Project 0 - Simple OpenMP Experiment

Metta Saketh Srinivasa Rao

mettas@oregonstate.edu

1. Tell what machine you ran this on

   I was connected to the flip server and ran the program on that.

2. What performance results did you get?
   For 1 threads, Peak Performance = 487.75 MegaMults/Sec For 4
   threads, Peak Performance = 1839.19 MegaMults/Sec
3. What was your 1-thread-to-4-thread speedup?
   To find the Speedup = (Peak performance for 4 threads) / (Peak performance for 1 thread)
   1839.19 / 487.75 = 3.77
4. Your 1-thread-to-4-thread speedup should be less than 4.0. Why do you think it is this way?

   As explained in the class on Monday, April 1st, each time a thread or process is created, it has an overhead. Let's assume that is 5 secs. Then when we create 4 or even 8 threads, each thread will have the same time requirement to be created, so when the process executes the program, it starts after the 5 seconds in both the cases. This way if we assume the program takes 30 seconds to execute with 1 thread and it will take 7.5 seconds with 4 threads, but we need to add the 5 seconds to the above time as well. So, 1-thread is 35 seconds and 4-thread is 12.5 seconds. In this way the performance will always be less than 4 times for 1-thread-to-4-thread.Theoretically, as the length of the array that is being executed increases the time can get infinitely close to 4 but never exactly 4 due to this issue.

5. What was your Parallel Fraction, Fp? (Hint: it should be less than 1.0, but not much less.) To find

   the parallel function: float Fp = (4./3.)*(1. - (1./S));

   Fp = 0.979664