CSCI-UA.0480-003
Parallel Computing
Lab 3

Before you start:

- To calculate the time of each program in this lab use the Linux command time.
- After you login to your CIMS account, you need to ssh to one of the following: cuda1, cuda2, cuda3, or cuda4
- The source code, containing both device and host code, has extension .cu
- You compile with **nvcc** progname.cu
- Don't forget to #include <cuda.h>
- A very useful API is **cudaGetDeviceProperties**() check it up.

1. Assume a reduction algorithm that finds the maximum of an array of 8192 integers. You will need to write a host function that fills the array with random integers between 1 and 100000.

   A. Write the sequential version of the program in C. Note that the sequential version will scan the array sequentially from start to end. Call it **seq8192.c**.

   B. Write a CUDA version of the program that does not take thread divergence into account. Call it **cuda81192.cu**.

   C. Update the version in B to take thread divergence into account. Call it **cudadiv8192.cu**.

   D. Update the program in C to make use of shared memory to reduce global memory bandwidth. Call it **cudashared8192.cu**.

Draw a bar graph that compares the execution time of each of the above 4 versions. That is, x-axis contains the 4 versions (for each one report the real, user, and sys) and the y-axis contains the time. So, we expect to see 12 bars (4 versions and 3 timing each).

2. Repeat problem 1 with an array of 65536 elements. Adjust the file names based on the new number.

3. What can we conclude from the results of problems 1 and 2 regarding the optimizations and the problem size?

What to submit (in single zip file called lab3.zip)?

- 8 source code files
- 1 pdf file for your conclusions of #3