Analyzing the Benefits of More Complex Cache Replacement Policies in Moderns GPU LLCs

Jarvis Jia and Matthew D. Sinclair
University of Wisconsin-Madison
jia44@wisc.edu ; sinclair@cs.wisc.edu

I. MOTIVATION

The gem5 simulator offers Classic and Ruby as two separate memory models for simulating on-chip caches. The Classic model, which originated from M5 [1], is a quick and simple option that allows for easy configuration, but only supports a basic MOESI coherence protocol. On the other hand, the Ruby model, which was developed by GEMS [2], is a more advanced and flexible option that can accurately simulate a wider range of cache coherence protocols and features [3], [4]. However, choosing between the two memory system models in gem5 is challenging for researchers as each has advantages and limitations which can be inconvenient. In particular, this has led to a bifurcation of effort where prior work has added replacement policies to Classic and Ruby in parallel – duplicating effort unnecessarily and preventing users from using a desired replacement policy if it is not implemented in the desired memory model (e.g., users could only use RRIP [5] in Classic).

Accordingly, we merged the cache replacement policies from Classic to Ruby, enabling users to use any of the replacement policies in either memory model. Gem5 currently has the capability to support 13 replacement policies, which can be used exchangeable within the Classic and Ruby cache models, including commonly used options like LRU, FIFO, PseudoLRU, and different types of RRIPs [6]. After combining the replacement policies for the Classic and Ruby cache models, we designed and integrated (into gem5’s nightly regressions) multiple corner case tests to verify and ensure the continued correct functionality of these policies [7]. Through these tests, we identified and fixed several bugs [8] [9] to ensure that the replacement policies operate correctly. Finally, with the newly enabled and verified functionality, we decided to use gem5 to study these policies in a GPU context. Specifically, we study GPU L2 caches, since GPU L1 caches are often used to stream data through and thus are unlikely to be significantly impacted by replacement policy.

II. METHODOLOGY

We used an AMD Vega 10 as the target GPU [10], [11], and studied various L2 cache sizes (256 KB - 512 MB) for all 13 replacement policies. To compare the impact of changing L2 cache size and replacement policies, we examine total GPU cycles and GPU L2 hit rates. We tested the replacement policies with the Rodinia benchmarks [12], [13]. For example,

III. CONCLUSION

Cache replacement policy plays an important role for memory design and optimization. It has a significant impact on the cache system’s hit rate and access latency, which has led to extensive efforts in both academia and industry to improve its effectiveness [14]. Despite the critical importance of cache replacement policies for efficient memory hierarchy design, there is limited research on their impact on GPUs. Our results show that, for write-through GPU LLC, caches replacement policy does not significantly impact overall results. However, we expect that write-back LLC caches and other, more cache sensitive GPU applications, will show larger benefits.
REFERENCES


