Comparative Analysis and Evaluation of Big Data Processing Frameworks and Tools

Bhupender singh thakur
Department of Computer Science, Himachal Pradesh University

Dr. Kishori Lal Bansal
Department of Computer Science, Himachal Pradesh University

ABSTRACT

With the rise in demand for data analytics big data has become a key technique for solving problems. Big data analytics enables for us to solve many problems. Due to large amounts of data and great computational power big data processing will give more fast and accurate result. Big Data Analytics is one of the top priorities of the organizations participating in the survey as they believe that it improves the performances of their organizations. New technologies are now making it easier to perform increasingly sophisticated data analytics on a very large and diverse datasets. So, this study gives insight into various big data processing technologies for the individuals, researchers, and organizations to help them in decision making regarding the selection of big data processing systems.

Keywords: big data processing, big data frameworks, computational power, datasets, data analytics, decision making, performance.

1. INTRODUCTION

Big data can be defined as the data which surpasses the processing capacity of traditional database systems. It is huge in size, is being generated and transferred at a very fast rate and contains a whole lot variety of non-textual data. To gain value from this data, we must choose an alternative way to process it [1]. Big data is very difficult to process using the traditional data management techniques such as RDBMS. Big data is huge amount of heterogeneous data being generated at a very fast speed. Integration of various services such as internet, mobile phones, and cloud computing is ever increasingly affecting our lives and as a result we are generating variety of colossal information. Until recent years, the common usage of information systems in business has been as a support function. They have provided an electronic equivalent to existing paper processes, automating forms and workflow. Although providing tremendous efficiencies, these systems had largely remained a digital exoskeleton, the inputs and outputs of which are mediated by physical realities—a phone call, a storefront, a factory floor. The arrival of the web began an important transition. Information systems are no longer just support systems in the middle of a business. They have become the business interface—the storefront, the marketing, the customer service. For many products and services that are digital in nature, information systems also form the means of delivery. The streams of log and user data generated from these systems give insight not only into the operation of a system itself, but into the proclivities and behaviour patterns of users. With the increase in mobile phones, computing has reached a large audience and it has provided opportunity to engage with people in any place and at any time. Also, with advancement in physical instrumentation, from sensors to robotics, have broadened the reach of algorithms. A computer might detect temperature change and accordingly make adjustment to an air conditioner or heater or it can reroute a taxicab due to excessive traffic. Rapid advancement of low cost 3-D printing has enabled the computers to instantiate artefacts in the physical world. In short, information system is transforming their role from digital exoskeleton to digital nervous system.

The effect of ecommerce surge, rise in merger/acquisition activity, increased collaborations, and the drive for harnessing information as a competitive catalyst is driving enterprises to higher levels of consciousness about how data is managed at its most basic level. With the advancement in technology the rate of data generation has risen exponentially. Today data is being generated at very high rate which has led to the accumulation of colossal amount of data around the world

2. BIG DATA CHARACTERIZATION

Data Evolution

First, there are multiple factors responsible for the evolution of data from being simple in structure and limited in size to complex in structure and unlimited in size. Following are some the main factors:

- Evolution of technology
- Social media
- Internet of things (IOT)
- Education and research
- Banking and finance

Characteristics of big data

Big data is characterized using different characteristics all of which starts with the letter 'V'. Doug Laney was first to characterize the data on the basis of 3 primary characteristics i.e. volume, velocity, and variety[2]. Since then many new characteristics have been identified of which following are the main:

- Volume – Storing huge amount of data generated overtime leads to colossal volume of data. The accumulated data is used to gain useful insights by different businesses.
- Velocity – It is the rate of data flow or data generation. Data is being generated and consumed at an alarming rate. The rate of data generation has exploded in past few years.
- Variety – Heterogeneous sources of data generates data in different formats. Big data is broadly classified into 3 types i.e. Structured, Semi-Structured, Unstructured.
- Veracity – Data transmission and operation can cause data loss and as a result big data contains huge amount of inconsistencies and uncertainty which needs to be removed.
- Value – Value refers to the useful information that is hidden in the data. Useful information is mined from the huge pools of data sets and then that data is analysed to extract valuable information.

3. PAPER STRUCTURE

This study categorises the Big Data processing frameworks as batch- only, stream-only, and hybrid. Also, the tools of Big Data processing can be classified into 3 types i.e. Data management tools, Data storage tools, and Data analytics tools. Following figure depicts the structure of this study:

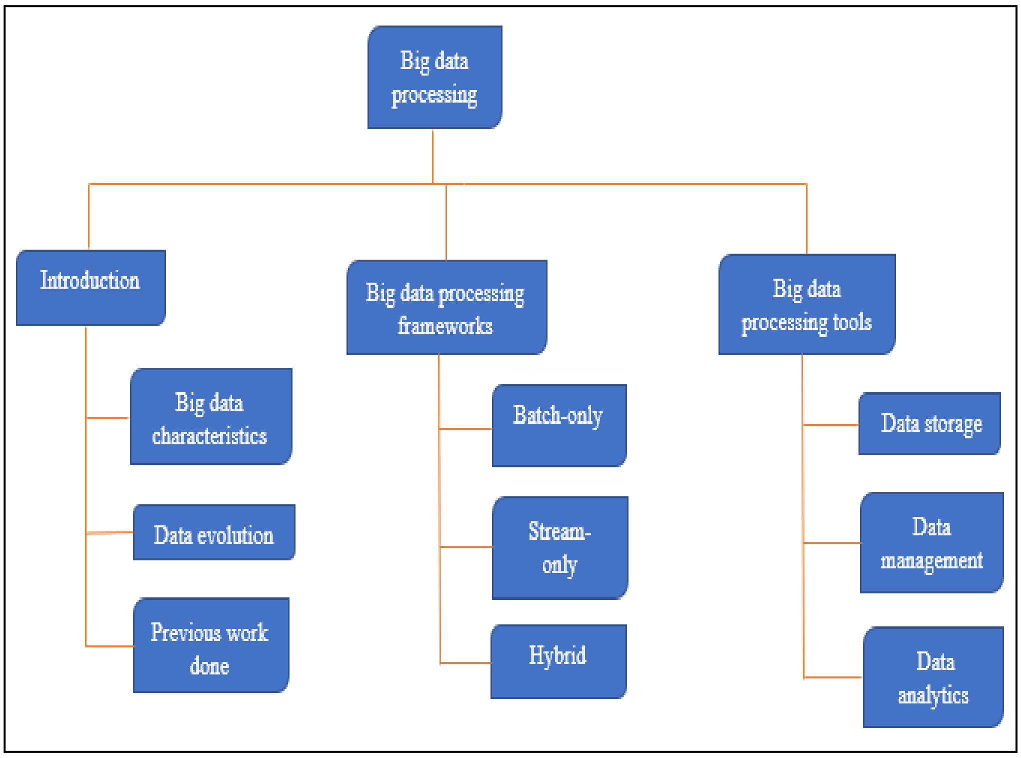

Figure 1. Organization of paper

4. PREVIOUS WORK DONE

This Section provides an overview of the previous studies in this field. In [1] the author defines big data as the data that surpasses the processing power of traditional database systems. It also describes the transformation of the role information systems from earlier being digital exoskeleton to now being the nervous system. In [2] the author accurately anticipated that During 2001/02, leading enterprises will increasingly use a centralized data warehouse to define a common business vocabulary that improves internal and external collaboration. Through 2003/04, data quality and integration woes will be tempered by data profiling technologies (for generating metadata, consolidated schemas, and integration logic) and information logistics agents. By 2005/06, data, document, and knowledge management will coalesce, driven by schema-agnostic indexing strategies and portal maturity. In [3] this study author has described an approach to make sense of historical data using computational tools and making historical data as huge digital data resources so as to make sense of huge amount of historical data. The author in [4] described big data as huge amount of complex data sets which for which the traditional data processing applications are not sufficient and distributed databases are needed. Firms like Google, eBay, LinkedIn, and Face book were built around big data from the beginning. In [5]: The author has proposed the lambda architecture for building suitable and reliable big data systems from scratch. Lambda architecture is basically meant for real time big data systems but this work also provides detailed insight into the working of batch processing systems.

5. BIG DATA PROCESSING

Big data processing consists of both storage and processing the data for analysis. The traditional tools and techniques were based on the principle of simple data structure, but big data is mainly composed of unstructured data. Therefore, big data requires non-traditional technologies to gather, organize, process and to gain insight from large data sets.

Classification of Big Data Processing Frameworks

Data transformed from being limited and simple into unlimited and complex in structure. When traditional data processing tools and technologies were exposed to the modern face of big data, they were unable to handle it efficiently. One of the main reasons for their inability to keep up was that, at the time these traditional tools and technologies were developed keeping in mind the limited size and simple structure of the data. Therefore, when confronted with the big data they were unable to process that type of data. So, new tools and technology was developed in order to handle the storage and processing of big data efficiently. NOSQL, Hadoop, Apache, Storm, Samza, and Flink are some of the popular big data processing frameworks. These frameworks further constitute various independent tools such as Pig, Hive, HDFS (Hadoop distributed file system, Elephant DB, Mongo DB, Cassandra, etc. Big data processing frameworks process the data in two ways. They either read the data from a fixed storage or they read the data from live streams entering the system. There are 3 main types of big data processing frameworks[6]:

- Batch-only processing – they perform batch-based storage data processing and have high processing latency. Apache Hadoop is the canonical example of batch processing
- Stream-only processing – they perform real-time data stream processing. They are time sensitive and have low processing latency. Apaches Storm and Apache Samza are the examples of stream-only processing.
- Hybrid processing – they can perform both types of processing on big data. Apache Spark, Apache Flink are the examples of hybrid processing frameworks.

Following are the most widely used big data processing frameworks:

1) Hadoop framework

Hadoop is an open source architecture used for building up big data processing system. It is a batch only processing framework and uses distributed architecture for the storage of data across multiple machines/nodes. It performs parallel processing of data stored in distributed fashion. Hence it provides high performance in terms of processing the data and therefore is very popular among various organizations and research scholars. The main components of Hadoop framework are:

- Hadoop common – It constitutes libraries and utilities used by other Hadoop modules.
- Hadoop Distributes file system – It is the Storage component of Hadoop framework. It stores data across multiple nodes using the master slave architecture.
- Hadoop MapReduce – It is responsible for the processing of large-scale data stored in Hadoop. It provides high performance as it computes data in parallel from across multiple nodes using Master-Slave architecture.
- Hadoop YARN – It is the resource manager of Hadoop framework. It manages computing resources and schedules user applications.

2) Spark framework

It is an Apache open-source in-memory cluster computing framework for big data processing. It is uses resilient distributed datasets (RDD) which are data sets stored in distributed fashion and available in read only mode. These datasets are maintained in a fault tolerant manner. It uses its own cluster management and hence cannot be categorised as modification of Hadoop. Spark enables significantly higher performance and functionality than other frameworks and it provides a general-purpose framework suitable for different big data processing tasks such as batch processing, interactive queries, stream processing, graph processing, etc.

Apache Spark is constituted of various components classified into following types [7]:

- Spark Core – It is foundation of the whole Apache spark project. It provides a programmable interface for allocating tasks in distributed fashion across the cluster, scheduling, and basic input and output functionalities.
- Data storage – It consists of the data storage components such as HDFS, Cassandra, HBase, Hive, etc.
- Cluster manager – It manages the resource sharing between Spark applications. Although Spark has an inbuilt cluster manager, it can also run on various other cluster managers such as Hadoop YARN, Apache Mesos, Amazon EC2, etc.
- Upper level Libraries – Many libraries have been built on top of Spark core for handling different types of workloads. Spark MLib is for machine learning tasks, Spark Streaming for Streaming analysis, Spark SQL for processing structured data, and GraphX for graph processing.

3) Flink framework

Apache Flink is an open source in-memory processing framework used for both batch and stream processing of big data. It has high fault tolerance to steadily recover the state of data streaming applications. This process creates continuous snapshots of the distributed data stream and operator state. In case of failure, the system can rely on these snapshots and revert back. Flink architecture can be described by 5 layers [8]:

- Data storage layer – This layer is responsible for reading and writing data from various sources such as HDFS, local files, etc.
- Deploy and resource management – It is responsible for management of resources and task planning using cluster manager. It also provides the environment for the execution of programs responsible for Cloud and Cluster environment.
- Kernel – It contains distributed stream data flow engine for real time processing.
- APIs – It provides 2 interfaces i.e. DataStream and DataSet for real-time and batch processing respectively.
- Library – In this layer programs are written. Java and Scala programming languages are supported.

4) Samza framework

Apache Samza is open source near real time event stream processing framework. Mostly Samza is used in combination with Apache Kafka which provides partitioned messages in a fault tolerant manner to Samza and Samza provides a framework to process those messages [9]. However, Samza and Kafka can be used independently. It follows distributed architecture which provides high throughput for colossal streams of data. Samza is built upon 3 main concepts which are responsible for its high scalability and operational robustness [10]:

- Partitioned Log Processing – the inputs to a Samza job are partitioned into disjoint subsets and each input partition is allocated to only one processing task. The number of partitions in the input decides the degree of parallelism of the job.

- Fault-tolerant Local State – in Samza the state of a task is recorded on the local disk of the processing node. It also maintains an in-memory cache for frequently used items. In case of a local disk failure Samza maintains a replicated log of the state record.

- Cluster-based Task Scheduling – Samza does not have an inbuilt mechanism for resource management and deployment. It uses existing cluster management software for that purpose. It supports 2 modes of distributed operation. One is Hadoop YARN and another is its Standalone mode.

5) Storm framework

Apache Storm is a distributed real-time processing framework. Storm is designed to process vast amount of stream data in a fault-tolerant and horizontally scalable manner. It is a stream processing framework that has the capability of highest ingestion rates. Storm works on the logic of Spout and Bolts, where a Spout is a source of streams and Bolt is responsible for the processing of these streams [11]. System of Spouts and Bolts is called a Topology. User provides topologies to the Storm application. A Topology is represented by a directed acyclic graph, where each node of the graph represents a Spout or a Bolt and the edges represent the flow of data between the nodes. A Storm cluster consists 2 types of nodes:

- Master node – It executes a daemon called Nimbus which distributes code in the cluster, assigns tasks to other nodes and monitors their progress and failures. It also runs another daemon called UI which is responsible for providing graphical user interface to the user for viewing the cluster status and managing topologies.

- Worker node – It executed a daemon called Supervisor which receives the work assigned to its machine by the master node. Each worker node executes a subset of a topology.

The coordination between the Master and Worker node is done using ZooKeeper which connects Nimbus with the Supervisor.

6) Giraph framework

Apache Giraph is a graph processing framework and is built on top of Hadoop framework. It uses distributed processing structure for large scale graph processing such as in the analysis of the interconnected web or social media. A Giraph computation receives graphs as its input. These graphs constitute vertices and edges which may represent different relations in different scenarios for example the vertices may represent a user in social media connected with other through edges representing friendship or friend request. The Giraph computations follows a sequence of iterations, called Supersteps and uses Bulk synchronous parallel computation model.

Every Giraph computation is a result of the coordinated activities between three network services [12]:

- Masters – The master service is responsible for running the master compute node which progresses the workers from one Superstep to another and monitors the health and statistics of all the workers.

- Workers – It is responsible for managing the state of graph partitions assigned to them. They also checkpoint their state so as to provide for easy recovery from a worker failure.

- Coordinators – It is responsible for providing distributed configuration, synchronization, and naming registry services.

Classification of Big Data Processing Tools

Big data frameworks are generally composed of various independent tools which interact with other tools to provide different types of services such as batch processing, stream or real-time processing. Following are the some of the widely used tools for big data processing and analysis. Big data processing tools can be categorized into three types namely Management tools, Storage tools, and Analytics tools [13].

- Data management tools: these tools are responsible for collecting, aggregating, and Transferring data between multiple sources such as backend logs and online services. For example, Apache flume, Apache Kafka, Apache Nifi.
- Data Storage tools: these tools consist of database management systems which provide the mechanism for data storage and retrieval. There are different types of databases such as relational databases, hierarchical databases, entity-relationship databases, object-based databases, etc. For example, Hadoop distributed file system, Cassandra, DynamoDB, HBase, CouchDB, MongoDB.
- Data Analytics tools: these tools include many different types of tools such as stream/batch Processing tools, scalable machine learning frameworks- that ease implementation of data analytics use cases such as collaborative filtering and sentiment analysis. For example, MapReduce, Apache Mahout, Apache Hive, Apache Pig, Statistical analysis Software (SAS).

6. RESULTS

In this study the big data processing frameworks and tools have been classified according to the types of functions and services provided by them. Big data processing frameworks are categorized according to the type of processing supported by the frameworks. The frameworks are classified as batch, streaming, and hybrid. These frameworks are composed of several tools which provides different functions to the framework. These tools function as a unit to provide various big data processing services such as storage, resource management, file system, workflow monitoring. The tools have been categorized by three categories i.e. data management, data storage, data analysis. Most of the tools are classified under one category such as HBase, Pig, Hive, Mahout, Flume, Sqoop, etc. Whereas some tools can be classified under more than one category. For example, SAS can provide services in all 3 categories.

Tool	Security	Fault tolerance	Scalability	Performance
Flume	High	Yes	Yes	High
Kafka	High	Yes	Yes	Very high
NiFi	Low	Yes	Yes	High

Table 1. Big data Management tools comparison table

Properties	Hadoop	Spark	Flink	Storm	Samza
Fault-Tolerance	High (Replication)	Low (resilient distributed dataset)	High (distributed snapshot check-up points)	Low	Low(log based)
Latency	High (minutes)	Medium (seconds)	Low	Very low (milliseconds)	Low (milliseconds)
Data format for analytics [14]	Structured/ Unstructured	Structured/ Unstructured	Structured/ Unstructured	Structured/ Unstructured	Structured/ Unstructured
Processing type	Batch only	Hybrid	Hybrid	Stream only	Stream only
Efficiency [14]	Low for real time applications & high for batch processing	High	High	High	High
File systems	HDFS, FTP, Amazon S3	HDFS, Amazon S3, Tachyon	HDFS, Amazon S3, Local FS	GPFS, Lustre	Kafka
Cluster management	YARN	Standalone, YARN, Mesos	YARN	YARN, Mesos	YARN
Language supported	Python, Java, C, C++, Ruby, Pearl, Groovy	Java, Scala, Pyhton and R	Java and Scala	Python, Ruby, JVM based languages	Python, Scala, Java
Computation type	Disk based	In-Memory	In-Memory	In-Memory	In-Memory

Table 2. Big data Frameworks comparison table

Tool	Fault tolerance	Architecture	Scalable [14]	Data Storage Format [14]	Single point failure [14]
Casandra	Yes(optional)	Distributed	Yes	Structured / Semi-structured / unstructured (schema less)	No
DynamoDB	Yes	Distributed	Yes (Optional)	Structured/ Unstructured	No
HBase	Yes	Distributed	Yes	Structured *i.e.* Tabular but not exactly row-oriented Relational Table	No
CouchDB	Yes	Distributed	Yes	Structured/ Unstructured	No
MongoDB	Yes (Replication)	Distributed	Yes (Horizontally using Sharding)	Structured/ Unstructured	No

Table 3. Big data Storage tools comparison table

Tool	Distributed computation	Efficiency	Latency time for query	Data Format for Analytics
MapReduce	Yes	Low for real time	Not applicable	Structured/ Unstructured
Mahout	Yes	High	Low	Structured/ Unstructured
Hive	Yes	High	Low (HiveQL)	Structured (RDBMS)
Pig	Yes	High	Low	Structured/ Unstructured
Apache Giraph	Yes	High	Low – In memory computation	Graph based

Table 4. Big data Analytics tools comparison table

7. CONCLUSION

In this study the Big Data processing frameworks and tools have been classified on the basis of the functions provided by them. Big Data processing frameworks are categorised based on the type of processing supported by the frameworks. The frameworks are classified as batch, streaming, and hybrid. These frameworks are composed of several tools which provides different functions to the framework. These tools functions in coordination with each other to provide various services such as storage, resource management, file system, workflow monitoring. The tools have been categorised into three categories i.e. data management, data storage, data analysis. Most of the tools are classified under one category such as HBase, Pig, Hive, Mahout, Flume, Sqoop, etc. Whereas some tools can be classified under more than one category. Tool such as SAS can provide services in each category. This study analysis four frameworks i.e. Hadoop, Flink, Spark, and Storm, based on different key performance indicators. The results show that Flink is better compared to the other frameworks, as it provides best performance in eight measures. Spark proved to be better than others in six measures, and Storm was better in four measures. Thus, personnel from organizations, researchers and individuals who, in order to analyse data and gain efficient results, wants to gain an overview of these tools based can select from these frameworks and tools based on the key performance indicators. This study aims to find some available computing and storage paradigms and tools that are being used in current scenario to address challenges of Big Data processing. We have categorized the survey into two streams. One stream contains study and survey of existing computing paradigms and tools used to perform computation on Big Data and the other stream gives a detailed survey of storage mechanisms and tools available today.

REFERENCES

[1] E. Dumbill, “Making Sense of Big Data,” *Big Data*, vol. 1, no. 1, pp. 1–2, 2013, doi: 10.1089/big.2012.1503.

[2] D. Laney, “3d data management controlling data volume velocity and variety,” *META Gr. Res. note*, p. 4, 2001.

[3] L. Hoffmann, “Looking back at big data: As computational tools open up new ways of understanding history, historians and computer scientists are working together to explore the possibilities,” *Commun. ACM*, vol. 56, no. 4, pp. 21–23, 2013, doi: 10.1145/2436256.2436263.

[4] M. Thomas, “A Review paper on Big Data,” *Int. J. Mob. Comput. Appl.*, vol. 2, no. 9, pp. 1–5, 2015, doi: 10.14445/23939141/ijmca-v6i2p101.

[5] N. Marz and J. Warren, *Big Data: Principles and best practices of scalable realtime data systems*. 2015.

[6] D. K. lal Bansal and Sanisha, “A comparitive study of open source big data processing frameworks for analysing big data,” 2017.

[7] S. Salloum, R. Dautov, X. Chen, P. X. Peng, and J. Z. Huang, “Big data analytics on Apache Spark,” *Int. J. Data Sci. Anal.*, vol. 1, no. 3–4, pp. 145–164, 2016, doi: 10.1007/s41060-016-0027-9.

[8] S. Alkatheri, S. A. Abbas, and M. A. Siddiqui, “Comparative Study of Big Data Frameworks,” *Int. J. Comput. Sci. Inf. Secur.*, vol. 17, no. January, pp. 1–4, 2020, doi: 10.1109/icict46931.2019.8977680.

[9] M. Kleppmann and J. Kreps, “Kafka , Samza and the Unix Philosophy of Distributed Data,” *IEEE Data Eng. Bull.*, vol. 38, no. 4, pp. 1–11, 2015.

[10] M. Kleppmann, “Apache Samza,” *Encycl. Big Data Technol.*, 2018.

[11] J. S. Van Der Veen, B. Van Der Waaij, E. Lazovik, W. Wijbrandi, and R. J. Meijer, “Dynamically Scaling Apache Storm for the Analysis of Streaming Data,” in *IEEE First International*

Conference on Big Data Computing Service and Applications, 2015, pp. 154–161, doi: 10.1109/BigDataService.2015.56.

[12] C. Martella, R. Shaposhnik, and D. Logothetis, *Practical Graph Analytics with Apache Giraph*. 2015.

[13] R. Ranjan, "Streaming Big Data Processing in Datacenter Clouds," *IEEE Cloud Comput.*, vol. 1, no. 1, pp. 78–83, 2014, doi: 10.1109/MCC.2014.22.

[14] B. R. Prasad and S. Agarwal, "Comparative Study of Big Data Computing and Storage Tools: A Review," *Int. J. Database Theory Appl.*, vol. 9, no. 1, pp. 45–66, 2016, doi: 10.14257/ijdta.2016.9.1.05.