ECE5640 **High Performance Computing** **Spring 2016**

Homework 2: Due Wednesday, February 3

Homework is due midnight on Blackboard on Wednesday, February 3. If you scan your homework, make sure it is legible!

Important: Submit two files. The first should be a pdf named `lastname_firstname_hw#.pdf` containing answers to all questions, including tables, graphs and screenshots of the output. The second should be a zip file named `lastname_firstname_hw#.zip` containing only codes, `readme.txt` and executables named according to the question numbers. For example the code in question 1 should be named `1.a`.

Numbered problems are exercises from the book: Introduction to High Performance Scientific Computing by Victor Eijkhout. The exercises are embedded in the chapters.

Each problem is worth 15 points unless otherwise noted.

1. (40 pts)

   This question makes use of the Euclidean distance code from Homework 1.

   (a) (20 pts) Write a multithreaded version of your code to compute Euclidean distance using OpenMP. The vectors should be split among the available threads and each thread should compute the distance of partial pairs. Then, each thread should have its own partial sum and then add to a shared variable called `totdist`. Each thread should use an openmp `atomic` pragma when it updates `totdist`. Your code shoud work on any number of input threads. Turn in your code.

   (b) (10 pts) Use the openmp reduction sum to accomplish the same thing as the atomic pragma in your code from the previous part. Turn in your code.

   (c) (10 pts) Analyze the run time of your code on the discovery cluster. Compare sequential, parallel, and reduction sum code. Run on different numbers of threads. Use vector sizes 10^7, 10^8 and 10^9. Give advice to someone running this code regarding how they should parallelize it. How many threads is optimal? Which version runs the fastest? Explain your answer. Turn in the script you use to run your file and the output file.

2. 2.1

3. 2.2

4. 2.3

5. 2.4