Nicholas Skinner

CS 475

**Project 0 - Simple OpenMP Experiment**

**Platform:**

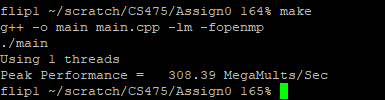
Tests for this project were conducted on the Oregon State University Flip Server.

**Performance:**

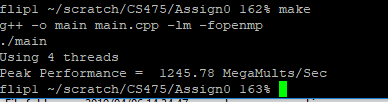
For this program, performance is calculated in Mega Multiples per second (MM/s). I used an array size of 1024 \* 1024, and had performed the computations 10 times to get an average performance. I used a makefile to compile and run the program, to compile it you should use the **make** command, to clean the directory, use **make clean**.

I ran the test with one thread and with four threads, and came to the results listed below.

1 Thread



4 Threads



**Speedup:**

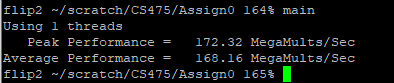
Calculating the speedup of our program is as simple as division, this division will take our 4 thread process and divide it by our 1 thread process to get a bit by bit comparison.

In our case, we will be taking 1245.78MMs and divide it by 308.39MMs

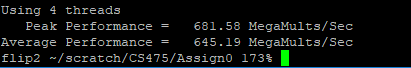
The result of the division is **4.03**, which seems to be a strange answer. I imagine this is due to variance of other users and usage of the machine.

I think the result should be just under 4 normally, due to overhead when tests are performed, i made a second attempt on the same machine to double check my answer. (I also found how to get the average as well as the peak, but I won't factor in the average, as the professor said not to.)

1 Thread



4 Threads



In this instance, we get a much more reasonable answer of **3.95**, it is in our expected range, and appears to be a more reasonable answer.

**Parallel Fraction:**

Our FP value is as follows

float Fp = (4./3.)\*( 1. - (1./S) );

Where S = our speedup from 1 to 4 threads

First Run FP:

Fp = (4./3.)\*( 1. - (1./4.03) );

**Fp = 1.0025**

Second Run FP:

Fp = (4./3.)\*( 1. - (1./3.95) );

**Fp = .99578**