Amy Stockinger (stockina@oregonstate.edu)
CS475 Project #0 – Simple OpenMP Experiment
Due: April 8, 2019

For this project, I ran a slightly modified version of the provided code on flip3 with uptime values around 3.00 and not using -O3. I moved the content of the given "main" function to a separate function and added a '(number of) threads' argument so it could be used to calculate for 1 and 4 threads in the same run. The main function then included speedup and parallel fraction calculations. This project seems to focus only on the parallelized loop, so I did not include filling the arrays with random numbers in my execution times.

My program had **array sizes of 100,000** with **20 tries**.

Results:

1 Thread	
Execution time	833.91 microseconds
Peak Performance	120.68 MegaMults/s
Average Performance	117.89 MegaMults/s
4 Threads	
Execution Time	209.71 microseconds
Peak Performance	477.22 MegaMults/s
Average Performance	460.13 MegaMults/s

Speedup = 833.91 / 209.71 = 3.98
Parallel Fraction = (4 / 3) * (1 – (1 / 3.98)) = 1.00

The peak performance and average performance numbers are close to each other, which is good.

If the speedup is less than 4.00, why do you think it is this way? It is almost exactly 4.00, and was in the 3.50-4.00 range for most runs, which is great—the program loop is 3.5-4x faster with 4 threads. The program is not so efficient on my personal computer, which the speedup is around 3.00, probably due to other background tasks and having less cores than flip.