COMPUTER ORGANIZATION AND ARCHITECTURE

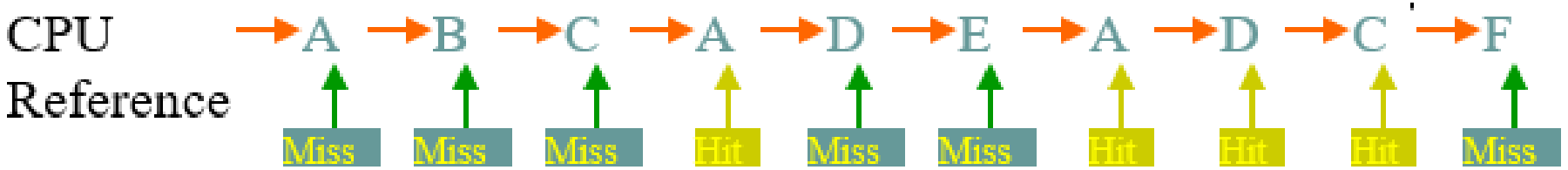
Course Code : CSE 2151

Credits : 04

REPLACEMENT ALGORITHMS

- Difficult to determine which blocks to be removed
- Least Recently Used (LRU) block
- The cache controller tracks references to all blocks as computation proceeds.
- Increase / clear track counters when a hit/miss occurs

REPLACEMENT ALGORITHMS



A	A	A	**A**	A	A	**A**	A	A	A
	B	B	B	B	E	E	E	E	F
		C	C	C	C	C	C	**C**	C
				D	D	D	**D**	D	D

Hit Ratio = 4 / 10 = 0.4

A	B	C	A	D	E	A	D	C	F
	A	B	C	A	D	E	A	D	C
		A	B	C	A	D	E	A	D
				B	C	C	C	E	A

LRU ALGORITHM

- The cache controller must track references to all blocks as computation proceeds.
- Suppose it is required to track the LRU block of a four-block set in a set-associative cache.
- A 2-bit counter can be used for each block.
- When a hit occurs,
  - the counter of the block that is referenced is set to 0.
  - Counters with values originally lower than the referenced one are incremented by one, and
  - all others remain unchanged.
- When a miss occurs and the set is not full,
  - the counter associated with the new block loaded from the main memory is set to 0, and
  - the values of all other counters are increased by one.
- When a miss occurs and the set is full,
  - the block with the counter value 3 is removed,
  - the new block is put in its place, and its counter is set to 0.
  - The other three block counters are incremented by one.
- It can be easily verified that the counter values of occupied blocks are always distinct

LRU ALGORITHM

- Assume counter values as shown

00	Block0
01	Block1
10	Block2
11	Block3

- Suppose hit occurs for block 2

01	Block0
10	Block1
00	Block2
11	Block3

LRU ALGORITHM

- A 4 × 10 array of numbers, each occupying one word, is stored in main memory locations 7A00 through 7A27 (hex). The elements of this array, A, are stored in column order, as shown below. Assume the data cache has space for only eight blocks of data.

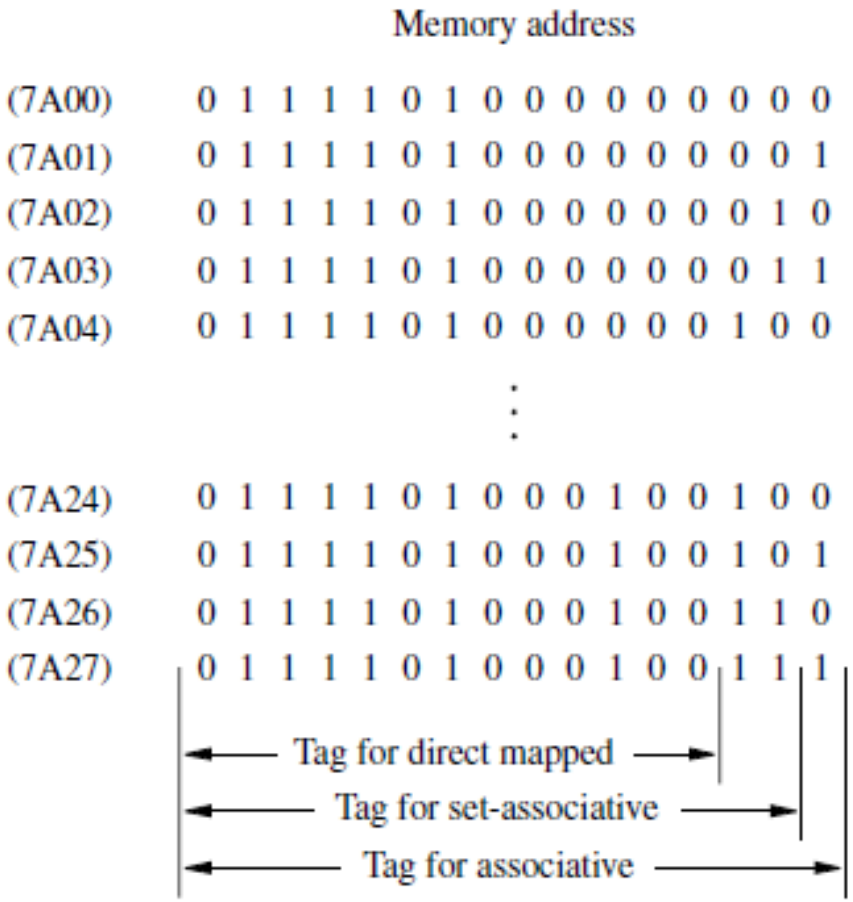
$$A(0,i) \leftarrow \frac{A(0,i)}{\left(\sum_{j=0}^{9} A(0,j)\right)/10} \qquad \text{for } i = 0, 1, \ldots, 9$$

```
SUM := 0
for j := 0 to 9 do
        SUM := SUM + A(0,j)
end
AVG := SUM/10
for i := 9 downto 0 do
        A(0,i) := A(0,i)/AVG
end
```

Figure 8.20 Task for example in Section 8.6.3.

LRU ALGORITHM

- No.of words in each block =1 (2^0=1)
  - Hence 0 bits to identify words within the block
- Direct mapped:
  - 8 blocks in cache: 3 bits to represent
- Set-associative:
  - 1 bit to identify set 0 or 1
- Associative:
  - All bits are considered tag



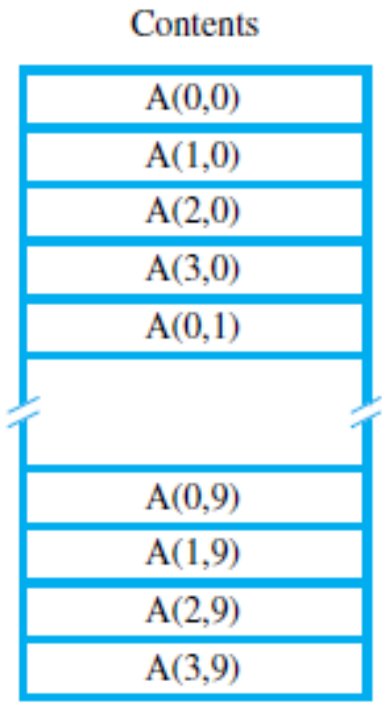


Figure 8.19 An array stored in the main memory.

DIRECT MAPPED CACHE

Block position	Contents of data cache after pass:								
	$j = 1$	$j = 3$	$j = 5$	$j = 7$	$j = 9$	$i = 6$	$i = 4$	$i = 2$	$i = 0$
0	A(0,0)	A(0,2)	A(0,4)	A(0,6)	A(0,8)	A(0,6)	A(0,4)	A(0,2)	A(0,0)
1									
2									
3									
4	A(0,1)	A(0,3)	A(0,5)	A(0,7)	A(0,9)	A(0,7)	A(0,5)	A(0,3)	A(0,1)
5									
6									
7									

Figure 8.21 Contents of a direct-mapped data cache.

ASSOCIATIVE MAPPED CACHE

Block position	Contents of data cache after pass:				
	$j = 7$	$j = 8$	$j = 9$	$i = 1$	$i = 0$
0	A(0,0)	A(0,8)	A(0,8)	A(0,8)	A(0,0)
1	A(0,1)	A(0,1)	A(0,9)	A(0,1)	A(0,1)
2	A(0,2)	A(0,2)	A(0,2)	A(0,2)	A(0,2)
3	A(0,3)	A(0,3)	A(0,3)	A(0,3)	A(0,3)
4	A(0,4)	A(0,4)	A(0,4)	A(0,4)	A(0,4)
5	A(0,5)	A(0,5)	A(0,5)	A(0,5)	A(0,5)
6	A(0,6)	A(0,6)	A(0,6)	A(0,6)	A(0,6)
7	A(0,7)	A(0,7)	A(0,7)	A(0,7)	A(0,7)

Figure 8.22 Contents of an associative-mapped data cache.

SET ASSOCIATIVE MAPPED CACHE

	Contents of data cache after pass:					
	$j = 3$	$j = 7$	$j = 9$	$i = 4$	$i = 2$	$i = 0$
Set 0	A(0,0)	A(0,4)	A(0,8)	A(0,4)	A(0,4)	A(0,0)
	A(0,1)	A(0,5)	A(0,9)	A(0,5)	A(0,5)	A(0,1)
	A(0,2)	A(0,6)	A(0,6)	A(0,6)	A(0,2)	A(0,2)
	A(0,3)	A(0,7)	A(0,7)	A(0,7)	A(0,3)	A(0,3)
Set 1						

Figure 8.23 Contents of a set-associative-mapped data cache

EXERCISE PROBLEM

- A byte-addressable computer has a small data cache capable of holding eight 32-bit words. Each cache block consists of one 32-bit word. When a given program is executed, the processor reads data sequentially from the following hex addresses:

  200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4

  This pattern is repeated four times.

  a) Assume that the cache is initially empty. Show the contents of the cache at the end of each pass through the loop if a direct-mapped cache is used and compute the hit rate.
  b) Repeat part a) for an associative-mapped cache that uses the LRU replacement algorithm.
  c) Repeat part (a) for a four-way set-associative cache.

TOPICS COVERED FROM

- Textbook 1:
  - Chapter 8: 8.6.2, 8.6.2