SDC

STORAGE DEVELOPER CONFERENCE

SNIA SANTA CLARA, 2017

# Self-Optimizing Caches


Irfan Ahmad

CachePhysics

# About

**Irfan Ahmad**
CachePhysics Cofounder
CloudPhysics Cofounder
VMware (Kernel, Resource Management),
Transmeta,30+ Patents
University of Waterloo
@virtualirfan

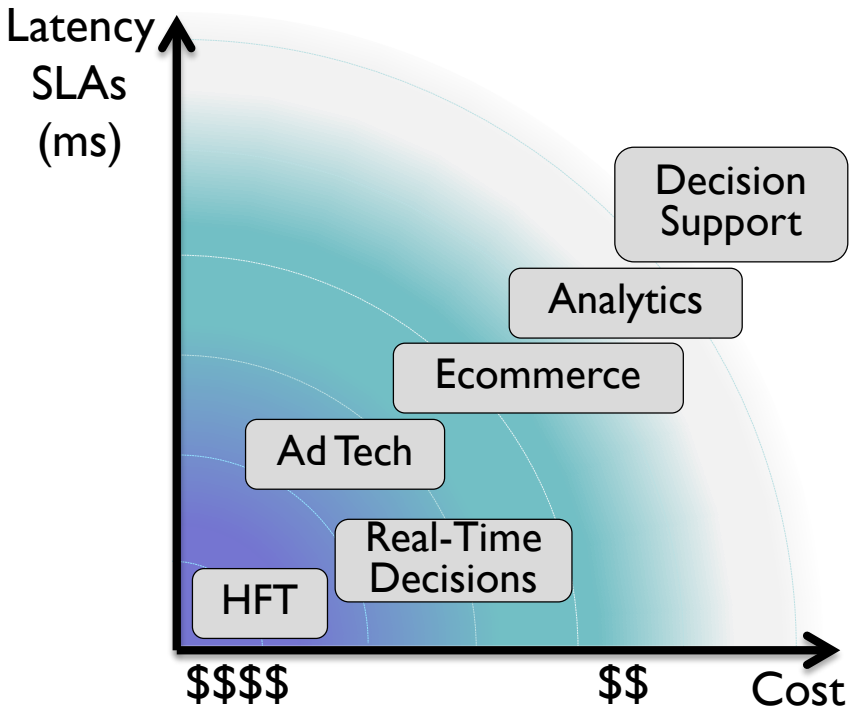
## CachePhysics

**Data Path Monitoring and Modeling Software**
Real-time Predictive Modeling of Data Access Patterns
Increasing Performance & Cost Efficiency of Existing Caches
Powering Next-Generation Self-Learning Caches





# Latency versus Cost



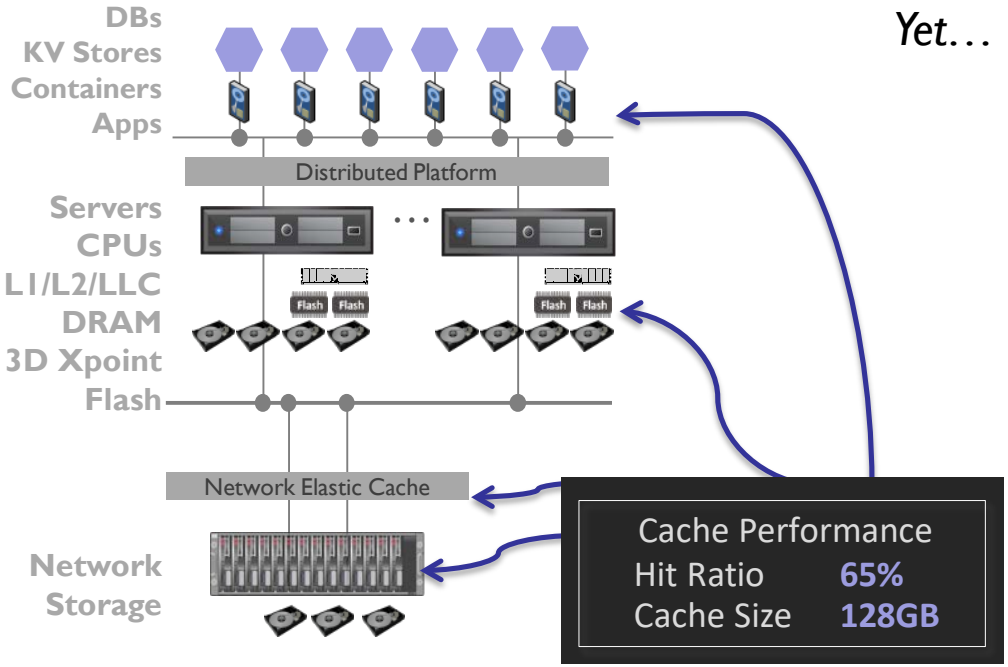

- DRAM
- Persistent Memory
- FLASH
- HDD

- Which apps need *low* latency?
- DRAM high $$ (10x v Flash)
- Where is the bottleneck?





# Caches are Critical to *Every* Applications




*Yet…*

## Intelligent Cache Management is Non-Existent

- Is this performance good?
- Can performance be improved?
- How much Cache for App A vs B vs …?
- What happens if I add / remove DRAM?
- How much DRAM versus Flash?
- How to achieve 99%ile latency of X µs?
- What if I add / remove workloads?
- Is there cache thrashing / pollution?
- What if I change cache parameters?

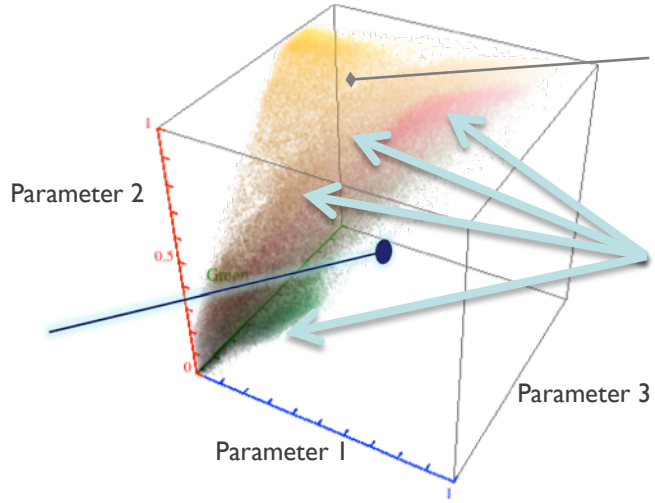




# Static vs. Self-Optimizing Caches

Today: parameters chosen based on benchmarking.

**One size does not fit all.**




Each point represents optimal cache settings for a single application

Applications have dramatically different *optimal* cache size and parameter settings.

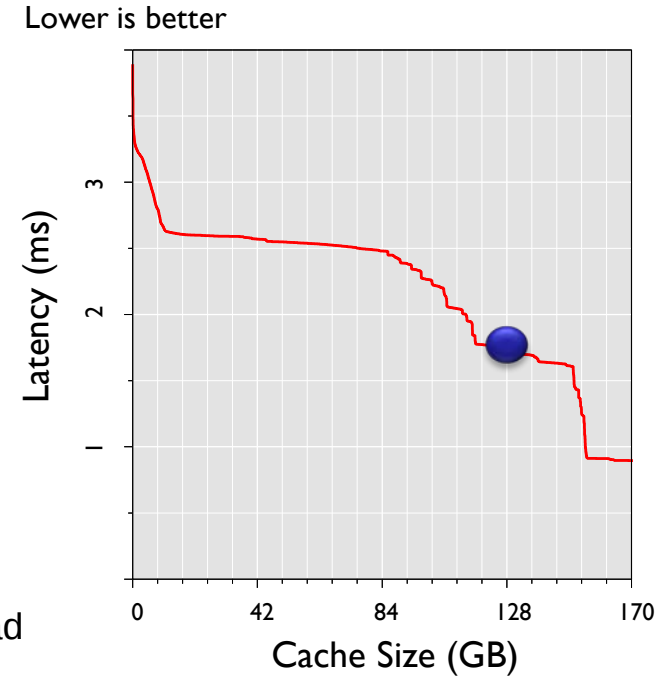
Static Caches Vulnerable to:

- Thrashing, Scan pollution
- Gross unfairness, Interference
- Unpredictability

⇒ Overprovisioning
⇒ Lack of Control

# Modeling Performance in Real-Time

| Cache Performance | |
|---|---|
| Hit Ratio | 65% |
| Cache Size | 128GB |

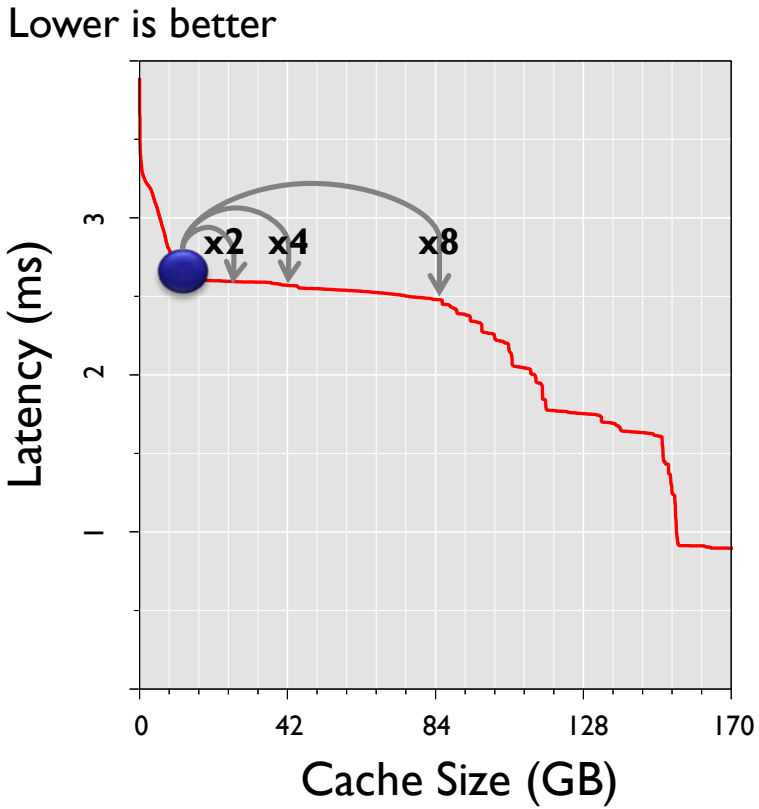



- Learn performance model of applications and cache
- Predict the performance of workload as *f(cache size, params)*





# Understanding Cache Models

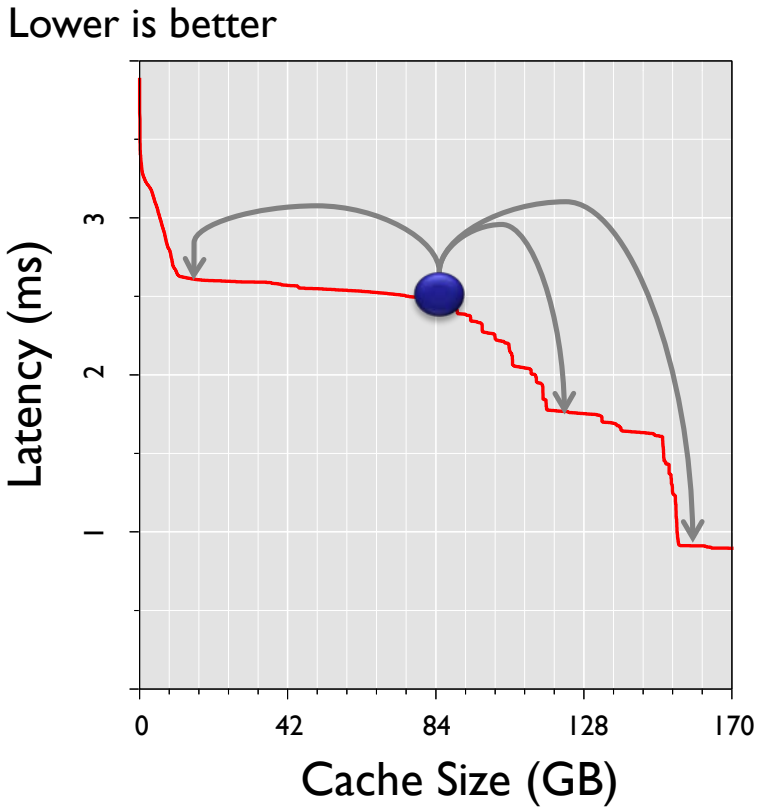



Models help decide useful increments of change.





# Understanding Cache Models (2)

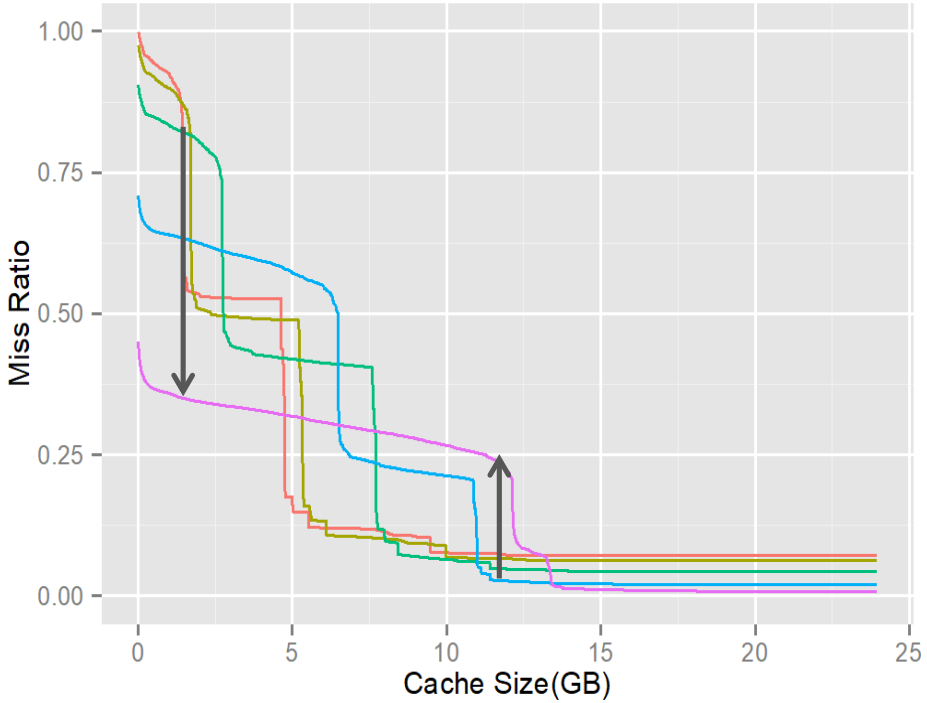



Often, most operating points are highly inefficient.





# Understanding Model-based Adaptation

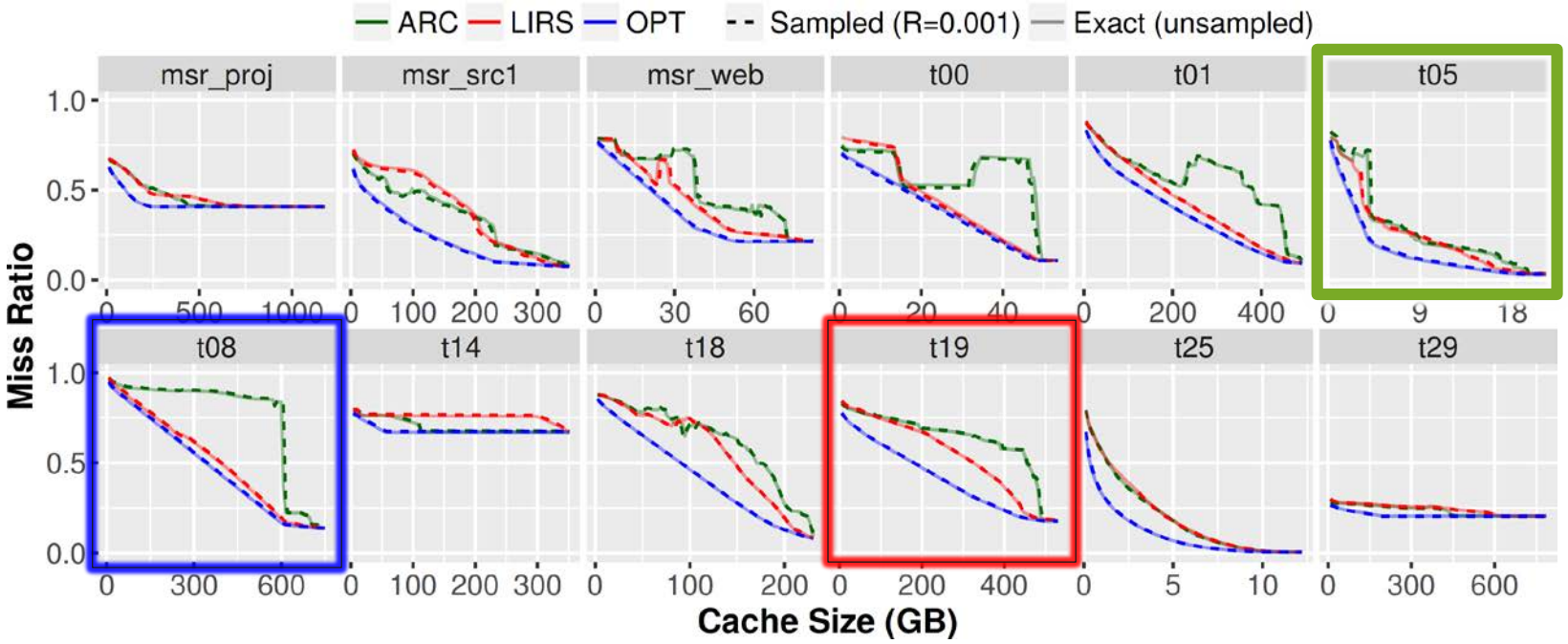



Single Workload. Prediction of performance under different policies.





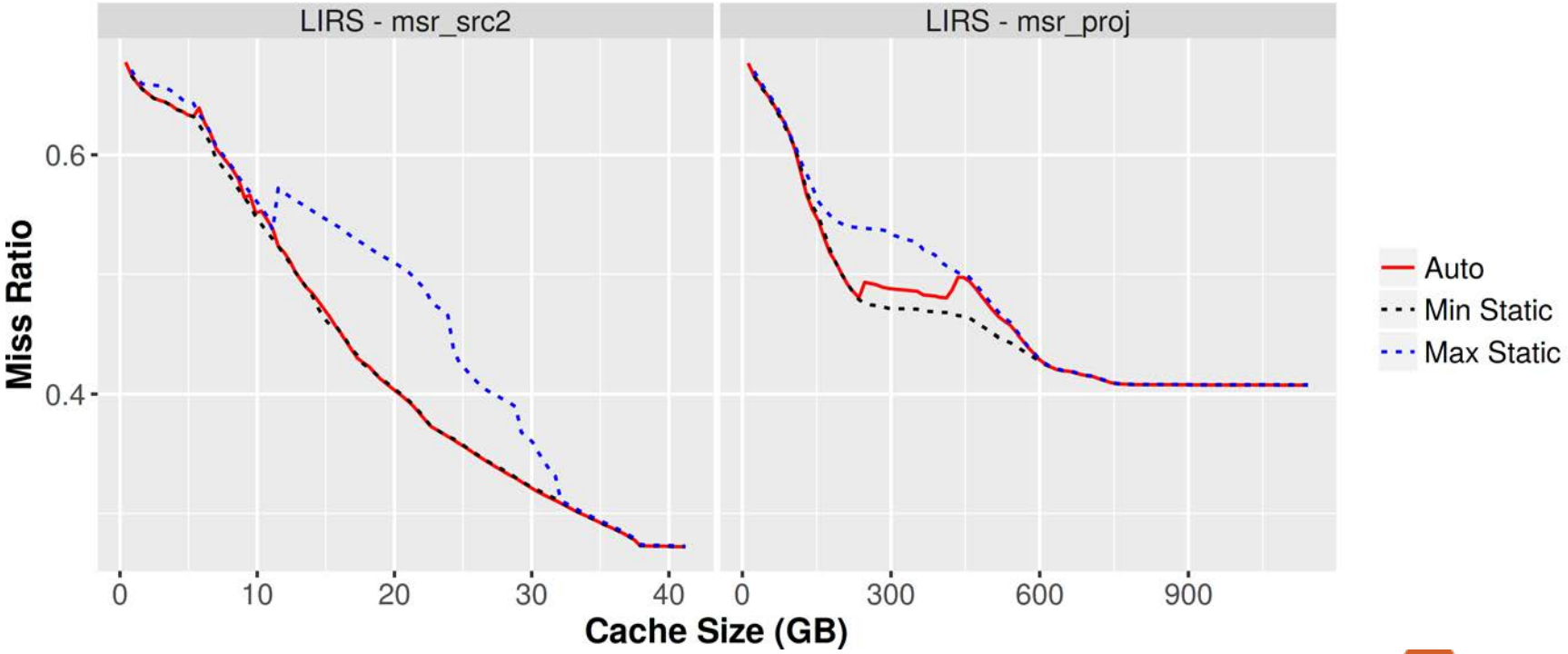
# Sample Models From Production Workloads







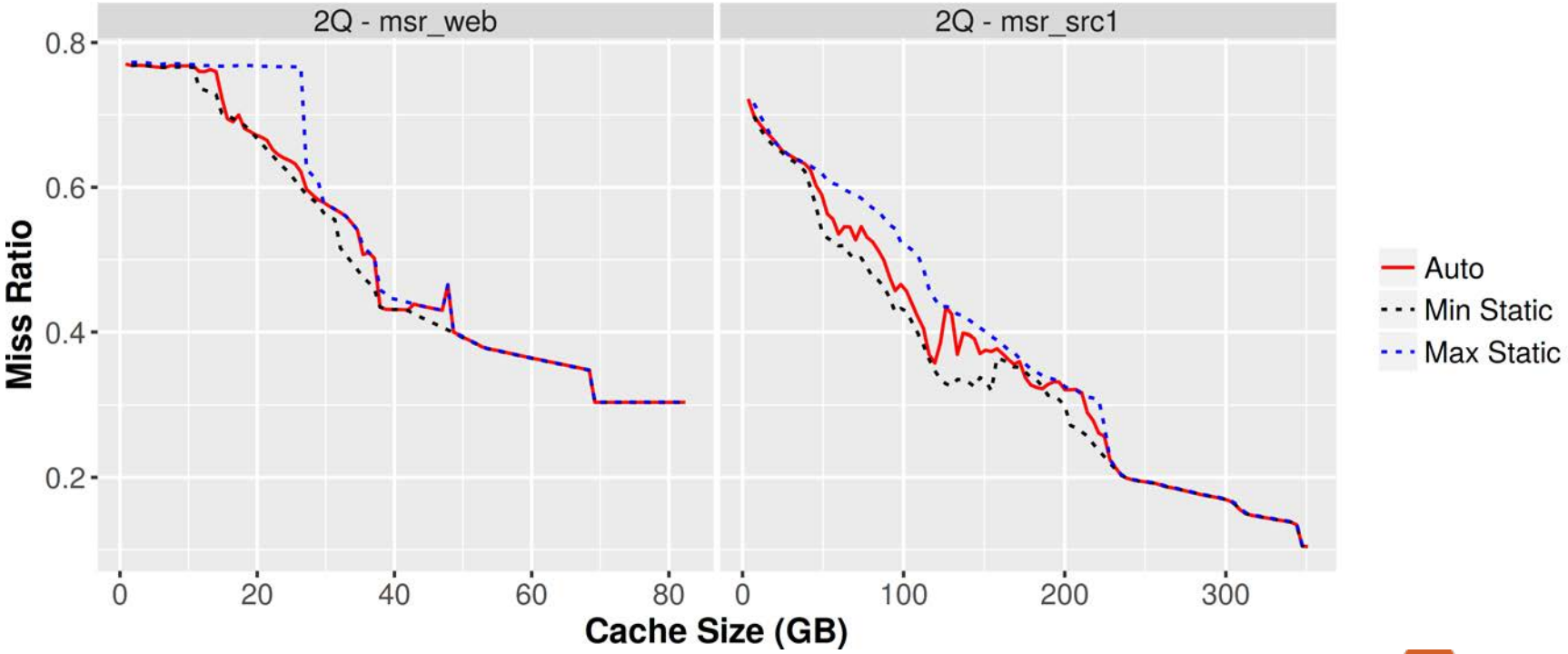

# LIRS Adaptation Examples



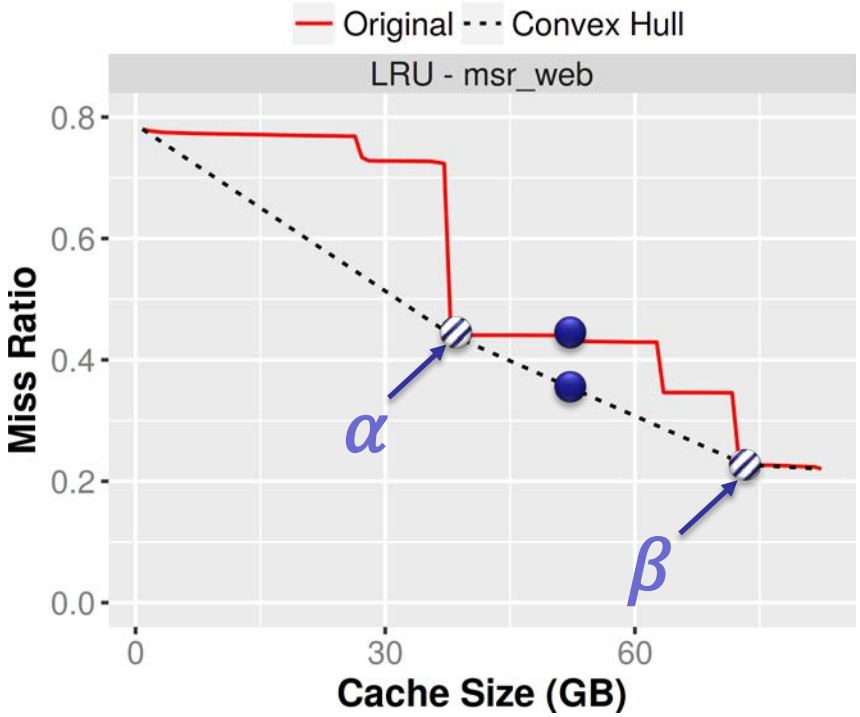





# 2Q Adaptation Examples



# Cliff Removal: New Class of Acceleration

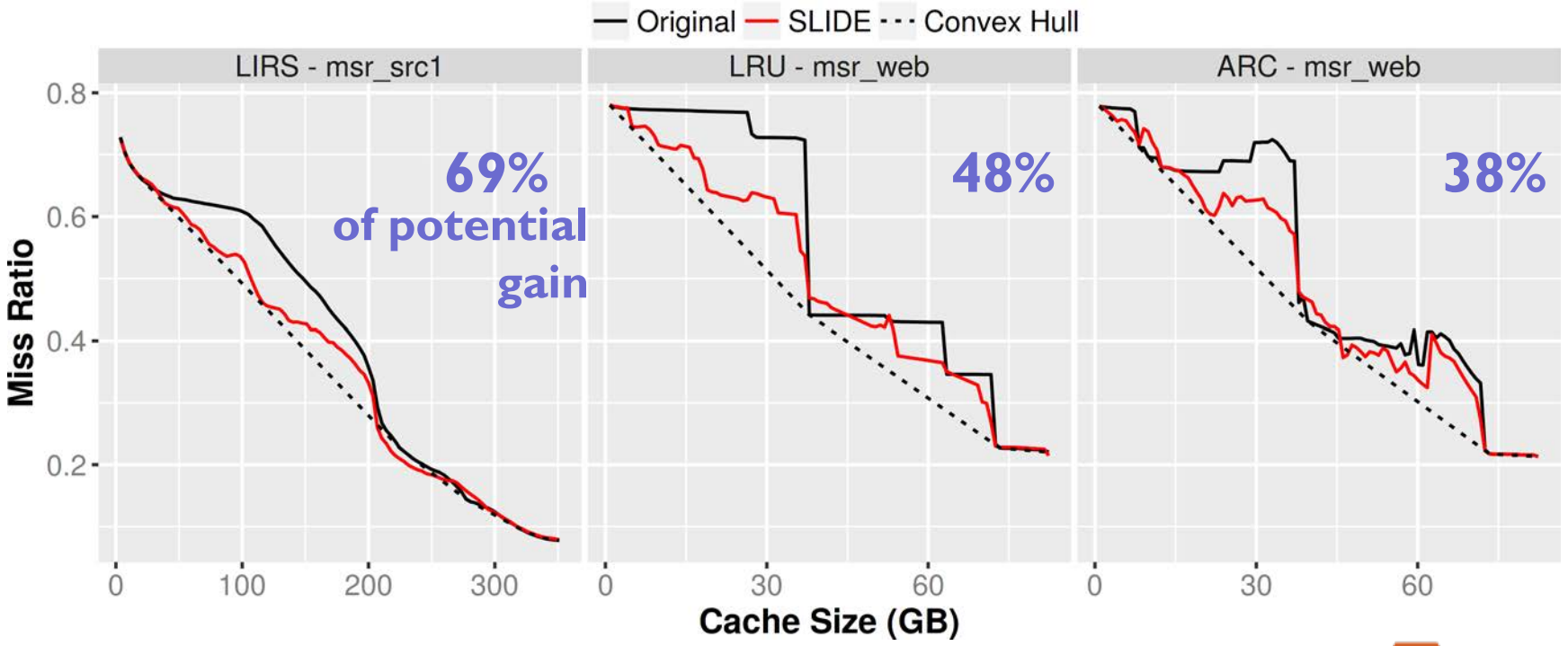



- Steer the curve?
  - Interpolate convex hull
  - Need Model (HPCA '15)
- Shadow partitions $\alpha$, $\beta$
  - Steer different fractions of refs to each
  - Emulate cache sizes on convex hull via hashing

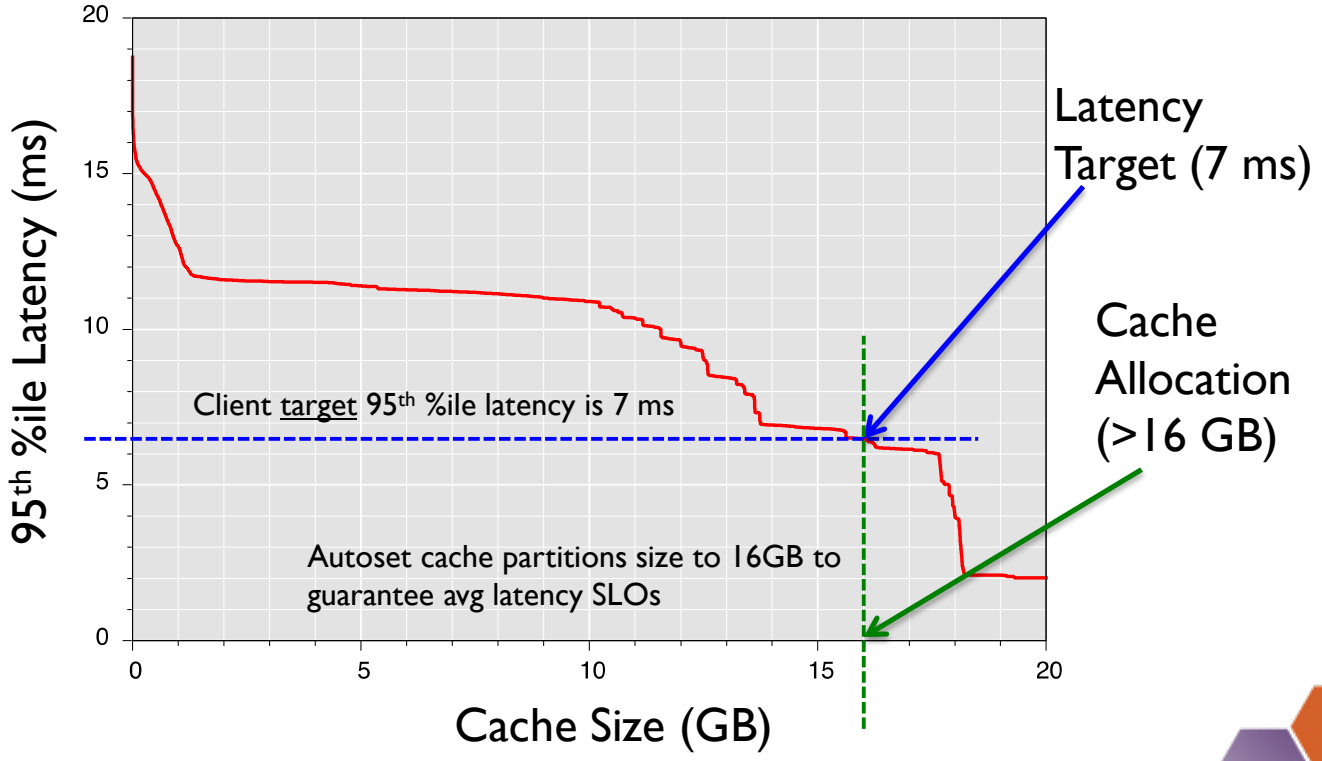




# Cliff Reduction Results

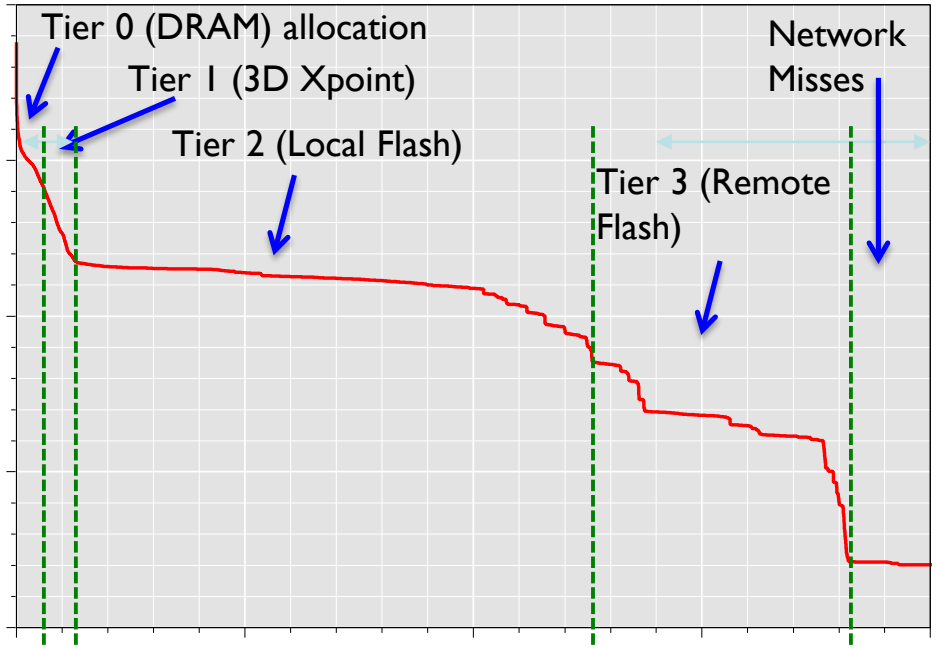


# Achieving Latency Targets




* Throughput targets can be implemented similarly

# Multi-Tier Sizing



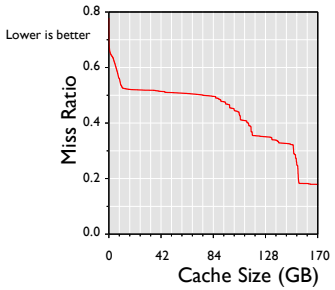

* Can model network bandwidth as a function of cache misses from each tier



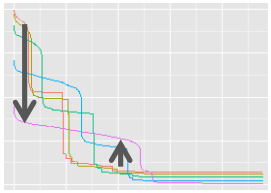


# Towards a Self-Optimizing Data Path

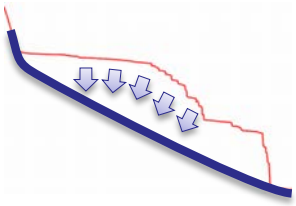
## Monitoring



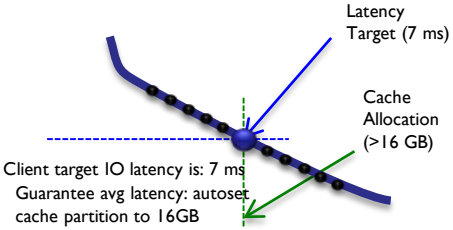

## Auto-Select Policies
(dynamic parameters)



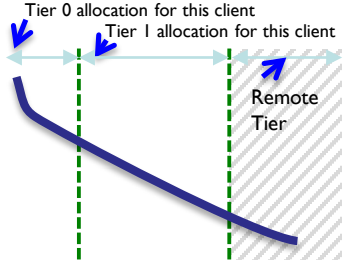
## Latency Reduction
(Thrashing Remediation)



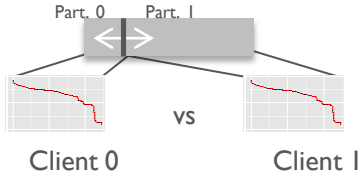
## Latency Guarantees




## Accurate Tiering

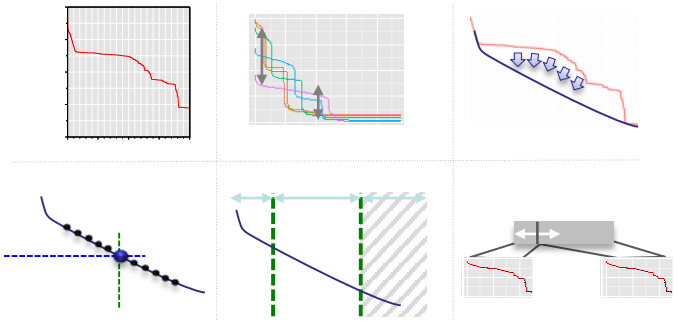



## Multi-Tenant Isolation




## Results:

- Safely quantify impact of changes
- Often 50-150% cache efficiency improvements ($$)
- Latency SLAs met
- Fewer production fire fights
- Higher consolidation ratios
- Accurate Capacity Planning



# CachePhysics

irfan@cachephysics.com 650-417-8559 @virtualirfan