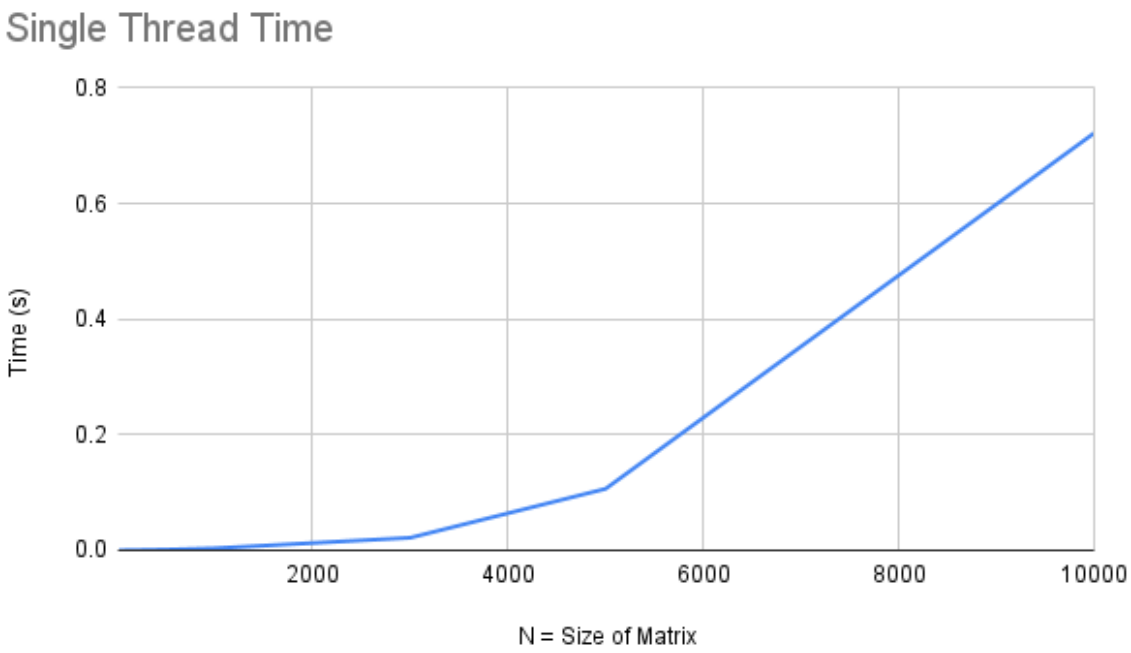
Homework 5 Report

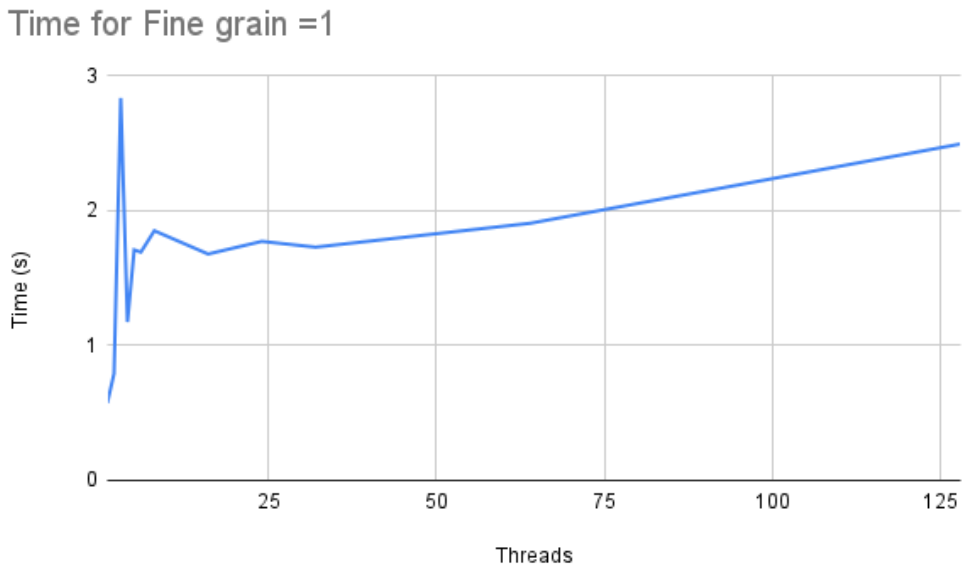
1. Single thread



N = matrix size	Single Thread Time
10	0.000002
50	0.000015
100	0.000066
500	0.001129
1000	0.003421
3000	0.021636
5000	0.106367
10000	0.721796

For transposition of a matrix varying the size of the matrix for a single thread the total execution time for each transposition with matrix size = n is increasing in exponential manner as can be seen from the graph plotted
The following is the actual time taken in seconds for a single thread

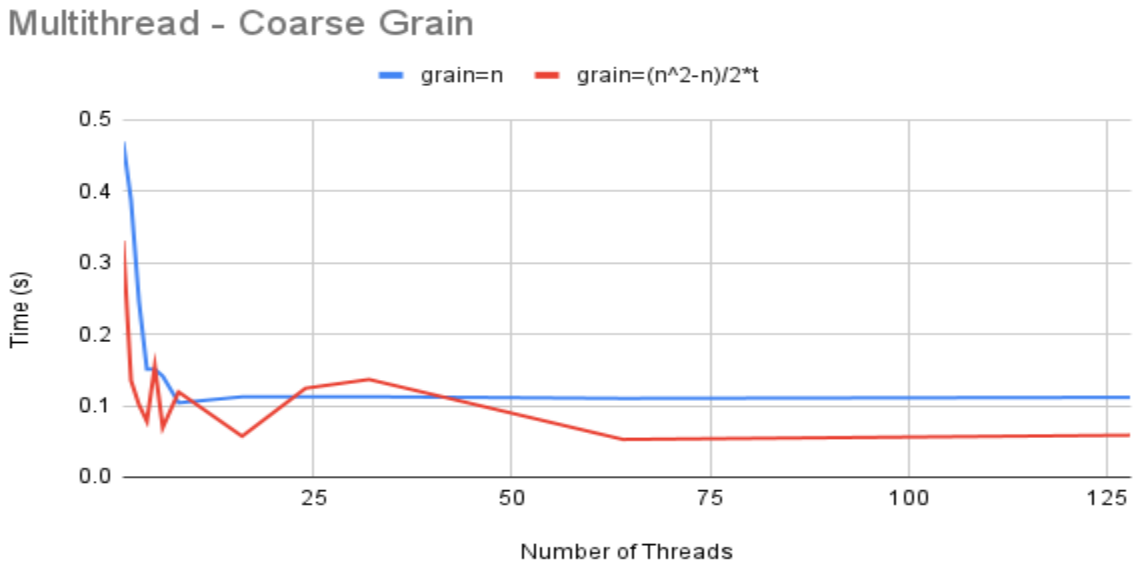
2. Multi-thread - Fine grain



Threads	Time (s) for Grain=1
1	0.573047
2	0.792543
3	2.836488
4	1.17508
5	1.709669
6	1.693302
8	1.851845
16	1.67911
24	1.772937
32	1.730157
64	1.907626
128	2.495408

For a multi threaded transposer where number of threads are varied from 1 to 128, with a grain=1 it can be seen that the time increases from thread 1 to 3 and then decreases from 3 to 4 and then smoothly increases for more number of threads. Since there are limited cores in the machine it decreases drastically initially and then increases with threads because of the extra overhead

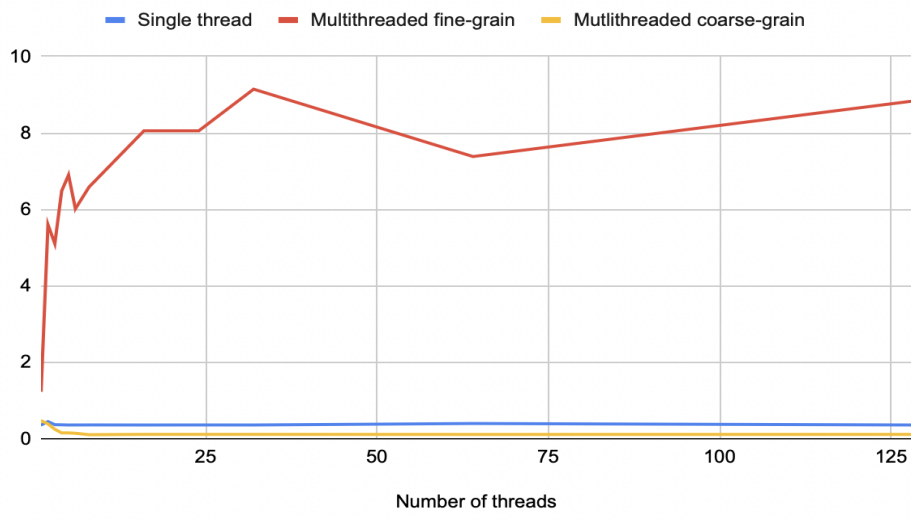
3. Multi-thread - Coarse grain



Threads	grain=n	grain=(n*n-n)/2*t
1	0.239285	0.330606
2	0.121003	0.135236
3	0.091622	0.101404
4	0.077662	0.078137
5	0.076714	0.155046
6	0.075994	0.069423
8	0.073215	0.119115
16	0.076621	0.057147
24	0.079634	0.124363
32	0.078145	0.136603
64	0.078383	0.052726
128	0.077973	0.058681

For multi threaded two grain sizes are observed grain=n and grain = (n*n-n)/2*t. As can be seen from the graph for grain=n there is a drastic decrease in time as number of threads are increased from 2 to 4 and then very slight increase as number of threads increases till 128. For grain = (n*n-n)/2*t there is fluctuation in time in initial thread values it decreases drastic from threads 3 to 4 then there is a slight increase at 5 to 8, finally for higher thread values from 16 it performs well as the grain size is equally divided among all the threads created

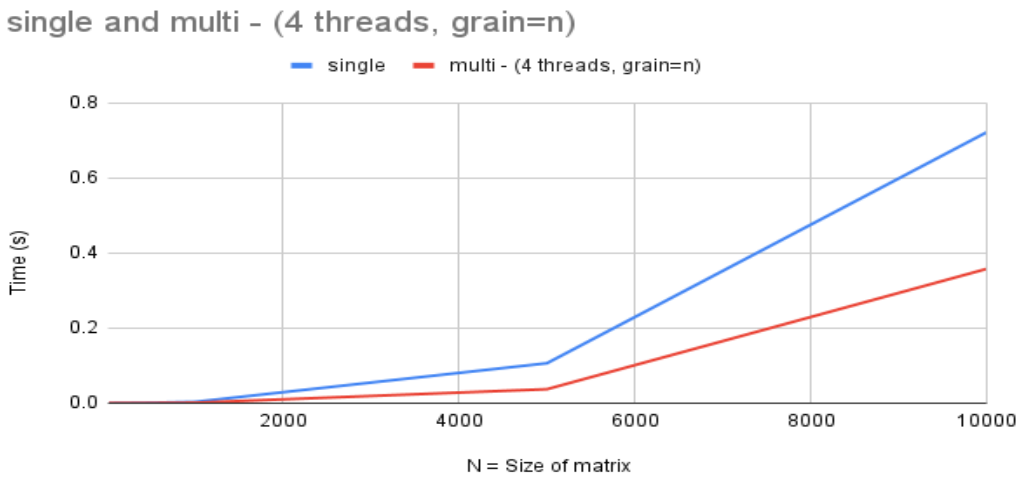
4. Plots/charts of the speedup obtained from fine and coarse-grain multithreaded versions.



Threads	Single	Multi(Grain=1)	Multi (Grain=n)
1	0.35806093	1.2367958	0.46922084
2	0.44660907	5.59896128	0.38749794
3	0.36632528	5.10244762	0.2462571
4	0.36244058	6.48185128	0.15095499
5	0.35790001	6.89737189	0.15095499
6	0.35821285	6.01588747	0.14123487
8	0.3594538	6.58544957	0.10395759
16	0.35773916	8.05444889	0.11235907
24	0.35861973	8.05444889	0.11235907
32	0.35797682	9.14394045	0.11246066
64	0.3969483	7.3815239	0.11015345
128	0.35786797	8.82920472	0.11157242

As can be seen from the above graph the multi threaded fine grain performs the worst since the time taken difference is a lot as compared to the single threaded or the multi threaded with coarse grain (grain =n) . The multi threaded coarse grain performs better than the single threaded one. For coarse grained the time taken is decreasing as the number of threads is increased till threads =8 and then there is a slight increase in time from when thread is changed from 8 to 128

5. Varying matrix size from n = 10 to n = 10,000.
   Plot of the fastest coarse grain (fixed grain) for a single threaded version (thread = 1)



N	single	multi - (4 threads, grain=n)
10	0.000002	0.000087
50	0.000015	0.000087
100	0.000066	0.00019
500	0.001129	0.000374
1000	0.003421	0.001159
3000	0.021636	0.010944
5000	0.106367	0.037173
10000	0.721796	0.35778

Above figure is a transposition of matrix as the size of matrix is varied fora single thread vs a multi thread with the number of thread=4 and coarse = n. The multi threaded performs better than the single thread as the size of the matrix is increased the difference between the execution time for single and multi thread also increases as can be seen from the graph the difference between red and blue line is increasing with increasing size of the matrix as.