MA513
Parallel Computing
Assignment-4 Report

Paranjay Bagga
160101050

Problem Statement 1 :-

Implementation of three different ***Matrix-vector multiplication*** algorithms with observation of overhead and speedup in each algorithm using MPI programming.

System Specifications :-

All the experiments have been performed on **Dell Inspiron 5559** laptop with **Intel i5-6200U dual core processor** and **8 GB RAM**. Each core has **2 threads.**

Algorithms :- *(Multiplication : $A_{NxN} * B_{Nx1} = R_{Nx1}$)*

1. **Row Division based :**
   Each processor has N/p number of rows and whole B vector initially (after data distribution by PE-0). In order to compute R vector, all processors multiply their corresponding partial matrix with whole B vector and thus holding N/p part of R vector individually. This partial result is finally shared by all the PEs with PE-0 (gathered).

2. **Column Division based :**

   Each processor has N/p portion of B vector & N/p no. of columns of A matrix initially (for easy communication, the Transpose matrix of A has been used to distribute data row wise partially). Then the partial result of the R vector is calculated by all processors which is then reduced so that the final R vector is received by PE-0.

3. **Block Division based :**

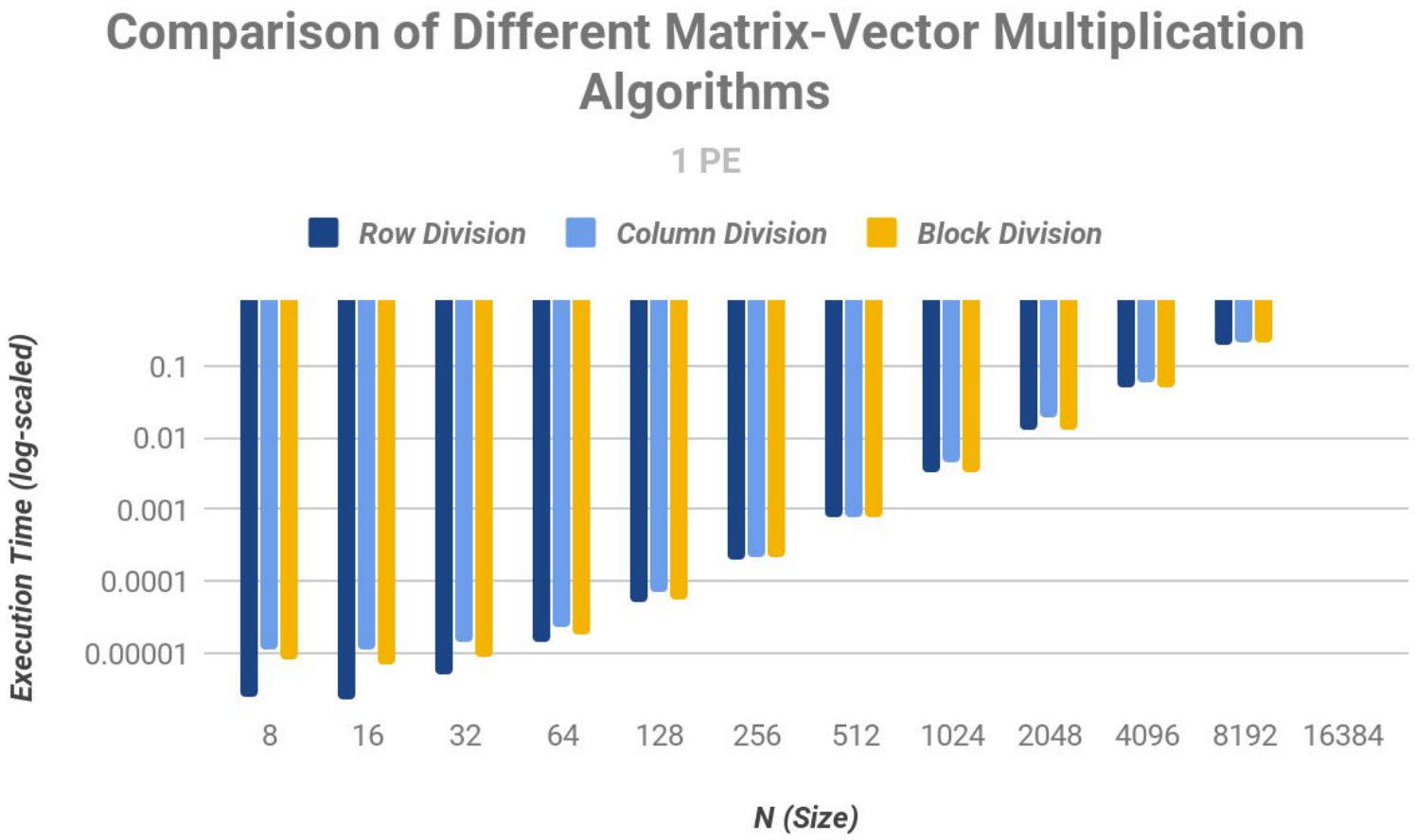
   The A matrix has been divided into p blocks with √p blocks in each row and √p blocks in each column, thus giving one block to each processor. This is accomplished by first scattering N/√p rows of the original NxN A matrix in the first column by PE-0. Following this, the matrices are block wise scattered to each PE by each processor of the first column. Then, each processor of the first row is given N/√p portion of B vector (which is scattered by the PE-0) and further it is broadcasted by them to each PE in their respective column. Now all the processors compute the N/√p portion of the partial R vector which is reduced into final R vector along the row. The first processor of each row again holds the final part of R vector. This final R vector is ultimately gathered by PE-0.

Experiments :-

The value of *N* is varied from **2^3(8)** upto **2^14(16284)**.
The results are shown in the tables and graphs below. The values in the table are average values of 4 executions to avoid any unexpected behaviour introduced as a result of thread switching.

1. Using 1 PE :

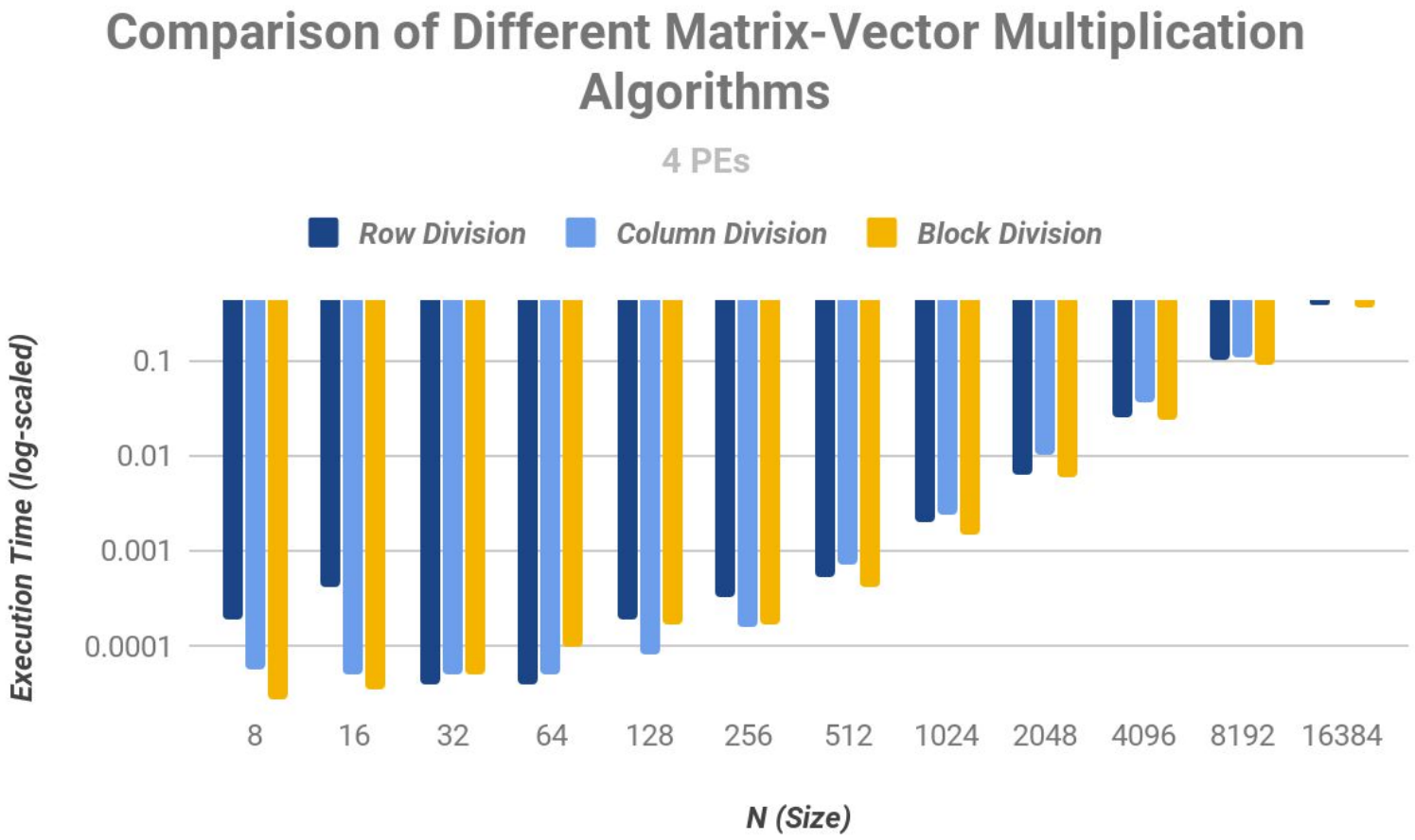
Size	Row Division	Column Division	Block Division
8	2.4E-06	1.13E-05	8.5E-06
16	2.2E-06	1.13E-05	6.9E-06
32	5.1E-06	1.42E-05	8.7E-06
64	1.38E-05	2.3E-05	1.8E-05
128	5.01E-05	7.2E-05	5.55E-05
256	0.0002021	0.0002152	0.0002096
512	0.0008084	0.0008063	0.000796
1024	0.0032018	0.0043716	0.0033133
2048	0.0126309	0.0192264	0.0127742
4096	0.0496019	0.0593319	0.0509108
8192	0.1993765	0.2073022	0.2031335
16384	0.7972415	0.798585	0.803736



2. Using 4 PEs :

Size	Row Division	Column Division	Block Division
8	0.0001846	5.64E-05	2.62E-05
16	0.0004105	4.82E-05	3.33E-05
32	3.82E-05	4.83E-05	4.89E-05
64	3.9E-05	5E-05	9.76E-05
128	0.0001889	7.83E-05	0.0001642
256	0.0003212	0.0001532	0.0001655
512	0.0005182	0.0007179	0.0004203

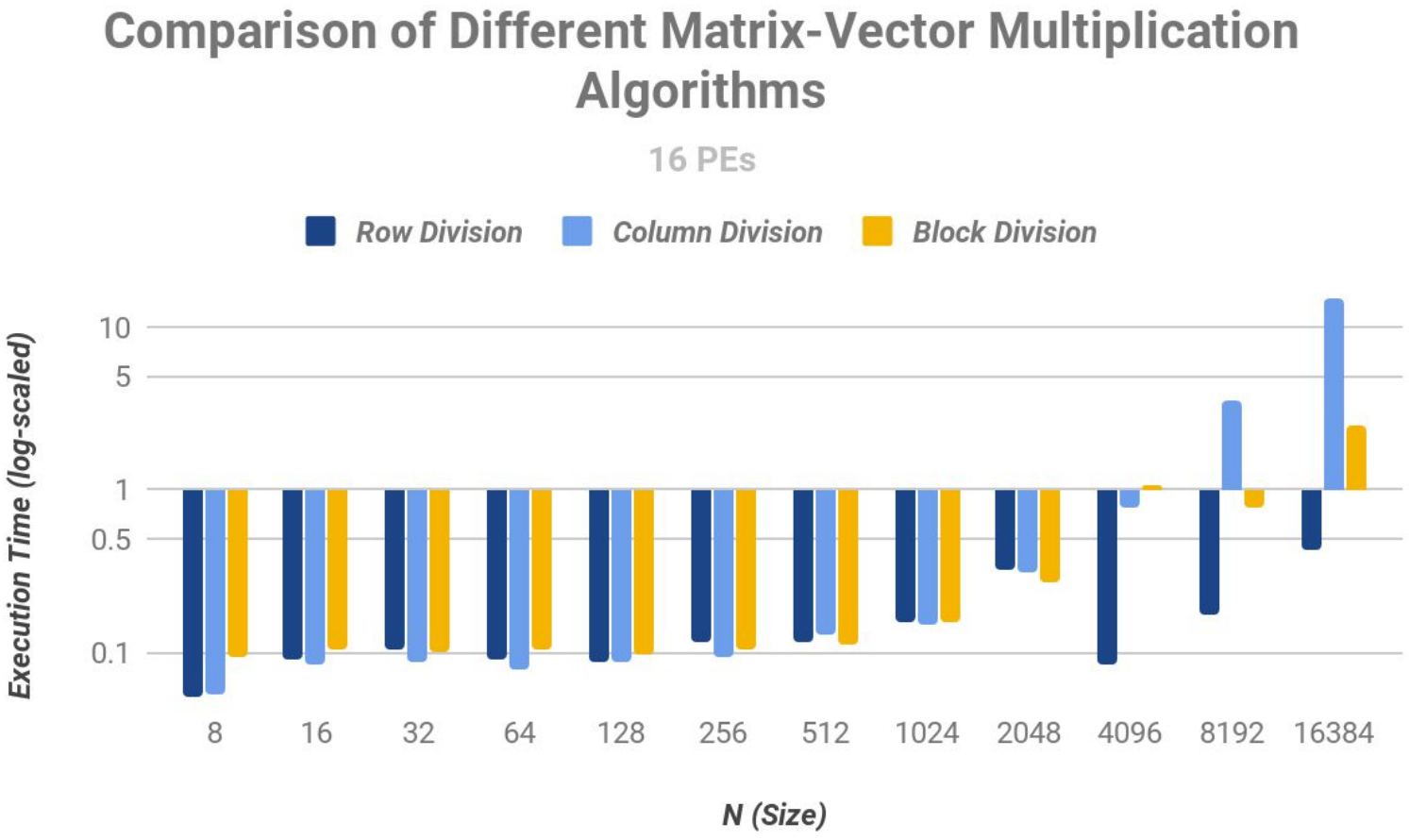
1024	0.0019554	0.0024332	0.001473
2048	0.0063958	0.0102061	0.0057853
4096	0.0250628	0.0362368	0.0230283
8192	0.099169	0.1098095	0.0915521
16384	0.369378	0.4220805	0.36295075



3. Using 16 PEs :

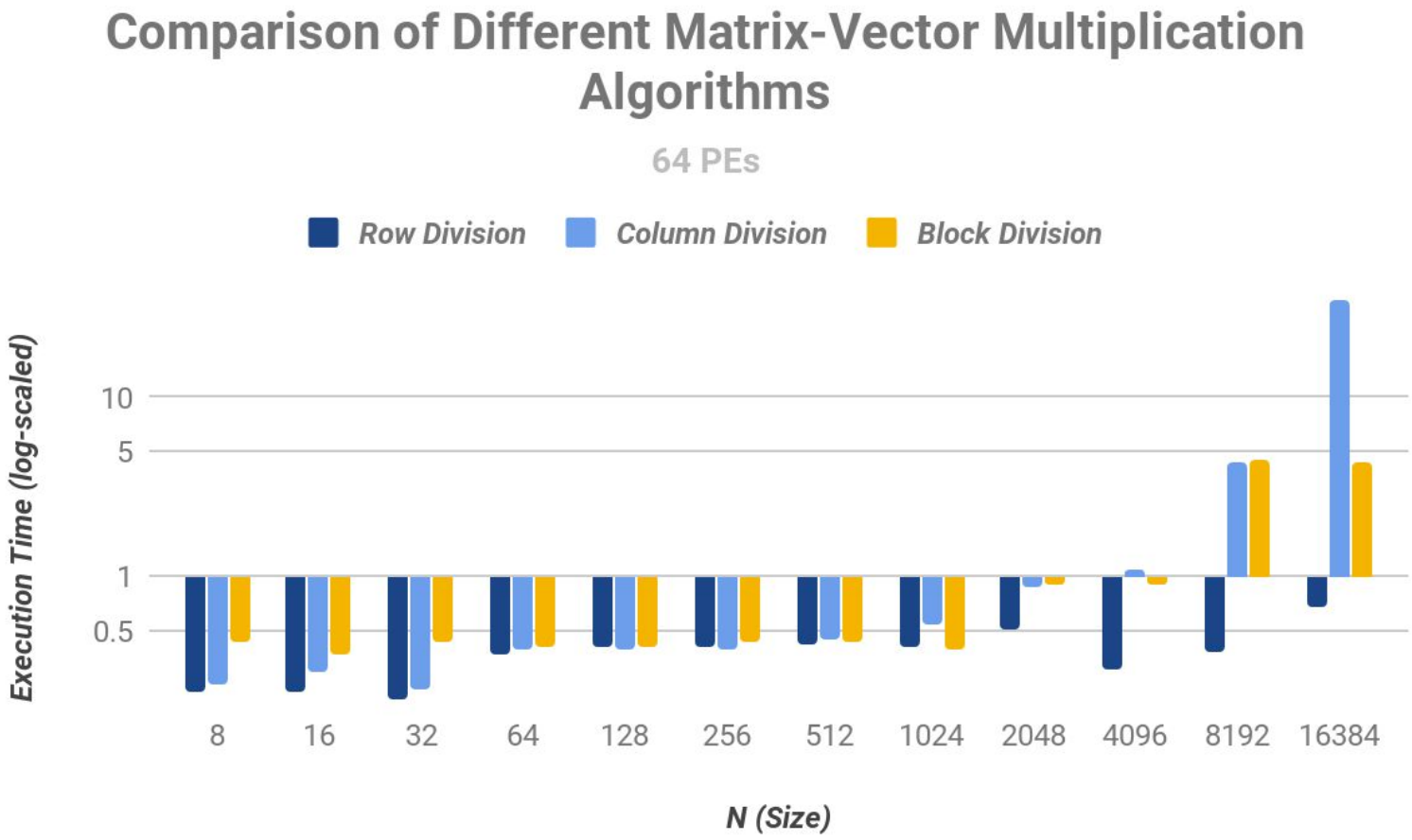
Size	Row Division	Column Division	Block Division
8	0.0521884	0.0556591	0.0926123
16	0.0903168	0.0851541	0.1032856

32	0.1040643	0.0863391	0.099999
64	0.0885938	0.078999	0.1036787
128	0.0874164	0.0862481	0.0965704
256	0.1170082	0.093696	0.1031648
512	0.1137181	0.1300625	0.1114262
1024	0.1502948	0.1497022	0.155167
2048	0.3223714	0.3147243	0.2671656
4096	0.0829516	0.7723864	1.0541617
8192	0.1701406	3.5568879	0.7750112
16384	0.42790975	14.83326625	2.47251925



4. Using 64 PEs :

Size	Row Division	Column Division	Block Division
8	0.22442825	0.2522365	0.43169175
16	0.22641475	0.291226	0.3616925
32	0.20255225	0.23128225	0.43268475
64	0.3683715	0.3922355	0.40553825
128	0.398254	0.384084	0.39852975
256	0.39681025	0.38683075	0.42892625
512	0.42159275	0.43982225	0.429166
1024	0.39781875	0.54171525	0.38795825
2048	0.4975175	0.86489225	0.888801
4096	0.30493	1.0870755	0.8851085
8192	0.37786825	4.29457875	4.50887725
16384	0.6628925	34.43415125	4.34788



Observations :-

- It is observed that the Block Division algorithm performs best in case when 4 PEs are used. This is followed by Row Division and then Column Division.
- In all other cases, Row Div. algorithm performs better than the two followed by Block Division and the worst by Column Division.
- There was only slight difference in case when 1 PE was used. But in case of 16 & 64 PEs, as N increased, the difference between the respective execution times began to become huge (from none) with Column Division Algorithm behaving worstly.