

# Prefetching (II): Using Processors-In-Memory (PIM) for Prefetching

---

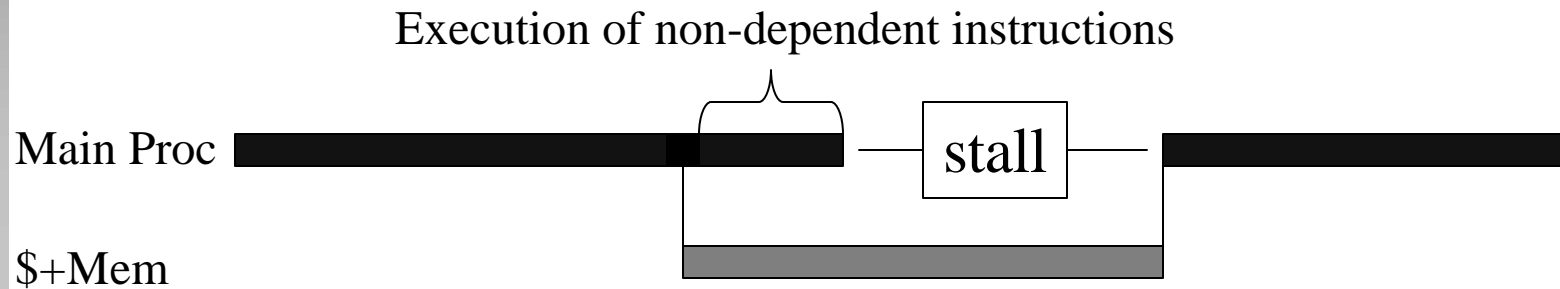
Instructor: Josep Torrellas  
CS533



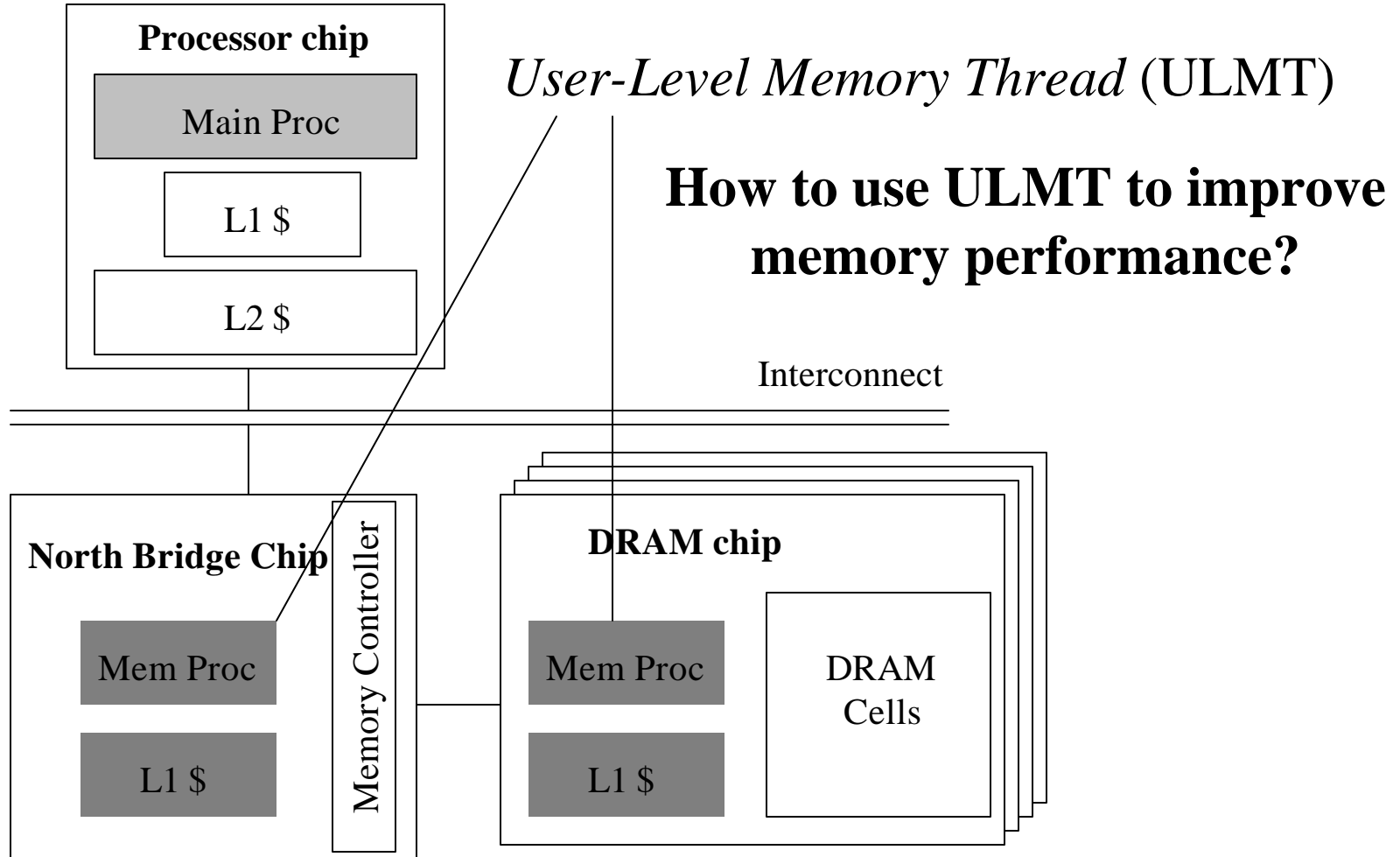
Copyright Josep Torrellas 2003

# Memory Stall Time

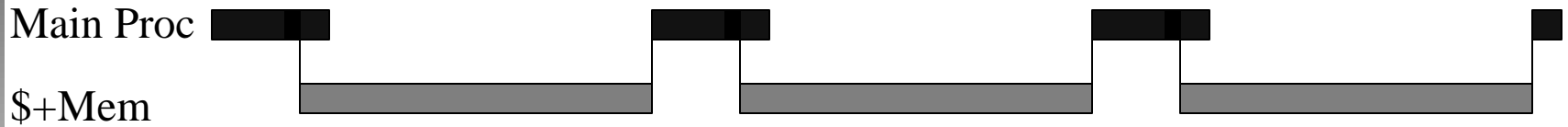
---



# Intelligent Memory



# Exploiting Intelligent Memory



## ULMT for Memory-Side Prefetching



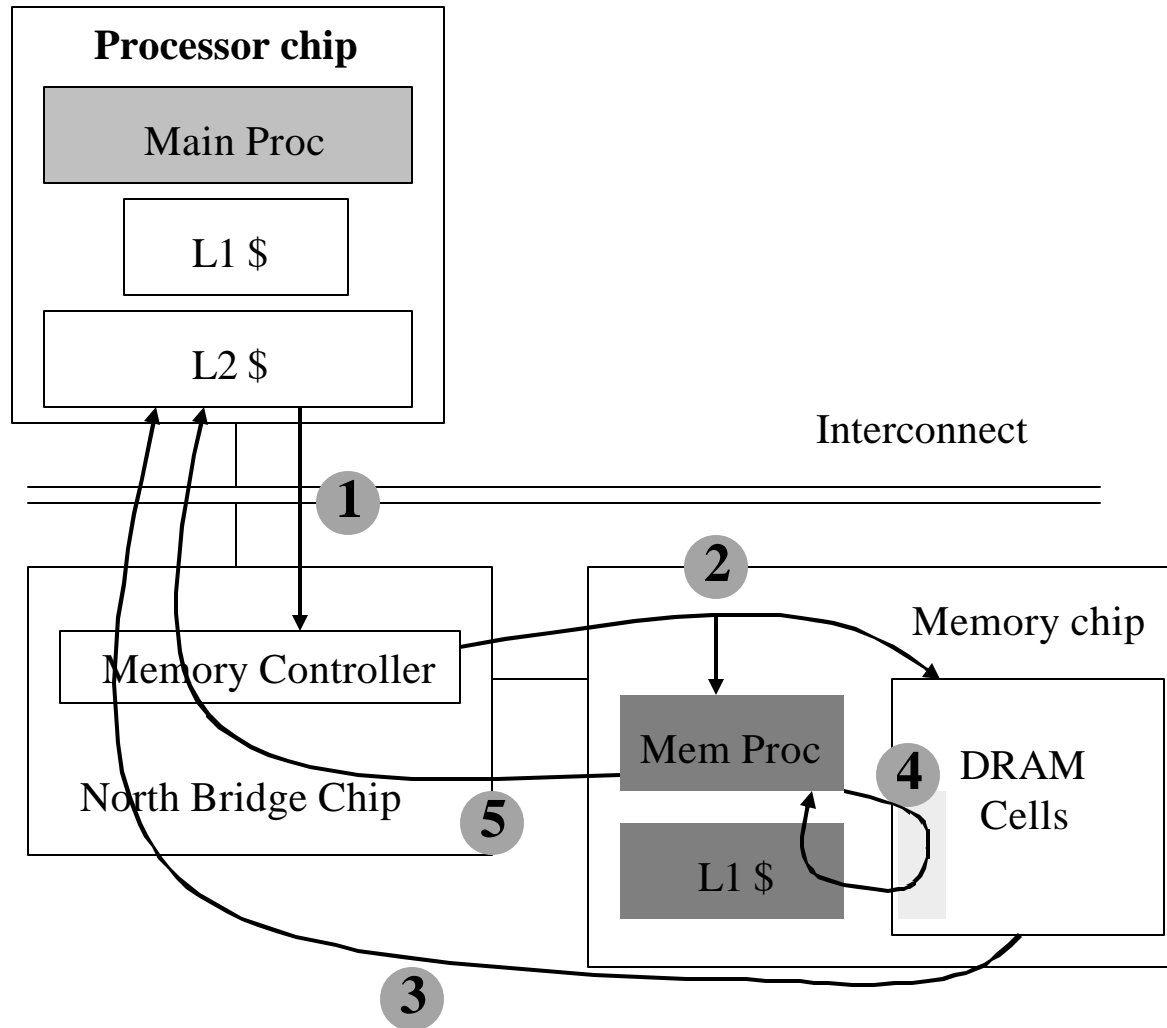


# Correlation Prefetching

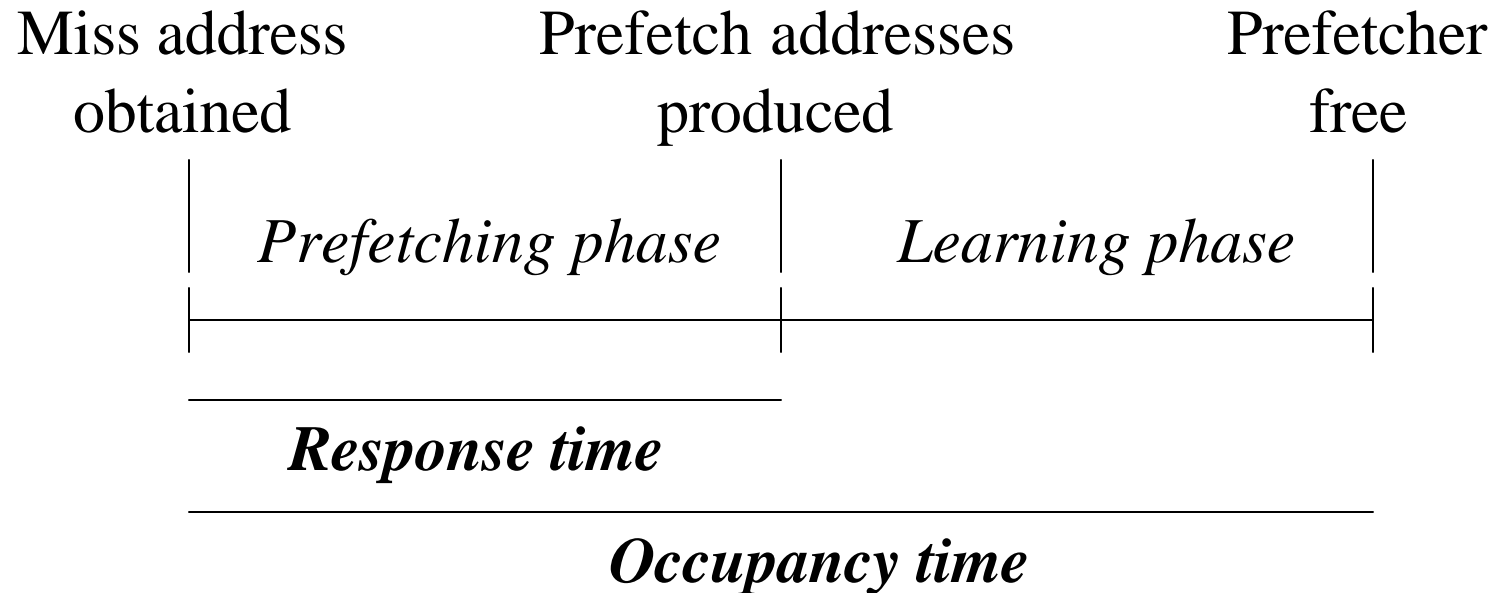
---

- ◆ Effective for irregular+regular apps
- ◆ Needs little info: cache miss addresses
- ◆ No compiler support
- ◆ Works with existing binaries
- ◆ How to implement the correlation table?
  - Past work: Expensive hardware
    - Few MBs of on-chip SRAM for the table (larger than cache!)
    - Table size has to scale with app working set
  - Paper proposes: Software
    - Is software fast enough? Yes.

# Communication Mechanism



# Timeline of the Prefetch Handler

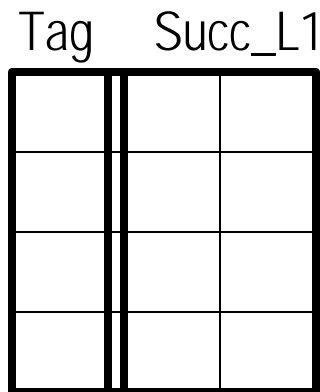


- ◆ Ideal algorithm:
  - lowest response time
  - occupancy time < time between misses

# Correlation Table

## Basic Organization <sup>[1]</sup>

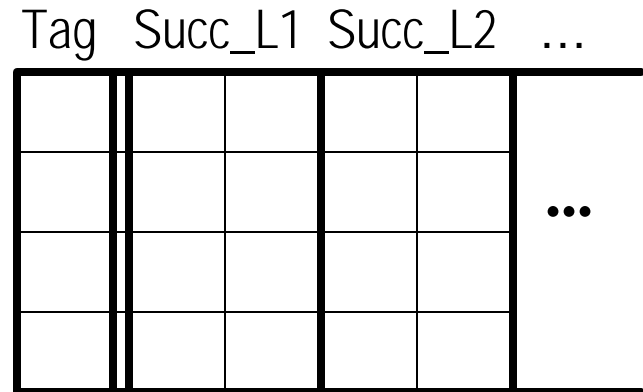
Addr of immediate successors



NumSucc = 2

## Advanced Organization

Addr of next immediate  
successors



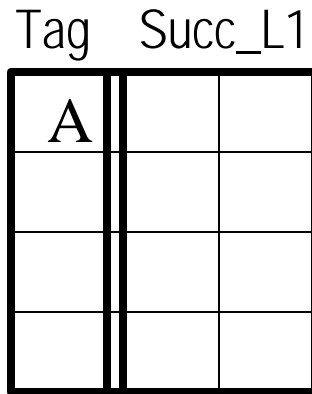
NumSucc = 2

NumLevels = n

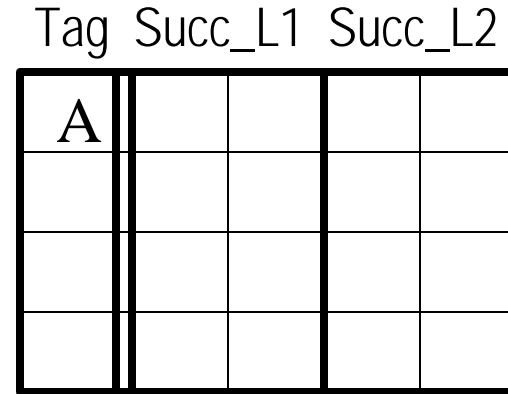
[1] Joseph&Grunwald, ISCA 97

# Learning Phase

## Basic Organization



## Advanced Organization



Current miss

A, B, C, A, D, C, ...

# Learning Phase

## Basic Organization

Tag	Succ_L1
A	B
B	

## Advanced Organization

Tag	Succ_L1	Succ_L2
A	B	
B		

Current miss

A, B, C, A, D, C, ...

# Learning Phase

## Basic Organization

Tag	Succ_L1
A	B
B	C
C	

## Advanced Organization

Tag	Succ_L1	Succ_L2
A	B	C
B	C	
C		

Current miss

A, B, C, A, D, C, ...

# Learning Phase

## Basic Organization

Tag	Succ_L1	
A	B	
B	C	
C	A	

## Advanced Organization

Tag	Succ_L1	Succ_L2	
A	B	C	
B	C	A	
C	A		

Current miss

A, B, C, A, D, C, ...

# Learning Phase

## Basic Organization

Tag	Succ_L1	
A	B	D
B	C	
C	A	
D	C	

## Advanced Organization

Tag	Succ_L1	Succ_L2		
A	B	D	C	
B	C		A	
C	A		D	
D	C			

Current miss

A, B, C, A, D, C, ...

# Learning Phase

## Basic Organization

Tag	Succ_L1	
A	B	D
B	C	
C	A	
D	C	

## Advanced Organization

Tag	Succ_L1	Succ_L2	
A	B	D	C
B	C		A
C	A		D
D	C		

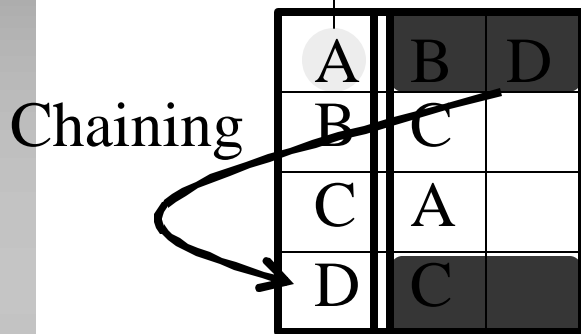
Current miss

A, B, C, A, D, C, ...

# Prefetching Phase

## Basic Organization

On miss A



Basic: 1 miss  $\Rightarrow$  *immediate successor*

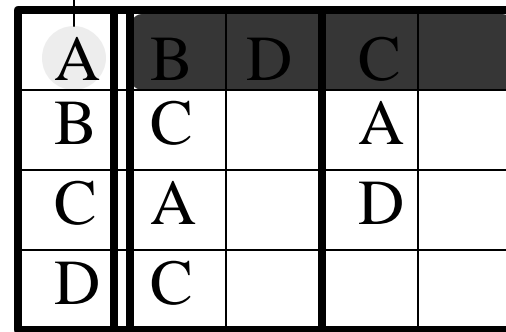
- no far ahead prefetching
- low coverage and late prefetches

Basic + Chaining:

- low accuracy
- high response time

## Advanced Organization

On miss A



Advanced: 1 miss  $\Rightarrow$  *several succ levels:*

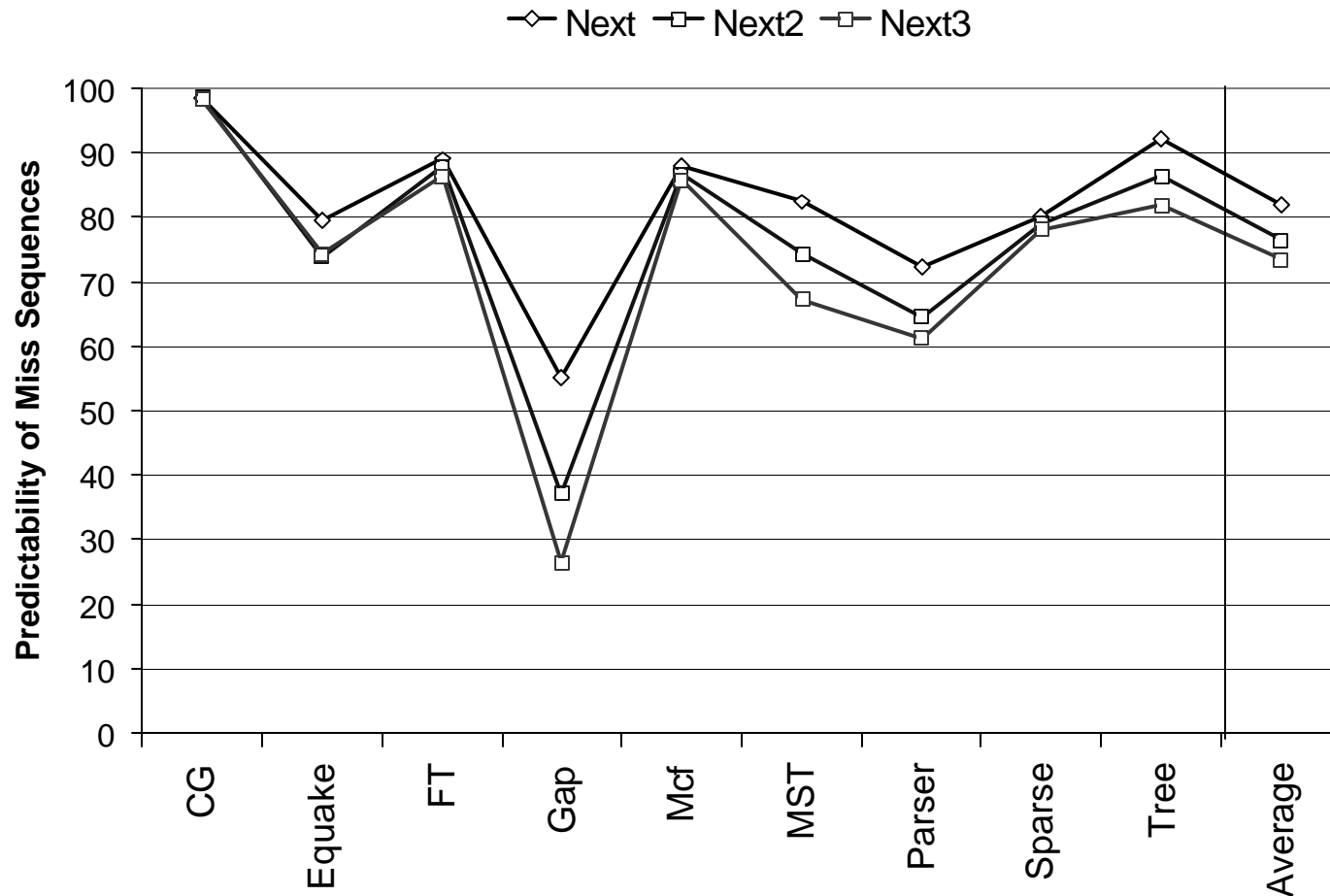
- + far ahead prefetching
- + high coverage
- + timely prefetches
- + high accuracy
- + low response time

# Simulation Environment

---

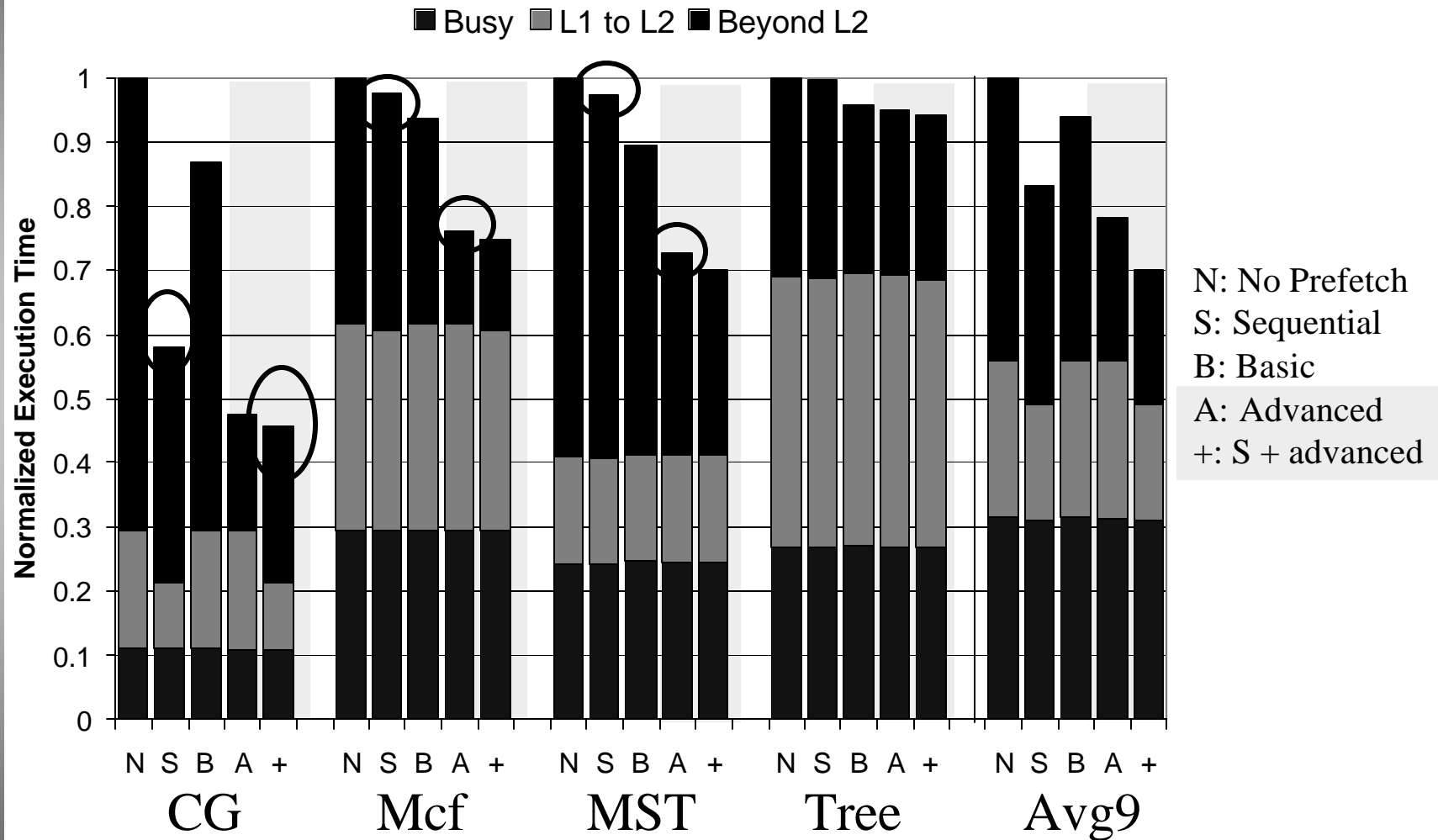
- ◆ Main processor:
  - 1.6 GHz, 6-issue OOO
  - L1: 2-way 16 KB; L2: 4-way 512 KB
  - Mem: 243 cycle RT
- ◆ Memory processor:
  - 800 MHz, 2-issue OOO
  - L1: 2-way 32 KB
  - Mem: 100 cycle RT (in NorthBridge), 56 cycle RT (in DRAM)
- ◆ Correlation table:
  - Application dependent, e.g. 64K entries, 3 levels, 2 successors
- ◆ Applications
  - Specint2000, Specfp2000, NAS, Olden

# Predictability of miss sequences

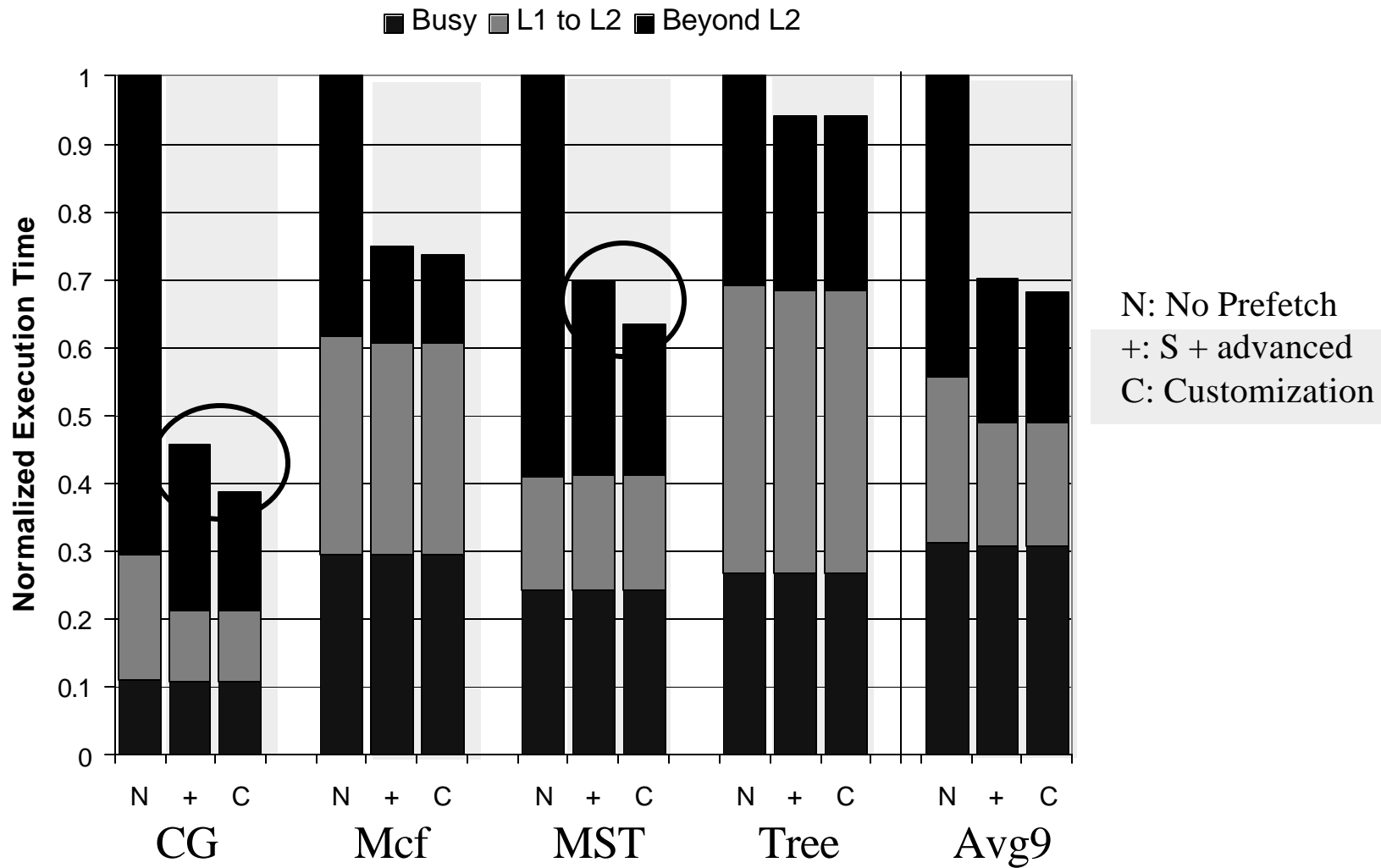


- ◆ Can predict miss sequences accurately

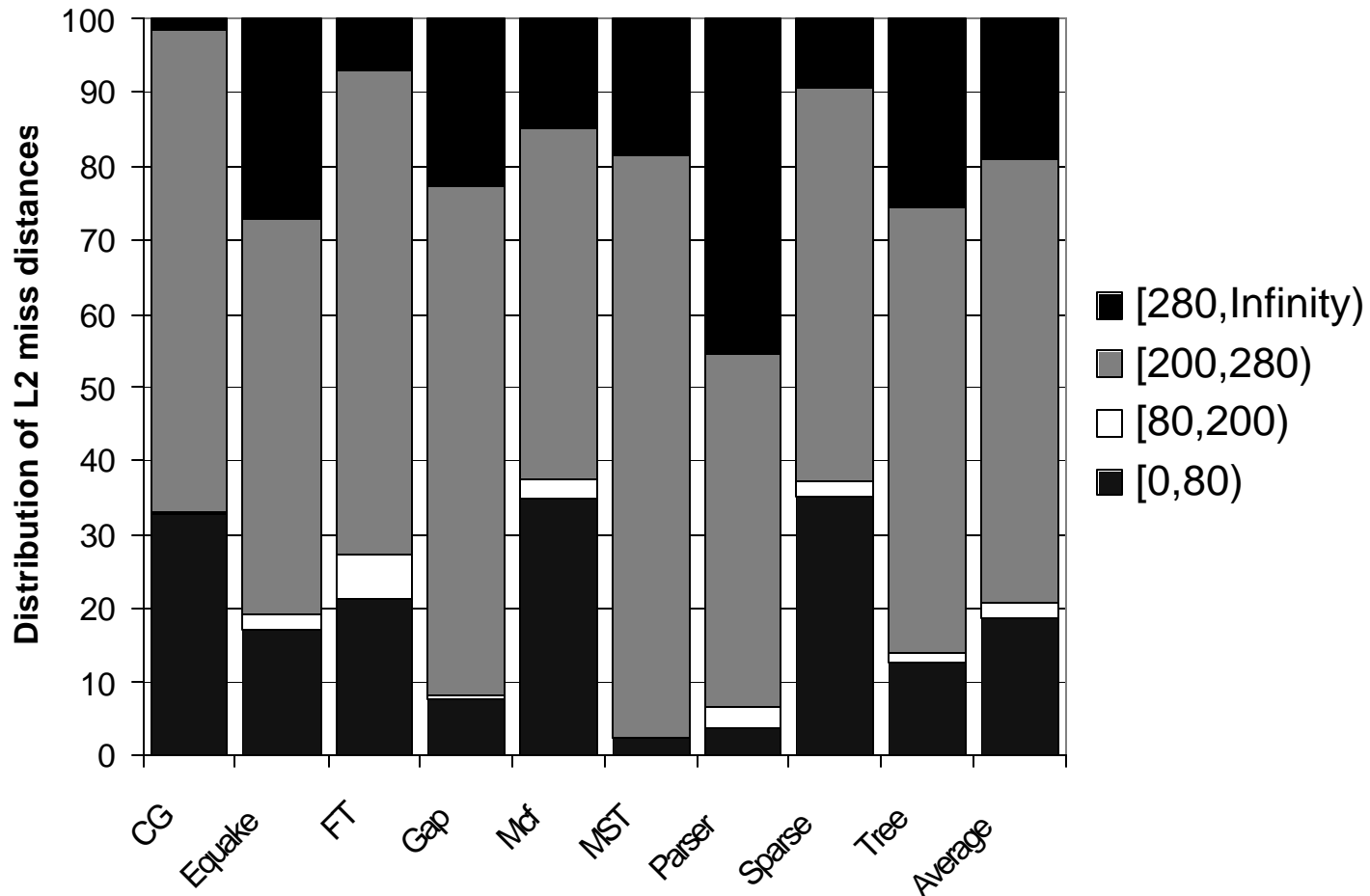
# Execution Time (Mem Proc in DRAM)



# Customization



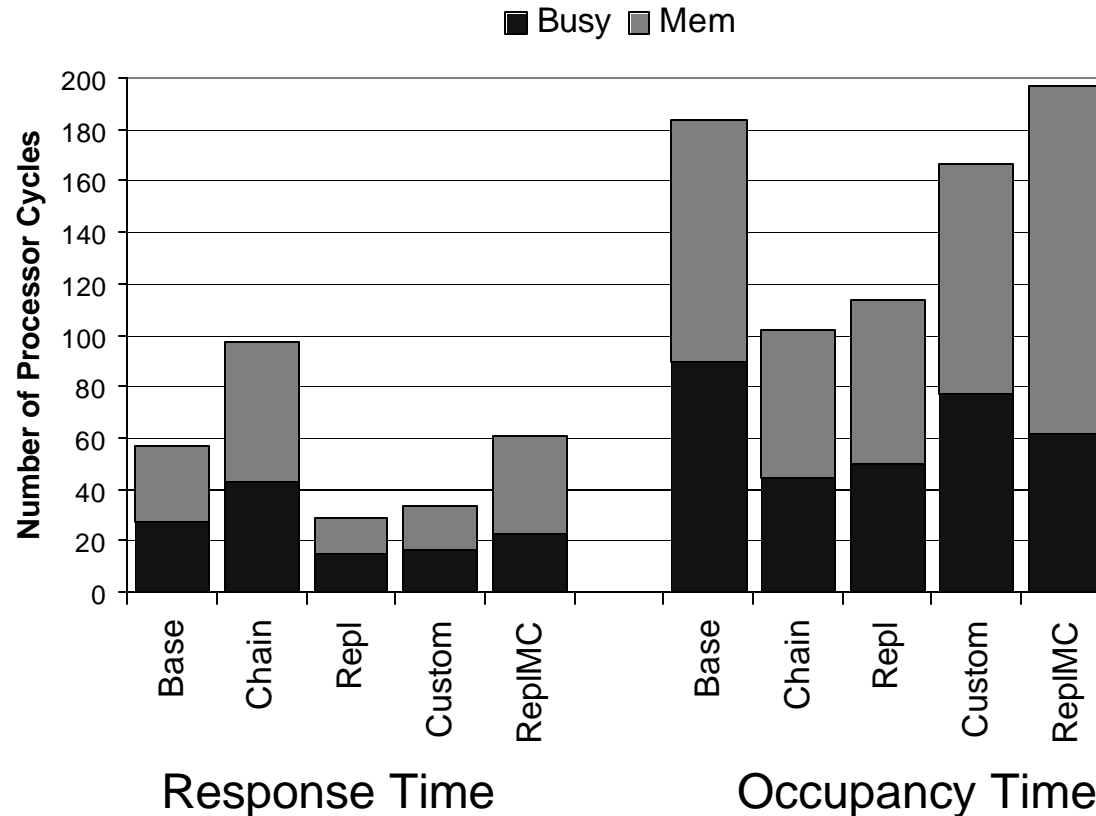
# Cycles Between Misses



- Occupancy time of prefetch thread must be  $< 200$  cycles



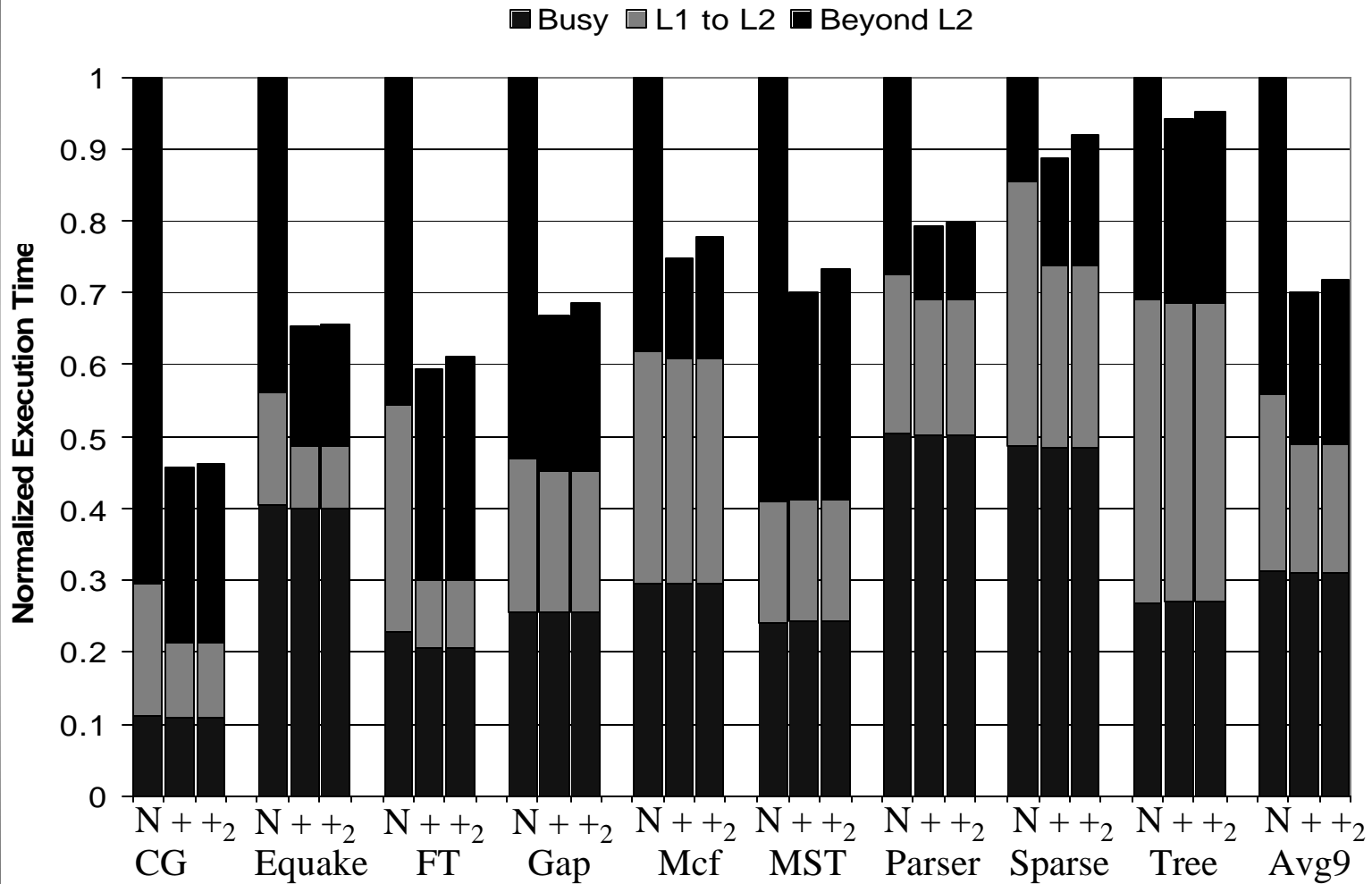
# Thread Response and Occupancy Time



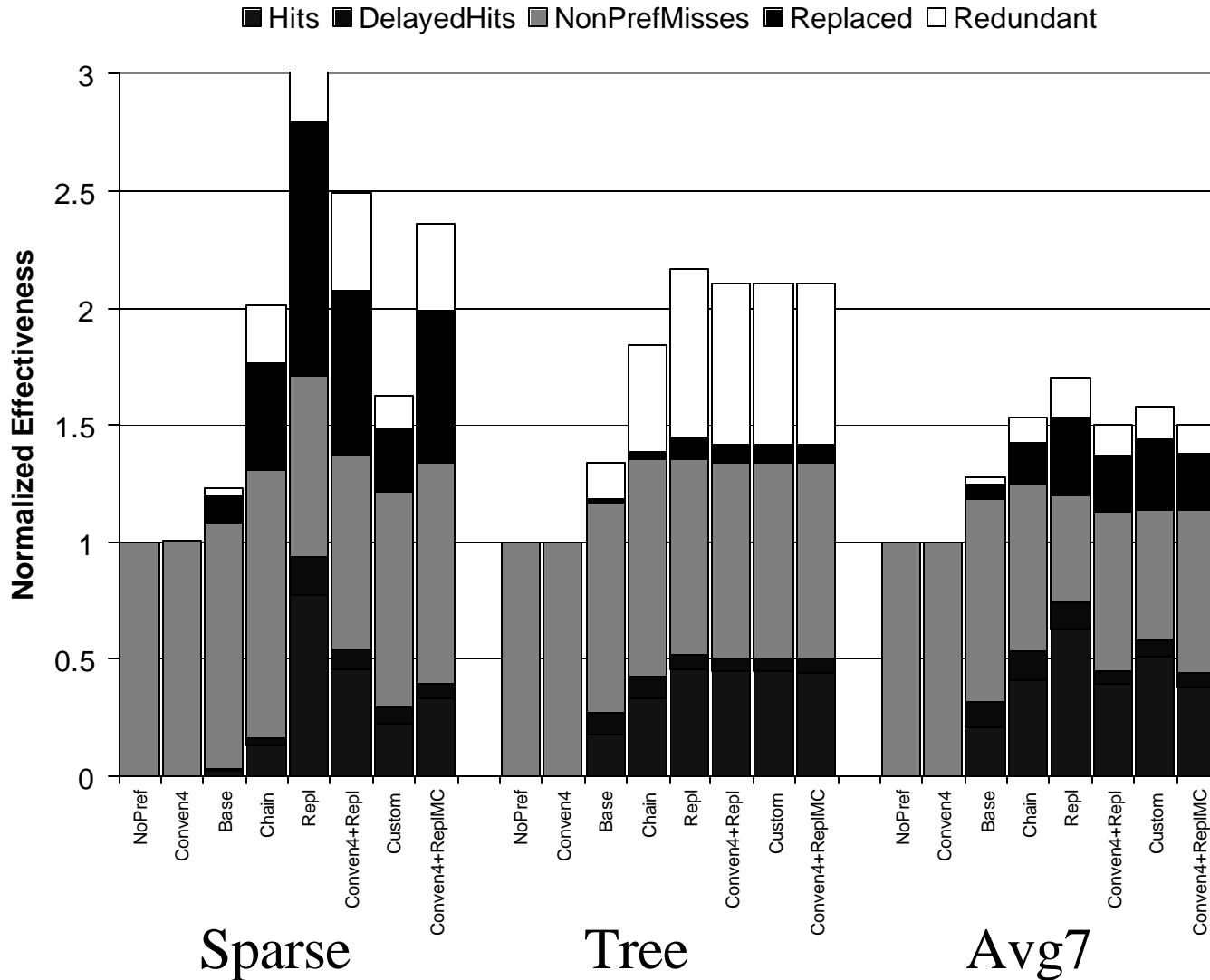
- All algo feasible: Occupancy Time < 200 cycles
- Advanced/Repl has the best Response Time



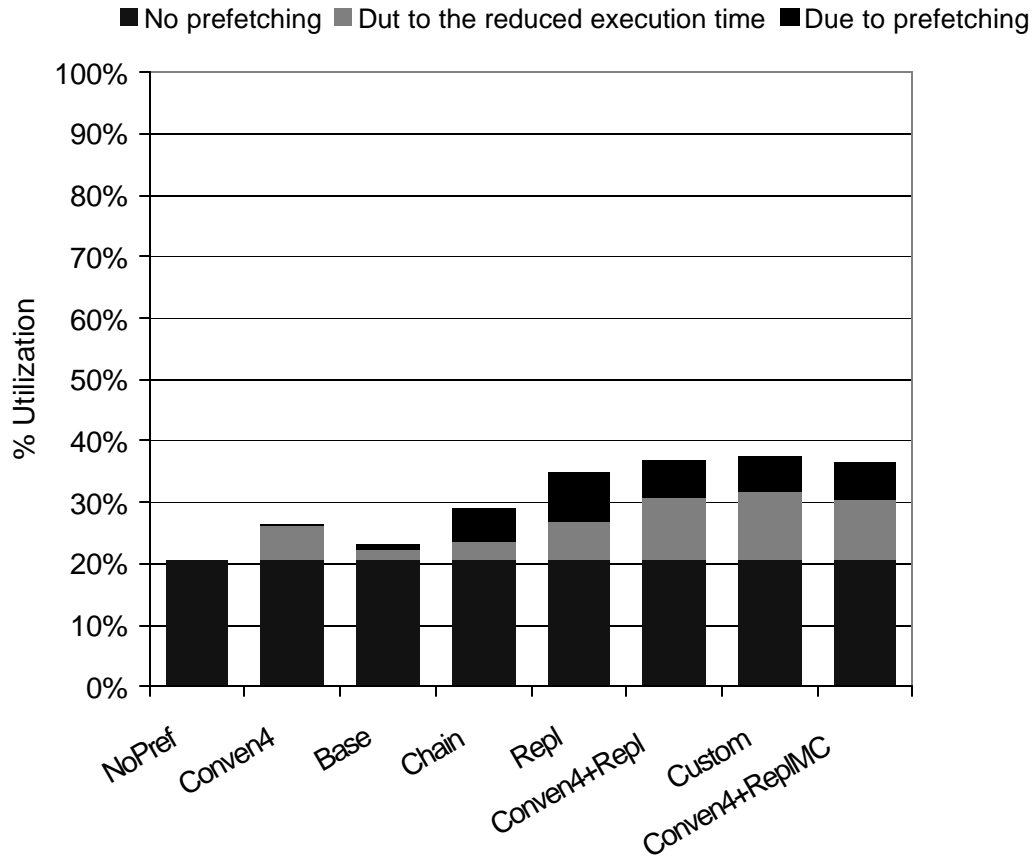
# DRAM vs. Mem Controller Chip



# Prefetching Effectiveness



# Bus Utilization



- ◆ Advanced: avg increase of 8%

