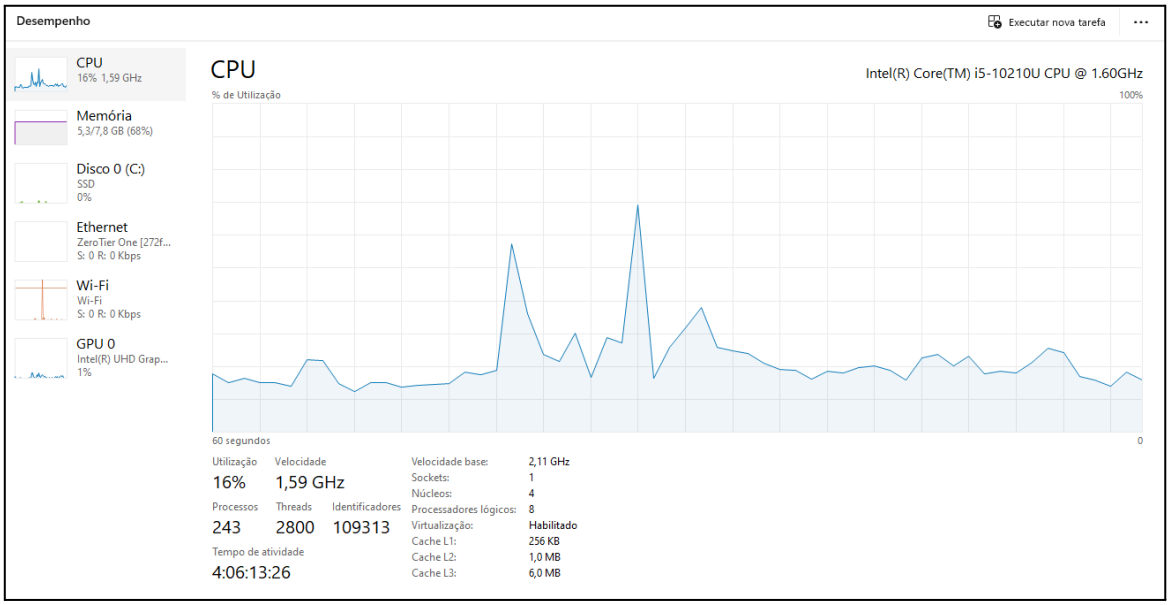
Aluno: Nicholas de Araujo Figueira DRE: 121088218

Relatório - Exercício 2 (Laboratório 4)



Para esse relatório, executou-se o programa de verificação da quantidade de número primos até um determinado valor. Avaliou-se para os casos 1000 (10^3), 1000000 (10^6) e 10000000 (10^7), considerando 1, 2, 4 e 8 threads. Deve-se atentar que o computador utilizado possui 4 núcleos.

Em seguida, conforme instruído, anotou-se o tempo de 5 iterações para cada caso e calculou-se a média dos tempos, a aceleração e eficiência. As equações de aceleração e eficiência são:

$$A_{(n,t)} = \frac{T_s(n)}{T_p(n,t)}; \; E_{(n,t)} = \frac{A(n,t)}{t}$$

Onde:

A = aceleração;

E = eficiência;

n = dimensão das matrizes;

t = quantidade de threads utilizadas;

Ts = tempo sequencial (nesse caso apenas uma thread);

Tp = tempo de processamento.

Forma de Utilização

Para compilar:

gcc -o atividade4 atividade4.c -lm -lpthread

Para rodar:

./atividade4 <N> <numThreads>

Onde N é o número até onde devem ser encontrados primos e numThreads o número de threads que devem ser utilizadas.

N = 1000 (10^3)

Sequencial (1 thread)						
	1°	**2°**	**3°**	**4°**	**5°**	**Média**
Tempo (s)	0.000225	0.000936	0.000450	0.000340	0.000309	0.000452

Aceleração: 1.000000
Eficiência: 1.000000

2 threads						
	1°	**2°**	**3°**	**4°**	**5°**	**Média**
Tempo (s)	0.000261	0.000258	0.000524	0.000826	0.000269	0.000428

Aceleração: 1.056075
Eficiência: 0.528037

4 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.000697	0.000480	0.000483	0.000716	0.000673	0.000610

Aceleração: 0.740984
Eficiência: 0.185246

8 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.000890	0.000585	0.000710	0.001221	0.001718	0.001025

Aceleração: 0.440976
Eficiência: 0.055122

N = 1000000 (10^6)

Sequencial (1 thread)						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.293747	0.293335	0.293747	0.288474	0.292432	0.292347

Aceleração: 1.000000
Eficiência: 1.000000

2 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.170001	0.175372	0.159906	0.168041	0.164570	0.167578

Aceleração: 1.744543
Eficiência: 0.872271

4 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.127649	0.124899	0.114800	0.112856	0.125395	0.121120

Aceleração: 2.413697
Eficiência: 0.603424

8 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	0.119221	0.122074	0.121929	0.114011	0.115087	0.118464

Aceleração: 2.467813
Eficiência: 0.308477

N = 10000000 (10^7)

Sequencial (1 thread)						
	1°	2°	3°	4°	5°	Média
Tempo (s)	7.350309	8.157334	7.410795	8.177984	7.547038	7.728692

Aceleração: 1.000000
Eficiência: 1.000000

2 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	4.002394	3.892042	3.955823	4.309828	4.122416	4.056501

Aceleração: 1.905261
Eficiência: 0.952630

4 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	2.257828	2.257114	2.421067	2.253308	2.183321	2.274528

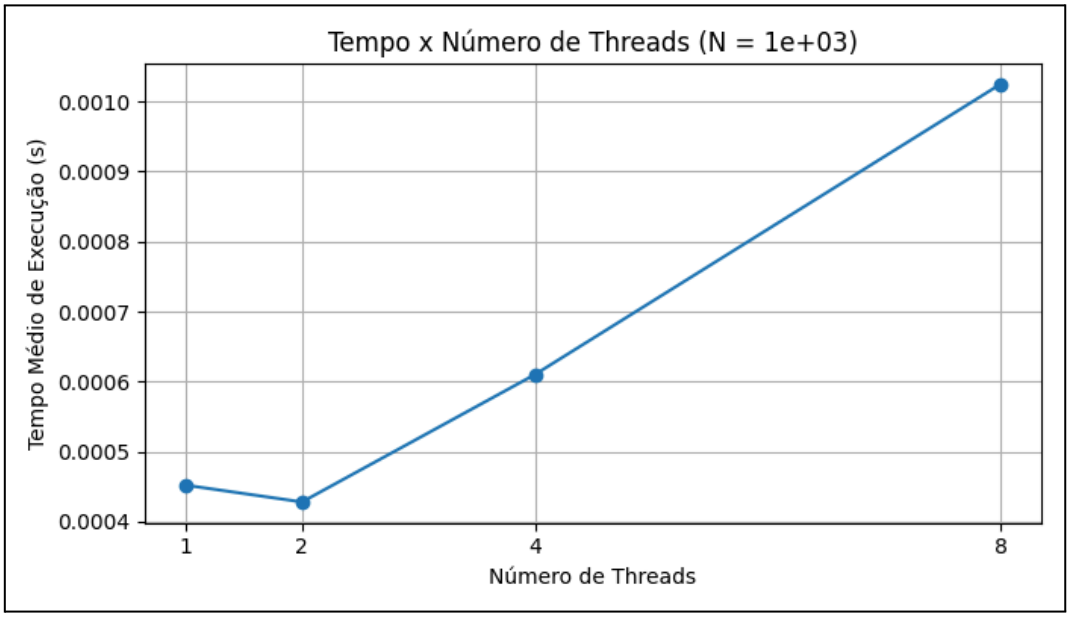
Aceleração: 3.397932
Eficiência: 0.849483

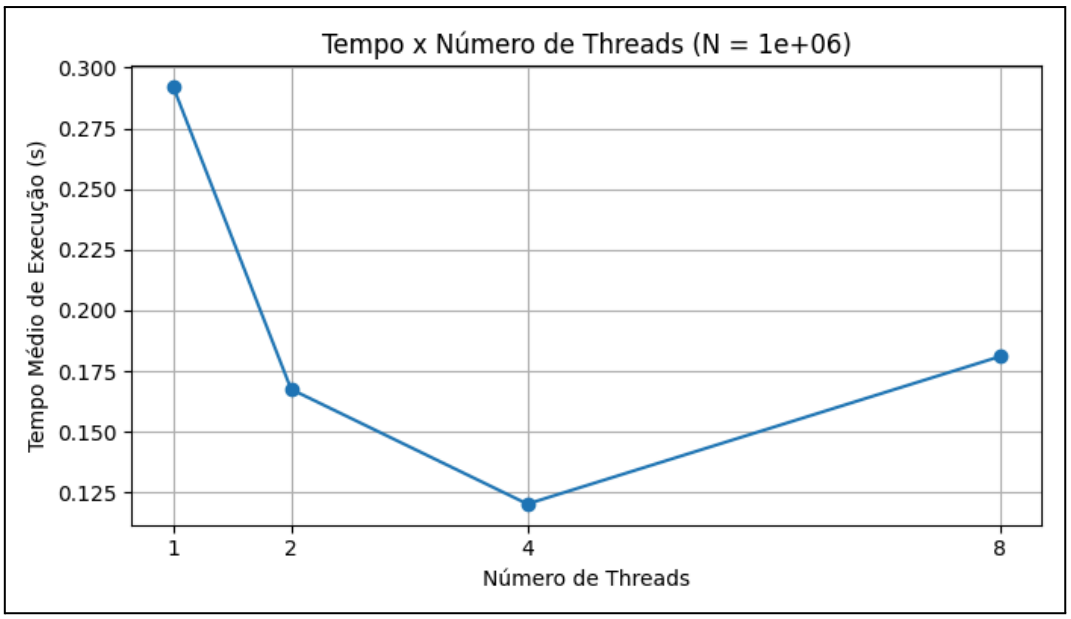
8 threads						
	1°	2°	3°	4°	5°	Média
Tempo (s)	1.633259	1.667859	1.679378	1.804603	1.680077	1.693035

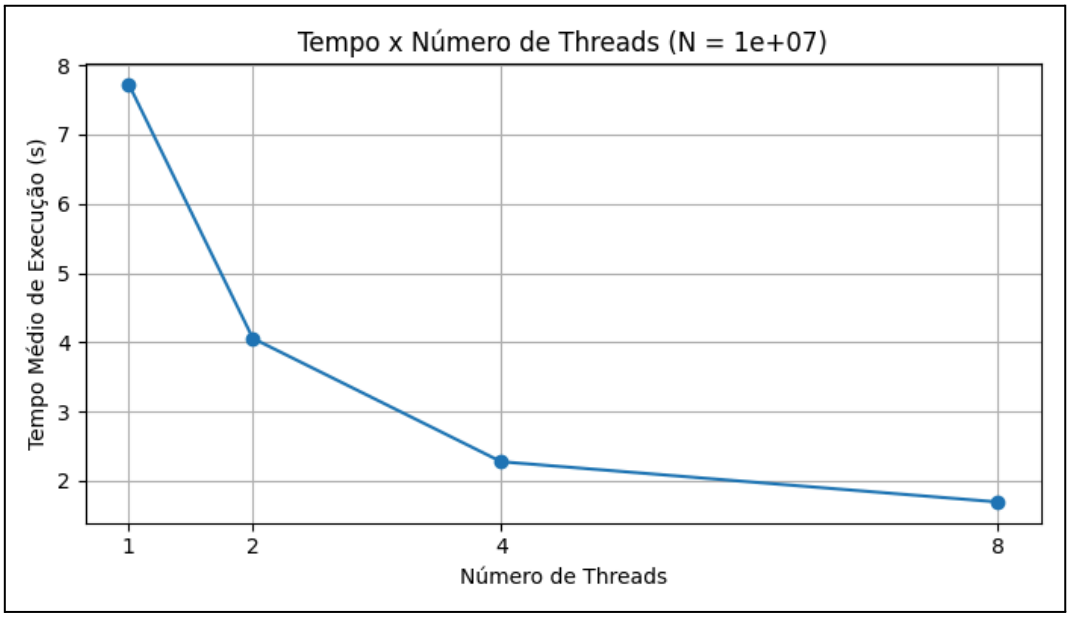
Aceleração: 4.564992
Eficiência: 0.570624

GRÁFICOS

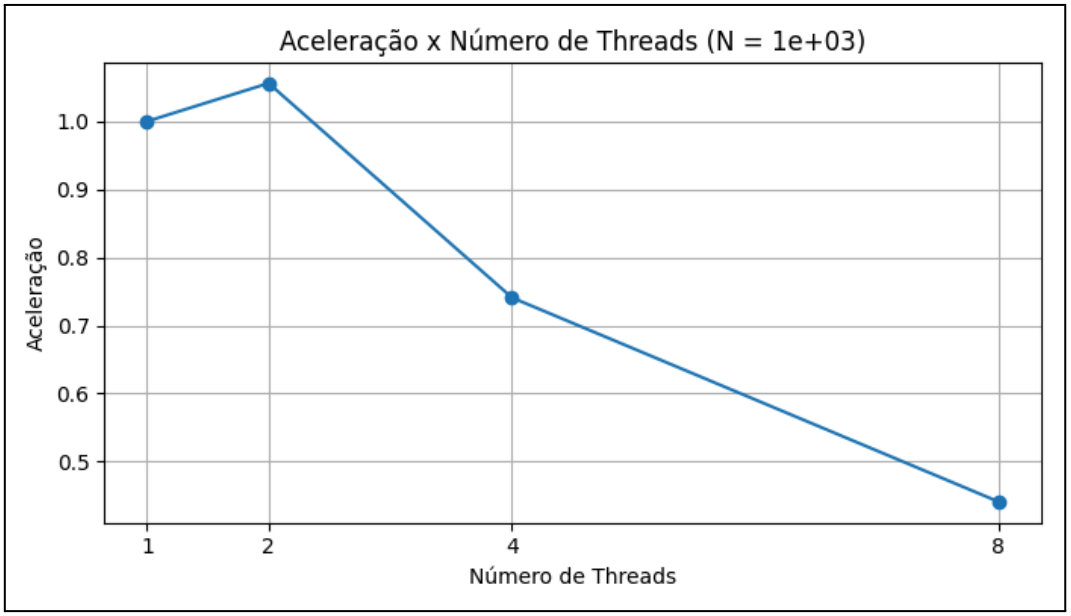
Tempo de execução x Número de Threads

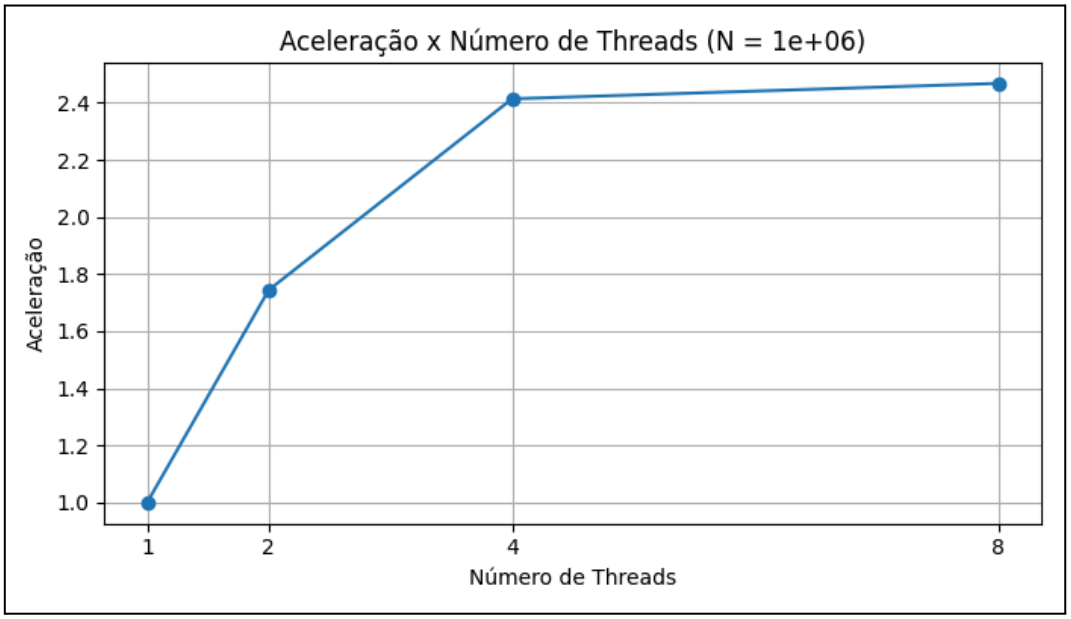


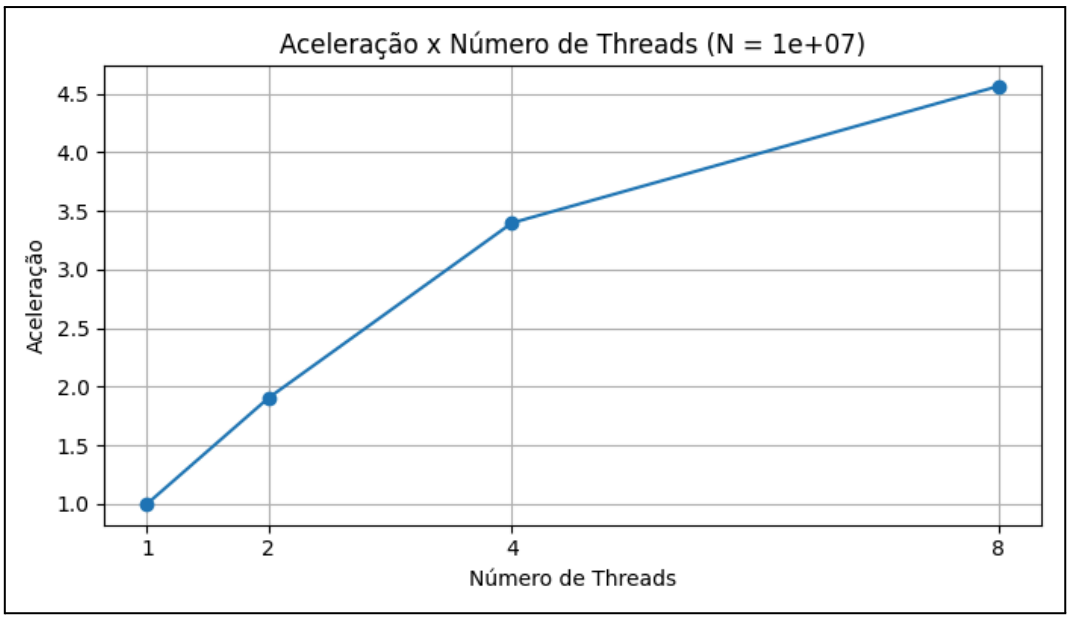

Tempo x Número de Threads (N = 1e+06)
Tempo Médio de Execução (s)
0.300
0.275
0.250
0.225
0.200
0.175
0.150
0.125
1
2
4
8
Número de Threads


Tempo x Número de Threads (N = 1e+07)
Tempo Médio de Execução (s)
8
7
6
5
4
3
2
1
2
4
8
Número de Threads

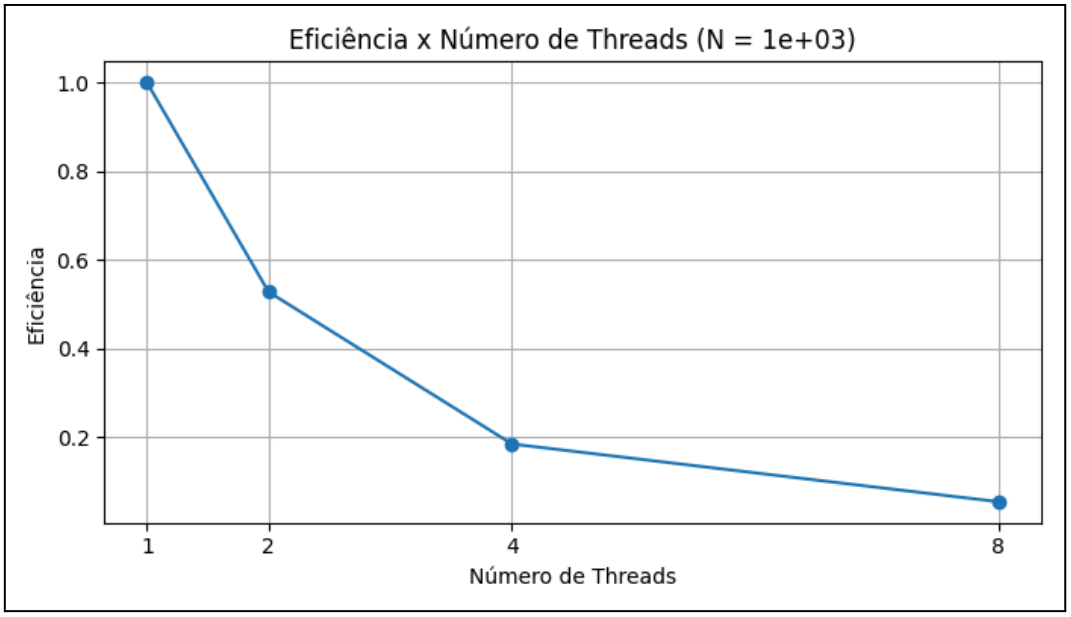
Aceleração x Número de Threads

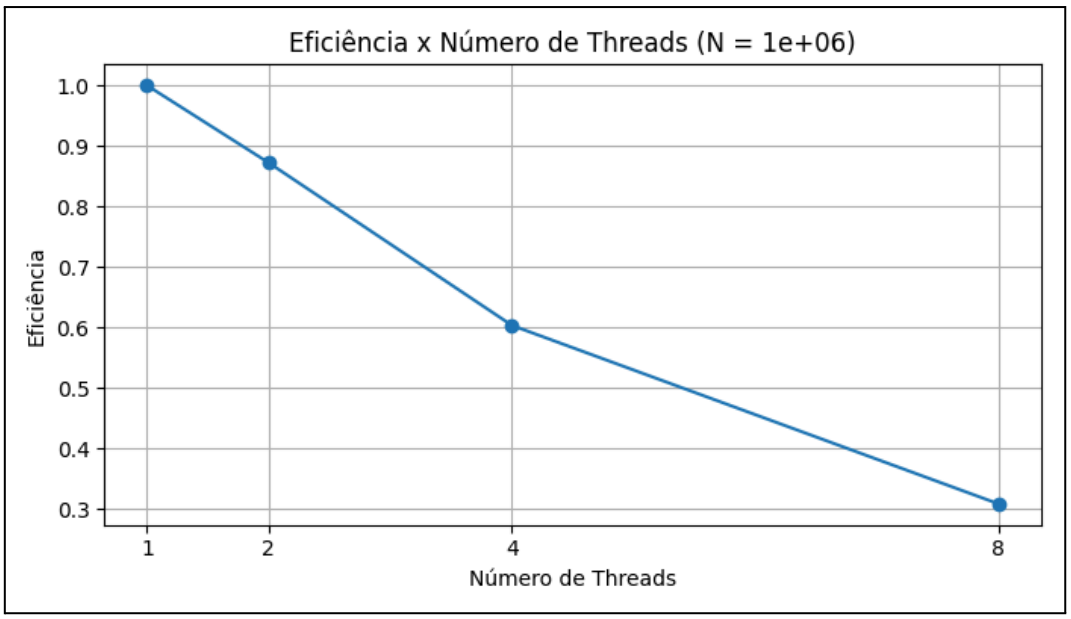


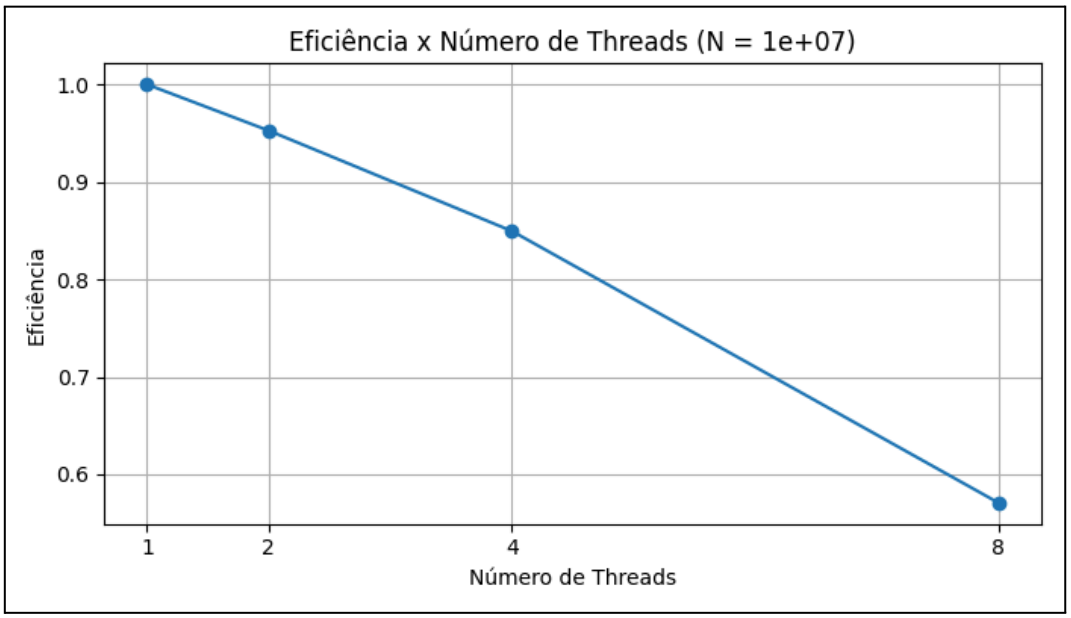




Eficiência x Número de Threads







Conclusões

- No caso de 10^3, a aceleração é baixa ou negativa com mais threads, o que indica que não há ganho de desempenho devido ao overhead criado e a sincronização necessária. Porém, isso não acontece para N maior;

- O aumento de threads não necessariamente melhora muito o desempenho, o que mostra que o número mais apropriado é o número de threads ser igual ao número de núcleos. Como o computador só tem 4 núcleos, o uso de 8 threads não foi muito útil;
- Quanto maior o N, melhor o ganho de desempenho, o que mostra que a concorrência brilha em aplicações e problemas pesados, vide a queda de 7,7 segundos para 1,8 no caso de N=10^7.