Student Instructions

The Sieve of Eratosthenes is a classic $n\ log(n)$ algorithm for determining a large list of prime numbers, and if your goal is to calculate a complete and comprehensive list of primes up to some value, is likely the fastest algorithm that you can use.

It works as follows. You start with a list of all numbers up to N. (For our computer algorithm we will in practice count to N-1 to make our array syntax easier). For each of these numbers, construct a list of whether the number is or is not prime. In the beginning, assume all numbers are prime until proven otherwise. This might be most easily done with an array of ints or booleans, so that `list[5]==0` would imply that 5 is prime, whereas `list[4]==1` would imply that 4 is not.

You have been provided a starter code that implements the Sieve of Eratosthenes algorithm in C, with additional includes and profiling instructions for OpenMP. Your goal is to discuss, plan, and implement a parallelization of the problem in OpenMP, and test the performance of your implementation.

1. Consider the starter code. Compile and run it as you would for any OpenMP program. `gcc -o sieve.starter sieve.starter.c -lm -fopenmp` For increasing list lengths, note the running time. Does the algorithm scale as expected?

2. For the starter code, note the amount of time required to build a list. Before attempting parallelization, make a hypothesis as to how long the list might need to be before you begin to see the benefit of parallelism.

3. Looking at the nested loop, determine whether there are any loop carried dependencies in either the inner or outer loop. What is the implication of a loop carried dependency in parallelization?

4. Implement a plan to parallelize using OpenMP. Focus on the outermost loop without a loop carried dependency. Parallelize your code.

5. Using your parallel code, compare parallel efficiency as you increase the problem size and number of threads. Display your results in a table, and discuss the effectiveness of parallelization for this problem at different problem sizes.