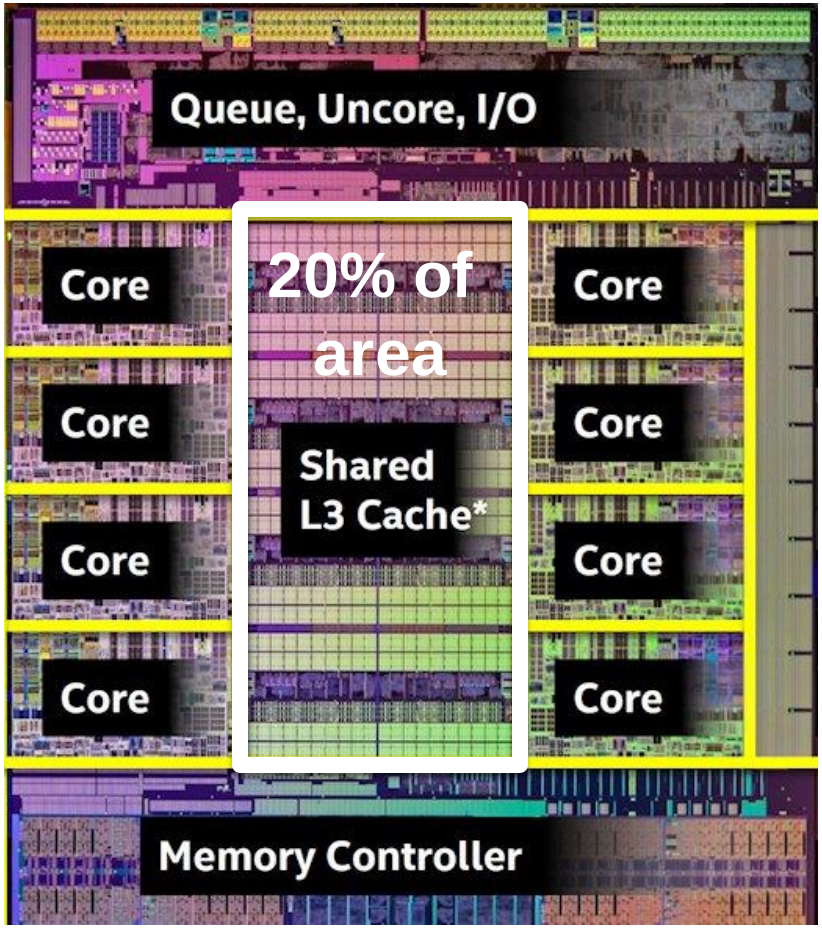
Speculative, Coordinated Memory Hierarchy Management

Paul V. Gratz and Daniel A. Jiménez

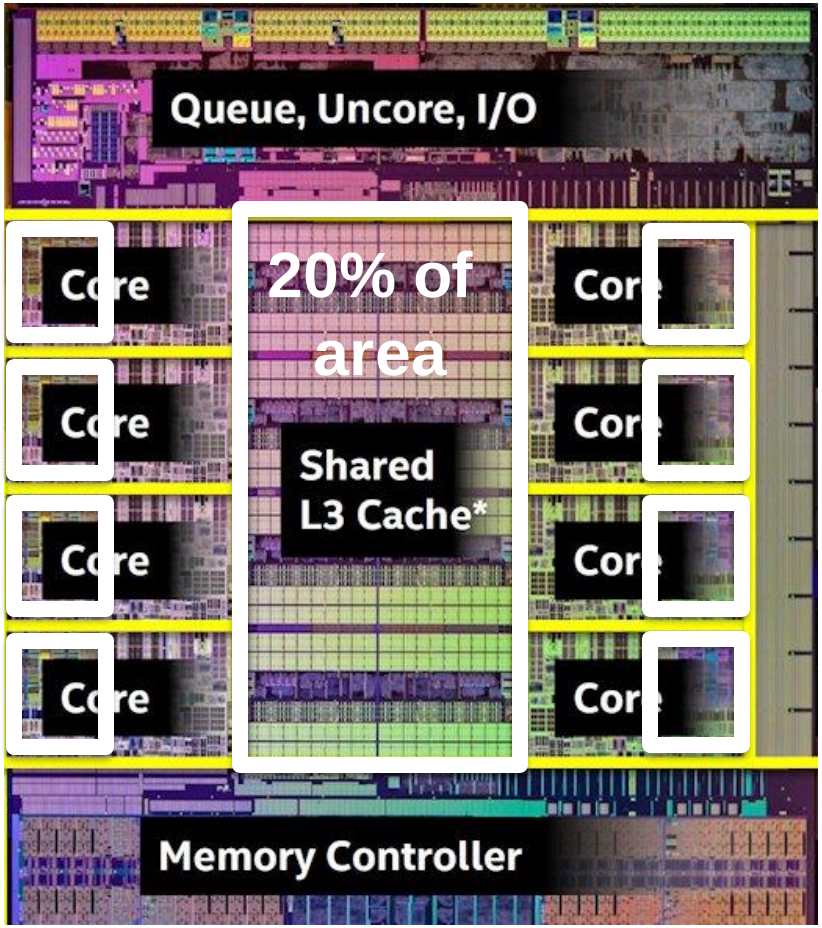
TEXAS A&M UNIVERSITY®

Cache: A huge on-chip structure



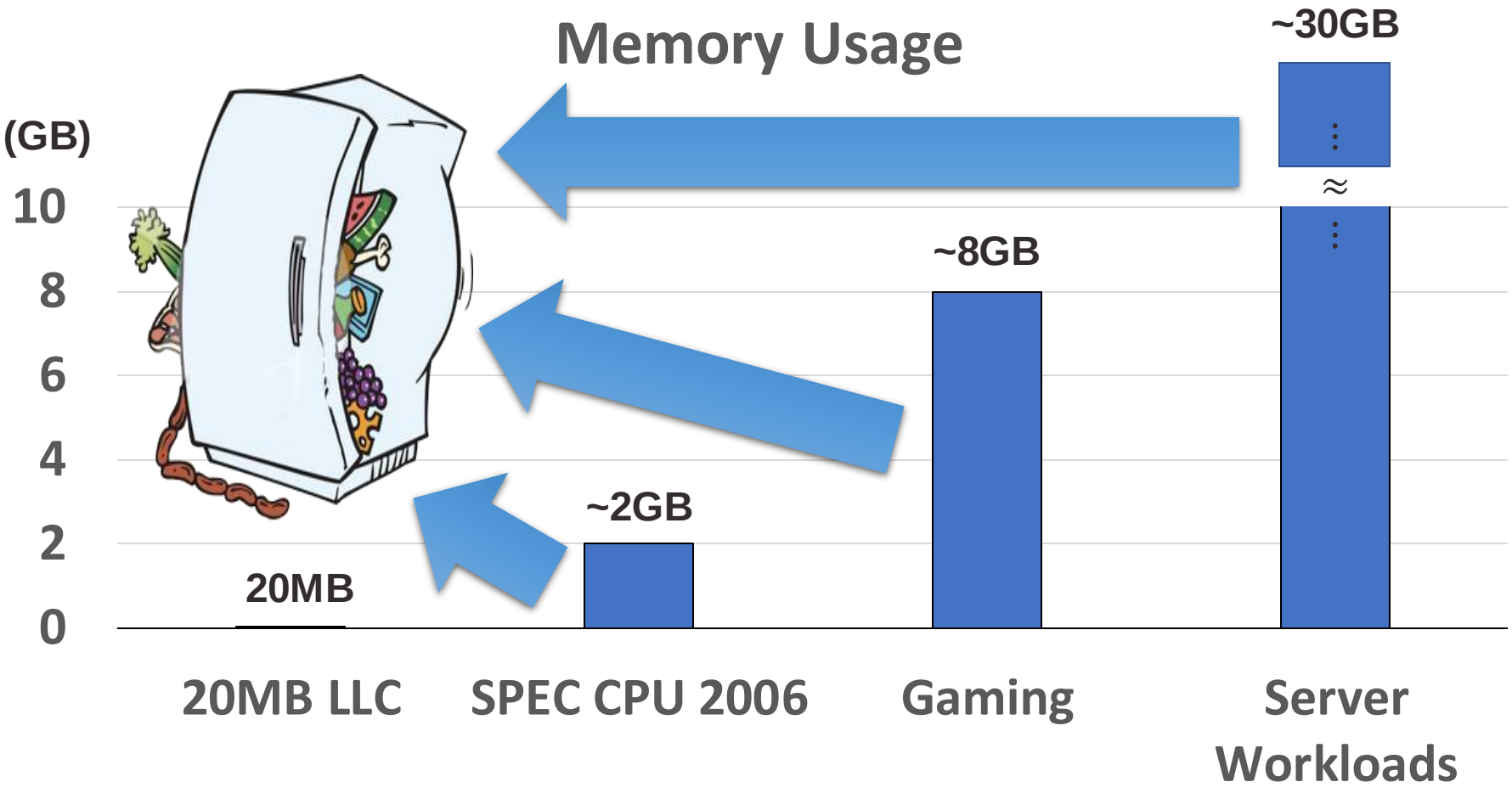
- 20MB Last-Level Cache (LLC) takes 20% of on-chip area

Cache: A huge on-chip structure

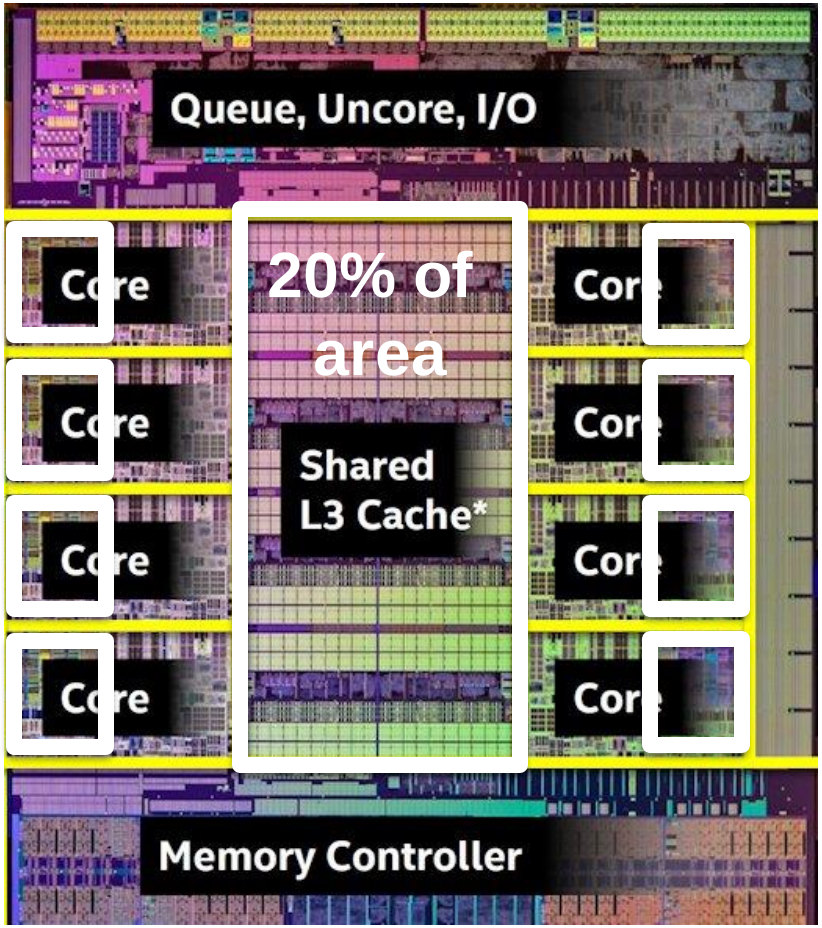


- 20MB Last-Level Cache (LLC) takes 20% of on-chip area
- Private caches also take substantial area

Cache: A small memory

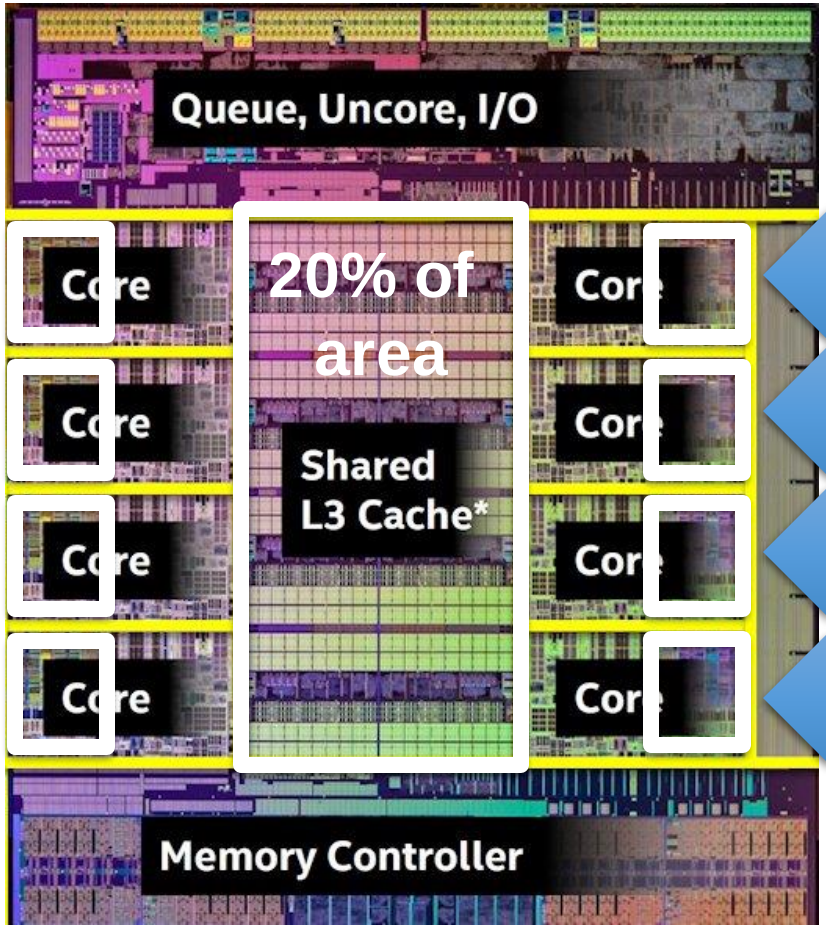


Cache: A huge on-chip structure



- ***Efficient cache management is highly important!***

Cache: A huge on-chip structure



- ***Efficient cache management is highly important!***

Data Prefetching

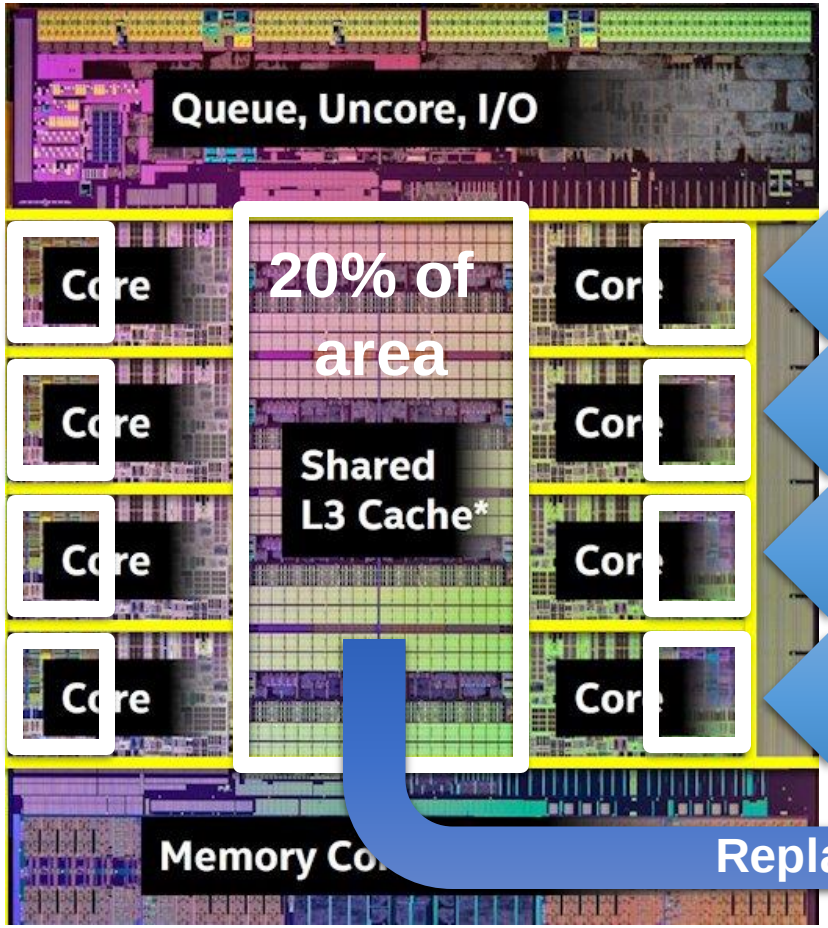
Data Prefetching

Data Prefetching

Data Prefetching

DRAM

Cache: A huge on-chip structure



- ***Efficient cache management is highly important!***

Data Prefetching

Data Prefetching

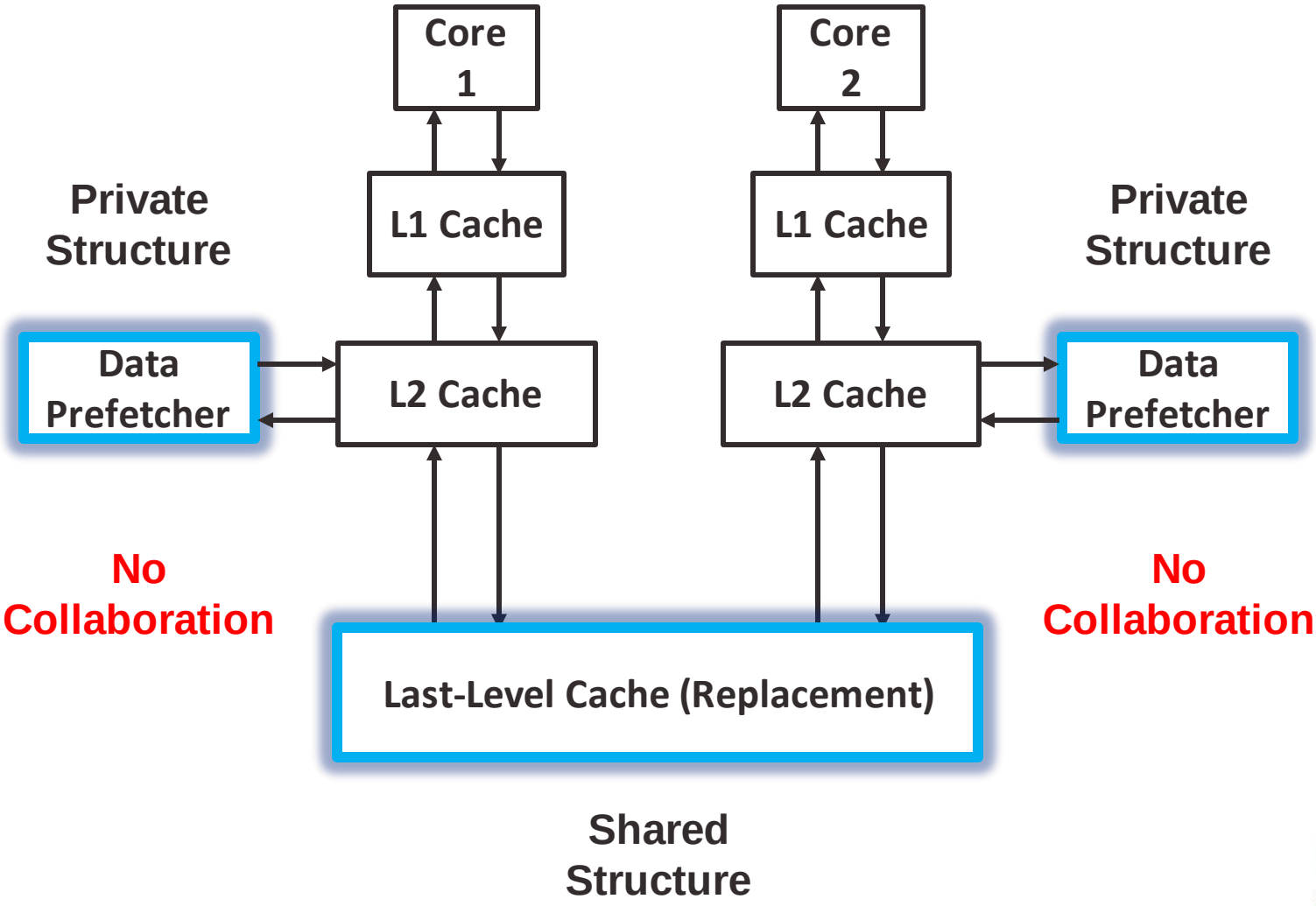
Data Prefetching

Data Prefetching

DRAM

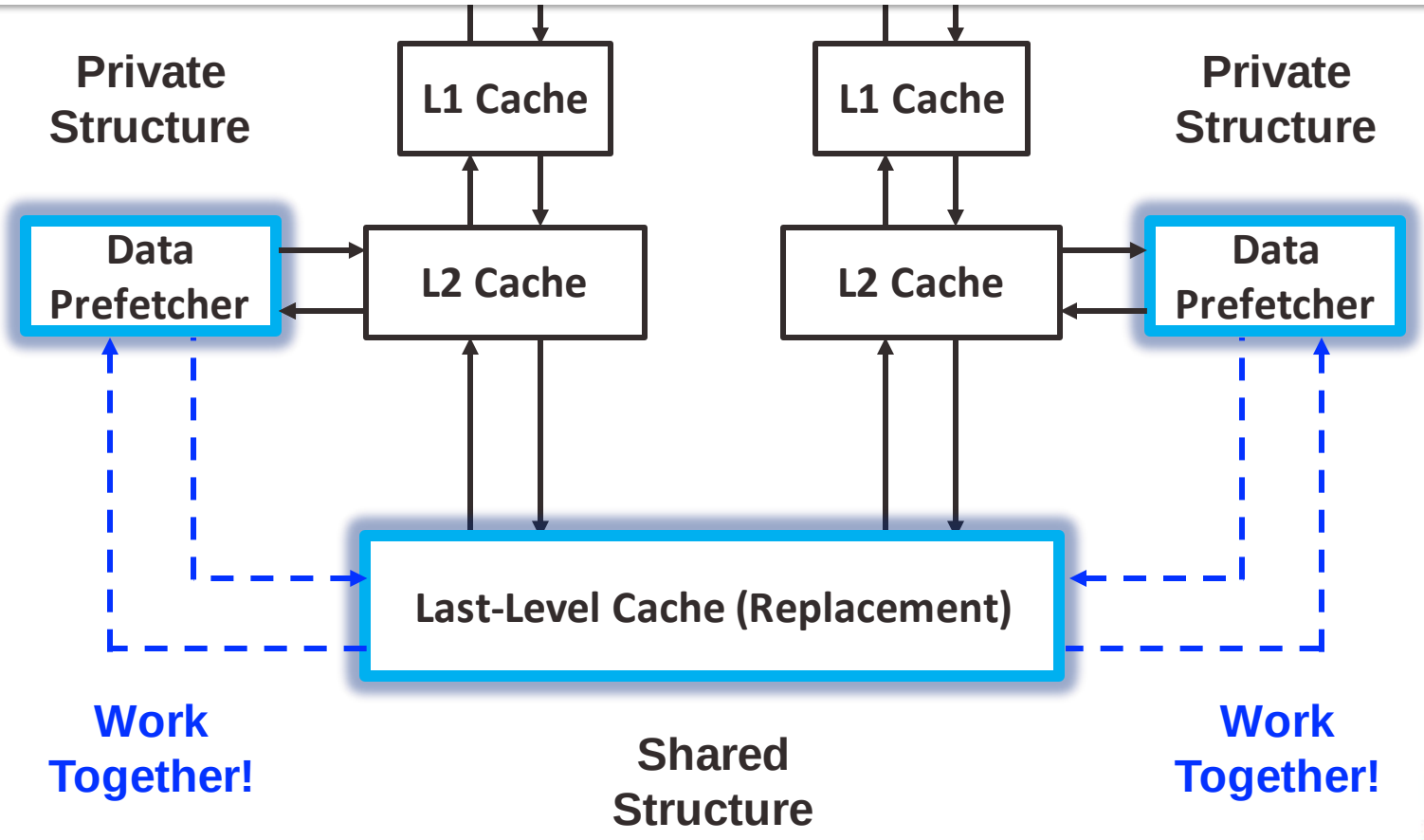
Replacement Policy

Cache Management



Cache Management

Can we build a holistic cache management system?



Overview

- Introduction
- Motivation
  - Related Works
  - Program Counter (PC) is not perfect
- Design
- Evaluation
- Conclusion

Why do we need a
"***Holistic***" cache management?

Related Works

- Prior works on the data prefetching algorithm
- Pure hardware-based prefetchers are listed

  - ❑ Temporal Memory Streaming (TMS) [Wenisch et al. ISCA '05]
  - ❑ Spatial Memory Streaming (SMS) [Somogyi et al. ISCA '06]
  - ❑ Spatio-temporal Memory Streaming (STeMS) [Somogyi et al. ISCA '09]
  - ❑ Access Map Pattern Matching (AMPM) [Ishii et al. ICS '09]
  - ❑ Unified Memory Optimization (UMO) [Ishii et al. ICS '12]
  - ❑ Irregular Streaming Buffer [Jain et al. ISCA '13]
  - ❑ Sandbox Prefetching [Pugsley et al. HPCA '14]
  - ❑ B-Fetch [Kadjo et al. MICRO '14]
  - ❑ Best-offset Prefetcher [Michaud HPCA '16]
  - ❑ Signature Path Prefetching [Kim et al. MICRO '16]

Related Works

Most data prefetching techniques do not consider the LLC replacement policy

- ❑ Temporal Memory Streaming (TMS) [Wenisch et al. ISCA '05]
- ❑ Spatial Memory Streaming (SMS) [Somogyi et al. ISCA '06]
- ❑ Spatio-temporal Memory Streaming (STeMS) [Somogyi et al. ISCA '09]
- ❑ Access Map Pattern Matching (AMPM) [Ishii et al. ICS '09]
- ❑ Unified Memory Optimization (UMO) [Ishii et al. ICS '12]
- ❑ Irregular Streaming Buffer [Jain et al. ISCA '13]
- ❑ Sandbox Prefetching [Pugsley et al. HPCA '14]
- ❑ B-Fetch [Kadjo et al. MICRO '14]
- ❑ Best-offset Prefetcher [Michaud HPCA '16]
- ❑ Signature Path Prefetching [Kim et al. MICRO '16]

Related Works

- Prior works on the LLC replacement policy
- Pure hardware-based replacement policies are listed

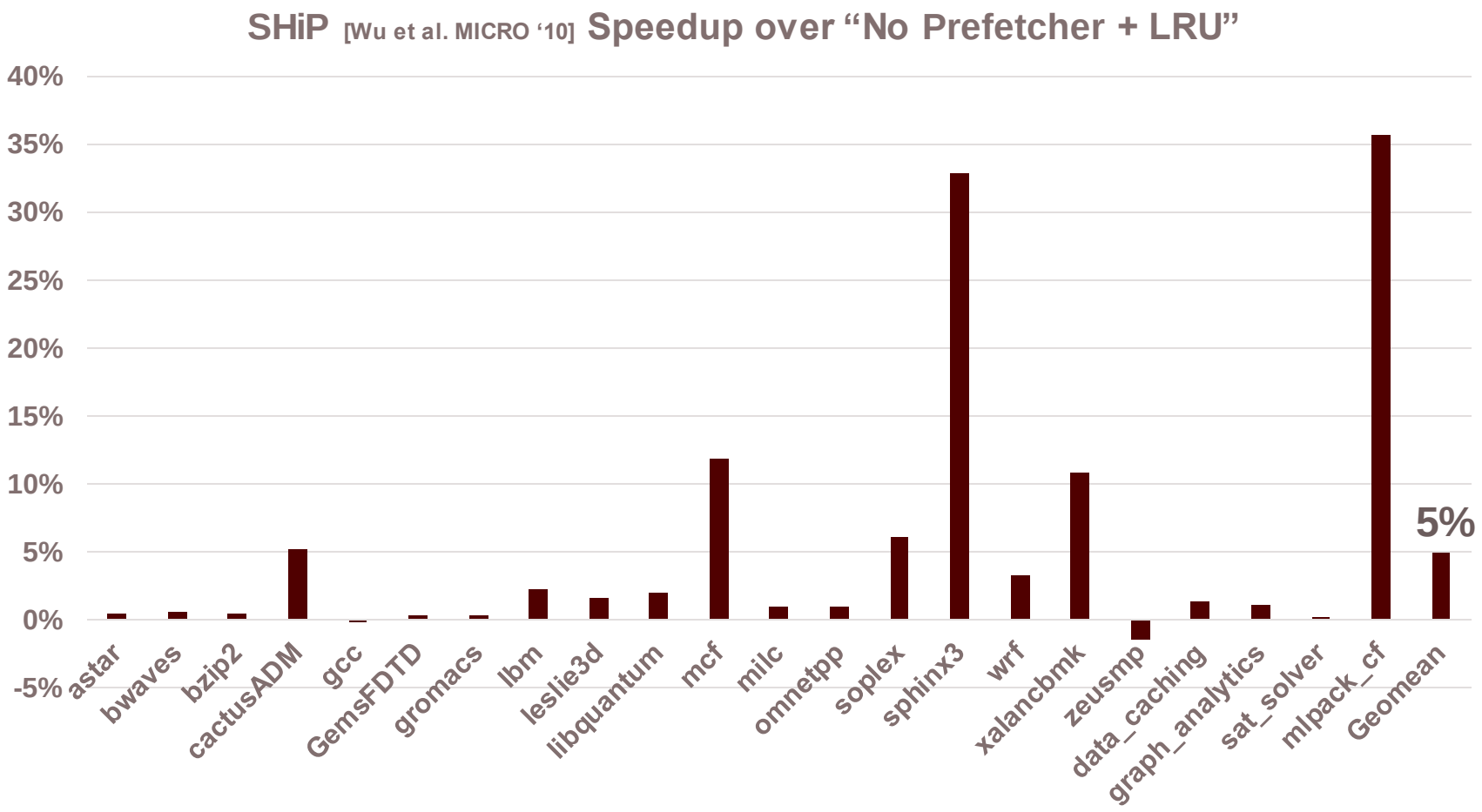
  - ❑ Dynamic Insertion Policy (DIP) [Qureshi et al. ISCA '07]
  - ❑ Re-Reference Interval Prediction (RRIP) [Jaleel et al. ISCA '10]
  - ❑ Sampling Dead Block Prediction (SDBP) [Khan et al. MICRO '10]
  - ❑ Signature-based Hit Prediction (SHIP) [Wu et al. MICRO '11]
  - ❑ PACMan [Wu et al. MICRO '11]
  - ❑ Evicted Address Filter (EAF) [Seshadri et al. PACT '12]
  - ❑ Minimal Disturbance Placement and Promotion (MDPP) [Teran et al. HPCA '16]
  - ❑ Hawkeye [Jain et al. ISCA '16]
  - ❑ Perceptron Learning for Reuse Prediction [Teran et al. MICRO '16]

Related Works

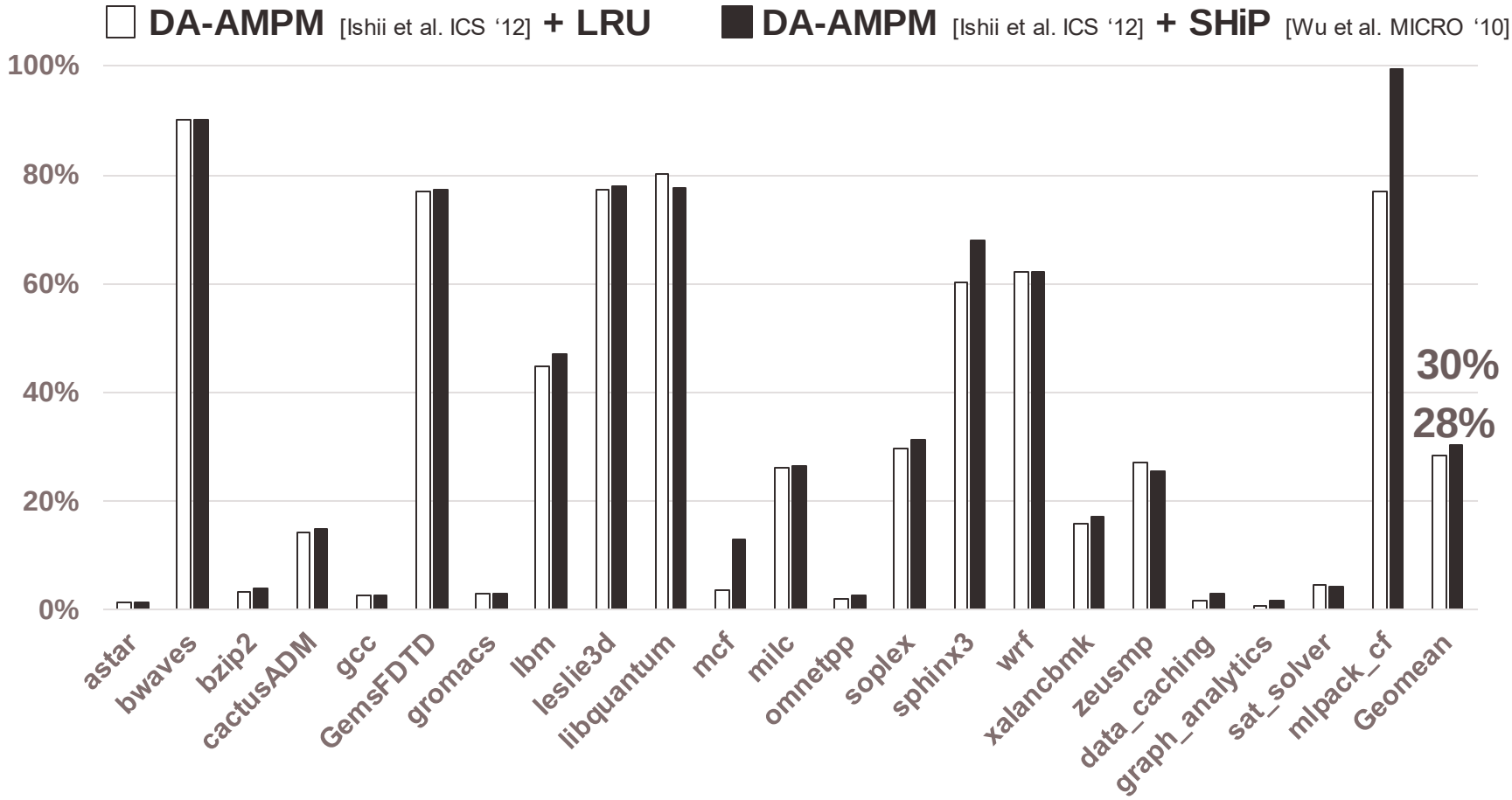
Most replacement policies do not consider prefetching

- Dynamic Insertion Policy (DIP) [Qureshi et al. ISCA '07]
- Re-Reference Interval Prediction (RRIP) [Jaleel et al. ISCA '10]
- Sampling Dead Block Prediction (SDBP) [Khan et al. MICRO '10]
- Signature-based Hit Prediction (SHIP) [Wu et al. MICRO '11]
- PACMan [Wu et al. MICRO '11]
- Evicted Address Filter (EAF) [Seshadri et al. PACT '12]
- Minimal Disturbance Placement and Promotion (MDPP) [Teran et al. HPCA '16]
- Hawkeye [Jain et al. ISCA '16]
- Perceptron Learning for Reuse Prediction [Teran et al. MICRO '16]

Related Works

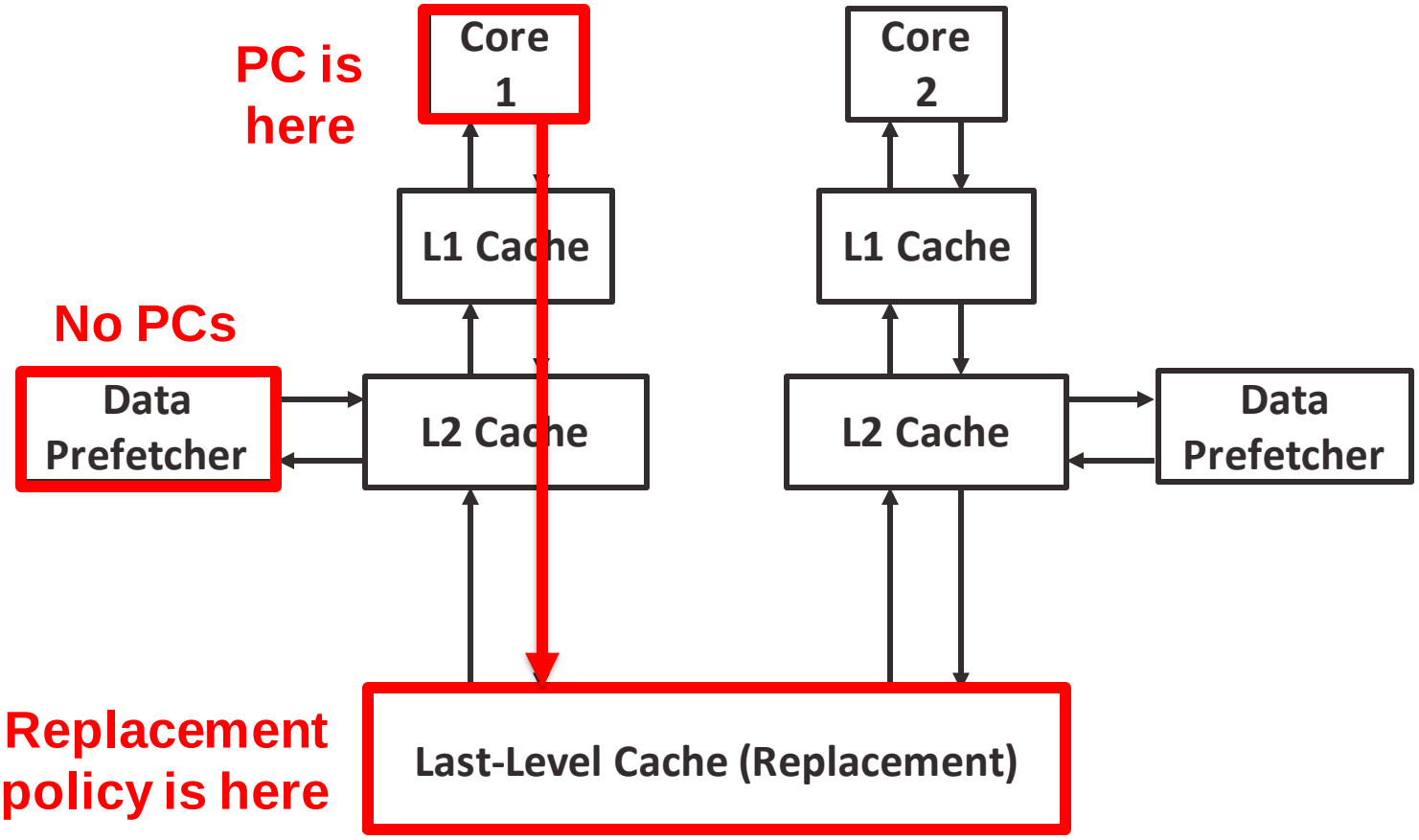


Related Works



Prefetch Interference

- Prefetch request does not have PCs



Hardware Complexity

- Hardware complexity

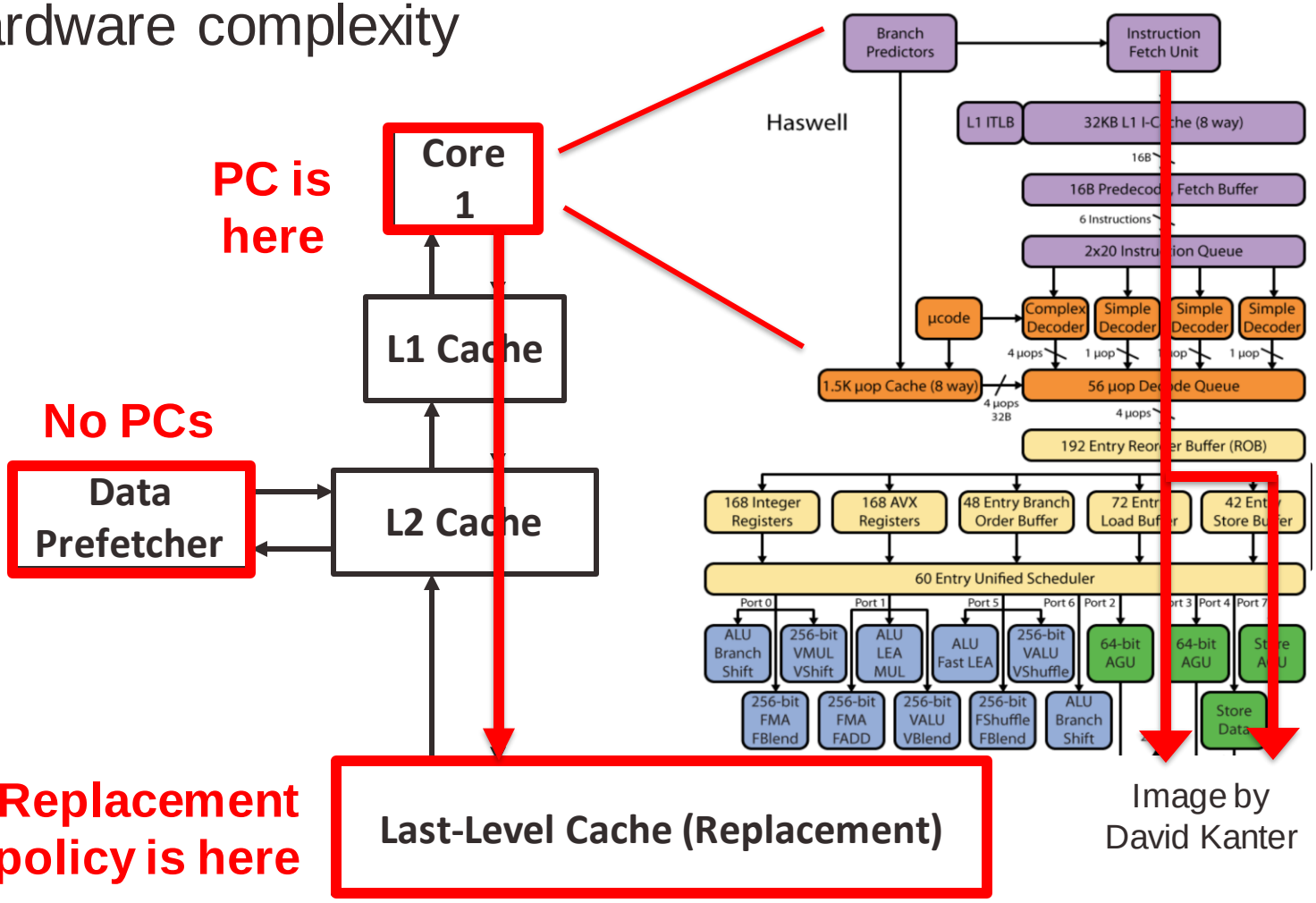


Image by David Kanter

PC is Not Perfect

- Prior works on the LLC replacement policy
- Pure hardware-based replacement policies are listed

  - ❑ Dynamic Insertion Policy (DIP) [Qureshi et al. ISCA '07]
  - ❑ Re-Reference Interval Prediction (RRIP) [Jaleel et al. ISCA '10]
  - ❑ Sampling Dead Block Prediction (SDBP) [Khan et al. MICRO '10]
  - ❑ Signature-based Hit Prediction (SHIP) [Wu et al. MICRO '11]
  - ❑ PACMan [Wu et al. MICRO '11]
  - ❑ Evicted Address Filter (EAF) [Seshadri et al. PACT '12]
  - ❑ Minimal Disturbance Placement and Promotion (MDPP) [Teran et al. HPCA '16]
  - ❑ Hawkeye [Jain et al. ISCA '16]
  - ❑ Perceptron Learning for Reuse Prediction [Teran et al. MICRO '16]

PC is Not Perfect

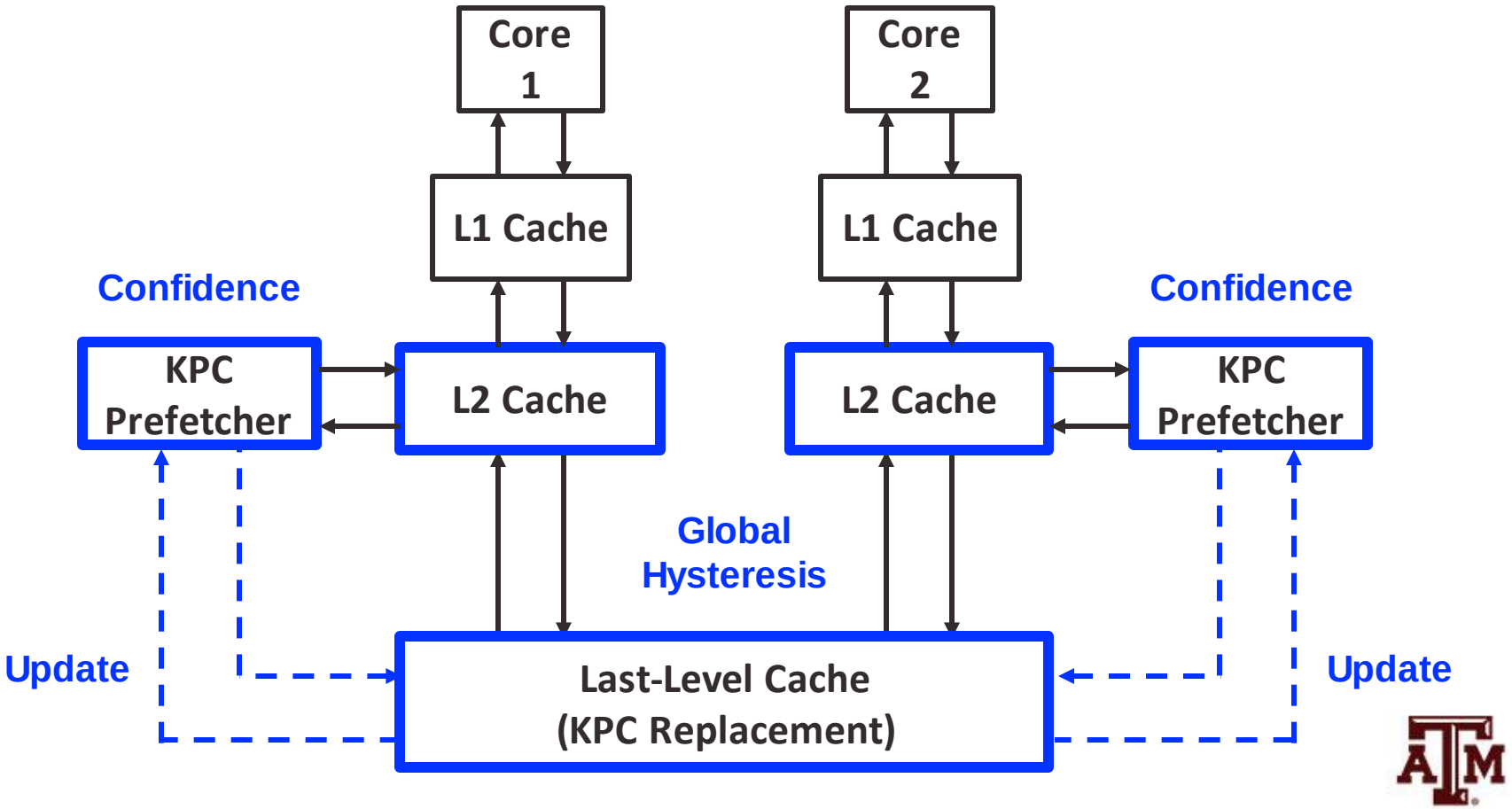
Top-performing replacement policies rely on Program Counter (PC)-based prediction

- ❑ Dynamic Insertion Policy (DIP) [Qureshi et al. ISCA '07]
- ❑ Re-Reference Interval Prediction (RRIP) [Jaleel et al. ISCA '10]
- ❑ Sampling Dead Block Prediction (SDBP) [Khan et al. MICRO '10]
- ❑ Signature-based Hit Prediction (SHIP) [Wu et al. MICRO '11]
- ❑ PACMan [Wu et al. MICRO '11]
- ❑ Evicted Address Filter (EAF) [Seshadri et al. PACT '12]
- ❑ Minimal Disturbance Placement and Promotion (MDPP) [Teran et al. HPCA '16]
- ❑ Hawkeye [Jain et al. ISCA '16]
- ❑ Perceptron Learning for Reuse Prediction [Teran et al. MICRO '16]

How can we design a
"***Holistic***" cache management?

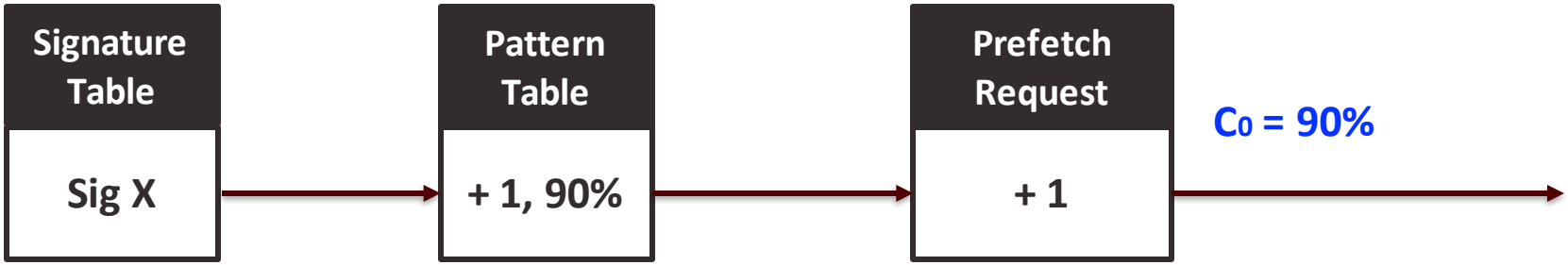
Kill the Program Counter (KPC)

- A simple, powerful, and holistic cache management system



KPC-P: Prefetching

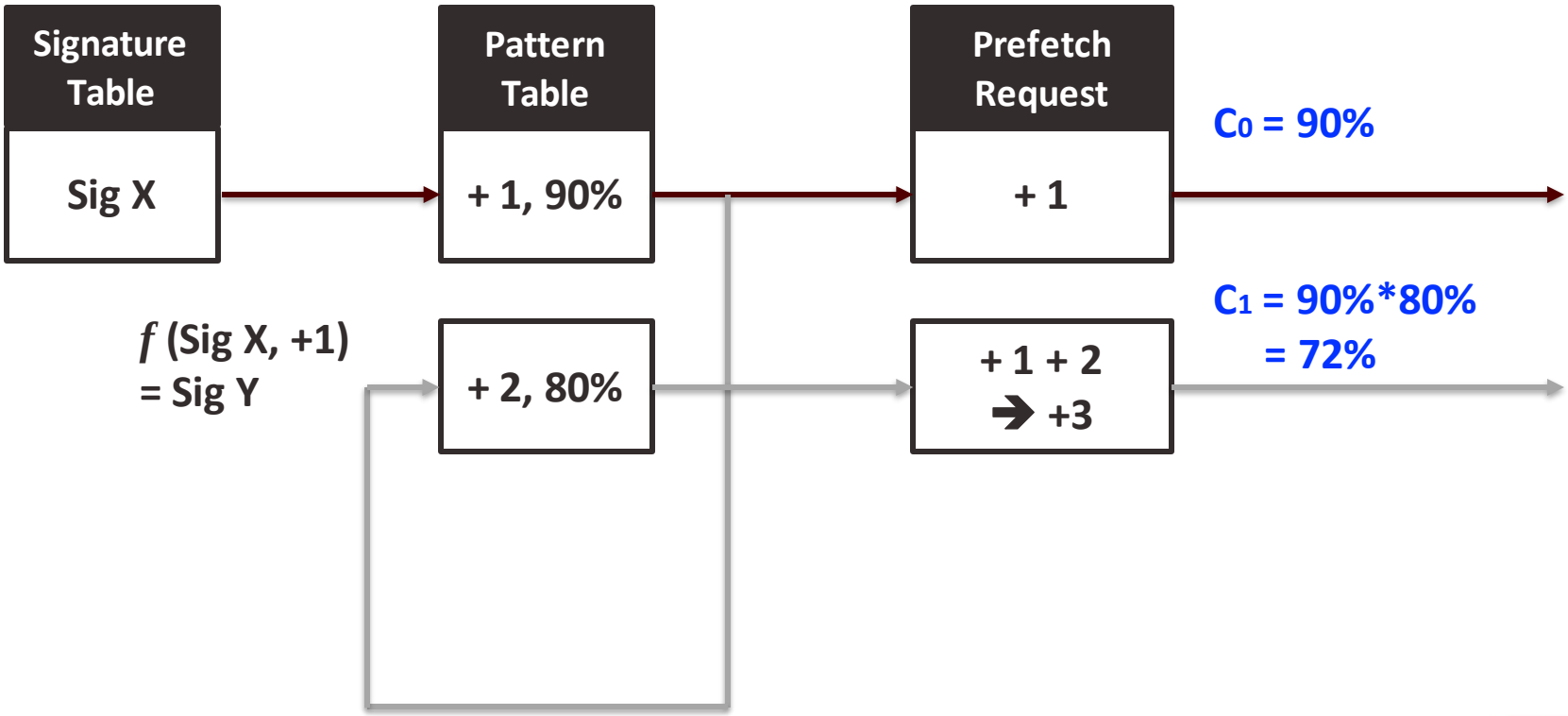
- Confidence-based prefetcher inspired by SPP [Kim et al. MICRO '16]



✓ Issue prefetch with confidence

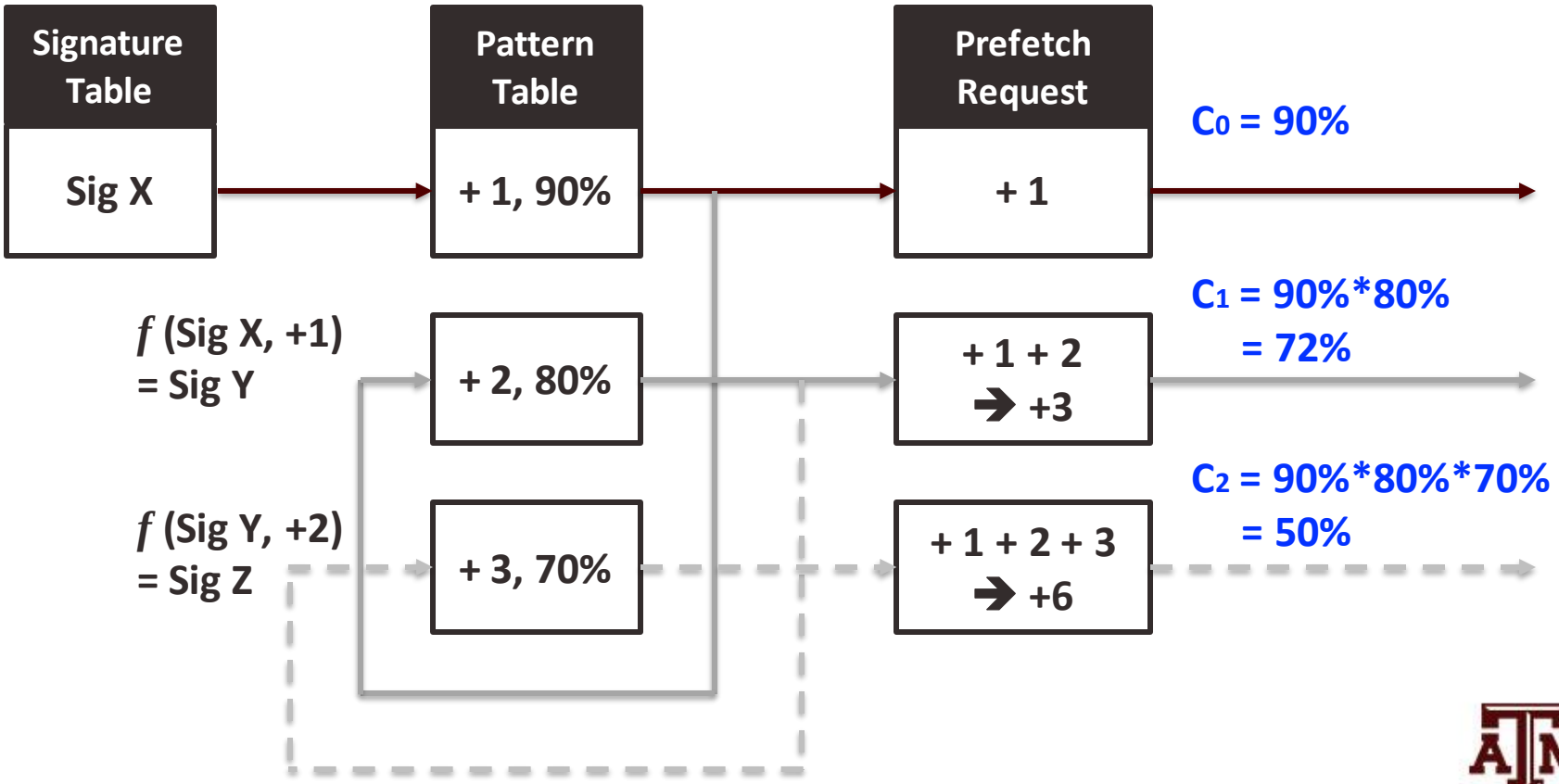
KPC-P: Prefetching

- Confidence-based prefetcher inspired by SPP [Kim et al. MICRO '16]



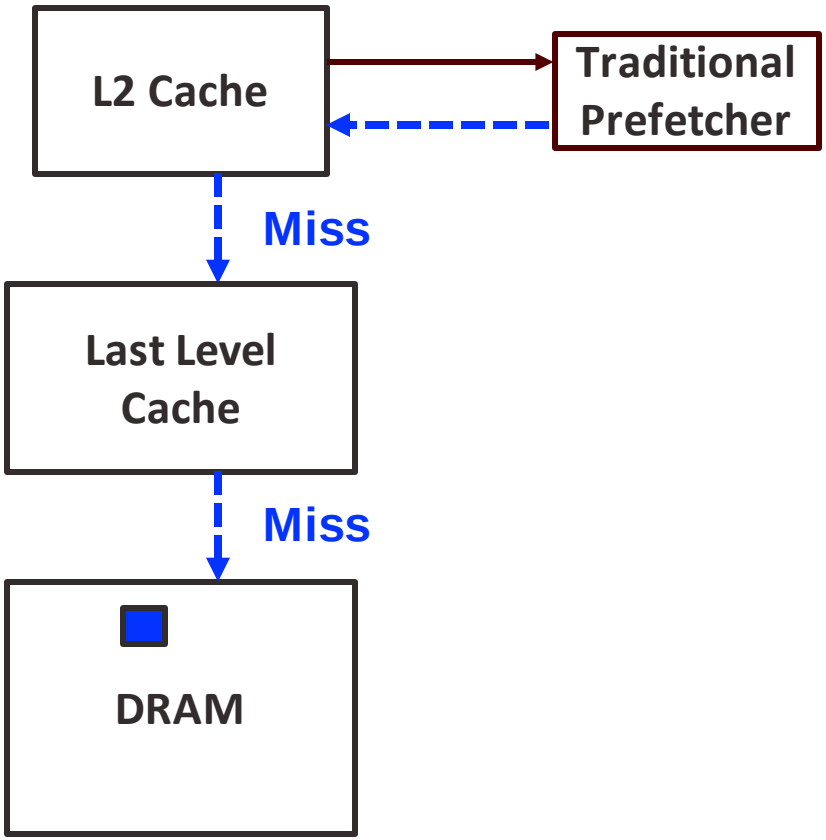
KPC-P: Prefetching

- Confidence-based prefetcher inspired by SPP [Kim et al. MICRO '16]



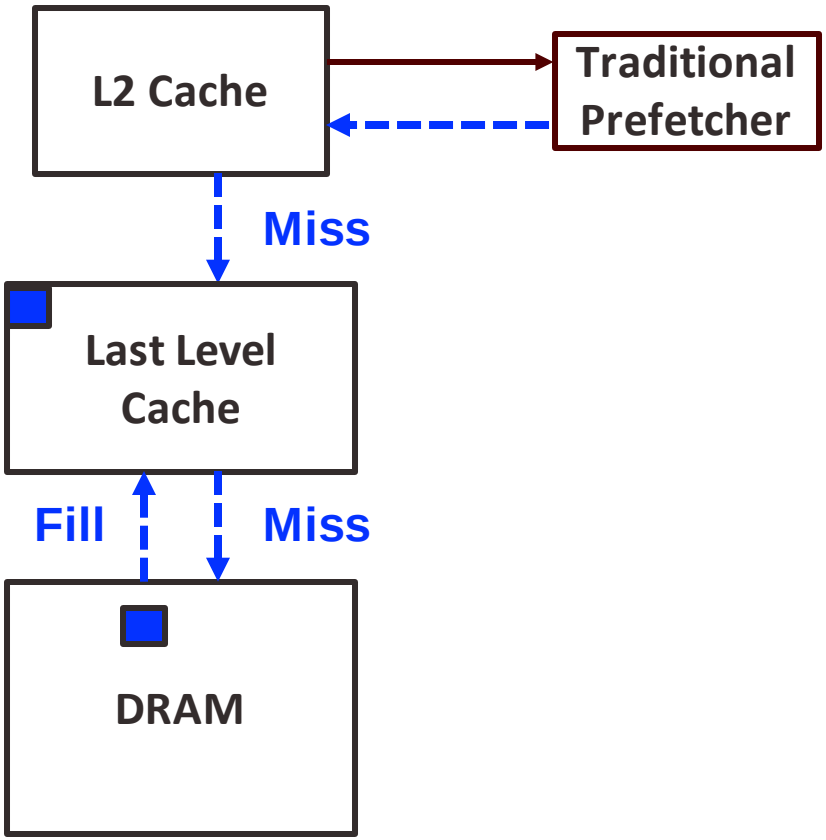
KPC-P: Caching Level Control

- Prefetch request miss
  - ❑ Fetch cache block from DRAM and fill both L2 and LLC



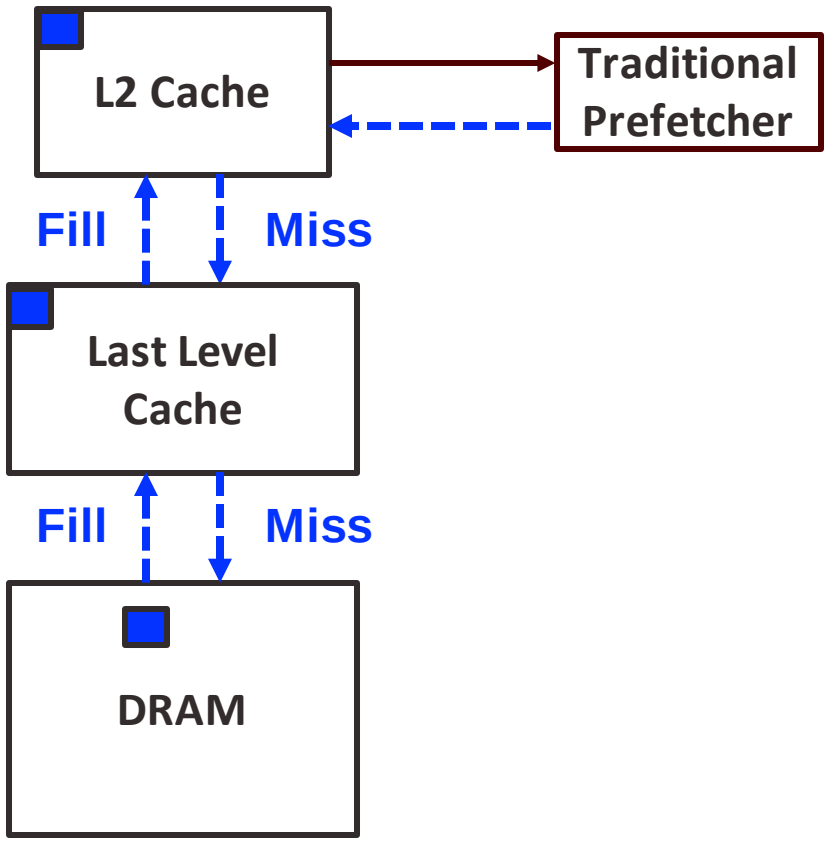
KPC-P: Caching Level Control

- Prefetch request miss
  - ❑ Fetch cache block from DRAM and fill both L2 and LLC



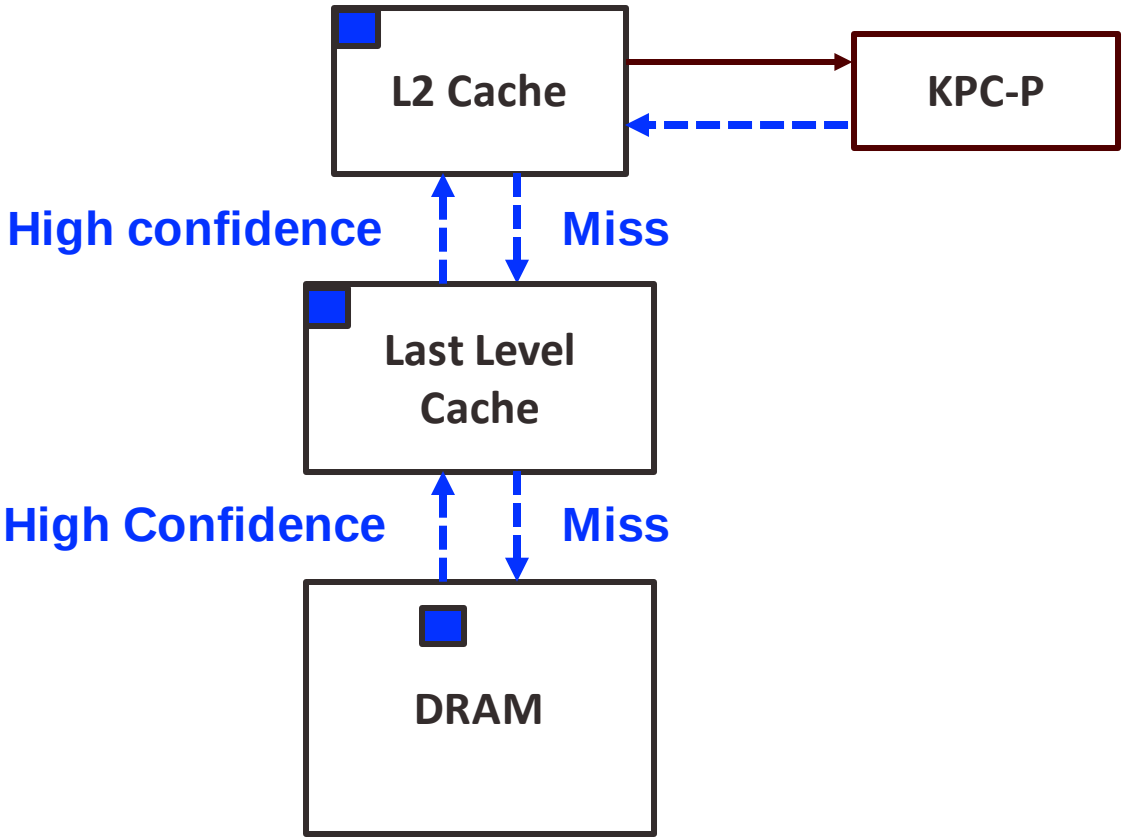
KPC-P: Caching Level Control

- Prefetch request miss
  - ❑ Fetch cache block from DRAM and fill both L2 and LLC



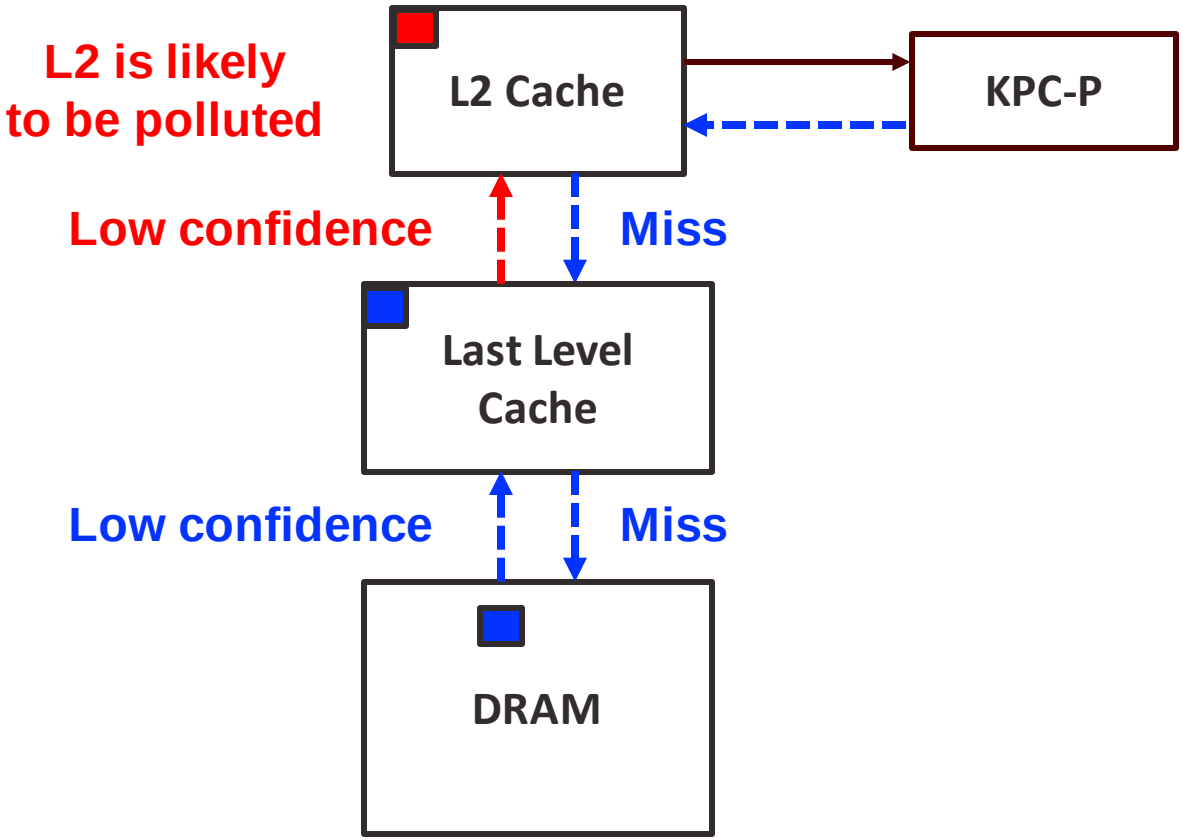
KPC-P: Caching Level Control

- Prefetch request miss (High confidence)
  - ❑ Accurate prefetches



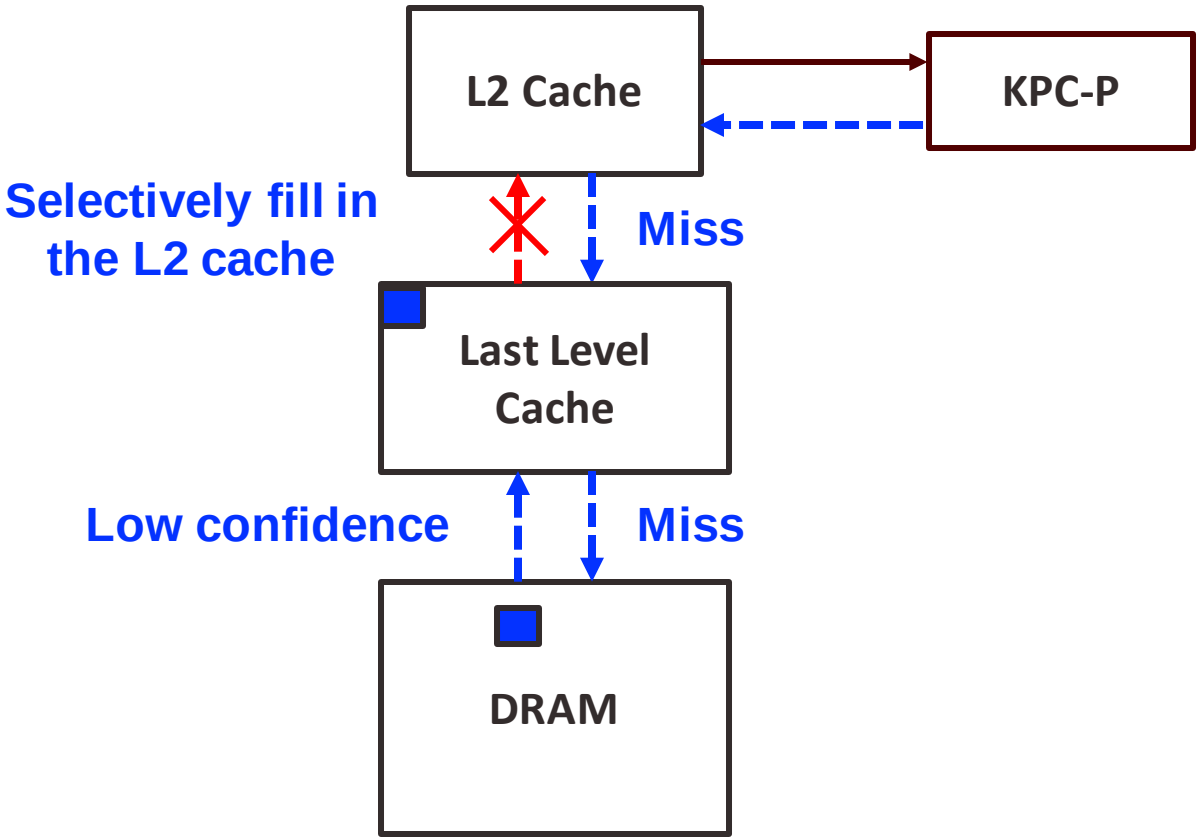
KPC-P: Caching Level Control

- Prefetch request miss (Low confidence)
  - ❑ A) Inaccurate prefetches B) Accurate but use distance is too long



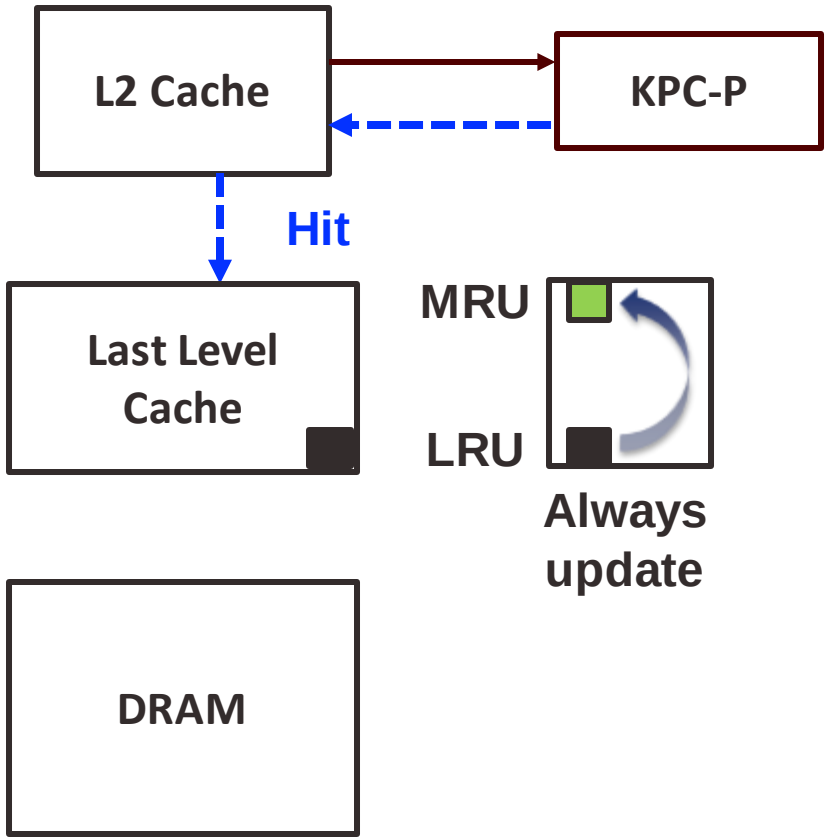
KPC-P: Caching Level Control

- Prefetch request miss (Low confidence)
  - ❑ A) Inaccurate prefetches B) Accurate but use distance is too long



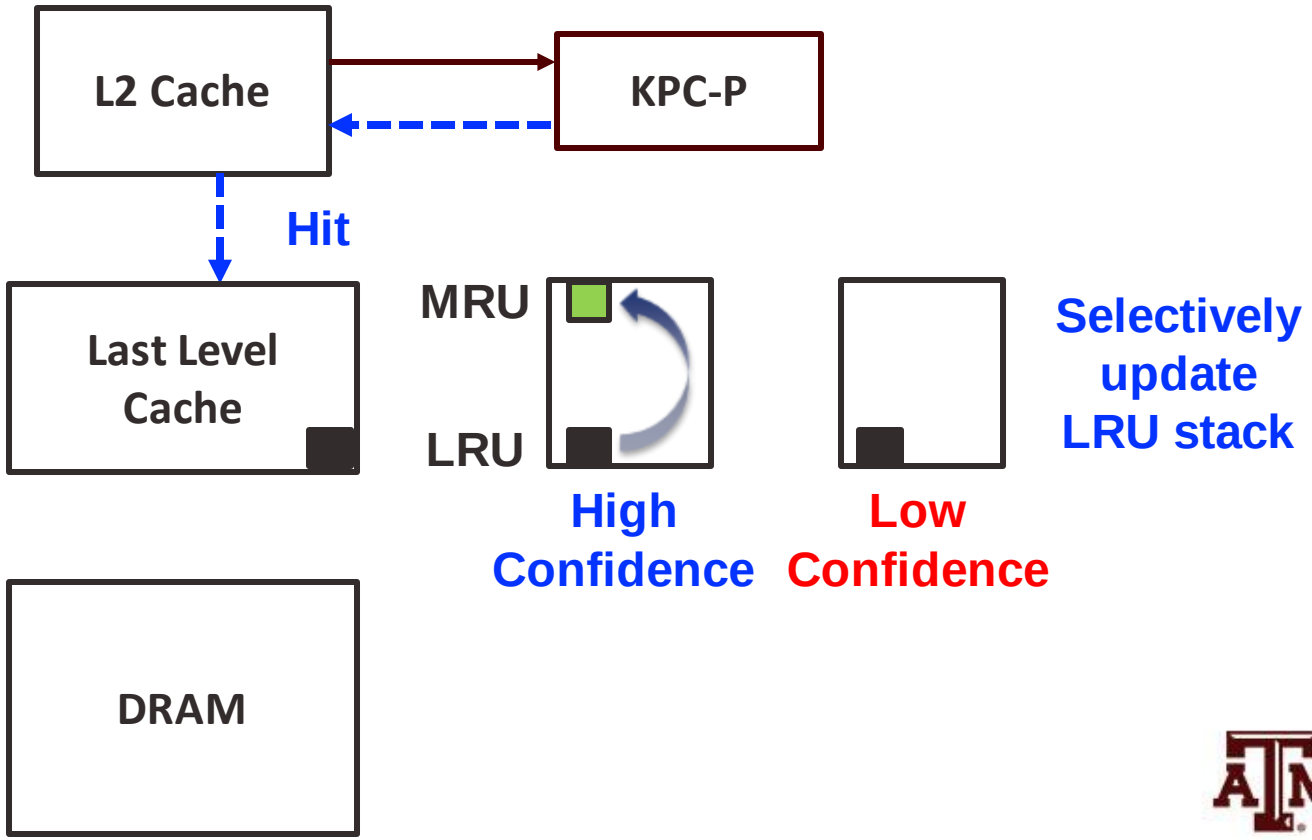
KPC-P: Placement Control

- Prefetch request hit in the LLC
  - ❑ Always update the LRU stack position



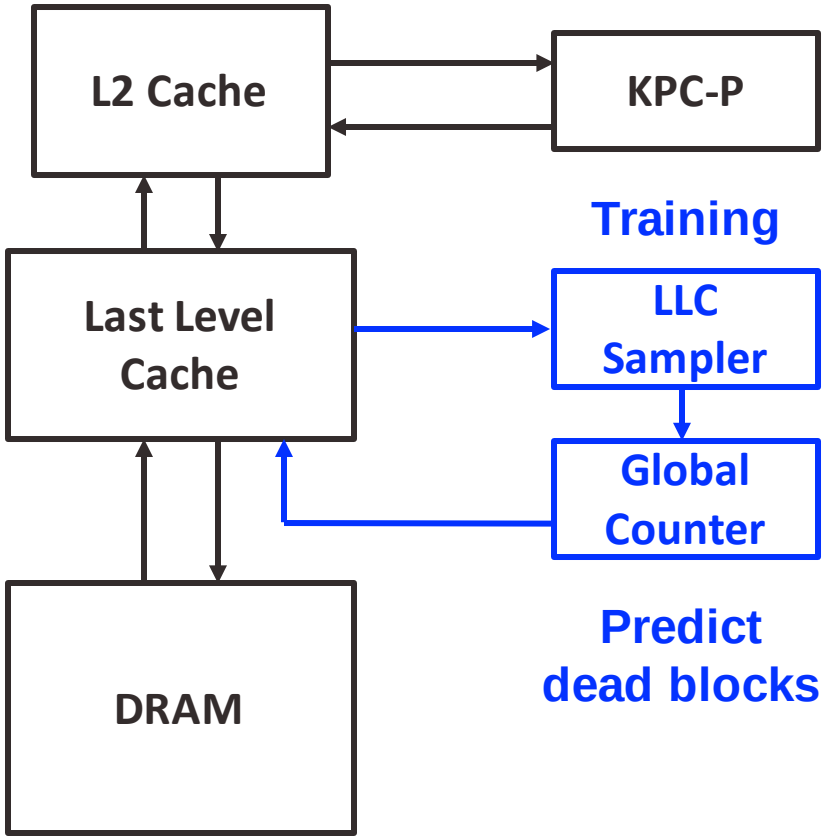
KPC-P: Placement Control

- Prefetch request hit in the LLC (Low confidence)
  - ❑ A) Inaccurate prefetches B) Accurate but use distance is too long



KPC-R: Replacement

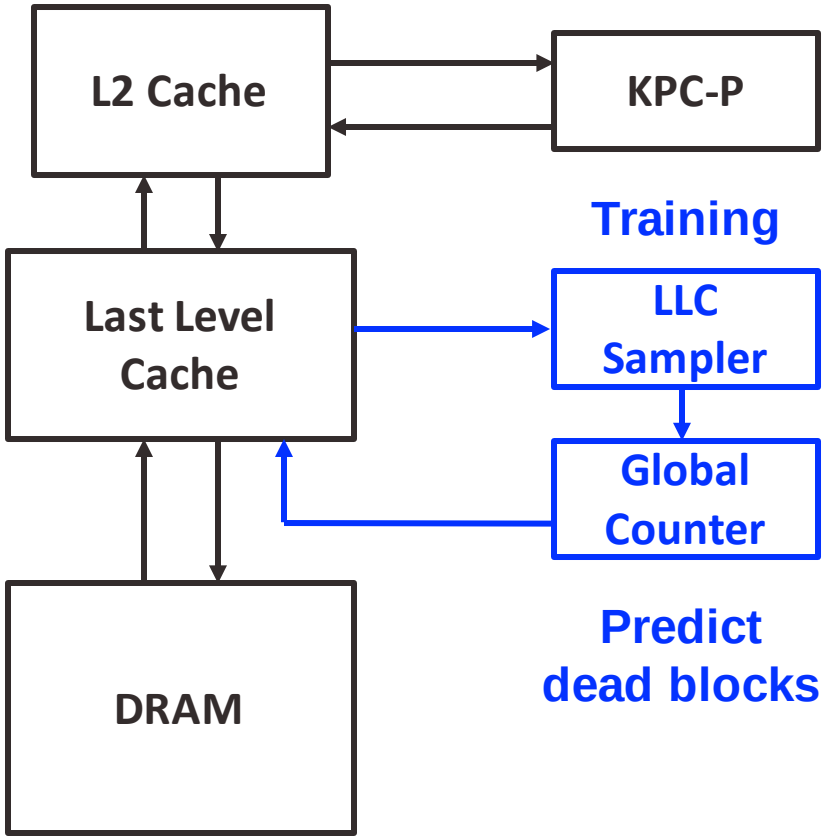
- Predict dead blocks with a simple global hysteresis counter
  - ❑ Decoupled global counters for demand and prefetch



✓ Block is replaced without reuse
➔ Increase "Global Dead Counter"

KPC-R: Replacement

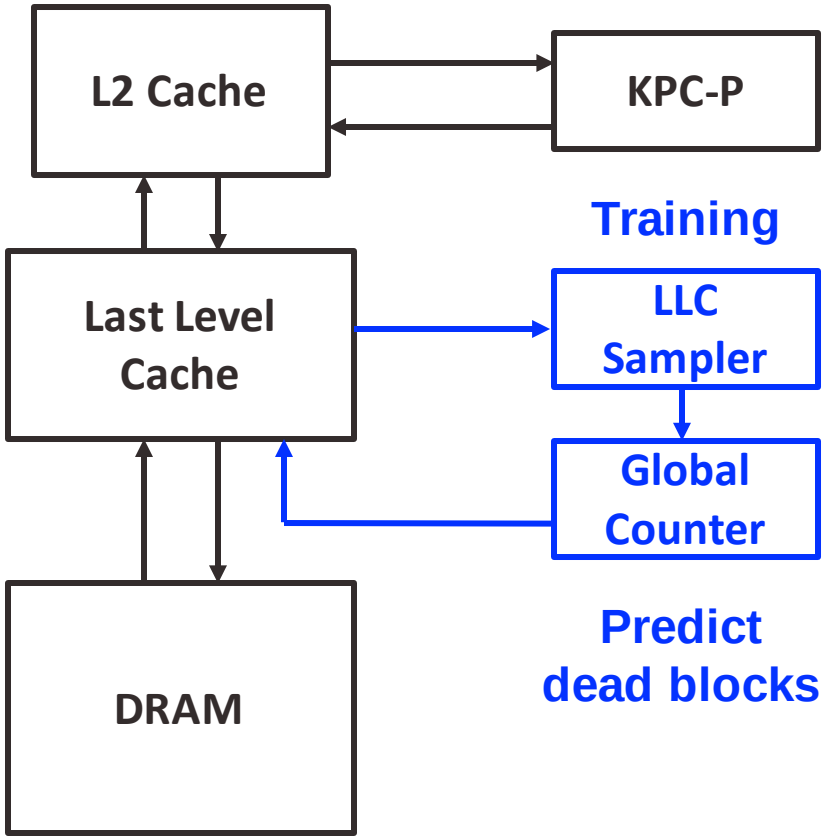
- Predict dead blocks with a simple global hysteresis counter
  - ❑ Decoupled global counters for demand and prefetch



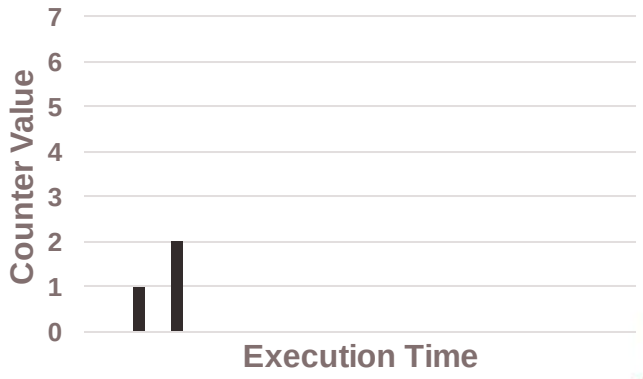
✓ Block is replaced without reuse
➔ Increase "Global Dead Counter"

KPC-R: Replacement

- Predict dead blocks with a simple global hysteresis counter
  - ❑ Decoupled global counters for demand and prefetch

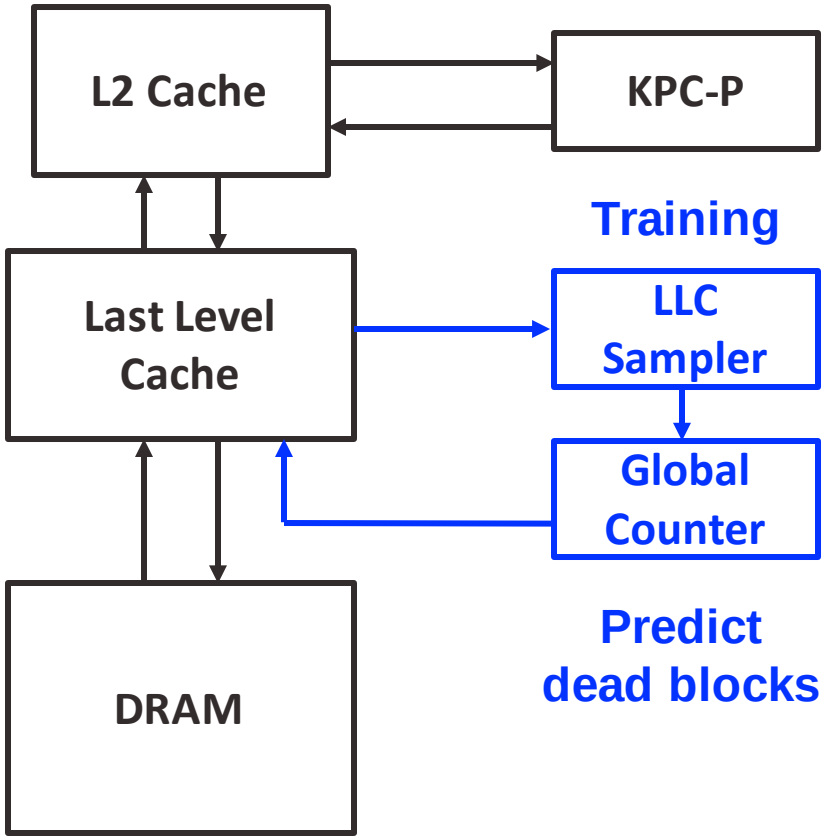


✓ Block is replaced without reuse
➔ Increase "Global Dead Counter"

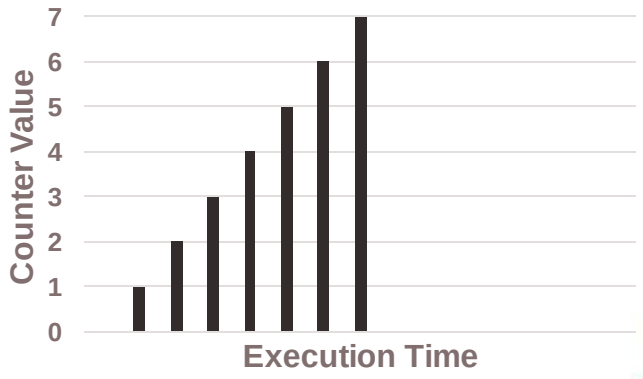


KPC-R: Replacement

- Predict dead blocks with a simple global hysteresis counter
  - ❑ Decoupled global counters for demand and prefetch

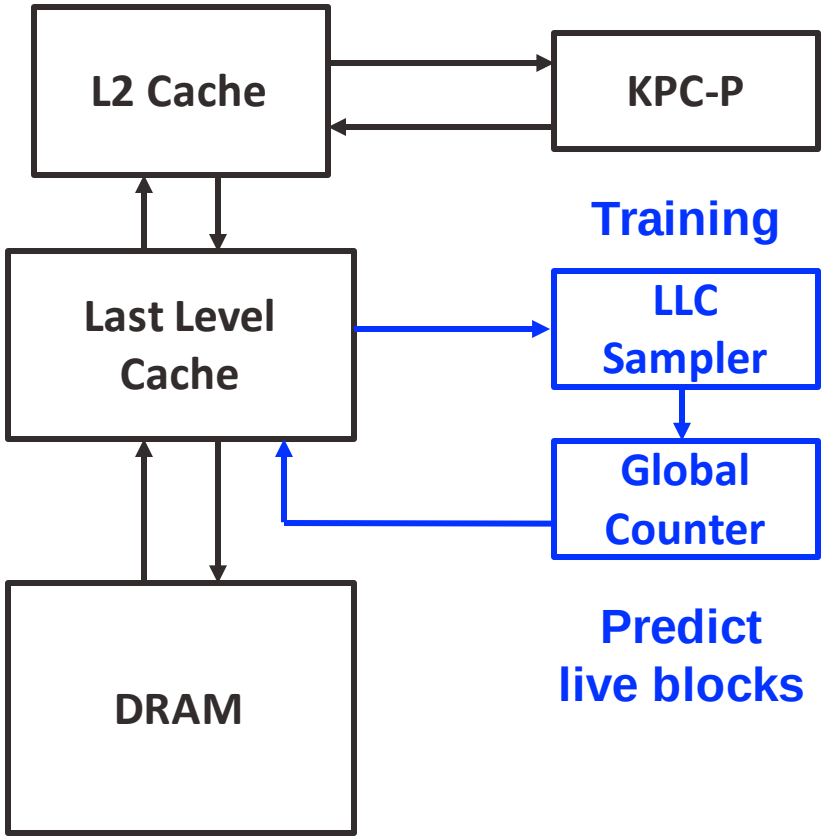


✓ Block is replaced without reuse
➔ Increase "Global Dead Counter"

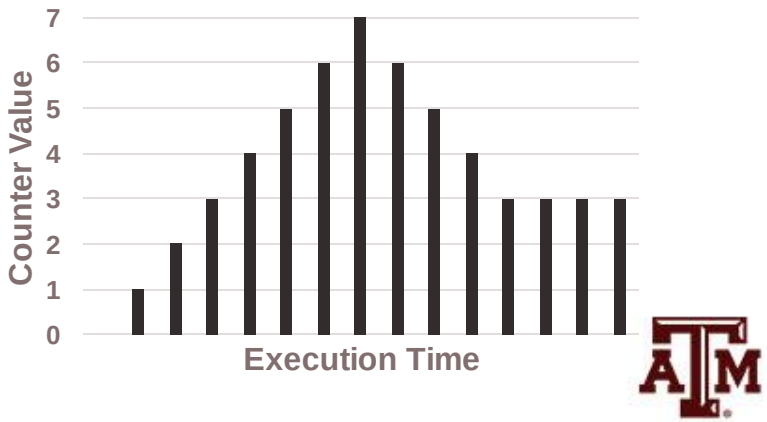


KPC-R: Replacement

- Predict dead blocks with a simple global hysteresis counter
  - ❑ Decoupled global counters for demand and prefetch



✓ Block is replaced without reuse
➔ Decrease "Global Dead Counter"

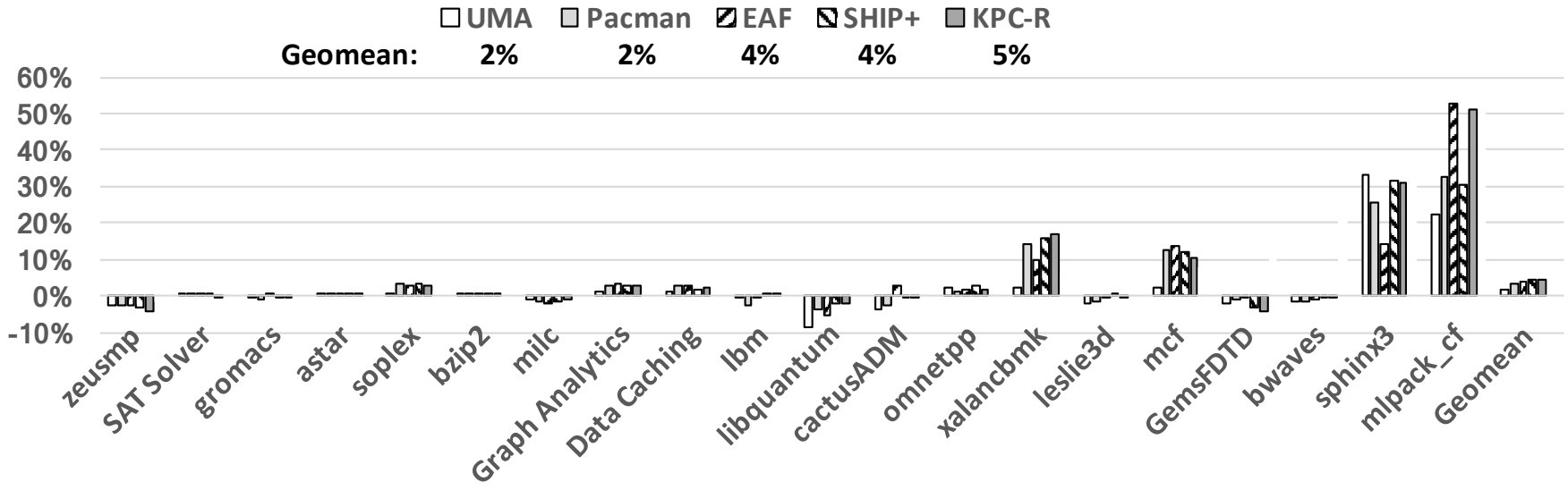


Evaluation

- Tested with ChampSim simulator
  - DA-AMPM (Baseline prefetcher)
  - LRU (Baseline replacement policy)
- Examined a diverse set of workloads
  - SPEC CPU 2006
  - CloudSuite (Server)
  - Mlpack (Machine learning)

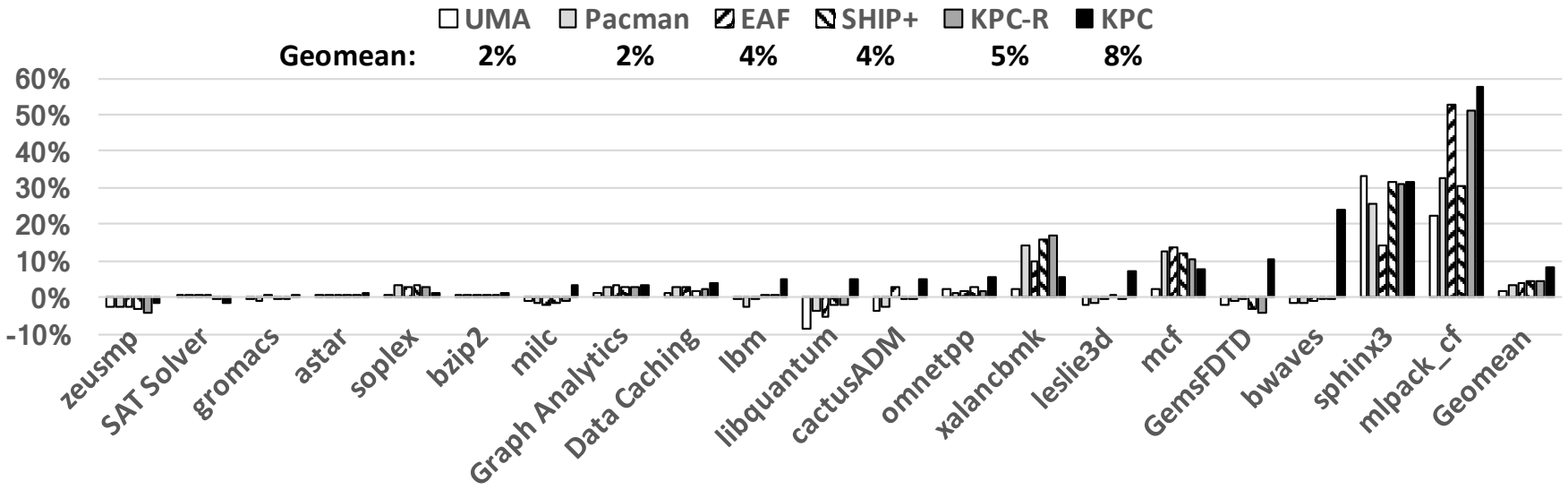
Evaluation

- Single-core performance
  - ❑ Baseline system: DA-AMPM prefetcher + LRU replacement
  - ❑ KPC-R shows similar performance to the state-of-the-art replacement algorithms without relying on PCs



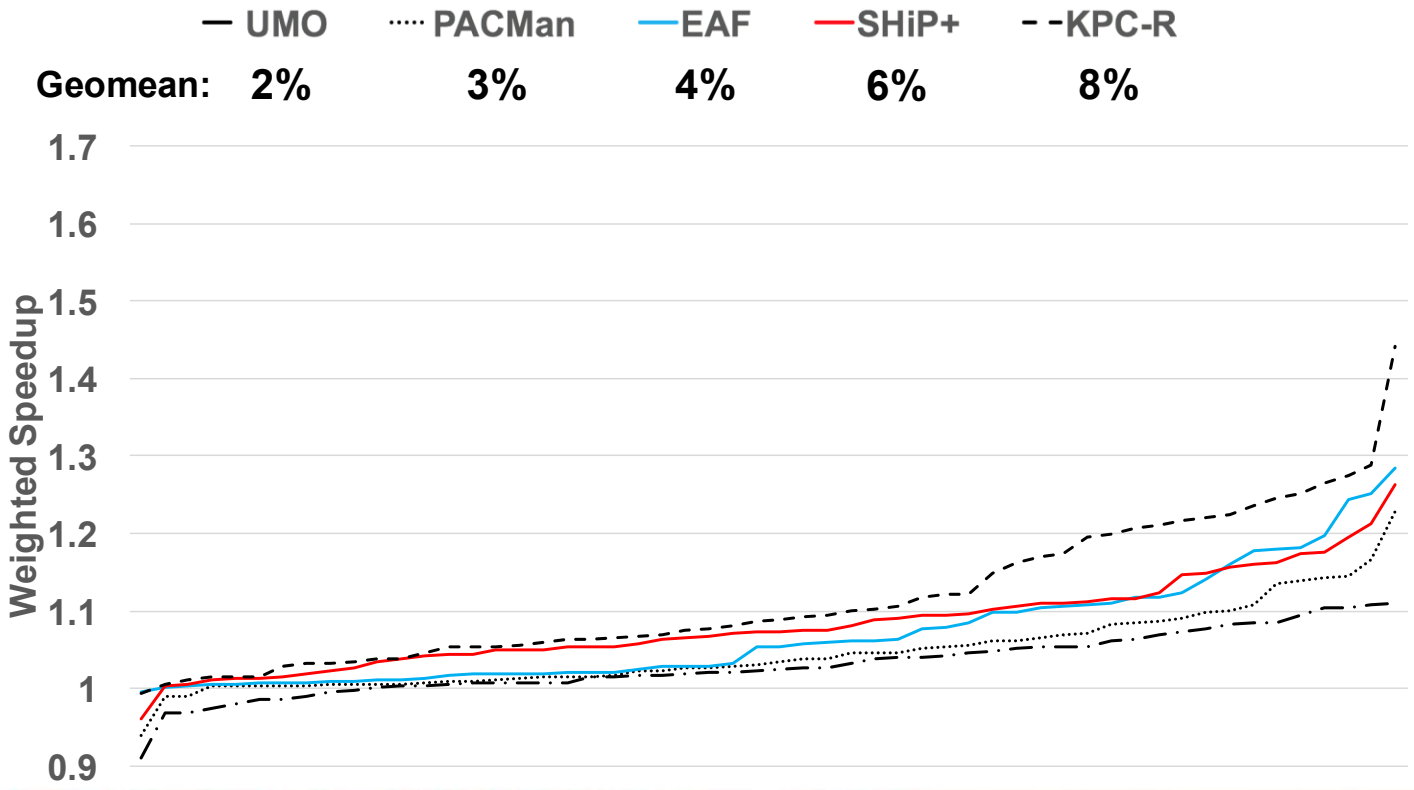
Evaluation

- Single-core performance
  - ❑ Baseline system: DA-AMPM prefetcher + LRU replacement
  - ❑ KPC (KPC-P + KPC-R) shows superior performance



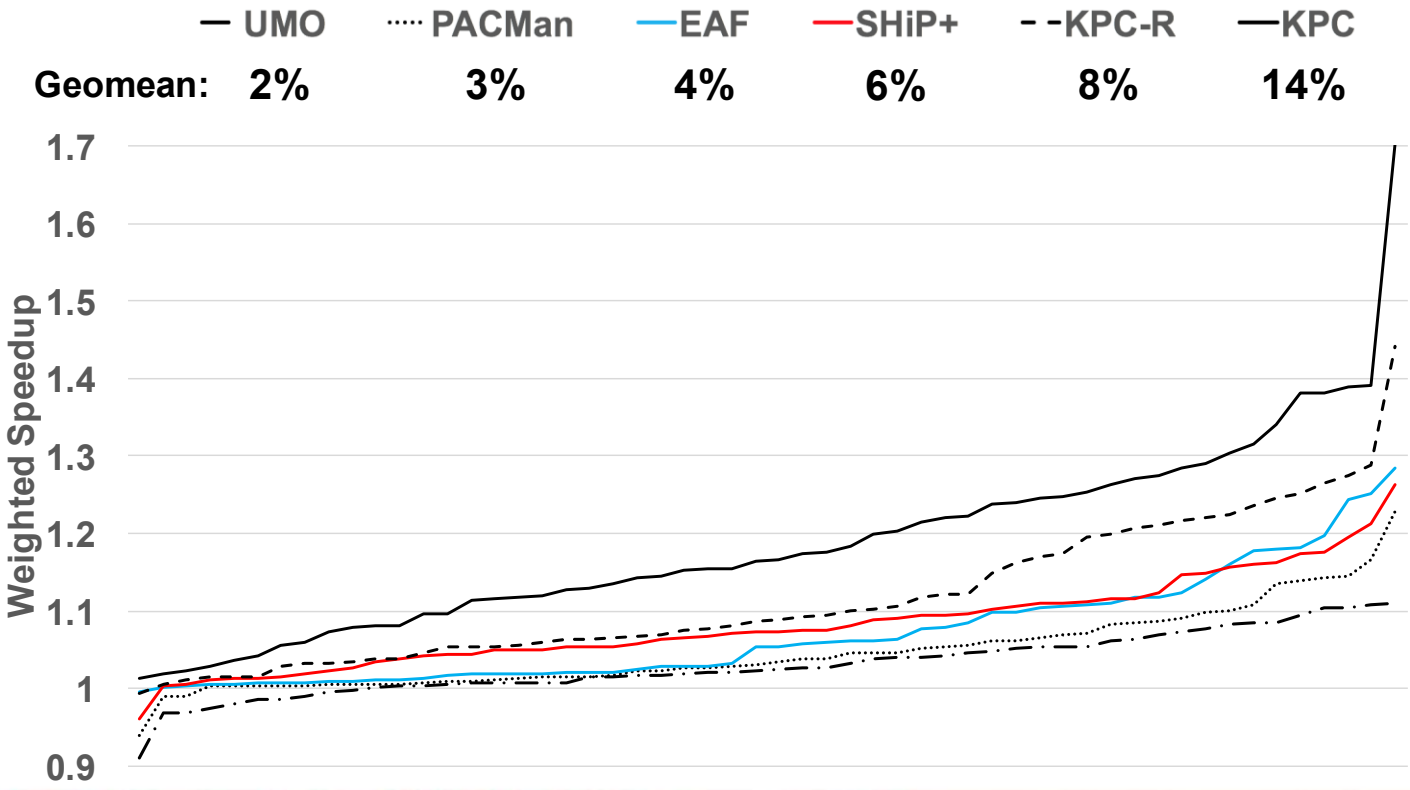
Evaluation

- Multi-core performance
  - ❑ KPC's performance gain becomes larger in multi-core environment
  - ❑ Direct result of accurate prefetching and replacement



Evaluation

- Multi-core performance
  - ❑ KPC's performance gain becomes larger in multi-core environment
  - ❑ Direct result of accurate prefetching and replacement



Conclusion

- Kill the PC reconstructs program behavior in the multi-level cache hierarchy
  - ❑ Prefetcher confidence guides the LLC replacement policy
  - ❑ Replacement policy decouples demand and prefetch requests
- Prefetching and replacement policy are tightly integrated into one unified solution
- Provide a holistic cache management system

Ongoing Work

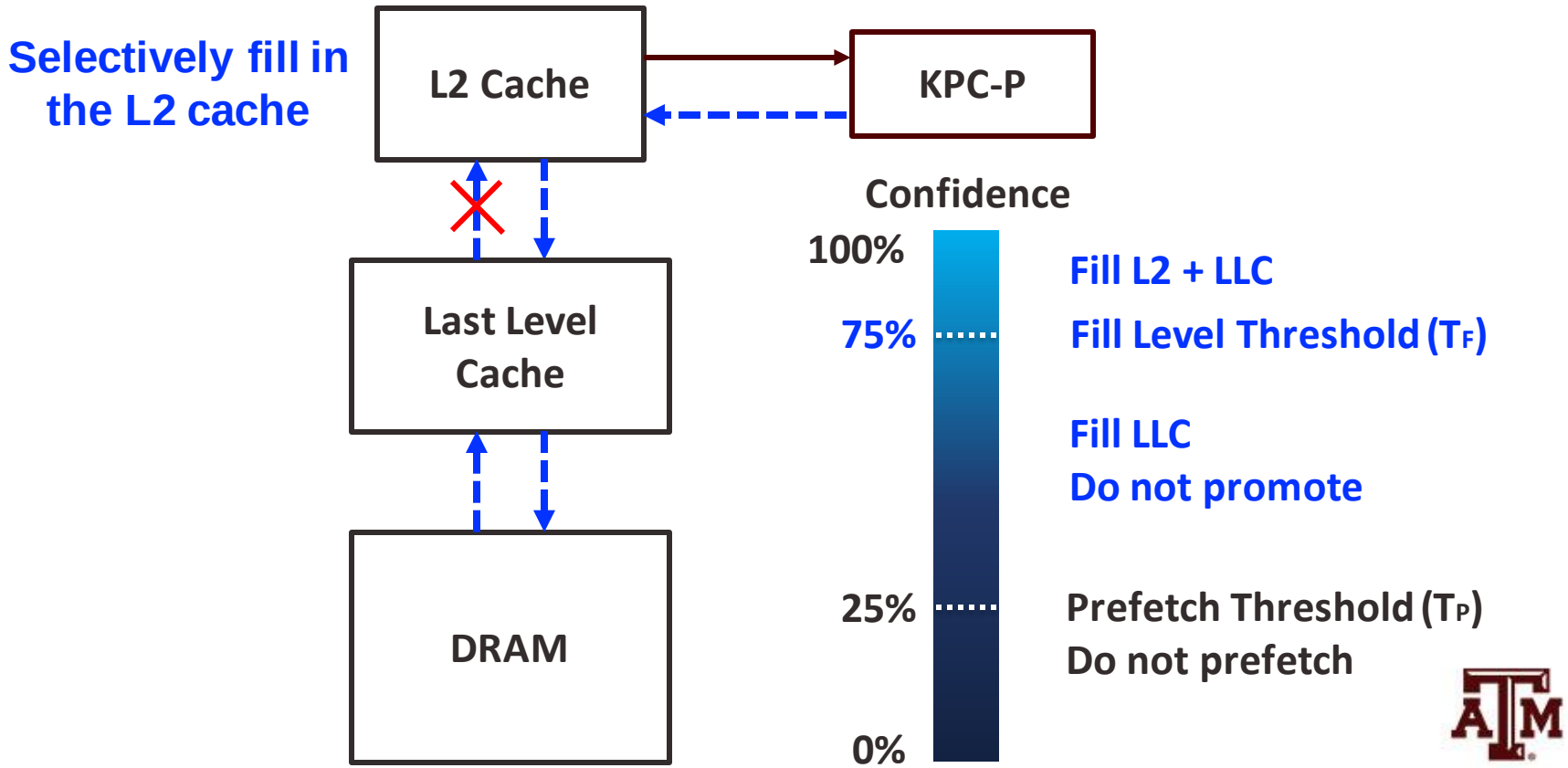
- Perceptron-based Prefetch Filtering[ISCA'19]
- Dynamic feature adaptation for microarchitectural speculation
- Page-balanced prefetching
- Hardware management of emerging NVM memory systems

Questions?

Backup

KPC-P: Placement Control

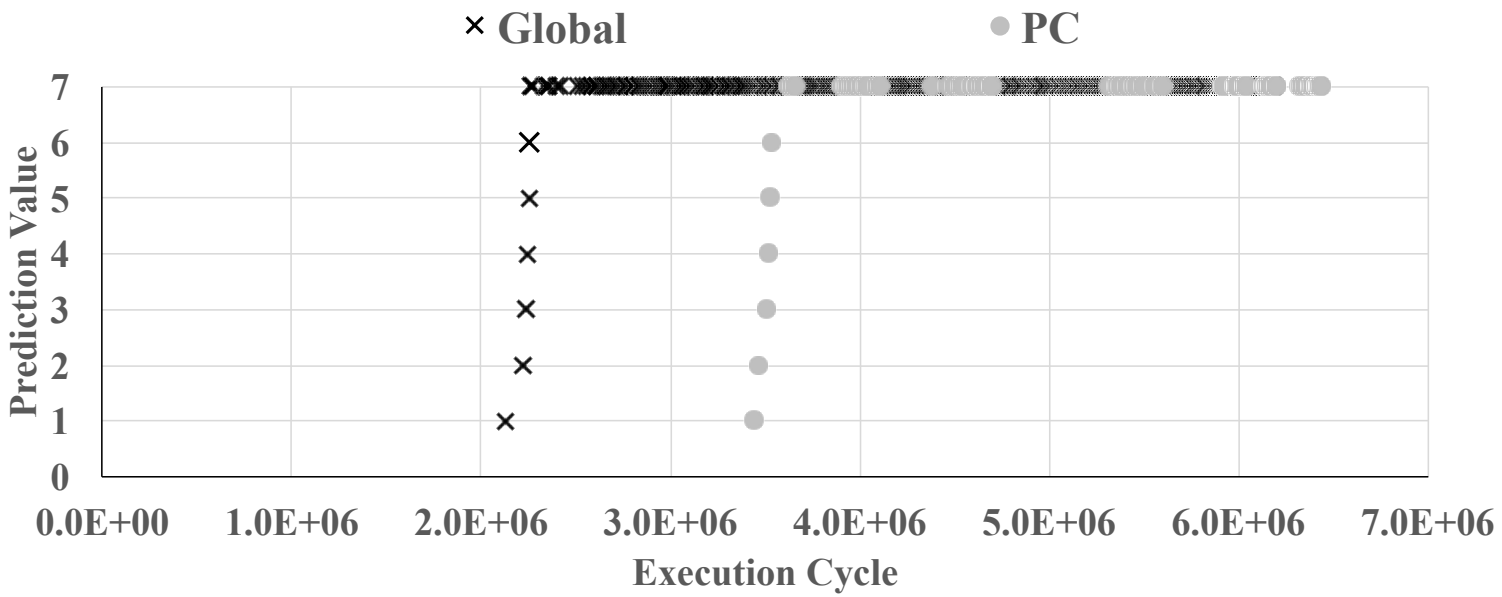
- Low confidence prefetches
  - ❑ A) Inaccurate prefetches B) Accurate but use distance is too long



Global Counters vs. PCs

- Why is a simple global hysteresis is sufficient?
  - ❑ Case 1) cactusADM

    Both Global and PC mechanisms quickly learn program behavior

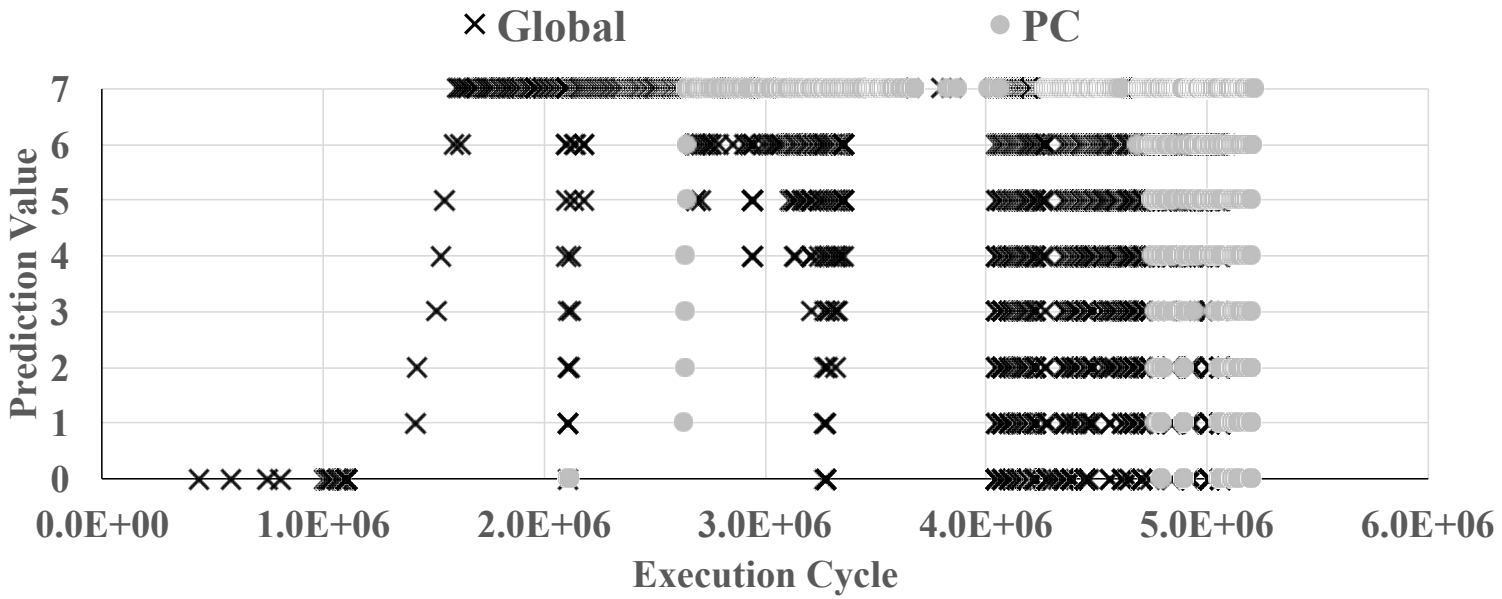


Global Counters vs. PCs

- Why is a simple global hysteresis is sufficient?
  - ❑ Case 2) sphinx3

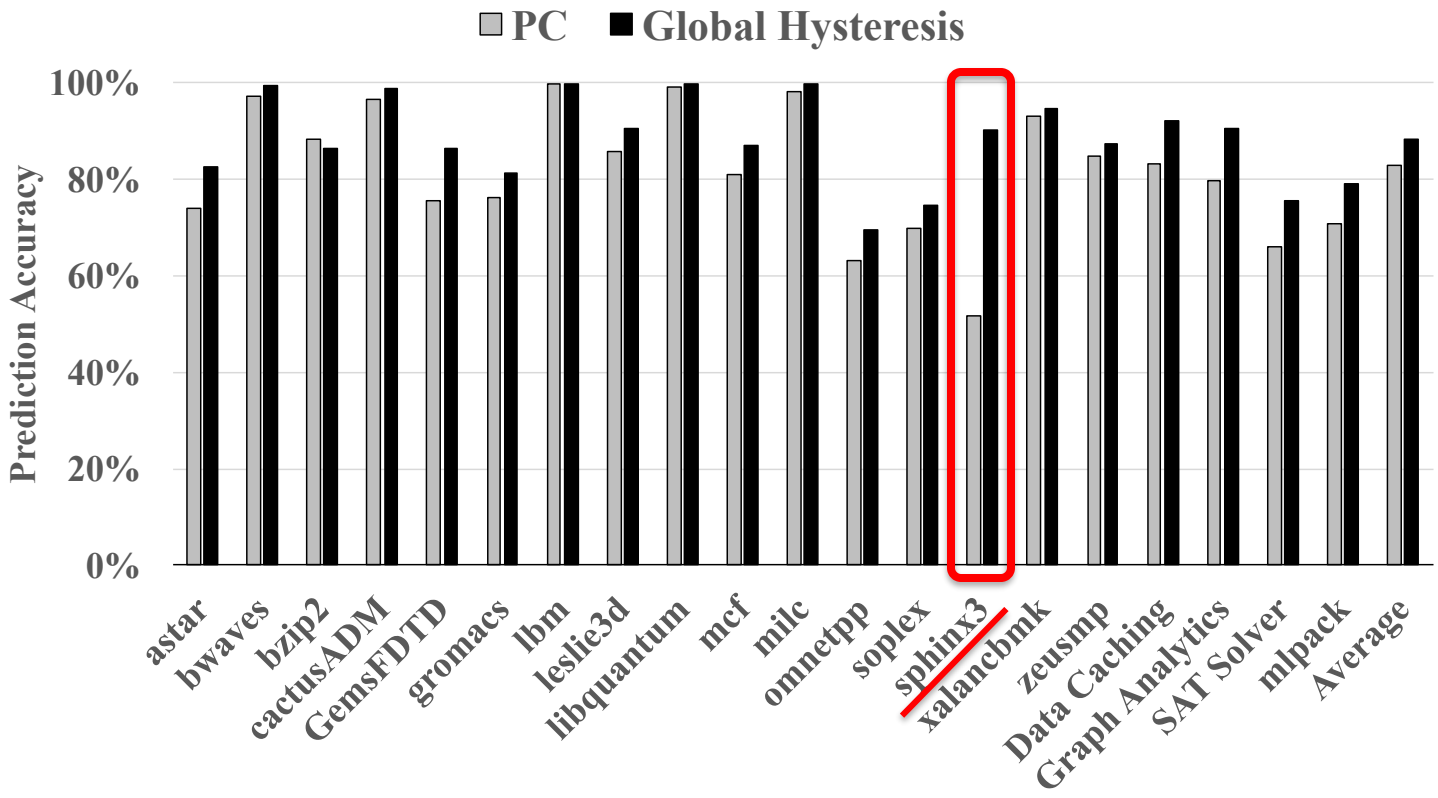
    Program behavior frequently changes over execution time

    KPC-R benefits from fast learning and adaptation



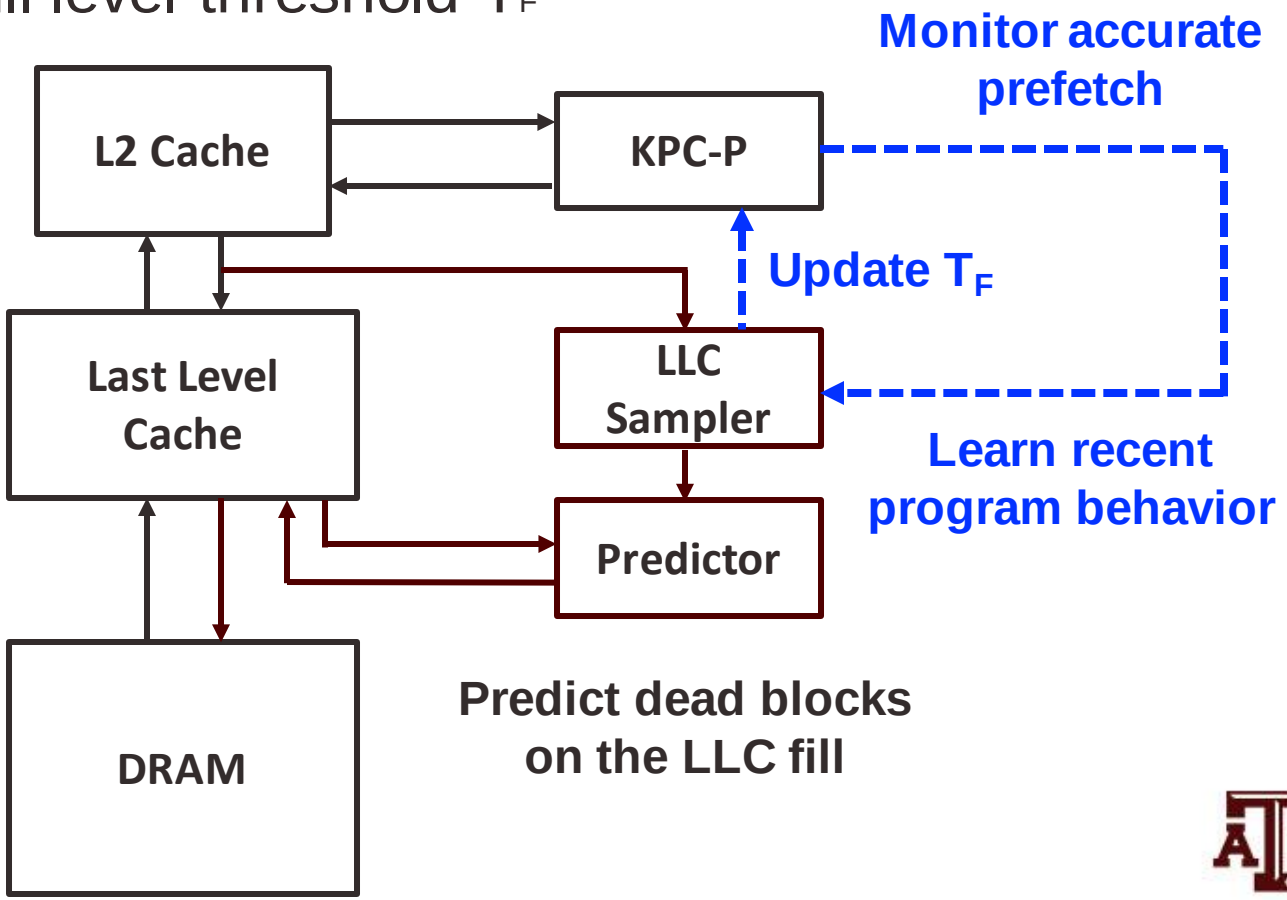
Global Counters vs. PCs

- Prediction accuracy: PC vs. Global Hysteresis



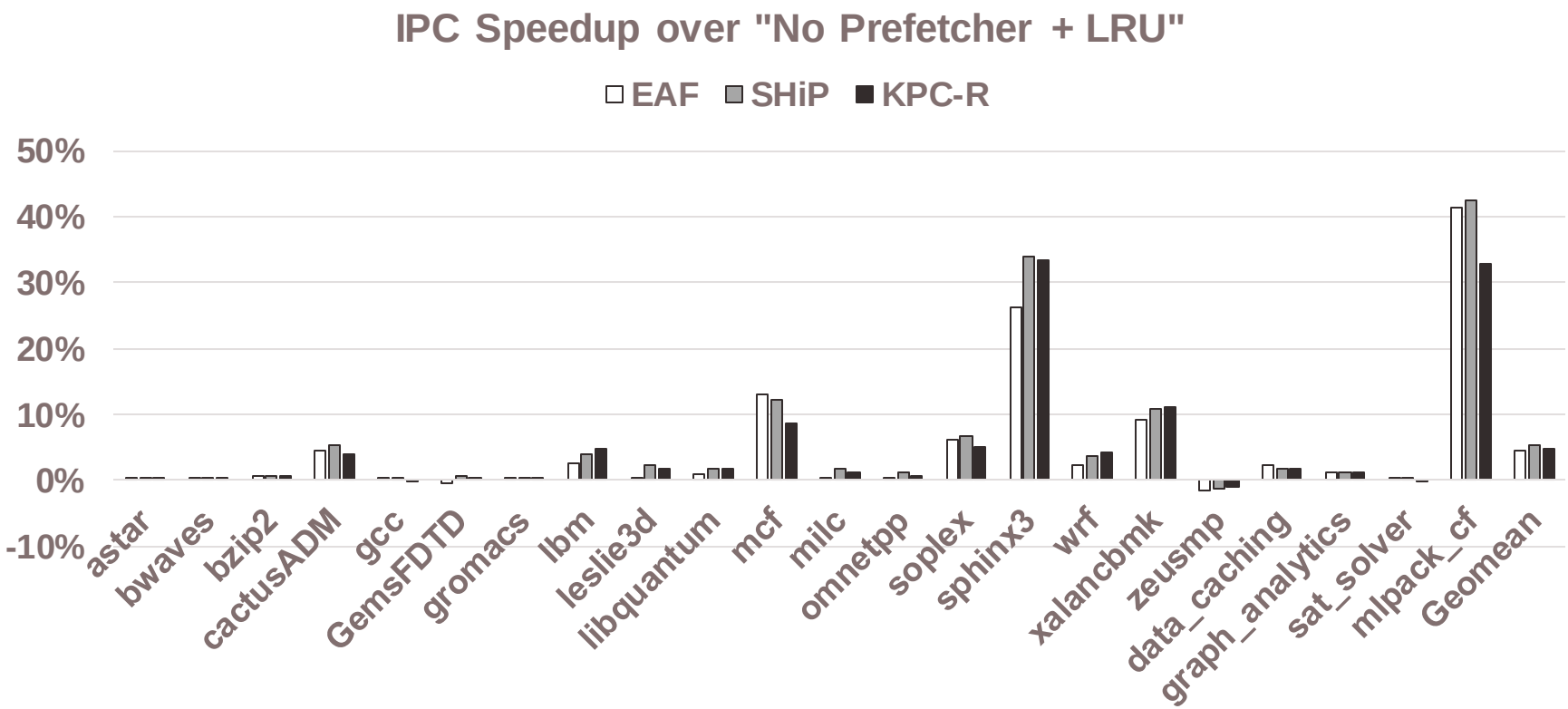
Fill Level Threshold

- Sampler monitors a subset of LLC accesses
- Update the fill level threshold T_F



KPC-R without Data Prefetcher

- Still shows similar or higher performance



Data Prefetchers

- KPC-R performance with recent data prefetchers
  - ❑ Similar or higher performance gain
  - ❑ Performance is maximized when KPC-R is integrated with KPC-P