The Entangling Data Prefetcher

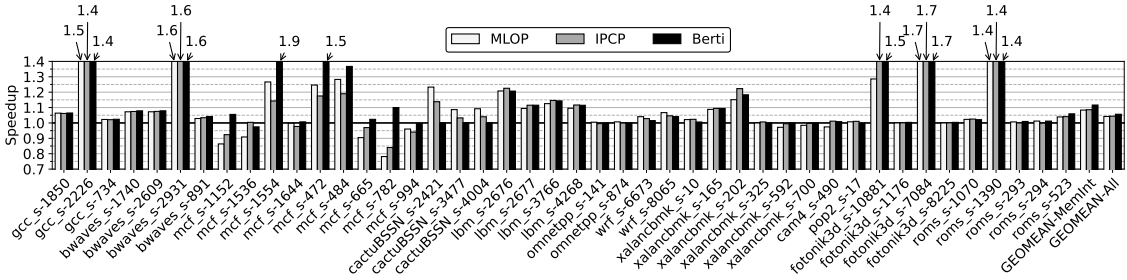
Agustín Navarro-Torres[1], Simranjit Singh[2], Biswabandan Panda[3], Alberto Ros[2]

[1]Universidad de Zaragoza (agusnt@unizar.es)
[2]University of Murcia (simranjit.singh@um.es, aros@ditec.um.es)
[3]Indian Institute of Technology Bombay (biswa@cse.iitb.ac.in)

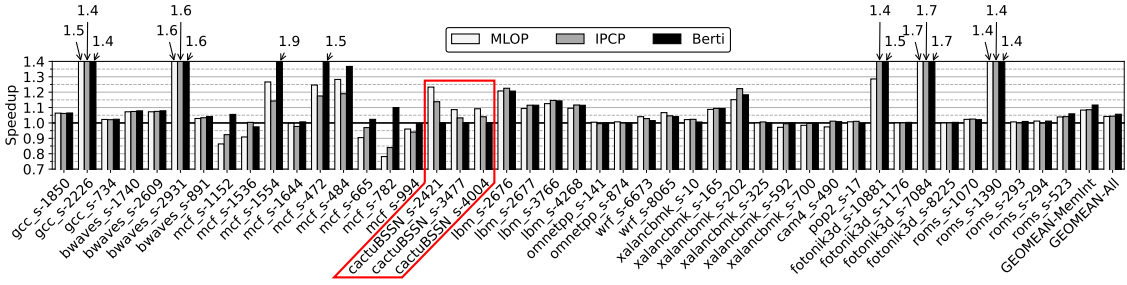
4th Data Prefetching Championship
February 1, 2026

Where does Berti fails?

Where does Berti fails?



Where does Berti fails?



In `cactuBSSN` Berti achieves 0% performance improvement

Why can Berti not cover `cactuBSSN`?

- Too many IPs
- *Very-long-time distance strides*

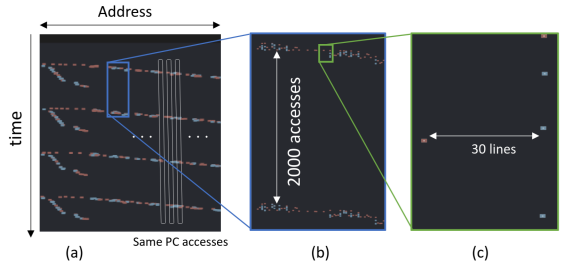
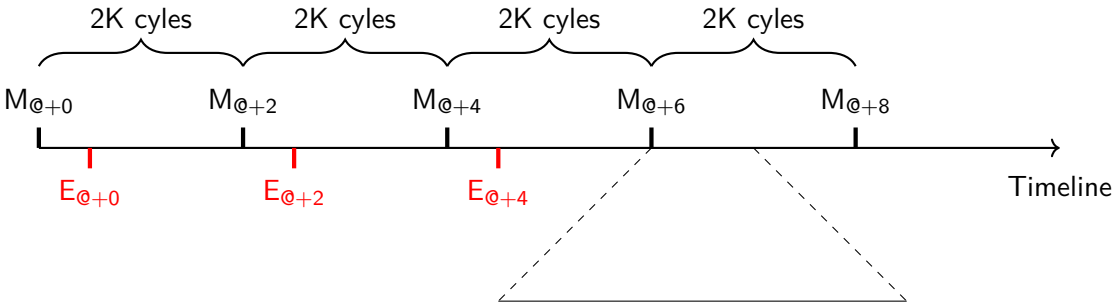


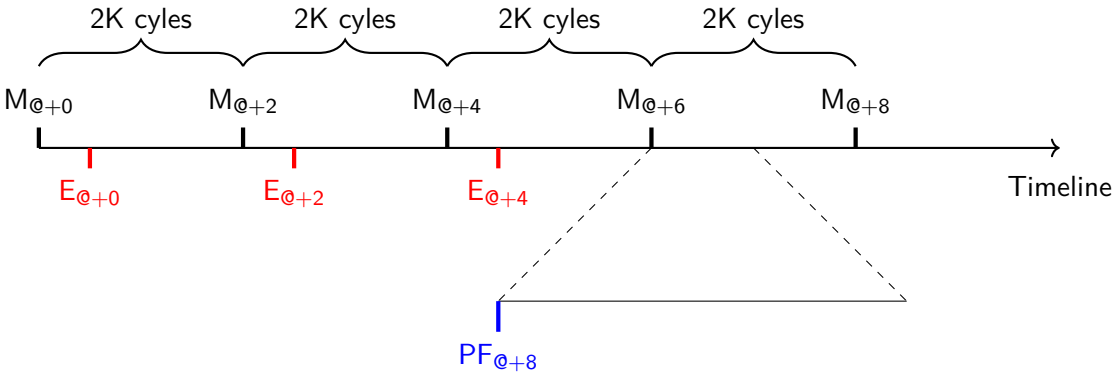
Fig. 1: A part of memory access pattern of `607.cactuBSSN_s-2421B`. The horizontal axis is the physical address and the vertical axis is time. [1]

[1] T-SKID: Timing Skid Prefetcher, *Tomoki Nakamura et. al*, DPC-3 and DATE

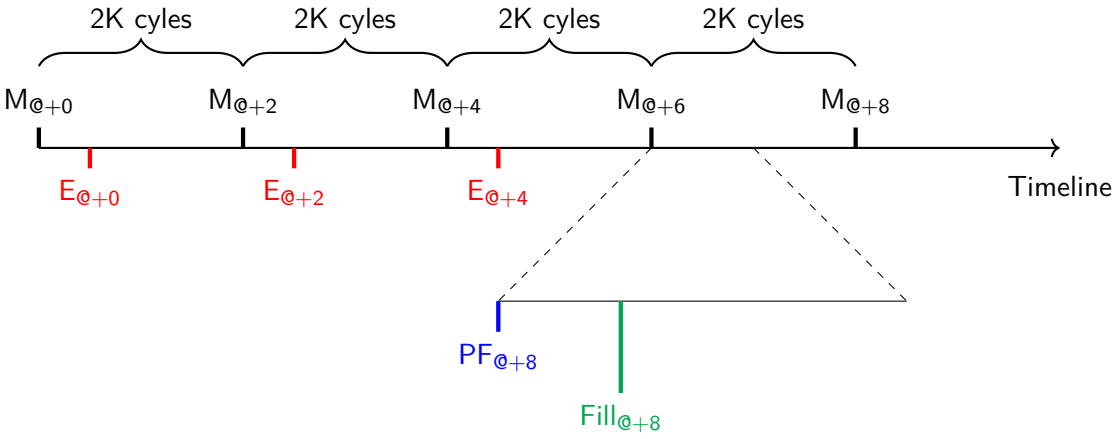
Why can Berti not cover cactuBSSN?



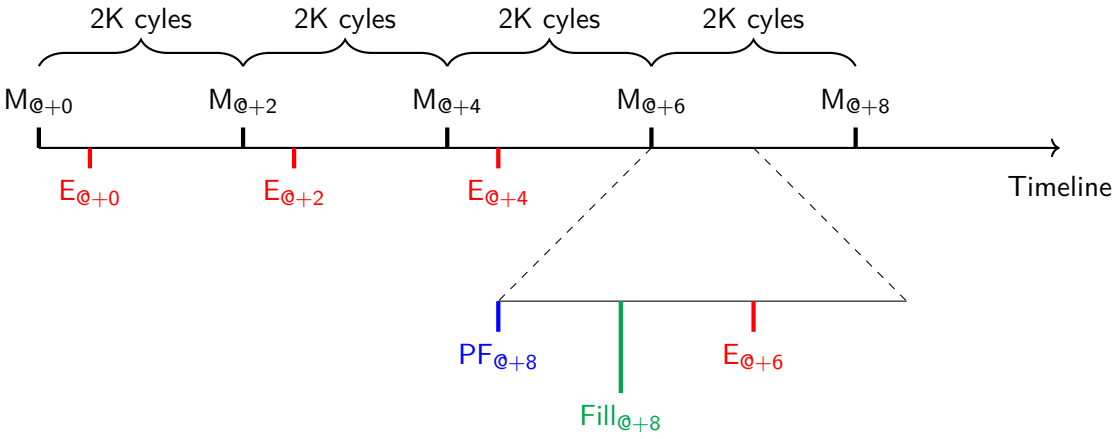
Why can Berti not cover cactuBSSN?



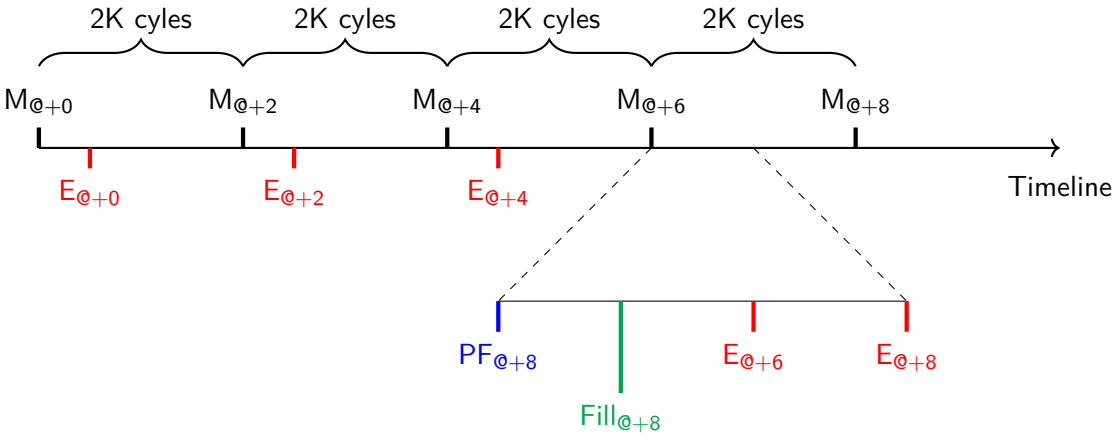
Why can Berti not cover cactuBSSN?



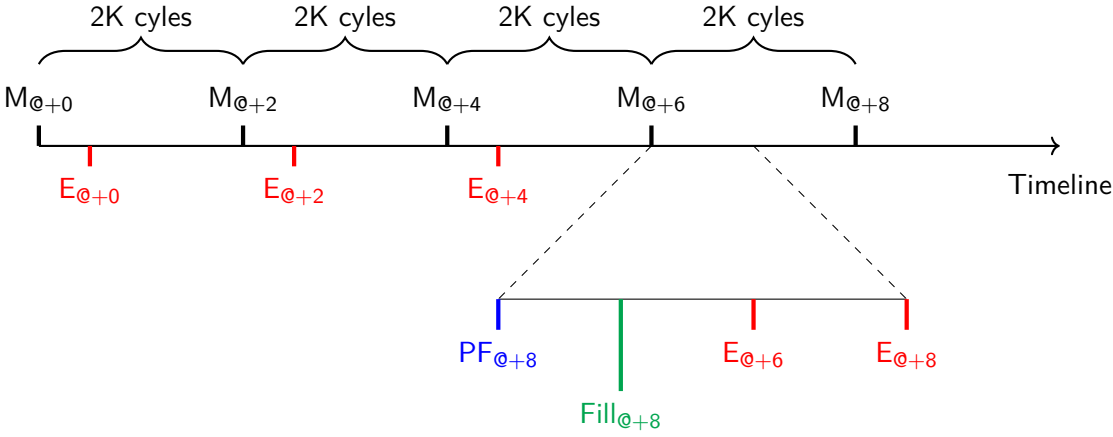
Why can Berti not cover cactuBSSN?



Why can Berti not cover cactuBSSN?



Why can Berti not cover cactuBSSN?



Berti fails to capture regular strides with large inter-access intervals (hundreds of cycles).

T-SKID

T-SKID[2] captures long-reuse and zero-strides patterns:

- The IP that uses and learns stride patterns **CAN** be different from the one that trigger
- It covers very-long-time distance & zero- stride
- Builds on top of an IP-Stride

[2] T-SKID: Timing Skid Prefetcher, *Tomoki Nakamura et. al*, DPC-3

T-SKID

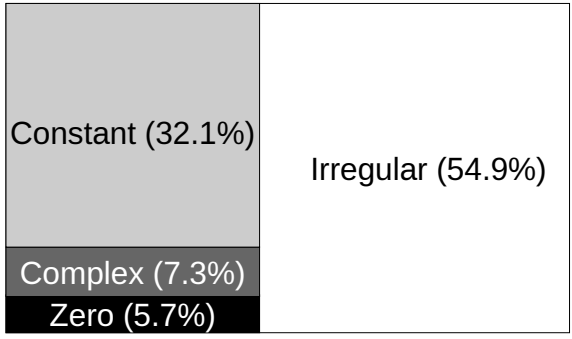
T-SKID[2] captures long-reuse and zero-strides patterns:

- The IP that uses and learns stride patterns **CAN** be different from the one that trigger
- It covers very-long-time distance & zero- stride
- Builds on top of an IP-Stride

It inherents all the issues of an IP-Stride prefetcher

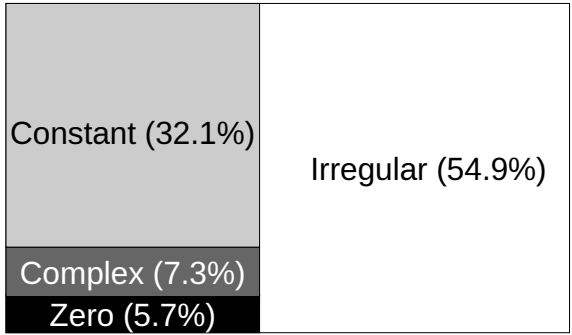
[2] T-SKID: Timing Skid Prefetcher, *Tomoki Nakamura et. al*, DPC-3

Which portions of patterns are not covered by Berti or T-SKID



SPEC CPU2017 patterns

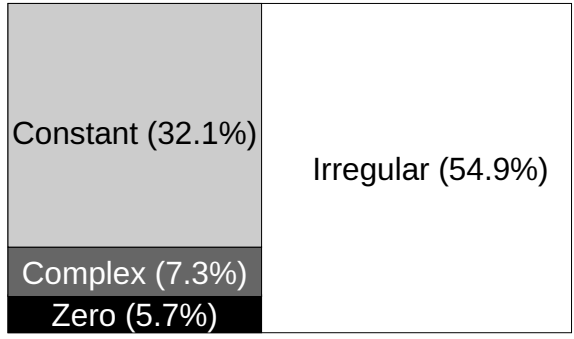
Which portions of patterns are not covered by Berti or T-SKID



SPEC CPU2017 patterns

Berti can not cover zero- and very-long-time distance strides (~5.7%)

Which portions of patterns are not covered by Berti or T-SKID



SPEC CPU2017 patterns

Berti can not cover zero- and very-long-time distance strides (~5.7%)

T-SKID can not cover complex stride patterns (~7.3%)

Can a prefetcher cover complex and long time distance patterns?

Can we have a prefetcher that covers both patterns at the same time?

Can a prefetcher cover complex and long time distance patterns?

Can we have a prefetcher that covers both patterns at the same time?

The Entangling Data Prefetcher (EDP)

Can a prefetcher cover complex and long time distance patterns?

Can we have a prefetcher that covers both patterns at the same time?

The Entangling Data Prefetcher (EDP)

- Cover complex patterns
- Cover very-long-time distance strides & zero-strides
- Improve timeliness

The Entangling Data Prefetcher

Complex patterns

The Entangling Data Prefetcher is built on top of Berti:

- Use **deltas** (distance between two or more accesses)
- It uses only **timely** deltas to prefetch

The Entangling Data Prefetcher

Complex patterns

The Entangling Data Prefetcher is built on top of Berti:

- Use **deltas** (distance between two or more accesses)
- It uses only **timely** deltas to prefetch

Complex patterns

Already covered thanks to Berti ☺

The Entangling Data Prefetcher

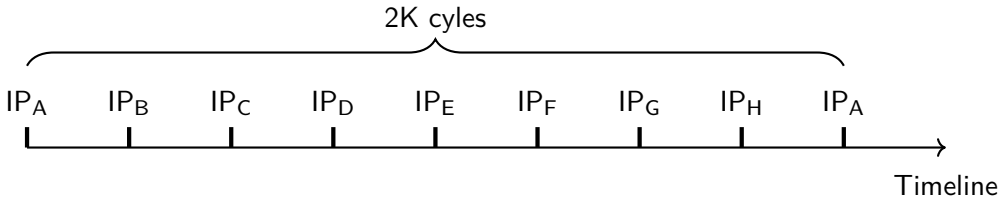
Long time distance patterns

Decoupling IP that learns patterns and IP that trigger patterns (similar to T-SKID)

The Entangling Data Prefetcher

Long time distance patterns

Decoupling IP that learns patterns and IP that trigger patterns (similar to T-SKID)

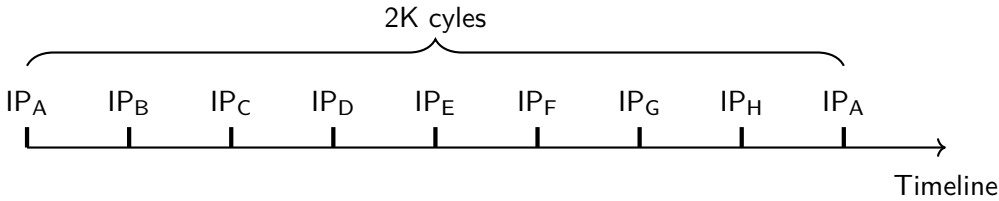


Why not use a previous IP to trigger prefetch requests

The Entangling Data Prefetcher

Long time distance patterns

Decoupling IP that learns patterns and IP that trigger patterns (similar to T-SKID)



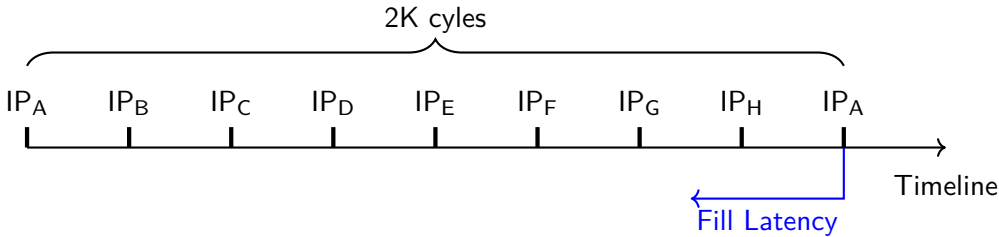
Why not use a previous IP to trigger prefetch requests

But which one?

The Entangling Data Prefetcher

Long time distance patterns

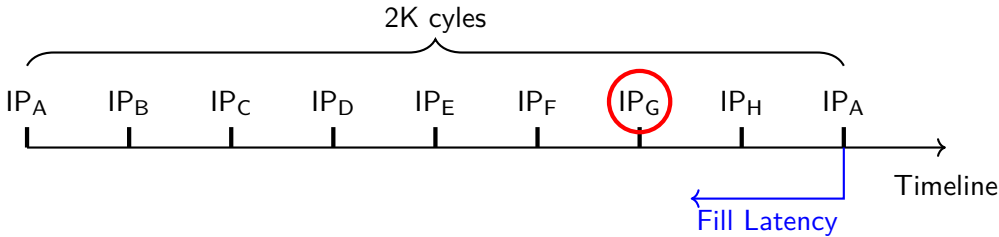
Decoupling IP that learns patterns and IP that trigger patterns (similar to T-SKID)



The Entangling Data Prefetcher

Long time distance patterns

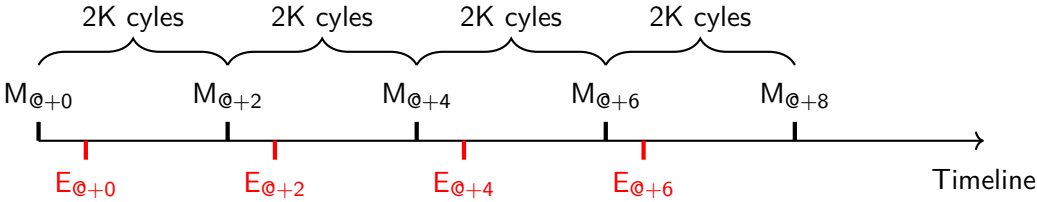
Decoupling IP that learns patterns and IP that trigger patterns (similar to T-SKID)



An IP sufficiently in advance

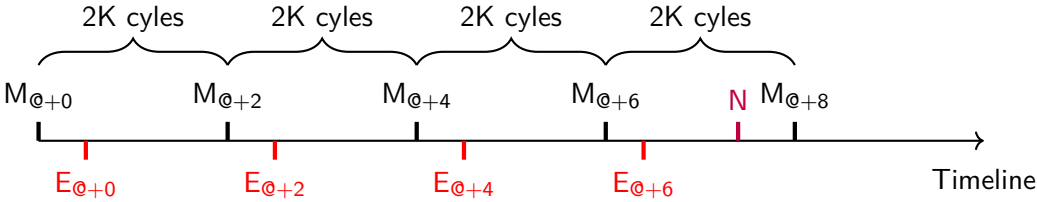
The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



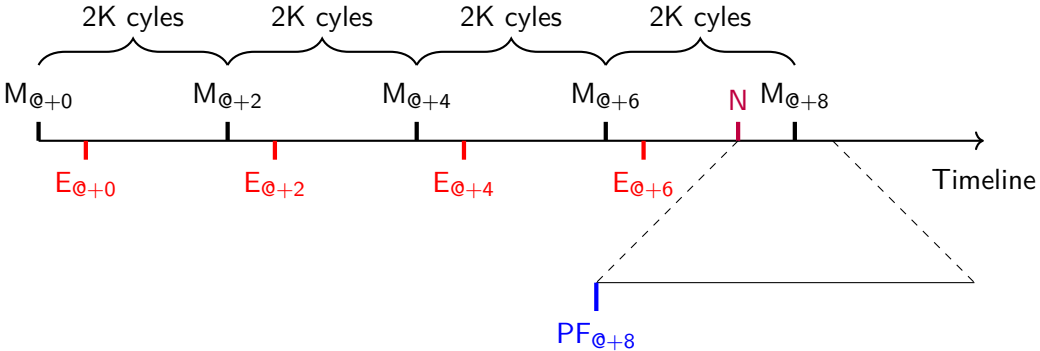
The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



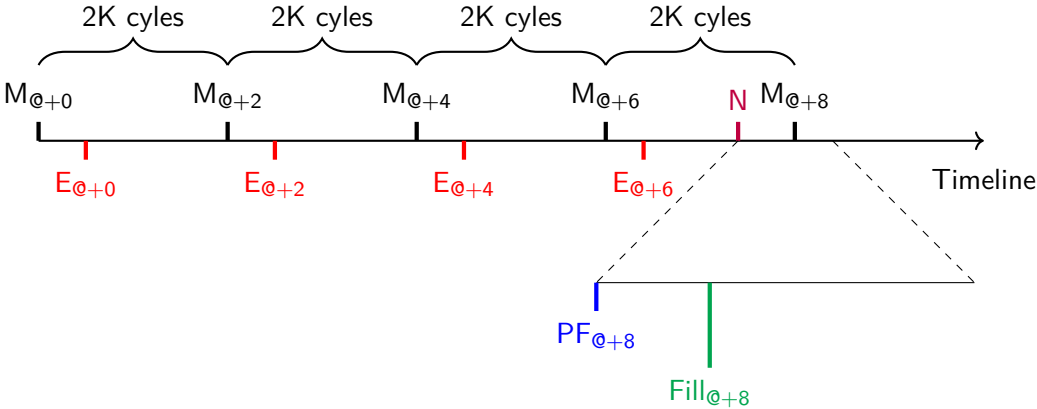
The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



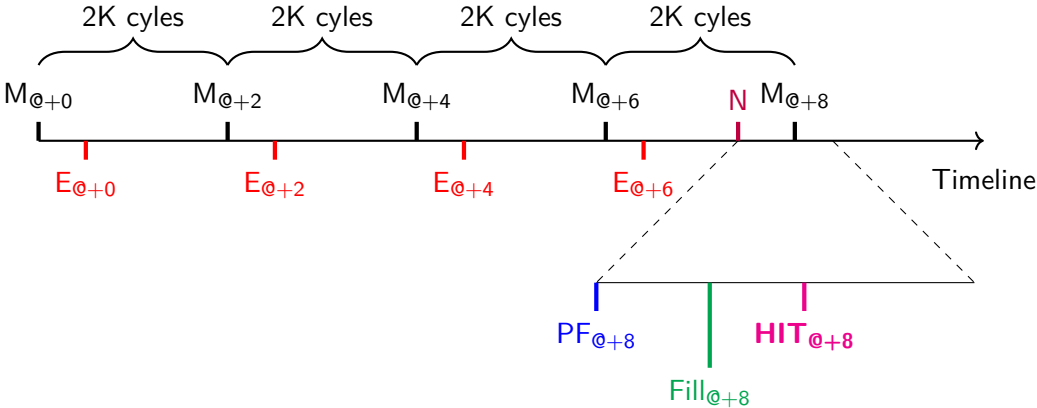
The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



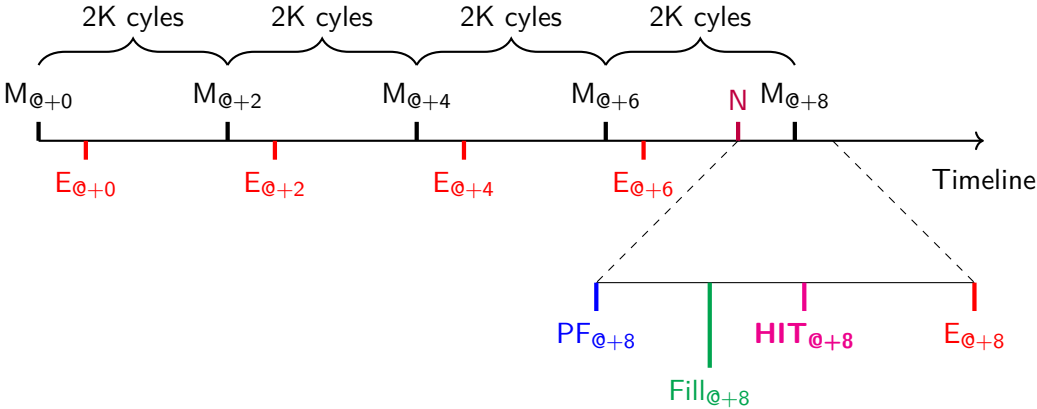
The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



The Entangling Data Prefetcher

A load triggers the prefetcher for another load →entangled pairs



The Entangling Data Prefetcher

Throttling the prefetcher

But... one IP can insert multiple requests for different IPs at the same time

The Entangling Data Prefetcher

Throttling the prefetcher

But... one IP can insert multiple requests for different IPs at the same time

It saturates L1D structures such as `MSHR`

The Entangling Data Prefetcher

Throttling the prefetcher

But... one IP can insert multiple requests for different IPs at the same time

It saturates L1D structures such as `MSHR`

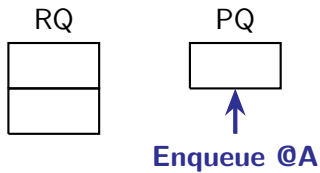
Two solutions:

1. Delay queue
2. A filter

The Entangling Data Prefetcher

A delay queue

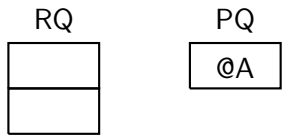
Under **low-contention**, the prefetch request enqueue a request and issue in the next few cycles



The Entangling Data Prefetcher

A delay queue

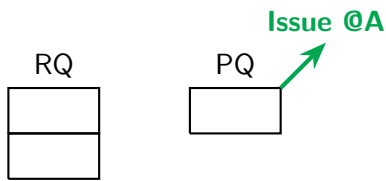
Under **low-contention**, the prefetch request enqueue a request and issue in the next few cycles



The Entangling Data Prefetcher

A delay queue

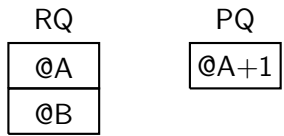
Under **low-contention**, the prefetch request enqueue a request and issue in the next few cycles



The Entangling Data Prefetcher

A delay queue

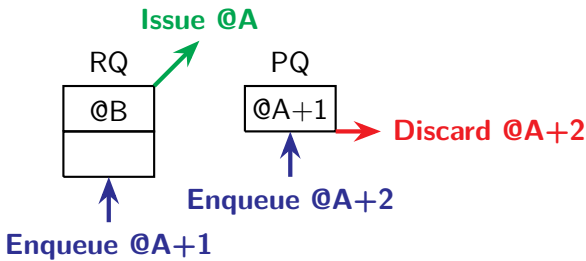
Under **high-contention**, the prefetch request stays in the PQ



The Entangling Data Prefetcher

A delay queue

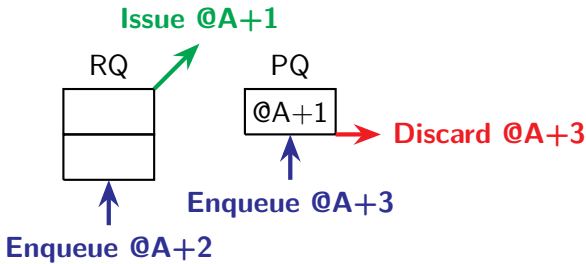
Under **high-contention**, the prefetch request stays in the PQ



The Entangling Data Prefetcher

A delay queue

Under **high-contention**, the prefetch request stays in the PQ

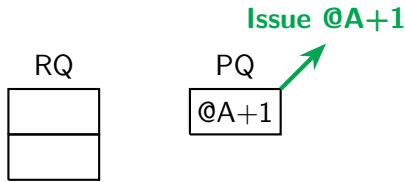


The Entangling Data Prefetcher

A delay queue

Under **high-contention**, the prefetch request stays in the PQ

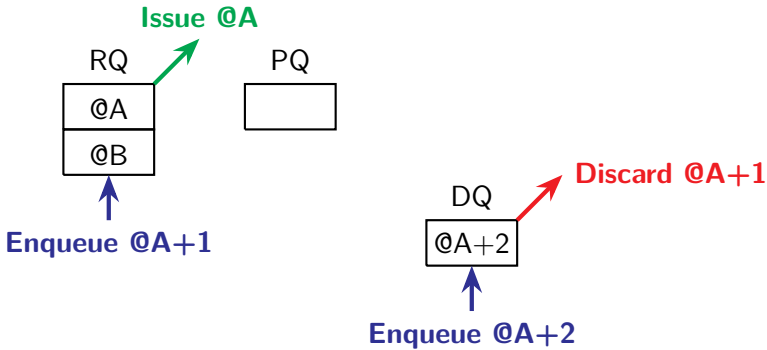
. . . and it becomes useless



The Entangling Data Prefetcher

A delay queue

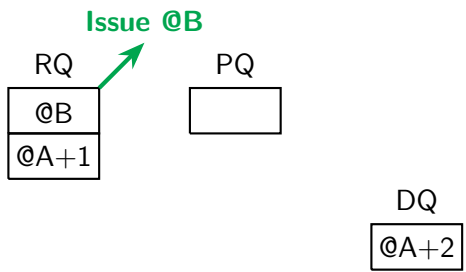
Our **delay queue** keep renewing the request to keep the useful one



The Entangling Data Prefetcher

A delay queue

Our **delay queue** keep renewing the request to keep the useful one

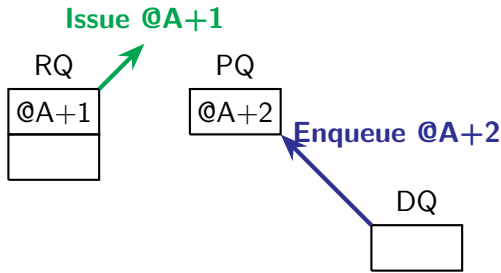


The Entangling Data Prefetcher

A delay queue

Our **delay queue** keep renewing the request to keep the useful one

. . . insert them into the PQ when they can be issued

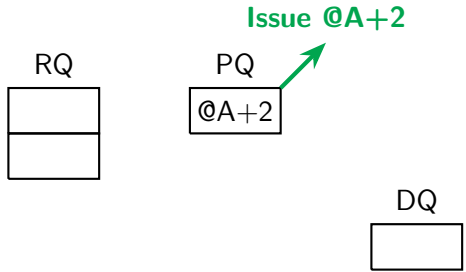


The Entangling Data Prefetcher

A delay queue

Our **delay queue** keep renewing the request to keep the useful one

. . . insert them into the PQ when they can be issued



The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

A *bloom* prefetch filter like:

The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

A *bloom* prefetch filter like:

- On every memory access (including prefetch issue) an entry is set to 1

The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

A *bloom* prefetch filter like:

- On every memory access (including prefetch issue) an entry is set to 1
- If a prefetch request accesses a filter entry with entry equal to 1 $\rightarrow$**DISCARD**

The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

A *bloom* prefetch filter like:

- On every memory access (including prefetch issue) an entry is set to 1
- If a prefetch request accesses a filter entry with entry equal to 1 →**DISCARD**
- Otherwise (entry equal to 0) →**ISSUE**

The Entangling Data Prefetcher

A prefetch filter

A filter that reject potential not useful prefetch requests

A *bloom* prefetch filter like:

- On every memory access (including prefetch issue) an entry is set to 1
- If a prefetch request accesses a filter entry with entry equal to 1 →**DISCARD**
- Otherwise (entry equal to 0) →**ISSUE**
- On a cache eviction, the entry is set to 0.

L2 & LLC

We include Pythia at L2:

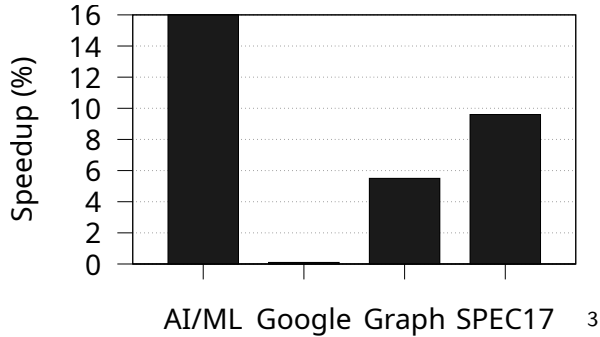
- Detects some patterns that are not detected by EDP
- It can issue prefetch request when L1D is under very high-contention

And a throttling mechanism in LLC:

- If too many cores access the LLC, it send a signal to reduce the aggressiveness of EDP
- If one core issues significant portion of non-demand/prefetch request to LLC
  $\rightarrow$disconnect EDP for that core

Evaluation

Singlecore



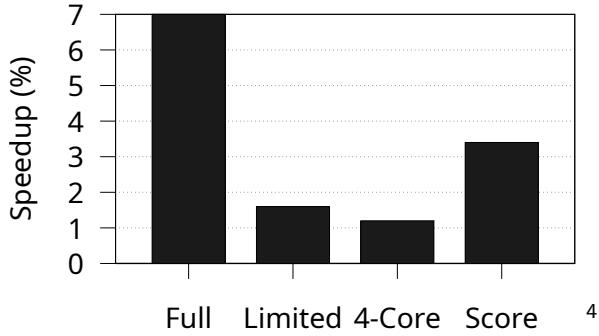
[3]

Speedup over 16% in AI/ML and over 9% in SPEC17

[3] Not including all release traces

Evaluation

All



[4]

7.1% over singlecore and 1.4% over multicore

[4] Not including all release traces

Thank you!

Any question?