Improvisation of Incremental Computing in Hadoop Architecture with File Caching

Mr.Alhad V. Alsi
M.E. 4th Sem Information Technology
Prof. Ram Meghe Institute of Technology & Research
Badnera-Amravati, India
e-mail:alhad.v.a@gmail.com

Prof. A P.Bodkhe
Principal
Prof. Ram Meghe Institute of Technology & Research
Badnera-Amravati, India
e-mail: principal@mitra.ac.in

***Abstract*—** Incremental data is a difficult problem, as it requires the continues development of well defined algorithms and a runtime system to support the continuous progress in computation. Many online data sets are elastic in nature. New entries get added with respect to the progress in the application. The Hadoop is a dedicated to the processing of distributed data and used to manipulate the large amount of distributed data. This manipulation not only contains storage but also the computation and processing of the data. Hadoop is used for data centric applications where data plays a vital role in making the decisions. Systems for incremental bulk data processing and computation can efficiently used for the updates but are not compatible with the non-incremental systems such as e.g., MapReduce, and more importantly and requires the programmer to implement application-specific incremental methodologies which ultimately increases algorithm and code complexity. Thus this paper discusses about the various aspects of the incremental computation and file caching.

Keywords- *data computing, Hadoop, MapReduce, parallel processing, distributated data systems,file caching.*

*****

I. INTRODUCTION

Traditional methods for data mining typically make the assumption that the data is centralized and static. But now a day these assumptions are no longer valid. These practises fails in processing and managing the input/output resources when data is dynamic and they impose communication overhead when data is in distributed environment. The efficient implementation of incremental data mining methods is hence becoming crucial for ensuring system scalability and facilitating knowledge discovery when data is dynamic and distributed.

MapReduce is emerging as an important programming model for data-intensive systems [9]. The model proposed by Google for dealing with bulk amount of data processing is very useful for ad-hoc parallel processing of random data. It shows good performance in case of batch parallel data manipulation and processing. MapReduce enables easy development of scalable parallel applications to process huge amount of data on large clusters of commodity environment [3]. It is a very popular open-source implementation. Hadoop, primarily developed by Yahoo and Apache, runs jobs on chunk of data. The actual application of Hadoop includes Facebook, Amazon, and Last.fm. Because of its high efficiency, scalability. MapReduce framework is used in many fields with a wide range of real time applications. Additionally, the MapReduce is used in traditional clusters, multi-core and multi-processor systems and heterogeneous environments. It is also used in systems with GPU and FPGA, and mobile systems. Such advantages attract researchers to apply it on incremental data processing [10].

A. HDFS

The Hadoop Distributed File System (HDFS) is designed to store huge amount of datasets reliably, and to stream those data sets at high bandwidth to user applications. HDFS provides scalable, fault-tolerant storage[19]. In HDFS files are stored across a collection of servers in a cluster. Files are then decomposed into blocks, and each block is written to more than one of the servers. This replication offers both fault-tolerance (loss of a single disk or server does not destroy a file) and performance (any given block can be read from one of servers. It does improves system throughput).

HDFS ensures data availability by continuously monitoring the servers in a cluster and the blocks that they manage. Individual block includes checksums. Before block is read, the checksum is verified, and if the block has been damaged it will be restored from one of its replicas stored on another node. All of the data is stored is replicated to some other node or nodes in the clusters of the respective distributed environment, from the collection of replicas when the server or disk fails,. In case of large clusters HDFS expects failure, hence organizations can spend less on servers and let software compensate for hardware issues. The resources grow with demand due distribution of storage and the computation across many servers.

B. Map and Reduce

MapReduce framework describe the techniques for simplified processing of dataset by providing simple programming model which eliminates a complex logic or infrastructure for parallel batch processing, scalability, data transfer, scheduling and fault tolerance[19]. MapReduce includes a software component called the job scheduler. It is responsible for choosing the servers and running the user job, It

also schedules multiple user jobs on shared clusters. The job scheduler communicates with the NameNode for the location of all of the blocks that make up the file or files required by a job. Each of those servers run the user's analysis code against its local block or blocks. MapReduce includes an input split that permits each block to be broken into separate individual records. The user code that implements a map job can be virtually anything. MapReduce allows developers to write and deploy code that runs directly on each DataNode server in the cluster. The code understands the format of the data stored in each block in the file. Further, it can implement simple algorithms or much more complex ones.

At the end of the map phase, results are collected and filtered by a reducer. This processing guarantees that data will be delivered to the reducer in sorted format. Hence output from all map jobs is collected and passed through a shuffle and sort process. The sorted output is passed to the reducer for further processing. Results are then written back to HDFS.

II. LITERATURE REVEIW

Frank McSherry and Rebecca Isaacs reported on the design and implementation of Naiad [6]. The paper describes a set of declarative data-parallel language extensions and an associated run time supporting computation which is efficient and incremental and iterative. This combination is enabled by a new computational model we call differential dataflow. Batch processing models such as MapReduce and Hadoop is proposed by Naid so as to support efficient incremental updates to the inputs in the manner of a stream processing system, but at the same time enabling arbitrarily nested fixed-point iteration. . In this technique incremental computation can be performed using a partial order on time. In the paper, they evaluated a prototype of Naiad that uses shared memory on a single multi-core computer. Naiad was applied to various computations, including several graph algorithms also and observe good scaling properties and efficient incremental re-computation.

Elastic Replica Management System introduces an active/standby storage model which takes advantage of a high performance complex event processing engine to distinguish the real time data types and brings an elastic replication policy for the different types of data [4]. Based on the access patterns of data , data in Hadoop can be classified into different types. Hot data - data having a large number of concurrent access and high intensity of access, while cold data- unpopular and rarely accessed data, normal data - rest of the data other than hot and cold. The ERMS introduces active/standby storage model which classifies the storage nodes into active nodes and stand by nodes. Processing efficiency depends on a number of threshold values and hence the careful selection of threshold values is needed and also memory requirement is high.

HadoopDB stores data in the local RDBMS's using ACID conforming DBMS engines which will affect the dynamic scheduling and fault tolerance of Hadoop [8]. The HadoopDB changes the interface to SQL and this is why the programming model is not as simple as Map/Reduce. Moreover changes are required to make Hadoop and Hive frameworks work together in HadoopDB. It can be utilized like parallel databases along with fault tolerance, ability to run in commodity server environments and software license cost as Hadoop and can reduce the data processing. Also, it can yield scalability, fault tolerance and flexibility of Map/Reduce systems.

Clydesdale is a research prototype built on top of Hadoop for structured data processing. It provides the key feature of MapReduce i.e.; the scalability, fault tolerance and elasticity [5].Along with that it increases the performance without making modifications to the underlying platform. To storing large amounts of dataset which further undergo processing, Clydesdale uses HOFS. Column oriented layout is used to store data and tries to collocate the columns of a given row to the specified node. For the operations in datasets Map tasks are carefully designed. SQL queries are represented as Map/Reduce programs in Java. The data structures for query processing can be shared by multiple threads and allows multiple tasks to execute successively on any node. Managing Clydesdale workloads along with Map/Reduce loads pose network load management problems and updates to dimension tables are not permitted. The Parallelism also may be limited for small data sets.

Pramod Bhatotia and Alexander Wiederwe described the implementation of a generic MapReduce framework, named Incoop, for incremental processing. It is capable of identifying the changes in the input data and enables the automatic modifications of the outputs by employing a fine-grained and efficient result re-use mechanism [13]. The efficiency is achieved by adopting recent advances in the area of programming languages to identify systematically the shortcomings of task-level memorization approaches, and address them using several novel techniques such as a storage system to store the input of consecutive iterations, a contraction phase that make the incremental computation of the reduce tasks more efficient and the scheduling algorithm for Hadoop that is aware of the location of previously computed results. The proposed framework Incoop was implemented by extending the Hadoop and further it was evaluated with a variety of real time applications. The case studies of higher-level services: incremental query (based on Pig) and log processing systems was also studied. The results showed significant performance improvements without changing a single line of application code.

III. PROPOSED WORK

The proposed system is focused on incremental computations [1]. In incremental datasets, map and reduce processes need to be run again every time even if there is some minute change in input. Thus the proposed system has suggested the caching technique to reuse the previously computed results. The states of the computations are stored . After caching would be done for the data that has big blocks upon a small change we can stop the map and same reduce job any time. In successive computation of the particular item, it checks the instance of the previously computed result is available. The result is then used and further computation on newly added data is performed. This saves the execution time and resources very well.

A. Association Algorithms

The row data was processed for the computation. The classification and clustering algorithms are used

Classification consists of predicting a certain outcome based on a given input. To predict the outcome, the algorithm processes a training set containing a set of attributes and the respective outcome. It tries to discover relationships between the attributes that would make it possible to predict the outcome. Further the algorithm is given a data set not seen before which contains the same set of attributes, except for the prediction attribute – not yet known. It analyses the input and produces a prediction. The accuracy for prediction defines how "good" the algorithm is. Main algorithms for classification are ID3 and C4.5. ID3 algorithm was originally developed by J. Ross Quinlan at the University of Sydney. ID3 algorithm induces classification models, or decision trees, from data. It is a supervised learning algorithm that is trained by examples for different classes. After being trained, the algorithm should be able to predict the class of a new item. ID3 identifies attributes that differentiate one class from another. All attributes must be known in advance, and must also be either continuous or selected from a set of known values. For instance, temperature and country of citizenship are valid attributes. To determine which attributes are the most important, ID3 uses the statistical property of entropy. It measures the amount of information in an attribute. This is how the decision tree, which will be used in testing future cases, is built. Imagine that you have a dataset with a list of predictors or independent variables and a list of targets or dependent variables. Onward, by applying a decision tree like J48 on that dataset would allow you to predict the target variable of a new dataset record[18][15].Decision tree J48 is the implementation of algorithm ID3 (Iterative Dichotomiser 3) developed by the WEKA project team. R includes this nice work into package RWeka[15].

Data Clustering is the process of making a group of abstract objects into classes of similar objects[16].

- A cluster objects of data can be treated as one group.
- In cluster analysis, the set of data is partitioned into groups based on data similar property and then assign the labels to the groups.

  The clustering has few advantages over classification. It is adaptable to changes and helps single out useful features that distinguish different groups.

IV. EXPERIMENTAL RESULT

In the proposed work, to solve the incremental processing file caching technique was used. caching is used for Retaining most recently accessed disk blocks. Repeated accesses to a block in cache can be handled without involving the disk. Caching reduce delays contention for disk arm. Cached data are still there during recovery and don't need to be fetched again.
Thus the results of the computations are stored in the cache. The results are then used further for the next iteration. The proposed work analyze the execution time in case of caching and without using the caching.

Following procedure is followed for the analysis of proposed solution:

- The row dataset of the super market is obtained for processing. It undergoes the cleaning procedure.
- Classification and clustering algorithms are applied on data. While the row data is also maintained.
- The structured data is maintain. The incremental processing without caching is performed on this structured data. in this case the states are not saved. Thus the reusability of the previous processing is not possible.

The proposed methods are comparing the processing with the parameter "Execution Time". The execution time for the job was observed in both the cases. The execution time in the proposed solution was much lesser than in case of the incremental computing. The proposed solution utilizes the Map and Reduce in a such fashion that the iterative Map and corresponding Reduce functions get optimized.

V. CONCLUSION

The proposed paper addresses the reusability of processed results in incremental data and computational techniques and/or methodologies. This new approach to incremental data computing is proposed for HDFS which will cluster similar documents in the same set of data nodes with minimal changes to the existing framework. In distributed systems, iterative computation computations can't be avoided. hence to improve the throughput, caching technique of iterative computations have been mentioned in this paper. Storing the states of the previously processed dataset proves beneficial when that dataset is incremental. Thus, to solve the issue of incremental computation file caching introduced the new approach towards the run time reusability in case of distributed environment.

ACKNOWLEDGMENT

I extend my sincere thanks to Prof. A. P. Bodkhe, Principal, Prof. Ram Meghe Institute of Technology & research, Badnera India, for his valuable support and guidance in pursuing my research work. My thanks are also to Prof. Ms.V.M.Deshmukh, Head of the department, Prof. Ram Meghe Institute of Technology & Research , Badnera India for her encouragement and continuous supervision.

REFERENCES

[1] Anam Alam and Jamil Ahmed."Hadoop Architecture and Its Issues".2014. International Conference on Computational Science and Computational Intelligence.2014

[2] Yanfeng Zhang,Qixin Gao,Lixin Gao,Cuirong Wang."iMapReduce: A Distributed Computing Framework for Iterative Computation".Springer.2012.

[3] Jeffrey Dean,Sanjay Ghemawat."MapReduce: Simplified Data Processing on Large Clusters".Communications of the ACM - 50th anniversary issue: 1958 - 2008 CACM Homepage archive Volume 51 Issue 1, January 2008.Pages 107-113

[4] Zhendong Cheng, Zhongzhi Luan,You Meng, Yijing Xu, Depei Qian."ERMS : An Elastic Replication Management System for HDFS".IEEE International Conference on Cluster Computing Workshops.2012

[5] Tim Kaldewey,Eugene J. Shekita,Sandeep Tata."Clydesdale: Structured Data Processing on MapReduce".EDBT '12 Proceedings of the 15th International Conference on Extending Database Technology.2012

[6] Frank McSherry Rebecca Isaacs Michael Isard Derek G. Murray."Composable Incremental and Iterative Data-Parallel Computation with Naiad". Microsoft Research Paper,2012.

[7] Matthew Eric Otey, Srinivasan Parthasarathy."Parallel and Distributed Methods for Incremental Frequent Itemset Mining". IEEE Transactions on systems,man and cybernetics-Part B: Cybernetics,Vol.34,No 6. December 2004

[8] Azza Abouzeid, Kamil Bajda-Pawlikowski,Daniel Abadi, Avi Silberschatz, Alexander Rasin."HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads ".Journal Proceedings of the VLDB Endowment VLDB Endowment Hompage archive Volume 2 Issue 1. Pages 922-933. August 2009

[9] MapReduce Design Patterns by Donald Miner and Adam Shook

[10] Hadoop Cluster Deployment by Danil Zburivsky

[11] Matthew Hayes,Sam Shah."Hourglass: a Library for Incremental Processing on Hadoop".2013 IEEE International Conference on Big Data.2013

[12] Cairong Yan,Xin Yang, Ze Yu, Min Li, Xiaolin Li."IncMR: Incremental Data Processing based on MapReduce".2012 IEEE Fifth International Conference on Cloud Computing.2012

[13] Pramod Bhatotia, Alexander Wieder, Rodrigo Rodrigues, Umut A. Acar, Rafael Pasquini."Incoop: MapReduce for Incremental Computations". Proceedings of the 2nd ACM Symposium on Cloud Computing Article No. 7 ACM New York, NY, USA .2011

[14] http://hadoop.apache.org

[15] Dr. Neeraj Bhargava, Girja Sharma,Dr. Ritu Bhargava,Manish Mathuria."Decision Tree Analysis on J48 Algorithm for Data Mining ".International Journal of Advanced Research in Computer Science and Software Engineering,Volume 3, Issue 6, June 2013.

[16] Aastha Joshi, Rajneet Kaur." A Review: Comparative Study of Various Clustering Techniques in Data Mining". International Journal of Advanced Research in Computer Science and Software Engineering. Volume 3, Issue 3, March 2013.

[17] Gaganjot Kaur, Amit Chhabra." Improved J48 Classification Algorithm for the Prediction of Diabetes". International Journal of Computer Applications, Volume 98 – No.22, July 2014.

[18] Rupali Bhardwaj , Sonia Vatta. "Implementation of ID3 Algorithm ". International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 6, June 2013

[19] Umesh V. Nikam, Anup W. Burange2. "Big Data and HADOOP: A Big Game Changer".International Journal of Advance Research in Computer Science and Management Studies,Volume 1, Issue 7, December 2013