PP ASSIGNMENT 3 (411764)

Count the number of primes b/w 1 and N using OpenMP

Sir, I’ve uploaded the code over github (https://github.com/ravijainpro/Parallel-Processing).

=====================Description=====================

A C++ code which counts the number of primes between 1 and N, using OpenMP to carry out the calculation in parallel. For each integer I, it checks whether any smaller J evenly divides it. The total amount of work for a given N is thus roughly proportional to 1/2*N^2.

In the BASH shell, the program could be run with 2 threads using the commands:

```
        1>      export OMP_NUM_THREADS=2
        2>      g++ -c -Wall -fopenmp prime_openmp.cpp
        3>      g++ -fopenmp prime_openmp.o
        4>      ./a.out
```

===========================CODE===========================

```
# include <cstdlib>
# include <iostream>
# include <iomanip>
# include <omp.h>

using namespace std;

int main ( int argc, char *argv[] );
void prime_number_sweep ( int n_lo, int n_hi, int n_factor );
int prime_number ( int n );

int main ( int argc, char *argv[] ) {
        int n_factor;
        int n_hi;
        int n_lo;

        cout << "\n";
        cout << "PRIME_OPENMP\n";
        cout << "  C++/OpenMP version\n";

        cout << "\n";
        cout << "  Number of processors available = " << omp_get_num_procs ( ) << "\n";
        cout << "  Number of threads =              " << omp_get_max_threads ( ) << "\n";
```

```
	n_lo = 1;
	n_hi = 131072;
	n_factor = 2;

	prime_number_sweep ( n_lo, n_hi, n_factor );

	n_lo = 5;
	n_hi = 500000;
	n_factor = 10;

	prime_number_sweep ( n_lo, n_hi, n_factor );

	cout << "\n";
	cout << "PRIME_OPENMP\n";
	cout << "  Normal end of execution.\n";

	return 0;
}

void prime_number_sweep ( int n_lo, int n_hi, int n_factor ) {
  int n;
  int primes;
  double wtime;

  cout << "\n";
  cout << "TEST01\n";
  cout << "  Call PRIME_NUMBER to count the primes from 1 to N.\n";
  cout << "\n";
  cout << "         N        Pi          Time\n";
  cout << "\n";

  n = n_lo;

  while ( n <= n_hi )
  {
    wtime = omp_get_wtime ( );

    primes = prime_number ( n );

    wtime = omp_get_wtime ( ) - wtime;

    cout << "  " << setw(8) << n
         << "  " << setw(8) << primes
         << "  " << setw(14) << wtime << "\n";

    n = n * n_factor;
  }

  return;
}

int prime_number ( int n ) {
  int i;
  int j;
  int prime;
  int total = 0;

  for ( i = 2; i <= n; i++ )
  {
    prime = 1;

    for ( j = 2; j < i; j++ )
    {
      if ( i % j == 0 )
      {
        prime = 0;
        break;
      }
    }
    total = total + prime;
  }

  return total;
}
```

```
==========================OUTPUT==========================

(base) ravijain@ravijain-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/openMP/Prime$ ./a.out

PRIME_OPENMP
  C++/OpenMP version

  Number of processors available = 8
  Number of threads =              2

TEST01
  Call PRIME_NUMBER to count the primes from 1 to N.

        N        Pi          Time

        1        0     0.000186855
        2        1       7.483e-06
        4        2     5.44199e-06
        8        4     5.87601e-06
       16        6     6.05201e-06
       32       11       5.861e-06
       64       18       1.007e-05
      128       31      2.4259e-05
      256       54      8.1105e-05
      512       97     0.000261008
     1024      172     0.000903265
     2048      309      0.00331418
     4096      564       0.0132808
```

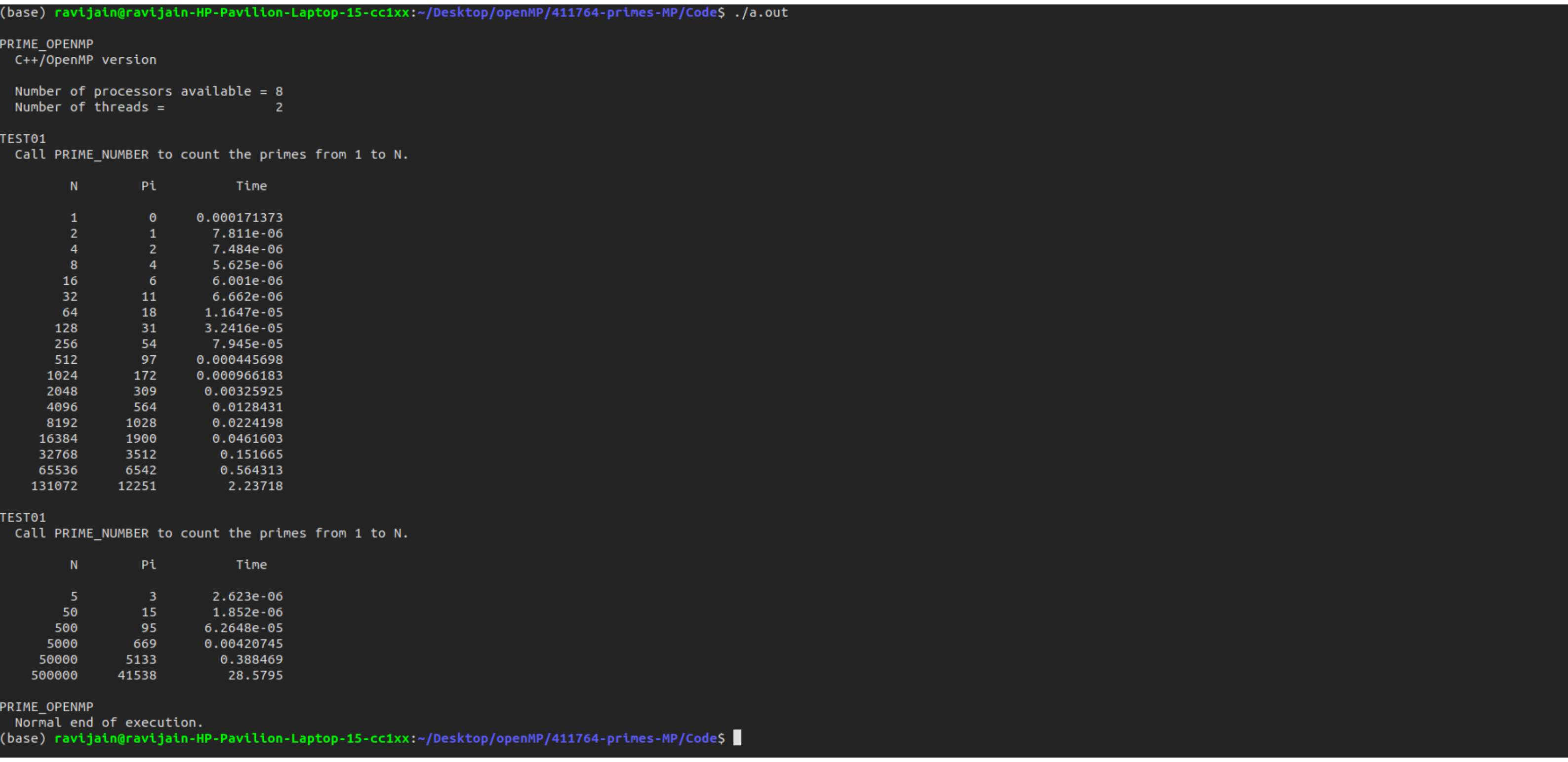
```
      8192      1028       0.0321918
     16384      1900       0.0611634
     32768      3512        0.172317
     65536      6542        0.599941
    131072     12251         2.05724

TEST01
  Call PRIME_NUMBER to count the primes from 1 to N.

         N        Pi          Time

         5         3     2.31601e-06
        50        15       1.668e-06
       500        95      5.8827e-05
      5000       669       0.0040472
     50000      5133        0.320499
    500000     41538         27.2001

PRIME_OPENMP
  Normal end of execution.
```



```
(base) ravijain@ravijain-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/openMP/411764-primes-MP/Code$ ./a.out

PRIME_OPENMP
  C++/OpenMP version

  Number of processors available = 8
  Number of threads =              2

TEST01
  Call PRIME_NUMBER to count the primes from 1 to N.

         N        Pi          Time

         1         0     0.000171373
         2         1       7.811e-06
         4         2       7.484e-06
         8         4       5.625e-06
        16         6       6.001e-06
        32        11       6.662e-06
        64        18      1.1647e-05
       128        31      3.2416e-05
       256        54       7.945e-05
       512        97     0.000445698
      1024       172     0.000966183
      2048       309      0.00325925
      4096       564       0.0128431
      8192      1028       0.0224198
     16384      1900       0.0461603
     32768      3512        0.151665
     65536      6542        0.564313
    131072     12251         2.23718

TEST01
  Call PRIME_NUMBER to count the primes from 1 to N.

         N        Pi          Time

         5         3       2.623e-06
        50        15       1.852e-06
       500        95      6.2648e-05
      5000       669      0.00420745
     50000      5133        0.388469
    500000     41538         28.5795

PRIME_OPENMP
  Normal end of execution.
(base) ravijain@ravijain-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/openMP/411764-primes-MP/Code$
```