



# Trace Cache

Leon Gu  
Dipti Motiani

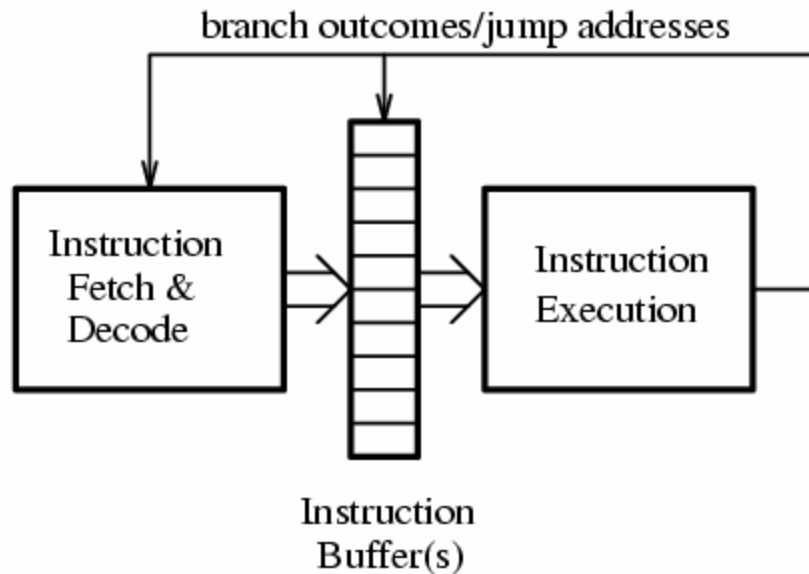
15-740: Computer Architecture, Fall, 2003

10/01/2003

# Papers

- Eric Rotenberg, Steve Bennett, and James E. Smith. *A Trace Cache Microarchitecture and Evaluation*, in *IEEE Transactions on Computers*, 48(2):111-120, February 1999.
- Bryan Black, Bohuslav Rychlik, and John Paul Shen. *The Block-based Trace Cache*, in *Proceedings of the 26th Annual International Symposium on Computer Architecture*, pages 196-207, May 1999.
- Michael Sung. *Design of Trace Cache for High Bandwidth Instruction Fetching*. Masters Thesis, May 1998
- Eric Rotenberg, Steve Bennett, Jim Smith. *Trace Cache: a Low Latency Approach to High Bandwidth Instruction Fetching*. April, 1996

# Superscalar Processors



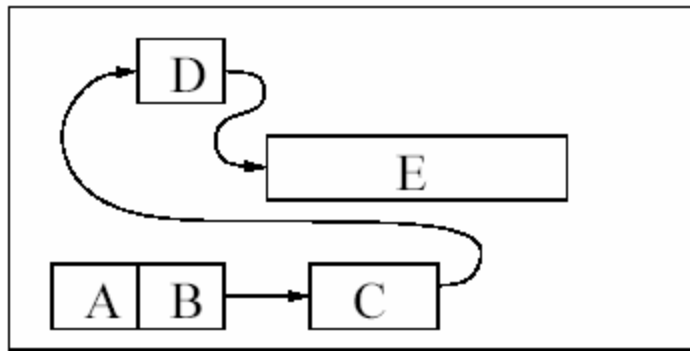
- Producer - Consumer
- Instruction Level Parallelism

# [ Motivation ]

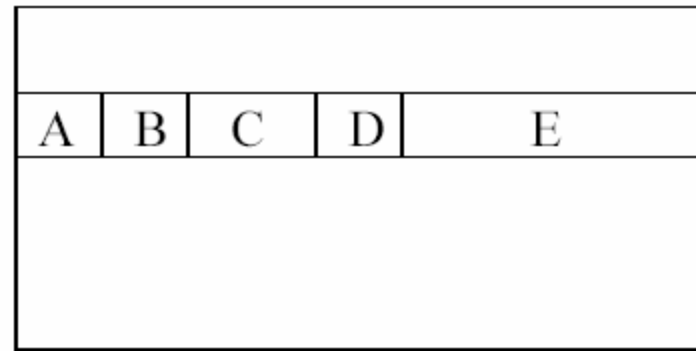
---

- Exploit I LP
- Fetch Bottleneck
  - Instruction cache misses
  - Branch prediction accuracy
  - Branch prediction throughput
  - Noncontiguous instruction fetching
  - Fetch unit latency

# [ Trace Cache ]



(a) Instruction cache.

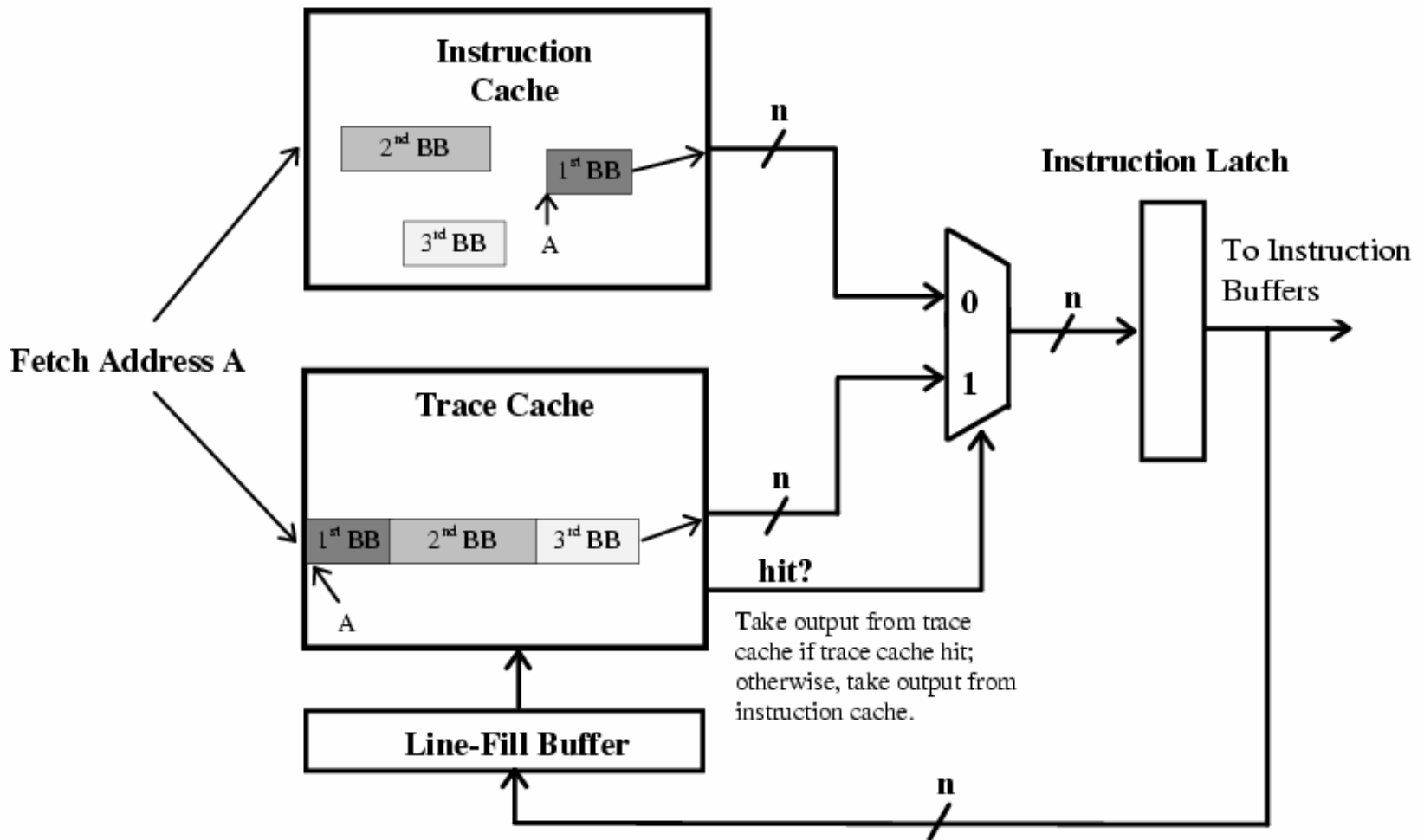


(b) Trace cache.

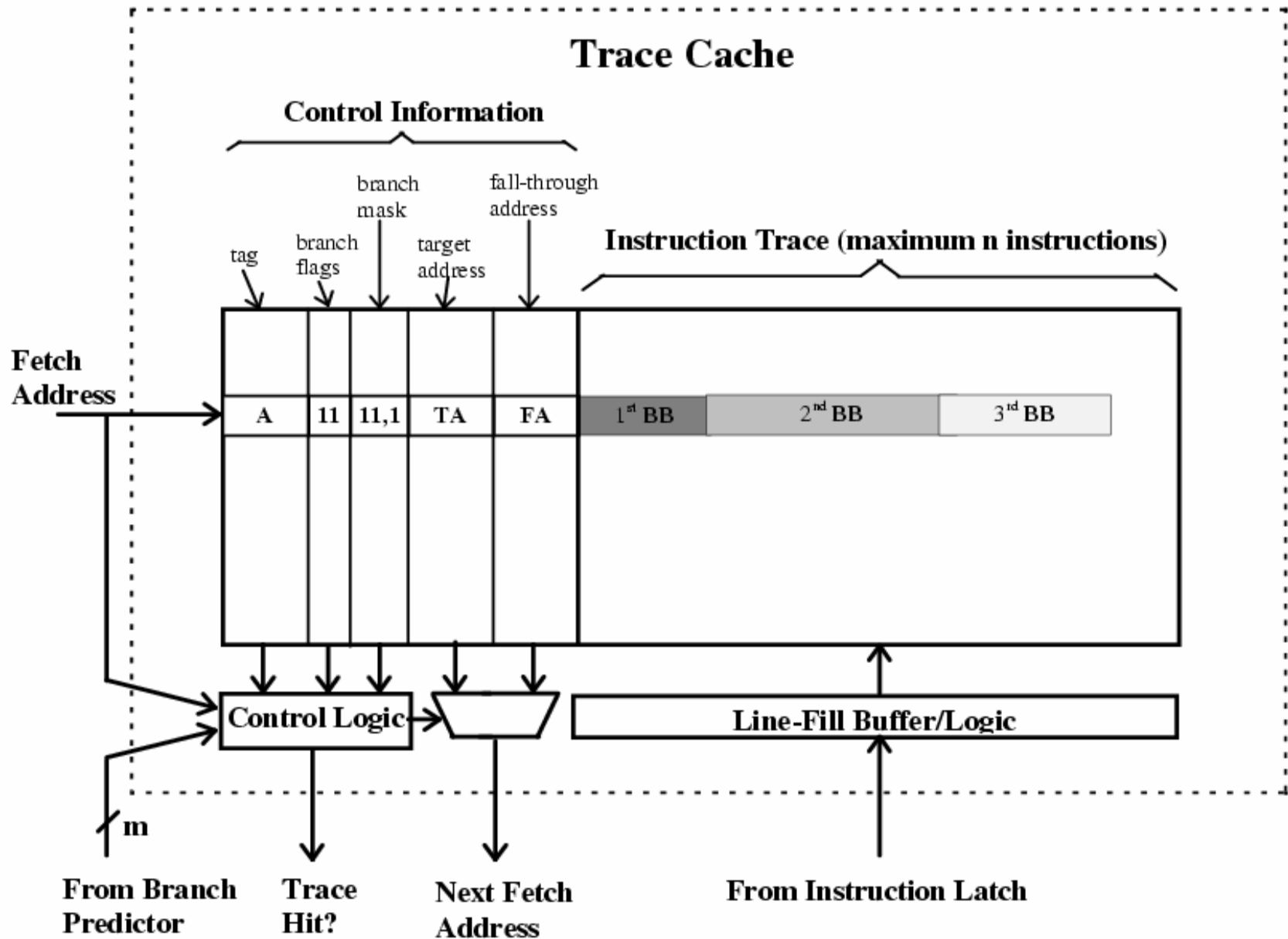
- A trace is a sequence of instructions starting at any point in a dynamic instruction stream.
- It is specified by a **start address** and the **branch outcomes** of control transfer instructions.

# Fetch Mechanism

- Trace cache is accessed in parallel with instruction cache.
  - Hit → Trace read into issue buffer
  - Miss → Fetch from instruction cache
- Trace cache hit if
  - Fetch address match
  - Branch predictions match
- Trace cache is NOT on the critical path of instruction fetch.



## Fetch Mechanism





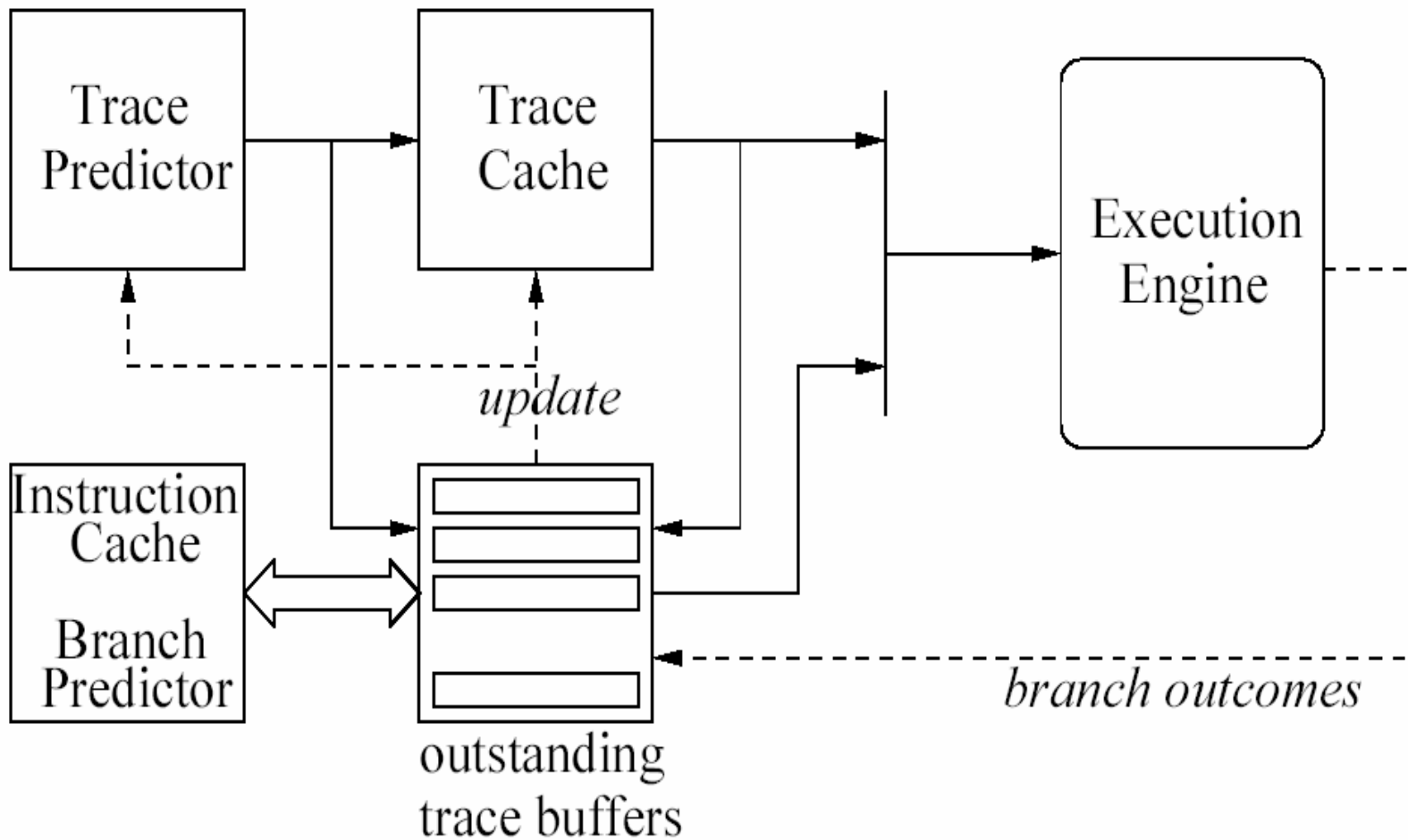
# [ Design Issues ]

- Trace Length
- Sizing
- Indexing
- Branch Throughput
- Fill Mechanism
- Partial Matches
- Associativity
- Replacement Policy

# [ Paper 1 ]

---

- Present a micro-architecture incorporating a trace cache
  - Control flow prediction and instruction supply at trace level
- Evaluate performance advantage
- Design issues – size and associativity



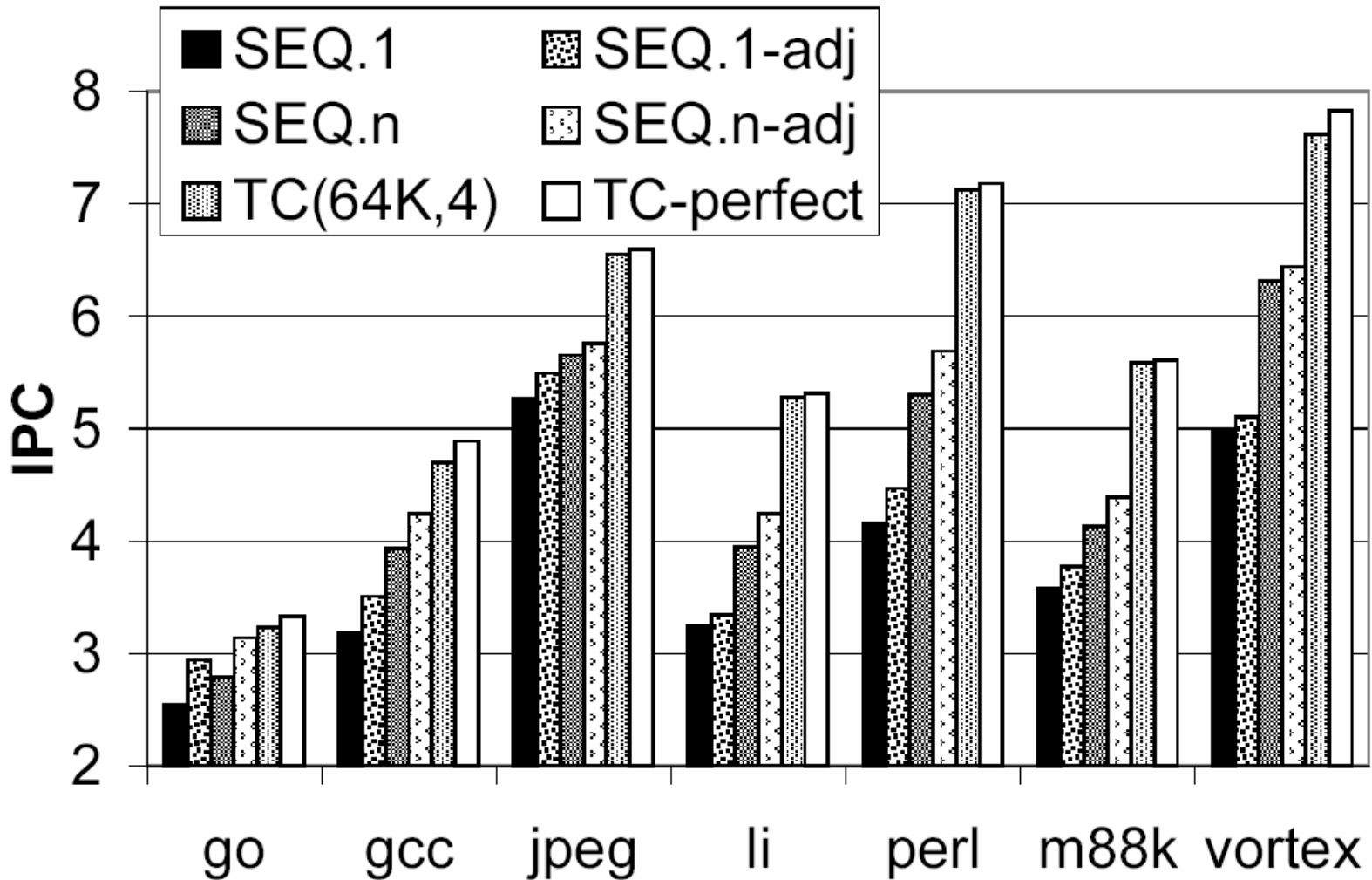
## Microarchitecture

# [ Microarchitecture ]

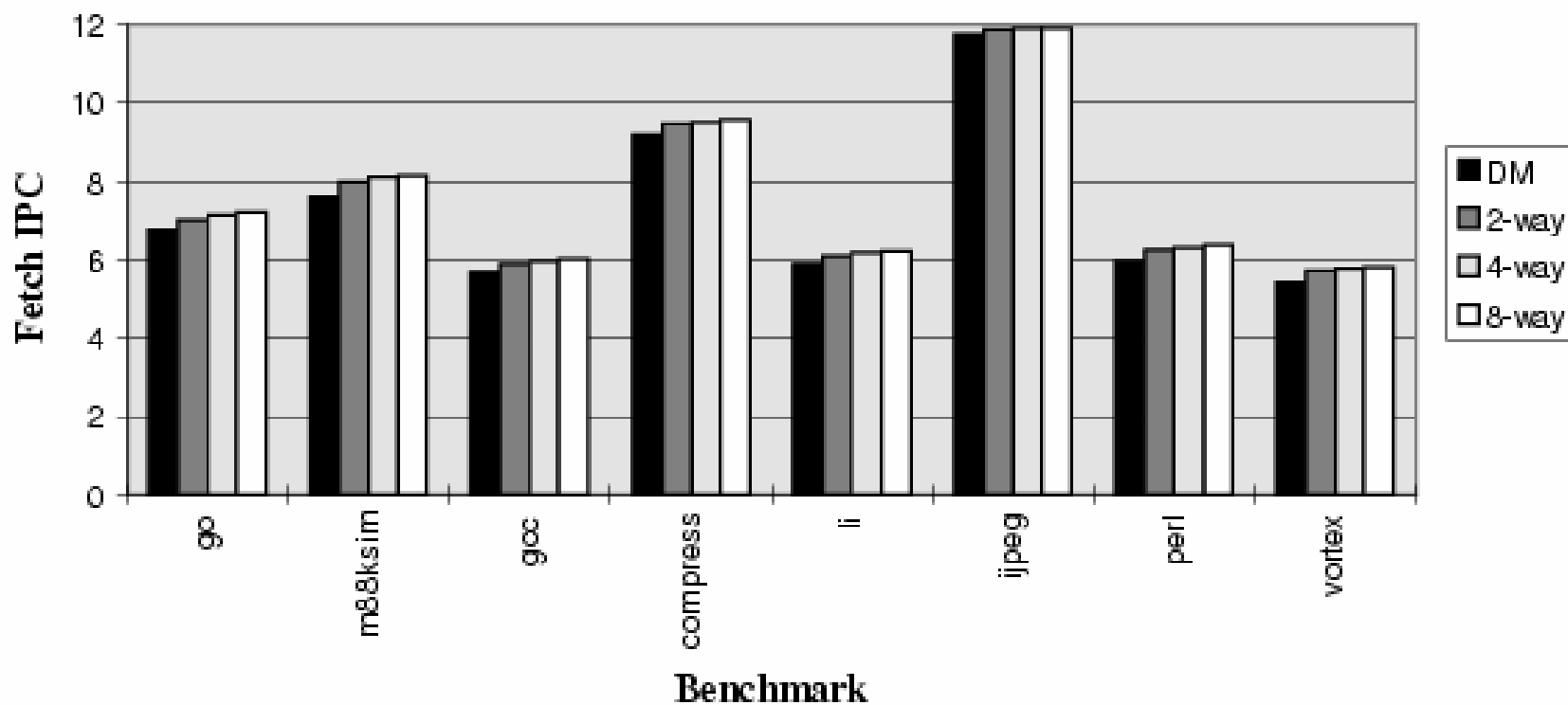
---

- Trace-level sequencing
- Instruction-level sequencing
- Next trace prediction
- Trace selection
- Hierarchical sequencing

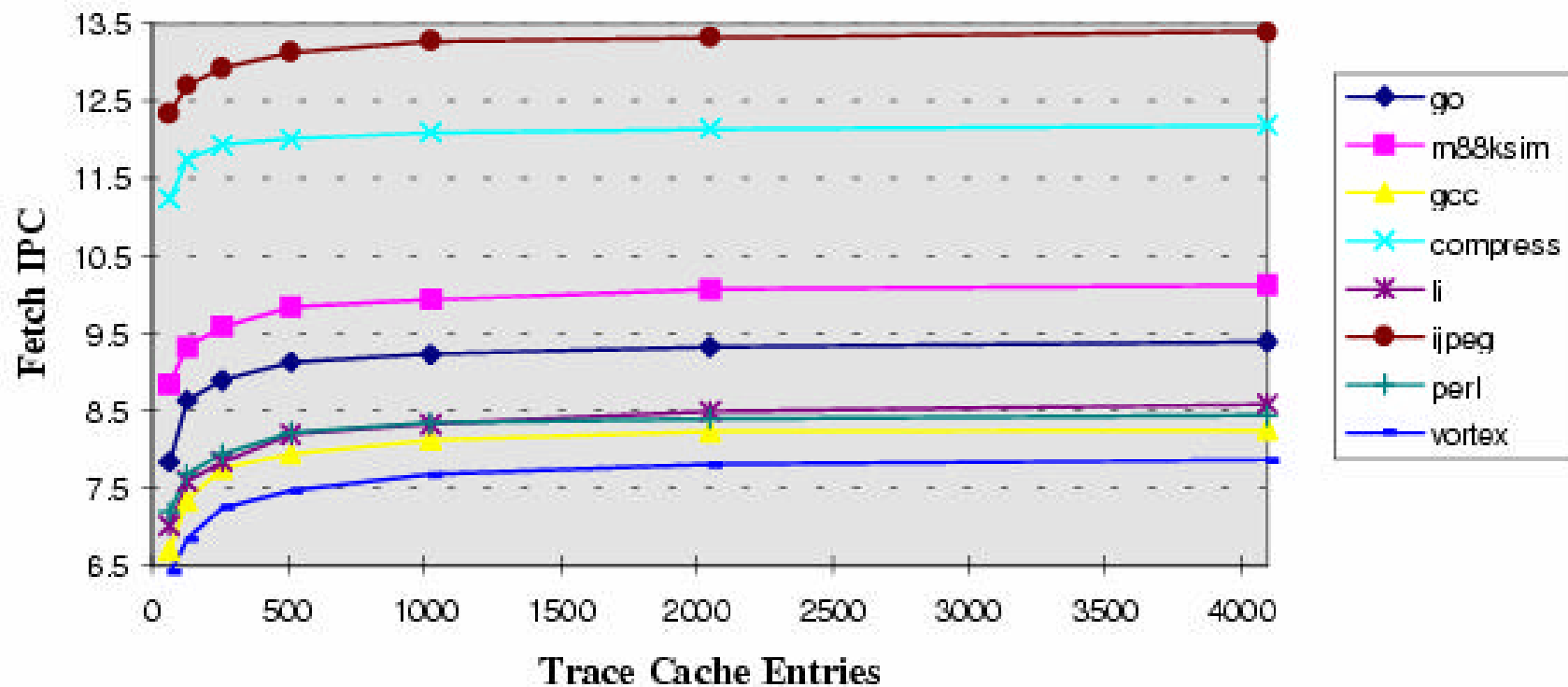
# Performance of Fetch Models



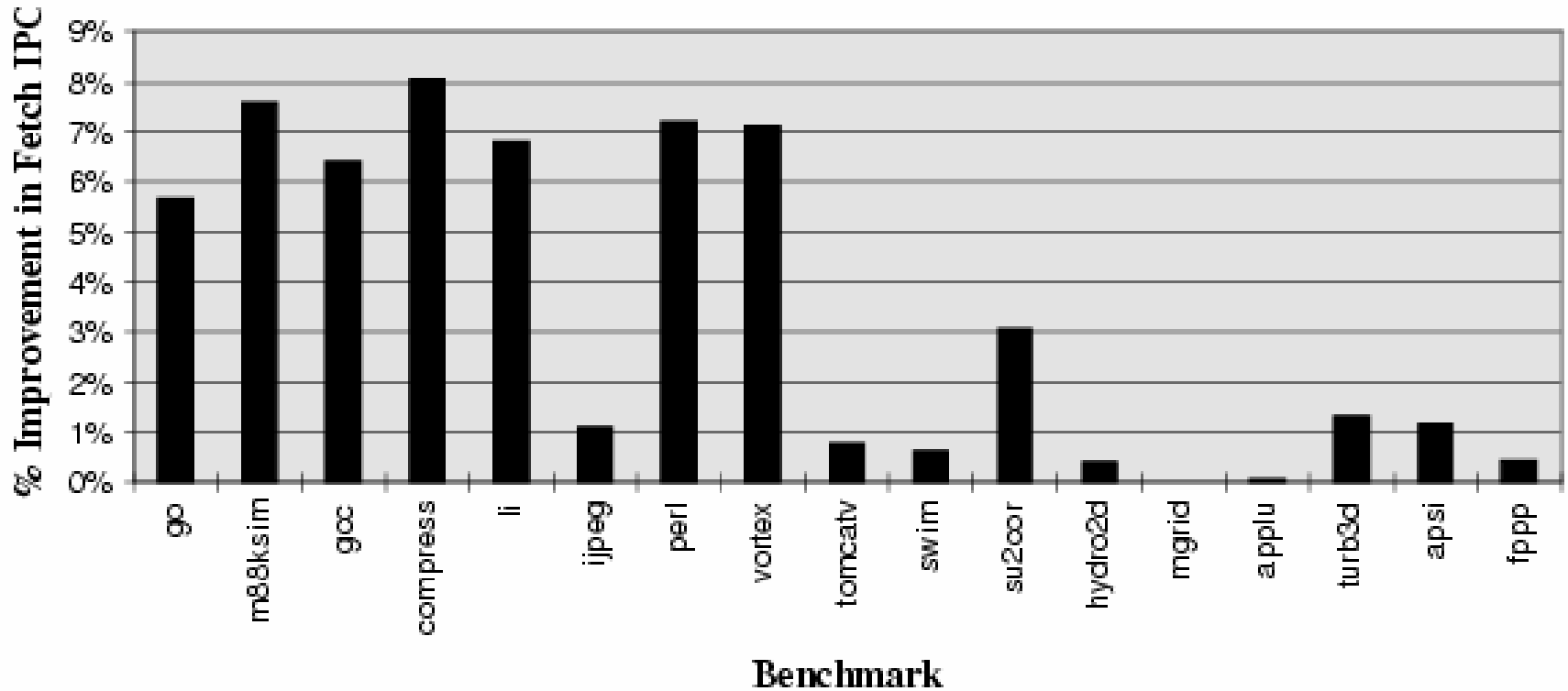
## Fetch IPC as a Function of Associativity



### Integer Fetch IPC as a Function of Trace Cache Size



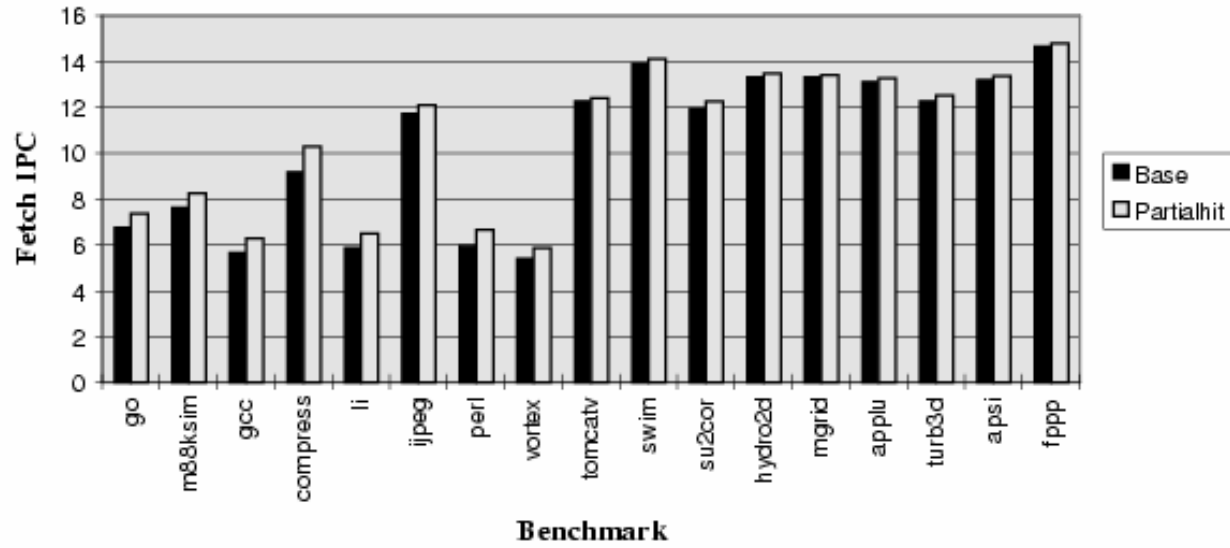
## IPC Improvement When Only Storing Complete Basic Blocks



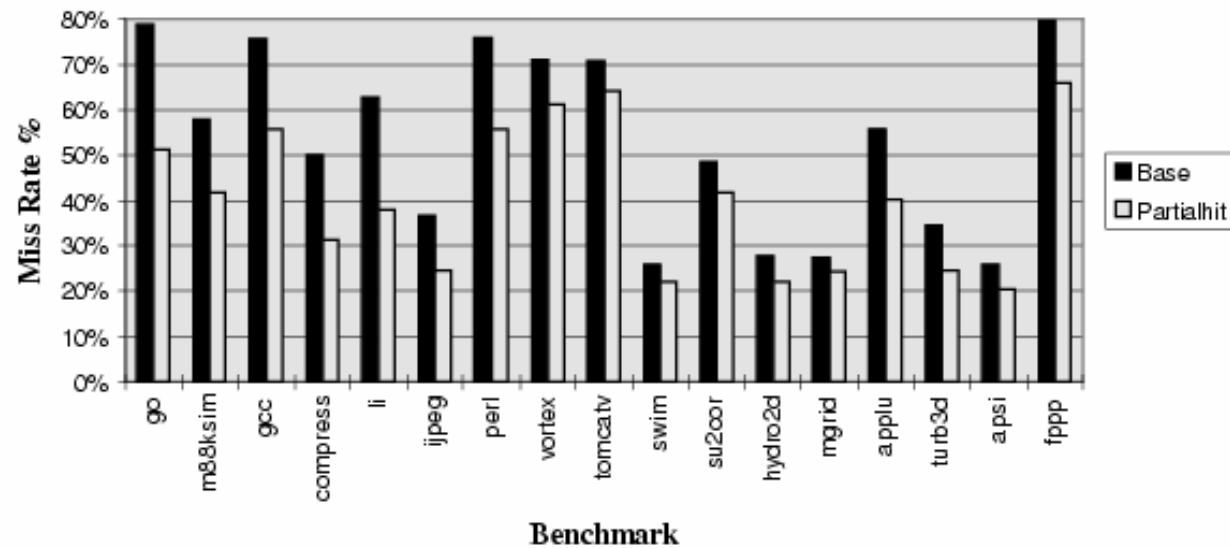
Fill Mechanism



### Fetch IPC from Allowing Partial Hits



### Trace Cache Miss Rate from Allowing Partial Hits



# [ Paper 1: Critique ]

---

- Power consumption
- Filling unit latency
- Duplication of instructions
- Liveness of traces
  
- Design issues

# [ Paper 2 ]

---

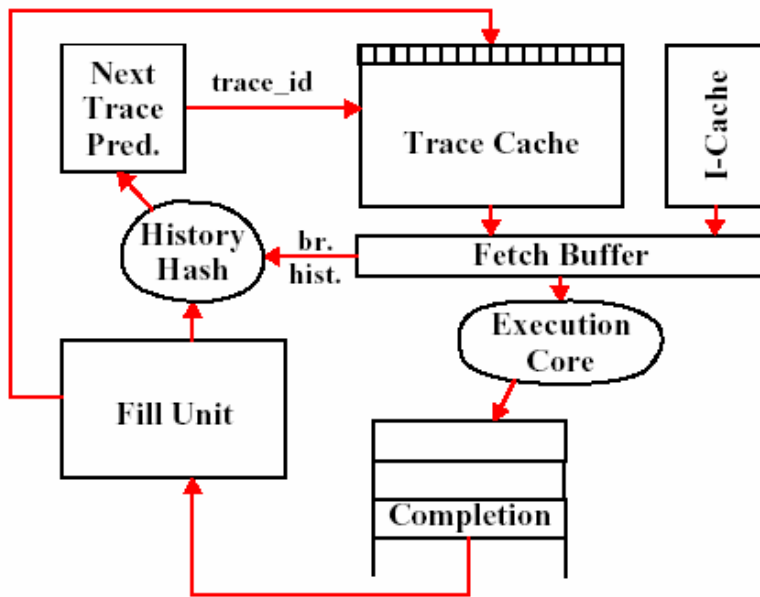
- Present a **block-based** trace cache implementation
  - Fetch address renaming
  - Basic block cache
- Performance comparison between conventional and block-based trace cache

# [ Motivation ]

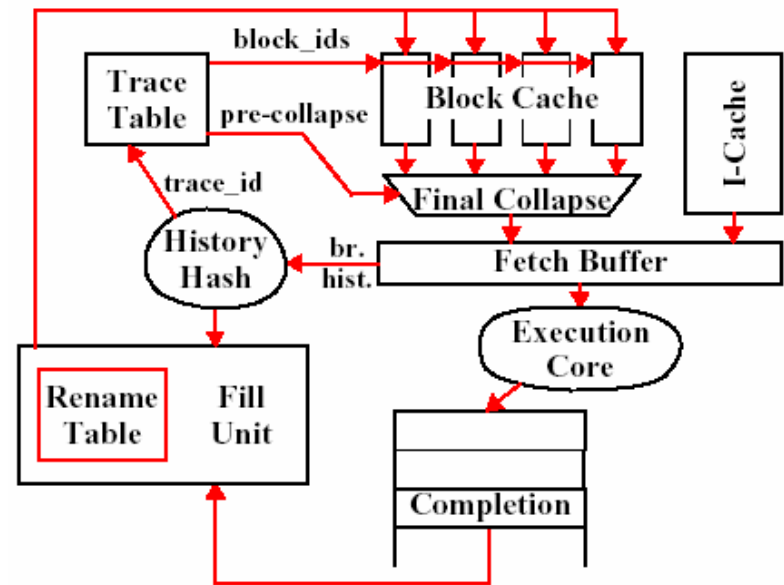
---

- Trace cache storage efficiency.
- Reduce the latency of indexing and associativity.
- Flexibility of trace construction and prediction.

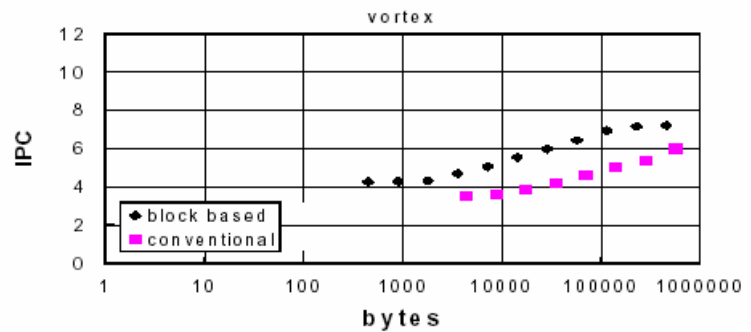
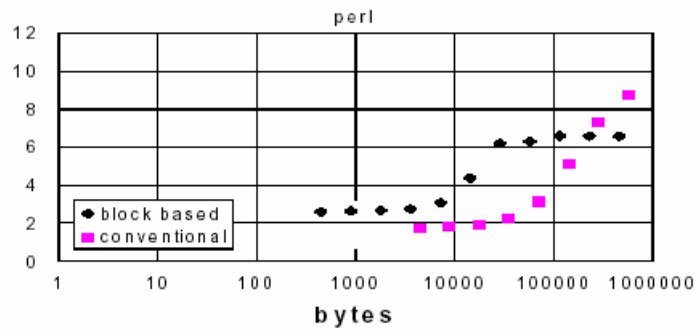
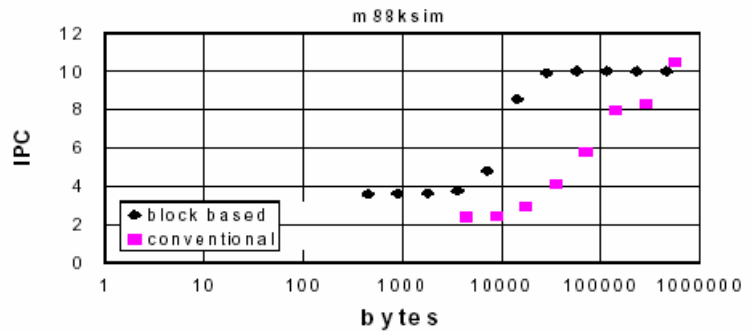
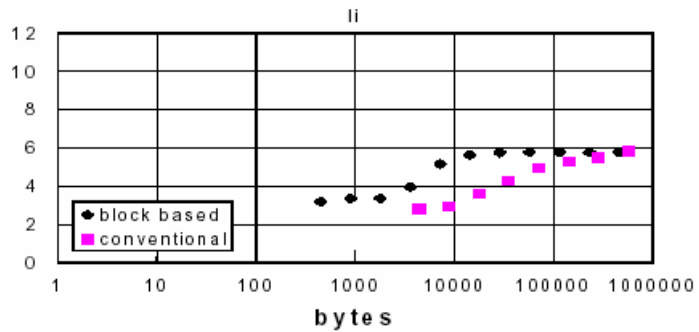
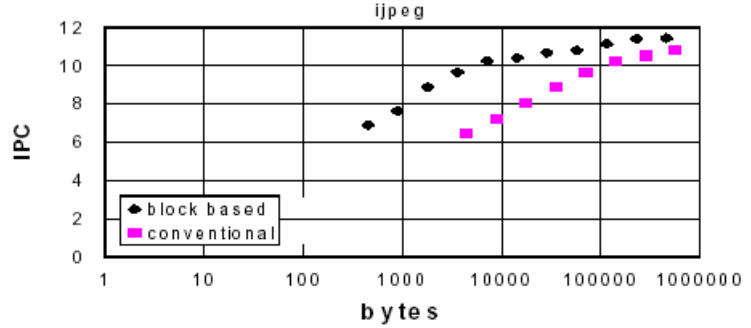
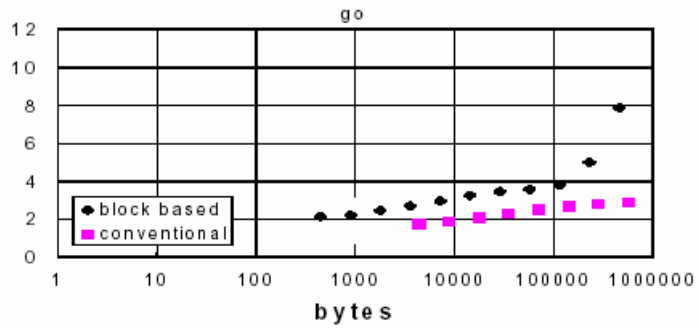
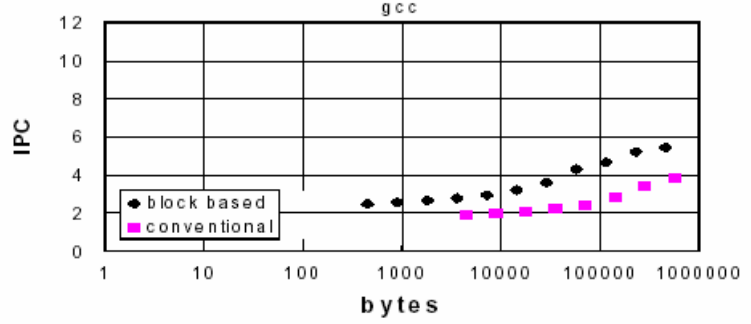
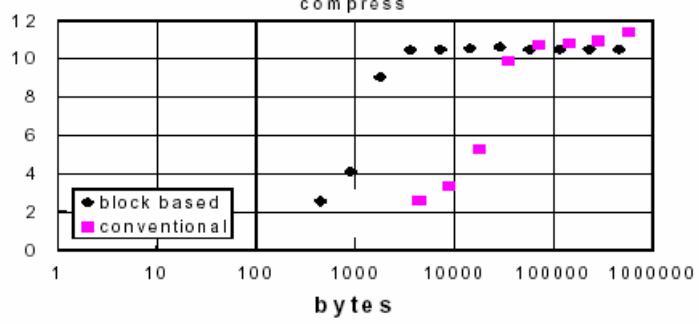
# Comparison



Conventional



Block-based



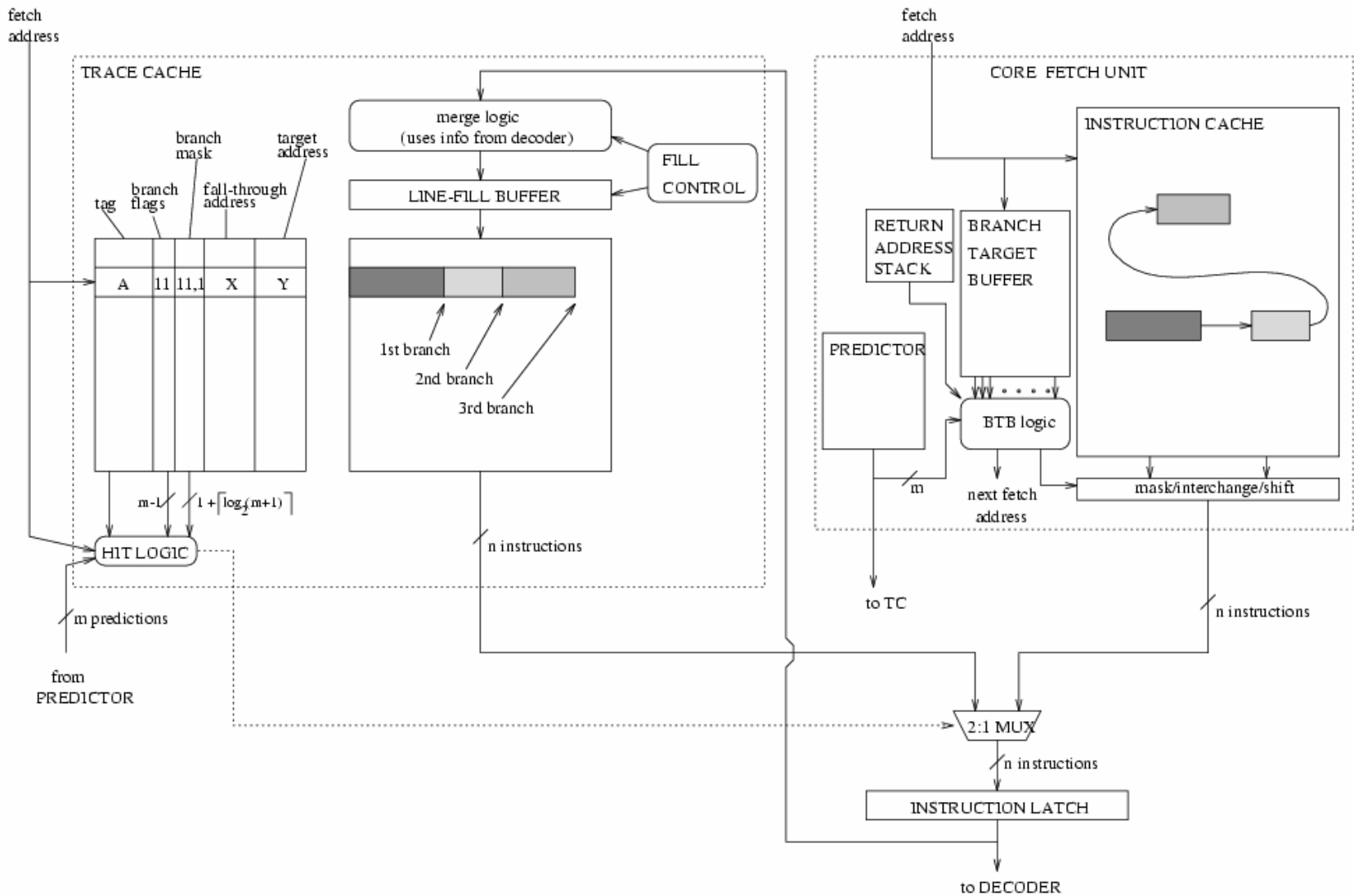
# Questions

- Dependence on branch prediction
- Other mechanisms
  - Branch Address Cache
  - Collapsing Buffer
    - *Trace Cache: a Low Latency Approach to High Bandwidth Instruction Fetching*
- Compiler Techniques ?

# Discussion - Research

- Replace instruction cache with trace cache?
- Reduce duplication and fragmentation
- Dynamic direction prediction trace cache
  - *Using Dynamic Branch Behavior for Power-Efficient Instruction Fetch. J. S. Hu, N. Vijaykrishnan, M. J. Irwin, M. Kandemir*
- Pentium4: Execution trace cache stores 12K decoded micro-ops





# Microarchitecture