Forward Error Correction (FEC) Code in the Distributed Storage System

[1]P. Kiruba Ponselvan, [2]K. Sudha
[1]P.G Scholar, Computer Science Engineering, Vandayar Engineering College, India.
[2]Assistant Professor, Computer Science Engineering, Vandayar Engineering College, India.

***Abstract*: To adjust execution and capacity effectiveness, present-day grouped record frameworks regularly first store information with replication, Forward Error Correction (FEC) is an Erasure coding method, it is a code for information insurance in which information is broken into parts, extended and encoded with repetitive information pieces and put away over an arrangement of various areas or capacity media. We contend that the generally utilized irregular replication does not consider eradication coding in its outline, in this manner raising both execution and accessibility issues in the resulting encoding task. We propose encoding-mindful replication, which painstakingly puts the imitations in order to (i) wipe out cross-rack downloads of information squares amid the encoding task, (ii) save accessibility without information movement after the encoding activity, and (iii) look after the load adjusting crosswise over reproductions as in arbitrary replication before the encoding activity. We direct broad HDFS-based testbed tests and discrete-occasion reproductions and exhibit the execution increases of encoding-mindful replication over arbitrary replication.**

Index Terms: Erasure coding, HDFS, I/O execution, hub disappointment, replication, space effectiveness.

I. INTRODUCTION

THE advancement of enormous information has made a wonder in application and arrangement improvement to remove, process what's more, store helpful data as it develops to manage new difficulties. Here, Apache Hadoop is one of the most famous parallel systems. In addition to the fact that it is utilized to accomplish high accessibility, Apache Hadoop is additionally composed to distinguish and handle the disappointments and also keep up the information consistency. Joining the advancement of Apache Hadoop, the Hadoop Distributed File System (HDFS) has been acquainted with giving the unwavering quality and high-throughput access for information-driven applications. Step by step, HDFS has turned into an appropriate stockpiling system for parallel and appropriated processing, particularly for Map Reduce motor, which was initially created by Google to adapt to the ordering issues on enormous information. To enhance the dependability, HDFS is at first prepared with a component that consistently imitates three duplicates of each datum record. This procedure is to keep up the necessities of adaptation to internal failure.

Sensibly, keeping no less than three duplicates makes the information more solid and more vigorous while enduring the disappointments. Be that as it may, this default replication technique still remains a basic downside concerning the execution angle. Instinctively, the reason for developing Apache Hadoop was to accomplish better execution in information control also, handling [1]. Along these lines, this reason ought to be painstakingly learned at each part. In the execution viewpoint, in light of the notable research of postponement booking [2], if the errand is put nearer to the required information source, the framework can accomplish speedier calculation what's more, better accessibility. The metric estimates the separation between the undertaking and the comparing information source can be alluded to as the information territory metric.

The principle purpose behind the change is twofold. To start with, the system overhead can be lessened on runtime because of the accessibility of the nearby information, thus no between correspondence is expected to exchange the required information from the remote hubs. Second, it is evident that the calculation can begin instantly on the input information which is locally accessible, thus no additional task scheduling exertion is expended. Thus, it is important to state that enhancing the information area would enormously upgrade the framework execution as far as accessibility what's more, computation time.

The objective of deletion coding is to empower information that ends up tainted sooner or later in the plate stockpiling procedure to be reproduced by utilizing data about the information that is put away somewhere else in the exhibit. Eradication codes are frequently utilized rather than customary RAID in light of their capacity to diminish the time and overhead required to reproduce information. The disadvantage of eradication coding is that it very well may be more CPU-concentrated, and that can convert into expanded dormancy. Deletion coding can be valuable with substantial amounts of information and any applications or frameworks that need to endure disappointments, for example, plate cluster frameworks, information networks, conveyed capacity applications, question stores and chronicled stockpiling. One basic current utilize case for eradication coding is protest based distributed storage. Deletion coding makes a numerical capacity to depict an arrangement of numbers so they can be checked for precision and recouped on the off chance that one is lost. Alluded to as polynomial introduction or oversampling, this is the key idea driving eradication codes. In scientific terms, the insurance offered by deletion coding can be spoken to in straightforward shape by the accompanying condition: $n = k + m$. The variable "k" is the first measure of information or images. The variable "m" remains for the additional or repetitive images that are added to give security from disappointments. The variable "n" is the aggregate number of images made after the eradication coding process. For example, in a 10 of 16 setup, or EC 10/16, six additional images (m) would be added to the 10 base images (k). The 16 information sections (n) would be spread crosswise over 16 drives, hubs or geographic areas. The first document could be remade from 10 checked sections. Deletion codes, which are otherwise called forward mistake revision (FEC) codes, were created over 50 years back. Distinctive composes have risen since that time. In one of the soonest and most normal composes, Reed-Solomon, the information can be recreated utilizing any mix of "k" images, or bits of information, regardless of whether "m" images are lost or inaccessible. For instance, in EC 10/16, six drives, hubs or geographic areas could be lost or inaccessible, and the first record would in any case be recoverable.

Despite the fact that there are a few examinations on this topic, not very many proactive arrangements are suggested that thoroughly think about the idea of the activity workload. Because of the way that the workload in Apache Hadoop comprises of short and long assignments together, these undertakings ought to be taken care of reasonably to quicken the calculation. Commonly, the Fair scheduler and postpone planning calculation [2] give the ideal information territory when the framework is loaded up with head-of-line occupations and short undertakings. Be that as it may, the long assignments, on the off chance that they are available would not be dealt with fittingly and in this manner make the framework imbalanced. One arrangement is to star effectively set up the potential replications previously planning the undertakings so as to divert and adjust the calculation. To do that, we plan to enhance the information area metric by changing the replication plot adaptively with respect to the ubiquity of the information document. Not exclusively is the idea of the entrance rate taken into account, yet the reproduction position is likewise precisely considered. Note that the entrance rate is characterized by the quantity of gets to in a given unit of time. Subsequently, the information documents in HDFS are recreated in light of their own entrance potential and in addition the general status of the framework.

By definition, the get to potential is perceived as how as often as possible the particular document may be perused in whenever age. For instance, say a document has an entrance capability of 32 inside the time of 5 seconds: this implies the document may be gotten to 32 times in the following 5 seconds. Furthermore, the foreseen results and get to designs are reserved in the information base with a specific end goal to immediately coordinate and rapidly fire the appropriate activity without having to re-compute a comparative info. Sporadically, each information document can be effectively imitated by an alternate yet proper technique. Further, keeping in mind the end goal to keep up the blame resistance for less often got to information records; an open source deletion code [3] is altered and connected to ensure the framework from the impacts of disappointments. At last, by actualizing this structure, the assignment execution time and capacity cost can be enhanced profiting the profitability of enormous information frameworks. In a rundown, the primary commitments of this exploration are as takes after. We composed a versatile replication administration (ARM) framework to give high accessibility to the information in HDFS by means of upgrading the information territory metric. Subsequently, the exceedingly neighborhood accessible information moves forward the execution of the Hadoop framework. It is worth taking note of that the eradication code is connected to keep up the unwavering quality. We proposed an intricacy decrease technique for the forecast system in both hyper-parameters learning what's more, preparing stages. This proposed strategy altogether builds the execution regarding the response rate for the replication system while still keeping the exactness of the expectation. We executed ARM in HDFS and completed an assessment keeping in mind the end goal to for all intents and purposes confirm the adequacy of the proposed technique as contrasted and the condition of the craftsmanship strategy.

To streamline the transmission, every server won't process the whole unit of coded information specifically; however, fabricate a pipeline by partitioning them into stripes, each with a settled estimate.

II. LITERATURE SURVEY

In the replication zone, there are two principle strategies: the proactive approach and the responsive one. For the proactive approach, the Scarlett arrangement [4] executes the likelihood as a perception and after that ascertains the replication plot for every datum document. The capacity spending plan restriction is additionally considered as a factor while appropriating the imitations. In spite of the fact that this arrangement takes after a proactive approach as opposed to utilizing limits, the entrance rate of the information record and additionally the reasonable situation for imitations isn't talked about altogether. Moreover in OPTIMIS [5], an intriguing answer for foreseeing the information document status has been proposed. In this approach, the information document is arranged and occupied with the restricted replication situations in view of the algorithmic expectation of the interest for information record usage. Be that as it may, the Fourier arrangement examination algorithm [6], which is normally utilized in the field of 'flag preparing', is decided for expectation without a convincing evidence of the viability. As an outcome, this wrong decision may result in poor expectation.

For the receptive approach, the financially savvy dynamic replication administration (CDRM) strategy [7] is a practical structure for replication in a distributed storage framework. At the point when the workload changes, CDRM ascertains the prevalence of the information document and decides the area in the cloud condition. Nonetheless, this system takes after a receptive model. Subsequently, by utilizing edge esteems, CDRM can't adjust well to the fast development of expansive scale frameworks. So also, DARE [8] is another responsive model of replication for HDFS. In this model, the creators announce that the probabilistic examining and focused maturing calculations are utilized freely on every hub to pick a replication conspire for every datum document, and also to choose the appropriate area for every copy. Be that as it may, there are two issues in this approach. In the first place, the issue of long errands, which exists in a sensible framework, isn't thought about deliberately.

Actually, the presence of this issue makes the framework unsteady. Second, the arrangement of the replication is judged without considering the document get to example and framework limit of the goal hubs. Therefore, DARE probably won't give the normal viability on a few frameworks. The versatile replication administration framework (ERMS) [9] considers a functioning/reserve show for information stockpiling in the HDFS bunch by executing the intricate occasion handling technique to characterize the information composes. The benefit of ERMS as contrasted and CDRM and DARE is that it powerfully changes the limits for measurements in view of the status of the HDFS bunch framework. What's more, ERMS is furnished with the eradication code to decide and delete disagreeable reproductions in order to spare the capacity. All things considered, despite the fact that CDRM, DARE and ERMS are produced in various courses, every one of them experience similar issues and constraints. Solidly, these arrangements endeavor to group and execute different imitating situations for each kind of information records by removing and preparing the out of date data.

Hence, these methodologies can't produce an ideal replication procedure for parallel frameworks. The detail of this claim is that when a few activities are taken care of the 'hot' information records, because of high inertness and deferral, these documents may not be 'hot' any longer when the activities are locked in. As a result, the recreating choice can't mirror the patterns of the information use. Also, in the ERMS approach, the deletion code setup isn't obviously determined. Therefore, the capacity unwavering quality of this approach is as yet not checked. Dialogs on eradication code are intriguing. Usually, it is acknowledged that the execution, as well as the unwavering quality is the required part of HDFS. To satisfy this prerequisite, the replication and the eradication code are two sorts of adaptation to internal failure methods attempting to get a similar objective. While the replication is reasonable for upgrading perused activity, it experiences an expansive stockpiling overhead of up to 200 percent [10].

Indeed, even with the quick decrease in the cost for the storeroom, this overhead has still turned into the real issue; this is on account of the volume and speed of Big Data significantly increment at a rate speedier than the framework, and are required for the capacity assets as well as for the calculation and system use [11]. Thus, numerous enterprises including Facebook, Microsoft and Google think about the eradication coding approach as an elective strategy for sparing the storehouse storage room while keeping a similar level of unwavering quality. What's more, the eradication coding approach has a long history of improvement in shared frameworks to guarantee the ideal adaptation to internal failure with an ease of capacity [12]. The most recent consequence of deletion coding is the applications to the Microsoft Azure Storage [13] and in addition the new form of the Google File System alongside a few modules of HDFS.

As of late, Facebook has begun utilizing an open source deletion code, in particular HDFS-RAID [14]. The HDFS-RAID utilizes the celebrated Reed-Solomon (RS) code, which is gotten from the greatest separation distinct (MDS) technique [15], to guarantee the unwavering quality. The setup for this technique is RS (10, 4), which remains for a gathering of 10 stripes and 4 equality hinders for every datum document. By utilizing this arrangement, HDFS-RAID can make due amid 4 square disappointments and makes just 40 percent overhead for the capacity. Regarding framework point of view, this technique can be viewed as hearty and capacity effective in contrast with numerous other replication approaches. Different techniques utilized in eradication coding are Pyramid codes [16] and HDFS-Xorbas [10], which take after the methodologies of the nearby reproduction code (LRC) [17] and the locally repairable codes (LRCs) [10], separately. Not at all like the MDS coding family, Pyramid codes (which are connected in Microsoft Azure Storage [13]) and HDFS-Xorbas offer better repairable highlights as far as system transmission capacity and plate I/O. In any case, the hindrance of these techniques is the higher extent of capacity overhead as contrasted and HDFS-RAID. We talk about one more deletion coding arrangement, the storageCore [3], which depends on HDFS-RAID. In this arrangement, the creators join the standard eradication code and RAID-4-like equality to build up the excess. By applying this mix, the arrangement offers a noteworthy change in repairability and read execution. Furthermore, the creators likewise furnish the entire source code with detail and reproduction as an open source venture.

This choice makes storageCore appropriate for prepared combination with different strategies in HDFS. Be that as it may, the burden of storageCore is an expansion away overhead of up to 20 percent in contrast with HDFS-RAID. As indicated by various examinations on deletion code, it is obvious to see that this branch of adaptation to non-critical failure has an imperative part in keeping up the unwavering quality for Big Data framework. Nonetheless, by containing just a single duplicate of every datum hinder, the eradication coding approaches need to reunify the information squares remotely. Notwithstanding when HDFS peruses the deletion coded hinders in parallel, the preparing time is as yet stretched by the inaccessible registering hub. To explain this issue, the greater part of the methodologies use the corrupted perusing, which really mitigates the unavoidable downside. Be that as it may, this purpose of configuration really decreases the throughput and by implication expands the calculation time. By looking at the related works, we have arrived at the conclusion that despite the fact that the exploration on replication and deletion code exists, relatively few analysts have completely endeavored to adjust the information territory and the unwavering quality inside a sensible cost of capacity asset. Besides, since Hadoop is step by step a mind boggling biological community, a quicker and more versatile replication component must be created.

III. EXISTING APPROACH

The objective of Mist is to give a general system to lessen the season of dispersing deletion coded information in the server farm. As appeared in Fig.3.1, Mist limits the number of collectors that contact the source straightforwardly and lets other two recipients reconstruct their coveted information from such beneficiaries.

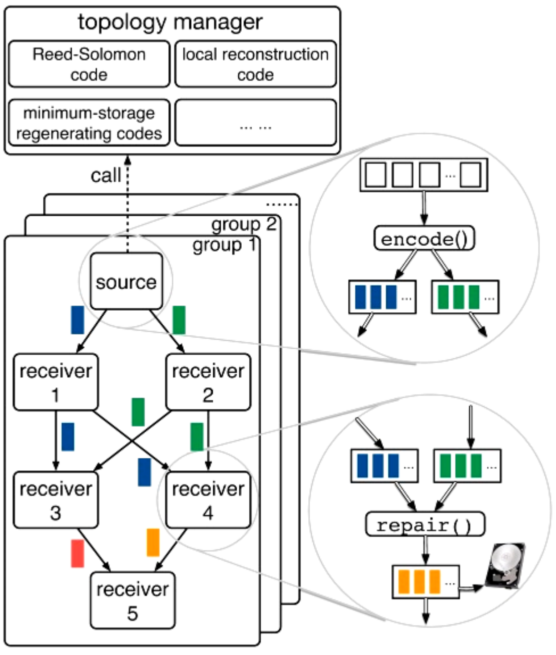

Fig. 3.1: The architecture of Mist

When we have in excess of 4 collectors, in any case, the 2 recipients that contact the source specifically will turn out to be new bottlenecks as every other beneficiary acquired information from them. In Mist, we fabricate staggered topologies for any number of collectors acquiring no more bottleneck anyplace in the topology that works with Reed-Solomon codes, as well as with all eradication codes when all is said in done. As it were, the collectors in the topology will be put into various levels with the end goal that all recipients in any level will get information from beneficiaries in another level nearer to the source. Other than a topology development for deletion codes, all in all, we additionally propose particular topology developments of neighborhood recreation codes [5] and least stockpiling recovering codes [8], two agent sorts of eradication codes intended for capacity frameworks in server farms, by exploiting their own particular properties.

Fig 3.1: demonstrates the design of Mist. In Mist, a gathering contains one source and different recipients. The topology administrator in Mist is in charge of producing the topology of the comparing eradication code. All beneficiaries will associate with different recipients or to the source as per the topology. The topology will likewise train the conduct of the source and every collector by actualizing two capacities: encode() to register eradication coded information at the source and repair() to process coded information from other coded information at the recipient. Along these lines, every recipient will register its coveted unit from the source or from different beneficiaries, store the relating units to neighborhood circles, and in the interim, send them to different collectors in the following level of the topology. At the point when there are numerous Mist bunches running in parallel, the topology supervisor does not keep running in a brought together way. Rather, the source in each gathering will run its own particular case of the topology chief to develop the pipelining topology of its own gathering, with the end goal that we can to countless Mist.

To streamline the transmission, every server won't process the whole unit of coded information specifically, however assemble a pipeline by partitioning them into stripes, each with a settled size. Once a stripe has been registered, the server will send it quickly to downstream beneficiaries. There might be an exchange off in picking the stripe measure, on the grounds that a bigger stripe will expand the computational postponement between two successive levels in the topology, while a little stripe will prompt a low throughput of encoding tasks [9]. In our approaching examinations, we will assess how different decisions of the stripe measure influence the scattering procedure.

IV. PROPOSED SYSTEM

Our plan handicaps the HDFS replication and includes the encoding and unraveling capacities executed by the DataNode. Extra sections are composed when they are produced and they are spared in particular DataNodes. DataTransferProtocol is utilized to send and get information from the DataNodes. A case of HDFS+ organization appears in figure 4.1

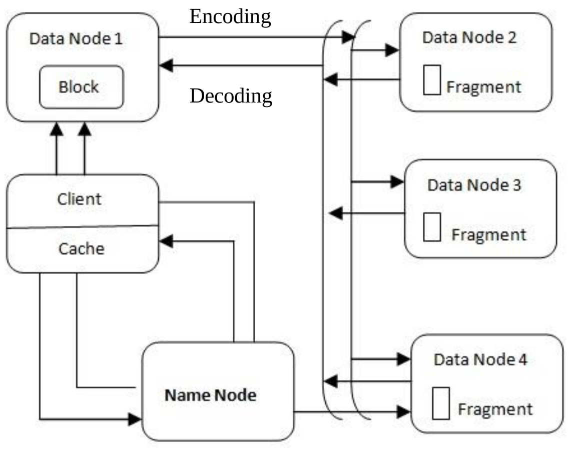

Figure 4.1 Proposed System Block Diagram

The compose activity in HDFS+ is as per the following: First, when the Client needs to compose a document, it sends compose demand to the NameNode. Second, the NameNode gives to the Client the rundown of accessible DataNodes to spare the information. Third, the Client picks the closest DataNode, and composes the squares into it. This DataNode turns into the essential DataNode. Fourth, the essential DataNode performs deletion encoding utilizing designed (n, k) values that gap the square into m sections and encode them into n pieces. The sections made are saved money on various DataNodes in the Hadoop group. Fifth, the affirmation of fruitful compose is sent to the NameNode.The Read task of HDFS+ is as per the following, First, when the Client needs to peruse a document, it sends a read demand to the NameNode. Second, the NameNode gives to the Client the rundown of DataNodes that it can recover information from. Third, the Client picks the most ideal DataNode to peruse from, and check on the off chance that it is an essential DataNode or not. Fourth, if the hub chose is the essential DataNode, at that point the Client recover information. In the event that the hub chose isn't the essential DataNode then it set that DataNode to be the essential DataNode. Fifth, the essential DataNode asks for sections from alternate DataNodes with parts for the asked for information. At the point when adequate sections have been acquired, the essential DataNode translates the information and supply it to the read application request. So that we could see that the proposed plot is superior to the first one as far as space and I/O execution.

V. IMPLEMENTATION

We actualize EAR as an augmentation to Facebook's HDFS [13]. The source code of our EAR execution is accessible at http://ansrlab.cse.cuhk.edu.hk/programming/ear.

5.1 Outline of Erasure Coding in HDFS

The first HDFS design [33] involves a solitary NameNode and different DataNodes. The NameNode stores the metadata (e.g., areas) for HDFS squares, while the DataNodes store HDFS squares. Facebook's HDFS actualizes deletion coding in a layer called HDFS-RAID [18], which deals with the eradication coded obstructs on HDFS. Over HDFS, HDFS-RAID acquaints a RaidNode with arranging the encoding task. The RaidNode likewise intermittently checks for any lost or adulterated squares and enacts recuperation for those squares. Facebook's HDFS bolsters between record encoding, with the end goal that the information squares of a stripe may have a place with various documents. HDFS-RAID executes encoding through MapReduce [11], which utilizes a solitary JobTracker to arrange numerous TaskTrackers on MapReduce preparing. To perform encoding, the RaidNode first acquires metadata from the NameNode and gatherings each k information hinders into stripes. It at that point presents a guide just MapReduce work (with no lessen undertaking) to the JobTracker, which doles out various guide assignments to keep running on various TaskTrackers, every one of which performs encoding for a subset of stripes. For each stripe, the capable TaskTracker issues peruse to k information obstructs in parallel from various DataNodes. When all k information squares are gotten, the TaskTracker figures the equality squares and thinks of them to HDFS. As of now, we utilize the Reed-Solomon codes [31] actualized by HDFS-RAID as our deletion coding plan. At last, the reproductions of the information squares are erased.

5.2 Consolidation

Figure 5.1 delineates how we adjust HDFS to incorporate EAR. Our alterations are insignificant, and they can be outlined in three angles. To begin with, we include the reproduction arrangement calculation of EAR into the NameNode. EAR restores the accompanying data: (I) which DataNodes the copies of an information square are to be put away, (ii) which information squares are encoded into a similar stripe in the consequent encoding activity, and (iii) which imitations of an information square are to be erased subsequent to encoding while at the same time guaranteeing rack-level adaptation to internal failure.

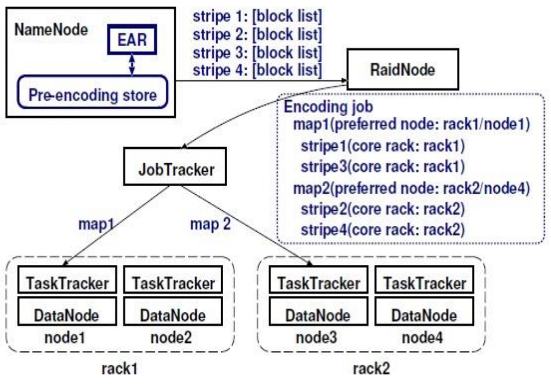

Figure 5.1 Consolidation of EAR into Facebook's HDFS.

We actualize a pre-encoding store in the NameNode to monitor each stripe and the rundown of information square identifiers that have a place with the stripe.We additionally alter how the RaidNode trains MapReduce to perform encoding of a stripe in the center rack. To accomplish this, we take note of that the MapReduce structure gives an interface to determine which favored hub to run a guide undertaking. In particular, the RaidNode first gets the rundown of information square identifiers for each stripe from the pre-encoding store. It at that point questions the NameNode for the imitation areas (as far as the hostnames of the DataNodes) of every datum square. With the returned areas, the RaidNode distinguishes the center rack of each stripe. At the point when the RaidNode introduces a MapReduce work for encoding, it guarantees that a guide undertaking encodes various stripes that have a typical center rack. This is finished by connecting the guide work with a favored hub, which we be one of the hubs in the normal center rack. For this situation, the JobTracker attempts to plan this guide assignment to keep running on the favored hub, or close-by hubs inside the center rack, in view of territory streamlining of MapReduce [11].

The above alterations still can't guarantee that the encoding activity is performed in the center rack, since all hubs in the center rack might not have enough assets to execute a guide errand for encoding and the JobTracker doles out the guide undertaking to a hub in another rack. This prompts cross-rack downloads for the encoding task. In this manner, we change the MapReduce structure to incorporate a Boolean banner in a MapReduce employment to separate on the off chance that it is an encoding work. On the off chance that the banner is valid, at that point the JobTracker just allots delineate to the hubs inside the center rack. Note that this alteration does not influence other non-encoding occupations.

VI. CONCLUSION

With a specific end goal to enhance the accessibility of HDFS by upgrading the information area, our commitment center around the following focuses. To start with, we outline the replication administration framework which is genuinely versatile to the normal for the information get to design. The approach not just master effectively plays out the replication in a prescient way, yet in addition, keep up the dependability by applying the deletion coding approach. Second, we propose an unpredictability decrease strategy to explain the execution issue of the expectation method. Truth be told, this intricacy decrease strategy essentially quickens the forecast procedure of the get to potential estimation. At long last, we actualize our strategy on a genuine group and check the viability of the proposed approach. With a thorough examination of the attributes of the document activities in HDFS, our uniqueness is to make a versatile answer for propelling the Hadoop framework. To encourage advancement, a few sections of the source code created to test our thought would be made accessible under the terms of the GNU general open permit (GPL).

VII. REFERENCES

[1] Dinh-Mao Bui, Shujaat Hussain, Eui-Nam Huh, and Sungyoung Lee. '**Adaptive Replication Management in HDFS Based on Supervised Learning**', IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 28, NO. 6, JUNE 2016.

[2] Fredrick RomanusIshengoma, **"HDFS+: Erasure Coding Based Hadoop Distributed File System,"** INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 2, ISSUE 9, SEPTEMBER 2013

[3] Runhui Li, Yuchong Hu, and Patrick P. C. Lee, **Enabling Efficient and Reliable Transition from Replication to Erasure Coding for Clustered File Systems**.

[4] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "**XORing Elephants: Novel Erasure Codes for Big Data**," in Proc. VLDB Endowment, 2013.

[5] K. Rashmi, N. Shah, and P. Kumar, "**Optimal Exact-Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction**," IEEE Trans. on Inform. Theory, vol. 57, no. 8, 2011.

[6] Maheswaran Sathiamoorthy, MegasthenisAsteris, DimitricPapailiopoulos, Alexandros G. Dimakis, RamkumarVadali, Scott Chen, **―XORing Elephants: Novel Erasure Codes for Big Data‖**, Proceeding PVLDB'13 Proceedings of the 39th international conference on Very Large Data Bases Pages 325-336, 2013.

[7] M. Spivak, S. K. Veerapaneni, and L. Greengard, "**The fast generalized Gauss transform," SIAM J. Sci. Comput**., vol. 32, no. 5, pp. 3092–3107, Oct. 2010.

[8] A. Datta and F. Oggier, "**Redundantly grouped cross-object coding for repairable storage**," in Proc. Asia-Pacific Workshop Syst., 2012, p. 2.

[9] 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.

[10] 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

[11] 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.

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