Boosting Data Centers Performance with the Entangling Instruction Prefetcher

Alberto Ros

University of Murcia, Spain

Dec 2, 2021

Data centers

- Data centers serve most devices
  - Internet of things, smartphones, self-driving cars, ...
  - Energy costs expected to reach 8% of the global consumption by 2030[1]

[1] Andrae et al. *On Global Electricity Usage of Communication Technology: Trends to 2030*, Callenges 2015.

Data centers

- Data centers serve most devices
  - Internet of things, smartphones, self-driving cars, ...
  - Energy costs expected to reach 8% of the global consumption by 2030[1]
- They run increasingly complex applicatons
  - Deep software stacks

[1] Andrae et al. *On Global Electricity Usage of Communication Technology: Trends to 2030*, Callenges 2015.

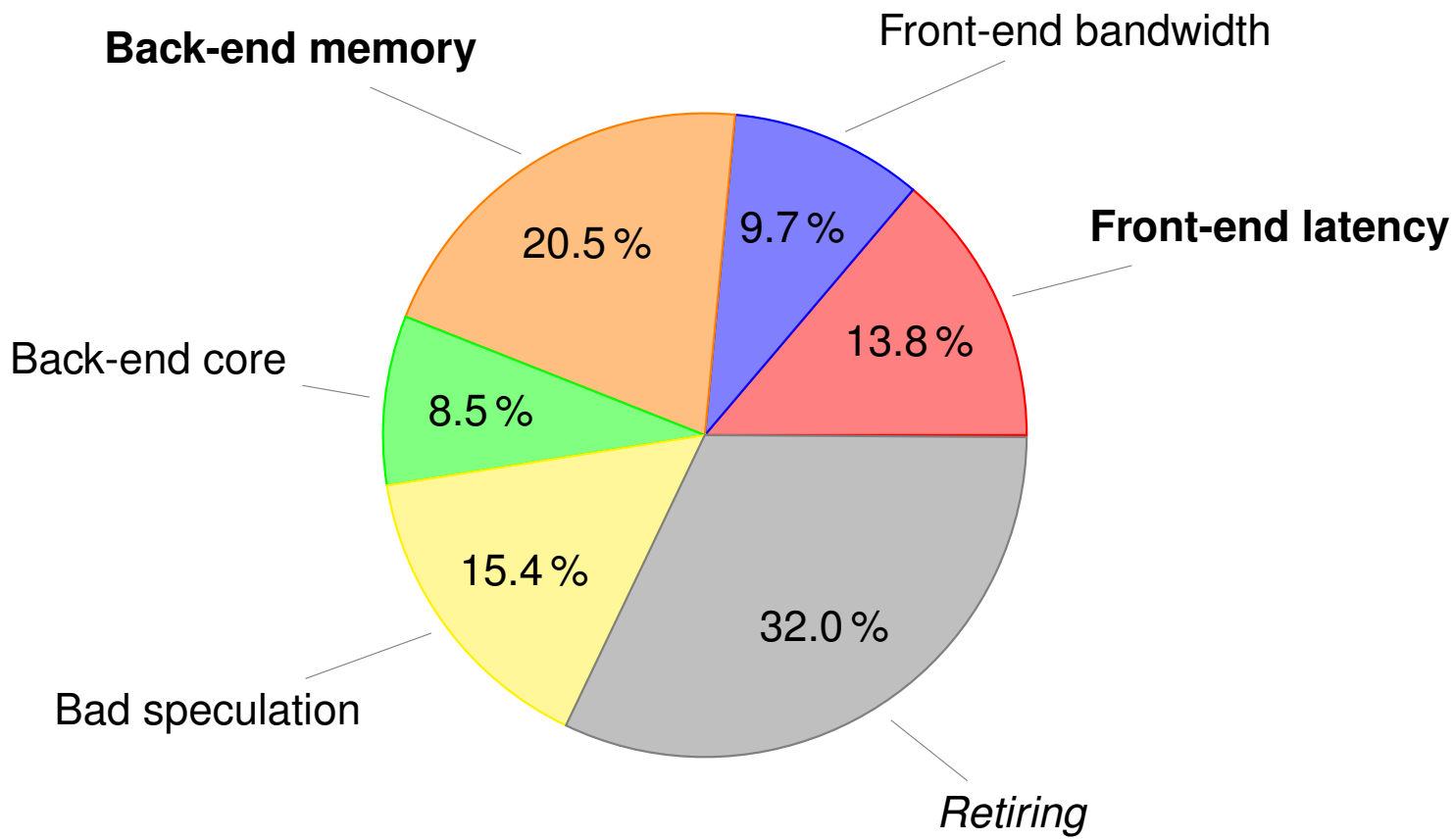
Data centers

- Data centers serve most devices
  - Internet of things, smartphones, self-driving cars, ...
  - Energy costs expected to reach 8% of the global consumption by 2030[1]
- They run increasingly complex applicatons
  - Deep software stacks
- Instruction footprint constantly growing
  - Far from fitting in small instruction caches (L1I)
  - And growing by 20% per year![2]

[1] Andrae et al. *On Global Electricity Usage of Communication Technology: Trends to 2030*, Callenges 2015.

[2] Kanev et al. *Profiling a warehouse-scale computer*, ISCA 2015.

DATA CENTERS BOTTLENECKS[3]



[3] Ayers et al. *AsmDB: Understanding and Mitigating Front-End Stalls in Warehouse-Scale Computers*, ISCA 2019.

Data centers bottlenecks

Front-end latency (13.8%)

- Dominated by instruction cache (L1I) misses
  - Hiting in the second level cache (L2) or last level cache (LLC)
- Latency more important than bandwidth
- Critical as processors need to keep the pipeline full

Back-end memory (20.5%)

- Due to data cache (L1D) misses
  - Many of them reaching main memory
- Cause significant stalls and late detection of bad speculation (15.4%)

PREFETCHING TO THE RESCUE

- High-performance processors would need a very large memory with a low access latency
- This is not possible due to technology limitations
- Computer architects already came with a solution to this problem: prefetching

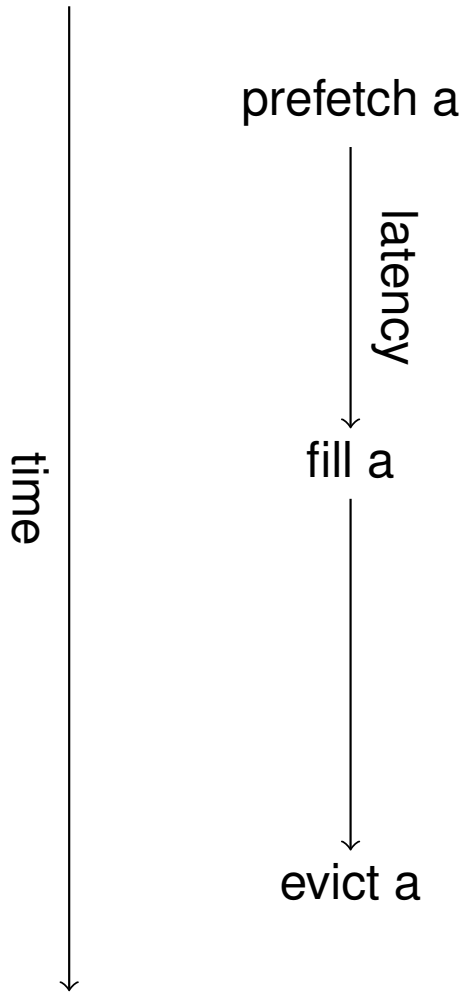
Prefetching to the rescue

- High-performance processors would need a very large memory with a low access latency
- This is not possible due to technology limitations
- Computer architects already came with a solution to this problem: prefetching

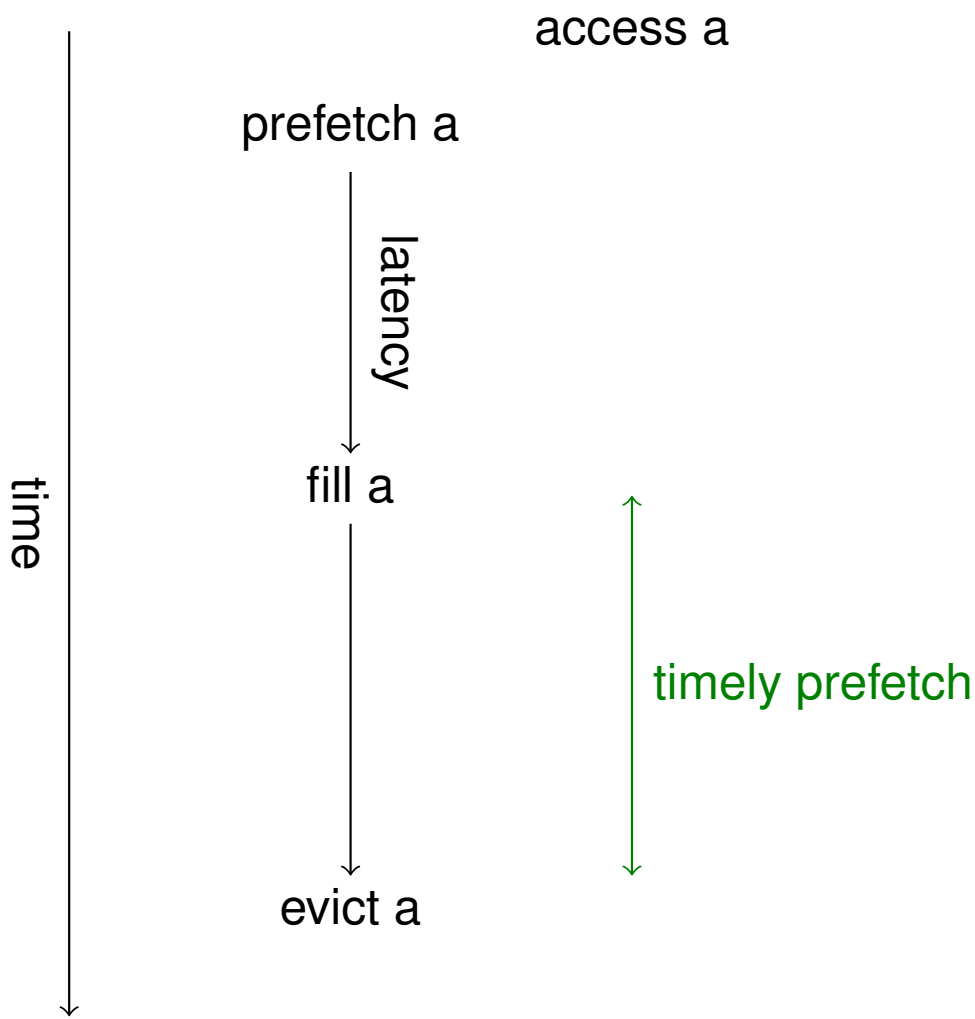
Prefetching

Predict which memory addresses will be accessed by the processor and fetch them before the processor requests them

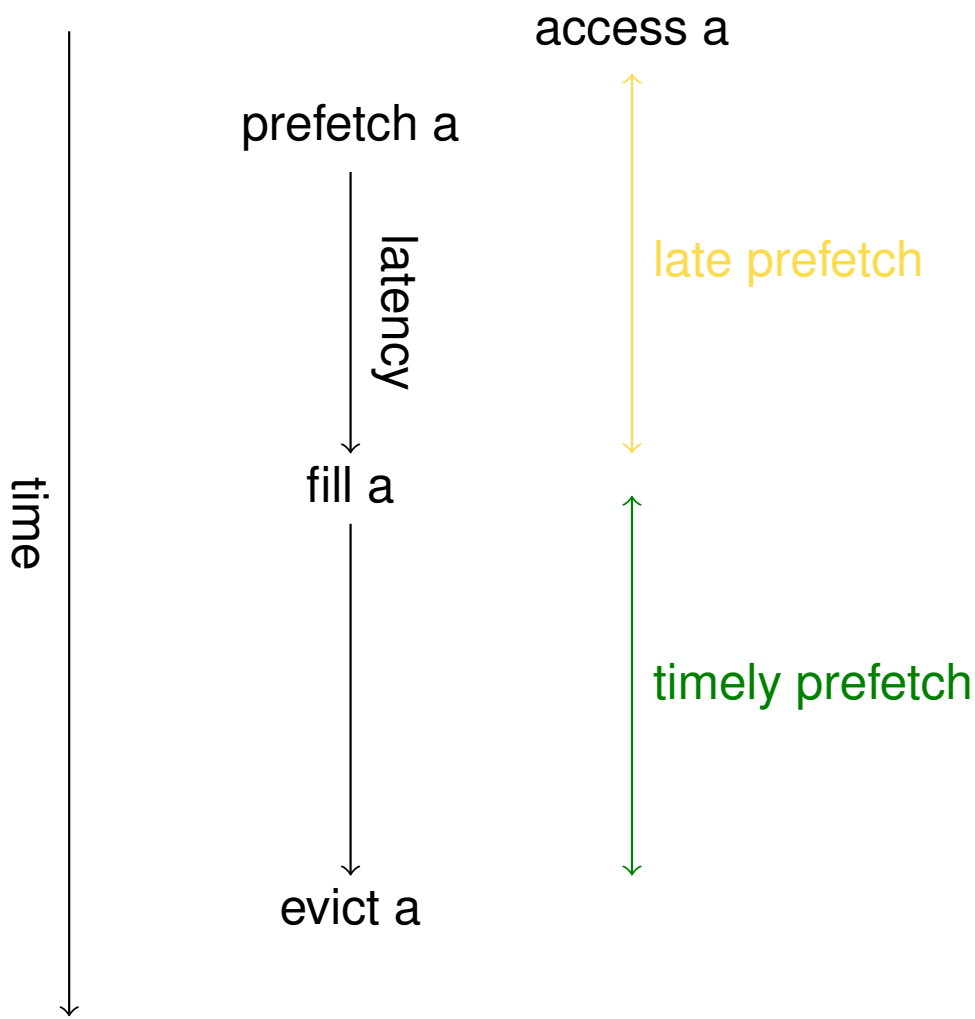
TIMELINESS IS THE KEY PROPERTY



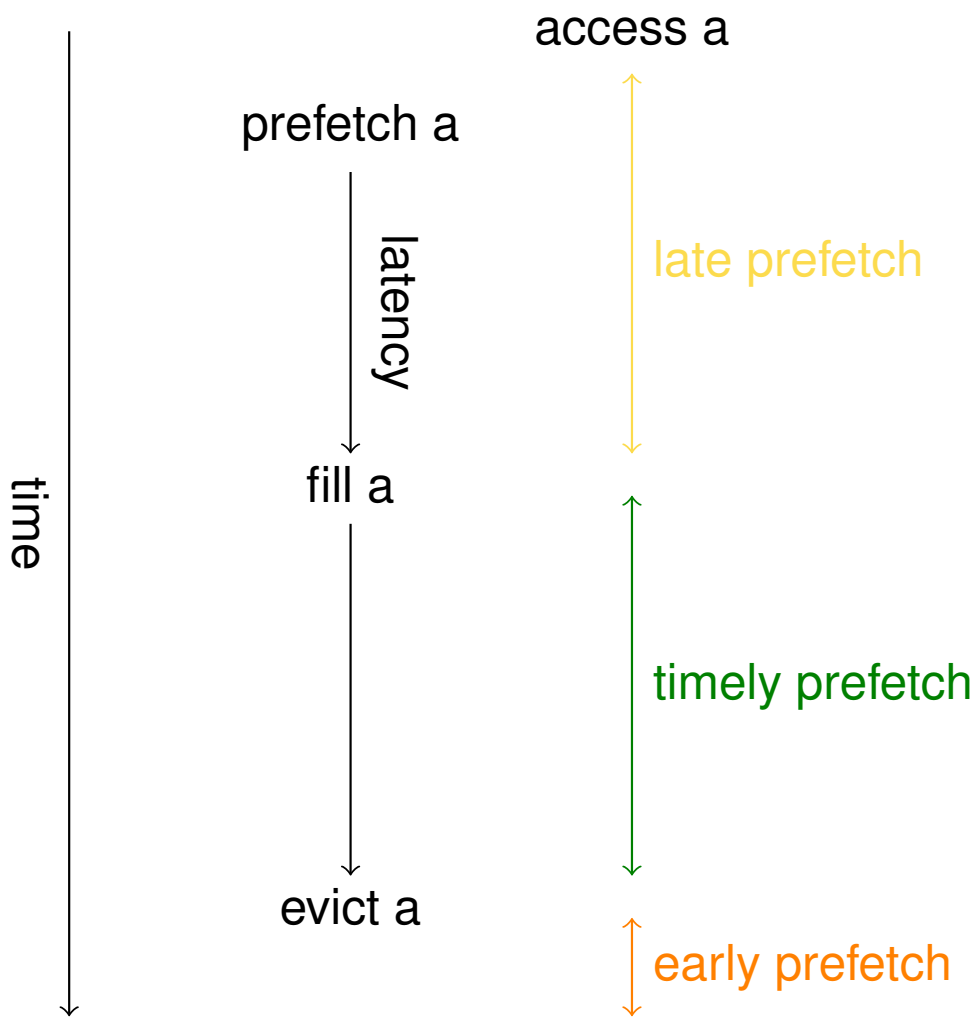
TIMELINESS IS THE KEY PROPERTY



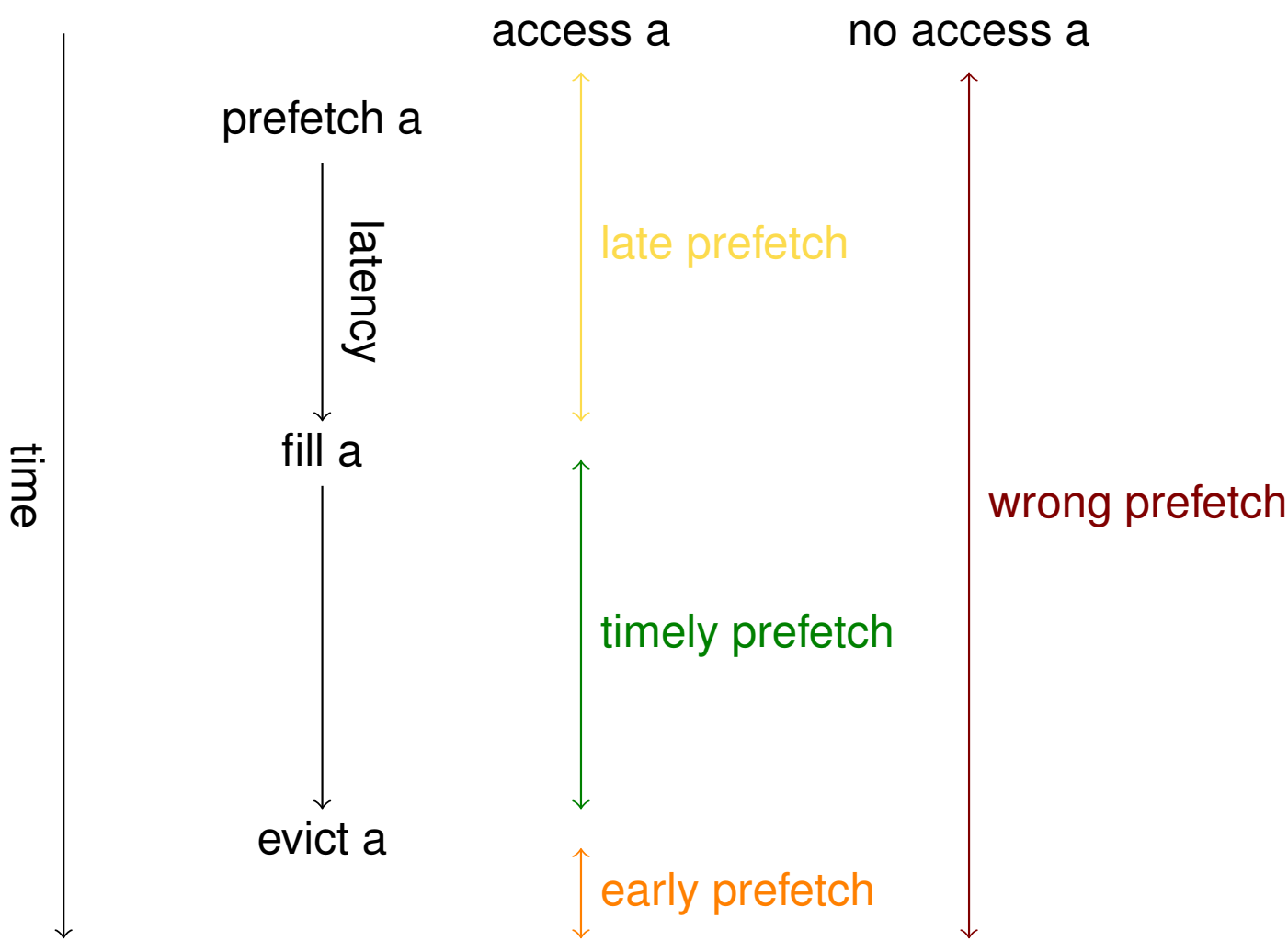
TIMELINESS IS THE KEY PROPERTY



TIMELINESS IS THE KEY PROPERTY



TIMELINESS IS THE KEY PROPERTY



TIMELINESS IS THE KEY PROPERTY

- Two prefetchers with a strong focus on timeliness

TIMELINESS IS THE KEY PROPERTY

- Two prefetchers with a strong focus on timeliness
  1. The BL∪E Data Prefetcher[4]
     - An LLC prefetcher
     - Winner of the 1st ML-based Data Prefetching Competition
     - Organized by Google
     - With a non-ML solution!

[4] Ros, *BL∪E: A Timely, IP-based Data Prefetcher*, ML-DPC-1 2021

TIMELINESS IS THE KEY PROPERTY

- Two prefetchers with a strong focus on timeliness
  1. The BL∪E Data Prefetcher[4]
     - An LLC prefetcher
     - Winner of the 1st ML-based Data Prefetching Competition
     - Organized by Google
     - With a non-ML solution!
  2. The ENTANGLING Intruction Prefetcher[5]
     - An L1I prefetcher
     - Winner of the 1st Instruction Prefetching Championship
     - Organized by Intel
     - Follow up paper published at ISCA'21

[4] Ros, *BL∪E: A Timely, IP-based Data Prefetcher*, ML-DPC-1 2021

[5] Ros and Jimborean, *The Entangling Instruction Prefetcher*, IPC-1 2020

The Entangling Instruction Prefetcher

- Server and cloud apps getting larger, far from fitting in L1I
  - $\Rightarrow$ stalls processor front-end, performance degradation

The Entangling Instruction Prefetcher

- Server and cloud apps getting larger, far from fitting in L1I
  - $\Rightarrow$ stalls processor front-end, performance degradation

- Prefetching instructions is fundamental for performance
  - Even when a decoupled front-end is implemented

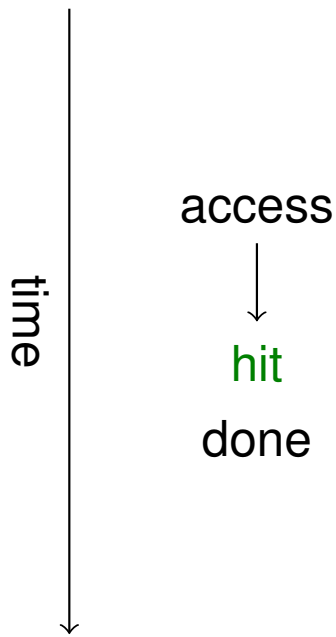
The Entangling Instruction Prefetcher

- Server and cloud apps getting larger, far from fitting in L1I
  - $\Rightarrow$ stalls processor front-end, performance degradation
- Prefetching instructions is fundamental for performance
  - Even when a decoupled front-end is implemented
- Solution: The Entangling instruction prefetcher[6]
  - Entangling: adaptive correlation based on latency
  - A cost-effective prefetcher
  - Prefetcher code is available[7]

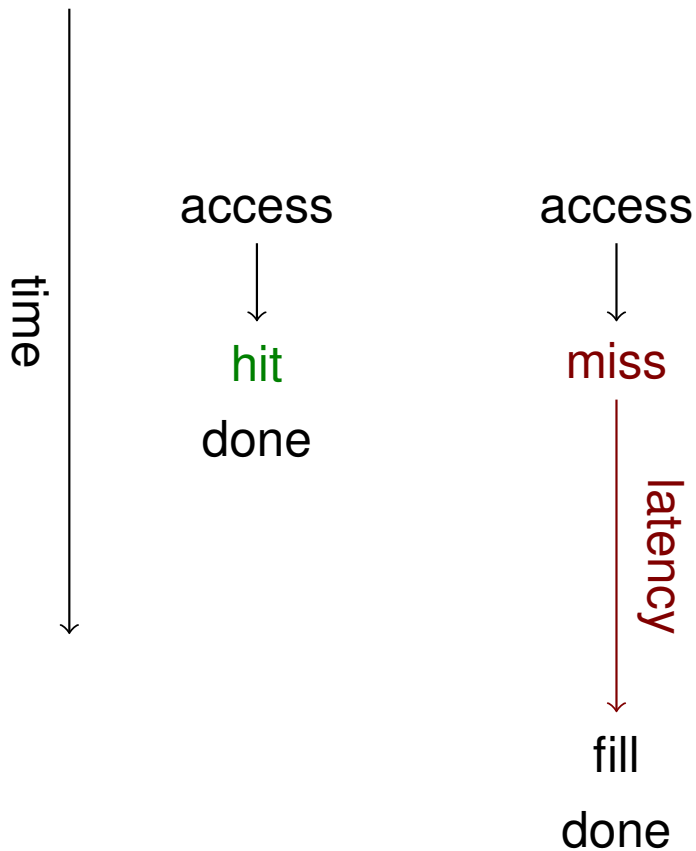
[6] Ros and Jimborean, *A Cost-Effective Entangling Prefetcher for Instructions*, ISCA 2021

[7] `https://github.com/alberto-ros/EntanglingInstructionPrefetcher`

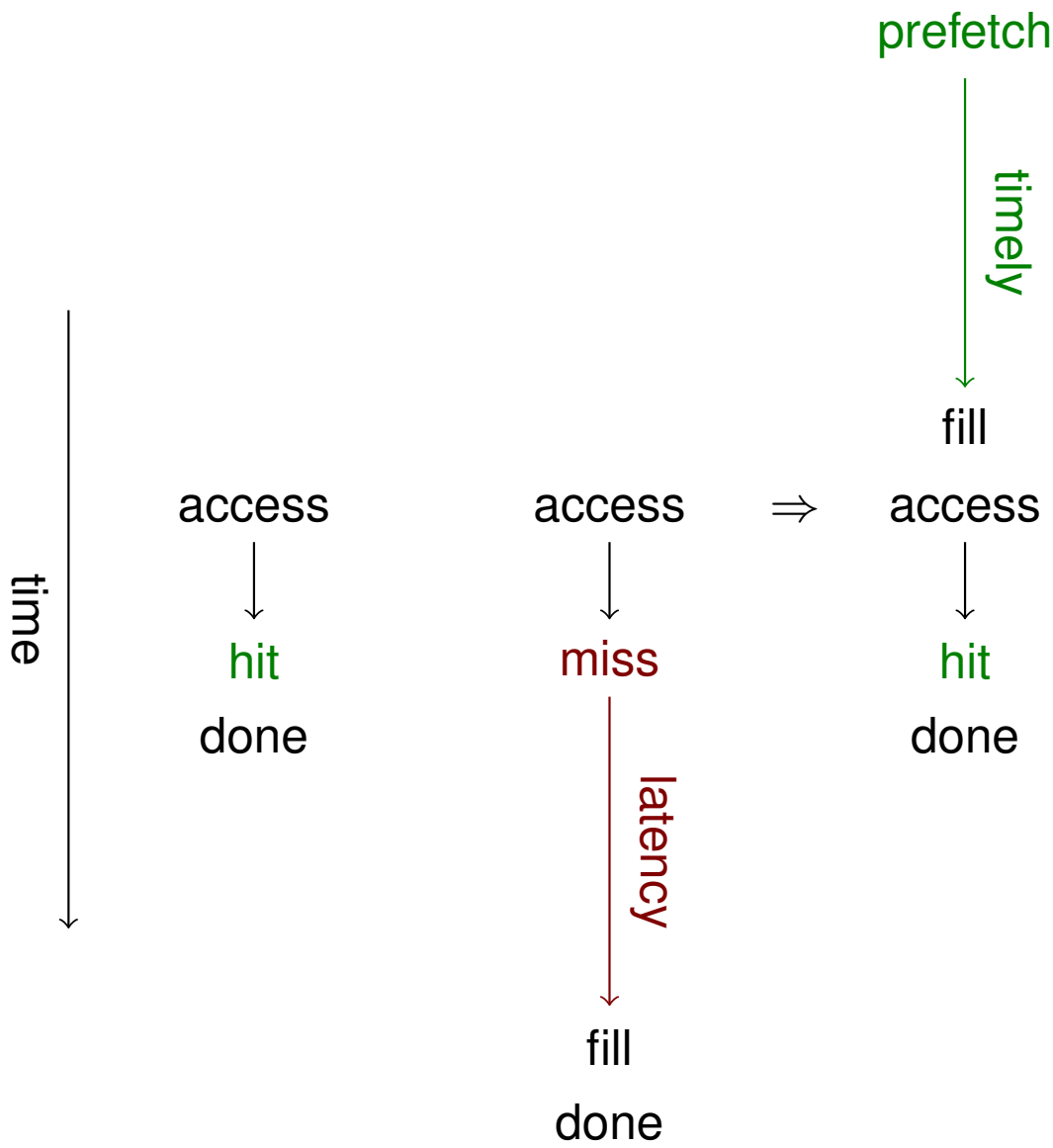
MOTIVATION: TIMELINESS



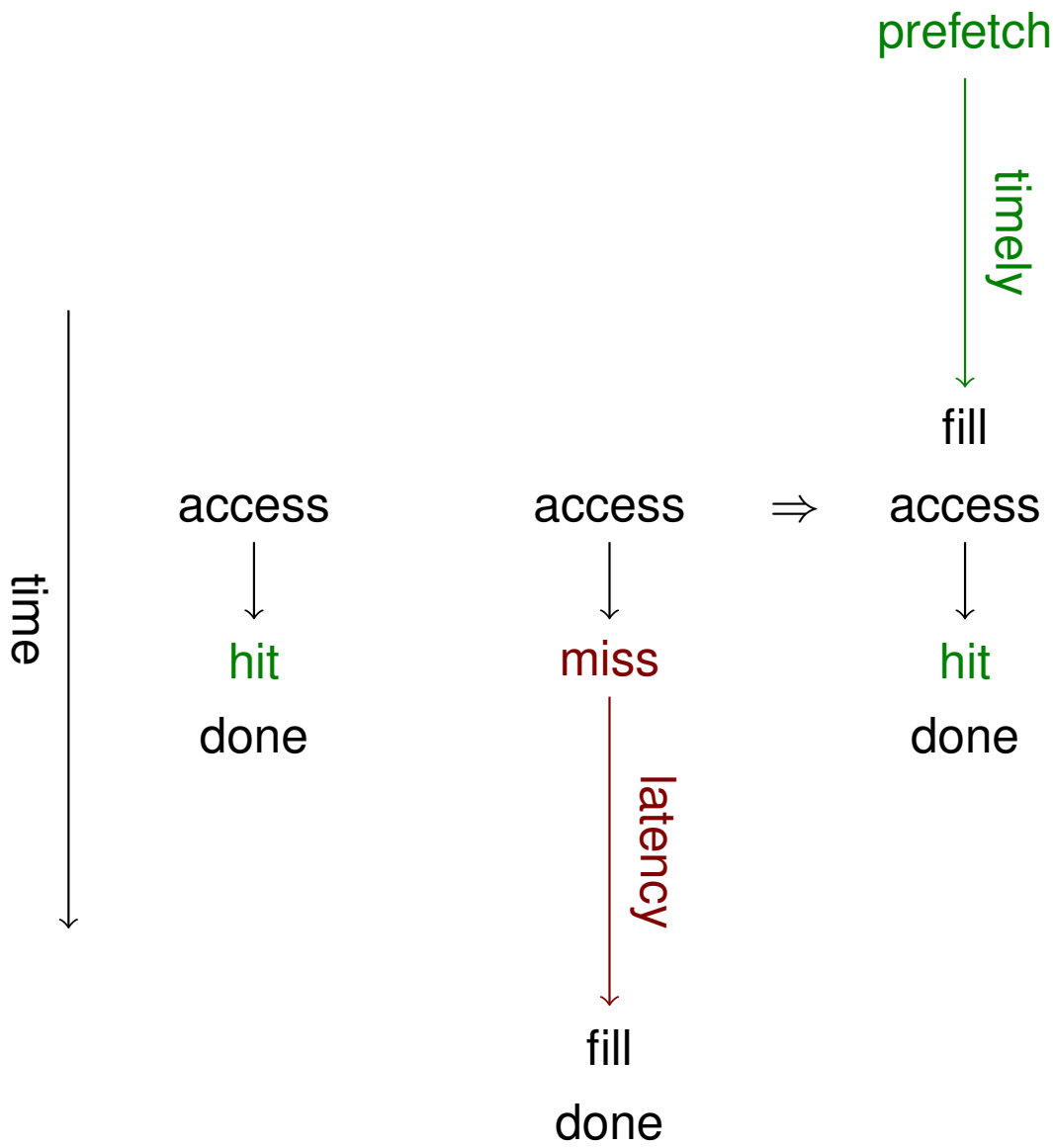
MOTIVATION: TIMELINESS



MOTIVATION: TIMELINESS



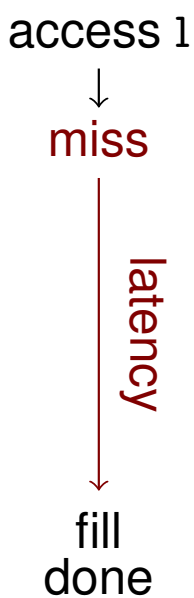
MOTIVATION: TIMELINESS



Timely prefetches for all misses:
Coverage 100%

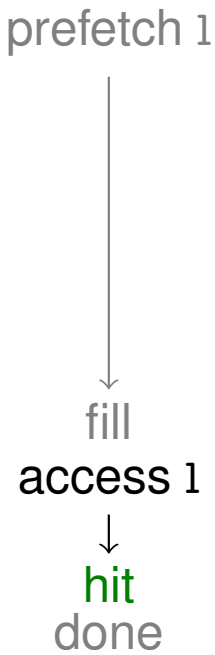
And only for misses:
Accuracy 100%

CONCEPT OF ENTANGLED ACCESSES

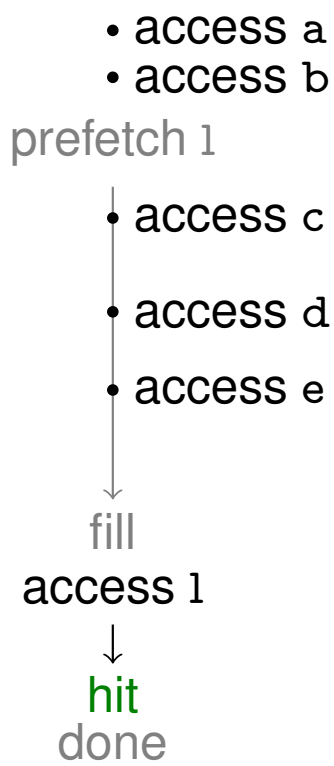


CONCEPT OF ENTANGLED ACCESSES

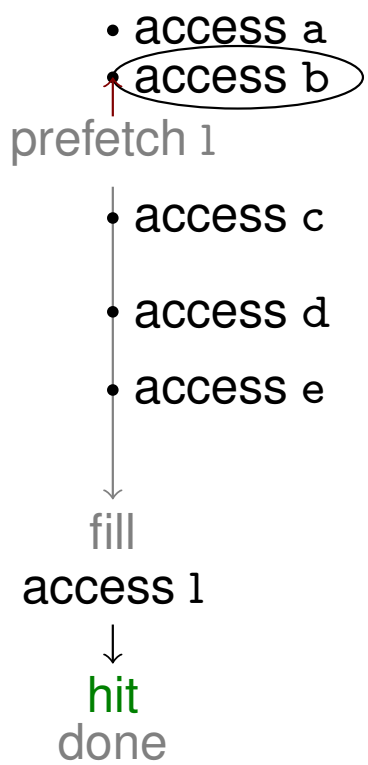
CONCEPT OF ENTANGLED ACCESSES



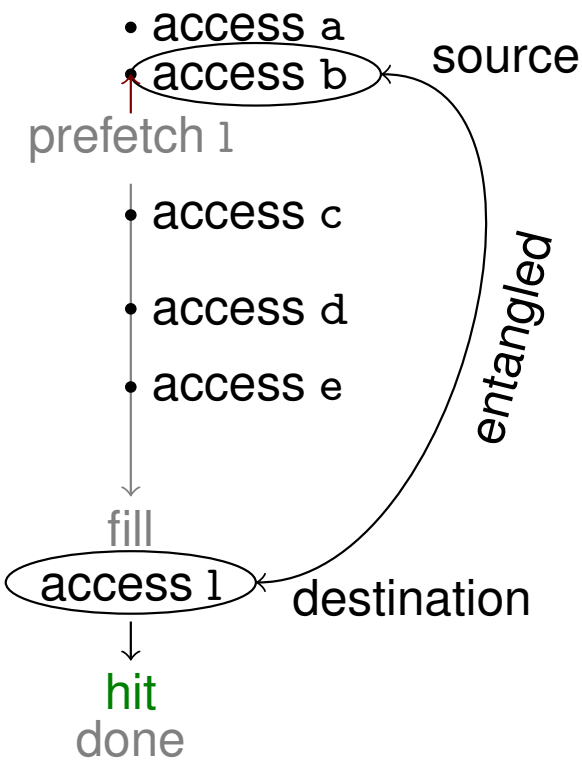
CONCEPT OF ENTANGLED ACCESSES



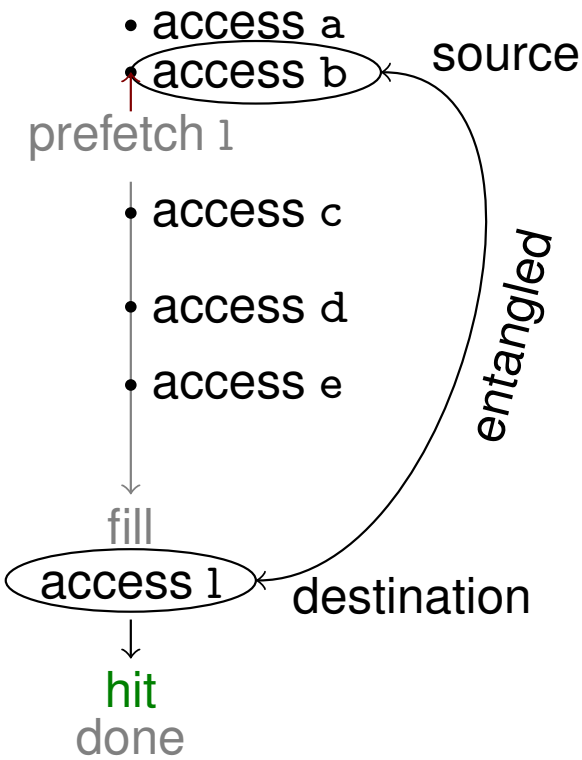
CONCEPT OF ENTANGLED ACCESSES

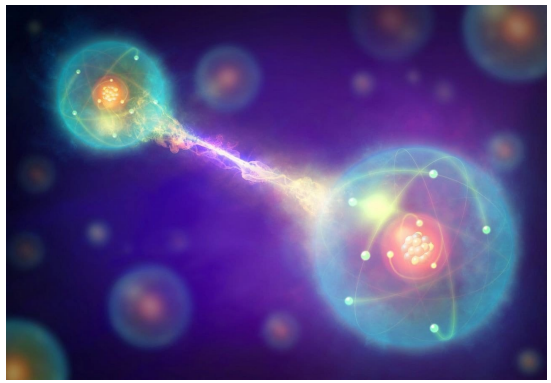


CONCEPT OF ENTANGLED ACCESSES



CONCEPT OF ENTANGLED ACCESSES

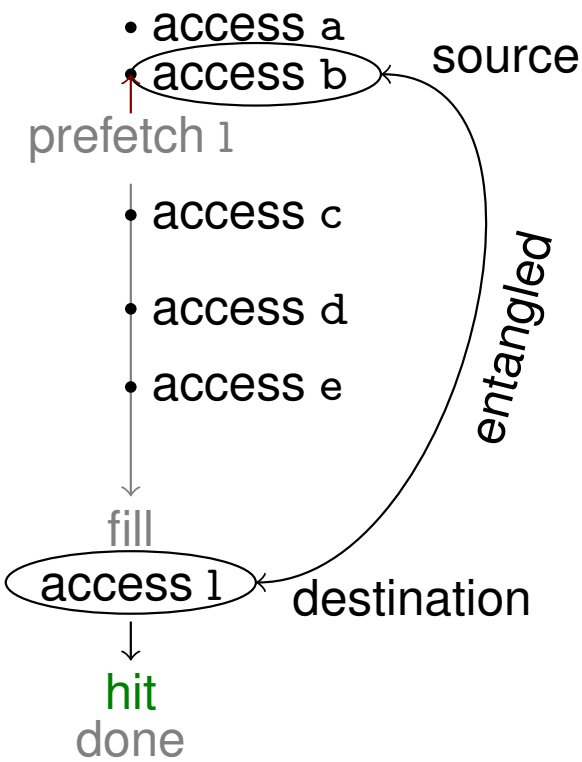


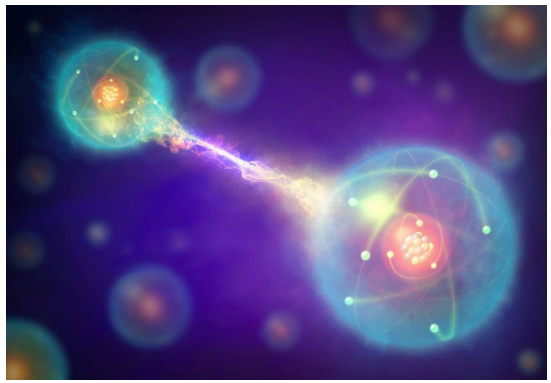


Quantum entanglement

(Image: © MARK GARLICK/SCIENCE PHOTO LIBRARY/Getty)

Concept of entangled accesses



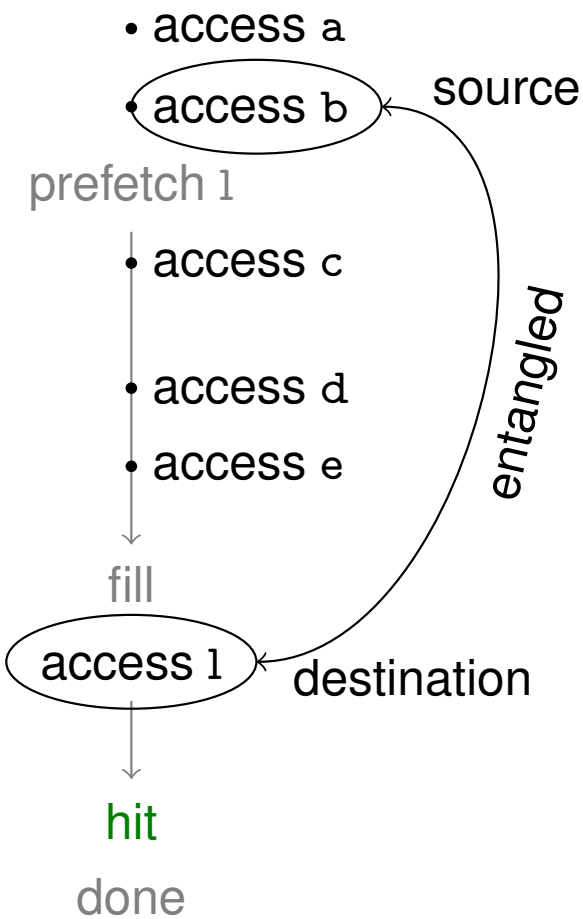


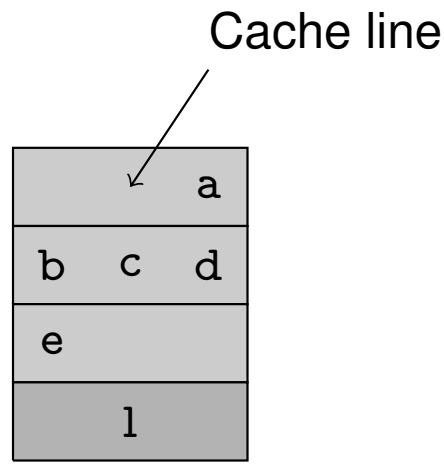
Quantum entanglement

(Image: © MARK GARLICK/SCIENCE PHOTO LIBRARY/Getty)

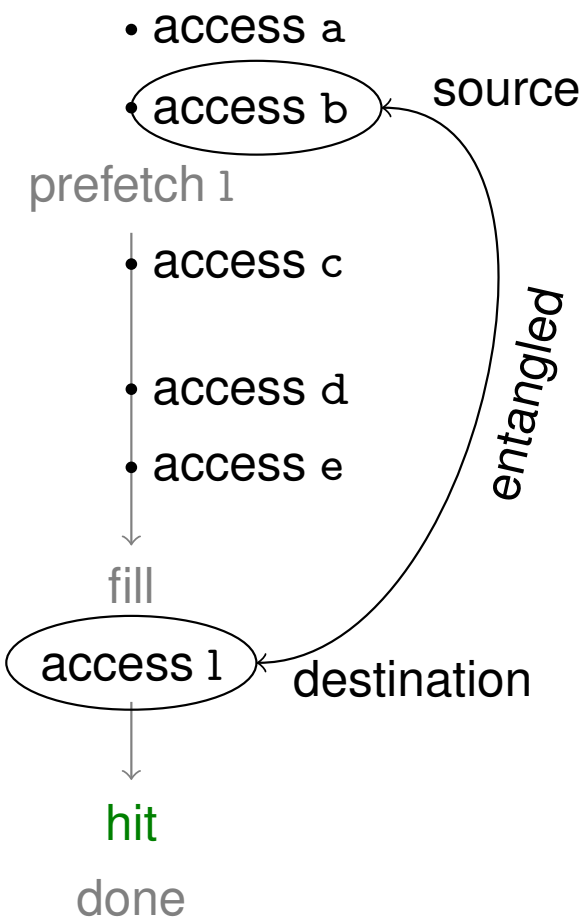
The Entangling Prefetcher for Instructions

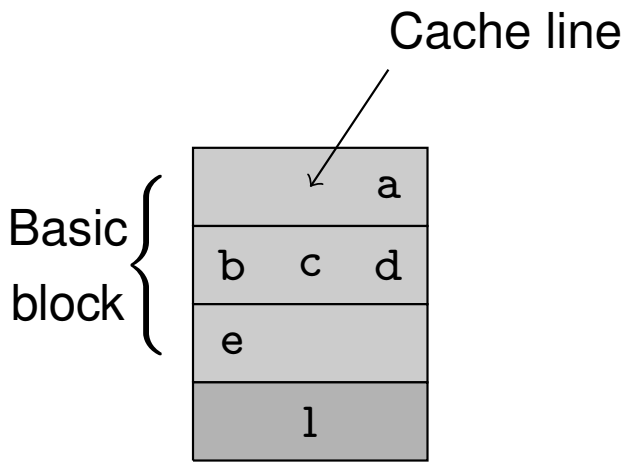
ENTANGLING CACHE LINES HEAD OF BASIC BLOCKS



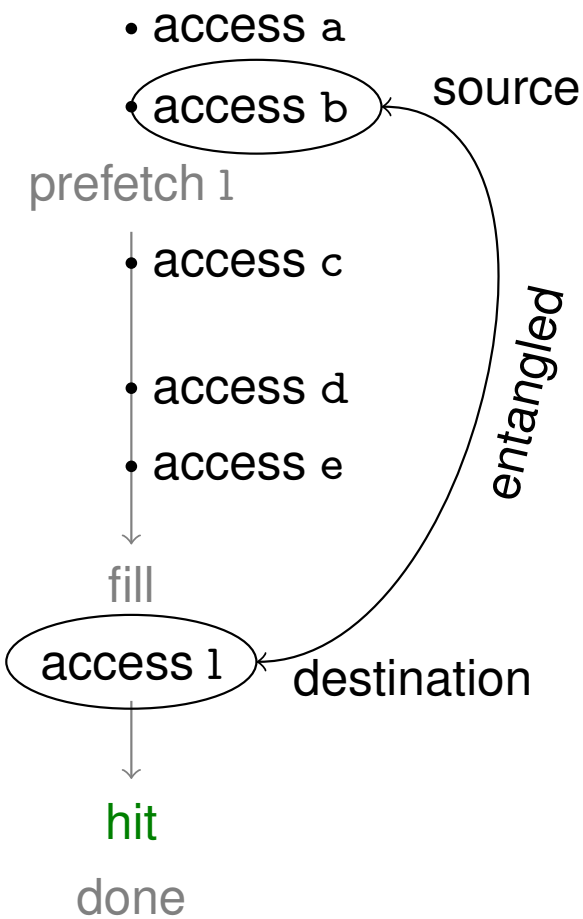


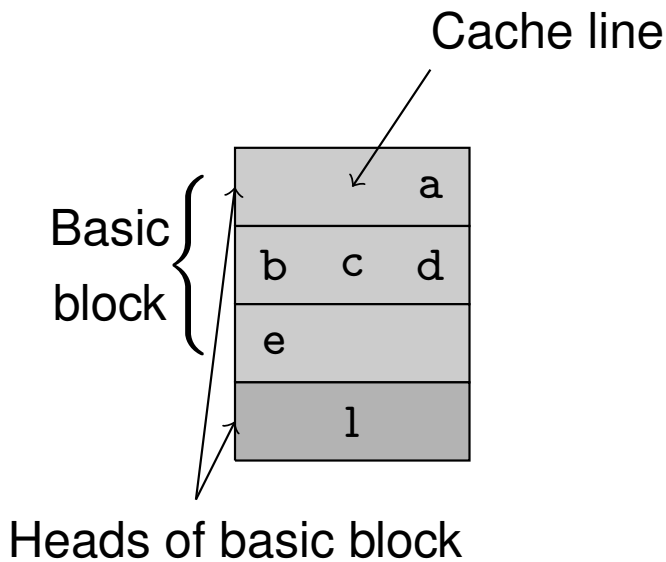
ENTANGLING CACHE LINES HEAD OF BASIC BLOCKS



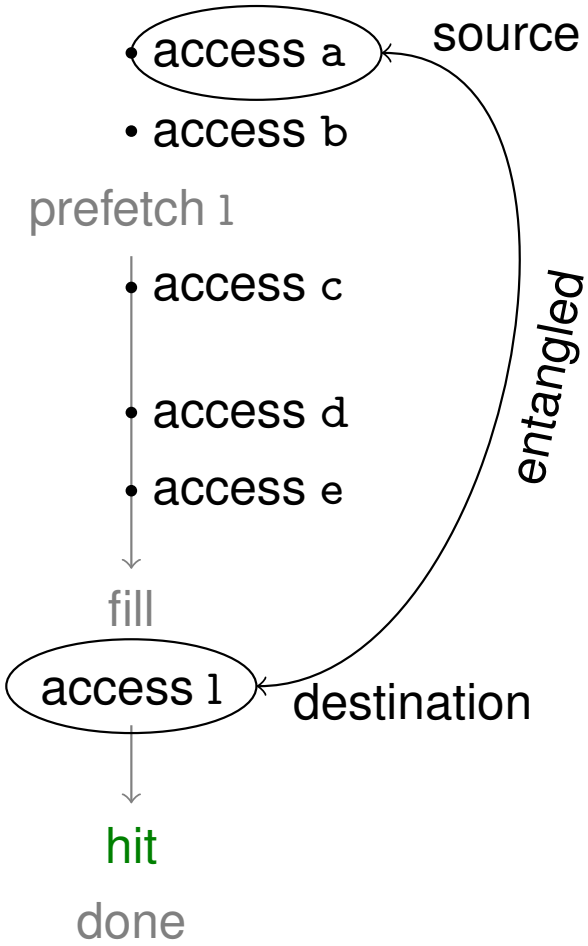


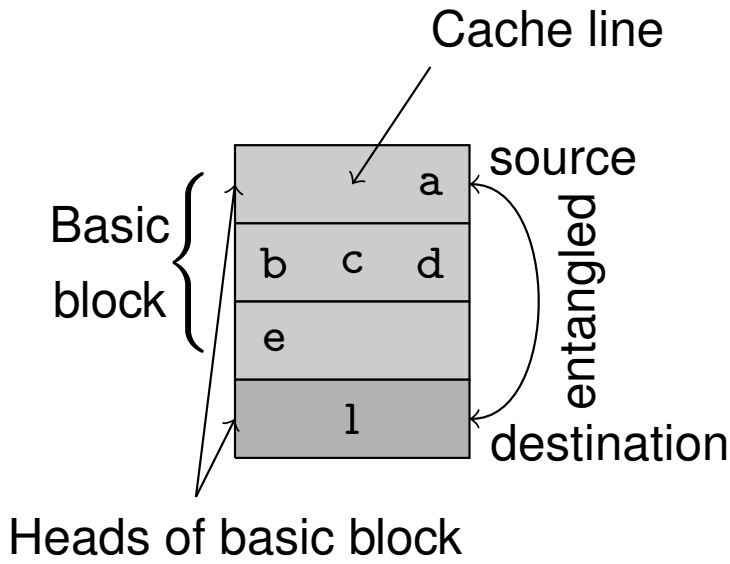
ENTANGLING CACHE LINES HEAD OF BASIC BLOCKS



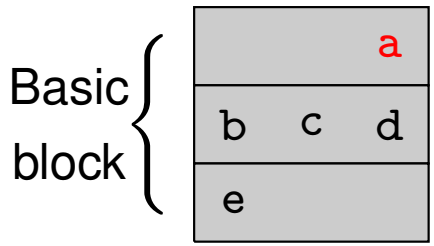


ENTANGLING CACHE LINES HEAD OF BASIC BLOCKS

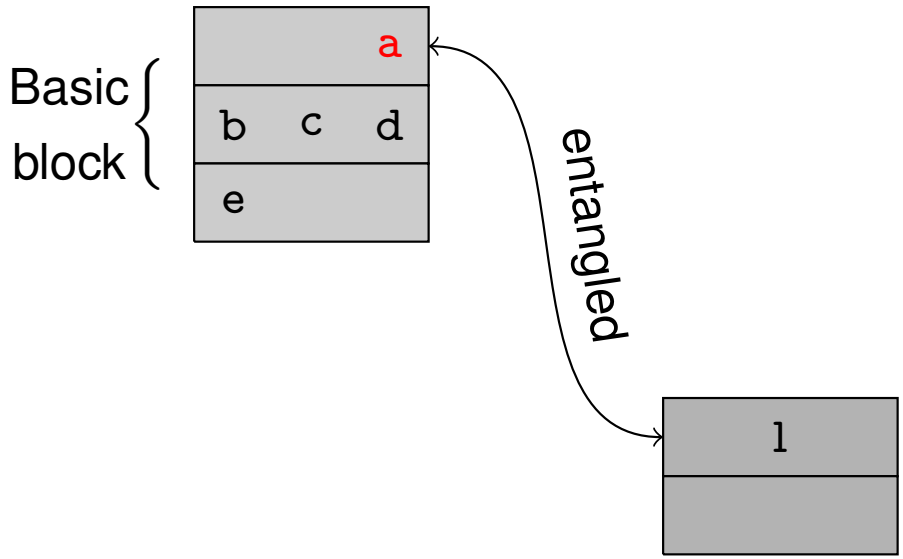




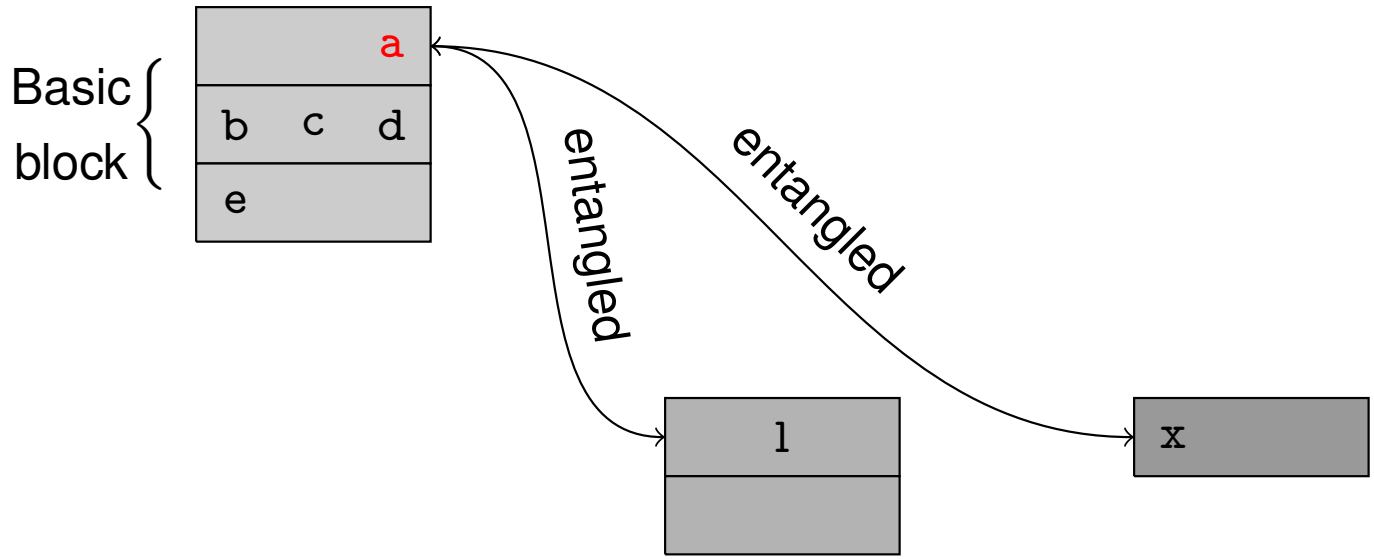
WHAT TO PREFETCH ON AN ACCESS TO a?



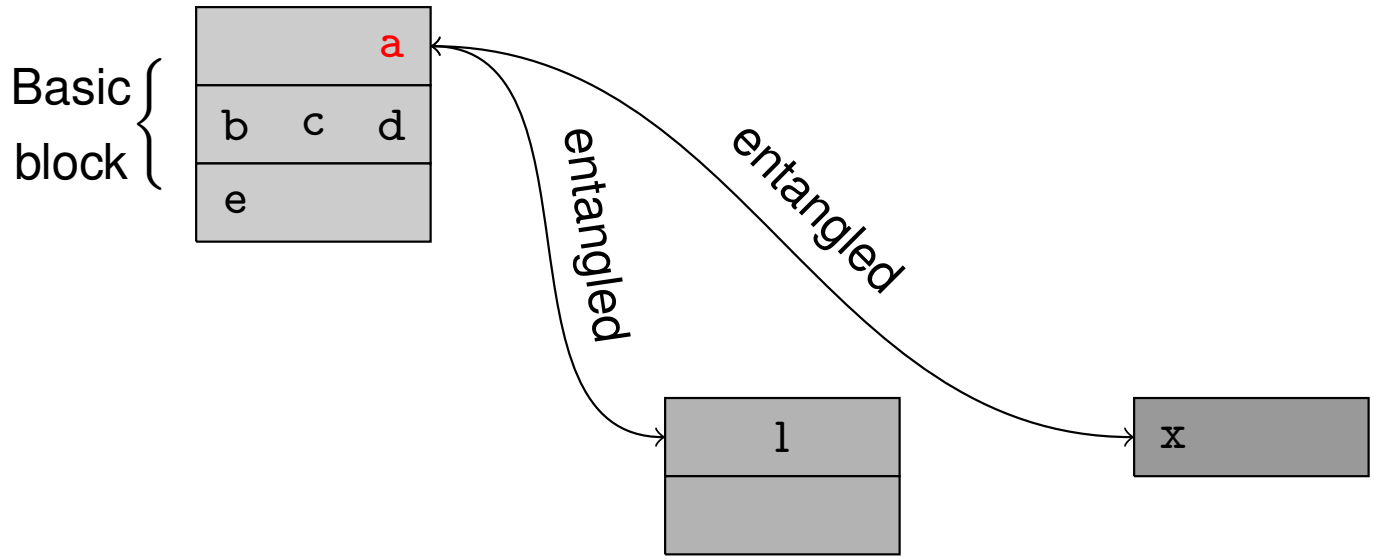
WHAT TO PREFETCH ON AN ACCESS TO a?



WHAT TO PREFETCH ON AN ACCESS TO a?

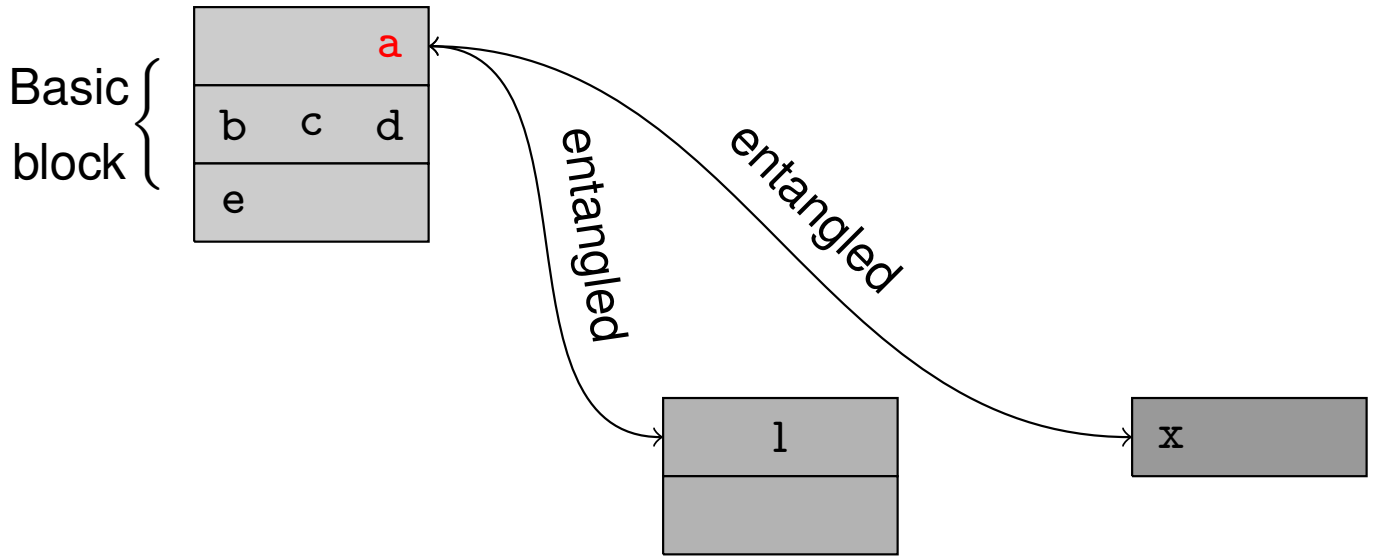


WHAT TO PREFETCH ON AN ACCESS TO a?



- Too much? (Max entangled = 6, Max BB size = 64)

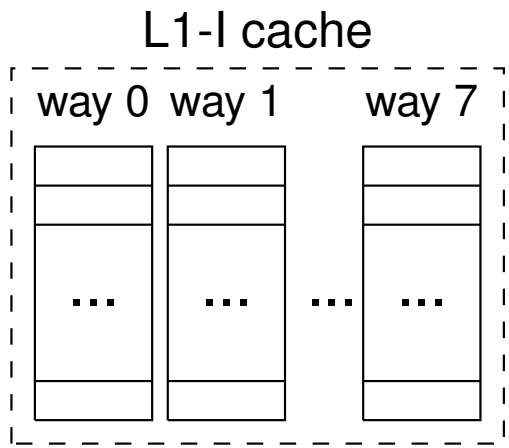
WHAT TO PREFETCH ON AN ACCESS TO a?



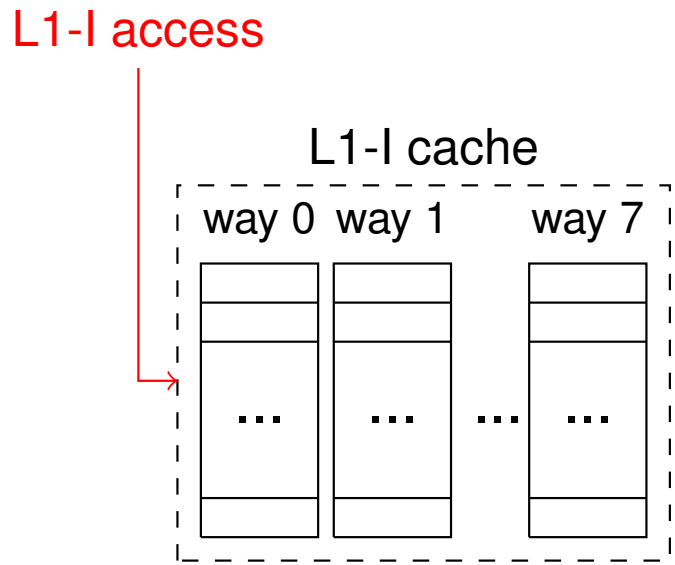
- Too much? (Max entangled = 6, Max BB size = 64)
  - Most of the time no prefetches are issued (no head of basic block)
  - Average number of prefetches per access to head of basic block ranging from ≈9 to ≈17
  - Remember: Front-end latency more imporant than bandwidth[8]

[8] Kanev et al. *Profiling a warehouse-scale computer*, ISCA 2015.

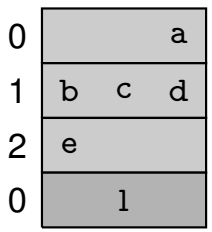
Design of the Entangling Prefetcher

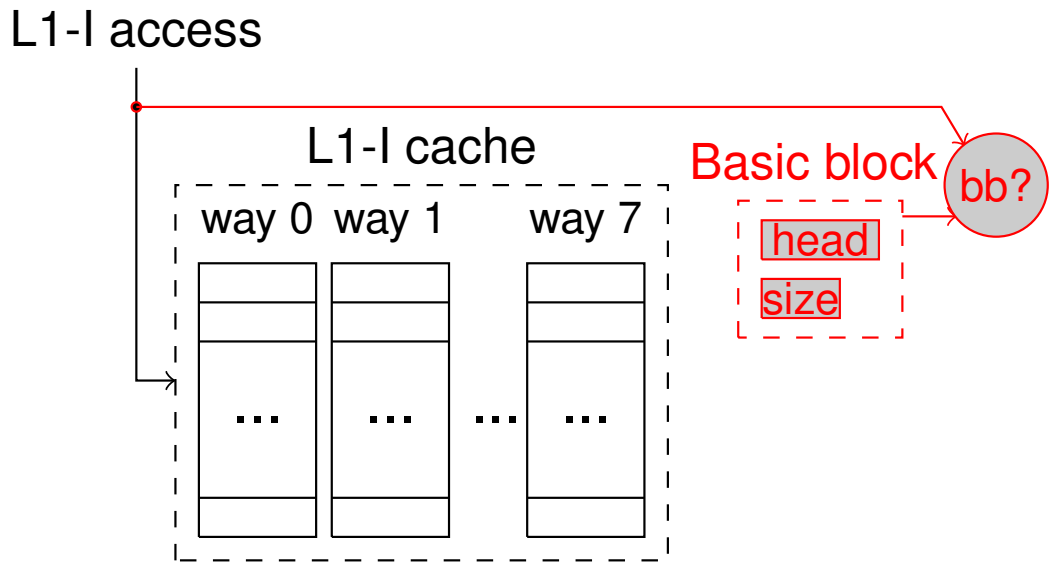


Design of the Entangling Prefetcher

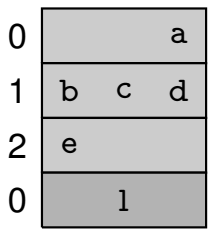


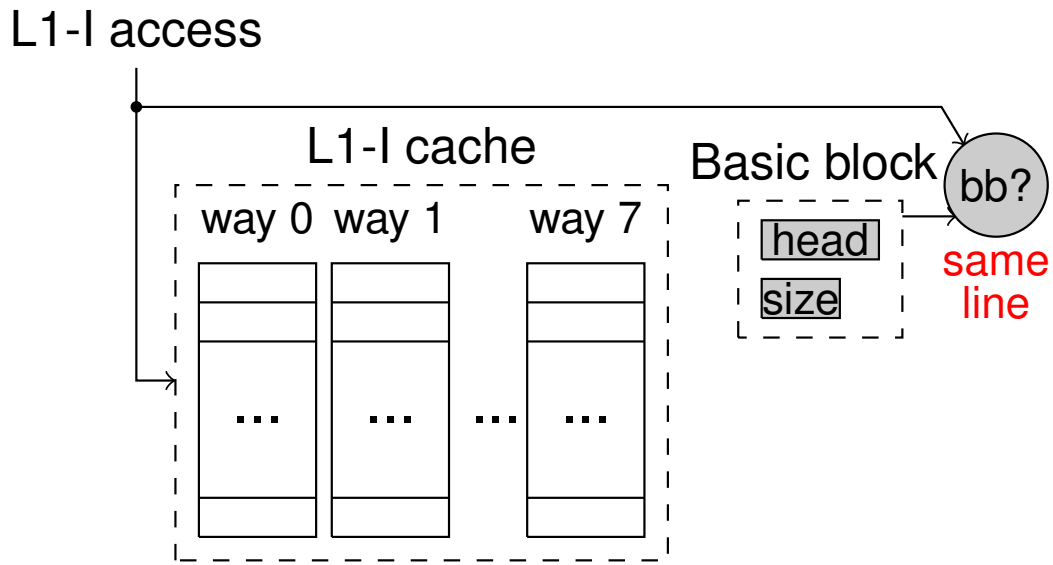
Design of the Entangling Prefetcher - Finding basic blocks



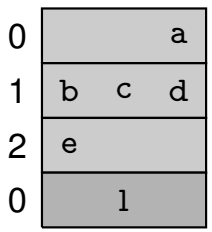


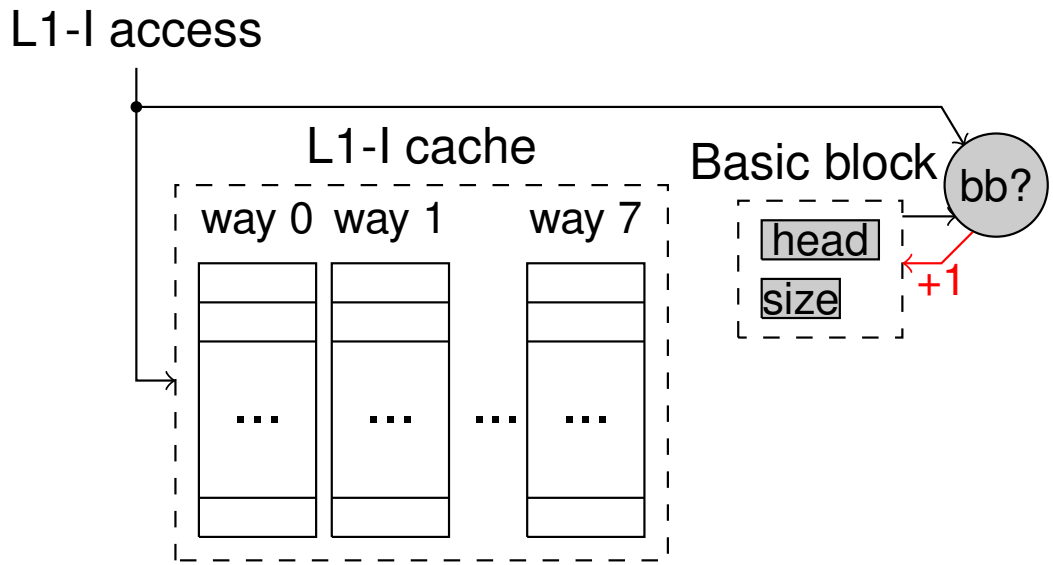
Design of the Entangling Prefetcher - Finding basic blocks



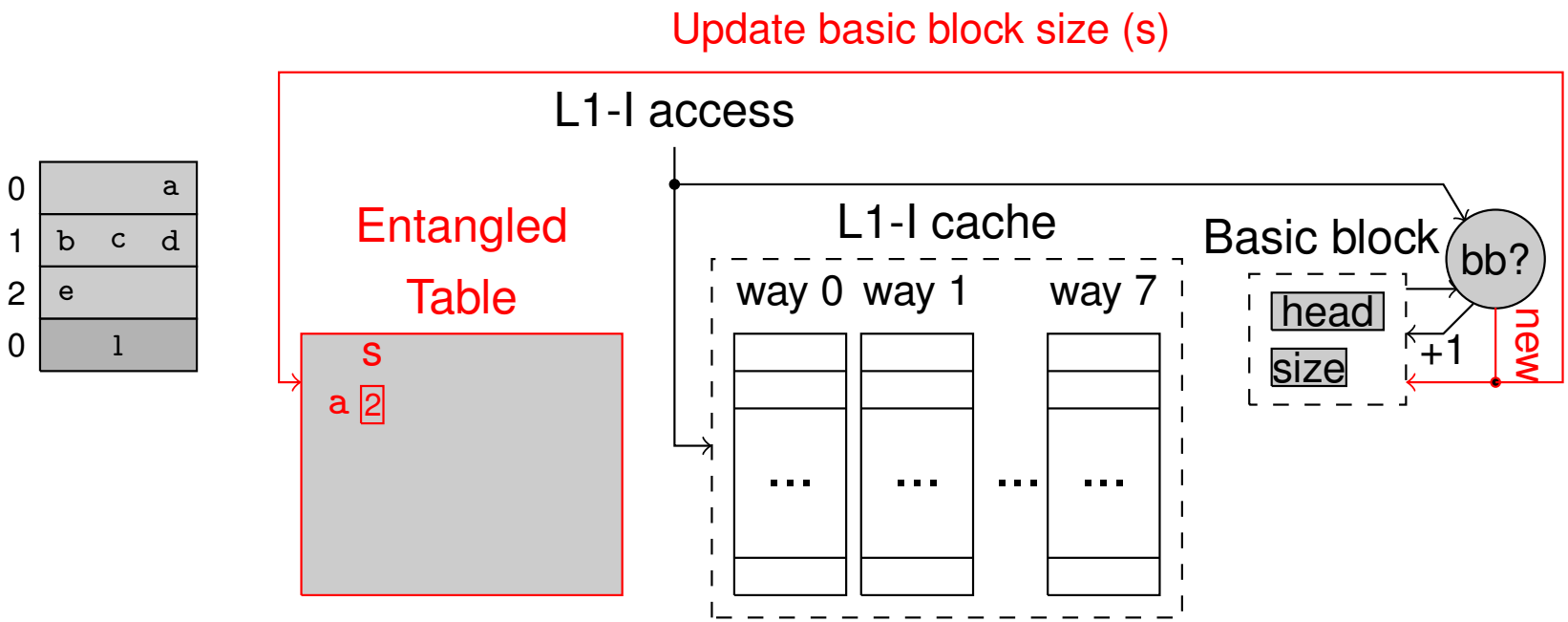


Design of the Entangling Prefetcher - Finding basic blocks

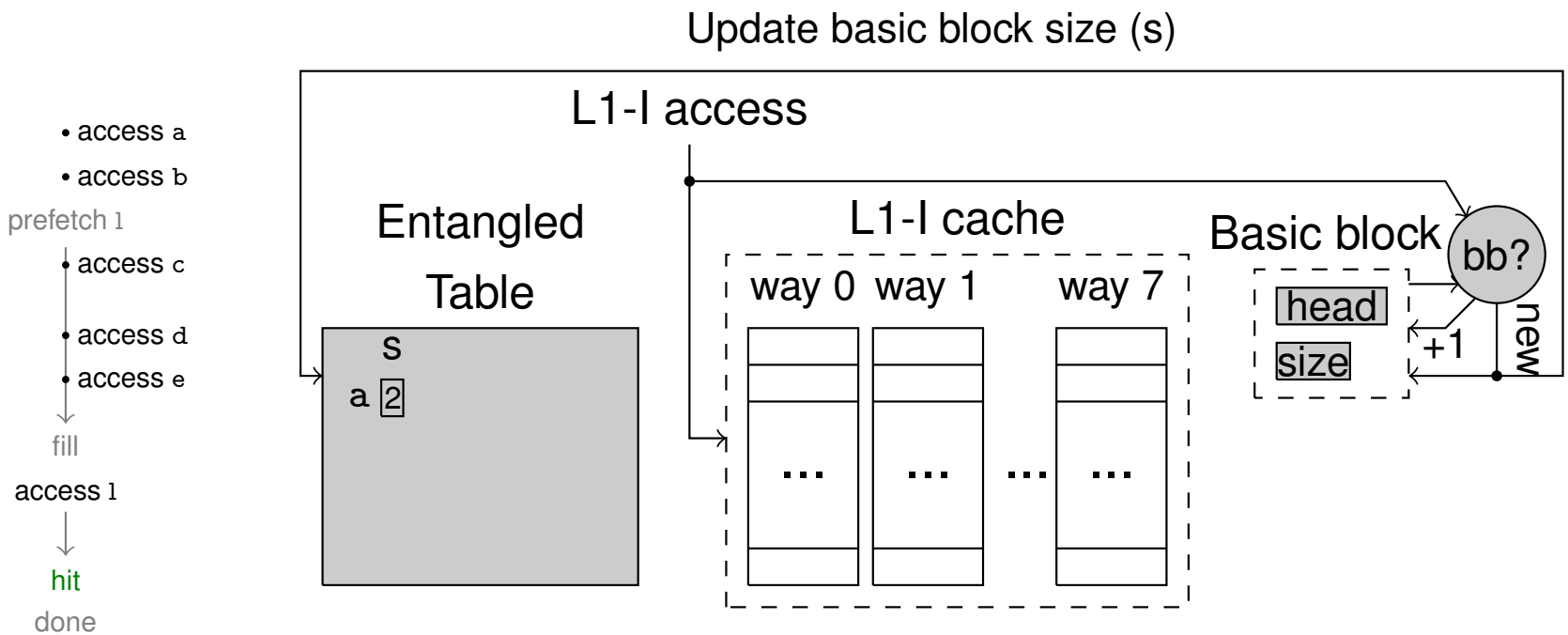




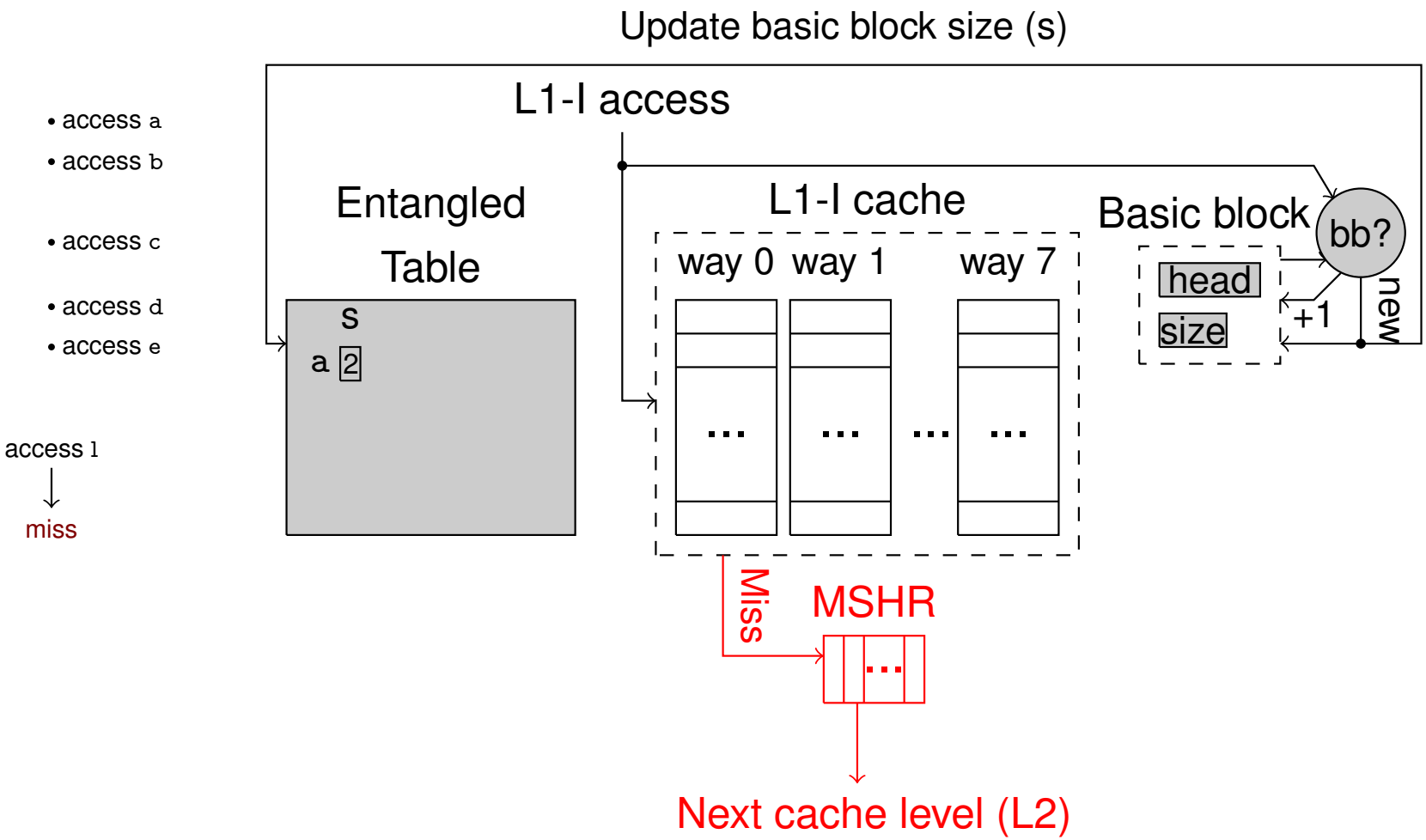
Design of the Entangling Prefetcher - Finding basic blocks



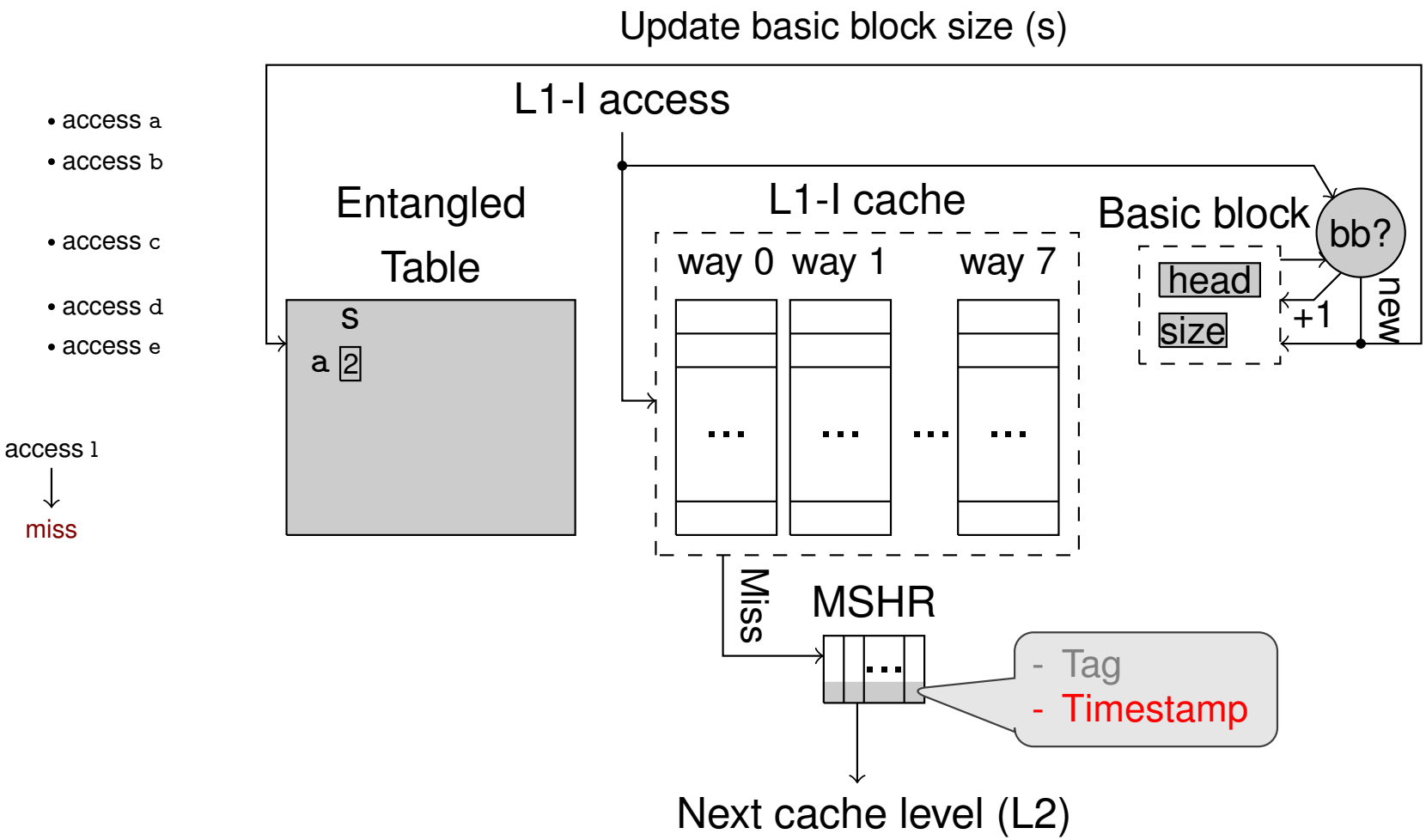
Design of the Entangling Prefetcher - Entangling cache lines



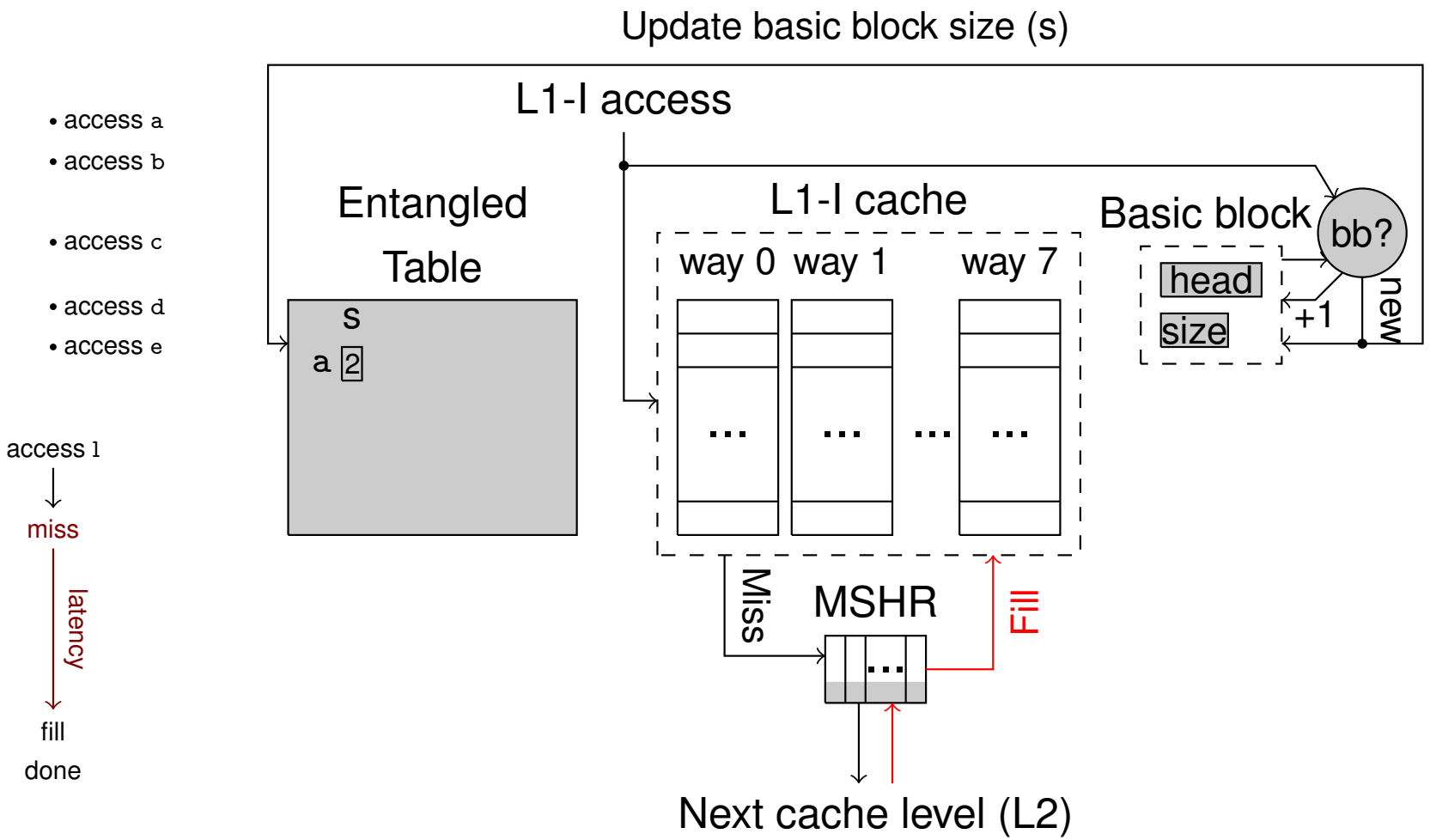
Design of the Entangling Prefetcher - Entangling cache lines



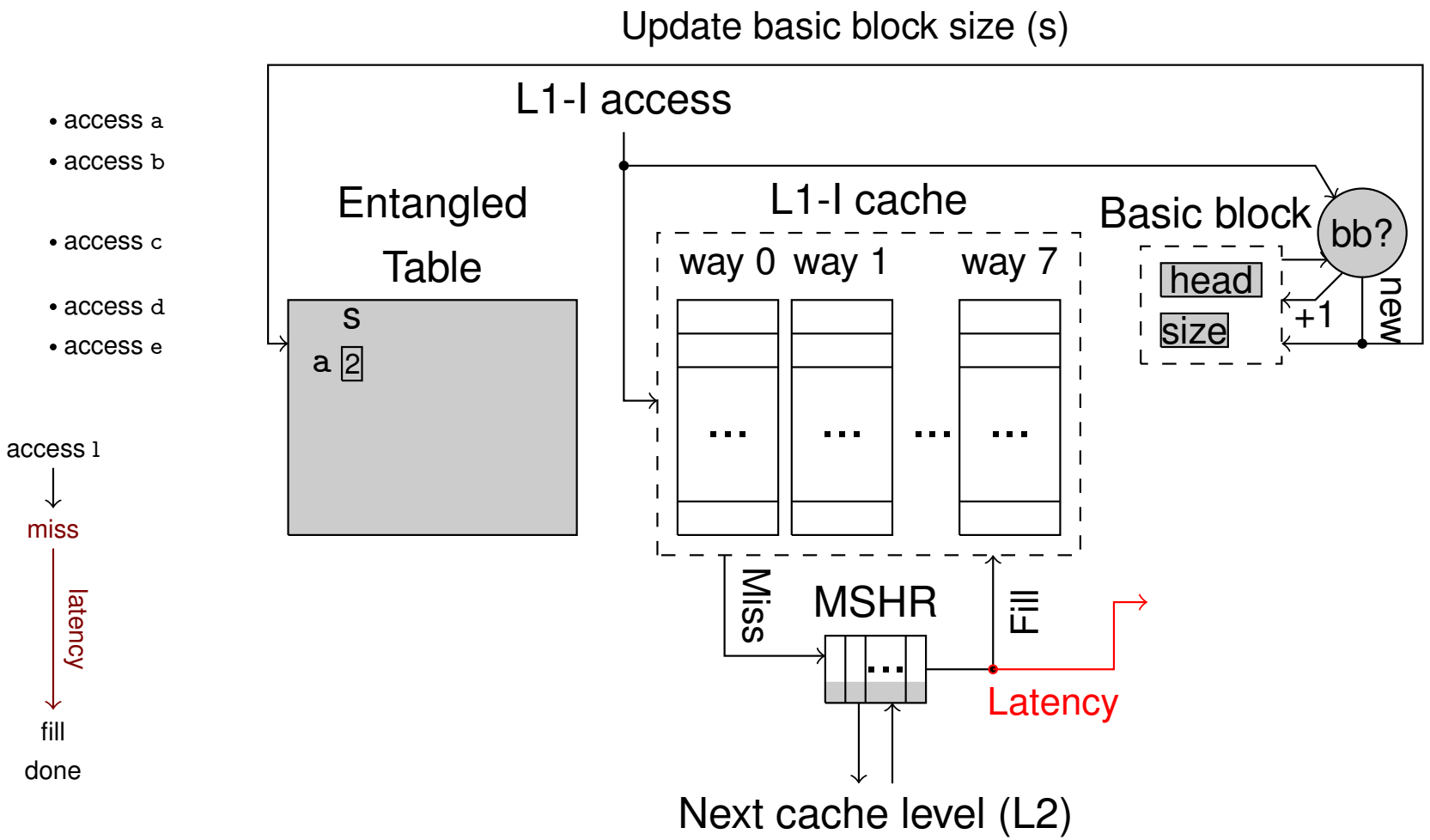
Design of the Entangling Prefetcher - Entangling cache lines



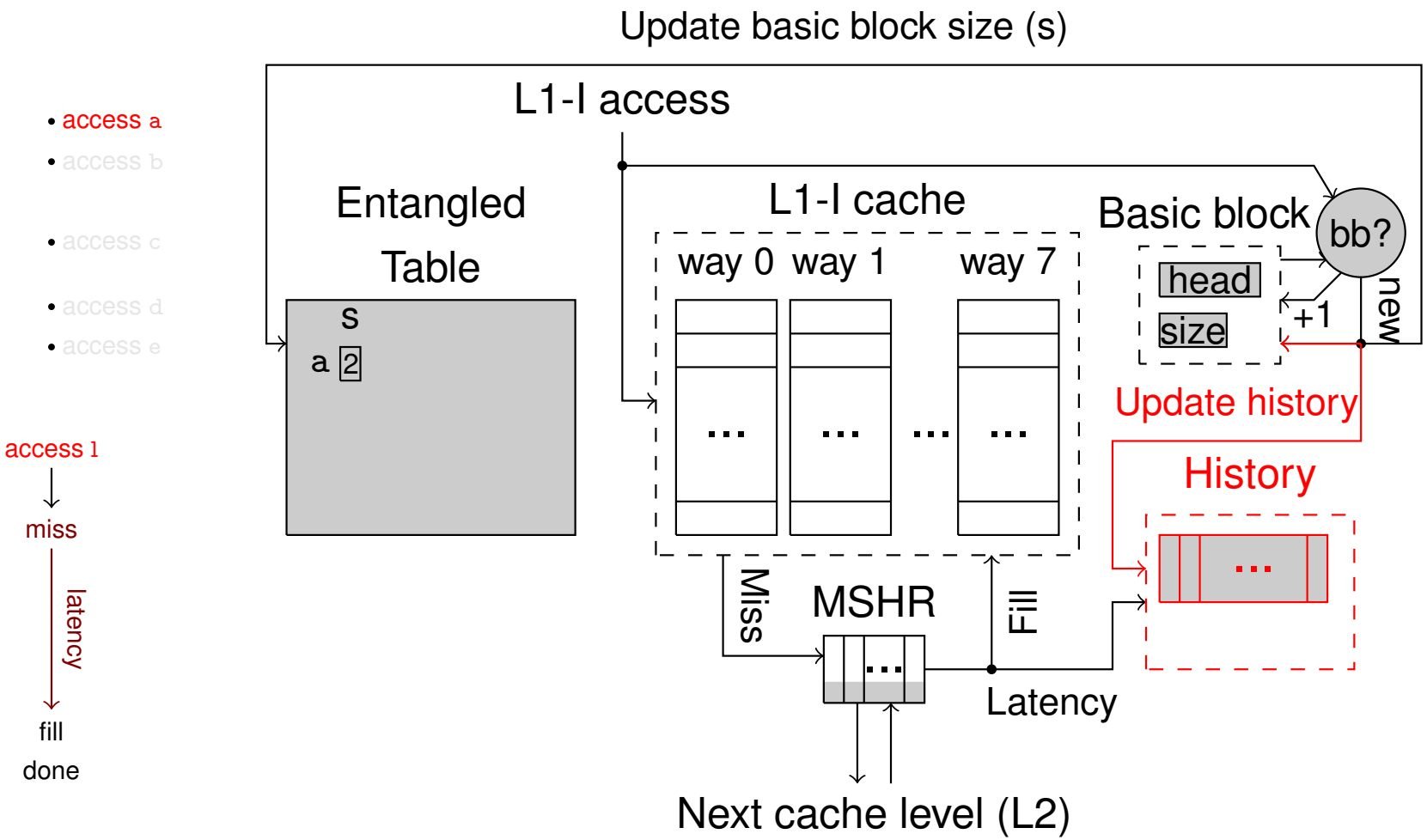
Design of the Entangling Prefetcher - Entangling cache lines



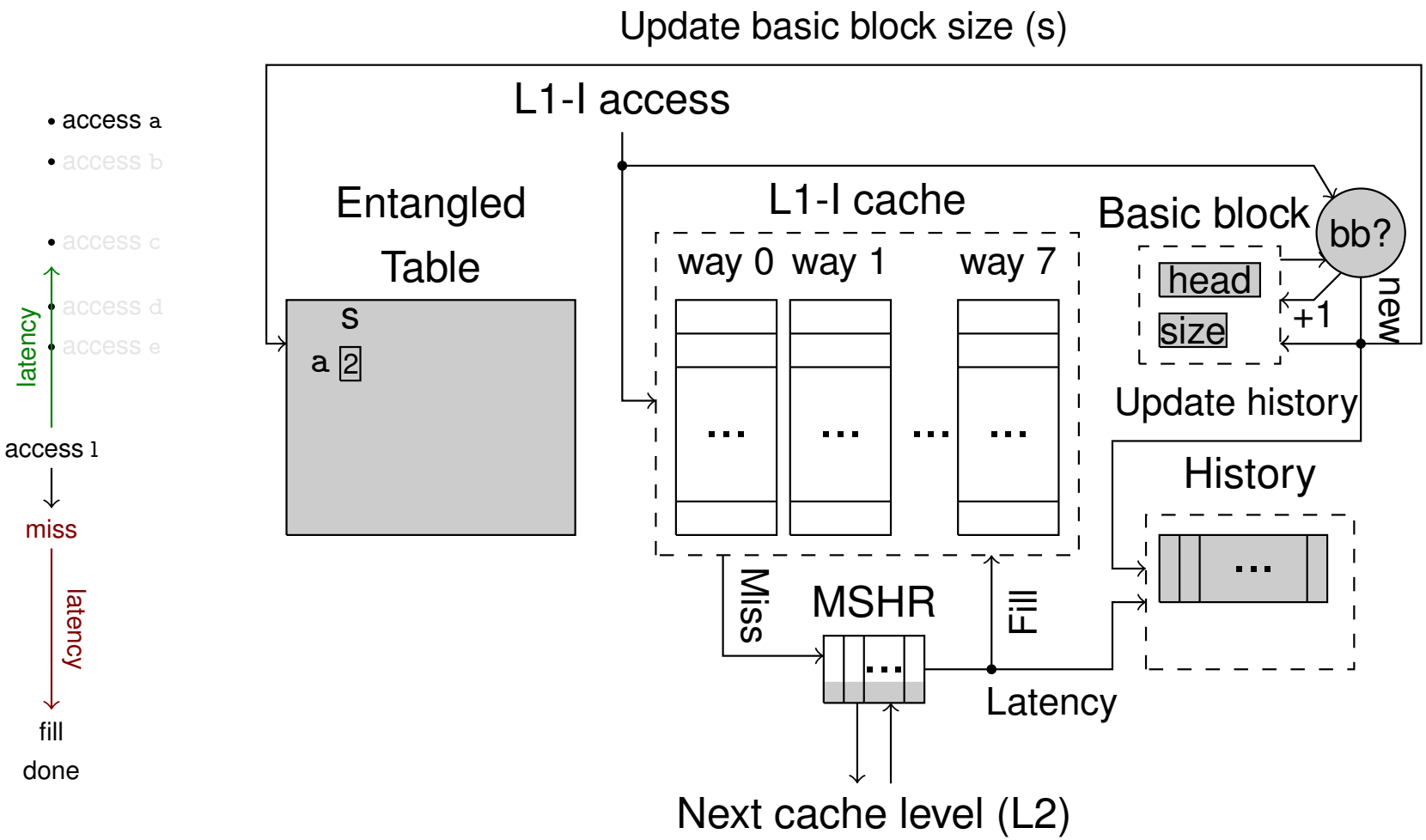
Design of the Entangling Prefetcher - Entangling cache lines



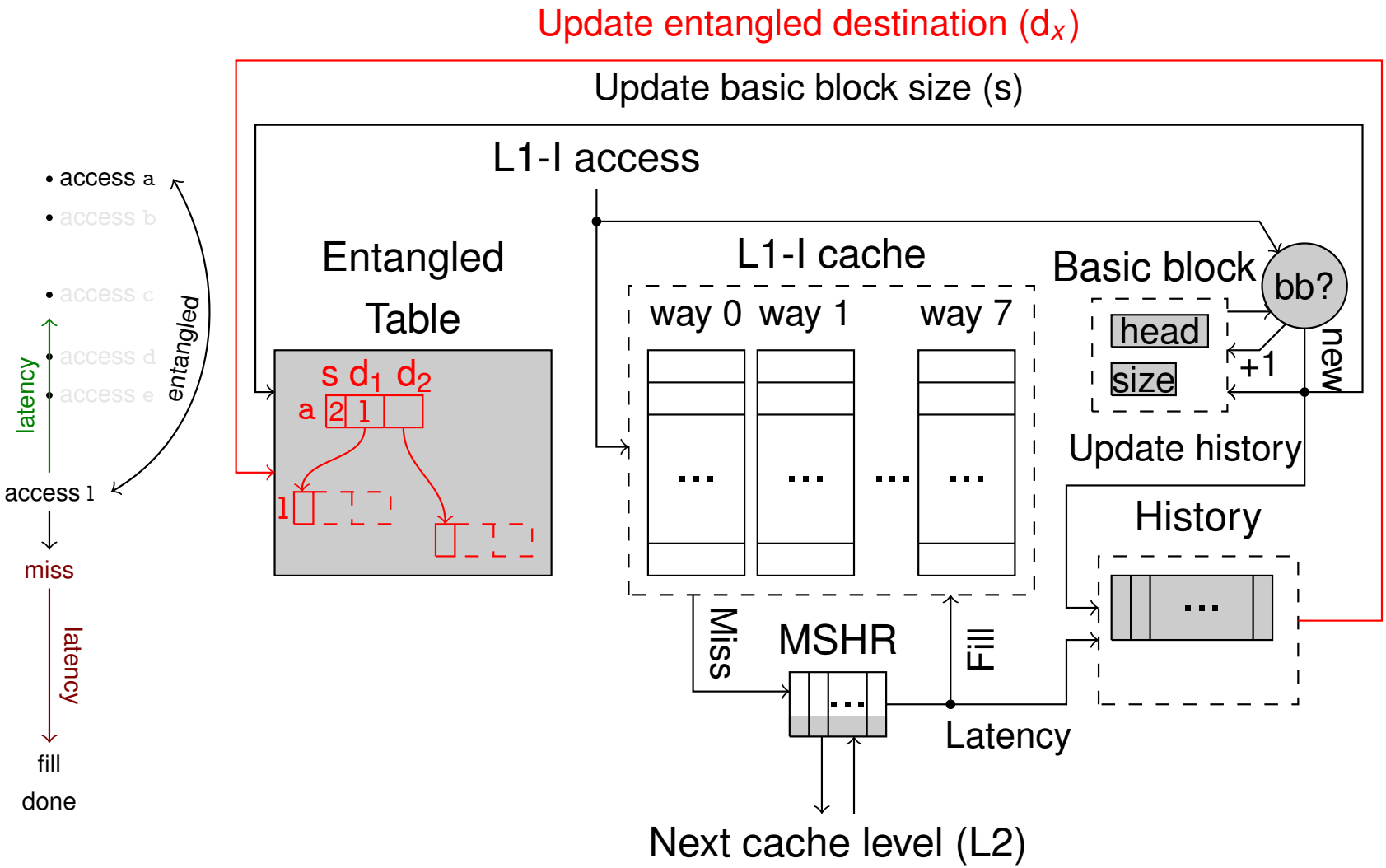
Design of the Entangling Prefetcher - Entangling cache lines



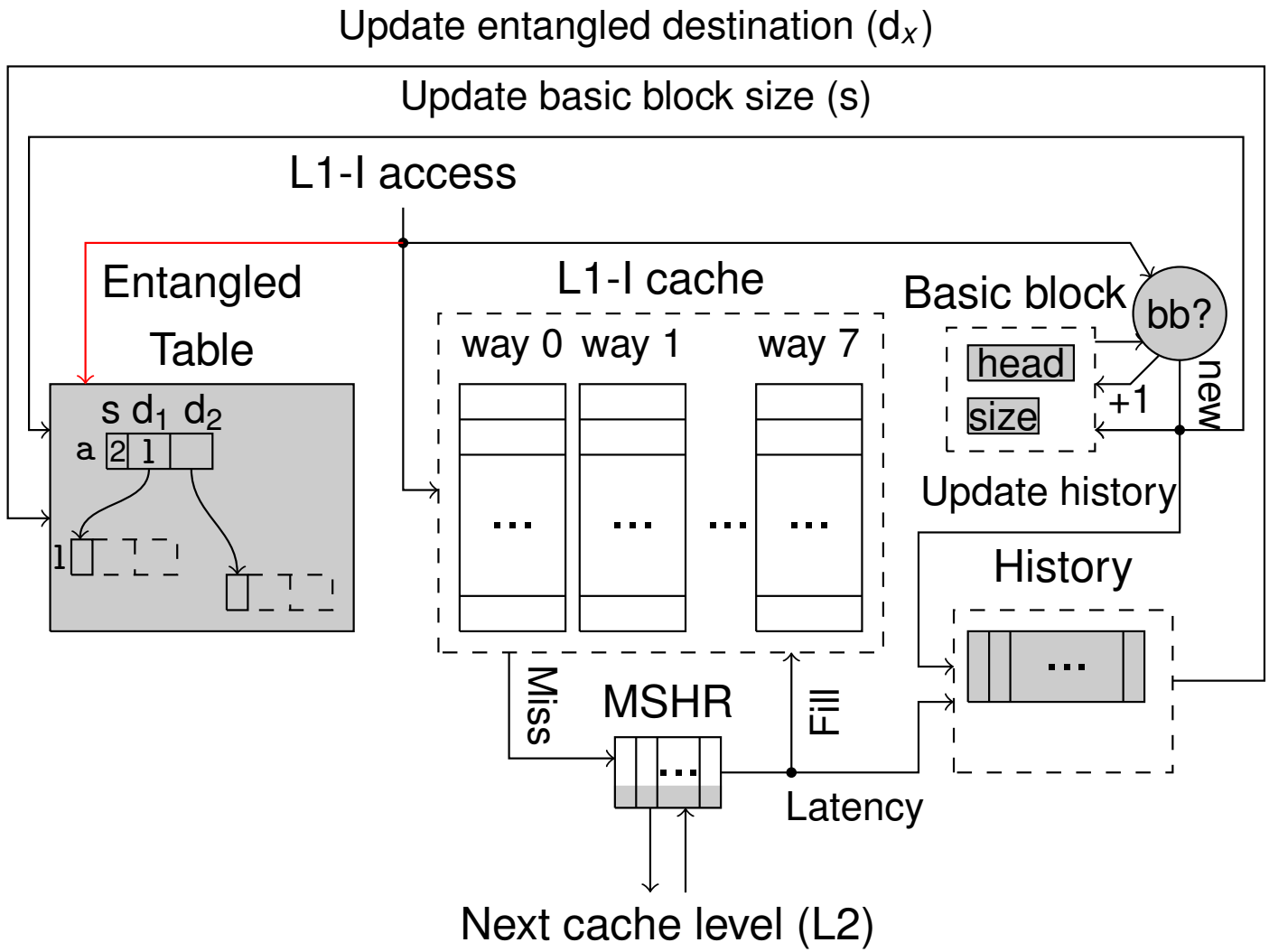
Design of the Entangling Prefetcher - Entangling cache lines



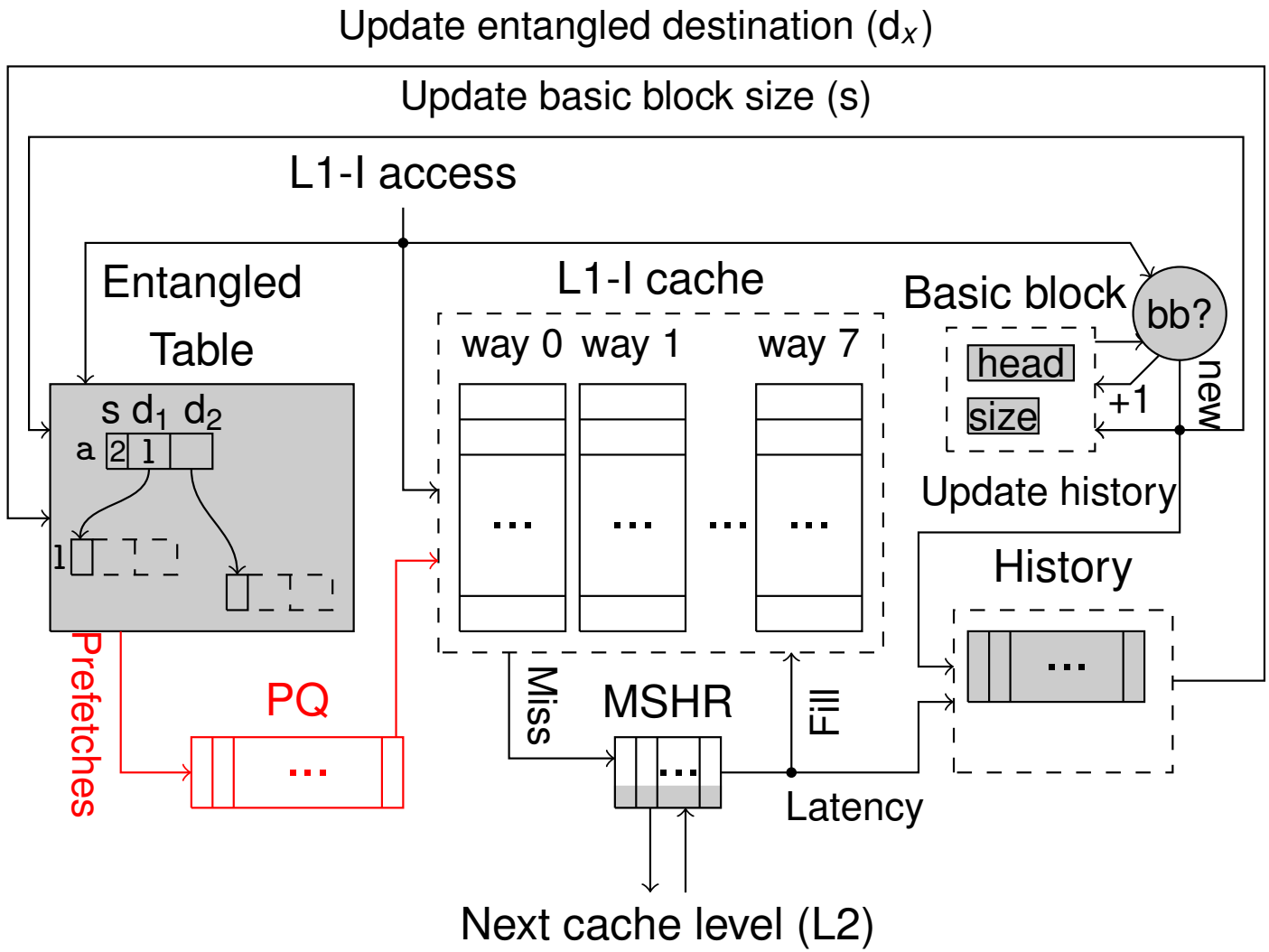
Design of the Entangling Prefetcher - Entangling cache lines



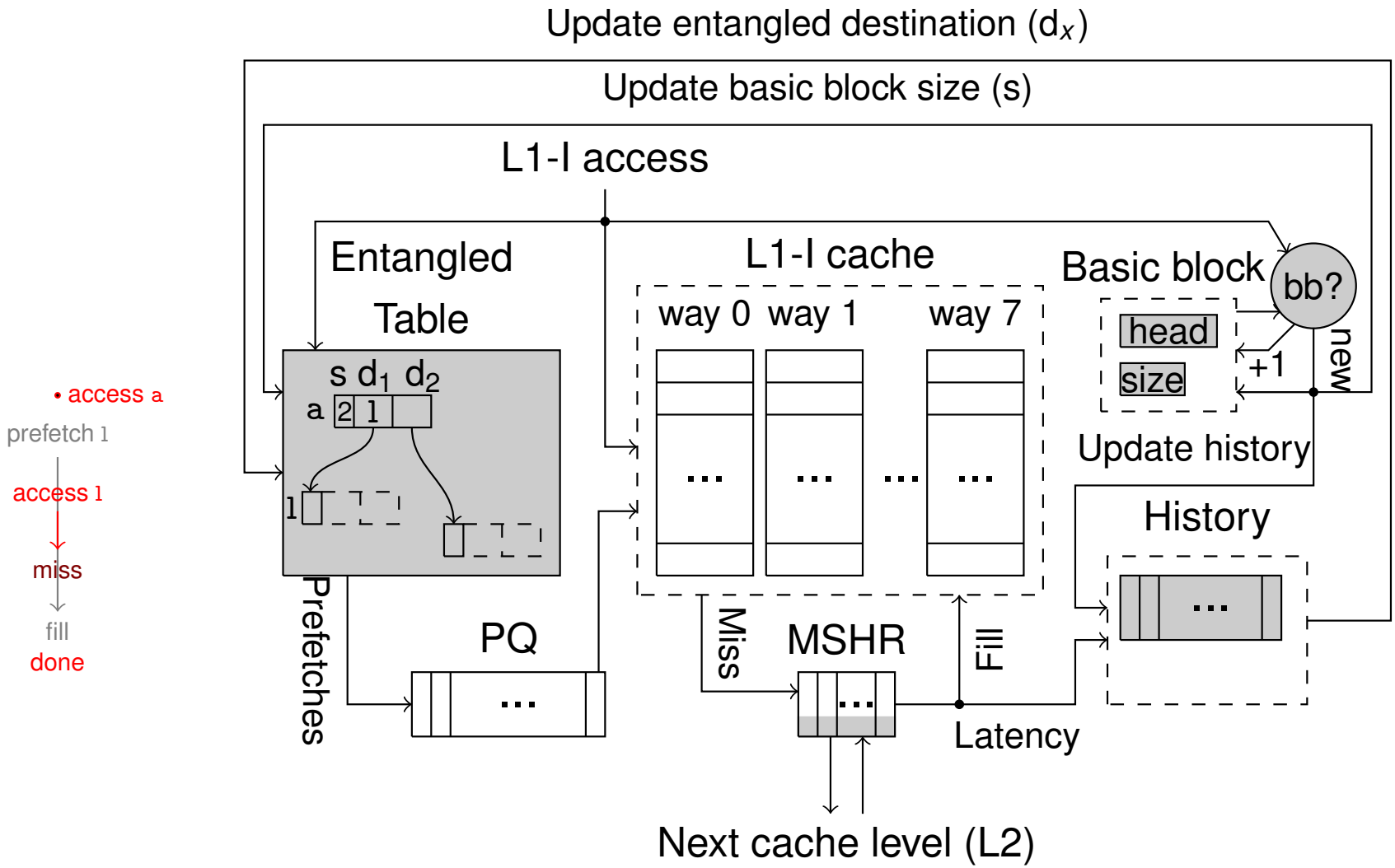
Design of the Entangling Prefetcher - Issuing prefetches



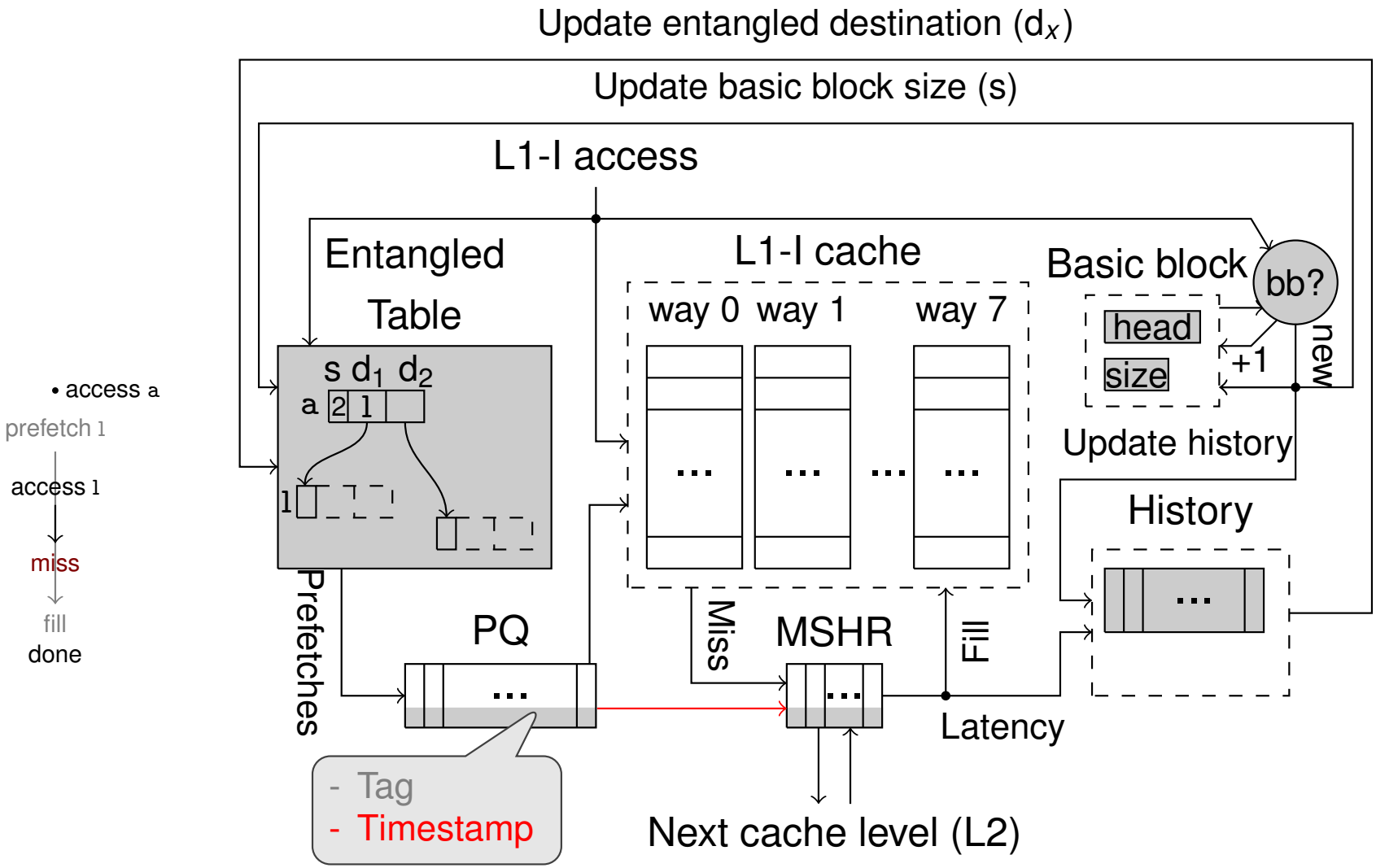
Design of the Entangling Prefetcher - Issuing prefetches



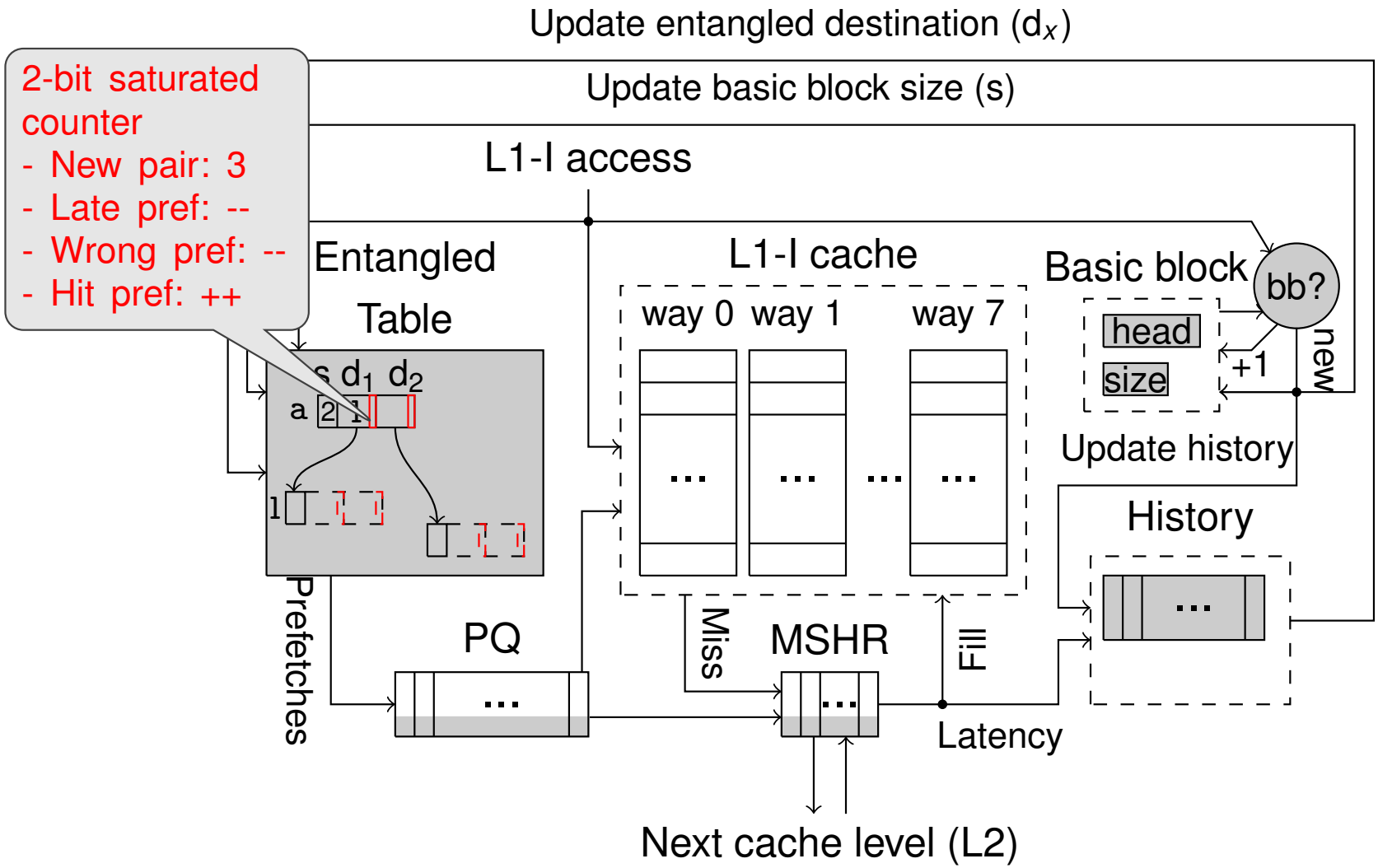
Design of the Entangling Prefetcher - Fixing late prefetches



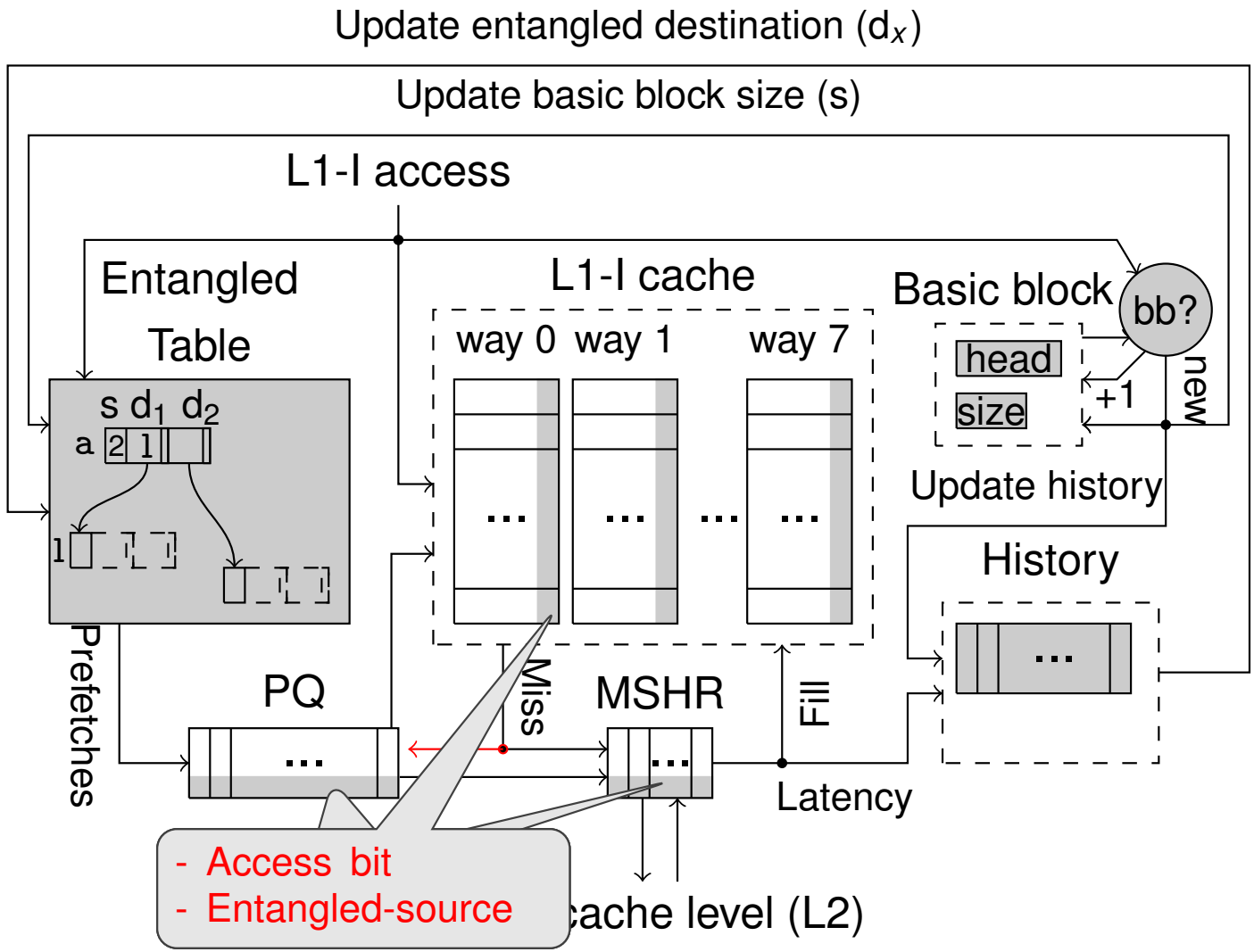
Design of the Entangling Prefetcher - Fixing late prefetches



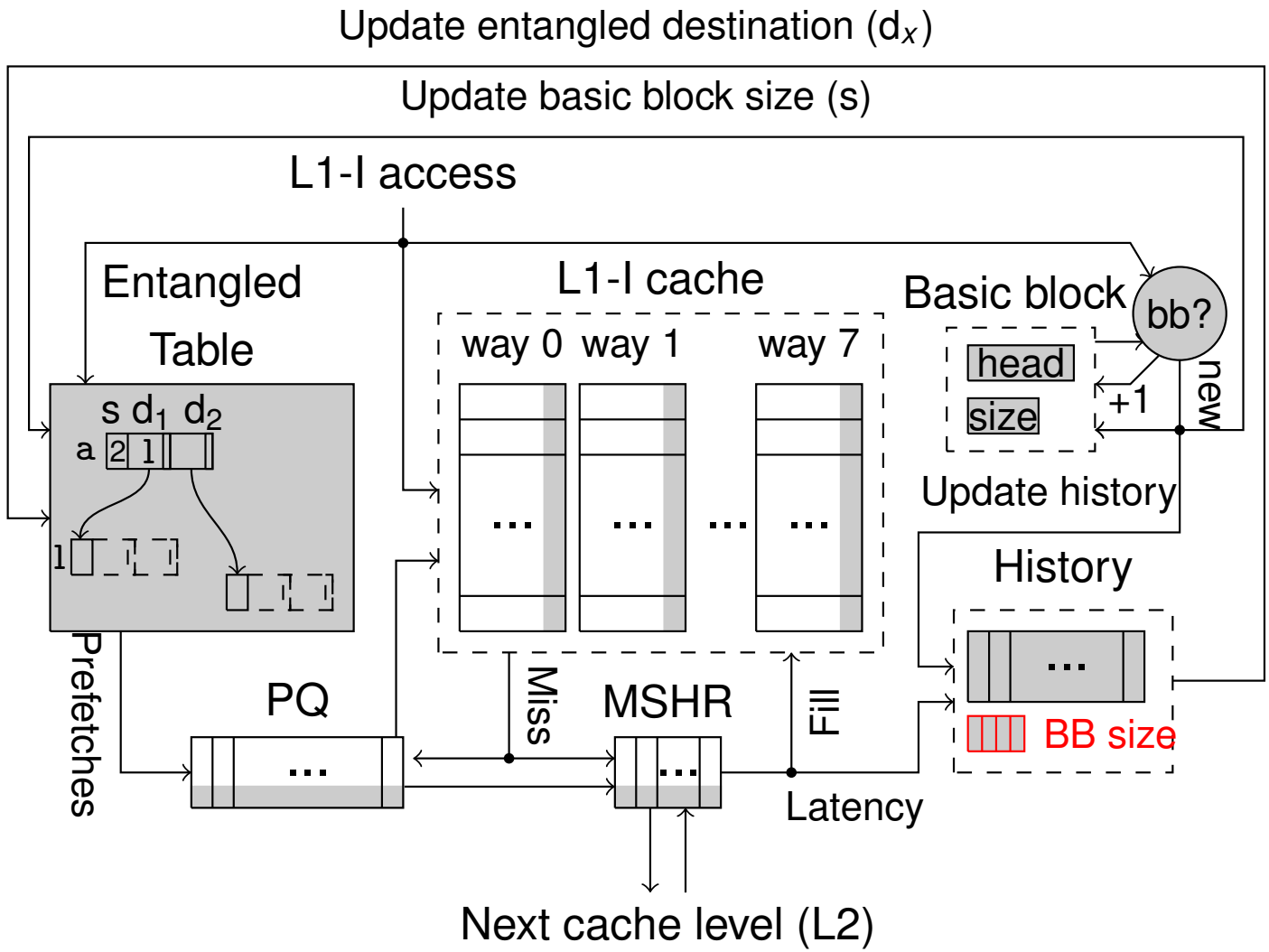
Design of the Entangling Prefetcher - Confidence for entangled pairs



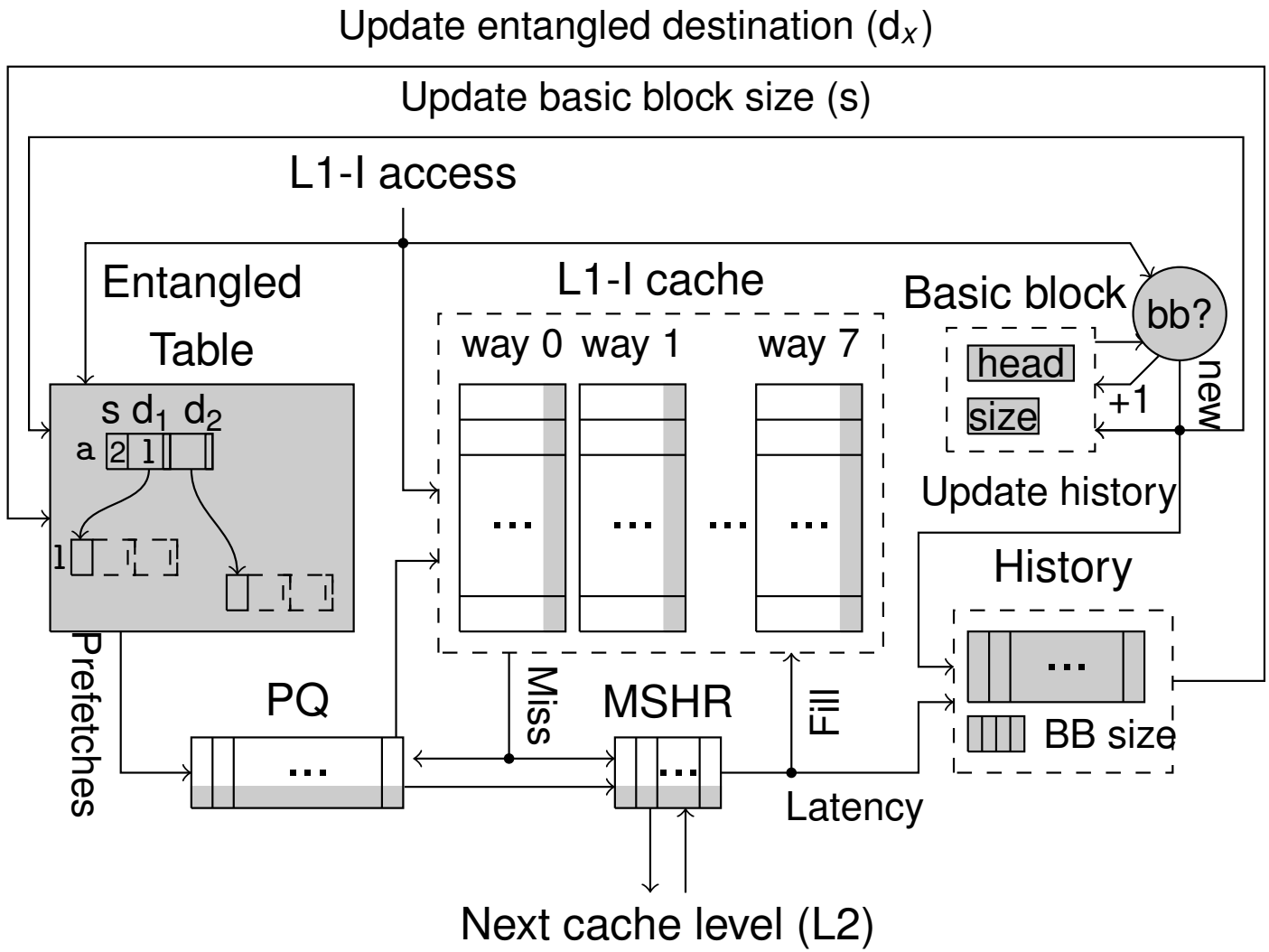
Design of the Entangling Prefetcher - Confidence for entangled pairs



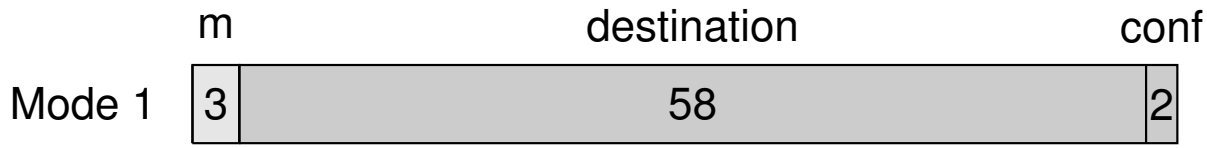
Design of the Entangling Prefetcher - Merging basic blocks



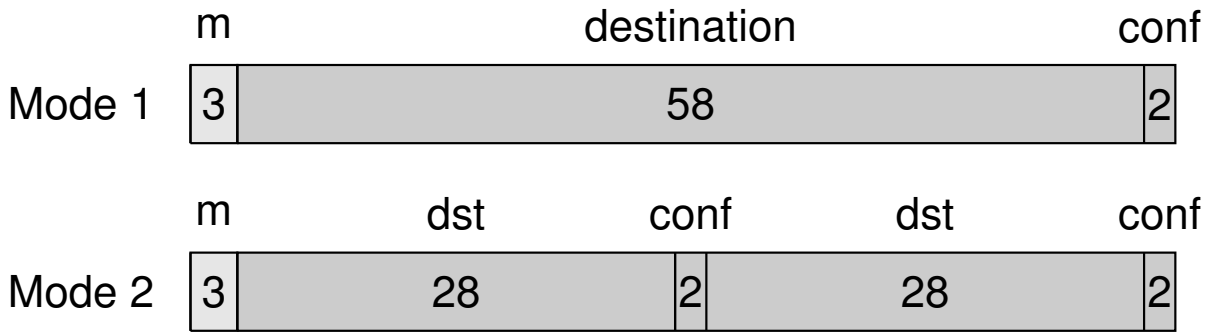
Design of the Entangling Prefetcher



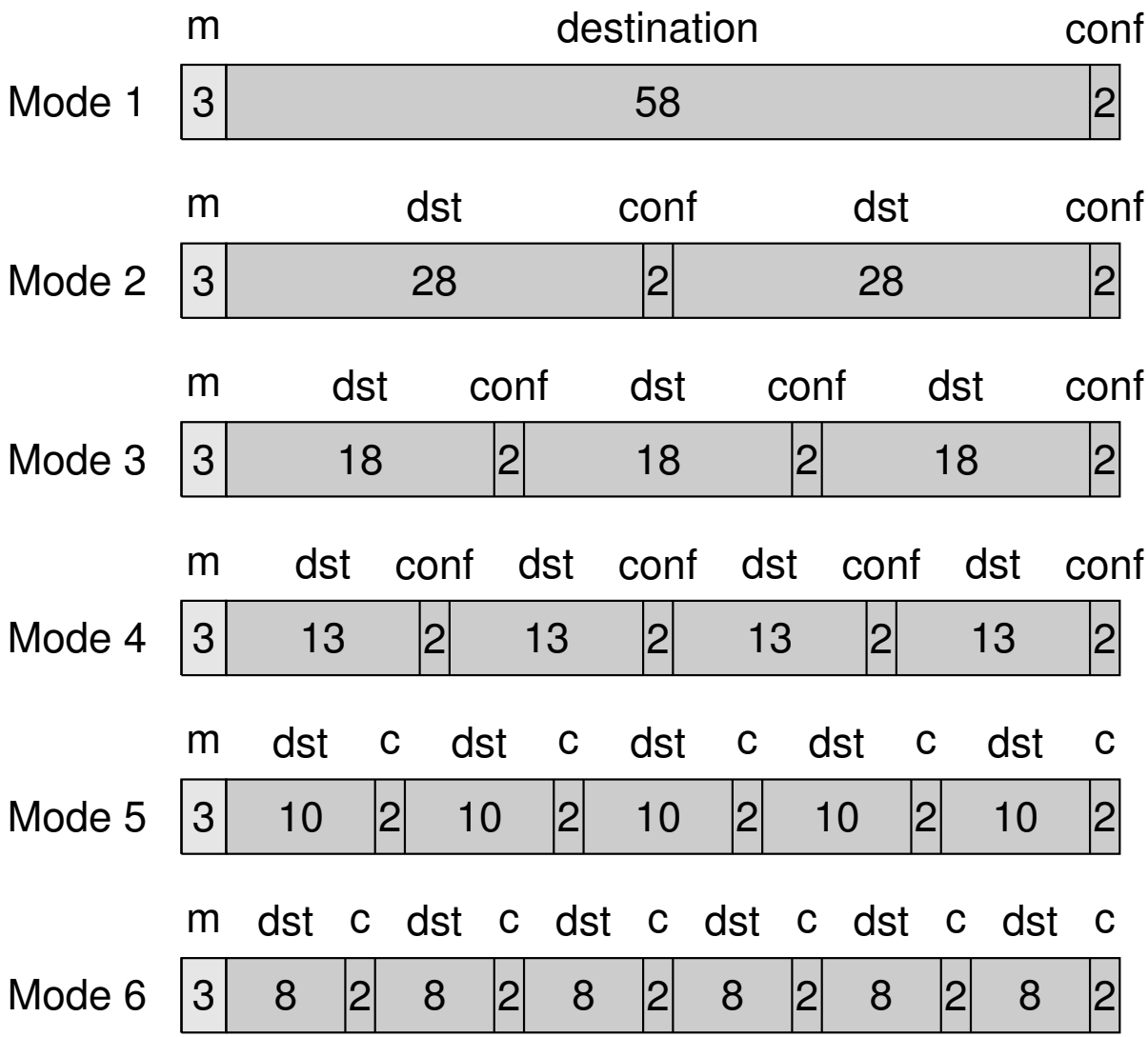
COMPRESSING DESTINATIONS



COMPRESSING DESTINATIONS



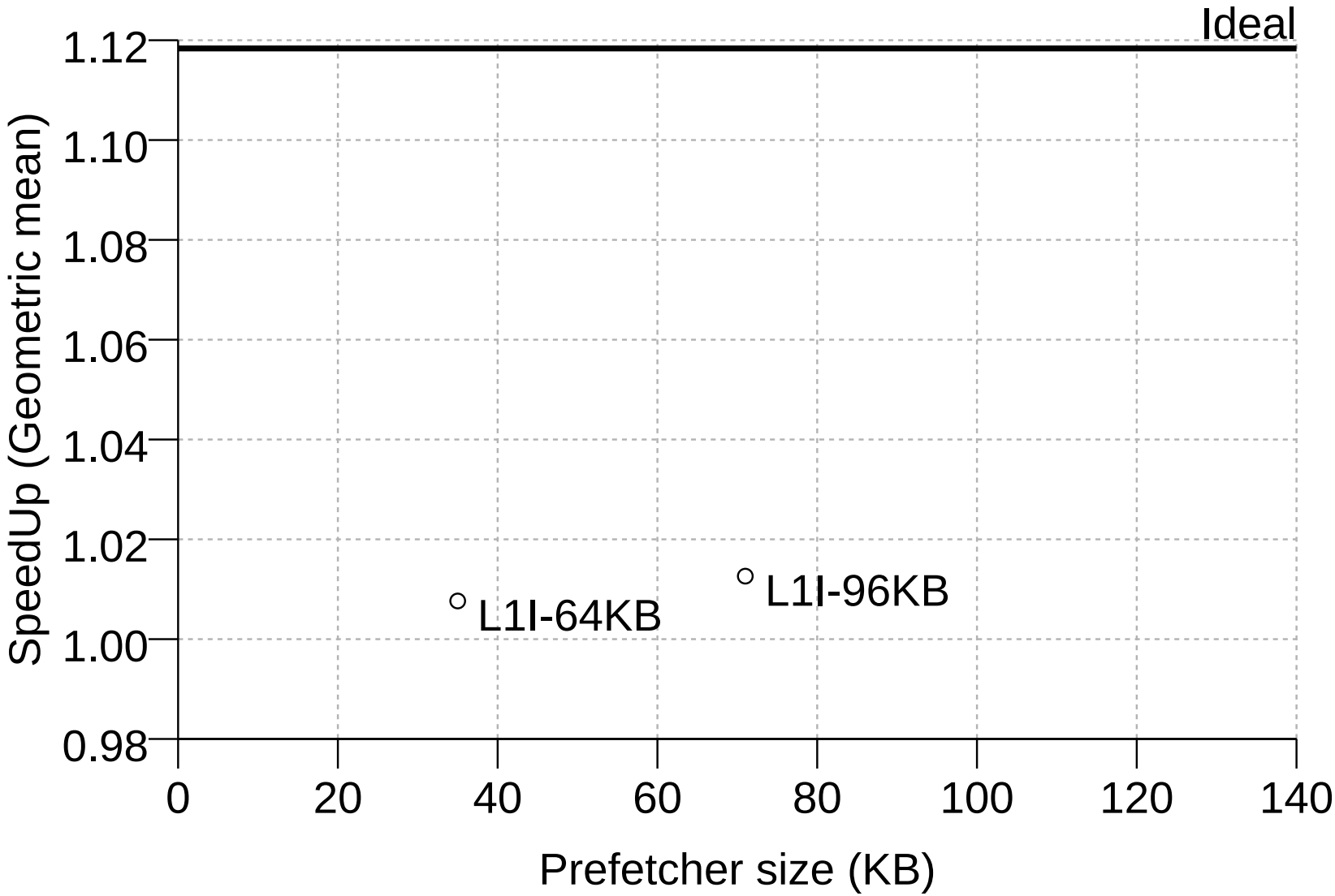
COMPRESSING DESTINATIONS



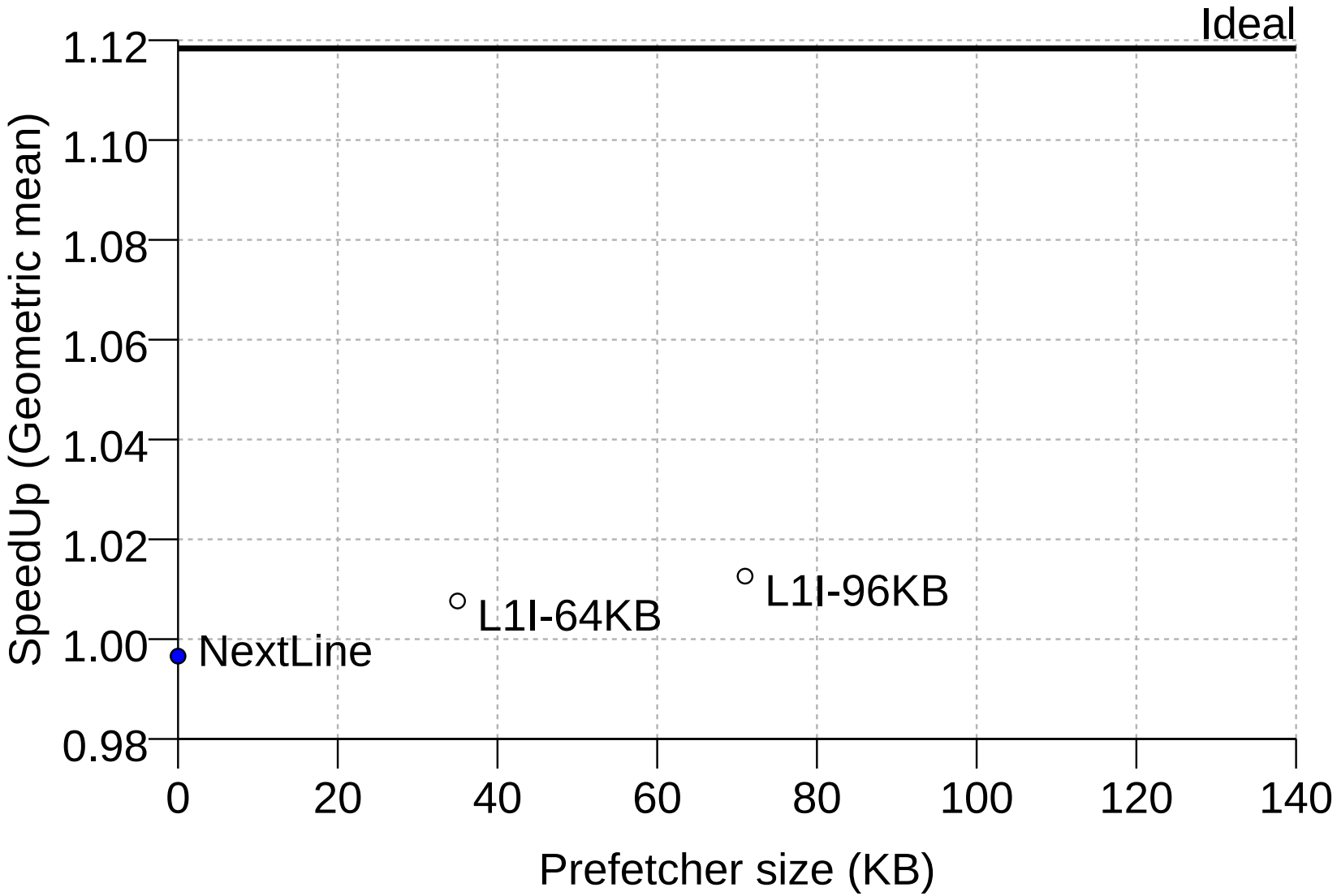
METHODOLOGY

- ChampSim develop branch (nov 2020)
- Baseline:
  - Sunny Cove-like system
  - Decoupled front-end (64-entry fetch queue)
  - 32KB L1I
- ENTANGLED:
  - *History buffer*: 16 entries
  - *Entangled table*: 2K, 4K and 8K entries
- Applications
  - 959 traces from the Championship Value Prediction (provided by Qualcomm)
  - Cloud Suite
- Analysis both for virtual and physical prefetching

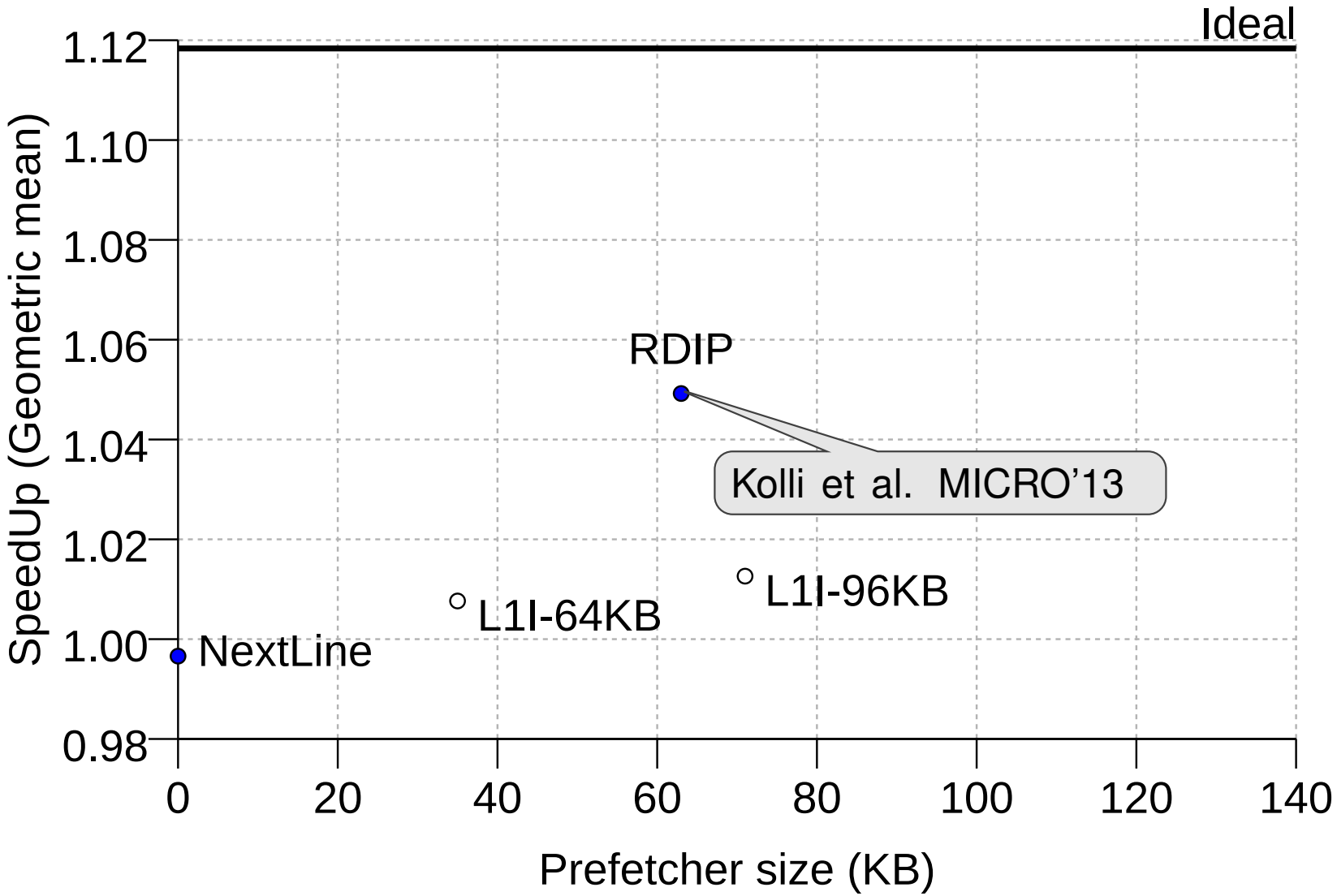
RESULTS: IPC VS MEMORY OVERHEAD



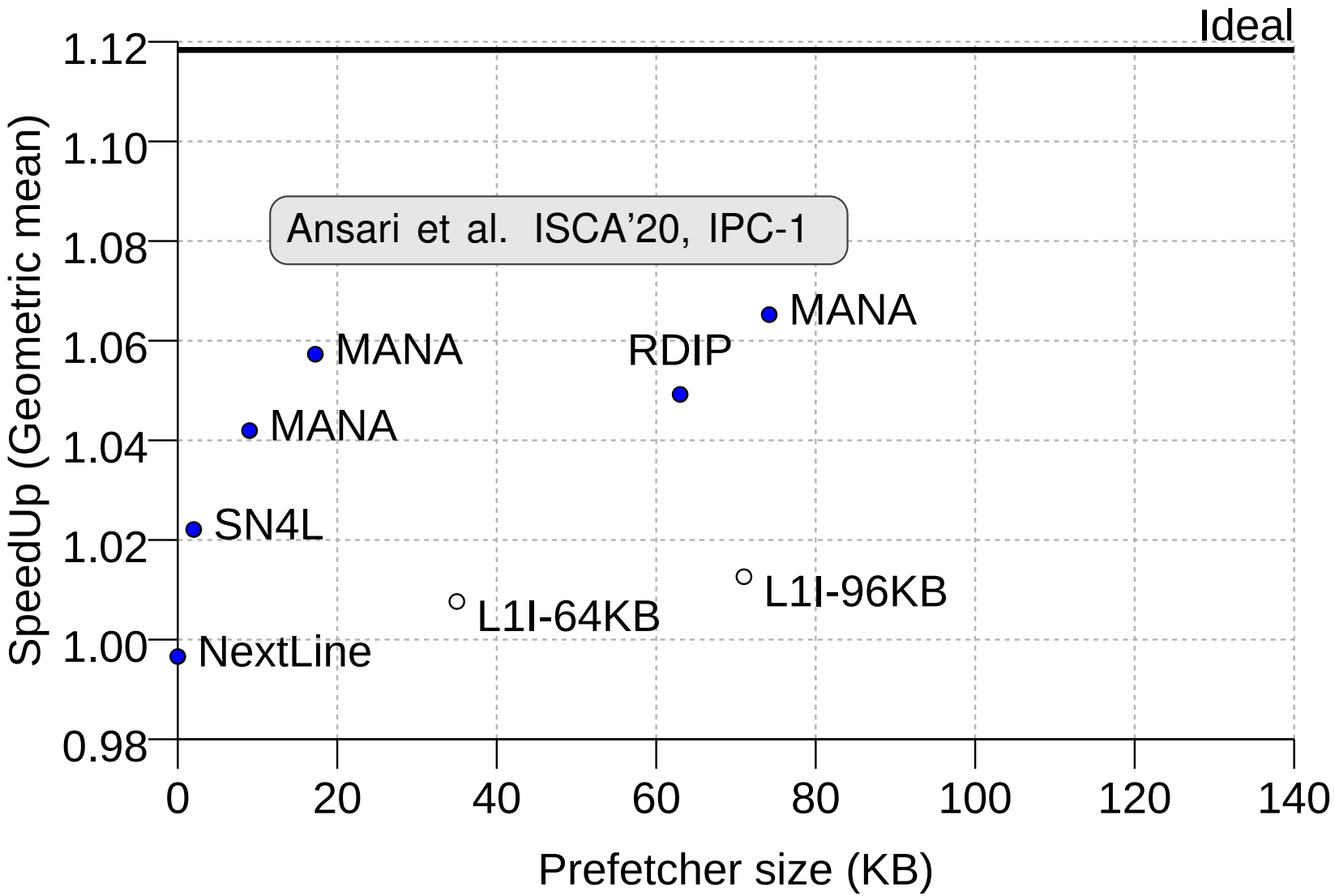
RESULTS: IPC VS MEMORY OVERHEAD



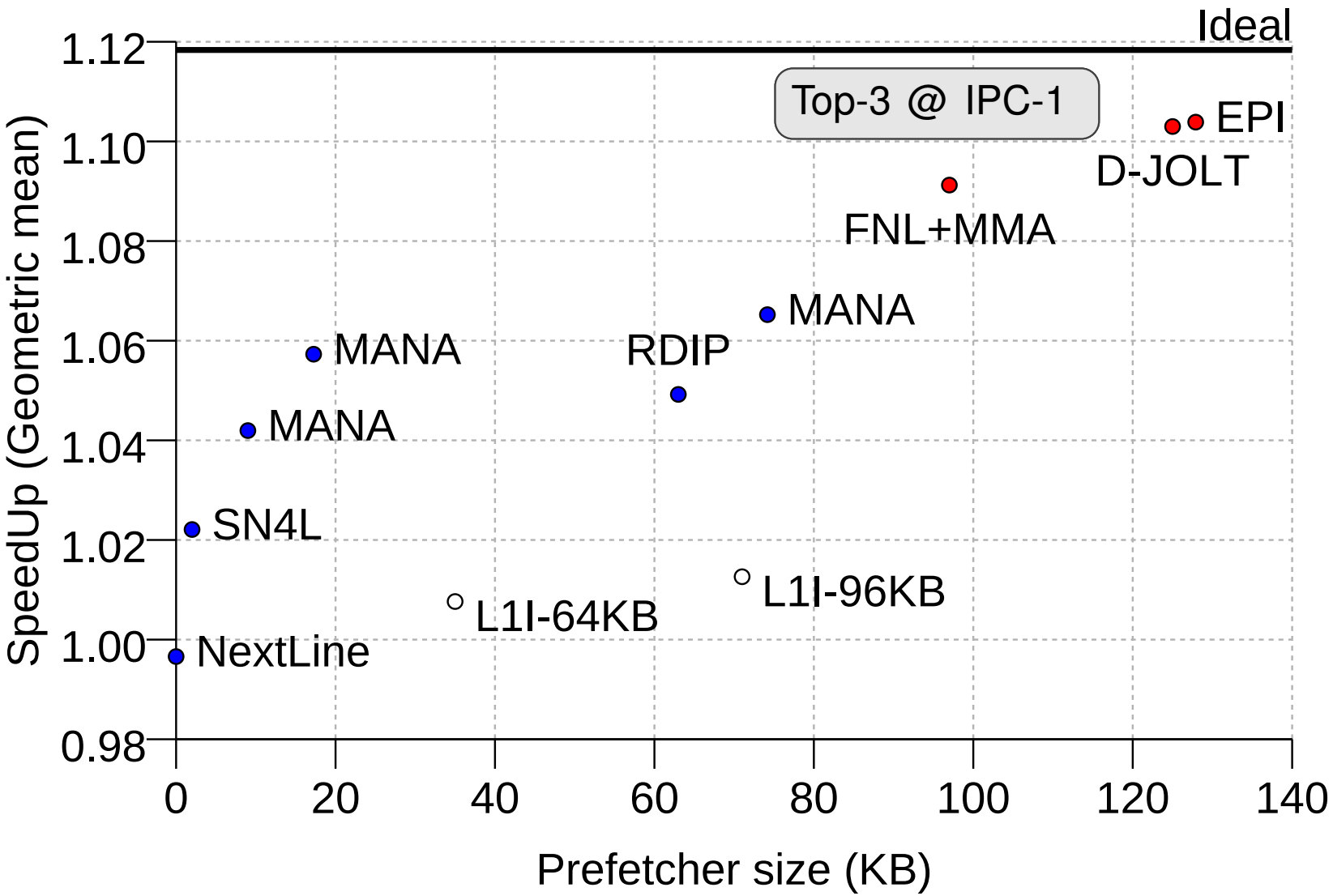
RESULTS: IPC VS MEMORY OVERHEAD



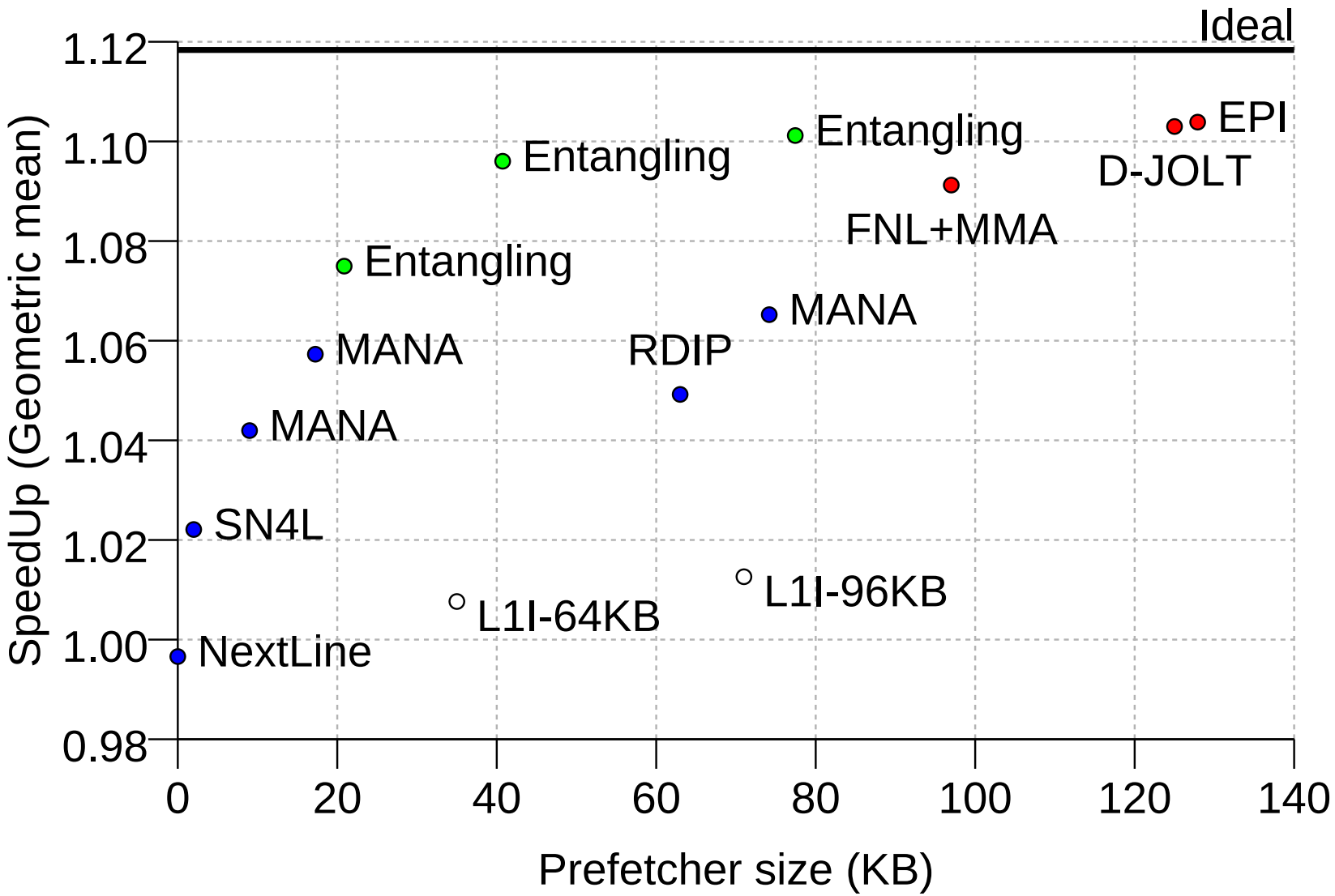
RESULTS: IPC VS MEMORY OVERHEAD



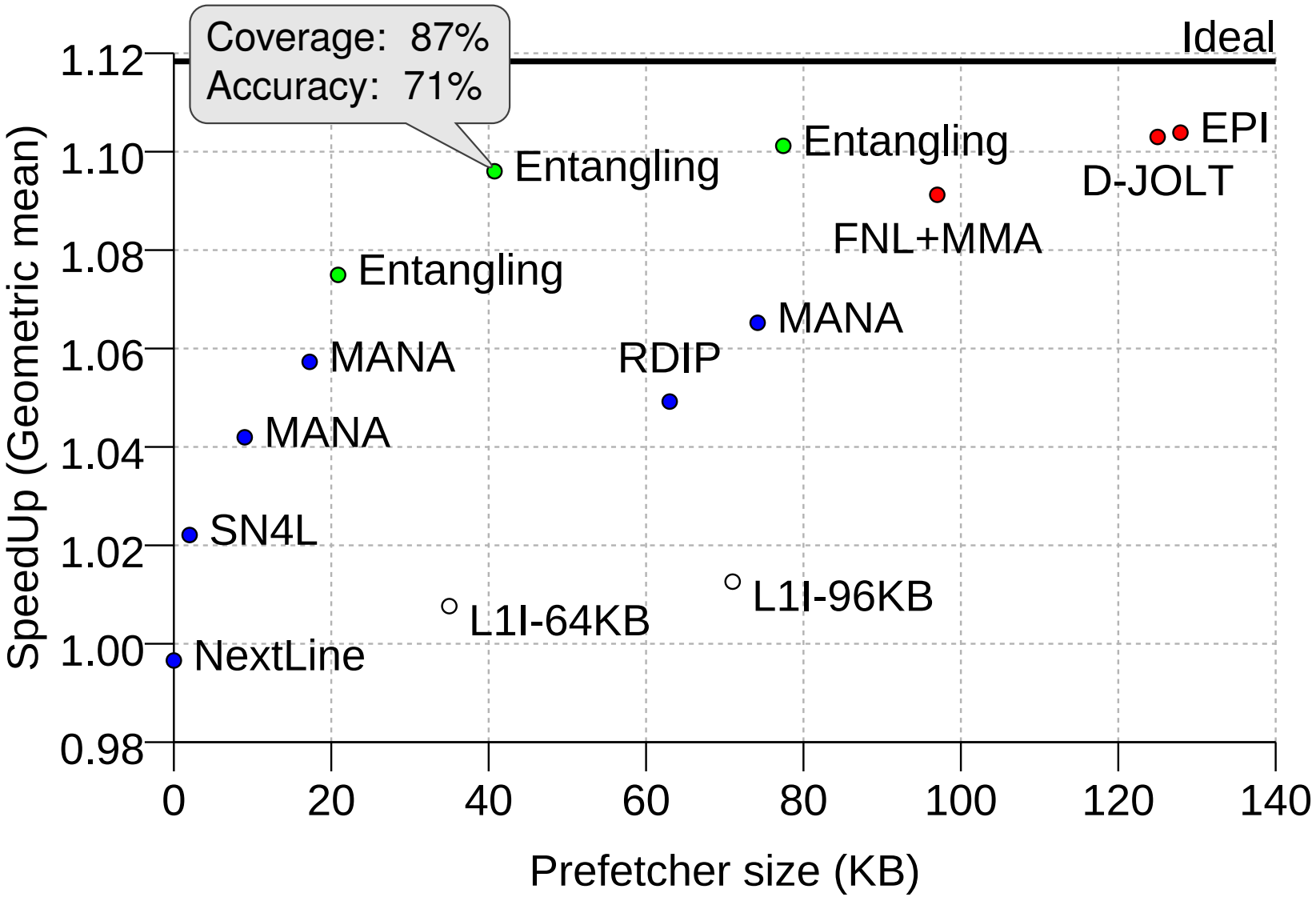
RESULTS: IPC VS MEMORY OVERHEAD



RESULTS: IPC VS MEMORY OVERHEAD



RESULTS: IPC VS MEMORY OVERHEAD



CONCLUDING REMARKS

- Data centers need good prefeteching techniques
- Timeliness as a key property for a prefetcher
- Entangle heads of basic blocks to trigger timely prefetches
- Near ideal L1I performance with just 40KB

Boosting Data Centers Performance with the Entangling Instruction Prefetcher

Alberto Ros

aros@ditec.um.es

Thank you!

ECHO, ERC Consolidator Grant (No 819134)