VOLATILITY
INDEX

1) IMPLEMENTATION

The project is implemented using two mapper and reducer chaining. First mapper reads the input line by line and gives outputs key as the company name and date with adjusted close price as value. The task of first reducer is to collect outputs from mapper and calculate Monthly rate of return of the company and after calculation Rate of return of each month, finally we are calculating Volatility index for that share. First reducer then write into a temporary file with company name as Key and Volatility index as value.

Now second mapper will read the output file of first mapper reducer pair it will emit all the keys to be same and value to be volatility concatenated with the stock name. In the reducer these iterable values will be added to a array which we will sort and get Top and Bottom 10 stocks according to volatility index.

2) PERFORMANCE AND EXECUTION LABELS

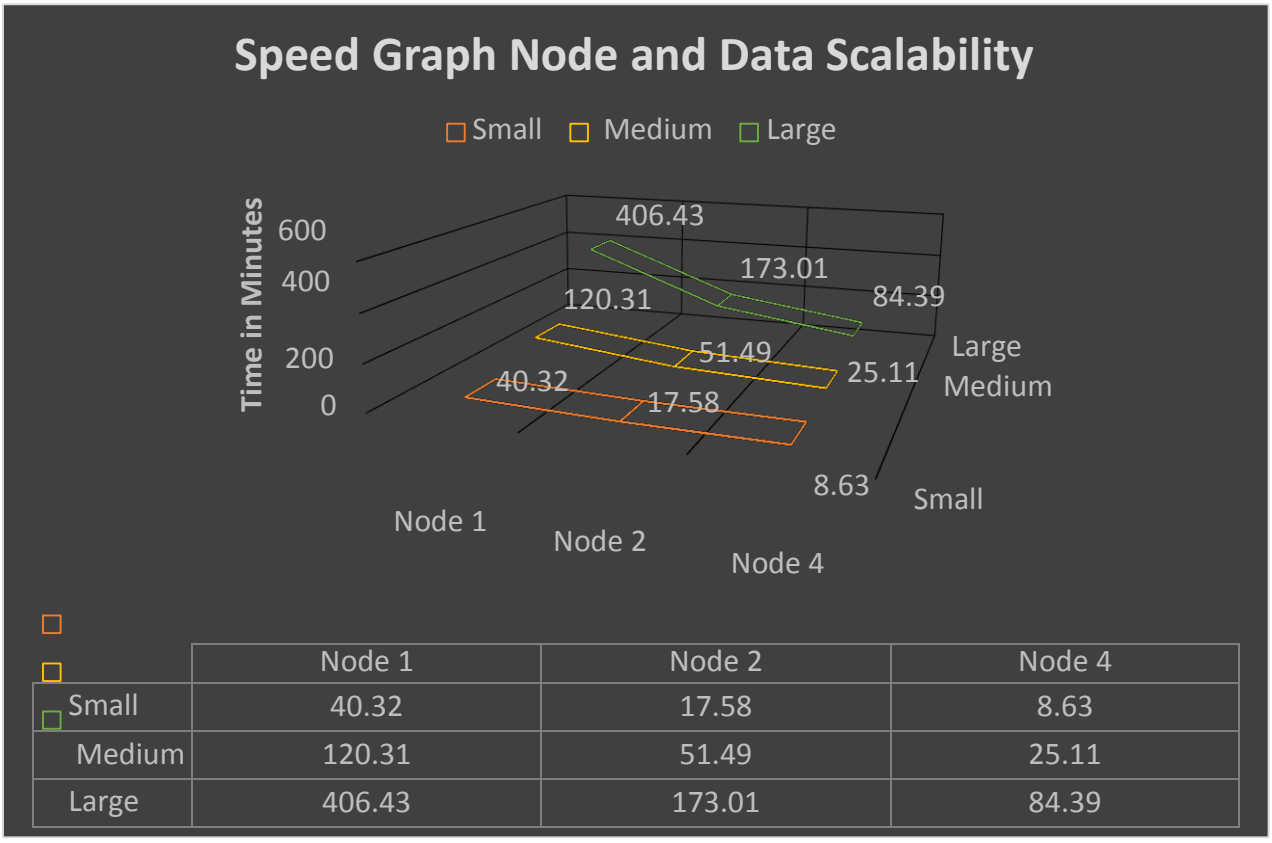
Problem Size	Execution Time: 1 node (12 cores)(minutes)	Execution Time: 2 nodes (24 cores)(minutes)	Execution Time: 4 nodes (48 cores)(minutes)
Small	40.32	17.58	8.63
Medium	120.31	51.49	25.11
Large	406.43	173.01	82.50

By measuring the values we can conclude that by increasing the data set size i.e. from small to medium to large the execution times on all node columns increases. Secondly we can say that as number of nodes increases for same data set size the performance increases. The execution

time by more than half if number of nodes are multiplied by 2. Hence as the number of nodes increase the performance also increase. The implementation is also scalable as it runs when data set is increased.

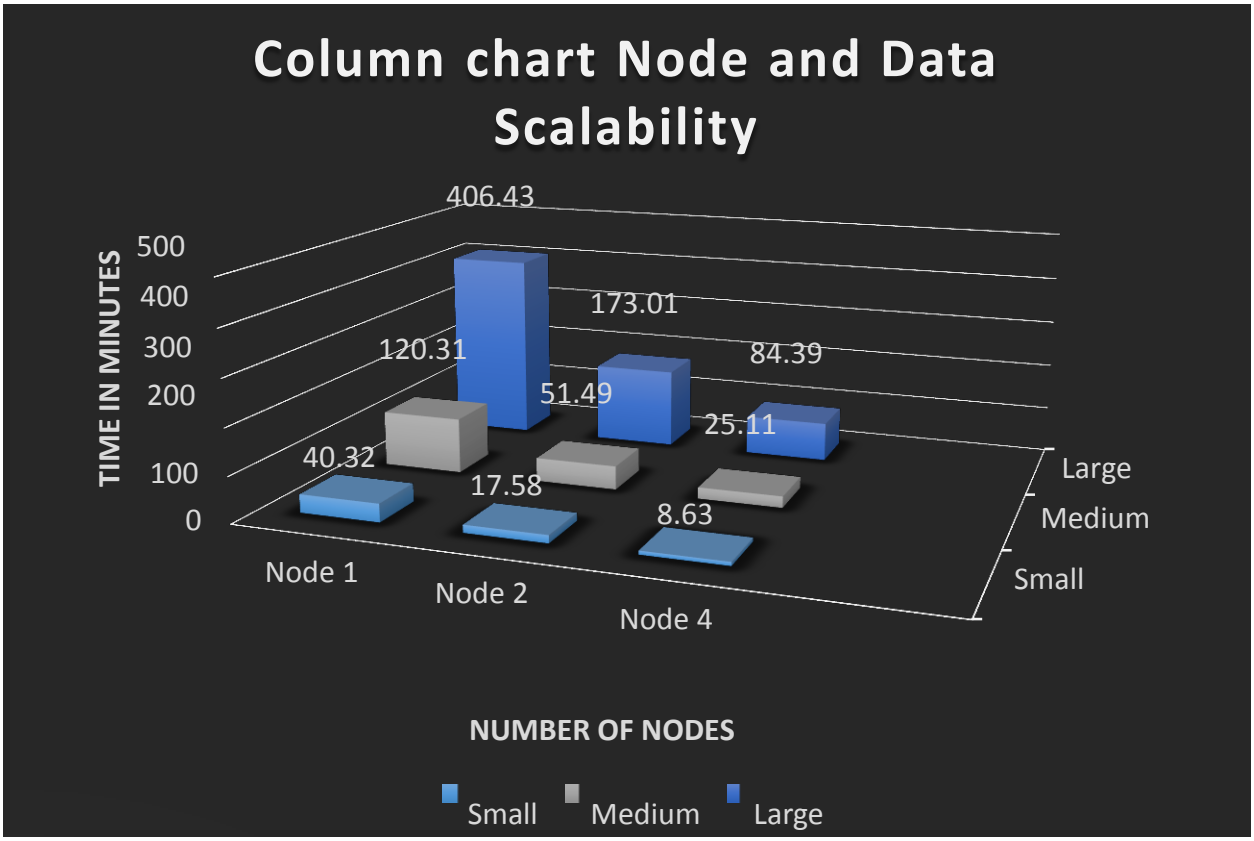
3) GRAPHS AND SCALABILITY

a) Composite Line Graph Representation- x-axis->Number of nodes, y-axis-> Time in minutes.



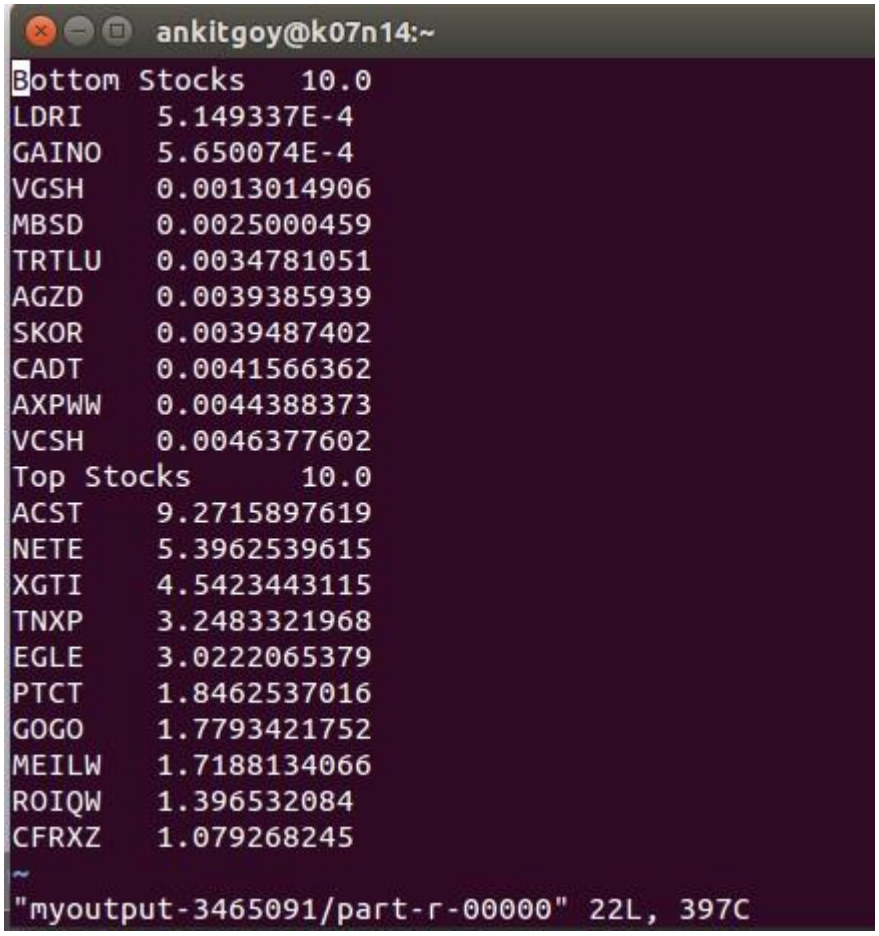
	Node 1	Node 2	Node 4
Small	40.32	17.58	8.63
Medium	120.31	51.49	25.11
Large	406.43	173.01	84.39

b) Column chart representation.

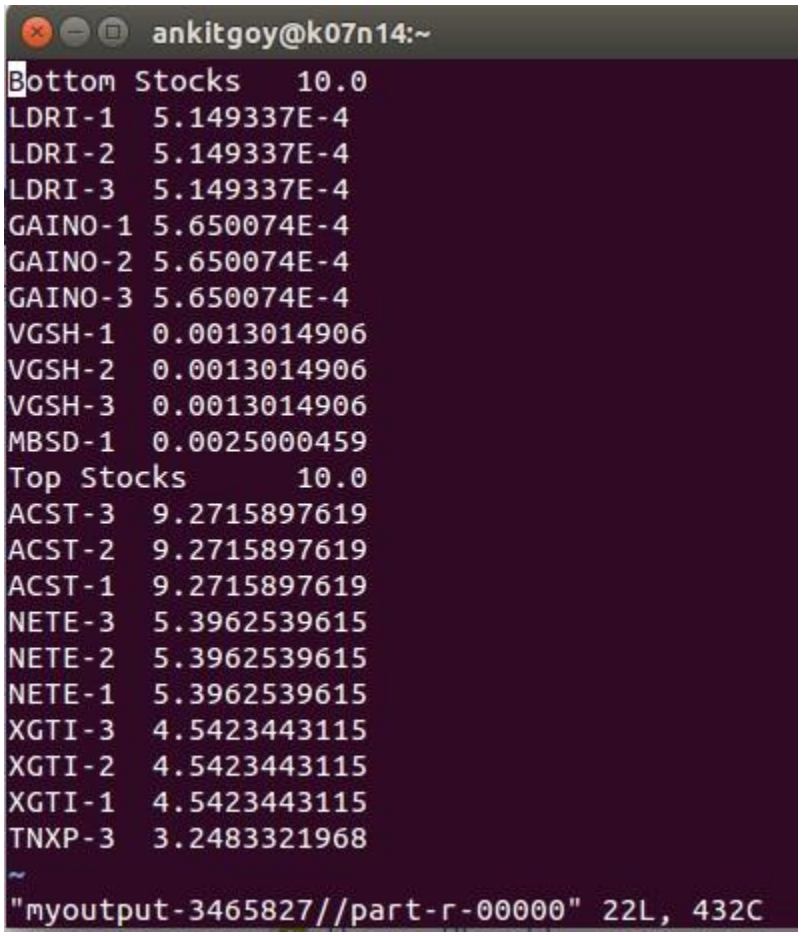


4) OUTPUT VALUES

a) Small data set



b) Medium data set



c) Large data set