Parallel Programming Homework 4

All-Pairs Shortest Path (Single-GPU)

CS 107062546 楊仲愷

1. Implementation

(1) How do you divide your data? What's your configuration?

a. $\left\lfloor \frac{|Vertex|}{32} \right\rfloor$

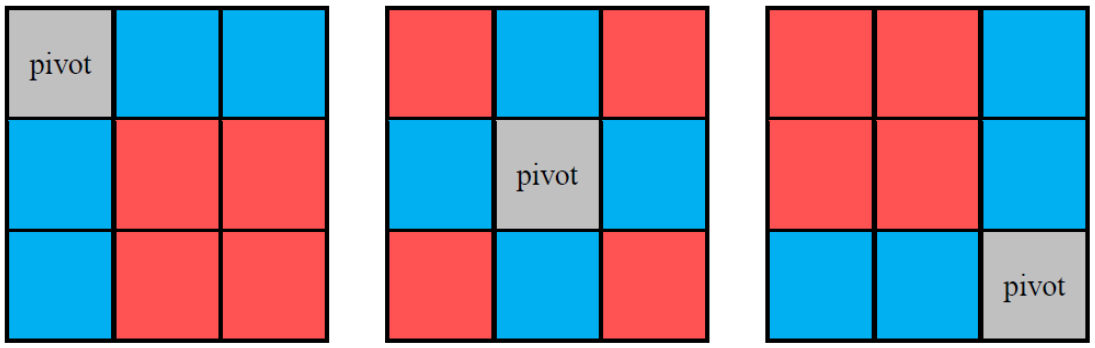
b. Blocking Factor: 32

c. Blocks: $(\frac{data\ \ amount}{32})^2$

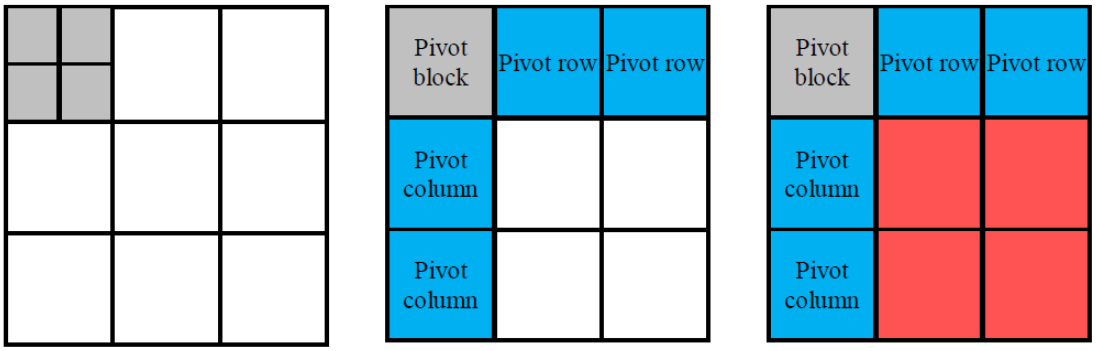
d. Threads: 32×32

(2) Briefly describe your impletion.

Based on the algorithm (i.e., blocked APSP algorithm),



(a) Round 1 (b) Round 2 (c) Round 3



(a) Phase 1 (b) Phase 2 (c) Phase 3

for each round, in phase 1, I divide the pivot block (i.e., the gray one) into 32 * 32 because of the maximum number of threads, and then execute the like FW algorithm in this block.

```
dim3 first(1);
dim3 second(num_round , 2);
dim3 third(num_round, num_round);
```

Afterwards, in phase 2, I have to utilize the block output based on last phase (i.e., phase 1). Thus, I have to firstly copy the value from global memory to shared memory or it will take a lot of time. I separate row and column into two different streams to calculate.

```
if (blockIdx.y == 1){
    #pragma unroll
    for (int m = 0; m < B && t_temp + m < n; m++){
        if (S_dist[s_temp + m] + pivot[m * B + shared_j] < S_dist[s_temp + shared_j]){
            S_dist[s_temp + shared_j] = S_dist[s_temp + m] + pivot[m * B + shared_j];
        }
    }
}else{
    #pragma unroll
    for (int m = 0; m < B && t_temp + m < n; m++){
        if (pivot[s_temp + m] + S_dist[m * B + shared_j] < S_dist[s_temp + shared_j]){
            S_dist[s_temp + shared_j] = pivot[s_temp + m] + S_dist[m * B + shared_j];
        }
    }
}
```

In the first condition, it will count the pivot row, and the other is to calculate the pivot column. Therefore, this will execute parallel. After counting all the distances, it has to be copied back to global memory.

Finally, in phase 3, take the top left block as an example, if I want to count its value, I have to get the output form the top one (pivot row) and the left one (pivot column) blocks. Similarly, it is necessary to copy the values from the blocks I needed.

Consider with how many times to execute it iteratively, and the pixel positions, these can be decomposed with the width or height, and the round times will also be the same.

2. Profiling Results

I take advantage of case p20k1 as measurement. I show occupancy, sm efficiency, shared memory load/store throughput, and global load/store throughput.

Block Dimension (32, 32):

Invocations	Metric Name	Metric Description	Min	Max	Avg
Device "GeForce GTX 1080 (0)"					
Kernel: phase1(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.494782	0.495898	0.495382
625	sm_efficiency	Multiprocessor Activity	0.00%	0.00%	0.00%
625	shared_load_throughput	Shared Memory Load Throughput	60.442GB/s	64.724GB/s	63.229GB/s
625	shared_store_throughput	Shared Memory Store Throughput	638.07MB/s	2.4272GB/s	988.18MB/s
625	gld_throughput	Global Load Throughput	2e+09GB/s	2e+09GB/s	2e+09GB/s
625	gst_throughput	Global Store Throughput	6e+08GB/s	7e+08GB/s	6e+08GB/s
Kernel: phase2(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.919195	0.925695	0.921890
625	sm_efficiency	Multiprocessor Activity	0.01%	0.01%	0.01%
625	shared_load_throughput	Shared Memory Load Throughput	1457.8GB/s	1521.9GB/s	1507.5GB/s
625	shared_store_throughput	Shared Memory Store Throughput	44.175GB/s	117.82GB/s	57.578GB/s
625	gld_throughput	Global Load Throughput	5e+07GB/s	5e+07GB/s	5e+07GB/s
625	gst_throughput	Global Store Throughput	2e+07GB/s	2e+07GB/s	2e+07GB/s
Kernel: phase3(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.905085	0.906583	0.905943
625	sm_efficiency	Multiprocessor Activity	1.91%	1.93%	1.92%
625	shared_load_throughput	Shared Memory Load Throughput	2749.9GB/s	2876.0GB/s	2837.3GB/s
625	shared_store_throughput	Shared Memory Store Throughput	84.679GB/s	88.658GB/s	87.386GB/s
625	gld_throughput	Global Load Throughput	3e+05GB/s	3e+05GB/s	3e+05GB/s
625	gst_throughput	Global Store Throughput	1e+05GB/s	1e+05GB/s	1e+05GB/s

Block Dimension (64, 64):

Invocations	Metric Name	Metric Description	Min	Max	Avg
Device "GeForce GTX 1080 (0)"					
Kernel: phase1(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.494874	0.496148	0.495385
625	sm_efficiency	Multiprocessor Activity	0.00%	0.00%	0.00%
625	shared_load_throughput	Shared Memory Load Throughput	61.733GB/s	64.724GB/s	63.614GB/s
625	shared_store_throughput	Shared Memory Store Throughput	653.98MB/s	2.5570GB/s	994.20MB/s
625	gld_throughput	Global Load Throughput	2e+09GB/s	2e+09GB/s	2e+09GB/s
625	gst_throughput	Global Store Throughput	6e+08GB/s	7e+08GB/s	6e+08GB/s
Kernel: phase2(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.919318	0.925248	0.921625
625	sm_efficiency	Multiprocessor Activity	0.01%	0.01%	0.01%
625	shared_load_throughput	Shared Memory Load Throughput	1497.5GB/s	1519.2GB/s	1514.1GB/s
625	shared_store_throughput	Shared Memory Store Throughput	45.716GB/s	117.55GB/s	57.828GB/s
625	gld_throughput	Global Load Throughput	5e+07GB/s	5e+07GB/s	5e+07GB/s
625	gst_throughput	Global Store Throughput	2e+07GB/s	2e+07GB/s	2e+07GB/s
Kernel: phase3(int, int, int, int*, int)					
625	achieved_occupancy	Achieved Occupancy	0.904866	0.906502	0.905906
625	sm_efficiency	Multiprocessor Activity	1.91%	1.93%	1.92%
625	shared_load_throughput	Shared Memory Load Throughput	2803.9GB/s	2869.0GB/s	2846.4GB/s
625	shared_store_throughput	Shared Memory Store Throughput	86.173GB/s	88.442GB/s	87.666GB/s
625	gld_throughput	Global Load Throughput	3e+05GB/s	3e+05GB/s	3e+05GB/s
625	gst_throughput	Global Store Throughput	1e+05GB/s	1e+05GB/s	1e+05GB/s

Discussion: In occupancy, block dimension with 16x16 is less than block dimension 32x32; hence, activate warp quantity is less at the same time. Because for each block, there are only 1024 threads launching with GeForce GTX 1080. Therefore, I can only set blocking factor as 32.

3. Experiment & Analysis

(1) System Spec

Run on the hades server.

(2) Time Distribution

Test cases: <vector, edge>

pp11k1: <11000, 505586>

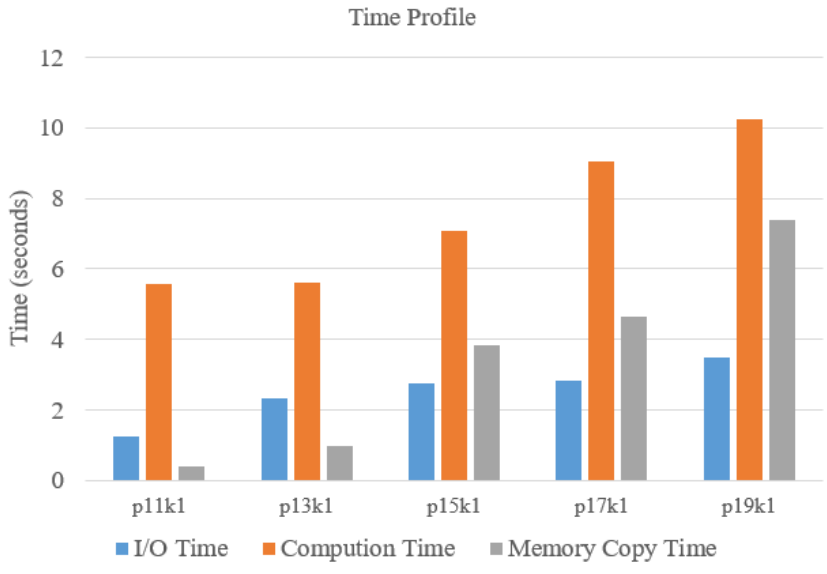
pp13k1: <13000, 1829967>

pp15k1: <15000, 5591272>

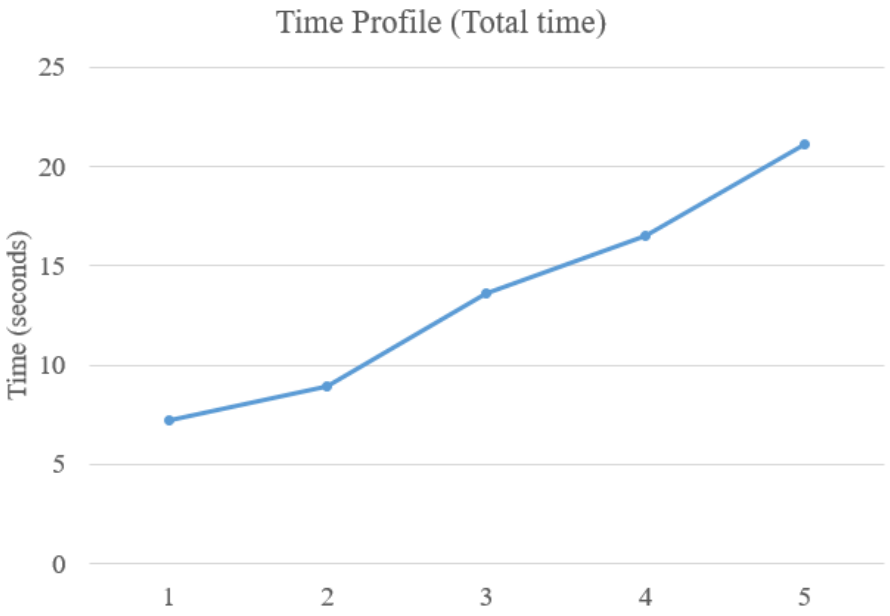
pp17k1: <17000, 4326829>

pp19k1: <19000, 333397>

Time Profile:



Total Time:

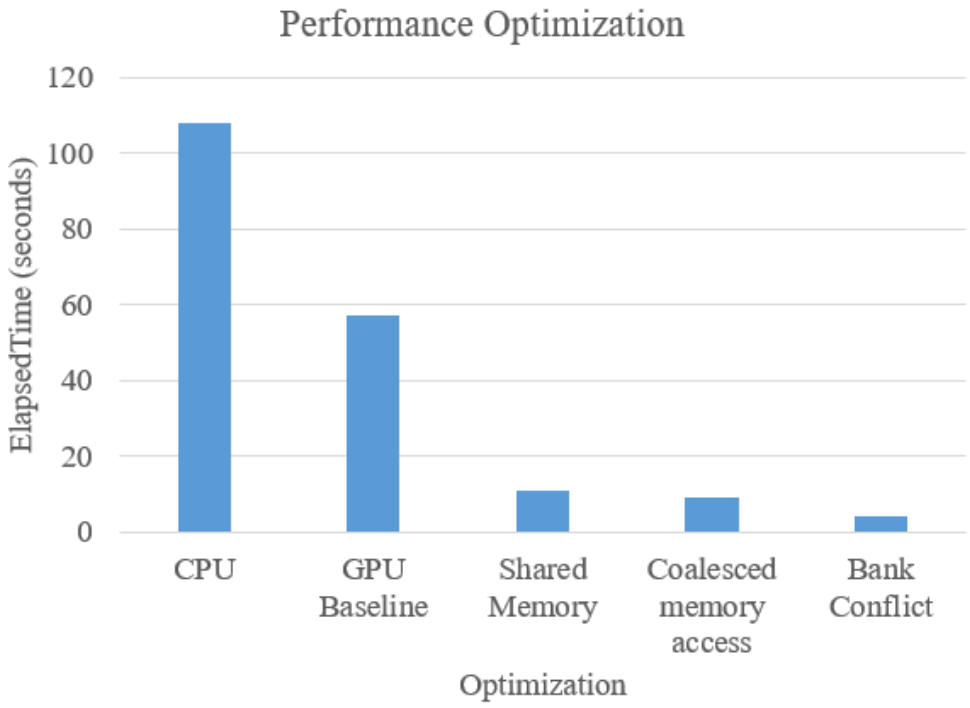


Value:

	I/O Time	mputionTiory Copy	Total Time
p11k1	1.24102	5.574629	0.386189	7.201838
p13k1	2.278541	5.621845	0.968774	8.86916
p15k1	2.784213	7.060258	3.82505	13.66952
p17k1	2.834678	9.05351	4.621716	16.5099
p19k1	3.48125	10.2576	7.375341	21.11419

(3) Optimization

I utilize cases c21.1 as measurement.



4. Conclusion

In this homework, I learned a lot about the speedup with making good use of memory (from global memory to shared memory). Also, I have to make them execute parallel, and how to allocate threads, stream, banks, etc., to optimize the performance. Moreover, I have to solve the problem with bank conflicts to avoid wasting of time.