ISSN:0975-9646

Big Data-Concepts, Tools and Foresight

Ravi Shaw[1], Samiddha Mukherjee[2], Nilanjan Haldar[3], Satyasaran Changdar[4]

[1,2,3,4] *Department of Information Technology, Institute of Engineering & Management, Kolkata, India*

***Abstract*--- The term, 'Big Data' in itself is self-explanatory-colossal amounts of data that cannot be handled by traditional data-handling techniques. Big Data is still in its infancy, and in the following literature survey we try to elaborate the concepts of it as conspicuously as possible. It commences with the concept of the subject in itself along with its properties and the two general approaches of dealing with it. The employment of diverse frameworks to deal with copious amounts of information such as Hadoop, MapReduce, DryadLINQ, SCOPE, and Jaql has been put across finely. To conclude the paper we have enlisted some of our own ideas giving a foresight on what could be a probable solution in solving the Big Data problem in near future. Throughout the course of this piece of literature, the authors intend to throw light upon the notions in the most palpable fashion incorporating in text several use-cases and illustrations.**

***Keyword*--- Big Data, 3 V's, Hadoop, Pig, Hive, Hbase, DryadLINQ, SCOPE, Jaql.**

I. INTRODUCTION

According to Eron Kelly, general manager at Microsoft SQL Server - "The amount of data produced in the next five years will be greater than the previous 5,000 years". Big Data involves deriving new insight from previously untouched data and integrating that insight into business operations. It is an upcoming field, which with passing time, will gain utmost importance and will be a pre-requisite for managing business, government and individual interests. The need of techniques to handle such massive amounts of data is growing by the hour as the world becomes more connected and new technological developments occurring by the minute result in the amount of data increasing at exponential rates. The practical definition of Big Data engulfs in it comprehensive coding skills, domain knowledge and statistics. Big Data is used in the fields of marketing, scientific research, customer interests amongst many others. This is a field which has been under extensive research in the present decade and several new algorithms and techniques are suggested every now and then. The World Bank organized the first WBG Big Data Innovation Challenge [1] in December 2014. This competition brought forward several out-of-the-box ideas such as using big data for climate smart agriculture and to predict poverty and user-focused Identification of Road Infrastructure Condition and Safety and so on and so forth.

Implementation of Big Data is a Herculean task- one involving handling data of large volume, velocity and variety. For the practical implementation of big data, Hadoop is the most commonly used software. Hadoop is not one single application but a collection of several software apps like HDFS, Pig, Hive, HBASE etc. as explained below. This literature survey deals with the applications, challenges and software implementations of big data.

What is Big Data?

"Big Data" is a term encompassing the use of techniques to capture, process, analyze and visualize potentially large datasets in a reasonable timeframe not accessible to standard IT technologies. By extension, the platform, tools and software used for this purpose are collectively called "Big Data technologies". [2]

In general, Big Data refers to data which due to its size, speed and format cannot be easily stored, manipulated or analyzed using traditional data handling algorithms and techniques. "*Every day, we create 2.5 quintillion bytes of data — so much that 90% of the data in the world today has been created in the last two years alone. This data comes from everywhere: sensors used to gather climate information, posts to social media sites, digital pictures and videos, purchase transaction records, and cell phone GPS signals to name a few. This data is "big data.*"[3]. The following are minute examples which deal with data of large size, type and speed- Walmart handles more than 1 million customer transactions every hour, Facebook handles 40 billion photos from its user base, decoding the human genome originally took 10 years to process but now it can be achieved in one week, CERN's Large Hydron Collider (LHC) generates 15 PB a year.

One way for describing Big Data is by looking at the 3 V's – volume, velocity and variety. Gartner analyst, Doug Laney [4] introduced the famous 3 V's concept back in a 2001 MetaGroup research publication, '*3D data management: Controlling Data Volume, Variety and Velocity*'. These three V's are the driving dimensions of Big Data quantification.

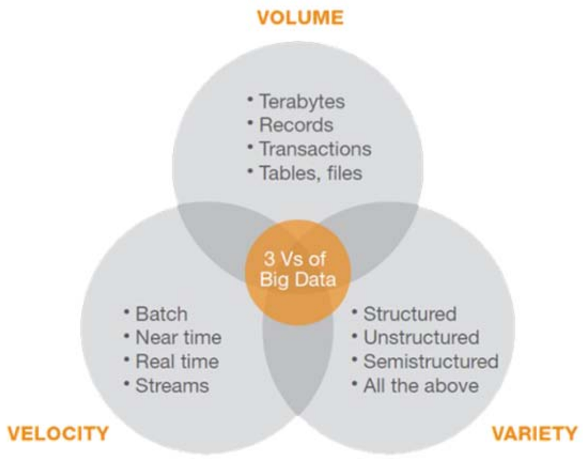

Fig. 1 schematic representation of the 3V's [5] of Big Data

1. *Volume* - Refers to the massive amount of data that is generated with every passing second. There are many factors which contribute to the increase in data volume. The size of available data has been increasing, a text file is of a few kilobytes, a sound file of a few megabytes while a full length movie is of a few gigabytes. In the past, excessive amount of data volume was a storage issue. But with decreasing storage costs this problem has been tackled while other issues have emerged over the years. The old model of data generation/consumption was – few companies were generating the data and all of us were consuming the data. The new model – all of us are generating the data and all of us are consuming the data. This data comprises of structured data as well as unstructured data streaming from social media sites and the increasing amount of information from sensors. For example, billions of smartphones generate variety (structured as well as unstructured) of data which did not exist a decade ago. E-commerce in particular, has exploded data management challenges along all the three dimensions. Petabyte sets are common these days and Exabyte is not far away. Following are the few cases where standard processing approaches to problems will fail due to Big Data -

- Large Synoptic Survey Telescope (LSST): “Over 30 thousands gigabytes (30TB) of images will be generated every night during the decade –long LSST survey sky.” [6]
- There is a corollary to Parkinson’s Law that states: “Data expands to fill the space available for storage.”[7]
- This is no longer true since the data being generated will soon exceed all available storage space.[8]
- 31 hours of video are uploaded to YouTube every minute.[9]

2. *Velocity* – With upcoming sources of data such as social media sites and mobile applications, data is being generated very fast. Initially, companies used to analyze data using a batch process. This batch process works when the incoming data rate is slower than the batch processing rate and when the result is useful despite the delay. In this era of smartphones, data is now streaming into the server in real time, in a continuous fashion and the result is only useful if the processing rate is faster. According to Facebook, its data system processes 2.5 million pieces of content each day amounting to 500+ terabytes of data daily. Facebook generates 2.7 billion like actions per day and 300 million new photos are uploaded daily. Every second, on average, around 6,000 tweets [10] are tweeted on Twitter which corresponds to over 350,000 tweets sent per minute, 500 million tweets per day and around 200 billion tweets per year. Google now processes over 40,000 search queries every second on average which translates to over 1.2 trillion searches per year worldwide [11].

3. *Variety* – From excel tables and databases, data is now losing its structure to hundreds of formats. Pure text, photo, audio, video, web, GPS data, relational databases, documents, SMS, pdf, flash, etc... This is a decade of smartphones, so one no longer has control over the input data format. Structure cannot be imposed like in the past in order to keep control over the analysis. As new applications are introduced, new data formats come to life. Data generated can be of any type – structured, semi-structured or unstructured. Structured data is the most traditional type of data, for example text. Social media sites, sensors embedded in smartphones and satellites are few of the many sources from which unstructured data is generated. Google uses smartphones as sensors to determine traffic conditions [12].

So now the question is, do we need to have all 3Vs to have Big Data. or just one to have Big Data? It is true that if we have all the 3V’s we definitely have Big Data but even if any one of them is present, it is too much for the existing standard approaches to process the data in order to give the desired output in time. One must understand that Big Data means that one cannot use the standard approach, as a result of which it presents a number of special challenges which are briefly discussed in this paper.

So the question is, how does one deal with the Big Data problem? Among the approaches to deal with Big Data, the following two are the most widely implemented:

1. Divide and conquer using Hadoop
2. Brute force using an “appliance” such as the SAP HANA (HIGH- Performance Analytic Appliance).

For the brute force approach, a very powerful server with terabytes of memory is used to crunch the data as one unit. The data set is compressed in memory. For example, for a Twitter data flow, that is pure text, the compression ratio may reach up to 100:1.

In the divide and conquer approach, the huge data set is broken into smaller parts (HDFS) and processed (MapReduce) in a parallel fashion using thousands of servers. The course of this paper discusses this approach in detail.

II. TOOLS TO DEAL WITH BIG DATA

Hadoop:

Hadoop is an Apache open-source framework designed by Doug Cutting and Mike Cafarella in 2005, for storing and processing large amount of data using a collection of low cost common hardware. It is extremely reliable; scalable that can scale up to hundreds or even thousands of nodes and is highly fault tolerant. The primary uses of Hadoop are index web searches, email spam detection, prediction in business and financial sectors, research oriented fields, sentiment analysis of people using social networking sites, and analysis of unstructured data like log files, text, click streams, etc. Hadoop is preferable under the following conditions:

- Complex data that needs to be processed.
- Conversion of unstructured data into structured data.
- Cases where parallel processing is needed like genome sequencing.
- Volume of data becomes too huge to be processed by RAM.
- Machine learning and pattern identification.
- Where different types of personalized coding are required to fit the job.

Hadoop platform mainly contains the following components:

1) Hadoop Common: It contains libraries and utilities that support other Hadoop modules.
2) Hadoop Distributed File System (HDFS): It is a distributed file system that stores and retrieves data through different clusters and provides high accessibility and throughput to different application programs.
3) Hadoop Yarn: Its main purpose is to schedule jobs and manage resource among the clusters.
4) Hadoop Map-reduce: Provide parallel processing for large volumes of data.

Hadoop Distributed File System (HDFS):
Hadoop is a distributed file system and framework, that performs analysis of large volumes of data sets using MapReduce programs [13]. It is highly fault tolerant as it works on low cost hardware. It has a master slave architecture where a single node controls a large number of sub nodes or clusters. After receiving the data, it divides and distributes it to different nodes in a cluster which promotes parallel processing. Moreover, it places multiple copies of the same data on different nodes, such that, if one node fails then that piece of information can be found elsewhere.

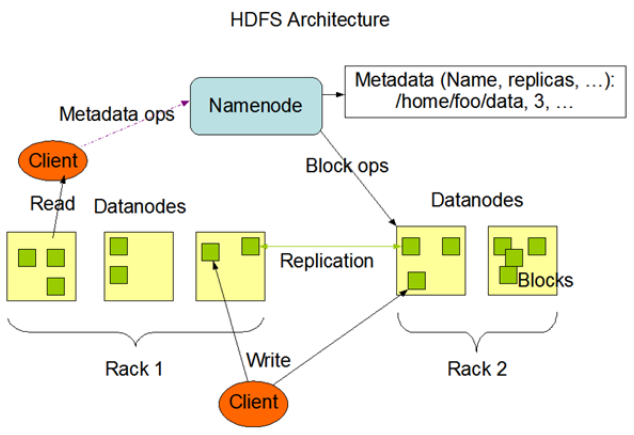

Fig. 2 The HDFS Architecture [14].

<u>*Name Node:*</u>A HDFS cluster has a name node (master node), which contains the metadata of the clusters under it. The name node manages each file's namespace, permissions, modifications and access, free space, number of active nodes, locations of files, what data is replicated in which data node. Each name node maintains a namespace tree and mapping of blocks to data nodes, holding the entire namespace image [15] in RAM, while having a cluster of thousands of data nodes and tens of thousands of HDFS clients. When a client wants to read a particular file it sends a request to the name node for the location of that particular data. Lastly each name node has a job tracker which manages resources, task lifecycle, determines where the fault has occurred, and allocates jobs to various task trackers that resides in data nodes.

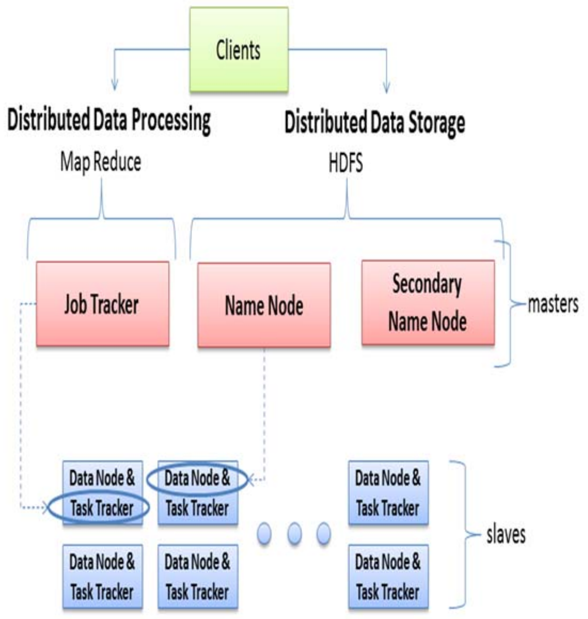

Fig. 3 Hadoop Client-Server Architecture [16].

<u>*Data Node:*</u>Data Nodes (slave nodes) are the ones, where actual data resides. Each data node has a task tracker that follows the instructions provided by the job tracker, and updating the job tracker with its progress periodically. During startup each data node performs a handshake to the master node through its task tracker, to verify the namespace id and version of the data node, and periodically sends a heartbeat message to the job tracker to say that it is alive and to keep the job tracker updated. If a name node does not receive a heartbeat from a particular data node for more than a predetermined amount of time, it considers the data node to be inactive and initiates searching of replica of that data in some other data node in that particular cluster. The data nodes also interact with each other to copy, move and keep replicas of data among themselves.

Map Reduce:
Map reduce[17] is a programming model used by Hadoop to make parallel processing of vast amount of data efficient, by breaking the entire task in two major process: mapping & reducing.
In mapping, the master node reads the huge amount of input, divides it into smaller jobs and finally distributing it into different clusters. In the meantime, the mapper also generates some intermediate results for reducers. The input to the mapper is generally given in the form of a key and a value (<key>, <value>) which is then transformed in to another key, value. Sometimes the output can have multiple entries of same key.
In reducing part, the master node records the outputs of each sub-node and by shuffling and combining it generates a new list of reduced output.

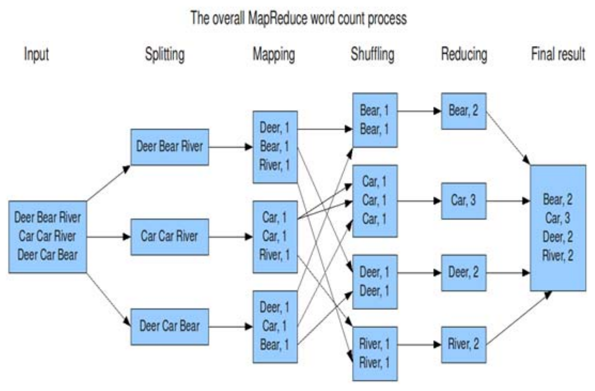

Fig. 4 Word Count using MapReduce [18].

Some popular MapReduce wrappers are:

1. *Pig:*

Pig was originally developed at Yahoo! Research around 2006 for creating and executing MapReduce jobs on very large data sets. In 2007, it was moved into Apache Software Foundation. Pig was developed to allow people using Hadoop to focus more on analyzing large data sets and spends less time on writing mapper and reducer programs. PIG is a scripting language that simplifies creation of the code that can run on Hadoop. According to *Alan Gates*, author of the book "Programming Pig"[19] – '*Pig provides a higher level of abstraction for data users, giving them access to the power and flexibility of Hadoop without requiring them to write extensive data-processing applications in low-level Java code.*' It provides an engine for executing data flows in parallel on Hadoop. It includes a language, Pig *Latin*, for expressing these data flows. Pig Latin, includes operators for many of the traditional data operations e.g. join, sort, filter, etc. as well as the ability for users to develop their own functions for reading, processing, and writing data. Pig is an Apache open source project. To run the Pig on machine or in Hadoop cluster, one must download and install it. One needs to download the Pig Package from Apache. It comes packaged with all of the JAR files needed to run on Pig.

Advantages:

- Programmers using Hadoop need not have to write mapper and reducer function, thereby decreased the development time. This is the biggest advantage.
- Anyone who does not know how to write perfect map-reduce or SQL for that matter could pick up and can write map-reduce jobs, therefore learning curve is not steep.
- Speaking of User defined functions (UDF), one could write their own user defined functions in Python.
- Enjoys everything that Hadoop offers parallelization, fault-tolerance with many relational database features.
- It is most effective tool for unstructured and messy large datasets. It is one of the best tool to make large unstructured data to structured.

Disadvantages:

- One of the major issues with Pig is that the errors that produce due to UDFS (Python) are not helpful at all. Even if the problem is related to syntax or type error it gives exec error.
- Even though it is has been around for quite some time, it is still in the development phase and is not mature.
- The commands are not executed unless either one dumps or stores an intermediate or final result. This increases the iteration between debugging and resolving the issue.
- Someone who knows SQL queries could write Hive queries but for writing queries in Pig one needs to learn the Pig syntax.

2. *Hive:*

Apache Hive is a data warehouse system for Apache Hadoop [20]. It has been widely used in organization to manage and process large volumes of data, such as eBay, Facebook, LinkedIn, Spotify, Toabao, Tenant, and Yahoo!. As an open source project, Hive has a strong technical development team working with diverse users and organizations. Hive team developer has solved more than 3000 issues in the recent years. With this rapid development over the years, Hive has been significantly updated by new innovations and research since the original Hive Paper [21] was published four years ago. Hive is used in 90% MapReduce programs in Facebook.

The following figure provides an overview of Hive's Architecture [21][18]-

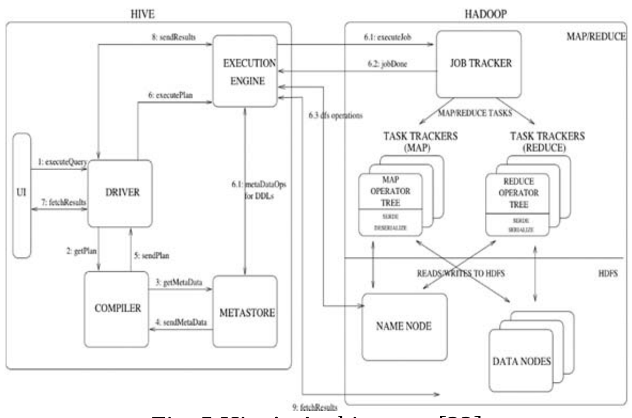

Fig. 5 Hive's Architecture [22]

Above Figure shows the major components of Hive and its interactions with Hadoop. As shown in the figure, the main components of Hive are:

- UI – The users submit their queries through the interface, which is used for the other system operations too. As of 2011, the system had a command line interface and a web based GUI was being developed.
- Driver – This is the component which receives the queries. This component implements the notion of session handles and provides execute and fetch APIs modeled on JDBC/ODBC interfaces.
- Compiler – The compiler parses the query, does semantic analysis on the different query blocks and query expressions and eventually generates an execution plan with the help of the table and partition metadata looked up from the metastore.
- Metastore- The metadata stores all the structure information of the various tables and partitions in the

warehouse including column and column type information, the serializers and deserializers necessary to read and write data and the corresponding HDFS files where the data is stored.

- Execution Engine – The plan which is created by the compiler is executed by the execution engine. The plan is a DAG of stages. The execution engine monitors the dependencies between these different stages of the plan and then executes these stages on the appropriate system components.
- HiveQL is an SQL - like query language for Hive. It mostly mimics SQL syntax for creation of tables, loading data into tables and querying the tables. HiveQL also allows users to embed their custom map-reduced scripts.

3. *HBase:*

HBase is an open source, distributed sorted map modeled after Google's BigTable [23]. HBase become first usable in 2007. Hbase is a part of Hadoop. HBase is written in Java. HDFS is a distributed file system that is well suited for the storage of large files. HBase, on the other hand, is built on top of HDFS and provides fast record lookups (and updates) for large tables. HBase uses HDFS for storage. It is column – oriented database. Each table consists of rows, each which has a primary key (row key). Each row may have any number of columns. Table schema only defines Column families (column family can have any number of columns).In HBase each cell value has a timestamp. Different sets of columns may have different priorities. Just as HDFS has a NameNode and slave nodes, and MapReduce has JobTracker and TaskTracker slaves, HBase is built on similar concepts. In HBase a master node manages the cluster and region servers store portions of the tables and perform the work on the data [24]. HBase is also sensitive to the loss of its master node.

Facebook currently uses Apache Hbase for FM as HBase comes with very good scalability and performance for workload and is a simpler consistency model. Facebook Messages (FM) [25] is a messaging system that enables Facebook users to send chat and email-like messages to one another, it is quite popular, handling millions of messages each day. FM stores its information within HBase and serves as an excellent case study. Users of FM interact with a web layer, which is backed by an application cluster, which in turn stores data in a separate HBase cluster. The application cluster executes FM-specific logic and caches HBase rows while HBase itself is responsible for persisting most data. Large objects (*e.g.*, message attachments) are an exception; these are stored in Haystack [26] because HBase is inefficient for large data. This design applies Lampson's advice to "handle normal and worst case separately" [27].

While all this approaches were used to deal with Big Data problem, Microsoft came up with other technologies to take on Google's Map-Reduce which include:

DryadLINQ:

DryadLINQ is a simple, powerful, and elegant programming environment for writing large-scale data parallel applications running on large PC clusters [28]. The goal of DryadLINQ combines two important pieces of Microsoft technology: the Dryad distributed execution engine and the .NET Language Integrated Query (LINQ).

Dryad provides a reliable, distributed computing on thousands of several for large-scale data parallel applications. LINQ enables developers to write and debug their applications in a SQL-like query language, relying on the entire .NET library and using Visual Studio. The term LINQ [29] refers to a set of .NET constructs for manipulating sets and sequences of data items.

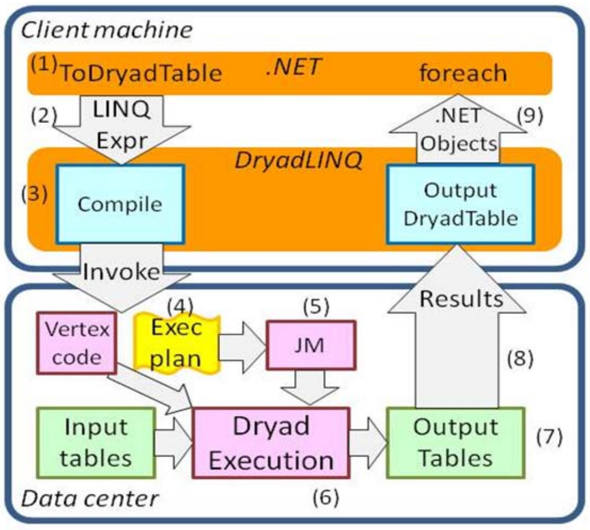

Fig. 6 The flow of execution when a program is executed by DryadLINQ [30].

DryadLINQ has the following features [31]:

- *Declarative programming:* The best features of SQL, functional programming and .Net are combined so that computations can be expressed in high-level language.
- *Automatic parallelization:* From sequential declarative code the DryadLINQ compiler generates highly parallel query plans spanning large computer clusters. DryadLINQ also exploits multi-core parallelism on each machine.
- *Integration with Visual Studio:* Programmers in DryadLINQ take advantage of the comprehensive VS set of tools: Intellisense, code refactoring, integrated debugging, build, source code management.
- *Integration with .Net:* All .Net libraries, Visual Basic and dynamic languages are included.
- *Job graph optimizations:*This optimization is done in static and dynamic as follows-
  - ➢ Static: A rich set of term-rewriting query optimization rules is applied to the query plan, optimizing locality and improving performance.
  - ➢ Dynamic: Run-time query plan optimizations automatically adapt the plan taking into account the statistics of the data set processed.

SCOPE:

SCOPE stands for "Structured Computations Optimized for Parallel Execution". SCOPE/COSMOS provides storage and computation for Back-End Batch data analysis of Microsoft's BING application. It is hybrid parallel database and MapReduce system. SCOPE is a SQL-like declarative language, which is highly extensible and flexible. SCOPE is fully integrated with .NET framework. Cosmos Storage System append-only distributed file system for storing petabytes of data. It is optimized for sequential I/O. The inside the storage system is compressed and replicated.
System complexity and parallelism are hidden from the users.

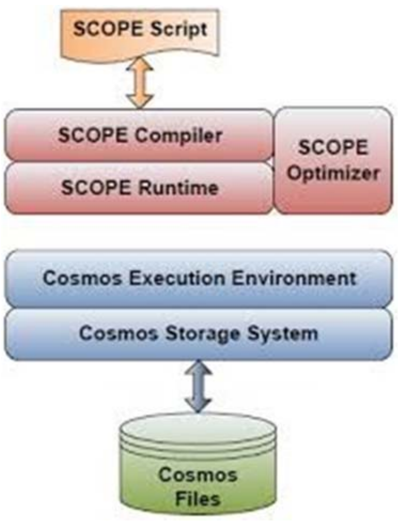

Fig. 7 Working of SCOPE [32].

SCOPE/Cosmos is a hybrid system of MapReduce and traditional parallel database. It is extensively used in cloud-scale data centers at Microsoft Bing. SCOPE seamlessly integrates optimization of both serial and parallel plans into a single uniform framework.

Jaql:

To address the needs of enterprise customers, IBM recently released InfoSphere BigInsights [33] and Cognos Consumer Insights [34] (CCI). Both BigInsights and CCI are based on Hadoop and include analysis flows that are much more diverse than those typically found in a SQL warehouse. Among other things, there are analysis flows to annotate and index intranet crawls [35], clean and integrate financial data [36,37], and predict consumer behavior [38].
Jaql, which is a declarative scripting language used in both BigInsights an CCI to analyze large semi-structured datasets in parallel on Hadoop. Jaql's data model is based on JSON, which is a simple format and standard (RFC 4217) for semi structured data. The main goal of Jaql is to simplify the task of writing analysis flows by allowing developers to work at a higher level of abstraction than low-level MapReduce programs. Jaql consists of a scripting language and compiler, as well as a runtime component for Hadoop, but we will refer to all three as simply Jaql. Jaql's design has been influenced by Pig [39], Hive [40], DryadLINQ [41], among others, but has a unique focus on the following combination of core features: (1) a flexible data model, (2) reusable and modular scripts, (3) the ability to specify scripts at varying levels of abstraction, referred to as physical transparency, and (4) scalability.

III. AUTHORS' PROPHECY

The three most important differentiators of Big Data as discussed previously, are volume, velocity and variety. However in the upcoming era, the more competitive differentiator will become velocity rather than volume. IBM's sixth annual analytics study in 2014 emphasized on this aspect apart from several others.
Like Big Data, another of the popular buzzwords today is Internet of Things. Everything that surrounds our life and makes it easier, be it our vehicles or our homes or any other major appliance that we use in our day to day functioning are now at all points of time connected to the Internet thus creating a sort of a network which is what is called the Internet of Things. These vast number of interconnected devices generate incessant streams of data wherein lies the scope of Big Data analytics. Like the smartphone which became a surreal reality only in the last decade, Internet of Things is the next big superstar of our world. We believe that once IoT is realized, analyzing the massive quantities of data which will be generated due to this, will be a challenge in itself and the present available solutions will not be sufficient.
As we have discussed earlier, there are two popular methods that till today are used to deal with Big Data. One of which is the Divide and Conquer method as used by Apache Hadoop whilst the other one is the Brute Force method where large chunks of data is crunched into smaller data which is employed by SAP in its trademark 'appliance' SAP-HANA. While the two methods are as of now being implemented individually, they are not sufficient for the humungous amount of data that will be generated every second in the next few coming years.
Having made such an exhaustive research on the present scenario of Big Data, we suggest the following probable solution-
Though individually applying any of the above two solutions may not solve the upcoming problem, an amalgamation of the two processes in a cognizant manner may be helpful in the field. However there are a few postulates that needs to be considered- At present the brute force method can only compress a few terabytes of data into a few gigabytes. However the rate at which data will be generated, it is essential that the compressing techniques be made more competent and capable of scaling-down data in a much larger ratio. We believe that the velocity of such compression must be bettered by leaps as compared to the present rate. The algorithms that we currently possess knowledge of, to implement the divide and conquer approach may be efficient for the present requirements. However, as the requirements increase exponentially, we suggest developing more potent algorithms which will optimize the data according to the project.
As the data pours in, we suggest that the data first be compressed to a considerable amount using the brute force method. After the compression has been executed

efficiently, the data should be processed using the divide and conquer approach to produce required results. Implementing this will considerably reduce the weight of the data generated and will produce optimized results as required.

Though there are a few ideas like this floating around but given the massive amounts of data which will generated in the upcoming days because of addition of IoT, and scientific and technological advancements, they are not at par with the upcoming requirements. This integration of the two processes needs to be made far more advanced in comparison to already available technology. We emphasize that this amalgamation be given utmost importance with respect to developing the technology and making it available comprehensibly to one and all with simplified architecture as it is the need of the hour.

IV. CONCLUSIONS

This literature survey traces the course of Big Data from its initiation to the present state. It elucidates the concepts of big data, its applications and popular tools to deal with Big Data. It discusses elaborately the challenges faced by Big Data and the future opportunities that could be harnessed. Big Data is an evolving field, where much of the research is yet to be done. At present, the most common implementation of this field has been via Hadoop. However with the exponentially proliferating amount of data, more pragmatic solutions to this issue needs to be fostered.

In the final section of the research report, we have hinted on a possible key to solving big data's big problem in the impending future. With the copious amounts of data which will be generated once we enter into the era of Internet of Things hook, line and sinker and progress in the fields of science and technology, finding an appropriate solution is a matter of immediate concern. We have suggested amalgamating the two most popular approaches to Big Data- the brute force method and the divide and conquer approach. Conjoining the two techniques to produce an approach which at first compresses the data and then applies the divide and conquer method to it can help us tackle the massive amounts of data which will be generated by a considerable amount.

To conclude, Peter Sondergaard, Senior Vice President of Gartner Research famously stated, "*Information is the oil of the 21st century and analytics is the combustion engine.*"

REFERENCES

[1] http://blogs.worldbank.org/voices/meet-winners-and-finalists-first-wbg-big-data-innovation-challenge

[2] Grand Challenge: Applying Regulatory Science and Big Data to Improve Medical Device Innovation, Arthur G. Erdman∗, Daniel F. Keefe, Senior Member, IEEE, and Randall Schiestl, IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, VOL. 19, NO. 3, MARCH 2013

[3] Apache Hive. Available at http://hive.apache.org

[4] http://www.forbes.com/sites/gartnergroup/2013/03/27/gartners-big-data-definition-consists-of-three-parts-not-to-be-confused-with-three-vs/

[5] http://www.exist.com/wp-content/uploads/2014/10/3Vsbigdata.png

[6] http://lsst.org/lsst/google

[7] http://en.wikipedia.org/wiki/Parkinson's_law

[8] http://www.economist.com/node/11412343

[9] http://www.youtube.com/t/press_statistics/?hl=en

[10] http://www.internetlivestats.com/twitter-statistics/

[11] http://www.internetlivestats.com/google-search-statistics/

[12] http://www.wired.com/autopia/2011/03/cell-phone-networks-and-the-fu...

[13] Jeffrey Dean and Sanjay Ghemawat: "MapReduce: Simplified Data Processing on Large Clusters". Proc. Sixth Symposium on Operating System Design and Implementation 2004.

[14] http://hadoop.apache.org/docs/r1.2.1/images/hdfsarchitecture.gif

[15] 8.2.2 (http://www.aosabook.org/en/hdfs.html)

[16] http://bradhedlund.s3.amazonaws.com/2011/hadoop-network-intro/Hadoop-Server-Roles-s.png

[17] Jefry Dean and Sanjay Ghemwat, MapReduce:A Flexible Data Processing Tool, Communications of the ACM, Volume 53, Issuse.1,January 2010, pp 31-36.

[18]http://www.rabidgremlin.com/data20/MapReduceWordCountOverview1.png

[19] Alan Gates, Programming Pig

[20] http://hadoop.apache.org/.

[21] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka,N. Zhang, S. Anthony, H. Liu, and R. Murthy. Hive – A Petabyte Scale Data Warehouse Using Hadoop. In ICDE,2010.

[22] https://cwiki.apache.org/confluence/display/Hive/Design

[23] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh,Deborah A.Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert Gruber. Bigtable: A Distributed Storage Systemfor Structured Data. In Proceedings of the 7th Symposiumon Operating Systems Design and Implementation (OSDI '06),pages 205–218, Seattle, Washington, November 2006.

[24] Apache HBase. Available at http://hbase.apache.org

[25] Kannan Muthukkaruppan. Storage Infrastructure Behind Facebook Messages. In Proceedings of International Workshop on High Performance Transaction Systems (HPTS '11), Pacific Grove, California, October 2011.

[26] Jason Sobel. Needle in a haystack: Efficient storage of billions of photos. http://www.flowgram.com/p/2qi3k8eicrfgkv, June 2008.

[27] Butler W. Lampson. Hints for Computer System Design. In Proceedings of the 9th ACMSymposium on Operating System Principles (SOSP '83), pages 33–48, Bretton Woods, New Hampshire, October 1983.

[28] http://research.microsoft.com/en-us/projects/DryadLINQ/

[29]https://cs.uwaterloo.ca/~kmsalem/courses/CS848W10/presentations/Aluc-Scope.pdf

[30]https://www.usenix.org/legacy/events/osdi08/tech/full_papers/yu_y/yu_y_html/systemoverview.jpg

[31] http://research.microsoft.com/en-us/projects/DryadLINQ/

[32]https://cs.uwaterloo.ca/~kmsalem/courses/CS848W10/presentations/Aluc-Scope.pdf

[33] Biginsights. http://www-01.ibm.com/software/data/infosphere/biginsights/

[34] Cognos Consumer Insight. http: //www-01.ibm.com/software/analytics/cognos/ analytic-applications/consumer-insight/.

[35] K. S. Beyer, V. Ercegovac, R. Krishnamurthy, S. Raghavan, et al. Towards a Scalable Enterprise Content Analytics Platform. IEEE Data Eng. Bull., 32(1):28–35, 2009.

[36] S. Balakrishnan, V. Chu, M. A. Hernandez, H. Ho, et al. ´ Midas: Integrating Public Financial Data. In SIGMOD, pages 1187–1190, 2010.

[37] A. Sala, C. Lin, and H. Ho. Midas for Government: Integration of Government Spending Data on Hadoop. In ICDEW, pages 122 –125, 2010.

[38] S. Das, Y. Sismanis, K. S. Beyer, R. Gemulla, et al. Ricardo: Integrating R and Hadoop. In SIGMOD, pages 987–998, 2010.

[39] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins. Pig Latin: A Not-So-Foreign Language for Data Processing. In SIGMOD, pages 1099–1110, 2008.

[40] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, et al. Hive - A Warehousing Solution Over a Map-Reduce Framework. PVLDB, 2(2):1216–1219, 2009

[41] Y. Yu, M. Isard, D. Fetterly, M. Budiu, et al. DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language. In OSDI, pages 1–14, 2008.