Machine Learning for Fine-Grained Hardware Prefetcher Control

Jason Hiebel Laura E. Brown Zhenlin Wang

jshiebel@mtu.edu lebrown@mtu.edu zlwang@mtu.edu

Department of Computer Science
Michigan Technological University

International Conference on Parallel Processing
August 2019

Hardware Prefetching

The Contextual Bandit Problem

Experimental Design

Results

Conclusion

Intel Hardware Prefetchers

L2 Cache	**DPL**	**Data Prefetch Logic** ascending/descending stream prefetcher
	ACL	**Adjacent Cache Line** spatial prefetcher
L1 Cache	**DCU**	**Data Cache Unit** ascending stream prefetcher
	DCU IP	**Data Cache Unit Instruction Pointer** ascending/descending stride prefetcher

Resource Contention

- Increased usage and contention of shared resources
  - last level cache
  - off-chip memory bandwidth
- Adverse performance in (some) multi-tenant workloads!

Core	Benchmark	Average Performance (IPC)		Speedup
		DPL Enabled	DPL Disabled	
1	xalancbmk	0.45	0.54	20%
2	fotonik3d	0.73	0.56	−23%
3	lbm	0.79	0.75	−5%
4	omnetpp	0.18	0.30	67%
				15%

Optimizing Prefetcher Usage

Static Recommendations (Liao et al., SC '09; Rahman et al., HPCC '15)

- Recommend workload-specific configuration
- Requires prior profiling/evaluation of workload

Dynamic Optimization (Jiménez et al., PACT '12)

- Periodically test configuration performance and exploit
- Requires enumerating configurations of interest

Optimizing Prefetcher Usage

- **Contextual Bandit** model for hardware prefetcher control
- Dynamic control using architectural metrics
  (branch mispredictions, cache misses, memory bandwidth usage, etc.)
- Learn model for prefetcher control using random profiling data

Hardware Prefetching

The Contextual Bandit Problem

Experimental Design

Results

Conclusion

The Contextual Bandit

Sequential model for decision making with **limited feedback**

(Langford and Zhang, NIPS '07; Beygelzimer and Langford, SIGKDD '09)

For iteration $t = 1, 2, \ldots$

1. Observe contextual information

   (cache, memory behavior)

2. Select an action

   (hardware prefetcher configuration)

3. Receive reward for the selected action

   (performance improvement)

Binary-Offset

(Beygelzimer and Langford, SIGKDD '09)

Off-Policy/Offline method

learn from logged profiling data using random actions

- Convert from bandit data to weighted classification data
- Learn control model using a weighted classifier

$$\begin{pmatrix} \text{context} \\ \text{action} \\ \text{reward} \end{pmatrix} \Longrightarrow \begin{pmatrix} \text{context} \\ \text{label} \\ \text{weight} \end{pmatrix} \Longrightarrow \text{model}$$

Contextual Information

Challenge

Identify relevent architectural behaviors (independent variables)

Solution

- Performance Monitoring Unit (PMU)
- Utilize domain expertise to select a subset of hardware events

Cache

`L1D:ALLOCATED_IN_M`
`L1D:M_EVICT`
`L1D:REPLACEMENT`
`L2_LINES_IN:ANY`
`L3_LAT_CACHE:MISS`

Memory Bandwidth

`OFFCORE_REQUESTS:DEMAND_DATA_RD`

DTLB Misses

`DTLB_LOAD_MISSES:WALK_COMPLETED`

Blocked Loads

`LD_BLOCKS:STORE_FORWARD`
`LD_BLOCKS:NO_SR`
`LD_BLOCKS:ALL_BLOCK`

Branch Mispredictions

`BR_MISP_RETIRED:ALL_BRANCHES`

Action Selection

Challenge

Exponential $\left(2^{4 \cdot \text{cores}}\right)$ system-wide configurations

Solution

- Myopic control: separate bandit **per-core** and **per-prefetcher**
- Binary action sets $\{ \text{enabled}, \text{disabled} \}$
- System-wide effects as part of the context and reward

Reward Formulation

Challenge

Maximize average workload speedup, $\frac{1}{n} \sum {IPC_i}^{\text{conf}} / {IPC_i}^{\text{base}}$

Solution

- Extract program phases using performance change-points
- Local (i) per-phase speedup vs average: $R_{(i),t}$
- Cross-core (i,j) per-phase speedup vs average: $R_{(i,j),t}$
- Average system-wide speedup:

$$R_{i,t} = \frac{1}{n}\left(R_{(i),t} + \sum_{i \neq j} R_{(i,j),t}\right) - 1$$

Hardware Prefetching

The Contextual Bandit Problem

Experimental Design

Results

Conclusion

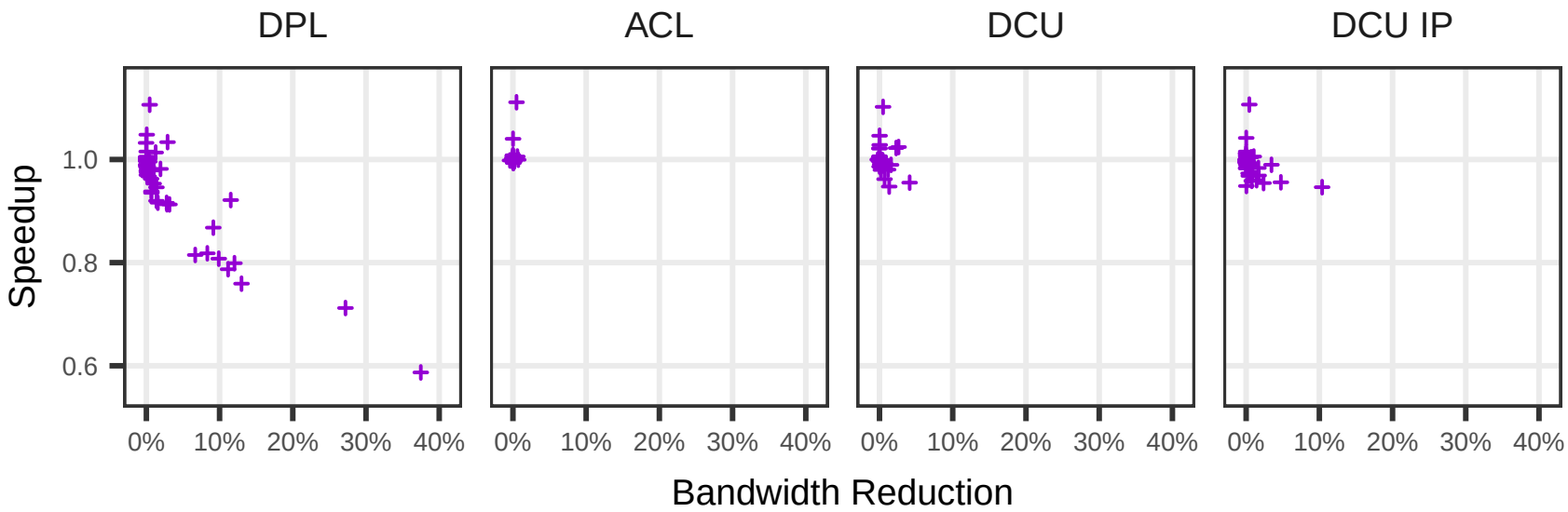
Evaluation Environment

	CPU		Cores	Memory	
Sandy Bridge	3.3 GHz	Core i5-2500	4	2x2 GB	(DDR3-1333)
Kaby Lake	3.6 GHz	Core i7-7700	4	4x8 GB	(DDR4-2400)
Broadwell	2.1 GHz	Xeon E5-2620 v4	8	4x32 GB	(DDR4-2400)

- Disable turbo-boost
- Disable energy saving features
- Disable hyper-threading (prefetchers on physical cores)

Workload Construction

- Generate 60 workloads consisting of four benchmarks
- Benchmark suites:
  SPEC CPU2006, SPEC CPU2017, PARSEC
- Determine benchmark sensitivity to each prefetcher (Broadwell)



Workload Construction

SPEC CPU2006

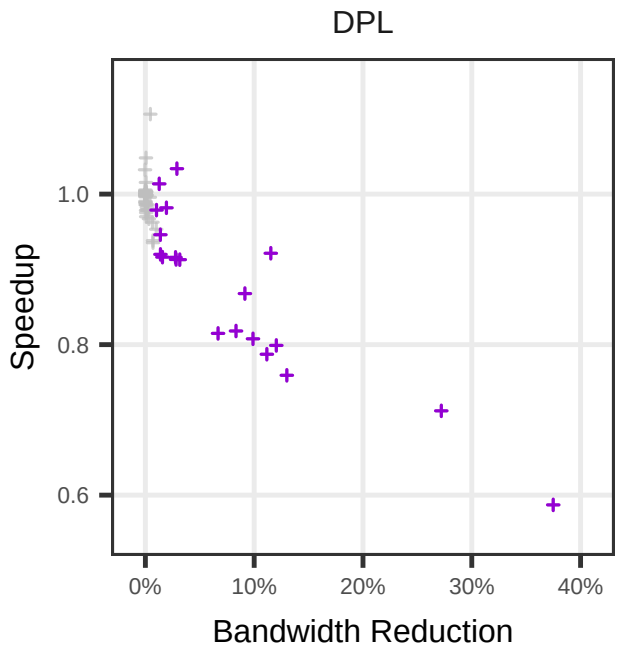
`bwaves`, `gcc`, `GemsFDTD`, `lbm`, `leslie3d`, `libquantum`, `mcf`, `milc`, `omnetpp`, `soplex`, `wrf`, `xalancbmk`

SPEC CPU2017

`bwaves_r`, `fotonik3d_r`, `gcc_r`, `lbm_r`, `mcf_r`, `omnetpp_r`, `roms_r`

PARSEC

`fluidanimate`



DPL Prefetcher Selection

Baselines

- **DPL Enabled** (on all cores)
- **DPL Disabled** (on all cores)
- **Best Static**

Dynamic Policies

- **Binary-Offset (Ind)**
  create specialized model for each workload
- **Binary-Offset (X)**
  create general model using X training workloads

Hardware Prefetching

The Contextual Bandit Problem

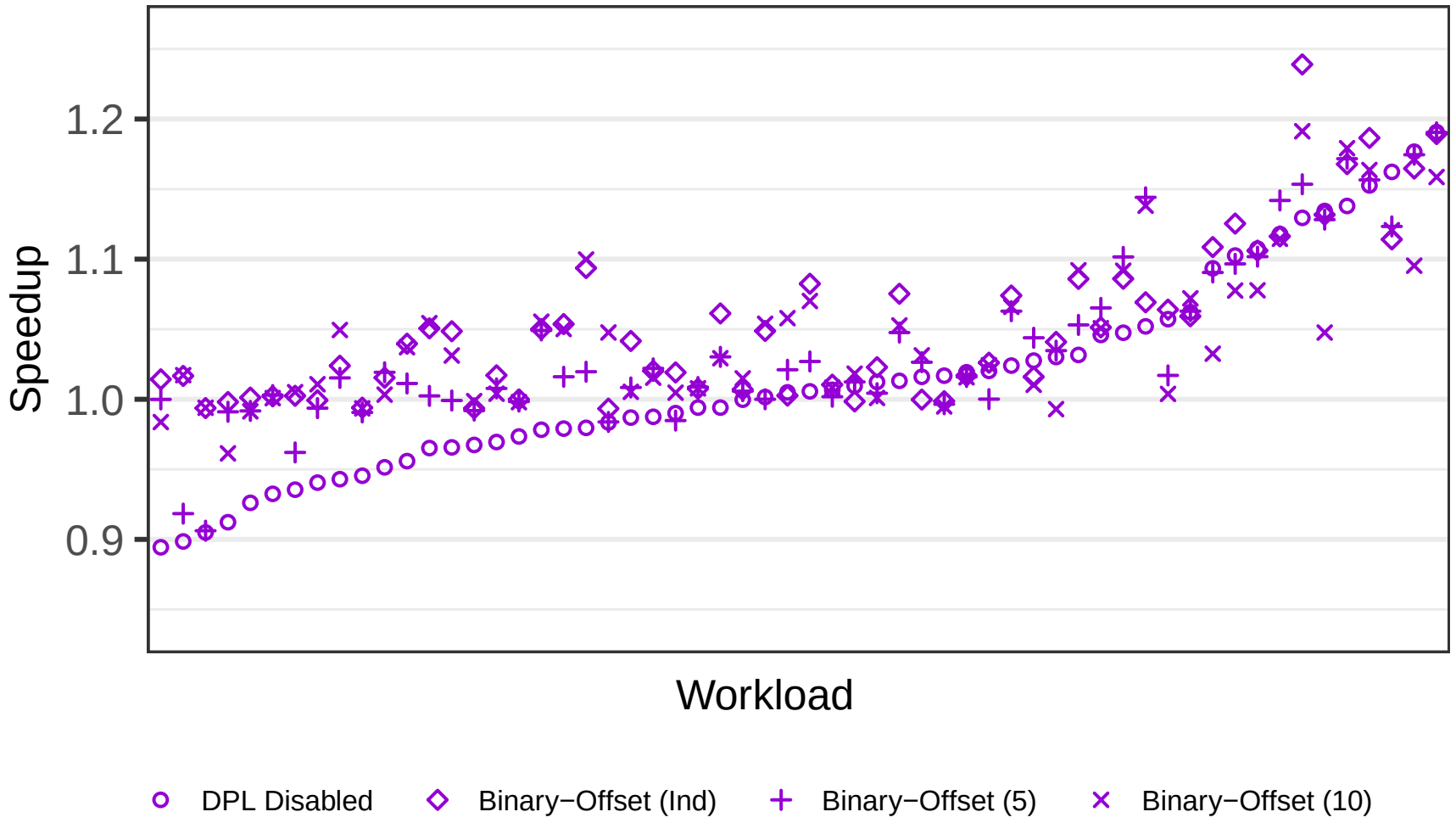
Experimental Design

Results

Conclusion

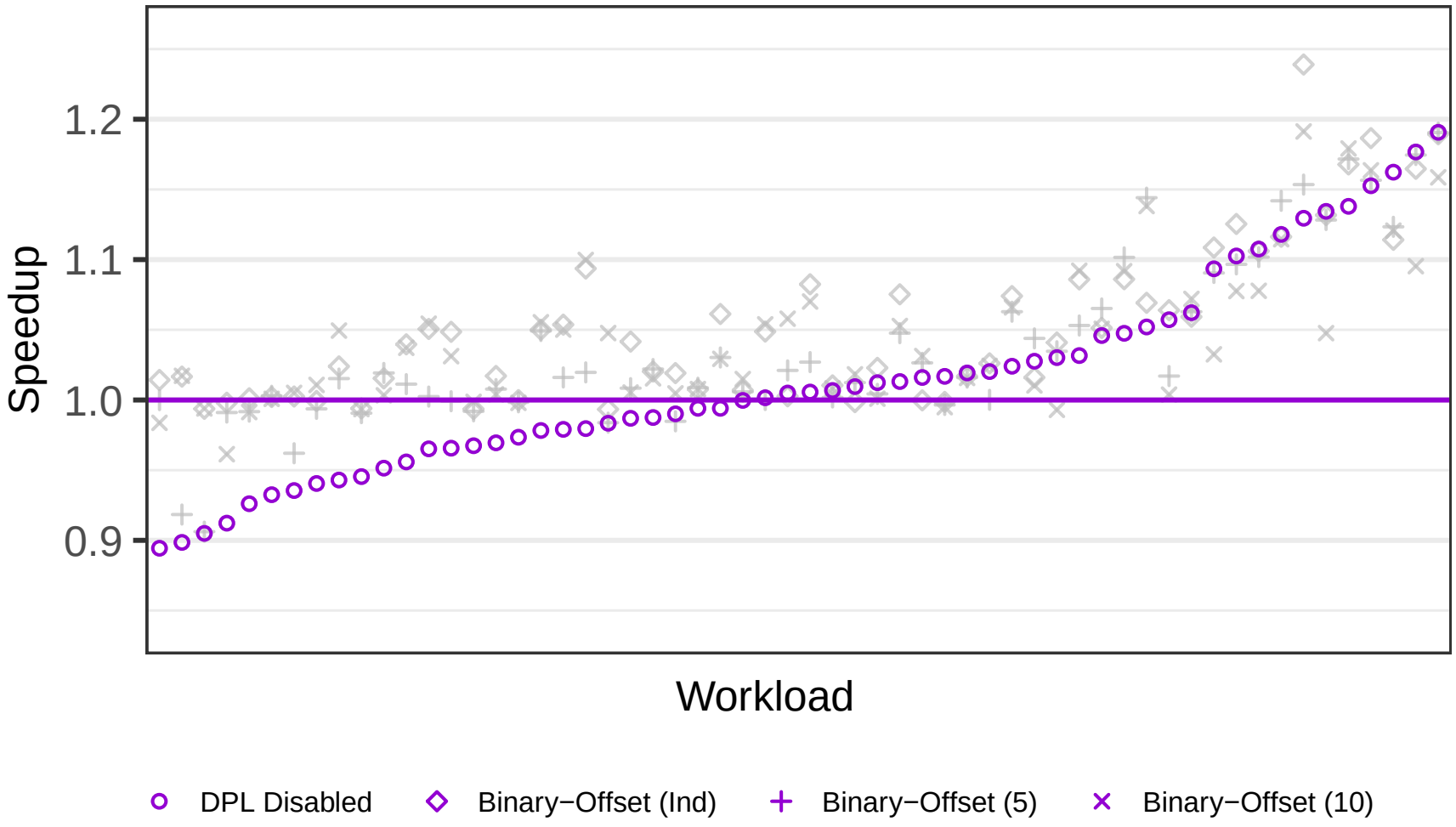
DPL Prefetcher (Sandy Bridge)

Overview



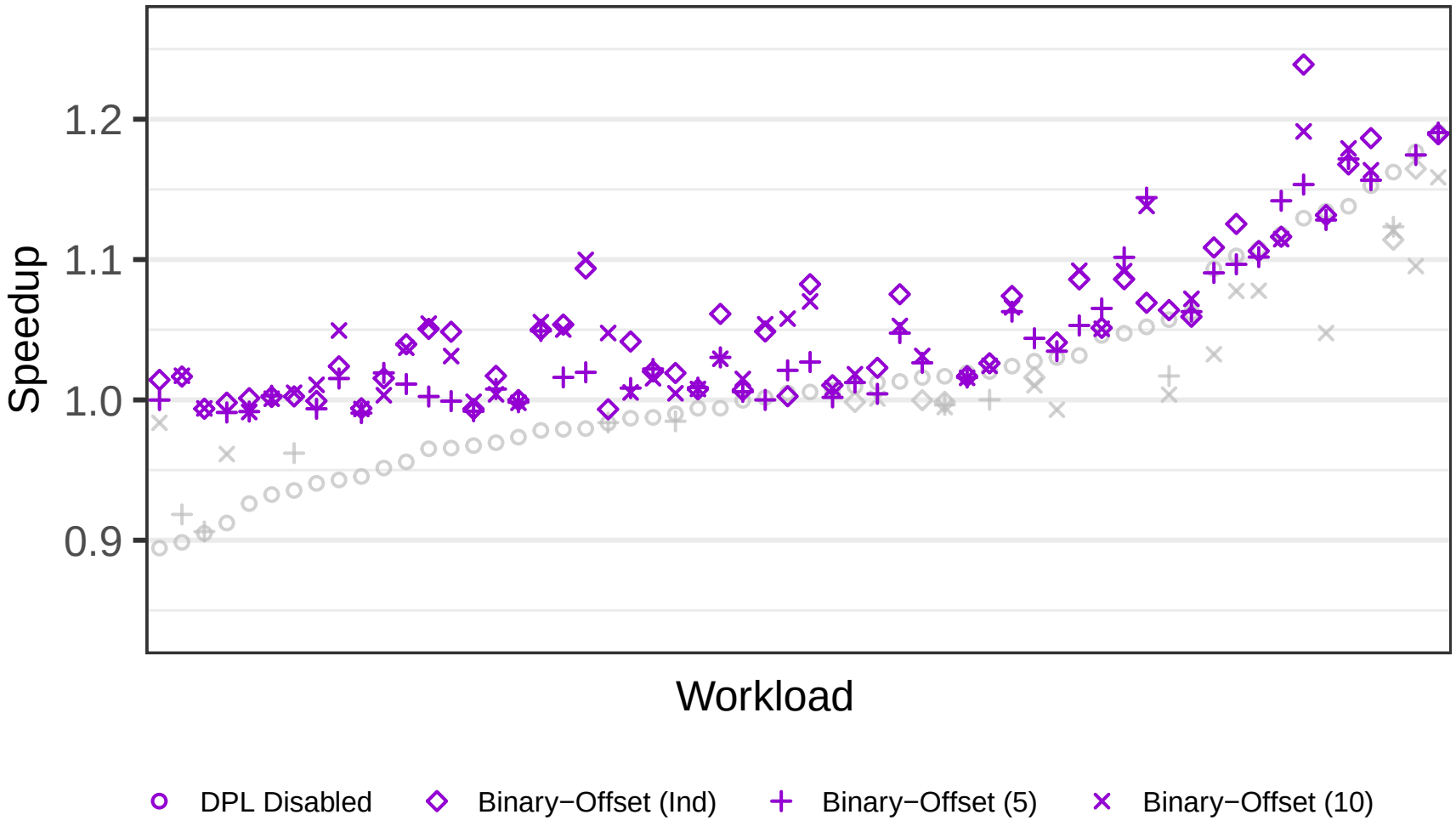
DPL Prefetcher (Sandy Bridge)

Baseline Performance



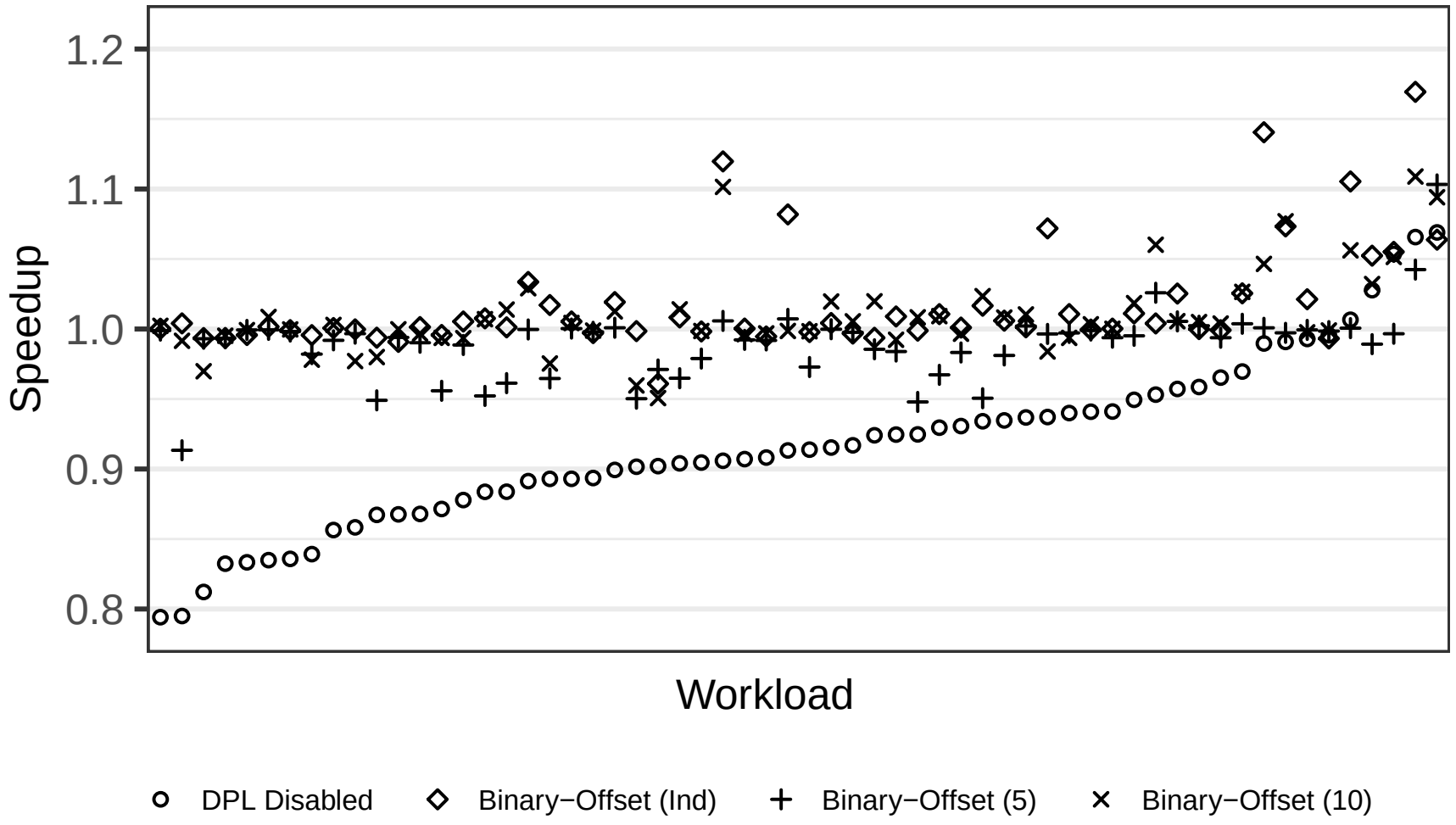
DPL Prefetcher (Sandy Bridge)

Outperforming “Best Static” Performance



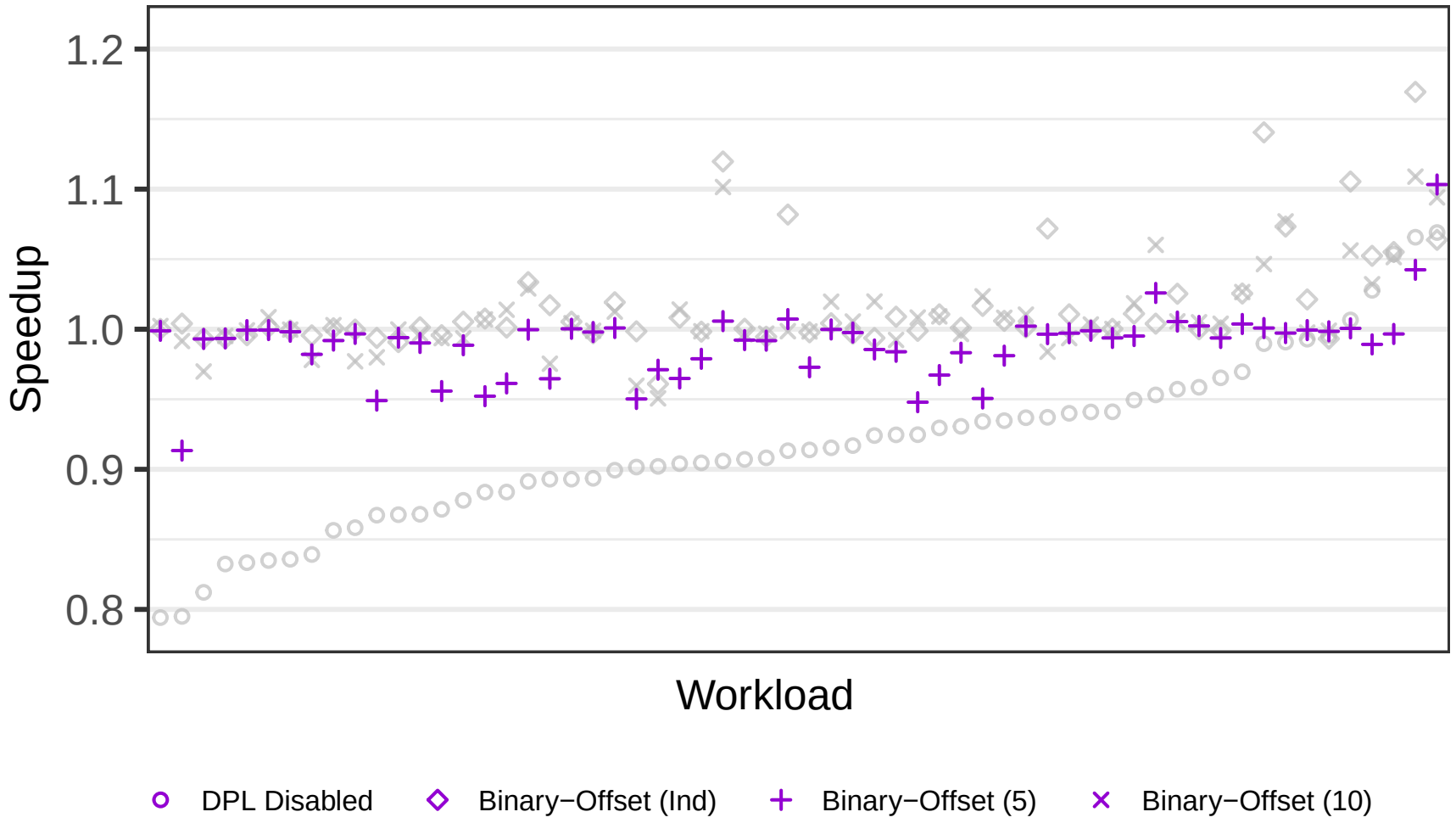
DPL Prefetcher (Kaby Lake)

Overview

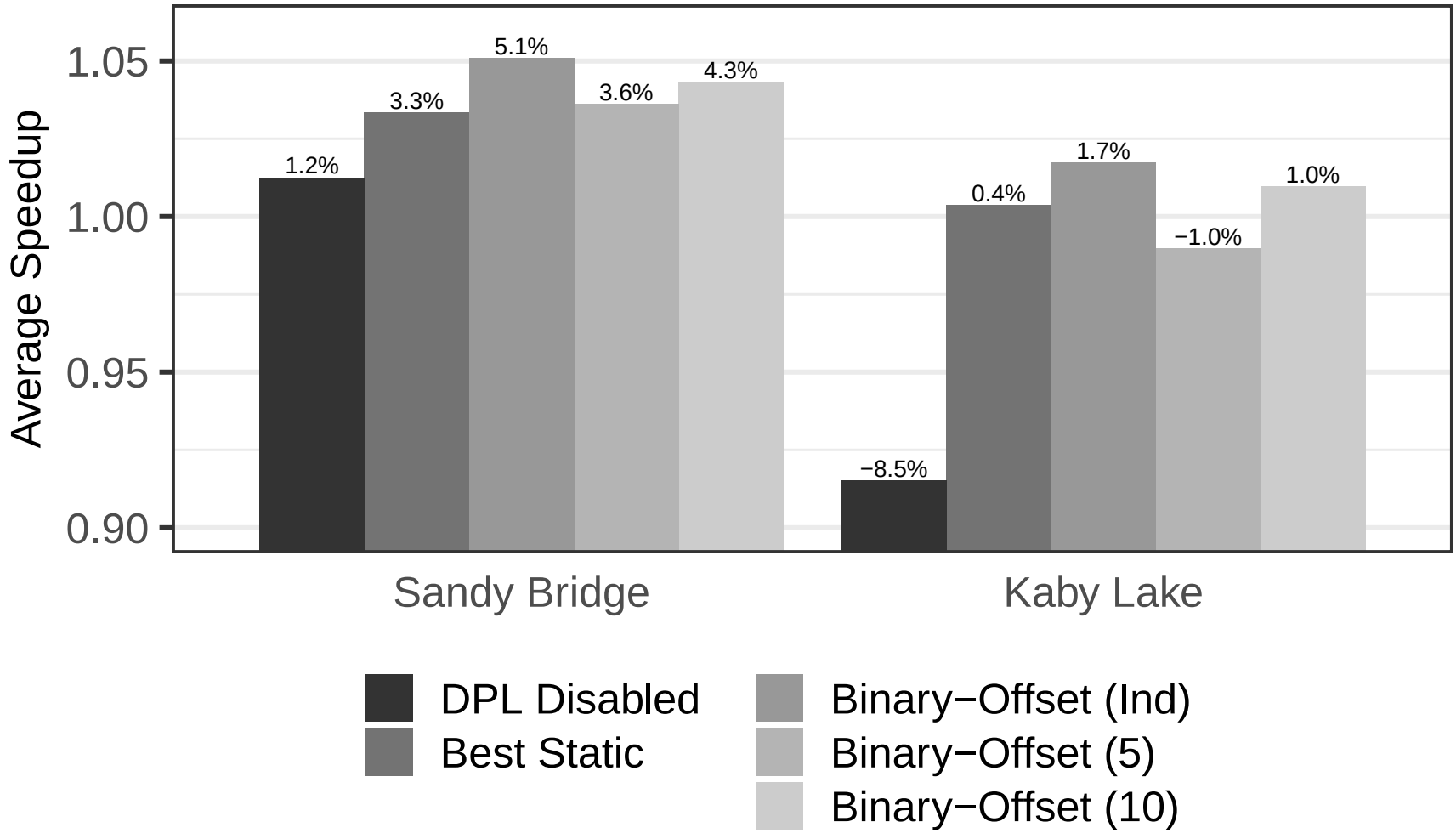


DPL Prefetcher (Kaby Lake)

Training Data Impact



DPL Prefetcher Selection



Hardware Prefetching

The Contextual Bandit Problem

Experimental Design

Results

Conclusion

Conclusion

- Contextual bandit model of hardware prefetcher control
  - independent control of hardware prefetchers per core
  - system-wide context, feedback
- Binary-Offset policy improves upon static configurations
  - dynamic control
  - general, workload-agnostic configuration control
  - random (non-enumerative) profiling data

Machine Learning for Fine-Grained Hardware Prefetcher Control

Jason Hiebel Laura E. Brown Zhenlin Wang

jshiebel@mtu.edu lebrown@mtu.edu zlwang@mtu.edu

Department of Computer Science
Michigan Technological University

International Conference on Parallel Processing
August 2019

References I

[1] Alina Beygelzimer and John Langford.

The offset tree for learning with partial labels.

In *Proceedings of 15th International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 129–38, 2009.

[2] Victor Jiménez, Roberto Gioiosa, Francisco J. Cazorla, Alper Buyuktosunoglu, Pradip Bose, and Francis P. O'Connell.

Making data prefetch smarter: Adaptive prefetching on POWER7.

In *21st International Conference on Parallel Architectures and Compilation Techniques*, PACT '12, pages 137–146, 2012.

[3] John Langford and Tong Zhang.

The epoch-greedy algorithm for contextual multi-armed bandits.

In *Advances in Neural Information Processing Systems 20*, NIPS, pages 817–824, 2007.

References II

[4] Shih-wei Liao, Tzu-Han Hung, Donald Nguyen, Chinyen Chou, Chiaheng Tu, and Hucheng Zhou.

Machine learning-based prefetch optimization for data center applications.

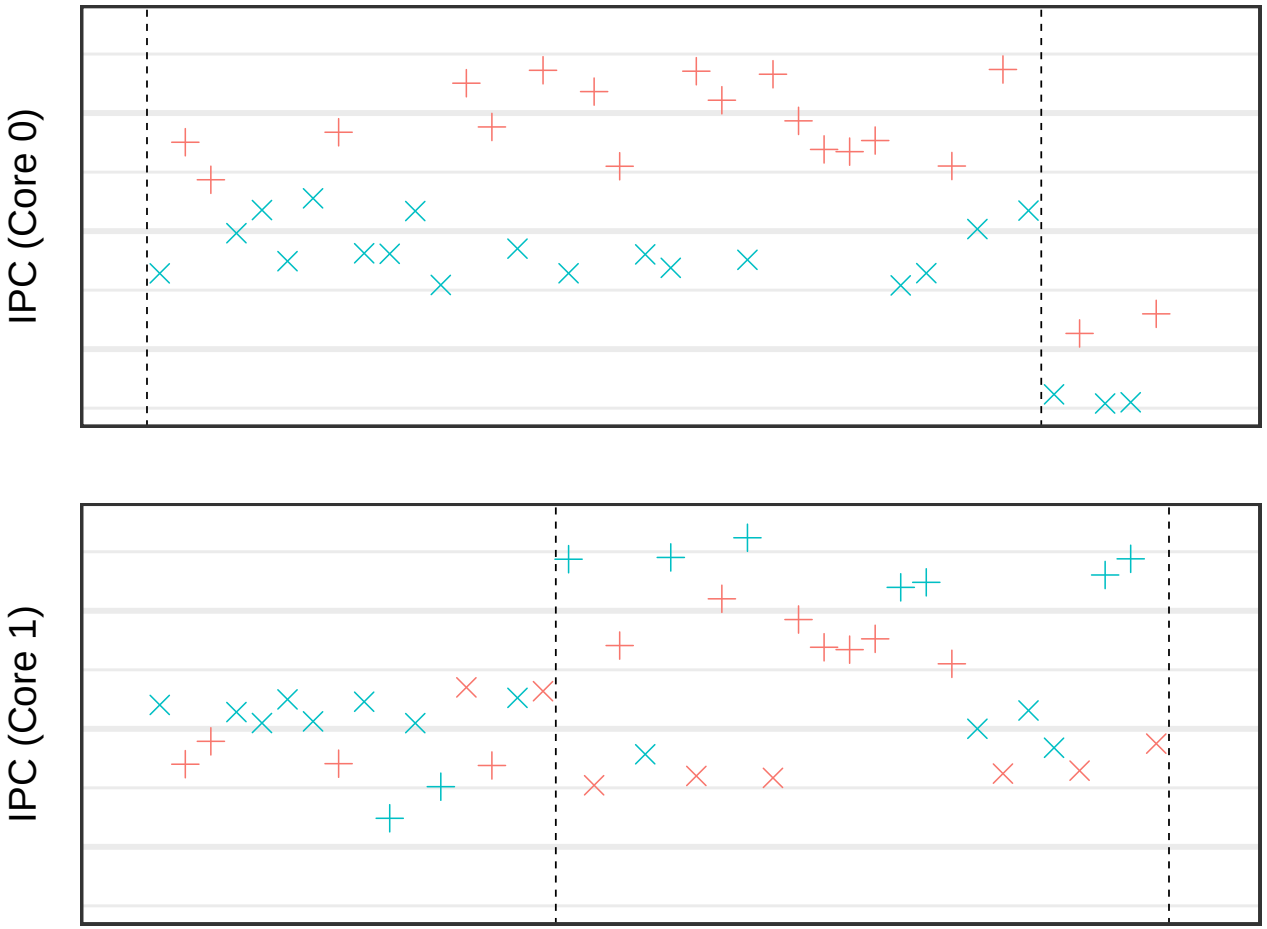
In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC '09, pages 1–10, 2009.

[5] Saami Rahman, Martin Burtscher, Ziliang Zong, and Apan Qasem.

Maximizing hardware prefetch effectiveness with machine learning.

In *Proceedings of the 17th International Conference on High Performance Computing and Communications*, pages 383–389, 2015.

Reward Formulation



Reward Formulation