PEKING
UNIVERSITY

LAMA: Optimized Locality-aware Memory Allocation for Key-value Cache

Xiameng Hu, Xiaolin Wang, Yechen Li, Lan Zhou, Yingwei Luo
Peking University
Chen Ding
University of Rochester
Song Jiang
Wayne State University
Zhenlin Wang
Michigan Technological University

Outline

- Background
- Existing Solutions
- LAMA design
- Evaluation
- Conclusion

Background

- The in-memory caches are vital components in today's web server architecture.
    - Memcached
    - Redis

WIKIPEDIA
The Free Encyclopedia

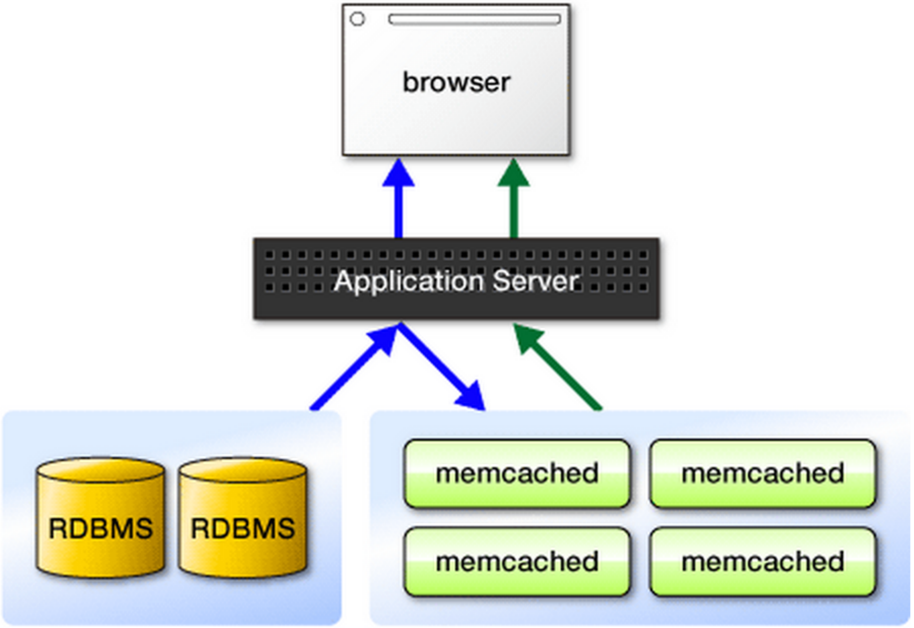
twitter

facebook

YouTube

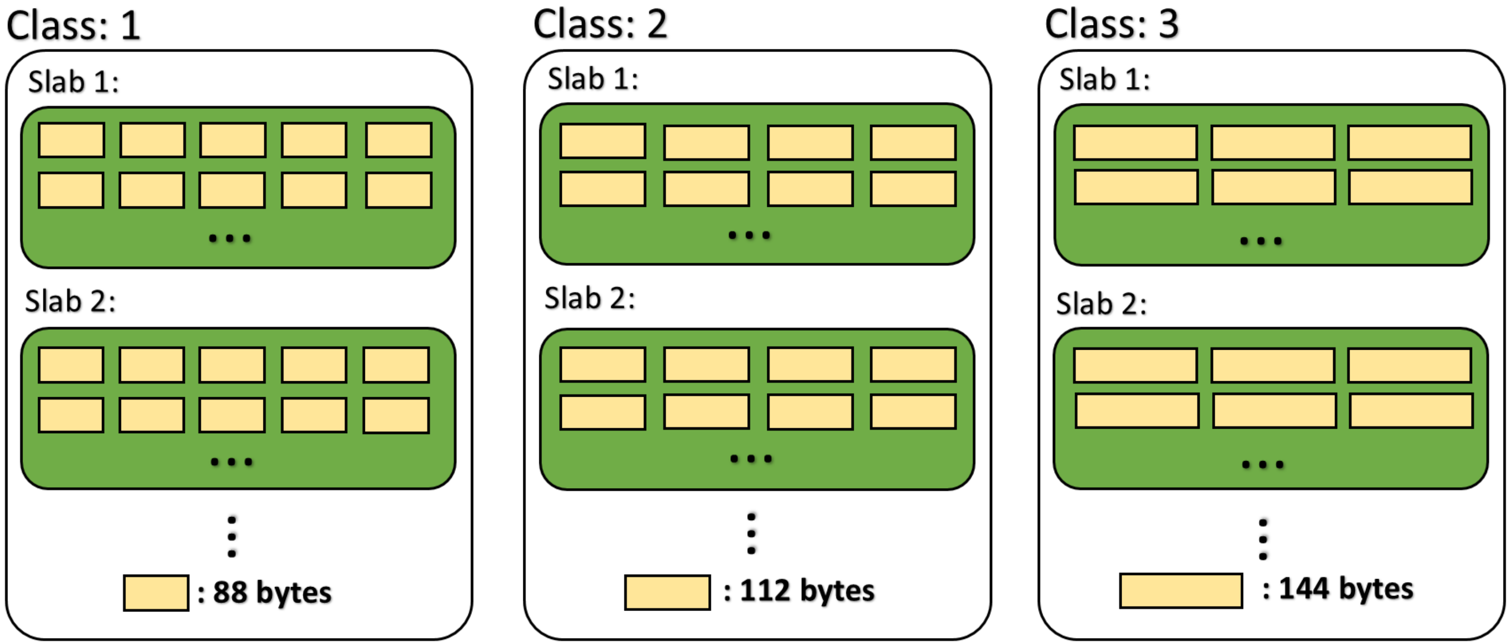
Memcached

- A high-performance, distributed memory object caching system.
  - Slab-based allocation.
  - Platform independent.
  - LRU eviction.



Memcached

- Split the space into different classes to store variable-size objects.
- Each class obtains its own memory space by requesting free slabs (1MB per slab).
- Each allocated slab is divided into slots of equal size.
- The slot size increases exponentially.



Memcached

- Default Memcached fills the cache at the cold start based on the demand.
- Demand-driven slab allocation may not deliver best performance.
- Default allocation results in slab calcification.

Example For Demand-driven Slab Allocation

- There are two classes of data references:
  - Class 1: “abcabcabc…”.
  - Class 2: “123456789…”.
  - Combined reference pattern: “a1b2c3a4b5c6a7b8c9…”.
  - There are four slabs and each slab contains one slot.

Default Allocation

Trace: a 1 b 2 c 3 a 4 b 5 c 6 a 7 b 8 c 9

Class 1 slabs : 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Class 2 slabs : 0 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

hits : 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Total hits: 0

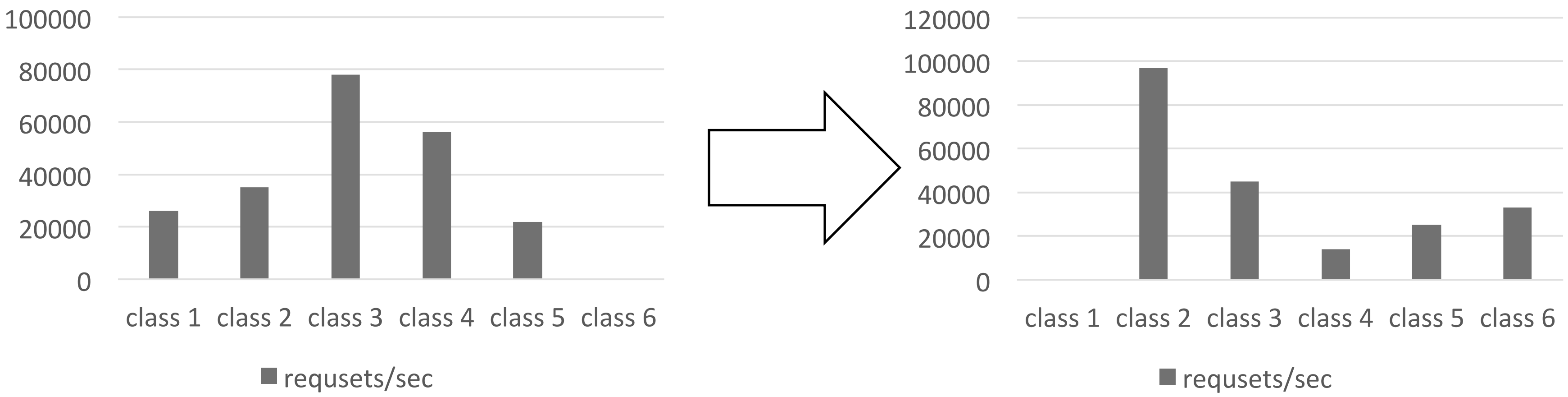
Optimal Allocation

Trace:	a	1	b	2	c	3	a	4	b	5	c	6	a	7	b	8	c	9
Class 1 slabs :	1	1	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Class 2 slabs :	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
hits :	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0

Total hits: 6

Slab Calcification

- The slab allocation is decided by the reference pattern in cold start period.
- When the workload behavior changes, slab allocation cannot adapt to the change in reference pattern.
- The cache performance will drop.



Existing Solutions

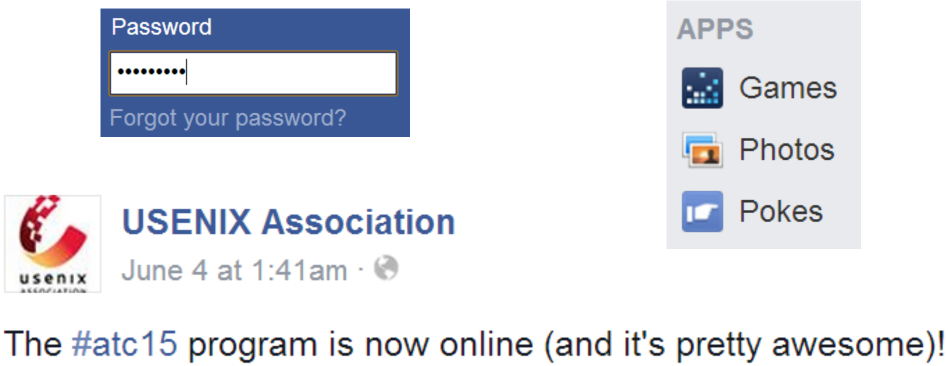
- Automove
  - Move a slab from a class with no evictions to one with the highest number of evictions in three consecutive monitoring windows(30 seconds).
- Twemcache (By Twitter)
  - Random slab eviction aims to balance the eviction rates among all classes.
- Periodic Slab Allocation (PSA) (ICC’14)
  - Move a slab from the class with the lowest risk to the class with the largest number of misses.
- Facebook Policy (NSDI’13)
  - Balance the age of the least recently used items among all classes, effectively approximating global LRU.

Locality-aware Memory Allocation (LAMA)

- Motivation
- Miss Ratio Curve
- Footprint Theory
- Minimal Miss Ratio
- Minimal Average Request Time

Motivation

- Why demand-driven allocation may not deliver best performance?
  - Different classes of data objects show different reference locality.



Password

Forgot your password?

APPS

- Games
- Photos
- Pokes

USENIX Association

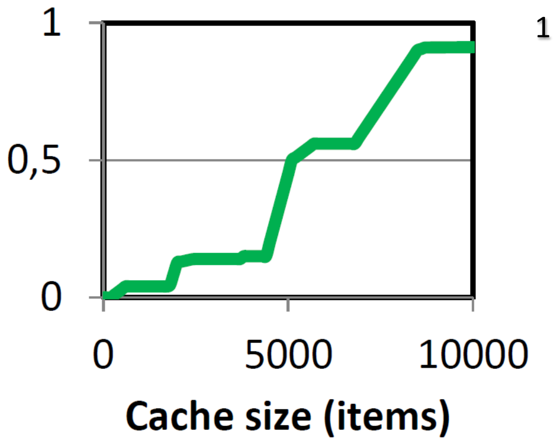
June 4 at 1:41am ·

The #atc15 program is now online (and it's pretty awesome)!

  - Some classes of data may be frequently requested but others not.
  - Allocating more slabs to cache frequently used data will increase cache performance.
- Existing solutions have been motivated by the same observation, but their performances are far from optimal.

Miss Ratio Curve

- What metric can be used to accurately describe data reference pattern?
  - Miss ratio curve (MRC) or Hit ratio curve (HRC).



- How to profile MRC online for each classes with low overhead?
  - Use footprint theory [PACT'11, ASPLOS'13]

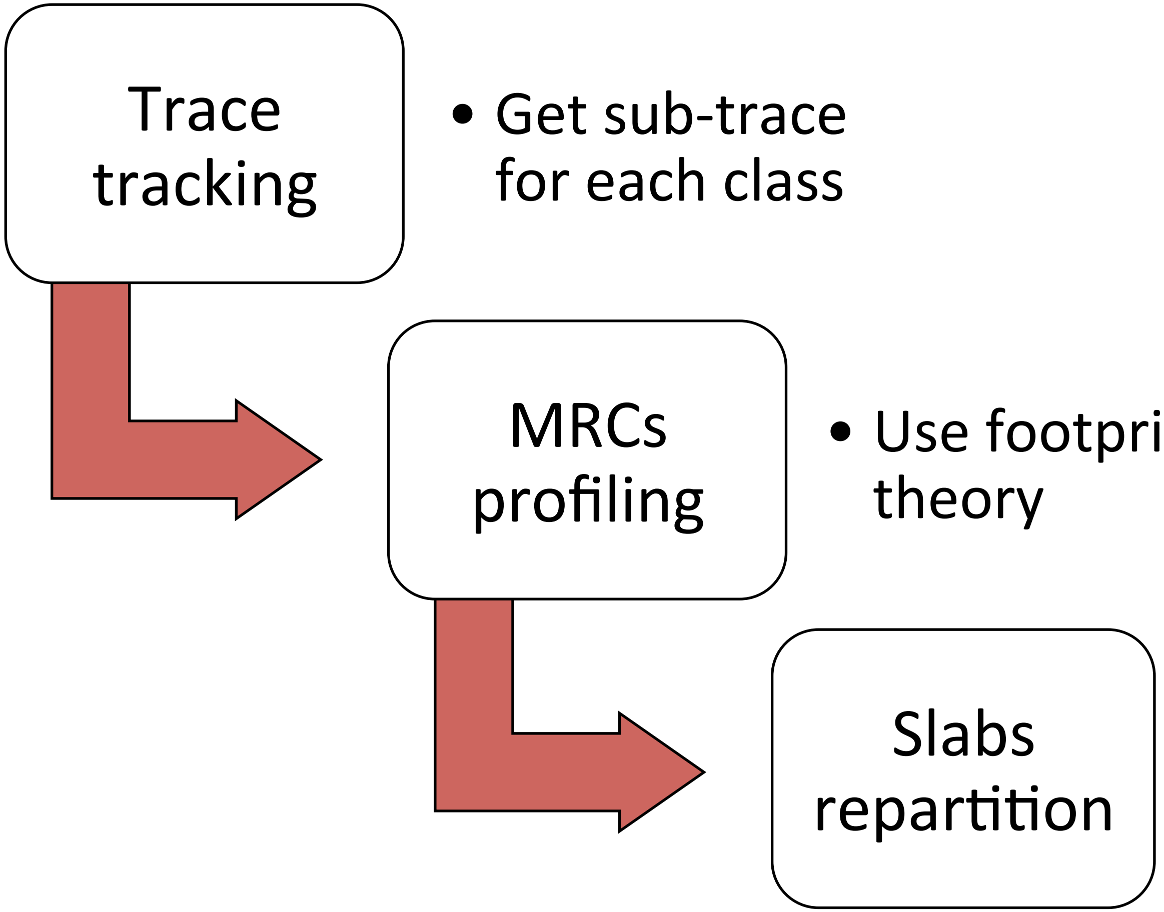
1. Dynamic performance profiling of cloud caches. In Proceedings of the 4th annual Symposium on Cloud Computing, page 59. ACM, 2013.

Footprint Theory

- Footprint is the number of data objects referenced in a giving trace.
- The footprint $fp(t)$ stands for the average data usage in any time window of length t in this trace.
- $fp(t)$ can be measured by a linear time algorithm.
- With $fp(t)$ distribution, the miss ratio of cache size x can be represented as the fraction of reuses that have an footprint larger than x.

$$MRC(x) = 1 - \frac{\sum_{\{t|fp(t)<x\}} r_t}{n}$$

LAMA Design



Trace tracking

- Get sub-trace for each class

MRCs profiling

- Use footprint theory

Slabs repartition

- Use dynamic programing

Minimal Miss Ratio

- How to use MRCs to find the best allocation for minimal miss ratio:
  - S_i : the number of slabs in class i.
  - I_i : the number of items per slab in Class i.
  - R_i: the number of requests for Class i.
  - MR_i: the miss ratio of class i.
  - The system miss ratio would be:

$$Miss\ Ratio = \frac{\sum_{i=1}^{n} R_i * MR_i}{\sum_{i=1}^{n} R_i} = \frac{\sum_{i=1}^{n} R_i * MRC_i(S_i * I_i)}{\sum_{i=1}^{n} R_i}$$

Minimal Average Request Time

- How to use MRCs to find the best allocation for minimal average request time (ART)?
  - $T_h(i)$: the average request hit time for Class i.
  - $T_m(i)$: the average request miss time (including retrieving data from database and setting back to Memcached).
  - The average request time ART_i of Class i now can be presented as:

$$ART_i = MR_i * T_m(i) + (1 - MR_i) * T_h(i)$$

  - The overall ART of the system is:

$$ART = \frac{\sum_{i=1}^{n} R_i * ART_i}{\sum_{i=1}^{n} R_i}$$

Slabs Repartition

- Each M references:
  - Calculate the best allocation according to the data locality measured as MRCs in this period.
  - Repartition only if the theoretical miss ratio difference between the new allocation and original allocation is above a certain threshold.
  - At each repartitioning, we choose at most N slabs with the lowest risk do reassigning.
- If the number of all slabs is MAX and there are n classes. The size of the solution space is MAX^n.
- We use dynamic programing to find the best allocation and the time complexity is $O(n * MAX^2)$.

Evaluation

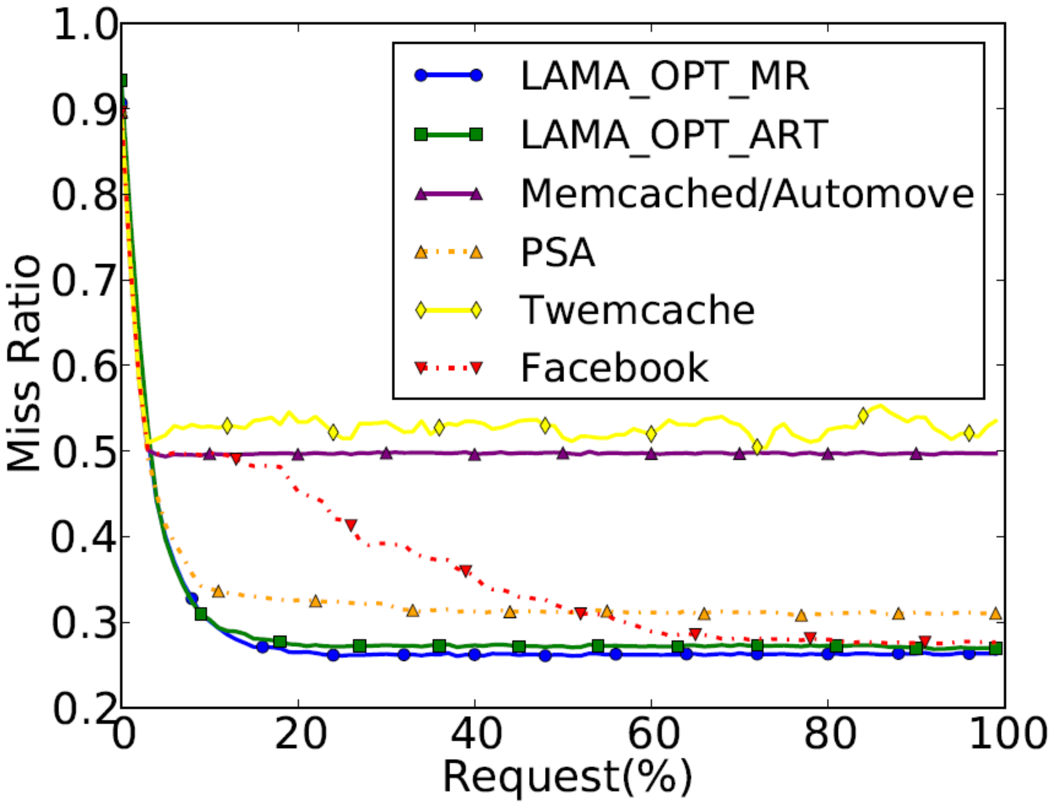
- We have implemented LAMA in Memcached-1.4.20.
- Experimental Setup:
  - Intel(R) Core(TM) I7-3770 with 4 cores, 3.4GHz, 8MB shared LLC.
  - 16GB memory.
  - Fedora 18 with Linux-3.8.2.
  - 4 server threads, one Memcached server.

Workloads

- The Facebook ETC workload to test the steady-state performance.
  - A general-purpose workload with the highest miss ratio in all Facebook's Memcached pools.
  - Generated by Mutilate.
  - 50 million requests to 7 million data objects.
- A 3-phase workload to test dynamic allocation.
  - Used to evaluate PSA.
  - 200 million requests to data items in two working sets, each of which has 7 million items.
  - Each phase has a different access pattern.
- A stress-test workload to measure the overhead.
  - Use the Memaslap generator of libmemcached.
  - To test the throughput of a given number of server threads.

Facebook ETC Miss Ratio

- Miss ratio from cold-start to steady state(512MB).



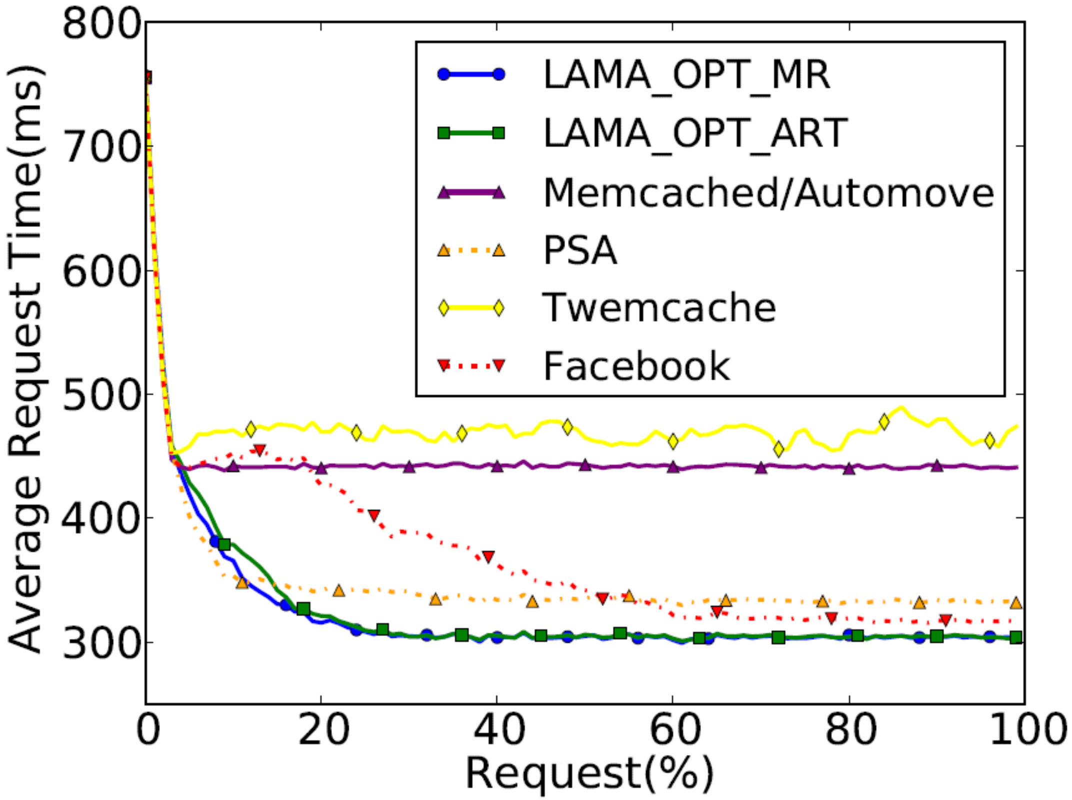
Observation:

LAMA_OPT_MR is

- 47.20% lower than Memcached.
- 18.08% lower than PSA.
- 5.40% lower than Facebook.

Facebook ETC Average Response Time

- Average request time from cold-start to steady state (512MB).



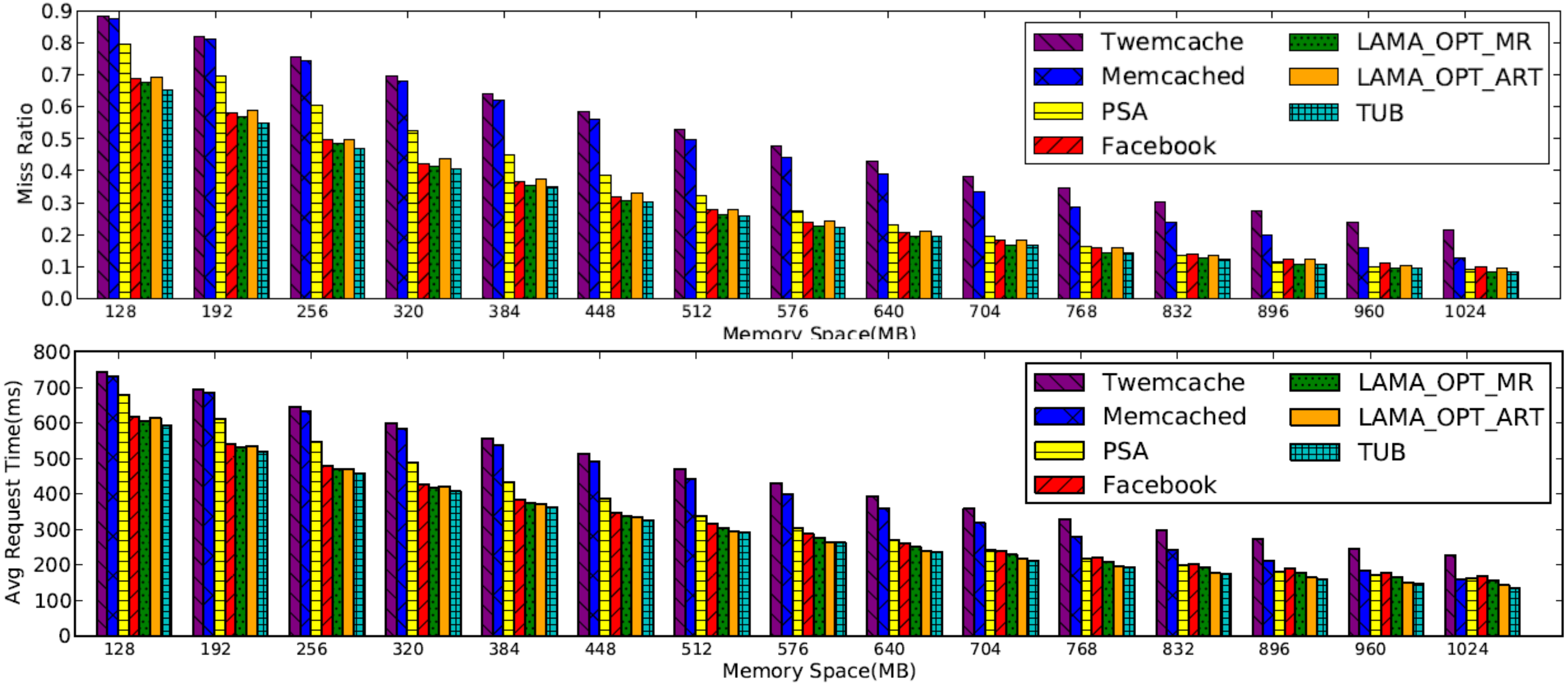
Observation:

LAMA_OPT_ART is

- 33.45% lower than Memcached.
- 13.17% lower than PSA.
- 6.70% lower than Facebook.

Facebook ETC Upper Bound Performance

- Steady-state miss ratio using different amounts of memory
- Theoretical Upper Bound (TUB): Using real MRCs measured by the full-trace reuse distance tracking.



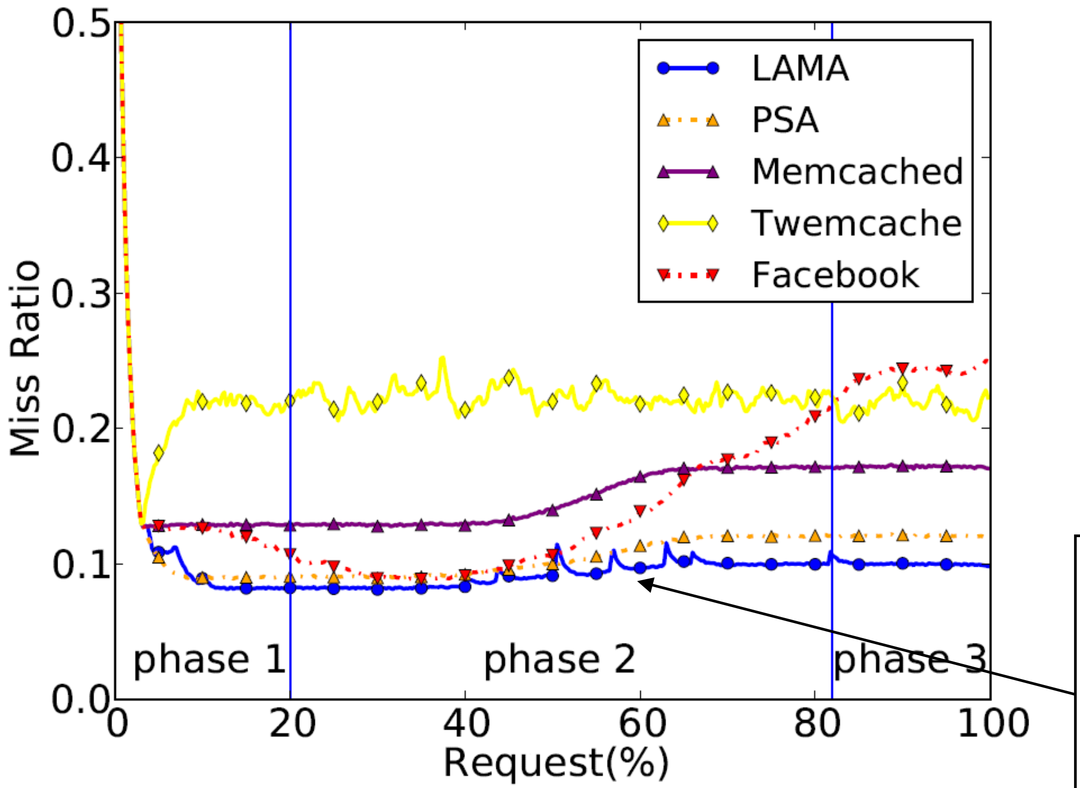
Facebook ETC Upper Bound Performance

- Conclusion (compared with Default Memcached):

	TUB	LAMA	FACEBOOK	PSA	Automove	Twemcache
Miss Ratio reduction	42.8% (25.5%–50.3%)	41.9% (22.4%–46.6%)	37.6% (21.0%–47.1%)	31.7% (9.1%–43.9%)	0%	-16.93% (-65.95%–0.90%)
Average request time reduction	28.3% (15.6%–34.4%)	26.4% (10.7%–33.9%)	19.9% (-0.5%–29.2%)	16.3% (2.0%–24.8%)	0%	-12.95% (-41.69%–-1.47%)

Slab Calcification

- Miss ratio over time by different policies (3-phased workload, 1024M).

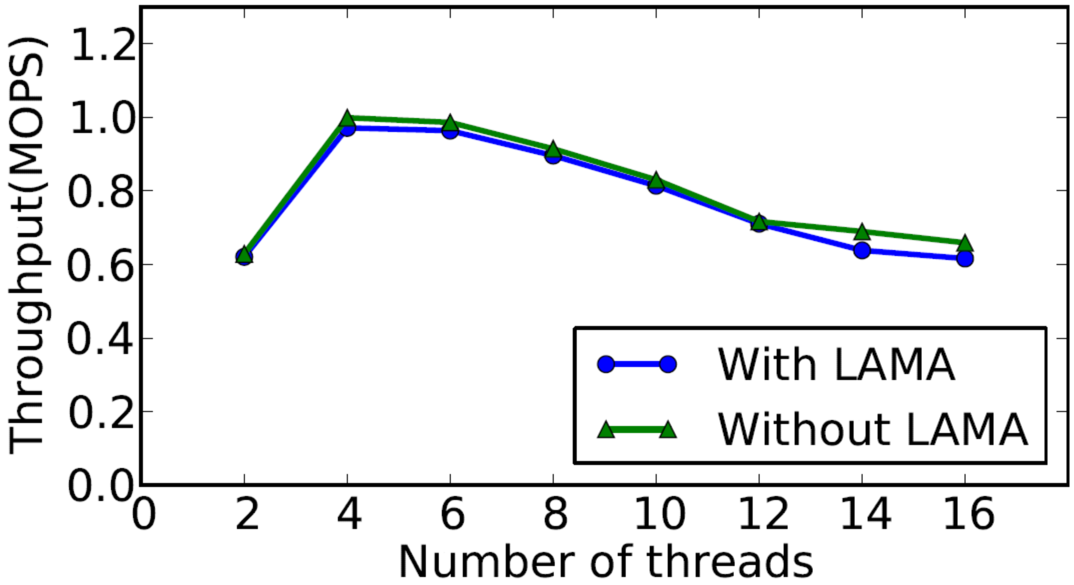


Observation: LAMA outperforms other techniques in each phase.

Dynamic allocation based on the previous access pattern.

LAMA Overhead

- Overall throughput as different number of threads are used (stress test workload)



Observation:
Average degradation of LAMA is only 3.14%.

Summary

- Compared with the default Memcached:
  - LAMA reduces the miss ratio by 42% using the same amount of memory.
  - LAMA achieves the same memory utilization (miss ratio) with 41% less memory.
  - LAMA outperforms four previous techniques in steady-state performance, convergence speed, and the ability to adapt to phase changes.
  - LAMA is close to optimal, achieving over 98% of the theoretical potential (TUB).

PEKING
UNIVERSITY

Thank you for your attention!

Q&A

Email: hxm@pku.edu.cn