CSE-587 Data Intensive Computing
Assignment-1
Hadoop Mapreduce

Karthik Kiran
kkiran@buffalo.edu

INTRODUCTION

Mapreduce is run on Hadoop to compute the volatility of stocks in NASDAQ. Daily data of 2970 stocks on NASDAQ market for 3 years from 01/01/2012 to 12/31/2014 is given. A characteristic of stocks named Volatility is computed for each stock. This
volatility index gives the earning potential of stocks for the period. Task is to find the Top 10 stocks with highest earning potential and Top 10 stocks with least earning potential for the given period.

DATA FORMAT

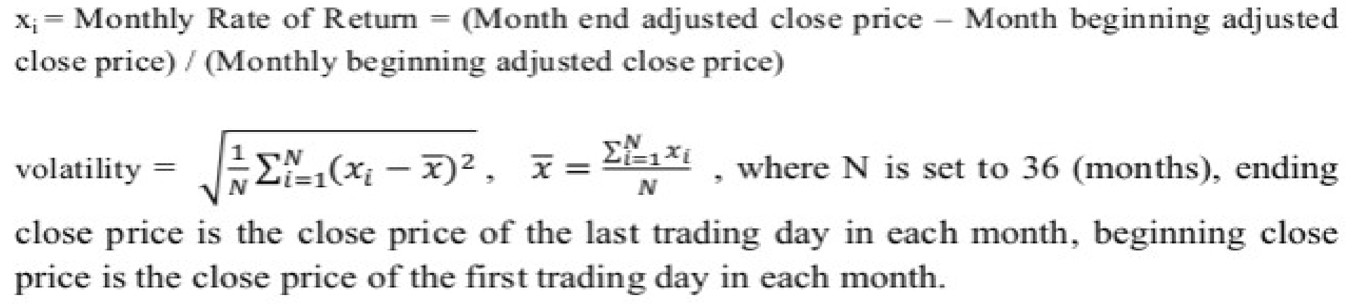
The data for each stock is provided in CSV files. There are 7 columns in each row. Each row represents a trading day.
Example:
Date ,Open,High ,Low ,Close ,Volume ,AdjClose
12/31/2014 , 112.82 , 113.13 , 110.21 , 110.38 , 41403400 , 110.38

Date represents the date of the stock.
Open represents the open price in that day of stock.
High represents the highest price in that day of stock.
Low represents the lowest price in that day of stock.
Close represents the close price in that day of stock.
Volume represents the volume in that day of stock.
AdjClose represents the close price in that day of stock.

DATASET

The complete dataset consists of 3 folders: small, medium and large. Small contains 2970 stocks. Medium contains 8910 stocks and large has 29700 stocks. The required task is run on all three sets on 1(12 cores), 2(24 cores) and 4(48 cores) nodes. Scalability of implementation on different data sizes and number of nodes is evaluated.

CALCULATIONS



x_i = Monthly Rate of Return = (Month end adjusted close price – Month beginning adjusted close price) / (Monthly beginning adjusted close price)

volatility = $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$, $\overline{x} = \frac{\sum_{i=1}^{N} x_i}{N}$, where N is set to 36 (months), ending close price is the close price of the last trading day in each month, beginning close price is the close price of the first trading day in each month.

IMPLEMENTATION

The complete computation is carried out in 3 phases. Each phase has 1 Mapper and 1 Reducer. So a total of 3 mappers and reducers are used. Each reducer writes the output into a file which is then read by the subsequent mapper. The flow and functionality of the implementation is as below:

Phase1:

- Input to this phase is the dataset which consists of CSV files where each file represents data for a single stock for the time period of 3 years.
- Mapper1 reads each input file line by line and generates a (key,value) pair as output.
- Each line is split using comma(,) as delimiter and the date and adj close price are stored. The file name which is also the stock name is retrieved within Mapper1.
- Stockname is appended with the year and month value. This is the key.
  Key = stockname+”,”+yearmonth
- The day from date column in each line is appended with the adjusted close price. This is considered as value for each key.
  Value = day+adjcloseprice
- For a single key, there would be multiple values which is equal to number of trading days for the particular month as provided in the dataset.
- The output of Mapper1 is input to Reducer1.
- Reducer1 processes each (key,list(values)) pair it receives from Mapper1.
- For each month, the monthly rate of return is calculated within Reducer1. This is done by obtaining first and last day adjusted close price values for a month and using the formula mentioned earlier.
- So for each key passed from Mapper1, Reducer1 would calculate the Monthly rate of Return.
- Key value written into Phase1 output is same as the key from Mapper1.
- Value is the Monthly Rate of Return.
- The output of Phase1 is written into a file which becomes input to Phase2.

Phase2:

- Mapper2 reads each line of the input file.
- Generates key by obtaining stockname from each line.
  Key = stockname
- Value is the Monthly Rate of Return value generated from Phase1.
  Value = Monthly Rate of Return
- Mapper2 generates as many (key,value) pairs as number of stocks in the dataset.
- Reducer2 processes each (key,list(value)) pair and calculates the volatility for each key, that is for each stock.
- Output of Reducer2 is written to a file with key as stockname and value as volatility.
- This Phase2 output file is input to Phase3.

Phase3:

- Mapper3 reads each line of input and obtains the stockname and corresponding volatility.
- Stockname and volatility are appended to form value.
  Value = volatility+"!"+ stockname
- Key is kept constant for all values generated within Mapper3, in my case its "A".
- Reducer3 processes the (key,list(values)) from Mapper3 and writes only the Top 10 and bottom 10 stocks based on volatility into the Output file. This output file gives 10 stocks with highest and 10 stocks with lowest earning potential.

RESULTS

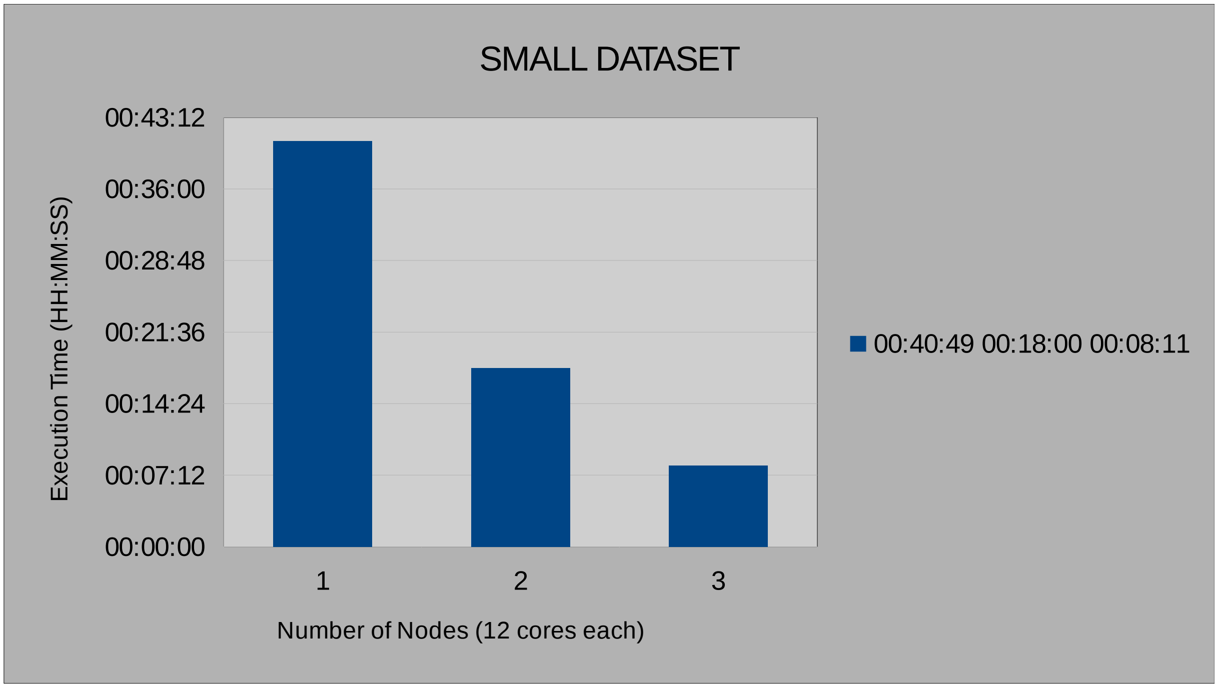
Problem Size	Execution Time: 1 node (12 cores) (hh:mm:ss)	Execution Time: 2 nodes (24 cores) (seconds)	Execution Time: 4 nodes (48 cores) (seconds)
Small	00:40:49	00:18:00	00:08:11
Medium	02:00:01	00:50:47	00:23:46
Large	06:46:17	02:48:34	01:15:21

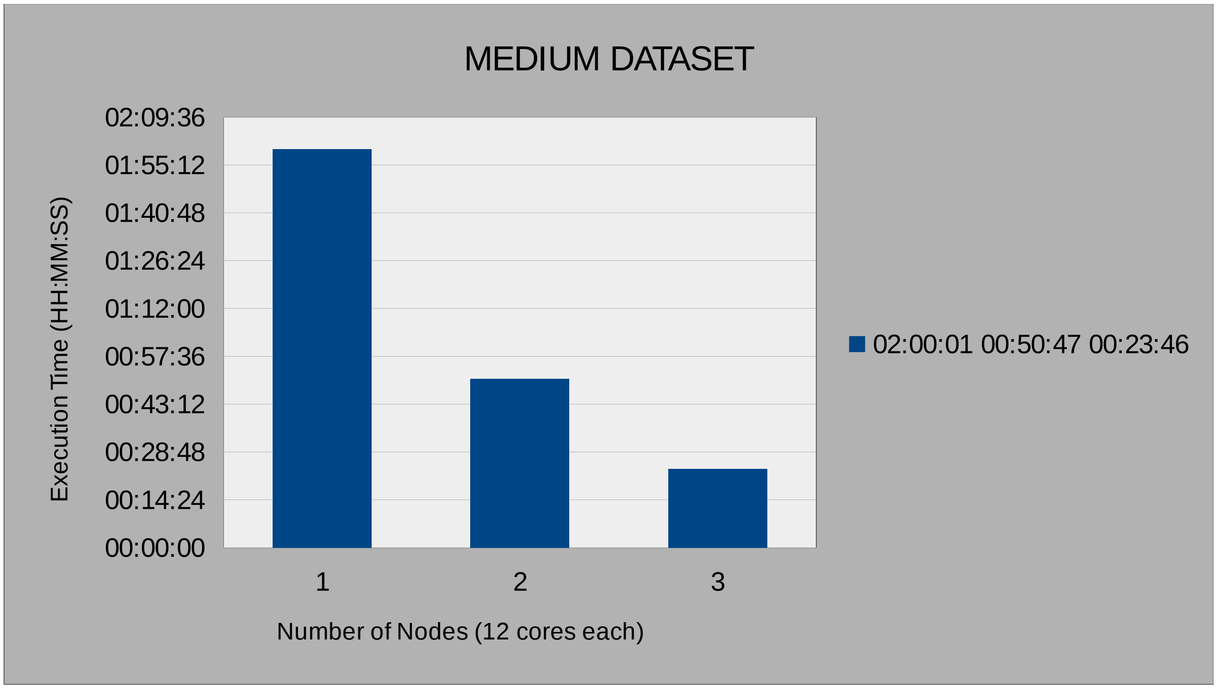
Analysis:

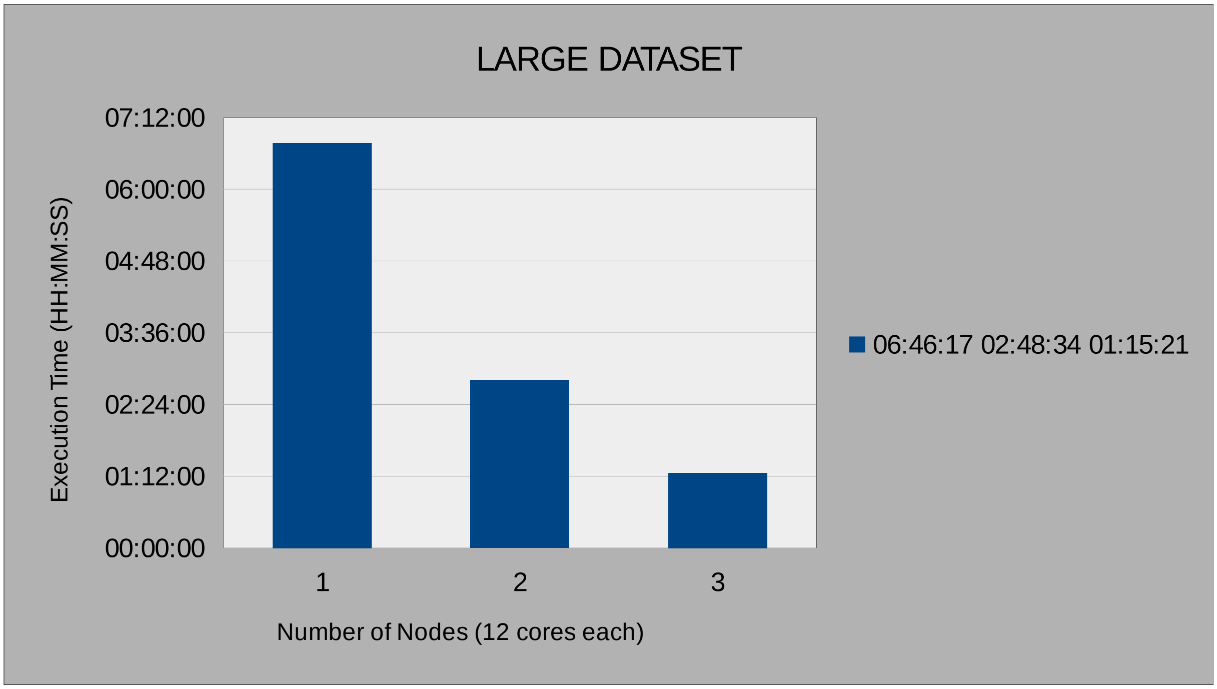
As the number of nodes increases, the time taken for execution reduces. As we can see for a particular dataset, the time take with 2 nodes is approximately half the time taken for execution with 1 node. Similarly, the time taken for execution with 4 nodes is approximately half the time taken for execution with 2 nodes. Hence, it can be inferred that the time taken for execution with N nodes is approximately half the time taken for execution with (N/2) nodes.

Also, the same output is obtained for each dataset using different number of nodes. Hence data scalability is satisfied by the program.

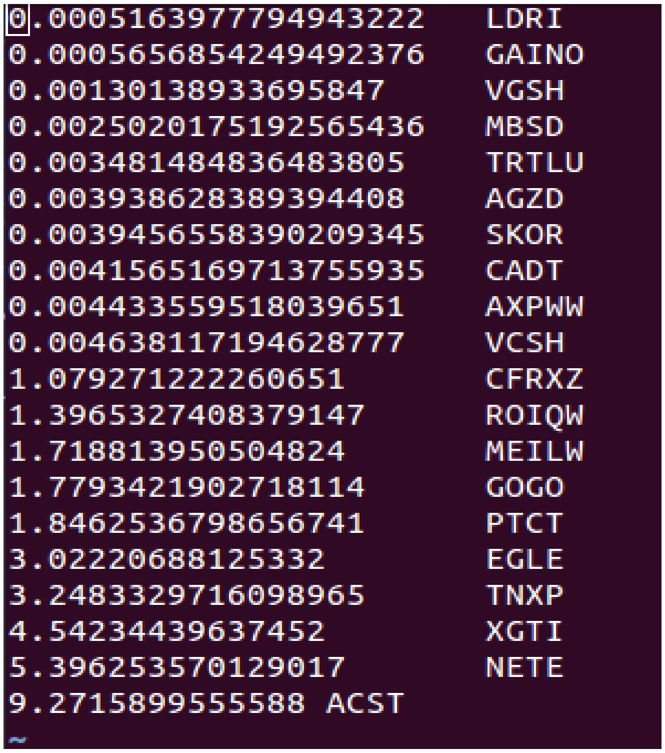
SPEEDUP GRAPHS



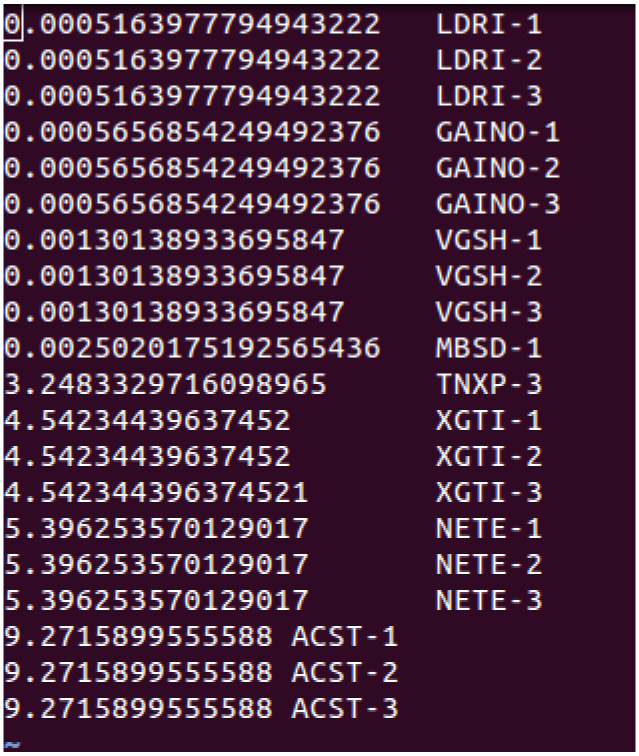




Top 10 and Bottom 10 Results for Small, Medium and Large Sets

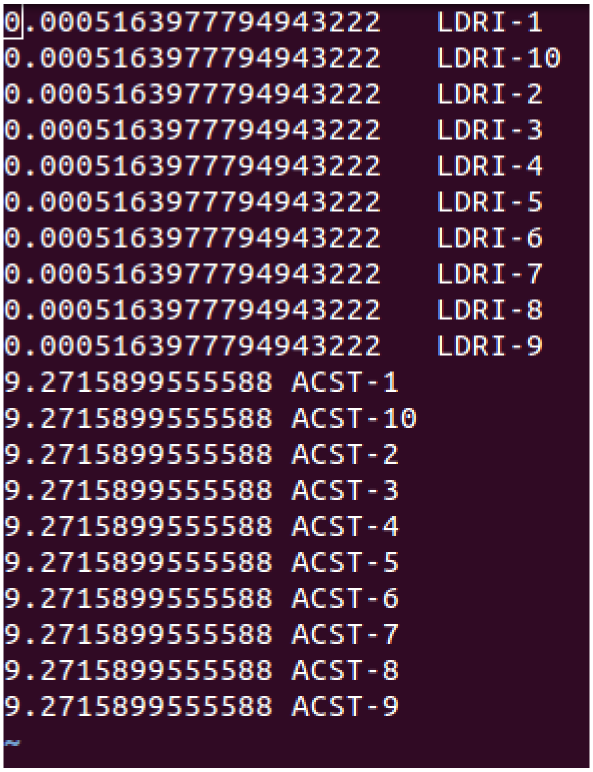


```
0.00051639777794943222    LDRI
0.0005656854249492376    GAINO
0.0013013893369584 7     VGSH
0.0025020175192565436    MBSD
0.0034814848364838 05    TRTLU
0.0039386283893944 08    AGZD
0.0039456558390209345    SKOR
0.0041565169713755935    CADT
0.0044335595180396 51    AXPWW
0.0046381171946287 77    VCSH
1.07927122226065 1       CFRXZ
1.3965327408379147       ROIQW
1.718813950504824        MEILW
1.7793421902718114       GOGO
1.8462536798656741       PTCT
3.02220688125332         EGLE
3.2483329716098965       TNXP
4.54234439637452         XGTI
5.396253570129017        NETE
9.2715899555588 ACST
~
```



```
0.00051639777794943222    LDRI-1
0.00051639777794943222    LDRI-2
0.00051639777794943222    LDRI-3
0.00056568542494923 76    GAINO-1
0.00056568542494923 76    GAINO-2
0.00056568542494923 76    GAINO-3
0.0013013893369584 7     VGSH-1
0.0013013893369584 7     VGSH-2
0.0013013893369584 7     VGSH-3
0.0025020175192565436    MBSD-1
3.2483329716098965       TNXP-3
4.54234439637452         XGTI-1
4.54234439637452         XGTI-2
4.542344396374521        XGTI-3
5.396253570129017        NETE-1
5.396253570129017        NETE-2
5.396253570129017        NETE-3
9.2715899555588 ACST-1
9.2715899555588 ACST-2
9.2715899555588 ACST-3
~
```

Small Output **Medium Output**



```
0.0005163977794943222   LDRI-1
0.0005163977794943222   LDRI-10
0.0005163977794943222   LDRI-2
0.0005163977794943222   LDRI-3
0.0005163977794943222   LDRI-4
0.0005163977794943222   LDRI-5
0.0005163977794943222   LDRI-6
0.0005163977794943222   LDRI-7
0.0005163977794943222   LDRI-8
0.0005163977794943222   LDRI-9
9.2715899555588 ACST-1
9.2715899555588 ACST-10
9.2715899555588 ACST-2
9.2715899555588 ACST-3
9.2715899555588 ACST-4
9.2715899555588 ACST-5
9.2715899555588 ACST-6
9.2715899555588 ACST-7
9.2715899555588 ACST-8
9.2715899555588 ACST-9
~
```

Large Output