Cache Modeling and Optimization using Miniature Simulations

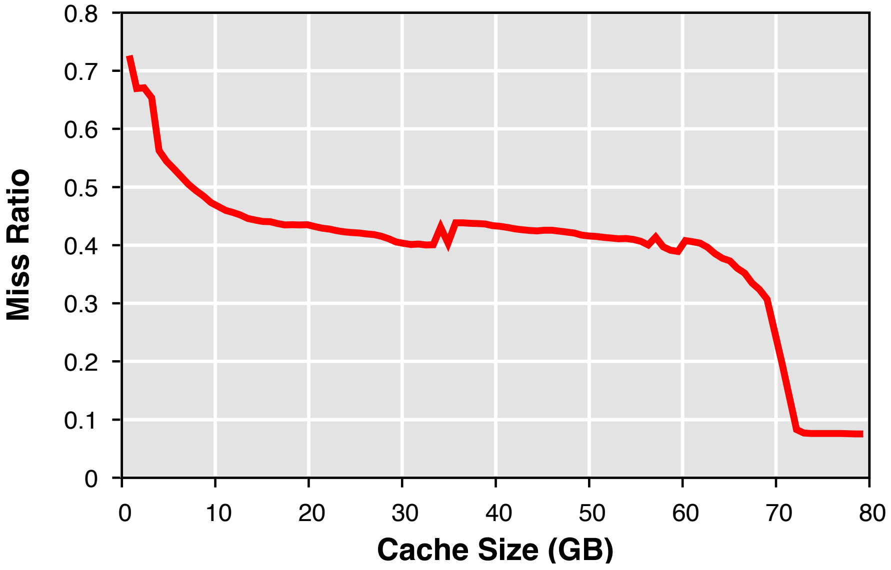
Carl Waldspurger *CachePhysics, Inc.*
Trausti Saemundsson *CachePhysics, Inc.*
Irfan Ahmad *CachePhysics, Inc.*
Nohhyun Park *Datos IO, Inc.*

USENIX Annual Technical Conference (ATC '17)
July 13, 2017

Motivation

- Caching important, ubiquitous
- Optimize valuable cache resources
  - Improve performance, QoS
  - Sizing, partitioning, tuning, cliff removal, …
- Problem: need accurate, efficient models
  - Complex policies, non-linear, workload-dependent
  - No general, lightweight, online approach

Cache Modeling



- Cache utility curves
  - Performance as *f*(size, …)
  - Miss ratio curve (MRC)
  - Latency curve
- Observations
  - Non-linear, cliffs
  - Non-monotonic bumps

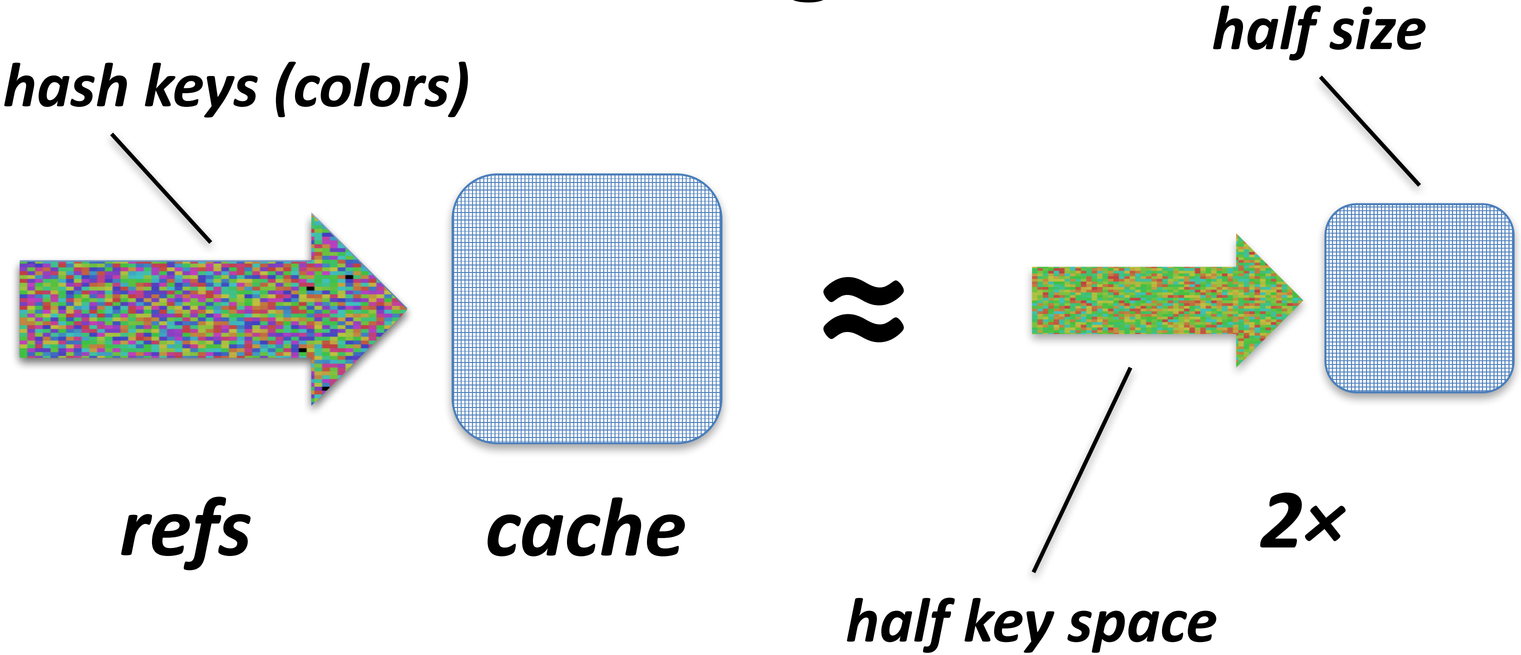
MRC Construction Methods

	Exact	Approximate
Stack Algorithms LRU, LFU, …	Mattson algorithm all sizes at once	Counter Stacks [OSDI ’14] SHARDS [FAST ’15] AET [ATC ’16]
Any Algorithm ARC, LIRS, 2Q, FIFO, …	separate simulation for each cache size	***miniature simulation*** ***[ATC ’17]***

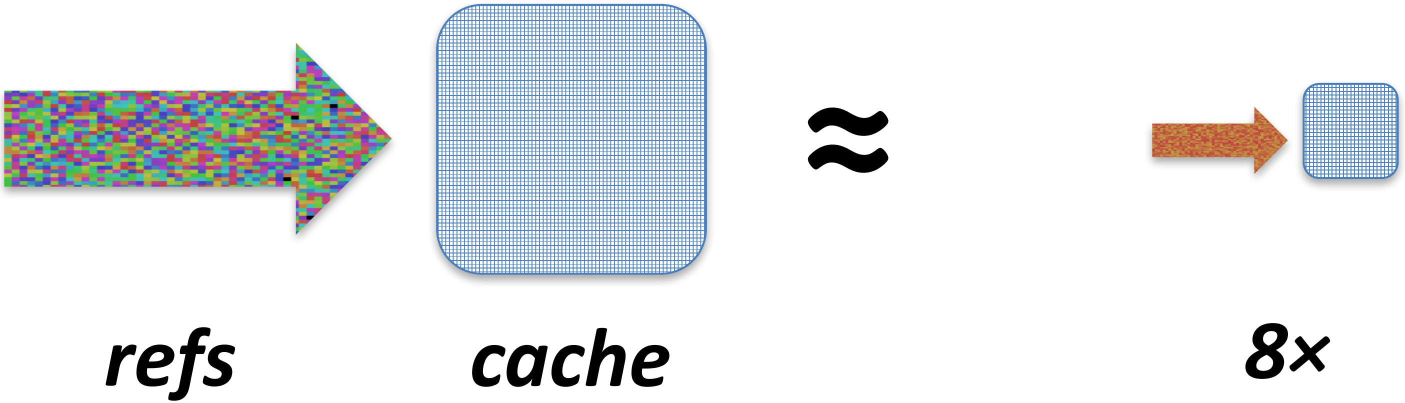
Miniature Simulation

- Simulate large cache using tiny one
- Scale down reference stream, cache size
  - Random sampling based on *hash*(key)
  - Assumes statistical self-similarity
- Run unmodified algorithm
  - LRU, LIRS, ARC, 2Q, FIFO, OPT, …
  - Track usual stats

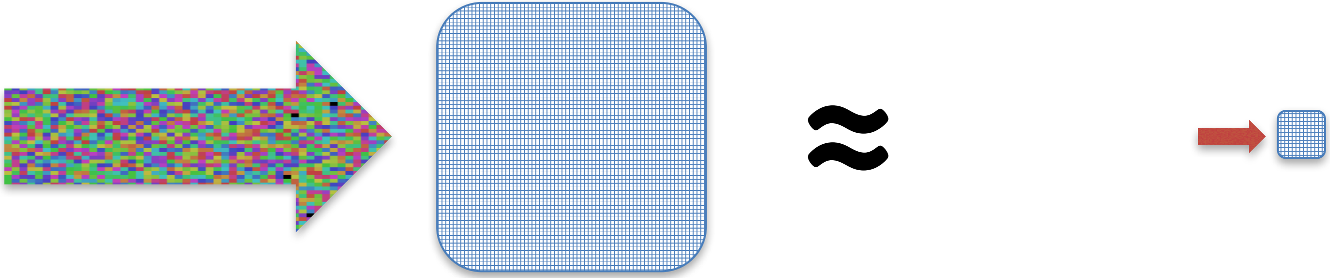
Scaling Down



Scaling Down

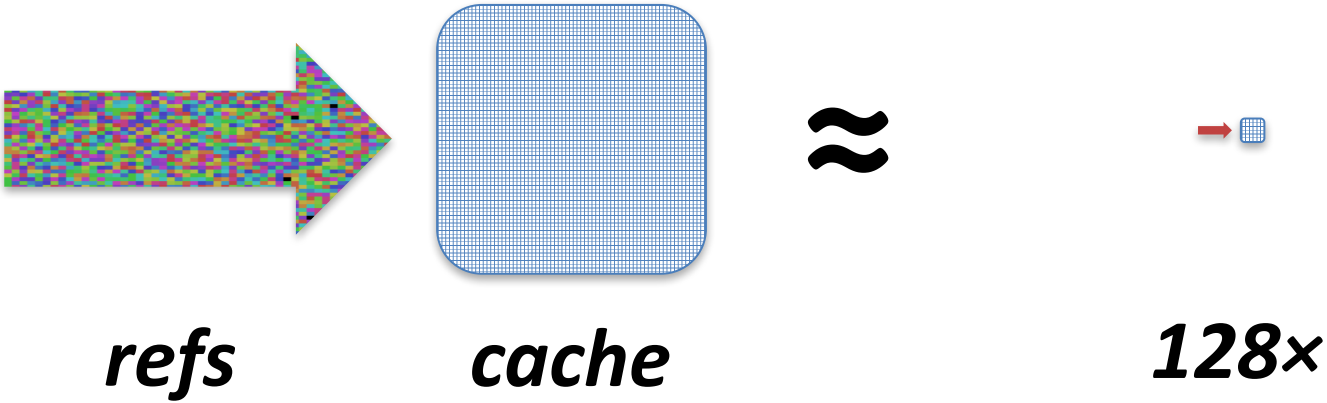


Scaling Down



refs *cache* *32×*

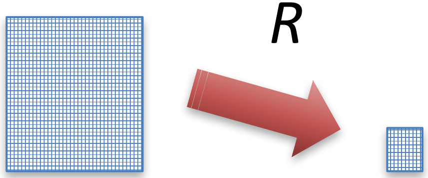
Scaling Down



Flexible Scaling

sampling rate

R



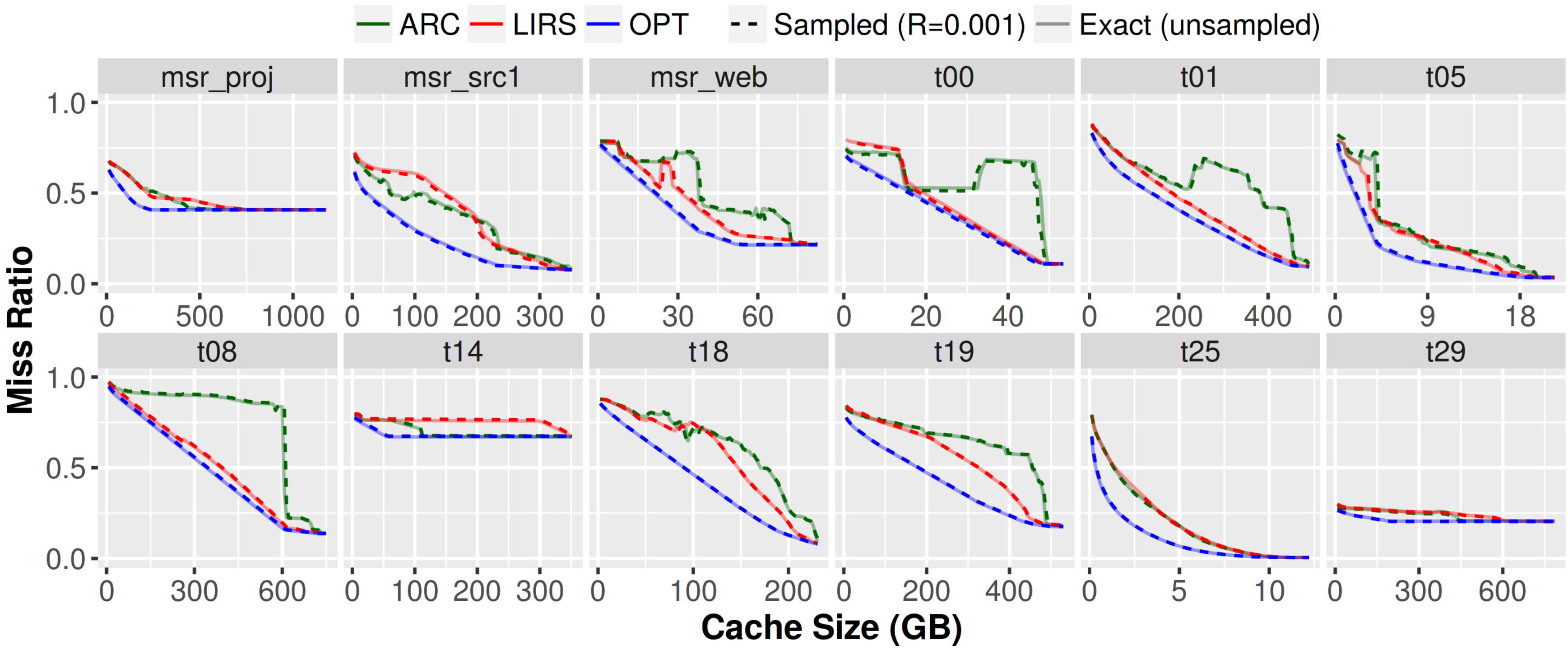
S_e *emulated size*

S_m *mini size*

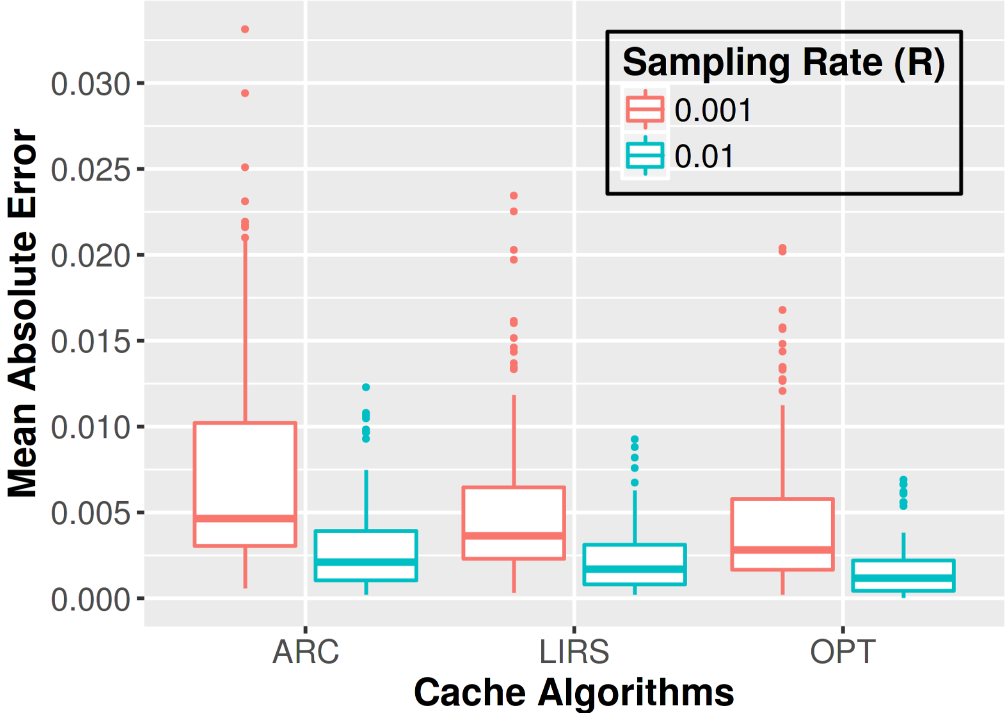
$$S_m = R \times S_e$$

- Time/space tradeoff
  - Fixed sampling rate R
  - Fixed mini size S_m
- Example: S_e = 1M
  - $R = 0.005 \Rightarrow S_m = 5000$
  - $S_m = 1000 \Rightarrow R = 0.001$

Example Mini-Sim MRCs



Mini-Sim Accuracy



- 137 real-world traces
  - Storage block traces
  - CloudPhysics, MSR, FIU
  - 100 cache sizes per trace
- Mean Absolute Error
  - | exact – approx |
  - Average over all sizes

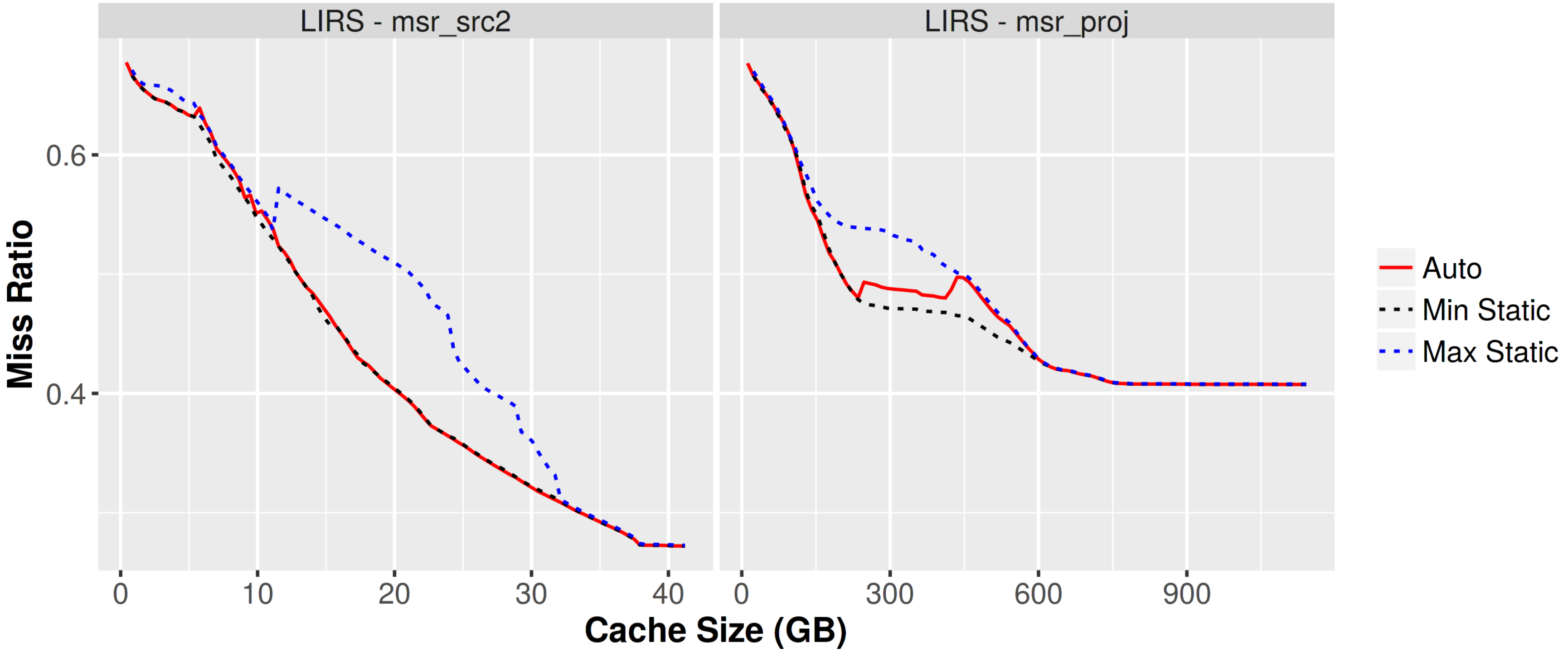
Mini-Sim Efficiency

- Variable costs
  - Both space and time scaled down by *R*
  - *R*=0.001 ⇒ simulation 1000× smaller, 1000× faster
- Fixed costs
  - Hashing overhead for sampling
  - Footprint for code, libraries, etc.
- Net improvement
  - *R*=0.001 ⇒ ~ 200× smaller, ~ 10× faster
  - Closer to 1000× if existing key hash or multiple sims

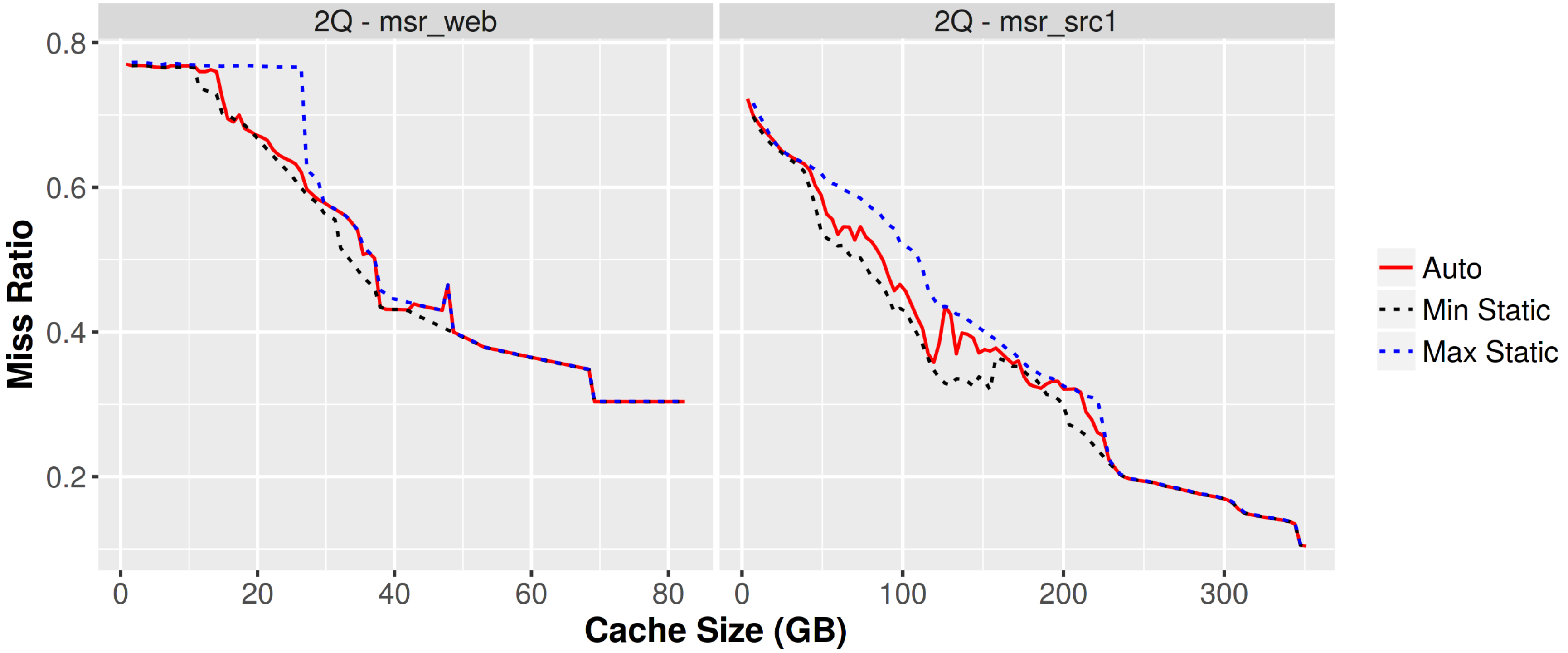
Mini-Sim Cache Tuning

- Dynamic multi-model optimization
  - Simulate candidate configurations online
  - Periodically apply best to actual cache
- Parameter adaptation experiments
  - LIRS *S* stack size, 5 mini-sims with f = 1.1 — 3
  - 2Q $A1_{out}$ size, 8 mini-sims with K_{out} = 50% — 300%
  - R = 0.005, epoch = 1M refs

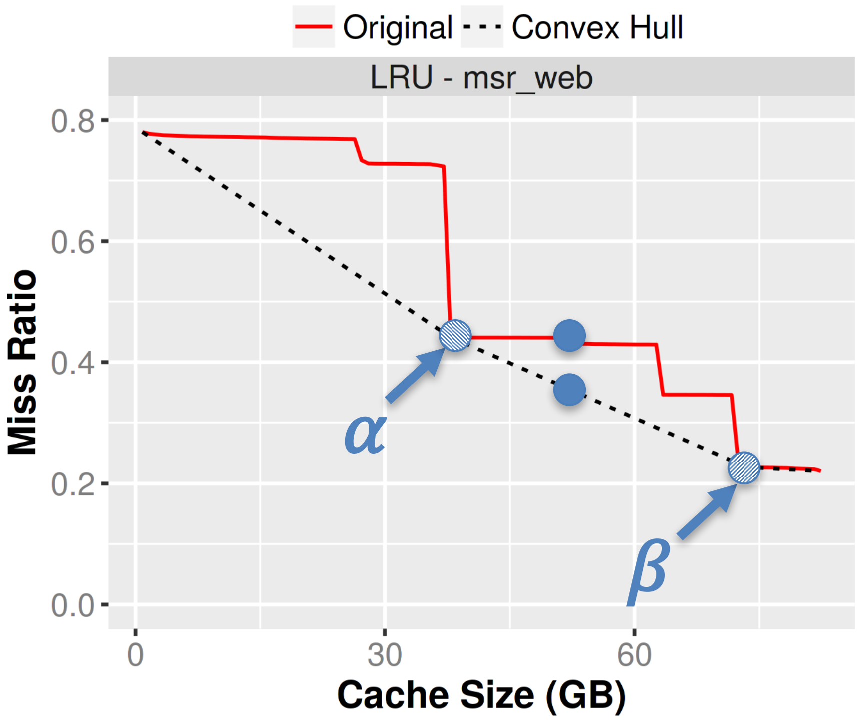
LIRS Adaptation Examples



2Q Adaptation Examples



Talus Cliff Removal



- Talus [HPCA ’15]
  - Needs MRC as input
  - Interpolates convex hull
- Shadow partitions α, β
  - Steer different fractions of refs to each
  - Emulate cache sizes on convex hull via hashing

Talus for Non-LRU Policies?

- Need efficient online MRCs
- Support dynamic changes?
  - Workload and MRC evolve over time
  - Resize partitions, lazy vs. eager?
  - Migrate cache entries in “wrong” partition?
- Not clear how to merge/migrate state

SLIDE: Transparent Cliff Removal

- ***S***harded ***L***ist with ***I***nternal ***D***ifferential ***E***viction
  - Single unified cache, no hard partitions
  - Defer partitioning decisions until eviction
  - Avoids resizing, migration, complexity issues
- New SLIDE list abstraction
  - No changes to ARC, LIRS, 2Q, LRU code
  - Replaces internal LRU/FIFO building blocks

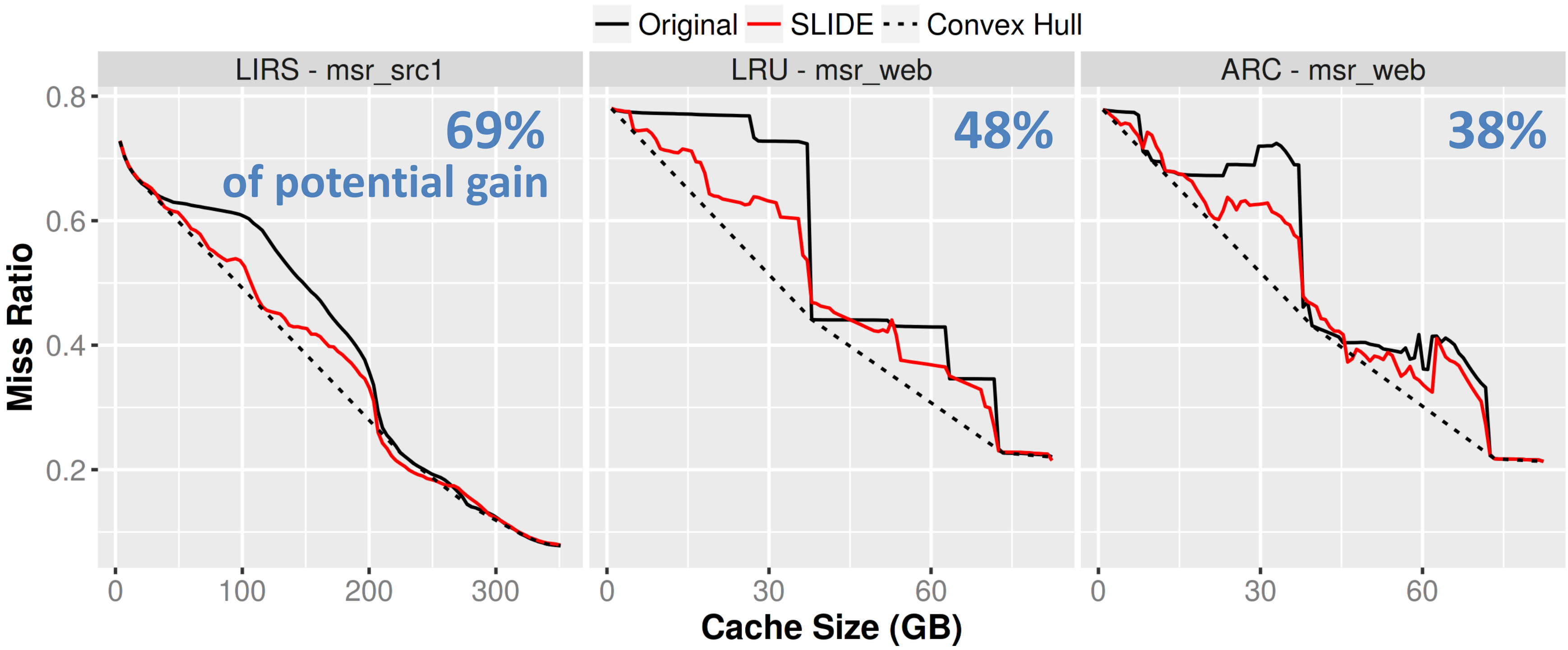
SLIDE List

- Augment conventional list
  - Per-item hash value
  - Hash threshold determines current "partition"
- Items totally ordered, no hard partitions
- Evict from tail of over-quota partition

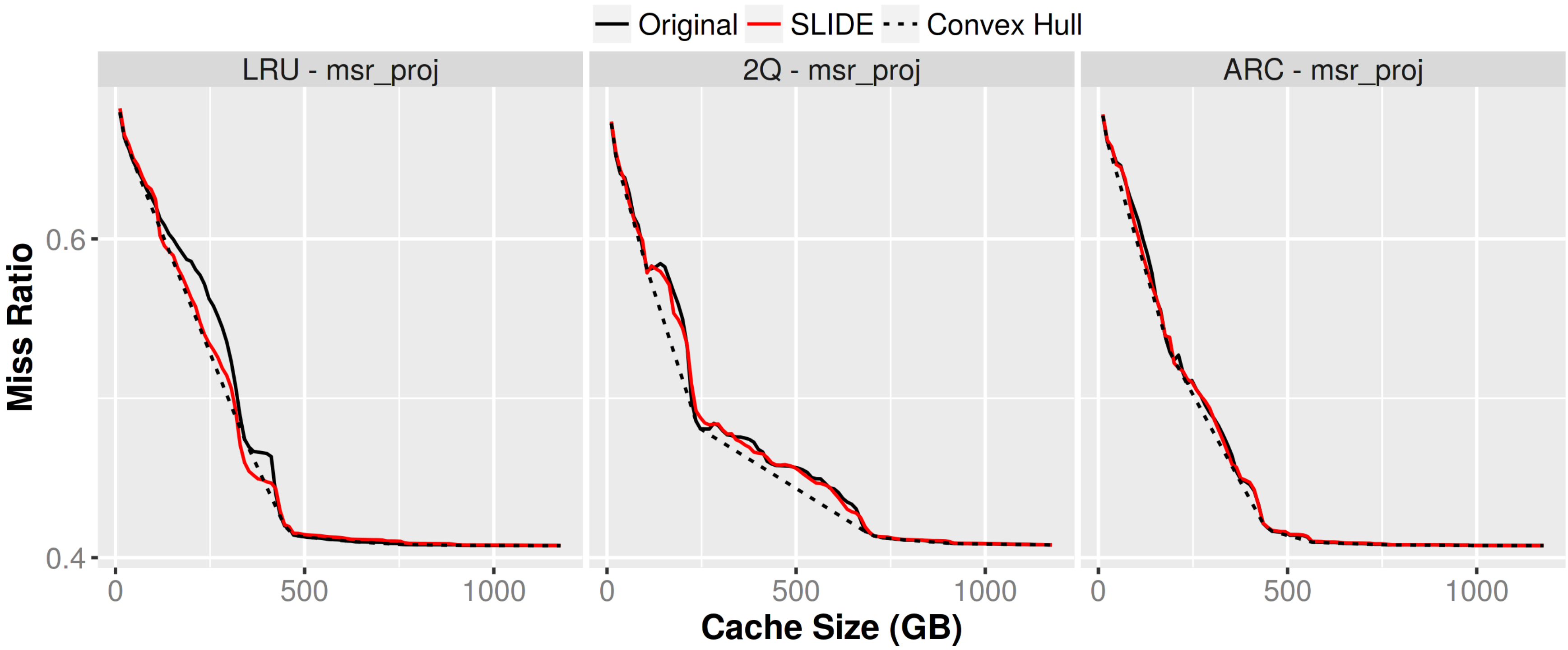
SLIDE Experiments

- Construct MRC online
  - 7 mini-sims {⅛, ¼, ½, 1, 2, 4, 8} × cache size
  - *R*=0.005, smoothed miss ratios
- Update SLIDE settings periodically
  - Discrete convex hull every epoch (1M refs)
  - Set new "partition" targets for SLIDE lists

SLIDE: Cliff Reduction



SLIDE: Little Impact without Cliffs



Conclusions

- Mini-sim extremely effective
  – Robust, general method (ARC, LIRS, 2Q, LRU, OPT, …)
  – Average error < 0.01 with 0.1% sampling
- Can optimize workloads/policies automatically
  – Dynamic parameter tuning
  – SLIDE transparent cliff removal