CUDA Parallel Programming Problem Set 6

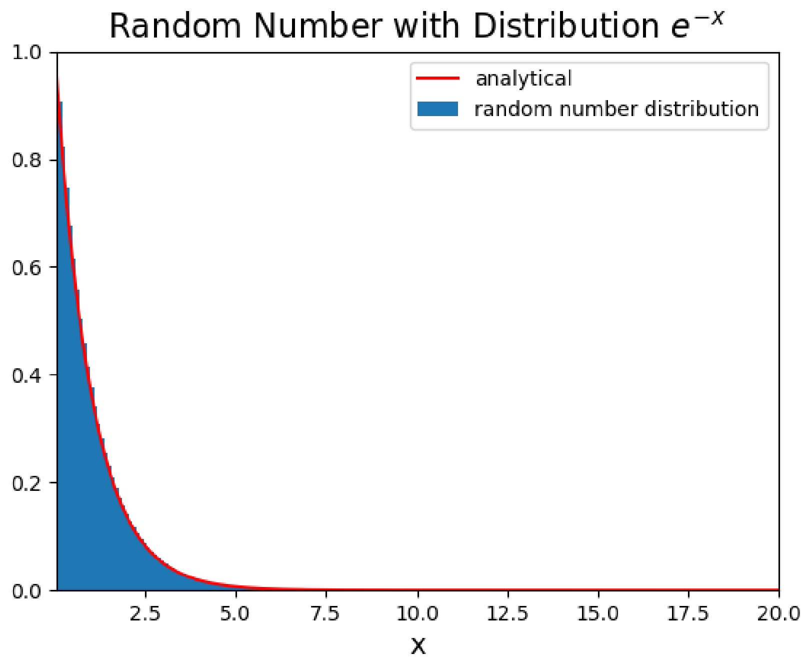
R08244002 Shin-Rong Tsai

• Histogram of a data set with exponential distribution

I. Result

Add the algorithm for generating exponential decay distribution in the code introduced in course, save in file *hist_1gpu_gmem.cu* and *hist_1gpu_shmem.cu*. To find the optimal block size for both GPU with global memory and shared memory, run through all the block size and grid size. Codes are saved in file *hist_1gpu_gmem_Optimize.cu* and *hist_1gpu_shmem_Optimize.cu*.

Each of them gives the same result for $[R_{min}, R_{max}] = [0{,}100]$ with $\mathrm{N} = 81920000$. Plot by *plot.py*.



II. Discussion

Since the time speed by CPU depends on the length of the data vector and the number of bins only, and they are all fall in the range between 388~392ms. So for rest of the discussion, I'll just compare the

$$\text{speed up rate} = \frac{cpu\ \ time}{gpu\ \ time\ \ total}$$

of GPU using global memory and GPU using shared memory directly.

A. Compare global memory vs shared memory with different number of bins.

I pick the maximum speed up rate for GPU-global memory in different block size and grid size, and compared with the maximum speed up rate for GPU-shared memory in different grid size, cause block size must equal to number of bins in GPU-shared memory.

# of bins / SpeedUp	128	256	512	1024
Global	3.33	3.64	3.75	3.66
Shared	6.19	6.22	6.24	6.26
$Shared/Global$	1.86	1.71	1.67	1.71

1. Speed up rate for GPU-global and GPU-shared are all larger than one, which means they are all faster than CPU. Since the calculating of histogram requires division, which is a rather heavy operation.
2. Using shared memory did speed up the whole process, since write to shared memory is a lot faster than write to global memory. The number of write to memory process is the same as data length, so $Shared/Global$ should not varies that much. I wonder why there is a boost in 128 number of bins.

B. Optimal block size and grid size for GPU-global memory.

The speed up rate for different number of bins act the same in different block size and grid size, so I'll only show the 1024 bins case.

Speed up rate of GPU-global memory, 1024 bins:

BlockSize / GridSize	2	4	8	16	32	64	128	256	512	1024
10^1	0.24	0.41	0.77	1.37	2.25	3.34	3.64	3.53	3.66	3.66
10^2	1.53	2.60	3.11	3.65	3.66	3.37	3.66	3.65	3.66	3.66
10^3	2.56	3.13	3.16	3.64	3.66	3.53	3.66	3.66	3.66	3.66
10^4	2.62	3.16	3.18	3.36	3.66	3.66	3.66	3.65	3.37	3.65
10^5	2.63	3.17	3.18	3.57	3.66	3.66	3.65	3.66	3.53	3.64
10^6	2.63	2.88	3.18	3.65	3.66	3.65	3.35	3.57	3.44	3.15

1. There is a balance between block size and grid size, since the number and power of streaming processor is limited.

2. The optimal block size and grid size for GPU-global memory is not that obvious, so long as $\text{BlockSize} \times \text{GridSize}$ falls in the yellow block, then it is optimized.

C. <u>Optimal block size and grid size for GPU-shared memory.</u>

Speed up rate of GPU-shared memory, with different number of bins, block size, and grid size:

BlockSize / # of bins / GridSize	2	4	8	16	32	64	128	256	512	1024
10^1	0.24	0.41	0.75	1.30	2.10	3.20	4.31	5.11	5.77	6.12
10^2	1.49	2.57	3.60	4.60	5.33	5.70	6.11	6.22	6.24	6.26
10^3	3.34	4.37	5.17	5.70	5.89	5.93	6.19	6.22	6.23	6.25
10^4	3.62	4.61	5.32	5.72	5.98	6.07	6.19	6.22	6.22	6.25
10^5	3.64	4.60	5.32	5.77	5.97	6.07	6.16	6.05	6.05	5.95
10^6	3.64	4.42	5.31	5.71	5.50	4.82	4.62	3.86	3.87	3.81

1. As number of bins becomes larger and with a proper grid size, the speed up rate is higher.
2. It also shows a balance between block size and grid size, though unlike GPU-global, the overall optimal block size and grid size is centered at $\text{block size} = 1024$ and $\text{grid size} = 100$. I guess that it is the matter of write to shared memory and write to global memory.