ATHENA

Synergizing Data Prefetching and Off-Chip Prediction via Online Reinforcement Learning

Rahul Bera, Zhenrong Lang

Caroline Hengartner, Konstantinos Kanellopoulos, Rakesh Kumar,
Mohammad Sadrosadati, Onur Mutlu

https://github.com/CMU-SAFARI/Athena

ETH zürich SAFARI NTNU

Executive Summary (I)

Background

- **Long memory access latency** significantly **limits performance** of modern high-performance processors
- **Prefetching** and **off-chip prediction** (OCP) are two orthogonal techniques proposed to hide long memory access latency

Key Insights

1. OCP and prefetching provide **complementary** performance benefits
2. Naively combining OCP with prefetching often **fails** to realize their full performance potential
3. Existing coordination policies leave a **large performance potential** behind

Goal

- **Autonomously coordinates** off-chip prediction with **multiple prefetchers**
- Provides **consistent performance benefits**, regardless of workloads and system configurations

Executive Summary (II)

Our Proposal

Athena

- Models the coordination between prefetchers and off-chip predictor (OCP) as a **reinforcement learning** (RL) problem
- **Observes multiple system-level** features over an epoch of program execution
- **Takes a coordination action** (i.e., enabling the prefetcher and/or OCP, and adjusting prefetcher aggressiveness)
- **Receives a numerical reward** to autonomously and continuously learn

Contributions

1. Introduces a **composite** reward framework that **isolates** the true **impact of Athena's own action** from **inherent variations in workload** behavior

2. Works as a prefetcher-OCP **coordinator** and a prefetcher **throttler**, at the same time, **without any additional hardware**

Evaluation

- Evaluated with **100** workloads, **6** prefetchers, **3** OCPs, **4** cache designs, and **a wide range** of main memory bandwidth configurations
- **Consistently** outperforms both heuristic- and learning-based policies in **every** prefetcher/OCP/bandwidth configurations on average by **3.6% - 10.3%**

https://github.com/CMU-SAFARI/Athena

Outline

Background

Motivation

Athena

Evaluation

Conclusion

Key Problem and Its Potential Solutions

Long memory access latency

remains a key performance bottleneck in modern processors

Key latency hiding techniques:

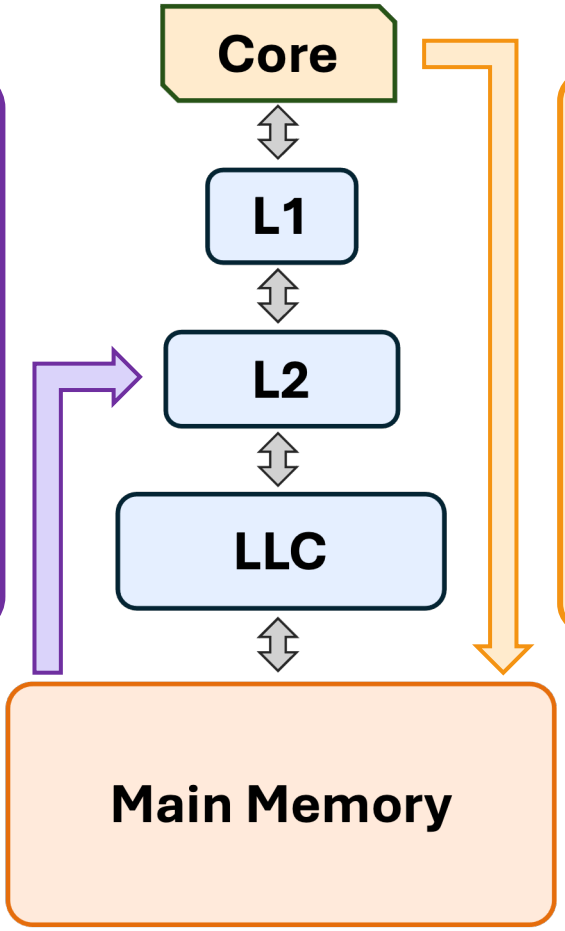
1. **Data prefetching**
2. **Off-chip prediction**

Data Prefetching and Off-Chip Prediction

Data prefetcher

Predicts address of future memory requests

- Hides the memory access latency
- Bandwidth overhead and cache pollution



Off-chip predictor

Predicts whether a given memory request would go off-chip

- Often more accurate predictions
- Lower timeliness than a prefetcher

How do they behave together?

Outline

Background

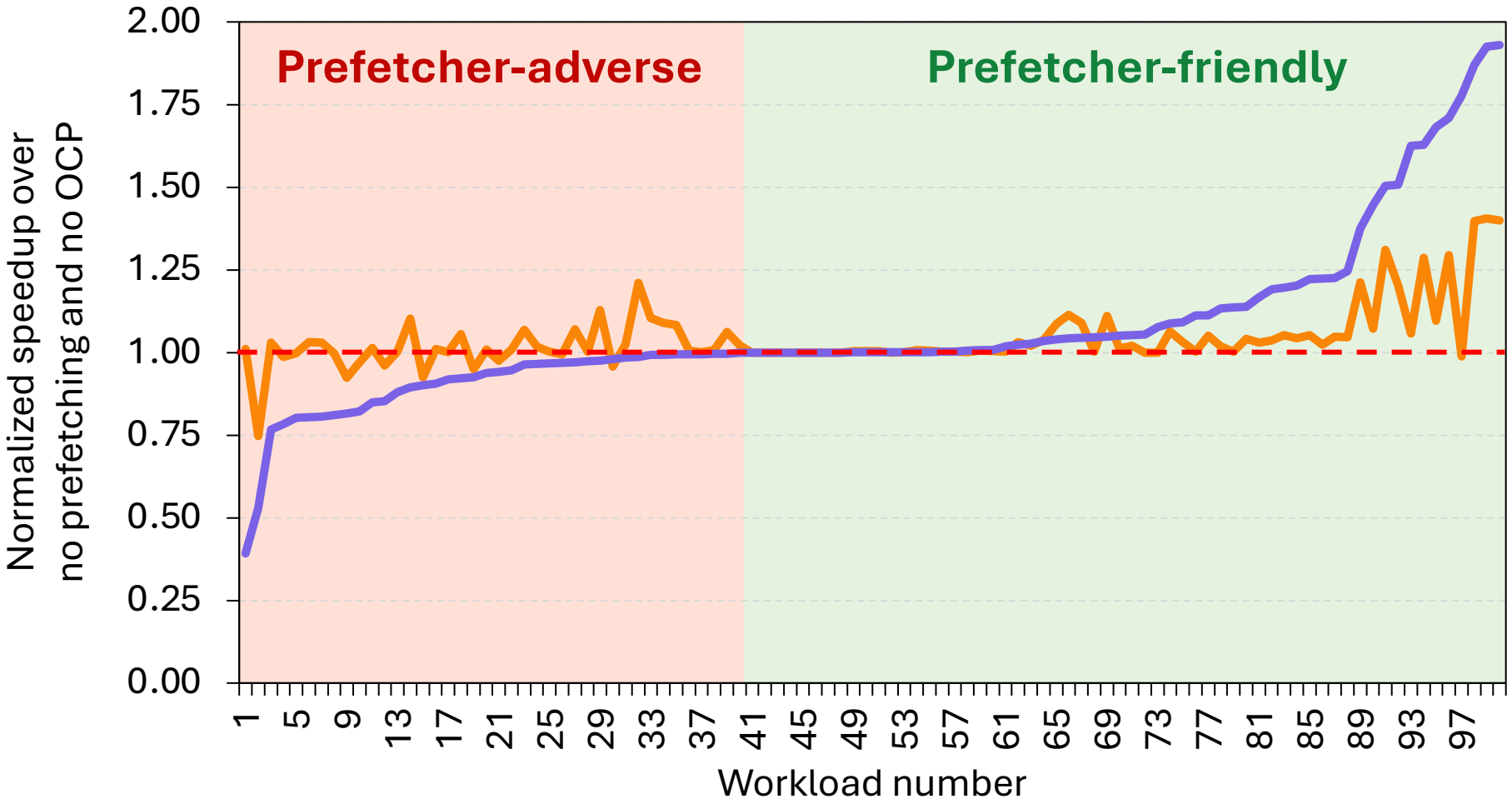
Motivation

Athena

Evaluation

Conclusion

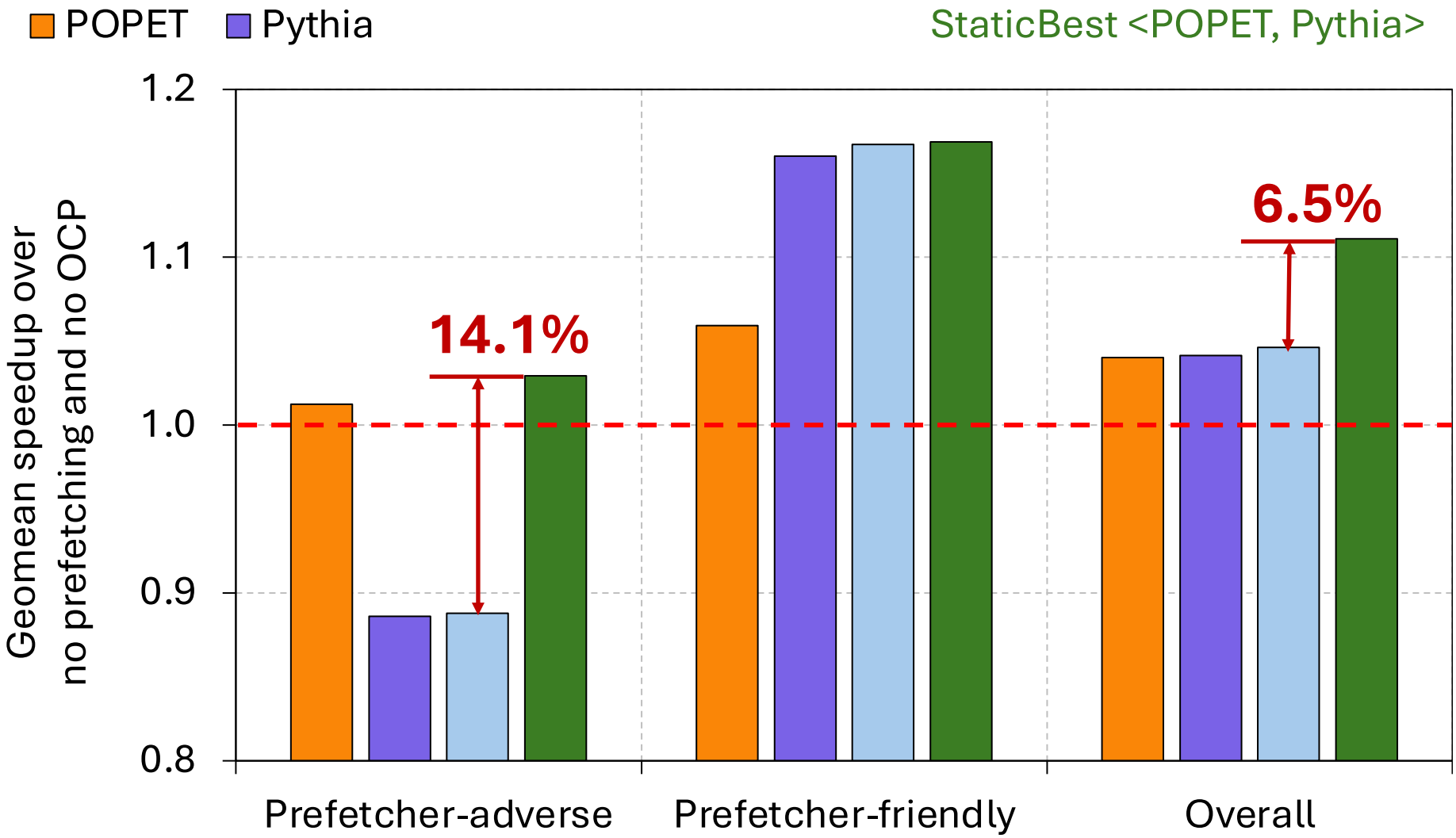
Ob. 1: Data Prefetching and Off-Chip Prediction Provide Complementary Performance Benefits



Ob. 1: Data Prefetching and Off-Chip Prediction Provide Complementary Performance Benefits

Data prefetching and off-chip prediction provide complementary performance benefits

Ob. 2: Naively Combining OCP with Prefetching Fails to Realize their Full Performance Potential



Ob. 2: Naively Combining OCP with Prefetching Fails to Realize their Full Performance Potential

Naively combining OCP with prefetching often fails to realize their full performance potential

Ob. 3: Existing Coordination Policies Fall Short

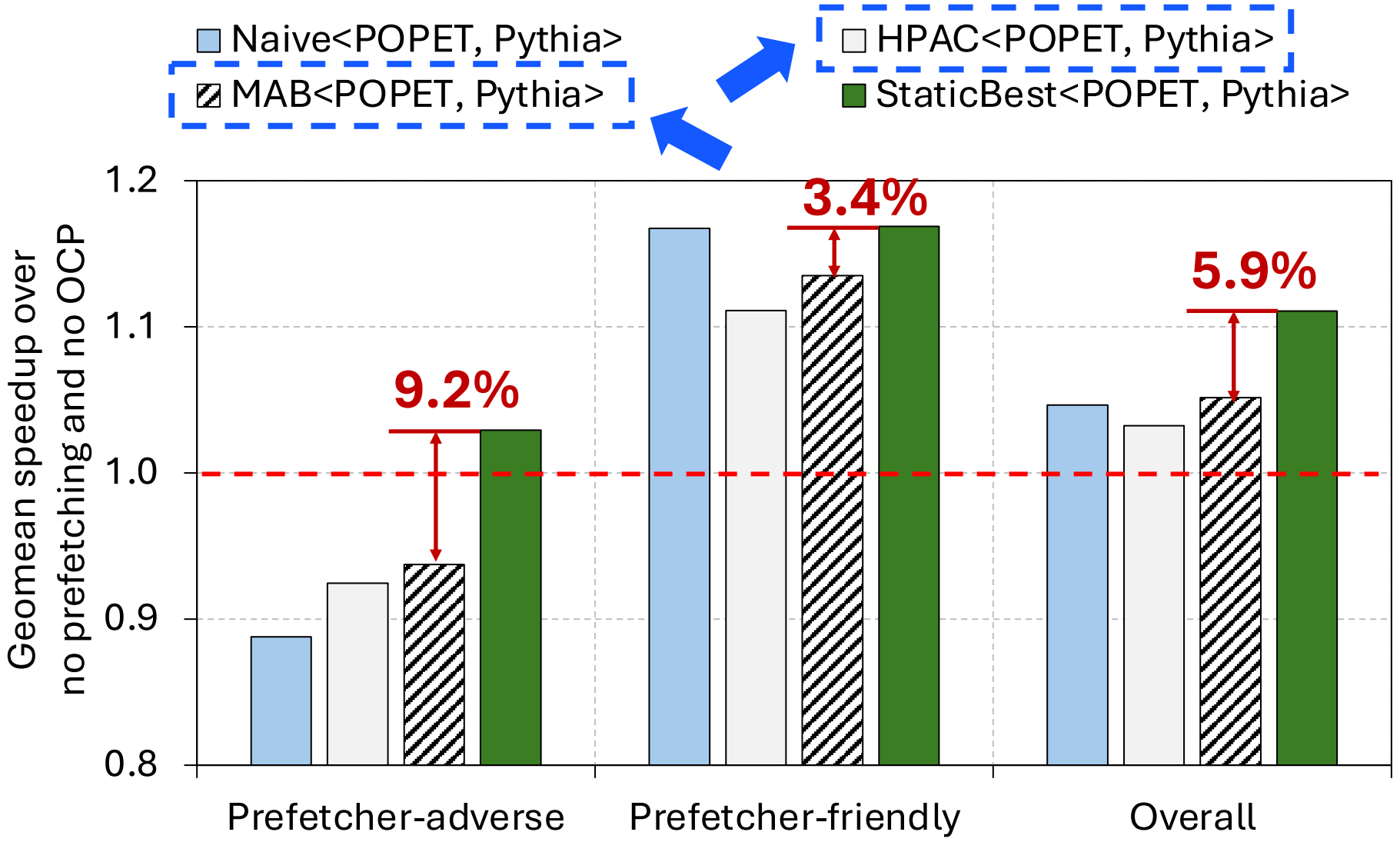
TLP [Jamet+, HPCA'24] is the **only** technique proposed to **combine OCP with a prefetcher**

Ob. 3: Existing Coordination Policies Fall Short

TLP [Jamet+, HPCA'24] is the **only** technique proposed to **combine OCP with a prefetcher**

Can coordinate OCP with prefetcher **employed only at the L1 data cache**

Ob. 3: Existing Coordination Policies Fall Short

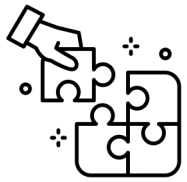


Ob. 3: Existing Coordination Policies Fall Short

Existing coordination policies leave a large performance potential behind

Our Goal

Design a **holistic framework** that



Autonomously synergizes OCP with
multiple prefetchers throughout the cache hierarchy

To deliver **consistent performance benefits**,
regardless of workloads and system configuration

Our Proposal

ATHENA

Formulates the coordination between prefetchers and OCP as a **reinforcement learning** problem

Outline

Background

Motivation

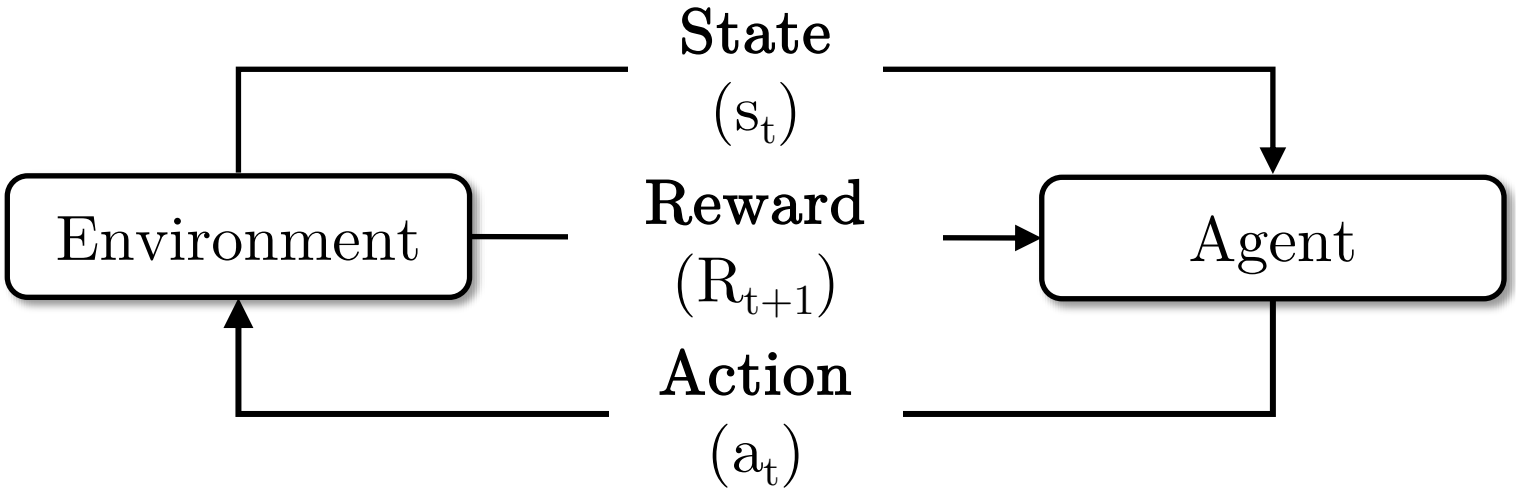
Athena

Evaluation

Conclusion

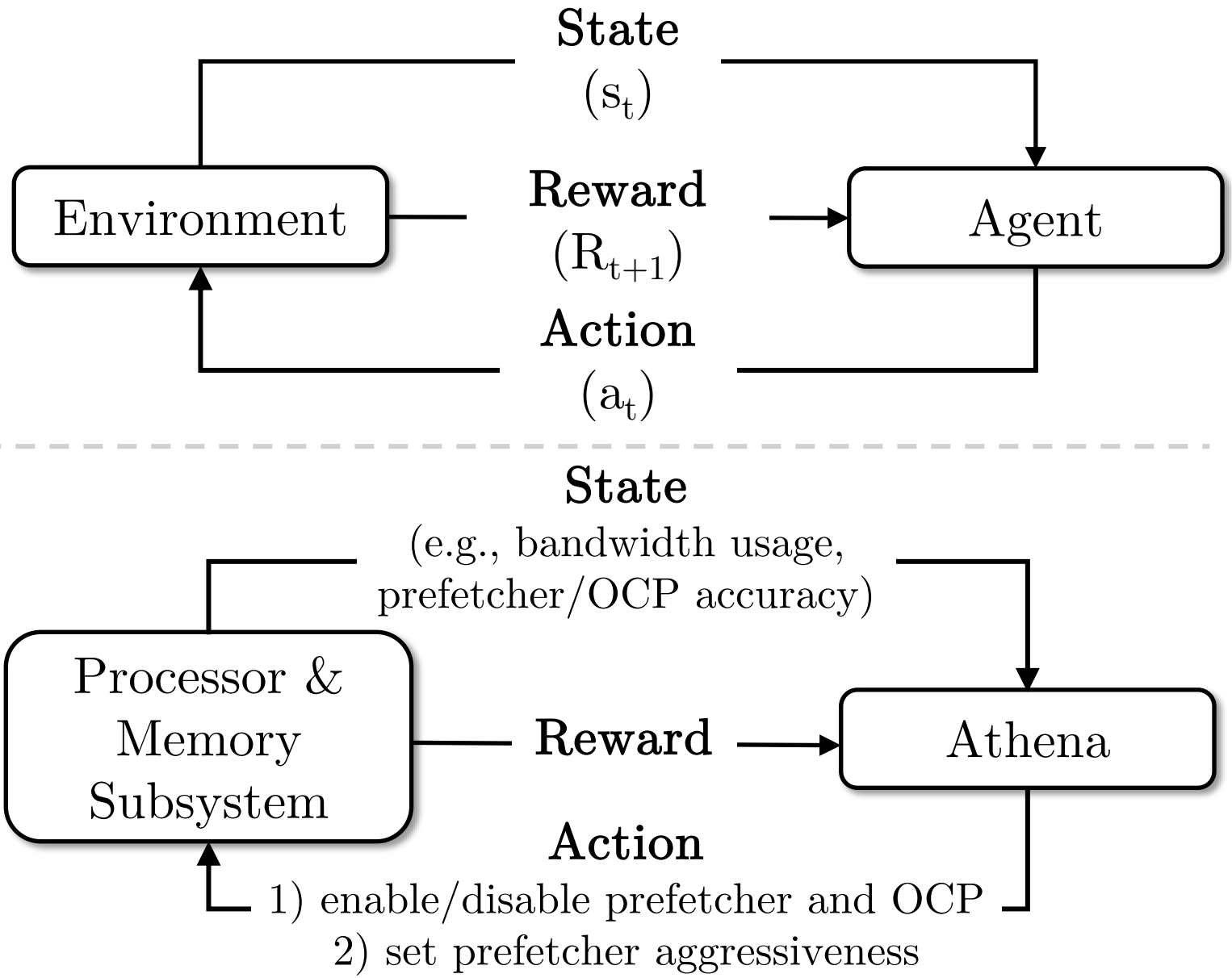
Basics of Reinforcement Learning (RL)

- An algorithmic approach to learn to take an **action** in a given **state** to maximize a numerical **reward**



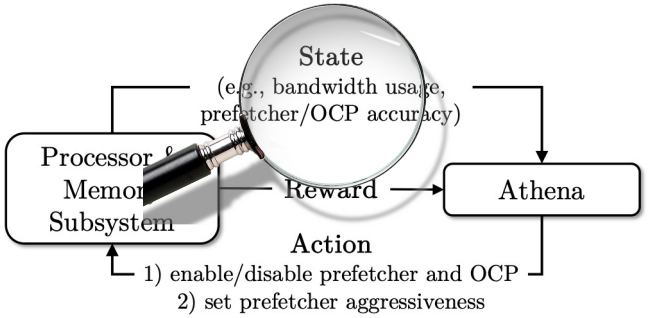
- Agent stores **Q-values** for *every* state-action pair
  - **Expected reward** for taking an action in a state
- Given a state, the agent selects the action that provides **the highest** Q-value

Formulating Coordination as RL



What is State?

- **Vector** of system-level features
- State := [Feature 1, Feature 2, ...]



- Captures **runtime behavior** of the memory subsystem

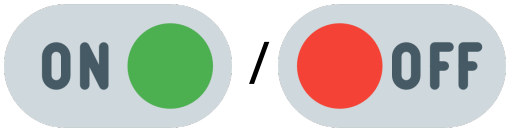
Example Features

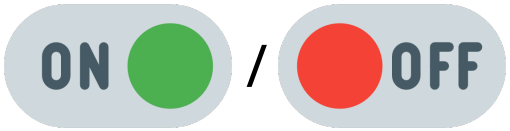
- Prefetcher accuracy
- OCP accuracy
- Bandwidth usage
- Cache pollution
- Prefetch bandwidth
- OCP bandwidth
- Demand bandwidth

Automated Design-Space Exploration

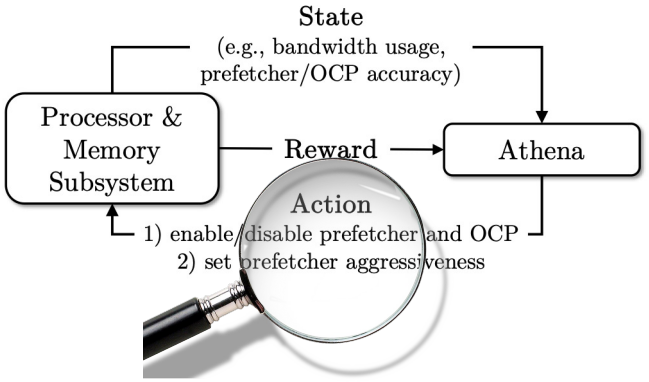
Perform offline **feature selection** to determine the **final** subset of features that Athena uses to construct the state vector

What is Action?

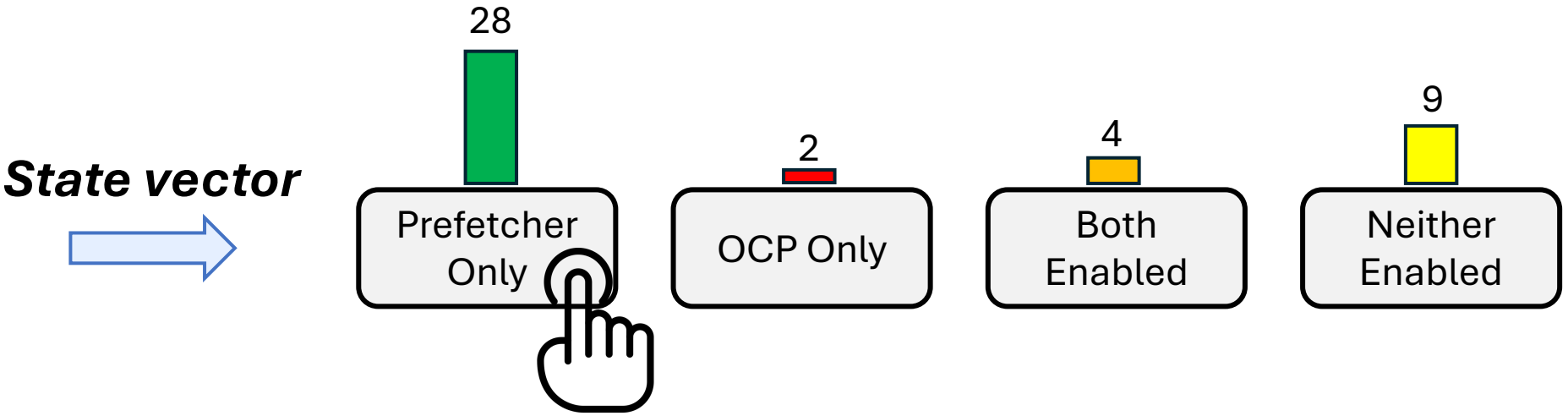
- Prefetcher 

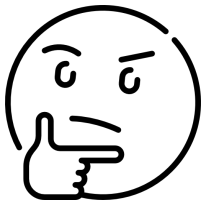
- OCP 

- Adjust the aggressiveness of prefetcher



What is Action?





The **magnitude** of the selected action's Q-value implicitly encodes Athena's **confidence** in taking that action

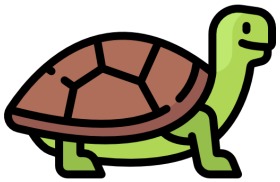
>> avg(, ,)

Q-Value-Driven Prefetcher Aggressiveness Control

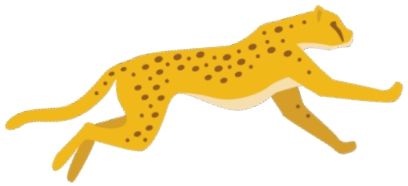
Key Idea: Use the **Q-value difference** to control prefetcher aggressiveness

1. Calculate Q-value difference
   $\Delta Q \leftarrow Q(a^*) - \text{avg}(Q(\text{remaining actions}))$
2. Drive prefetcher aggressiveness using ΔQ

Low ΔQ → Low aggressiveness



High ΔQ → High aggressiveness



Q-Value-Driven Prefetcher Aggressiveness Control

Key Idea: Use the **Q-value difference** to control prefetcher aggressiveness

1. Calculate Q-value difference
   $\Delta Q \leftarrow Q(a^*) - \text{avg (remaining actions)}$
2. Drive prefetcher aggressiveness using ΔQ

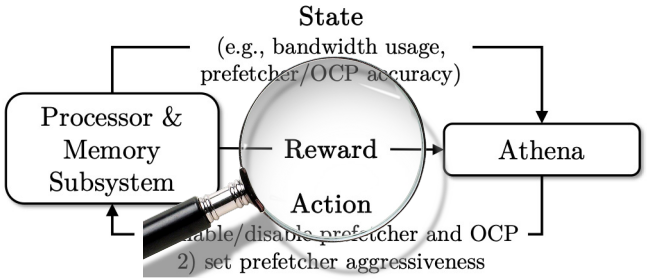
Low $\Delta Q \rightarrow$ Low aggressiveness

High $\Delta Q \rightarrow$ High aggressiveness

Athena works as a prefetcher-OCP coordinator and a prefetcher throttler, at the same time, using the same hardware

What is Reward?

- Defines the **objective** of Athena



- Intuitive reward: instructions committed per cycle (IPC)

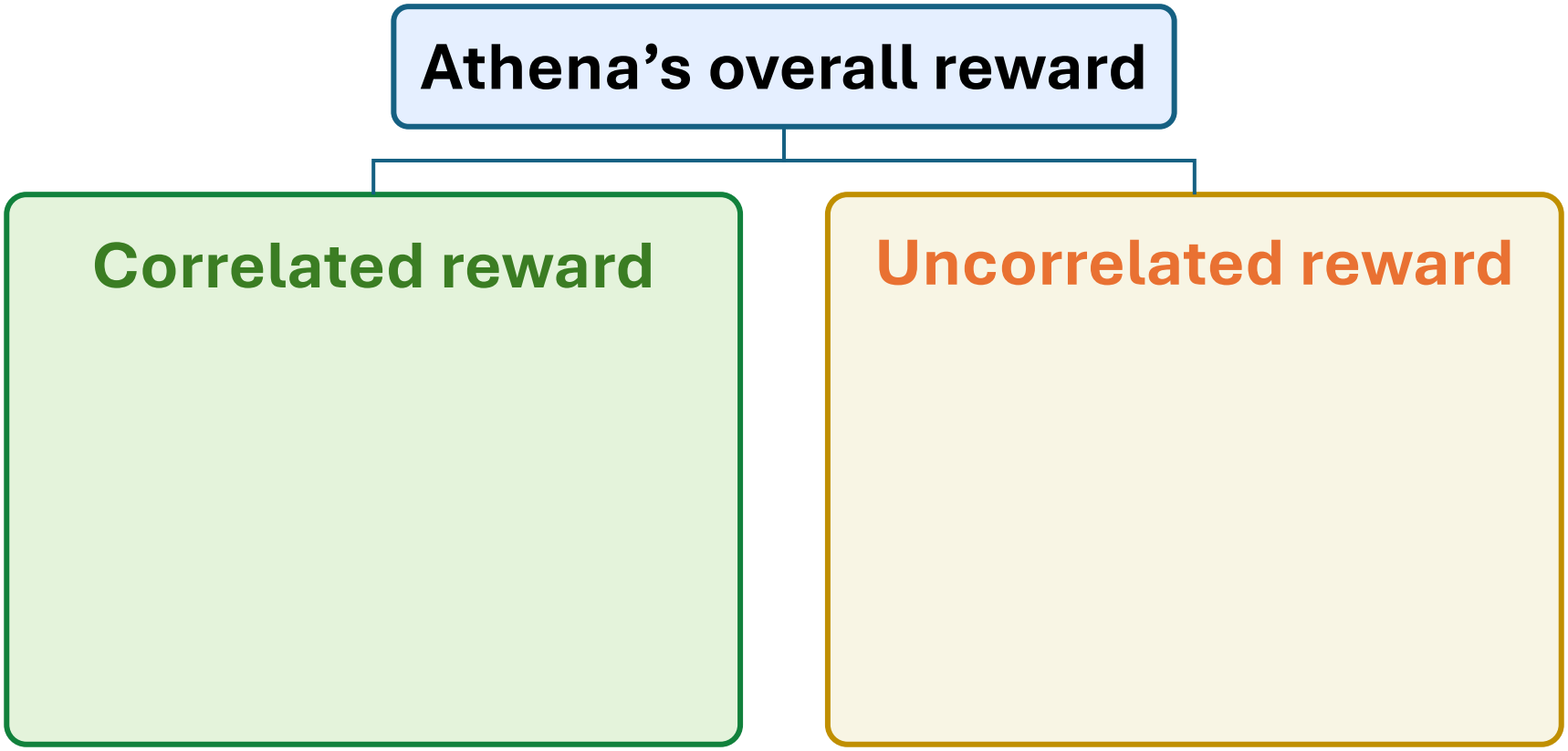
Change in IPC

Coordination actions taken by Athena

Inherent variation in workload behavior

Using IPC as **the sole reward** can be **unreliable** and may **mislead** the learned policy

Athena's Composite Reward Framework



Allows Athena to autonomously learn a coordination policy
by **isolating** the **true impact of its actions**
from **inherent variations** in workload behavior

Athena's Composite Reward Framework

Allows Athena to autonomously learn a coordination policy

Potentially broadly applicable to many other microarchitectural decision-making processes

Final Configuration of Athena

State

Four features

- Prefetcher accuracy
- OCP accuracy
- Bandwidth usage
- Prefetcher-induced cache pollution

Action

Coarse-grained action

- Enable prefetcher
- Enable OCP
- Enable both
- Enable none

Fine-grained action

- Q-value driven prefetcher aggressiveness control

Reward

Correlated reward

- Cycles
- LLC misses
- LLC miss latency

Uncorrelated reward

- Load instructions
- Mispredicted branches

More in the Paper

- **Detailed design of Athena**
  - Architecture to store and retrieve Q-values
- **Automated design space exploration**
  - Feature selection
  - Reward tuning
  - Hyperparameter tuning
- **Mechanism to measure system state**
  - Prefetcher accuracy
  - OCP accuracy
  - Cache pollution
  - Main memory bandwidth usage
- **Storage and latency overhead analysis**

More in the Paper

Athena: Synergizing Data Prefetching and Off-Chip Prediction via Online Reinforcement Learning

*Rahul Bera[1] *Zhenrong Lang[1] Caroline Hengartner[1] Konstantinos Kanellopoulos[1]
Rakesh Kumar[2] Mohammad Sadrosadati[1] Onur Mutlu[1]

[1]ETH Zürich [2]NTNU

https://www.arxiv.org/pdf/2601.17615

Outline

Background

Motivation

Athena

Evaluation

Conclusion

ChampSim Trace-Driven Simulation Methodology

Prefetchers

- **IPCP** [Pakalapati+, ISCA'20]
- **Berti** [Navarro-Torres+, MICRO'22]
- **Pythia** [Bera+, MICRO'21]
- **SPP+ PPF** [Bhatia+, ISCA'20]
- **SMS** [Somogyi+, ISCA'06]
- **MLOP** [Shakerinava+, DPC3'19]

System Configurations

- **# cores**: 1- 8 cores
- **Main memory bandwidth**: 1.6 GB/s – 12.8 GB/s
- **4 cache designs** with various prefetchers at different cache levels

Off-Chip Predictors

- **Hermes** [Bera+, MICRO'22]
- **HMP** [Yoaz+, ISCA'99]
- **TTP** [Jalili+, HPCA'22]

Coordination Policies

- **HPAC** [Ebrahimi+, MICRO'09]
- **MAB** [Yoaz+, ISCA'99]
- **TLP** [Jamet+, HPCA'24]

Workloads

- **100 single-core** workloads
- **180 four-core and eight-core** workload mixes

CMU-SAFARI / Athena

Type / to search

Code | Issues 1 | Pull requests | Agents | Actions | Projects | Security | Insights | Settings

Athena Public

Edit Pins | Watch 0 | Fork 1 | Star 2

main | 1 Branch | 2 Tags | Go to file t | Add file | Code

rahulbera Added Zenodo DOIs — 1cf9be6 · 3 days ago — 11 Commits

branch	initial commit	last month
config	initial commit	last month
inc	initial commit	last month
logo	Updated README	5 days ago
prefetcher	initial commit	last month
replacement	initial commit	last month
scripts	Updated the scripts to match the Fig numbers from the p...	5 days ago
src	initial commit	last month
.clang-format	initial commit	last month
.gitignore	Updated README	5 days ago
Makefile	initial commit	last month
README.md	Added Zenodo DOIs	3 days ago

About

A reinforcement learning based policy to dynamically coordinate off-chip predictor with multiple data prefetchers, as described in the HPCA2026 paper by Bera and Lang et al.: https://arxiv.org/abs/2601.17615

arxiv.org/pdf/2601.17615

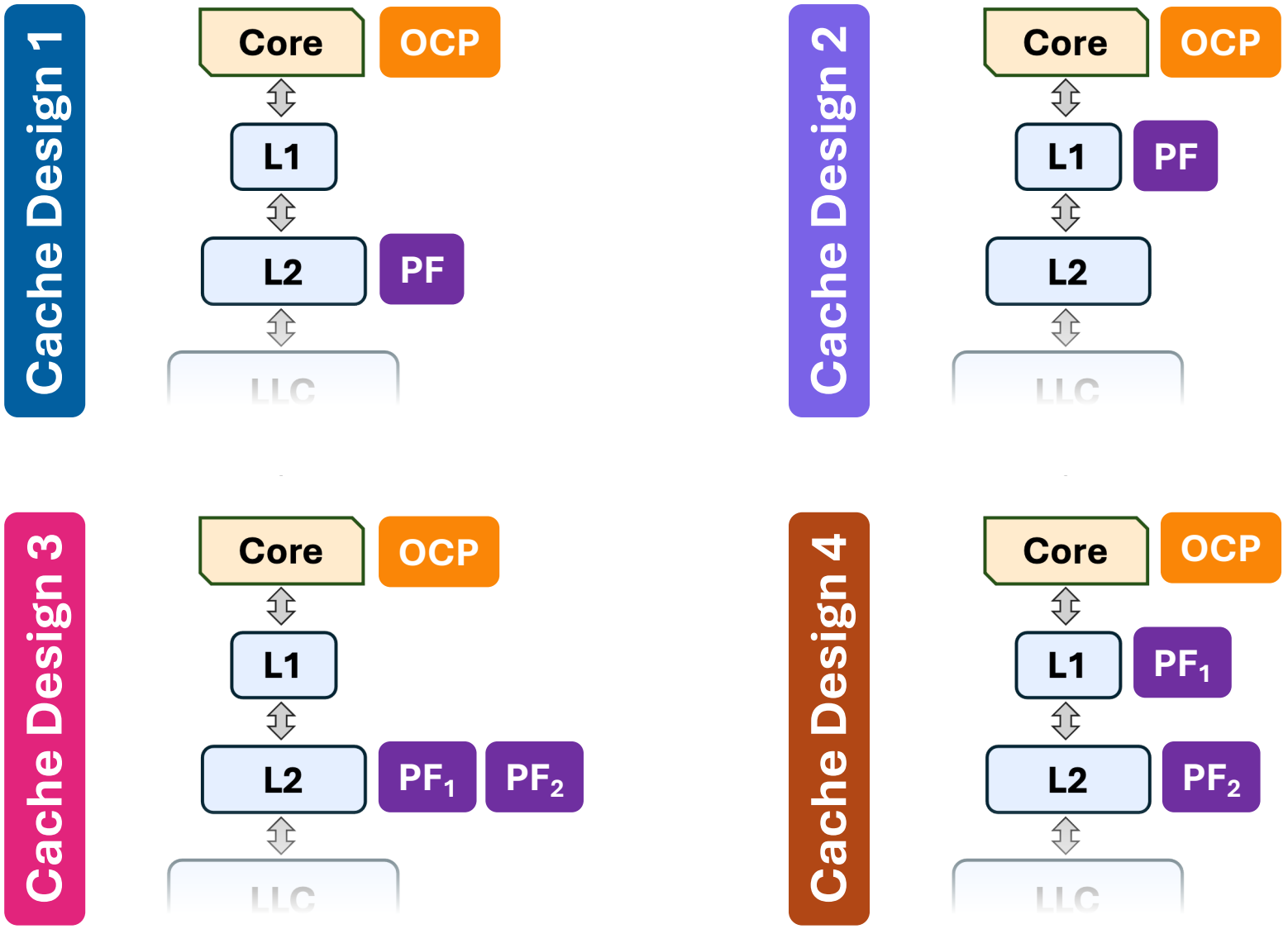
machine-learning | reinforcement-learning | computer-architecture | microarchitecture | prefetching | champsim-simulator | champsim-tracer

Readme
Activity
Custom properties
2 stars
0 watching
1 fork
Report repository

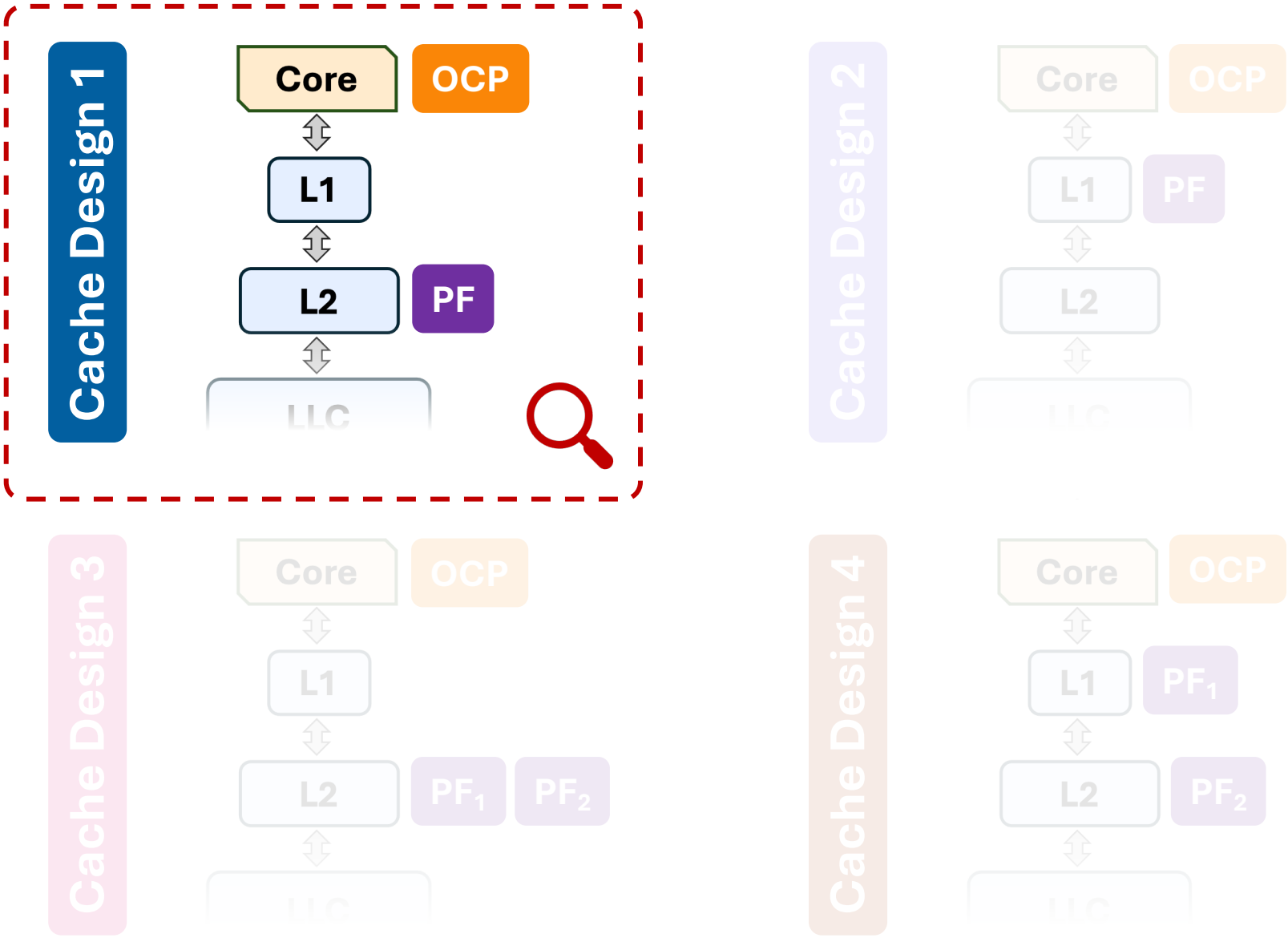
https://github.com/CMU-SAFARI/Athena

Open-sourced and artifact-evaluated with all three badges

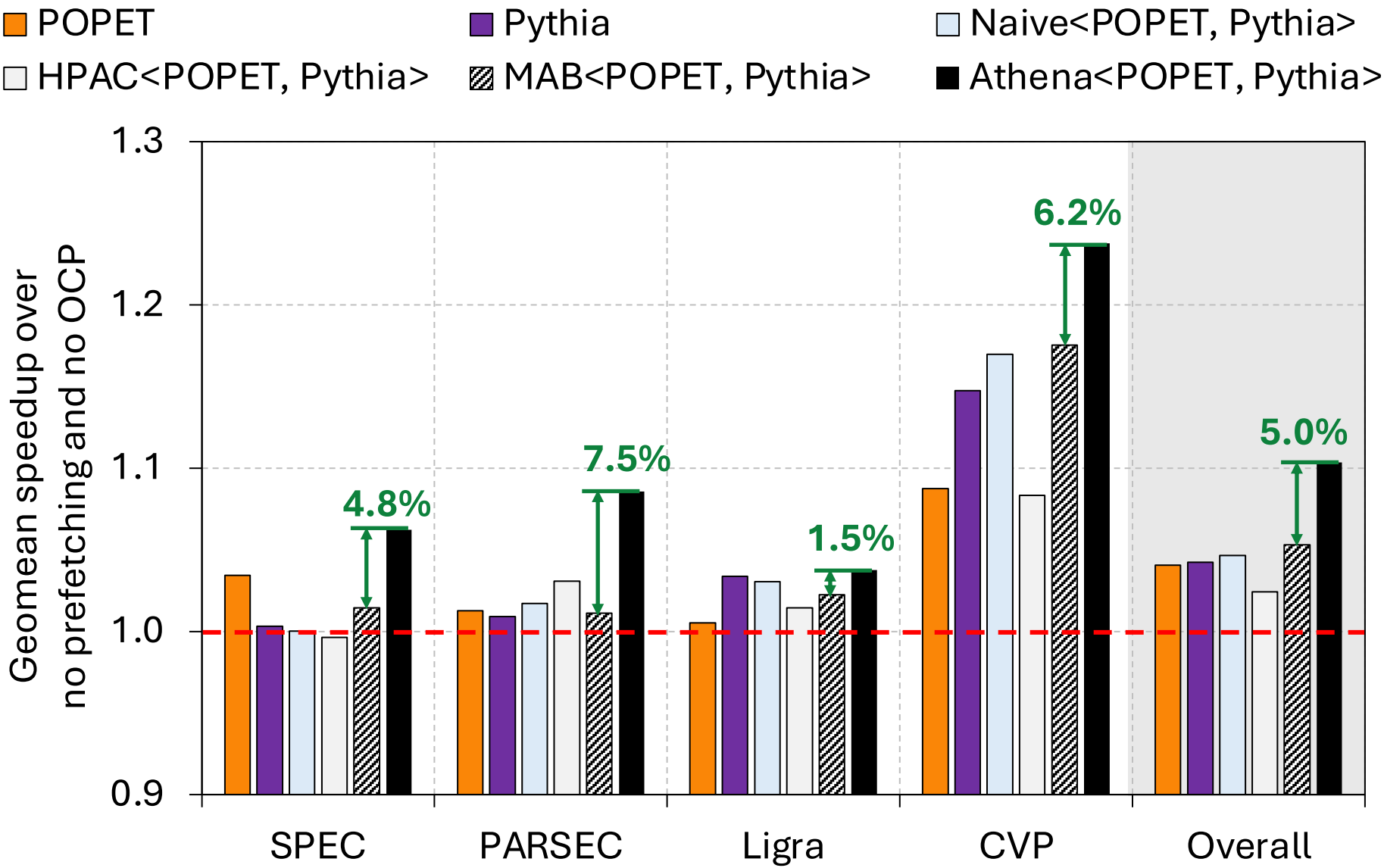
Evaluated Cache Designs



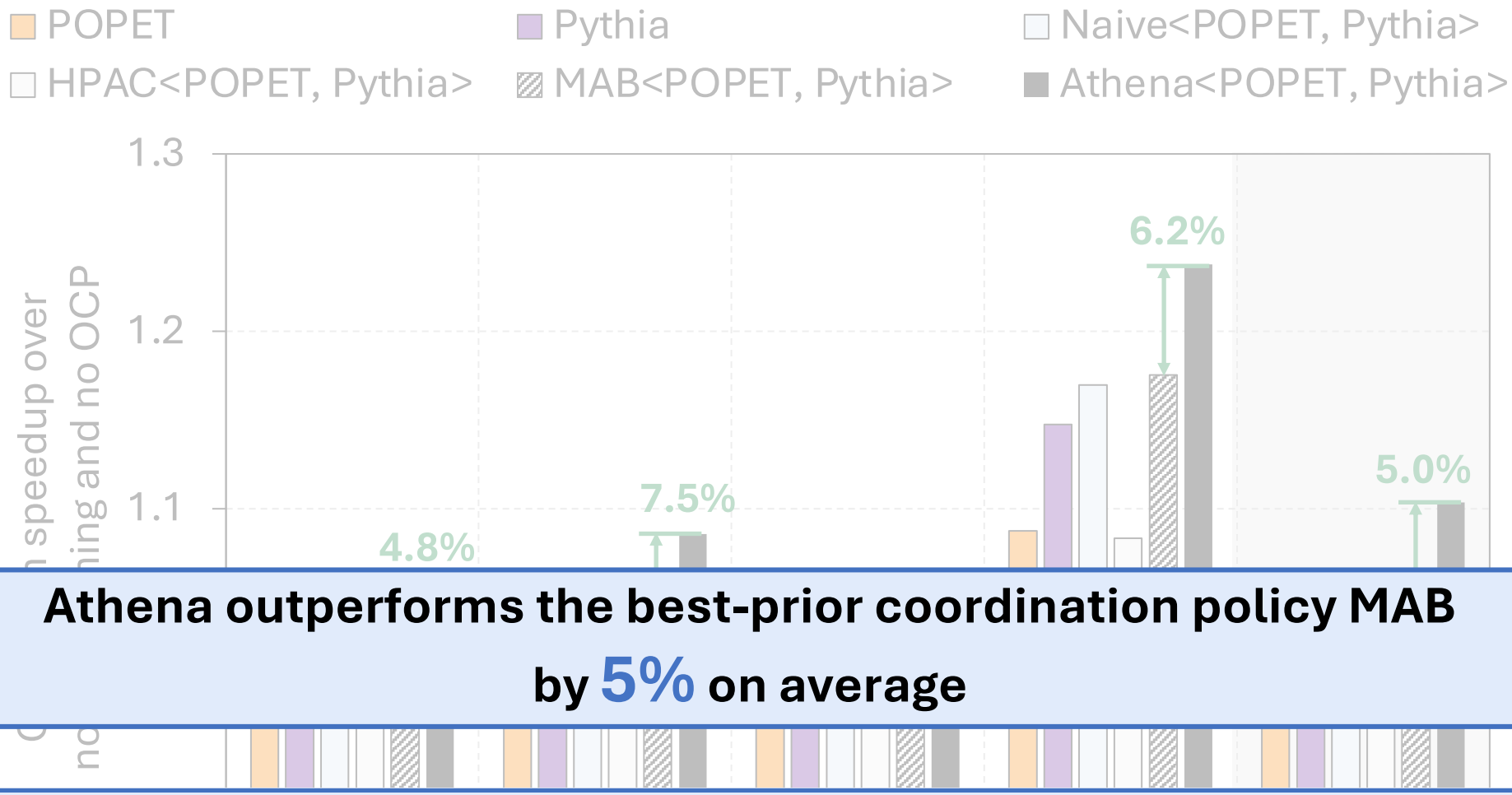
Evaluated Cache Designs



Speedup in Cache Design 1



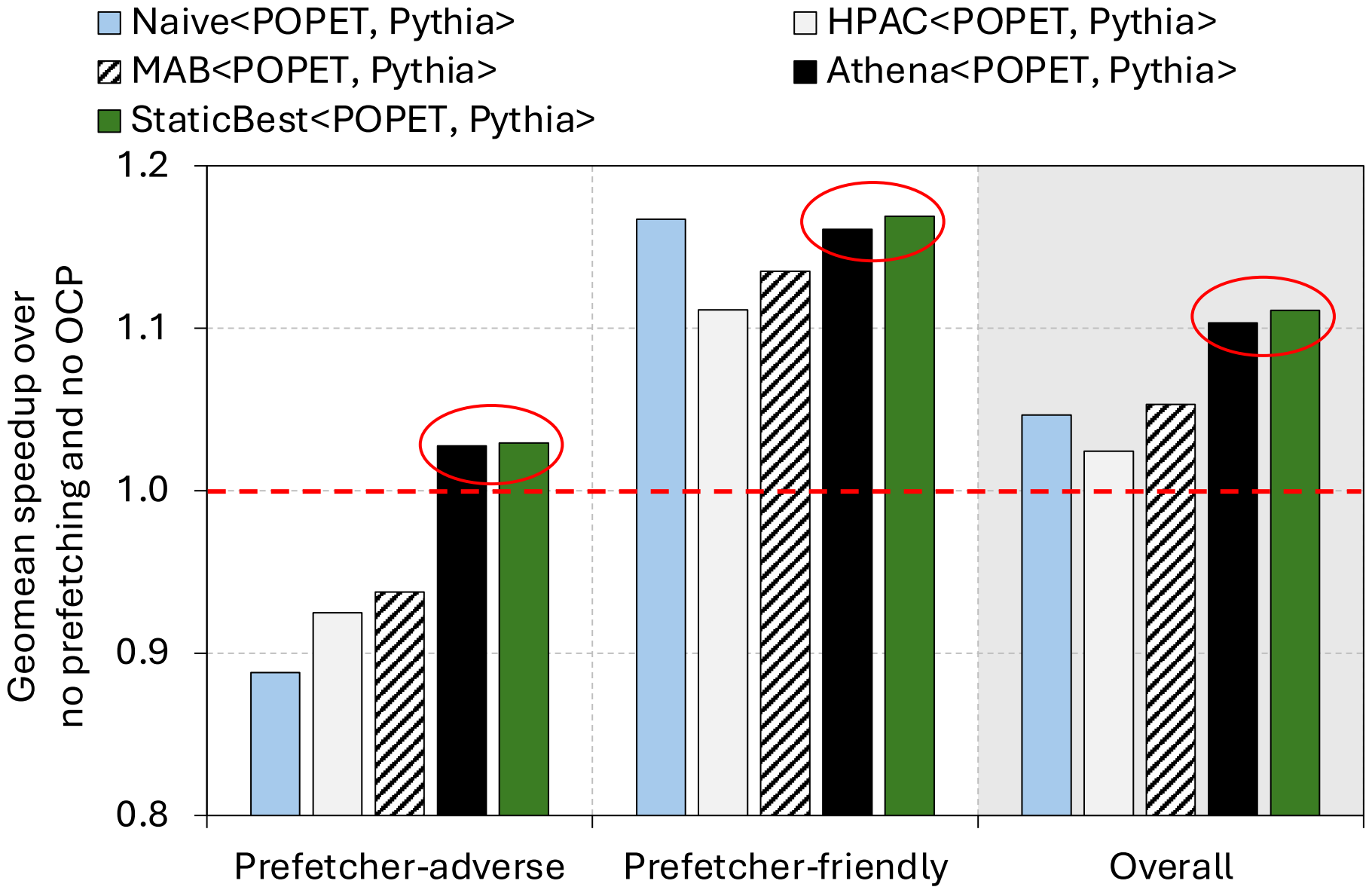
Speedup in Cache Design 1



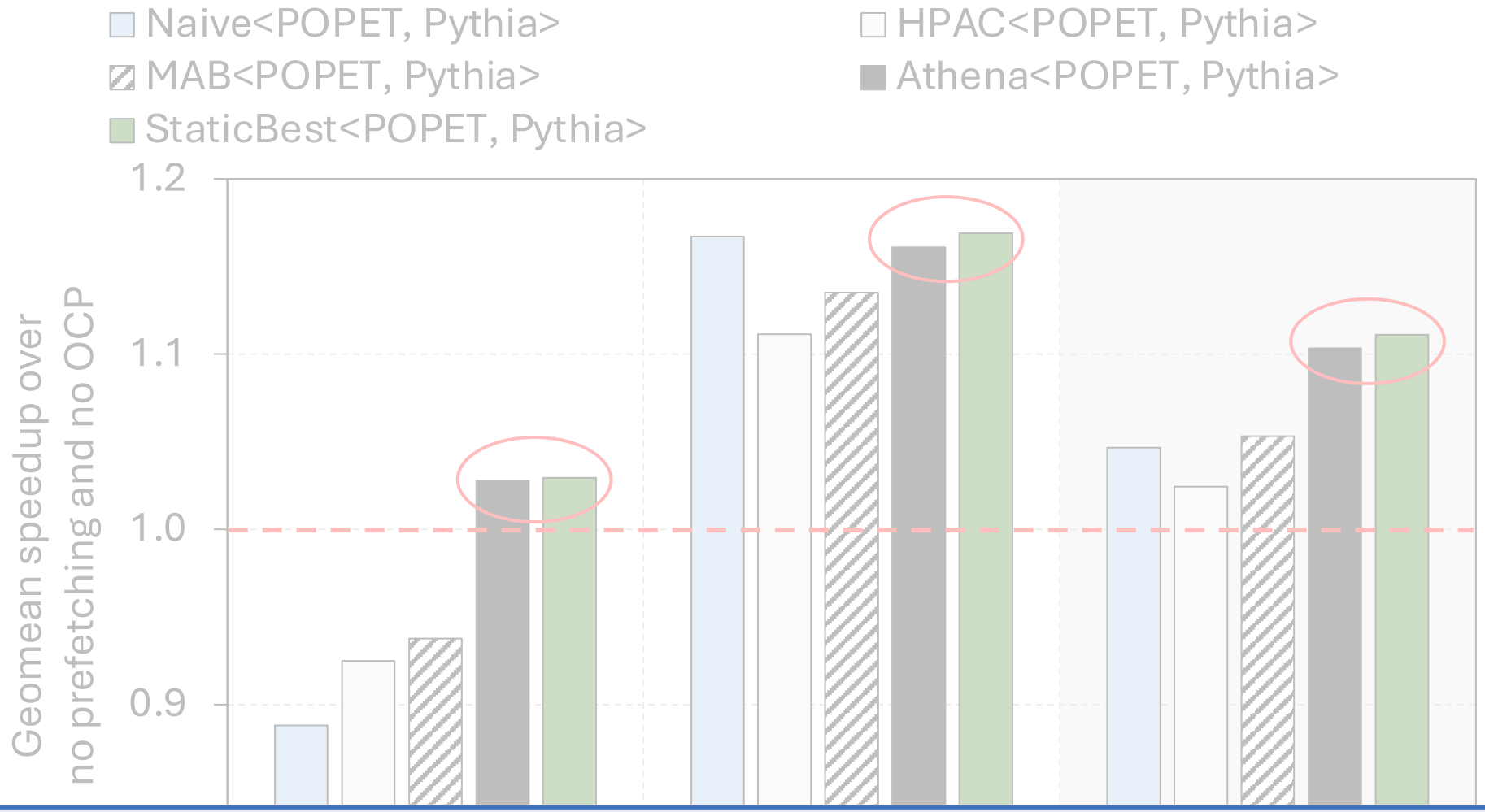
Athena outperforms the best-prior coordination policy MAB by 5% on average

Athena consistently outperforms prior prefetcher control policies in all workload categories

Speedup in Cache Design 1

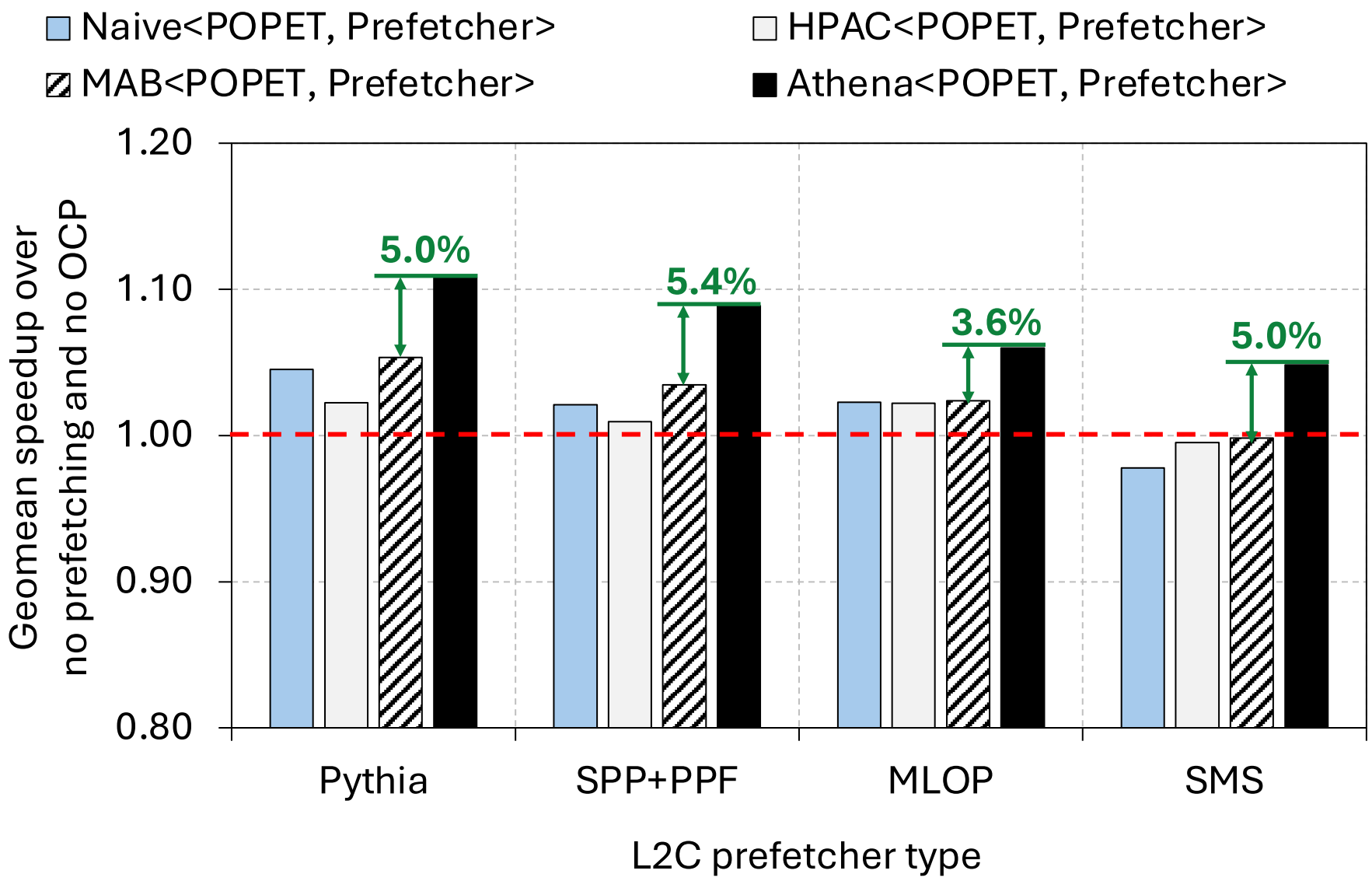


Speedup in Cache Design 1

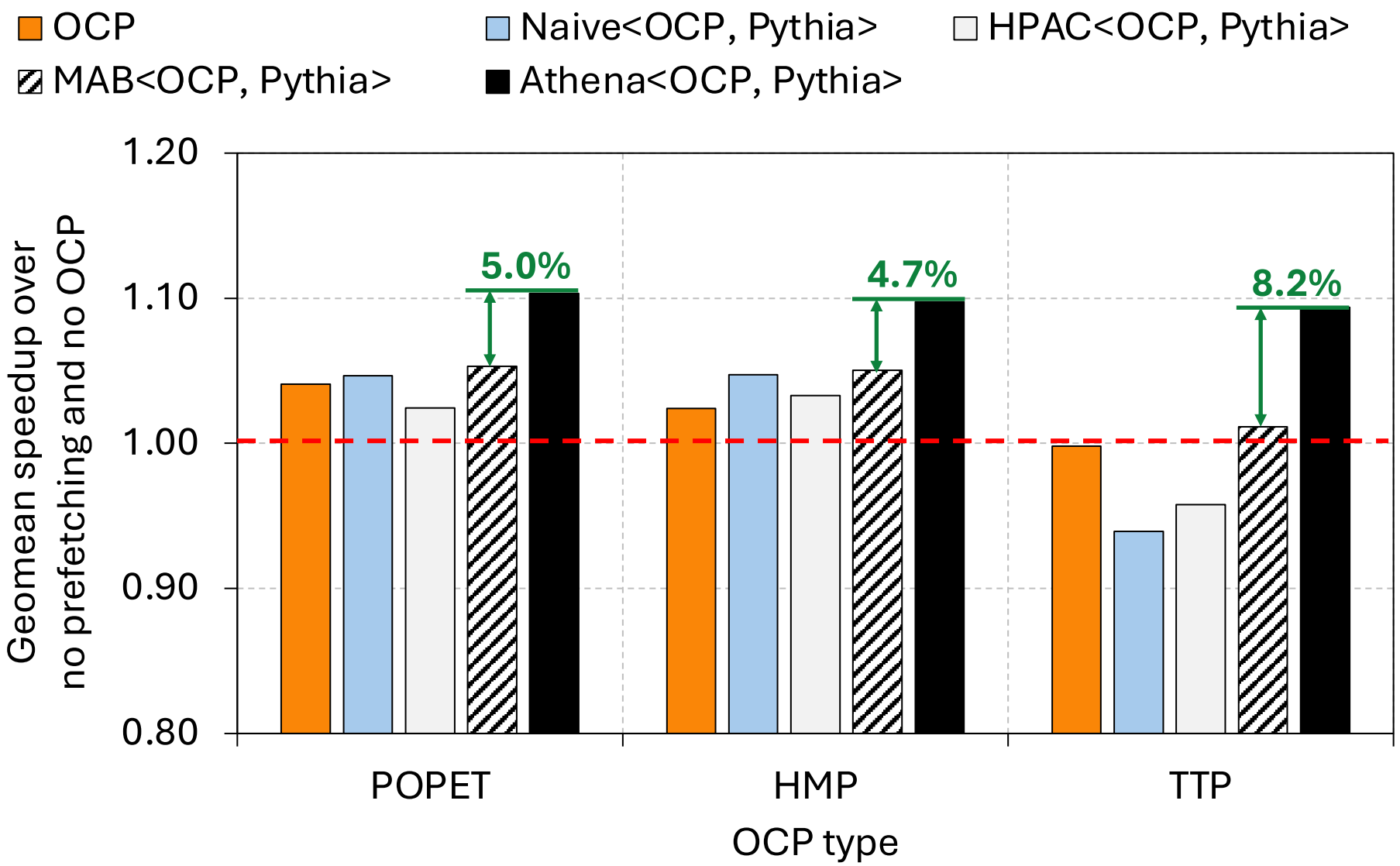


Athena provides similar performance gains as the StaticBest combination

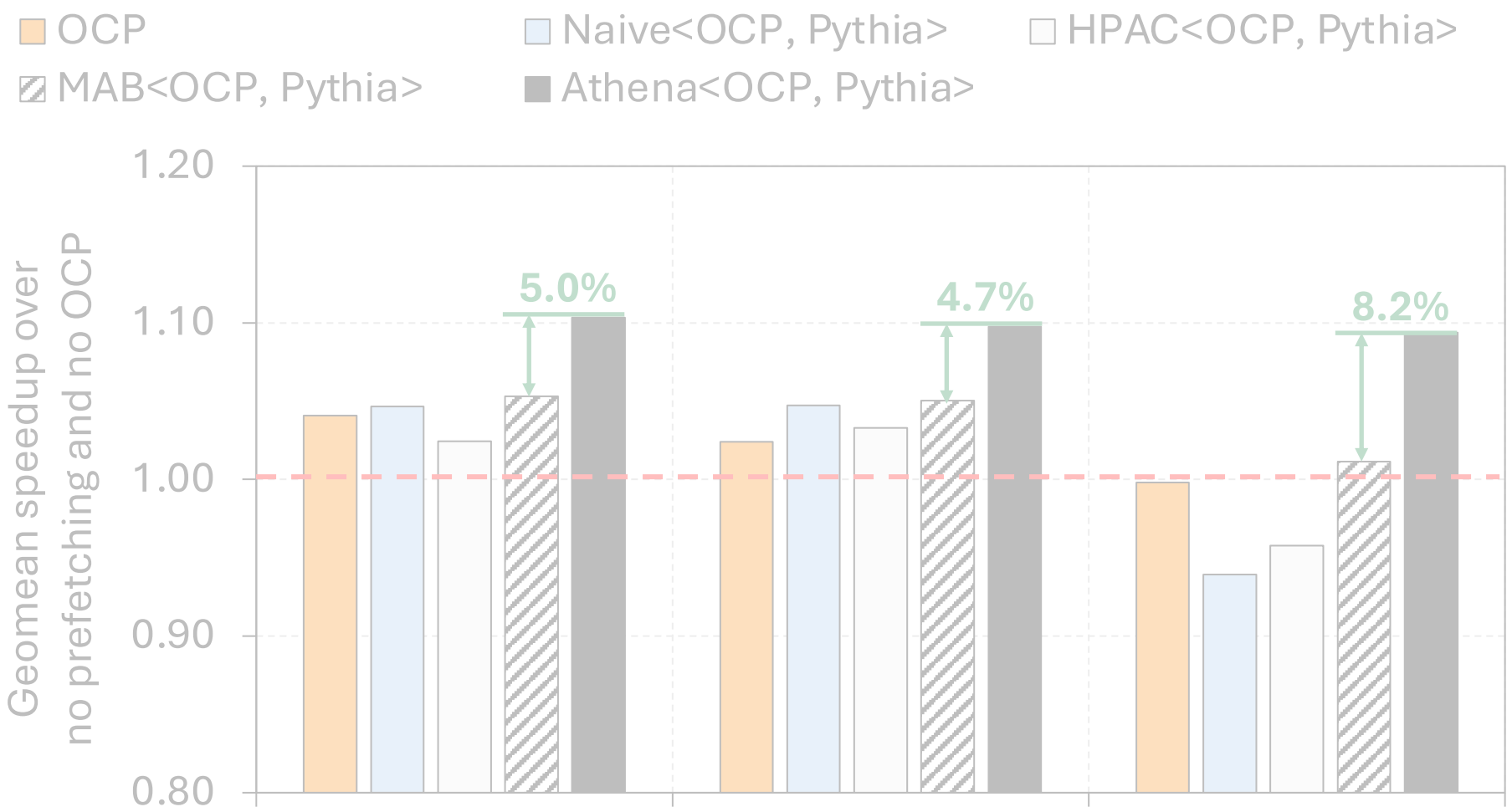
Performance Sensitivity to Varying Prefetchers



Performance Sensitivity to Varying OCPs



Performance Sensitivity to Varying OCPs



Athena provides consistent performance benefits across diverse prefetcher and OCP types

More Evaluations in the Paper

- Performance improvement in **three other cache designs**
- **Performance sensitivity study** across diverse system configurations
  - Varying L1D prefetcher
  - Varying OCP request issue latency
  - Varying main memory bandwidth
- **Understanding Athena using a case study**
- **Employing Athena for prefetcher-only management**
- **Additional results** in the extended version on arXiv:
  - Effect of Athena on main memory request
  - Effect of Athena on LLC load miss latency
  - Performance improvement of Athena on **359** additional unseen traces

More Evaluations in the Paper

Athena: Synergizing Data Prefetching and Off-Chip Prediction via Online Reinforcement Learning

*Rahul Bera[1] *Zhenrong Lang[1] Caroline Hengartner[1] Konstantinos Kanellopoulos[1]
Rakesh Kumar[2] Mohammad Sadrosadati[1] Onur Mutlu[1]

[1]ETH Zürich [2]NTNU

https://www.arxiv.org/pdf/2601.17615

Athena is Open Sourced


Open Research Objects


Research Objects Reviewed


Results Reproduced

100 Workloads

- **SPEC CPU 2006 & 2017**
- **PARSEC 2.1**
- **Ligra graph processing workloads**
- **Real-world INT, FP**

Six Prefetchers

- **IPCP** [Pakalapati+, ISCA'20]
- **Berti** [Navarro-Torres+, MICRO'22]
- **Pythia** [Bera+, MICRO'21]
- **SPP+PPF** [Kim+, MICRO'16]
- **SMS** [Somogyi+, ISCA'06]
- **MLOP** [Shakerinava+, DPC3'19]

Three OCPs

- **POPET** [Bera+, MICRO'22]
- **HMP** [Yoaz+, ISCA'99]
- **TTP** [Jalili+, HPCA'22; Bera+, MICRO'22]

Four Policies

- **TLP** [Jamet+, HPCA'24]
- **HPAC** [Ebrahimi+, MICRO'09]
- **MAB** [Gerogiannis+, MICRO'23]
- **Athena (Our Proposal)**

https://github.com/CMU-SAFARI/Athena

Outline

Background

Motivation

Athena

Evaluation

Conclusion

ATHENA

The **first** reinforcement learning-based mechanism to coordinate off-chip predictor and multiple prefetchers

Contributions

1. Introduces a **composite** reward framework that **isolates** the true **impact of Athena's own action** from **inherent variations in workload** behavior

2. Works as a prefetcher-OCP **coordinator** and a prefetcher **throttler**, at the same time, **without any additional hardware**

Evaluation

- Evaluated with **100** workloads, **6** prefetchers, **3** OCPs, **4** cache designs, and **a wide range** of main memory bandwidth configurations
- **Consistently** outperforms both best-prior heuristic- and learning-based policies in **every** prefetcher/OCP/bandwidth configuration

https://github.com/CMU-SAFARI/Athena

ATHENA

Synergizing Data Prefetching and Off-Chip Prediction via Online Reinforcement Learning

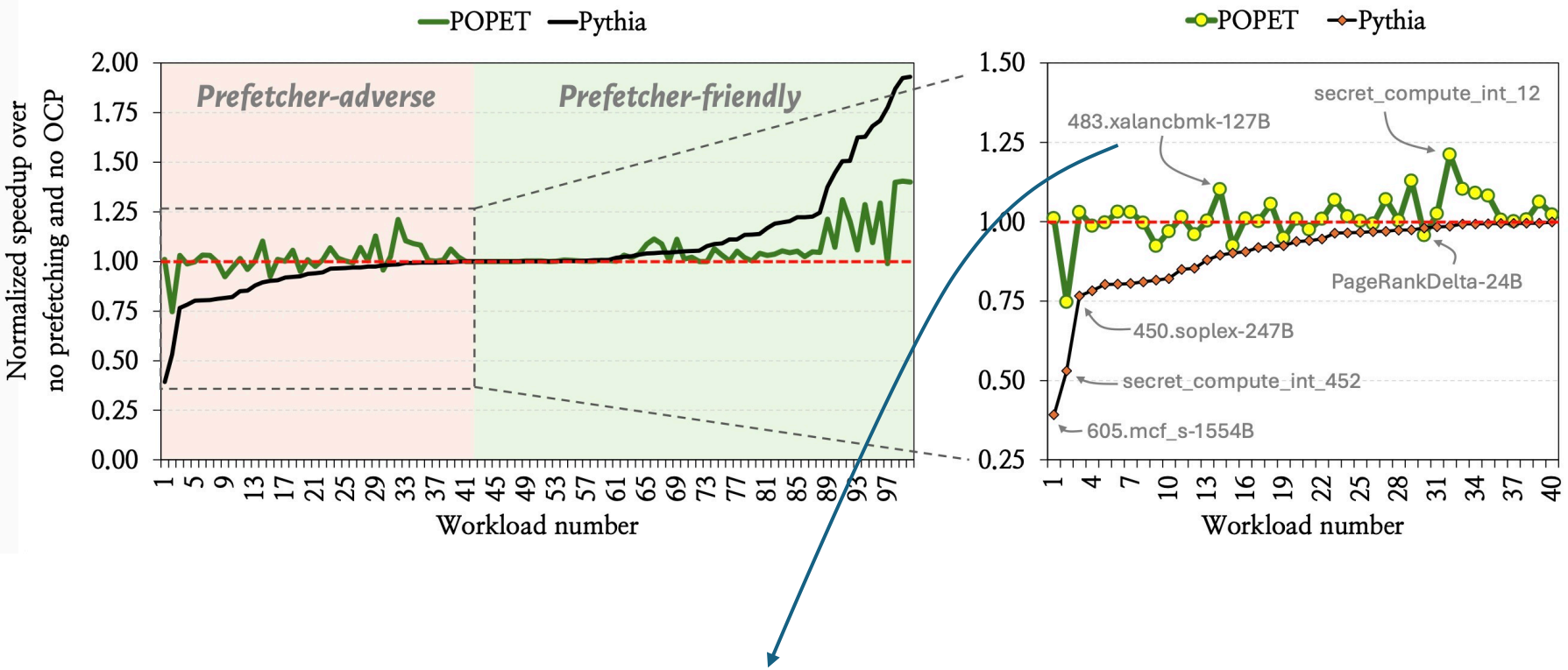
Rahul Bera, Zhenrong Lang, Caroline Hengartner, Konstantinos Kanellopoulos, Rakesh Kumar, Mohammad Sadrosadati, Onur Mutlu

ETH zürich SAFARI NTNU

BACKUP

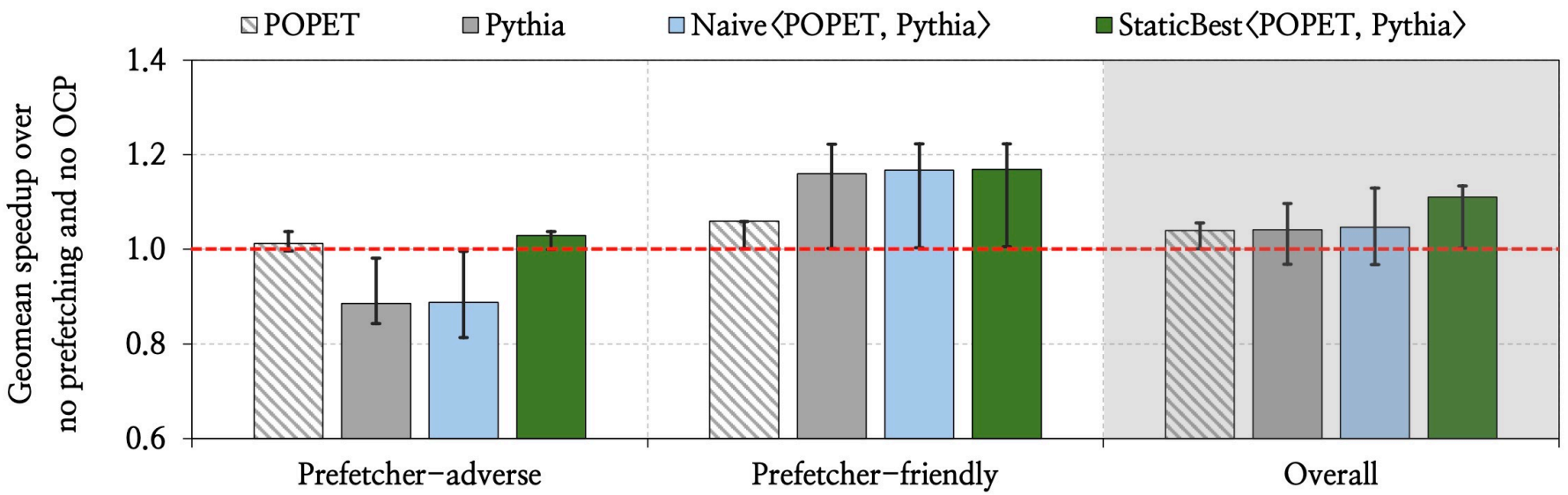
Index

POPET vs. Pythia Performance Line Graph

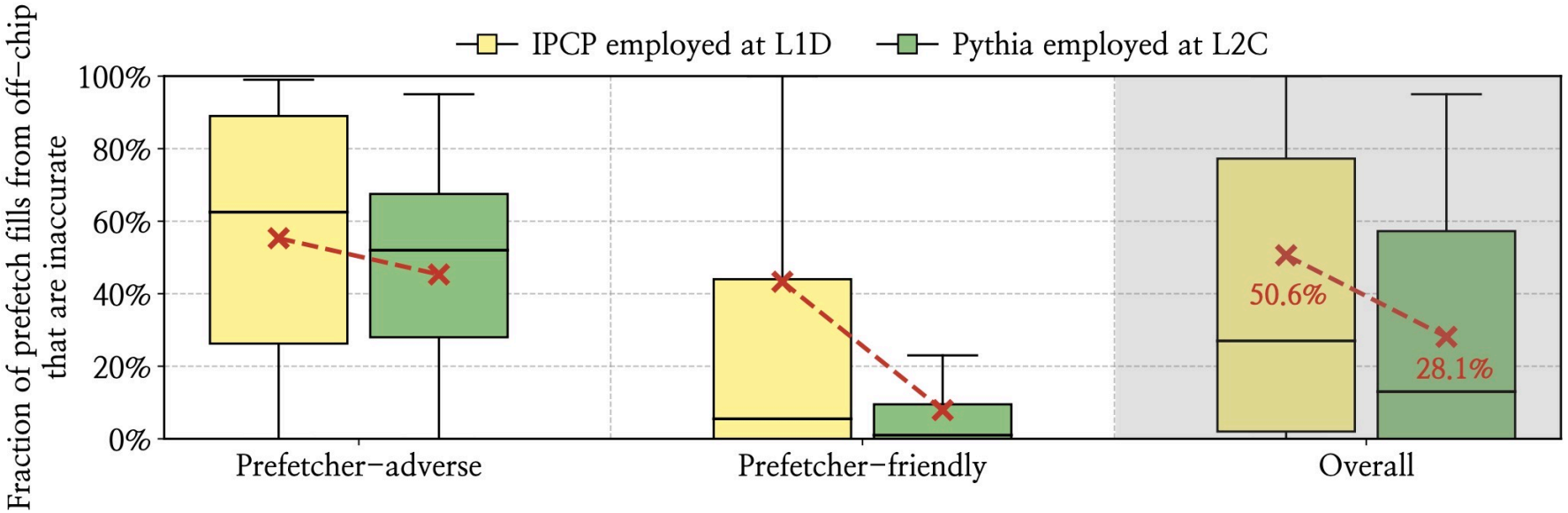


	Accuracy	Performance
Pythia	28.7%	-10.5%
POPET	84.1%	+10.3%

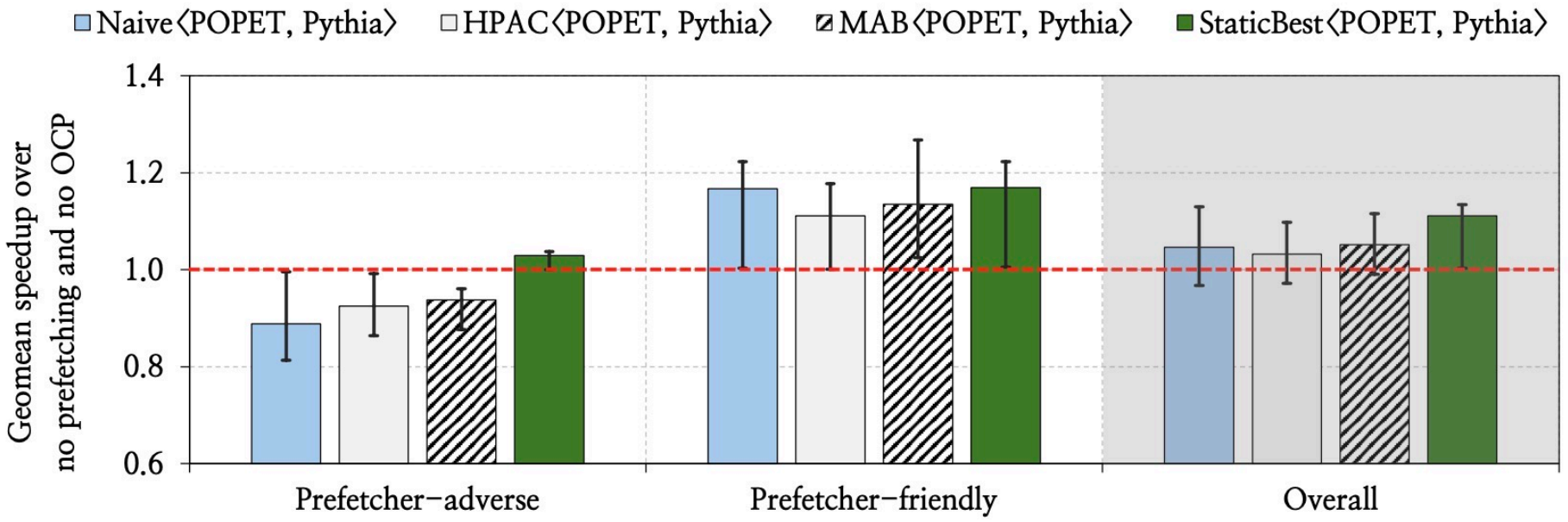
Athena Performance Headroom



Off-Chip Prefetch Fills are not Always Inaccurate



Prior Works Leave Performance Potential Behind



Candidate Features Considered

Feature	Measurement	Rationale
Prefetcher accuracy	$\frac{\text{\# demand hits}}{\text{\# prefetches issued}}$	Effectiveness of prefetching
OCP accuracy	$\frac{\text{\# off-chip demand hits}}{\text{\# off-chip predictions}}$	Effectiveness of OCP
Bandwidth usage	$\frac{\text{used main memory bandwidth}}{\text{peak main memory bandwidth}}$	Memory bus pressure
Cache pollution	$\frac{\text{\# prefetch-evicted demand misses}}{\text{\# total demand misses}}$	Interference by prefetcher
Prefetch bandwidth	$\frac{\text{\# prefetch requests to DRAM}}{\text{\# total DRAM requests}}$	Prefetcher's share of memory traffic
OCP bandwidth	$\frac{\text{\# OCP requests to DRAM}}{\text{\# total DRAM requests}}$	OCP's share of memory traffic
Demand bandwidth	$\frac{\text{\# demand requests to DRAM}}{\text{\# total DRAM requests}}$	Demand's share of memory traffic

Q-Value-Driven Prefetcher Aggressiveness Control

Algorithm 1 Q-value-driven prefetcher aggressiveness control

procedure SELECTPREFETCHDEGREE
  $a^* \leftarrow \arg\max_{a \in \mathcal{A}} Q(a)$
  $avg \leftarrow$ average Q-value of all remaining actions
  $\Delta Q \leftarrow Q(a^*) - avg$ ▷ *compute Q-value confidence*
  $r \leftarrow \min(1, \Delta Q / \tau)$ ▷ *normalize confidence w.r.t. hyperparam* τ
  $d \leftarrow \lfloor r \cdot d_{\max} \rfloor$ ▷ *adjust prefetch degree*
  return d

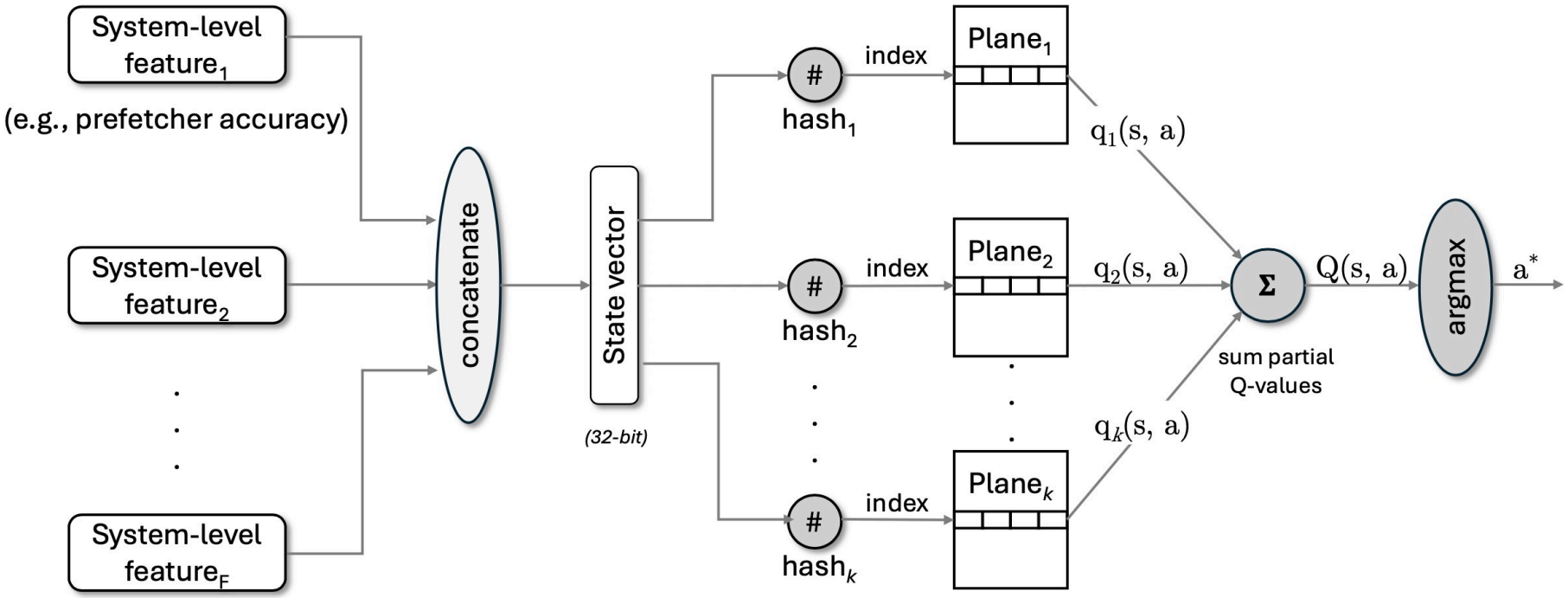
Reward Framework

$$R_t = R_t^{corr} - R_t^{uncorr}$$

$$R_t^{corr} = \sum_i \lambda_i \cdot \Delta M_{i,t}^{corr} \quad R_t^{uncorr} = \sum_j \lambda_j \cdot \Delta M_{j,t}^{uncorr}$$

	Constituent metric	Weight
R_t^{corr}	# Cycles	λ_{cycle}
	# LLC misses	$\lambda_{\text{LLC}_\text{m}}$
	LLC miss latency	$\lambda_{\text{LLC}_\text{t}}$
R_t^{uncorr}	# Load instructions	λ_{load}
	# Mispredicted branches	λ_{MBr}

Q-Value Retrieval from QVStore



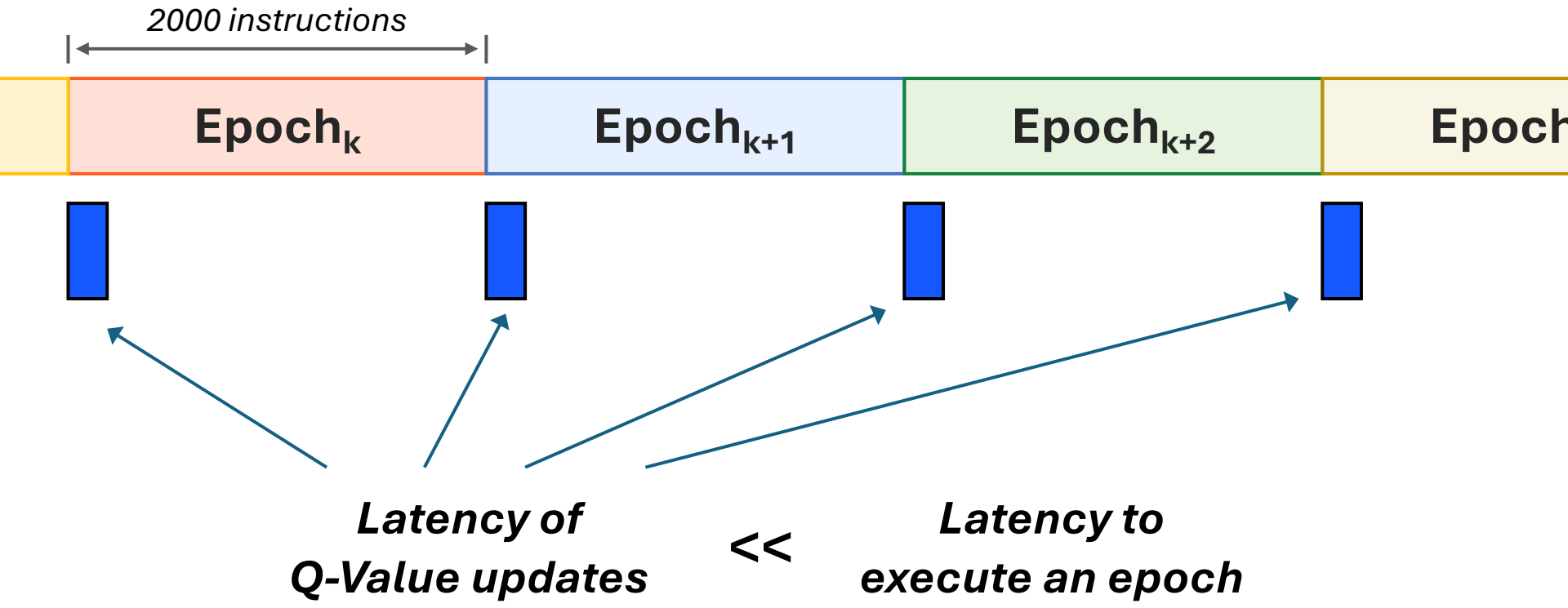
Final Configuration of Athena

Category	Final Values
Selected Features	(1) prefetcher accuracy, (2) OCP accuracy, (3) bandwidth usage, (4) prefetch-induced cache pollution
Reward Weights	$\lambda_{\text{cycle}} = 1.6$, $\lambda_{\text{LLC}_\text{m}} = 0.0$, $\lambda_{\text{LLC}_\text{t}} = 0.0$, $\lambda_{\text{load}} = 0.6$, $\lambda_{\text{MBr}} = 1.0$
Hyperparameters	$\alpha = 0.6$, $\gamma = 0.6$, $\epsilon = 0.0$, $\tau = 0.12$, Epoch length (N) = $2K$ instructions

Storage Overhead of Athena

Structure	Description	Size
QVStore	# planes = 8, # rows = 64 # columns = 4, entry size = 8 bits	2 KB
Accuracy Tracker	4096-bit Bloom filter, 2 hashes	0.5 KB
Pollution Tracker	4096-bit Bloom filter, 2 hashes	0.5 KB
Total		**3 KB**

Why is Athena's Latency Overhead Minimal?



Evaluated System Parameters

Core	6-wide fetch/issue/commit, 512-entry ROB, 128-entry LQ, 72-entry SQ, perceptron branch predictor [89], 17-cycle misprediction penalty
L1I/D	Private, 48KB, 64B line, 12-way, 16 MSHRs, LRU, 4/5-cycle round-trip latency
L2C	Private, 1.25MB, 64B line, 20-way, 48 MSHRs, LRU, 15-cycle round-trip latency [8]
LLC	Shared, 3MB/core, 64B line, 12-way, 64 MSHRs/slice, SHiP [183], 55-cycle round-trip latency [7, 8]
Main Memory	1 rank per channel, 8 banks per rank, 64-bit data bus, 2 KB row buffer, $t_{RCD} = t_{RP} = t_{CAS} = 12.5$ ns; DDR4 DRAM with 3.2 GB/s per core

Evaluated Workloads

Suite	# Traces	Example Workloads
SPEC CPU 2006	29	astar, gobmk, GemsFDTD, leslie3d, libquantum, milc, omnetpp, sphinx3, …
SPEC CPU 2017	20	bwaves, cactuBSSN, cam4, fotonik3d, gcc, lbm, mcf, xalancbmk, …
PARSEC	13	canneal, facesim, fluidanimate, raytrace, streamcluster, …
Ligra	13	BC, BFS, BFSCC, CF, PageRank, PageRankDelta, Radii, Triangle, …
CVP	25	integer, floating point, …

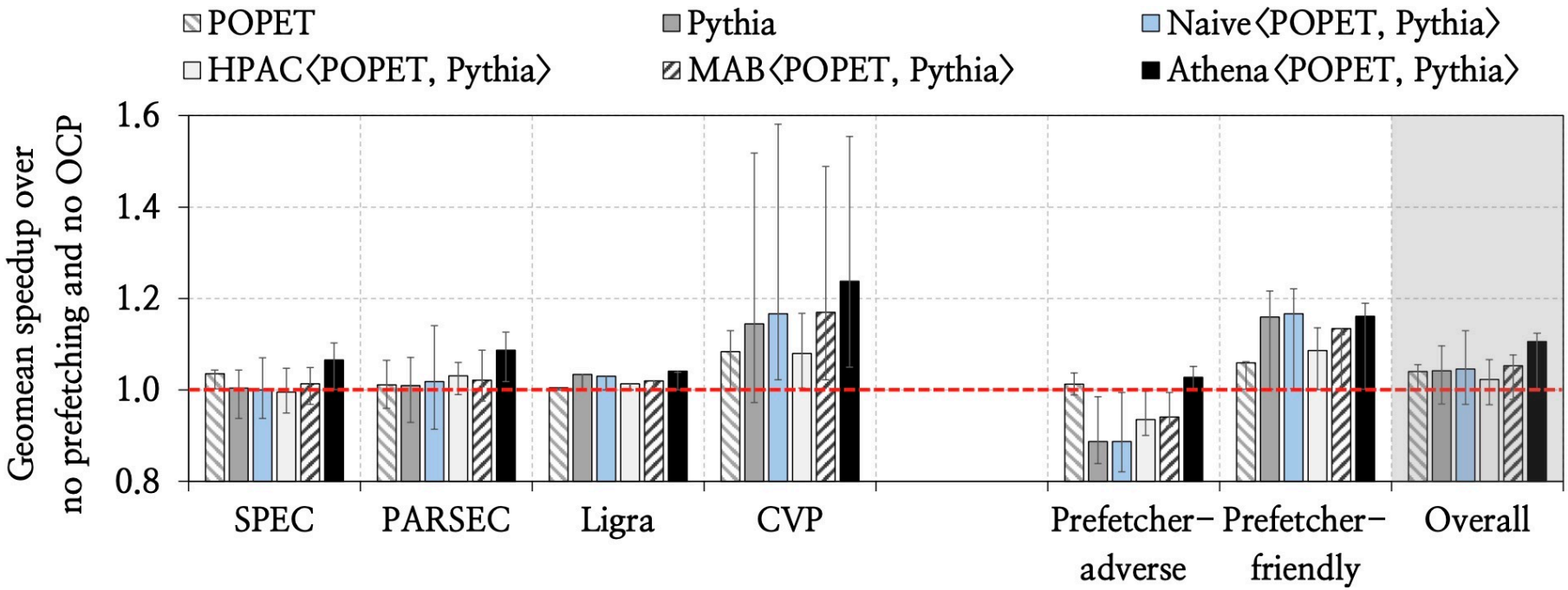
Evaluated Cache Designs

Design#	Description	Comparison Points
CD1	OCP + 1 L2C prefetcher	HPAC, MAB
CD2	OCP + 1 L1D prefetcher	HPAC, MAB, TLP
CD3	OCP + 2 L2C prefetchers	HPAC, MAB
CD4	OCP + 1 L1D + 1 L2C prefetcher	HPAC, MAB, TLP

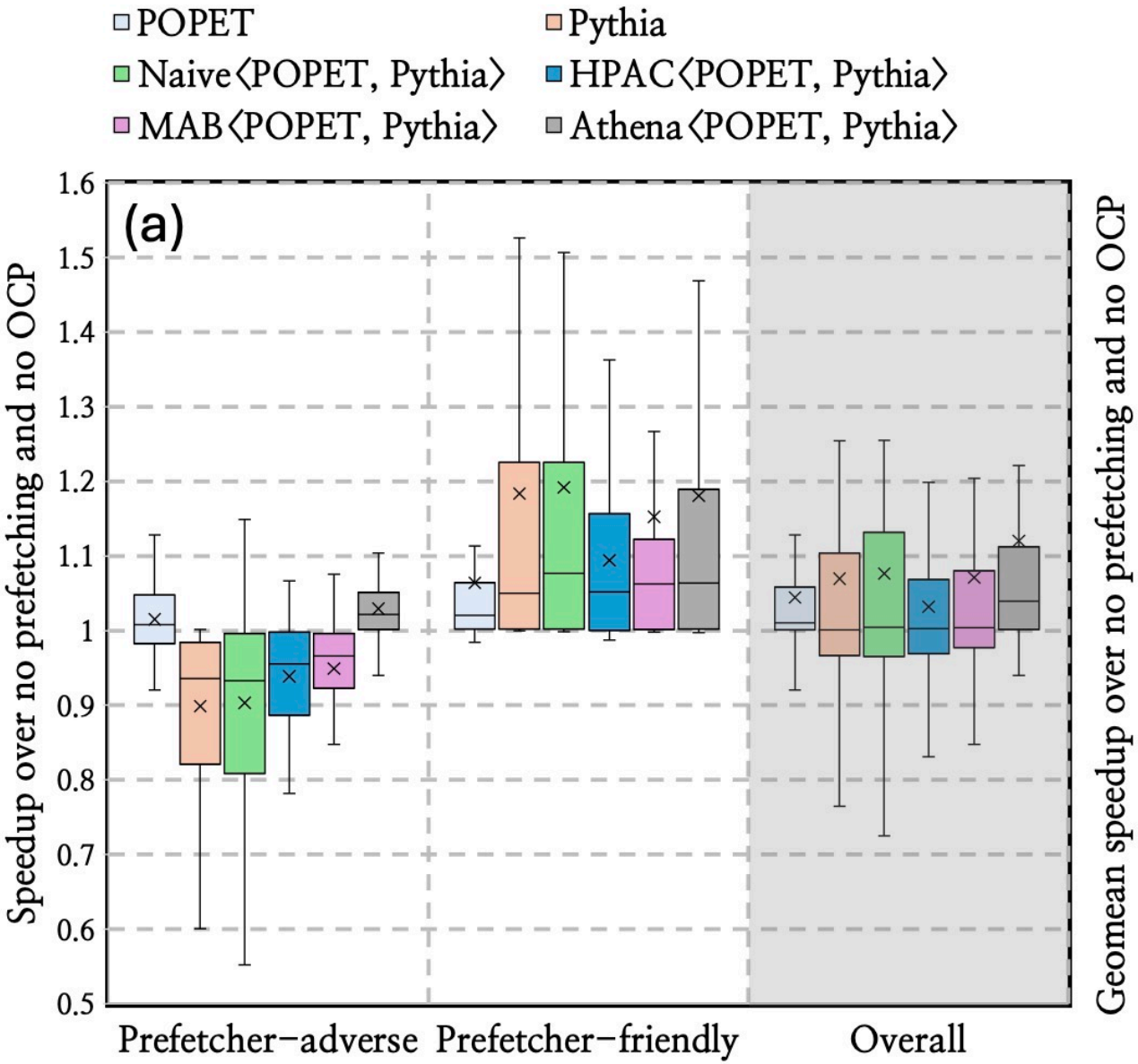
Storage Overhead Comparison

Prefetchers	L1D	**IPCP** [139], as an L1D-only prefetcher	**0.7 KB**
		Berti, as configured in [137]	**2.55 KB**
	L2C	**Pythia**, as configured in [21]	**25.5 KB**
		SPP+PPF, as configured in [25, 99]	**39.3 KB**
		MLOP, as configured in [156]	**8 KB**
		SMS, as configured in [165]	**20 KB**
OCPs		**POPET** [20], with 5 features	**4 KB**
		HMP [196], with 3 component predictors	**11 KB**
		TTP [83], with metadata budget ~L2 cache size	**1536 KB**
Policies		**TLP**, as in [84]	**6.98 KB**
		HPAC [54], adapted for OCP	**0.5 KB**
		MAB [68], adapted for OCP	**0.1 KB**
		Athena (this work)	**3 KB**

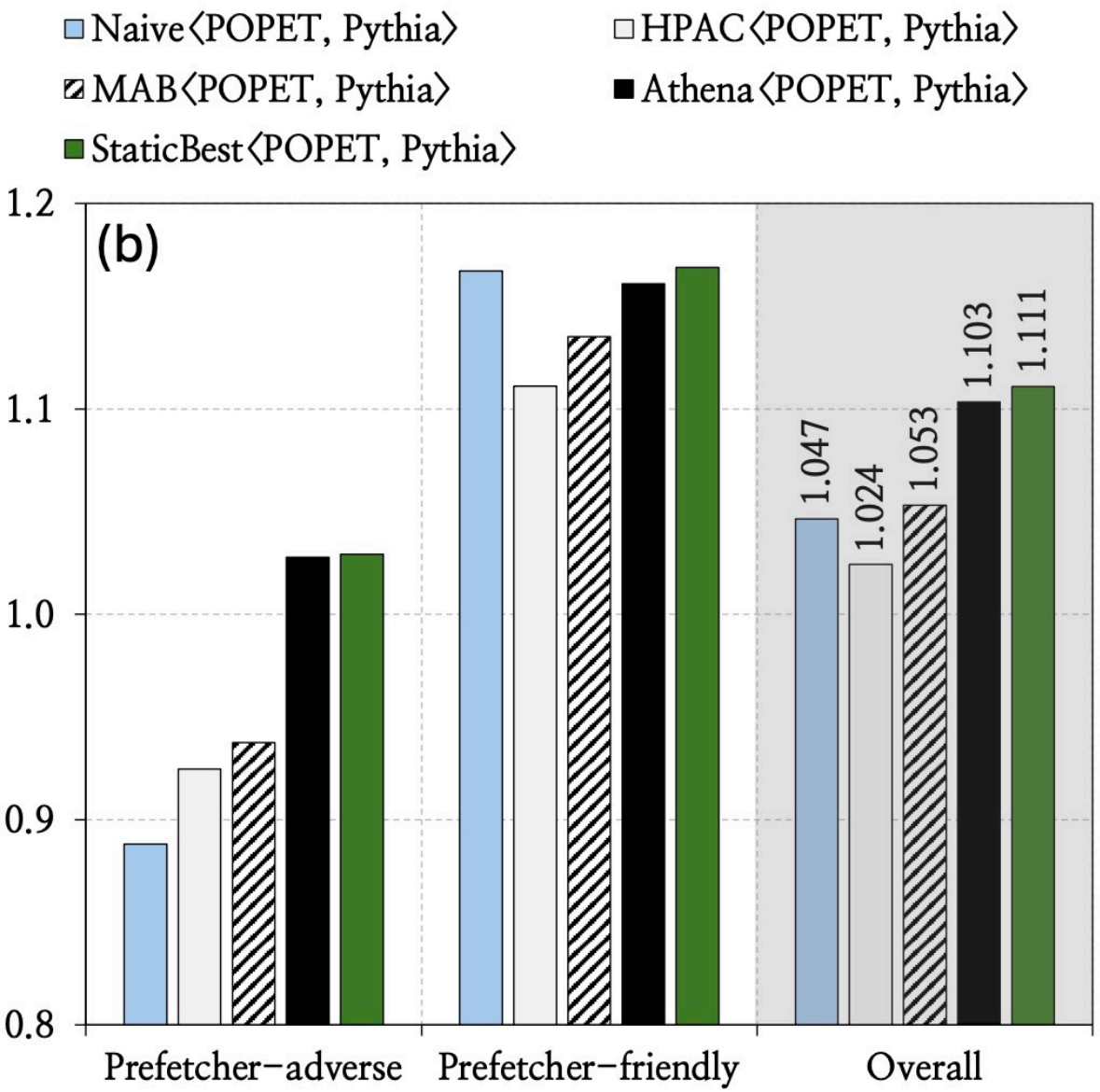
Speedup in CD1



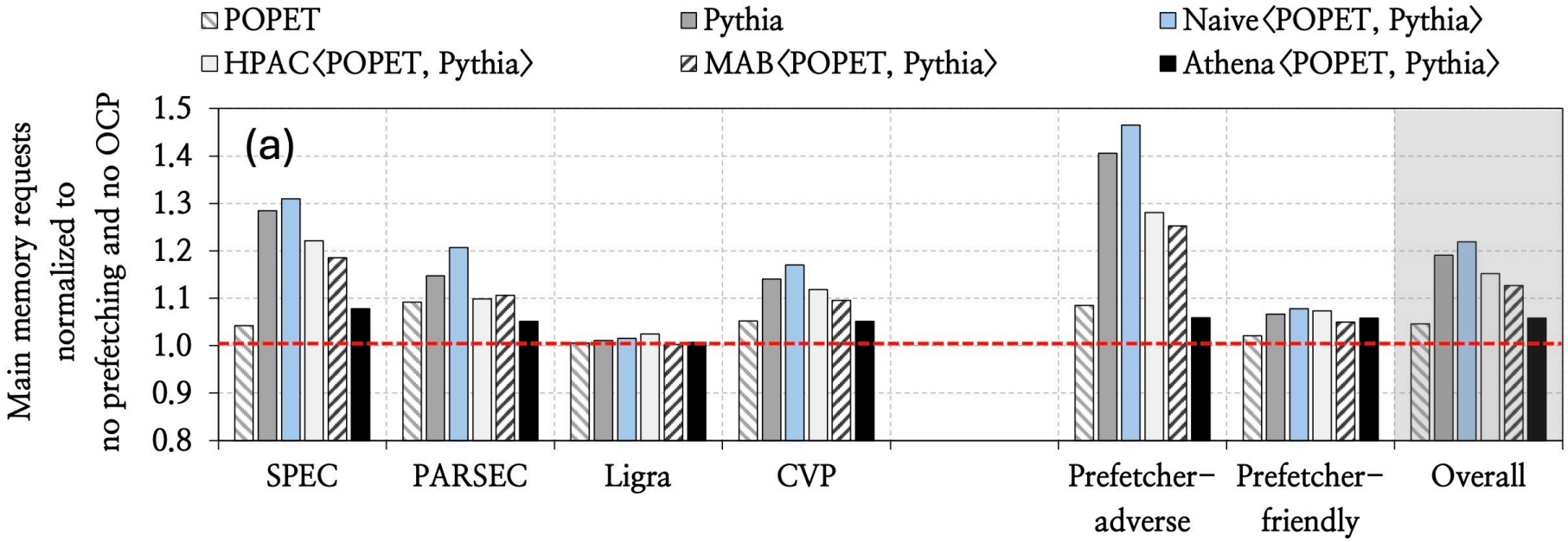
CD1 Speedup – Deepdive



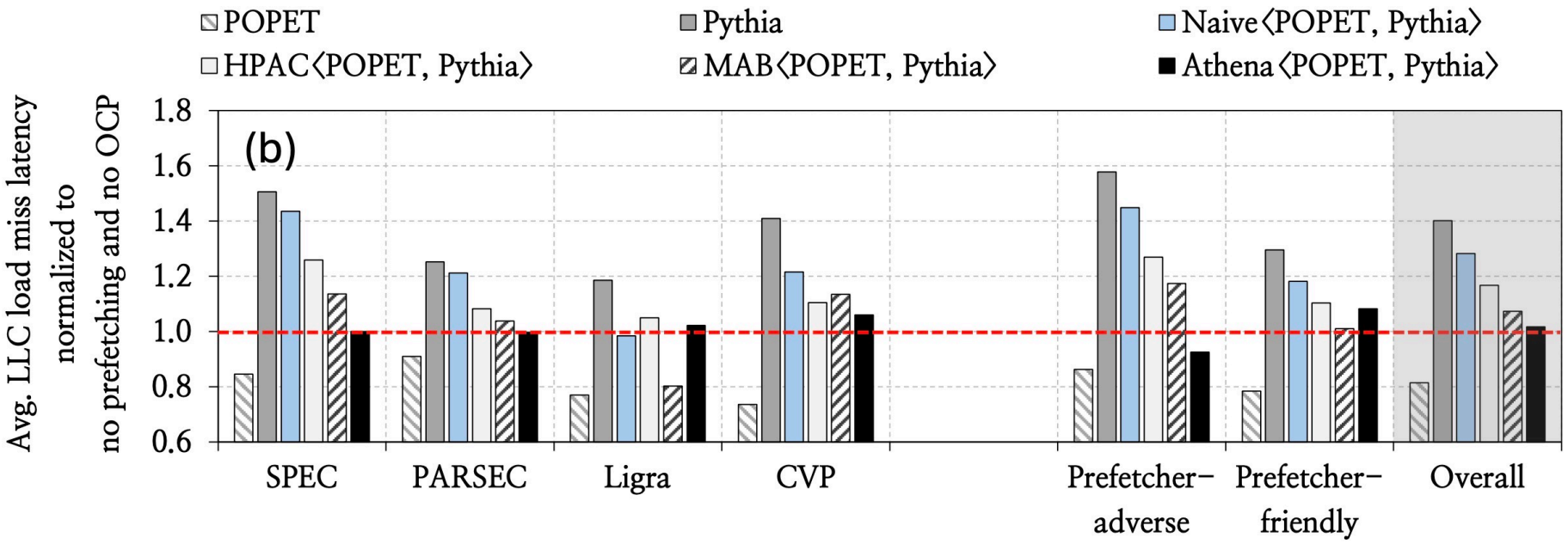
CD1 Speedup – Compared to StaticBest



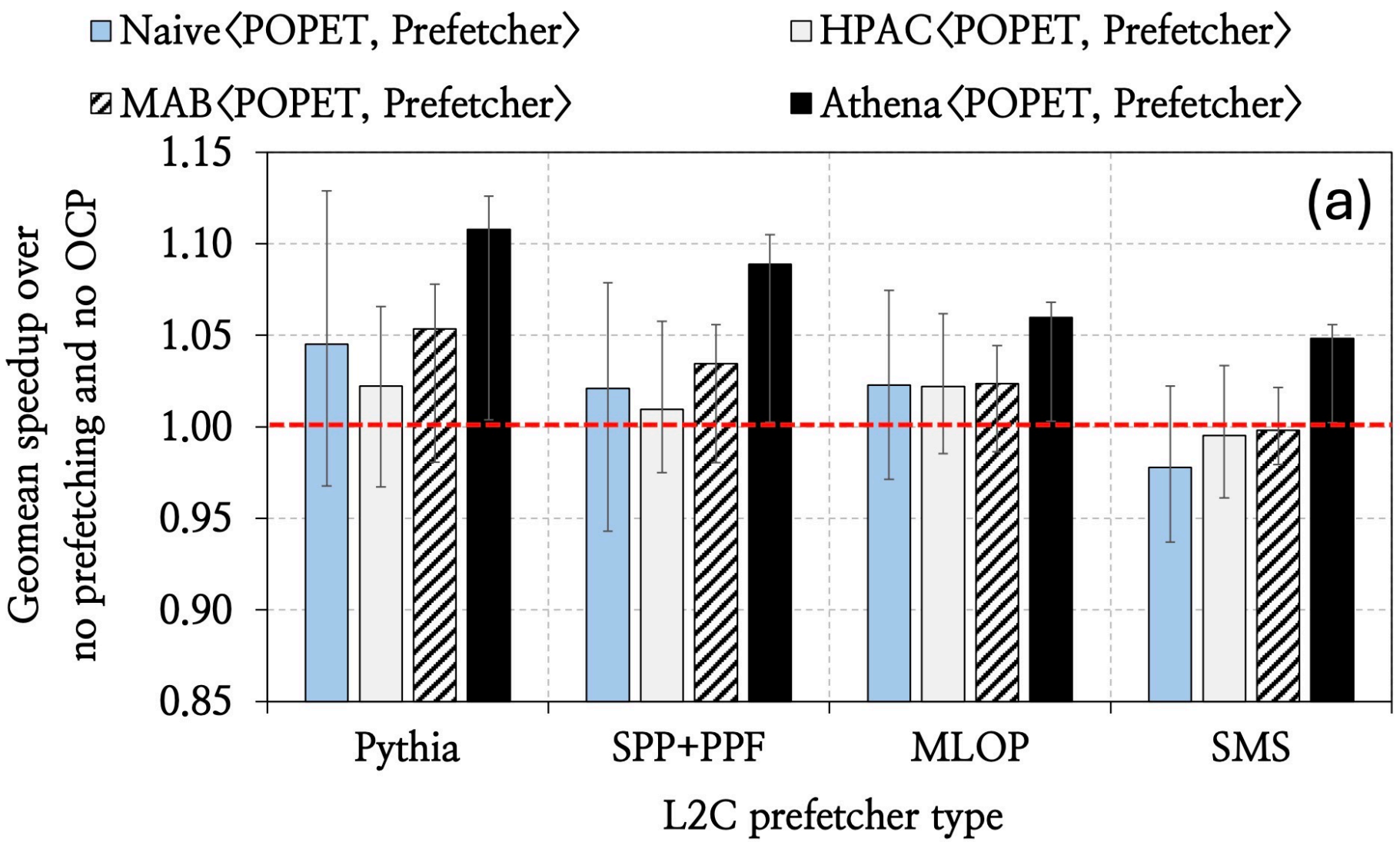
CD1 – Impact on Main Memory Requests



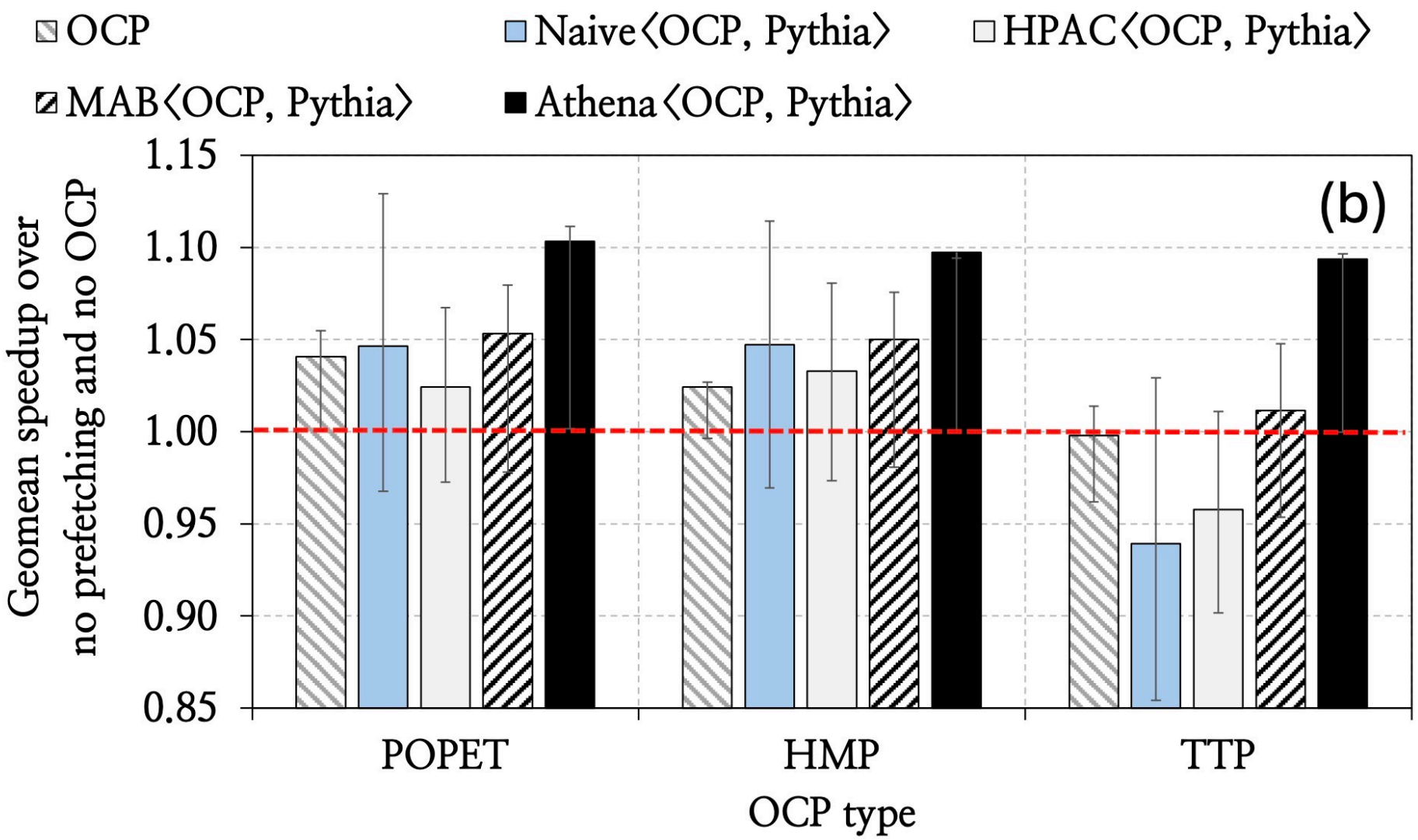
CD1 – Impact on LLC Load Miss Latency



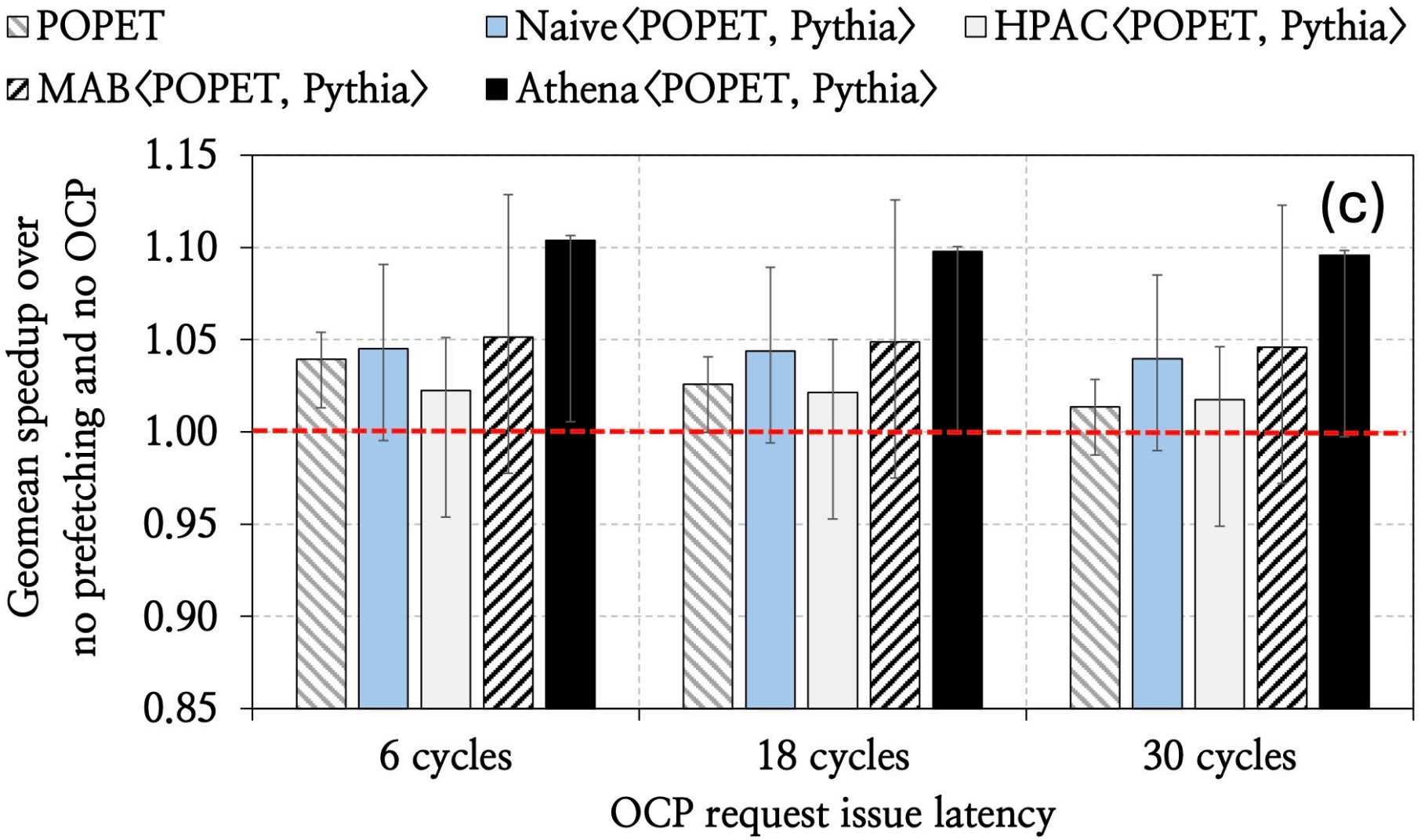
CD1 – Sensitivity to L2C Prefetcher Type



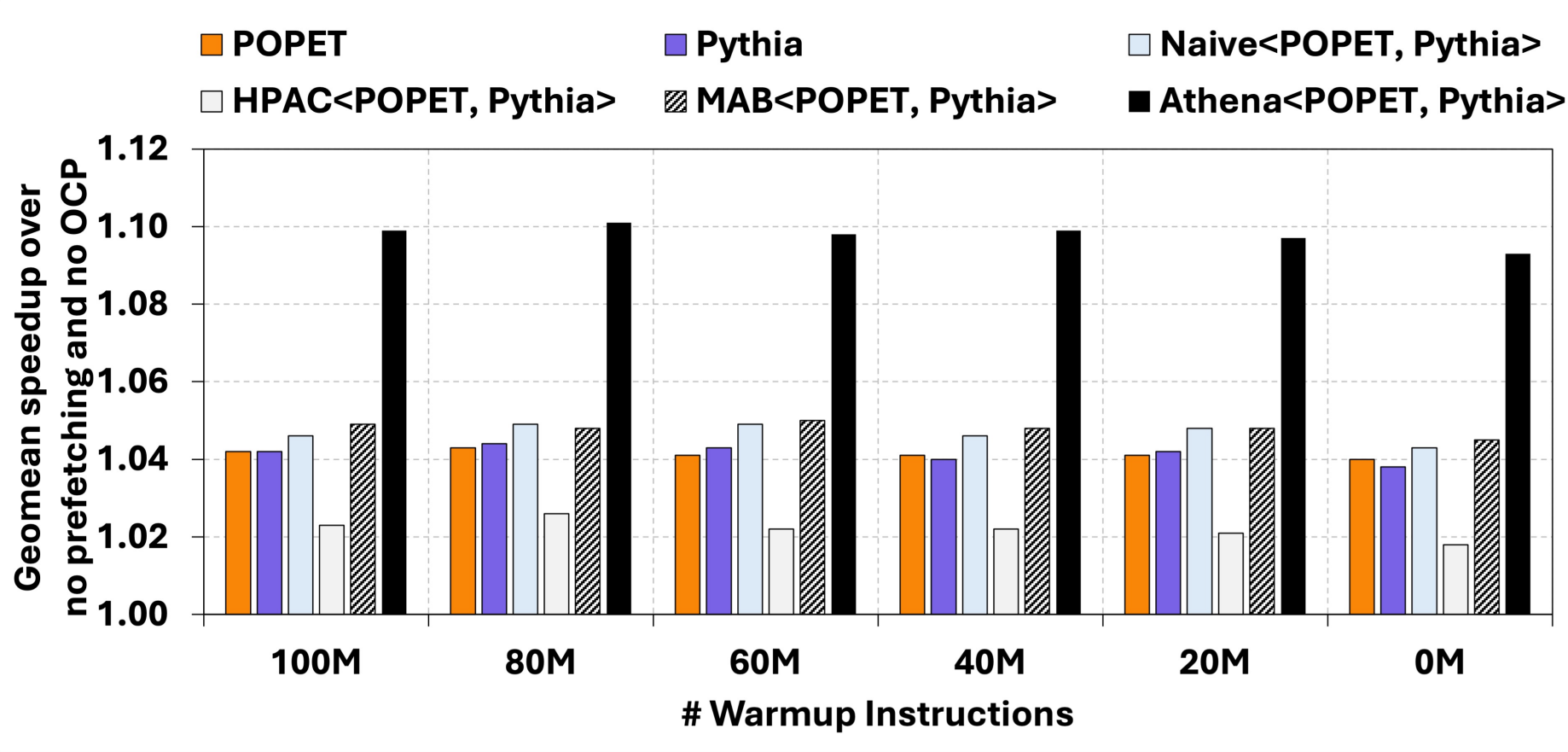
CD1 – Sensitivity to OCP Type



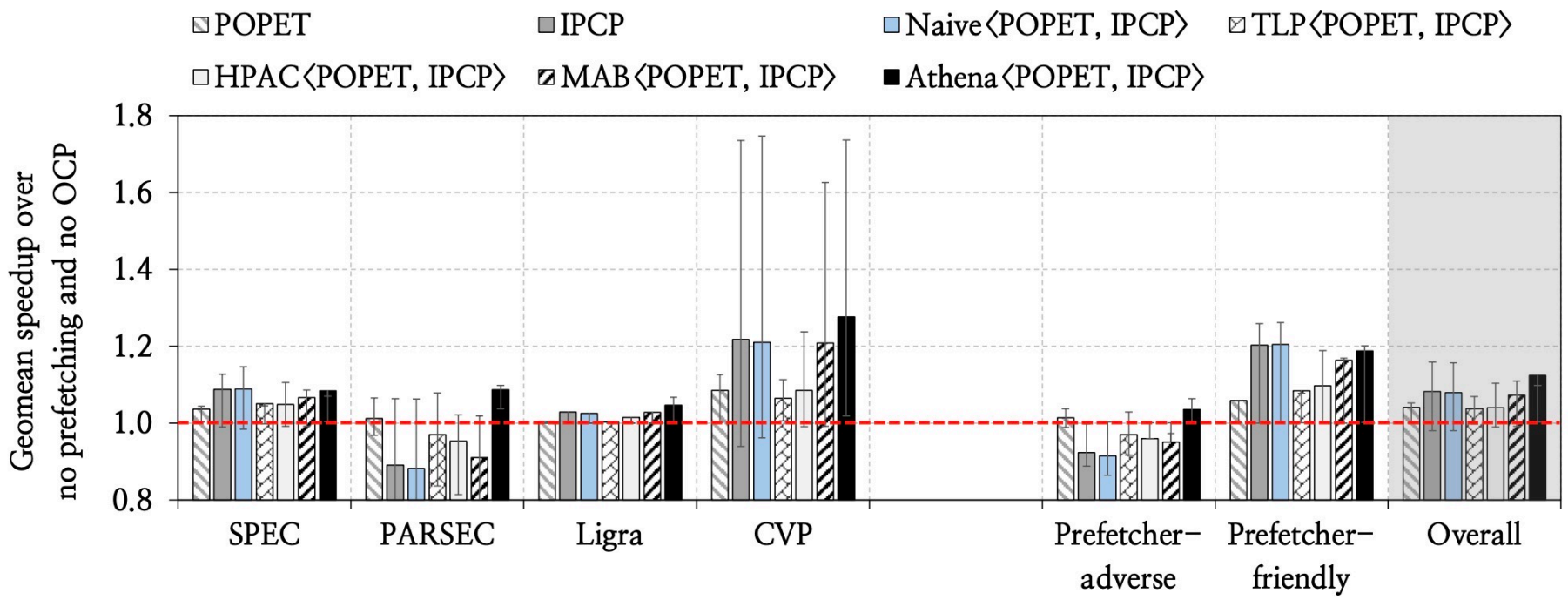
CD1 – Sensitivity to OCP Request Issue Latency



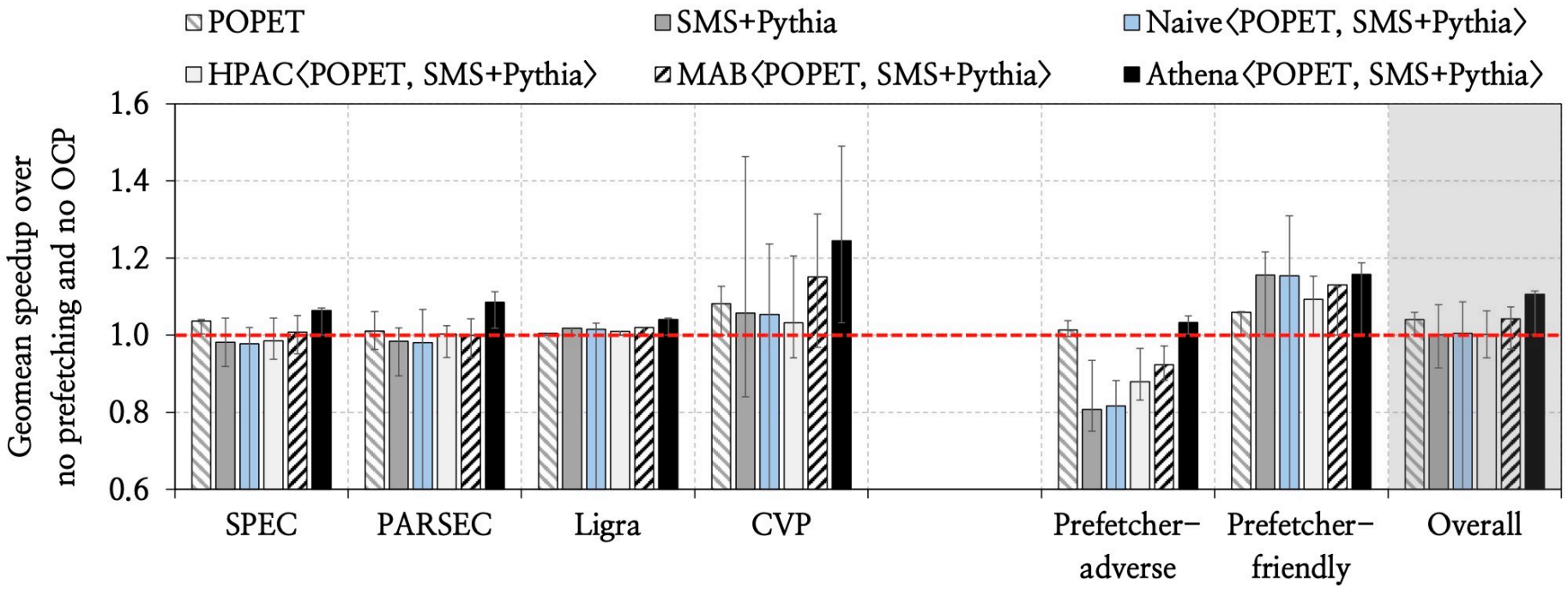
CD1 – Sensitivity to Warmup Length



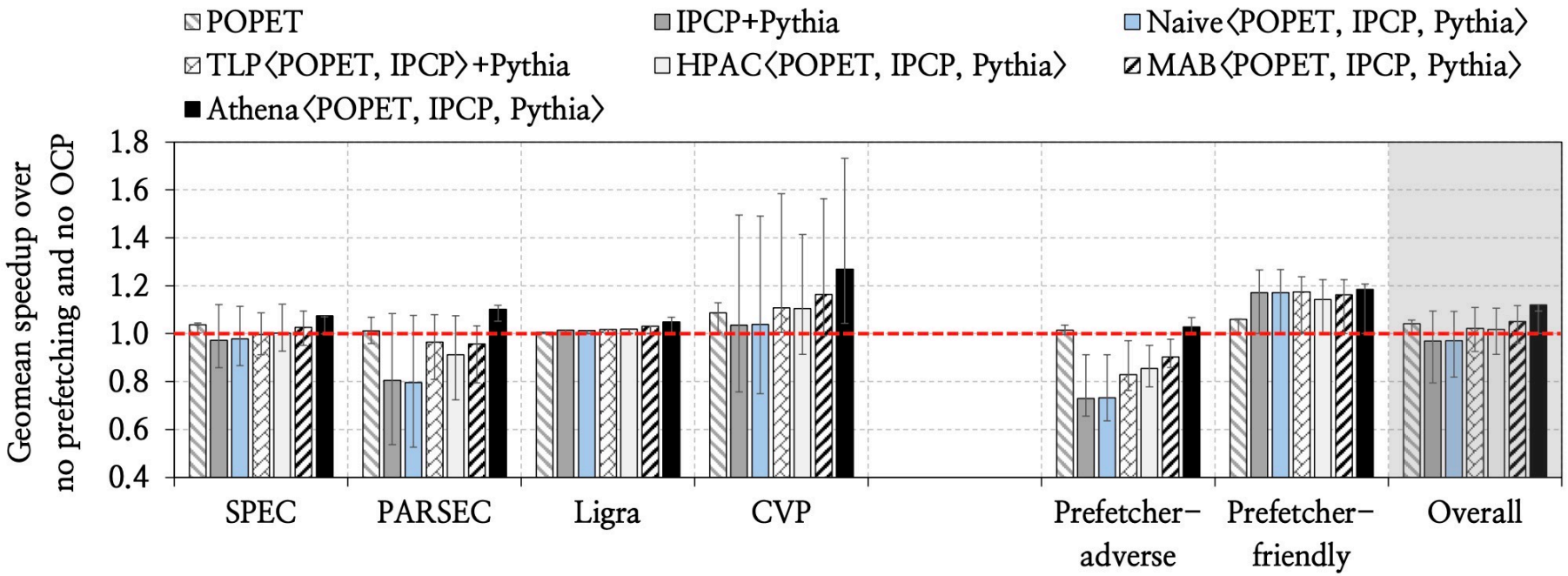
Speedup in CD2



Speedup in CD3



Speedup in CD4



CD4 – Sensitivity to L1D Prefetcher Type

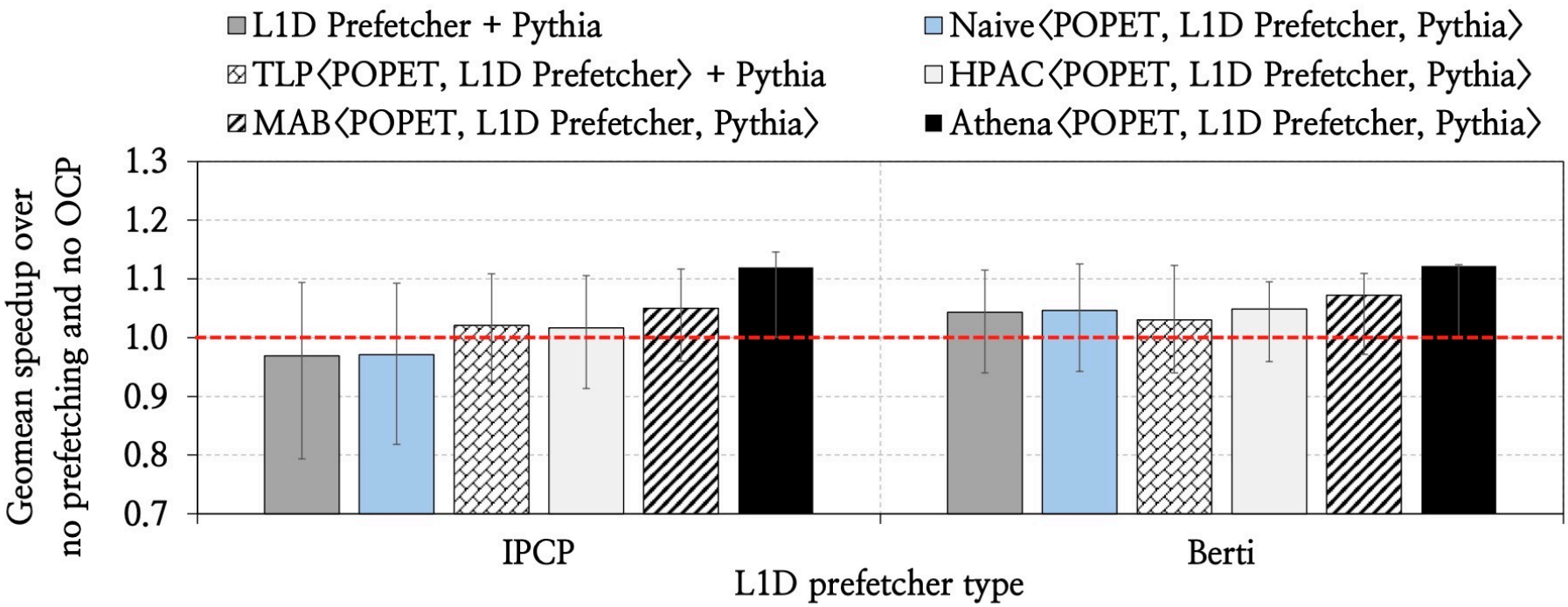
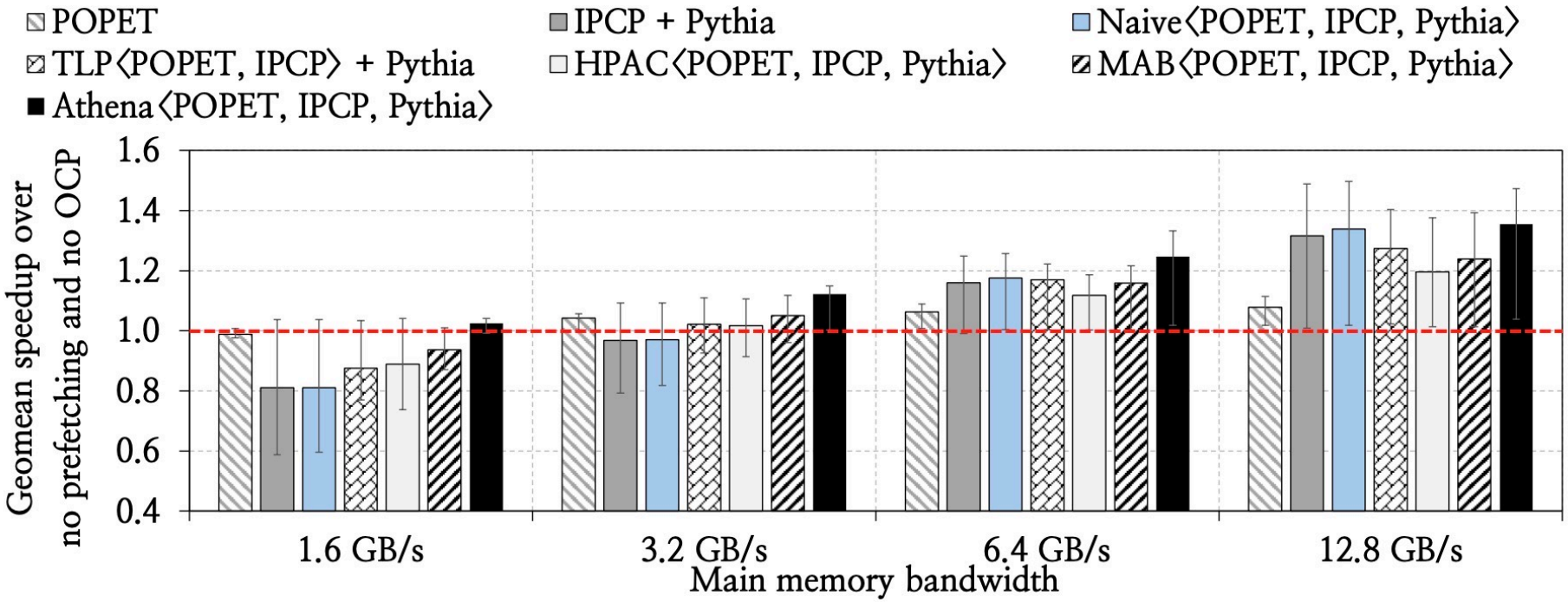
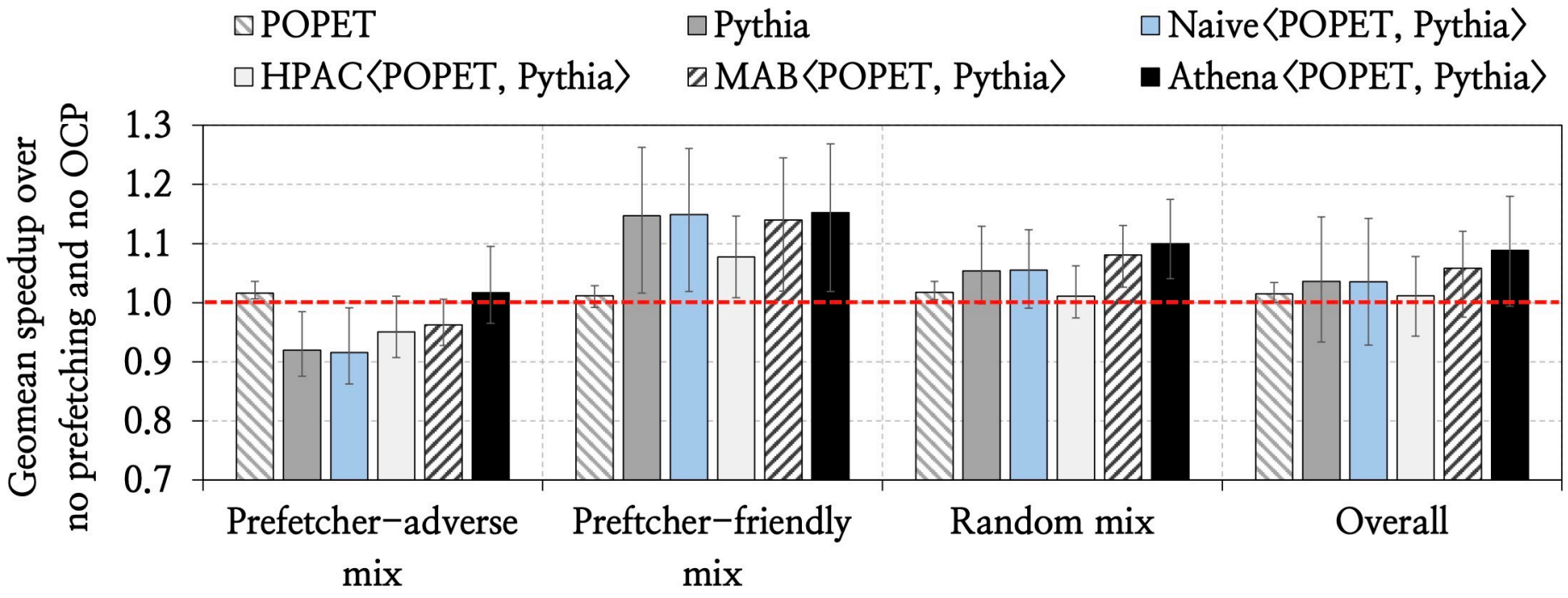


Figure 13: Performance sensitivity to prefetching mechanism at L1D in CD4.

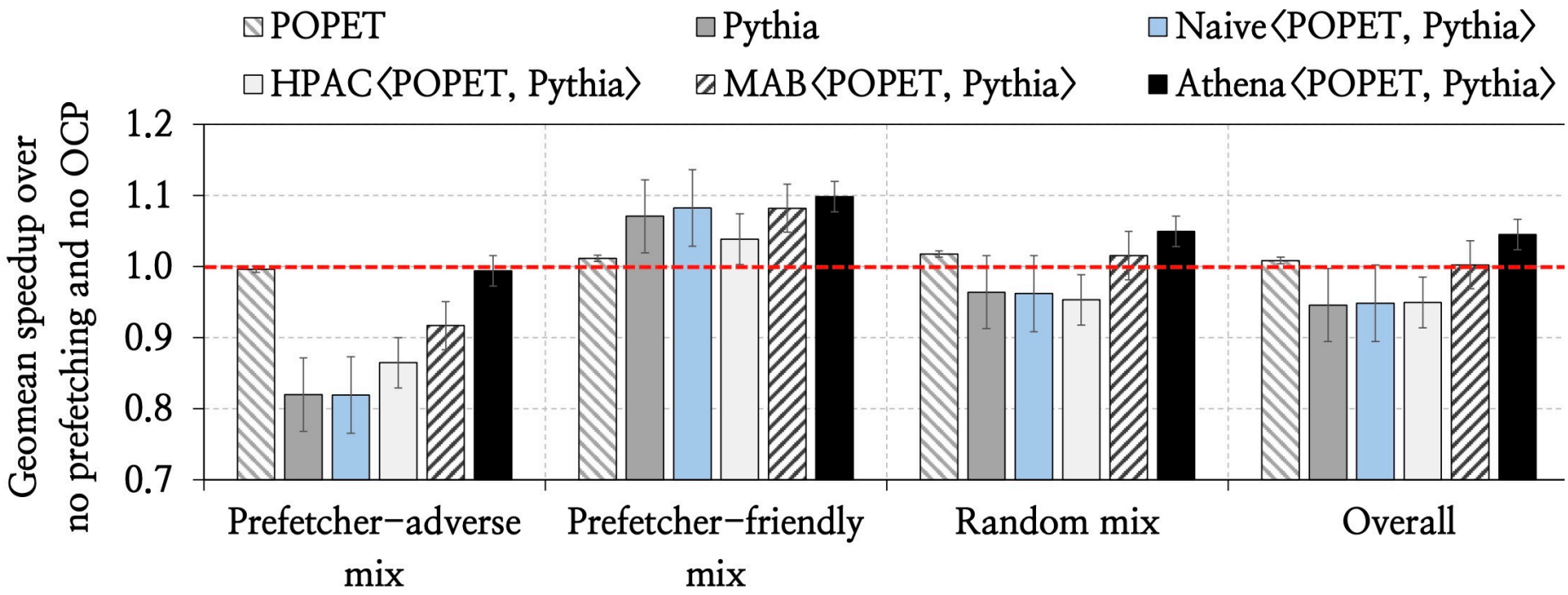
CD4 – Sensitivity to Main Memory Bandwidth



Four-Core Evaluation Results

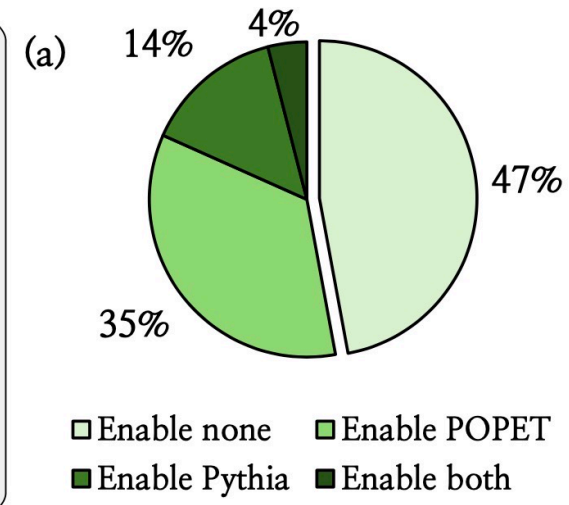


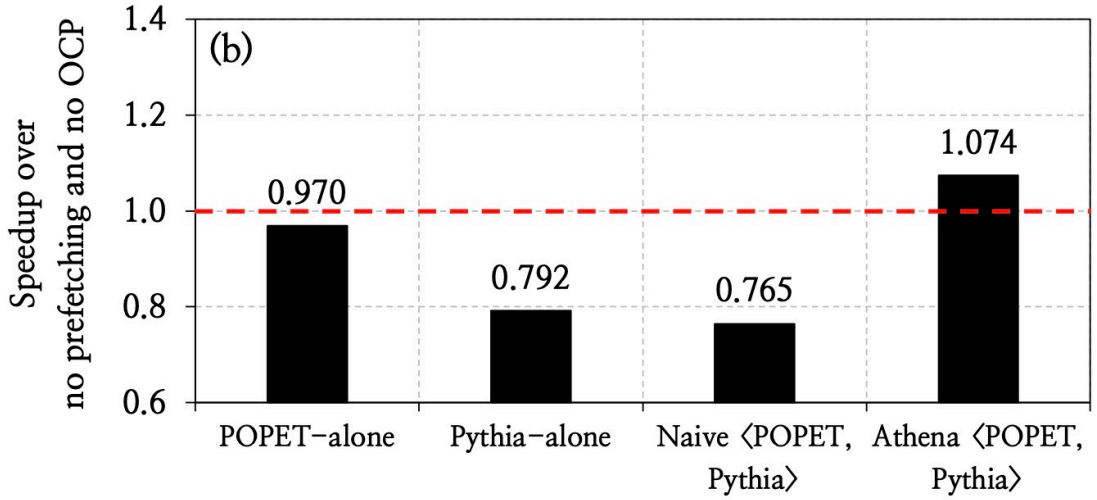
Eight-Core Evaluation Results



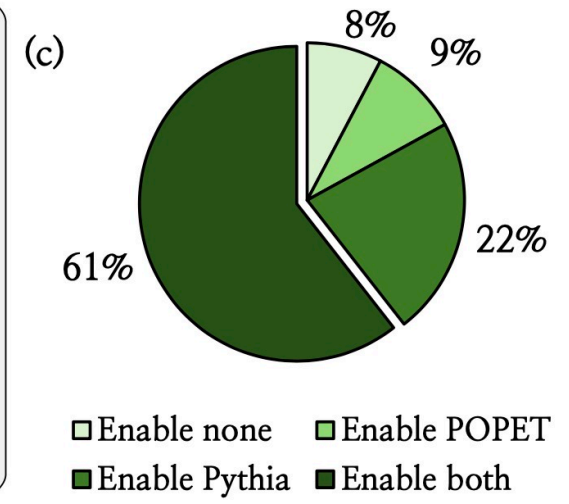
Understanding Athena using a Case Study

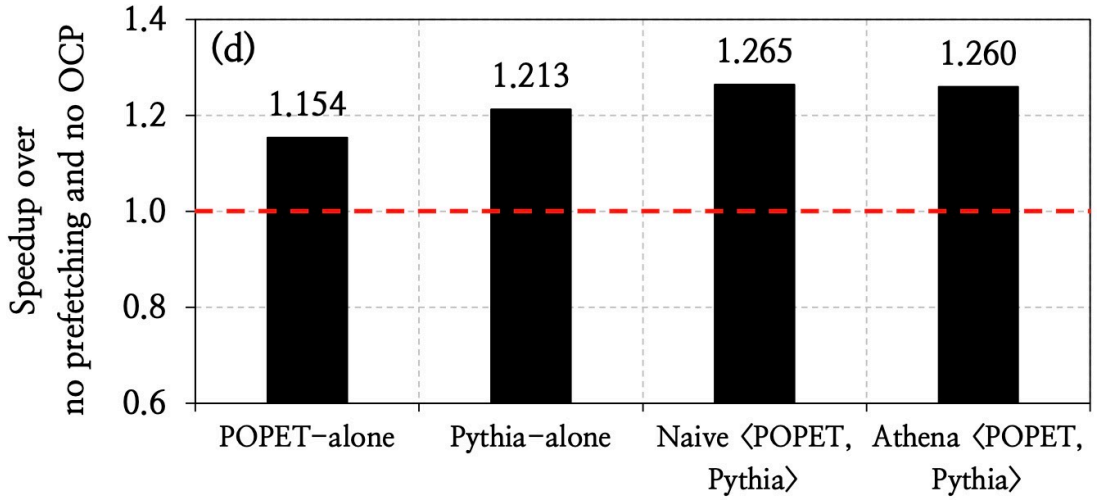
With 3.2 GB/s main memory bandwidth



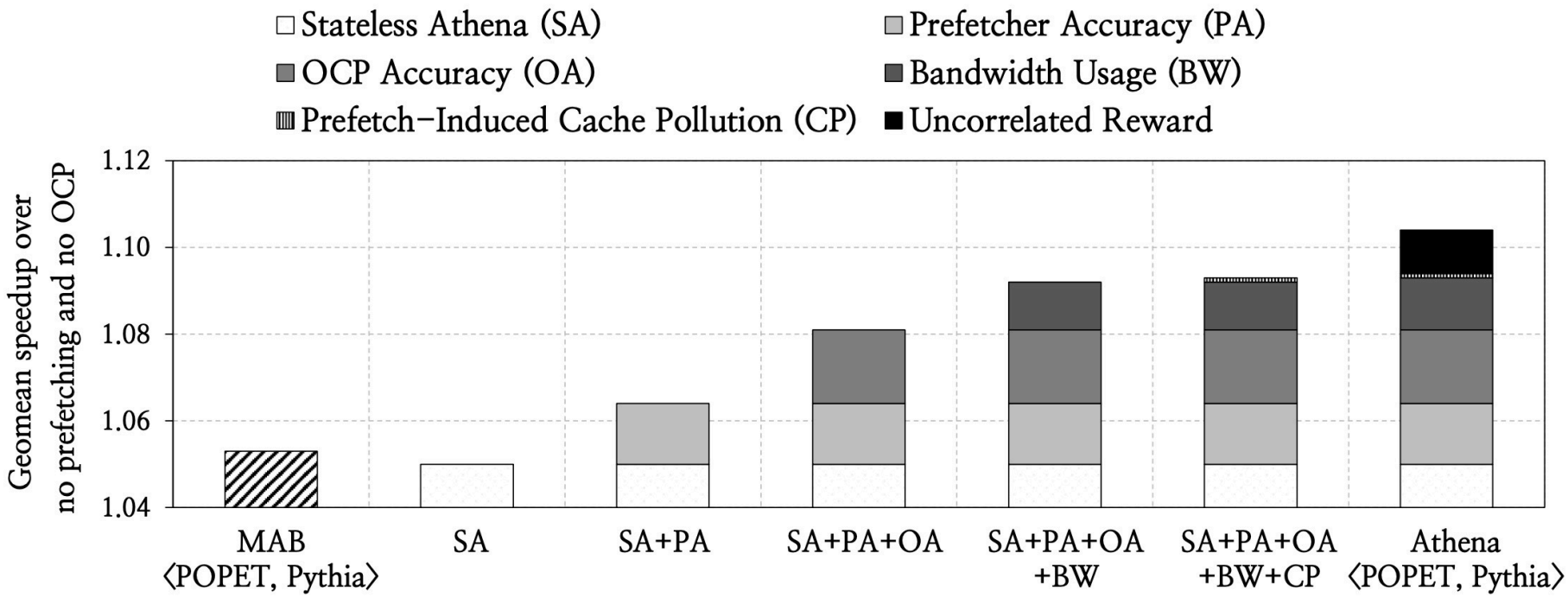


With 25.6 GB/s main memory bandwidth

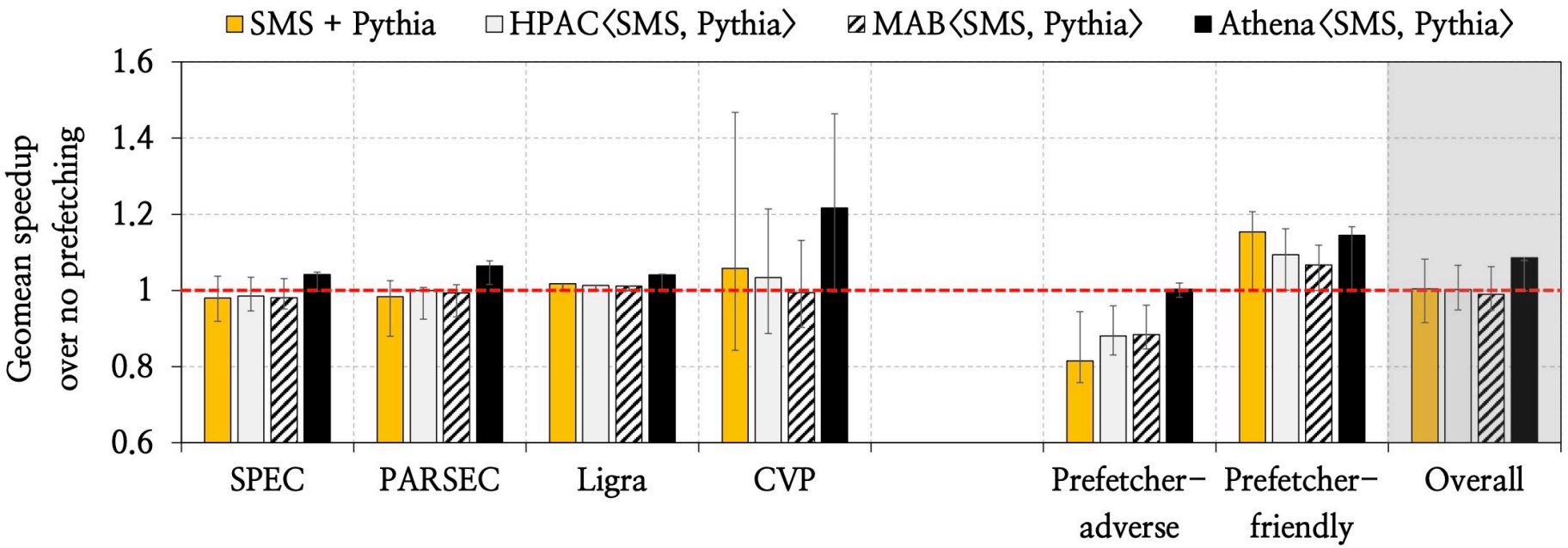




Athena Ablation Study

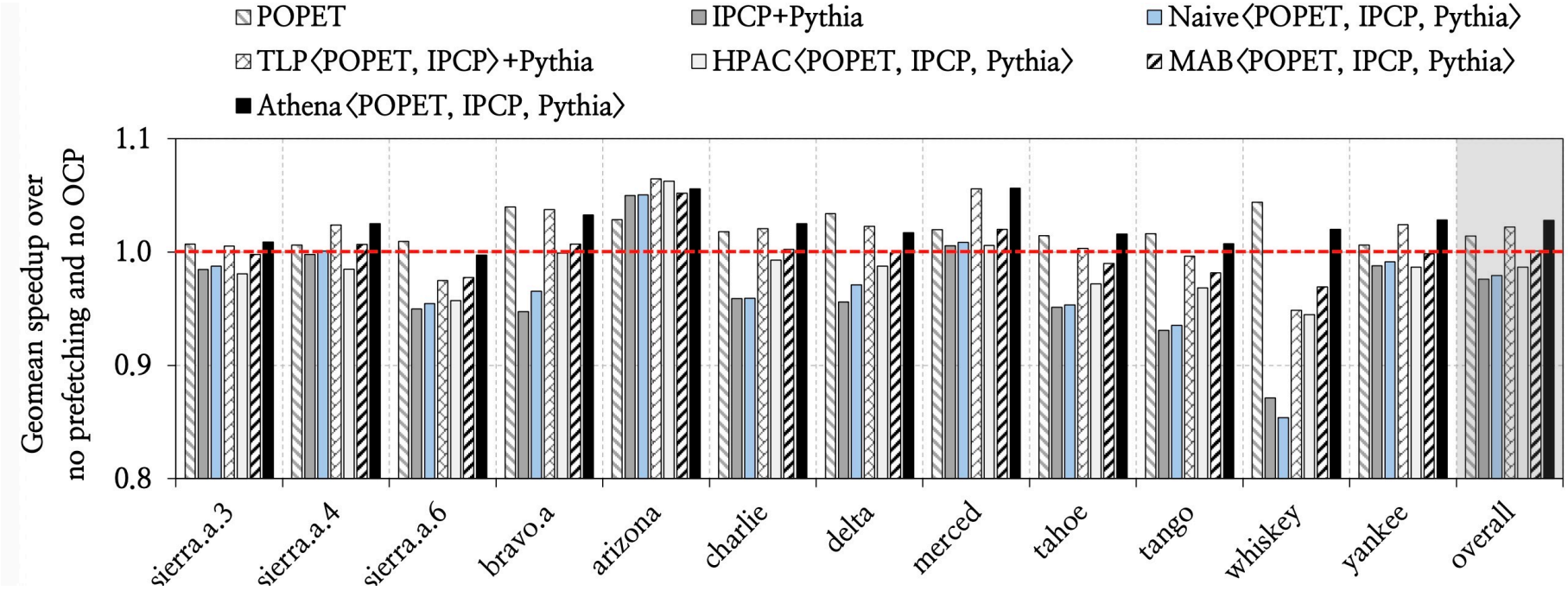


Athena for Prefetcher-Only Management



Athena on Unseen Workloads

Across **359** Google datacenter workload traces released in DPC4



Old Slides