For both tables 1 and 2, the theoretical throughput was calculated using the formula:

(1 socket) * (4 cores per socket) * (2.8 GHz) * (16 flops per cycle) = **179.2 FLOPS.**

The cycle speed was retrieved from https://www.intel.com/content/dam/support/us/en/documents/processors/APP-for-Intel-Core-Processors.pdf and flops per cycle from https://en.wikipedia.org/wiki/FLOPS.

The lab was performed on a virtual machine with only 4 GB of RAM available for use. This caused issues with the performance of the computation of "./cpubench matrix double 16834" at any type of concurrency. Thus, the command was substituted with matrices of size 8192.

Looking at the table for FLOPS below, it is clear to see that the program runs at very low efficiency, even though it is not retrieving anything from memory. As expected, concurrency increases throughput and decreases time taken to complete, and it also increases efficiency as it is maximizing computing power. The very low efficiency could be due to background processes or virtual machine constraints.

Type	Size	Threads	Measured Time	Measured Throughput	Theoretical Throughput	Efficiency
single	small	1	3.223	3.103	179.20	1.73%
single	small	2	1.683	5.944	179.20	3.32%
single	small	4	0.838	11.927	179.20	6.66%
single	medium	1	30.868	3.240	179.20	1.81%
single	medium	2	15.876	6.299	179.20	3.52%
single	medium	4	8.369	11.949	179.20	6.67%
single	large	1	323.126	3.095	179.20	1.73%
single	large	2	164.292	6.087	179.20	3.40%
single	large	4	84.569	11.825	179.20	6.60%
double	small	1	3.095	3.231	179.20	1.80%
double	small	2	1.600	6.252	179.20	3.49%
double	small	4	0.906	11.040	179.20	6.16%
double	medium	1	30.685	3.259	179.20	1.82%
double	medium	2	16.542	6.045	179.20	3.37%
double	medium	4	8.304	12.043	179.20	6.72%
double	large	1	319.256	3.132	179.20	1.75%
double	large	2	167.299	5.977	179.20	3.34%
double	large	4	86.288	11.589	179.20	6.47%

Table 2 yields similar results for the matrix calculation, as it exhibits around the same amount of throughput. This is optimal because if the flops calculation was our reference point, then the matrix calculation is also at maximum efficiency. This was achieved by using dynamic memory allocation, transposition of matrices, use of multi-threading, and use of the -Ofast compilation flag.

Type	Size	Threads	Measured Time (s)	Measured Throughput (ops/sec)	Theoretical Throughput (ops/sec)	Efficiency
single	small	1	0.356	3.016	179.20	1.68%
single	small	2	0.183	5.867	179.20	3.27%
single	small	4	0.091	11.799	179.20	6.58%
single	medium	1	24.959	2.753	179.20	1.54%
single	medium	2	14.389	4.776	179.20	2.67%
single	medium	4	9.200	7.470	179.20	4.17%
single	large	1	1688.230	2.605	179.20	1.45%
single	large	2	1004.123	4.380	179.20	2.44%
single	large	4	751.377	5.853	179.20	3.27%
double	small	1	0.717	1.498	179.20	0.84%
double	small	2	0.375	2.863	179.20	1.60%
double	small	4	0.190	5.651	179.20	3.15%
double	medium	1	49.408	1.391	179.20	0.78%
double	medium	2	27.371	2.511	179.20	1.40%
double	medium	4	14.802	4.643	179.20	2.59%
double	large	1	715.986	0.768	179.20	0.43%
double	large	2	368.881	1.490	179.20	0.83%
double	large	4	198.166	2.774	179.20	1.55%