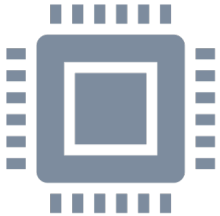
UCLA Samueli School of Engineering

MemLiner: Lining Up Tracing and Application for a Far-Memory-Friendly Runtime

Chenxi Wang*, **Haoran Ma* (co-first author)**, Shi Liu, Yifan Qiao,
Jonathan Eyolfson, Christian Navasca, Shan Lu, Guoqing Harry Xu

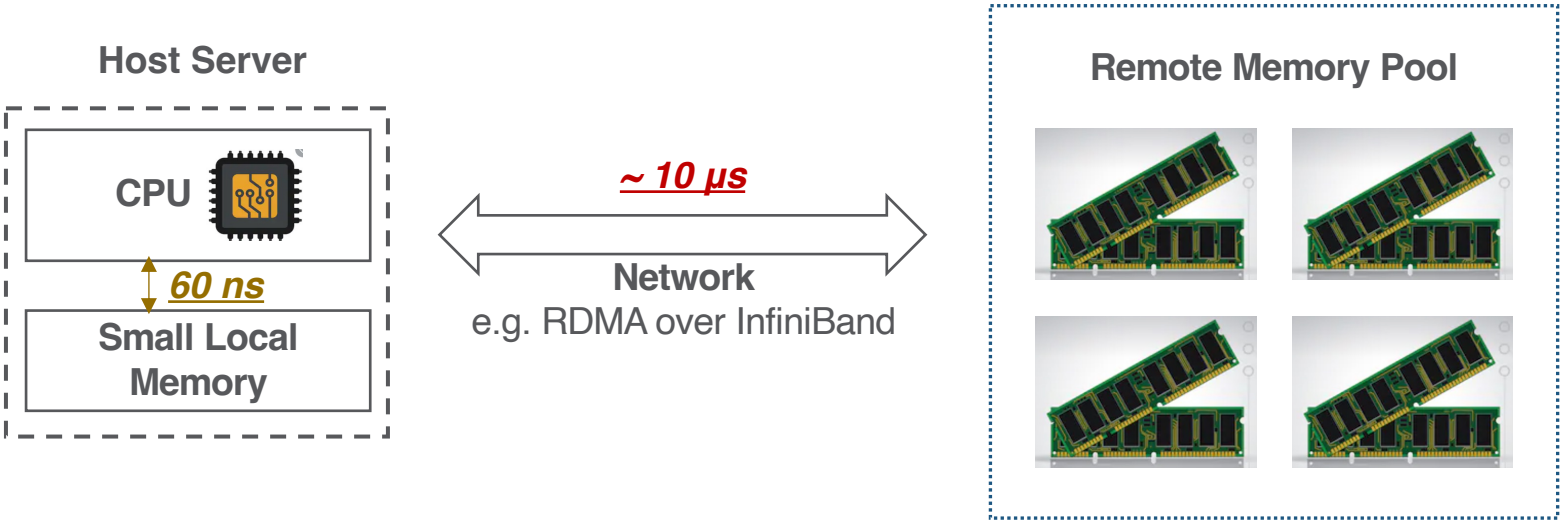
Memory Capacity Bottleneck in Datacenters



Growing imbalance between processor computation and memory capacity

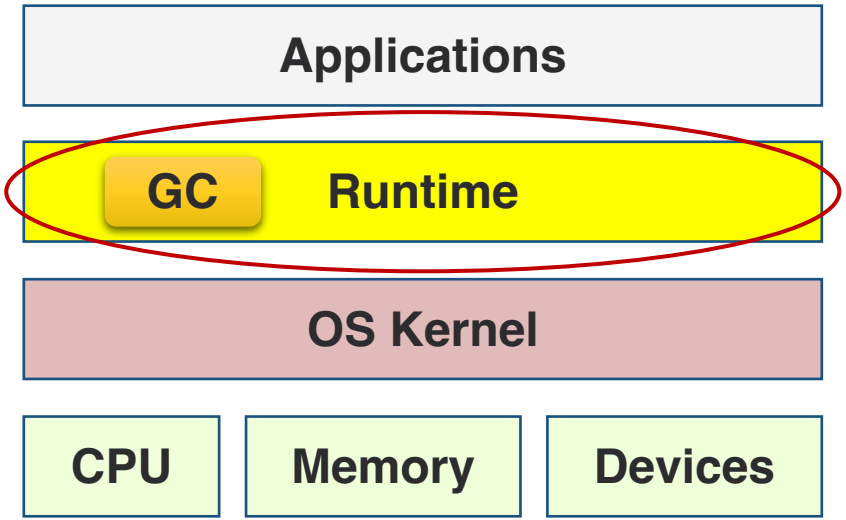
Memory underutilization in datacenters

Far-Memory System

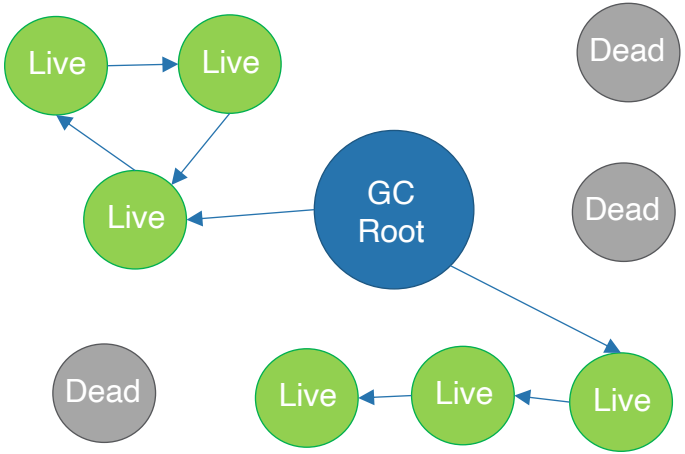


High-level Languages

Applications written in high-level languages are dominant in datacenter workloads.



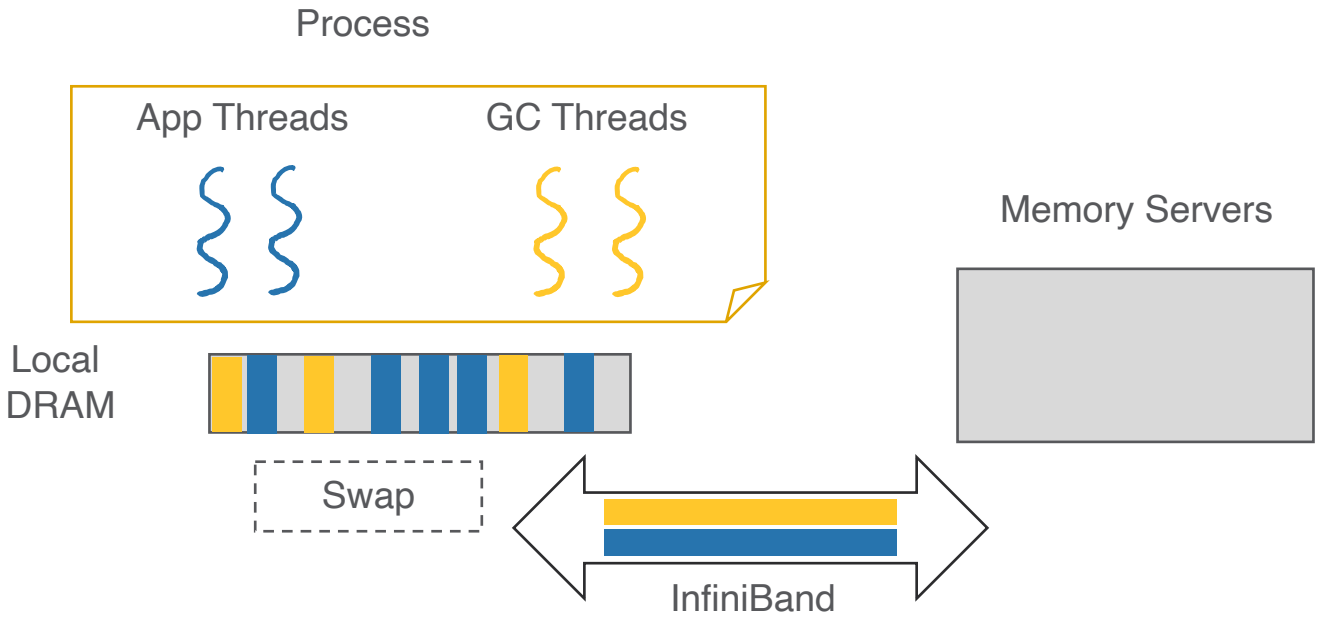
Garbage Collection



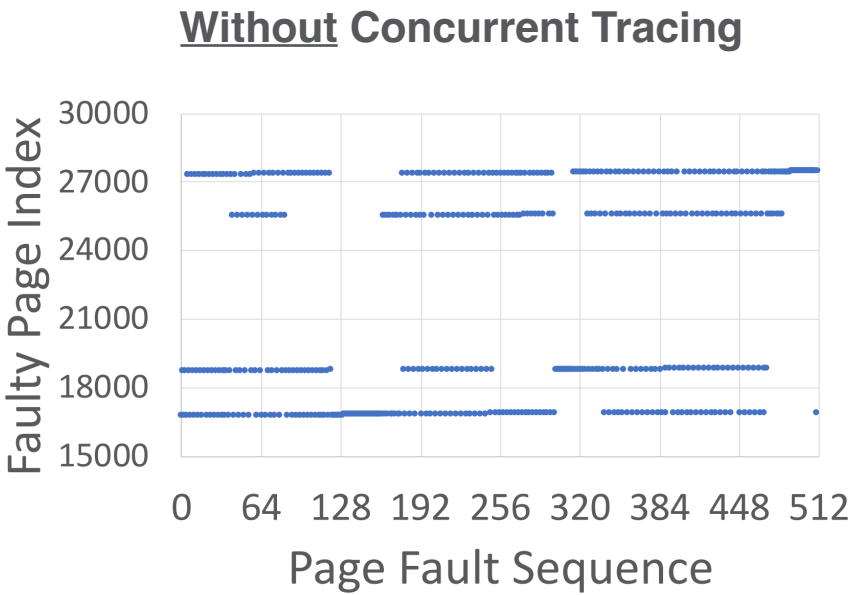
Tracing is done concurrently with applications

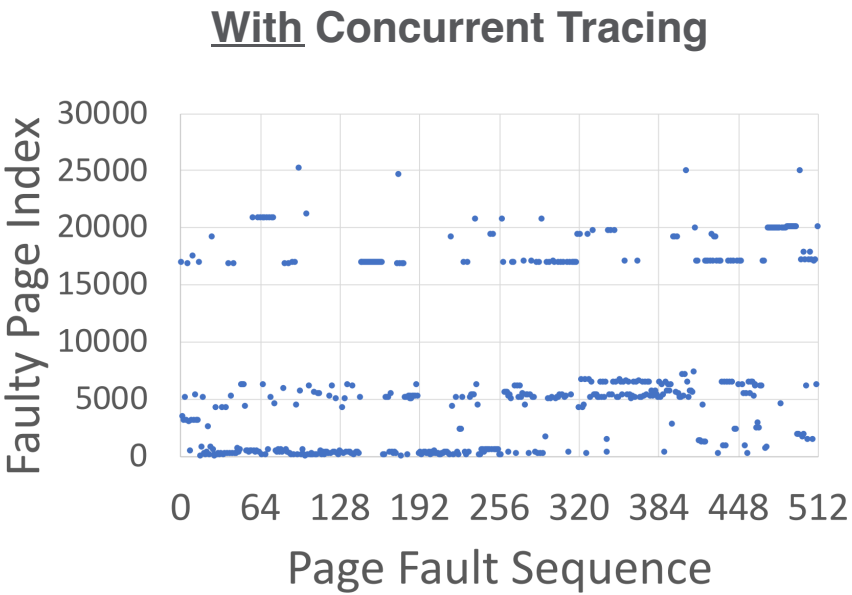
Local Mem Ratio	25%	13%
Slowdown	2.6x	3.4x

Resource Competition

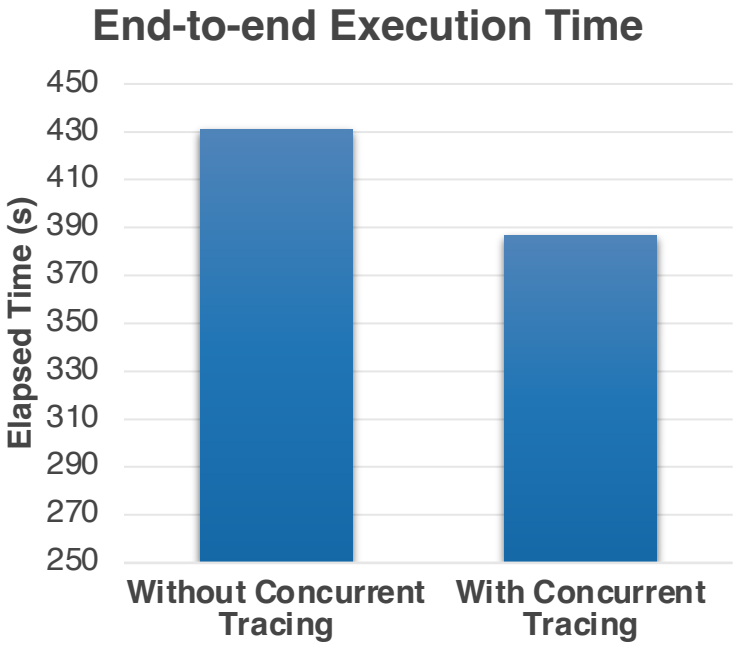


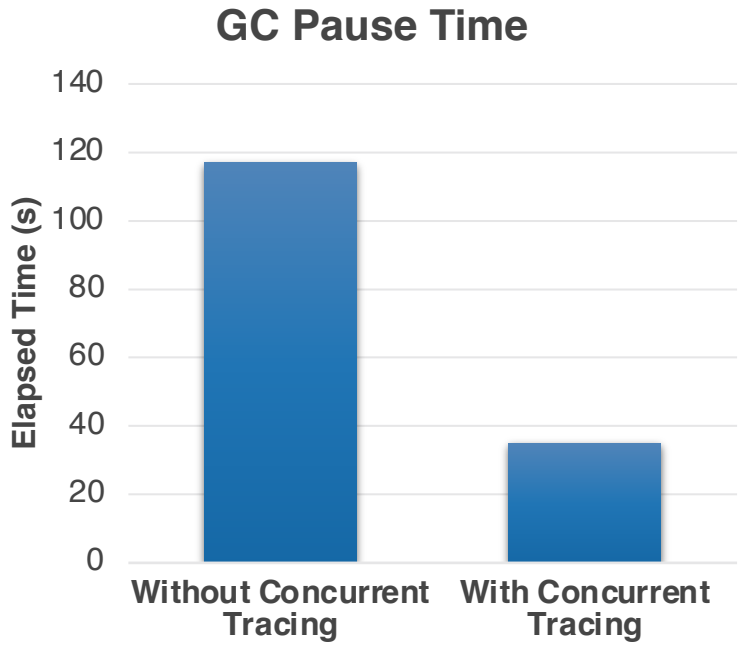
Ineffective Prefetching





Can we disable concurrent tracing?





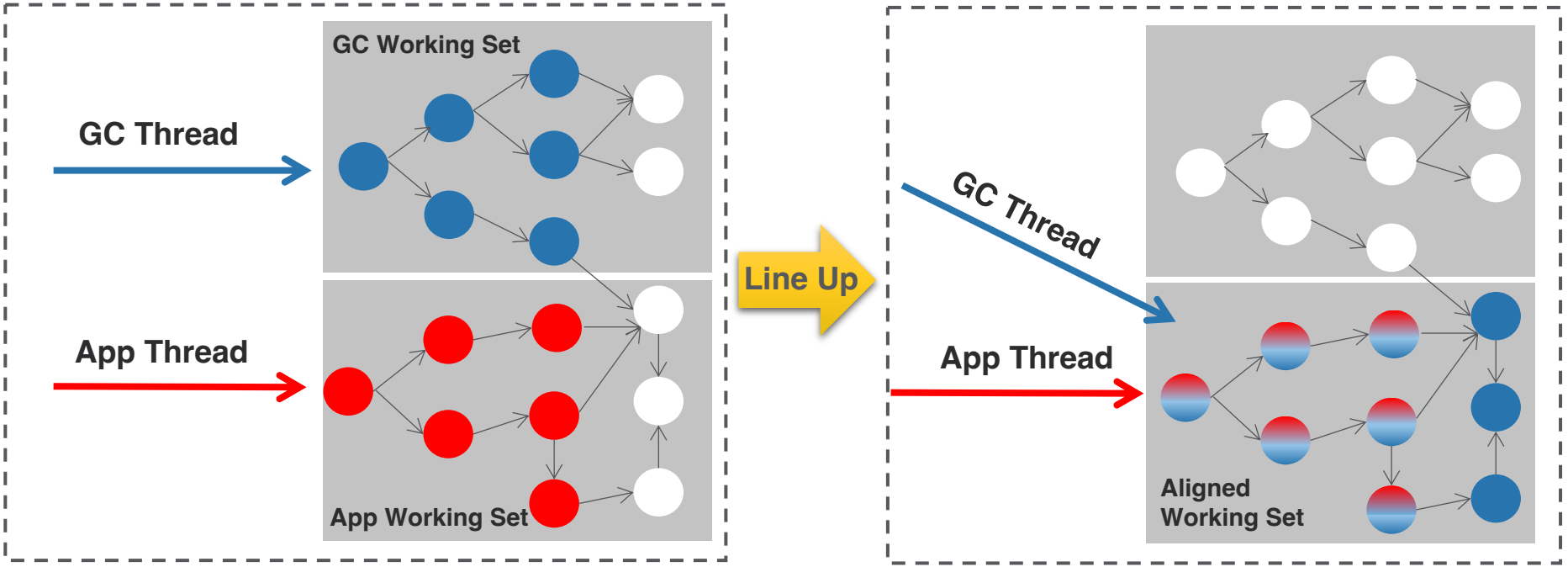
Are application and garbage collection completely unrelated?

Observations

1. Application and GC are just temporally unaligned

2. Changing object access order in GC is possible

Key Design Idea



(a) Current runtime

(b) MemLiner runtime

Object Classification

1. **Local Objects**: Currently being accessed by application threads
   - GC threads should touch
2. **Incoming Objects**: In remote memory, will soon be accessed by app threads
   - GC threads should touch
3. **Distant Objects:** In remote memory, will **not** be accessed by app threads soon
   - GC threads should **delay** the access

Challenges in Classifying Objects

Local

How to inform GC threads accessed objects

Incoming

What kind of objects will be accessed by app threads soon

Distant

How to estimate the location of objects

Barriers

Read Operation

Pre-read Barrier

$a = b.f$ or $a = b[i]$

Post-read Barrier

Write Operation

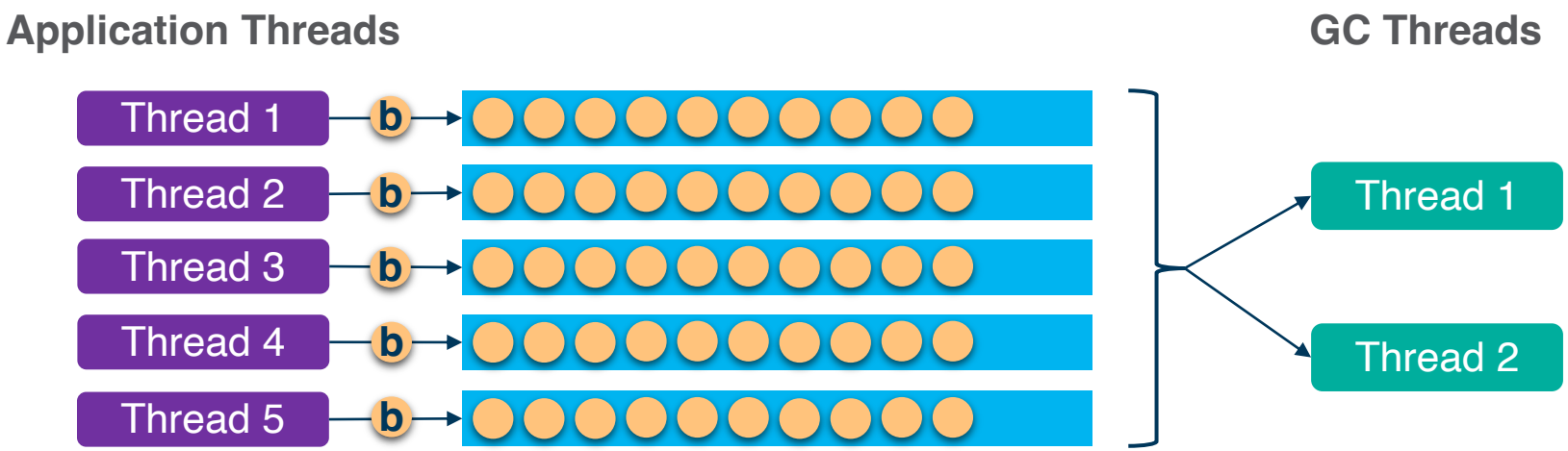
Pre-write Barrier

$b.f = a$ or $b[i] = a$

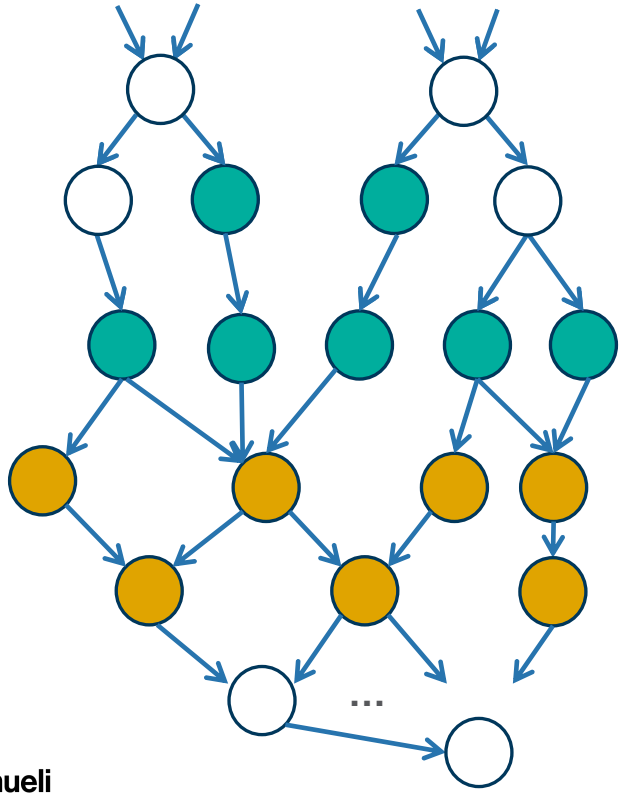
Post-write Barrier

Local Objects

$$a = \boldsymbol{b}.f$$
$$\boldsymbol{b}.f = a$$



Incoming Objects



Currently being accessed by app

In remote memory, used by app soon

Distant Objects

Counter maintained by kernel

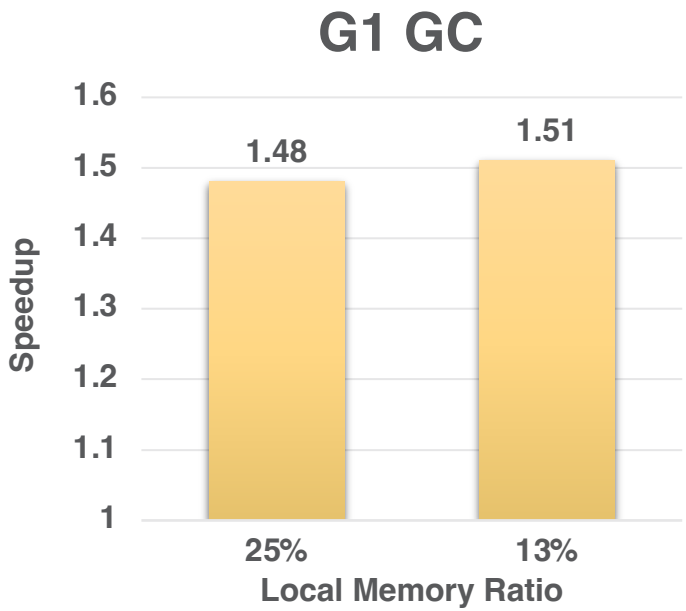
Object reference maintained by runtime

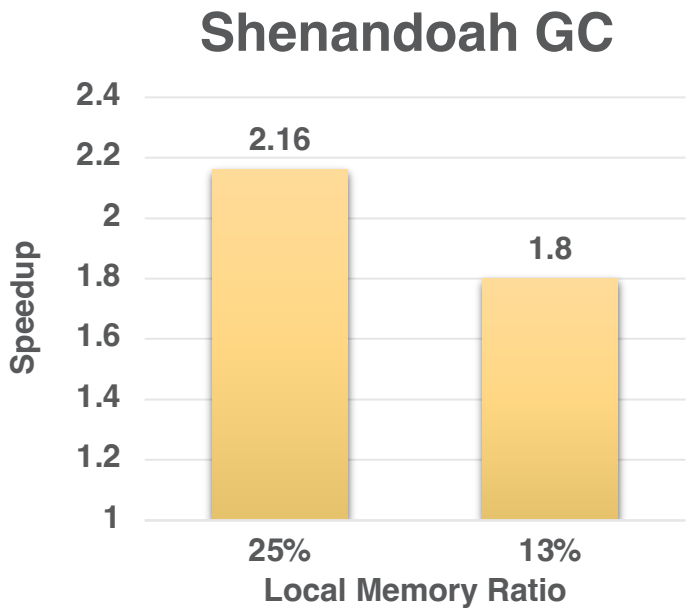
$$Diff(\text{Current Epoch}, \text{Timestamp (4 bits)}) < \delta$$

Benchmarks

- MemLiner is implemented in two widely-used garbage collectors:
  - G1 GC
  - Shenandoah GC
- Evaluated MemLiner on ***12 workloads*** using a range of local memory ratios
- MemLiner is run on two swap systems: Fastswap and Leap

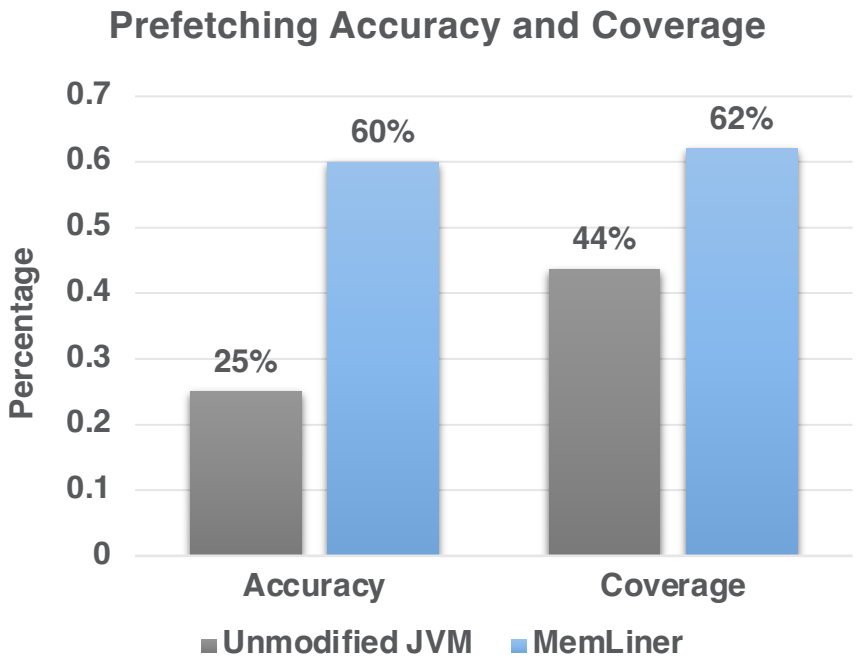
Results: Throughput



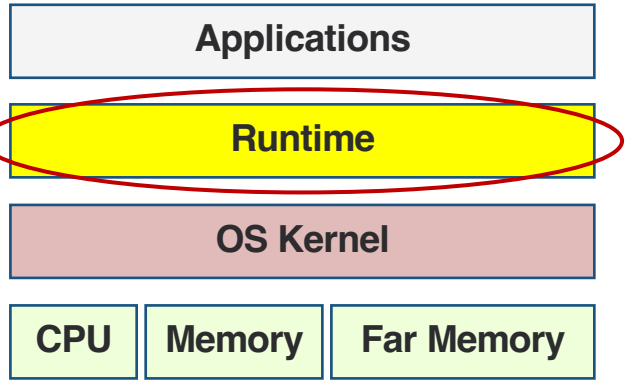


Results: Prefetching Effectiveness

- An average of **1.6x** speed up under 25% local memory on **Leap**
- Reduces 58% of on-demand swap-ins, and 53% of total swap-ins on average.



Key Takeaways



- Runtime should also be taken into consideration when hardware changes
- Runtime serves as a semantic bridge between application and underlying OS/hardware architecture

Thank you! Code at https://github.com/uclasystem/MemLiner.

Q&A