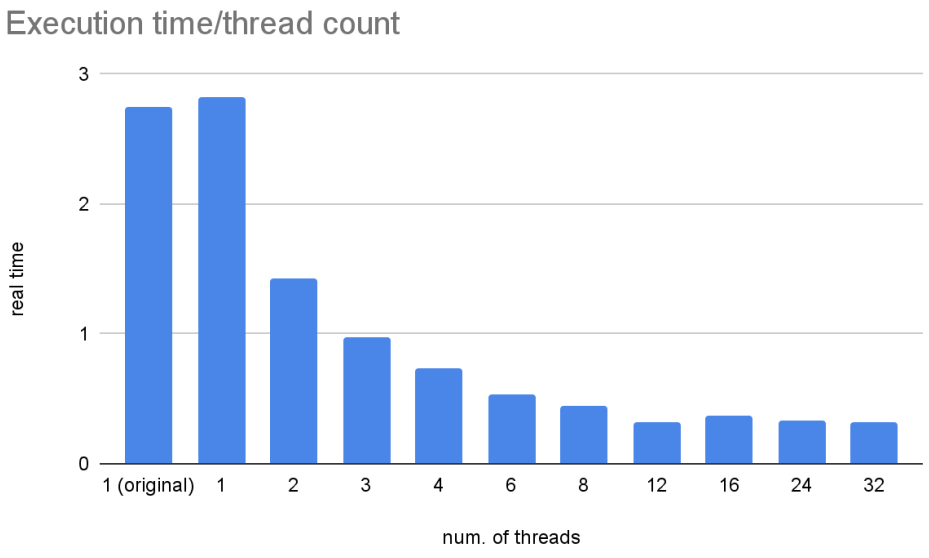
A3 Report

Gabrielle Pili
30064830

Question 2 - written for `calcpi.cpp`

PART A - Record your timings in a table and create a corresponding bar graph.
Results of running *r*=50000 and `time` command

Threads	Timing (s)
1 (original)	2.742
1	2.826
2	1.425
3	0.969
4	0.740
6	0.530
8	0.446
12	0.326
16	0.373
24	0.337
32	0.320



PART B - When you run your implementation with N threads, you should see N-times speed up compared to the original single threaded program. Do you observe this in your timings for all values of N?

Yes, there is a N-times speed up for up to 3 threads.

- For 1 thread it is running at 2.826 seconds
- For 2 threads, it runs very close to ½ of the first thread at 1.425 seconds
    - close to 2.826 secs / 2 = **1.413 seconds**
- For 3 threads, it runs very close to ⅓ of the first thread at 0.969 seconds
    - Close to 2.826 secs / 3 = **0.942 seconds**
- For 8 threads, it runs about ⅛ of the first thread at 0.446 seconds
    - close to 2.826 secs / 8 = **0.35325 seconds**

When we get to 8 threads, we don't really see exactly N-times speed up. Given this observation, we can say that there are 6-cores on the CPU.

PART C - Why do you stop seeing the speed up after some value of N?

The speed up changes after some value of N because there are a limited number of cores on a CPU. We will only see speed ups when the **number of threads <= number of cores**.

Each thread uses 1 core in a single processor. In this case, we can see that the number of cores would be about 12 on the linuxlab computers since it slows down when there are 16 threads. When the *number of threads* ***becomes greater*** than *the number of cores* in a CPU, this is where we see it slow down.

Question 4 - written for `detectPrimes.cpp`

Are the timings what you expected them to be? If not, explain why they differ.

medium.txt			
# threads	**Observed timing**	**Observed speedup compared to original**	**Expected speedup**
Original program	17.6209	1.0	1.0
1	17.6200	1.0	1.0
2	9.2191	1.91	2.0
3	6.2713	2.81	3.0
4	4.6556	3.78	4.0
8	3.2562	5.41	8.0
16	2.7764	6.35	16.0

The timings are expected up until 8 threads. Threads 1-4 have about **N** speed ups.

- **Example #1:** For N=4 and T=17.62 secs, the speed up is 3.78 which is almost N times speed up.

From 8-16 threads, I was expecting to also see a speed up of close to *N* times so the results were disappointing. As mentioned in **Q2 PART C**, the *number of threads becomes greater than the number of cores* in the CPU.

- **i.e.** running `lscpu` in my ssh session gives me information that there are only 6 cores. And since 8-16 are all greater than the number of available cores, we stop seeing *N* times speed up.

```
Model name:            Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
  CPU family:          6
  Model:               158
  Thread(s) per core:  2
  Core(s) per socket:  6
```