CPSC 457 — Principles of Operating Systems
ASSIGNMENT 2

Name: Haohu Shen
UCID: 30063099

3. **Written question (5 marks)**

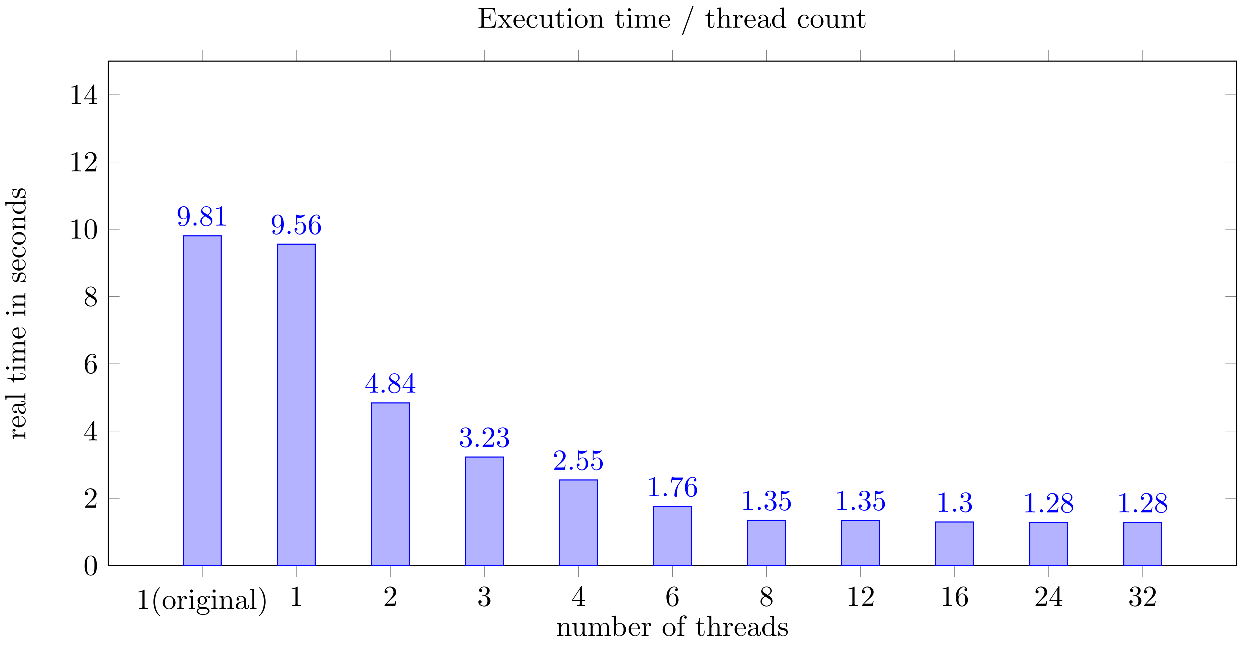
   Time your multithreaded solution from Q1 on the numbers in *test3.txt* file using the time command. Record the real-time for 1 thread, then 2, 3, 4, 6, 8, 12, 16, 24 and 32 threads. Also record the timings for the sample solution I provided.

   Solution.

   (a) The table of the timings is shown below:

Threads	Timing(s)
1 (original)	9.805
1	9.557
2	4.837
3	3.227
4	2.546
6	1.756
8	1.347
12	1.346
16	1.296
24	1.277
32	1.278

   (b) The bar graph of the timings is shown below:



(c) Answer the following questions:

a) With N threads you should see an N-times speed up compared to the original single threaded program. Do you observe this in your timings for all N?

When N changes from 1 to 2, we can clearly see it is approximately 1-time speed up compared to the original single threaded program, also it is approximately k-times speed up compared to the original single threaded program when $k = 2, 3, 4, 6, 8$. The trend of speeding up stops and it becomes stable when $k \geq 8$.

b) Why do you stop seeing the speed up after some value of N?

One reason is that since the number of concurrent threads is increasing, the creation and the termination of threads are expensive and time-consuming. The second reason is that since the server for grading the assignment only have 8 CPUs and only 1 thread per core, thus when we assign more than 8 threads, only 8 cpu core threads are being used, thus most of threads (which are created on our code) that we assigned have to wait until a CPU core is free to run them, which is also time-consuming.