CPA Lab-Report
Lab 2 Prime Numbers

Simon Birrer, Dominic Schürmann

November 3, 2015

Date Performed:	October 27, 2015
Students:	Simon Birrer
	Dominic Schümann
Instructor:	Francisco Javier Piris Ruano

Contents

1 Biggest prime storable in 8 bytes

The Source for the solution is in the file *primo_grande.c*. In the figure 1 you will see the changes applied to function primo of the original code.

```
1    int numberOfThreads;
2    int offset;
3    #ifdef _OPENMP
4    #pragma omp parallel
5    numberOfThreads = omp_get_num_threads();
6    offset = 2 * numberOfThreads;
7    #else
8    numberOfThreads = 1;
9    offset = 2;
10   #endif
11
12    #pragma omp parallel private(i)
13    {
14      #ifdef _OPENMP
15      int threadIndex =  omp_get_thread_num();
16      int startIndex = (2 * threadIndex) + 3;
17      #else
18      int startIndex = 3;
19      #endif
20      for (i = startIndex; p && i <= s; i += offset)
21        if (n % i == 0) p = 0;
22    }
```

Figure 1: code changes primo grande

1.1 Compiling without OpenMP

To use the program in both ways, either with or without OpenMP , we used the preproccessor directives. Now the compiler decides upon the arguments if the code will use OpenMP or not for the passages where OpenMP function calls will happen.

This can be seen in figure 1 in lines 3-10 and 14-19.

1.2 Time measurement of parallelized version

In figure 2 are the measured times of executing the program with different numbers of threads using kahan.

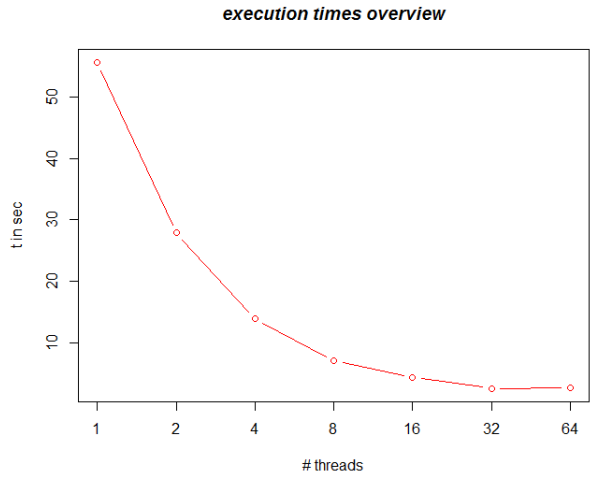

Figure 2: execution times for exercise 1.2

Since a node of kahan has 32 cores, the execution with 32 threads was the fastest. In addition the performance decreases if the number of threads will be increased. This is shown in figure 2 and the overhead is even more visible in figure 3 .
TODO Describe why 64 slower

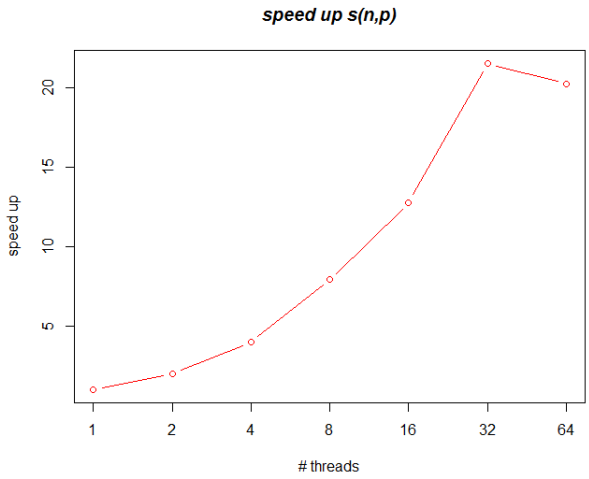

Figure 3: speed up for exercise 1.2

2 Count primes in a range

The Source for the solution of exercise 2.1 is in the file *primo_numeros_1.c* and for exercise 2.2 in file *primo_numeros_2.c*

2.1 Exercise 1 with reduction clause

2.1.1 scheduling distribution

- static 0, without chunk
- static 1, with chunk
- dynamic

2.2 Exercise 2 printing workload

3 appendix

3.1 measurements of exercise 1

number or threads	1	2	4	8	16	32	64
time in seconds	55.571	27.850	13.931	7.019	4.349	2.580	2.741

Table 1: execution time

TODO include formula for speed up and speed up table

List of Figures

List of Tables

References