Click Stream Data Analysis Using Hadoop

Krishna Chand Reddy Gaddam  
*Governors State University*

Sivakrishna Thumati  
*Governors State University*

Follow this and additional works at: [http://opus.govst.edu/capstones](http://opus.govst.edu/capstones)

Part of the [Systems Architecture Commons](https://opus.govst.edu/capstones/101)

**Recommended Citation**

[http://opus.govst.edu/capstones/101](http://opus.govst.edu/capstones/101)

For more information about the academic degree, extended learning, and certificate programs of Governors State University, go to [http://www.govst.edu/Academics/Degree_Programs_and_Certifications/](http://www.govst.edu/Academics/Degree_Programs_and_Certifications/)

Visit the [Governors State Computer Science Department](http://www.govst.edu/Academics/Degree_Programs_and_Certifications/)

This Project Summary is brought to you for free and open access by the Student Capstone Projects at OPUS Open Portal to University Scholarship. It has been accepted for inclusion in All Capstone Projects by an authorized administrator of OPUS Open Portal to University Scholarship. For more information, please contact opus@govst.edu.
Abstract

The objective of this project is to collect Click Stream data of USA Government websites which is high in volume and velocity, and store it for analysis in a cost effective manner for enhanced insight and decision making. I expect to learn how to process this data in an engineer’s way. I have plenty of tools in my hand like map reduce, pig, streaming and many more. But for a given business case it is very important to know which tools should be used to achieve the objective. In brief this is what I expect to learn.

The Hadoop-ecosystem, State-of-the-art in Big Data age is perfectly suitable for click stream data analysis. To achieve the objective mentioned, it is very much necessary to have scalable systems at low cost which can operate at great speeds and bring out wonderful insights. Perfect answer for this is Hadoop.

Keywords: Hadoop, Click Stream, Pig, Python, Json, Mapper, Reducer, Namenode, Datanode and HDFS.
## Table of Contents

1) Introduction  
2) What is Hadoop?  
3) Apache Pig  
4) How to execute the Project?  
5) Commands for Running the Project  
6) Software Requirements  
7) Results  
8) References
1. Introduction

Click Stream are records of users’ interaction with a website or other computer application. Each row of the Click Stream contains a timestamp and an indication of what the user did. Every click or other action is logged—hence the term “Click Stream”. In some circumstances, what the website does is also logged. This is useful when the website does different things for different users, such as post recommendations.

I have developed a Big Data approach to Click Stream analysis that allows producing aggregates for reporting as well as session statistics like User_Agent, Country_Code, Know_User, Encoding_User_Login, Referring_URL, Timestamp, GeoRegion, Geo_city_name and Time Zone. A typical approach is to load the data into a Hadoop HDFS.

2. What is Apache Hadoop?

The processing of large data sets across clusters of commodity servers for Apache Hadoop open-source software project that enables the distribution. It is a very high degree of fault tolerance, designed to scale up from a single server to thousands of machines. Rather than relying on high-end hardware, the resiliency of these groups to recognize and manage failures at the application layer is the ability of the software.

Economics and large-scale computing will change the dynamics of Hadoop. Its effect can be boiled down to four main features (Scalable, Flexible, Cost-Effective, Fault tolerant).

The project includes these modules:

- **Hadoop Common**: The most common utilities that support the other Hadoop modules.

- **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data.

- **Hadoop Map Reduce**: A YARN-based system for parallel processing of large data sets.
3. Apache Pig

The pig was initially developed at Yahoo! To allow people using Apache Hadoop to focus more on analyzing large data sets and spend less time having to write mapper and reducer programs. Like actual pigs, who eat almost anything, the Pig programming language is designed to handle any kind of data—hence the name!

Pig consists of two components: the first language itself, which is called piglatin (yes, people of different naming projects tend to have a sense Associate Del Hadoop convention naming their mood), and the second is an environment execution, where pig Latin runs programs. Think about the relationship between a Java Virtual Machine (JVM) and a Java application. In this section, we’ll just refer to the entity as a whole Pig.

- **Ease of Programming:** It is trivial to achieve parallel execution of tasks simple data analysis, “embarrassingly parallel”. Complex tasks transformation composed of multiple interrelated data is explicitly coded as sequences of data flow, making them easier to write, understand and maintain.
- **Optimization and opportunities:** The way in which tasks are coded allows the system to optimize performance automatically, allowing the user to focus on the semantics rather than efficiency.
- **Extensibility:** Users can create their own functions to do special-purpose processing.
4. How to execute the Project?

Step by step explanation of the method I have used:

**Step 1:**

In the below link, click stream data for a period of 1 year is archived in numerous small files and provided. [http://www.usa.gov/About/developer-resources/1usagov.shtml](http://www.usa.gov/About/developer-resources/1usagov.shtml) downloading the data from the link provided is not that simple as general file download looks. There are several thousands of files, which has click stream data for a given day. To download these files manually, would definitely take a week or more. So the best way to answer this problem is to open a connection with the website server, get all the URL’s present and start downloading the data. There are many programming languages that can do this task. But I feel Java is the best as there are many already built libraries which can do this job. So we choose Java platform to get this job done and library used is Jsoup. Using this program, first we got all the links present in the given website and filtered them to get the links which have these archived data. Now iterated over all the links and downloaded files to local disk. It took one entire day to download data from all the archived data links provided in the website.

**Step 2:**

As there are numerous files, we unzipped all the files on Linux platform and then linked together in a sequence in to a single file.

**Step 3:**

Now we have to load this data on to HDFS.

**Step 4:**

Data provided is in JSON Format. And In this project our interest is to find the top 10 websites per country and per month. So the fields we require would be only URL, month and country. So pre-processing this data and extract the required fields is pretty much important as this would decrease the processing time and storage on Hadoop. Since this data is already loaded on to Hadoop, now we can use the map reduce model to get the required fields. And what we foresee is there is no need of reducer for this data pre-processing operation. So it is enough to have identity reducer to store the output onto the HDFS. For this operation, we could have used Pig inbuilt JsonReader function to extract the required fields, but this function is provided from pig 1.0 version, but we have an old version, so only we could not use pig for this operation. But using pig inbuilt function would have saved us a lot of time.
Another option we have is to write a UDF and register it, but in general on Hadoop Jython should be installed to register UDF, since we do not have Jython and installing that would might unstabilize the existing environment. So we did not go for that. Java can be used for this, but Java is not suitable for the data pre-processing as there would be separate libraries built for json reading, which should be added to classpath and it adds more complexity to the existing map reduce model.

Finally, we are left with Hadoop streaming. In this case we can use any language which can read from standard input and write to standard output. So we have options like python, Perl, c, c++ and many more, but we chose python, as by default it is present in Linux environment. And python is very good for data processing. In mapper phase, python will read each line from standard input, decode the JSON format and store it in a dictionary which is very similar to map (key-value) in Java. Now required fields are URL, country and month, getting the URL and country is simple. But getting month from time stamp is a bit complex, in which we need to decode it to human readable format and then store month in a separate variable.

Now emit these values from the mapper. After this they undergo shuffle and sort phase, followed by reducer which writes the output to HDFS, here we use an identity reducer since their aggregation is required. This step is the most important step which discards all the unnecessary data and only picks the required fields.

**Step 5:**

Data now has only three tab separated fields. Now process this data and find:

1. The top 10 most popular sites in terms of clicks
2. The top-10 most popular sites for each country, and
3. The top-10 most popular sites for each month.

To find the top 10 most popular sites in terms of clicks, using Java will take at least 200 lines of code, and need many hours of testing on Hadoop platform. To avoid all these it is better to use high level interfaces built on Hadoop. We have two options Hive and Pig. We chose pig, which is procedural and has an option of storing results and various points. Following six lines of code will give the top 10 most popular sites in terms of clicks.

```
Clicks = LOAD 'MiniProject7/output14/part-00000' USING PigStorage(' ') AS (url: char array, country: char array, month: int);
load data with fields as url, country and month.
grpd = GROUP clicks BY url;
now group them by URL.
```
cnt = for each grpd generate group, COUNT(clicks) as URLCount;

Find out the count in each group.

dorder = order cnt by url desc;

Arrange them in descending order.

top10 = limit dorder 10;

Consider only top ten values.
STORE top10 INTO 'top10Urls';
Store the top ten URL.

Now to find the top ten URL in each country and per each month, pig alone cannot achieve this. So for grouping data pig is used.

grpCountryUrl = GROUP clicks BY (country, url);
Group the data by country and URL.

countryUrlCount = FOREACH grpCountryUrl GENERATE
FLATTEN (group) AS (country, url),
COUNT (clicks) AS Country_url_count;
Now count the url in each country.

STORE countryUrlCount INTO 'countryUrlCount';
Store this data back to Hadoop.

After Data is processed into {Country, URL, and Count} Format, the rest of the work in finding the top URLs per Country or Month is done through map-reduce. We have created a project named Project7 in eclipse and wrote the following classes for processing the data into top 10 URL per Country or top 10 URL per Month,

- CountryURLCount
- CountryURLMapper
- CountryURLReducer
- MonthURLCount
- MonthURLMapper
- MonthURLReducer
- URLCount

CountryURLCount is the main class to be run for map-reduce job of finding top10 URLs per country. In it Mapper is set to CountryURLMapper and Reducer is set to CountryURLReducer. Job is configured. CountryURLMapper is the mapper class about the job of finding top10 URLs per Country. In it, we read data in the format {Country, URL, Count} as line and split the line with a tab as the separator. Mapper casts the data into the Country as Mapper Output Key and URLCount as Mapper output value.

CountryURLReducer is the reducer class on the job of finding top10URLs per Country. The input key is the country which comes as a sorted key value from the shuffle and sort mechanism of Map-Reduce and input value is the iterator containing URLCount Objects. In reducer we then collect all the URLs and their counts per country in Reducer. We then perform sorting of the data and then collect top 10 URLs per Country. We write the result in context with key as Country and value as URL into a separate file. The output folder for this Reducer is countryUrlOutput. In that folder one can find the results for top10 URLs per Country.

MonthURLCount is the main class to be run for map-reduce job of finding top10 URLs per month. In it Mapper is set to MonthURLMapper and Reducer is set to MonthURLReducer. Job is configured. MonthURLMapper is the mapper class about the job of finding top10 URLs per Month. In it, we read data in the format {Month, URL, Count} as line and split the line with a tab as the separator. Mapper casts the data into the month as a Mapper output key and URLCount as Mapper output value.

MonthURLReducer is the reducer class on the job of finding top10URLs per Month. The input key is the Month which comes as a sorted key value from the shuffle and sort mechanism of Map-Reduce and input value is the iterator containing URLCount Objects. In reducer we then collect all the URLs and their counts per Month in Reducer. We then perform sorting of the data and then collect top 10 URLs per Month. We write the result in context with key as Month and value as URL into a separate file. The output folder for this Reducer is MonthUrlOutput. In that folder one can find the results for top10 URLs per Month.

URLCount is a custom class to store the information of URL and its count. This is used as output value in both CountryURLMapper and MonthURLMapper. As a mapper output value it must implement Writable Interface. Since we are doing sorting of objects of URLCount, URLCount must also implement Comparable Interface. It is required that all Writable implementations must have a default constructor so that the Map Reduce framework can instantiate them, then populate their fields by calling readFields (). We must also override methods like hashcode (), equals () and ToString from Java. Lang. Object
6. Commands for Running the Project

Command for copying data to HDFS: hadoop fs -put usagov.json usagov.json

Command for streaming the data and convert to json format:


Command for running the pig:

For Top 10 URLs:

```pig
 clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS (url:chararray, country, :chararray, month:int);
grpd = GROUP clicks BY url;
cnt = foreach grpd generate group, COUNT(clicks) as urlcount;
dorder = order cnt by urlcount desc;
top10 = limit dorder 10;
STORE top10 INTO 'top10Urls';
```

For Month list: pig -x mapred month.pig
```pig
 clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS (url:chararray, country :chararray, month:int);
cntMonthlyUrl = GROUP clicks BY (month, url);
MonthlyUrlCount = FOREACH cntMonthlyUrl GENERATE FLATTEN (group) AS (month, URL), COUNT (clicks) AS month_url_count;
STORE MonthlyUrlCount INTO 'MonthlyUrlCount';
```

For Country: pig -x mapred Country.pig
```pig
 clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS (url:chararray, country :chararray, month:int);
grpCountryUrl = GROUP clicks BY (country, url);
countryUrlCount = FOREACH grpCountryUrl GENERATE FLATTEN (group) AS (country, url), COUNT (clicks) AS Country_url_count;
STORE countryUrlCount INTO 'countryUrlCount';
```
7. Software requirements

Operating System : Cloudera
Technology : Hadoop, Pig, Python
Web Technologies : Json

8. Results for Project

Cluster Summary of Hadoop File System:
Downloading Data from http://www.usa.gov/About/developer-resources/1usagov.shtml

The best way to get this data from this URL is open a connection with website server get all the URL’s present and start downloading data by using Java platform. Data downloaded in zip format and then extracted and Linux platform gunzip command. So data looks in below format of the list of files.
After Running the below all scripts the Namenode format, shown as below:
Top 10 URLs in terms of Number of Clicks

The pig code to get Top 10 URL’s:

```pig
clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS (url:chararray,country:chararray,month:int);
grpd = GROUP clicks BY url;
cnt = foreach grpd generate group, COUNT(clicks) as urlcount;
dorder = order cnt by urlcount desc;
top10 = limit dorder 10;
STORE top10 INTO 'top10Urls';
```

After running above script the output shown as in below for top 10 URL’s clicked mostly.

```
http://www.sasa.gov 19718
http://api.4j.iw.gov 138450
http://www.sasa.gov 64984
http://www.sasa.gov 23826
http://www.irc.cop.roax.gov 23898
http://portal.hud.gov 15618
http://answers.sas.gov 10060
http://www.whitehouse.gov 8764
http://op睢.xasa.gov 8325
http://op睢.xasa.gov 7902
```
Results - TOP 10 URLS per Country

The Pig code for Top 10 URL’s for each country:

```pig
pig -x mapred Country.pig
clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS (url:chararray,country:chararray,month:int);
grpCountryUrl = GROUP clicks BY (country,url);
countryUrlCount = FOREACH grpCountryUrl GENERATE FLATTEN(group) AS (country,url),COUNT(clicks) AS Country_url_count;
STORE countryUrlCount INTO 'countryUrlCount';
```

After running above script the output shown as in below for top 10 URL’s for each country.
Results - TOP URLS per Month

Pig code for Top URL’s per Month:

```
pig -x mapred month.pig
clicks = LOAD 'LargeFile/part-00000' USING PigStorage(' ') AS
?url:chararray,country:chararray,month:int);
cntMonthlyUrl = GROUP clicks BY (month, url);
MonthlyUrlCount = FOREACH cntMonthlyUrl GENERATE FLATTEN(group) AS (month,
?url),COUNT(clicks) AS month_url_count;
STORE MonthlyUrlCount INTO 'MonthlyUrlCount';
```

After running above script the output shown as in below for top URL’s per month.
9. Reference

- http://www.usa.gov/About/developer-resources/1usagov.shtml