PC-2018/19 Hadoop and Storm sentiment analysis

Riccardo Malavolti
UniFi L.M.Ing. Informatica
riccardo.malavolti@stud.unifi.it

Davide Nesi
UniFi L.M.Ing. Informatica
davide.nesi@stud.unifi.it

Abstract

We present our study on two very different approaches to distributed computing and analysis. In this paper we will study the two different solutions proposed by Apache Hadoop and Apache Storm. Both solutions will be tested using a benchmark problem. For our case study we choose the sentiment analysis of Tweets. In the end we will discuss application scenarios and main aspects of both technologies.

Future Distribution Permission

The author(s) of this report give permission for this document to be distributed to Unifi-affiliated students taking future courses.

1. Introduction

Processing large amount of data can be sometime over the possibility of a single computer or server. It is not always possible to scale vertically the resources allocated to a computational unit. Very often old hardware just sit unused while it could actually increment performance for complex computational task working in parallel with newer hardware. For this and many others reasons scaling a system horizontally may be the best option depending on the type of task. Both Hadoop and Storm are system that allow horizontal scaling. In the next sections we will discuss about general concepts of batch processing, real time processing and sentiment analysis.

1.1. Batch processing

Batch processing is a two phases process. In a first place all data are collected and stored in one or multiple data sets. In a second phase data are processed. There is in this process a very defined separation between data acquisition and data processing. Batch processing usually is something a user requests and waits for results. The whole idea behind batch processing is to process very large amount of stored data.

1.2. Realtime processing

Realtime processing is a process where data flowing from a source are immediately processed. Usually data comes in stream or small batches and they are processed as soon as they arrive. This type of data processing is very convenient when it is needed to see the progress of a data analysis in time while data are still coming. Doing so it is possible to see the evolution of a data while it is being received and processed.

1.3. Sentiment analysis

Sentiment analysis is the process of analyzing and categorizing a set of information (usually text) coming from human actions and thoughts. Sentiment analysis usually categorize the text author's sentiment toward a specific topic. Classical example of sentiment analysis comes from the analysis performed on tweets. This type of analysis can be very helpful to study the relationship between large mass of people and a certain object (a company, a product, a person, etc...).

2. Used technologies

In this study we considered two widely adopted technology for batch and realtime processing. The first is Apache Hadoop and the second one is Apache Storm. Hadoop is widely used in both research and business application and it is an older

technology. Storm is a new technology but it surged as one of the most important realtime data analysis tool. In our study we used LingPipe to perform sentiment analysis and classify tweets.

2.1. Hadoop

Hadoop is a library framework build to allow distributed processing of big amount of data, born in 2003. Hadoop is mostly writte in Java. Hadoop works on large static data sets. Hadoop uses big clusters of computers in order to solve a given problem faster. Hadoop itself is designed to handle all the computer in the clusters managing resources and errors. Hadoop organize the cluster by giving computational units a role. In a cluster there is one master node and multiple workers node. The master node consists of a DataNode, NameNode, JobTraker and TaskTraker. All this components allow the master node to manage the cluster and keep all workers busy on the job. Workers have DataNode and TaskTraker. Hadoop Distributed File System (HDFS) is a distributed file system that allows all the cluster to store data on machines and have it always available. Hadoop's MapReduce is an implementation of the MapReduce programming model. In this programming model there are mainly two components: the mapper and the reducer. The mapper orders or filters data, and the reduce perform some sort of aggregation of the data. This particular programming model allows Hadoop a very high level of scalability.

2.2. Storm

Storm is a distributed realtime computation system. Storm works on stream of data performing analysis in real time. Storm is actually Hadoop for realtime analysis. The inner working mechanism of Storm is indeed different from Hadoop. In Storm a cluster execute a topology, not a series of jobs. In Storm data is organized as tuple, a generic ordered list of elements. A topology is a graph of computation. A topology starts with a spout where data are received and emitted to the rest of the topology. Bolts are the node in the topology where data is processed with simple processing logic. Bolts can be linked together forming a full topology. In Storm's clusters there are two types of nodes: master nodes and worker nodes. Master nodes execute a scheduling component called Nimbus which assign tasks to worker while monitoring them. Worker nodes runs a Supervisor to manage topology executions assigned by Nimbus.

2.3. LingPipe

LingPipe is a tool for processing text. It can perform a huge variety of tasks on text extracting relevant info from raw text. LingPipe can be used to classify the topic of a text, clustering, sentence detection, sentiment analysis and much more.

2.4. Cassandra

Cassandra is a database that makes of scalability and high availability one of the point of strength. Being often associated to bot Storm and Hadoop it was a natural choice for us.

3. Implementation

For our implementations we decide to follow an historical path going from Hadoop to Storm. Doing so we had the possibility to appreciate changes made in distributed parallel computing in time. Both Hadoop and Storm are tested using the sentiment analysis problem. To determine a text sentiment we used LingPipe after have training it on a data set of already classified tweets. In the next few subsection we will discuss the implementation choices made in our study.

3.1. Feeding

In order to allow both Hadoop and Storm to perform analysis on a given data set we decided to use a feeding script. The feeding script draws from a large data set of stored tweet and distributes tweets to both Hadoop and Storm. Given the big difference in the inner workings of Hadoop and Storm the feeding procedure is different for the two framework. The feeding script

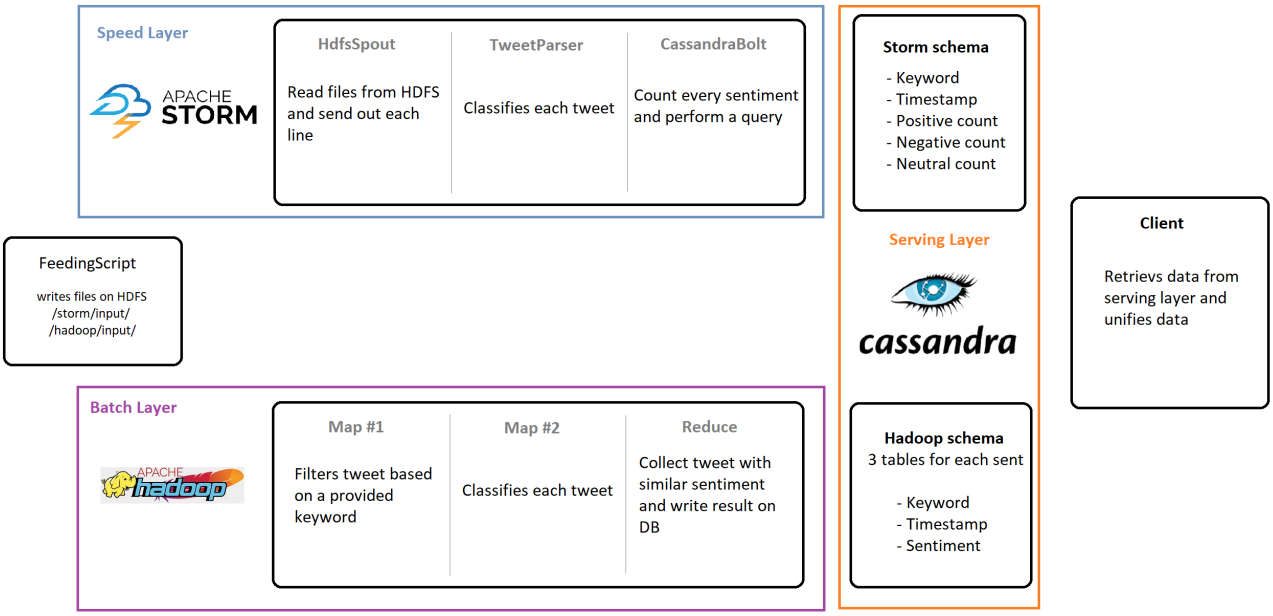

Figure 1. Architecture

feeds Hadoop splitting the whole data set in different files. There is a size limit to the single Hadoop file of 10 Mb. Feeding Storm is a process that sends once in a while a tweet from the database to a Storm available file. Storm available files are limited in size to 1Mb.

3.2. LingPipe

We used LingPipe to classify the tweets' sentiment in positive and negative. Doing so requested a training phase for LingPipe. The training was performed using a small data set of annotated tweets. After the training process the classifier we trained was ready to just classify tweets. For our study accuracy in classification was not the main concern.

3.3. Hadoop

In this phase we decided to implement a Map-MapReduce solution. We decided to go this way to allow the final user to be able to select a topic on which perform the tweets analysis. Our firs mapper is in fact a filter passing to the further mapper only tweets that contains a specific query string. The second mapper is the element where the real sentiment analysis is performed. In this mapper all tweets filtered and passed on by the first mapper are analyzed and classified, then passed to the mapper. In the final step of our Hadoop implementation we used a mapper to collect and count all tweets having the same sentiment classification. In the end all data from the reduce are sent to a database.

3.4. Storm

The topology developed using Storm is composed by a spout and two bolts. The initial node of the topology is just a text spout collecting data from files and passing them to the rest of the topology. The initial spout is connected to the main processing bolt. This bolt process each tweet giving to the last bolt all the sentiment analysis results for every tweet. The last bolt connects to Cassandra and stores all the information inside Cassandra itself. Queries can be performed and will be executed on this last bolt.

3.5. Cassandra

Cassandra is the database we used in this study. Its simple integration with Storm and Hadoop allowed us to use it seamlessly with the framework used. Cassandra has two different schemas: one

for Hadoop and one for Storm. We decided to with this approach to better underline the characteristics of both frameworks. Storm schema has a timestamp field, used to mark down the data processing flow.

4. Tests and results

The first tests we made were conducted on local machines without a real cluster. Then we decided to test the system inside a real cluster using Google Cloud Platform. The computational power we had to try the system we produced is risible confronted with a real scale architecture but it can provide a good measure on how even with a very small cluster of machines with limited computational capability the speed of computation is something very hard to achieve with a single computational unit.

5. Conclusions

In the end we can conclude how both Hadoop and Storm provide very interesting framework for distributed data processing. In this study we experimented with both framework successfully and we managed to implement a sentiment analysis system based on LingPipe sentiment analysis. The system we developed shows how different requirements can lead to adopt even only one of the frameworks we studied. Hadoop and Storm have a strong polarization over batch and stream processing respectively and this is the main point where to take decision on the system to use.