Data Intensive Computing

Report for Assignment 3

Compute the Volatility of Stocks in NASDAQ in HIVE and PIG

1. Method and Implementation

In this assignment we were given the data of 2970 stocks on NASDAQ market for 3 years from 01/01/2012 to 12/31/2014. The work happening in the specific jobs (Hive and Pig) is explained below.

Type of DataSet:

1. We have been provided with a data set of comma separated files with the monthly trading value of the stocks.
2. The data rows are in descending order of the monthly dates. For ex. the data for a particular month starts from the last traded day in the month and ends with the first traded day of the month.
3. This structure of data is being utilized in the strategy..

Sample Input data:

```
Date,Open,High,Low,Close,Volume,Adj Close
2014-12-31,50.68,50.68,50.68,50.68,000,50.55
```

HIVE

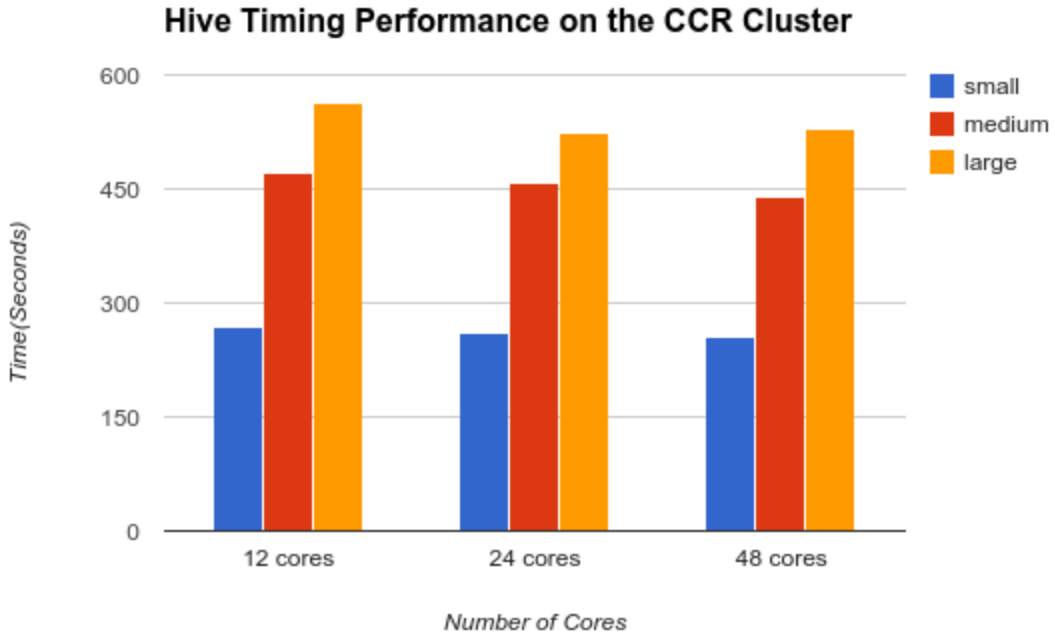
Strategy

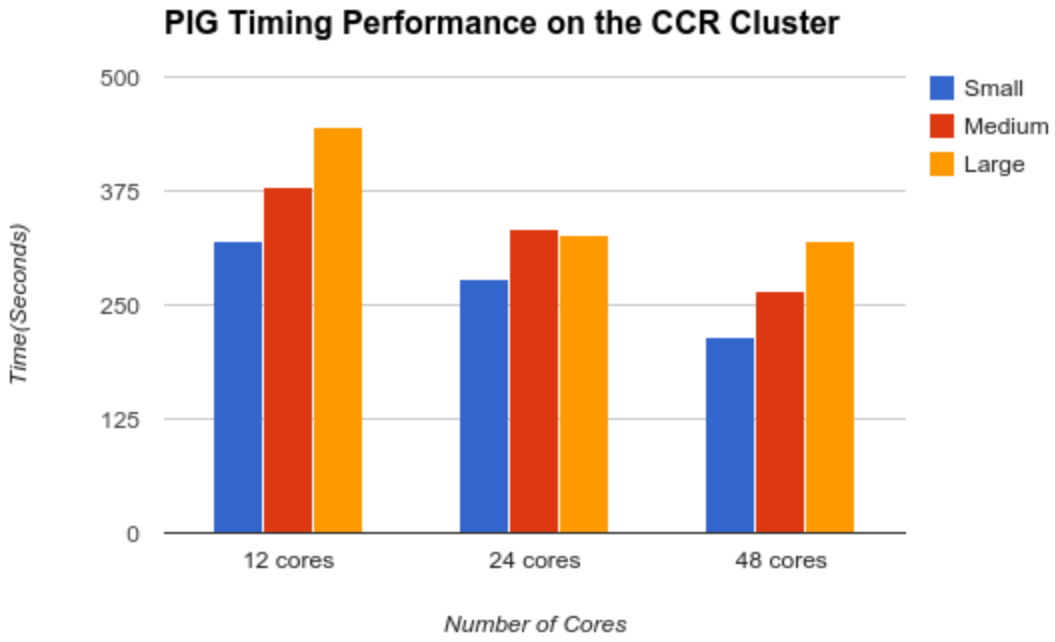
- Load data into a table and remove the header using filter.
- Filter the workable data with file name into another table storing the date as substring of year and month also.
- Finding the min and max date.
- Applying the double join with the filtered all data table to find the adjusted close price for max and min date.
- Use the unbiased standard deviation inbuilt method of stddev_samp of the hive.
- Removed the 0 values and sort the data in both order and stored in a table
- Stored the sorted data into the output files using the command in SLURM file.

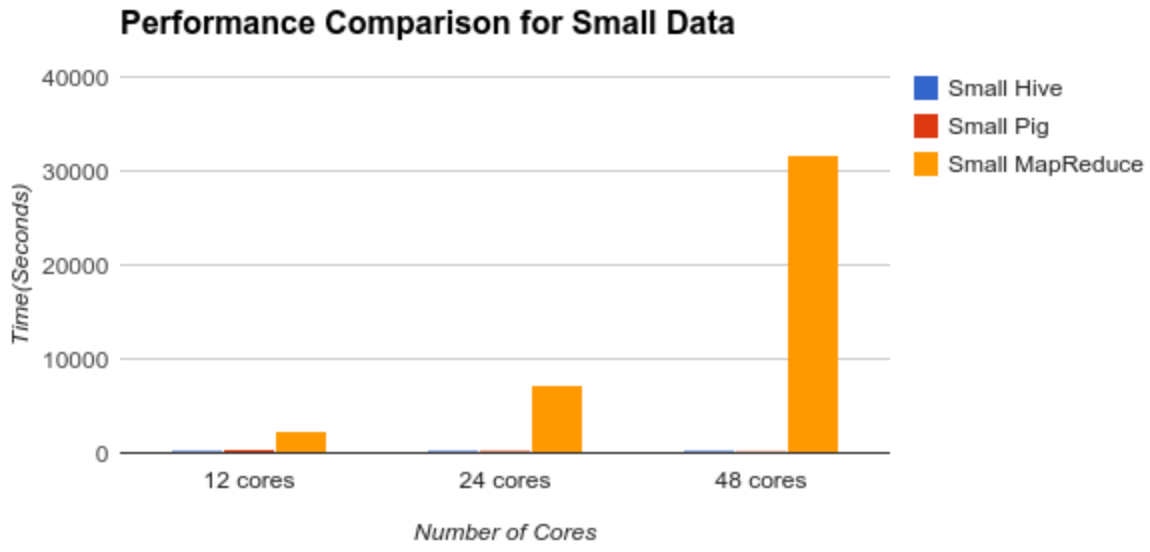
PIG

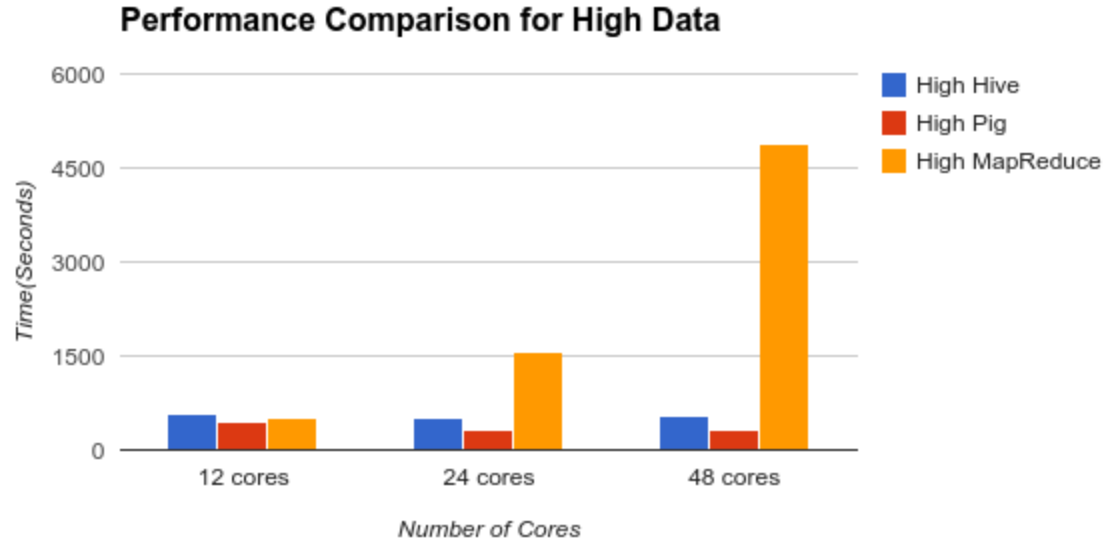
Strategy

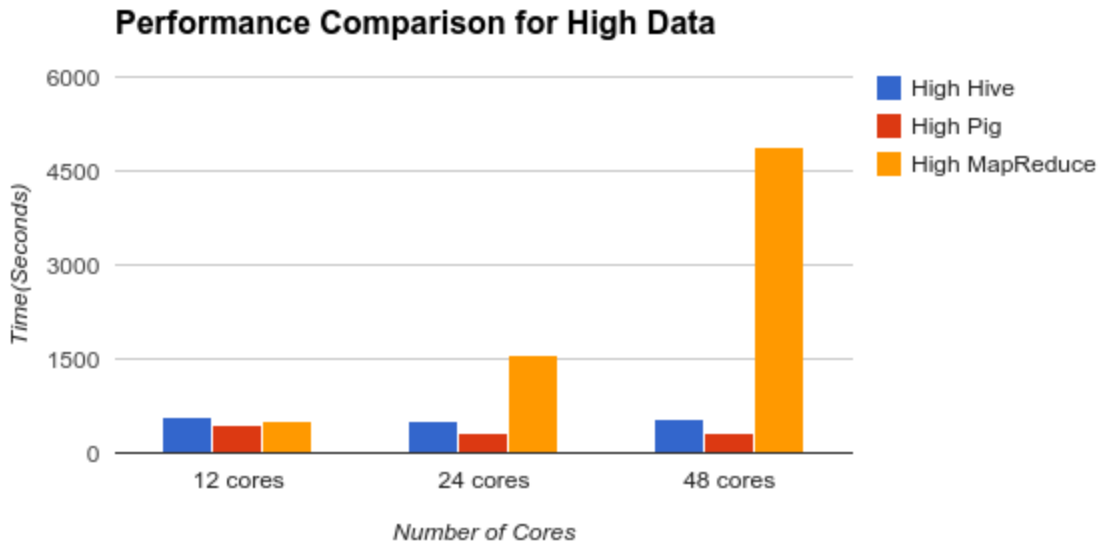
- Load the data into the pig variable.
- Filtered the data, removing the header and collected the useful data.
- With the substring of date's year and month and stock name, grouped the data into a bag of tuple.
- Using the values of tuple in the bag, found out the max and min dates of the month using the filtered data.
- Joined the variables with min and max date data with the actual filtered data on the basis of the dates, so that the remaining data will be filtered out and we can find out the prices corresponding to those dates.
- Calculated value of xi with the resultant variable of the last point.
- Checked the condition when the total month is 1 only so that those stocks could be filtered out.
- Calculated the mean value and the total months for that stock.
- Then there are multiple steps including Join, Foreach and Group statements to find the summation, product, square root and other number crunching operations.
- Then the sorting and limiting the data into the required number of rows were done.


Hive Timing Performance on the CCR Cluster
600
450
300
150
0
Time(Seconds)
12 cores
24 cores
48 cores
Number of Cores
small
medium
large


PIG Timing Performance on the CCR Cluster
500
375
250
125
0
Time(Seconds)
12 cores
24 cores
48 cores
Number of Cores
Small
Medium
Large


Performance Comparison for Small Data
40000
30000
20000
10000
0
Time(Seconds)
12 cores
24 cores
48 cores
Number of Cores
Small Hive
Small Pig
Small MapReduce


Performance Comparison for High Data
6000
4500
3000
1500
0
Time(Seconds)
12 cores
24 cores
48 cores
Number of Cores
High Hive
High Pig
High MapReduce



Experiment and Discussion

Performance on Data Scaling
As the size of the data increased, the processing time got increased. This increase was not very linear, since the records read by the Hive and Pig Mappers will gets increase and the reducer was getting more records. There are some functions in Hive and Pig which are computationally costly and running them on parallel file system will have an impact on the time performance. Some of them are Join, Group etc.

Performance on Node Scaling
Since the number of nodes got increased with the increase in the number of processors, the execution time was also reduces. This shows the parallel nature of Hadoop Distributed file system performance. Since now the reducers are doing task in parallel, hence the reduction in time was noticed.

For Pig
Note: Time in Seconds

Problem Size	Execution Time: 1 node (12 cores)	Execution Time: 1 node (24 cores)	Execution Time: 1 node (48 cores)
Small	321	380	445
Medium	279	333	327
Large	215	265	321

For Hive

Note: Time in Seconds

Problem Size	Execution Time: 1 node (12 cores)	Execution Time: 1 node (24 cores)	Execution Time: 1 node (48 cores)
Small	269	504	564
Medium	261	486	525
Large	256	457	530