# CS398 Report A4

## Speed up for CPU vs GPU for different image sizes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CPU | | | GPU | | | | |
| A | B | Total | A | B | Copy | Total | Speedup |
| 512x512 | 0.0002 | 0.00075 | 0.00099 | 0.00014 | 0.00008 | 0.00026 | 0.00042 | 236% |
| 1024x1024 | 0.00071 | 0.00293 | 0.00373 | 0.00018 | 0.00015 | 0.00049 | 0.00076 | 491% |
| 2048x2048 | 0.00286 | 0.01199 | 0.01507 | 0.00039 | 0.00032 | 0.00141 | 0.00204 | 739% |

Table 1. Speed tests for each size of image comparison with speed up %.

From the table above, we can observe the difference between each A and B for CPU and GPU. The GPU total output takes into account the copy-ing of the buffer from GPU back to CPU as the memory and can only be used after the data is copied back from GPU.

## Answered Questions

For the section below we are only considering an image size of 512x512.

## Histogram

### Describe all optimizations you tried regardless of whether you committed to them or abandoned them and whether they improved or hurt performance.

Used privatization for reading and accumulating values in each block of threads and reducing number of floating point operations for setting colors. (putting it in A instead of B ).

### Were there any difficulties you had with completing the optimization correctly.

No. Instead the scan algorithm given in the slides was more difficult than the optimizations done.

### Which optimizations gave the most benefit.

Privatization of the values. Reducing number of floating point operations only affected higher number of image sizes.

### For the histogram kernel, how many global memory reads are being performed by your kernel? explain.

Number of global memory reads: width\* height\* channel size: 512\*512\*1 = 262, 144.

### For the histogram kernel, how many global memory writes are being performed by your kernel? explain.

Number of global memory writes: 256 \* block numbers. 256 \* (512-1/32) + 1)^2. 256\*16\*16: 65,536.

### For the histogram kernel, how many atomic operations are being performed by your kernel? explain.

Global writes: 256\*16\*16: 65,536.

Internal adds : 512 \* 512: 262,144  
Total: 256\*16\*16 + 512\*512: 327,680

### For the histogram kernel, what contentions would you expect if every element in the array has the same value?

It will be linear adding as every element will write into the same block element and every block will write into the same global element.

### For the histogram kernel, what contentions would you expect if every element in the input array has a random value?

As the elements are more diverse, there will be less compare and swap operations. Therefore each thread can write into the data parallelized.

## Scan

### How many floating operations are being performed in your reduction kernel? EXPLAIN.

Number of floating point operations performed:  
Block size \* number of floating point operations in block: 128 \* (3 divides, 1 SetColor)

: 128 \* (6+3) = 1152

### How many global memory reads are being performed by your kernel? EXPLAIN.

Number of global memory reads performed:

2\* block size = 256. (reading into shared memory)

### How many global memory writes are being performed by your kernel? EXPLAIN.

Number of global memory writes performed:

2\* block size = 256. (writing from shared memory).

### What is the minimum, maximum, and average number of real operations that a thread will perform? Real operations are those that directly contribute to the final reduction value.

Min = 2 + 6 +3 = 11.

Max = 2\*log(128) + 6 + 3 = 14 + 9 = 23.

Average = (max – min)/ 2 = 34/2 = 17.

### How many times does a single thread block synchronize to reduce its portion of the array to a single value?

Out of loop: 2 times. Once after pre and once after post.

In loop: 2\*log(128): 14

Total: 16

### Suppose a you want to scan using a binary operator that’s not commutative, can you use a parallel scan for that?

No. The algorithm does not preserve order.

### Is it possible to get different results from running the serial version and parallel version of scan? EXPLAIN.

Yes. Using Non-Commutative operators.