Lab 4

CMPUT 398

Objective

It this lab you will implement three versions of matrix multiplication. The first implementation will be a C version that runs on the CPU, the second will be basic dense matrix multiplication routine written in CUDA, and the third will be a tiled dense matrix multiplication routine using shared memory.

This lab will be submitted as one zipped file through eclass. Details for submission are at the end of the lab.

Instructions

Edit the code where the TODOs are specified and perform the following:

- Allocate device memory
- Copy host memory to device
- Initialize thread block and kernel grid dimensions
- Invoke CUDA kernel
- Copy results from device to host
- Deallocate device memory
- Implement the CPU matrix-matrix multiplication routine
- Implement the basic GPU matrix-matrix multiplication routine
- Implement the matrix-matrix multiplication routine using shared memory and tiling

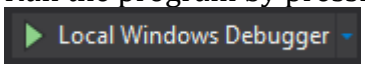
Local Setup Instructions

Steps:

1. Download “Lab4.zip”.
2. Unzip the file.
3. Open the Visual Studios Solution in Visual Studios 2013.
4. Build the project. Note the project has two configurations.
    a. Debug
    b. Submission

But make sure you have the “Submission” configuration selected when you finally submit.

5. Run the program by pressing the following button:



Make sure the “Debug” configuration is selected and the project you wish to run is selected in the “Solution Explorer”. To select the project just make sure you click on it before you run. The title of the program is printed at the top of the output console if you are unsure.

Running the program in Visual Studios will run one of the tests located in “Dataset/Test”.

Testing

To test run all tests located in “Dataset/Test”, first build the project with the “Submission” configuration selected. Make sure you see the “Submission” folder and all three executables are in the folder: CPU_MatMul.exe, GPU_MatMul.exe, and OPT_MatlMul.exe. If you are missing one of the executables because of build errors the test script should still work.

To run the tests, click on “Testing_Script.bat”. This will take a couple of seconds to run and the terminal should close when finished. The output is saved in “Marks.js”, but to view the calculated grade open “Grade.html” in a browser. If you make changes and rerun the tests, then make sure you reload “Grade.html”. You can double check with the timestamp at the top of the page.

In the test script if GPU_MatMul.exe fails then OPT_MatMul.exe will fail since the goal is to see the speed up between the basic matrix multiplication and the tiled matrix multiplication. See the Mark Breakdown for more information.

Report

Create a report on speed up of the tiled dense matrix multiplication compare to the normal gpu matrix multiplication algorithm using NSIGHT on the last test case (number 9). You can just put the two screenshots from NSIGHT (just like the previous labs) for both algorithm, as well as the speed up in a pdf or doc file. You will lose mark if you do not include this report in your submission. The file name should be “report.pdf” saved in the “Submission” folder.

Submission

After you build your project with the “Submission” configuration selected (see above) you will see a new folder called “Submission” in the project directory. If you open the folder you will see your three executables. Finally zip the folder “Submission” and upload the zipped folder to eclass. Remember to add the report also.

If you don’t see this folder then one of two things happened. First your build could have failed and you should check for any errors in your build log in Visual Studios. Also you might have had the “Debug” configuration selected (see above). Make sure the “Submission” configuration is selected.

Mark Breakdown

Note that you will get zero marks if the Tiled Matrix Multiplication is not significantly faster than the Basic Matrix Multiplication. If you implemented the Tile Matrix Multiplication correctly and using shared memory you should have no issues. You should be getting a speed up of about 2 to 3.

It is important that you **do not** add or remove any print statements or wb functions. We use them for part of your grading so if you make changes the marking script could fail.

Part	Breakdown (%)
CPU Code	10 %
Basic Matrix Multiplication	40 %
Tiled Matrix Multiplication	50%