International Journal of Engineering and Techniques - Volume 2 Issue 6, Nov – Dec 2016

RESEARCH ARTICLE OPEN ACCESS

A review on Joining Approaches in Hadoop Framework and Skewness Associate to it

Bibhudutta Jena
Kiit University, Bhubaneswar

Abstract:

Today's era is generally treated as the era of data on each and every field of computing application huge amount of data is generated. The society is gradually more dependent on computers so large amount of data is generated in each and every second which is either in structured format, unstructured format or semi structured format. These huge amount of data are generally treated as big data. To analyze big data is a biggest challenge in current world. Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage and it generally follows horizontal processing. Map Reduce programming is generally run over Hadoop Framework and process the large amount of structured and unstructured data. This Paper describes about different joining strategies used in Map reduce programming to combine the data of two files in Hadoop Framework and also discusses the skewness problem associate to it.

Keywords— ***Hadoop, Mapreduce, Vertical scaling, Joining approaches, Data skewness***

I. INTRODUCTION

Big data means really large amount of data, which are not possible to processed using traditional computing techniques. Big data is not merely a data, rather it has become a complete subject, which involves various tools, technqiues and frameworks. The major sources of big data creation are black box data, social media data, Power grid data, search engine data etc.

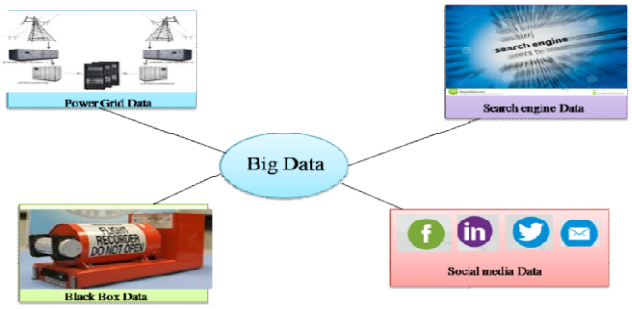

Fig 1.Sources Of big data

The above figure [1] describes the different sources from which the big data are generated. In current world big data are also used in different application domain such as Banking sector, Health care industry, Telecom industry etc.So different tools and framework are used to analyze the big data. Hadoop is one of the open source framework which is generally used to store and process the large amount of data which is either in unstructured format or in semi structured format. It basically consist of two main components generally termed as HDFS and Mapreduce Programming.

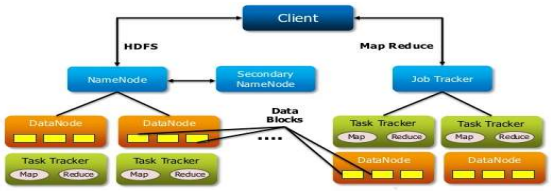

Fig 2.Hadoop Architecture

The figure [2] describes the basic architecture open source HADOOP framework which is generally used to operate the large amount of data by the use

of two main components i.e HDFS and Map Reduce. .Section II presents a literature survey on various optimization techniques used in HADOOP. Section III presents a brief idea about Hadoop Map Reduce Join. The motivation for this work is laid out in Section III. Section IV represents about the data skewness. Finally, Section V concludes the paper. Section VI provides an idea about future work.

II. LITERATURE SURVEY

The paper [1] Optimizing the big data in Hadoop framework by implementing NLS during map reduce programming. It does not follow a master slave architecture and sometime multiple name nodes are not coexist and cooperate with each other and proposes a technique or a service named as NLS (Name Node Location Service) .Similarly the paper [2] explores the Hadoop data placement policy in detail and proposes a modified block placement policy that increases the efficiency of the overall system. Also, the idea of placing data across the cluster in a way that a node possessing higher I/O capabilities is allocated more data is addressed, thereby reducing inter-node traffic and increasing overall performance. The goal of the paper [3] is to implement an load-balancing algorithm in the Hadoop framework to sort a list of timestamps. and it also gives an idea how to gathers locally the necessary information, combines them and produces a distribution of the data in order to avoid skew. The main objective of the paper [4] is to optimize the big data in hadoop map reduce framework by minimizing the load on task tracker. MapReduce is a monitoring technique and a operating model for distributed computing based on java. The MapReduce algorithm generally comprises of map task and reduce task. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs) [5] .Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job. The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers [6].

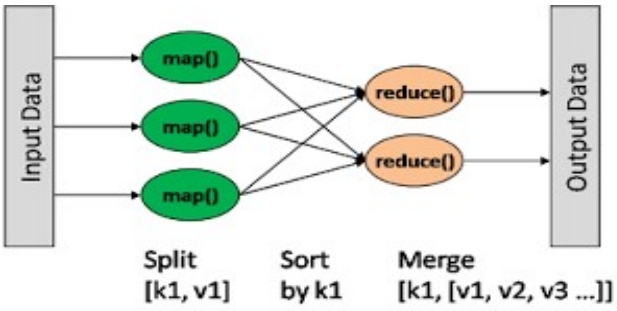

Fig 3. MapReduce architecture

The above figure [3] describes the architecture Map reduce framework which consist of two tasks map task and reduce task. As the name suggest first the map task will run then reduce task will run and finally the reduced output will provide to the customer the Hadoop framework.

III. MAP REDUCE JOIN APPROACHES

In Map reduce programming joining operation is performed to combine the data of two larger file and produce the combined output. Joining two large dataset can be achieved using MapReduce Join. However, this process involves writing lots of code to perform actual join operation [7]. Joining of two datasets begin by comparing size of each dataset. If one dataset is smaller as compared to the other dataset then smaller dataset is distributed to every datanode in the cluster. Once it is distributed, either Mapper or Reducer uses smaller dataset to perform lookup for matching records from large dataset and then combine those records to form output records. Basically two types of join operation can be perform in Hadoop framework during map reduce programming. These are:

A. Reduce side Joining

Suppose we want to perform join operation among two larger files in Hadoop framework then we follow the reduce side join approach to join the two files contents and produced combined output. In this approach firstly the content of two files are divided in to small parts and assigned to the mappers according to their division, because mappers cann't be set manually in Hadoop framework [8]. Then contents from all the mappers

of file 1 and file 2 will come to a reducer and the join operation will be perform on the reducer and from the reducer we get the combined join output., In this case we can manually set the number of reducers according to the user requirement.

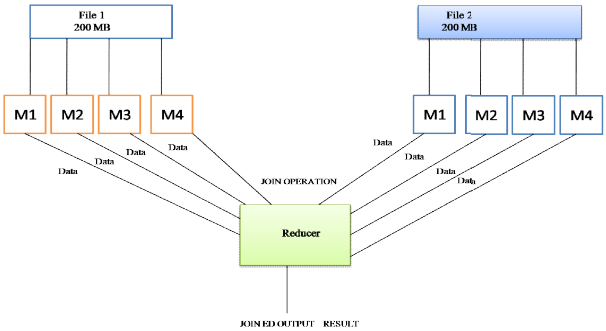

Fig 4.Reduce side join approach

The figure [4] describes the reduce side join approach. In the proposed figure there are two files having 200 MB of data. The content of first file has assigned to 4 mappers similarly the content of second file has distributed among 4 mappers manually. As shown in the figure the output from all the mappers will come to a system where reduce program is running which is termed as reducer. On the reducer the join operation has performed and from the reducer finally combine join output will produce. In this the number of reducer can be configurable. When more than one reducer will use in this approach then the output from the mappers will assign to the reducer according to the Hash algorithm.

B. Map side Joining

This type of join approach will follow when join operation will perform between one large files and one small file. In this approach cache memory [9] concept is used to store the content of smaller file and less number of mappers will require in comparison to the reduce side joining. In this approach no reducer will use for join operation.

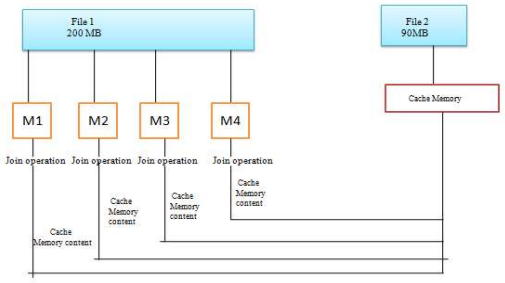

Fig 5.Map side joining approach

The figure [5] gives an idea about Map side joining approach. According to it there are two files having size 200 MB and 90 Mb, as the size of one file is very small map side joining is used here to combine the content of two files. In this approach the content of the small file will move to the cache memory which size has predefined by Hadoop developer and it must be larger than the size of the small file, and the cache memory will available to each and every mappers of the larger file and perform the join operation and provide the output. In this approach the number of reducer will be automatically decrease because the file with less size will not require any mappers for join operation. In the below table the paper has described some basic difference between the reduce side joining and map side joining.

C. Comparison between reduce side joining and map side joining

TABLE1. Reduce side Join vs Map Side Join

Reduce side Join	Map Side Join
This type of join operation will be possible when both the files are of larger size.	This type of join operation will be possible if one file size is much more smaller in comparison to the other file.
The number of mappers will be more in this type of joining.	The number of mappers will be less in comparison to the reduce side join.
Reducer will be configurable manually and require for the join operation.	Reducer will not require to perform the join operation.
Cache memory concept is not present here.	Cache memory concept is present here.
More chance to go through from data skewness.	Less chance to go through data skewness

The above table [1] points out some major differences between the reduce side join approach

and the map side join approach. Data skewness is a major problem for hadoop framework during the join operation. In the next section the paper has described the data skewness problem in hadoop framework.

IV. DATA SKEWNESS

Data skew is an imbalance in the load assigned to different reduce tasks [10],[11] .The load is a function of the assigned key to a reducer and the number of records and the number of bytes in the values per key. Data skewness always effect the performance of the system and minimize the execution time of the Hadoop framework. The paper has described why data skewness will actually happen in the below figure.

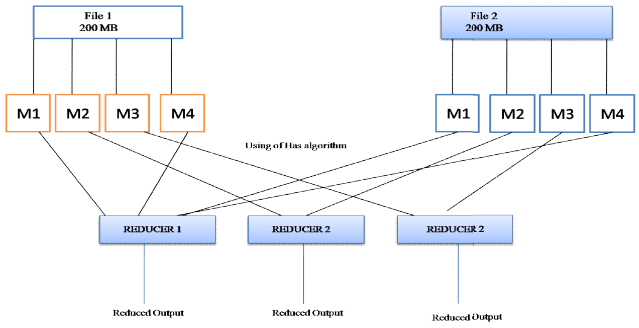

Fig 6. Data skewness problem

According to the above figure [6] during the reduce side joining approach when the number of reducer will be more then the output of the mapper will assign to the different reducer by using the hash algorithm concept. So in this approach sometimes the load will be more on some reducer so that reducer will take a lot of time to process the data and provide the output, Hence it will affect the performance of the entire framework by maximizing the execution time and this problem is generally treated as data skew problem. The chance of data skewness is more during the reduce side joining process in Hadoop Framework. To overcome from this different techniques and approaches are used. Basically hash join algorithm is used to overcome from data skewness though the hash algorithm has explained below.

A .Hash Algorithm

Step 1.Consider two datasets P and Q. The algorithm for a simple hash join will look like

Step 2.For all p <– P do

Step 3 Load p into in memory hash table H

Step 4.end for

Step 5.for all q <– Q do

Step 6.if H contains p matching with q then

Step 7 .add <p,q> to the result

end if

end for

This algorithm usually is faster than the Sort-Merge Join, but puts considerable load on memory for storing the hash-table.

V. CONCLUSION

Big Data analytics is still in the initial stage of development, since existing Big Data optimization frameworks and tools are very restricted to solve the actual Big Data problems completely, though further scientific investments from both governments and enterprises end should be contributed into this scientific paradigm to develop some new optimization techniques for big data. The paper has discussed about different approaches of joining which are generally used in Hadoop Framework during map reduce programming [12]. It also gives a brief idea about the data skewness which is generally occurred during reduce side joining operation in hadoop framework and discussed the hash algorithm which is generally used to overcome from data skewness.

VI. FUTURE SCOPE

In the case of Reduce-Side Cascade Join, the intermediate result is suitable for input to a Map-Side Join. It might be possible to run a pre-processing step in parallel to the first join such that the rest of the datasets are ready for joining by the

time the first join produces the intermediate result. This may involve exploring the scheduling mechanisms of Hadoop or any other implementation of Map/Reduce. Till now different research is going on to find out alternative techniques to overcome from data skewness.

REFERENCES

1. HUI JIANG1, KUN WANG1,YIHUI WANG, MIN GAO,YAN ZHANG, "Energy Big Data: A Survey" , Digital Object Identifier 10.1109/ACCESS.2016.2580581.

2. Tim Mattson , " HPBC 2015 Keynote Speaker - Big Data: What happens when data actually gets big " , Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International.

3. Carson K. Leung, Hao Zhang, "Management of Distributed Big Data for Social Networks ",2016 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing.

4. Mohand-Saïd Hacid , Rafiqul Haque, " Blinked Data: Concepts, Characteristics, and Challenge " , 2014 IEEE World Congress on Services.

5. Nicolas Sicard, B´en´edicte Laurent, Michel Sala, Laurent Bonnet, " REDUCE, YOU SAY: What NoSQL can do for Data Aggregation and BI in Large Repositories " 2011 22nd International Workshop on Database and Expert Systems Applications.

6. Tim Mattson , " HPBC 2015 Keynote Speaker - Big Data: What happens when data actually gets big " , Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International.

7. Rodney S. Tucker, Jayant Baliga, Robert Ayre, Kerry Hinton, Wayne V. Sorin, Energy consumption in IP networks, in:
Proceeding of Optical Communication, 2008, pp.1.

8. Shekhar Srikantaiah, Aman Kansal, Feng Zhao, Energy Aware consolidation for cloud computing, in: Proceedings of the 2008 Conference on Power Aware Computing and Systems, 2008.

9. Aman Kansal, Feng Zhao, Jie Liu, Nupur Kothari, Arka A. Bhattacharya, Virtual machine power metering and provisioning, in: Proceedings of the 1st ACM Symposium on Cloud computing, 2010, pp. 39–40.

10. Xiaobo Fan, Wolf-Dietrich Weber, Luiz Andre Barroso, Power provisioning for a warehouse-sized computer, in: Proceedings of the 34th Annual International Symposium on Computer Architecture, 2007, pp. 13–23.

11. Muhammad H. Alizai, Georg Kunz, Olaf Landsiedel, Klaus Wehrle, Power to a first class metric in network simulations, in: Proceedings of the Workshop on Energy Aware Systems and Methods, February, 2010.

12. Jayant Baliga, Robert Ayre, Wayne V. Sorin, Kerry Hinton, Rodney S. Tucker, Energy consumption in access networks, in: Proceedings of Optical Fiber Communication/National Fiber Optic Engineers, February, 2008, pp. 1–3.