Inception of Big Data with Hadoop and Map Reduce

Prof. Yoginee Surendra Pethe
Department of Computer Application
Assistant professor, Shri Ramdeobaba College of Engineering & Management Nagpur, India.

Abstract-*Big Data is a term used to describe large collections of data that may be unstructured, and grow so large and quickly that it is difficult to manage with regular database or statistical tools. Therefore, Big Data solutions based on Hadoop and other analytics software are becoming more and more relevant.This massive amount of data can be analyzed by using Hadoop. Hadoop is an open source software project that enables the distributed processing of large data sets across clusters of commodity servers. It is designed to scale up from a single server to thousands of machines, with a very high degree of fault tolerance. The technologies used by big data application to handle the enormous data are Hadoop, Map Reduce, Apache Hive, No SQL and HPCC. In this paper I suggest various methods for furnishing to the problems in hand through Map Reduce framework over Hadoop Distributed File System (HDFS). Map Reduce is a Minimization technique which makes use of file indexing with mapping, sorting, shuffling and finally reducing. Map Reduce techniques have been studied in this paper which is implemented for Big Data analysis using HDFS.*

Keywords-Bigdata, Hadoop, Hadoop Components, HDFS, Mapreduce.

I. INTRODUCTION

With all the devices available today to collect data, such as RFID readers, microphones, cameras, sensors, and so on, we are seeing an explosion in data being collected worldwide.

The need of big data generated from the large companies like facebook, yahoo, Google, YouTube etc for the purpose of analysis of enormous amount of data also Google contains the large amount of information.

Big Data is a term used to describe large collections of data (also known as datasets) that may be unstructured, and grow so large and quickly that it is difficult to manage with regular database or statistical tools. Other interesting statistics providing examples of this data explosion are: There are more than 2 billion internet users in the world today, and in 2014 there will be 7.3 billion active cell phones. Twitter processes 7TB of data every day, and 500TB of data is processed by Facebook every day. Interestingly, approximately 80% of these data are unstructured. With this massive quantity of data, businesses need fast, reliable, deeper data insight. Therefore, Big Data solutions based on Hadoop and other analytics software are becoming more and more relevant. There are various technologies in the market from different vendors including Amazon, IBM, Microsoft, etc., to handle big data.

Imagine this scenario: You have 1GB of data that you need to process. The data is stored in a relational database on your desktop computer and this desktop computer has no problem handling this load. Then your company starts growing very quickly, and that data grows to10GB and then 100GB. You start to reach the limits of your current desktop computer. So you scale-up by investing in a larger computer, and you are then OK for a few more months. When your data grows to 10TB, and then 100TB, you are quickly approaching the limits of that computer. Moreover, you are now asked to feed your application with unstructured data coming from sources like Facebook, Twitter, RFID readers, sensors, and so on. Your management wants to derive information from both the relational data and the unstructured data and wants this information as soon as possible.

Hadoop is an Apache open source framework written in Java that allows distributed processing of large dataset across cluster of computers using simple programming model. It is a framework originally developed by Doug Cutting who named it after his son's toy elephant. Hadoop uses Google's MapReduce and Google File System technologies as its foundation. Hadoop creates cluster of machines and coordinates work among them. It is optimized to handle massive quantities of data which could be structured, unstructured or semi-structured, using commodity hardware, that is, relatively inexpensive computers. This massive parallel processing is done with great performance. Hadoop replicates its data across multiple computers, so that if one goes down, the data is processed on one of the replicated computers. It is a batch operation handling massive quantities of data, so the response time is not immediate. Hadoop is not suitable for OnLine Transaction Processing workloads where data is randomly accessed on structured data like a relational database. Also, Hadoop is not suitable for OnLine Analytical Processing or Decision Support System workloads where data is sequentially accessed on structured data like a relational database, to generate reports that provide business intelligence. Hadoop is used for Big Data. It complements OnLine Transaction Processing and OnLine Analytical

Processing. It is NOT a replacement for a relational database system.

Map Reduce by itself is capable for analyzing large distributed data sets; but due to the heterogeneity, velocity and volume of Big Data, it is a challenge for traditional data analysis and management tools. A problem with Big Data is that they use NoSQL and has no Data Description Language (DDL) and it supports transaction processing. Also, web-scale data is not universal and it is heterogeneous. For analysis of Big Data, database integration and cleaning is much harder than the traditional mining approaches. Parallel processing and distributed computing is becoming a standard procedure which are nearly non-existent in RDBMS. Map Reduce has following characteristics:

- It supports Parallel and distributed processing
- It is simple and its architecture is shared-nothing which has commodity diverse hardware (big cluster).
- Its functions are programmed in a high- level programming language (e.g. Java, Python)
- It is flexible.
- Query processing is done through NoSQL integrated in HDFS as Hive tool.

A unique challenge for researchers system and academicians is that the large datasets needs special processing systems. Map Reduce over HDFS gives Data Scientists the techniques through which analysis of Big Data can be done. HDFS is a distributed file system architecture which encompasses the original Google File System. Map Reduce jobs use efficient data processing techniques which can be applied in each of the phases of MapReduce namely Mapping, Combining, Shuffling, Indexing, Grouping and Reducing. All these techniques have been studied in this paper for implementation in Map Reduce tasks.

II. CHARACTERIZATION OF BIG DATA

Big Data is a term that refers to dataset whose volume (size), complexity and rate of growth (velocity) make them difficult to captured, managed, processed or analyzed by conventional technology and tools such as relational databases.

A. Volume

The name 'Big Data' itself is related to a size which is enormous. Size of data plays very crucial role in determining value out of data. Also, whether a particular data can actually be considered as a Big Data or not, is dependent upon volume of data. Hence, 'Volume' is one characteristic which needs to be considered while dealing with 'Big Data'. Volume refers to amount of data. Volume represent the size of the data how the data is large. The size of the data is represented in terabytes and petabytes.

B. Variety:

Variety makes the data too big. Variety refers to heterogeneous sources and the nature of data, both structured and unstructured. During earlier days, spreadsheets and databases were the only sources of data considered by most of the applications. Nowadays, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. is also being considered in the analysis applications. This variety of unstructured data poses certain issues for storage, mining and analyzing data.

C. Velocity:

The term 'velocity' refers to the speed of generation of data. How fast the data is generated and processed to meet the demands, determines real potential in the data. The data comes at high speed. Sometimes 1 minute is too late so big data is time sensitive. Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks and social media sites, sensors, mobile devices, etc. The flow of data is massive and continuous.

D. Value:

The potential value of Big data is huge. Value is main source for big data because it is important for businesses, IT infrastructure system to store large amount of values in database. Value deals with what value should come out which data. How big data will enable the user to get better result from data stored?

E. Veracity:

Veracity refers to noise, biases and abnormality when dealing with high volume, velocity and variety of data, the all of data are not going 100% correct and there will be dirty data. It describes the quality of data. Is the data noiseless or conflict free? Accuracy and completeness is concerned.

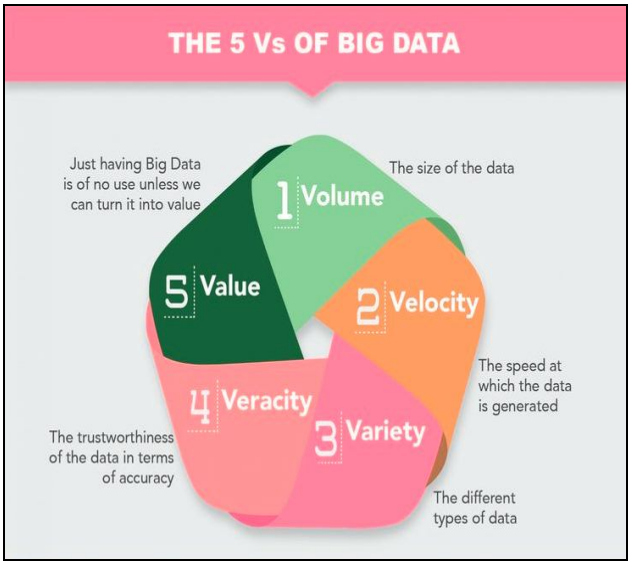

Fig.1. Characterization of Big Data

III. HADOOP AND HDFS

Hadoop is generally ssen as having two parts: a file system (Hadoop Distributed File System) and a programming paradigm (MapReduce). One of the key component of Hadoop is the redundancy built into the environment. Not only is the data redundancy stored in multiple places across the cluster but programming model is such that failures are expected and are resolved automatically by running portions of the program on various servers in the cluster. Hadoop is a scalable, fault-tolerant Virtual Grid operating system architecture for data storage and processing. It runs on commodity hardware, it uses HDFS which is fault-tolerant high-bandwidth clustered storage architecture. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage Hadoop consists of two component.

It runs MapReduce for distributed data processing and is works with structured and unstructured data. HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file. An application can specify the number of replicas of a file. The replication factor can be specified at file creation time and can be changed later. Files in HDFS are writing-once and have strictly one writer at any time. The NameNode makes all decisions regarding replication of blocks. It periodically receives a Heartbeat and a Blockreport from each of the DataNodes in the cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode.

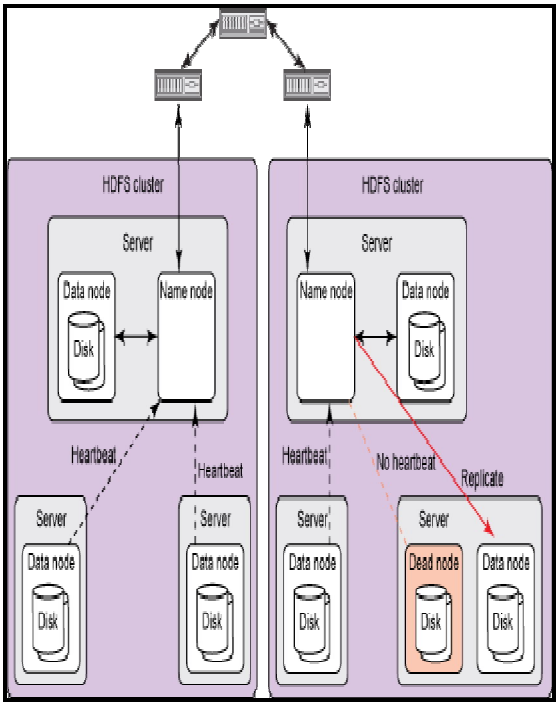

Fig.3. Architecture of Hadoop

A. HDFS (Hadoop Distributed File System)

HDFS is a file system designed for storing very large files with streaming data access pattern, running clusters on commodity hardware. HDFS manages storage on the cluster by breaking incoming files into pieces called ‘blocks’ and storing each blocks redundantly across the pool of the server. HDFS stores three copies of each file by copying each piece to three different servers. Size of each block is 64MB. HDFS architecture is broadly divided into following three nodes which are Name Node, Data Node, HDFS Clients/Edge Node.

- Name Node

It is centrally placed node, which contains information about Hadoop file system. The main task of name node is that it records all the metadata & attributes and specific locations of files & data blocks in the data nodes. Name node acts as the master node as it stores all the information about the system .and provides information which is newly added, modified and removed from data nodes.

- Data Node

It works as slave node. Hadoop environment may contain more than one data nodes based on capacity and performance. A data node performs two main tasks storing a block in HDFS and acts as the platform for running jobs.

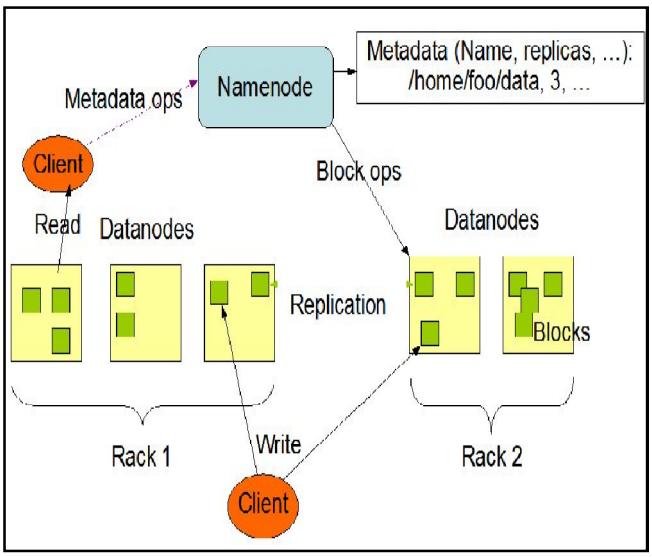

Fig.4. HDFS Architecture

- HDFS Clients/Edge node

HDFS Clients sometimes also know as Edge node. It acts as linker between name node and data nodes. Hadoop cluster there is only one client but there are also many depending upon performance needs.

B. MapReduce Framework

Map Reduce is the heart of Hadoop. MapReduce is defined as a programming model for processing and generating large sets of data. There are two phases in MapReduce, the "Map" phase and the "Reduce" phase. The system splits the input data into multiple chunks, each of which is assigned a map task that can process the data in parallel. Each map task reads the input as a set of (key, value) pairs and produces a transformed set of (key, value) pairs as the output. The framework shuffles and sorts output of the map tasks, sending the intermediate (key, value) pairs to reduce task which groups them into final results.

IV. HADOOP COMPONENTS

A. HBases

Hbase is distributed column oriented database where as HDFS is file system. But it is built on top of HDFS system. HBase is a management system that is open-source, versioned, and distributed based on the BigTable of Google. It is Non-relational, distributed database system written in Java. It runs on the top of HDFS. It can serve as the input and output for the MapReduce. For example, read and write operations involve all rows but only a small subset of all columns.

B. Avro:

Avro is data serialization format which brings data interoperability among multiple components of apache Hadoop. Most of the components in Hadoop started supporting Avro data format. It works with basic premise of data produced by component should be readily consumed by other component. Avro has following features: Rich data types, Fast and compact serialization, Support many programming languages like java, Python.

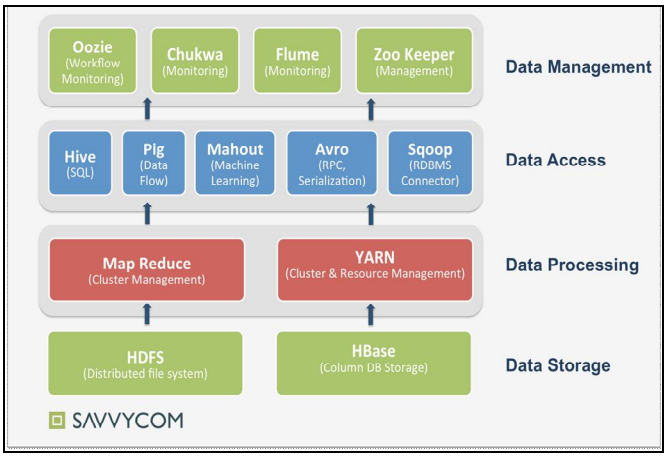

Fig.5. Hadoop Components

C. Pig:

Pig is platform for big data analysis and processing. Pig adds one more level abstraction in data processing and it makes writing and maintaining data processing jobs very easy. Pig. can process tera bytes of data with half dozen lines of code.

D. Hive:

Hive is data warehouse software. It manages large datasets in distributed computers. It allows writing SQL like queries to process and analyzing the big data stored in HDFS.

It provides the SQL interface and relational model. Hive infrastructure is built on the top of Hadoop that help in providing summarization, query and analysis.

E. Sqoop:

Sqoop is tool which can be used to transfer the data from relational database environments like oracle, mysql and postgresql into hadoop environment. Sqoop is a command-line interface platform that is used for transferring data between relational databases and Hadoop.

F. Zookeeper:

Zookeeper is a distributed coordination and governing service for hadoop cluster. It is a centralized service that provides distributed synchronization and providing group services and maintains the configuration information etc. In hadoop this will be useful to track if particular node is down and plan necessary communication protocol around node failure.

G. Mahout:

Mahout is a library for machine-learning and data mining. It is divided into four main groups: collective filtering, categorization, clustering, and mining of parallel frequent patterns. The Mahout library belongs to the subset that can be executed in a distributed mode and can be executed by MapReduce.

V. MAP REDUCE

MapReduce is a programming model for processing large-scale datasets in computer clusters. The MapReduce programming model consists of two functions, map() and reduce(). Users can implement their own processing logic by specifying a customized map() and reduce() function. The map() function takes an input key/value pair and produces a list of intermediate key/value pairs. The MapReduce runtime system groups together all intermediate pairs based on the intermediate keys and passes them to reduce() function for producing the final results.

Map

(in_key,in_value)>list(out_key,intermediate_value)**Reduce**

(out_key,list(intermediate_value))>list(out_value)

The signatures of map() and reduce() are as follows :
map (k1,v1) ! list(k2,v2)and reduce (k2,list(v2)) !list(v2)

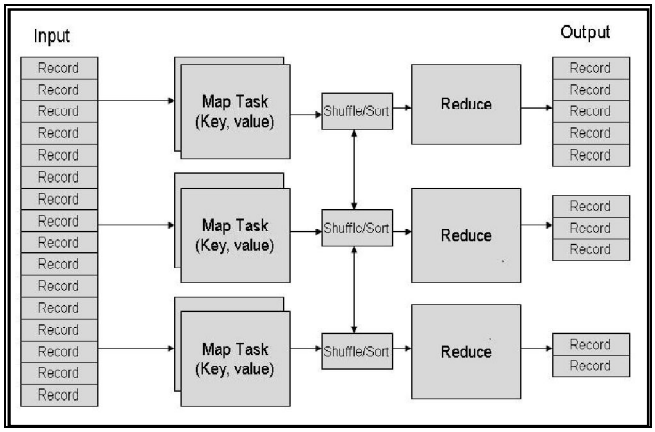

Fig.6. Map Reduce Architecture and Working

Large scale data processing is a difficult task, managing hundreds or thousands of processors and managing parallelization and distributed environments makes is more difficult. Map Reduce provides solution to the mentioned issues, as is supports distributed and parallel I/O scheduling, it is fault tolerant and supports scalability and has inbuilt processes forstatus and monitoring of heterogeneous and large datasets as in Big Data.

A. MapReduceComponents

- **NameNode**– managesHDFSmetadata,doesn'tdealwithfilesdirectly.
- **DataNode**–storesblocksofHDFS– defaultreplicationlevelforeachblock:3.
- **JobTracker**– schedules,allocatesandmonitorsjobexecutiononslaves– TaskTrackers.
- **TaskTracker**–runs MapReduceoperations. HadoopMapReducecomesbundledwithalibraryofgenerallyusefulmappers,reducers,andpartitioners.

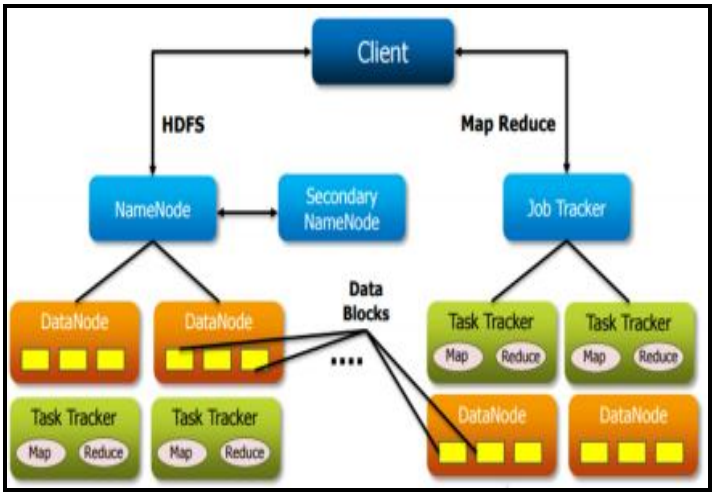

Fig.7. Map Reduce working through Master /Slave

B. Map Reduce Techniques

- **Prepare the Map() input**–the "Map Reduce system" designates Map processors, assigns the K1 input key value each processor would work on, and provides that processor with all the input data associated with that key value.
- **Run the user-provided Map()code**–Map() is run exactly once for each K1 key value, generating output organized by key values K2.
- **"Shuffle" Map output to the Reduce processors**–the Map Reduce system designates Reduce processors, assigns the K2 key value each processor should work on, and provides that processor with all the Map-generated data associated with that key value.
- **Run the user-provided Reduce() code**–Reduce () is run exactly once for each K2 key value produced by the Map step.
- **Produce the final output**–the Map Reduce system collects all the Reduce output, and sorts it by K2 to produce the final outcome.

Sample Program

```
function map(String name, String document):
 // name: document name
 // document: document contents
 for each word w in document:
   emit (w, 1)
function reduce(String word, Iterator partialCounts):
 // word: a word
 // partialCounts: a list of aggregated partial counts
 sum = 0
 for each pc in partialCounts:
   sum += ParseInt(pc)
 emit (word, sum)
```

VI. CONCLUSION

The paper describes the concept of Big Data along with Operational vs. Analytical Systems of Big Data. The paper also focuses on Big Data processing problems. These technical challenges must be addressed for efficient and fast processing of Big Data. In this paper I have tried to cover all the details of Hadoop, Hadoop components and future scope.

The need to process enormous quantities of data has never been greater. Not only are terabyte- and petabyte-scale datasets rapidly becoming commonplace, but there is consensus that great value lies buried in them, waiting to be unlocked by the right computational tools. Big Data analysis tools like Map Reduce over Hadoop and HDFS, promises to help organizations better understand their customers and the marketplace, hopefully leading to better business decisions and competitive advantages. For engineers building information processing tools and applications, large and heterogeneous datasets which are generating continuous flow of data, lead to more effective algorithms for a wide range of tasks, from machine translation to spam detection. MapReduce can be exploited to solve a variety of problems related to text processing.

REFERENCES

[1] Jens Dittrich, Stefan Richter and Stefan Schuh, " Efficient OR Hadoop: Why Not Both?," Datenbank-Spektrum, Volume 13, Issue 1 , pp 17-22

[2] Humbetov, S, "Data-Intensive Computing with Map-reduce and Hadoop," in Proc. 2012 Application of Information and Communication Technologies (AICT), IEEE ,6th International Conference pp. 5

[3] Hadoop Tutorial, Apache Software Foundation, 2014, Available: http://hadoop.apache.org/

[4] Sherif Sakr, Anna Liu and Ayman G. Fayoumi, " The family of mapreduce and large-scale data processing systems," ACM Computing Surveys, Volume 46 Issue 1, October 2013, Article No. 11

[5] Aditya B. Patel, Manashvi Birla and Ushma Nair, " Addressing Big Data Problem Using Hadoop and Map Reduce," in Proc. 2012 Nirma University International Conference On Engineering, pp. 1-5.

[6] Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Rasin, A., and Silberschatz, A. 2009. HadoopDB: An architectural hybrid of mapreduce and dbms technologies for analytical workloads. Proc. VLDB Endow. 2, 1,922–933.

[7] Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Rasin, A., and Silberschatz, A. 2010. HadoopDB in action: Buildingreal world applications. In Proceedings of the 36th ACM SIGMOD International Conference on Management of Data.

[8] Parallel DataProcessing with MapReduce: A Survey :www.cs.arizona.edu/~bkmoon/papers/sigmodrec 11.pdf

[9] Jyoti Nandimath, Ankur Patil, Ekata Banerjee,Pratima Kakade and Saumitra Vaidya, " Big Data Analysis Using Apache Hadoop," IEEE IRI 2013, August 14-16, 2013, San Francisco, California, USA

[10] Jens Dittrich and JorgeArnulfo Quian´eRuiz, " Efficient Big Data processing in Hadoop Mapreduce,"Proceedings of the VLDB Endowment, Volume 5 Issue 12, August 2012, Pages 2014-2015

[11] D. Harris,(2013),"The history of Hadoop:From 4 nodes to the future of data".Accessed: https://gigaom.com/2013/03/04/the-history-of-hadoop-from-4-nodes-to-the-future-of-data/.

[12] Vance, Ashlee (2009-03-17). "Hadoop, a Free Software Program, Finds Uses Beyond Search". The New York Times. Archived from the original on 11 February 2010. Retrieved 2010-01-20.

[13] Apache Hadoop." http://hadoop.apache.org.

[14] Fries, Sergie. "MapReduce: Fast Access to Complex Data." Data Management and Data Exploration Group. 1 Jan. 2014.

[15] Marx, Vivien. "Nature." The Big Challenges of Big Data 498 (2013)