SEPTEMBER 8TH, 2015 PRE-CLASS QUESTIONS

ELLIOT CARTEE

Question 1. Suppose 16N is significantly larger than the size of our L3 cache. What is the memory-based AI of this code? (Hint: What is the memory-based AI of just the innermost loop?)

Let us first consider the innermost loop:

```
for (k = 0; k < N; k++)
        tmp += A[i,k] * B[k,j]
```

Here we see that we are computing a total of $2N$ flops each time we complete this innermost loop. Since $16N$ is much larger than our L3 cache, we are getting no reuse of the cache, and therefore are also transferring $16N$ bytes from memory to cache every time we go through this entire loop. Since we are computing no other flops, the arithmetic intensity is:

$$(\#\text{ flops}) \ / \ (\#\text{ bytes transferred between memory and cache}) = \frac{2N}{16N} = \frac{1}{8}$$

Question 2. If the cache is substantially larger than $16N$, but substantially smaller than $8N^2$, then we cannot fit an entire matrix into cache, but we can fit an entire row (or column). Now in the innermost loop we are still doing $2N$ flops, but asymptotically we only need N memory accesses to write to the cache, as A is usually stored in cache and reused, while B is not. In this case we would have an arithmetic intensity of:

$$(\#\text{ flops}) \ / \ (\#\text{ bytes transferred between memory and cache}) = \frac{2N}{8N} = \frac{1}{4}$$

Question 3. If the cache is big enough to hold all of A ,B, and C, then we will have to transfer $3N^2$ double precision floating point numbers in and out of the cache, meaning that in total we are transferring $(2 \text{ transfers }) * (8 \text{ bytes per double }) * (3N^2 \text{ doubles }) = 24N^2$ bytes transferred.

Meanwhile, we compute $2N$ flops for each entry of C, meaning that we compute $2N^3$ flops total, giving us an arithmetic intensity of:

$$(\#\text{ flops}) \ / \ (\#\text{ bytes transferred between memory and cache}) = \frac{2N^3}{24N^2} = \frac{N}{12}$$

Question 4. For a problem size of N, there are $2N^2$ double precision floating point numbers that need to fit into the cache. Since each double is 8 bytes of memory, this means that the total memory required for problem size N is $16N^2$ bytes.

For the L1 cache with a size of 32KB, in order to fit the entire problem into cache, we need that $16N^2 \leq 2^{15}$. This happens when $N \leq \sqrt{2^{15}/16} = 45.25...$

For the L2 cache with a size of 256KB, in order to fit the entire problem into cache, we need that $16N^2 \leq 2^{18}$. This happens when $N \leq \sqrt{2^{18}/16} = 2^7 = 128$

For the L3 cache with a size of 6MB, in order to fit the entire problem into cache, we need that $16N^2 \leq 6 \cdot 2^{20}$. This happens when $N \leq \sqrt{6 \cdot 2^{20}/16} = 627.07...$ (I'm not sure of the exact number of bytes in a 6MB cache since this is not a power of 2....)

For each cache, you will have arithmetic intensity $\frac{1}{8}$ when $16N$ bytes is larger than the cache size, as this corresponds to the case of Problem 1 where a single row will not fit in the cache. You will get arithmetic intensity $\frac{1}{4}$ when the cache size is between $16N$ and $16N^2$, as here you can fit a row into the cache but not an entire matrix, as in Problem B. You will get arithmetic intensity $\frac{N}{12}$ when the cache size is greater than $24N^2$ as then you can fit both matrices into the cache, as in Problem 3.

Question 5. Your machine becomes CPU-bound when the arithmetic intensity is greater than the ratio between peak flop rate and peak memory bandwidth. This occurs when:

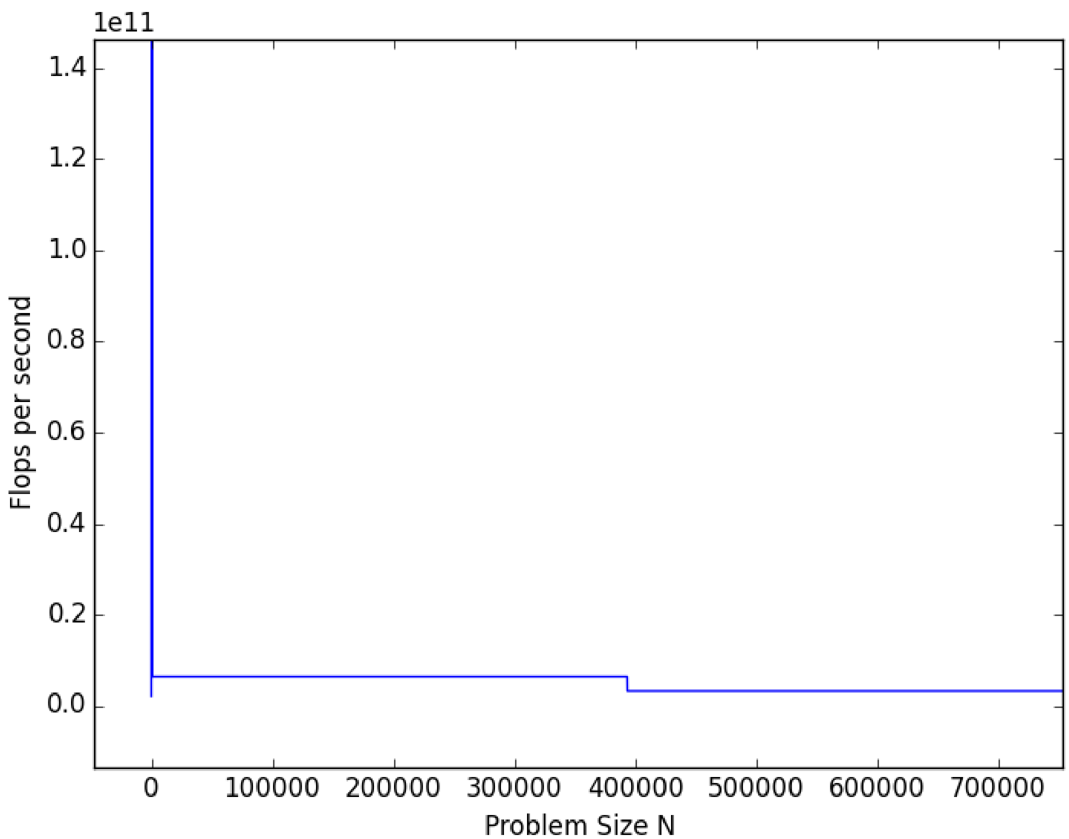
$$\text{arithmetic intensity} > \frac{\text{peak flop rate}}{\text{memory bandwidth}} = \frac{(4 \text{ cores })(16 \text{ flops per cycle })(2.4 \cdot 10^9 \text{ Cycles per second })}{25.6 \cdot 10^9 \text{ bytes per second}}$$

$$\text{arithmetic intensity} > 6 \text{ flops per byte transferred}$$

Question 6. Based on my above answer for Question 5, in order for this machine to become CPU-bound, the arithmetic intensity will need to be greater than 6 flops per byte. Based on my answer for Question 3, this occurs when the problem size N is bigger than 72, but all three matrices can fit in the cache, so when $24N^2$ bytes can fit in the cache. For the 6MB L3 cache, this occurs when $N \leq \sqrt{6 \cdot 2^{20}/24} = 512$. So the full size range is $72 \leq N \leq 512$.

Question 7. A plot of flops per second will show an increasing line for $N < 72$ as the machine is memory bound and the arithmetic intensity increases. Then a horizontal line for $72 \leq N \leq 512$ where the machine is CPU-bound. Then two more flat lines corresponding to the cases of Questions 1 and 2.

A plot made using `matplotlib` follows. It was generated using the code `flops_plot.py` located in this directory


1e11
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Flops per second
0
100000
200000
300000
400000
500000
600000
700000
Problem Size N