

# Trace Cache Sampling Filter

Michael Behar,<sup>‡</sup> Avi Mendelson<sup>†</sup> and Avinoam Kolodny<sup>‡</sup>

<sup>‡</sup> Technion, Israel Institute of Technology

<sup>†</sup> Intel Corporation Haifa, Israel

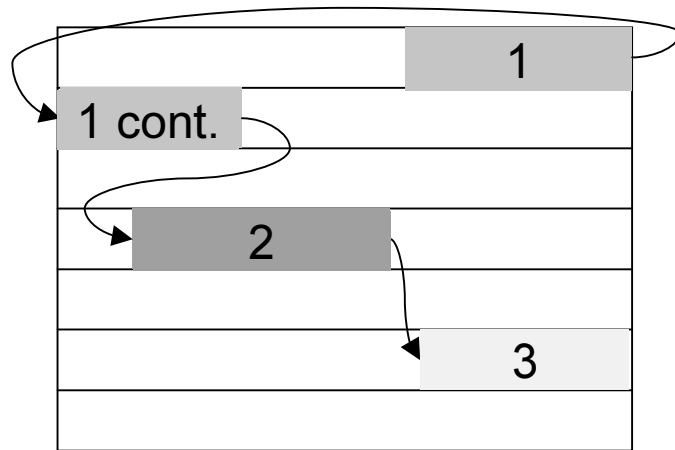


# Outline

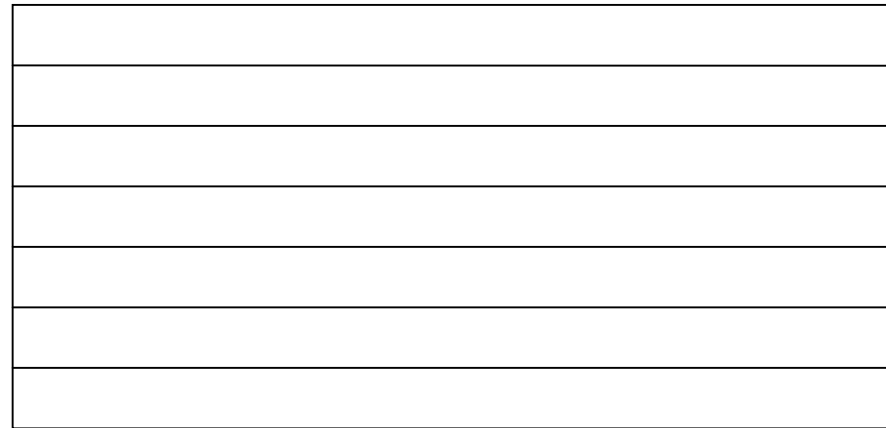
- Introduction
  - Trace cache redundancies
  - “Hot/Cold” Principle
- The sampling Filter
  - Filter Structure
  - Why it works?
  - The SF and FTC-MTC combination

# The Trace Cache\*

## Instruction Cache



## Trace Cache



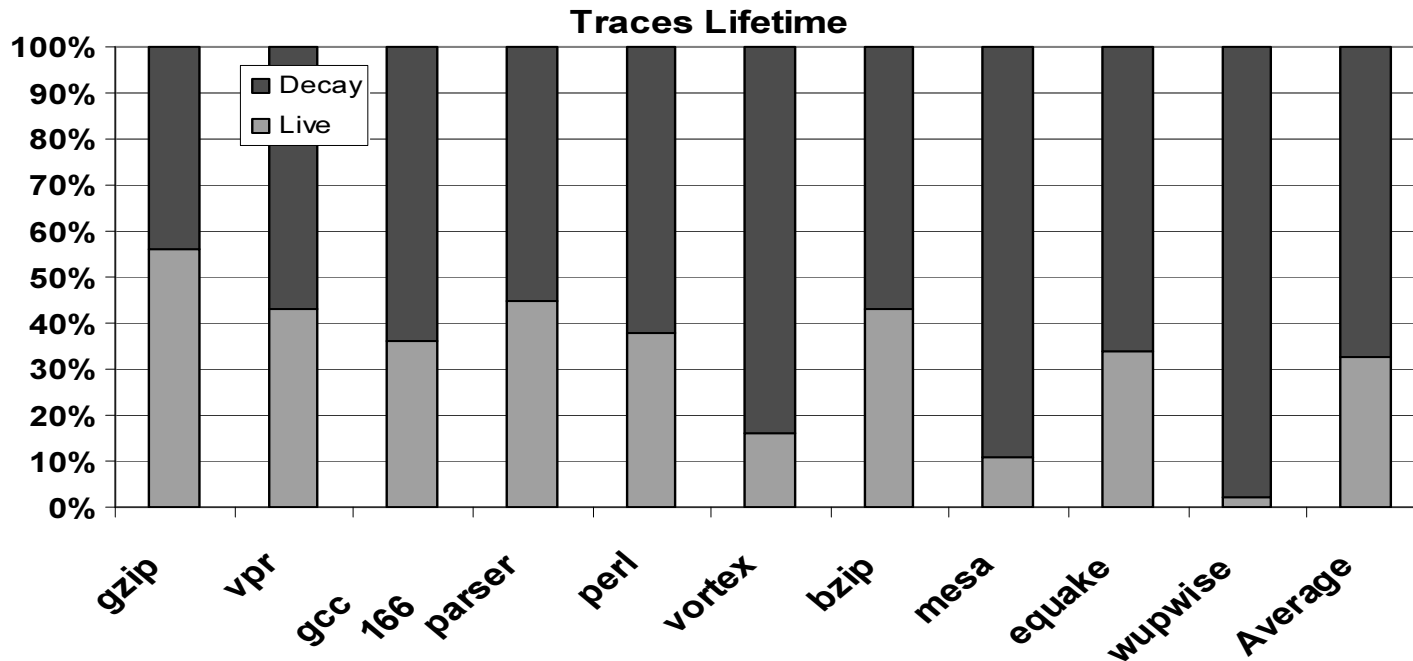
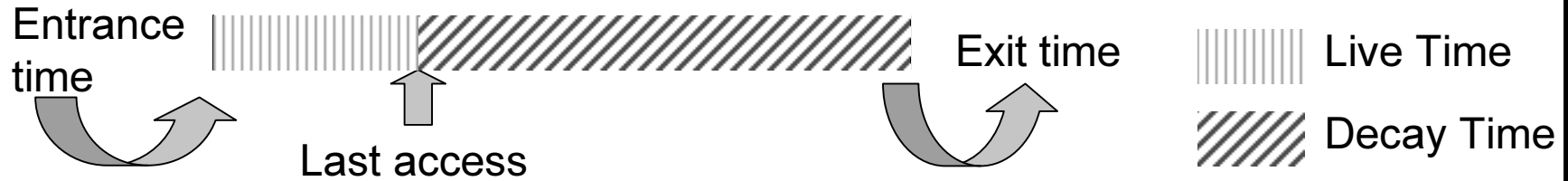
Store the Instructions in their dynamic order rather than in their compiled static order.

# Trace Cache Drawbacks

- Fragmentation
- Indexability
- Basic block duplication
- Duplication between the trace and instruction caches
- Short Live time



# Live-time Problem



\*for a 32-traces Trace Cache



# How to use a trace cache?

- Memory inefficient.
- Must be very large.
  - Longer access time
  - Higher power consumption.

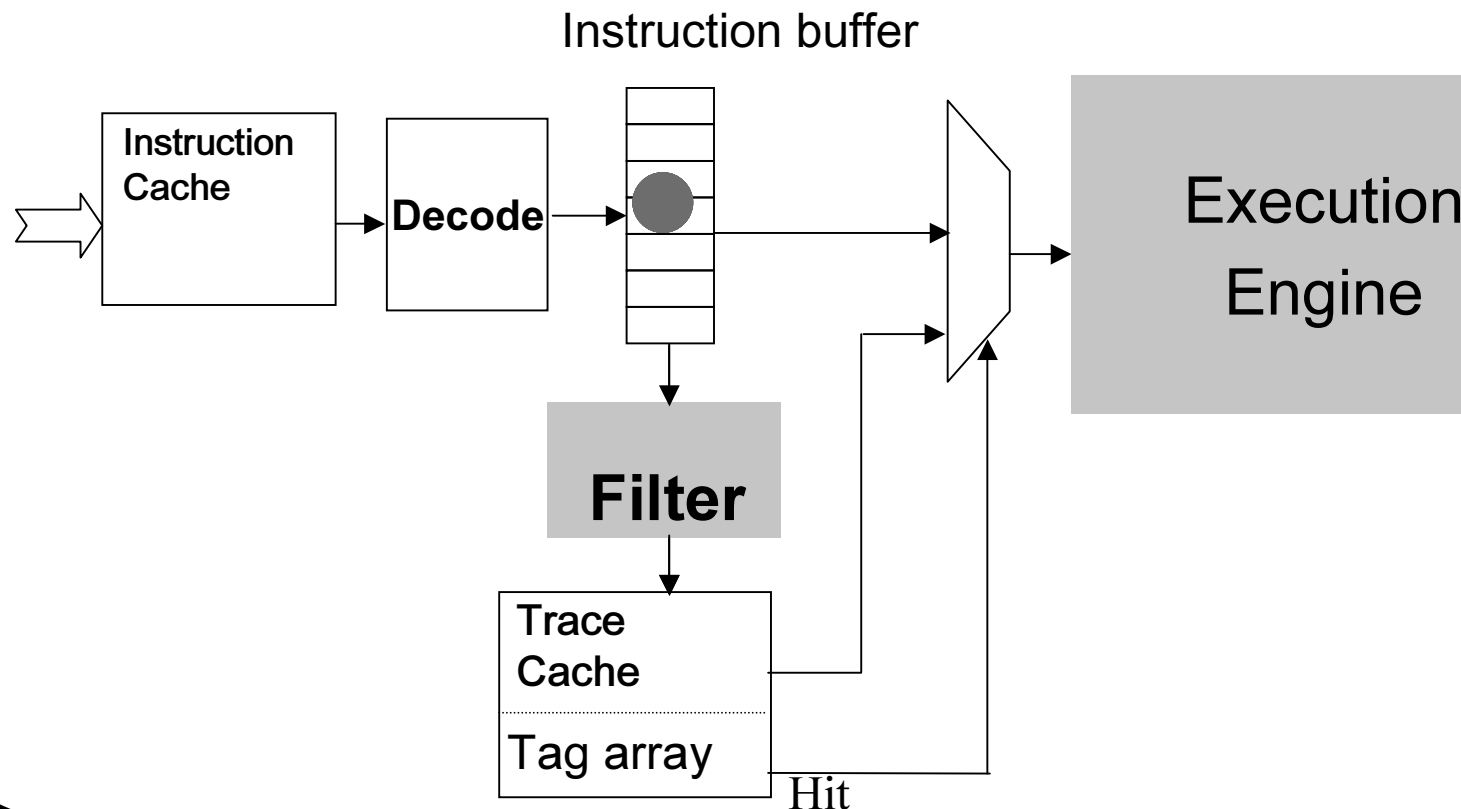
## **The desired solution:**

- A small but **smart** Trace Cache
- A backup Instruction cache



# How can we make the trace cache smart ?

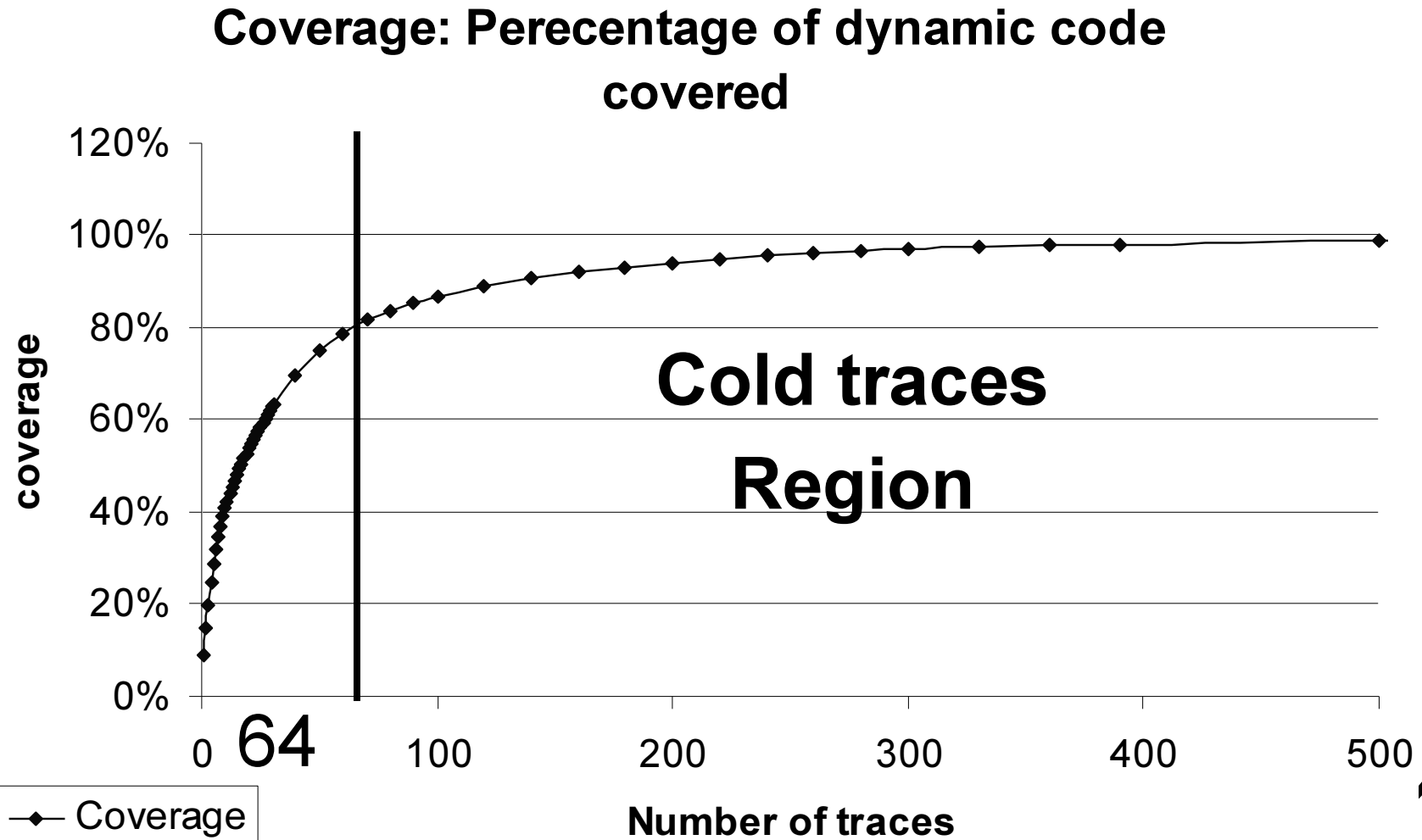
- Use selective storage: Filtering



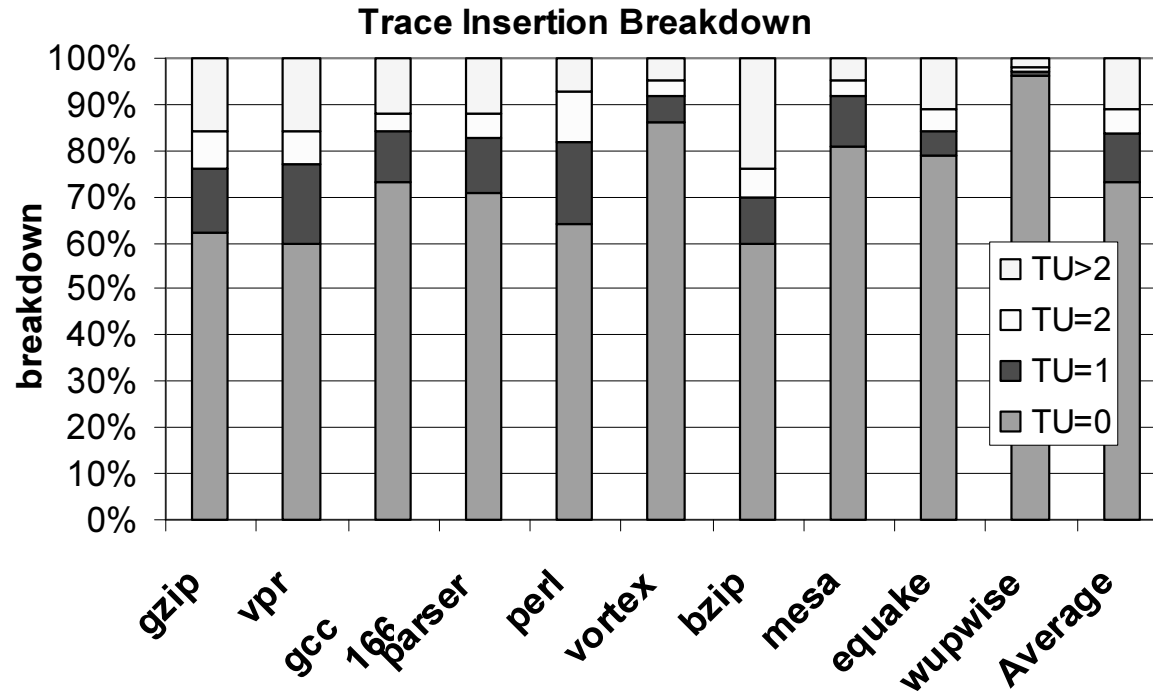
# Hot/Cold Principle (20%/80%)

- **“Hot traces”**: Traces that are executed many times out of the trace cache and contributes most of the retired instructions.
- **“Cold traces”**: Traces that are rarely executed from the trace cache but contributes most of the writes to the cache.

# Hot traces potential



# Trace Insertion Breakdown

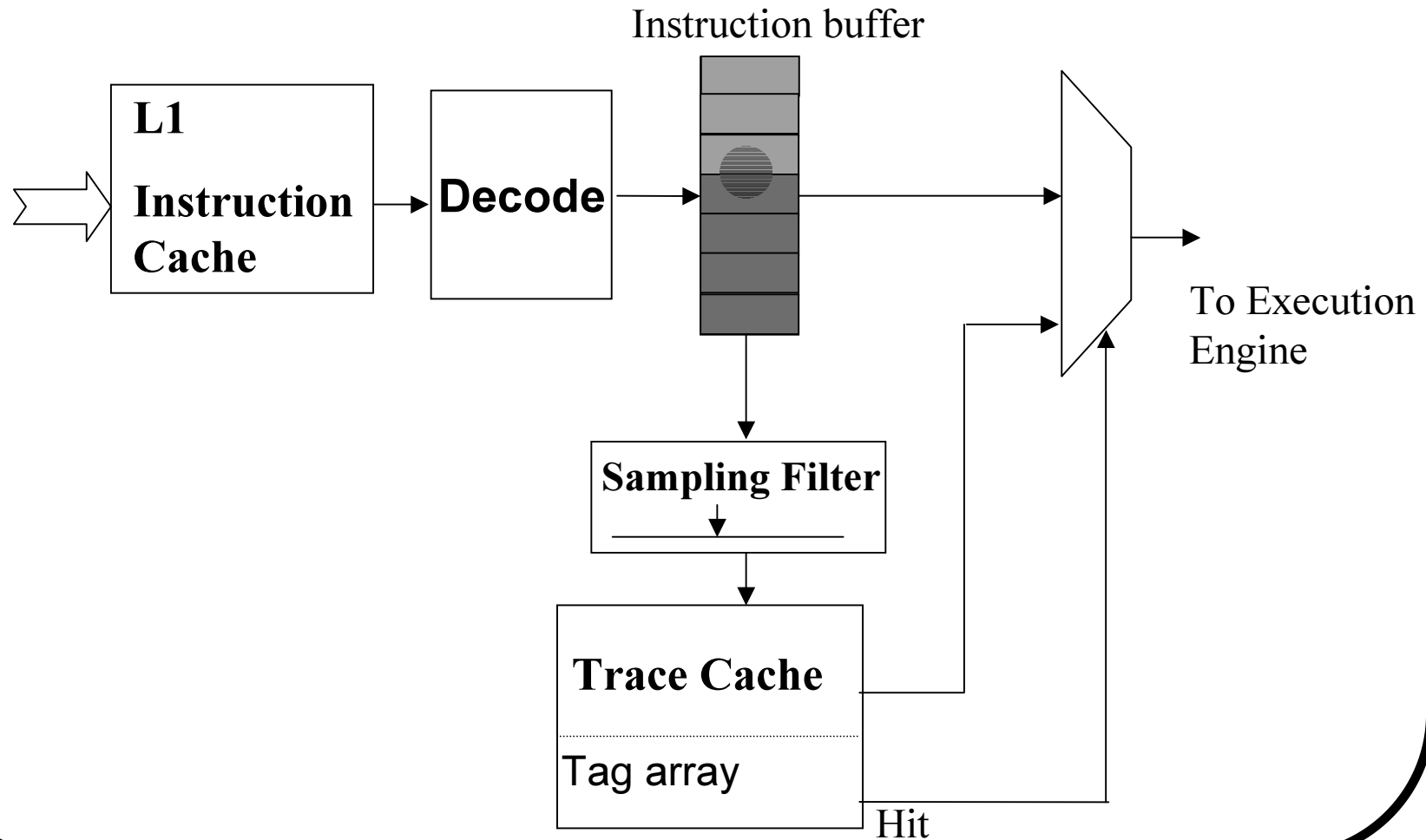


- 73% of insertion useless
- Only 10% of insertions result with more than two hits

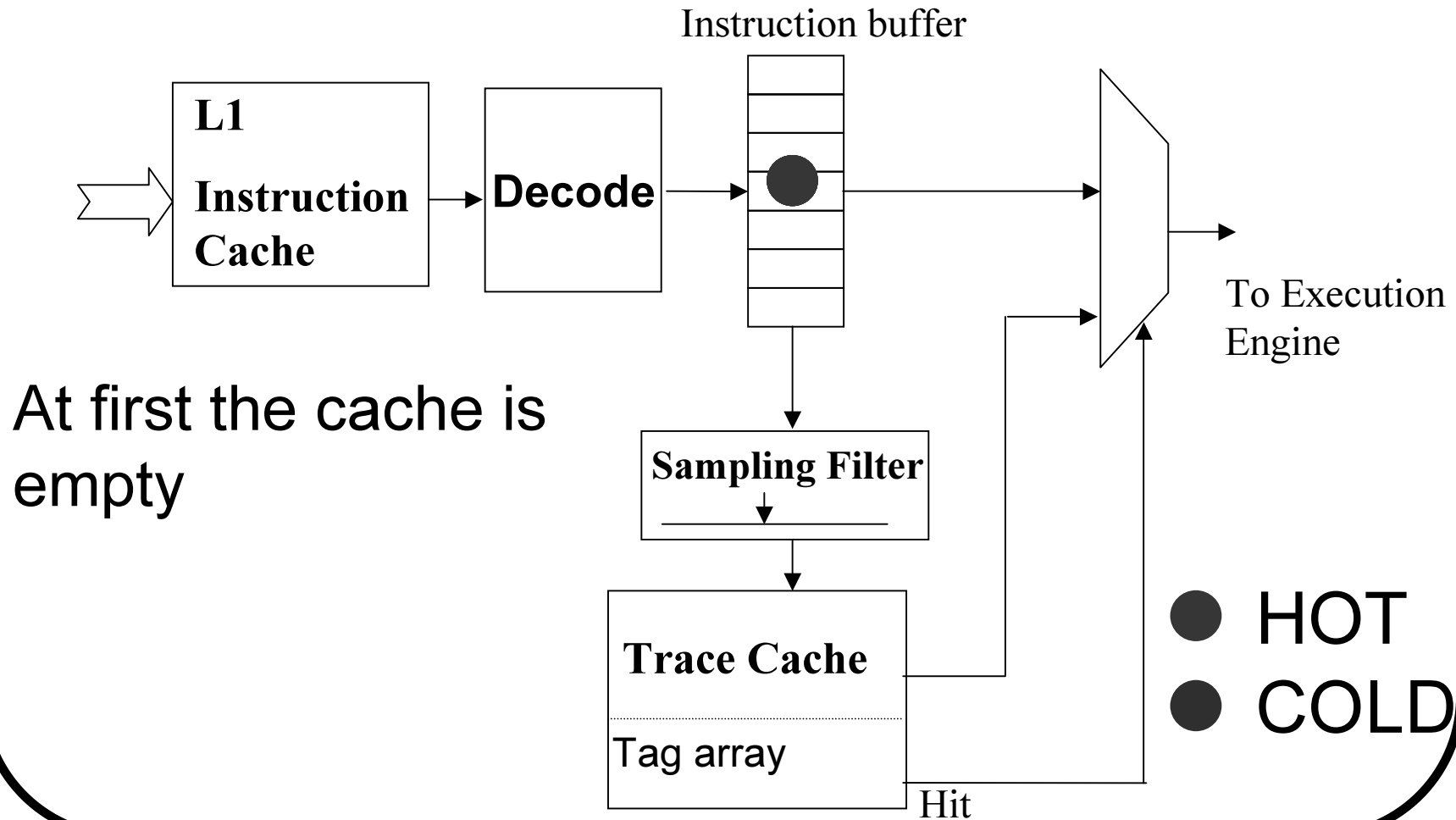
\*for a 32-traces Trace Cache



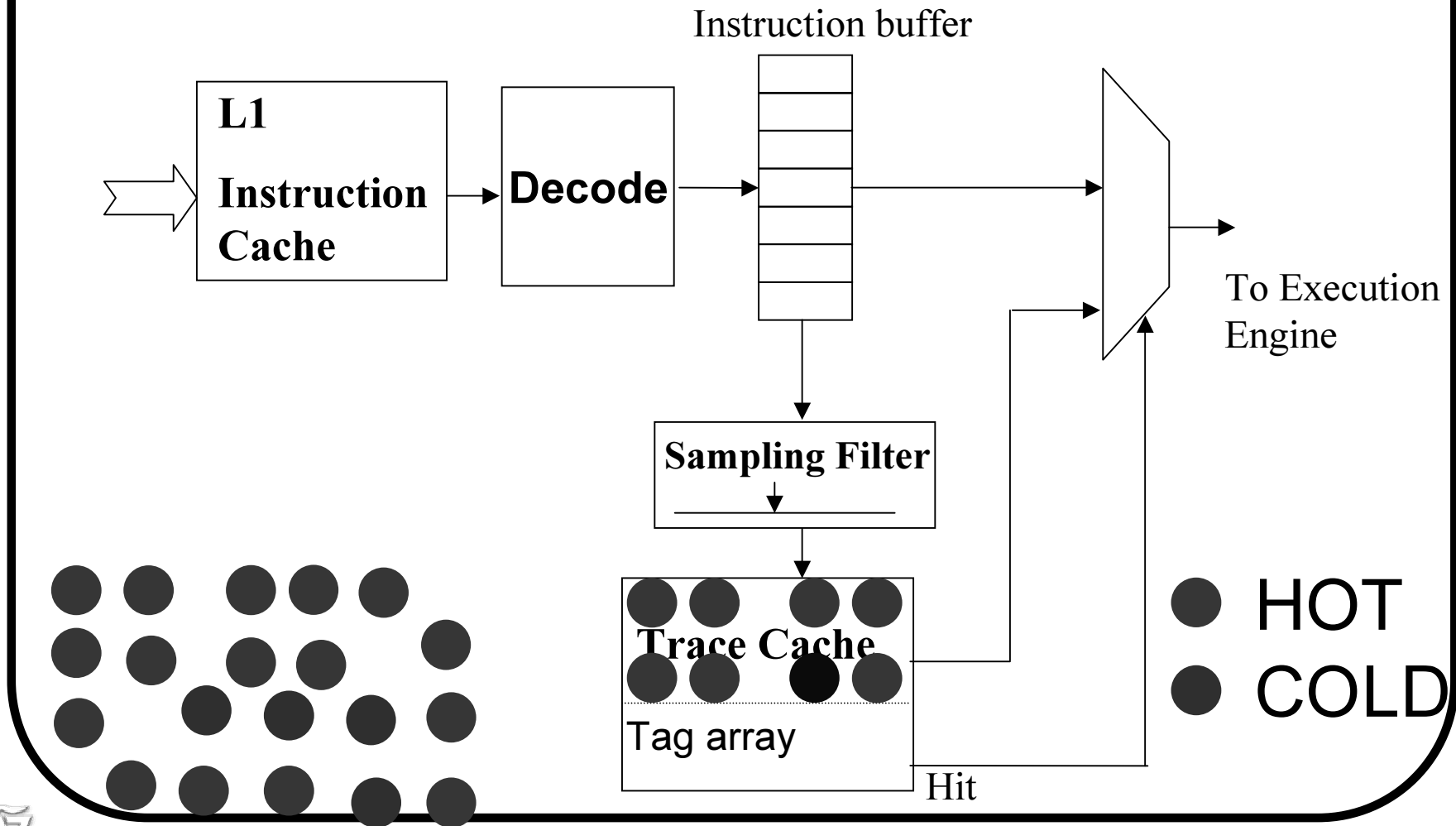
# The Sampling Filter



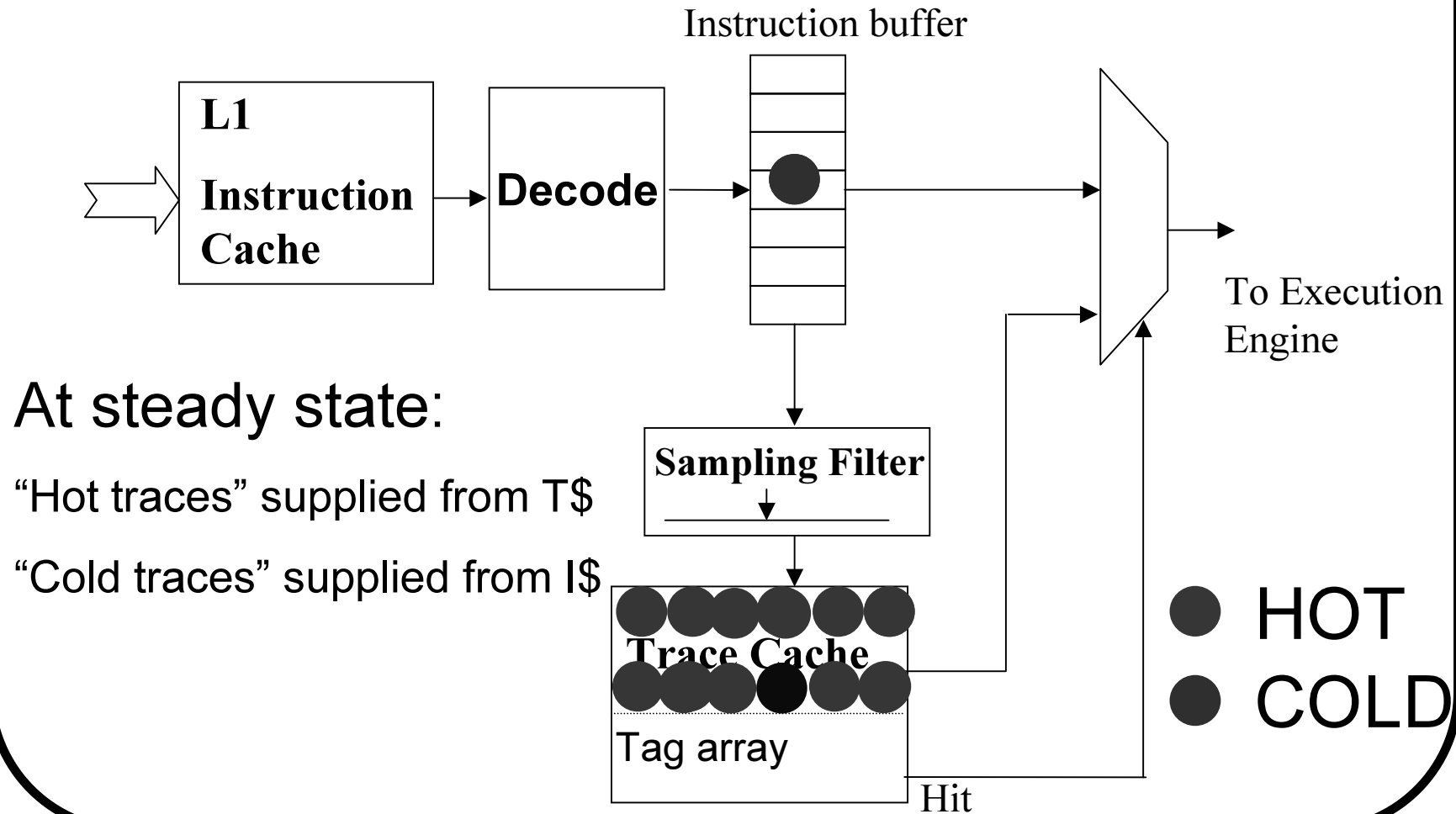
# Catching Frequent Traces



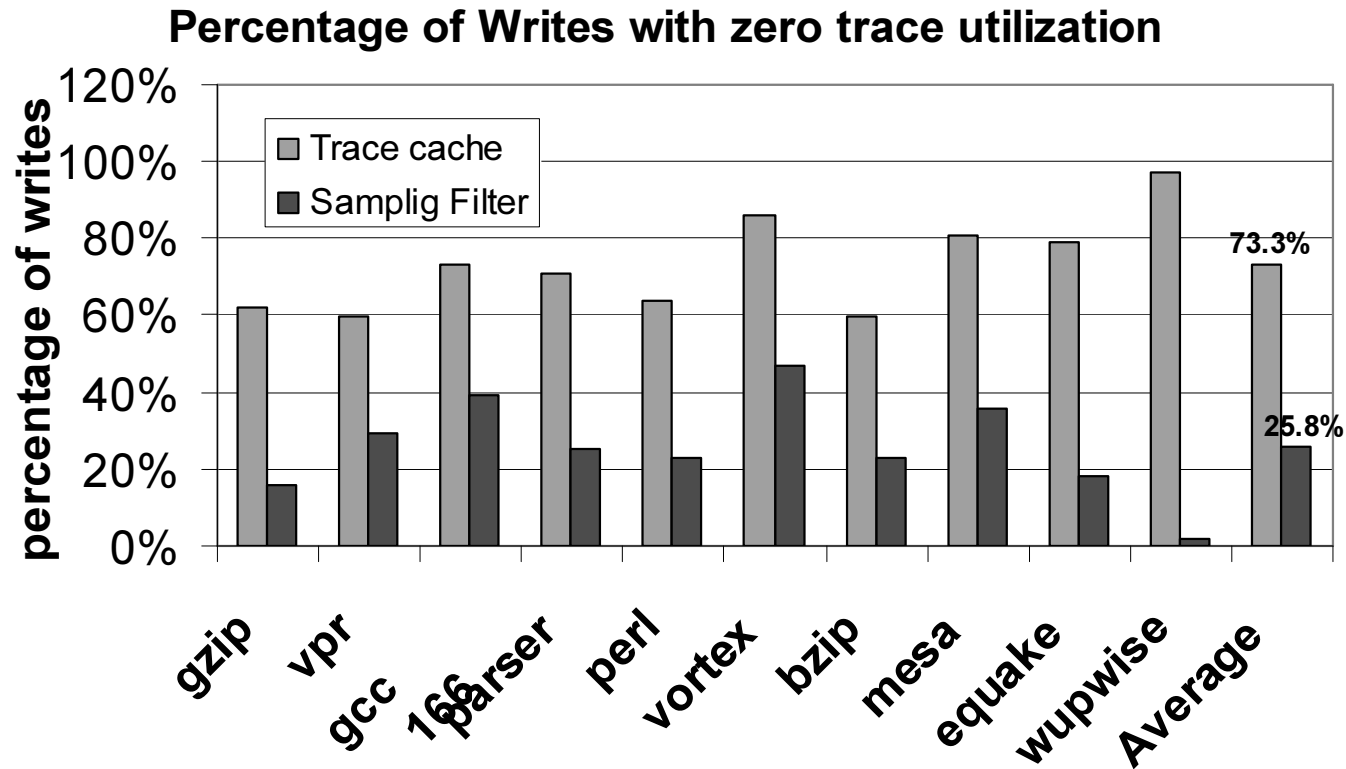
# Catching Frequent Traces



# At Steady State



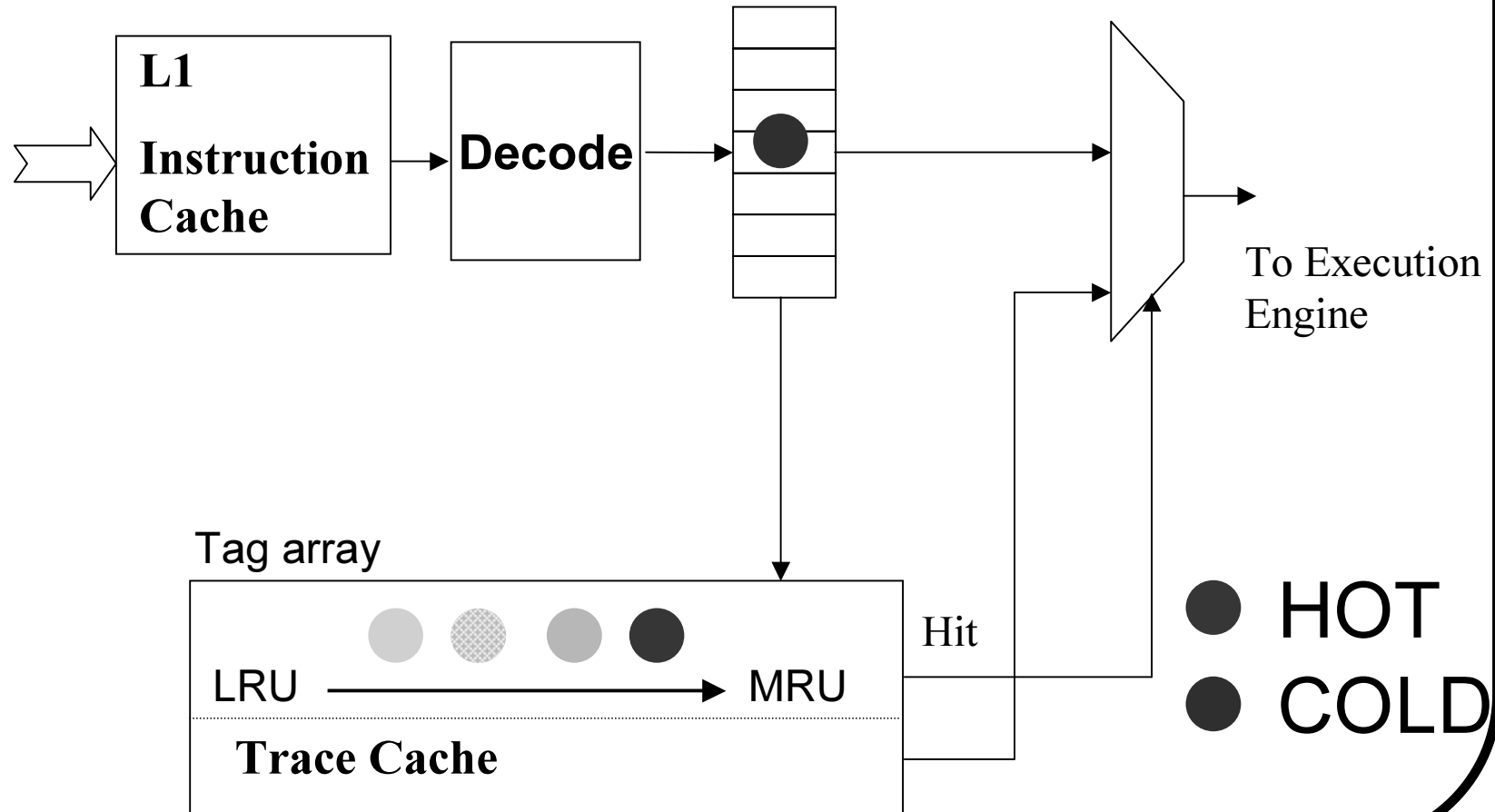
# Sampling reduces useless writes



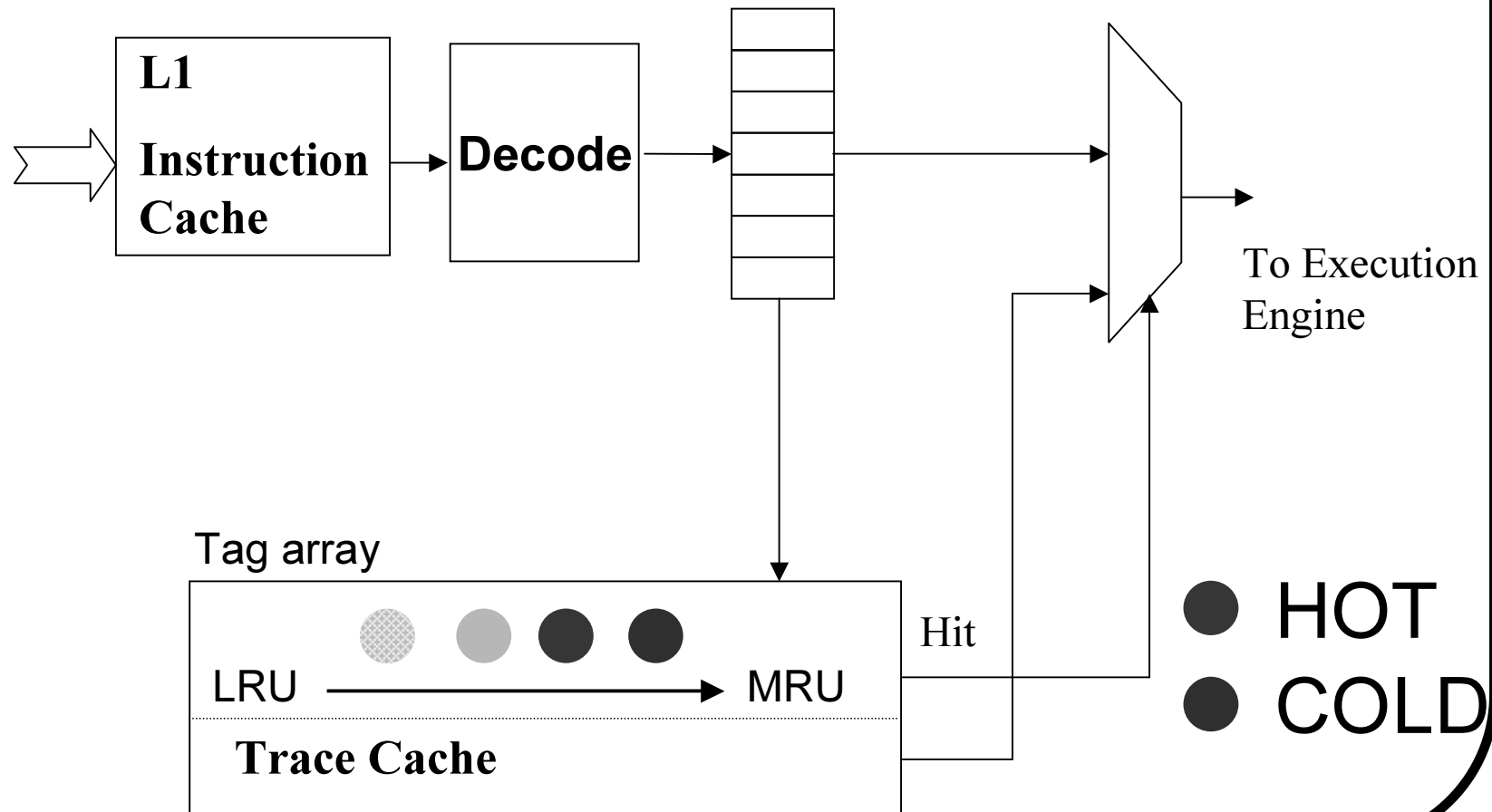
Trace utilization: the number of time a trace is used prior to its eviction from the cache.



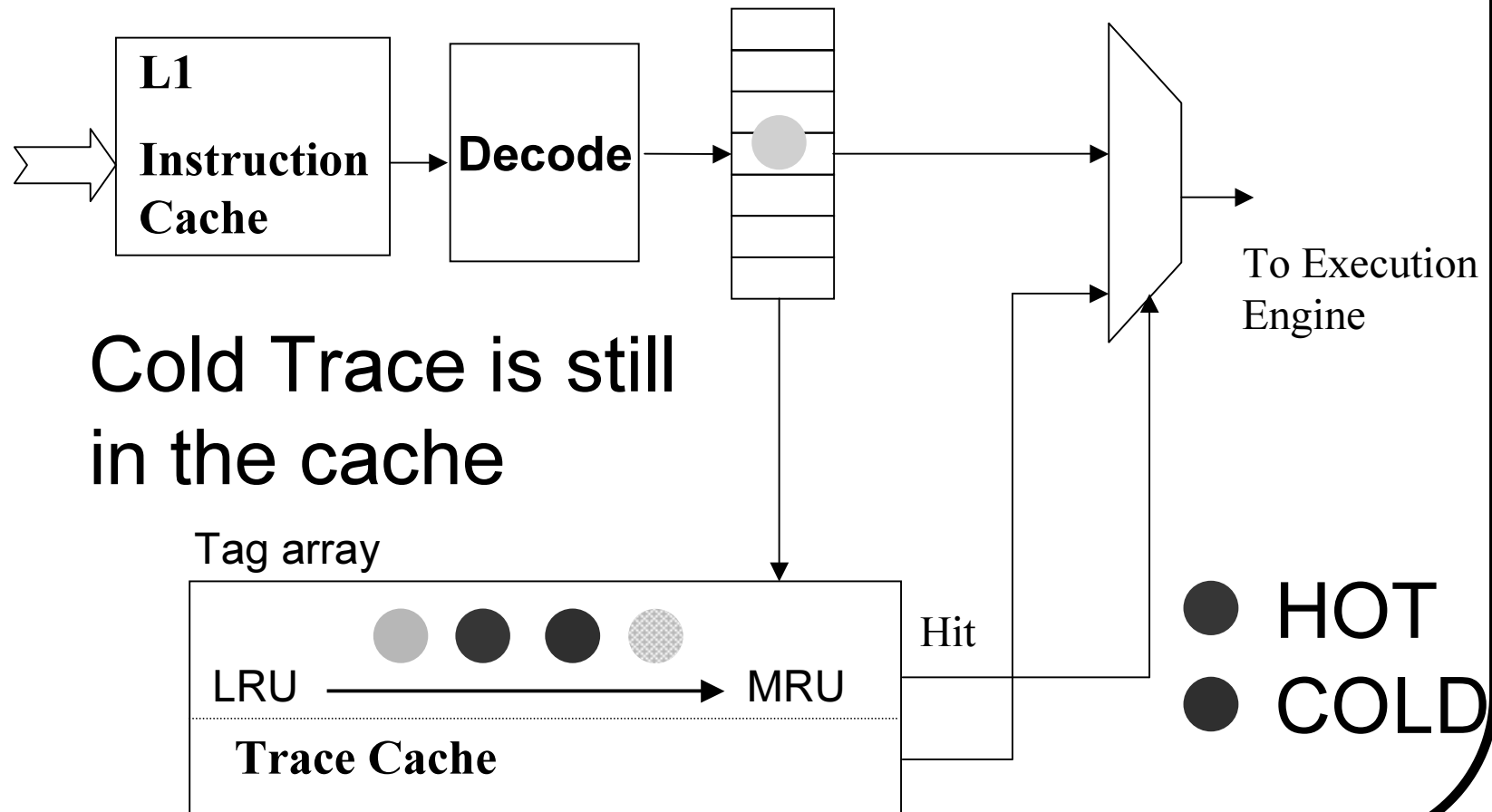
# LRU Replacement Mechanism



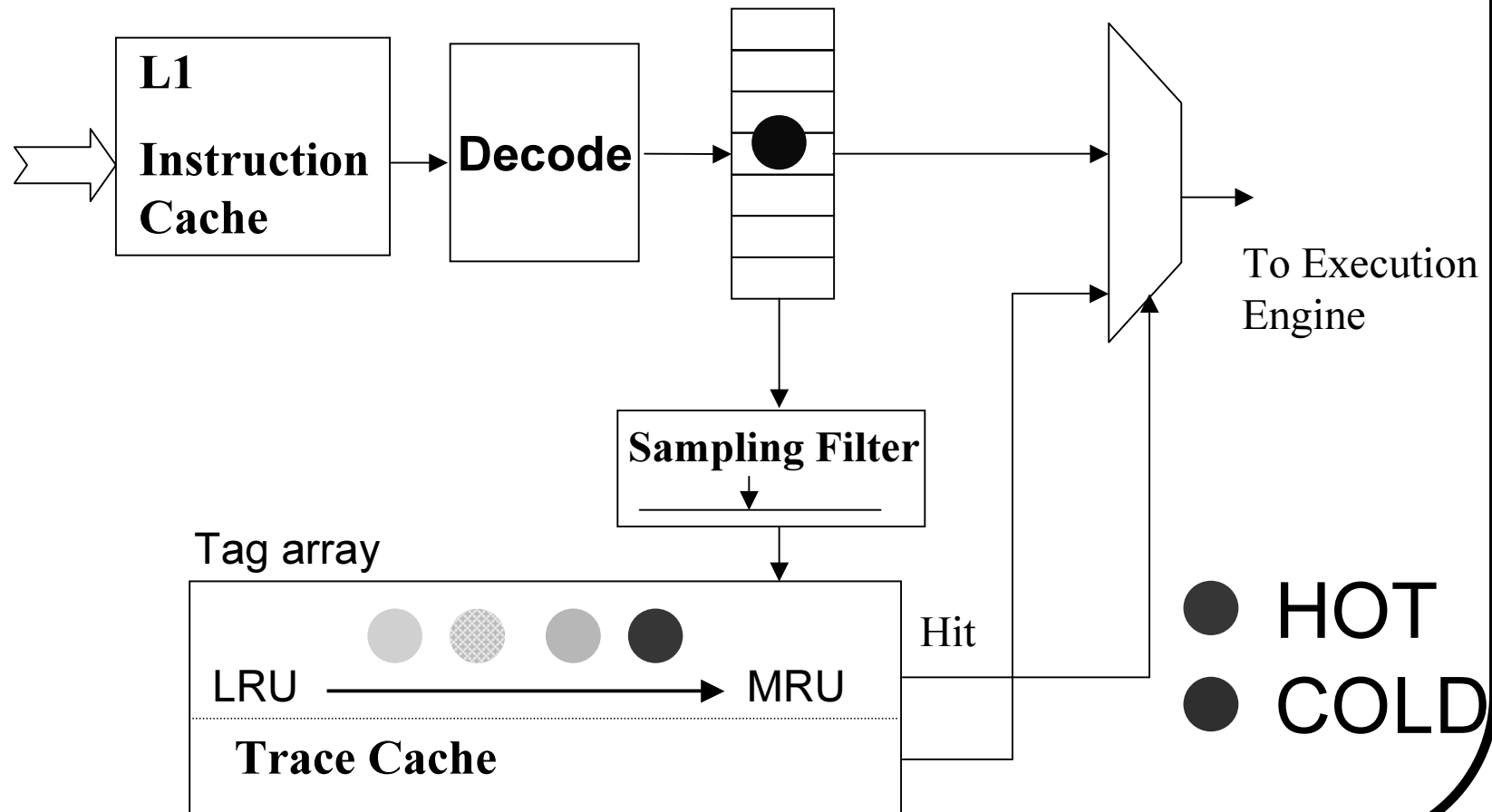
# LRU Replacement Mechanism



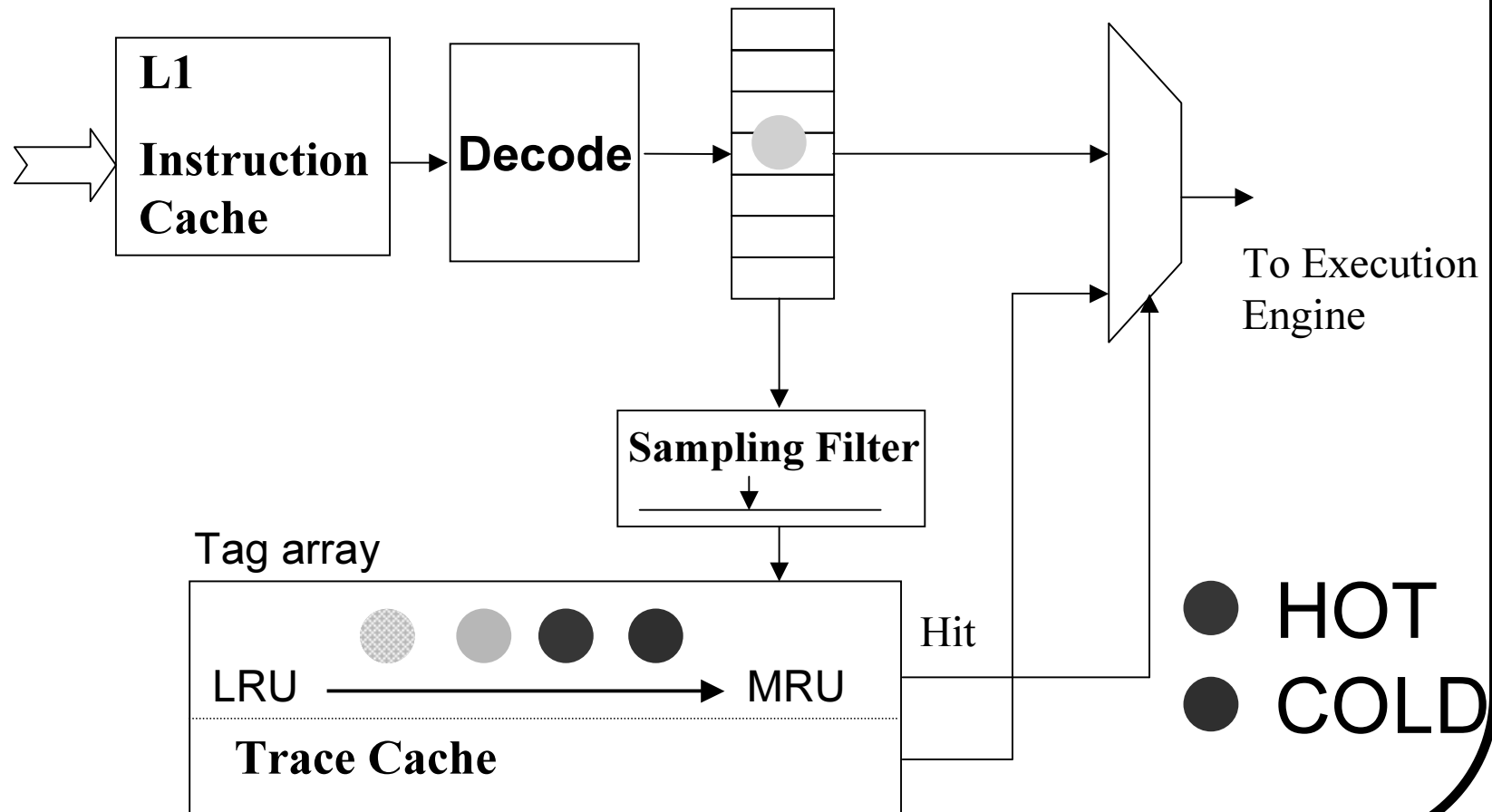
# LRU Replacement Mechanism



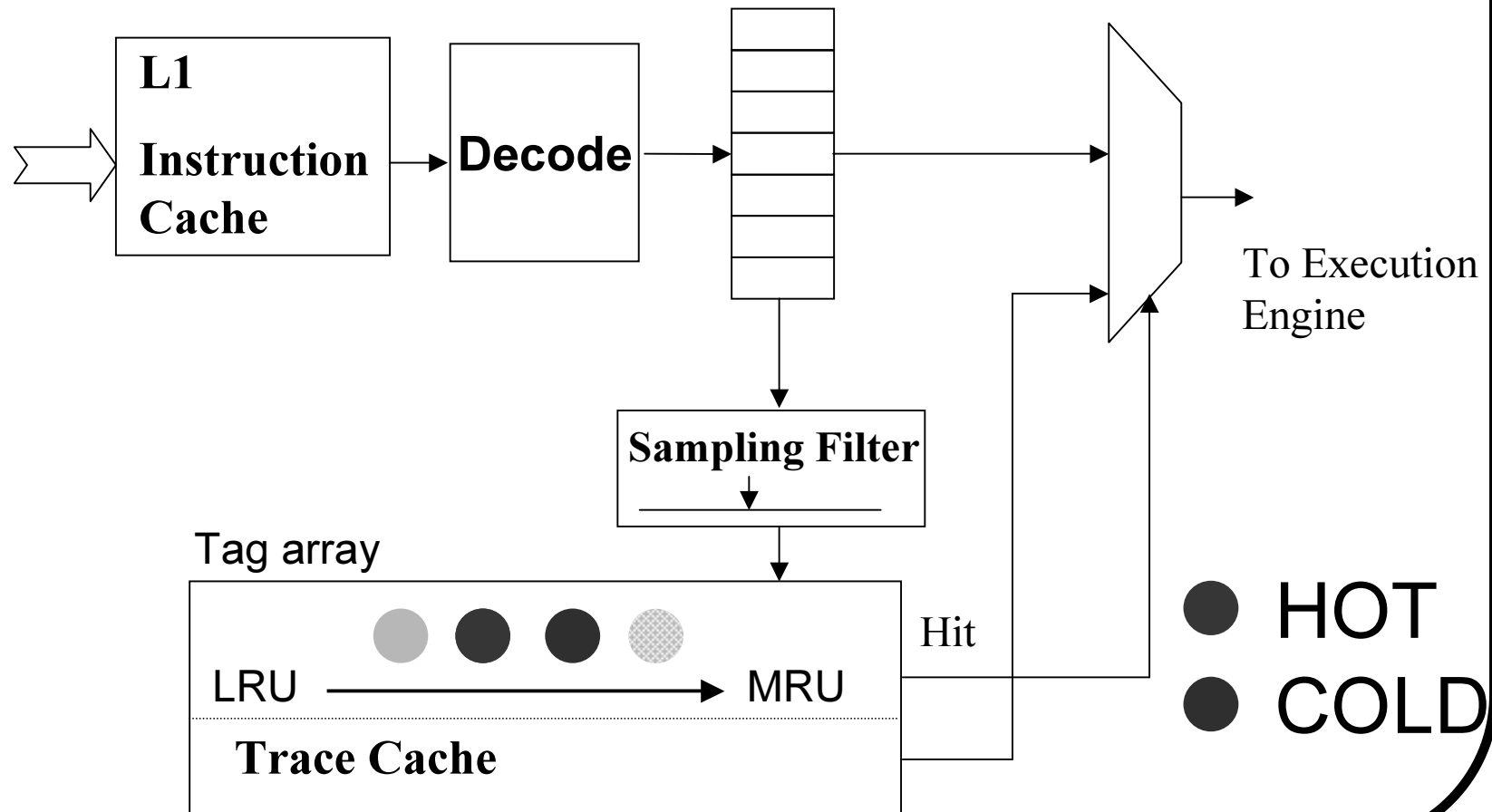
# LRU with a Sampling Filter



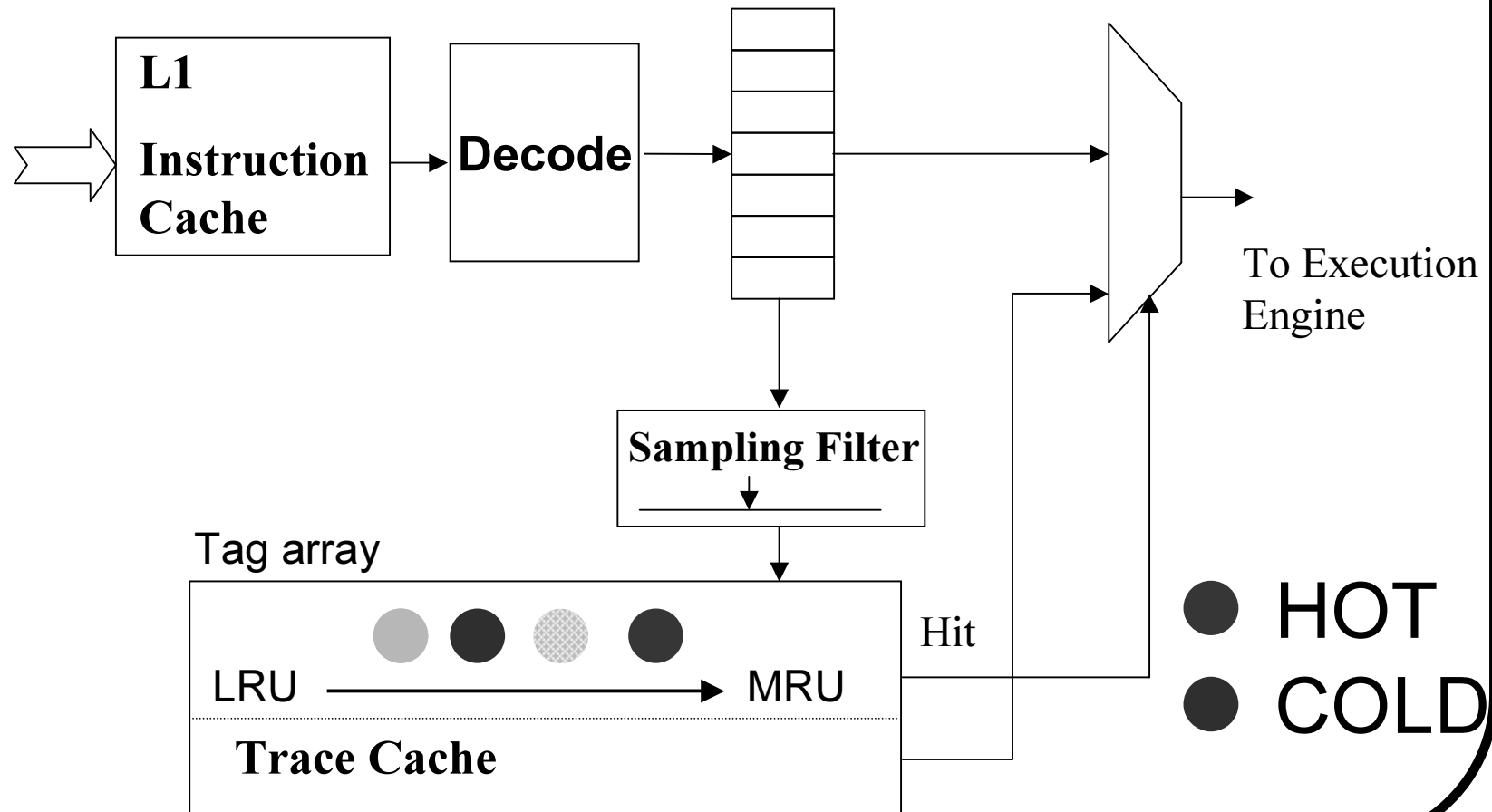
# LRU with a Sampling Filter



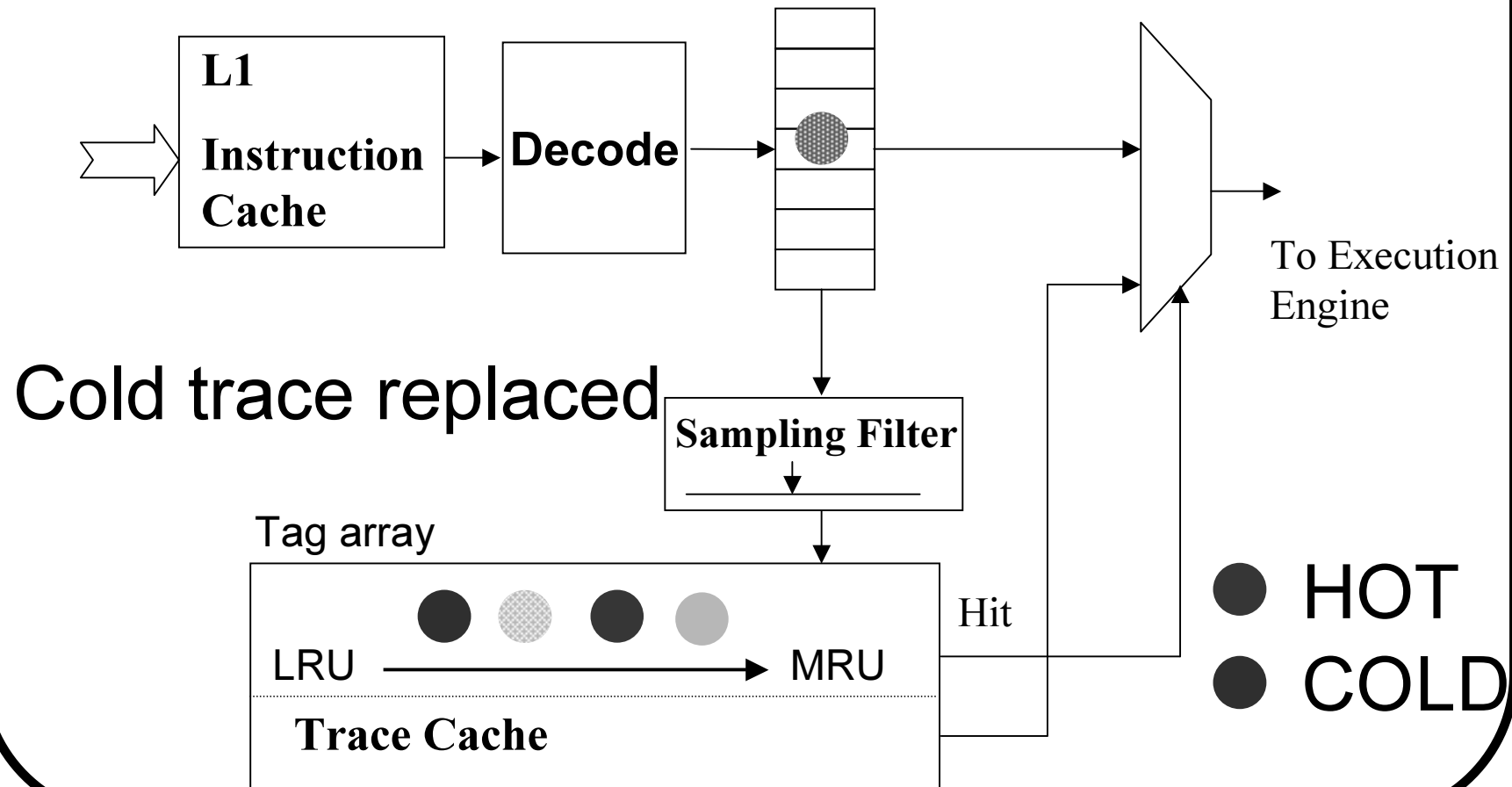
# LRU with a Sampling Filter



# LRU with a Sampling Filter



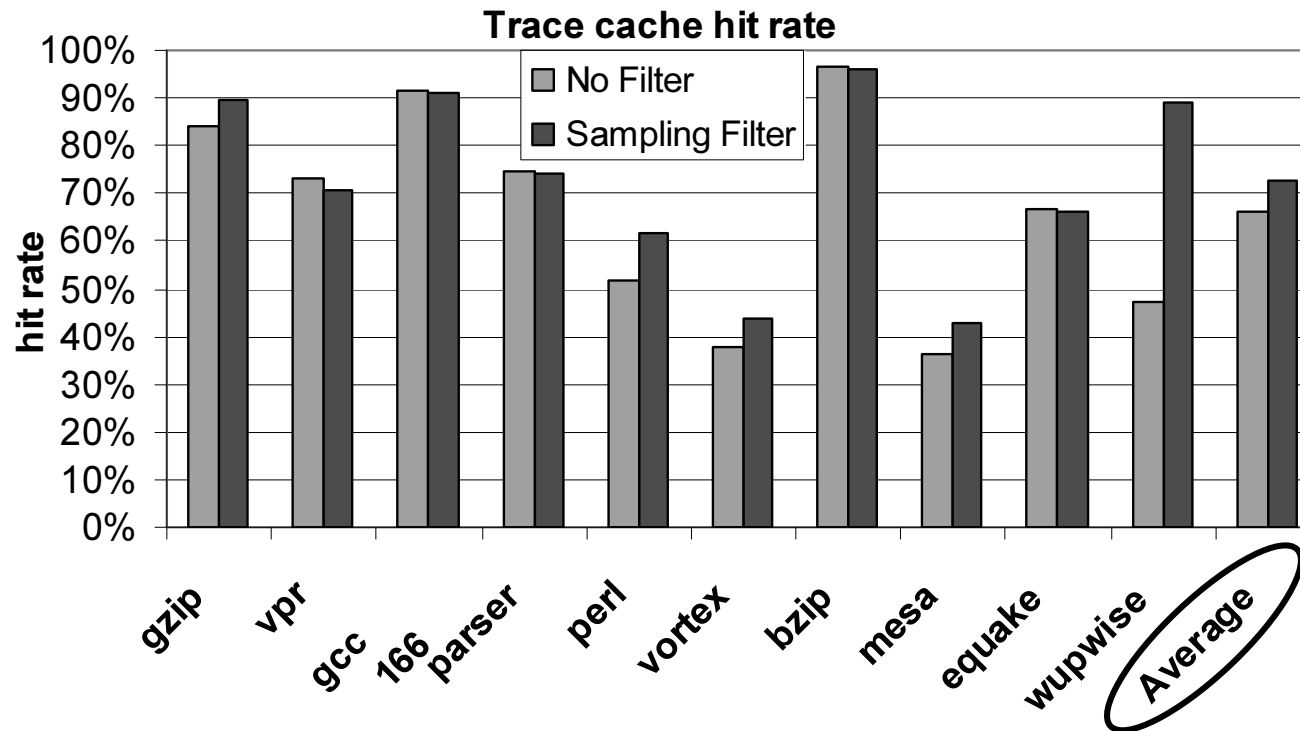
# LRU with a Sampling Filter



# L1 Caches Decoupling

- Traces retained longer in the cache
- Mirrored code in the Instruction cache is replaced by the LRU.
- Optimally:**
  - “Hot code” in the Trace cache
  - “Cold code” in the Instruction cache

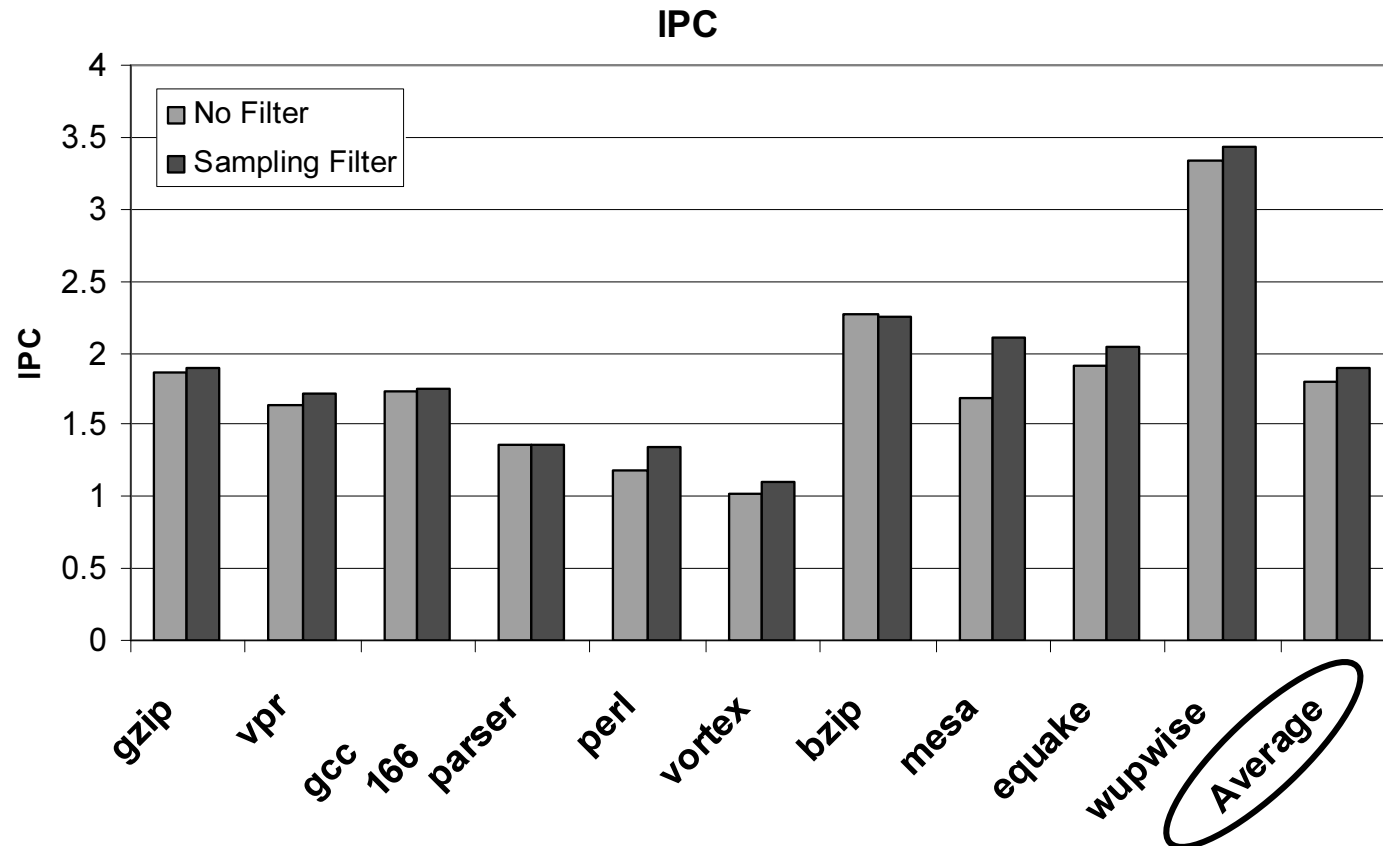
# Sampling Filter Hit Rate



The Average hit rate increases from 66% to 72.6%.



# Sampling Filter IPC



Average IPC is increased by 6%

# Sampling Filter Effect On Builds

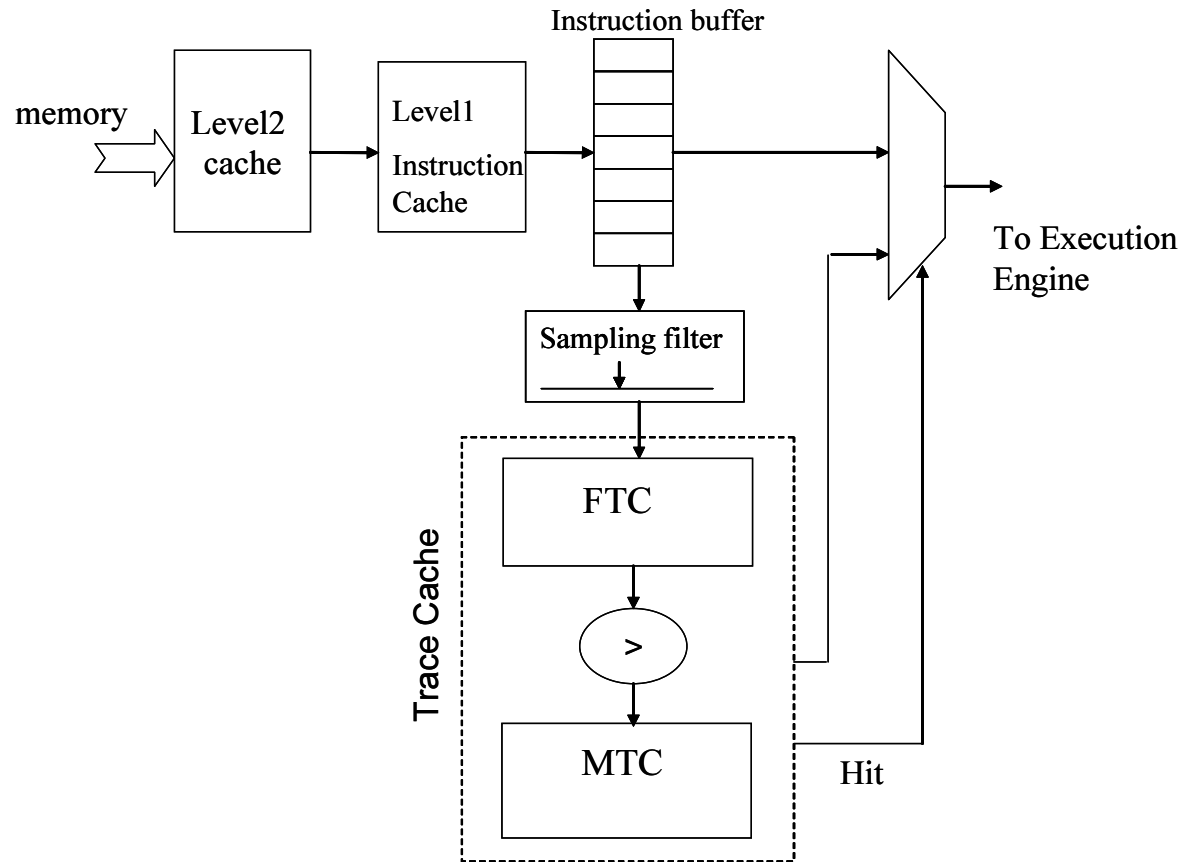
Build per 1000 retired instructions			
Benchmark	No-Filter	S.F.	Ratio
Gzip	22.54	0.80	28.1
Vpr	34.69	1.84	18.8
Gcc166	7.06	0.38	18.8
Parser	32.20	1.76	18.3
Perl	7.89	0.30	26.5
Vortex	57.55	2.59	22.2
Bzip	4.37	0.28	15.7
Mesa	53.70	2.38	22.6
Equake	35.77	1.57	22.7
Wupwise	34.84	0.36	96.0
Average	29.06	1.23	29.0

Utilization (Hit/Insertion)			
Benchmark	No-Filter	S.F.	Ratio
Gzip	5.13	160.69	31.3
Vpr	2.45	41.65	17.0
Gcc166	10.37	191.43	18.5
Parser	2.84	51.13	18.0
Perl	1.19	36.84	31.0
Vortex	0.57	14.4	25.3
Bzip	28.1	435	15.5
Mesa	0.59	15.02	25.5
Equake	1.97	39.77	20.2
Wupwise	0.9	162.39	180.4
Average	5	115	21.2

@ Sampling Rate 1/20



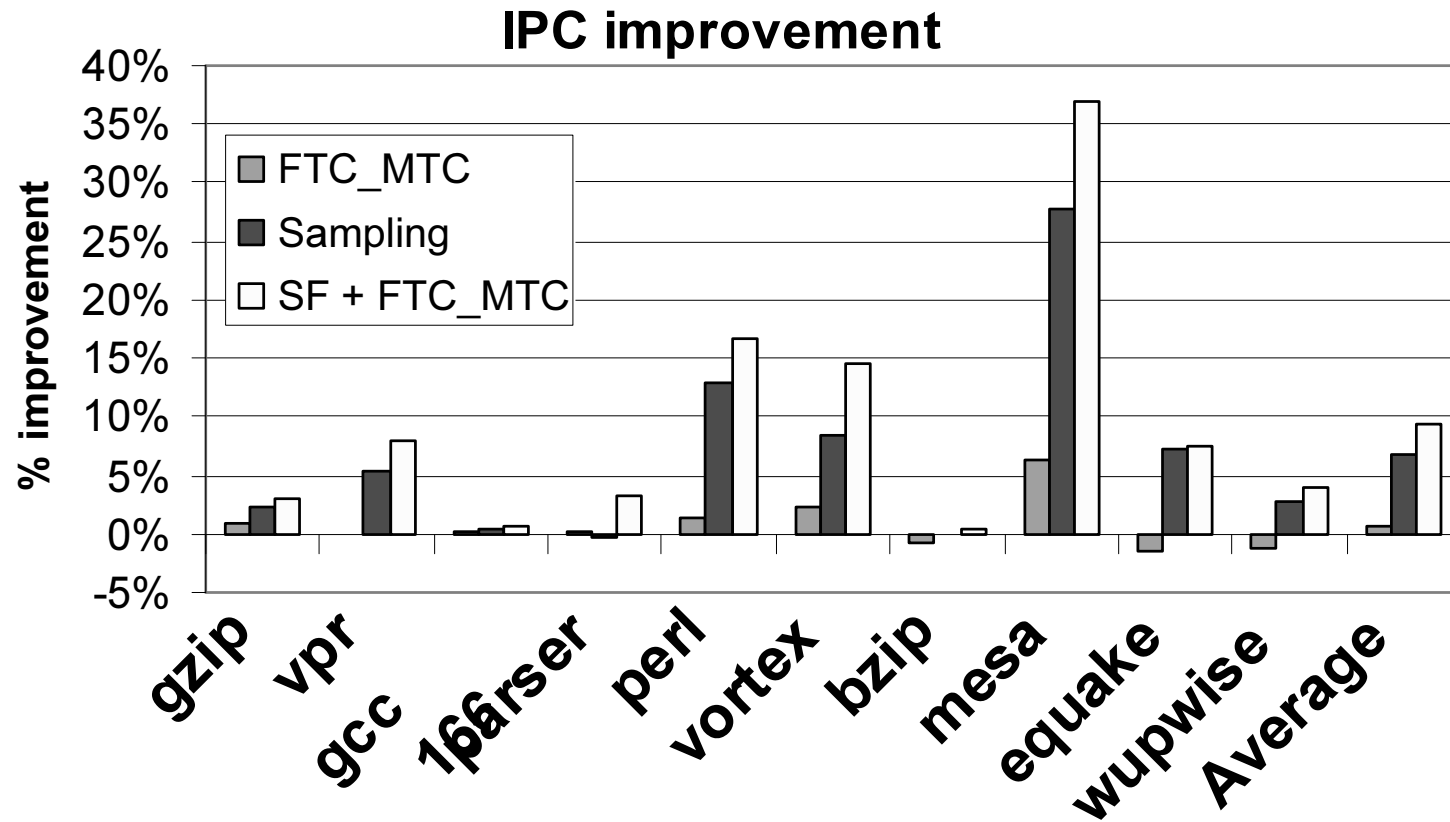
# Sampling Filter with The MTC-FTC



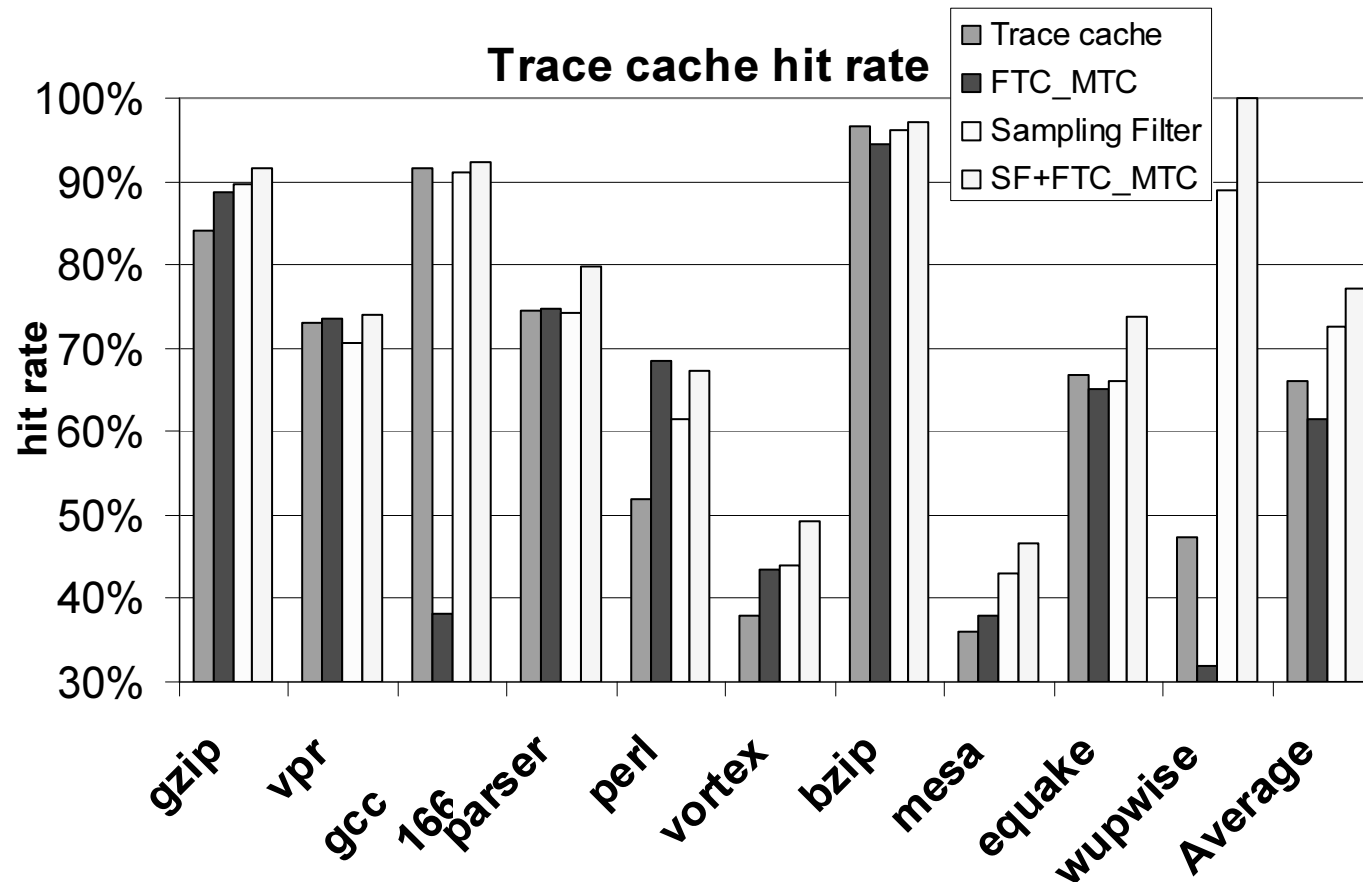
# Sampling Filter with The MTC-FTC

- SF Improves the Filter Trace Cache performance (FTC)
- Better FTC-MTC filter decision
- The “Hot Traces” are kept tightly in the cache.

# IPC



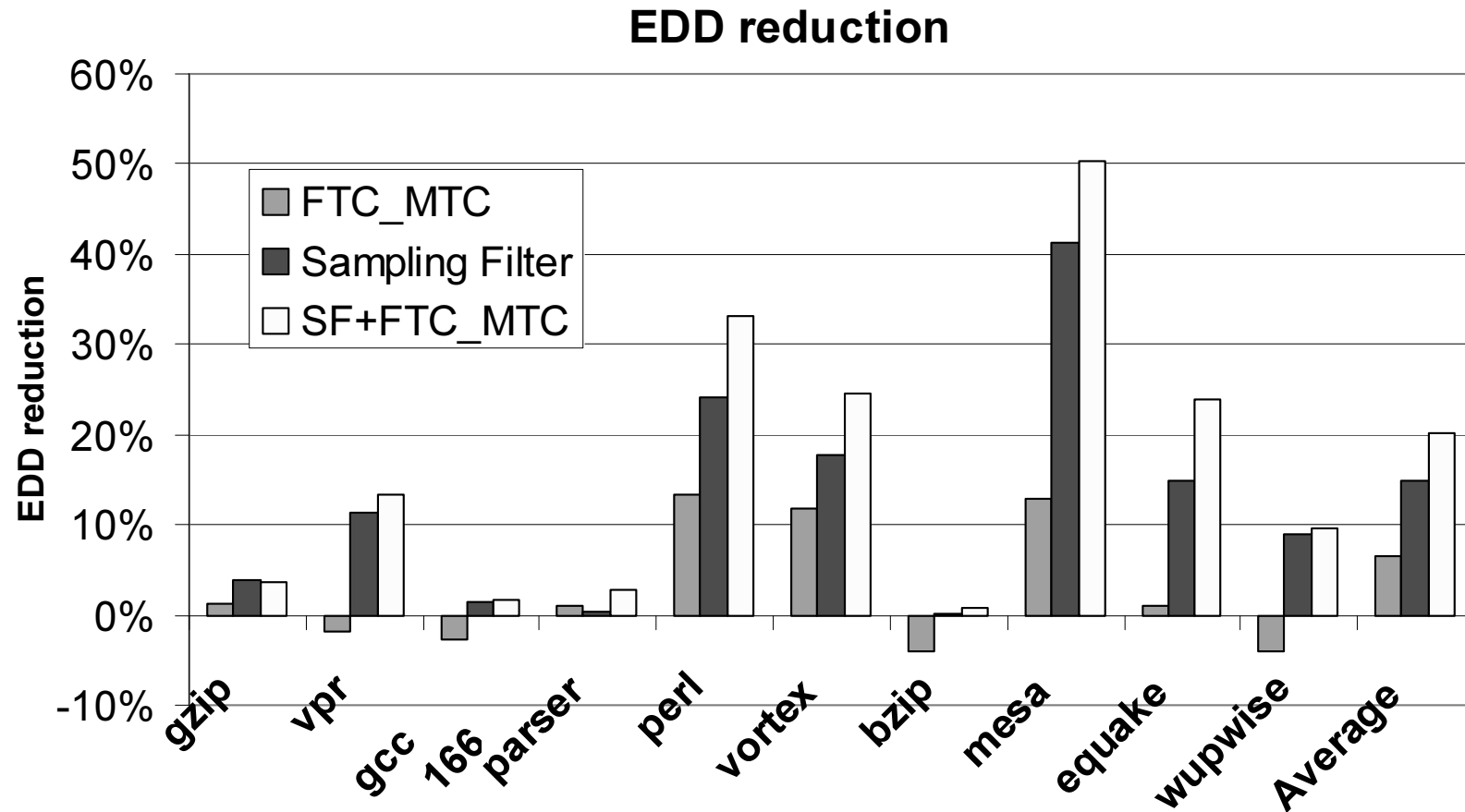
# Hit Rate



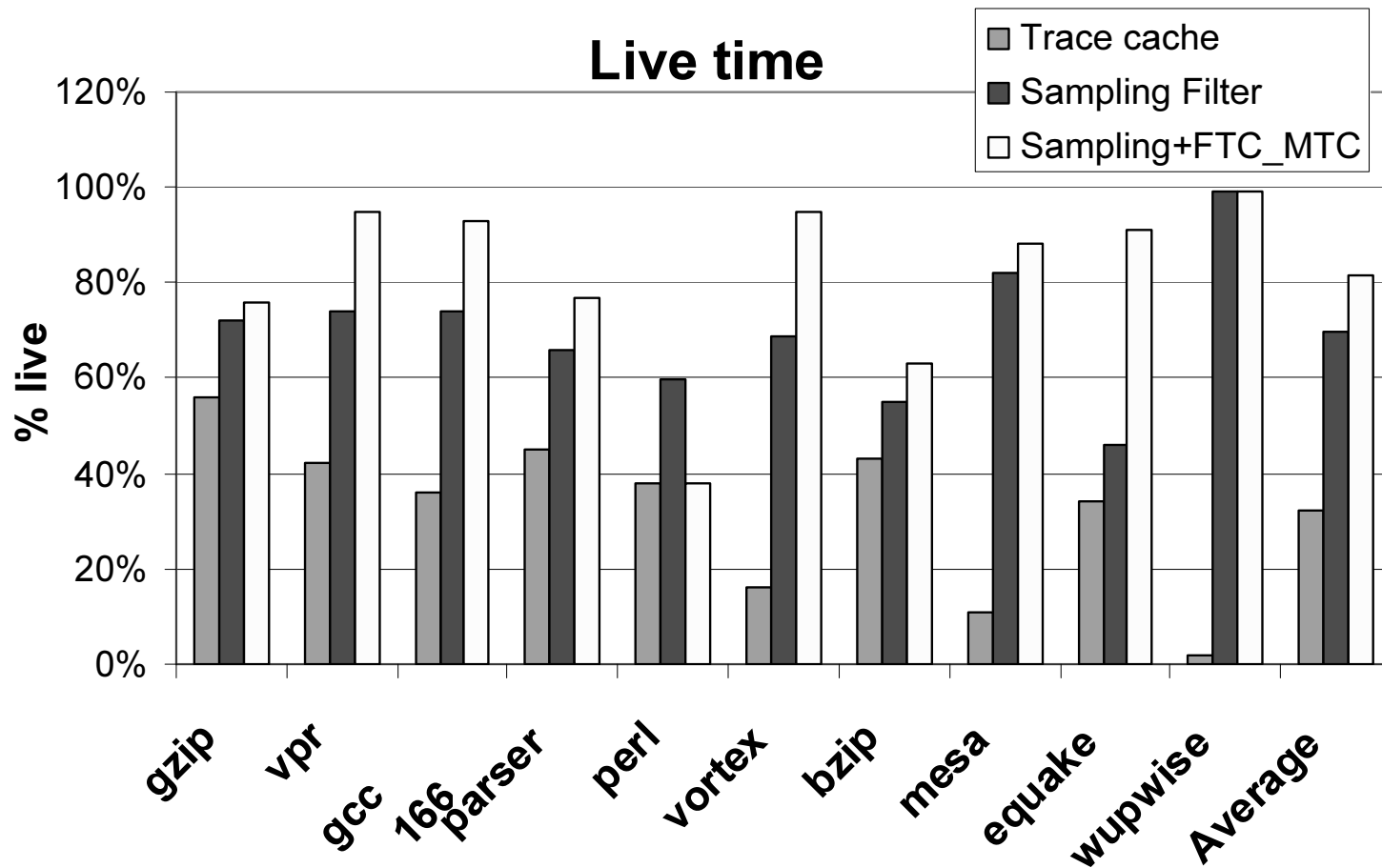
Average Hit rate improves from 66% to 77.2%



# Energy Delay Square Product



# Lifetime



# Conclusions

- “Hot/cold” Filtering can increase the trace cache utilization.
- A simple random selection of traces can improve the trace cache performance.
- Applicable for many architectural structures.

# Questions?



**The Bahai Temple and gardens - Haifa**