cudahw

目标

本次课程作业通过编写CUDA版本的矩阵矩阵乘法（GEMM，包括SGEMM和DGEMM）使同学熟悉GPU上的CUDA编程模型。鼓励大家尝试不同的优化策略。

问题描述

在数学领域中，矩阵乘法将两个矩阵进行相乘，得出另一个矩阵。矩阵运算是许多科学计算问题的基础，应用广泛。GPU和CPU处理器相比，具有较高的计算能力，特别适合矩阵运算等能够进行高并发的算术操作。本次作业在GPU上实现矩阵－矩阵运算。下式中，α, β为常数，A、B、C为矩阵。

$$C = \alpha A * B + \beta C$$

在我们的实验框架中为了简便，设置$\alpha = 1.5, \beta = 0.5$，其中矩阵的大小，`A`为`M*K`，`B`为`K*N`，`C`为`M*N`

框架介绍

在实验框架中已经实现了baseline版本（`gemm`函数）和库函数版本（cublas，性能很好）。并且已经和这两种版本进行了性能的比较。

同学们只需要在`include/gemm.h`里面实现你自己的gemm函数（注意不要修改接口）。

考虑到要达到最优的性能，一些同学可能会根据输入的参数（M，N，K）做一些预处理之后再进行计算，所以在`include/gemm.h`的`myGEMM`预留好了两种接口，一种是无需预处理（将`preprocess`设置为`false`），只需要把自己的gemmkernel实现在`myGEMM`函数里即可；另一种是需要预处理（请将`preprocess`设置为`false`），这种情况下需要让`myGEMM`返回自己计时得到的时间（使用了预处理的话，自己计时请注意**在恰当的位置**添加cuda同步函数（`cudaDeviceSynchronize()`），防止kernel还没结束就结束计时，导致得到的运行时间偏小）

How to run

```
git clone https://github.com/xxcclong/hw_cuda_gemm.git
cd hw_cuda_gemm
nvcc -std=c++11 main.cu -I./include -lcublas -lcurand -o gemm
srun -p gpu ./gemm --m 300 --n 400 --k 500 --iter 10
```

其中可以自己用命令行参数设置M，N，K来测试性能

提交

假设你的学号是`123456`

请提交，注意保证**提交的gemm头文件能编译通过且结果正确**

```
|-- gemm_123456.h
`-- report_123456.pdf
```

如果在项目文件做了额外的对性能的测试的话，也可以提交工程文件

```
|-- gemm_123456.h
|-- hw_cuda_gemm_123456
|   |-- include
|   |   |-- args.hxx
|   |   |-- gemm.h
|   |   `-- util.h
|   `-- main.cu
`-- report_123456.pdf
```

对于报告的要求，希望可以多看到一些对不同规模矩阵的测试和分析（毕竟框架已经完成绝大部分的profiling的代码了）。以及如果自己的优化是一步步做出来的，可以写上每一步代码修改的思路以及性能的改变以及自己的理解和分析。

注意事项

- 课程集群一共只有2个GPU节点，请注意不要让自己的任务死在上面
- 得分标准：实现优于baseline的版本（4）+ 报告性能分析以及改进（4）+ performance bonus（1）
- 因为框架的实现里难免有问题或者表述不清的地方，所以采用git的方式布置作业(https://github.com/xxcclong/hw_cuda_gemm)，如果对框架有修改会在群里通知大家pull repo（pull repo的时候注意不要把自己的改动弄丢了）