International Journal for Modern Trends in Science and Technology
Volume: 03, Special Issue No: 02, March 2017
ISSN: 2455-3778
http://www.ijmtst.com

Flexible Replication Management for Frequently Accessed Data Files in HDFS Using Hadoop

V.Susmitha[1] | P.Poojitha[2] | Ch.Aasritha[3] | V.Lavanya[4] | V.Vidya Sagar[5]

[1,2,3,4,5]Department of IT, Andhra Loyola Institute of Engineering & Technology, Vijayawada, Andhra Pradesh, India.

To Cite this Article
V.Susmitha, P.Poojitha, Ch.Aasritha, V.Lavanya and V.Vidya Sagar, "Flexible Replication Management for Frequently Accessed Data Files in HDFS Using Hadoop ", *International Journal for Modern Trends in Science and Technology*, Vol. 03, Special Issue 02, 2017, pp. 24-31.

ABSTRACT

Many applications based on Apache Hadoop are greatly enlarging due to the vast 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 data processing by applying a static replication by default. Due to the characteristics of parallel operations on the application layer, the access rate for each data file in HDFS is absolutely different. Consequently, maintaining the same replication mechanism for every data file leads to adverse effects on the performance. In considering defects of the HDFS replication, this paper introduce an access to dynamically replicate the data file depend on the predictive analysis. With the help of probability theory, the utilization of each data file can be anticipated to create a analogues replication strategy. Finally, the popular files can be afterwards replicated according to their own access potentials. For the halting low potential files, an erasure code is applied to maintain the reliability. Hence, our approach together improves the availability while keeping the reliability in connection to the default scheme.

KEYWORDS: *Replication, HDFS, Proactive Prediction, Optimization, Bayesian Learning, Gaussian Process.*

I. INTRODUCTION

The enlargement of big data has build a anomaly in application and solution development to store, access, process and retrieve useful information as to deal with new challenges. Everywhere, Apache Hadoop is one best acclaimed parallel framework. Not alone is it used to achieve high availability, Apache Hadoop is also designed to handle and detect the failures as well as maintains the data consistency. Improving with the development of Apache Hadoop, the Hadoop Distributed File System (HDFS) has been introduced to provide the reliability and high-throughput access for data centric applications. Gradually, HDFS has become a suitable storage framework for parallel and distributed computing, especially for Map Reduce engine, which was originally developed by Google to cope with the indexing problems on big data.

To improve the reliability, HDFS is uniformly replicates three copies of every data file supplied with a mechanism initially. Mainly this strategy is to maintain the requirements of fault tolerance. To make the data more reliable, more robust and tolerating the failures by fairly we can keep at least three copies. Moreover, this default replication strategy still remains a demanding drawback

According to the performance aspect, the main ambition of inventing Apache Hadoop

- To achieve better performance in data possible manipulation and processing .

- Therefore, this function should be studied carefully at every component level.
- According to the performance aspect, based on the well-known research of delay scheduling.
- If the task is placed closer to the required data source, the system can achieves faster computation and better availability.

The metric measures the distance between the task and the corresponding data source can be referred to as the data locality metric. The main reason for the improvement is twofold.

- Mainly to reduce network overhead on runtime due to the availability of the local data, by that inter-communication is needed to transfer the required data from the remote nodes.
- However, it is clear that the computation can start immediately on the input data which is locally available, and so no extra task-scheduling effort is consumed.
- Consequently, it is meaningful to say that improving the data locality would immensely enhance the system performance in terms of availability and calculation time.

Although there are some studies on this subject matter, very few proactive solutions are proposed that rigorously consider the nature of the job workload. Due to the fact that workload in Apache Hadoop consists of short and long tasks together, these tasks should be handled fairly to accelerate the computation. Typically, the Fair scheduler and delay scheduling algorithm provide the optimal data locality when the system is filled with head-of-line jobs and short tasks. However, the long tasks, if they are present would not be treated appropriately and thus make the sys- tem imbalanced. One solution is to pro-actively prepare the potential replications before scheduling the tasks in order to redirect and balance the computation. To do that, we aim to improve the data locality metric by changing the replication scheme adaptively with regards to the popularity of the data file. Not only is the nature of the access rate taken into ac- count, but the replica placement is also carefully considered. Note that the access rate is defined as the number of accesses in a given unit of time. Subsequently, the data files in HDFS are replicated based on their own access potential as well as the overall status of the system.

In summary, the main contributions of this research are as follows.

- We designed an flexible replication management (FRM) 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 FRM 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.

II. RELATED WORKS

In the replication area, there are two main methods:

- the proactive approach
- the reactive approach

For the proactive approach, the Scarlett solution implements the probability as an observation and then calculates the replication scheme for each data file. The memory storage limitation is considered as a factor while distributing the replicas. For this proactive approach is introduced instead of using thresholds. 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 [6], 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.

For the reactive approach, the cost-effective dynamic replication management (CDRM) method is a cost-effective framework for replication in a cloud storage sys-tem. When the workload changes, CDRM calculate the popularity of the data file and determines 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 [8] is another reactive model of replication for HDFS. In this model, In this model, 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 exists in 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 considering the 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) [9] 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 chosen to 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 not clearly 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 same goal. While the replication is suitable for enhancing read operation, it suffers from a large storage overhead of up to 200%. Even with the rapid decline in the cost for the storage facility, this overhead has still become the major problem; this is because the volume and velocity of Big Datadramatically increase at a rate faster than the infrastructure, and are required not only for the storage resources but also for the computation and network utilization.
According to a number of studies on erasure code, it is clear to see that this branch of fault tolerance possesses an important role in maintaining the reliability for Big Data system. However, 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. To solve this issue, most of the approaches choose to utilize the degraded reading, which actually mitigates the unavoidable drawback. However, this point of design actually reduces the throughput and indirectly increases the computation time.

By examining the related works, we have come to the conclusion that although the research on replication and erasure code exists, not many researchers have thoroughly attempted to balance the data locality and the reliability within a reasonable cost of storage resource. Furthermore, since Hadoop is gradually a complex ecosystem, a faster and more adaptive replication mechanism must be developed.

III. METHODOLOGY

3.1 Motivation

In many parallel and distributed systems equipped with Map Reduce engine, the processing jobs usually comprise a series of consecutive phases, namely

- map
- shuffle

- re-duce.

In the beginning, map phase reads the input from disk and prepares the intermediate data for other phases. Due to this. Replicating uniformly or increasing the replication factor is not the key to accelerate the computation as well as reduce the slot contention and hotspot issue. Note that the slot contention happens when the number of concurrent tasks accessing the data file surpasses the number of replicas. Consequently, the tasks with no locally available data have to request for remote access or wait for the next available turns on the same data.

- Automatically it decreases the system performance. In the other hand
- The hot-spot issue, which is recognized as the attractive nodes to many tasks, makes the system imbalanced and wastes the idle computational capability.

These issues must be solved to fulfill the capability of big data system, especially.To minimize the effect of slot contention and hot-spot issue, many approaches choose to improve the data locality and conduct the load balancing as seen in the Related Works section. Thus, this reason motivates us to design a predictive approach (FRM) to truly enhance the data locality with regard to the system utilization and reliability. By proposing FRM, we expect that our study can be useful to any org-nizations or companies, which are interested in optimizing the performance within an affordable cost.

3.2 Approach analysis

As we discussed in Related Works section, 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 frequent data files may have more replicas 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% [18]. On the other hand, over 90% of data files might have a minimum access potential [18]. 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 replication. 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 thus results 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 erasure coding 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 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.

3.3 Domain analysis

As stated pro-actively improving the data locality based on the prediction method. For that reason, it is necessary to discuss the properties of input and output. Intuitively, the predictive computation mostly relies on the heartbeat (the periodic information generated by the computing nodes to indicate their operational status), which is collected by the HDFS logging component. This heartbeat is periodic, noise-free and consists of access rate as well as access type with regards to the time epoch. Basically, there are two kinds of **access types:**

- remote access
- local access.

Local access is dispatched from the tasks on the local machine, while remote access comes from the other servers in the same rack or from the servers located on the different racks, and is also known as the inter-communication access.

3.4 Cost analysis

The replication and the data locality are not in a linear relationship, only the storage space

consumption; the network bandwidth and the disk I/O are linearly related to the replication factor. In other words, the quantitative changes in the replication factors affect the storage cost and the communication bandwidth, which might subsequently degrade the currently running tasks. Unfortunately, this is an unavoidable issue for any solution attempting to improve the data locality and/or the reliability. Frankly, there is no ideal solution to achieve the best functionality without such a trade-off, especially when compared with the state of the art solution. Because of that, our goal is to design an architecture that improves the metric of interest (the data locality) but still maintains the same level of reliability at a reasonable price. For the reliability, as discussed above, the maintenance is held on two sets: the replication set and the erasure set. In fact, the reliability of the replication set is not a problem. First, due to the over replication, the redundancy of this set is kept at a high level as compared with the default scheme. Second, because the replication set occupies less than 10% of total data files, the space and transferring cost for this set is much less than that of the triple replication approach. For over 90% of data files which belong to the erasure set, it is proved in the that the reliability of this set is also maintained at a much cheaper cost than the default replication scheme. Given the fact that 80% of data accesses go toward less than 10% of stored bytes, our proposed architecture attempts to identify and replicate this small potential replication set to serve the major aforementioned percentage of data accesses. In this case, a maximum of 10% of data files get replicated with the adaptive factors, which consequently cuts down on the resources used and produces less contention on the currently running tasks as compared with other solutions.

IV. PROPOSED ARCHITECTURE

4.1 System description

The main function of the proposed architecture is to dynamically scale the replication factors as well as to conveniently schedule the placement of replicas based on the access probable of each data file. Additionally, to reduce the calculation time, the knowledge base and heuristic technique are implemented to detect the sameness in the access pattern between in-processing files and the anticipate ones. By definition, the access pattern is actually a set of eigenvectors represent the feature properties of processed data. Two files with similar access behaviors are deal with the same replication strategy. However, because these techniques are minor parts and popularly used in various systems, discussing them is not within the outlook of this paper. Constructed as a component of HDFS, the proposed approach (FRM) takes responsibility in advise the replication over the HDFS nodes. Intuitively, an overview of FRM is described in Figure 1. In this architecture, the regular physical servers as well as the cloud virtual machines can be used as and hint to as nodes. For this system configuration, FRM can be considered as a replication scheduler which can collaborate with any Map Reduce job scheduler. In fact, FRM helps the Fair scheduler and delay scheduling algorithm to affected the drawback of long tasks. Following is the description explaining the operation of ARM. To begin with, the system starts by periodically collecting the heartbeat. After that, This heartbeat is sent to the heuristic detector as the training data. This training data is related with the access patterns, which are derive from the predictor component and hoard at the knowledge base. If there is a match, the access potential is then retrieved from the arrangement and directly passed to the predictor component without any calculation. Otherwise, the training data is continuously sent instead as described in Figure 2. In that case, most of the calculation belongs to the hyper-parameter learning and training phases of the prediction. To solve this issue, the hyper generator is design to reduce the computational complication of the hyper-parameter learning phase. After that, the training phase can start to evaluation the access potential.

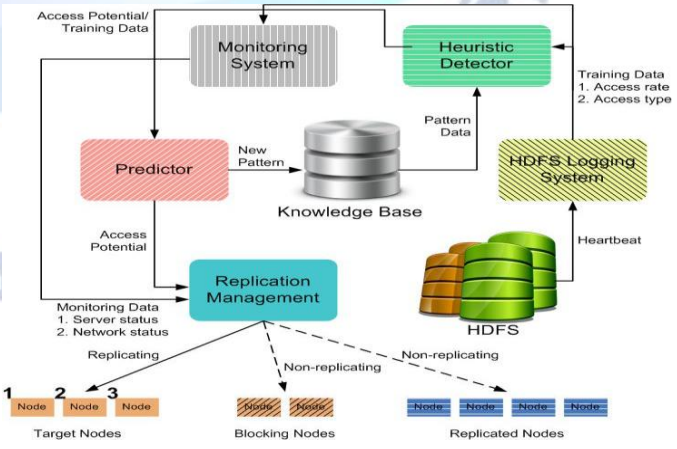

Fig. 1: Architecture of Flexible Replication Management (FRM) system.

Finally, the access potential of the target file is passed on to the replication management fundamental. In addition, a new pattern is also extracted and stored at the knowledge base for the next evaluation.

4.2 Replication management

The purpose of this section is to explain how the replication management chooses the correct position of the replica. Logically, by placing the possible replicas on low utilization nodes (low blocking rate nodes), the replication management helps to allocate the tasks to these idle nodes and maintains the computation. The blocking rate is calculated based on the information provided by the monitoring system. Based on Ganglia framework, the monitoring system is simple, robust and easy to configure for monitoring most of the required metrics. After plugging into the HDFS nodes, the monitoring system can collect statistics via Ganglia API. Because Ganglia receives most of the metrics provided by HDFS, there is almost no difference between this factor and the heartbeat. The only extra information is the system statistic, which consists of CPU utilization, RAM utilization, disk I/O and network high frequency. This design helps to merge the data sources for computational accesibility, especially for blocking rate calculation.

In order to complete the replication management, we assume that the replication management component collects all the elements and generates the replication approaches. From this acceptance, the access potential is used to scale the number of file copies. Then, the only issue remaining is related to selecting the placement of the replicas. As mentioned above, this duty is mainly based on the statistics fetch from the monitoring system to calculate the blocking rate and attach the replicas. Using the parallel and distributed system theory [21], only a few demanding factors can be considered to judge the blocking rate of the server. These factors include the network high frequency, the number of co-existing accesses and the capability of the server. Following is the mechanism to calculate the blocking rate.

$$BR(Si)= (\lambda\ i\ \tau\ i\ )\ ^c\ i/c\ i\ ^![(\lambda\ i\ \tau\ i\ )\ ^k/k!]^{-1}$$

where $_i$ is the moderate mapping time of the aforementioned tasks in S_i. Thereafter, by calculating the blocking rate, it is easy for the replication management component to select a location to allocate the replicas. As described in Figure 1, only the computing node satisfying two conditions is chosen as the destination. The first condition is low-blocking rate and the second one is to not store the desired replicas in advance.

V. Prediction Model

5.1 Back ground

The objective of prediction (forecasting) in the proposed approach (FRM) is to assume 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 used to perform on the access rate to obtain 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 should be engaged to benefit the localization.

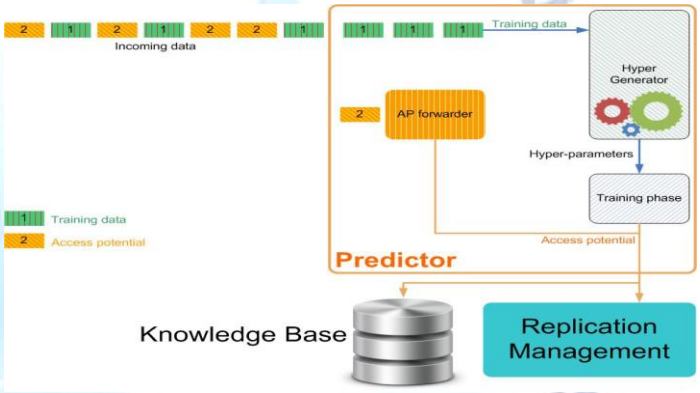

Fig. 2: Working mechanism of Predictor component

$$Wi =\eta r\ j^\ (i)/\psi\ j\ a\ i.$$

where W_i is the weighted factor corresponding to the access potential of the file i, $= _{max}= _{min}$ is the ratio of the largest to the smallest transmission rate, a_i is the total number of accesses to the file i, $r_j^{(i)}$ is the number of remote accesses coming from the connection j to the file i, and $_j$ is the transmission rate of that connection. 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.

Algorithm 1: Hyper-parameter learning phase

Data: Access array. This is the latest history of access

Result:rate of each data file with regards to time step.

```
Initialize value for (0) = [l0; f 0]; !; RMSE;
/* Fast Fourier Transform of input data
*/
y^= nufft1d1(y);
for k=1 to sizeof(y^) do
/* step_size is equivalent to in
the Equation (22) and (23)
step _size=decay _function(k);
j=random(1,sizeof (y^));
/* partial derivative of FrMLL w.r.t
l
r l = partial l(y^ ; ! ; l(k 1); (k 1));
/* partial derivative of FrMLL w.r.t
f
r f = partial _ f (y^ ; ! ; l(k 1)
/* update hyper-parameters
l(k) = l(k 1)+step _size* r l;
f(k) = f(k 1)+step _size* f ;
Compute r(k) ( (k)); r
rMLL
Compute RMSE(k) = RMSE(FrMLL);
if (RMSE (k) RMSE) then
break();
end
end
return ( ) = [l( ); ( )
f ];
```

Before proceeding with the prediction of the access potential, the GPR model requires the determination of the hyper-parameters as a prerequisite. By definition, hyper- parameters, which can be found in the covariance function, are the free parameters making the adaptations to the prediction process if the dataset at runtime. Finding hyper-parameters is one of the most expensive steps in constructing the aforementioned prediction model. However, not much progress has been made in terms of performance improvement, especially in dealing with a large dataset.

VI. PERFORMANCE EVALUATION

6.1 Experiments

Two experiments are used to evaluate the performance of the proposed approach (ARM). The first experiment is conducted on the Facebook cluster traces, namely the Statistical Workload Injector for Map Reduce (SWIM) sampling the historical Map Reduce cluster traces, the SWIM provides an efficient method to measure the effectiveness of the solution, which is intended to improve the HDFS replication on the realistic dataset. The data access patterns in these sets of traces are adjusted to follow the Zipf-like distribution, i.e., over 90% of data accesses focus on less than 10% of data files as mentioned in the Cost Analysis. Basically, the TeraSort stress test builds a sample key structure by selecting the subsets from the input before submitting the job and pushing this key structure into HDFS.

TABLE 1: System Configuration

	Configuration
Computing Nodes	01 Name Node, 16 Data Nodes
Platform	64bit
CPU	Intel®Core™ Xeon E5520 2.3GHz 4 cores
Storage	4x1TB for Name Node 4x1TB for each Data Node
Memory	16GB for Name Node 12GB for each Data Node
Network	Gigabit Ethernet Controller
OS	Cent OS 6.5 (final) Kernel: 2.6.32-431.el6.x86_64
Software	Apache Hadoop 0.20-append

In this experiment, the Tera Sort is performed on 1TB input data. Additionally, the Hadoop version used in the experiments is also modified to accept the dynamic replication factor as well as the flexible placement decision which is made by the FRM system. Finally, the configuration of the experimental cluster used for the experiments can be found in Table 1

6.2 Implementation

Since there is no way to directly trigger the operations on files in HDFS, an indirect heartbeat analysis is chosen as the alternative technique to collect the read/write operations. To do that, a log parsing script is implemented in Java by using the socket listener to manipulate the log file. Basically, the needed information consists of access time, access IP, target file, file operation and user authority who issues the operation. For the comparison purpose, the de-fault replication mechanism, ERMS and OPTIMIS are implemented to compare with the proposed ARM. All these approaches are implemented along with the Fair scheduler. Note that the delay scheduling algorithm is not engaged to avoid confusing the experiment result. Separately, the delay

scheduling is subsequently compared with ARM via the test on various durations of task.

6.3 Metrics

ARM is measured at two levels: the prediction level and the system level. At the prediction level, the metrics of interest are the completion time and the accuracy of prediction. Meanwhile, at the system level, the data redundancy and the availability (measured by the data locality) are the metrics of interest. In addition, some relevant factors such as network traffic, execution time and utilization of knowledge base are also considered. For the data locality evaluation, the metric is calculated in percentage as follows.

$$M_{dl} = \frac{Access_{local}}{Access_{total}}$$

where M_{dl} is the data locality metric of the target file estimated by the fraction of the local access $Access_{local}$ and the total access $Access_{total}$. In the other hand, the data redundancy metric is evaluated via the mean quantity of replicas and the shape of replication factor distributions.

VII. CONCLUSION

In order 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. For further development, some parts of the source code developed to test our idea would be made available under the terms of the GNU general public license (GPL).

VIII. LITERATURE SURVEY

Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. The core storage primitive: Cross-object redundancy for efficient data repair & access in erasure coded storage. Scarlett: Coping with skewed content popularity in mapreduce clusters. Enabling proactive data management in virtualized hadoop, clusters based on predicted data activity patterns. Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster

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/10.1145/1966445.196647

[5] G. Kousiouris, G. Vafiadis, and T. Varvarigou, "Enabling proac-tive 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 Inter-national Conference on, Sept 2010, pp. 188–196.