International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.4,No.7, July 2015 DOI:10.15693/ijaist/2015.v4i7.59-63

A Survey On Big Data Processing Methods

Basil C Sunny
PG Student/Department of CSE
AdiShankara Institute of Engineering and Technology, Kalady, Kerala, India

Ramesh R
Assistant Professor/Department of CSE
AdiShankara Institute of Engineering and Technology, Kalady, Kerala, India

***Abstract*—Today, in a digital universe in which information and technology play important roles in dictating the quality of our lives. A zeta byte of data passed through the Internet in the past year; IDC predicts that this digital universe will explode to an unimaginable eight Z bytes by 2015. Conventional data processing technologies are now unable to process this data within a tolerable elapsed time. Big Data with large-volume, heterogeneous, autonomous sources with distributed and decentralized control, and seeks to explore complex and evolving relationships among data. These characteristics make it an extreme challenge for discovering useful knowledge from the Big Data. Data Scientist uses different methods to handle Big Data which starts from conventional data mining algorithms to complex distributed processing methods. This paper surveys different approaches used to handle big data from the beginning of big data era.**

Index terms - Big data, Distributed Processing, Knowledge Extraction, Mapreduce.

I. INTRODUCTION

Data Mining is the technology to extract the knowledge from the data. It is used to explore and analyze the same. Data Mining has also been termed as data dredging, data archeology, information discovery or information harvesting depending upon the area where it is being used. The data mining environment produces a large volume of the data usually called the Big Data or Very Large (VL) Data. Michael Cox and David Ellsworth were among the first to use the term big data literally, referring to using larger volumes of scientific data for visualization [1]. Big data includes data sets with sizes beyond the capability of normally used data mining tools to capture, curate, manage, and process the data within a tolerable elapsed time.Table 1 shows Hubers description of data set sizes [2][3].

Byte	**10^6**	**10^8**	**10^{10}**	**10^{12}**	**$10^{>12}$**
Size	Medium	Large	Huge	Monster	**VL**

A general definition of big data is as the three V's: volume, velocity and variety, but in the modern context its up to seven V's. Several factors contribute to the increase in data size. Transaction-based data stored through the years. Social Medias generate unstructured data streams. Large amounts of sensor and machine-to-machine data being captured. Big data volume was a storage issue in the past. Other Big Data issues emerged in recent years includes how to determine relevance within Big data volumes and how to use analytic to generate meaning from relevant data. Data is streaming in at high speed and must be dealt with in a timely manner. Sensors, RFID tags and smart metering are driving the need to deal with torrents of data in near-real time. Most organizations are now facing the challenge of reacting quicklytoday in traditional databases are Structured. Structure-less data includes text documents, email, video, audio, stock ticker data and financial transactions.

Main sources which produce big data are social networks, software as a service and cloud applications, data warehouse appliances, network and in-stream monitoring technologies, legacy documents and scientific experiment results. The ubiquity of personal computing technology, especially mobile computing, has produced an abundance of staggeringly large data sets. Face book alone logs over 25 terabytes (TB) of data per day. A data set representing 10^{12} objects, each with ten features, stored in short integer (4 bytes) format would require 40 TB of storage (most high-performance computers have <1 TB of main memory). A zeta byte of data passed through the Internet in the past year; IDC predicts that this digital universe will explode to an unimaginable eight Z bytes by 2015. Hence, Table I will continue to be pertinent for many years.

Conventional data processing technologies are now unable to process big data within a tolerable elapsed time. All state of-the-art implementations of data mining algorithms operate by loading the whole training dataset into the main (RAM) memory of a single machine or simple machine clusters that have static processing and storage capacity configurations. This approach has two key problems. First, the data can simply grow too big over time to fit into the available RAM. Second, most of big data applications produce data spread across multiple distributed data sources (including streaming sources). Moving all the datasets to a centralized machine is thus expensive. When the data mining algorithms' computational complexity exceeds the available RAM, the algorithms don't scale well and they never finish or are unable to process the whole training dataset. Hadoop-Mapreduce framework [4] is one of the best solutions suggested by the data scientist handle above mentioned problems of Big Data.

II. RELATED WORK

Digital world produces big data with huge volume, different in data structure which includes decentralized and distributed sources. Finding relationships among such data have high value, but extremely complex task to establish. The characteristics of the big data make it extremely difficult for knowledge extraction. Data produced from various sources like social networks, public and private clouds, sensor networks need to be stored and analyzed due to the value of

the knowledge resides in the data from an analytical view point. Such data need highly scalable as well efficient storage schema and also needs highly time consuming analytical operations.

To handle this two different approaches are commonly proposed by big data scientists. One way is to produce informative version of big data using different clustering techniques which generates summaries of the entire data. Such clustering techniques aim to produce a good quality of clusters/summaries. Therefore, they would hugely benefit everyone from ordinary users to researchers and people in the corporate world, as they could provide an efficient tool to deal with large data. Every traditional clustering method are not suited for big data, so researchers have been selected candidate algorithms from data mining and modified it according to the processing needs. Second way is use advantage of distributed processing and cloud computing to hand the big data produced from multiple sources. Data mining application frameworks and Data Center Cloud Technologies contains different methods for such processing.

A. Big Data Clustering

1. Fuzzy-C Means (FCM)

The FCM[5] algorithm is a partion-based "soft" clustering method in which clustering is done based on the membership values of each objects in the data sets. A membership function is used to generating the membership values of data objects. It is possible to cluster one object in more than one cluster due to different degrees of belief produced by the membership function. A center point for each cluster is chosen and the objects are assigned to clusters based on center point and the membership values. Intracluster variance is minimized and intercluster variance is maximized in fuzzy c means algorithm.

FCM pseudo-code

Input: Given the dataset, set the desire number of clusters c, the fuzzy parameter m (a constant > 1), and the stopping condition, initialize the fuzzy partition matrix, and set stop = false.

Step 1: Do

Step 2: Calculate the cluster centroids and the objective value J.

Step 3: Compute the membership values stored in the matrix.

Step 4: If the value of J between consecutive iterations is less than the stopping condition, then stop = true.

Step 5: While (!stop)

Output: A list of c cluster centers and a partition matrix are produced.

A modified FCM called incremental multiple medoids based fuzzy clustering(IMMFC)[6] is proposed for to handle complex patterns in large data sets. In this approach dataset is divided into different data chunks and medoids or centroids of each chunk is generated which will used to represent clusters. A data chunk is a portion of data in the given big data set. Representatives from different data chunks are analyzed to extract useful information. IMMFC keeps track of relationships among selected representatives in each data chunk to improve the overall clustering process.

2. BRICH

BIRCH algorithm [7] is hierarchal-based clustering algorithm which generates a special dendrogram based data structure called clustering feature tree (CF tree). BIRCH does not need the entire data set in memory for processing. It scans portions of data in an incremental and dynamic fashion to build the CF tree. BRICH is a two phase approach: an in-memory CF tree is build first by performing incremental scanning on dataset and secondly leaf needs are clustered.

BIRCH pseudo-code

Input: The dataset, threshold T, the maximum diameter (or radius) of a cluster R, and the branching factor B

Step 1: (Load data into memory) An initial in-memory CF-tree is constructed with one scan of the data. Subsequent phases become fast, accurate and less order sensitive.

Step 2: (Condense data) Rebuild the CF-tree with a larger T.

Step 3: (Global clustering) Use the existing clustering algorithm on CF leaves.

Step 4: (Cluster refining) Do additional passes over the dataset and reassign data points to the closest centroid from step #3.

Output: Compute CF points, where CF = (# of points in a cluster N, linear sum of the points in the cluster LS, the square sum of N data SS).

3. DENCLUE

The DENCLUE [8] which is a density-based algorithm analytically models the cluster distribution according to the sum of influence functions of all of the data points. The influence function can be seen as a function that describes the impact of a data point within its neighborhood. Then density attractors can be identified as clusters. Density attractors are local maxima of the overall density function. In this algorithm, clusters of arbitrary shape can be easily described by a simple equation with kernel density functions.

DENCLUE pseudo-code

Input: The dataset, Cluster radius, and Minimum number of objects

Step 1: Take dataset in the grid whose each side is of 2σ.

Step 2: Find highly dense cells, i.e. find out the mean of highly populated cells.

Step 3: If d (mean(c1), mean(c2)) < 4a, then the two cubes are connected.

Step 4: Now highly populated cells or cubes that are connected to highly populated cells will be considered in determining clusters.

Step 5: Find Density Attractors using a Hill Climbing procedure.

Step 6: Randomly pick point r.

Step 7: Compute the local 4σ density.

Step 8: Pick another point (r+1) close to the previous computed density.

Step 9: If den(r) < den(r+1) climb, then put points within (σ /2) of the path into the cluster.

Step 10: Connect the density attractor based cluster.

Output: Assignment

4. OptiGrid

OptiGrid algorithm [9] is designed to obtain an optimal grid partitioning. This is achieved by constructing the best cutting hyperplanes through a set of selected projections. These projections are then used to find the optimal cutting planes. Each cutting plane is selected to have minimal point density and to separate the dense region into two half spaces. After each step of a multi-dimensional grid construction defined by the best cutting planes, OptiGrid finds the clusters using the density function.

OptiGrid pseudo-code

Input: The dataset (x), a set of contracting projections P = {P0, P1, …..,Pk}, a list of cutting planes BEST CUT ϕ, and CUT ϕ.

Step 1: For i=0,...,k, do

Step 2: CUT best local cuts Pi(D), CUT SCORE, Score best local cuts Pi(D)

Step 3: Insert all the cutting planes with a score >= min cut score into BEST CUT

Step 4: Select the q cutting planes of the highest score from BEST CUT and construct a multidimensional grid G using the q cutting planes.

Step 5: Insert all data points in D into G and determine the highly populated grid cells in G; add these cells to the set of clusters C.

Refine C: For all clusters Ci in C, perform the same process with dataset Ci

Output: Assignment of data values to clusters.

5. Expectation-Maximization (EM)

EM algorithm[9] is designed to estimate the maximum likelihood parameters of a statistical model in many situations, such as the one where the equations cannot be solved directly. EM algorithm iteratively approximates the unknown model parameters with two steps: the E step and the M step. In the E step (expectation), the current model parameter values are used to evaluate the posterior distribution of the latent variables. Then the objects are fractionally assigned to each cluster based on this posterior distribution. In the M step (maximization), the fractionalassignment is given by re-estimating the model parameters with the maximum likelihood rule.

EM pseudo-code

Input: The dataset (x), the total number of clusters (M), the accepted error for convergence (e) and the maximum number of iterations.

E-step: Compute the expectation of the complete data log-likelihood.

M-step: Select a new parameter estimate that maximizes the Q-function.

Iteration: increment t = t + 1; repeat steps 2 and 3 until the convergence condition is satisfied.

Output: A series of parameter estimates $\{\theta_0, \theta_1, \ldots, \theta_T\}$, which represents the achievement of the convergence criterion.

The main disadvantage of above explained candidate clustering algorithms are they need more powerful main frame systems to execute with respect to the increase in data size. Due to high cost, only large organizations which regularly need to analyze big data can afford this type of computers. But still when data size increases, processing power is also need to be increased. Data Center Cloud Technologies gives alternative solution for this problem because such technologies are on-demand and pay-per use. The cost factor is still very high which leads to a distributed processing architecture.

B. Distributed Big Data Processing

In Distributed processing, big data is divided into small chunks and passed to distributed systems for parallel processing. Then the results are aggregated together to form the final.

1.Parallel-Horus

Parallel-Horus[10] is a distributed, user transparent programming model which uses sequential c++ programs to enable distributed multimedia content analysis. This framework is implemented as a grid execution model which efficiently uses grid resources and a significance performance improvement is achieved. Framework allows the users to describe programs as algorithmic patterns which contains a set of computations and dependency exist with that computations.

2. Data Flow Graphs

Data Flow[11] Graphs are used as a programming model for distributed large scale big data processing. DFG is a directed acyclic graph where each vertex represents a program and edges represent data channels. At runtime the vertices of the DFG will be scheduled to run on the available machines of the cluster. Edges represent the flowof the processed data through the vertices. Data communication between the vertices is abstracted from the user and can be physically implemented in several ways. Based on the DFG model, one of the most famous implementation of a DFG based processing framework is Microsoft's Dryad which enables the processing of iterative jobs by defining processes to perform only on the portion of updated data between two consecutive iterations. The iterations are thus performed on set of updates rather than the whole dataset.

3. Map Reduce model

The Map Reduce[4] model proposed by Google for simplified processing of big data through distributed clusters. The Map Reduce model is composed of two principal phases: Map and Reduce. The data model in Map Reduce is the key/value pair. During the Map phase, the associated Map function is executed on every key/value pair of the input data, producing in turn intermediate key/value pairs. Between the Map and the Reduce phase is the shuffling phase where the intermediate key/value pairs are sorted and grouped by the key. This result in a set of key/value pairs where each pair contains all the values associated to a particular key. These key/value pairs are then partitioned and grouped on the key then passed on to the Reduce phase during which the Reduce function is applied individually on each key and the associated values for that key. The Reduce phase can produce zero or one output value. Between each phase the intermediate outputs are stored on the file system which is most of the time distributed for fault-tolerance but also for data localization.

Big Data Clustering	Distributed Processing
▪ Cannot scale well to size above Peta Bytes. ▪ Need high cost hardware to run ▪ Time consuming ▪ Data must be moved to a central system ▪ Not suitable for every category of data	▪ Can scale up and down according to the size of data ▪ Parallel processing increases the throughput and efficiency ▪ On demand service avoids hardware costs ▪ No need to move data to a central system ▪ Suitable for different flavors of data

Comparisonbetween big data Clustering methods and distributed processing Methods shown in the table 2.

III. DISTRIBUTED BIG DATA SYSTEMS

Distributed processing methods can handle different types of big data ie,.structured, unstructured and semi structured data. Distributed processing methods are applicable for different flavors data that are available in different areas. Following are some distributed big data systems that are build using the methods mentioned in the above section.

A.sksOpen

sksOpen[12] is an Online Indexing and Querying System for Big Geospatial Data which allows users to easily create indices of spatial objects and to query and visualize the results and share them via unique URLs. This system is implemented on MapReduce frame and uses TerraFly API.

B. ClubCF

Clustering-based Collaborative Filtering approach (ClubCF)[13] is a service recommender system which classifies similar services together to improve service recommender. ClubCF divides available services to different clustering based on service similarity in logic. A collaborative filtering algorithm is applied on the clusters to retrieve the requested service. Searching space is reduce from available services to clustered service in ClubCF which improves the execution time and performance. Hadoop frame work is used to implement ClubCF for online service recommendation.

C. Parallel CCL

A huge amount of Big data is image and video data generated by sources like image sensor networks and the analysis of such data is an important challenge in big data. Parallel Connected Component(CCL) [4] is a high performance image analysis system which is suitable for large image and video sets. Parallel CCL can process streaming video data which reduces the memory requirements and thus hardware cost to build the system comparatively smaller.

D. KASR

Keyword-Aware Service Recommendation method, named KASR[15] is a Personalized Service recommender systems which provide appropriate recommendations to users. Keywords are used to indicate users' preferences, and a user-based Collaborative Filtering algorithm is adopted to generate appropriate recommendations. To improve its scalability and efficiency in big data environment, KASR is implemented on Hadoop, a widely-adopted distributed computing platform using the MapReduce parallel processing paradigm.

E. Distributed Video Transcoding System

The paper[16] presents a distributed video transcoding system which uses Map- Reduce framework, FFmpeg and an open source distributed computing model. In this model, video is splitted into several parts and then stored in HDFS. Each mappers transcode video clips using FFmpeg and the reducers are used to merge results to form the final target video.[12] presents a video monitoring framework based on Apache Hadoop. The framework provides a Hadoop cluster based processing, responsive video distribution network for handling.

IV. CONCLUSION

Big data storage and analysis is one of the hot topic in modern digital world. This paper gives an overview of the entire big data paradigm from the beginning of big data era. Initially, data scientists selected candidate clustering algorithms in data mining with some modifications for processing large datasets. Processing capabilities of these algorithms are directly proportional to data size and available memory of the processing computer. To overcome disadvantages of big data clustering algorithms, distributed processing methods are introduced. These methods divide big data into manageable data chunks, process them individually in separate nodes and the results are aggregated together. Distributed processing methods can scale up as well as mange different categories of data. Finally, some existing big data systems from different areas are mentioned which use distributed processing.

REFERENCES

[1]. Michael Cox and David Ellsworth (1997) Application-controlled demand paging for out-of- core visualization, ACM.

[2]. P. Huber (1997) Massive data sets workshop: The morning after in Massive Data Sets. Wash- ington, DC: Nat. Acad., pp. 169184.
[3]. R. Hathaway and J. Bezdek (2006) Extending fuzzy and probabilistic clustering to very large data sets, Comput. Statist. Data Anal., vol. 51, pp.215234.
[4]. J. Dean and S. Ghemawat (2008) Mapreduce: simplified data processing on large clusters, Commun. ACM, vol. 51, pp. 107113, Jan.
[5]. J. C. Bezdek, R. Ehrlich, and W. Full. "Fcm: The fuzzy c-means clustering algorithm". Computers & Geosciences, 10(2):191–203, 1984.
[6]. Yangtao Wang and Jian-Ping Mei, "Incremental Fuzzy Clustering with Multiple Medoids for Large Data", IEEE Transactions On Fuzzy Systems.
[7]. T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. ACM SIGMOD Record, volume 25, pp. 103–114, 1996.
[8]. A. Hinneburg and D. A. Keim. "An efficient approach to clustering in large multimedia databases with noise". Proc. of the ACM SIGKDD Conference on Knowledge Discovery ad Data Mining (KDD), pp. 58–65, 1998.
[9]. A. Fahad, N. Alshatri and Z. Tari, "A Survey of Clustering Algorithms for Big Data: Taxonomy & Empirical Analysis", IEEE Transactions on Emerging Topics in Computing.
[10]. Frank J. Seinstra, Jan-Mark Geusebroek, Dennis Koelma, Cees G.M. Snoek, Marcel Worring, and Arnold W.M. Smeulders (2007) High Performance Distributed Video Content Analysis with Parallel-Horus, IEEE.
[11]. Nam-Luc Tran, SabriSkhiri, Arthur Lesuisse and Esteban Zimanyi. "AROM: Processing Big Data With Data Flow Graphs and Functional Programming", IEEE 4th International Conference on Cloud Computing Technology and Science, 2012.
[12]. Yun Lu and Mingjin Zhang," sksOpen: Efficient Indexing, Querying, an Visualization of Geo-spatial Big Data", 12th International Conference on Machine Learning and Applications, 2013.
[13]. Rong Hu and Wanchun Dou," ClubCF: A Clustering-based Collaborative Filtering Approach for Big Data Application", IEEE Transactions On Emerging Topics In Computing, 2013.
[14]. S.M. Najmabadi, M. Klaiber, Z. Wang, Y. Baroud and S. Simon, Stream Processing of Scientific Big Data on Heterogeneous Platforms – Image Analytics on Big Data in Motion, IEEE 16th International Conference on Computational Science and Engineering, 2013.
[15]. ShunmeiMeng, Wanchun Dou, Xuyun Zhang, Jinjun Chen, "KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications", IEEE Transactions On Parallel And Distributed Systems, 2013.
[16]. Chenwei Song, WenfengShen, Lianqiang Sun, Zhou Lei and WeiminXu (2014) Distributed Video Transcoding Based on MapReduce, IEEE ICIS.

Authors Profile

Basil C Sunny received the **Diploma**in computer science and engineering from the Govt. Polytechnic, Chelad, Technical Education, Kerala, India, in 2009 and **B. Tech**. in computer science and engineering from the MBITS, Nellimattom, MG University, Kottayam, India, in 2013.Currently doing **M.Tech.** incomputer science and engineering in MG University, Kottayam, Kerala, India. His research interest includes Big Data Analytics, Map reduce processing, Distributed parallel processing, Medical video analysis

Ramesh R received **B.Tech** Degree from Mahatma Gandhi University in 2010 and **M.Tech** in computer science and engineering from M.G.University. Currently working as Assistant professor at AdiShankara Institute of Engineering& Technology. His research interest includes Network Security, Big Data Processing