Fetch Me If You Can: Evaluating CPU Cache Prefetching and Its Reliability on High Latency Memory

DaMoN 2025 | June 23 | Berlin, Germany

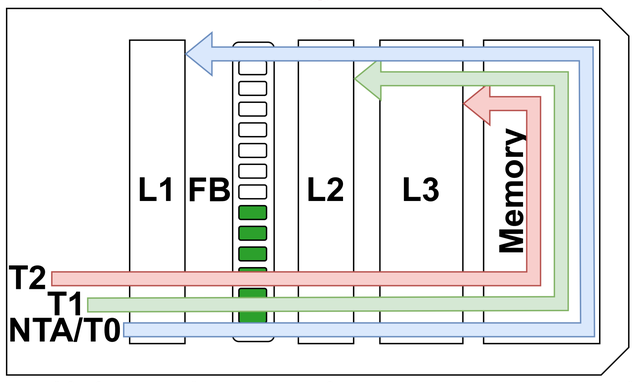
Fabian Mahling, Marcel Weisgut, Tilmann Rabl

Abstract

A larger distance between a CPU core and memory usually results in higher access latency. Software prefetching algorithms claim to hide memory access latencies by moving data to the CPU cache before accessing the data. We analyze to what extent software prefetching can hide increased memory access latencies. We evaluate these on seven systems with different memory technologies and access latencies. We show that prefetching can increase performance by up to 2.6× and 2.8× for B+-Tree and binary search workloads. We find that CPU fill buffers and a workload's memory intensity dictate how much access latency can be hidden. CPUs implement prefetches differently. We introduce microbenchmarks that identify concrete target cache and eviction strategies for different prefetch localities across x86 and ARM architectures. When the fill buffers are full, CPUs either drop prefetches (weak reliability) or halt until all can be executed (strong reliability). We introduce microbenchmarks identifying a CPU's reliability. When prefetching 8 KiB B+-Tree nodes, weak reliability achieves a speedup of 2× while strong reliability degrades performance with a slowdown of 2.5× for lookup workloads.

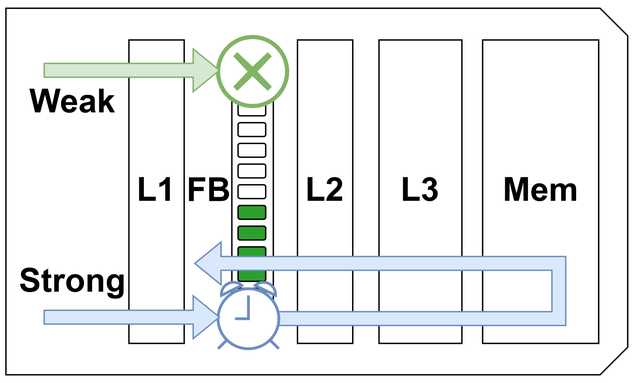
Background

Prefetching Localities



Localities determine target cache level and eviction behavior.

Prefetching Reliability



When hardware resources are sparse, the CPU …

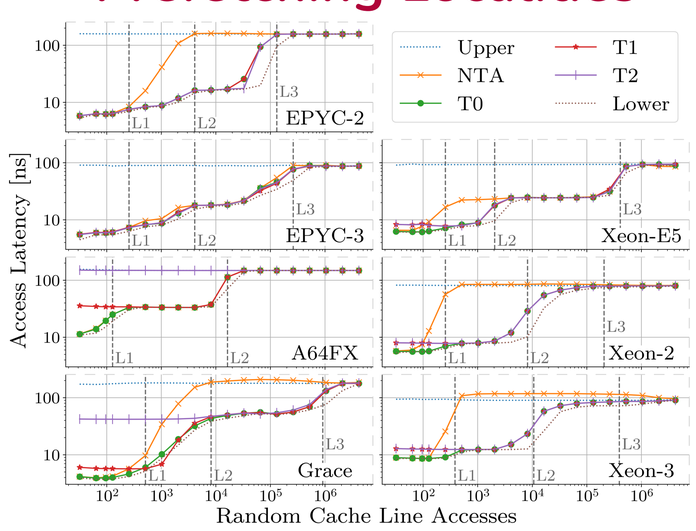
Weak: drops the prefetch.

Strong: stalls to guarantee prefetch execution.

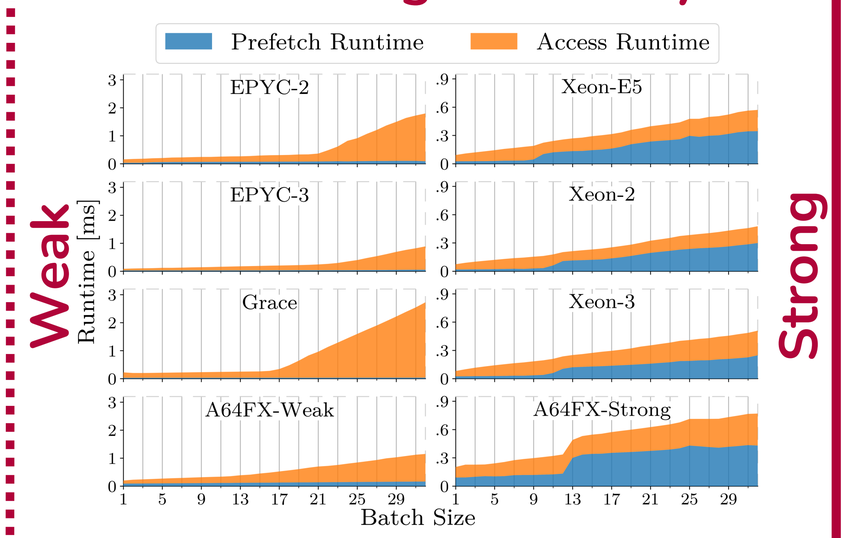
Different CPU Prefetching Implementations

Problem: Prefetching **target cache levels**, **eviction behavior**, and **reliability** are CPU specific.

Prefetching Localities



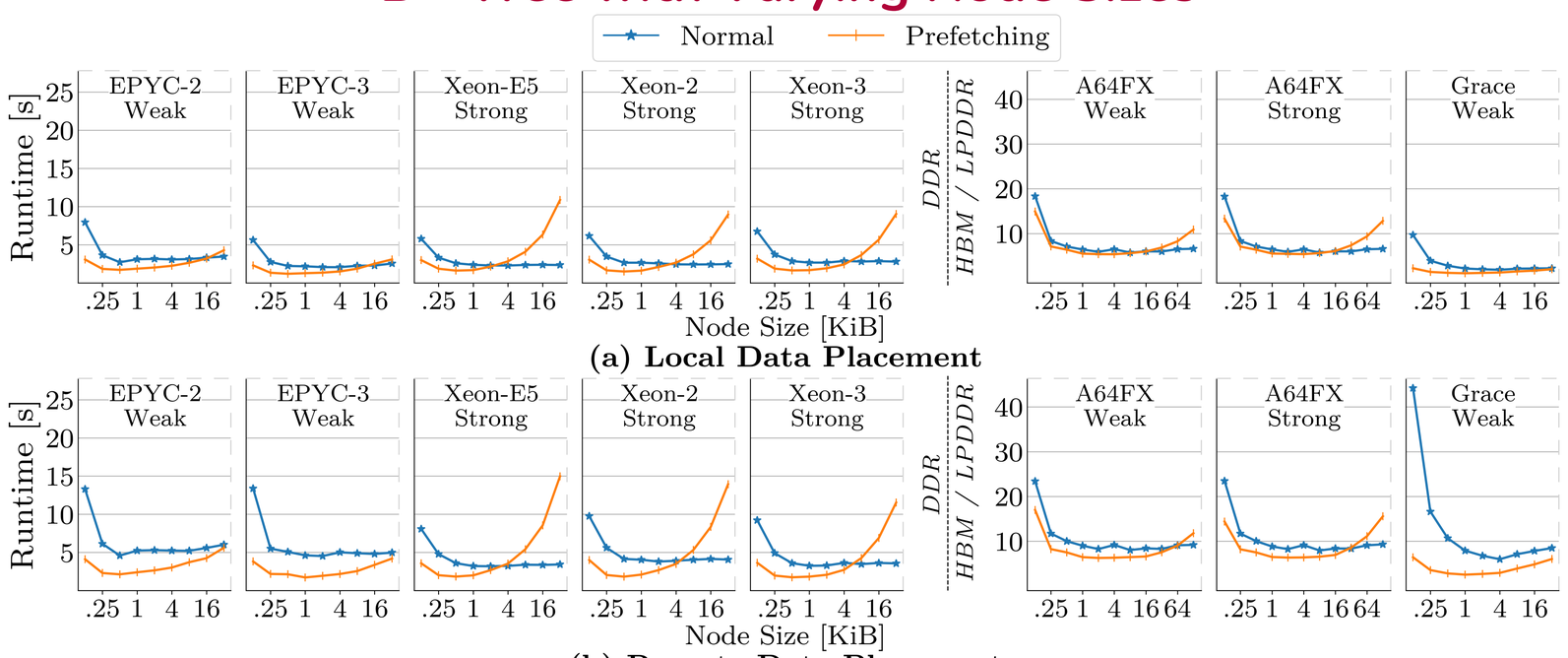
Prefetching Reliability



Results: Prefetching localities *T0* and *NTA* lead to consistent target cache level and eviction behavior on most used systems. We can also approximate the fill buffer (FB) size using our reliability benchmark.

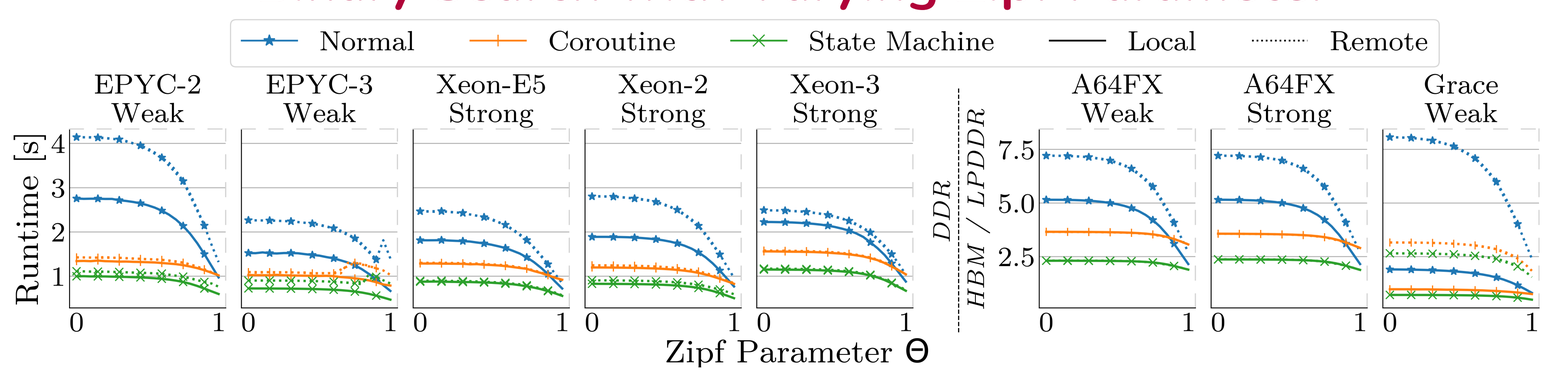
Prefetching on High Access Latency Memory

B+-Tree with Varying Node Sizes



Findings: Prefetching reliability impacts performance when prefetching speculatively (not every prefetched cache line is subsequently accessed).

Binary Search with Varying Zipf Parameter



Findings: Prefetching can hide increased access latencies if enough FB slots are available.

HPI Hasso Plattner Institut

Fabian Mahling
Hasso Plattner Institute, Potsdam, Germany
fabian.mahling@student.hpi.de

This work was partially funded by the German Research Foundation (ref. 414984028), European Union's Horizon 2020 research and innovation programme (ref. 957407), and by SAP.