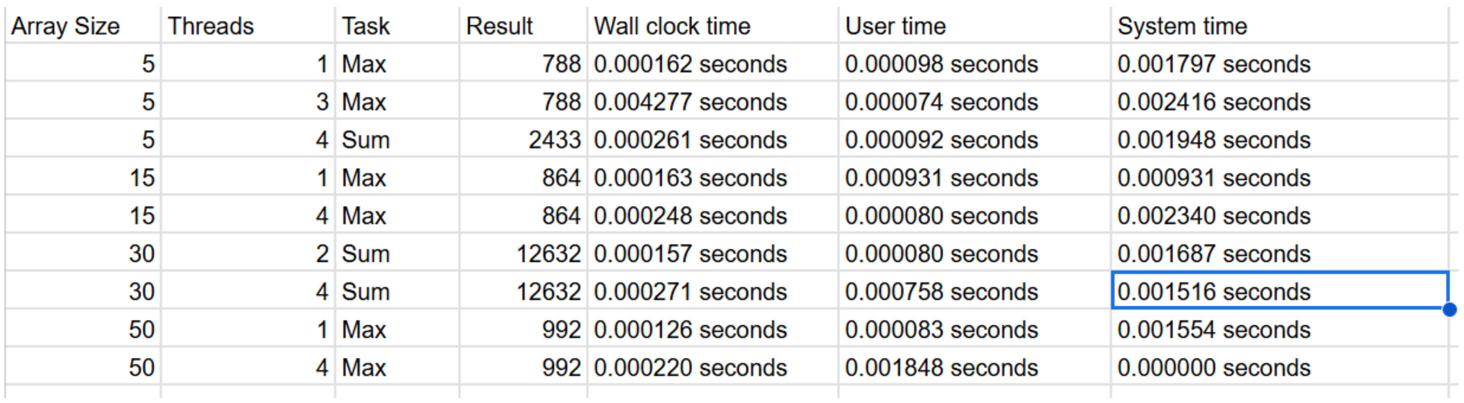
The table below shows the results of experiments conducted with different array sizes and thread counts. For an array of size 5, using 1 thread for the "Max" task achieves the lowest wall clock time (0.000162 seconds). Adding threads increases the wall clock time, as observed with 3 threads (0.004277 seconds). This means that for very small datasets, the extra work involved in managing multiple threads is more costly than the advantages of using them. With larger arrays (size 15 or 30), multithreading becomes more beneficial but shows diminishing returns. For 15 elements, using 4 threads for the "Max" task slightly increases the wall clock time (0.000248 seconds) compared to using 1 thread (0.000163 seconds). However, the system time increases significantly (0.002340 seconds) due to thread overhead. With 50 elements, the results are inconsistent: For the "Max" task, wall clock time is slightly higher with 4 threads (0.000220 seconds) compared to 1 thread (0.000126 seconds). This suggests that the additional threads adds extra work without a corresponding improvement in performance. User time increases significantly with 4 threads (0.001848 seconds), indicating that thread management and synchronization are costly for this dataset size. In general, multithreading works better for larger arrays, where the benefits outweigh the costs. For smaller datasets, the extra overhead can hurt performance.



Array Size	Threads	Task	Result	Wall clock time	User time	System time
5	1	Max	788	0.000162 seconds	0.000098 seconds	0.001797 seconds
5	3	Max	788	0.004277 seconds	0.000074 seconds	0.002416 seconds
5	4	Sum	2433	0.000261 seconds	0.000092 seconds	0.001948 seconds
15	1	Max	864	0.000163 seconds	0.000931 seconds	0.000931 seconds
15	4	Max	864	0.000248 seconds	0.000080 seconds	0.002340 seconds
30	2	Sum	12632	0.000157 seconds	0.000080 seconds	0.001687 seconds
30	4	Sum	12632	0.000271 seconds	0.000758 seconds	0.001516 seconds
50	1	Max	992	0.000126 seconds	0.000083 seconds	0.001554 seconds
50	4	Max	992	0.000220 seconds	0.001848 seconds	0.000000 seconds