CS 354 - Machine Organization & Programming
Thursday, October 17, 2019

Project p3 (6%): DUE at 10 pm on Monday, October 28th

Homework hw3 (1.5%): DUE TOMORROW at 10 pm on Friday, October 18th

Last Time

Footers
Explicit Free List
Explicit Free List Improvements
Heap Caveats
Memory Hierarchy

Today

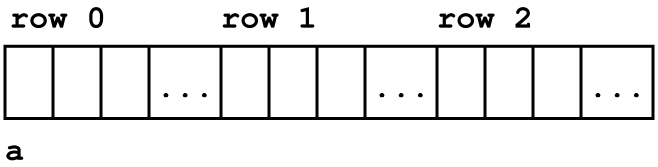
Memory Hierarchy (from last time)
Locality (from last time)
Bad Locality
Rethinking Addressing
Caching Basic Idea

Next Time

Designing Caches & Varying Set Size
Read: B&O 6.4.3 - 6.4.4

Bad Locality

Why is this code bad?



```
int a[ROWS][COLS];

for (int c = 0; c < COLS; c++)
   for (int r = 0; r < ROWS; r++)
      a[r][c] = r * c;
```

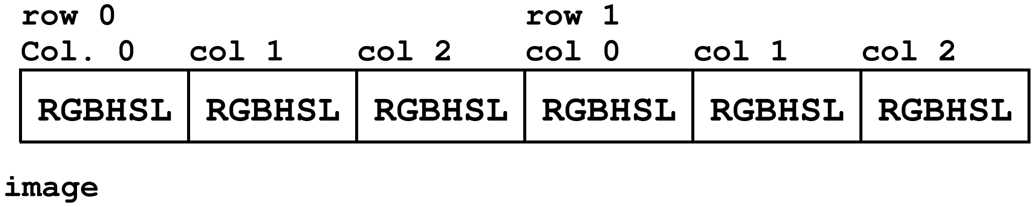
→ How would you improve the code to reduce stride?

Key Questions for Determining Spatial Locality:

1.

2.

Why is this code bad?



```
struct {
   float rgb[3];
   float hsl[3];
} image[HEIGHT][WIDTH];

for (int v = 0; v < 3; v++)
   for (int c = 0; c < WIDTH; c++)
      for (int r = 0; r < HEIGHT; r++) {
         image[r][c].rgb[v] = 0;
         image[r][c].hsl[v] = 0;
      }
```

➢ How would you improve the code to reduce stride?

Good or bad locality?

- Instruction Flow:
  sequencing?
  selection?
  repetition?

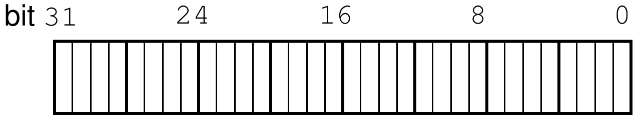
- Searching Algorithms:
  linear search

  binary search

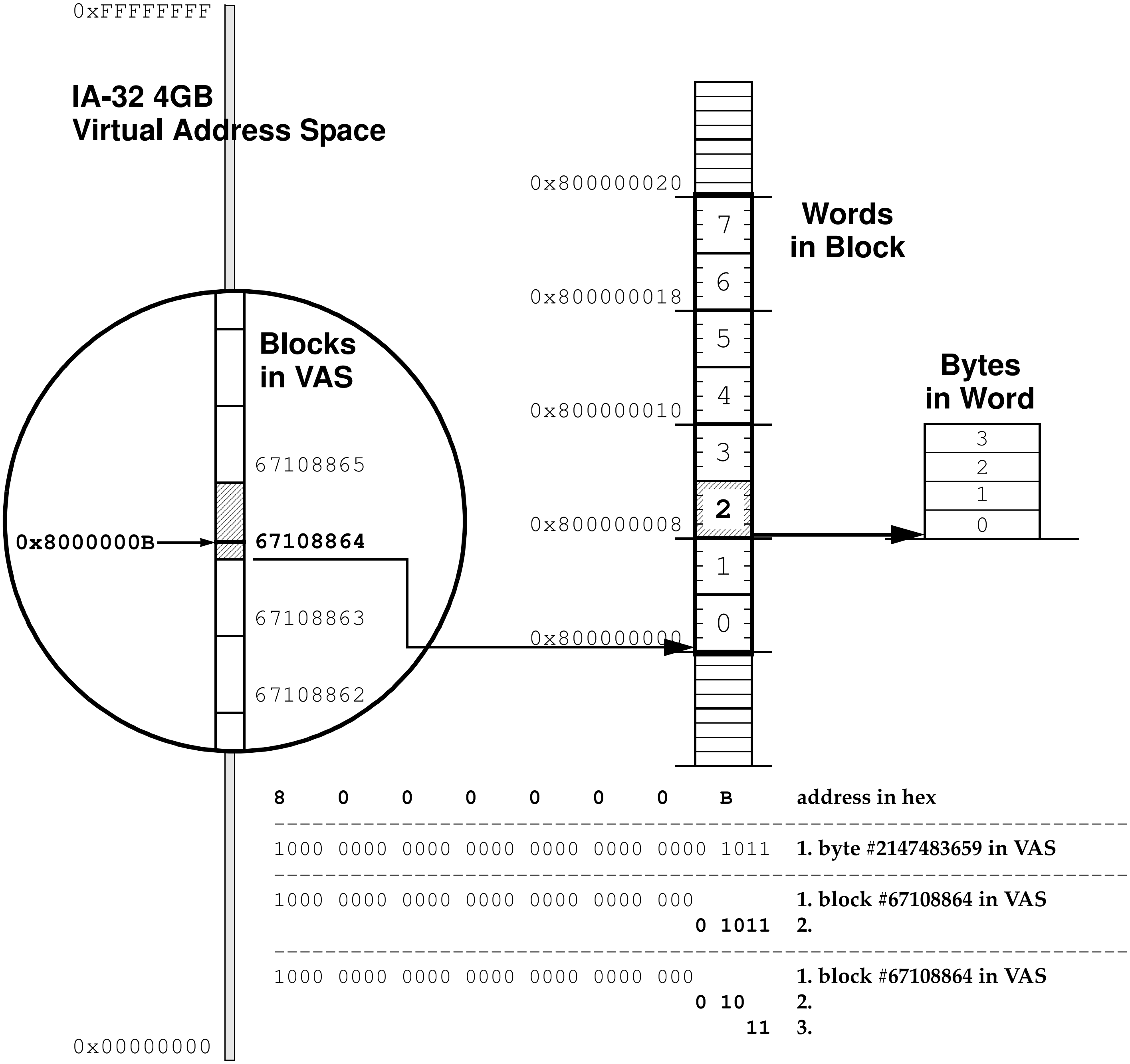
✵ *Programs with good locality*

Rethinking Adressing

- ✵ *An address identifies*
- ✵ *An address can be*



cache block is 32 bytes in IA-32
    step 1.
    step 2.
word is 4 bytes in IA-32



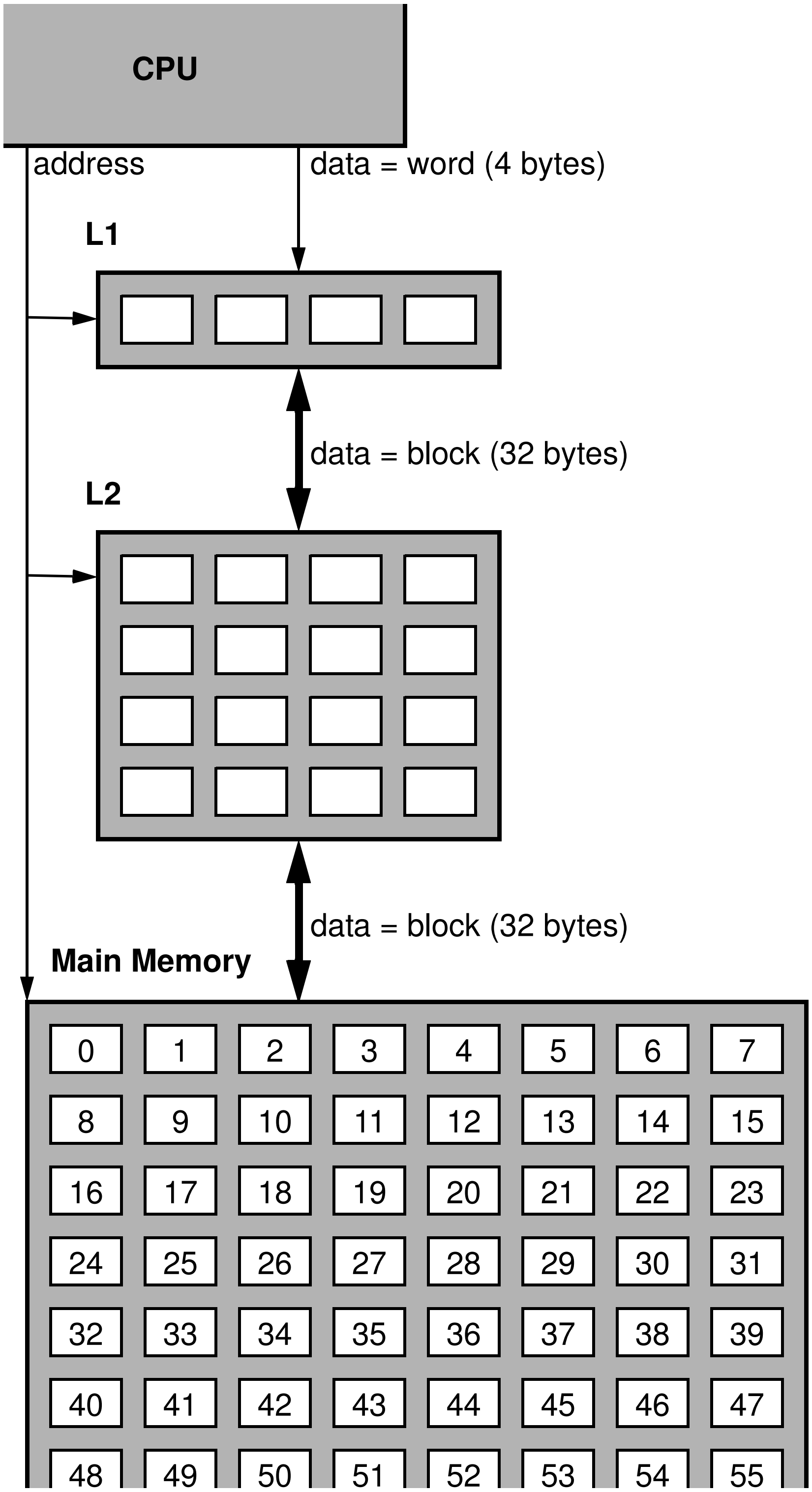
```
8    0    0    0    0    0    0    B       address in hex
---------------------------------------------------------------------------
1000 0000 0000 0000 0000 0000 0000 1011    1. byte #2147483659 in VAS
---------------------------------------------------------------------------
1000 0000 0000 0000 0000 0000 000          1. block #67108864 in VAS
                                  0 1011   2.
---------------------------------------------------------------------------
1000 0000 0000 0000 0000 0000 000          1. block #67108864 in VAS
                                  0 10     2.
                                      11   3.
```

Basic Caching Idea

Assume memory is divided into 32 byte blocks, and all needed blocks are already in main memory.
Cache L1 has 4 locations to store blocks and L2 has 16 locations to store blocks.

Consider the CPU accessing the following blocks in this sequence:

22,11,22,44,11,33,11,22,55,27,44



cache miss

cold miss

capacity miss

conflict miss

cache hit

placement policies

1.

2.

replacement policies

1.

2.

victim block

working set