1.Introduction

The primary purpose of this paper is to provide an in-depth analysis of different platforms available for performing big data analytics. This paper surveys different platforms available for big data analytics and assesses the advantages and drawbacks of each of these platforms based on various metrics such as scalability, data I/O rate, fault tolerance, real-time processing, data size supported and iterative task support.

This is an era of Big Data. Big Data is driving radical changes in traditional data analysis platforms. To perform any kind of analysis on such voluminous and complex data, scaling up the hardware platforms becomes imminent and choosing the right hardware software platforms becomes a crucial decision if the user's requirements are to be satisfied in a reasonable amount of time. There are several big data platforms available with different characteristics and choosing the right platform requires an in-depth knowledge about the capabilities of all these platforms. Especially, the ability of the platform to adapt to increased data processing demands plays a critical role in deciding if it is appropriate to build the analytics based solutions on a particular platform. Typically, when the user has to decide the right platforms to choose from, he/she will have to investigate what their application/algorithm needs are.

2.Type of computing platforms

One will come across a few fundamental issues in their mind before making the right decisions.

- How quickly do we need to get the results?
- How big is the data to be processed?
- Does the model building require several iterations or a single iteration?

Clearly, these concerns are application/algorithm dependent that one needs to address before analyzing the systems/platform-level requirements. At the systems level, one has to meticulously look into the following concerns:

- Will there be a need for more data processing capability in the future?
- Is the rate of data transfer critical for this application?
- Is there a need for handling hardware failures within the application?

The big data platforms can be categorized into the following two types of scaling:
Scaling is the ability of the system to adapt to increased demands in terms of data processing. To support big data processing, different platforms incorporate scaling in different forms.

- **Horizontal Scaling**

Horizontal scaling involves distributing the workload across many servers which may be even commodity machines. It is also known as "scale out", where multiple independent machines are added together in order to improve the processing capability. Typically, multiple instances of the operating system are running on separate machines.

- **Vertical Scaling**

Vertical Scaling involves installing more processors, more memory and faster hardware, typically, within a single server. It is also known as "scale up" and it usually involves a single instance of an operating system.
A comparison of advantages and drawbacks of horizontal and vertical scaling
Horizontal scaling

- Increases performance in small steps as needed
- Software has to handle all the data distribution and parallel processing complexities
- Financial investment to upgrade is relatively less
- Limited number of software are available that can take advantage of horizontal scaling
- Can scale out the system as much as needed

Vertical scaling

- Most of the software can easily take advantage of vertical scaling
- Requires substantial financial investment
- System has to be more powerful to handle future workloads and initially the additional performance in not fully utilized
- It is not possible to scale up vertically after a certain limit

it limits the scaling ability of a platform since it will require substantial financial investment. To handle future workloads, one always will have to add hardware which is more powerful than the current requirements due to limited space and the number of expansion slots available in a single machine. This forces the user to invest more than what is required for his current processing needs.

On the other hand, horizontal scale out gives users the ability to increase the performance in small increments which lowers the financial investment. Also, there is no limit on the amount of scaling that can done and one can horizontally scale out the system as much as needed. In spite of these advantages, the main drawback is the limited availability of software frameworks that can effectively utilize horizontal scaling.

2.1 Horizontal scaling platforms

Some of the prominent horizontal scale out platforms include peer-to-peer networks and Apache Hadoop. Recently, researchers have also been working on developing the next generation of horizontal scale out tools such as Spark to overcome the limitations of other platforms.

2.1.1 Peer-to-peer networks

Peer-to-Peer networks involve millions of machines connected in a network. It is a decentralized and distributed network architecture where the nodes in the networks (known as peers) serve as well as consume resources. It is one of the oldest distributed computing platforms in existence. Typically, Message Passing Interface (MPI) is the communication scheme used in such a setup to communicate and exchange the data between peers. Each node can store the data instances and the scale out is practically unlimited (can be millions of nodes). The major bottleneck in such a setup arises in the communication between different nodes. Broadcasting messages in a peer-to-peer network is cheaper but the aggregation of data/results is much more expensive. In addition, the messages are sent over the network in the form of a spanning tree with an arbitrary node as the root where the broadcasting is initiated. MPI, which is the standard software communication paradigm used in this network, has been in use for several years and is well-established and thoroughly debugged. One of the main features of MPI includes the state preserving process i.e., processes can live as long as the system runs and there is no need to read the same data again and again as in the case of other frameworks such as MapReduce (explained in section “Apache Hadoop”). All the parameters can be preserved locally. Hence, unlike MapReduce, MPI is well suited for iterative processing. Another feature of MPI is the hierarchical master/slave paradigm. When MPI is deployed in the master–slave model, the slave machine can become the master for other processes. This can be extremely useful for dynamic resource allocation where the slaves have large amounts of data to process.

MPI is available for many programming languages. It includes methods to send and receive messages and data. Some other methods available with MPI are ‘Broadcast’, which is used to

broadcast the data or messages over all the nodes and 'Barrier', which is another method that can put a barrier and allows all the processes to synchronize and reach up to a certain point before proceeding further. Although MPI appears to be perfect for developing algorithms for big data analytics, it has some major drawbacks. One of the primary drawbacks is the fault intolerance since MPI has no mechanism to handle faults. When used on top of peer-to-peer networks, which is a completely unreliable hardware, a single node failure can cause the entire system to shut down. Users have to implement some kind of fault tolerance mechanism within the program to avoid such unfortunate situations. With other frameworks such as Hadoop (that are robust to fault tolerance) becoming widely popular, MPI is not being widely used anymore.

2.1.2 Apache Hadoop

Apache Hadoop is an open source framework for storing and processing large data sets using clusters of commodity hardware. Hadoop is designed to scale up to hundreds and even thousands of nodes and is also highly fault tolerant. The Hadoop platform contains the following two important components:

- Distributed File System (HDFS) is a distributed file system that is used to store data across cluster of commodity machines while providing high availability and fault tolerance.
- Hadoop YARN is a resource management layer and schedules the jobs across the cluster. MapReduce

The programming model used in Hadoop is MapReduce which was proposed by Dean and Ghemawat at Google. MapReduce is the basic data processing scheme used in Hadoop which includes breaking the entire task into two parts, known as mappers and reducers. At a high-level, mappers read the data from HDFS, process it and generate some intermediate results to the reducers. Reducers are used to aggregate the intermediate results to generate the final output which is again written to HDFS. A typical Hadoop job involves running several mappers and reducers across different nodes in the cluster.

- Limitations of MapReduce

One of the major drawbacks of MapReduce is its inefficiency in running iterative algorithms. MapReduce is not designed for iterative processes. Mappers read the same data again and again from the disk. Hence, after each iteration, the results have to be written to the disk to pass them

onto the next iteration. This makes disk access a major bottleneck which significantly degrades the performance.

2.2 Vertical scaling platforms

The most popular vertical scale up paradigms are High Performance Computing Clusters (HPC), Multicore processors, Graphics Processing Unit (GPU) and Field Programmable Gate Arrays (FPGA). High performance computing (HPC) clusters HPC clusters, also called as blades or supercomputers, are machines with thousands of cores. They can have a different variety of disk organization, cache, communication mechanism etc. depending upon the user requirement. These systems use well- built powerful hardware which is optimized for speed and throughput. Because of the top quality high-end hardware, fault tolerance in such systems is not problematic since hardware failures are extremely rare. The initial cost of deploying such a system can be very high because of the use of the high-end hardware. They are not as scalable as Hadoop or Spark clusters but they are still capable of processing terabytes of data. The cost of scaling up such a system is much higher compared to Hadoop or Spark clusters.

The communication scheme used for such platforms is typically MPI. Since fault tolerance is not an important issue in this case, MPIs'lack of fault tolerance mechanism does not come as a significant drawback here.

2.2.1 Multicore CPU

Multicore refers to one machine having dozens of processing cores. They usually have shared memory but only one disk. Over the past few years, CPUs have gained internal parallelism. More recently, the number of cores per chip and the number of operations that a core can perform has increased significantly. Newer breeds of mother boards allow multiple CPUs within a single machine thereby increasing the parallelism. Until the last few years, CPUs were mainly responsible for accelerating the algorithms for big data analytics.

The parallelism in CPUs is mainly achieved through multithreading. All the cores share the same memory. The task has to be broken down into threads. Each thread is executed in parallel on different CPU cores. Most of the programming languages provide libraries to create threads and use CPU parallelism. The most popular choice of such programming languages is Java. Since multicore CPUs have been around for several years, a large number of software applications and programming environments are well developed for this platform. The developments in CPUs are not at the same pace compared to GPUs. The number of cores per CPU is still in double digits with

the processing power close to 10Gflops while a single GPU has more than 2500 processing cores with 1000Tflops of processing power. This massive parallelism in GPU makes it a more appealing option for parallel computing applications.

The drawback of CPUs is their limited number of processing cores and their primary dependence on the system memory for data access. System memory is limited to a few hundred gigabytes and this limits the size of the data that a CPU can process efficiently. Once the data size exceeds the system memory, disk access becomes a huge bottleneck. Other computing platforms are also available those are as follow.

- Dynamo

In 2006 Amazon developed Dynamo, which uses a key-value pair storage system. Dynamo is a highly available and scalable distributed data store built for Amazon's platform. Dynamo is used to manage services that need high reliability, availability, consistency, performance and cost effectiveness. The following models are also developed to support big data management and processing.

- HBase

HBase is an open source, non-relational, distributed database developed after big table. It works on the top of Hadoop Distributed file system and provides big-table like capabilities for Hadoop

- Apache Hive

Apache Hive is a data warehouse infrastructure built on top of Hadoop. It provides data summarization, query, and analysis of big data

- Berkeley Data Analytics Stack(BDAS)

The Berkeley Data Analytics Stack (BDAS) is an open source data analytics stack that integrates software components being built by the UC Berkeley AMP Lab for computing and analyzing big data. Many systems in the stack provide higher performance over other big data analytics tools, such as Hadoop. Nowadays, BDAS components are being used in various organizations.

The key open source components of the stack are:

Spark, a computation engine built on top of the Hadoop which support iterative processing (e.g. Machine Learning algorithms), and interactive queries. Spark gives an easy-to-program interface that is available in Java, Python, and Scala. Spark Streaming, a new component of Spark provides high Sherin A et al.

Shark is a significant data warehouse system. It runs on top of Spark. It allows users to run unmodified Hive queries on existing Hive workhouses because of backward-compatibility with Apache Hive,

Mesos, a cluster manager that gives an adequate platform for performing resource isolation and sharing efficiently across distributed applications.

- ASTERIX

ASTERIX is an Open Source System for big data management and analysis. With the help of ASTERIX Semi structured data can be easily ingested, stored, managed, indexed, retrieved and analyzed. Many of the drawbacks of Hadoop and similar platforms such as single system performance, difficulties of future maintenance, inefficiency in extracting data and awareness of record boundaries etc. are easily overcome by ASTERIX

- SciDB

SciDB is an open-source data management and analytics software system (DMAS) that uses a multi- dimensional array data model. SciDB is designed to store petabytes of data distributed over a large number of machines and used in scientific, geospatial, financial, and industrial applications to tackle, big data mining, very-large-scale parallel machine learning and data mining algorithms (ML- DM) are developed for e.g. Hadoop Map Reduce, NIMBLE, Big Cloud-Parallel Data Mining (BC- PDM).

- NIMBLE

An open source toolkit for the Implementation of Parallel Data Mining and Machine Learning Algorithms on Map Reduce for large datasets. It allows users to compose parallel ML-DM algorithms using reusable (serial and parallel) building blocks that can be efficiently manipulated using almost all parallel programming models such as Map Reduce. It runs on top of Hadoop.

- Big Cloud-Parallel Data Mining(BC-PDM)

Big Cloud Parallel Data Mining mainly relies on cloud computing and works on top of Hadoop and mainly used in intelligence data analysis.

- Graph Mining tools

Graphs are widely used in data mining application domains for identifying relationship patterns, rules, and anomalies. Certain examples for domains include the web graph, social networks etc.

3.Basic algorithms of computing platform

3.1 Decision tree induction classification algorithms

In the initial stage different Decision Tree Learning was used to analyze the big data. In decision tree induction algorithms, tree structure has been widely used to represent classification models. Most of these algorithms follow a greedy top down recursive partition strategy for the growth of the tree. Decision tree classifiers break a complex decision into collection of simpler decision. Hall. et al. proposed learning rules for a large set of training data. The work proposed by Hall et al generated a single decision system from a large and independent subset of data. An efficient decision tree algorithm based on rainforest frame work was developed for classifying large data set.

3.2 Evolutionary based classification algorithms

use domain independent technique to explore large spaces finding consistently good optimization solutions. There are different types of evolutionary algorithms such as genetic algorithms, genetic Sherin A et al. / International Journal of Computer Science & Engineering Technology (IJCSET) programming, evolution strategies, evolutionary programming and so on. Among these, genetic algorithms were mostly used for mining classification rules in large data sets. Patil et al. proposed a hybrid technique combining both genetic algorithm and decision tree to generate an optimized decision tree thus improving the efficiency and performance of computation. An effective feature and instance selection for supervised classification based on genetic algorithm was developed for high dimensional data.

3.3 Partitioning based clustering algorithms

In partitioning based algorithms, the large data sets are divided into a number of partitions, where each partition represents a cluster. K-means is one such partitioning based method to divide large data sets into number of clusters. Fuzzy- CMeans is a partition based clustering algorithm based on Kmeans to divide big data into several clusters

3.4 Hierarchical based clustering algorithms

In hierarchical based algorithms large data are organized in a hierarchical manner based on the medium of proximity. The initial or root cluster gradually divides into several clusters. It follows a top down or bottom up strategy to represent the clusters. Birch algorithm is one such algorithm based on hierarchical clustering. To handle streaming data in real time, a novel algorithm for extracting semantic content were defined in Hierarchical clustering for concept mining. This

algorithm was designed to be implemented in hardware, to handle data at very high rates. After that the techniques of self-organizing feature map (SOM) networks and learning vector quantization (LVQ) networks were discussed in Hierarchical Artificial Neural Networks for Recognizing High Similar Large Data Sets. SOM consumes input in an unsupervised manner whereas LVQ in supervised manner. It subdivides large data sets into smaller ones thus improving the overall computation time needed to process the large data set.

3.5 Density based clustering algorithms

In density based algorithms clusters are formed based on the data objects regions of density, connectivity and boundary. A cluster grows in any direction based on the density growth. DENCLUE is one such algorithm based on density based clustering.

3.6 Grid based clustering algorithms

In grid base algorithms space of data objects are divided into number of grids for fast processing. OptiGrid algorithm is one such algorithm based on optimal grid partitioning.

3.7 Model based clustering algorithms

In model based clustering algorithms clustering is mainly performed by probability distribution. Expectation- Maximization is one such model based algorithm to estimate the maximum likelihood parameters of statistical
models.

4.Application area

- Future Healthcare
- Market Basket Analysis
- Manufacturing Engineering
- Fraud Detection
- Financial Banking

5.Challenges of big data

5.1 Building a global unifying theory of big data mining

Many techniques are designed for performing classification or clustering individually, but there is no theoretical framework that unifies different tasks such as classification, clustering and association rules and so on. Therefore, building a global unifying theory for mining big data is an active research area.

5.2 Scaling up to meet the growing needs of large data sets

In order to meet the growing demands of data, we need to scale up both in terms of capacity and performance measures effortlessly. Big data needs more capacity, scalability, and efficient processing capabilities without increasing the resource demands. In traditional systems, storage architectures were designed in such a way to scale up with the growing needs of data. But it really affects the performance capacity of the storage systems. So organizations dealing with big data should design an optimal storage architecture which offers the features such as scalability, high performance, high efficiency, operational simplicity, interoperability and so on to manage growth.

5.3 Building efficient big data mining platform

To handle big data and its characteristics, an efficient big data Processing and computing framework is needed. Traditional data mining algorithms only needed all the data to be loaded into the main memory and perform the operation of data mining. In medium scale data processing, parallel computing is used with limited number of processors. As big data applications are characterized by autonomous sources and decentralized controls, consolidating distributed data sources to a centralized node for mining is systematically discouraging due to the potential transmission cost and privacy issues. So building an efficient platform to mine big data is essential.

5.4 Building efficient mining algorithms/models for big data

With the exponential growth of data, traditional data mining algorithms have been unable to meet large data processing needs. In order to deal with big data, an efficient model that deals with cost effective computation of huge, heterogeneous, sparse, incomplete, complex data are needed. The main drawbacks of big data mining algorithms are lack of user-friendly interaction support, quality and performance. Data mining algorithms usually needs scanning of entire data for obtaining perfect statistics and there may require intensive computing. Therefore, it is essential to improve the efficiency and performance of data mining algorithms to handle big data.

5.5 Maintaining security, trust and data integrity

Security is a major concern with big data. In order to ensure security, organizations need to establish policies, which are self-configurable. Another major issue in the field is trust of data sources which are not well – known and not at all verifiable. Data Integrity should be maintained by adopting the best practices in the industry.

6.what can be done

As we have seen computing platforms have different application and they used to simplify different tasks. Platform for managing the various data sources including data management, access, programming models, schedulers, security etc. The platform includes various tools for accessing other data platforms using streaming, web services, APIs. Other data platforms include data services from relational data stores, google data, social networking etc.

7.Conclusion

We are living in a digital world of big data where massive amounts of heterogeneous, autonomous, complex and evolving data sets are constantly generated at unprecedented scale. In this paper, an overview of big data along with it types, basic algorithms are discussed. This paper reviews about the various big data mining platforms and algorithms. To support big data mining, high-performance computing platforms are required. It is understood that interestingness of discovered patterns, developing a global unifying theory, building efficient mining platforms or algorithms, privacy preserving, security, trust and data integrity are the major challenging issues in the current big data mining scenario. It is known that big data mining is an emerging trend in all science and engineering domains and also a promising research area. In spite of the limited work done on big data mining so far, it is believed that much work is required to overcome its challenges related to the above mentioned issues.

Reference

Singh and Reddy Journal of Big Data 2014
http://www.journalofbigdata.com/content