CUDA Parallel Programming
Homework 4

40647007S 朱健愷

April 19, 2021

1 Source codes

1.1 File Layout

- vecDotProduct_NGPU/vecDotProduct_ngpu.cu - First problem's main code used to calculate vector dot product that can utilize multi GPU
- headDiffusion2D_NGPU/heatDiffusion2D_NGPU.cu - Second problem's main code used to calculate heat diffusion in 2D grid using $\omega = 1$
- Makefile - Script to auto generate executable from code
- vecDotProduct_NGPU/experiment.sh - Script to auto generate results of first problem's vector dot product using multiple GPUs with different block size and grid size
- heatDiffusion_NGPU/experiment.sh - Script to auto generate results of solving second problem's thermal equilibrium temperature distribution on a square plate with different block size
- vecDotProduct_NGPU/result/1GPU_block_*_grid_*/ - Output result of solving first problem's vector dot product with 1 GPU and different block size and grid size, the suffix represented the block size and grid size respectively. In this directory contains statistical result of calculating vector dot product.

- vecDotProduct_NGPU/result/2GPU_block_*_grid_*/ - Output result of solving first problem's vector dot product with 2 GPU and different block size and grid size, the suffix represented the block size and grid size respectively. In this directory contains statistical result of calculating vector dot product.
- heatDiffusion2D_NGPU/result/1GPU_block_* - Output result of solving second problem's 2D heat diffusion using 1 GPU and different block size, the suffix represent the block size. In this directory contains initial phi data, phi data calculated by GPU, and statistical result of solving 2D heat diffusion problem.
- heatDiffusion2D_NGPU/result/2GPU_block_* - Output result of solving second problem's 2D heat diffusion using 2 GPU and different block size, the suffix represent the block size. In this directory contains initial phi data, phi data calculated by GPU, and statistical result of solving 2D heat diffusion problem.
- notebook/*.png - Plots concluding output result

1.2 Usage

Make code in both vecDotProduct_NGPU/ and heatDiffusion2D_NGPU/ directories Run the experiment.sh script in both vecDotProduct_NGPU/ and heatDiffusion2D_NGPU/ directories

```
cd vecDotProduct_NGPU
make
sh experiment.sh
cd ../heatDiffusion2D_NGPU
make
sh experiment.sh
```

And it will produce two problems statistical result using different block size and grid size.

1.3 Code design

In first problem, the program divide the vector into N equal portion of the original vector and use N GPU to calculate their dot product, and finally used CPU to gather N GPU's vector dot product result.

In second problem, it divide the grid into $Nx \times Ny$ equal porition of the original grid and use $Nx \times Ny$ GPU to calculate the heat diffusion equation. While they are calculating the boundary, they will determine whether this is the true boundary which represent the edge of the plate or just the divided boundary that border to another portion of grid calculated by another GPU, and let CPU calculate the difference to determine whether it will perform the next grid update to minimize the error in consecutive iteration.

2 Result

2.1 Experiment environment

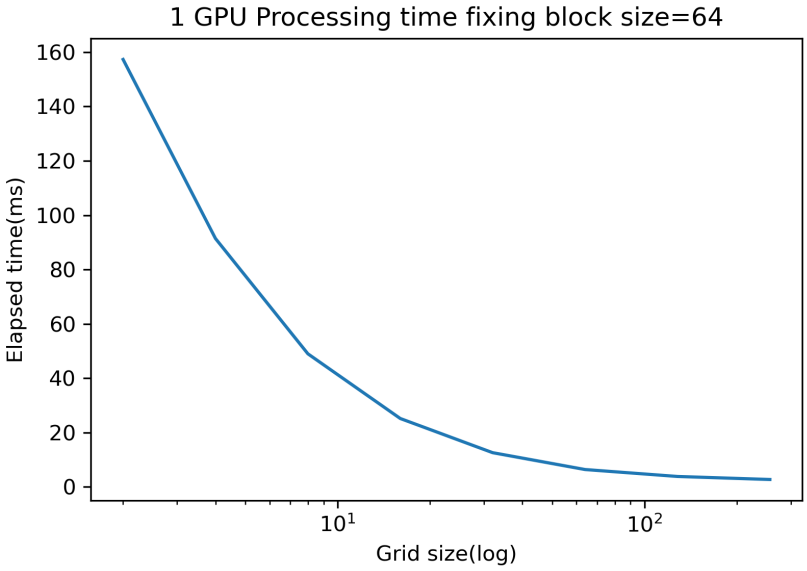
I ran my code on workstation provided in course, below is the Setup of workstation

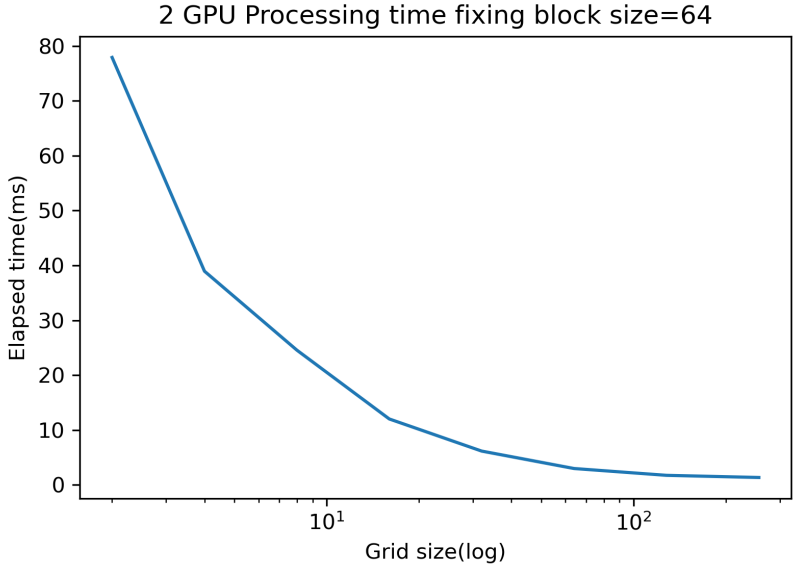
- Operating system: Linux version 4.19.172 (root@twcp1)
  (gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1))
- CPU: Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz
- GPU: Nvidia GTX 1060 6GB
- Memory: 32GB

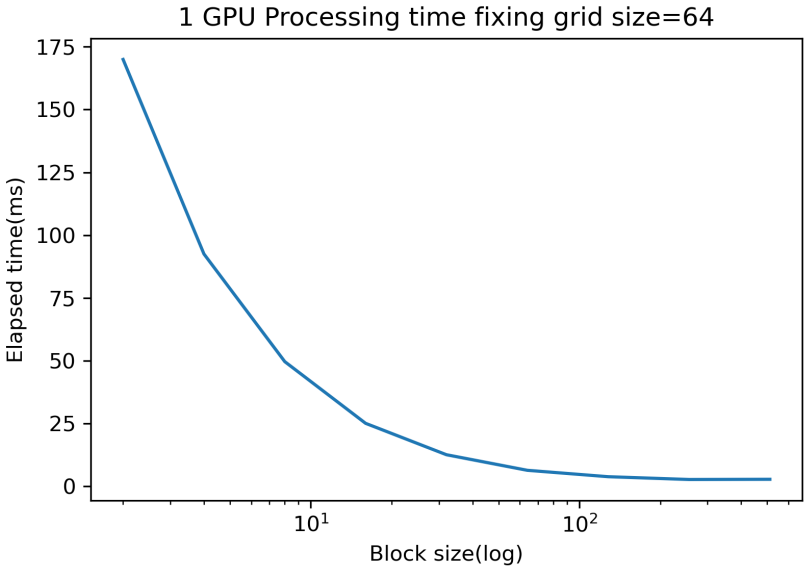
2.2 Performance

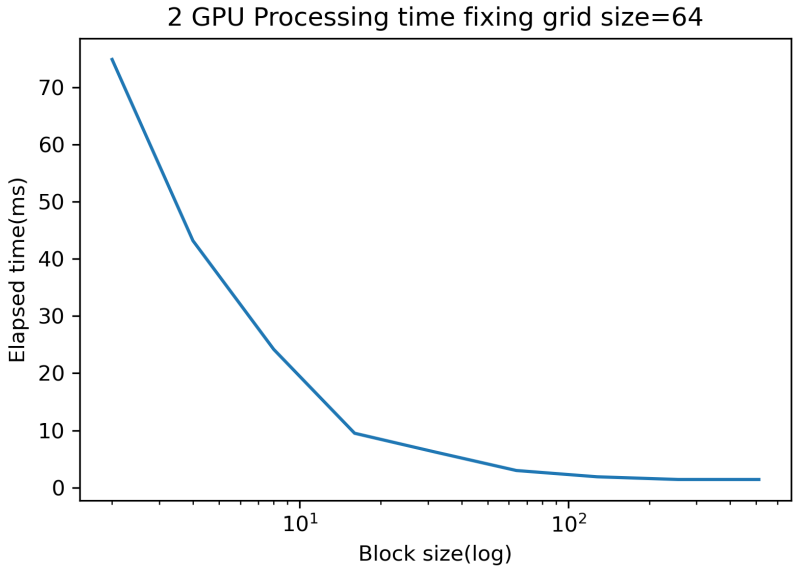
2.2.1 First problem

Below four figures showed 1 GPU case and 2 GPU case performing dot product fixing one of block size or grid size and change the other one respectively

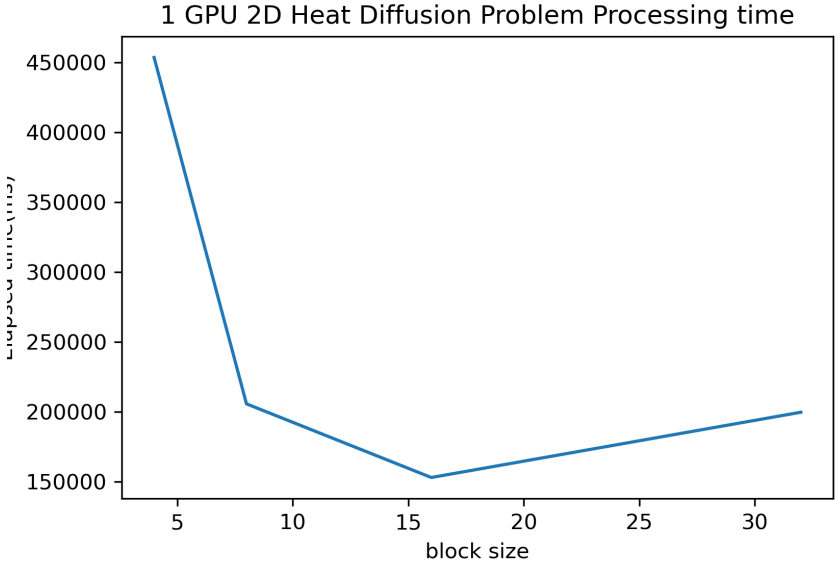

1 GPU Processing time fixing block size=64
Elapsed time(ms)
160
140
120
100
80
60
40
20
0
10^1
10^2
Grid size(log)

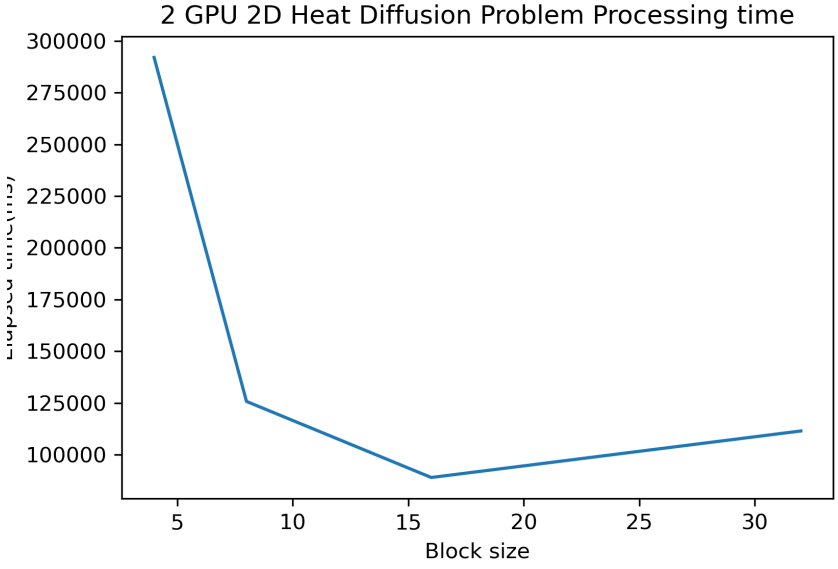

2 GPU Processing time fixing block size=64
Elapsed time(ms)
80
70
60
50
40
30
20
10
0
10^1
10^2
Grid size(log)





Below two figures showed 1 GPU case and 2 GPU case solving the 2D heat diffusion problem using different block size.


1 GPU 2D Heat Diffusion Problem Processing time
450000
400000
350000
300000
250000
200000
150000
5
10
15
20
25
30
block size


2 GPU 2D Heat Diffusion Problem Processing time
300000
275000
250000
225000
200000
175000
150000
125000
100000
5
10
15
20
25
30
Block size

2.2.2 Observation

We can observe that the performance of small block size setup in both 1 GPU case and 2 GPU case yield the worse performance in my experiment. This may because of when we do parallel reduction, we usually need many threads in block in order to collect each block's difference, and then gather the result using CPU, and more blocks in grid in order to collect dot product data which range are out of basic per threads in grid. So the block size and grid size do affect the performance a lot.

The result of utilizing 2 GPU in both problems helps improve the performance in nearly every block size and grid size setup(except for very large grid or block size setup), in my observation, they spent nearly half times of calculation time as 1 GPU cases in both problem. But same as the observation I observed in the previous assignment, the performance increase until block size or grid size increased to some extent.

The optimal block size and grid size setup for first problem in my experiment is 256 block size and 256 grid size for 1 GPU case, 256 block size and 64 grid size for 2 GPU case.

The optimal block size setup for second problem in my experiment is 16 block size for 1 GPU case and 16 block size for 2 GPU case.

3 Discussion

I thought their speed up is due to divide the question into two equal portion and solve them. So we can see the approximately one times speed up compared to 1 GPU case in both problem. Although the second problem demanded that 2 GPU have to communicate between boundary to solve the problem, that's not the bottleneck, because in this two problem, synchronizing between 2 GPU, 4 GPU or even more GPU requires fix GPU bridge amount(In first problem 0 bridge is required, in second problem up to 4 bridges for each GPU is required), unless the problem scale is arbitrary large or too many GPU to synchronize that had filled the GPU bandwidth of transferring data.