Twitter Streaming API Using Apache Spark in Big Data Analytics

Sakshi

M.Tech Scholar, Department of Computer Science & Applications, Kurukshetra University, Kurukshetra-136119

Email: sakshi28sept@gmail.com

Shuchita Upadhyaya

Professor, Department of Computer Science & Applications, Kurukshetra University, Kurukshetra-136119

Email: suchita_bhasin@yahoo.com

----------**ABSTRACT**----------

Analysing big data is very common requirement of today; all such requirements become difficult to access when analyzing bulk of data source such as social networking sites which are having a lot of information on day basis. Twitter is the micro blogging and popular site providing social networking services today. To analyze bigger amount of tweets from twitter to get different patterns and extract relevant information is a big challenge. Apache spark is the platform that is used to analyze big data efficiently. In order to find out the context used, we make out relevant patterns from big data by using Twitter Streaming API. This paper explores the concept of twitter streaming API to extract relevant information from enormous amount of data from twitter tweets.

Keywords – Apache Spark, Big Data, Resilient distributed Dataset, Twitter Streaming API, Tweets.

1. INTRODUCTION

Apache spark is a fast, general purpose and distributed processing platform that uses distributed memory abstraction to process large volume of data efficiently. Apache spark is flexible and scalable computing system consists of powerful API and higher order tools that are compatible with hadoop. Basically it has privilege to run through yarn, or standalone cluster mode. Spark has the fast computing memory that caches the data to be processed and gives the output 100 times faster than hadoop platform. Spark performs scheduling tasks, performs multithreading, and manages executor states. Apache spark platform is capable of utilizing the existing components that exits in hadoop eco-system for handling data in large volume. Hadoop eco-system has made spark to grow exponentially in field of data handling and analytics. Apache Spark is built to perform distributed computing, batch processing, interactive queries, streaming process and machine learning. Spark can store data in HDFS, Cassandra, Hbase, Hive, and any other Hadoop input format. Apache spark runs on Spark Web UI. [1, 2, 3]

Apache spark is based on feature RDD (resilient distributed datasets) that is read-only collection of objects that is partitioned across a set of jobs. RDD helps in recovery of lost partition, which means RDD can be reconstructed if node fails. Elements of RDD do not exist in physical storage although RDD contains enough information to process the data in reliable mode on every node. [4, 5, 6]

Apache spark based on scala, java, and python API libraries. Spark requires no changes in scala or compiler plug-ins to run it. Python API uses standard Cpython for implementation and can call the existing C libraries for

python. There is no need of hadoop platform to run apache spark if running on single mode. [3] Working with a cluster there is need of some form of shared file system that can be mounted on each node, for example NFS.

All jobs in apache spark consist of series of operators and runs on various datasets that are prebuilt on some data. Various prebuilt operators in a task are used to construct a direct acyclic graph that is possible to optimize by combining and rearranging operators. For Example if have to submit a job carried out by map and filter operation, then optimizer will rearrange the order of these operators. [7, 8] Filtering process reduce the number of records that are processed by map operation. Spark is small code based platform in which system is divided into various layers with some pre-assigned responsibilities. All layers and code are independent of each other. Apache spark processes the data 100 times faster than any other data analytics platform. [9, 10]

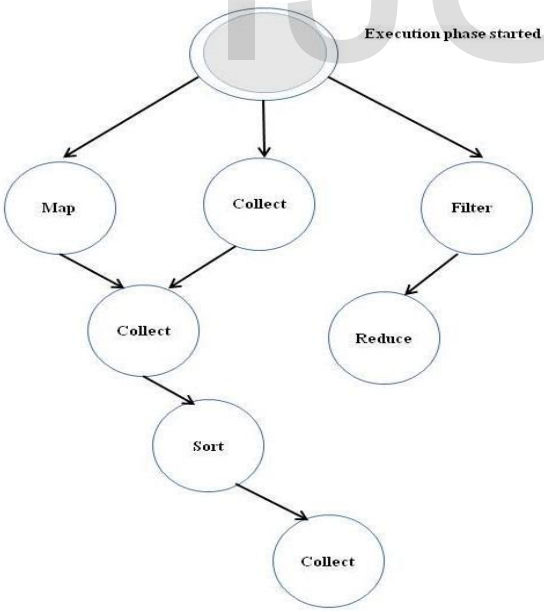

Figure 1: Directed acyclic graph on apache spark map and filter operations

Apache spark uses directed acyclic graph execution engine that is used to generate a directed acyclic graph when the user runs an action on Resilient Distributed Dataset (RDD) that consider all the transformation dependencies. [11]

2. APACHE SPARK STREAMING

Spark streaming in apache spark uses the same abstraction for real time event streaming. Spark is used to offer data streaming that groups incoming data into micro-batches for processing by the apache spark platform. This can be completed through a discretized stream that is used for representation of set of RDD (Resilient distributed datasets). This made the batch processing as available for streaming as well it allows MLib and GraphX. MLib and GraphX are used to operate with steaming data. [12] New Data Frame API is used with query optimizer that has equal performance for Scala, Java, Python, and R. There is performance increase of 10-100x the in Hive, due to in-memory caching of RDDs & better spark abstractions. [13]

Spark streaming uses micro batch processing execution model, which cannot have any effect on the applications because batches are short as 0.5 seconds created by RDD. In most of the applications streaming over big data is done over a large window and the latency for getting data gets higher (for example sensors reading data in every 10 seconds). The benefit of using micro model batch processing is that it enables exact once semantics that means system can recover all intermediate state and give results on failure. [14]

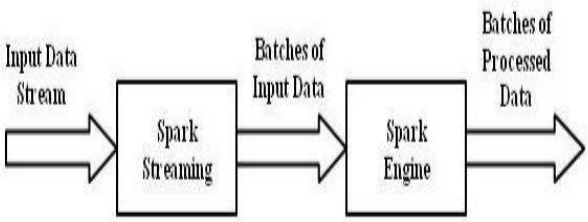

Figure 2: Spark streaming based on discretized stream

Spark Stream provides the ability to continuously compute transformations on data, For Example map join, reduce etc. Once we receive data from various

sources then it creates a batch of input stream and store. [15]

Various features of apache spark streaming over map reduce:

- Apache spark supports both check pointing based recovery and lineage based recovery. Fault is very common to come along, lineage based recovery is faster than check pointing based recovery because replicating nodes in the network are much slower for high-throughput data flow.

- Spark uses thread pool execution of tasks that enables to schedule the processes faster as in milliseconds and map reduce takes sometimes minutes in cluster environment. [16]

Layers in apache spark streaming are defined below that are used for scheduling and dispatching of task in work environment.

2.1 The topmost layer in spark environment is the **interpreter**; Spark uses a Scala interpreter, with some modifications. In the compiler graph to manage the spark-shell of apache spark.

2.2 User creates code in the command line that creates an operator graph by creating RDD's and operators on it.

2.3 User runs an action like collect, then the Graph is submitted to a direct acyclic graph scheduler then **DAG scheduler**s divides the operator graph into map and reduce stages.

2.4 DAG scheduler is used to pipeline the operators to optimize the graph in a single stage to give a result of set of stages by DAG scheduler.

2.5 All the stages processed by DAG scheduler are passed to the **Task Scheduler**. Task scheduler is used for launching the task by cluster manager like spark standalone/yarn/mesos/daemon. Task scheduler does not have any dependency among stages that leads to no replication among nodes.

2.6 At last **Worker** executes the task on the slave, worker have only knowledge about the code that has been passed to it. [17, 11]

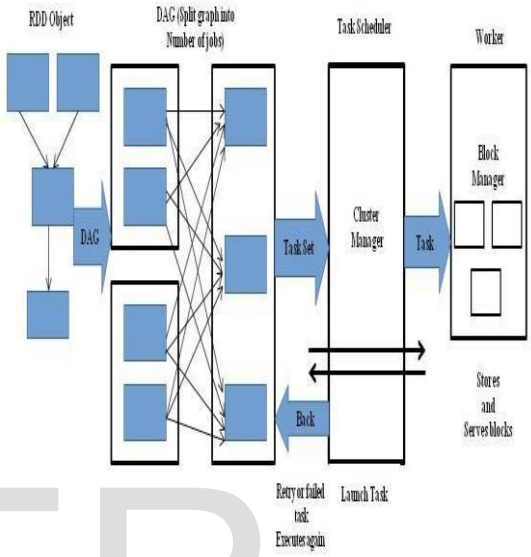

Figure 3: Apache Spark in-built Layers for processing of Data

Spark Streaming has two types of operator:

- Transformation operator: It is used to map a data stream to the one or more root streams. It can share data both independent of each interval or share data across intervals.

- Output operator: It is an action operator that allows writing data to external systems for example: print or save data streaming. [11]

3. TWITTER STREAMING API

In twitter, tweets are generally open for public known as the public timeline of twitter. Twitter as an organization itself processes approximately 10k tweets per second, they analyze the data with fast rate to ensure that each and every tweet get processed followed by dependency policy and restricted words are used by twitter. Since twitter uses many kinds of API's for general users and we can obtain tweets from there. Streaming API was

released in 2010 that allows real time access to the users. Streaming API only provides 10% tweets from general tweets that are randomly selected. [18]

API's are used for formation of largest integrated standard libraries for big data that can lead to interesting and relevant design patterns and decision to enable efficient composition of workloads. [7]

The streaming API is used to provide the push deliveries of tweets and other events to be happening on twitter. It is used for real time applications or low latency applications. [13]

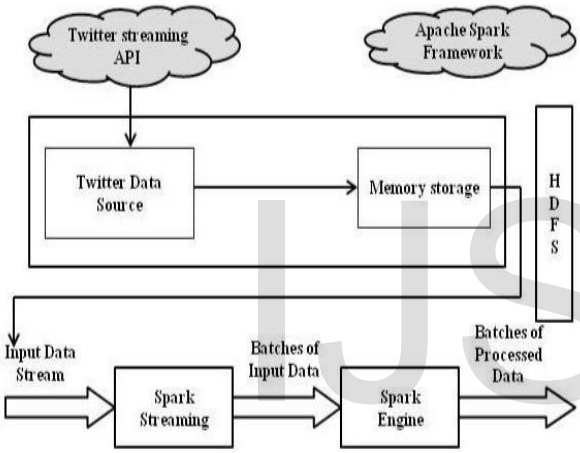

Figure 4: Apache Spark Data Processing for Twitter Streaming [13]

4. MATERIAL AND METHODS

The nature of this paper is to extract the relevant and meaningful information from a huge amount of data by analysing data in real time. Twitter is the social networking site that maintains an enormous amount of data on today basis. In this paper, the data processing is done by using Apache Spark and Twitter streaming API as shown in fig 4:

Apache spark: Apache spark is open source and distributed computing platform that is used for analyze of big data. Apache Spark process is used for estimating launch on cluster". This allows spark to run on single node on distributed computing platform such as ports of the execution time that is considered for average number of tasks executing that particular time. [19]

$$\text{Jobs to be executed} = \{ \text{level}_i \mid 0 =< i =< M \}$$

$$\text{Execution Time} = \{ \text{Task}_{i,j} \mid 0 =< j =< N \}$$

Here, M is the number of stages in the job assigned and N is the number of task assigned in a stage. For Processing, we need to operate the worker nodes in the standalone mode. Worker node extracts sufficient amount of sample input data to process the blocks of input by using CPU core. Hence this can configure one block of Hadoop Distributed File System (HDFS) to the size of sample that can be computed by the block size and number of tasks to be executed in parallel i.e. *Block Size * Number of Tasks*. [19]

Twitter Streaming API: To evaluate this model, we need to run spark on standalone where every node is called a worker node. For experimental evaluation, we run apache spark in standalone mode on top of HDFS with its default settings. Twitter stream API fetches the tweets to the apache spark framework for data collection used to record execution profiles and performance metric. [18] Twitter Tweets can be counted easily on command line by the file dataset of twitter.

Scala: Scala is a object oriented and functional language; it treats every value as an object and every function as a value. To compile scala code on apache spark we require plug-in to be added in scala tool repository. Scala repository then include scala and apache spark as dependencies. [20]

5. PROPOSED WORK

Spark is built on the top of Hadoop system, a "parallel processing platform that lets the multiple parallel application share a single node and provide API to be

hadoop, yarn, and daemon services that can process the data. [21] Apache Spark platform works on various

stages to extract patterns through twitter steaming API. Following scenarios will be executed for experiment purpose on live streams of tweets on twitter.

5.1 Apache spark reads the twitter tweets from a set of documents fetched by twitter streaming API.

5.2 Count the number of times each word appears in the text file.

5.3 Filter out all the words that are less than a specific number of times in the tweets.

5.4 For the remaining set we count the number of times each letter occurs.

5.5 Fetch the top ten words that are collected during a particular period of time.

5.6 Calculate number of a particular “word” being used in the tweets twitted in particular period of time.

These steps carry out a defined pattern for the analysis and relevant information from an enormous amount of data from the twitter tweets. Our framework can process large amount of stream data in real time.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented twitter streaming API, based on apache spark that maintains In-memory computing, parallel processing and fault tolerance. Apache spark’s small size code and wide range of built in functions make its origin to both applications oriented and system oriented level for big data analysis. In addition, building on hadoop greatly reduced the efforts that have to go in Apache spark. Hadoop HDFS is used to store the data processed over spark platform as storage processed after map and reduce operations. This paper shows the analysis of a number of tweets in a various aspects. In future, we aspire to work on other real time streaming API’s in our framework and compare them. Implementation of the proposed work using Apache Spark is in progress.

REFERENCES

[1] "Apache Spark," April 2016. [Online]. Available: http://spark.apache.org/. [Accessed April 2016].

[2] "Apache Hadoop," [Online]. Available: http://hadoop.apache.org/. [Accessed 2016].

[3] "Significance of Apache Spark," 2016. [Online]. Available: https://www.mapr.com/blog/. [Accessed April 2016].

[4] I. Cordova and T. Moh, "DBSCAN on Resilient Distributed Datasets," 2015.

[5] "Apache Spark Introduction," 2016. [Online]. Available: http://www.infoq.com/articles/. [Accessed April 2016].

[6] M. Zaaharia, M. Chowdhury, T. D. A. Das, J. Ma, M. McCauley, J. ,. Franklin, S. Shenker and I. Stoica, "Resilient Distributed Datasets: A Fault Tolerant Abstraction for In-Memory Cluster computing," 2015.

[7] M. Armbrust, T. Das, A. Davidson, A. Ghodsi, A. Or, J. Rosen, I. Stoica, P. Wendell and R. Z. M. Xin, "Scaling Spark in the Real World: Performance and Usability," in *International Conference on Very Large Databases*, 2015.

[8] L. Rodriguez-Mazahua, C.-A. Rofriguez-Enriquez, L. Sanchez-Cervantes, L. Garcia-Alcaraz and A. Hernandez, "A general perspective of Big Data: Applications, Tools, Challenges and Trends," 2015.

[9] D. Singh and C. K. Reddy, "A Survey on Platform for Big Data Analytics," *Journal of Big Data*, 2014.

[10] P. I. Stocia, "Conquering Big Data With Spark," in *International Conference on Big Data*, 2015.

[11] M. Solaimani, M. Iftekhar, L. Khan, B. Thuraisingham and J. I. Burton, "Spark-based Anomaly Detection Over Multi-source VMware Performance Data In Real-time," 2014.

[12] A. N. Richter, T. M. Khosgoftaar, S. Landset and T. Husanin, "A Multi-Dimensional Comparison of Toolkits for Machine Learning With Big Data," in *IEEE*, 2015.

[13] A. G. Shoro and T. R. Soomro, "Big Data analysis: Ap Spark Perspective," *Global Journals Inc.*, vol. 15, no. 1, 2015.

[14] "Apache Spark Streaming," [Online]. Available: http://spark.apache.org/streaming/. [Accessed 2016].

[15] A. Ilari, M. Rautiainen, M. Salmi, S. Pirttikangas and J. Riekki, "Low Latency Analytics for Streaming Traffic Data with Apache Spark," in *International Conference on Big Data*, 2015.

[16] S. Aggarwal and B. R. Prasad, "High Speed Streaming Data Analysis of Web Generated Log Streams," in *International Conference on Industrial and Information Systems*, Sri Lanka, 2015.

[17] C. Anagnostopoulos and P. Triantafillou, "Learning to Accurately COUNT with Query-Driven Predictive Analytics," in *International Conference on Big Data*, 2015.

[18] Y. Arakawa, S. Tagashira and A. Fukuda, "Relationship Analysis between User's Contexts and Real Input Words through Twitter," in *Globecom 2010 workshop on ubiquitous computing and networks*, 2010.

[19] M. Zaharia, T. Das, H. LI, T. Hunter, S. Shenker and I. Stioca, "A fault-Tolerant Model for Scalable Stream Processing," 2012.

[20] "Scala Learn," [Online]. Available: http://www.scala-lang.org/documentation/.

[21] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, J. Franklin, S. Shenker and S. Ion, "Fast and Interactive Analytics over hadoop data with spark".