Improve Dynamic Replication Management in Hadoop Distributed File System (HDFS)

V. Siva Krishnaveni[1], M. Praveena[2]
[12]*Lecturer, Dept. Of Computer Science, Sri Durga Malleswara Siddhartha Mahila Kalasala, A.P., India.*

Abstract- Apache Hadoop is one of the best known platforms for distributed storing and processing of big data across clusters of computers. The storage component of Hadoop, Hadoop Distributed File System (HDFS) maintains a default replication factor for each file as three, which is placed in separate nodes. HDFS provides high performance access to data by applying a static and default replication strategy. Though HDFS ensures high reliability, scalability and high availability, its static and default approach in data replication requires large amount of storage space. With a replication factor of three, a file is copied three times in different nodes. In this paper we propose an efficient dynamic data replication management system, which consider the popularity of files stored in HDFS before replication. This strategy dynamically classifies the files to hot data or cold data based on its popularity and increases the replica of hot data by applying erasure coding for cold data. The experiment results show that the proposed method effectively reduces the storage Utilization up to 50% without affecting the availability and fault tolerance in HDFS.

Keywords- Big data, Distributed File System, Dynamic data replication, Optimization, reliability.

I. INTRODUCTION

In HDFS, in order to ensure data availability and to reduce the chance of data loss, each file is replicated across a number of machines. The default replication factor in HDFS is to create three replicas for each file. HDFS replication strategy will not consider whether a particular file is popular or not. Unnecessary replication of non-popular file will result in storage overhead. In the proposed strategy, a dynamic data replication algorithm is used to manage the replicas in HDFS. The Replication Management System in the proposed algorithm manages the replication of files in HDFS. This module classifies the data files into hot data or cold data. After classifying the data, the replication factor is increased for the hot data and its replica is placed in data node. For placement of replicated data Hadoop's random placement strategy is used. Erasure coding is applied for cold data to ensure availability. Replication Management System does these tasks with the help of HDFS Logging System. The logging system provide details such as the number of files accessed, their source, the nodes which accessed them, frequency of access for each file, etc.

The Logging system obtains all these information from HDFS and provides it to the Replication management System. a Hadoop cluster was setup comprising of 10 nodes. The physical Hadoop cluster comprises of one master node and nine slave nodes and the version of a Hadoop distribution is 2.7. The master node acts as both name node and data node, thus a cluster of ten data nodes is formed. Each node is equipped with Intel Core i5 (3.30GHz) CPU and 8 GB RAM. Files were copied into HDFS from the local file system. Files of different size and types are considered for the experimentation of the algorithm. Various types of files including text, audio, video files with sizes ranging from The 600MB to 4GB were considered for this purpose. The files were divided into blocks by HDFS with default block size of 128MB. Initially, the replications for the files were three, which is the default replication count in HDFS. These files were accessed randomly from different nodes, at different time intervals. The log files were analyzed. Then in regular intervals the algorithm is executed in the HDFS cluster. The algorithm checks the access count for each file and calculates their popularity. Based on this popularity value and threshold, files are classified into two – hot or cold. Replication count for hot files is incremented and cold files are encoded using Reed-Solomon erasure code. The performance of the algorithm was analyzed comparing the result obtained by using Hadoop default replication strategy.

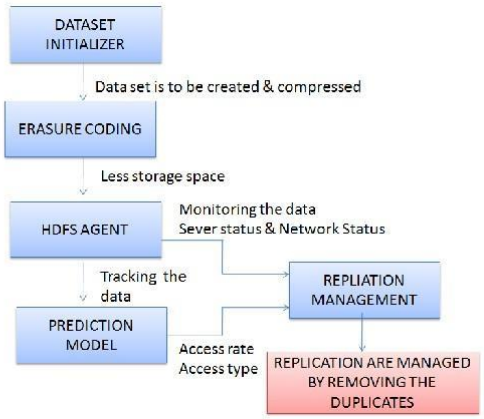

Fig.1: Block Diagram of HDFS

II. LITERATURE SURVEY

Enabling proactive data management in virtualized hadoop clusters based on predicted data activity patterns
AUTHORS: G. Kousiouris, G. Vafiadis, and T. Varvarigou
Hadoop clusters are gaining more and more in popularity based on their ability to parallelize and complete large scale computational tasks on big data. Service offerings of this type have appeared in the recent years, covering a need for dynamic and on-demand creation of such data analytics frameworks. The aim of this paper is to provide a mechanism for offering such virtual clusters as a service, with built-in intelligence functionalities for efficient management. The target of these mechanisms is to predict future demand of the files in the HDFS cluster and dynamically manipulate the according replication factor for availability purposes, in order to improve performance and minimize storage overhead.
To this end, real data have been utilized as a dataset input to the prediction framework, based on Fourier series analysis, due to the latter's ability to capture different periodicities that can influence service usage. Multiple time-step ahead prediction is performed in order to enable proactive management (e.g. suitable replication strategy). We describe the framework's architecture, necessary modifications to the client side of Apache Hadoop for data logging and the results of the applied method on two real world datasets. This paper presents a cost-effective dynamic replication management scheme referred to as CDRM. A novel model is proposed to capture the relationship between availability and replica number. CDRM leverages this model to calculate and maintain minimal replica number for a given availability requirement. Replica placement is based on capacity and blocking probability of data nodes. By adjusting replica number and location according to workload changing and node capacity, CDRM can dynamically redistribute workloads among data nodes in the heterogeneous cloud. We implemented CDRM in Hadoop Distributed File System (HDFS) and experiment results conclusively demonstrate that our CDRM is cost effective and outperforms default replication management of HDFS in terms of performance and load balancing for large-scale cloud storage.

III. PROPOSED SYSTEM

❖ We designed an adaptive replication management (ARM) system to provide high availability for the data in HDFS via enhancing the data locality metric. As a result, the highly local available data improves the performance of the Hadoop system. It is worth noting that the erasure code is applied to maintain the reliability.
❖ We proposed a complexity reduction method for the prediction technique in both hyper-parameter learning and training phases. This proposed method significantly increases the performance in terms of reaction rate for the replication strategy while still keeping the accuracy of the prediction.
❖ We implemented ARM in HDFS and did an evaluation in order to practically verify the effectiveness of the proposed method as compared with the state of the art method.

ADVANTAGES OF PROPOSED SYSTEM

❖ The main function of the proposed architecture is to dynamically scale the replication factors as well as to efficiently schedule the placement of replicas based on the access potential of each data file.
❖ Additionally, to reduce the calculation time, the knowledge base and heuristic technique are implemented to detect the similarity in the access pattern between in-processing files and the predicted ones.
❖ By definition, the access pattern is actually a set of eigenvectors describing the feature properties of processed data.
❖ Two files with similar access behaviors are treated with the same replication strategy. However, because these techniques are minor parts and popularly used in various systems, discussing them is not within the scope of this paper.

IV. REALATED WORK

Dataset Initializer:
Before a new transactional replication, the Local file system must contain files with the same sequence and data at the system. The initial dataset is typically a file that is created and compressed and applied by the Distribution Agent. The initial dataset can also be supplied through a backup or other means, then it is been converted to the Sequence file.
Replication places shared locks on all files published as part of replication for the duration of initial generation. This can prevent updates from being made on the publishing files. Concurrent file processing, the default with transactional replication, does not hold the share locks in place during the file generation, which allows users to continue working uninterrupted while replication creates initial files.
Erasure Code Approach:
In order to maintain the fault tolerance for less frequently accessed data files, an open source erasure code is modified and applied to protect the system from the effects of failures. By containing only one copy of each data block, the erasure coding approaches have to reunify the data blocks remotely. Even when HDFS reads the erasure coded blocks in parallel, the processing time is still lengthened by the unavailable computing node.
Even though the replication process is fully postponed for the erasure coded files, the minimum replicas of the files are still kept so they can provide access without inducing degradation on the reading. Specially, the erasure coded file

still has a chance to return to the replication set if there is any remote access firing for its remaining replicas. When this occurs, the restriction on the original file is simply lifted, enabling replication once more. To deal with any potential failures, the modified HDFS would search the replication set first. If there is no functional copy of the file, the erasure coding reconstruction process is triggered to fix the problem. Due to the nature of the proposed method, it could be considered as a hybrid solution of erasure code and replication.

Log Reader:

The Log Reader Agent typically runs continuously, but can also run according to a schedule you establish. When executing, the Log Reader Agent first reads the publication transaction log (the same database log used for transaction tracking and recovery) The agent copies those transactions in batches to the distribution database at the Distributor. The Log Reader Agent uses the internal stored procedure to get the next set of commands marked for replication from the log. The dataset analyzer then becomes the proc. Only committed transactions are sent to the distribution database. After the entire batch of transactions has been written successfully to the distribution database, it is committed.

Prediction Model

The objective of prediction in the proposed approach (ARM) is to anticipate the access potential of the data file. To obtain this target, the Bayesian learning and Gaussian process are employed as the inference technique and probability framework, respectively. These techniques are supposed to perform on the access rate to create the access potential. For the access type, because of the different impacts between the local access and remote access to the transmission rate, a weighted access scheme has been engaged to benefit the localization.

The idea of the weighted access scheme is that it reflects the viewpoint that the data files possessing higher remote access rate should have a higher potential to be replicated than the others. By using this factor, not only the access rate, but also the access type can contribute to scale the access potential.

Replication Manager

Finally Replication Manager runs over the Hadoop processing system and generates a loop to replicate the file in the Hadoop systems. The replication manager manages the replicas over the Hadoop system. It achieves the high reliability.

V. CONCLUSION & FUTURE WORK

Big Data is high-volume, high-velocity and high- variety information that demands cost-effective, innovative forms of information processing for enhanced insight and decision making. The extraction, storage and processing of big data is beyond the ability of traditional data processing techniques., to improve the availability of HDFS by enhancing the data locality, our contribution focuses on following points. First, we design the replication management system which is truly adaptive to the characteristic of the data access pattern. The approach not only pro-actively performs the replication in the predictive manner, but also maintains the reliability by applying the erasure coding approach. Second, we propose a complexity reduction method to solve the performance issue of the prediction technique. In fact, this complexity reduction method significantly accelerates the prediction process of the access potential estimation. Finally, we implement our method on a real cluster and verify the effectiveness of the proposed approach. With a rigorous analysis on the characteristics of the file operations in HDFS, our uniqueness is to create an adaptive solution to advance the Hadoop system.

In future work can be improved by optimizing the placement strategy of replicas in Hadoop cluster. Also a comparison can be made based on the performance of proposed strategy and existing method in Hadoop.

VI. REFERENCES

[1]. "What is apache hadoop?" https://hadoop.apache.org/, accessed: 2015-08-13.

[2]. M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling," in Proceedings of the 5th European conference on Computer systems. ACM, 2010, pp. 265–278.

[3]. K. S. Esmaili, L. Pamies-Juarez, and A. Datta, "The core storage primitive: Cross-object redundancy for efficient data repair & access in erasure coded storage," arXiv preprint arXiv:1302.5192, 2013.

[4]. G. Ananthanarayanan, S. Agarwal, S. Kandula, A. Greenberg, I. Stoica, D. Harlan, and E. Harris, "Scarlett: Coping with skewed content popularity in mapreduce clusters." in Proceedings of the Sixth Conference on Computer Systems, ser. EuroSys '11. New York, NY, USA: ACM, 2011, pp. 287–300. [Online]. Available: http://doi.acm.org/10.1145/1966445.1966472

[5]. G. Kousiouris, G. Vafiadis, and T. Varvarigou, "Enabling proactive data management in virtualized hadoop clusters based on predicted data activity patterns." in P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on, Oct 2013, pp. 1–8.

[6]. A. Papoulis, Signal analysis. McGraw-Hill, 1977, vol. 191.

[7]. Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, "Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster." in Cluster Computing (CLUSTER), 2010 IEEE International Conference on, Sept 2010, pp. 188–196.

[8]. C. L. Abad, Y. Lu, and R. H. Campbell, "Dare: Adaptive data replication for efficient cluster scheduling." in CLUSTER. IEEE, 2011, pp. 159– 168.

About Authors:

V. SIVA KRISHNA VENI is presently working as Lecturer in Computer Science Department, Sri Durga Malleswara Siddhartha Mahila Kalasala, Vijayawada. Subjects interested in cloud computing and big data.

M. PRAVEENA is presently working as Lecturer in Computer Science Department, Sri Durga Malleswara Siddhartha Mahila Kalasala, Vijayawada. Subjects interested in cloud computing and big data.