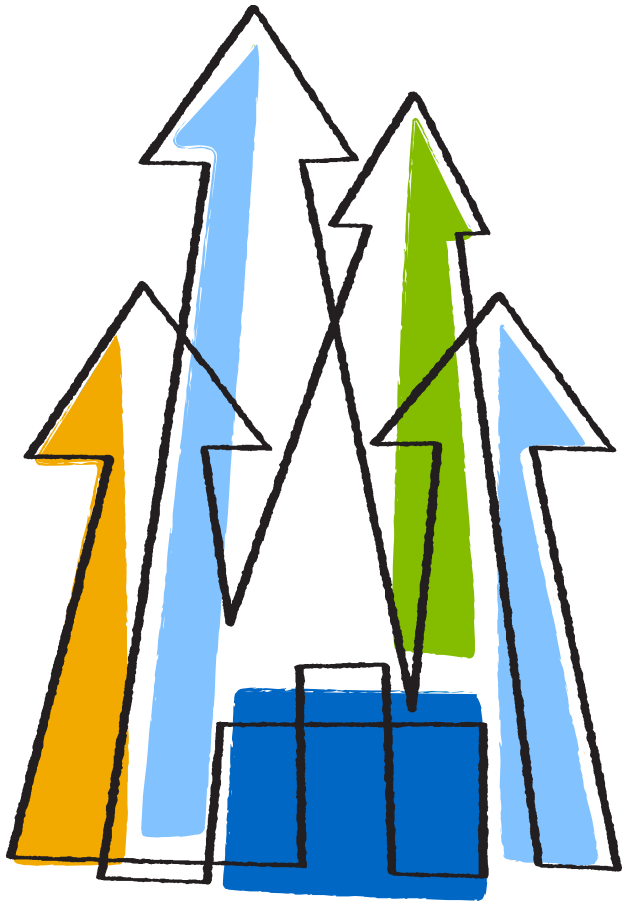
NetApp

Go further, faster®

SLO-aware Hybrid Store



Priya Sehgal, Kaladhar Voruganti,
Rajesh Sundaram

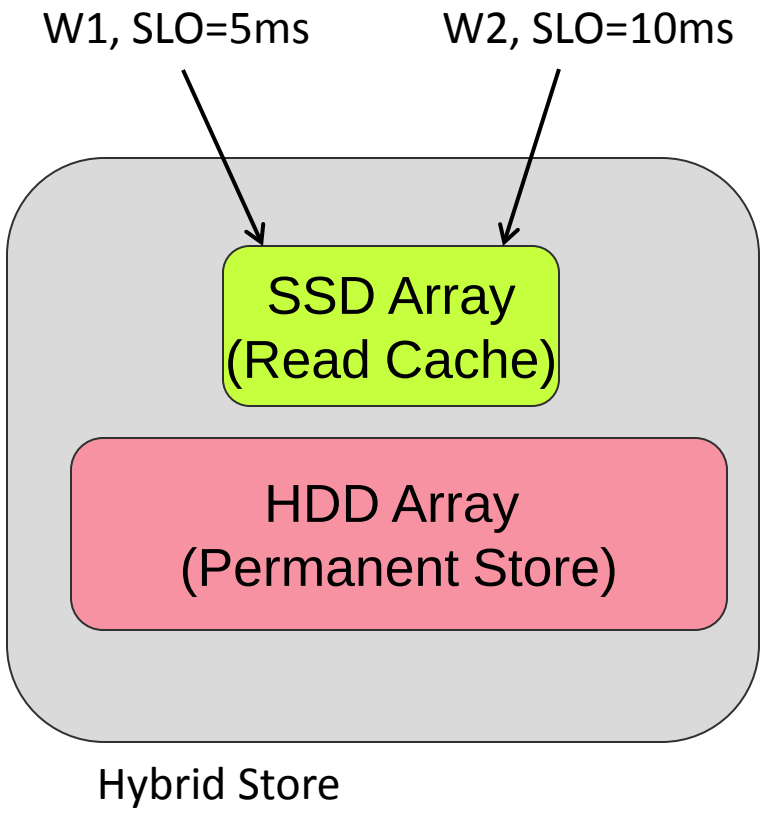
20th April 2012

Introduction

- What is SLO?
  - Service Level Objective
  - Specification of application requirements
  - Technology-independent

- Examples
  - Performance : Average I/O latency, throughput
  - Capacity
  - Reliability
  - Security, etc.

Problem and Motivation



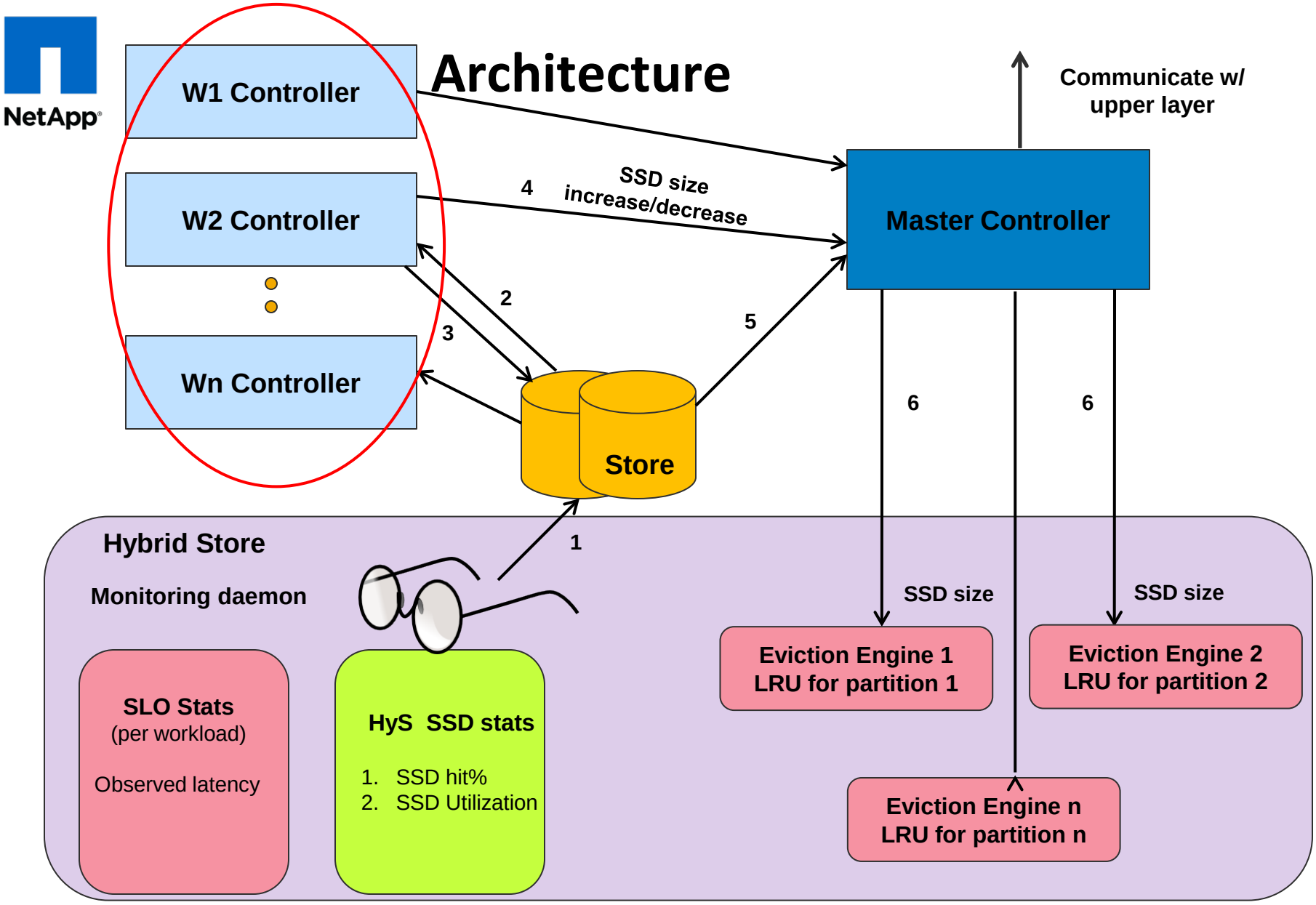
- Assumptions:
  - W1 and W2 working on different data sets
  - Assign SLO to workloads
  - SLO = Latency (in ms)
  - SSD tier: read cache only
    - **no** write cache
- Problems:
  - SLO inversion
  - SLO violation
  - Sub-optimal SSD Utilization

Need: Bring SLO-awareness to SSD caching (read) in HyS

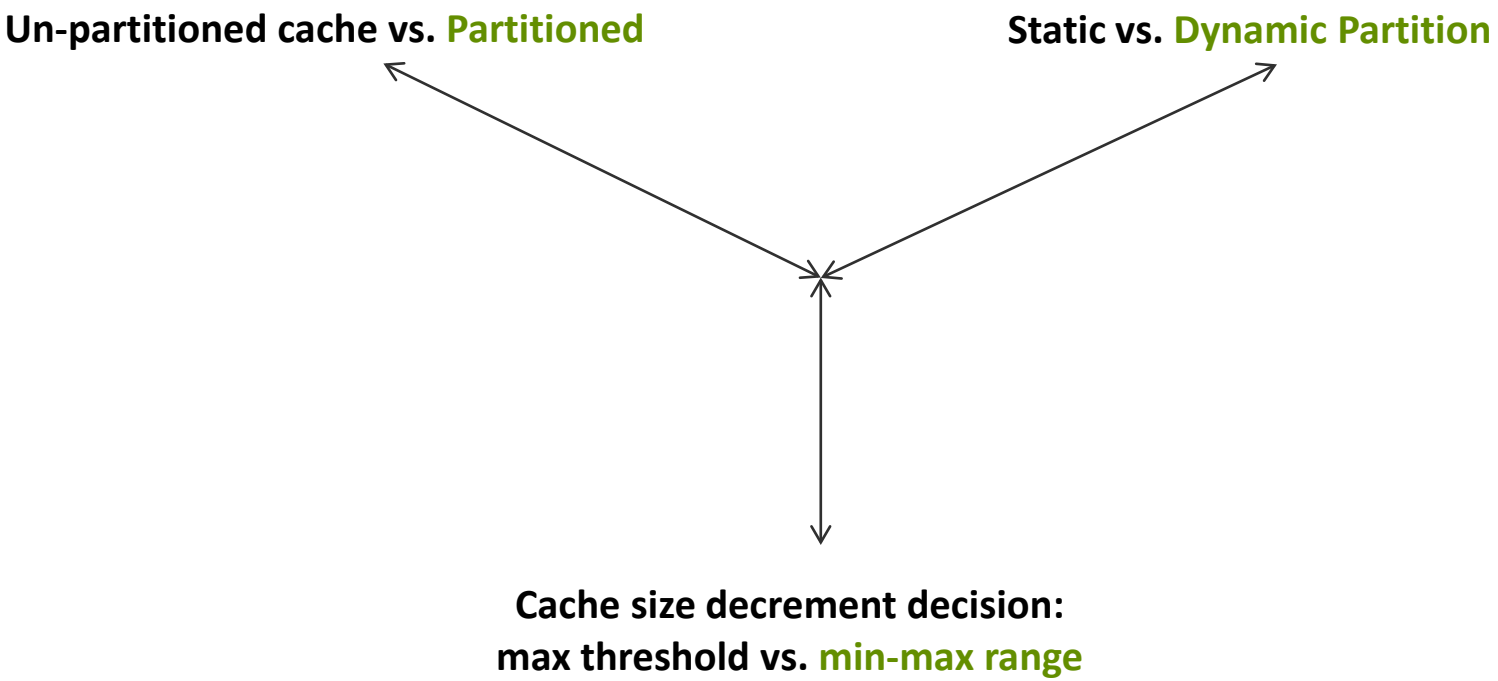
Solution – Trailer

Per-workload cache partitioning and dynamic cache sizing

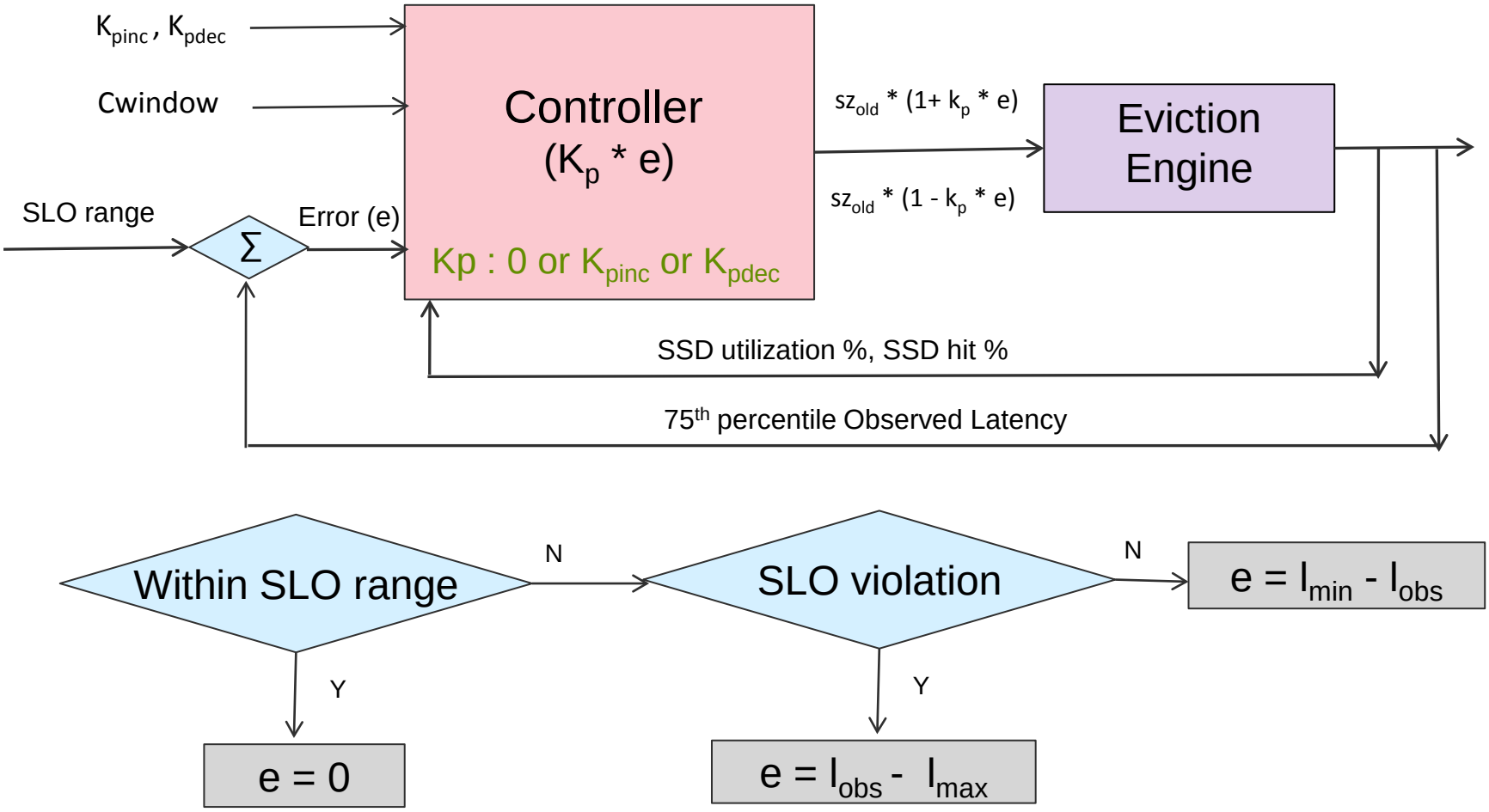
Architecture

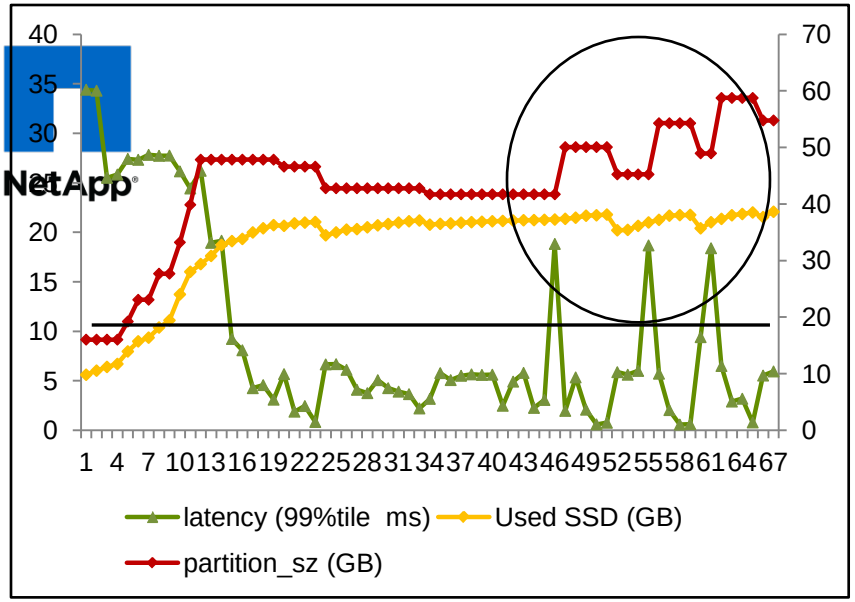


Controller Design Space Dimensions

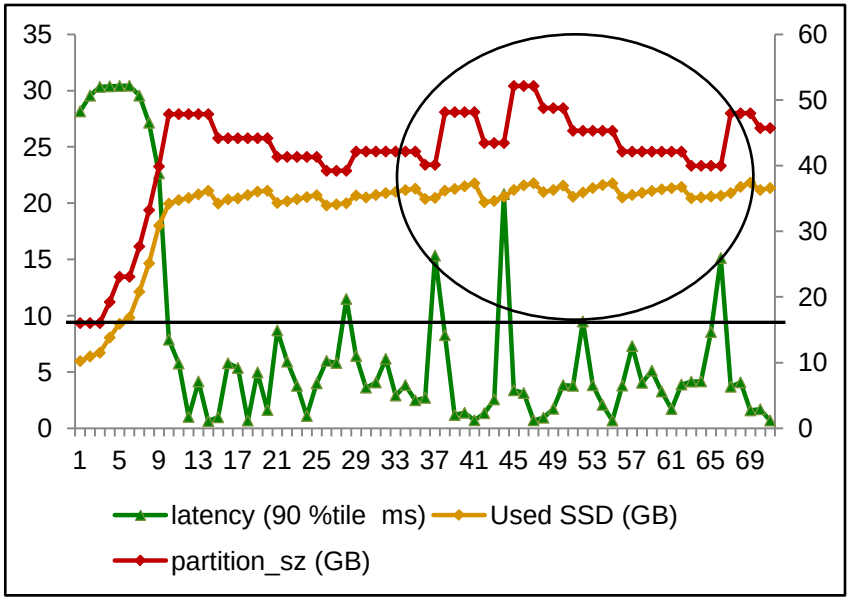


Error-aware Feedback Controller (EAFC)

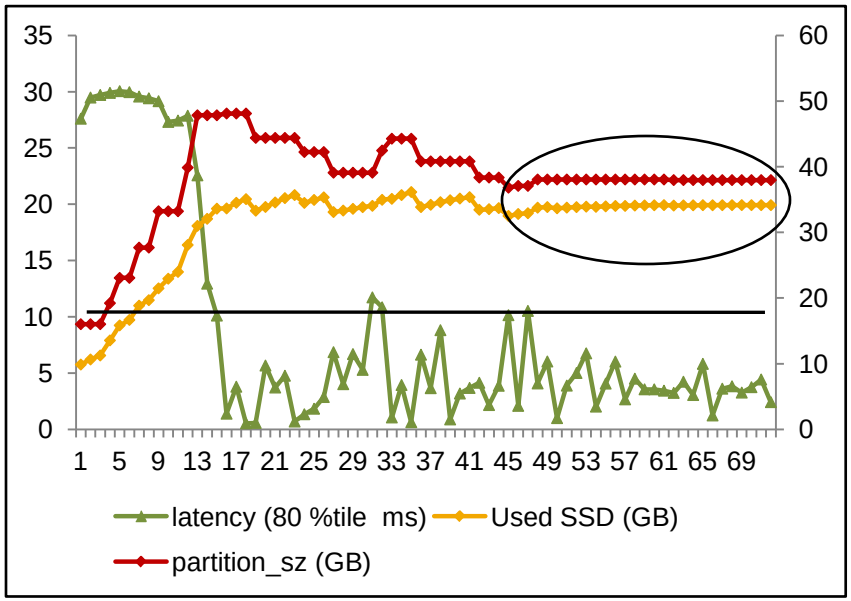




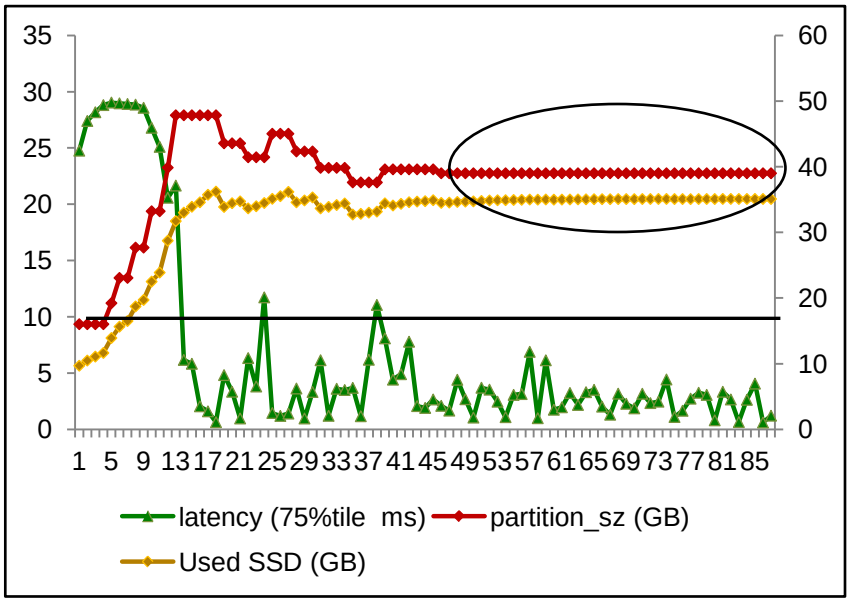
a. SLO Target = 99 percentile



b. SLO Target = 90 percentile



c. SLO Target = 80 percentile

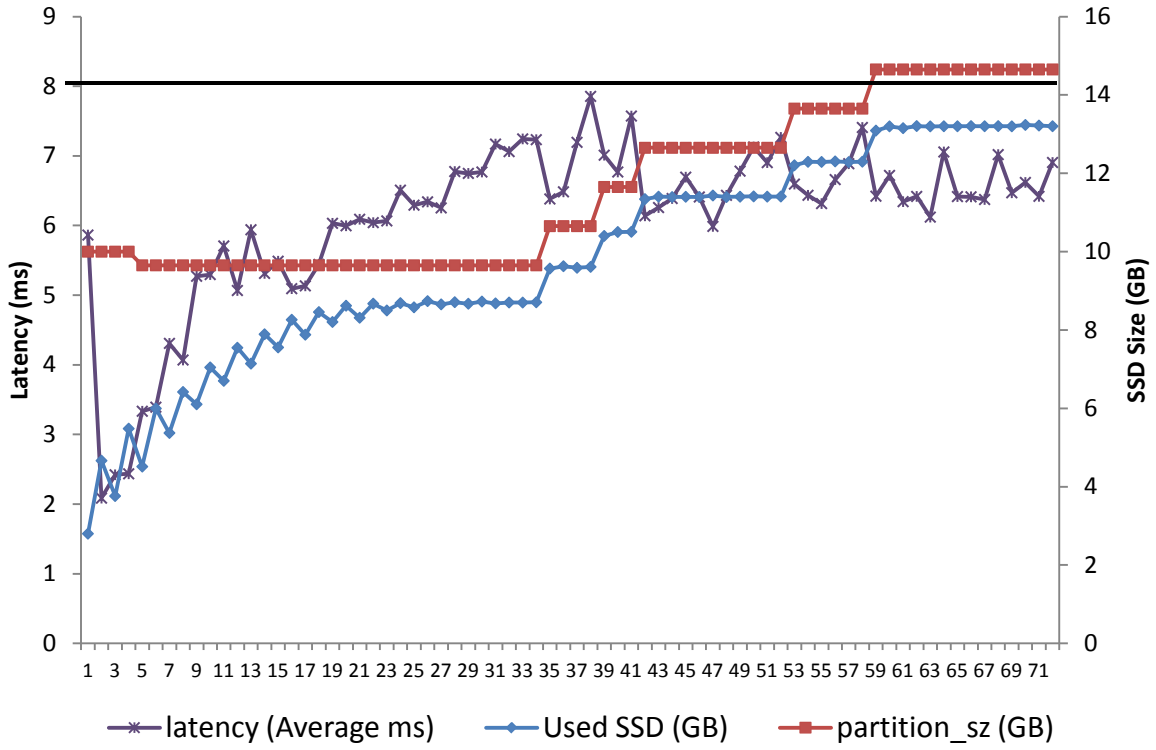


d. SLO Target = 75 percentile

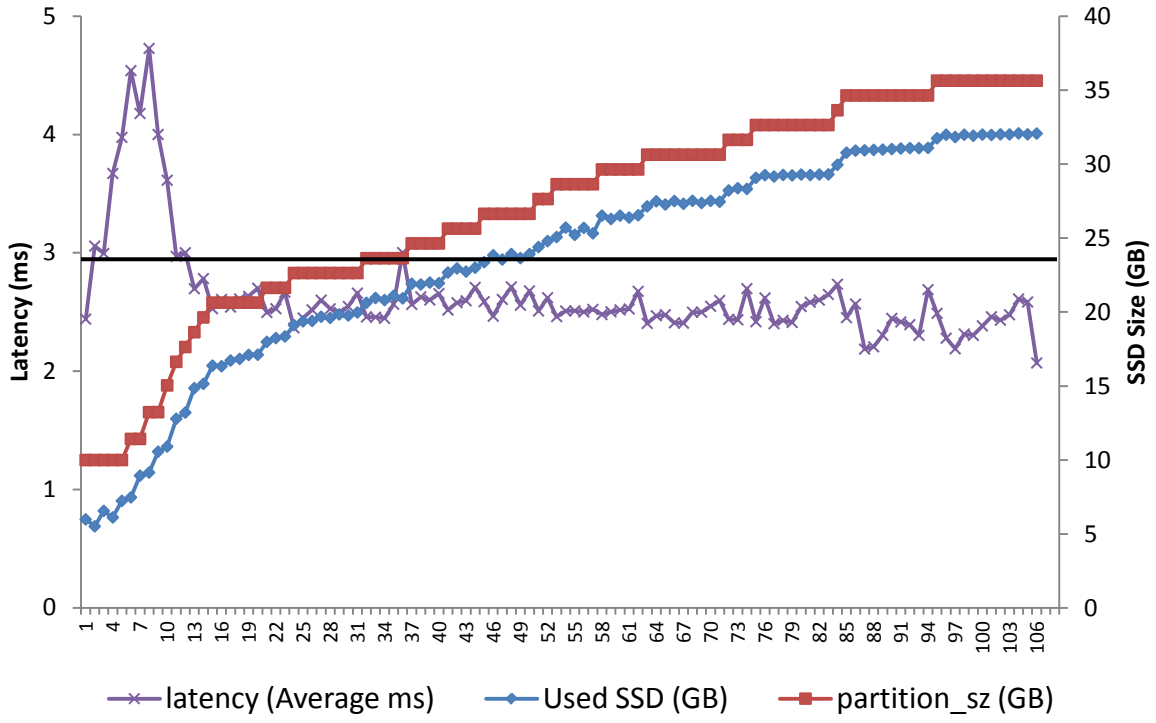
Evaluation

- Hybrid Store prototype
  - 1 workload per volume
    - 1 workload: All I/Os coming to a volume
  - HDD Space: 1TB
  - SSD Space: 160GB
  - RAM Size: 16 GB
- Workload : SPECsfs 2008-like
  - No. of threads = 20
  - Load/thread = 250 IOPS → 5000 IOPS
  - Total WSS = 70 GB
  - Target Latency = 3ms and 8ms

SPECsfs 2008 (SLO target = 8ms)



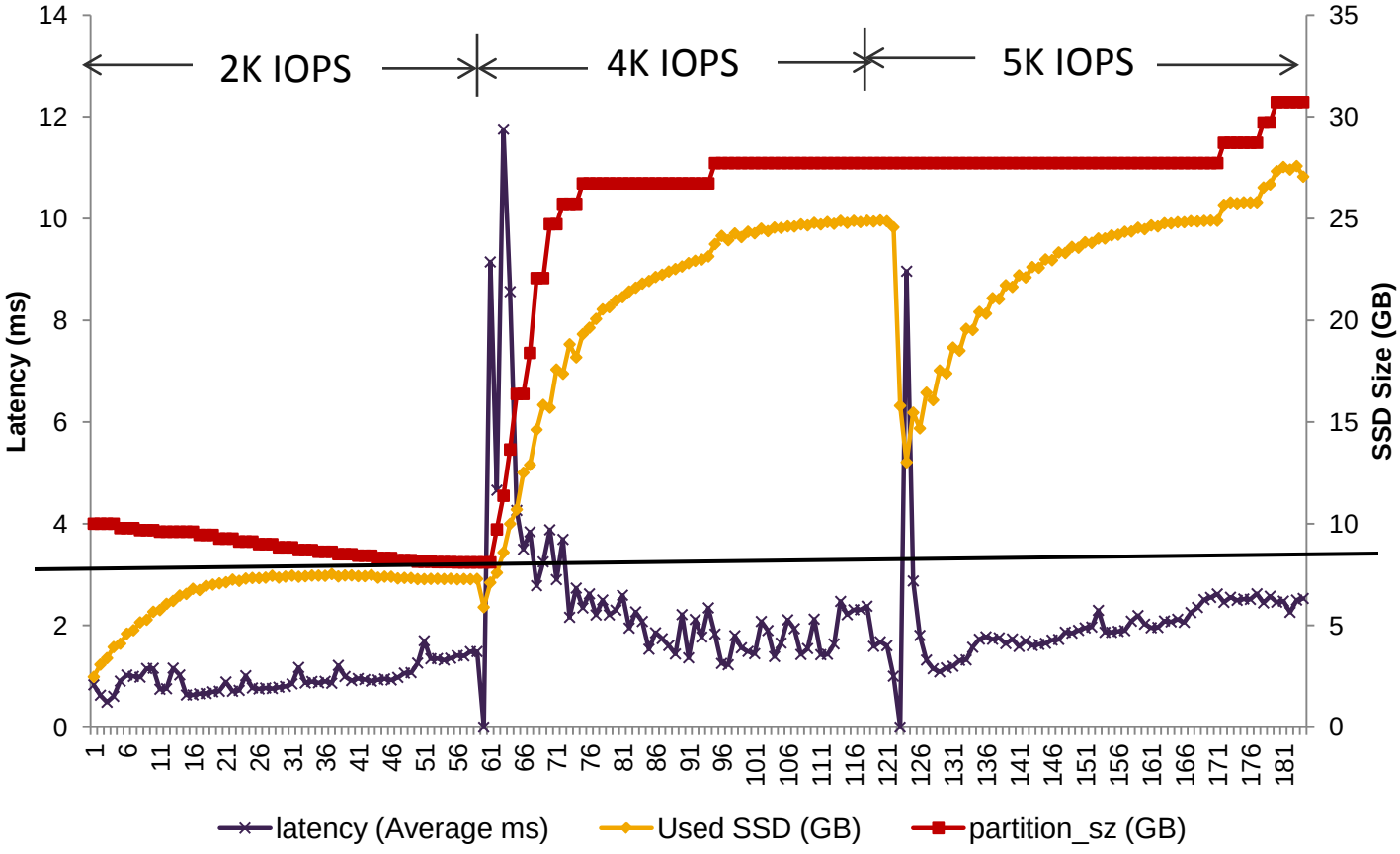
SPECsfs 2008 (SLO target = 3ms)



EAFC sizes SSD cache depending upon SLO requirement

Stringent SLO → More % of WSS needs to be cached

SPECsfs2008 (Varying loads)



EAFC adapts to changes in WSS and loads

Conclusion

- Insights
  - It is not necessary to cache whole WSS to meet certain latency targets
  - 75th percentile SLO conformance yields close to optimal SSD size meeting average SLO almost all the time
  - A cache sizer needs only few 100 history points to set the appropriate SSD size → light-weight

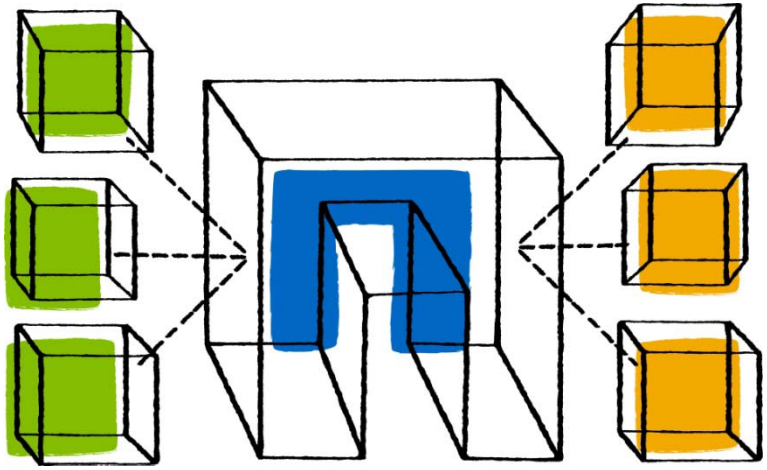
- Objectives Met
  - SLO met close to 100%
  - With close to optimal amount of SSD
  - Improving SSD utilization
  - Without much computation and memory overheads

NetApp

Thank you

NetApp

Backup Slides



References

[1] J.Chang and G.S. Sohi, “Cooperative cache partitioning for chip multiprocessors”, *In Proc. ICS’07*, 2007.

[2] S.Kim, D. Chandra, and Y.Solihin, ”Fair Cache Sharing and Partitioning in a Chip Multiprocessor Architecture”, *In Proc. PACT’04*, 2004.

[3] R.Lyer, “QoS: A framework for enabling qos in shared caches cmp platforms”, *In Proc.ICS’04*, pp.257-266, 2004.

[4] H. S. Stone, J. Turek, and J.L. Wolf, “Optimal partitioning of cache memory”, *IEEE Transactions on Computers.*, 41(9), 1992.

[5] G. E. Suh et al., “Dynamic partitioning of shared cache memory”, *In Journal of Supercomputing, 28(1),* 2004*.*

[6] M. K. Qureshi and Y.N. Patt, “Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches”, *In Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture.*

[7] Pawan Goyal, et al., “CacheCOW: QoS for Storage System Cache”, *In Eleventh International Workshop on Quality of Service*, 2003.

Sizing based on SLO needs

Test Case	Size (Vanilla)	Size (EAFC)	% space savings
SPECsfs (3ms)	70 GB	36 GB	48%
SPECsfs (8ms)	70 GB	15 GB	78%