A8 Spark Report

Pankaj Tripathi, Kartik Mahaley, Shakti Patro, Chen Bai

March 26, 2016

The report encompasses a detailed comparison of the two solutions, one with Hadoop MapReduce and other with Spark in terms of performance, software engineering and the graph plotted considering the code being run on AWS EMR Cluster.

Introduction:

The price of a ticket depends, in part, on the fuel consumed, figuring out which airline is cheapest requires a little bit of work. The problem statement of the assignment is to write a Spark MapReduce job that ranks carriers and plots the evolution of prices of the least expensive carrier over time.
The assignment requires taking input "-time=N" where N is a natural number representing scheduled flight minutes. For each year and each carrier, estimate the intercept and the slope of a simple linear regression using the scheduled flight time to explain average ticket prices. For a given year, compare carriers at N by computing intercept+slope*N. The least expensive carrier is the carrier with the lowest price at N for the most years. We also need to plot, for each week of the entire dataset, the median price of the least expensive carrier.

Design:

The design that we have chosen to implement the task involves three phases furnished below.
1. **Phase-1:Developing Pair RDDs**
2. **Phase-2:Modeling and Prediction**
3. **Phase-3:Finding cheapest carrier and plotting graphs**

A detailed description of these phases and design is provided in the implementation section.

Implementation:

In this section, we not only provide detailed explanation to our implementation but also provide a comparison in terms of software engineering between the Hadoop MapReduce and Spark solution.
Hadoop model: In previous model we read files from Hadoop MapReduce, cleaned it by passing it thorugh sanity test. Once the sanity test is completed we map the data with Key{Year, Carrier} and Value{Price, ActualElapsedTime}. We then in the same job find the cheapest carrier and then pass it to next MapReduce job. In next MapReduce job we simply display the median values for the cheapest carrier each year and then plot the same. Lines of Code in this model was approximately 579 LOC.

Phase-1: Developing Pair RDDs

In this phase we read the data, passed it through the sanity test and then we created Java Pair RDDs with Keys{Year, Carrier} and Values{Price, ActualElapsedTime}. We obtained a list of values using group by key.

Phase-2:Modeling and Predicition

In this phase the list obtained from previous phase for each key is used to create a model using Simple Linear Regression. The package used for modelling the Linear regression is Apache's Math Commons package. We refrained from using Spark's Mlib because creating nested RDDs is not allowed in Spark. This made it difficult for us to model for each key. The output form this phase is {Year, Carrier, Estimated Price}

Phase-3:Finding cheapest carrier and plotting graphs

In this phase we call collect as map function to the output ie. the RDDs obtained from the phase-2. This gives a Java Map which we can easily operate on. We perform some operations on this map to get the cheapest carrier. These operation include creating another map where we store for each year the cheapest carrier. This is followed by creating a TreeMap with Carriers as key and a count which is number of times the carriers were cheapest in the respective years. This helps us find the cheapest airline overall. We then plot, for each week of the entire dataset, the median price of the that carrier with the help of the same RDD created in Phase-1.

Execution:

The phases with there execution details are furnished below. Total time taken for Hadoop is 44 mins and for Spark model is 11 mins.

Spark Model:
Model Creation: Phase-1 is executed on AWS.
Master: 1
Slave: 10
Type: m3.xlarge
Time Minutes: 11

Hadoop Model: Executed on AWS.
Master: 1
Slave: 3
Type: m3.xlarge
Time Minutes: 44

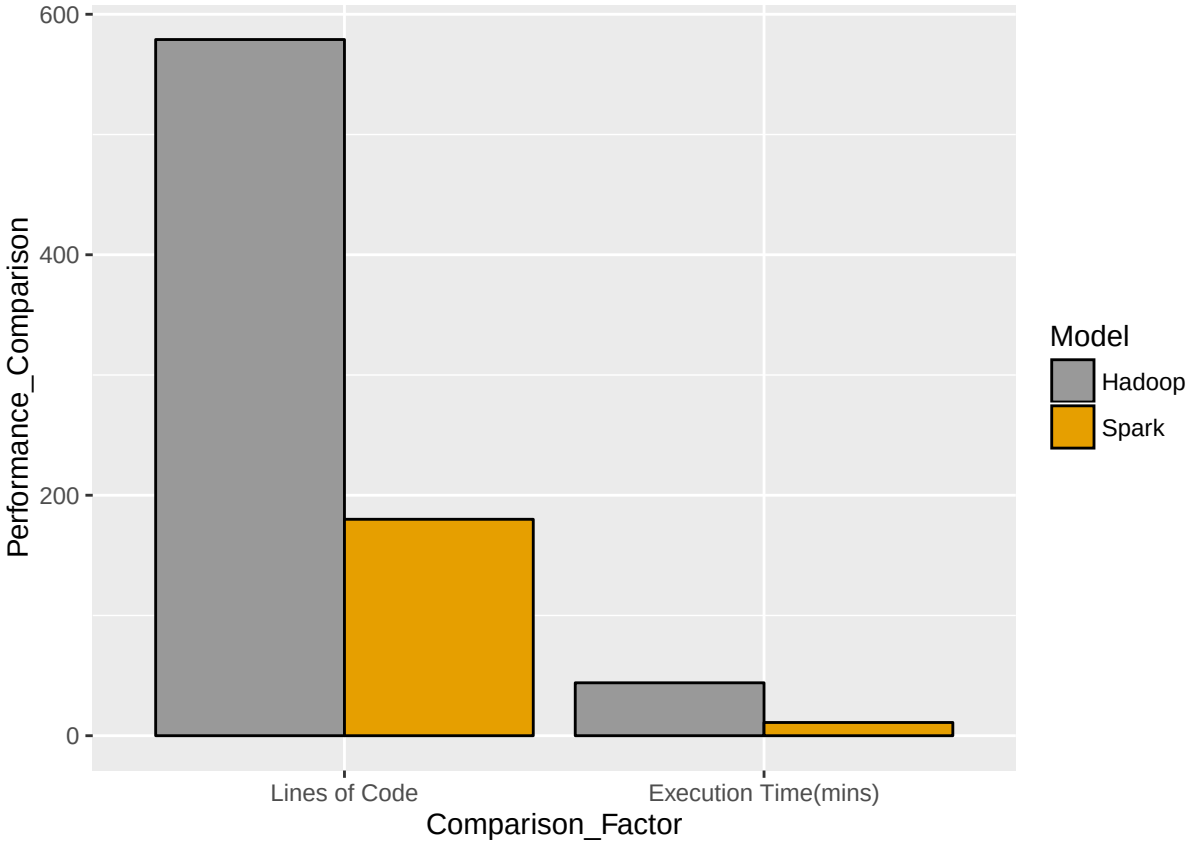
Task Distribution:

Phase-1: Shakti Patro, Kartik Mahaley
Phase-2: Shakti Patro, Chen Bai
Phase-3: Pankaj Tripathi
Packaging: Kartik Mahaley
Code Maintainence: Chen Bai, Shakti Patro
Report: Pankaj Tripathi

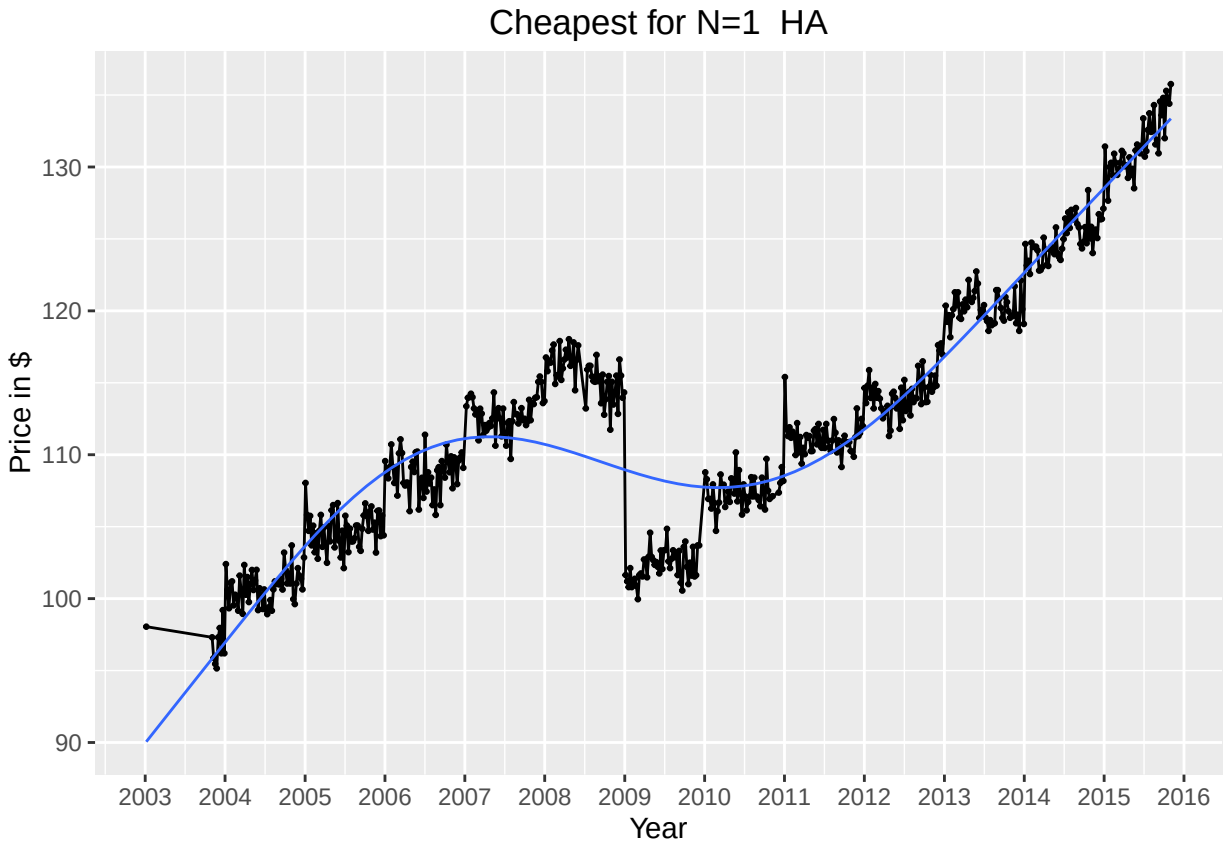
Graphs:

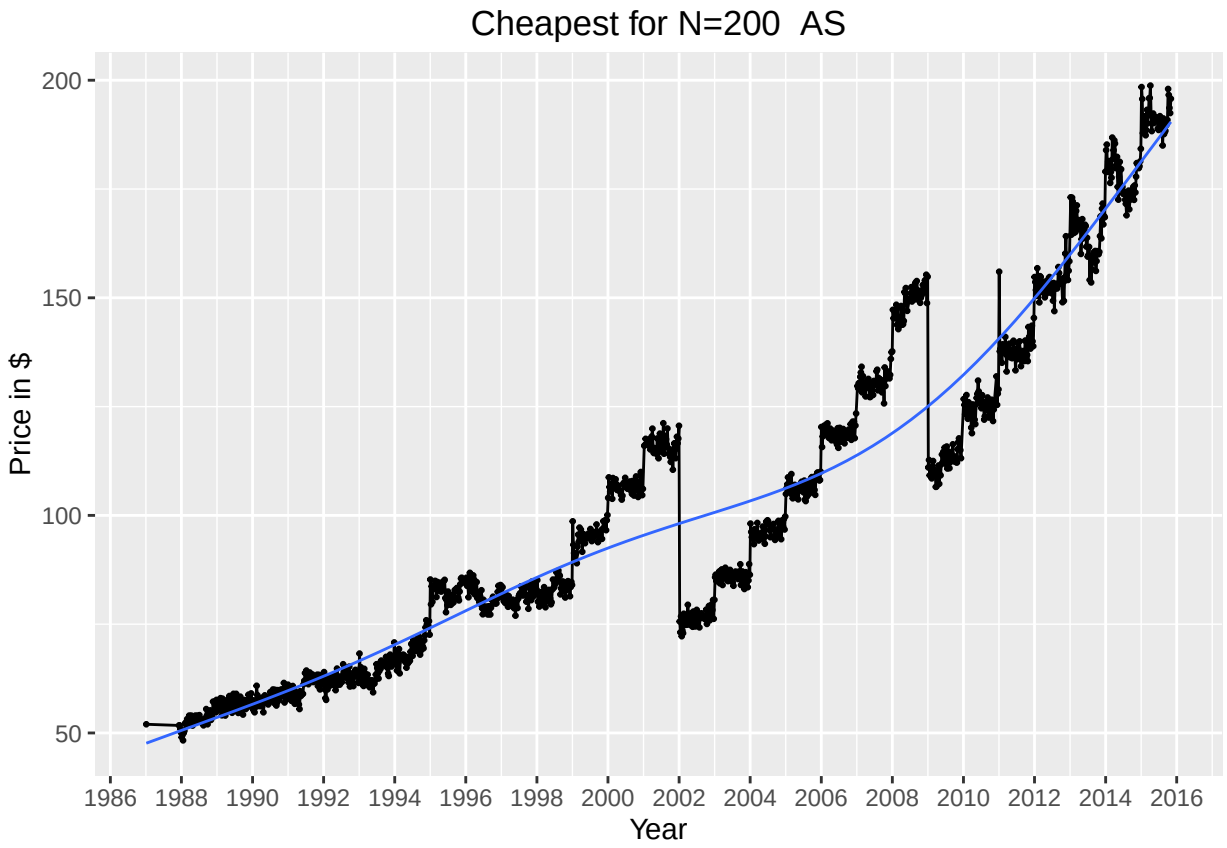
Performance Comparison:

Model	Lines of Code	Execution Time(mins)
Hadoop	579	44
Spark	180	11



Cheapest Airline Price Detail Graphs:





Observation:

Hadoop vs Spark:

Factor	Hadoop	Spark
Difficulty	Bit Difficult to program	Easy to program
Ease of Coding	Writing Hadoop Mapreduce pipelines is complex and lengthy	Spark code is compact
Execution Time	Slower	Faster
Machine Learning Libraries	Has no libraries	Has libraries for Machine learning

Graphs:

For N=1 the cheapest airline is HA. HA went active in 2003 after which it saw a subsequent increase in its price. This was followed by a fall in price from 113$ to 103$ in 2009.
For N=200 the cheapest airline is AS. AS went active in 1986 after which it went inactive. Then went active the next year after which it saw an increase in its prices. There was a drop in 2002 in the price from 122$ to 75$ and then it again saw an increase in its prices before seeing another drop in 2009 from 165$ to 130$.

Conclusion:

The modeling strategy that we followed with Spark was some what similar to that we did with the Hadoop. As shown in the graph for performance comparison, the number of lines of code in Hadoop is almost 3 times than that taken in Spark. As far as time for execution is concerned the performance with Hadoop is 5 times to that of Spark.