding read retry statistics via S.M.A.R.T. commands
- ❑ Use RAID as the test vehicle

? How to most effectively implement the system-assisted mode?

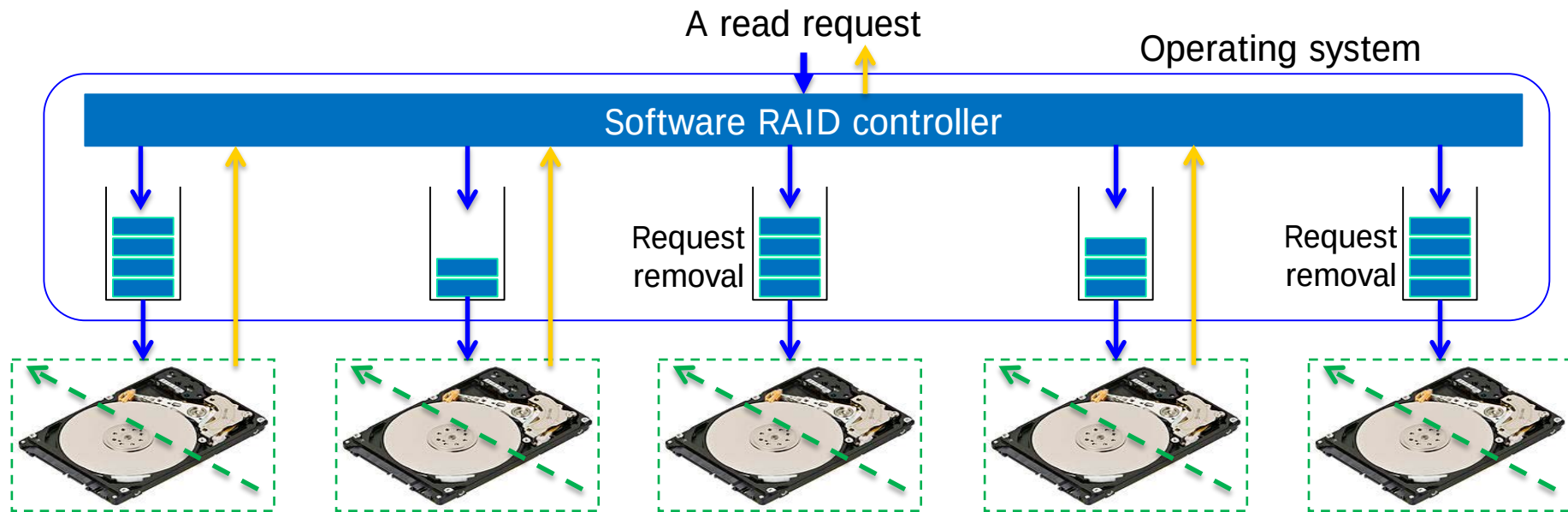
? How to improve the sector failure tolerance of the system-assisted mode?

? For each read request, how to decide which mode we should choose?

Pro-active Design Approach

? How to most effectively implement the system-assisted mode?

➡ Runtime variation among HDDs (e.g., sector failure rate, queue depth)



Pro-active Design Approach

? How to improve the sector failure tolerance of the system-assisted mode?

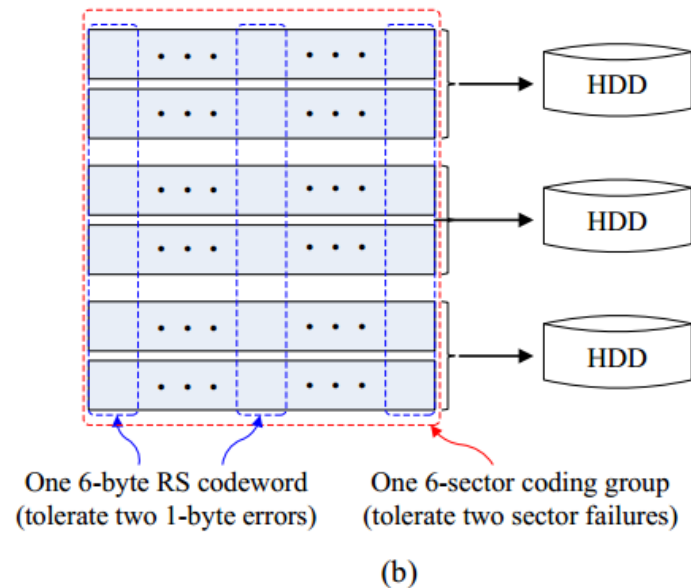
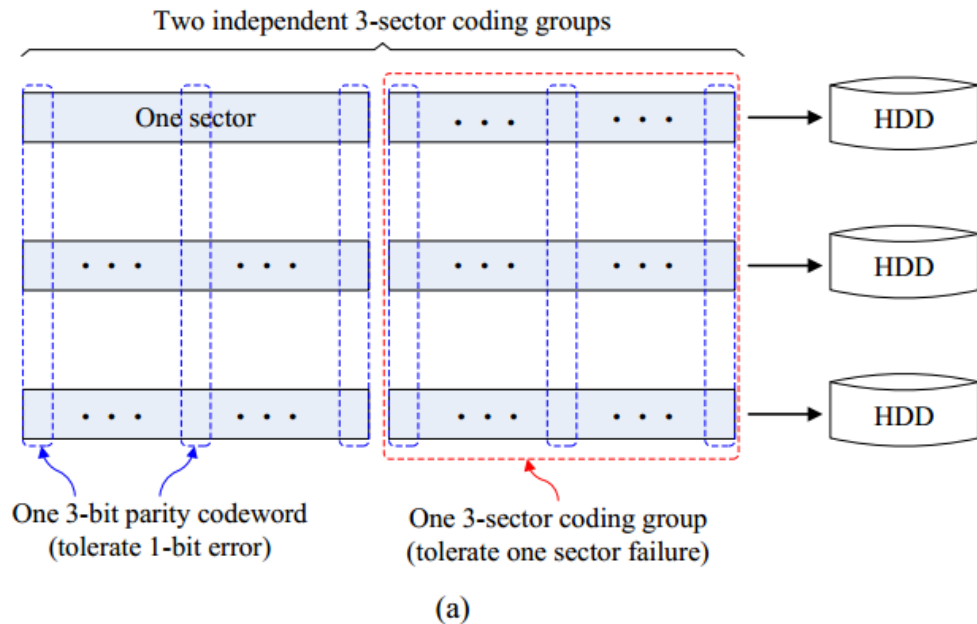
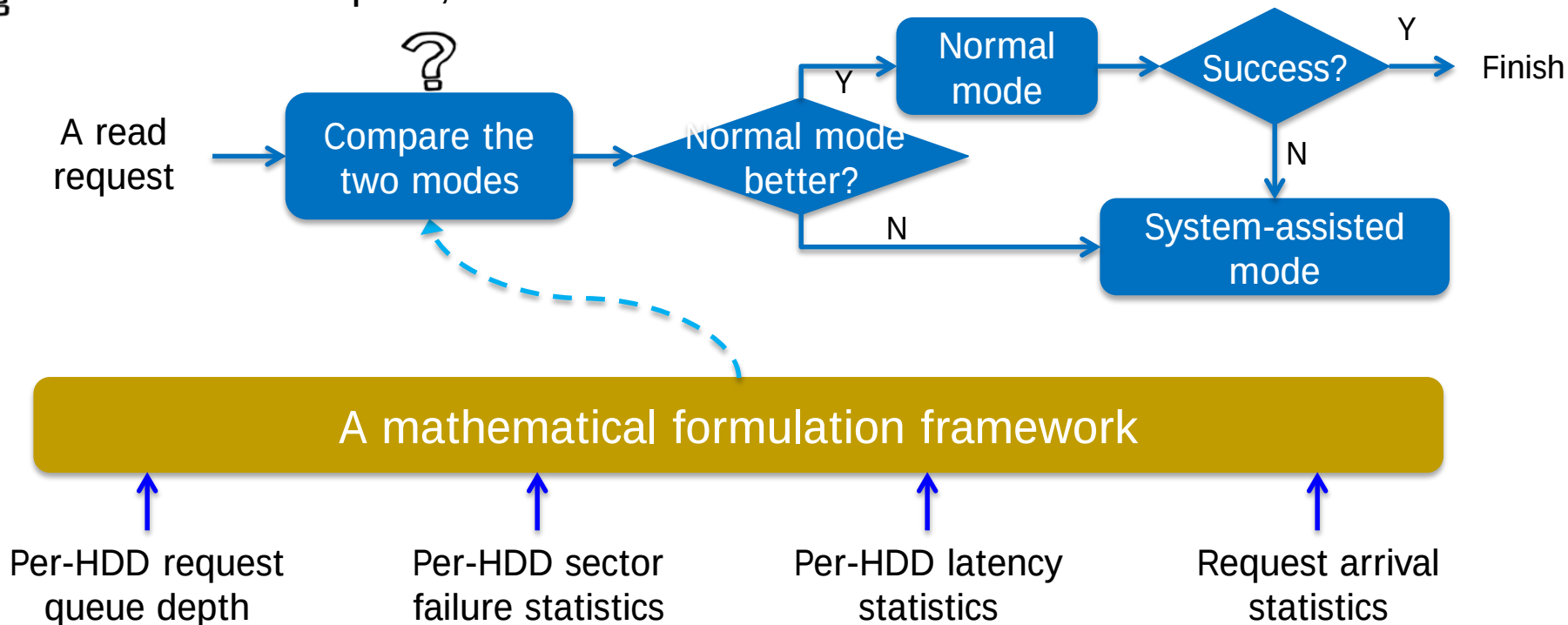


Illustration of (a) conventional RAID and (b) proposed eRAID on 3 HDDs with $m = 2$ and $k = 1$.

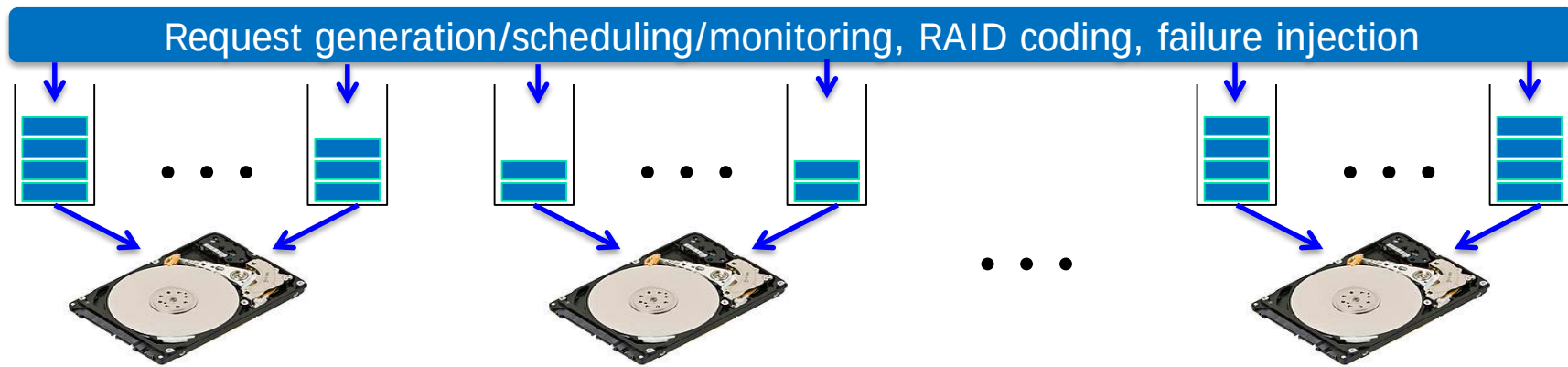
Pro-active Design Approach

? For each read request, how to decide which mode we should start with?



Pro-active Design Approach

- ❑ An experimental platform to facilitate the research



- ❑ To emulate intra-HDD read retry
 - ✓ Increase the read request size to force additional disk rotations
 - ✓ For example, assume 1.2MB per track → convert a 4kB read request to a 3.6MB read request to mimic the read retry with 3 disk rotations

Experiments

- ❑ A server with dual-socket Intel Xeon E5-2630 2.2GHz CPUs (10 cores per socket) and 64GB DRAM
- ❑ Six 2TB 7200rpm SATA HDDs form a RAID-5 with the stripe size of 8kB
- ❑ Total 192 user-space threads to concurrently dispatch read requests to all the six HDDs
- ❑ Assume 3 rotations or 5 rotations per read retry

Experiments

❑ Impact of HDD fail-slow on the average and tail read latency

Average read
latency

Rotations per retry	Retry rate	Read request size		
		8kB	24kB	40kB
3	0	16ms	41ms	107ms
	1%	18ms	48ms	221ms
	2%	19ms	64ms	269ms
5	1%	18ms	56ms	284ms
	2%	22ms	90ms	553ms

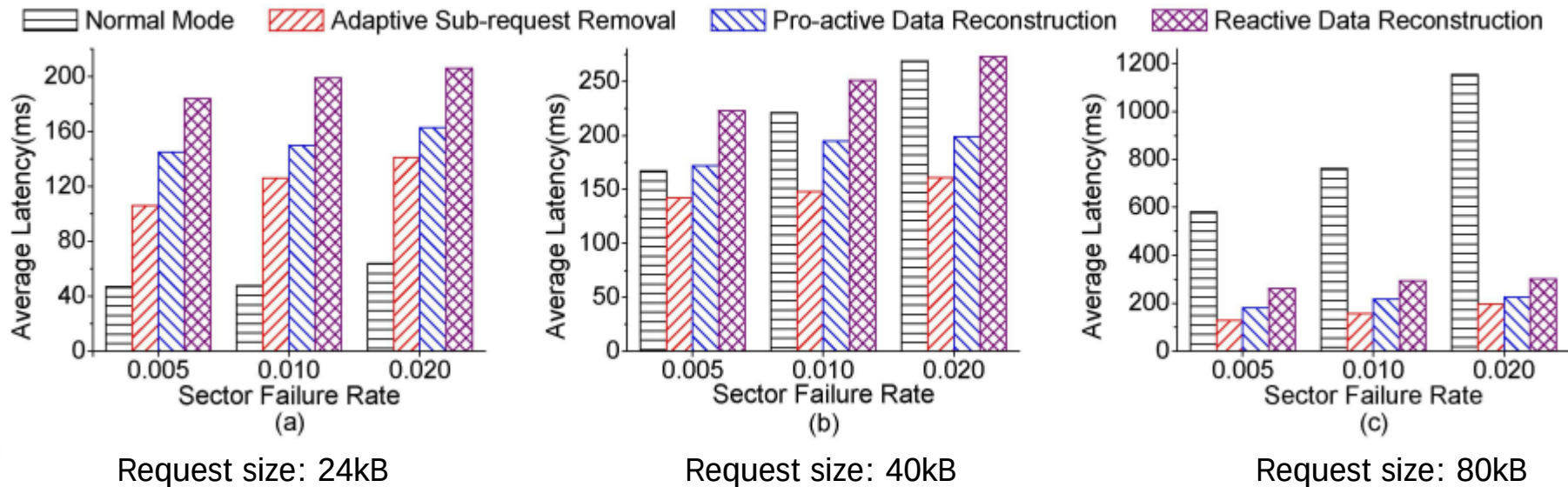
99% tail read
latency

Rotations per retry	Retry rate	Read request size		
		8kB	24kB	40kB
3	0	43ms	169ms	832ms
	1%	63ms	236ms	1,712ms
	2%	68ms	512ms	2,190ms
5	1%	81ms	243ms	2,513ms
	2%	98ms	530ms	3,336ms

Experiments

❑ Implementation of system-assisted mode

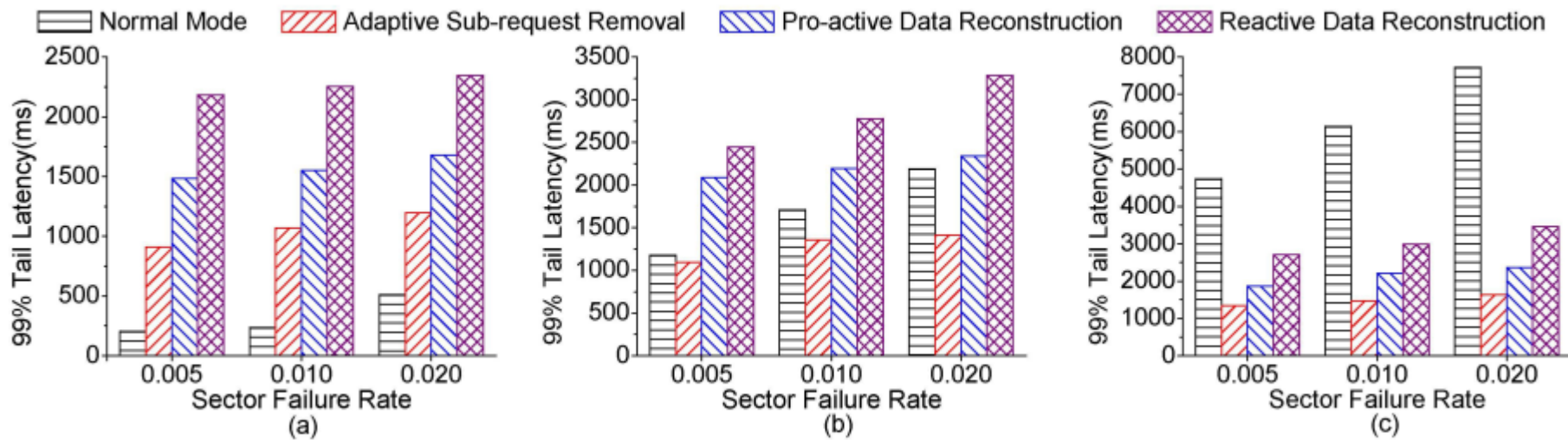
1. Proposed: Pro-active data reconstruction w. adaptive request removal
2. Pro-active data reconstruction (without adaptive request removal)
3. Reactive data reconstruction (without adaptive request removal)



Experiments

❑ Implementation of system-assisted mode

1. Proposed: Pro-active data reconstruction w. adaptive request removal
2. Pro-active data reconstruction (without adaptive request removal)
3. Reactive data reconstruction (without adaptive request removal)



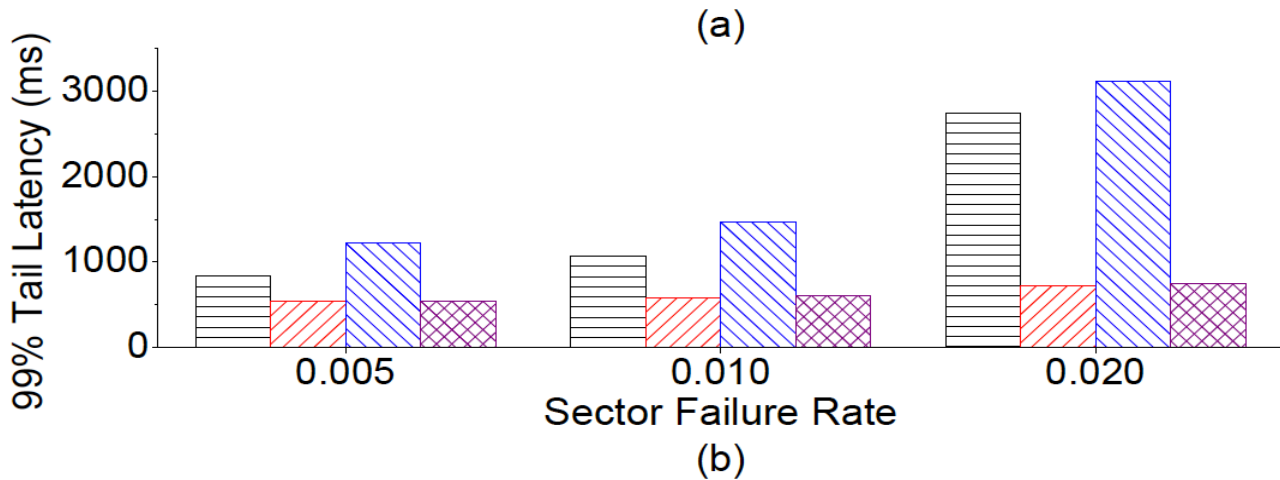
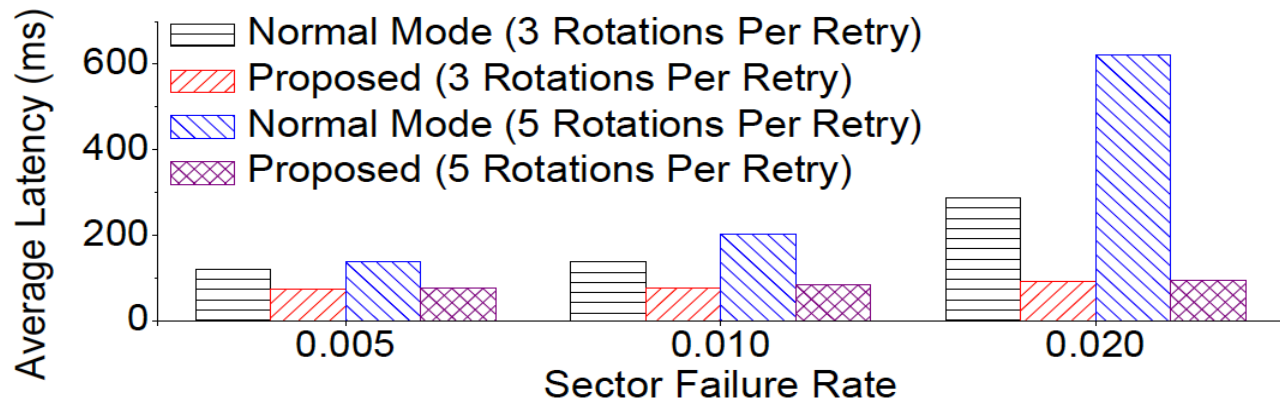
Request size: 24kB

Request size: 40kB

Request size: 80kB

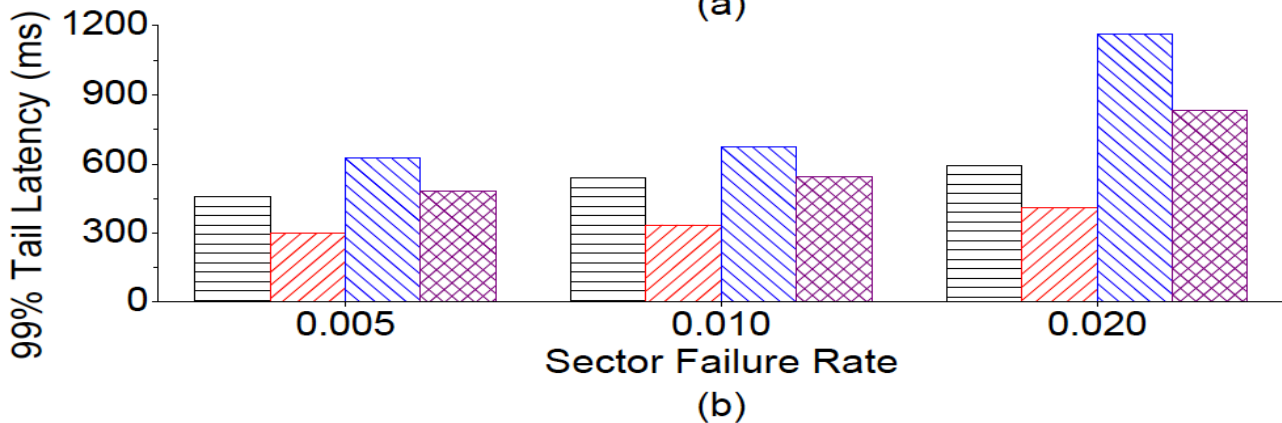
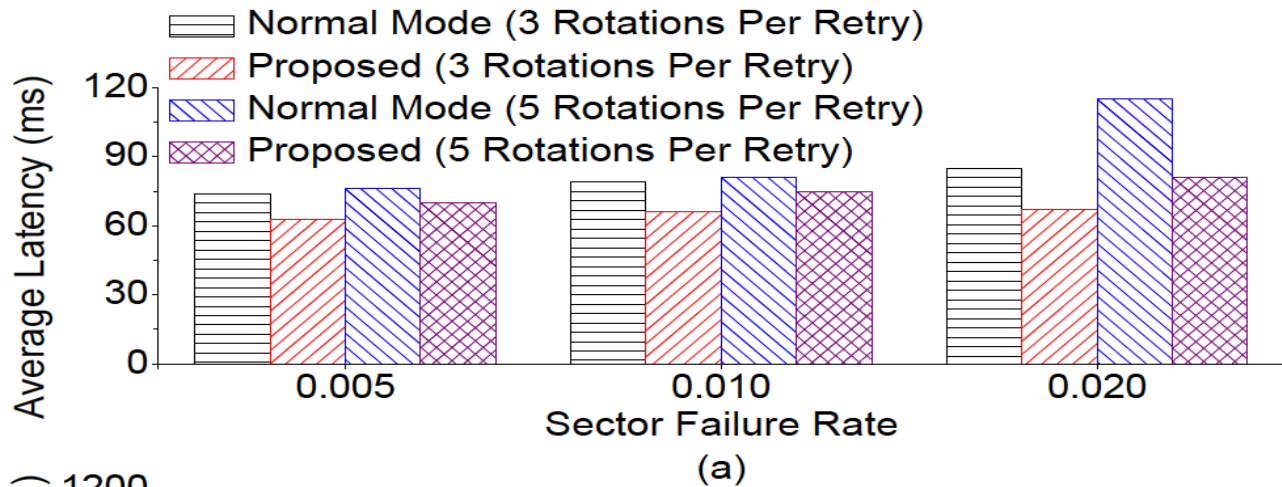
Experiments

- Read-only workloads with read request size 8kB~80kB
- Mean of request arrival time: 8ms
- All the HDDs** are subject to the same sector failure rate



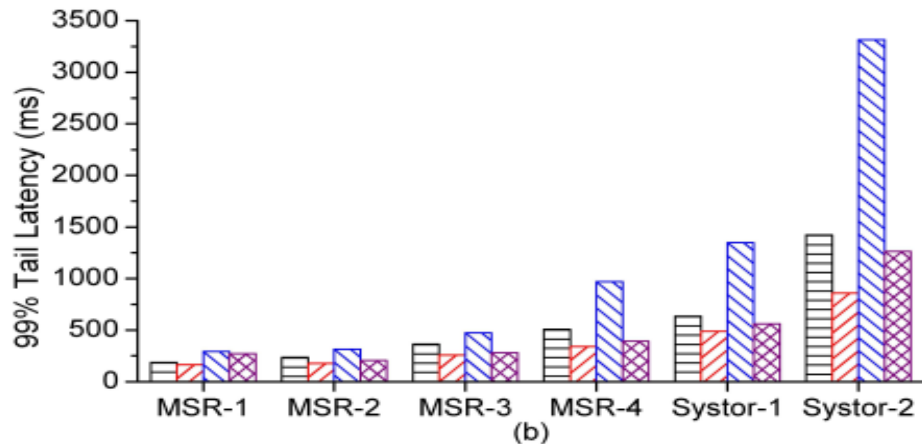
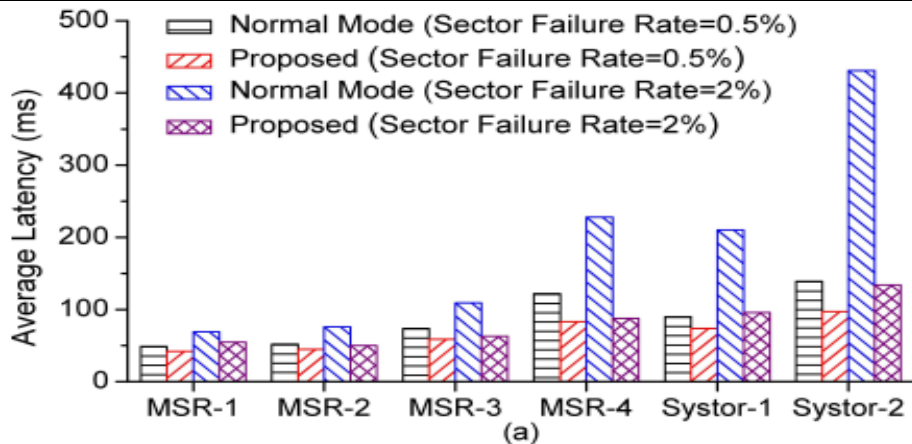
Experiments

- ❑ Read-only workloads with read request size 8kB~80kB
- ❑ Mean of request arrival time: 8ms
- ❑ **Only one HDD** is subject to the high sector failure rate



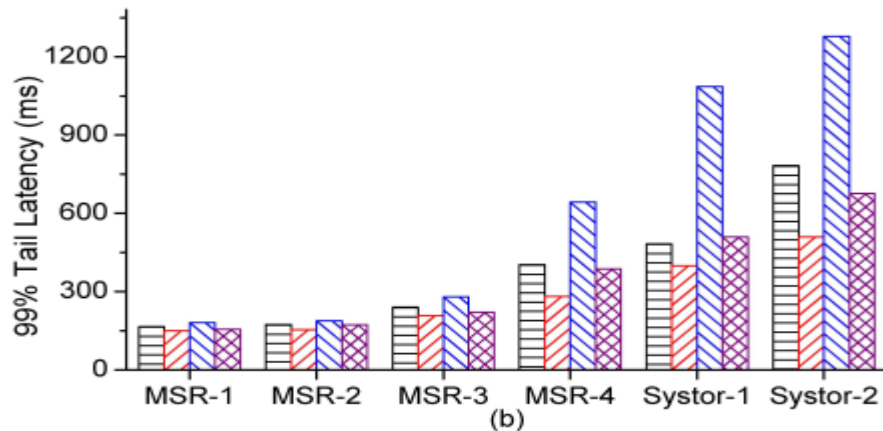
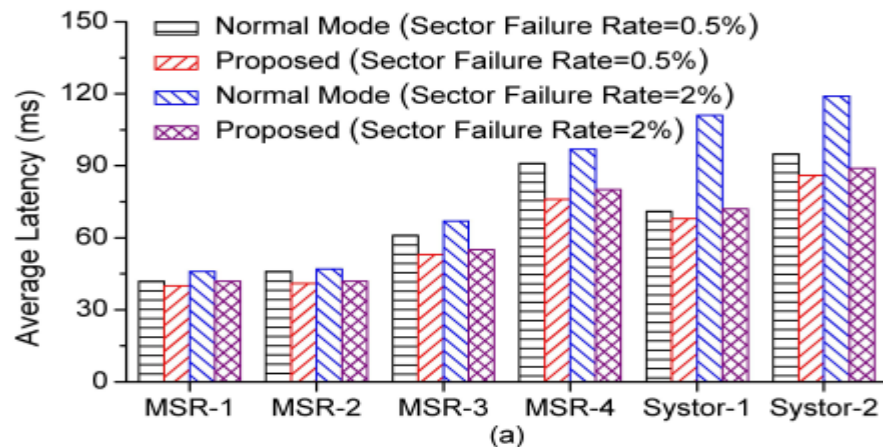
Experiments

- Measured average and 99-percentile read latency under six different traces.
- All the HDDs** are subject to the high sector failure rate.



Experiments

- Measured average and 99-percentile read latency under six different traces.
- Only one HDD** is subject to the high sector failure rate.



Conclusion and Future Work

Conclusion:

- A strategy that can most effectively implement the system-assisted mode.
- A design technique to enhance existing redundancy coding schemes.
- A mathematical framework to quantitatively formulate the impact of the *system-assisted mode* on the overall system read latency performance.
- Experiments in the context of a RAID-5 system consisting of six 2TB 7200rpm SATA HDDs.

Future Work:

- Integration with SMR HDDs (in particular host-managed SMR HDDs).