Parallel matrix norms using pthreads

Lee Kai Yang (23205838)
kai.y.lee@ucdconnect.ie

November 12, 2023

1 Introduction

The aim of this experiemnt is to discover the difference between the serial and parallel algorithm for computing matrix norms. The parallel algorithm was imlemented using **pthreads** and the benchmark was ran on a machine with 8 x Intel(R) Xeon(R) E5-2620 v3 @ 2.40GHz processors. The program uses the number of processors as the number of threads hence the results for parallel algorithm shown in following sections uses 8 threads.

All the matrix multiplication algorithms used in this experiemnt are written manually without using any third-party dependencies both for the serial algorithm and the parallel algorithm.

2 Dependence of program execution time on matrix size

Since the matrix size n must be divisible by the number of threads, the benchmark was ran with n ranging from 128 to 1792 with a step of 128. Figures below show the execution time for the serial and parallel algorithm with increasing matrix size n.

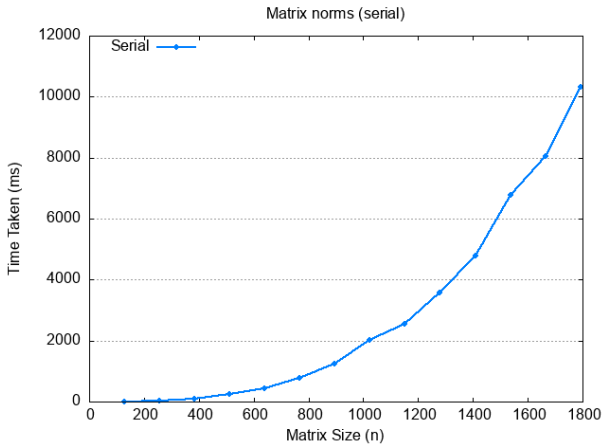

Figure 1: Matrix norm using serial algorithm

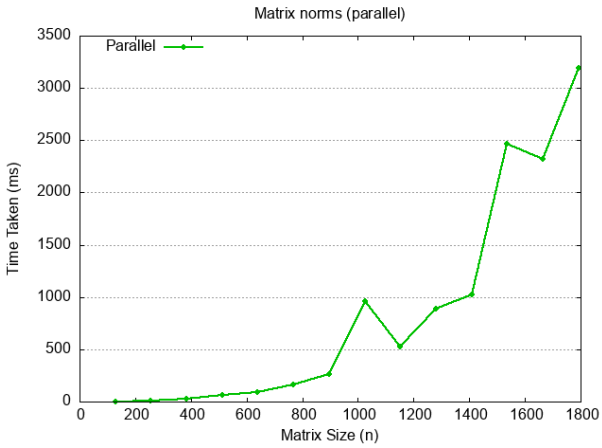

Figure 2: Matrix norm using pthreads

From the results above, we can deduce that as matrix size n increases, execution time increases. This is true for both cases because the number of computation needed increases.

3 Speedup of parallel algorithm over serial algorithm

To further investigate on the speedup of the parallel algorithm over the serial algorithm, Figure 3 below shows the plot of the execution time for the serial algorithm and the parallel algorithm on the same graph. Moreover, Table 1 below includes the execution time for both algorithms for different matrix size n.

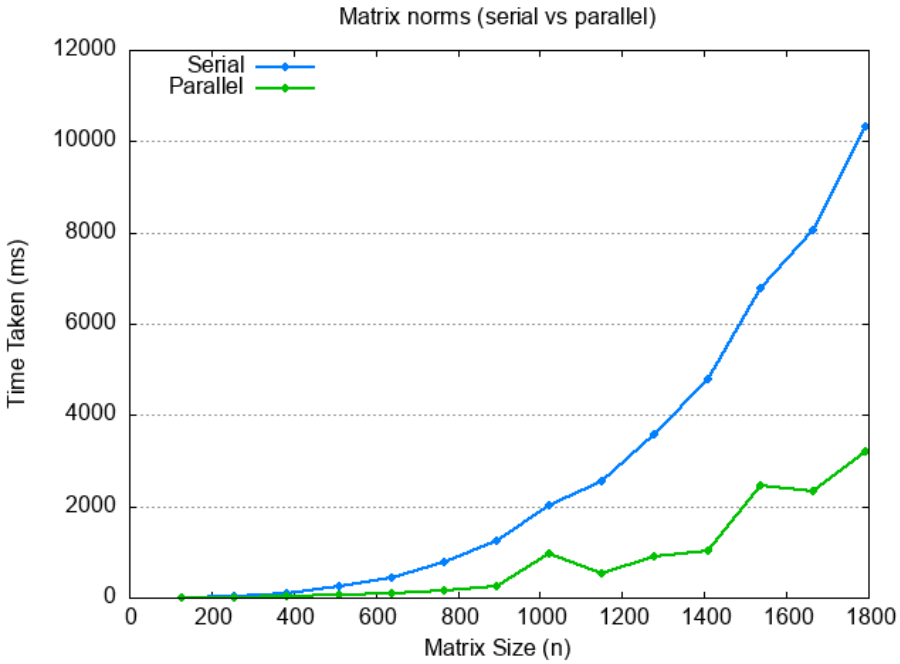

Figure 3: Matrix norm benchmark

Matrix size (n)	Time taken for serial (ms)	Time taken for parallel (ms)
128	0.97	10.83
256	5.66	33.25
384	15.36	65.15
512	34.5	117.55
640	62.42	196.83
768	117.13	317.01
896	169.87	480.37
1024	240.73	3257.47
1152	340.48	3223.21
1280	462.89	6557.70
1408	608.636	8467.80
1536	778.77	11587.90
1664	985.34	13826.90
1792	1217.72	18577.88

Table 1: Time taken for serial and parallel algorithm

From the results shown above, the average time taken for the non-blocked algorithm is 4970.56ms and 3546.13ms for the blocked algorithm. The speedup is 40% calculated using the formula:

$$\frac{(avg\ time_{blocked} - avg\ time_{non-blocked})}{avg\ time_{non-blocked}} \cdot 100\%$$

.