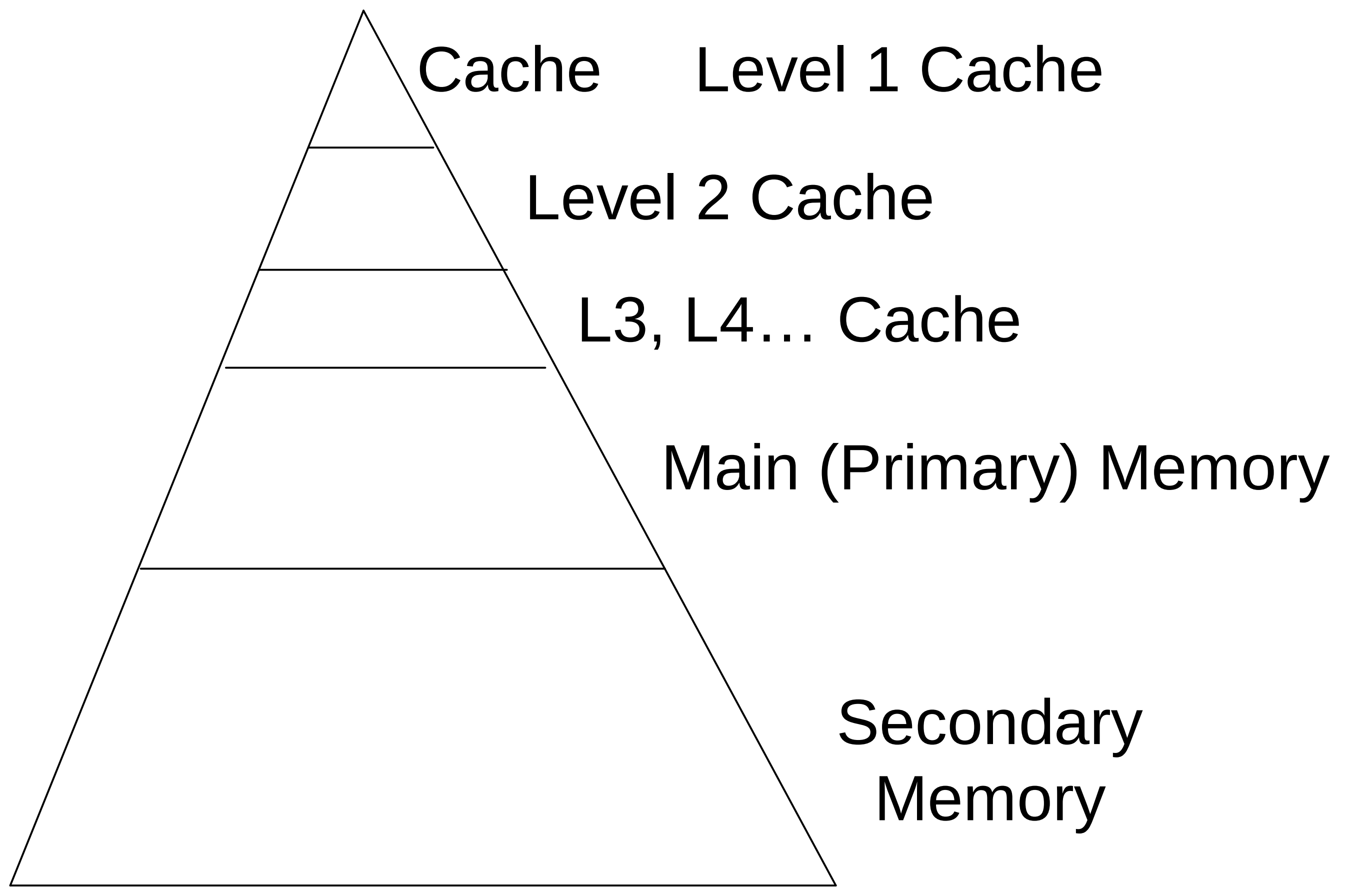
High Performance Computing
Lecture 29

Matthew Jacob

Indian Institute of Science

Memory Hierarchy Progression



Cache and Programming

- Objective: Learn how to assess cache related performance issues for important parts of our programs
- Will look at several examples of programs
- Will consider only data cache, assuming separate instruction and data caches
- Data cache configuration:
  - Direct mapped 16 KB write back cache with 32B block size

Tag : 18b	**Index: 9b**	**Offset: 5b**

Example 1: Vector Sum Reduction

```
double A[2048], sum=0.0;
for (i=0; i<2048, i++) sum = sum +A[i];
```

- To do analysis, must view program close to machine code form (to see loads/stores)
  - Recall from static instruction scheduling examples how loop index i was implemented in a register and not load/stored inside loop
  - Will assume that both loop index i and variable sum are implemented in registers
- Will consider only accesses to array elements

Example 1: Reference Sequence

- load A[0] load A[1] load A[2] … load A[2047]
- Assume base address of A (i.e., address of A[0]) is 0xA000, 1010 0000 0000 0000
  - Cache index bits: 100000000 (value = 256)
- Size of an array element (double) = 8B
- So, 4 consecutive array elements fit into each cache block (block size is 32B)
  - A[0] – A[3] have index of 256
  - A[4] – A[7] have index of 257 and so on

100000001 00000
100000001 01000
100000001 10000
100000001 11000

Example 1: Cache Misses and Hits

A[0]	0xA000	256
A[1]	0xA008	256
A[2]	0xA010	256
A[3]	0xA018	256
A[4]	0xA020	257
A[5]	0xA028	257
A[6]	0xA030	257
A[7]	0xA038	257
..	..	..
..	..	..
A[2044]	0xDFE0	255
A[2045]	0xDFE8	255
A[2046]	0xDFF0	255
A[2047]	0xDFF8	255

Example 1: Cache Misses and Hits

A[0]	0xA000	256	Miss	Cold start
A[1]	0xA008	256	Hit	
A[2]	0xA010	256	Hit	
A[3]	0xA018	256	Hit	
A[4]	0xA020	257	Miss	Cold start
A[5]	0xA028	257	Hit	
A[6]	0xA030	257	Hit	
A[7]	0xA038	257	Hit	
..	..	..	..	
..	..	..	..	
A[2044]	0xDFE0	255	Miss	Cold start
A[2045]	0xDFE8	255	Hit	
A[2046]	0xDFF0	255	Hit	
A[2047]	0xDFF8	255	Hit	

Cold start miss: we assume that the cache is initially empty. Also called a Compulsory Miss

Hit ratio of our loop is 75% -- there are 1536 hits out of 2048 memory accesses

This is entirely due to spatial locality of reference.

If the loop was preceded by a loop that accessed all array elements, the hit ratio of our loop would be 100%, 25% due to temporal locality and 75% due to spatial locality

Example 1 with double A[4096]

Why should it make a difference?

- Consider the case where the loop is preceded by another loop that accesses all array elements in order
- The entire array no longer fits into the cache – cache size: 16KB, array size: 32KB
- After execution of the previous loop, the second half of the array will be in cache
- Analysis: our loop sees misses as we just saw
- Called Capacity Misses as they would not be misses if the cache had been big enough

Example 2: Vector Dot Product

```
double A[2048], B[2048], sum=0.0;
for (i=0; i<2048, i++) sum = sum +A[i] * B[i];
```

- Reference sequence:
  - load A[0] load B[0] load A[1] load B[1] …
- Assume base addresses of A and B are 0xA000 and 0xE000
- Again, size of array elements is 8B so that 4 consecutive array elements fit into each cache block

Tag : 18b	**Index: 9b**	**Offset: 5b**

Example 2: Vector Dot Product

Tag : 18b	Index: 9b	Offset: 5b

		Tag	Index	Offset
A[0]	0xA000	000000000000000010	100000000	00000
B[0]	0xE000	000000000000000011	100000000	00000
A[1]	0xA008	000000000000000010	100000000	01000
B[1]	0xE008	000000000000000011	100000000	01000
A[2]	0xA010	000000000000000010	100000000	10000
B[2]	0xE010	000000000000000011	100000000	10000
A[3]	0xA018	000000000000000010	100000000	11000
B[3]	0xE018	000000000000000011	100000000	11000
..	..			
..	..			
..	..			

Example 2: Cache Hits and Misses

A[0]	0xA000	256	Miss	Cold start
B[0]	0xE000	256	Miss	Cold start
A[1]	0xA008	256	Miss	Conflict
B[1]	0xE008	256	Miss	Conflict
A[2]	0xA010	256	Miss	Conflict
B[2]	0xE010	256	Miss	Conflict
A[3]	0xA018	256	Miss	Conflict
B[3]	0xE018	256	Miss	Conflict
..	..	..	..	
..	..	..	..	
B[1023]	0xFFF8	511	Miss	Conflict

Conflict miss: a miss due to conflicts in cache block requirements from memory accesses of the same program

Hit ratio for our program: 0%

Source of the problem: the elements of arrays A and B accessed in order have the same cache index

Hit ratio would be better if the base address of B is such that these cache indices differ

Example 2 with Packing

- Assume that compiler assigns addresses as variables are encountered in declarations
- To shift base address of B enough to make cache index of B[0] different from that of A[0]

  double A[2048], d1, d2, d3, d4, B[2048];
- Base address of B is now 0xE020
  - 0xE020 is 1110 0000 0010 0000
  - ❑ Cache index of B[0] is 257; B[0] and A[0] do not conflict for the same cache block
- ■ Hit ratio of our loop would then be 75%