Leveraging Page Size Information to Enhance Data Cache Prefetching

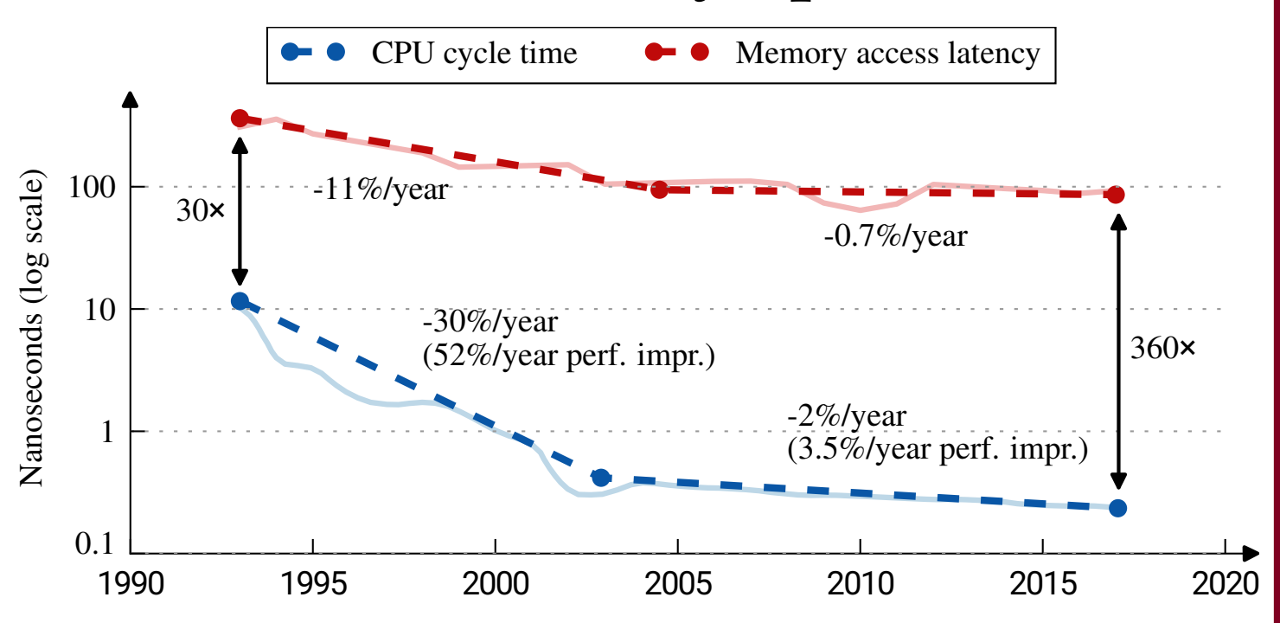
Georgios Vavouliotis[1,3], Gino Chancon[2], Lluc Alvarez[1,3], Paul V. Gratz[2], Daniel A. Jiménez[2], and Marc Casas[1,3]

[1]*Barcelona Supercomputing Center* [2]*Texas A&M University* [3]*Universitat Politècnica de Catalunya*

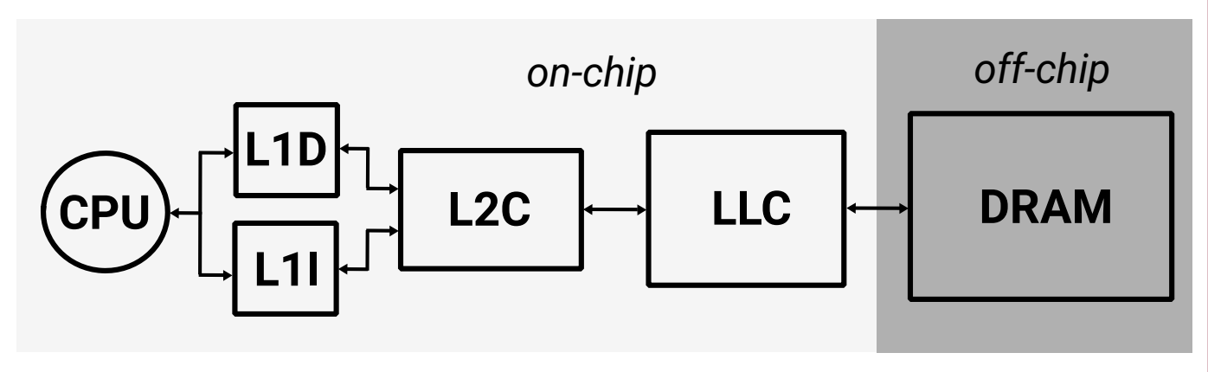
Memory Bottleneck

CPU vs RAM Speeds

- Discrepancy between processor and main memory speeds



- Architects use on-chip cache hierarchies to reduce the latency cost of accessing main memory



- On-chip caches partially reduce main memory accesses due to limited capacity

Cache Prefetching

Fundamental Idea

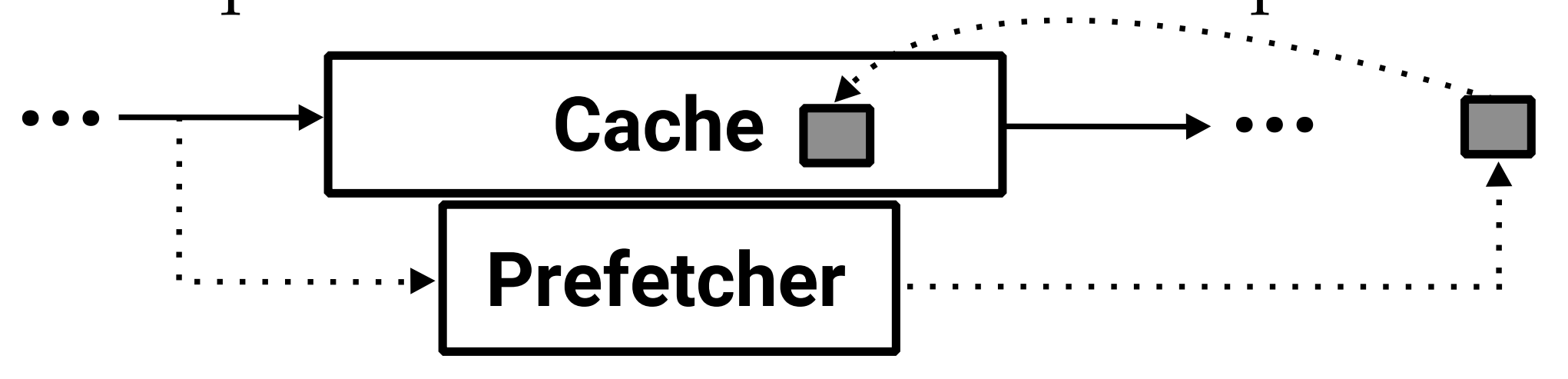
- Proactively fetch data blocks into the on-chip caches before they are explicitly requested

Why?

- HPC and Big Data workloads feature massive data footprints that do not fit in the on-chip caches

Cache Prefetching in Practise

- Prefetching can be applied on all cache levels
- On a cache access, the prefetcher takes as input the requested block address and issues prefetches



- Today, all HPC chips employ different data cache prefetchers to capture heterogeneous access patterns (e.g., Intel IceLake, AMD Zen)

Virtual Memory Sub-System

Overview & Page Size

- Modern systems implement paging-based virtual memory
- The standard page size is 4KB in most systems
- Modern OSes provide support for larger page sizes (e.g., 2MB and 1GB pages)

Prior Work on Cache Prefetching & Opportunities

- Numerous cache prefetchers have been proposed in recent literature [1-3]
- All prior works propose cache prefetchers that use complex prefetching algorithms to capture more distinct patterns than the previously proposed designs

Common Aspects of Cache Prefetchers

- Prior cache prefetchers that operate on the physical address space assume 4KB pages
- These cache prefetchers do not permit prefetching beyond 4KB physical page boundaries because physical contiguity is not guaranteed

Opportunity for Improving Cache Prefetching

- Modern systems vastly use larger page sizes to reduce address translation overheads
- Physical pages are equally sized with the virtual pages (e.g., when a virtual page is 2MB the corresponding physical page is also 2MB)
- Intuitively, limiting cache prefetchers to prefetch within 4KB boundaries when larger page sizes are used results in sub-optimal performance gains

Physical Machine Measurements

- Cache intensive SPEC'06, SPEC'17, and GAP workloads on an Intel Xeon machine
  - More than 80% of the allocated pages are 2MB pages for these workloads

Objective of this Work

- Enhance the performance of all prior and new spatial cache prefetchers that operate on the physical address space by exploiting the page size information

Methodology

- ChampSim simulator (L1D: 48KB, L2C: 512KB, LLC: 2MB, DRAM: 8GB)
- Workloads: 14 SPEC'06, 12 SPEC'17, 12 GAP, and 49 Qualcomm traces from CVP-1 contest

Evaluated L2 Cache Prefetchers

- SPP [1] • VLDP [2] • BOP [3]

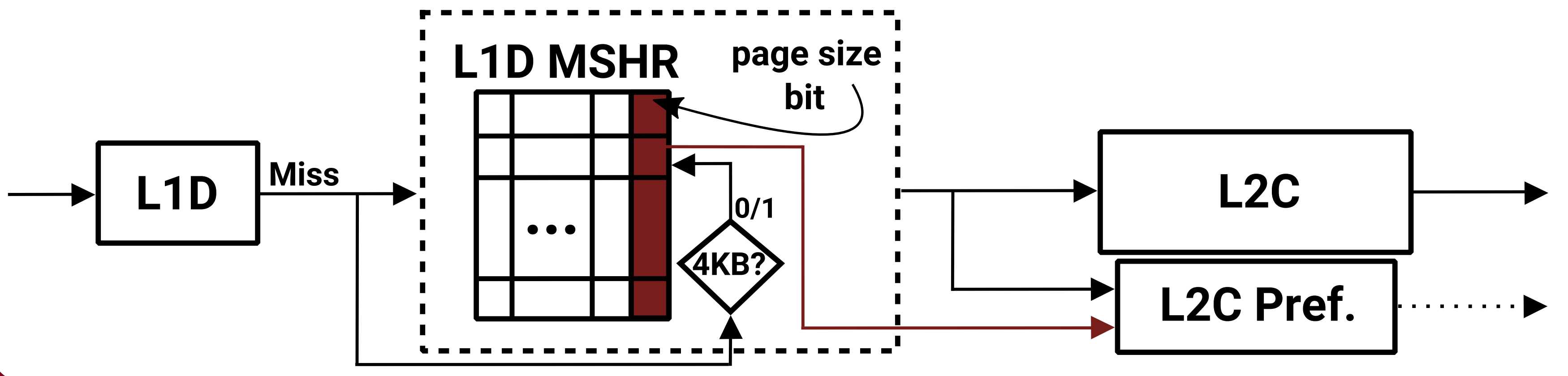
Baseline

Performance improvement is computed over a baseline without prefetching at any cache level

Design & Evaluation

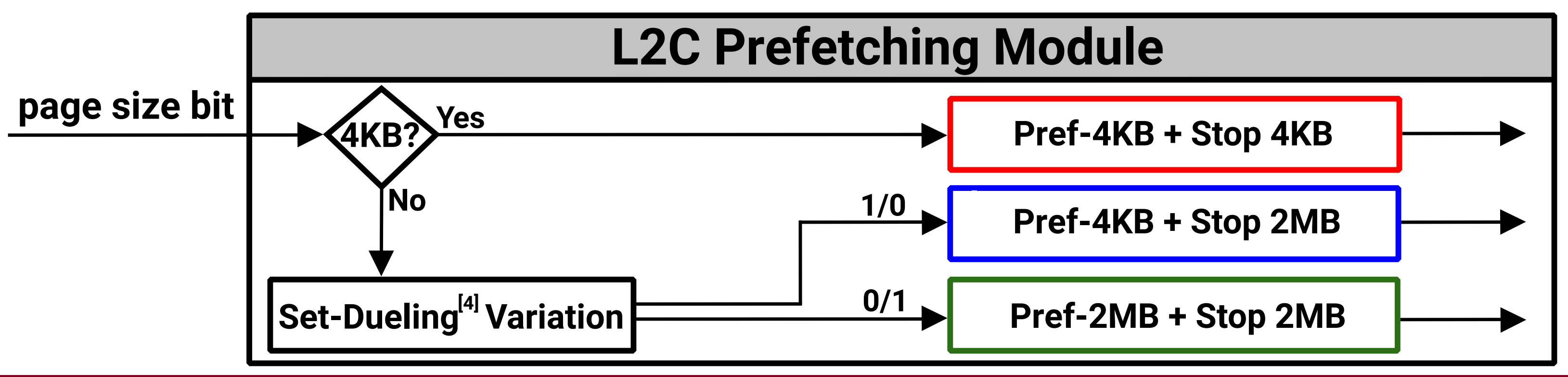
Page Size Propagation Scheme

- L1D Caches are VIPT –> On L1D misses page size is known
- L2C prefetchers are engaged on L1D misses
- Enhance L1D MSHR with one bit, indicating the page size



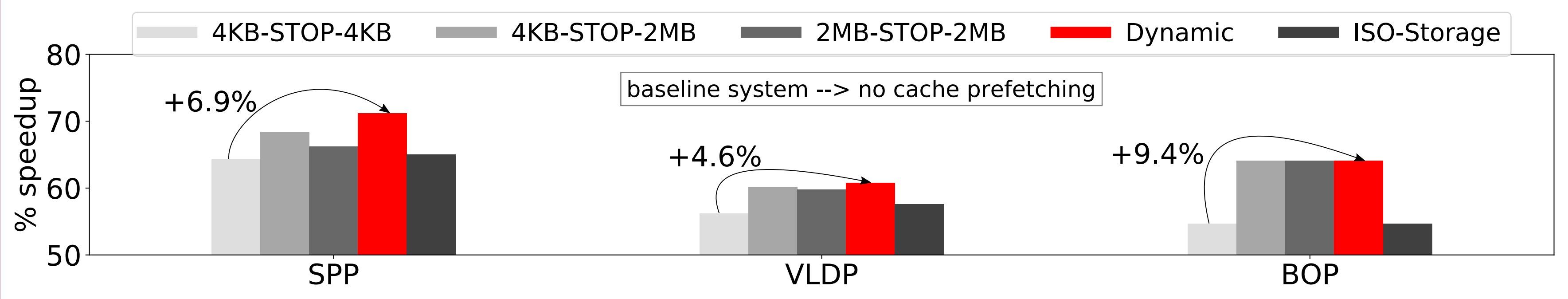
Exploiting the Page Size Information for Improving L2C Prefetching

- The L2C prefetching module consists of two engines
  - Pref-4KB: L2C prefetcher that considers 4KB pages for its structures
  - Pref-2MB: L2C prefetcher that considers 2MB pages for its structures
    - Pref-4KB and Pref-2MB are identical but consider different page sizes
    - Pref-4KB and Pref-2MB capture different access patterns
- When page size is 4KB, Pref-4KB is always consulted and stops prefetching at 4KB boundaries since physical contiguity is not guaranteed
- When page size is 2MB
  - A Set-Dueling [4] variation activates either Pref-4KB or Pref-2MB
    - If Pref-4KB is activated, it does not stop prefetching at 4KB boundaries (but in 2MB boundaries) since Pref-4KB is aware that the block resides in a 2MB page
    - If Pref-2MB is activated, it stops prefetching at 2MB boundaries



Preliminary Performance Results

- System with only 2MB pages (>80% of the pages are 2MB for these workloads)
- Evaluated versions of each L2C prefetcher (SPP, VLDP, BOP)
  - Pref-4KB-Stop-4KB –> 4KB structures + stop prefetching at 4KB boundaries
  - Pref-4KB-Stop-2MB –> 4KB structures + stop prefetching at 2MB boundaries
  - Pref-2MB-Stop-2MB –> 2MB structures + stop prefetching at 2MB boundaries
  - Pref-Dynamic –> both versions of the L2C prefetcher + Set-Dueling variation
- Geomean speedups presented in the following figure consider the entire workload set



- Simply propagating the page size information (Pref-4KB-STOP-2MB) improves performance over the state-of-the-art approach (Pref-4KB-STOP-4KB) by 4.1% (SPP), 4.0% (VLDP), and 9.4% (BOP) because it enables more timely prefetching
- BOP-2MB and BOP-Dynamic perform the same as BOP-4KB-STOP-2MB since BOP does not store the physical pages in any structure
- Pref-Dynamic provides the largest speedups among the evaluated scenarios; SPP-Dynamic outperforms its standard version (SPP-4KB-STOP-4KB) by 6.9% (geomean)
- Doubling the size of each cache prefetcher (ISO-Storage) provides negligible benefits

Conclusions

- Propagating the page size to L2C prefetchers has potential for large performance gains
- Applicable to all prior and new spatial L2C prefetchers
- Applicable to LLC prefetching by propagating the page size bit through the L2C MSHR
- Potential impact on future industrial microarctitectural designs

[1] J. Kim et al., "Path confidence based lookahead prefetching", MICRO'16
[2] M. Shevgoor et al., "Efficiently prefetching complex address patterns", MICRO'15
[3] P. Michaud, "Best-offset hardware prefetching", HPCA'16
[4] Qureshi et al., "Adaptive insertion policies for high performance caching", ISCA'07