Project Euler Problem 10: Summation of Primes

Maria Cristina Menendez Ortiz

July 30, 2014

1 Design

1.1 The problem

Find the sum of all the primes below n.

$$S = 2 + 3 + 5 + 7 + ... + n \tag{1}$$

1.2 STAPL

1.2.1 Pseudocode

Input: n.
Output: The sum of all the primes bellow n.
Fill the array with sequential values, from 1 to n.
Replace all array values that are not prime numbers with the value 0.
Compute the sum of all the values in the array.

1.2.2 Components

- stapl::array<int> c(n);
- stapl::array_view<stapl::array<int>> v(c);
- stapl::iota(v, 1);
  Fills the array with sequentially increasing values, starting with 1.
- stapl::replace_if(v, condition, 0);
  Replace all elements satisfying a condition. It is used to replace all non-primes with 0.

- stapl::accumulate(v, 0);
  Compute the sum of the values in the array.

2 Experiments

The experiments were compiled using GNU g++ 4.7.2 with -O3 optimization level and performed on a Cray XE6m with 24 nodes, each with 16 or 32 cores and 2GB of memory per core. The system has a total of 576 cores. Scalability for p processors is defined as the ratio of the execution time for 1 processor and p processors. In other words, $\frac{T_1}{T_p}$. The expected behavior is a linear trend in the graph of the scalability.

2.1 Graphs

Table 1 shows the time, in seconds, that it takes to execute the program.

	Time (s)		
Processors	10000	100000	1000000
1	0.23043	28.7658	3196.25
2	0.240441	25.6255	2707.72
4	0.151582	16.1191	1706.55
8	0.0898	9.42793	993.644
16	0.04909	5.05372	531.798
32	0.025705	2.50125	263.115
64	0.014446	1.2427	130.809
128	0.00873	0.621217	65.2315
256	0.007398	0.31261	32.601
512	0.005095	0.157881	16.2636

Table 1: Relation between the input and the time.

Figure 1 is comparing the time that it takes the processors to finish the process with respect to the number of input.

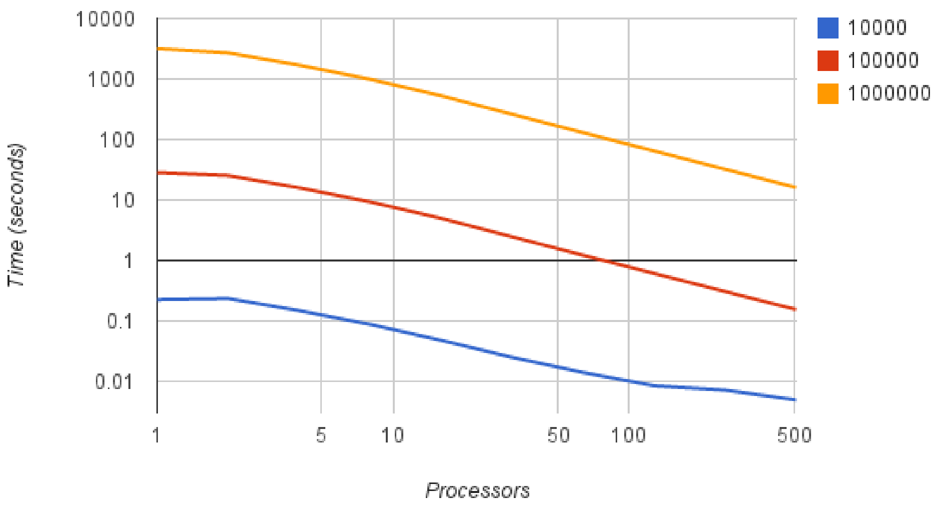
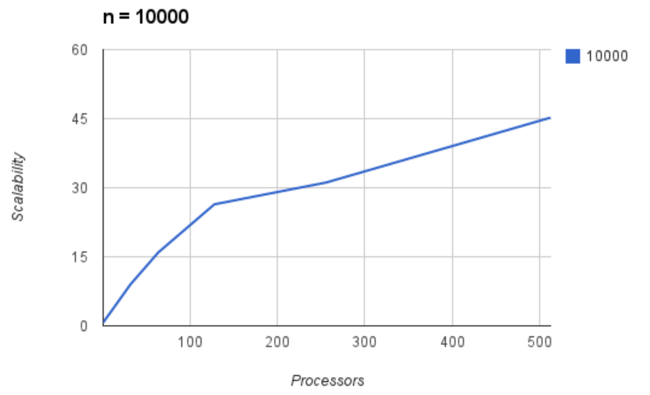


Figure 1: Relation between the input and the time.

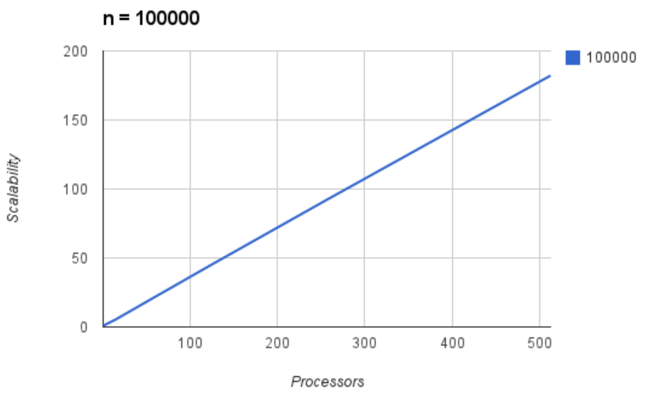
Table 2 shows the scalability for various input size.

Scalability

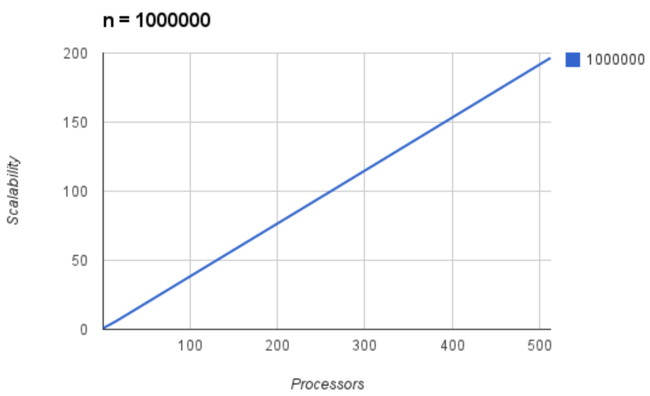
Processors	10000	100000	1000000
1	1	1	1
2	0.9583640061	1.122545902	1.180421166
4	1.520167302	1.784578543	1.872930767
8	2.566035635	3.051125751	3.216695315
16	4.694031371	5.692005097	6.010270817
32	8.964403812	11.50056972	12.14773008
64	15.95112834	23.14782329	24.43448081
128	26.395189	46.30555828	48.99856664
256	31.14760746	92.0181696	98.04147112
512	45.22669284	182.1992513	196.527829



n = 10000
60
45
30
15
0
100
200
300
400
500
Scalability
Processors
10000



n = 100000
200
150
100
50
0
100
200
300
400
500
Scalability
Processors
100000



n = 1000000
200
150
100
50
0
100
200
300
400
500
Scalability
Processors
1000000

2.2 Analysis

The graph above illustrates the difference of scalability when we have three different inputs. When the input is equal to *10,000* the scalability increases 45 times, but when the input is 10 and 100 times bigger the scalability increases more than 100 times. In other words, the efficiency raises when the number of inputs and processors increase.
With the last two inputs we observe the expected behavior.