Associative caching

2/3/25

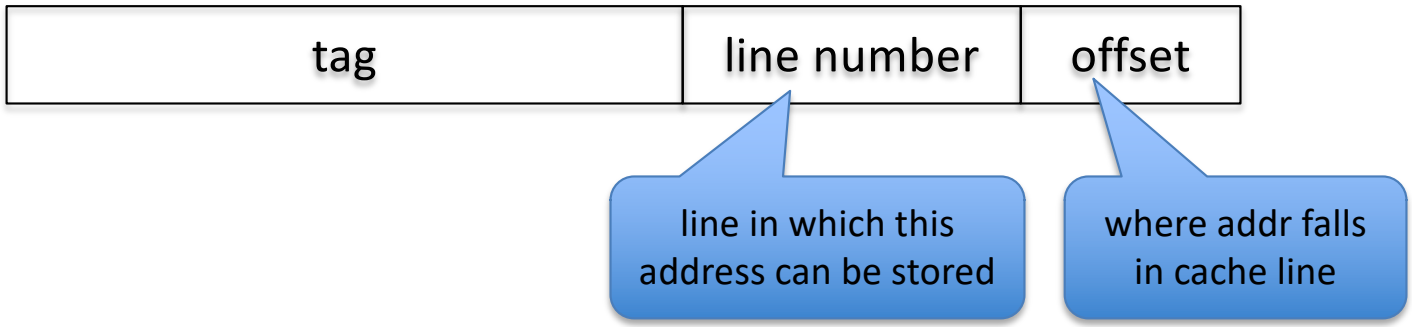
Administrivia

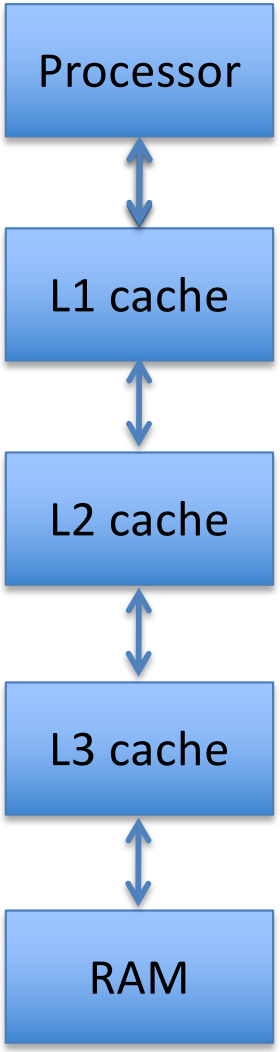
- HW 5 (linked lists in C) due Thursday night

- Midterm out Friday morning
  - Multi-day takehome due sometime next week
  - No class on Monday (2/10)
  - Covers everything up to (and including) caching

Recall: Idea of caching

- Have small, fast memory near the processor and bigger, slower memory far way
- Locality:
  - temporal: Likely to reuse memory addresses soon
  - spatial: Likely to use memory addresses near those we use now





Associative caches

- Fully-associative caches: Any data block can use any line

Associative caches

- Fully-associative caches: Any data block can use any line

What is the advantage of this?

How many bits would the tag occupy in a system with a 32-bit address space where the fully-associative cache stores $1024=2^{10}$ lines, each holding 4 bytes?

A. 18
B. 20
C. 26
D. 30
E. None of the above or cannot be determined

How many bits would the tag occupy in a system with a 32-bit address space where the fully-associative cache stores $1024=2^{10}$ lines, each holding 4 bytes?

A. 18
B. 20
C. 26
D. 30
E. None of the above or cannot be determined

Associative caches

- Fully-associative caches: Any data block can use any line
  - Replacement strategies: FIFO, LRU

Associative caches

- Fully-associative caches: Any data block can use any line
  - Replacement strategies: FIFO, LRU
  - Approximate LRU with just a couple of bits

Associative caches

- Fully-associative caches: Any data block can use any line
  - Replacement strategies: FIFO, LRU
  - Approximate LRU with just a couple of bits

- K-way set associative: Lines organized into sets, each data block can use any line in its set

How many bits would the tag occupy in a system with a 32-bit address space where the 4-way set associative cache stores $1024=2^{10}$ lines, each holding 4 bytes?

A. 18
B. 20
C. 22
D. 24
E. None of the above or cannot be determined

How many bits would the tag occupy in a system with a 32-bit address space where the 4-way set associative cache stores $1024=2^{10}$ lines, each holding 4 bytes?

A. 18
B. 20
C. 22
D. 24
E. None of the above or cannot be determined

How many bits does the tag use in a system with a 32-bit address space where the 8-way set associative cache stores $2048=2^{11}$ lines, each holding $16=2^4$ bytes?

A. 14
B. 17
C. 20
D. 24
E. None of the above or cannot be determined

How many bits does the tag use in a system with a 32-bit address space where the 8-way set associative cache stores $2048=2^{11}$ lines, each holding $16=2^4$ bytes?

A. 14
B. 17
C. 20 (= 32 - (11 - 3) - 4)
D. 24
E. None of the above or cannot be determined

What happens on a read to address 0011100?

(Cache is 2-way set associative, LRU, 2-byte lines)

Set of 2 slots

#	valid	tag	data
0	T	0011	???
1	F	0010	???
2	T	1110	???
3	T	1000	???
4	T	1011	???
5	F	0011	???
6	T	1011	???
7	F	0010	???

A. Hit
B. Miss – loads line 3
C. Miss – loads line 4
D. Miss – loads line 5
E. None of the above or can't be determined

What happens on a read to address 0011100?

(Cache is 2-way set associative, LRU, 2-byte lines)

Set of 2 slots

#	valid	tag	data
0	T	0011	???
1	F	0010	???
2	T	1110	???
3	T	1000	???
4	T	1011	???
5	F	0011	???
6	T	1011	???
7	F	0010	???

A. Hit
B. Miss – loads line 3
C. Miss – loads line 4
D. Miss – loads line 5
E. None of the above or can't be determined

What happens on a read to address 0011011?

(Cache is 2-way set associative, LRU, 2-byte lines)

Set of 2 lines

#	valid	tag	data
0	T	0011	???
1	F	0010	???
2	T	1110	???
3	T	1000	???
4	T	1011	???
5	F	0011	???
6	T	1011	???
7	F	0010	???

A. Hit
B. Miss – loads line 3
C. Miss – loads line 4
D. Miss – loads line 5
E. None of the above or can't be determined

What happens on a read to address 0011011?

(Cache is 2-way set associative, LRU, 2-byte lines)

Set of 2 lines

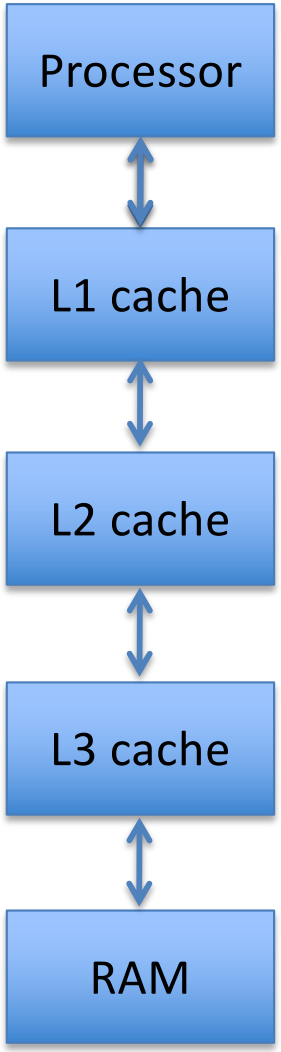
#	valid	tag	data
0	T	0011	???
1	F	0010	???
2	T	1110	???
3	T	1000	???
4	T	1011	???
5	F	0011	???
6	T	1011	???
7	F	0010	???

A. Hit
B. Miss – loads line 3
C. Miss – loads line 4
D. Miss – loads line 5
E. None of the above or can't be determined (miss – loads 2 or 3)

What about writes?

- Write-thru cache: Always send writes all the way down the memory hierarchy

- Write-back cache: Don't propagate changes down the hierarchy until data block is evicted
  - Add "dirty bit" to each line



Recall: Example of caching effects

```
int[][] array = new int[size][size];
for(int i=0; i < size; i++)
    for(int j=0; j < size; j++)
        /* DO SOMETHING */
```

Version 1: array[i][j] *= 2;

Version 2: array[j][i] *= 2;

Relevant detail: Java stores a 2D array as an array of array references

Recall: Example of caching effects

```
int[][] array = new int[size][size];
for(int i=0; i < size; i++)
    for(int j=0; j < size; j++)
        /* DO SOMETHING */
```

Version 1: array[i][j] *= 2;

Version 2: array[j][i] *= 2;

Which version should have better cache performance?

A) Version 1

B) Version 2

Relevant detail: Java stores a 2D array as an array of array references

Recall: Example of caching effects

```
int[][] array = new int[size][size];
for(int i=0; i < size; i++)
    for(int j=0; j < size; j++)
        /* DO SOMETHING */
```

Version 1: array[i][j] *= 2;

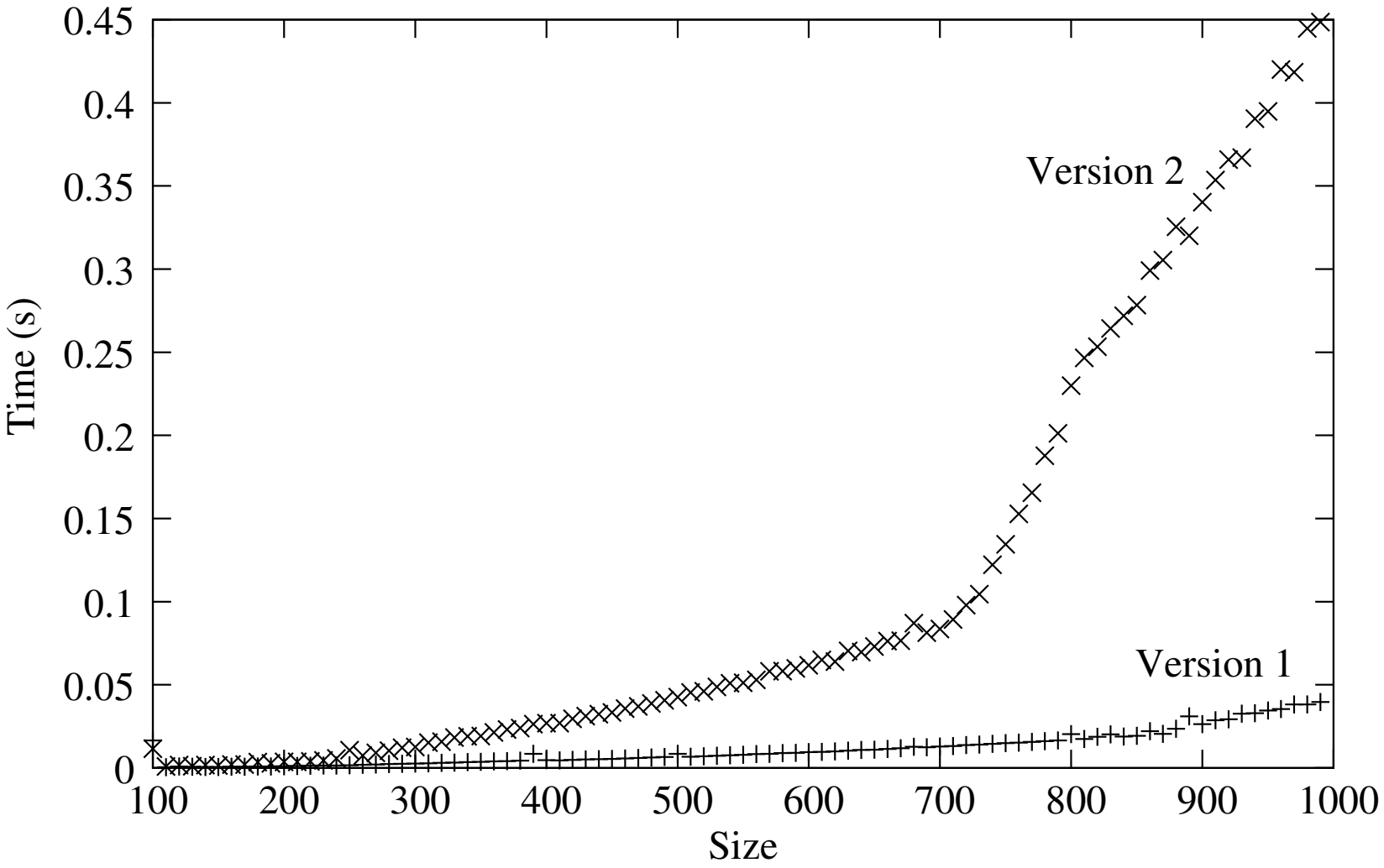
Version 2: array[j][i] *= 2;

Which version should have better cache performance?

A) Version 1

B) Version 2

Relevant detail: Java stores a 2D array as an array of array references


0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Time (s)
100
200
300
400
500
600
700
800
900
1000
Size
Version 2
Version 1