I. Introduction and Motivation

- Once a page is mapped in memory, the kernel isn't notified of subsequent accesses to it.

- To determine page temperature, the kernel must regularly scan all mapped pages checking their referenced bits.

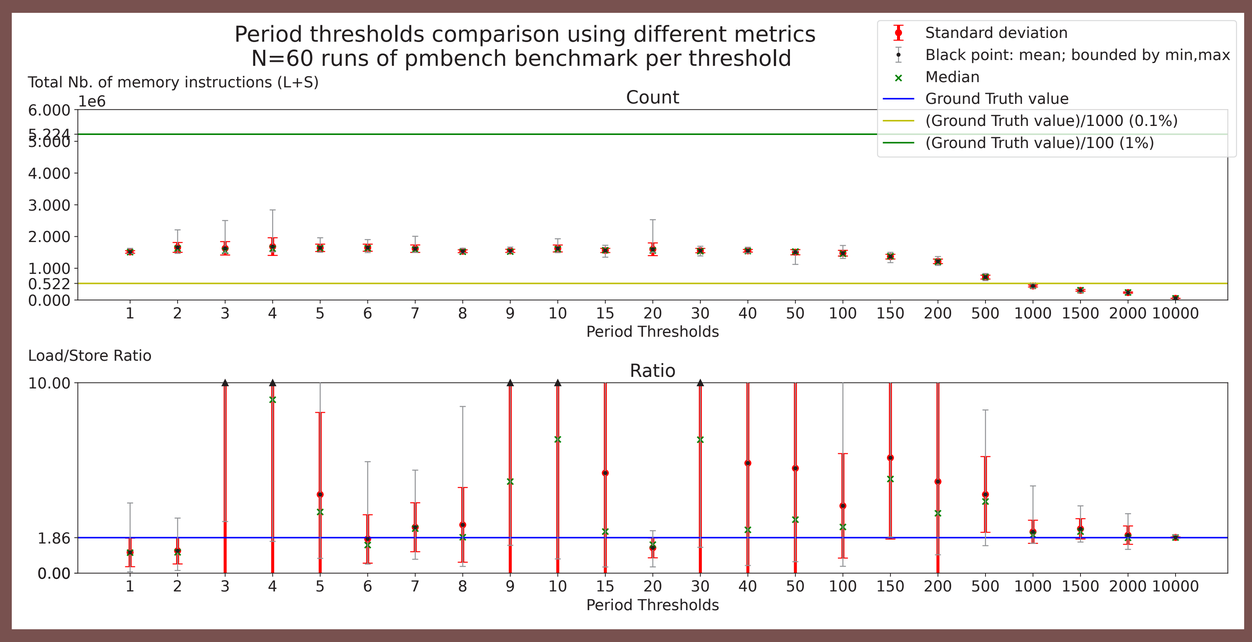
- Recently, **hardware sampled memory traces** have become more accurate and complete.

We can use them as input for **online page eviction algorithms** to achieve better accuracy at low overhead!

II. Background

How can we sample memory traces? **PEBS** on Intel counts events and periodically samples data in memory. We can use it to get memory traces easily!

→ How does sampling period influence memory trace completeness?



→ Higher threshold is more workload resemblant, but yields less data

EVALUATING PARTIAL MEMORY TRACES FOR PAGE TEMPERATURE DETECTION

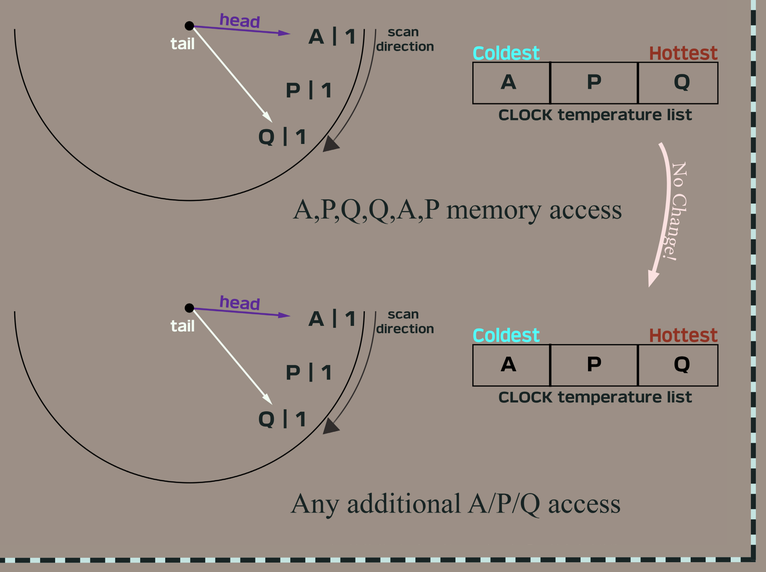
Victor Garvalov

Shaurya Patel

Alexandra Fedorova

III. Evaluation Method

Observation: Some page caching algorithms might yield **same output**, although they are given **more information**:



Idea:

- Simulate different algorithms with different levels of memory trace completeness

- Monitor how each performs (# page faults) and

- See how each individual additional memory trace impacts the algorithm by normalizing its internal representation of the page cache (to a "temperature list")

- **Compare** amongst each other using **Manhattan Distance**.

Benchmarks:
- STERAM
- PMBENCH

Simulated algorithms:
1. LRU-2
2. GCLOCK (i=2)
3. ARC
4. CAR

IV. Future Work

- Evaluate the impact of memory trace completeness using CPU-intensive benchmarks and real-life applications

- Explore the feasibility and benefits of PEBS sampling directly in the kernel to obtain more samples at a lower overhead cost

- Test our approach with state-of-the-art nearly-optimal algorithms designed for databases systems

- Develop a **Set-Duelling algorithm** that dynamically selects the best strategy (partial memory traces vs. reference bits) based on current memory pressure