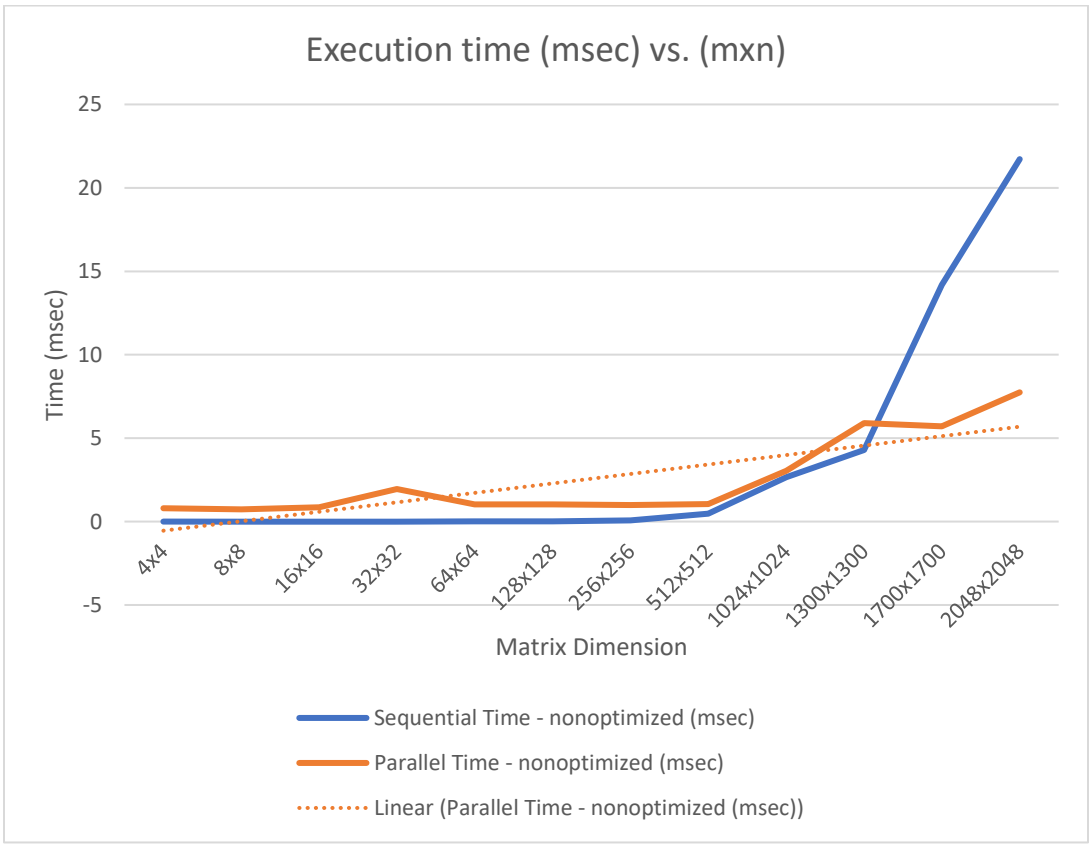
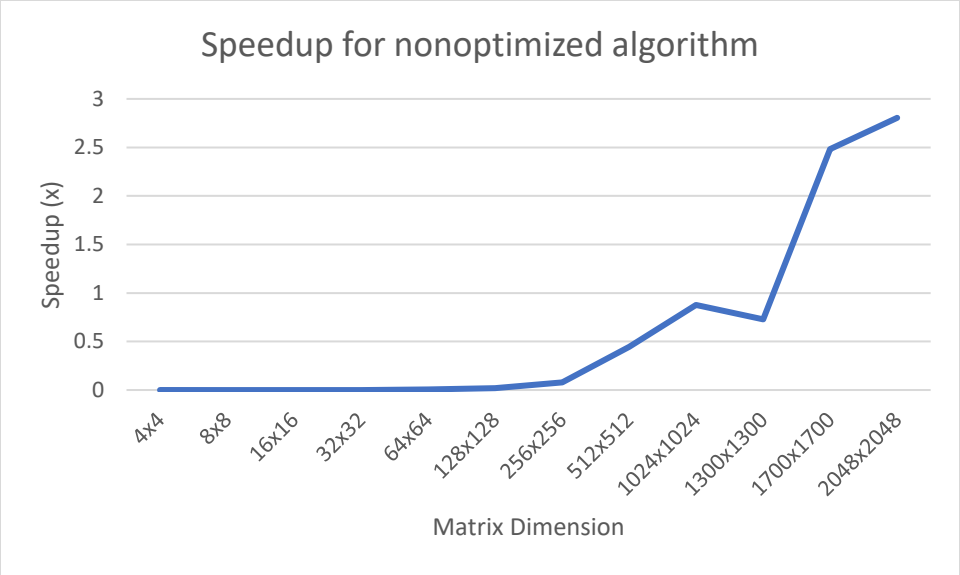
Haniye Kashgarani

Test 1, Take Home

Prob. 1:

Matrix Dimension	Sequential Time - nonoptimized (msec)	Parallel Time - nonoptimized (msec)	Speedup
4x4	0.0001	0.7922	0.000126231
8x8	0.00016	0.73197	0.000218588
16x16	0.00042	0.86207	0.000487199
32x32	0.00144	1.94619	0.000739907
64x64	0.00529	1.0298	0.00513692
128x128	0.02085	1.03052	0.020232504
256x256	0.07943	0.9923	0.080046357
512x512	0.47006	1.05555	0.445322344
1024x1024	2.65937	3.03928	0.875
1300x1300	4.29792	5.89966	0.728502998
1700x1700	14.183	5.71214	2.48295735
2048x2048	21.72	7.74396	2.804766554



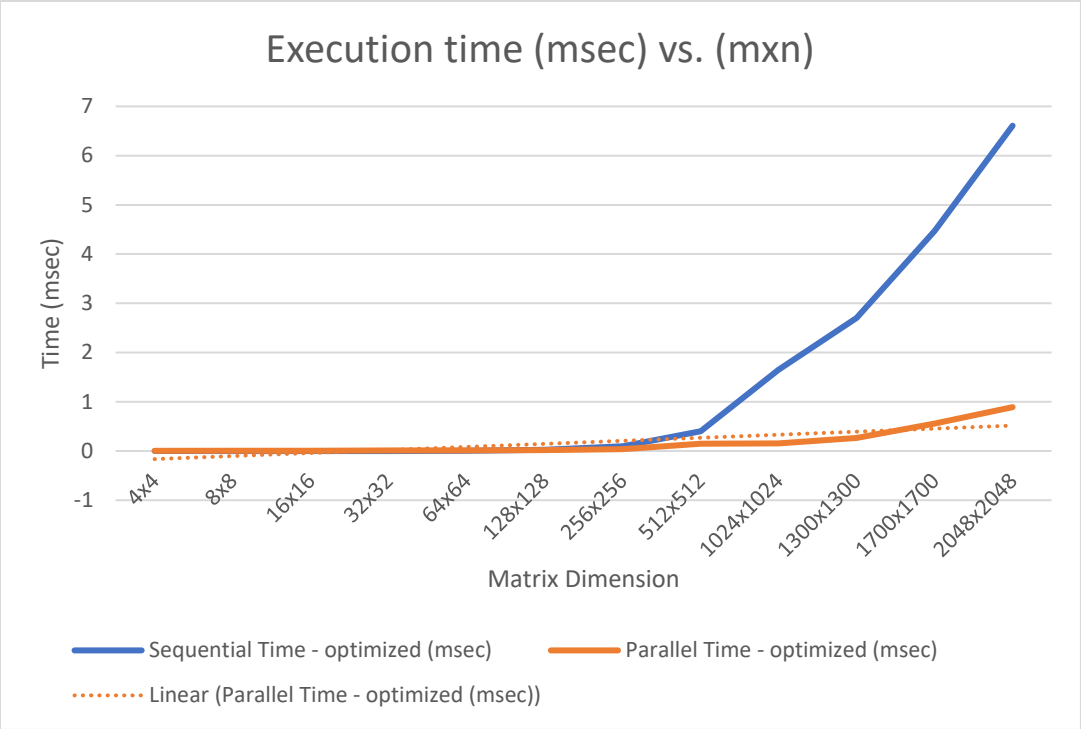


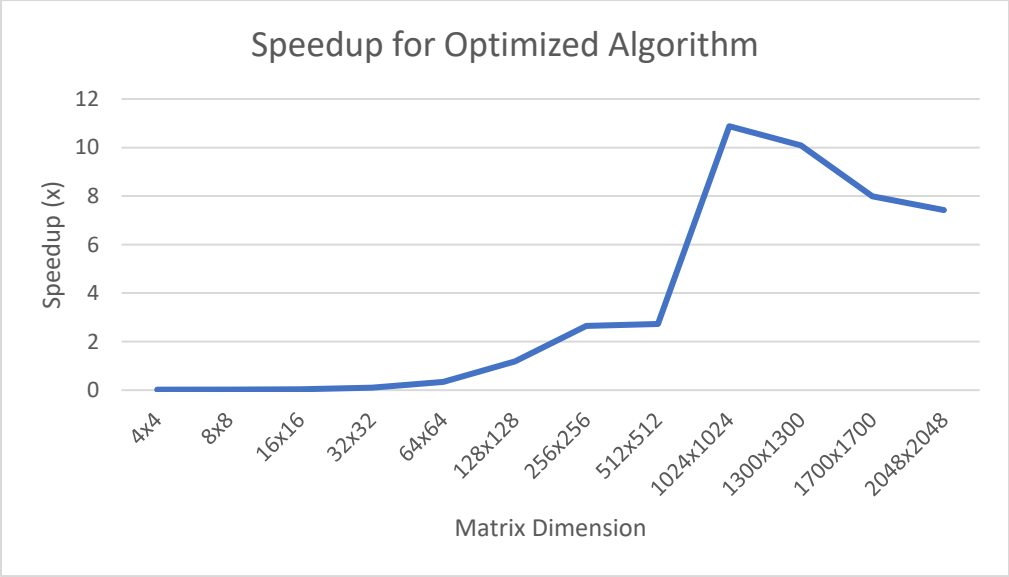
The computational complexity of the algorithm is $O(M^2)$ and the computational complexity of the parallel version should not change since the algorithm has not been modified.

Prob. 2:

The problem with the implementation is poor performance and this is because od memory access. For addressing this issue we can transpose the matrix so we can have locality of memory access this way. The performance will improve because of two reasons. 1. When we transpose the matrix the memory is accessed in a coalesced approach rather that stride access. 2. With the coalesced access, the cache will contain elements of transposed matrix which are accessed next. This reduces the memory access time significantly. With the original matrix, with the stride access the cache would not contain the next element for performing the operation.

Matrix Dimension	Sequential Time - optimized (msec)	Parallel Time - optimized (msec)	Speedup
4x4	0.00008	0.00383	0.020887728
8x8	0.00013	0.00534	0.024344569
16x16	0.00026	0.00703	0.036984353
32x32	0.00102	0.01023	0.099706745
64x64	0.00402	0.01203	0.334164589
128x128	0.02155	0.01827	1.179529283
256x256	0.09589	0.03622	2.647432358
512x512	0.40024	0.14676	2.727173617
1024x1024	1.65036	0.15172	10.87766939
1300x1300	2.70252	0.26807	10.08139665
1700x1700	4.46947	0.5595	7.988328865
2048x2048	6.60766	0.89079	7.417752781


Execution time (msec) vs. (mxn)
Time (msec)
7
6
5
4
3
2
1
0
-1
4x4
8x8
16x16
32x32
64x64
128x128
256x256
512x512
1024x1024
1300x1300
1700x1700
2048x2048
Matrix Dimension
Sequential Time - optimized (msec)
Parallel Time - optimized (msec)
Linear (Parallel Time - optimized (msec))


Speedup for Optimized Algorithm
Speedup (x)
12
10
8
6
4
2
0
4x4
8x8
16x16
32x32
64x64
128x128
256x256
512x512
1024x1024
1300x1300
1700x1700
2048x2048
Matrix Dimension