Skewness a challenge for MapReduce in Big Data

[1]Vishal A. Nawale
Pune University, MIT College of engineering,
Kothrud, Pune 411038, India
vish.nawale@gmail.com

[2]Prof. Priya Deshpande
Pune University, MIT College of engineering,
Kothrud, Pune 411038, India
priyardeshpande@gmail.com

Abstract:

MapReduce has emerged as a popular computing model used for processing and analyzing big size data using commodity hardware. Now days virtualized cloud centres also have Hadoop ecosystem which open source implementation of MapReduce paradigm for hosting and analyzing big size data. Sometimes it is observed that because of uneven partitioning some reducer has more data to process than other reducer which in turn causes performance degradation. This is called as data skewness. In this paper we are going to study various techniques used to deal with data skew and partitioning strategies and various advantages and limitations of the same.

Keywords: Data Skew, MapReduce, Hadoop, Partition Skew.

I. INTRODUCTION

Big Data happens when the data you have to process is bigger than what you can process in the given time with current technologies. Massive amount of data is constantly produced through online shopping, social media, public transport, GPS, etc. This generated data which is stored for a long duration of time results into big data. The data produced in last 2 years is more than the entire digital data produced in the history of mankind. The data that is created each day amounts to about 2.5 quintillion bytes and the rate at which it is produced increases every day. Big data refers to the exponential growth and storage of such data that can be both structured and unstructured that exceeds the processing capacity of traditional database management systems.

Big data is created on different machines spread throughout the world. The major challenge is not the storage but retrieval and analysis of this data that is stored on different machine at different location. Hadoop is a framework which facilitates with distributed processing of large data sets across clusters of commodity computers using a simple programming model.

Hadoop mainly consists of a distributed file system and a MapReduce paradigm which enables for distributed storage and processing of data. The Hadoop Distributed file system consists of a NameNode that acts as a master for all the DataNodes. The NameNode orchestrates the movement of data blocks into and across the DataNode whose responsibility is to store data blocks. Similarly a job tracker directs the appropriate task tracker to perform MapReduce operation where a job tracker acts a master for all the task trackers.

MapReduce is a special form of Directed Acyclic Graph that is applicable to a wide range of use cases. The Map function transforms the input data in key-value pairs, each of which will be sorted according to the key and will be sent to same node that is assigned for each key. The Reduce function will then merge the values into a single result. Thus the combination of map and reduce function allow for distributed processing of input data.

Although MapReduce paradigm drastically reduces the execution time of a program there are some limitations associated with it. Some of the reduce functions tend to finish instantly whereas some might require considerably more time. Such limitation is a result of skewness. Skewness is the variation in the execution time of MapReduce operations because of imbalance in the amount of data assigned to each task. Some tasks require more time for execution than others, for instance, some partitions of an operation take significantly longer to process their input data than others, slowing down the entire computation. Such task that execute slower than its other companion tasks is known as a straggler. The presence of these stragglers can therefore significantly increase the job execution time and result in lower cluster throughput.

II. LITERATURE SURVEY

Today, researchers, governments, and companies are accumulating creating very large amount to process. Analysis of such big size data is challenging task because of complexity of data. Number of frameworks and tools can be used for such type of analysis. One- of the well known frameworks is Hadoop which is open source implementation of MapReduce paradigm.

A. HADOOP

Hadoop is framework based on mapreduce paradigm. Hadoop ecosystem consists of number of tools like hdfs, hive, hbase etc. HDFS is managed by daemon process called as NameNode. NameNode manages metadata for blocks. Actual file is divided into blocks of 64MB and each block is replicated at more than one node depending on the configuration file parameter called as dfs distributed file

system replication factor. Each slave node run daemon process called as DataNode. [5]

B. MAPREDUCE PARADIGM

MapReduce Paradigm is suggested at Google for distributed processing of data intensive task. Major advantages of MapReduce paradigm are

1) It can be executed on commodity hardware.

2) Scalable

3) Fault tolerant.

Application writer have to represent the actual algorithm in terms of mapper and reducer.

```
map(k1,v1){

emit(k2,v2)

}

reducer(k2,v2)

{

do some processing

write(k3,v3)

}
```

Execution of jobs submitted by the user will be handled by Job tracker daemon on master node. The actual work horses are called as TaskTracker.

C. SKEWNESS IN MAPREDUCE ENVIRONMENT

Skewness in MapReduce environment occur at

1) Mapper side

2) Reducer Side

3) Because of straggler node.

Hadoop framework works on the principal of data locality. Each mapper will get same size data to process in terms of Input Split. But the complexity of this data may vary. This causes skewness at mapper side. Sampling the input data in this case can be the solution. But still it is overhead.

The mapper output is given as input to reducer. Before that it is shuffled and ordered using partitioning scheme. Some reducer may get large size key to process causing data skewness at the reducer level.

In heterogeneous cluster environment some nodes may have less compute power and thus can hamper overall performance of the system. Such nodes are called as straggler nodes. We can avoid this thing using speculative execution where we can execute the same task on available nodes with high computing resources. But still it is overhead.

D. SOLUTIONS FOR DATA SKEWNESS

There are number of solutions suggested for data skew mitigation as follows:

In [1], Libra can be used to manage the data skewness on reduce side application. It avoids execution of pre-run sample tasks in order to decide the distribution of intermediate data. It does sampling during the normal execution of map tasks only. Reducer can immediately starts consuming the data as soon as this starting map tasks completes. Libra also partitions the keys which are large in size and also consider the heterogeneous nodes while executing this policy.

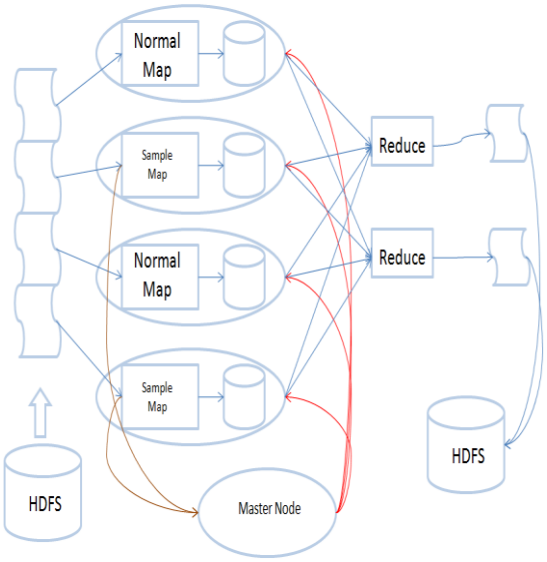

Figure 1: LIBRA system architecture[1]

In [2], author suggested efficient way for mitigating the skewness resulted from the stragglers nodes. This work consider maximum cost performance model to decide the straggler task. For this it consider the current progress rate of task using the history of execution of previous stages and accordingly it forecast the future requirement. This approach is quite scalable, which performs very well in both small clusters and large clusters. It also considers both homogeneous and heterogeneous clusters. To accurately identifying straggler nodes three methods are used by MCP as follows:

1. Uses process bandwidth and progress rate in a current phase to decide slow tasks
2. Calculates tasks remaining time and makes prediction about process speed using EWMA (Exponentially Weighted Moving Average)
3. Makes use of cost-benefit model with consideration about load on the cluster to determine which task to backup on another node

In [3], SkewTune works by identifying the idle nodes and then it offload the data on straggler task identified by considering highest expected running time as metric, to this ideal node using the efficient partitioning scheme. The system is transparent to the user and introduces very less overhead. In SkewTune MapReduce data processing engine consists of following important characteristics:

1. Consists of Coordinator-worker architecture

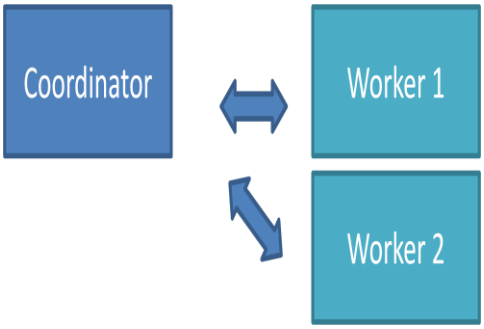

Figure 2: Coordinator-worker architecture

2. De-coupled Execution – operators execute independently of each other
3. Independent record processing
4. Per-task progress estimation – Remaining time
5. Per-task statistics – Such as total number of (un)processed bytes and records

SkewTune uses Late Skew Detection in which tasks in consecutive phases are decoupled, map tasks can process their input and produce their output as fast as possible and slow remaining tasks are replicated when slots become available. SkewTune uses range partitioning to preserve original output order of the UDO.

In [4], the author discussed data skewness at mapper and reducer side also. Mitigation at both the side reducer as well as mapper side is the main motivation behind this work. In heterogeneous environment to improve mapreduce performance author suggests load balancing approach for reduce phase, this approach is based on two components as follows:

1. It predicts performance of reducers that will run on heterogeneous nodes with the support of vector machine models.
2. Balances skewed data among reduce tasks with the help of Heterogeneity-aware partitioning (HAP).

E. Challenging Issues

In MapReduce paradigm to do analysis on huge amount of data in this scenario data skewness issue is much more important. Because of this data skew performance of MapReduce degrades and ultimately makespan of mapreduce jobs increases. To deal with skew present in mapreduce application we have to implement efficient load balancing and partitioner which removes skew to a greater extent which will improve the performance and makespan of mapreduce jobs.

F. Summary

Presence of skew in input data will degrade the performance of mapreduce jobs to deal with this problem various authors suggested their solutions, we will do some analysis on their solutions provides as follows:

In [1], LIBRA results significant performance improvement in both synthetic and real workloads. It has very negligible overhead in the absence of skew. Again it has a large cluster split and arrangement for heterogeneous environments.

In [2], MCP got success in minimizing the skew by detecting straggler nodes with improvement in cluster throughput. Limitation for this paper will be as it is considering speculative execution at the end of a stage.

In [3], SkewTune partitions the input data on straggler nodes on another idle node then also it preserves the order of output. It shows a improvement over Hadoop by a factor of 4 though there are some limitations such as:

1. While running a skew mitigation task it occupies more task slots than regular.
2. It considerably mitigates skew in reduce phase but cannot improve copy and sort phases, this may be the performance bottleneck for some applications.

In [4], PM-SVM can predict accurately the performance of reducers; large keys are also detected by load balancer. Load balancer has low overhead and achieves a speedup of 11.7% over native Hadoop. Future work for this is to consider load balancing in map phase also.

III. PROPOSED SYSTEM

A very efficient partitioning scheme will be used to partition the intermediate data which is output from map task. Partitioning algorithm calculates the frequency of key and accordingly partitions among reducers with taking

consideration of performance of reducers and also takes care of heterogeneity parameter while taking decision of partitioning.

IV. CONCLUSION

In this paper, we have studied various causes of skew in mapreduce applications and various techniques which solve or minimize the problem of skew for the jobs in mapreduce applications with this job can be resulted in minimized makespan and improves cluster throughput. We have also summarized various challenges of data skew and straggler nodes with the gaps in the current solutions by the various authors. In future work we will more focus on Data skew present in all stages of mapreduce framework and also provide solution for the partition scheme which will evenly distribute the workload on homogeneous environment and as per the performance of reducer in heterogeneous environment.

V. REFERENCES

[1] Q. Chen, J. Yao, and Z. Xiao, "LIBRA: Lightweight Data Skew Mitigation in MapReduce," Parallel and Distributed Systems, IEEE Transactions on (Volume: PP , Issue: 99), 2014.

[2] Q. Chen, C. Liu, and Z. Xiao, "Improving mapreduce performance using smart speculative execution strategy," IEEE Transactions on Computers (TC), vol. 63, no. 4, 2014.

[3] Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, "Skewtune: Mitigating skew in mapreduce applications," in Proc. of the ACM SIGMOD International Conference on Management of Data, 2012.

[4] FAN Yuanquan, WU Weiguo, XU Yunlong, CHEN Heng, Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, Shaanxi Province, P. R. China. "*Improving MapReduce Performance by Balancing Skewed Loads.*" Aug 2014 Published In Communications, China(Volume:11, Issue:8).

[5] Apache Hadoop. [Online]. Available: http://hadoop.apache.org/