Drake Seifert

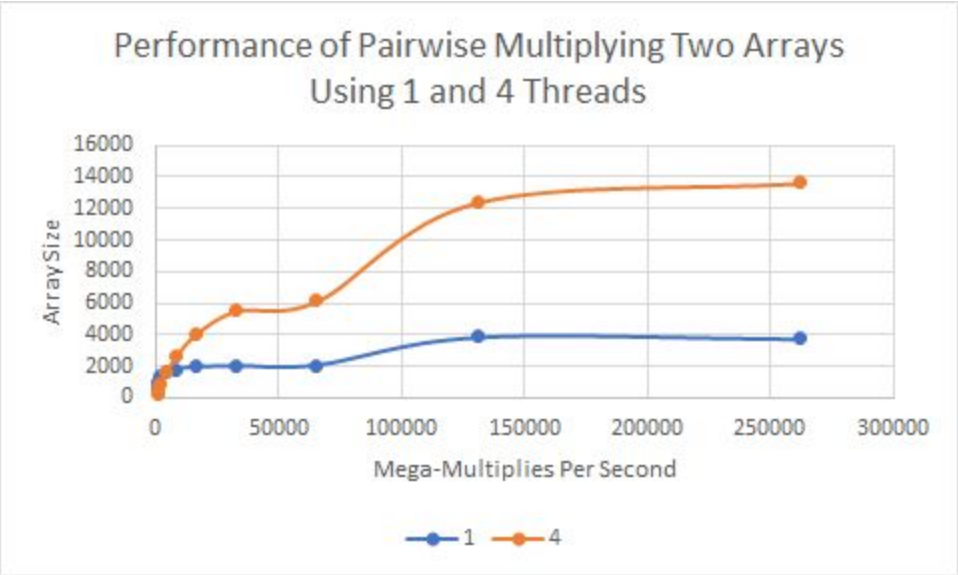
CS 475 - Project 0

Machine Tests Were Ran on

All tests were compiled and ran on the OSU FLIP servers.

Performance Results

NUMTRIES = 1000 for all tests
Using g++ compiler version 7.3.0
Uptime load average: All three values greater than 1 and less than 2



4-Thread-to-One Speedup

S = (Execution time with one thread) / (Execution time with four threads)

From Piazza, I learned that S can be calculated using performance using the following equation:
S = (Performance with four threads) / (Performance with one thread)
Using the last data points with the largest variance between performances,
S = 13624.73 / 3730.52 = 3.65

Reason for This Behavior

As the array size increases, eventually a single thread will begin to max out its performance. This can be seen when the array size reaches about 130,000 elements. This number is

drastically improved when using multiple threads however, because each thread's individual performance can be compounded together for overall better efficiency.

Interesting behavior can be seen for small array sizes, typically less than 4000 elements large. That is, the single thread in these cases outperforms the multithreaded counterpart. This may be because the overhead of spawning and managing multiple threads may began to exceed the performance gains of multithreading when the array size is small enough to be quickly calculated by a single thread.

Parallel Fraction

Fp = (4 / 3) * (1 - (1 / S))
Fp = (4 / 3) * (1 - (1 / 3.65))
Fp = 0.726