Adaptive Replication Management using Predictor for HDFS

[1]I.Sugapriya,[2]K.BhuvaneswariM.Sc[SW].,
[1]Pursuing M.E, Dept of CSE
[2]Assistant Professor (SG), Department of Computer Science and Engineering

Abstract—The number of applications based on Apache Hadoop is dramatically increasing due to the robustness and dynamic features of this system. At the heart of Apache Hadoop, the Hadoop Distributed File System (HDFS) provides the reliability and high availability for computation by applying a static replication by default. However, because of the characteristics of parallel operations on the application layer, the access rate for each data file in HDFS is completely different. Consequently, maintaining the same replication mechanism for every data file leads to detrimental effects on the performance. By rigorously considering the drawbacks of the HDFS replication, this paper proposes an approach to dynamically replicate the data file based on the predictive analysis. With the help of probability theory, the utilization of each data file can be predicted to create a corresponding replication strategy. Eventually, the popular files can be subsequently replicated according to their own access potentials. For the remaining low potential files, an erasure code is applied to maintain the reliability. Hence, our approach simultaneously improves the availability while keeping the reliability in comparison to the default scheme. Furthermore, the complexity reduction is applied to enhance the effectiveness of the prediction when dealing with Big Data.

Index Terms—Replication, HDFS, proactive prediction, optimization, Bayesian learning, Gaussian process

I. INTRODUCTION

The evolution of big data has created a phenomenon in application and solution development to extract process and store useful information. Apache Hadoop is one of the most renowned parallel frameworks. Hadoop Distributed File System (HDFS) has been introduced to provide the reliability and high-throughput access for data-centric applications. To improve the reliability, HDFS is initially equipped with a mechanism that uniformly replicates three copies of every data file. This strategy is to maintain the requirements of fault tolerance. Reasonably, keeping at least three copies makes the data more reliable and more robust when tolerating the failures. However, this default replication strategy still remains a critical drawback with regards to the performance aspect.

II.RELATED WORK

2.1 Pro-Active approach for Replication

The Scarlett solution [1] implements the probability as an observation and then calculates the replication scheme for each data file. The storage budget-limitation is also considered as a factor when distributing the replicas. Although this solution follows a proactive approach instead of using thresholds, the access rate of the data file as well as the suitable placement for replicas is not discussed thoroughly.

Likewise in OPTIMIS [2], an interesting solution for anticipating the data file status has been proposed. In this approach, the data file is classified and engaged in the limited replication scenarios based on the algorithmic prediction of the demand for data file utilization. However, the Fourier series analysis algorithm [3], which is usually used in the field of 'signal processing', is chosen for prediction without a compelling proof of the efficacy. As a consequence, this inappropriate choice may result in poor prediction.

2.2 Reactive approach for Replication

For the reactive approach, the cost-effective dynamic replication management (CDRM) method [4] is a cost-effective framework for replication in a cloud storage system. When the workload changes, CDRM calculate the popularity of the data file and determine the location in the cloud environment.

However, this technique follows a reactive model. As a result, by using threshold values, CDRM cannot adapt well to the rapid evolution of large-scale systems. Similarly, DARE [5] is another reactive model of replication for HDFS. In this model, the authors declare that the probabilistic sampling and competitive aging algorithms are used independently on each node to choose a replication scheme for each data file, as well as to decide the suitable location for each replica. However, there are two issues in this approach. First, the problem of long tasks, which existsin a realistic system, is not considered carefully. In fact, the existence of this issue makes the system unstable. Second, the placement of the replication is judged without consideringthe file access pattern and system capacity of the destination nodes. For these reasons, DARE might not provide the expected effectiveness on some systems.

The elastic replication management system (ERMS) [6]takes into account an active/standby model for data storage in the HDFS cluster by implementing the complex event processing method to classify the data types. The advantage of ERMS as compared with CDRM and DARE is that it dynamically changes the thresholds for metrics based on the status of the HDFS cluster system. In addition, ERMS is equipped with the erasure code to determine and erase unpopular replicas so as to save the storage.

Nevertheless, although CDRM, DARE and ERMS are developed in different ways, all of them encounter the same problems and limitations. Concretely, these solutions try to classify and implement various replicating scenarios for each type of data files by extracting and processing the obsolete information. For that reason, these approaches cannot generate an optimal replication strategy for parallel systems. The detail of this claim is that when some actions are chosento handle the 'hot' data files, due to high latency and delay, these files may not be 'hot' anymore by the time the actions are engaged. As a consequence, the replicating decision cannot reflect the trends of the data utilization. Additionally, in the ERMS approach, the erasure code configuration is notclearly specified. For that reason, the storage-reliability of this approach is still not verified.Discussions on erasure code are interesting. Commonly, it is accepted that not only the performance, but also the reliability is the mandatory aspect of HDFS. To fulfill this requirement, the replication and the erasure code are two types of fault tolerance techniques trying to obtain the samegoal. Other methods employed in erasure coding are Pyramid codes and HDFS-Xorbas, which follow theapproaches of the local reconstruction code and the locally repairable codes respectively.

Unlikethe MDS coding family, Pyramid codes and HDFS-Xorbas offerbetter repairable features in terms of network bandwidthand disk I/O.

However, the disadvantage of these methodsis the higher magnitude of storage overhead as comparedwith HDFS-RAID. Storage Corewhich is based on HDFS-RAID. In this solution, the standard erasure code and RAID-4-like parity are combined to establish the redundancy. By applying thiscombination, the solution offers an impressive improvementin reparability and read performance. Thisdecision makes storage Core suitable for ready integrationwith other methods in HDFS. However, the disadvantage ofstorageCore is an increase in storage overhead of up to 20percent in comparison to HDFS-RAID.

According to a number of studies on erasure code, it isclear to see that this branch of fault tolerance possesses animportant role in maintaining the reliability for Big Datasystem. However, by containing only one copy of each datablock, the erasure coding approaches have to reunify thedata blocks remotely. Even when HDFS reads the erasurecoded blocks in parallel, the processing time is still lengthenedby the unavailable computing node. To solve thisissue, most of the approaches choose to utilize the degradedreading, which actually mitigates the unavoidable drawback.However, this point of design actually reduces thethroughput and indirectly increases the computation time.

By examining the related works, led to theconclusion that although the research on replication and erasurecode exists, not many researchers have thoroughlyattempted to balance the data locality and the reliabilitywithin a reasonable cost of storage resource. Furthermore,since Hadoop is gradually a complex ecosystem, a faster andmore adaptive replication mechanism must be developed.

The high datalocality is critical to the performance and the availability of HDFS. Theoretically, our prediction technique intends toimprove the data locality by creating the individual replication scheme for each data file based on its own access potential. Naturally, some popular data files may have morereplicas than others because these files possess more potential to be utilized by various tasks. The percentage of high potential files can be measured in less than 10 percent.

On the other hand, over 90 percent of data files might have a minimum access potential. As a consequence, the replication for this large percentage of data files should be limited,as they are clearly not necessary to perform the prediction. A suitable strategy in this situation is to leave these low access potential files with only one replica. This strategy might save some space as well as reduce the computation, but it also reduces the reliability and puts the whole system in danger if any failure happens. In order to maintain the reliability, the idea is to partially integrate the erasure coding solution so as to take care of the low access potential files. For reference, the storage core is chosen because of the efficiency in terms of network bandwidthand computation costs. The details of the entire erasure coding process are not the focus of this research and can be found in the original paper.

As time progresses, thedata files are divided into two sets: the replication set and theerasure set. Only the files in the replication set have their access potentials calculated and replicated over to the system. When the access potential of a file decreases and thusresults in its replica quantity being less than or equal to the minimum number of replicas, the status of the file is marked as restricted for replicating and this file is moved to the erasure set. As soon as the file transfer is finished, the erasurecoding process begins, encoding this file at the block level. Even though the replication process is fully postponed for the erasure coded file, the minimum replicas of the file are still kept so they can provide access without inducing degradation on the reading.

Specially, the erasure coded file stillhas 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 setfirst. 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.

2.3 SYSTEM OVERVIEW

With the help of probability theory, the use of each data file can be predicted to create a matching replication scheme. Finally, the popular files can be consequently replicated according to their particular access possibilities. For the remaining low possible files, an erasure code is applied to maintain the reliability. This project plans an adaptive replication management (ARM) system to provide high availability for the data in HDFS through enhancing the data locality metric. As a result, the highly local available data improves the performance of the Hadoop system. Adaptive replication management is implemented (ARM) in HDFS and an estimate in order to practically verify the effectiveness of the proposed method.

Hyper-parameter settings could have a big impact on the prediction accuracy of the trained model. Optimal hyper-parameter settings often differ for different datasets. Therefore they should be tuned for each dataset. Since the training process doesn't set the hyper-parameters, there needs to be a Meta process that tunes the hyper-parameters. This is calledhyper-parameter tuning.

III.PROBLEM STATEMENT

Data replication has been widely used as a mean of increasing the data availability of storage systems such as Google file system, Ceph file system and HDFS. If multiple copies of block exist on different datanodes, the chances of at least one copy being accessible increase. When one data node fails, the data is still available from the replicas and service need not be interrupted. While replication has above advantage, the management cost will significantly increase with the number of replica increasing. Too much replicas may not significantly improve availability, but bring unnecessary spending instead.

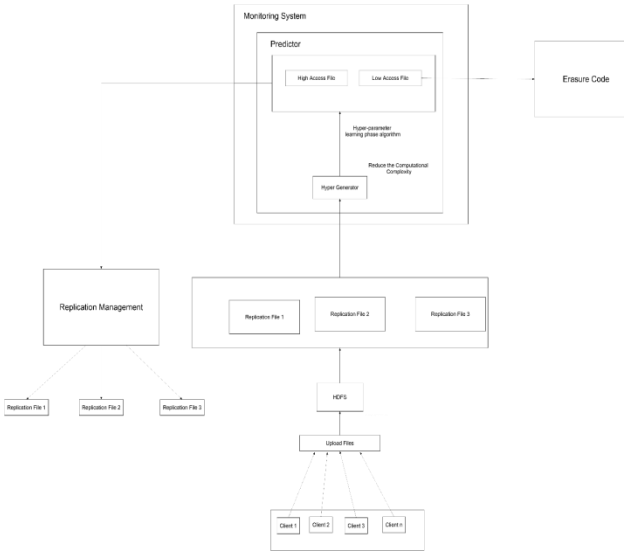
The block size is the unit of work for the file system. Every read and write is done in full multiples of the block size. The block size is also the smallest size on disk a file can have. If the 16 byte Block size, then a file with 16 bytes size occupies a full block. If the replication files are higher than the number of block, the replication will continue according to block size. Otherwise the replication process is stopped.

IV.SYSTEM MODEL

A.PROPOSED SYSTEM

The proposed method significantly increases the performance in terms of response rate for the replication strategy while still keeping the accuracy of the prediction. With the help of probability theory, the use of each data file can be predicted to create a matching replication scheme. Finally, the popular files can be consequently replicated according to their particular access possibilities. For the remaining low possible files, an erasure code is applied to maintain the reliability.Replicating uniformly or increasing the replication factor is not the key to accelerate the computation as well as reduce the slot contention and hot-spot issue. To minimize the effect of slot contention and hot-spot issue, many approaches choose to improve the data locality and conduct the load balancing. Most of existing methods are either maladaptive or

inaccurate to provide the suitable replication strategies coping with various data access patterns. It is worth noting that beside the growth in storage cost, the diversity of data access patterns is more critical affecting the performance, the replica management and the balance of the system



PREDICTOR MANAGEMENT

The purpose of this section is to describe how the replication management chooses the placement for the replica. Theoretically, by placing the potential replicas on low utilization nodes (low blocking rate nodes), the replication management helps to redirect the tasks to these idle nodes and balance the computation. The blocking rate is calculated based on the information provided by the monitoring system. The monitoring system is simple, robust and easy to configure for monitoring most of the required metrics. After plugging into the HDFSnodes, the monitoring system can collect statistics. The only extra information is thesystem statistic, which consists of CPU utilization, RAM utilization, disk I/O and network bandwidth. This design helps to unify the data sources for computational convenience, especially for blocking rate calculation.

HYPER-PARAMETER LEARNING PHASE ALGORITHM

Hyper-parameter settings could have a big impact on the prediction accuracy of the trained model. Optimal hyper-parameter settings often differ for different datasets. Therefore they should be tuned for each dataset. Since the training process doesn't set the hyper-parameters, there needs to be a meta process that tunes the hyper-parameters. Hyper-parameter tuning is imposed.

Hyper-parameter tuning is a meta-optimization task. Each trial of a particular hyper-parameter setting involves training a model an inner optimization process. The outcome of hyper-parameter tuning is the best hyper-parameter setting, and the outcome of model training is the best model parameter setting.

For each proposed hyper-parameter setting, the inner model training process comes up with a model for the dataset and outputs evaluation results on hold-out or cross validation datasets. After evaluating a number of hyper-parameter settings, the hyper-parameter tuner outputs the setting that yields the best performing model. The last step is to train a new model on the entire dataset (training and validation) under the best hyper-parameter setting. The high data locality is critical to the performance and the availability of HDFS. Theoretically, our prediction technique intends to improve the data locality by creating the individual replication scheme for each data file based on its own access potential. Naturally, some popular data files may have more replicas than others because these files possess more potential to be utilized by various tasks.

CONCLUSION AND FUTURE WORK

An adaptive replication management (ARM) system is designed to provide high availability for the data in HDFS by enhancing the data locality metric. The highly local available data increased the presentation of the Hadoop system. To maintained 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. This scheme proposed a complexity reduction method for the prediction technique using the hyper-parameter learning phase algorithm. This proposed method is increased the performance in terms of response rate for the replication strategy even though protected the accuracy of the prediction. The probability theory, the used of each data file can be predicted to create a matching replication scheme. ARM is implemented in HDFS and organized an evaluation in order to practically verify the effectiveness of the proposed method as compared with the state of the art method.

The future work persists on improve the mechanism of replica's placement and re-movement in the future to get the minimum access potential of the topology. Also it is intend to enhance the performance of the system and believe that the proposed system will be cost effective and efficient utilization of computer machines.

REFERENCES

[1] 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 Proc. 6th Conf. Comput.Syst., 2011, pp. 287–300.

[2] G. Kousiouris, G. Vafiadis, and T. Varvarigou, "Enabling proactive data management in virtualized hadoop clusters based on predicted data activity patterns," in Proc. 8th Int. Conf. P2P, Parallel,Grid, Cloud Internet Comput., Oct. 2013, pp. 1–8.

[3] A. Papoulis, Signal Analysis. New York, NY, USA: McGraw-Hill, 1977, vol. 191.

[4] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, "Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster," in Proc. IEEE Int. Conf. Cluster Comput., Sep. 2010,
pp. 188–196

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

[6] Z. Cheng, Z. Luan, Y. Meng, Y.Xu, D. Qian, A. Roy, N. Zhang, and G. Guan, "Erms: An elastic replication management system for hdfs," in Proc. IEEE Int. Conf. Cluster Comput. Workshops, Sep. 2012, pp. 32–40.

[7] M. Sathiamoorthy, M. Asteris, D.Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "Xoring elephants: Novel erasure codes for big data," Proc. VLDB Endowment, vol. 6, no. 5, pp. 325–336, 2013.

[8] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: A scalable and fault-tolerant network structure for data centers," ACM SIGCOMM Comput. Commun.Rev., vol. 38, no. 4, pp. 75–86, 2008.

[9]` B. Calder, J. Wang, A. Ogus, N. Nilakantan, A. Skjolsvold, S. McKelvie, Y. Xu, S. Srivastav, J. Wu, H. Simitci, et al., "Windows azure storage: a highly available cloud storage service with strong consistency," in Proc. 23rd ACM Symp. Oper. Syst. Principles, 2011, pp. 143–157.

[10] A. Duminuco and E. Biersack, "Hierarchical codes: How to make erasure codes attractive for peer-to-peer storage systems," in Proc. 8th Int. Conf. Peer-to-Peer Comput., 2008, pp. 89–98.