Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DATA REPLICATION USING MODIFIED D2RS IN CLOUD COMPUTING FOR PERFORMANCE IMPROVEMENT

[1]S.KIRUBAKARAN, [2]Dr.S. VALARMATHY AND [3]C.KAMALANATHAN
[1]Assistant Professor (Sr.Gr), [2]Professor& Head, [3]Assistant Professor (Sr.Gr)
Department of Electronics and Communication Engineering
Bannari Amman Institute of Technology, Sathyamangalam
E-mail : [1]kirubakaran0283@gmail.com

ABSTRACT

In cloud computing environments failures are normal rather exceptional. To advance the system availability, the frequently used data should get replicated to multiple locations to make the users to access from the nearby site. To decide a reasonable number and right location of replicas is a challenging task in cloud computing. Here, we recommend a Modified Dynamic Data Replication Strategy (MDDRS) to decide a reasonable number and right location of replicas and we compare both the modified dynamic data replication strategy and Dynamic Data Replication Strategy (DDRS). The DDRS has three different stages which are the identification of data file to replicate, number of replicas to be created and placing new replicas. We modify the popularity degree in the first stage of normal dynamic data replication strategy and the other two stages are similar to the normal dynamic data replication strategy. In addition we incorporate the synchronous and asynchronous updation of replica data file when we update the main data center.

Keywords: *Cloud Computing, Data Replication, Popularity Degree.*

1. INTRODUCTION

The success of contemporary technologies extremely depends on its usefulness of the world's norms, its ease of use by end users and most significantly its degree of information security and control. Cloud computing is a new and rising information technology that shifts the way IT architectural solutions are suggest by means of moving towards the theme of virtualization: of data storage, of local networks (infrastructure) as well as software [1-2]. It is a large-scale disseminated computing paradigm driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, highly available, and configurable and reconfigurable computing resources (e.g., networks, servers, storage, applications, data) can be hastily provisioned and released with least management effort in the data centers. Services are carried on demand to outdoor customers over high-speed Internet with the X as a service (XaaS)" computing architecture, which is broken down into three different parts: applications", platforms", and infrastructure".

From a sociological standpoint on the other hand, in the cloud, applications are reachable anywhere, any-time, and storage becomes infinite for all intents and purposes. And the users can admittance the powerful applications, platforms, and services delivered over Internet. Also, high availability, high fault tolerance and high efficiency access to cloud data centers where failures are normal rather than outstanding are significant issues, due to the vast data support. Data replication allows dipping user waiting time, speeding up data access and increasing data availability by providing the user with diverse replicas of the same service, all of them with a coherent state. Replication is a frequently used technique in the cloud, such as GFS (Google file system) [3], HDFS (Hadoop Distributed File System) [4]. However, cloud data centers have grown steeply in both size and number, and the dynamically scalable and totally virtualized resources are provided as a service over the Internet [5].

Replication is used to progress system availability (by directing traffic to a replica after a failure), avoid data loss (by recovering lost data from a replica), and improve performance (by spreading load across multiple replicas and by making low-latency access available to users around the world). However, there are diverse approaches to replication. Synchronous replication assures all copies are up to date, but potentially incurs high latency on updates. Moreover, availability may be impacted if synchronously

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

replicated updates cannot accomplish while some replicas are offline. Asynchronous replication excludes high write latency (in exacting, making it appropriate for wide area replication) but permits replicas to be stale. Moreover, data loss may take place if an update is lost due to breakdown before it can be replicated [9].

In order to attain the dynamic data replication, there are three vital problems that must be solved. 1) Which data should be replicated and when to replicate in the cloud systems to meet the users' necessities on waiting time reduction and data access speeding up are significant issues for further research, as the wrongly selected and too early replicated data will not reduce the waiting time or speed up data access. 2) How many suitable new replicas should be created in the cloud to meet a reasonable system availability requirement is another important issue to be thoroughly investigated. With the number of new replicas increasing, the system maintenance costs will considerably increase, and too many replicas may not enhance availability, but bring unnecessary spending instead. 3) Where the new replicas should be located to meet the system task successful execution rate and bandwidth consumption necessities is also an imperative issue to be explored in detail. By keeping all replicas active, the replicas may advance system task successful execution rate and bandwidth consumption if the replicas and requests are logically distributed. However, appropriate replica placement in ultra-large-scale, dynamically scalable and totally virtualized data centers is much more complicated [7].

In this paper we recommend a technique for synchronous and asynchronous based data replication in cloud computing. We initially check the system availability and thereafter we apply the modified D2RS (Dynamic Data Replication Strategy) algorithm. The dynamic data replication strategy consists of three different stages. The first stage is to identify which data file should be replicated and when to replicate in cloud computing to reduce the waiting time. We modify the popularity degree in the first stage of the D2RS algorithm [7]. The modification of the popularity degree is based on double exponential moving average function. The second stage of the D2RS algorithm is to find the required number of replicas and the third stage is to find where to place the new replica data files. In normal dynamic data replication strategy, the popularity degree is calculated between the start time and present time but in our modified dynamic data replication strategy, we calculated the access frequency based on time factor and users and the access frequency based on time factor is calculated using double exponential moving average function. We also incorporate the synchronous and asynchronous updation for the newly created replicas. In synchronous updation, the record which we update in the main datacenter will get update simultaneously to the replicas in the sub datacenters but in asynchronous updation, the record which we update in the main datacenter will get update to the replicas after a specified time interval using the asynchronous agent.

The rest of the paper is structured as follows: the second section shows the review of related works, the third section shows system architecture, the fourth section shows our proposed technique, the fifth section delineates the obtained result and the sixth section concludes our recommended technique.

2. REVIEW OF RELATED WORKS

A handful of researches are available in the literature for data replication model in cloud computing environment. Here, the review of recent works from these topics is presented. Storage systems are crucial building blocks for cloud computing infrastructures. Although high performance storage servers are the eventual solution for cloud storage, the execution of low-cost storage system remains an open issue. To address this problem, Julia Myint and Thinn Thu Naing [6] have implemented the efficient cloud storage system with inexpensive and commodity computer nodes that are organized into PC cluster based datacenter. Hadoop Distributed File System (HDFS) was an open source cloud based storage platform and designed to be deployed in low-cost hardware. PC Cluster based Cloud Storage System was implemented with HDFS by enhancing replication management scheme. Data objects were distributed and replicated in a cluster of commodity nodes located in the cloud. That system provided optimum replica number as well as weighting and balancing among the storage server nodes. The experimental results showed that storage was balanced depending on the available disk space, expected availability and failure probability of each node in PC cluster.

Failures are normal rather than exceptional in the cloud computing environments. To improve system availability, replicating the popular data to multiple suitable locations is an advisable choice, as users

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

can access the data from a nearby site. This is, however, not the case for replicas which must have a fixed number of copies on several locations. How to decide a reasonable number and right locations for replicas has become a challenge in the cloud computing. Da-Wei Sun *et al* [7] developed a dynamic data replication strategy was put forward with a brief survey of replication strategy suitable for distributed computing environments. It included: 1) analyzing and modeling the relationship between system availability and the number of replicas; 2) evaluating and identifying the popular data and triggering a replication operation when the popularity data passes a dynamic threshold; 3) calculating a suitable number of copies to meet a reasonable system byte effective rate requirement and placing replicas among data nodes in a balanced way; 4) designing the dynamic data replication algorithm in a cloud. Experimental results demonstrated the efficiency and effectiveness of the improved system brought by the proposed strategy in a cloud.

Mohamed-K Hussein and Mohamed-H Mousa [8] have proposed an adaptive replication strategy in the cloud environment. The strategy investigates the availability and efficient access of each file in the data center, and studied how to improve the reliability of the data files based on prediction of the user access to the blocks of each file. The proposed adaptive replication strategy redeployed dynamically large-scale different files replicas on different data nodes with minimal cost using heuristic search for each replication. The proposed adaptive strategy was based on a formal description of the problem. The strategy identified the files which were popular file for replication based on analyzing the recent history of the data access to the files using HLES time series. Once a replication factor based on the popularity of the files was less than a specific threshold, the replication signal was triggered. Hence, the adaptive strategy identifies the best replication location based on a heuristic search for the best replication factor of each file. Experimental evaluation demonstrated the efficiency of the proposed adaptive replication strategy in the cloud environment.

3. SYSTEM ARCHITECTURE

This section shows the system architecture of the cloud computing. The Fig.1 shows the sample cloud system topology.

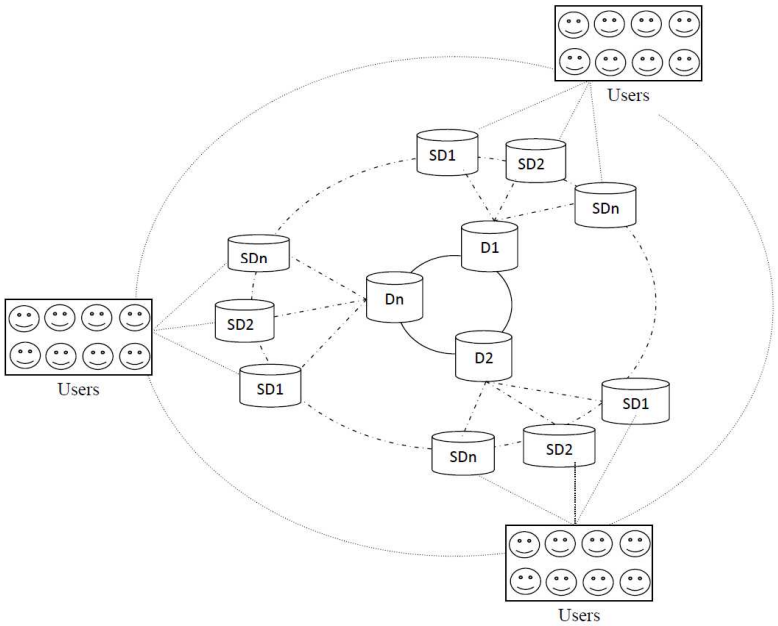


Fig.1 Sample Cloud System Topology

In the Fig.1, $D1, D2, ..., Dn$ are the main data centers and $SD1, SD2, ..., SDn$ are sub data centers of the main data centers $D1, D2, ..., Dn$. Here, each main data centers have n number of sub data centers. The sub data centers may or may not have another set of sub data centers. The users will use the least set of sub data centers. Each main data centers arte interlinked with each other and each set of sub data centers are also interlinked with each other to transfer the data from one data center to another. It is not sure that all the data in the data

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

center $D1$ would be in the data center $D2$ and also it is not sure that the entire data in the sub data center $SD1$ of the main data center $D1$ is present in the sub data center $SD2$ of the main data center $D1$. If the user who uses the data center $SD1$ requires a data which is not in $SD1$ but which is present in $SD2$, the data center $SD1$ request a replica of the data to $SD2$ or to its main data center. Another case is that if a data in the data center $SD1$ is used by more number of users and if a new user comes to use the same data, he needs to wait to use that data. So to avoid the waiting time, we need to create a replica of that data in the data center $SD1$.

The Fig.2 shows the sample cloud data center architecture. It contains users, scheduling manager, replica manager and data centers. The task given by the user is first send to the scheduling manager. The scheduling manager then gives the path to the particular data center to access the file. The path selection by the scheduling manager is based on the number of users uses a data center to avoid congestion.

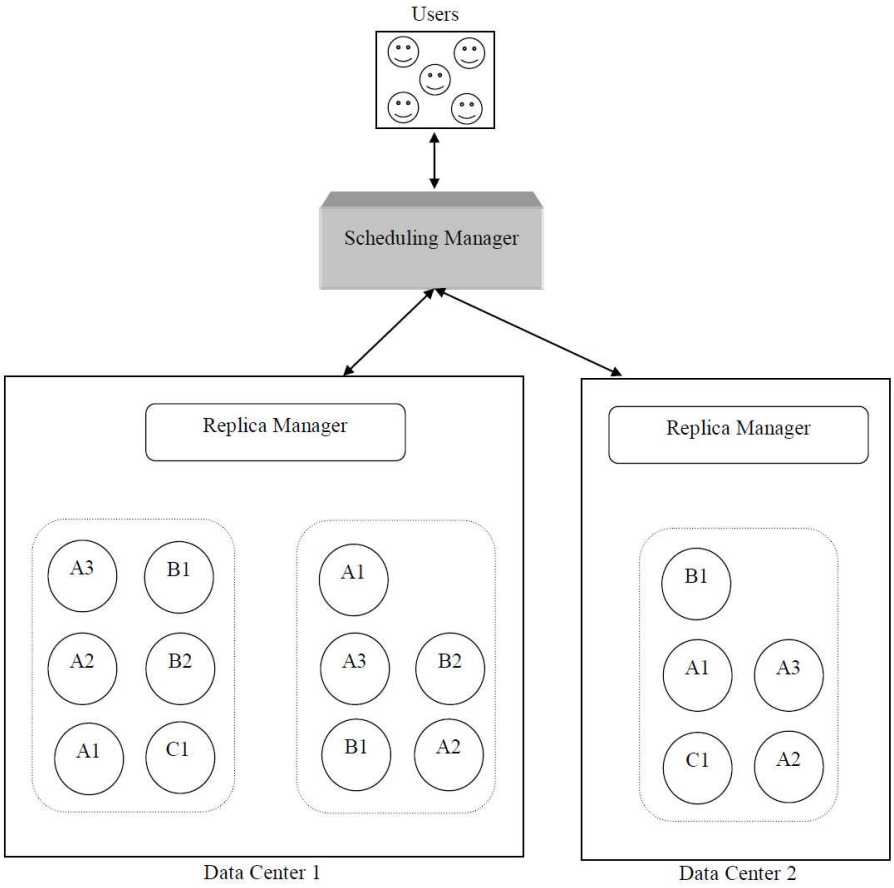


Fig.2 Sample Cloud Data Center Architecture

4. PROPOSED SYNCHRONOUS AND ASYNCHRONOUS BASED DATA REPLICATION

This section delineates our recommended technique of synchronous and asynchronous based data replication in cloud computing. The processes involved in our recommended technique are probability of file availability and unavailability, which file and when to replicate, total number of replication, where to place the replica and the synchronous and asynchronous update after replication. The Fig.3 shows the block diagram of our recommended technique.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

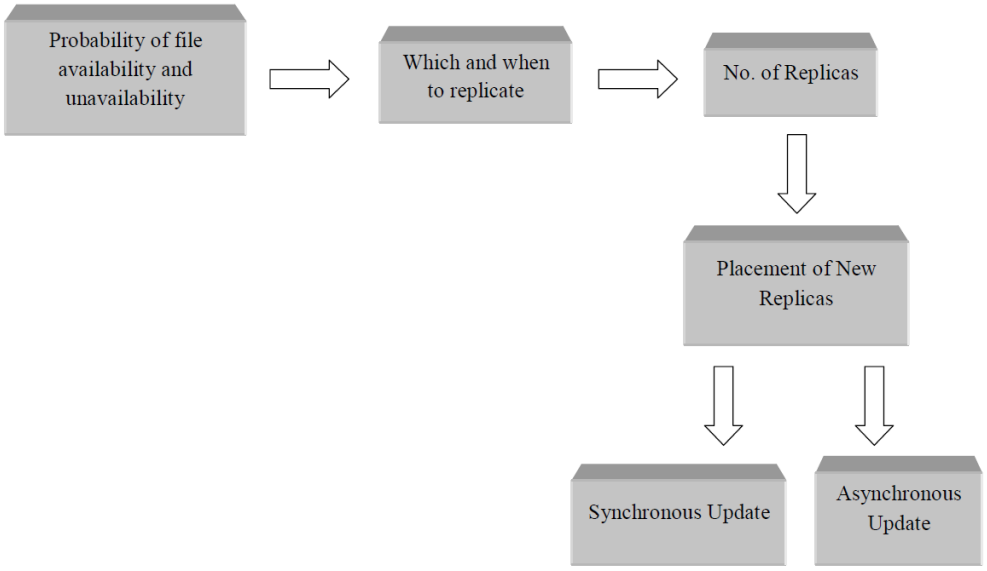


Fig.3 Block Diagram Of Our Proposed Technique

The availability of file and unavailability of the file is calculated to find the system byte effective rate [7]. Thereafter, we apply the modified dynamic data replication strategy (D2RS) algorithm. The normal dynamic data replication strategy [7] consist of three stages that are (i) which data file should be replicated and when to replicate in the cloud system, (ii) how many replicas to be created in the cloud system and (iii) where we have to place the new replicas. The modification we do is the calculation of the popularity degree which is the access frequency based on time factor that is used in the first stage of the normal dynamic data replication strategy. Except the modification, all the process is same as [7] and we incorporate the synchronous and asynchronous updation in our technique. In normal dynamic data replication strategy, the popularity degree is calculated considering the start time and the present time but in our modified dynamic data replication strategy, the popularity degree is calculated using access frequency based on time factor and access frequency based on users.

4.1. Popularity Degree

The popularity degree of a block is the access frequency based on the time factor and users. The popularity degree is calculated from the start time t_s to the present time t_p. In our recommended technique, the popularity degree is calculated using access frequency based on time factor and the access frequency based on users. The access frequency based on time factor is calculated using double exponential moving average function. The popularity degree is calculated as follows:

$$pd = F_1 + F_2 \quad (1)$$

where,

pd → Popularity Degree

F_1 → Access frequency based on time factor

F_2 → Access frequency based on users

The access frequency based on the time factor F_1 is calculated based on the double exponential moving average. It is shown by the equation below:

$$F_1 = 2[f(an_t)] - f(f(an_t)) \quad (2)$$

where,

$f(an_t)$ → Access frequency based on time interval

$f(f(an_t))$ → Access frequency based on the time interval of the time interval we split

The procedure to find the access frequency based on the time interval we split is as follows: Initially we split the time intervals from the present time t_p to start time t_s. A sample splitting of time intervals from the present time t_p to start time t_s is shown in the Fig.4. Thereafter, we calculate the access frequency within the time interval which we split and eventually, we sum all the access frequency we obtained for the respective time intervals. The access frequency based on the time interval we split is shown by an equation below:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$f(an_t)_{t\to t_s \text{ to } t_p} = \alpha an_{t_p} + (1-\alpha)an_{t_{p1}} + (1-\alpha)^2 an_{t_{p2}} + (1-\alpha)^3 an_{t_{p3}} + \ldots\ldots\ldots + (1-\alpha)^k an_{t_s} \quad (3)$$

Where,

an_{t_p} → Access frequency with respect to the time interval t_p

α → Constant

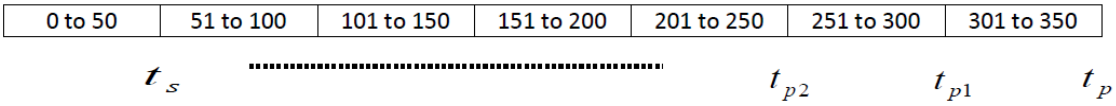


Fig.4 Sample Time Splitting

The Fig.4 shows the sample splitting of time between t_s and t_p. In this figure, t_p is the present time which takes the interval from 301 to 350 seconds and t_{p1} has the time interval from 251 to 300 seconds and t_{p2} has the time interval from 201 to 250 seconds and eventually, t_s is the start time that has the time interval from 0 to 50 seconds.

After calculating the access frequency based on time interval, we have to calculate the access frequency based on the time interval of the time interval we split i.e. we have to split the time interval of $t_p, t_{p1}, t_{p2}, \ldots\ldots\ldots, t_s$. The equation given below shows the formula to find the access frequency based on the time interval of the time interval we split.

$$f(f(an_t)) = f(an_{t_p}) + f(an_{t_{p1}}) + f(an_{t_{p2}}) + \ldots\ldots\ldots + f(an_{t_s})$$

Using the access frequency based on time interval $f(an_t)$ and the access frequency based on the time interval of the time interval $f(f(an_t))$, we can find the value of the access frequency based on time factor F_1.

The access frequency based on the user F_2 is calculated as follows: initially we need to find the access frequency of each user separately from the start time t_s to the present time t_p. Thereafter, the access frequency of each user is multiplied with the weight value of the respective users and eventually, we need to sum all the values. It is shown by the equation below:

$$F_2 = \sum_{\substack{i=1 \\ t\to t_s \text{ to } t_p}}^{N} an_{ui} * W_{ui} \quad (4)$$

an_{ui} → Access frequency of i[th] user

W_{ui} → Weight value of i[th] user

N → Total number of users

4.2. Synchronous and Asynchronous Update

This section explains the synchronous update and asynchronous update of our recommended technique. The Fig.5 shows a sample diagram for the synchronous and asynchronous update.

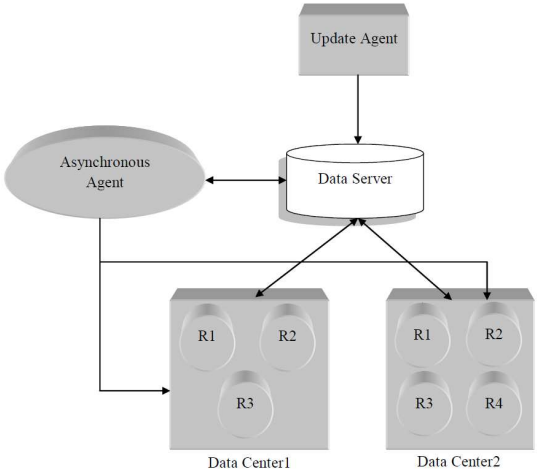


Fig.5 Sample Block Diagram For Synchronous And Asynchronous Update

4.2.1.Update Agent

This is the agent who updates i.e. adding or deleting any data in the data server. The data in the data server will not be same for every time we open a page. It would get change when the update agent updates the page in the data server. The update agent is the authorized agent to alter any information in the data server and except the update agent no one can do any alteration in the data server.

4.2.2.Data Server

The data server is the main data center that consists of many sub data centers. The main data center contains a collection of information and it shares that information through the cloud and the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

users can use that information from the cloud network. So any alteration of information in the main data server would change the same copy present in the sub data centers which provides the information to the users.

4.2.3. Data Center

The data centers are the sub data centers which are connected with the main data center and share the information from the main data center. The users use the data from the sub data center in the cloud network and the files which are replicated are present in the sub data center. The sub data centers contain the data which is in the main data center. If a file is demanded by more number of users, it would create a replica of that file to avoid congestion and waiting time. The replicas created in the sub data center should get change when we alter the source file of that replica in the main data center.

4.2.4. Asynchronous Agent

The asynchronous agent is an agent that updates the replica in the sub data center after a specified time interval. The alteration done in the main data center should get updated in the replica present in the sub data centers. Usually, when we update the information in the main data center, the appropriate replicas present in the sub data centers would also get updated automatically. But we cannot be sure that all the replicas present in the data centers would get updated. The usual simultaneous alteration of data in the sub data centers when the source file is altered in the main data center is called synchronous update. The asynchronous agent updates all the replicas in the sub data centers for every specified time interval. So each replica in the sub data centers would get updated in the specified time interval even if the replica was not updated using synchronous update.

5. RESULTS AND DISCUSSION

This section delineates the results we obtained for our recommended technique and compare the performance with the existing technique and also explains the comparison of synchronous and asynchronous updation.

5.1. Experimental Setup

Our recommended technique is applied using java that uses java development kit version 1.6 and the system we used has the following configuration: i5 processor, 4GB RAM. We used three different datasets for processing our recommended technique and the datasets are Financial, Medical and RDB. We compare our modified dynamic data replication strategy with the normal dynamic data replication strategy using those three datasets we used.

5.2. Experimental Process

This section shows the experimental process of our recommended technique. The Fig.5 shows a sample dataset in the data center with its contents.

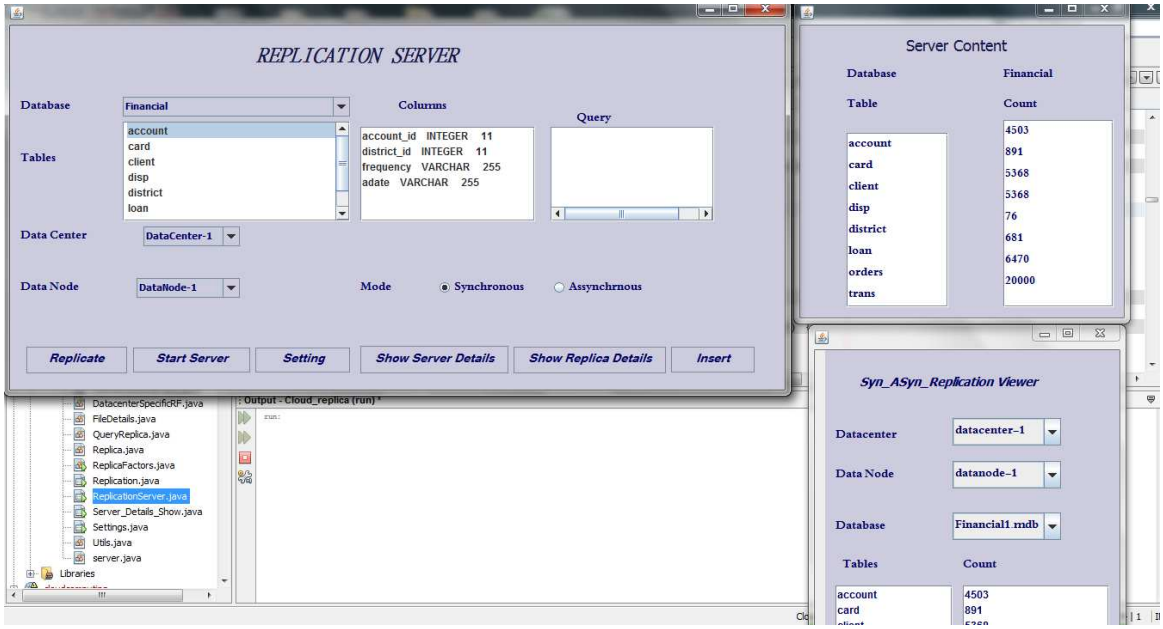


Fig.5 Sample Dataset In The Datacenter With Its Contents

Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In this figure, the 'replication server' window shows that the financial dataset contains the tables as account, card, client, etc. and the 'column' section shows the records in the account table. The 'server content' window in the Fig.5 shows the total number of records in each table present in the financial dataset.

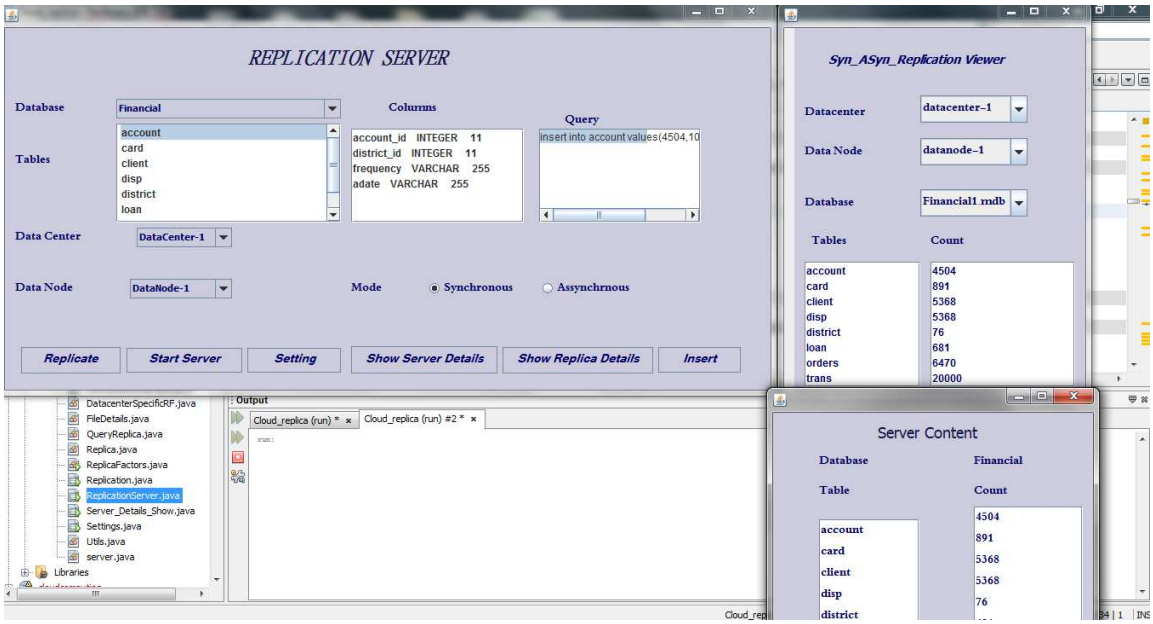


Fig.6 Synchronous Updation

The Fig.6 shows the screenshot for synchronous updation after adding a record in the account table of the financial dataset. In 'replication server' window the option for updation we selected is synchronous. So when we update a record in the server, it will get updated in the replica simultaneously. We can see it in the 'Syn_Asyn_Replication Viewer' and in the 'Server Content' window. The 'server content' window shows the details in the server and the 'Syn_Asyn_Replication Viewer' window shows the details in the replica. In both windows we can see the total number of records in the account table as 4504 but before updating the server it was 4503 that is shown in Fig.5.

Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

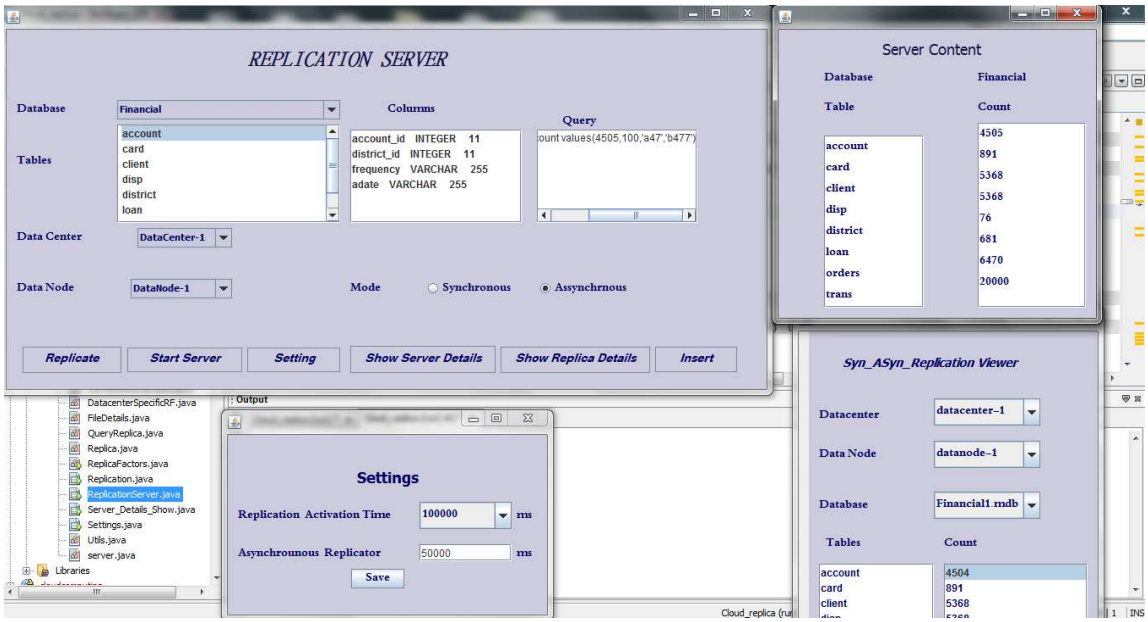


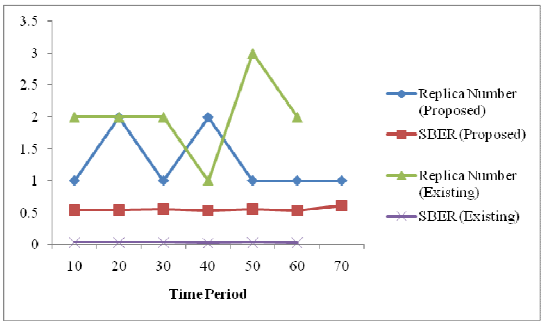
Fig.7 Asynchronous Updation

The Fig.7 shows the screenshot for asynchronous updation after adding a record in the account table of the financial dataset. In 'replication server' window the option for updation we selected is asynchronous. So when we update a record in the server, it will not get updated in the replica simultaneously. After adding a record, the 'server content' window shows the account table has 4505 records and the 'Syn_Asyn_Replication Viewer' window shows the account table has 4504 records only. It implies that the added record is not updated simultaneously in the replica. The Fig.8 shows the screenshot for the query given by the user.

Fig.8 Client Query

5.3. Performance Comparison

This section shows the performance of our technique compared with the normal dynamic data replication strategy based on the number of replica and system byte effective rate with respect to time interval. We check the performance with different values of adjustable parameter α [7]. The values of adjustable parameter are set based on different system performance. The better the requested system performance, the greater α can be selected. We also compare the synchronous and asynchronous updation based on the execution time.



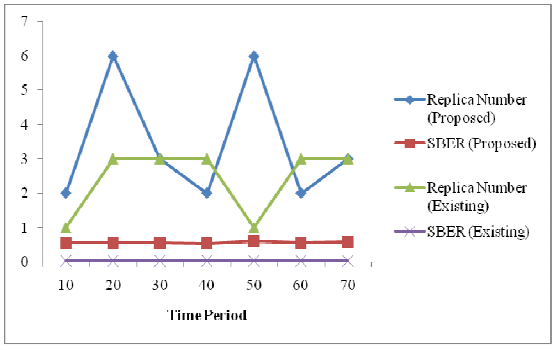
Graph 1 Performance Comparison when $\alpha = 0.2$

The Graph.1 shows the number of replicas and system byte effective rate in different time intervals for our proposed and existing technique when the adjustable parameter $\alpha = 0.2$. The values in the graph is taken by checking the replica numbers we obtained for our proposed technique and the existing technique at different time periods and the system byte effective rate is calculated after generating replicas. Here in this Graph.1, for the first time period the replica number we obtained for our proposed technique is one and the replicas we obtained for the existing technique is two and the system byte effective rate we obtained for our proposed technique is 0.545 and for the existing technique is 0.03645 for the same time period. The system byte effective rate is the number of bytes

Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2

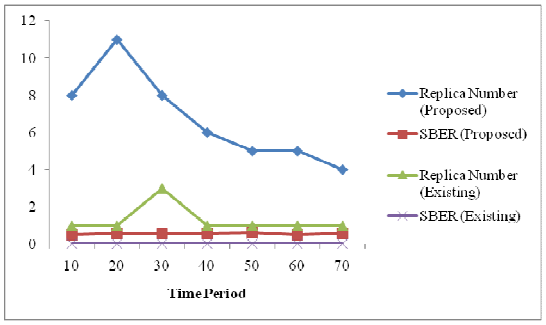
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

potentially available and the total number of bytes requested by all tasks. Here, the system byte effective rate comparison shows that our technique has better system availability rate for the varied time periods.



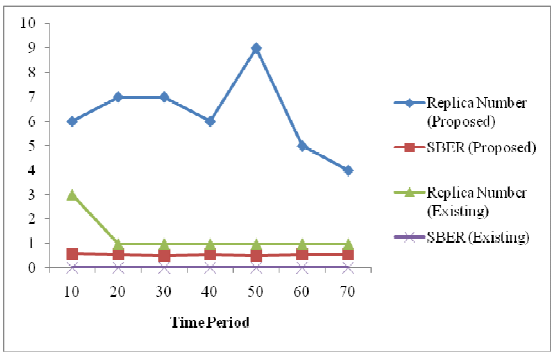
Graph 2 Performance Comparison When $\alpha = 0.4$

The Graph.2 shows the number of replicas and system byte effective rate in different time intervals for our proposed and existing technique when the adjustable parameter $\alpha = 0.4$. Here, at the third time interval we got the replica numbers as three for our proposed technique and existing technique and at that time interval, the system byte effective rate of our proposed technique is 0.5705 and for the existing technique is 0.03209. This shows that our proposed technique has better system availability ratio compared to the existing technique.



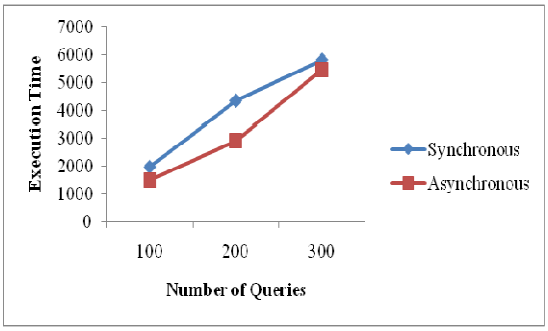
Graph 3 Performance Comparison When $\alpha = 0.6$

The Graph.3 shows the number of replicas and system byte effective rate in different time intervals for our proposed and existing technique when the adjustable parameter $\alpha = 0.6$. For each time interval, the number of replicas would get vary for our proposed technique and for the existing technique and the system byte effective rate for our proposed technique on each time interval is better compared to the existing technique after generating replicas.



Graph 4 Performance Comparison When $\alpha = 0.8$

The Graph.4 shows the performance comparison between MDDRS technique and the DDRS when $\alpha = 0.8$. It shows that the system byte effective rate of our proposed technique is high compared to the existing technique. The Graph 5 shows the performance comparison between the synchronous and asynchronous updation based on the execution time. When the queries given to execute is hundred, the time taken for execution is 1984 ms based on synchronous updation and it is 1500 ms based on asynchronous updation. When we give 200 queries, the time taken for execution is 4353ms in synchronous updation and 2924ms in asynchronous updation. When the query given is 300, the execution time is 5827ms for synchronous updation and 5481ms for asynchronous updation.



Graph 5 Execution Time For Synchronous And Asynchronous Updation

In synchronous updation, the record we change in the main server will get updated simultaneously to the client servers but in asynchronous updation, it will not get updated in the client servers simultaneously. So after the synchronous updation, when we give the query the processing time is high compared to the asynchronous updation because the data in the client server based on synchronous updation would be more compared to the data in the client server based on asynchronous updation. The updation in asynchronous method will take place after a certain time period. Therefore, after the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

updation is done in asynchronous mode and if we give the query, the execution time would be same for both modes of updations.

6. CONCLUSION

In this paper we have proposed a modified dynamic data replication strategy algorithm with synchronous and asynchronous updation. We have modified the popularity degree which is the access frequency based on time factor. Generally, the dynamic data replication strategy consist of three stages which are identification of data file to replicate, number of replicas to be created and where to place the replicas. The popularity degree is used in the first stage of the dynamic data replication strategy and we have modified the first stage of the dynamic data replication strategy. The modification is based on the double exponential moving average function and the access frequency based on users. We have compared the modified dynamic data replication strategy and the normal dynamic data replication strategy [7] in terms of system availability based on the replica numbers. The performance comparison showed that our proposed technique has better system availability than the existing technique. This indicates that our proposed technique replicated more recently accessed file and increased the system availability. We have also compared the execution time of synchronous and asynchronous updation and it showed that the execution time for asynchronous updation is less compared to the synchronous updation.

REFERENCES

[1] Leavitt N, 2009, "Is Cloud Computing Really Ready for Prime Time?," *Computer*, Vol. 42, pp. 15-20, 2009.

[2] Weinhardt C, Anandasivam A, Blau B, Stosser J, "Business Models in the Service World", IT Professional, vol. 11, pp. 28-33, 2009.

[3] Ghemawat S, Gobio H, Leung S T. "The Google file system,"*ACM SIGOPS Operating Systems Review*,vol.37,no.5,pp.29-43. 2003

[4] Shvachko K, Hairong K, Radia S, Chansler R. "The Hadoop distributed file system". *In Proc. the 26th Symposium on MassStorage Systems and Technologies,* Incline Village, NV, USA, May 3-7, pp.1-10,2010.

[5] Wang S.S, Yan K Q, Wang S C. "Achieving efficient agreement within a dual-failure cloud-computing environment," *Expert System with Applications*, vol.38, no.1, pp.906-915, 2010.

[6] Julia Myint and Thinn Thu Naing, "Management of Data Replication for Pc Cluster Based Cloud Storage System," *International Journal on Cloud Computing: Services and Architecture* (IJCCSA), Vol.1, No.3, November 2011.

[7] Sun DW, Chang GR, Gao S.,"Modeling a dynamic data replication strategy to increase system availability in cloud computing environments," *Journal of Computer Science and Technology*, vol. 27, no.2, pp. 256-272 Mar. 2012.

[8] Mohamed-K Hussein, Mohamed-H Mousa, "A Light-weight Data Replication for Cloud Data Centers Environment," *International Journal of Engineering and Innovative Technology* (IJEIT), Vol.1, no.6, June 2012.

[9] Brian F. Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, Russell Sears, "Benchmarking Cloud Serving Systems with YCSB", *in Proceedings of the 1st ACM symposium on Cloud computing*, pp. 143-154, 2010.