International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 04 Issue: 06 | June -2020 ISSN: 2582-3930

System Architecture and Development using Spark for Effective Resource Management in NAWEC Ltd, The Gambia

Pierre Gomez 1, Prof. A. Ananda Rao 2, Dr. P. Radhika Raju 3

1 M-Tech Scholar, Department of Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu, Andhra Pradesh, India

2 Professor, Department of Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu, Andhra Pradesh, India

3 Ad-hoc Assistant Professor, Department of Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu, Andhra Pradesh, India

---

Abstract - In recent years, there has been an exponential surge in the amount of digital data that have been produced. This poses tremendous opportunities as well as computational challenges. The data sizes in many instances have outperformed the capabilities of single machines, hence users need new systems to scale out computations to multiple nodes. The increase of data sizes led to the development of new cluster programming models targeting diverse computing jobs. These include MapReduce which support batch processing; Dremel was developed by Google for interactive SQL queries and Pregel for iterative graph algorithms etc. In the open-source world, Apache Hadoop stack systems like Storm and Impala were also specialized.In this paper we will explore large-scale computing architecture customized for real-time data processing of big data applications in spark; and to provide relevant views on how spark can be used to support a wider class of applications than MapReduce, without compromising its automatic fault tolerance.**Key words —Big data,spark, Hadoop, cluster, efficiency.**

1. Introduction

Companies now a days have been engulfed in this phenomenal data growth. This research looks at developing an architecture using spark for effective resource management in NAWEC. A national company like NAWEC (National Water and Electricity Company) Ltd, that provides Water and Electricity services to the populace in the entire country is not exempted from these data surge. In fact, the growing demands of customers increases day by day and the challenge for a system to meet up with these demands is inevitable. The present use of terminal server system will soon be incapable of handling the demands of the customers, throughout the country. Hence the need for a more efficient computing system tailored towards meeting the growing demands is a necessity. The exponential increase in the amount of digital Data generated daily; is due to the fact that computing systems has being in-cooperated into every aspect of the company's daily operation. The data generated from these systems is increasing and in varying structure. Introducing spark in the computing environment will radically transform the system by increasing data storage capacity, processing power and easy access; thereby facilitating resource management.

A. Introduction to Spark

Apache spark is a cluster computing framework for larger-scale data processing. Spark does not use MapReduce as an execution engine, but uses its own distributed runtime engine, or can be integrated with Hadoop to run on YARN and work with Hadoop file formats and storage backends like HDFS. Spark can keep large working dataset in memory between jobs.

The use of spark in this situation will make the computing platform more efficient and able to serve the needs of the customers. In Apache Hadoop stack systems, spark is more efficient than MapReduce in *multi-pass* applications that require low-latency data sharing across multiple parallel operations. Spark first extends its' data-sharing abstraction using 'Resilient Distributed Dataset' (RDD) for fault tolerant computation in a cluster. However, a number of higher-level APIs have been developed in spark. This include MLlib library for machine learning, SparkSQL for data manipulation using SQL queries, GraphX library for processing large graph. These simple extensions are used to capture a wide range of processing workloads that previously neededseparate engines. Spark also provides built in APIs for the following programing languages such as Python, R, Java and Scala. The experiment in this research will be carried out using PySpark. In spark there are *Iterative algorithms*, including many machine learning algorithms and graph algorithms like PageRank; *Interactive data mining*, where a user would like to load data into RAM across a cluster and query it repeatedly and *Streaming applications* that maintain aggregate state over time [2].

B. Spark features

Fig. 1 Spark features [6]

Speed: Spark speed on memory is about 100 times faster than Hadoop MapReduce for large-scale data processing. It is able to achieve this speed through controlled partitioning. It manages data using partitions that help parallelize distributed data processing with minimal network traffic.

Supports multiple languages: Spark provides built-in APIs in Java, Scala, Python or R. Therefore, you can write applications in different languages. Spark comes up with 80 high-level operators for interactive querying.

Advanced Analytics: Spark not only supports 'Map' and 'reduce'. It also supports SQL queries, Streaming data, Machine learning (ML), and Graph algorithms.

Lazy Evaluation:

Apache spark delay its evaluation until it is necessary to do so. This contributes to the speed of spark for transformation, it adds the transformations to DAG (Directed Acyclic Graph). DAG only get executed when the driver requests some data.

Spark unlike prior systems supports real time data processing (fig 2). This makes it suitable for the type of infrastructure that NAWEC needs for her computing platform. Data parallelism in spark leads to reduction in the processing time fig 3.

Fig 2: Real time data processing in spark [6]

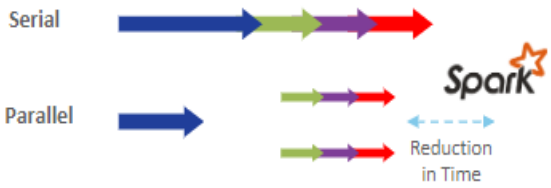

Fig 3. Data parallelism in Spark [6]

In spark, it is easier to develop application because they use a unified API. Combining processing tasks in spark is more efficient than the prior systems, which may require writing data to a storage before

passing it to another engine. Spark has the capability of running diverse functions over a dataset in memory. In spark, new applications can be enabled (such as interactive queries on a graph and streaming machine learning) which is not possible with previous systems [3].

2. Related work

There are significant volume of work that looked into various aspects of Big data processing systems in various distributed settings, in this paper, the focus is on recent efforts that investigates the use of spark in data processing and resource management in a rapidly growing company. The challenge of increasing sizes of datasets make it more demanding to explore efficient ways of processing and managing the data and various resources within the network.

3. Propose system

Thefocus of this research as stated above is to examine how to utilize the power of spark in a growing company for effective resource management. As established earlier,spark is more efficient than a number of parallel processing frameworks in use today. In this section a multi-node cluster comprising of one master node and three worker nodes will be setup and configured on a virtual machine for experimental purpose only. The system specification of the cluster is show in fig 4 below.

No	Node	RAM	CPU	Disk	OS
1	Master	2GB	1	64GB	CentOS
3	Worker Nodes	2GB each	1	32GB each	CentOS

Fig 4: Cluster Nodes specifications

The operating system used in the cluster is CentOS 8 and Hadoop, java, spark and python has been installed and configured to carry out the experiment. The cluster is shown in fig 5 below.

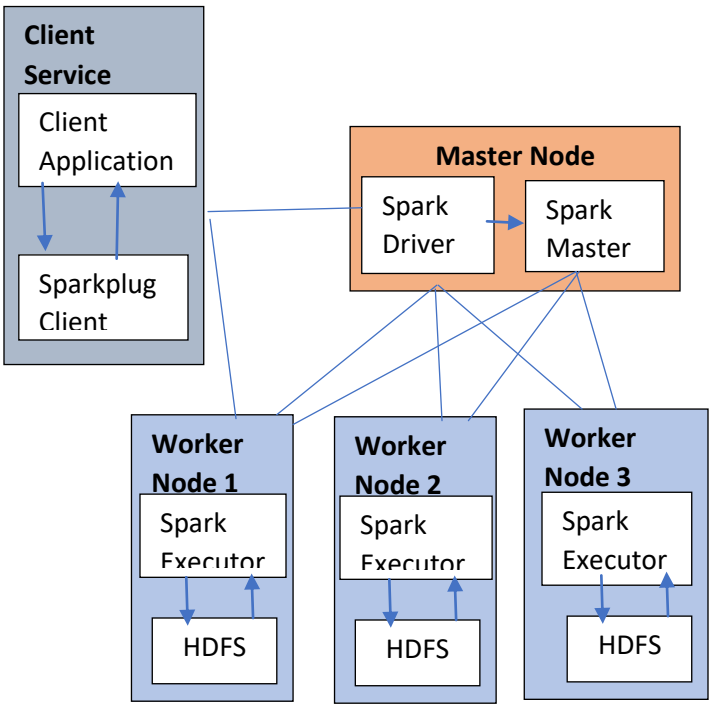

Fig 5. Spark Multi-node cluster

The cluster in fig 5 is a master, slave architecture; with two main processes.When the client sent a job from the client service machine, the **master node**,havethe ***driver program***, which drives the application. The code written behaves as a driver program or if you are using the interactive shell, the shell acts as the driver. In the driver program a **spark context** is created. The spark context act as a gateway to all spark functionalities. TheSpark context works with the ***cluster manager*** to manage various jobs in the cluster. A job is split into multiple tasks which are distributed over the worker node. Anytime an RDD is created in Spark context, it can be distributed across various nodes and can be cached there [6]. ***Worker nodes*** are the slave nodes whose job is to basically execute the tasks. These tasks are then executed on the partitioned RDDs in the worker node and hence returns back the result to the Spark Context [6].

The example below shows a python code using spark SQL context to read a csv file into a **DataFrame** and displays the records.

```
from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)

# Create the DataFrame
df =
sqlContext.read.csv("/home/pgomez/Project_Data/county_facts.csv")
# Show the content of the DataFrame
df. show()
```

A number of rows and columns can be selected from the dataframe just by using a select statement as shown below.

```
df.select('Date','open','High','low','close','volume','Adj Close').show()
```

```
+-------------------+---------+---------+---------+---------+--------+---------+
|               Date|     Open|     High|      Low|    Close|  Volume|Adj Close|
+-------------------+---------+---------+---------+---------+--------+---------+
|2015-12-31 00:00:00|    41.41|41.619999|40.799999|    40.82| 1481100|39.634133|
|2015-12-30 00:00:00|    41.68|    41.93|41.470001|41.509998| 1991000|40.304086|
|2015-12-29 00:00:00|41.610001|41.959999|41.490002|41.639999| 1697100|40.430311|
|2015-12-28 00:00:00|41.389999|41.599998|40.959999|41.400002| 1604800|40.197285|
|2015-12-24 00:00:00|    41.27|41.759998|     41.0|41.509998| 1393300|40.304086|
|2015-12-23 00:00:00|40.290001|41.310001|     39.5|41.259998| 3754500|40.061349|
|2015-12-22 00:00:00|39.009998|40.080002|38.790001|40.080002| 2531000|38.915633|
|2015-12-21 00:00:00|    39.23|39.509998|38.439999|    38.84| 2900600|37.711654|
|2015-12-18 00:00:00|    39.77|    39.93|39.029999|    39.09| 5390900|37.954392|
|2015-12-17 00:00:00|    40.43|40.490002|    39.68|39.990002| 3023700|38.828247|
|2015-12-16 00:00:00|40.150002|40.540001|39.630001|40.290001| 2067900|39.119531|
|2015-12-15 00:00:00|40.150002|40.330002|39.630001|39.830002| 2144800|38.672895|
|2015-12-14 00:00:00|39.610001|    39.91|    39.16|39.900002| 1856900|38.740862|
|2015-12-11 00:00:00|39.540001|40.110001|    39.27|39.470001| 2348700|38.323353|
|2015-12-10 00:00:00|40.189999|40.470001|    39.93|40.150002| 1608800|38.983599|
|2015-12-09 00:00:00|    39.98|40.919998|39.900002|40.220001| 2178300|39.051565|
```

This shows that the cluster setup is correctly carried out and the daemons are active.

3.3. System Architecture for NAWEC

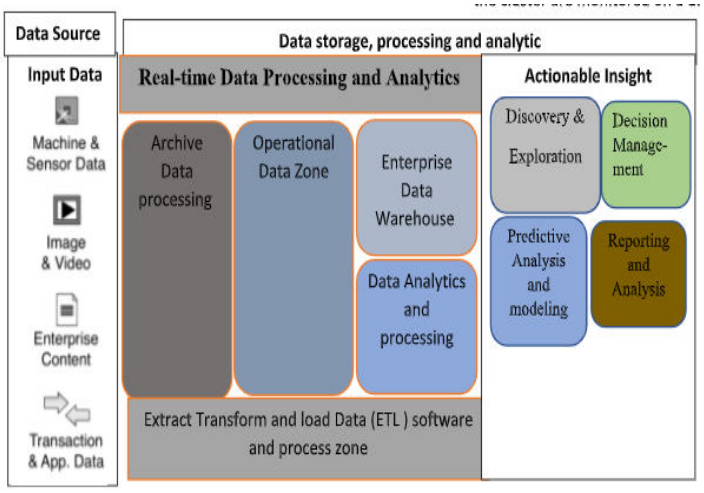

Fig 6. System Architecture

The architecture is for alarge-scale computing platform customized for real-time data processing of big data applications using spark;the cluster is divided into two main segments, that is Data source and Data processing, analysis and storage. Data type includes structured, semi-structured and unstructured data; however, our main focus is structured and semi-structured data. This hybrid architecture is used to increase the efficiency of the cluster. The different compartmentalizationensures that there is no bottle neck during operation and on the network.

Data input can be processed and analyze in real-time, whiles storing when necessary in the Enterprise Data Warehouse. Operational Data Zone will also be in use as required by user including analytics and processing section. Data from legacy system are stored in the Archive.

Extraction Transformation and load (ETL) software, to extra, transform and integrate data from other systems to the cluster. In the **Actionable Insight section** decision making, reporting and analysis of data. All connected systems on the cluster are monitored on a dashboard.

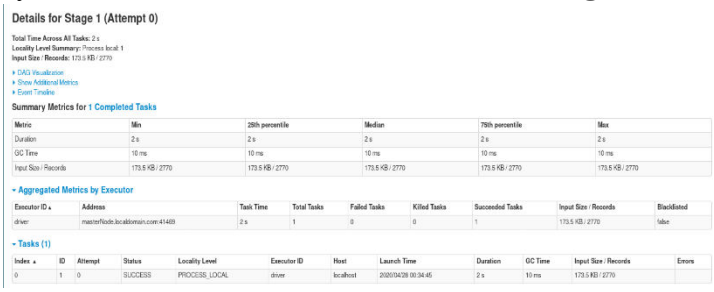

Fig 7: execution summary

4. Experiment and result

The experiment is carried out in a spark cluster using Pyspark and SparkSQL to receive and analyze data. A dataset of over 2500 records was inserted into the cluster and the time taken to process the data was 2s as shown in fig 7, in a virtual machine cluster, the output time is satisfactory, considering that the virtual machines configurations are not very high compare to physical server. When implemented on high end servers, processing time will very small.

Estimating Execution Time

Spark job is executed in multiple stages, and each stage contains multiple tasks. In this paper the following notation is used to represent an apachespark job:

Job= {Stagei | 0 ≤ i ≤ Y} equation 1
Stagei = {Taski, j | 0 ≤ j ≤ N} equation 2

Here Yrepresentsnumberofstagesinajoband N isthenumber of tasks in a stage. And different stages within a job are executed sequentially, the execution time of a job is here represented as the sum of the execution time of each stage plus the job startup time and the job cleanup time as follows:

$$\text{JobTime} = \text{Startup} + \sum_{s=1}^{Y} \text{stageTime} + \text{Cleanup}$$

Furthermore, within each stage, as one CPU core executes one task at a time, in a cluster with a number of worker nodes represented as "H worker nodes", the number of tasks P that can run in parallel can be calculated as follows:

$$p = \sum_{i=1}^{H} CoreNumi$$

CoreNumi represents the number of CPU cores of working nodesi and H is the number of working nodes in a cluster [5].

5. Conclusions and future work

This paper concentrated on spark and a custom architecture for a developing company for efficient implementation of real time big data processing in spark cluster, one of the most widespread models for large scale data processing in today's warehouse-scale computers.Given the rapid growth of big data applications and their financial, social, and health benefits, the demand for higher throughput, computing capacity, and performance will only increase in foreseeable future. This increasingly intensifies the need and interest for further research in this area to fill in the gaps of the growing industry [1]. Emerging technology suitable for real-time big data processing in production datacenters is a clear sign that the trend towardlarge-scale custom computing systems has only begun [1].

Reference

[1] Heterogeneous Architectures for Big Data Batch Processing in MapReduce ParadigmM. Goudarzi, Senior Member, IEEE Computer Engineering Department, Sharif University of Technology, Tehran, I.R.Iran

[2] Apache Spark: A Unified Engine for Big Data Processing. COMMUNICATIONS OF THE ACM | NOVEMBER 2016 | VOL. 59 | NO. 11

[3] Scaling Spark in the Real World: Performance and Usability. Vol. 8, No. 12

[4] Performance PredictionforApacheSparkPlatform. 978-1-4799-8937-9/15 $31.00 © 2015 IEEE DOI 10.1109/HPCC-CSS-ICESS.2015.246

[5] SparkR: Scaling R Programs with Spark. SIGMOD '16, June 26–July 1, 2016, San Francisco, CA, USA.

[6] https://www.edureka.co/blog/spark-architecture/#Overview

[7] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, 2010.

[8] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2012, pp. 2–2.

[9] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of resource provisioning cost

in cloud computing," Services Computing, IEEE Transactions on, vol. 5, no. 2, pp. 164–177, 2012.

[10] P. Patel, D. Bansal, L. Yuan, A. Murthy, A. Greenberg, D. A. Maltz, R. Kern, H. Kumar, M. Zikos, H. Wu et al., "Ananta: Cloud scale load balancing," in ACM SIGCOMM Computer Communication Review, vol. 43, no. 4. ACM, 2013, pp. 207–218.

[11] Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," IDC iView: IDC Analyze the future, vol. 2007, pp. 1-16, 2012.

[12] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Int'l Symp. on Operating System Design and Implementation (OSDI), 2004.

[13] L. A. Barroso, J. Clidaras, and U. Hölzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines (2nd Edition): Morgan & Claypool Publishers, 2013.

[14] Apache Storm project; http://storm.apache.org

[15] Armbrust, M. et al. Spark SQL: Relational data processing in Spark. In Proceedings of the ACM SIGMOD/PODS Conference (Melbourne, Australia, May 31–June 4). ACM Press, New York, 2015.