18-645: How to write fast code

Project #2 – Manycore Optimization

Due: Mar 17, 2017, 8PM PST, 11PM EST

The goal of this project is to use your understanding of parallel computing resources in a manycore microprocessor to optimize two fully functional applications. The applications are Matrix Multiple and k-means Clustering.

For a functional description of the applications, please refer to:

http://en.wikipedia.org/wiki/Matrix_multiplication

http://en.wikipedia.org/wiki/k-means

The code optimization techniques you may want to consider are explained in Module 3.1~4.

For project "`matrix_mul`" CUDA implementation, you can modify any functions in the file "`matrix_mul.cu`".

The CUDA code for **Matrix-to-Matrix Multiplication** that is provided for this project is implemented for power of 2 input matrix sizes only. Your task is to:

1. *optimize this code to achieve **150 GFLOPS***
2. *have a working version for any input sizes*

For project "`kmeans`" CUDA implementation, you can modify any functions in file "`cuda_kmeans.cu`".

The CUDA code for **k-means** that is provided for this project fails for test cases 3 and 4. Understand why it fails.

Hint: Focusing on "`compute_delta`" kernel function call, function and arguments

Your task is to:

1. *update this code to work for any test case requested*
2. *achieve 1.5x speedup above the implementation provided*

Grading criteria

- □ 30% - Correctness - Correctness of the results (program output)
- □ 30% - Performance –
    - MatrixMultiply:
      For CUDA version, achieving **at least 150 GFLOPS**
    - K-means:
      For CUDA version, achieving **at least 1.5x** speedup compared to initial **CUDA version**
- □ 30% - Write up – Clearly describing, for each performance optimization,
    - how the speed up works
    - what is the expected speed up
    - what is the observed speed up
    - an explanation of any difference between the expected and observed speed ups
- □ 10% - Code quality - Good coding practices and well commented code

Guidelines for the write up:

Minimum of one 8.5x11 page write-up for each optimization. The write up should include:

- Optimization goal:
    - Hardware resources being optimized toward?
      (cache? SIMD? multicore?)
    - What is the specification of the hardware you are optimizing for?
- Optimization process:
    - Data considerations
    - Parallelization considerations
- Optimization results:
    - Performance before optimization
    - Performance after optimization

Two teams with the fastest project in the class will be asked to do a 10min presentation each on what they tried.

We will look at the code of the slowest two implementations as a class. The class will discuss why their code is slow.