CSR:Small: Effective Sampling-Based Miss Ratio Curves – Theory and Practices

Zhenlin Wang, Michigan Technological University

(CSR1618384, Oct. 2016 – Sept. 2019)

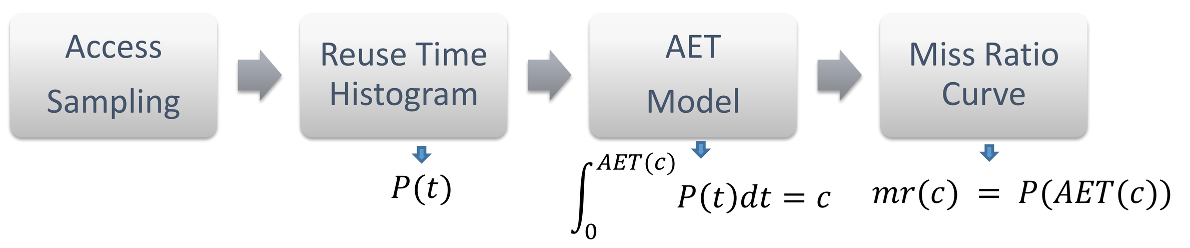
Objectives

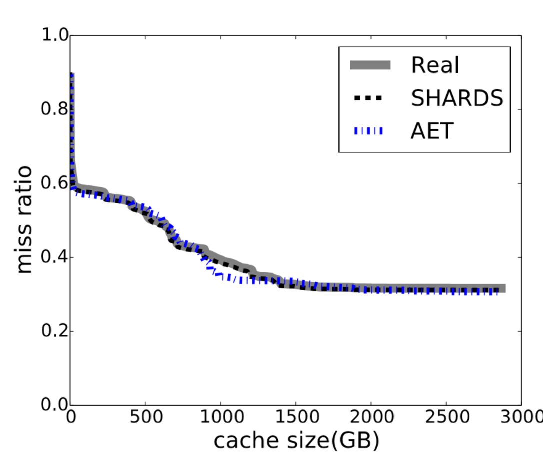
- Develop a new cache model based on reuse time distribution and a novel concept of average eviction time (AET) to construct miss ratio curves effectively [1]. Focus on in-depth study of the hypotheses behind the model and develop theoretical foundation for sampling.
- Conduct a systematic comparison of recent cache models with respect to their assumptions of access distribution: the reuse distance-based models such as SHARDS and Counter Stacks, and the reuse time-based models: the footprint theory, Statstack, and AET [2].
- Study theory and practice for hardware cache partitioning. Develop effective online miss ratio curve approaches that exploit the recent Intel Cache Allocation Technology (CAT) [3].
- Investigate theory and practice for hugepages and exploit huge pages to reduce TLB pressure. The two classes of pages, regular or huge, introduce a new challenge to the AET model with nonuniform miss penalties and block granularities. We propose a study on composability of AET-based MRCs and examine the impact of different page sizes and miss penalties [4].
- Research theory and practice for key-value memory cache management and its interaction with hypervisor-level dynamic memory management [5]. We propose to develop management algorithms for distributed memory cache, taking advantage of composability of the AET model, and investigate the interaction between application-level and hypervisor-level memory management [2].

Miss Ratio Curve (MRC) using Average Eviction Time (AET)

AET Process:

1. Find reuse time distribution through sampling
   - For all t, find $P(t)$, the probability for an access with a reuse time greater than t
   - Reuse time is the time between a reference and its next reuse
2. AET Model: Given $P(t)$, find average eviction time of cache size c
3. Miss ratio curve: given cache size c, find miss ratio

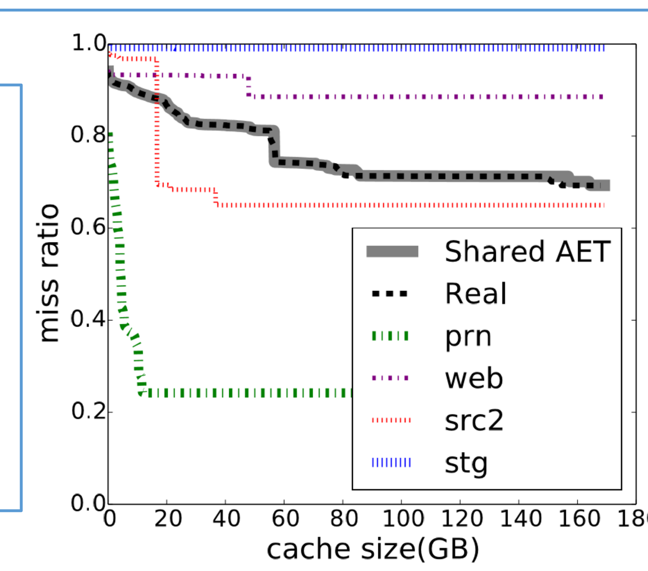


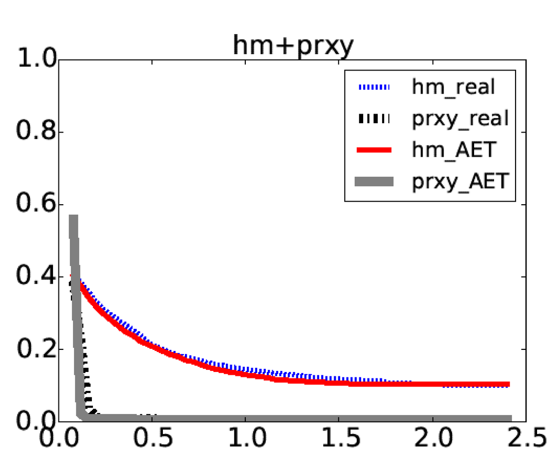


AET shows comparative prediction accuracy when compared to SHARDS [6], using the MSR traces by Microsoft Research Cambridge
- SHARDS is a reuse distance-based model
  - Applies spatial sampling to reduce cost
  - Uses approximation w.r.t. sampling rate to estimate reuse distances
- AET is a reuse time-based model
  - Explores a variety of sampling
  - Needs to model the relationship between reuse time and reuse distance

AET model is composable: With reuse-time distribution of each individual program or trace, AET can model the co-run MRC.

The graph on the right shows the MRCs of four MSR traces and the co-run MRC



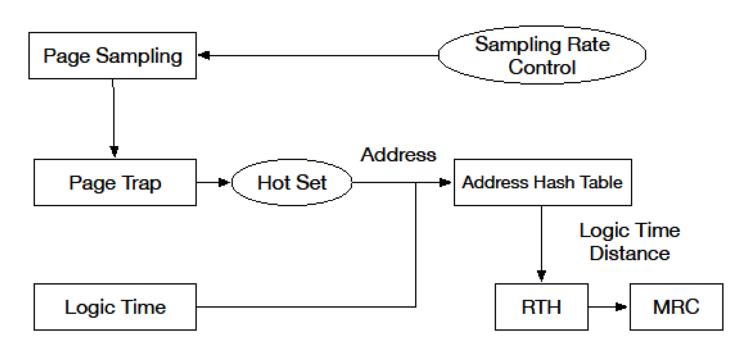


AET model can be extended to model cache hierarchy. The graph on the left shows the L2 MRCs of two MSR traces where each trace has a private L1 (40MB) and shares an L2 cache with the other

Summary: AET shows advantages in both time and space complexities

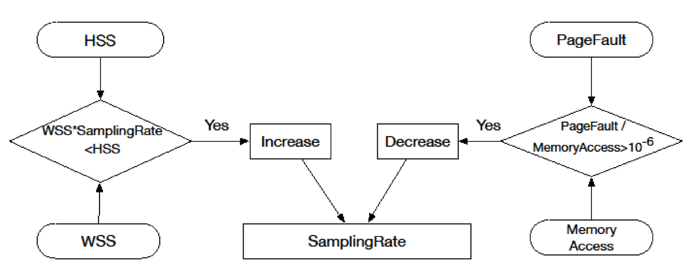
	Time complexity	Space complexity	Memory	Runtime	Composability	Correctness
Stack Processing	$O(NM)$	$O(N)$	10GB	> 1 day	No	accurate
Search Tree	$O(N\log M)$	$O(M)$	21GB	482 secs	No	accurate
Scale Tree	$O(N\log\log M)$	$O(M)$	17GB	333 secs	No	bounded err
Footprint	$O(N)$	$O(M)$	17GB	50 secs	Yes	conditional
Counter Stacks	$O(N\log M)$	$O(\log M)$	80MB	1034 secs	No	bounded err
SHARDS	$O(N)$	$O(1)$	2.3MB	29.6 secs	No	conditional
AET model	$O(N)$	$O(1)$	1.7MB	30.5 secs	Yes	conditional

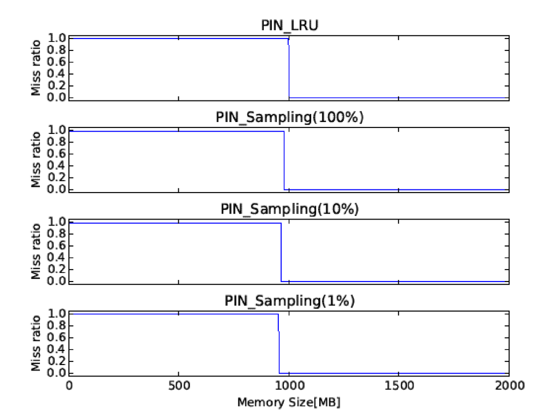
AET-Based Working Set Size (WSS) Prediction for Virtual Machines



Apply AET to predict WSS of a virtual machine:
- To track a page access, set a reserved bit to trap into the hypervisor
- Balance between #VM exits and MRC accuracy
  - Sampling (random or spatial)
  - Hot set: only trap/model cold pages

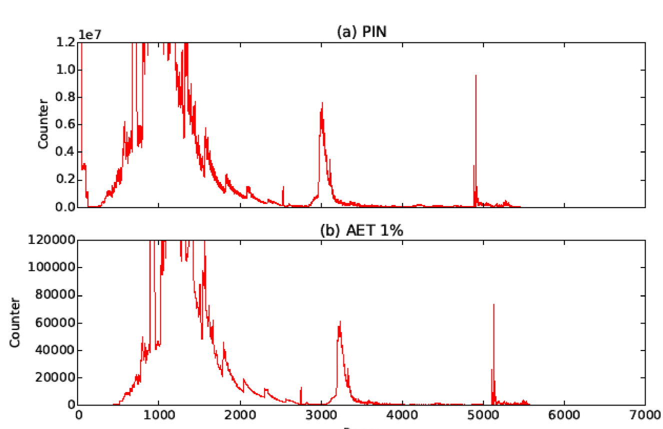
Sampling rate control:
- Empirically, one over a million soft page faults yields acceptable overhead
- Need to track a sufficient number of cold pages
- Balance between hot set size (HSS) and sampling rate

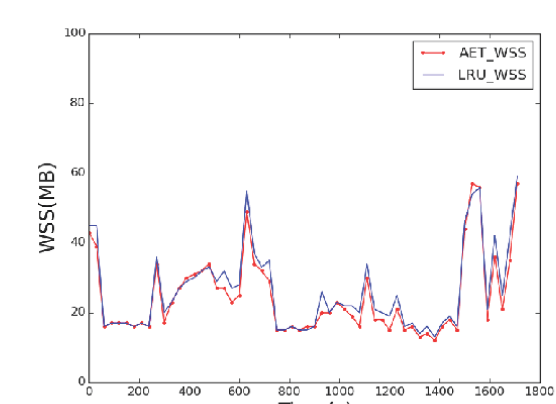


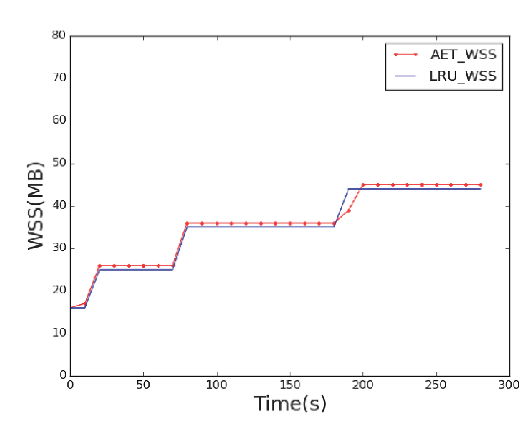


Without hot set, the AET model shows high accuracy with low sampling rate. The result on the right shows 1% sampling rate delivers the same MRC as 100% (a micro-benchmark on Pin). In [1], we show 1/10[4] sampling rate is acceptable

Trapping short reuse times is prohibitive. Hot set filters short reuse times as shown on the right. The RTH of *mcf* in SPEC 2006 beyond the 512-page hot set is accurate with the counts reduced to 1% (sampling rate) of the actual.



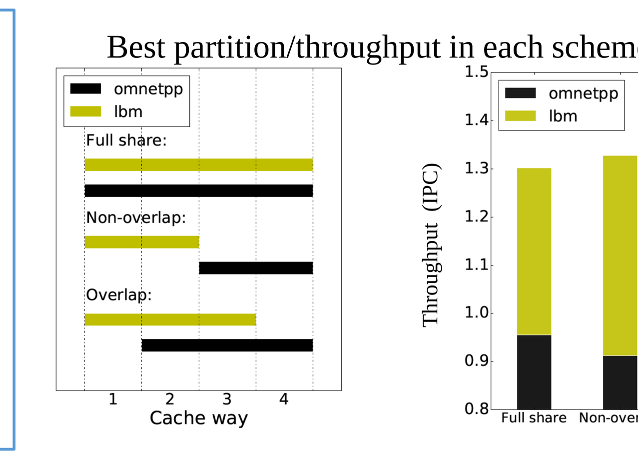


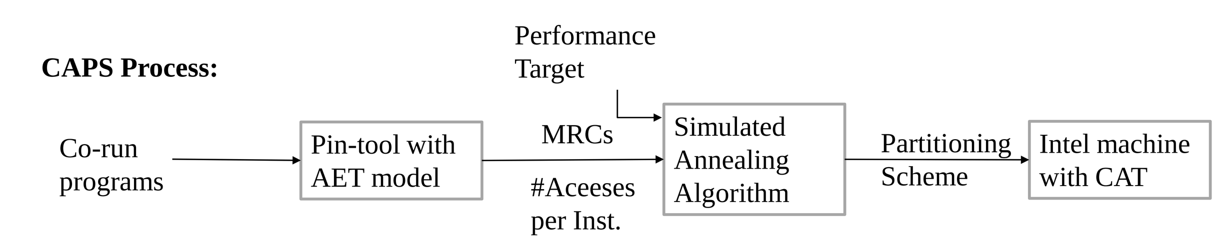


The two figures compare the predicted WSSs along the execution of *mcf* (left) and *omnetpp* (right): The AET-based approach is comparable to the reuse-distance-based approach which models LRU stack directly

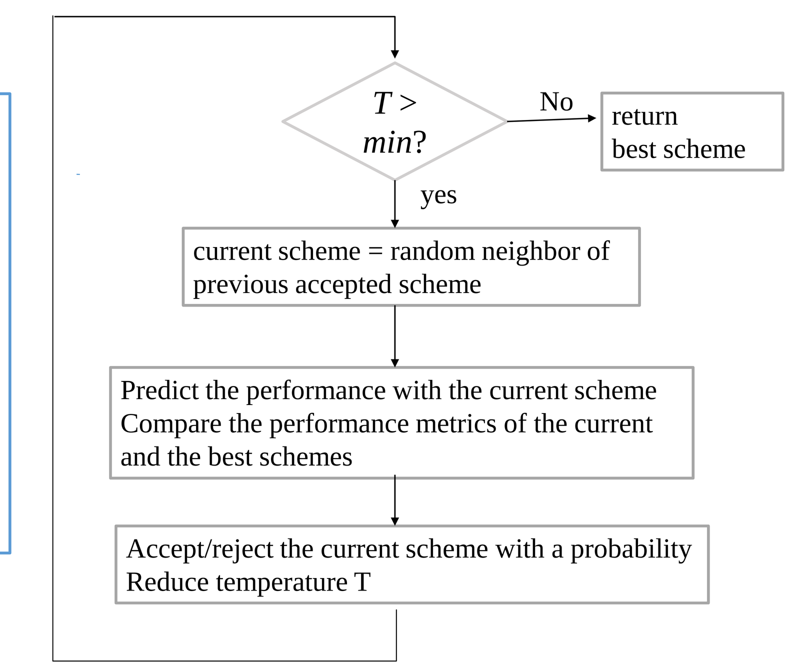
Cache Allocation Through Partial Sharing (CAPS)

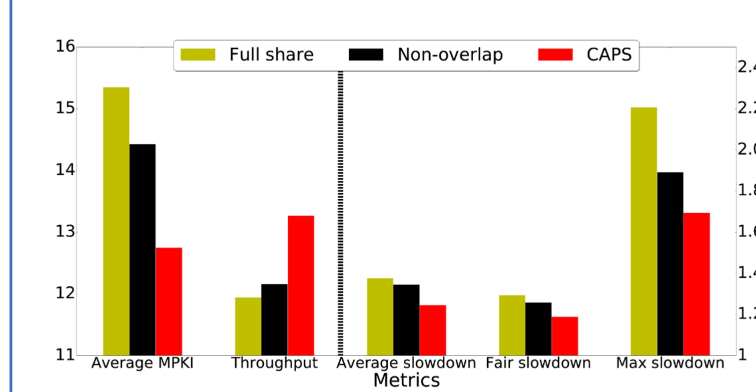
Recent Intel cache allocation technology (CAT) enables **partial sharing** (overlapping) among multiple cores/programs
- A program can have its dedicated ways while sharing some ways with others
- Partial sharing can deliver best performance
- Exponential solution space is the key challenge





Simulated annealing algorithm:
- Probabilistic algorithm to search a large space
- Applies random walk and a probability, $f(T)$, to accept a worse state to avoid getting stuck in a local optimum
- Uses MRCs and an iterative cache occupancy model to predict target performance
- Controls search using temperature

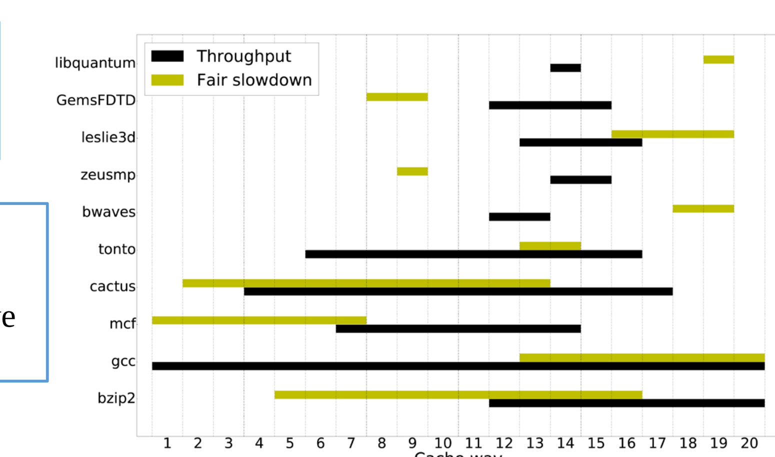




Average performance across 75 workloads:
- Disjoint partitioning (non-overlap) performs better than full sharing by avoiding cache contention
- Partial sharing explores the potential of CAT better by allowing partition overlapping
- The improvement space is significant, such as 11% on average in throughput

Case study on one workload mix:
- Different performance target yields different partitioning schemes

Challenges and ongoing work:
- Make cache-level AET on-the-fly
- Simulated annealing is too expensive
- Modeling multithreaded workload



References and Publications

1) Hu et al., “Kinetic Modeling of Data Eviction”, USENIX Annual Technical Conference, June 2016.
2) Hu et al., “Fast MRC Modeling Using Average Eviction Time”, Submitted to ACM Transactions On Storage (TOS).
3) Hu et al., “Optimal Symbiosis and Fair Scheduling in Shared Cache”, IEEE Transactions on Parallel and Distributed Systems (TPDS), April 2016.
4) Wang et al., “Evaluating the impacts of hugepage on virtual machines”, Science China Information Sciences 60:012103, Jan. 2017.
5) Hu et al., “Optimized Locality-aware Memory Management for Key-value Cache”, IEEE Transactions on Computers (TC), May 2017.
6) Waldspurger et al., “Efficient MRC Construction with SHARDS”, USENIX FAST, Feb. 2015.

Collaborators

- **Peking University:** Yingwei Luo, Xiaolin Wang, Xianmeng Hu, Lan Zhou, Zhigang Wang, Fan Hou. Taowei Luo, Zihui Huang, Yaocheng Xiang
- **Michigan Tech:** Nilufer Onder, Daniel Byrne, Wei Kuang
- **University of Rochester:** Chen Ding

Michigan Technological University