18-640 Fall 2014
Project 5 – Performance and Power Analysis of CUDA application
Due on: 12/8/2014 11:59 pm EST

Task 1: Install and build GPGPU-Sim

1. Download CUDA Toolkit and GPU Computing SDK (https://developer.nvidia.com/cuda-toolkit-40).

2. Download and install the dependencies listed below:
   gcc g++ make makedepend xutils bison flex zlib python-pmw, python-ply python-numpy python-matpotlib libpng12-dev doxygen graphviz libcppunit-doc libcppunit-dev

3. Setup the below environment variables:
   1. CUDAHOME: cuda install location
   2. CUDA_INSTALL_PATH: cuda install location
   3. NVIDIA_CUDA_SDK_LOCATION: sdk location
   4. PATH: add CUDAHOME/bin
   5. LD_LIBRARY_PATH: add $CUDAHOME/lib

4. Download gpgpu-sim using the below command:
   git clone git://dev.ece.ubc.ca/gpgpu-sim
   If you have not set up git, please following the instructions in the below link: https://help.github.com/articles/set-up-git/

5. Run the below commands inside v3.x directory in gpgpu-sim:
   source setup_environment release
   make
   make docs

If the above steps are followed correctly, you should be able to install gpgpu-sim successfully.

Task 2: Analyze the power and Performance statistics of the simulation output **30 points**

1. The folder for project 5 includes CUDA Source code for basic matrix multiplication, test program and makefile used to compile the program.

2. A basic version of the matrix multiplication program is provided under the folder /afs/ece/class/ece640/project/project5. Run make inside the folder to compile it.

3. In order to run the application, you have to copy the contents of v3.x/configs/GTX480/ to your project folder and run source setup_environment under gpgpu-sim/v3.x folder.

4. Experiment with the .config file by changing various configuration parameters. [Do not change -gpgpu_ptx_sim_mode parameter]. Experiment with different block and grid sizes and report the results.

5. Run the compiled executable for matrix multiplication and obtain the following statistics after running the program:
Gflops,gpgpu_simulation_rate,avg row locality,
L1I_total_cache_miss_rate, L1D_total_cache_miss_rate,
L2_total_cache_miss_rate

6. Also obtain the avg_power, min_power, max_power statistics from the below log file:
gpgpusim_power_report__Date.log.

Task 3: Optimize the program **50 points**

1. Use any two optimization techniques to accelerate the matrix multiplication program and report the performance statistics.
Ensure that the code compiles and runs successfully.

2. Vary the block size and grid size and report power statistics.

3. Experiment with the below matrix dimensions:
4 * 4, 32 * 32, 64 * 64, 128 * 128, 256 * 256, 512 * 512, 1024 * 1024, 2048 * 2048

NOTE: You have to ensure that config files are copied to the application folder and run source setup_environment under gpgpu-sim/v3.x/ everytime you login to the machine.

Questions:
20 points

1. What is the optimal block size or grid size for large matrices(1024 and 2048) that results in high performance? Justify your answer.

2. How does performance vary with each optimization? Explain the reasons for speed up/slow down observed with each optimization.

3. How does power vary with each optimization?

4. For matrix of very small dimensions (2 * 2, 4 * 4, 8 * 8), it is generally observed that multithreaded application runs faster than CUDA application. State possible reasons for this behavior.