Introduction of Skyline Query in Big Data with Hadoop Architecture

Mitali Chauhan[1] Hitesh Patel[2]

[1]ME Student [2]Assistant Professor

[1,2]Department of Information Technology Engineering

[1,2]Kitrc-Kalol, Gandhinagar-382721

Abstract— Big Data is a broad term for datasets so large and complex that create difficulties to process on large dataset using on hand database management tools. In this paper we are presenting the 5v's characteristics of the Big Data and also technology which is used to handle Big Data. During the last decades, management of Data and storage have become increasingly distributed. So requirement of the advanced query operators, such as skyline queries, are necessary in order to help users to handle the huge amount of available data by identifying a set of interesting data objects. Here we also discuss the concept of skyline which filters out the set of interesting points from the large set of data points from database.

Key words: Big Data, Skyline query, Hadoop, MapReduce, HDFS

I. Introduction

Big Data is the collection of complex and large data sets, which are difficult to capture, process, store, search and the analysis of data/ information using conventional database management tools and traditional database management system [11].The main difficulty in handling such large amount of data is because that the volume is increasing rapidly in comparison to the computing resources [2]. Data (big data) is generated by machine, generated by humans and also generated by Mother Nature [5].The Big data term which is being used now a days is kind of misnomer as it points out only the size of the data not putting too much of attention to its other existing properties [2].Big data can neither be worked upon by using traditional SQL like queries nor can the relational database management system (RDBMS) be used for storage [5]. Hadoop an open source distributed data processing system is one of the prominent and well known solutions [5].

Big Data can be described by following characteristics:

A. Volume:

It is the data at rest. Usually, Terabytes and Exabyte's of existing data to process for accurate analysis [11]. The social networking sites existing are themselves producing data in order of terabytes everyday and this amount of data is definitely difficult to be handled using the existing traditional systems [2].

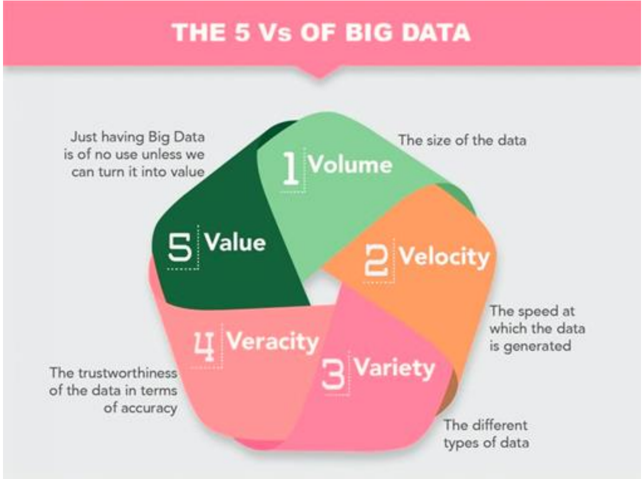
THE 5 Vs OF BIG DATA
1 Volume
The size of the data
2 Velocity
The speed at which the data is generated
3 Variety
The different types of data
4 Veracity
The trustworthiness of the data in terms of accuracy
5 Value
Just having Big Data is of no use unless we can turn it into value

Fig. 1: 5v's of big data

B. Velocity:

It is the data in motion usually; streaming data has a few milliseconds to a few seconds for a response [11]. This characteristic is not being limited to the speed of incoming data but also speed at which the data flows [2]. Volume influences the velocity as the increase in data volume can reduce the rate at which the data is captured and transmitted [3]. For example the data from the sensor devices would be constantly moving to the database store and this amount won't be small enough on performing the analytics on the data which is constantly in motion [2].

C. Variety:

Variety corresponds to the different forms of data like web Pages, Web Log Files, social media sites, e-mail, documents, and sensor devices data both from active passive devices [2], comprising unstructured, semi-structured, structured, texts, logs, etc., with unstructured data forming the major portion of big data [3].

D. Veracity:

It is the data in doubt. Data is uncertain due to inconsistency, incompleteness, ambiguities, latency, deception and model approximations [11]. The uncertainty constraints are always higher in big data, as it is extracted from multiple sources, which could rise to new issues of trustworthiness of data [3].

E. Value:

The value of data is determined by the various factors i.e.; quality of information, statistical information, survey data, data from reliable and well reputed sources [3]. It is all good and well having access to big data but unless we can turn it into value. It becomes very costly to implement IT infrastructure systems to store big data, and businesses are going to require a return on investment [5].

II. Architecture

At a very high level, Hadoop has two main components Map reduce and file system HDFS [6].MapReduce is the processing part of Hadoop and manages the jobs. HDFS refers to Hadoop Distributed File system stores all the data redundantly required for computations [6]. Origin of Hadoop comes from Google File System GFS and MapReduce which become Apache HDFS and Apache MapReduce respectively [6].

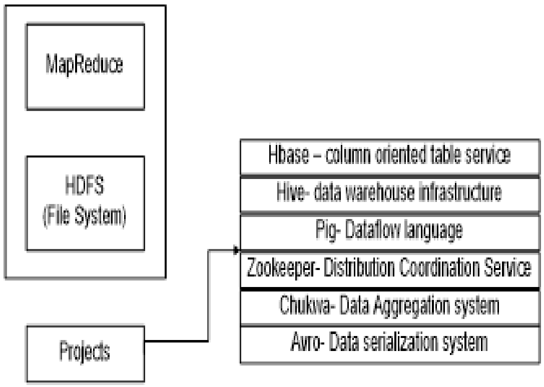
MapReduce
HDFS (File System)
Hbase – column oriented table service
Hive- data warehouse infrastructure
Pig- Dataflow language
Zookeeper- Distribution Coordination Service
Chukwa- Data Aggregation system
Avro- Data serialization system
Projects

Fig. 2: high level architecture of Hadoop

A. Mapreduce:

MapReduce is the heart of Hadoop, which is a programming model meant for large clusters. It has a parallel computing framework and is obscure to responsibilities like parallelization, fault tolerance, data distribution and load balancing [11]. The computation of MapReduce takes a set of input key/value pairs and generates a set of output key/value pairs and is divided into two functions: Map function and Reduce function [11].

The map tasks produce a sequence of key-value pairs from the input and this is done according to the code written for map function [2]. These value generated are collected by master controller and are sorted by key and divided among reduce Tasks [2]. The sorting basically assures that the same key values ends with the same reduce tasks. The Reduce tasks combine all the values associated with a key working with one key at a time. Again the combination process depends on the code written for reduce job [2].

B. Hdfs:

The HDFS is distributed file system component of Hadoop designed to store and support large amount of data sets on commodity hardware reliably and efficiently and also has master / slave architecture, with master as Name Node and slave as Data Node [6]. All the individual files in HDFS are divided into fixed size blocks. A cluster of machines with storage capacity are used for the storage of these blocks [11]. HDFS stores all the file system metadata on single Name Node. The application data is stored as multiple copies on a number of slave Data Nodes, which are usually one per node in cluster [6].

1) NameNode acts as the master of the system and is responsible for the management of blocks on the DataNodes [11]. NameNode itself doesn’t store any data neither does any data flows through the name node. It only determines and keeps track of mapping of file blocks to Data Node, thereby acting as a repository for all HDFS metadata [6]. It is the single entry point for a failure happening in Hadoop cluster. The replication factor, specified by the application is stored in the NameNode [11].Any application requiring data, first contacts the NameNode which provides locations of data blocks containing the file. While storing/writing data to HDFS, NameNode chooses a group of (by default three) to store the block replicas [6]. The client application then pipelines the data to the Data Node nominated by Name Node [6].
2) DataNodes acts as slaves and it is deployed on each machine in the cluster. DataNode stores the actual data [11]. Data Nodes are responsible for storing the blocks of file as determined by the Name Node [6]. Data file to be stored is first split into one or more blocks internally [6]. Data Nodes serve the read write requests from file system’s client data [6]. These are also responsible for creating, deleting and replicating blocks of file after being instructed by the Name Node [6].

III. Skyline Query

Before the entry of skyline into database management there is a problem called maximum vector problem or Pareto optimum[1].skyline queries have been actively studied to support multi-criteria decision analysis[4] and also for decision making applications; skyline queries help users make intelligent decisions over complex data, where different and often conflicting criteria are considered[7].

For example, consider a database that contains information about hotels [1]. Each tuples of the database is represented as a point in a data space consists of numerous dimensions [1]. Assume a user is looking for a hotel that is cheap as possible and as close to the beach. To illustrate the idea of dominance relationships, Fig.1 gives hotel finding example in this example user is looking for a hotel based on two criteria, minimum price and minimum distance to the user standing location [1]. Fig.1a lists 9 hotel records and their values and Fig.1b depict the representation of the hotel in a 2D space [1]. Hotels p4, p7, p8 and p9 are all dominated by other points so skyline which return points that are not dominated by any other points. Consider the point, p7 which is dominated by p5 as it is more expensive than p5 but both have the same distance value [1].

The skyline query retrieves all hotels for which no other hotel exists that is cheaper and closer to beach. So the skyline result set which consist of {p2, p3, p1, p5, p6} [1].In database systems, queries specialized to search for the non-dominated data points are called skyline queries and their corresponding result set is known skyline set and Individual data points in a skyline result set are known as skyline Points[1].

Hotel	Price	Distance
P1	3	3
P2	1	6
P3	2	4
P4	3	7
P5	5	2
P6	7	1
P7	6	2
P8	4	4
P9	6	6

Table 1: Skyline query Dataset

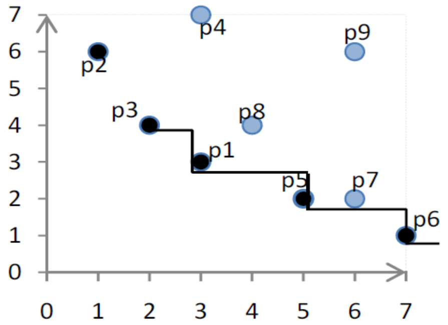

Fig. 3: An Example of skyline query

The database system at your travel agents' is unable to decide which hotel is best for you, but it can at least present you all interesting hotels and all hotels that are not worse than any other hotel in both dimensions so we can say this is set of interesting hotels the Skyline [13].

The same considerations also hold for a variety of applications (e.g., electronic marketing places or real-estate databases for houses), where the user is interested in mobiles, cars, houses, or other products likewise, a user who is interested in buying a car wants to find a good trade-off between minimum mileage, minimum age, and minimum price[7].With the growing number of applications that include uncertainty, e.g., sensor readings, human reading errors, and data imperfection, it became essential to support skyline queries over uncertain data[10]. As nowadays data is increasingly stored and processed in a distributed way, skyline processing over distributed data has attracted much attention recently [7].

A. Skyline Query:

Given a set of points, the skyline query proceeds a set of points (referred to as the skyline points), such that any point is not dominated by any other point in the dataset [12].

B. Skyline Query Processing:

Given a set of points, the skyline query is examined for the interesting point in the datasets is referred as Skyline Query Processing [12].

C. Range Based Skyline Query:

Given a set of points, the skyline query based on the range is examined for the interesting point in the datasets is referred as Range Based Skyline Query Processing [12]

D. Sql Extensions:

In order to specify Skyline queries, proposed to extend SQL‟s SELECT statement by an optional SKYLINE of clause as follows with the extension of the SQL and thus represented for making the processing for the skyline [12].

SELECT ... FROM ... WHERE...
GROUP BY ... HAVING...
SKYLINE OF [DISTINCT] d1 [MIN | MAX |
DIFF], ..., dm [MIN | MAX | DIFF]
ORDER BY...

d1;:::; dm denotes the dimensions of the Skyline; e.g., price, distance to the beach, or rating. MIN, MAX, and DIFF specify whether the value in that dimension should be minimized, maximized, or simply be dissimilar [12]. For example, the price of a hotel should be minimized (MIN annotation) whereas the rating should be maximized (MAX annotation). The optional DISTINCT specify how to deal with duplicates [12].

Skyline query processing in distributed environments poses inherent challenges and requires non-traditional techniques due to the distribution of content and the lack of global knowledge. There are various different distributed systems with different requirements and unique characteristics that have to be exploited for efficient skyline processing [7].

A good distributed skyline algorithm should achieve the following goals:

1) Minimization of bandwidth consumption. We measure bandwidth in the total number of points transmitted over the network [9]. Precisely speaking, some extra bandwidth is needed for sending other synchronizing messages and the packet headers in order to enforce the underlying network protocol [9].
2) Progressiveness. That is, ideally, the algorithm should quickly output some early results soon after the beginning and produce a majority of the remaining results well before the end of execution [9].
3) Adaptability to user preferences. The algorithm should allow the flexibility of returning skyline points in different orders [9].

	Filter points	Routing	Result	propagation Optimization goal
DSL	All local skyline points	Overlay	Same path	Response time (network communication cost and load balancing)
SSP/Skyframe	Most dominating point	Overlay	Direct	Network communication cost/response time
iSky	Most dominating point and threshold	Overlay	Direct	Response time and network communication cost
SSW	Nearest neighbor	Overlay	Direct	Scalability (network communication cost and contacted peers)
SFP	Most dominating point	Exhaustive	Direct	Network communication cost
DDS	All local skyline points	Routing index	Same path	scalability (network communication cost and contacted peers)
SKYPEER	Threshold	Flooding	Same	Response time

			path	(computational time and transferred data)
SKYPEER +	Threshold	Routing index	Same path	Response time (computational time, transferred data and contacted super-peers)
BITPEER	No filter points	Flooding	Same path	Response time
PaDSkyline	Multiple filter points	Exhaustive	Direct	Response time (parallelism and network communication cost)
AGiDS	No filter points	Exhaustive	Direct	Response time
FDS	Multiple filter points	Exhaustive	Direct	Network communication cost
SkyPlan	Multiple filter points	Exhaustive	Same path	Response time (parallelism and network communication cost)

Table 2: Overview of the features and objectives of the different distributed approaches [8]

IV. CONCLUSION

Big data provides an opportunity for big analysis leading to big opportunities to advance the quality of life. In this paper details about Big Data have been based on Hadoop Framework and also described the concept of big data, its importance and the existing projects. This paper provides a survey of existing approaches for skyline computation. Skyline queries retrieve the non-dominated points from a large database system based on the user preference so it can be used in preference based applications. It successfully eliminates all the dominated points by using some efficient technique.

REFERENCES

[1] Angel C Bency, S Deepa Kanmani,” A SURVEY OF SKYLINE PROCESSING IN VARIOUS ENVIRONMENT”, Indian Journal of Computer Science and Engineering (IJCSE), ISSN : 0976-5166 Vol. 5 No.1 Feb-Mar 2014

[2] Avita Katal Mohammad Wazid R H Goudar,” Big Data: Issues, Challenges, Tools and Good Practices”, IEEE, 2013

[3] Cameron Seay, Rajeev Agrawal, Anirudh Kadadi, Yannick Barel,” Using Hadoop on the Mainframe:A Big Solution for the Challenges of Big Data”, 12th International Conference on Information Technology - New Generations, IEEE, 2015

[4] Hyountaek Yong, Jongwuk Lee, Jinha Kim, Seung-won Hwang *,”Skyline ranking for uncertain databases *”, Elsevier Inc, 2014

[5] Ishwarappa , Anuradha J“A brief Introduction on Big Data 5Vs Characteristics and Hadoop Technology”, Published by Elsevier, 2015

[6] Kamalpreet Singh, Ravinder Kaur,” Hadoop: Addressing Challenges of Big Data”, IEEE, 2014

[7] Katja Hose, Akrivi Vlachou,” Distributed Skyline Processing: a Trend in Database Research Still Going Strong”,ACM,March 26–30, 2012,

[8] Katja Hose • Akrivi Vlachou,” A survey of skyline processing in highly distributed environments”, Springer-Verlag , The VLDB Journal (2012) 21:359–384

[9] Lin Zhu, Yufei Tao, and Shuigeng Zhou,” Distributed Skyline Retrieval with Low Bandwidth Consumption”, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 21, NO. 3, MARCH 2009

[10] Mohamed E. Khalefa, Mohamed F. Mokbel, Justin J. Levandoski,” Skyline Query Processing for Uncertain Data *”, ACM, October 26–30,2010

[11] Pradeep Adluru, Srikari Sindhoori Datla, Xiaowen Zhang*, “Hadoop Eco System for Big Data Security and Privacy”, IEEE,2015

[12] R.Prema Steffi 1, S.Sundaramoorthy 2,” Survey on Skyline Queries with Its Algorithms and Operators”, International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 2 Issue 11, November – 2013

[13] Stephan Borzsonyil Donald Kossmann2 Konrad Stocker',” The Skyline Operator*”,IEEE, 2001