OPEN ACCESS INTERNATIONAL JOURNAL OF SCIENCE & ENGINEERING

# LEVERAGING FILE REPLICATION IN DATA-INTENSIVE CLUSTERS WITH ENERGY ADAPTABILITY

**Miss. Trupti Farande[1] ,Prof. Vandana Navale[2]**

*PG Students Department of Computer Engineering, Dhole Patil College of Engineering, Wagholi, Near Eon IT park., Pune-412207, Maharashtra, India[1]*

*Assistant Professor, Department of Computer Engineering, Dhole Patil College* of Engineering, Wagholi, Near Eon IT park., Pune-412207, Maharashtra, India[2]

*farandetrupti@gmail.com[1]*

**Abstract: File replication is a common strategy to improve data reliability and availability in large clusters. Reliability for each file under server failures based on the relationship between file reliability and replication factor when servers have a certain probability to fail. In the existing system more number of replica creates it require more energy consumption, time and storage space and sometime system fails and cannot send immediate response to the user request. In the proposed system increasing number of file replica according to user request or user priority and vice versa. Propose strategies in reducing file read latency, replication time and power consumption in large cluster. If multiple user send request for one file then system create multiple replica according to priority and get immediate response to the user.**

**Keywords**: Data-intensive clusters, file replication, replica placement, energy-efficient

## I INTRODUCTION

In this system some file create and store three replicas for each file in randomly selected servers across different racks. However, they neglect the file heterogeneity and server heterogeneity, which can be leveraged to further enhance data availability and file system efficiency. As files have heterogeneous popularities, a rigid number of three replicas may not provide immediate response to an excessive number of requests, So we propose a dynamic transmission rate adjustment strategy to prevent potential incast congestion when replicating a file to a server, a network aware data node selection strategy to reduce file read latency, and a load-aware replica maintenance strategy to quickly create file replicas under replica node failures. Random selection of replica destinations requires keeping all servers active to ensure data availability, which however wastes power consumption. The random selection of replica destinations does not consider destination bandwidth and request handling capacity, network congestions may occur due to capacity limitation of some links and server may become overloaded by data requests

## II LITERATURE SURVEY

In this Paper, The planned three genuine key swaps over protocols for parallel network file system (pNFS). Our procedures present three attractive compensations over the obtainable Kerberos based pNFS procedure. Primary, the metadata server implementing our procedures has much subordinate workload than that of the Kerberos based move toward. Subsequent, two our procedures make available frontward confidentiality: one is incompletely frontward protected [with admiration to manifold assemblies within an occasion era], at the same time as the additional is completely onward protected [with admiration to an assembly). Next, we have intended a procedure which not only make available onward confidentiality, other than is too escrowing gratis[1].

Author presents, Hadoop uses HDFS as the fundamental storage backend though Hadoop was originally planned to work with other FS too. Other types of FS are also supported by 0THadoop i.e. KFS, S3 etc. Hadoop with

HDFS file system have few challenges, one of them is Lustre as backend for Hadoop. Given the disadvantages of HDFS, Lustre looks much promising storage backend for Hadoop when compared with HDFS.[2]

The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks. By distributing storage and computation across many servers, the resource can grow with demand while remaining economical at every size. We describe the architecture of HDFS and report on experience using HDFS to manage 25 petabytes of enterprise data at Yahoo!.[3]

As new science teams adopt computational science as part of their discovery process, it becomes ever more important that parallel file systems cater to less “ideal” access patterns. In this work we have pursued a strategy of enabling a greater degree of latency hiding and reducing number of messages and I/O operations as a way of improving the performance of a parallel file system under a load with many small and independent I/O operations. We have described specific techniques that implement this strategy with roots in a variety of fields, including message passing libraries, databases, local file systems, and parallel file systems. [4]

In this paper, it is described that the small files processing strategy, the IPFP algorithm can reduce memory cost greatly and improve the efficiency of data access, thus avoids memory overflow and reduces I/O overhead. Meanwhile, the IPFP algorithm is migrated to the MapReduce environment, which can complete frequent itemsets mining efficiently and thus enhance the overall performance of FP-Growth algorithm. The experimental results show that IPFP algorithm can make a breakthrough where PFP algorithm has its defects in handling massive small files datasets, and has a good speedup and a higher mining efficiency.[5]

In this paper, we presented a new object-based storage model, ASG; introduced its architecture and primitives; and described a couple of use cases based on this model. As the use cases clearly show, the ASG storage model is flexible and can act as a starting point for building complex storage applications. Features supported by the ASG storage model make it possible to support requirements of common data models.[6]

## III SYSTEM ARCHITECTURE

In this paper, proposed EAFR to reduce file read latency, power consumption and replication completion latency. EAFR adaptively increases the number of replicas for hot files to alleviate intensive file request loads, and thus reduce the file read latency, and also decreases the number of replicas for cold files without compromising their read efficiency.

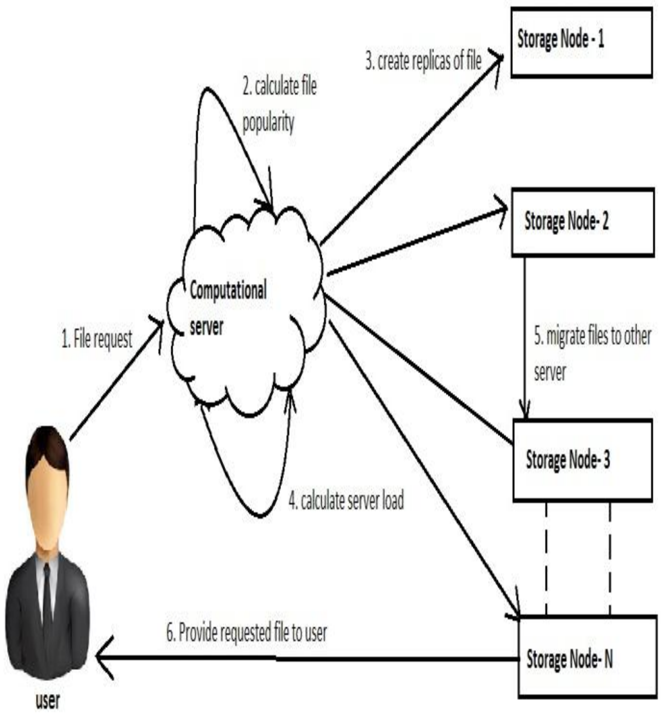
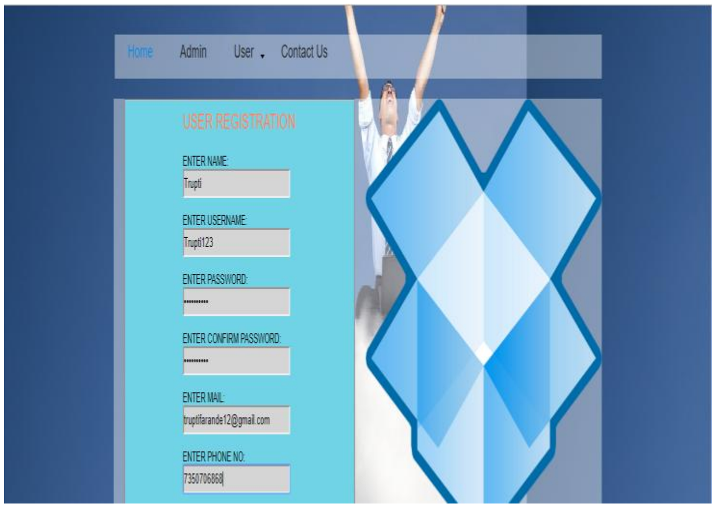


**Figure 1: System architecture**

## IV RESULTS



USER REGISTRATION

ENTER NAME: Trupti

ENTER USERNAME: Trupti123

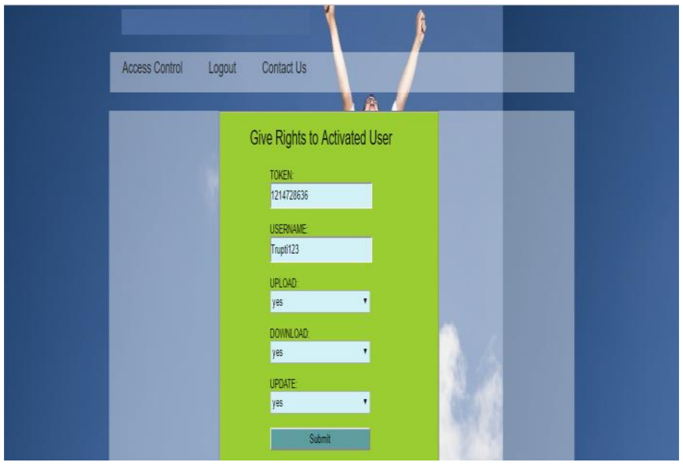
ENTER PASSWORD: ••••••••••

ENTER CONFIRM PASSWORD: ••••••••••

ENTER MAIL: truptifarande12@gmail.com

ENTER PHONE NO: 7350706868

**Figure. 2: Registration Page**



Give Rights to Activated User

TOKEN: 1214728636

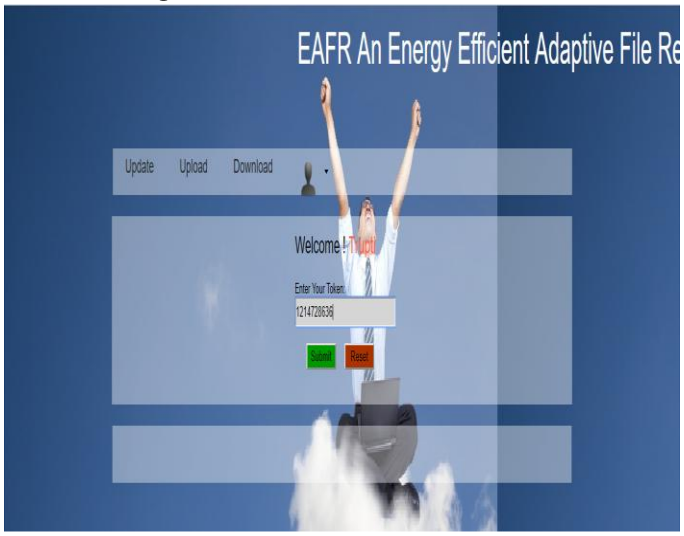
USERNAME: Trupti123

UPLOAD: yes

DOWNLOAD: yes

UPDATE: yes

Figure. 3: User Activation



Welcome ! Trupti

Enter Your Token: 1214728636

Figure. 4: Token Validation Page

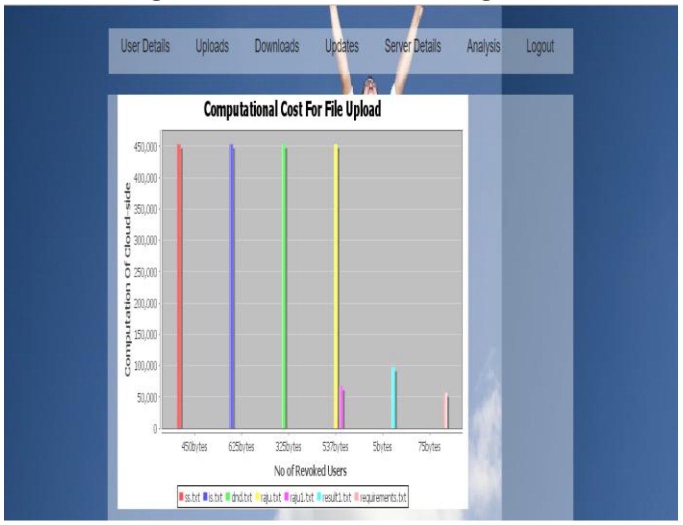


Figure. 5: Time Analysis

## V CONCLUSION AND FUTURE WORK

In this Paper the popularity of data-intensive clusters places demands for file systems such as short file read latency and low power consumption. In this system a dynamic transmission rate adjustment strategy to prevent potential incast congestion when replicating a file to a server, a network aware data node selection strategy to reduce file read latency, and a load-aware replica maintenance strategy to quickly create file replicas under replica node failures. In future system can be extend by adding server failure tolerance also to increase system file security. The data partitioning algorithms is used for better performance and security.

## REFERENCES

1) M. Rengasamy , "Energetic Key Exchange Protocol Authentication for Similar Network File Systems" October 2015
2) Sagar S. Lad P,NaveenKumarP, " Comparison study on Hadoop's HDFS with Lustre File System " November 2015
3) Konstantin Shvachko, Hairong Kuang , " The Hadoop Distributed File System " 2010 IEEE
4) Philip Carns, Sam Lang, Robert Ross, " Small-File Access in Parallel File Systems "
5) Sankalp Mitra , Suchit Bande, " Efficient FP Growth using Hadoop - (Improved Parallel FP-Growth) " July 2014
6) engiz Karakoyunlu, " Toward a Unified Object Storage Foundation for Scalable Storage Systems "
7) Z. Cheng, et al., “ERMS: An elastic replication management system for HDFS,” in Proc. CLUSTER Workshops, 2012, pp. 32–40.
8) Q. Chen, J. Yao, and Z. Xiao, “Libra: Lightweight data skew mitigation in MapReduce,” IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 9, pp. 2520–2533, Sep. 2014.
9) Verma, G. Dasgupta, T. Nayak, P. De, and R. Kothari, “Server workload analysis” in Proc. Conf. USENIX Annu. Tech. Conf., 2009, pp. 28–28.
10) J. Luo, L. Rao, and X. Liu, “Temporal load balancing with service delay guarantees for data center energy cost optimization,” IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 3, pp. 775–784, Mar. 2014.
11) K. Shvachko, K. Hairong, S. Radia, and R. Chansler, “The hadoop distributed file system,” in Proc. IEEE 26th Symp. Mass Storage Syst. Technol., 2010, pp. 1–10.