John Brooks
01/26/2020

Homework 1

Implementation

The implementation is done using nested for loops. A temporary variable for storing the product matrix element sum was not used so as to increase any effect observed by optimization of locality. The algorithm is a simple naive implementation. To improve code readability and maintainability, the code which actually performs the operation of equation 1 was abstracted to its own function. This may have adversely affected performance. Inlining this function somehow resulted in worse performance. If the implementation is not subject to change, then it may merit being put into a macro definition to avoid function call overhead.

The results show that the array of structures storage was superior to the simple array storage in every test. It is assumed that due to the locality of the real and imaginary components in the array of structure implementation that CPU cache is able to increase the algorithm performance.

The tests were executed on the Yoko server.

Floating Point Operations (FLOP) Computation

For each element of the A matrix (M * N), there are a total of 4 FLOP to compute the real component and 4 FLOP to compute the imaginary component. See in equation 1 the general form of the real computation which involves two multiplications and two additions. The imaginary computation is similar.

$$sum = sum + ac - bd \tag{1}$$

Therefore, for each computation, the performance in FLOPS is found by equation 2.

$$FLOPS = (M * N * 8) / t \tag{2}$$

	Execution Time (s)		MFLOPS	
Matrix Size (NxN)	**Simple Array**	**AoS**	**Simple Array**	**AoS**
64	0.000014445	0.000008301	2268	3947
128	0.000138629	0.000078042	945	1679
256	0.000568098	0.000378166	922	1386
512	0.002411294	0.001798671	869	1165
1024	0.010492993	0.007170019	799	1153
2048	0.029871061	0.019953932	1123	1681

John Brooks
01/26/2020