Xiaoying Li
lixiaoyi@oregonstate.edu
Project #0 Simple OpenMP Experiment

1. Tell what machine you ran this on.

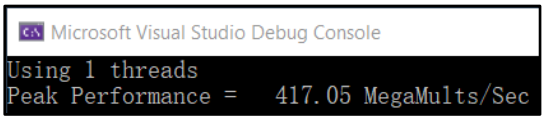
I ran the program on my own machine – Intel Core i5.

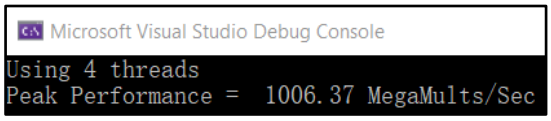
2. What performance results did you get?

The performance was calculated with "Mega-Multiplies per second" using the main program provided by instruction.

Execution time results for 1 thread: Peak Performance = 417.05 MegaMults/Sec.

Execution time results for 4 threads: Peak Performance = 1006.37 MegaMults/Sec.





3. What was your 4-thread-to-one-thread speedup?

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

= (Performance with four threads) / (Performance with one thread)

= 1006.37 / 417.05 ≈ 2.42

4. If the 4-thread-to-one-thread speedup is less than 4.0, why do you think it is this way?

In theory, the performance result for 4 threads is expected to be 4 times of the performance result for 1 thread, which means the 4-thread-to-one-thread speedup is expected to be 4.0. But in practice this might not happen, like the 4-thread-to-one-thread speedup I got is 2.42, which is less than 4.0.

In my opinion, because in practice not all physical threads are available entirely to run the concurrent program. There are always other tasks and programs that are running on the system and sharing these resources. So, every thread is likely to be interrupted by other programs and the execution time will increase. Another reason is multi-threads is taking advantage of idle recourses, so the overhead for creating threads and thread-intercommunications can also make the speedup non-ideal.

5. What was your Parallel Fraction, Fp?

Parallel Fraction float Fp = (4. / 3.) * (1. – (1. / S)) = (4. / 3.) * (1. – (1. / 2.42)) ≈ 0.78