ISSN: 0975-766X
CODEN: IJPTFI
Available Online through
www.ijptonline.com
Research Article

RECEIVING FILES FROM SYSTEM SERVER USING DATA PERFECTING TECHNIQUE ON CLOUD

[1,]S.Vijay, [2]Mrs.J.Jackulin Reeja
Master of Computer Application, Sathyabama University, Chennai-600119.
Email: Jaivijay920@gmail.com

Received on 21-02-2016 *Accepted on 19-03-2016*

Abstract

Cloud fundamentally handles highly voluminous data and every user of cloud can store and then access such data to the tune of gigabytes. Cloud applications with capability to offer very quick access to data have been quite frequently emerging. Therefore, it becomes very important to offer great performance along with efficiency and reliability. One among the primary issues found in design of the distributed file systems happens to be reducing latency while remote files are being accessed, and solutions include file pre-fetching technologies and cache replacement. Here, we are making use of pre-fetching method on repository servers pertaining to distributed file systems toward cloud computing. The technique Piggybacking happens to be one bi-directional information transformation method on the grid layer. In the suggested system, we propose the said method toward minimizing the work load on the cloud server. Prefetching file data of client nodes gets piggybacked on to real customer I/O requisitions, after which they get forwarded to concerned storage server.

Prefetched data may be pushed on to related customer machine from storage server. Also, in this method, prefetching forecasts the future batch access functions toward directing in advance which data to be fetched on storage servers. Prefetching method is advantageous to distributed file systems pertaining to cloud scenario for achieving improved I/O operation.

The mechanism initially examines disk I/O courses for predicting ensuing I/O access which the depository servers may fetch information early, and eventually convey the prefetched information to the appropriate file systems toward potential future usages.

Keywords: Cloud computing, Piggybacking, prefetching, distributed file system.

1. Introduction

Internet access has been becoming readily available to everybody during the recent years. Cloud computing happens to be one internet-oriented technology. Software and hardware are being used by cloud computing as resources for providing service across the internet. In the present period of time, cloud computing has been made use of for enabling end users to produce and utilize software without having to worry about the implementation of technical data at anytime from anywhere [9] [8]. Across the network, resources are being utilized and post computation, they get delivered in the form of services in the cloud computing. Here, the technology of Cloud computing has been implanted in three services that happen to be easily usable, only a click away, and pay when you make use of the amenity. Cloud storage happens to be a service related to developers for storing and accessing information in the cloud. Normally, any file system that has been deployed in one given distributed computing scenario is known as distributed file system and it always is remained to be some backend depository system for providing I/O services to different types of information intensive exercises on the cloud computing situations [1], [2], [7], [6]. Cloud is finding demand on the fields such as web hosting and web services. In the recent time, Internet amenity file systems have been developed extensively toward management of data present in huge scale services on cloud computing principles. Certain private organizations may also have cloud systems for insuring their works get completed in the fastest possible manner. A major of the applications being executed on cloud need number files toward processing and computation. These systems involve heavy expenses for the purpose of processing all such file requests. The latent period required for accessing and transmitting files outruns the time for computation to a substantial level. Piggybacking happens to be on bi-directional data transfer method in the grid layer (OSI pattern). It makes maximum use of the dispatched data settings from the receiver to sender, in addition, confirming that data setting dispatched by sender has been successfully received (via ACK acknowledgement). This in practice means that in place of dispatching one acknowledgement in one single frame, it gets piggy-backed on to the information setting. In the suggested system we adopt, the method is gathering cloud file and distributes pre-fetched information to the appropriate client machines in proactive manner. The information regarding customer nodes gets piggybacked to actual client I/O requisitions and further gets forwarded to related storage server properly. The user file system can dispatch suitable I/O requisitions toward storage server for the purpose of reading the appropriate information and reduced file processes through batched I/O requisitions via prefetching.

2. Related Work

J.Mogul and T.V.Padmanabhan introduced the concept that the enduring victory of the popular World Wide Web relies on quick time of response. People make use of the Web for accessing data from remote websites but they hate having to wait for a long time toward their results. Latency of web document retrieval depends on many elements like propagation time, bandwidth of the network, and speed of client and server computers. Even though there are many proposals toward latency reduction, it is practically tough for pushing it to the spot where it turns insignificant. Regarding prefetching processes that are employed in real life distributed file systems, Ceph file structure [32] is one distributed file system which offers great reliability, scalability, and performance. Ceph system maximizes the division between Meta data and data management via replacement of allotment tables by some pseudo-arbitrary information distribution operation (CRUSH) devised toward dynamic and heterogeneous clusters of non-dependable object depository devices (OSDs). These may leverage device intellect through distributing failure detection, information replication, and restoration of semi-independent OSDs running some specialized domestic object file structure. A dynamic shared metadata bunch offers extremely effective metadata management while seamlessly adapting with a vast range of common purpose and also scientific calculating file structure workloads. For managing data, cloud structures depend on distributed file systems (DFS). Examples for these are Hadoops HDFS [8] AND Googles GFS [9]. There are many strategies toward cloud information retrieval. Upon being given some query, the matching information are getting retrieved from DFS and forwarded to a group of functioning nodes toward parallel checking. This uses a simple strategy for query processing and is found to be suitable for any particular purpose pertaining to an individual organization. Contrary to this, in the open service cloud structure like EC2 of Amazon, various customers arrange their own products of software on the same cloud structure. Processing nodes get shared among different customers and therefore data management turns more complex. Hence, one more effective information access service becomes necessary in place of checking. In [4] Dr.G.Shoba and D.Pratipa suggested that as information gets complicated and also number of clients is on the rise, searching files needs to be permitted through usage of multiple keyword according to interest of the end users. Conventional searchable encoding structures permit clients for searching in encoded cloud information via keywords that assist just Boolean search meaning that irrespective of whether or not a keyword is present in the file, with no relevance of information files and queried keyword. The process of searching data in cloud by making use of single keyword graded search results to be too crude

output and privacy of data is opposed while employing ranking on server side on the basis of order-protection encryption (OPE). And this strategy assures top-n recapture of multi keyword across the encoded cloud information with great confidentiality and practical effectiveness with the help of vector space pattern and TRSE, in which majority of the works of computing can be performed on server while user can participate in ranking.

3. Proposed Work

3.1 Overview

Cloud computing happens to be one fast progressing technology in which resources like platform, applications, and storage devices are being shared across the internet. It is being used extensively by multiple clients in the medium and small business sector. In the system proposed by us, we introduce one piggyback method regarding prefetching information during the time a user requests the cloud for information. Here, prefetching method is being employed for reducing the cloud storage process and sending result to user quickly. One key issue found in design related to distributed file systems will be reducing latent period during remote file access, with solutions such as file prefetching and cache replacement technologies. We propose in our paper an innovative prefetching strategy wherein the primary concept is providing an improved I/O operation. Piggybacking procedure is being used for holding early search files of user from cloud storage along with verifying client input query every time a client requests for information with cloud server. Under the system we propose, an access has been produced for enabling the user to be in a position to access piggybacking file.

3.2 Overall Architecture

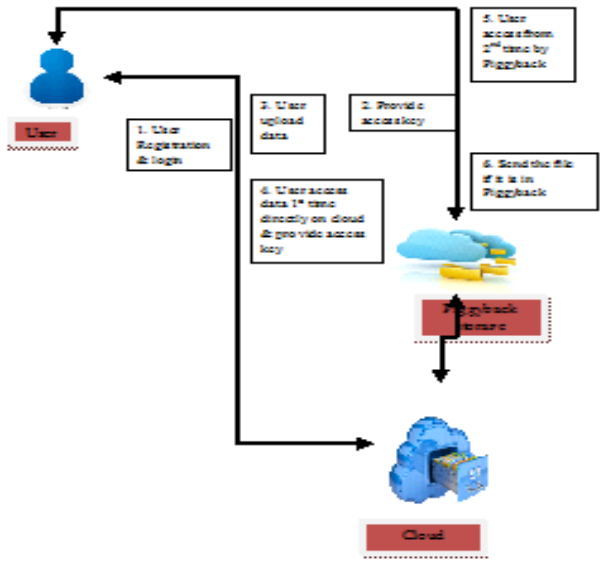

Figure-1: Data Prefetching process by Piggyback system.

3.3 Client Access Piggyback Depository

In the said system, the user has to first register for accessing files, for storing information on cloud storage, and foe getting prefetched information from piggyback depository. Piggyback repository is being run on behalf of cloud storage with regard to each user. Upon registering their details, users will receive verification code toward accessing piggyback repository.

In normal case, when the user is accessing cloud repository for the first time, piggyback procedure may not have started. But from the second attempt onward, it begins the procedure of verification of client input output requisition, and examines whether or not client input file has been stored in given storage space. When a file requested by user is recognized on piggyback repository, then it gets prefetched and also dispatched back to the client. By making use of this procedure, it is possible to reduce waiting time of user and minimize the cloud storage process.

3.4 Using Piggyback Method for Prefetching

In the prefetching approach that has been presented, the information gets prefetched via repository servers after examining disk I/O traces and then information is pushed proactively into the related customer file system toward catering to potential applications requisitions. For this purpose, it becomes necessary for the repository servers to understand data about user file arrangements and applications. Piggybacking procedure conveys relevant data from customer node to repository servers toward supplying modeling diskette I/O models and forwarding prefetched data. When a user sends some logical I/O requisition to repository server, customer file structure will piggyback data of client file structure and also the application.

It is possible via storage servers to register disk I/O incidents along with client data that plays a crucial role in categorizing patterns of access and establishing destination customer file structure toward prefetched data. Customer data is being piggybacked to repository servers such that repository servers are enabled to register disk I/O functions that accompany data about associated logical I/O incidents. Data regarding logical access consists of file indicator offset, node information, and the size requested. Data regarding related physical access includes ID stripe, ID of storage server, ID block, and size of request. One among the techniques mentioned aims to direct prediction of physical access through examination of disk I/O tracks. Two neighboring I/O procedures present in raw disk track may not be having any dependency.

3.5 The Piggybacking I/O

The I/O process belonging to random stream or sequential stream verifies whether working group size pertaining to the chosen systems is found to be small when compared with the entire storage size even though size displays substantial variation. The said working group algorithm maintains a working group (W(t, T)) period t as group of access destinations cited inside process time period gap (t-T, t). This algorithm compares working group size with a threshold for indicating whether some access model change has occurred.

4. Result and Discussion

In this section, we present the results of data prefetching experiments on the cloud storage based on the client server model. Here we apply prefetching process on the cloud computing technology due to reduce the cloud storage process time and user waiting time.

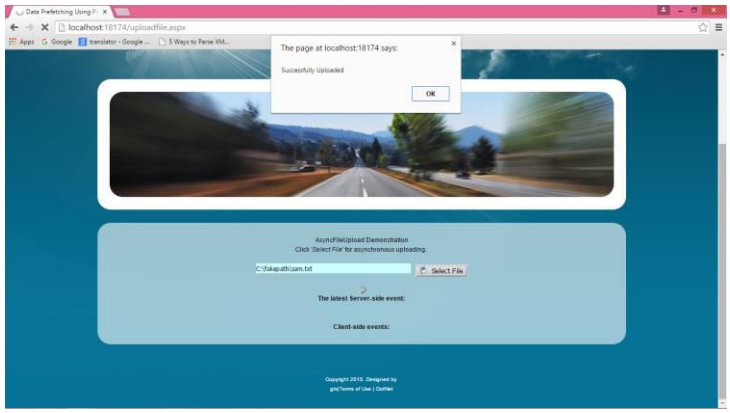

Figure-2: File uploading to cloud by piggyback Storage.

The above screen shows the file uploading process to cloud storge using piggyback techique. Before data uploading to stroage space user have to register and get secret key from piggyback system. This secret key is verified at file searching time.

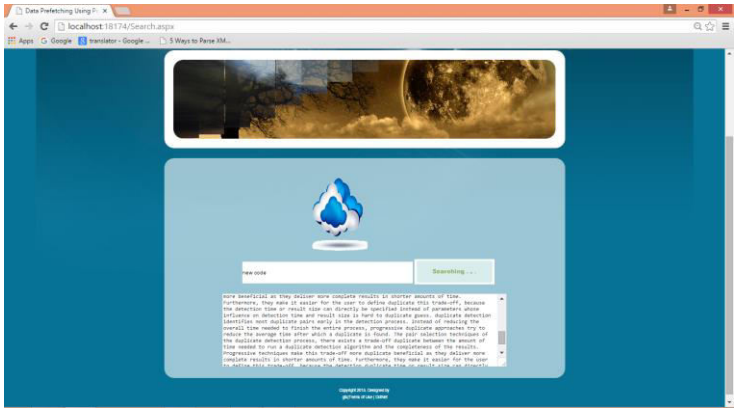

Figure-3: File pre-searchig in piggyback storage.

Once user uploaded the files in cloud storage. Users needs to access the file when they want. Such situation user first enter into piggyback storage with secret key and provide the file name to in the searching place. Once user enter the file name piggyback start the searching in piggyback storage before enter into the cloud storage.

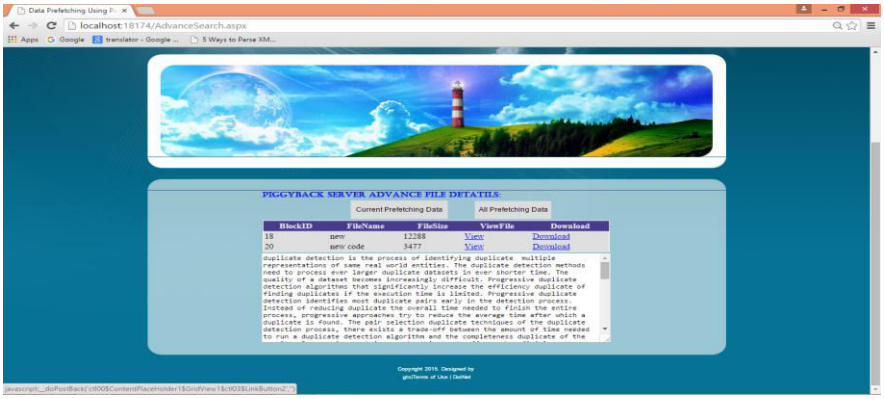

Figure-4: File Prefetching Result in piggyback storage.

The above figure explains the data displaying on piggyback storage. Once user enter the file name in piggyback storage, it starts the searching in that storage for file prefetching. If file is found in piggyback storage it will show the file details to user.

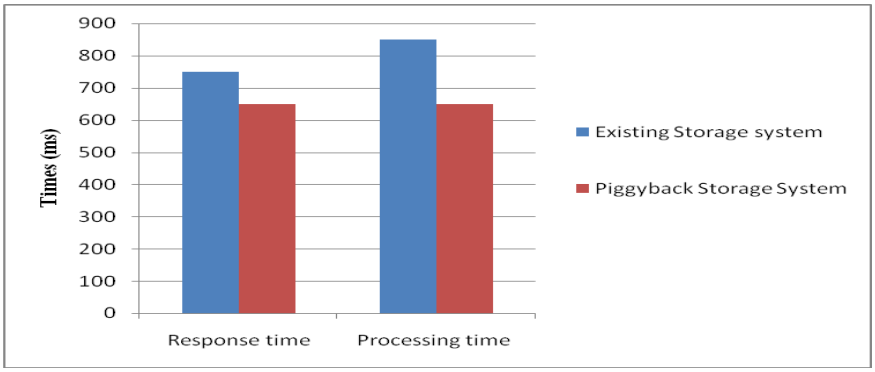

Figure-5: Process comparison of Existing and proposed system.

The above figure3 shows advantages of proposed system with existing system. Compare with existing system our piggyback system provides more efficiency to user for sending the data to user and minimize the process time.

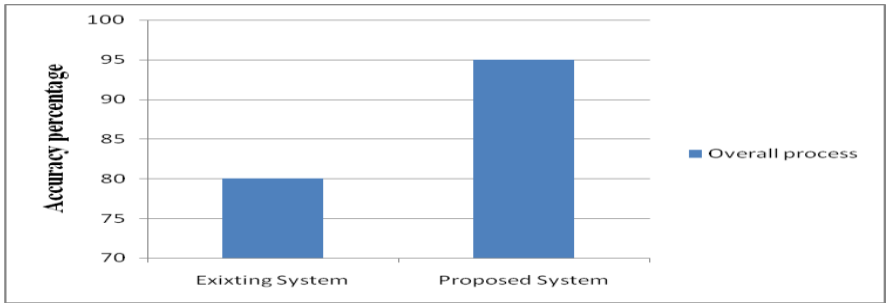

Figure-6: Overall process of Proposed and Existing System.

The above figure shows the experimental result of overall process comparison. The main aim of the proposed system provides the better pre fetching technique on cloud computing technology. Because cloud computing is now days accessed by many user for storage purpose. So avoiding the process time of cloud storage and response time to user here we propose piggyback method, this method helps to cloud storage and users for efficiently provide a better solution.

5. Conclusion

The client and server model in cloud computing is quickly becoming more extensive because of its quick application deployment running in server side and minimum cost, or so-called server related computing. In this proposed system we suggested piggyback method file prefetching process in cloud computing system. We propose the piggyback storage system with access key and based on the access key user can only access the cloud storage and piggyback storage system. We show the method feasibility by conducting experiments and implementing prototype.

The experimental results demonstrate that file prefetching waiting time in remote location is minimized and hit ratio is maximized in more cases than existing system.

6. References

1. D.Pratiba, Dr.G.Shobha and Vijaya Lakshmi.P.S "Efficient Data Retrieval From Cloud Storage Using Data Mining Technique" International Journal on Cybernetics & Informatics (IJCI) Vol. 4, No. 2, April 2015.
2. S.S and A. Basu, "Performance of eucalyptus and open stack clouds on future grid,"International Journal of Computer Applications, vol. 80,no.13,pp.31-37, 2013.
3. Hsiao-Ying Lin; Tzeng, W.-G.; , "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding," Parallel and Distributed Systems, IEEE Transactions on , vol.23, no.6, pp.995- 1003, June 2012.
4. Wang Cong, Wang Qian, Ren Kui, Cao Ning and Lou Wenjing , "Toward Secure and Dependable Storage Services in Cloud Computing," Services Computing, IEEE Transactions on , vol.5, no.2, pp.220-232, April-June 2012.
5. J. Stribling, Y. Sovran, I. Zhang and R. Morris et al. Flexible, widearea storage for distributed systems with WheelFS. In Proceedings of the 6th USENIX symposium on Networked systems design and implementation (NSDI'09), USENIX Association, pp. 43–58, 2009.
6. D.Nurmi, R.Wolski, C.Grzegorczyk, G.Obertelli,S.Soman, L.Youseff and D.Zagorodnov, "The eucalyptus open-source cloud- computing system," CCGRID 20009.9th IEEE/ACM International Symposium, 2009.

7. J. Kunkel and T. Ludwig, Performance Evaluation of the PVFS2 Architecture, In Proceedings of 15th EUROMICRO International Conference on Parallel, Distributed and Network-Based Processing, PDP '07, 2007

8. E. Shriver, C. Small, and K. A. Smith. Why does file system prefetching work? In Proceedings of the USENIX Annual Technical Conference (ATC '99), USENIX Association, 1999.

9. N. Nieuwejaar and D. Kotz. The galley parallel file system. Parallel Computing, 23(4-5):447–476, 1997.

Corresponding Author:

S.Vijay*,

Email: *Jaivijay920@gmail.com*