ISSN:0975-9646



# Demands Apply Base Data Replication Over Cloud Computing

Musmade Anjali J.[#] , S. M. Rokade [*]

[#]*Computer Engineering Department, SVIT COE, Nasik, University of Pune ,India.*
[*]*Head of Computer Engineering Department, SVIT COE, Nasik, University of Pune ,India.*

***Abstract*** **- Cloud computing is an important mechanism for utilizing computing services. So most of the applications are developed in cloud computing environment. As there are number of applications they have different quality of service (QoS) requirements. Due to the data corruption in data nodes, numbers of applications are unable to reach their successful outcomes. To support such type of applications continuously we are performing data replication on the basis of QoS requirements of the corresponding application. For performing such type of data replication we are developing an algorithm using some concepts of HQFR [High QoS First Replication] algorithm. Along with the QoS requirement our main goal is to minimize data replication cost. So we are developing another algorithm which is inspired from MCMF [Minimum Cost Maximum Flow] algorithm. At the end we will propose an efficient scheme for data replication on the basis of QoS requirement.**

***Index Terms***—**Cloud Computing, Data replication, Quality of service.**

## I. INTRODUCTION

Cloud computing is becoming an important mechanism for utilizing computing services worldwide. Cloud computing have different features like transparency in resource allocation and service provisioning facilities. There is a rapid growth in new information services and business oriented applications via internet. Because of the flexible nature of cloud computing environment most of the applications are developed in this environment. As there is tremendous increase in data intensive applications there is necessity of new efficient techniques for processing of a huge volume of data. Cloud computing have focus on scalability and availability of large scale applications. Apache Hadoop can be regarded as a typical example of cloud computing.

Cloud computing system processes a large volume of data in the network for performing a huge number of applications. In the network, there are number of nodes. As there are huge number of nodes there is more possibility of hardware failure or system failure. Due to hardware failure some data stored at node may get corrupted. Simultaneously an application in running state which has request of that corrupted data cannot access that data. Due to this data corruption, most of the applications are unable to reach their successful outcomes. So for supporting such type of applications continuously by avoiding data corruption a new technique is introduced in cloud computing system which is nothing but data replication. In data replication we maintain more than one replica that means copies of each data block to avoid data corruption. There are different techniques used for data replication in cloud computing. But very few of them concerned about quality of service requirements of applications.

In this paper, we are focusing on quality of service requirements of applications while performing data replication. According to our knowledge very few papers are concerned with this quality of service(QoS) requirement problem in data replication. Here QoS requirement is considered from the aspect of request information of an application. We are trying to investigate the problem of applications regarding QoS requirements while performing data replication. The problem is stated as; due to limited space for replication some high QoS data get stored at lower performance node and cannot reach to the appropriate application to give correct results. Sometimes low QoS data get stored at high performance node, that data is not in use for longer time, in this way it is wastage of data replication space at high performance node. This type of lower frequency data blocks which are not in use by any application or they cannot meet their appropriate application, are known as QoS violated data replicas. We are trying to solve this QoS problem in data replication and will propose an efficient technique to improve this problem. Along with this our goal is to minimize the total data replication cost and to minimize the number of QoS violated data replicas [1].

In our newly proposed technique we will use some algorithms. The idea for first algorithm is taken from HQFR algorithm. HQFR stands for High QoS First Replication. In this algorithm the application having high QoS will be considered first and it can perform data replication. Along with the minimization of replication cost our aim is to minimize the number of QoS violated data replicas. For achieving such type of goal we will introduce another algorithm which gives optimal solution to the QoS requirements in data replication. We are also interested in finding efficient energy consumption of nodes in cloud computing system. Our goal is to perform data replication by considering the QoS requirements of application and also the energy consumption requirements of applications.

Apart from this our paper is distributed in following sections. Section II describes the related work. Section III gives the system model. Section IV describes efficient schemes for data replication. And lastly we conclude our paper work.

## II. LITERATURE SURVEY

To tolerate failures of application in the cloud computing various concepts are introduced. We will have look to them.

Some techniques are introduced to avoid data corruption in hadoop distributed file system (HDFS). For NameNode failure checkpoint method is used. In this method NameNode periodically combines the existing checkpoint and journal to create a new checkpoint and an empty journal. Where checkpoint is persistent record of the image stored in the local host's native file system and journal is storage of changes in the image. The checkpointNode usually runs on a different host from the NameNode since it has the same memory requirements as the NameNode. It takes the current checkpoint and journal files from the NameNode, merges them locally and returns the new checkpoint back to the NameNode. Creating checkpoints is one way to protect the file system metadata [2].

Another concept of BackupNode is discussed in the same paper. It is similar to CheckpointNode, creating periodic checkpoints but in addition it keeps record of an inmemory, up to date image of the file system namespace which is always synchronized with the NameNode. If NameNode fails, the BackupNode's image in memory and the checkpoint on disk is a record of the latest namespace state [2].

A new technique is introduced to tolerate DataNode failure which is nothing but the data replication. In this technique they maintain more than one copies of each data block. In HDFS default data replication factor is two. When a new block is created, it places the first replica on the node where the writer is located, the second and third replicas on two different nodes in a different rack and the rest are mounted on random nodes with restrictions such that no more than one replica is placed at one node and no more than two replicas are placed in the same rack when the number of replica is less than twice the number of racks [2].

There are many techniques introduced to improve data availability and reliability in cloud computing system, but very few of them investigate the problem of QoS requirements of application in data replication.

The problem is investigated of placing the replicas of an object in content distribution systems to meet the QoS requirements of clients with the objective of minimizing the replication cost. In this paper QoS requirements are specified in the form of a general distance metric. They consider two classes of service models which is replica aware services and replica blind services. They proposed efficient algorithms to compute the optimal locations of replicas under different models. Their goal is to find a replica placement that satisfies all requests without violating any range constraint, and minimize the update and storage cost at the same time. They show that this QoS aware replica placement problem is NP- complete for general graphs and they provide two heuristic algorithms called l-Greedy-Insert and l-Greedy-Delete, for general graph and a dynamic programming solution for tree topology. Since l-Greedy-Insert starts by inserting replicas into a empty replica set & l-Greedy-Delete starts by deleting replicas from a full replica set, the execution time of these two algorithms depends on the number of replicas in the optimal solution. So l-Greedy-Insert is more efficient than l-Greedy-Delete when optimal solution has few replicas and l-Greedy-Delete become more efficient when optimal solution has many replicas [3].

A simple problem formulation is NP-complete is shown firstly. They present different types of heuristics for object replication that satisfies the specified access time deadlines and they try to achieve low storage overhead. Their goal was to minimize the number of replicas presented in system. They proposed a simple algorithm which finds the solution of QoS aware problem. This algorithm may be known as Greedy MSC [Greedy Minimum Set Covering [4].

Authors presented a new heuristic algorithm for QoS aware problem which is based on the idea of cover set. It determines the positions of a minimum number of replicas expected to satisfy certain quality requirements. Their placement algorithm exploits the data access history for popular data files and computes replica locations by minimizing QoS satisfaction for a given traffic pattern [5].

Authors proposed a new heuristic algorithm, called Greedy-Cover, which finds good solutions for QoS aware replica placement problem in general graph. Algorithm helps to decide the positions of the replicas to improve system performance and satisfy the quality requirements specified by the user simultaneously [6].

Authors proposed a replica replacement strategy to make dynamic replica management effective. A dynamic replacement strategy is proposed for a domain based network where a weight of each replica is calculated to make the decision for replacement [7].

All the above explained algorithms are not suitable for solving our QoS aware problem. In our system number of replicas is fixed for specific data block. So it is not easy to minimize replication cost with fixed number of replicas. Our QoS aware problem is concerned about minimization of replication cost, minimization of QoS violated data replicas and minimization of energy consumption.

## III. SYSTEM MODEL

For designing our replication strategy we refer to the architecture of Hadoop Distributed File System. The basic architecture of HDFS cluster is consisting of two major nodes NameNode and DataNode [2].

Fig.1 shows the architecture of HDFS. Multiple DataNodes are mounted in a rack. There are number of racks at a NameNode. They communicate with each other through switches. Files and directories are represented on the NameNode by i nodes, which record attributes like permissions, modification and access times, namespace and disk space quotas. The file content is divided into large numbers of blocks and each block of the file is independently replicated at multiple DataNodes. NameNode keeps the record of datablocks stored in DataNodes. DataNodes send heartbeats to the NameNode to represent its proper functioning.

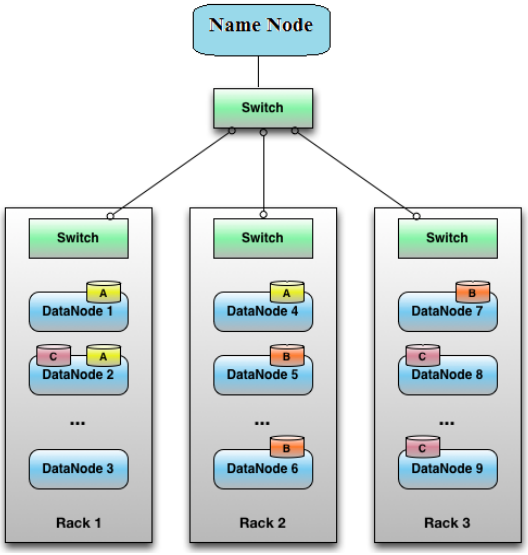

Fig.1: Hadoop Distributed File System

In a cloud computing cluster of thousands of nodes, failures of a node (most commonly storage faults) are daily occurrences. A replica stored on a DataNode may become corrupted because of faults in memory, disk, or network. To avoid the data corruption, the data replication technique is used which provide high data availability. Because of the heterogeneous nature of node, the data of a high-QoS application may be replicated in a low-performance node (the node having slow communication and disk access latencies). After that, if data corruption occurs in the node running the high-QoS application, the data of that application will be revert from the low-latency node. Since the low-performance node has slow communication and disk access latencies, the QoS requirement of the high-QoS application may be violated. Note that the QoS requirement of an application is defined from the aspect of the request information.

We are trying to investigate the problem of QoS requirements satisfaction while performing data replication in cloud computing. The problem is given as, due to limited space for replication some high QoS data get stored at lower performance node and cannot reach to the appropriate application to give correct outcomes. Sometimes low QoS data get stored at high performance node, that data is not in use for longer time, in this way it is wastage of data replication space at high performance node. This type of lower frequency data blocks which are not in use by any application or they cannot meet their appropriate application, are known as QoS violated data replicas. We are trying to solve this QoS problem in data replication and will propose an efficient technique to improve this problem. Along with this our goal is to minimize the total data replication cost and to minimize the number of QoS violated data replicas.

## IV. AN EFFICIENT DATA REPLICATION TECHNIQUE

In this section, we will represent two efficient algorithms for solving QoS aware problem in data replication. The main goal of our scheme is to minimize replication cost and to minimize QoS unsatisfied data replicas count. We are also interested in finding an efficient algorithm for performing data replication with minimum energy consumption. We have some assumptions for solving our QoS aware problem:

- *Assumption 1*: consider a cloud computing system with a set of storage node (S). These nodes can be run applications along with storing data. The functionality of storage nodes is similar to the storage nodes in HDFS.
- *Assumption 2*: Let r be the requested node such as r $\epsilon$ S. If node r is running its application writes a data block b to the disk of node r. Then the replication request is forwarded from node r to the cloud computing system. Then the copies of block b are replicated to other nodes in the cloud computing system.
- *Assumption 3*: Now let q be the satisfied node. That mean a replica copy of block b from node r is stored as dr block at node q. Let T be the time required to store dr. dr is associated to the replication cost (RC) and total access time of replication (AC).
- *Assumption 4*: When node r cannot read its data block b due to data corruption then it will retrieve the data replica dr from the node q, but if access time AC is greater than the time T, then dr becomes QoS violated (unsatisfied) data replica.
- *Assumption 5*: Our main goal is to minimize the total number of Qos violated data replicas and the total replication cost for all data blocks.

Now we will see the algorithms:

*A.* ***HQFR algorithm:*** As the name indicates High QoS First Replication algorithm. The main thing is that we are considering the QoS requirement from the aspect of request information and its access time only. In HDFS the data is divided into 64MB data blocks. The replication factor is two in HDFS. There are two numbers of copies of data block other than the original one. And that two copies are stored on different DataNodes or different data racks. And the NameNodes keeps track of all the replicas other than original copy and they mounted on different data racks to avoid rack failure.

***Basic idea of algorithm:*** As the name indicates the applications with high QoS should be replicated first. According to our knowledge the high QoS application have stricter requirements in the response time of a data access time than the normal applications. High QoS requirement application should take precedence over the low QoS requirement application to perform data replication.

We have to sort all the applications according to their QoS requirement in a way, the application with high QoS should come first and then the lower one. If the data replication space is limited then first stores the data replicas of high QoS applications. When the high QoS application reads a corrupted data replica , its QoS requirements can be supported continuously by retrieving the data replica from high performance node.

Fig.2 shows HQFR algorithm below:

**Input:** A set of requested nodes $S_r$.
**Output:** QoS-aware data replica placement.
1: Sort the requested nodes in $S_r$ based on their associated access time.
2: **for** each requested node $r_i$ in the sorted list of the requested nodes **do**
3: $S_q{}^{r_i} \leftarrow$ Find the corresponding qualified nodes of $r_i$ using Eq. (1) and (2) to verify all storage nodes.
4: Select the $r_f$ qualified nodes from $S_q{}^{r_i}$ which have smaller data replica access time than other qualified nodes in $S_q{}^{r_i}$.
5: **for** each selected qualified node $q_j$ **do**
6: Store one data replica from $r_i$ to $q_j$.
7: Update the available replication space of $q_j$.
8: **end for**
9: **if** $|S_q{}^{r_i}| < r_f$ **then**
10: $S_{uq}{}^{r_i} \leftarrow$ Find the corresponding un-qualified nodes of $r_i$ only using Eq. (1).
11: Select the $r_f - |S_q{}^{r_i}|$ un-qualified nodes from $S_{uq}{}^{r_i}$ which have smaller data replica access time than other qualified nodes in $S_{uq}{}^{r_i}$.
12: **for** each selected un-qualified node $uq_j$ **do**
13: Store one data replica from $r_i$ to $uq_j$.
14: Update the available replication space of $uq_j$.
15: **end for**
16: **end if**
17: **end for**

Fig.2 : HQFR Algorithm

While processing these replication request we have to find the qualified node's list which helps to satisfy the QoS requirements of the appropriate application while running. The QoS requirement is given in the form of access time of that data block which is requested by an application. Note that while finding qualified node it should satisfy two conditions:

- The requested node $R_i$ and its qualified node $Q_j$ should not be mounted in the same rack. They should belong two different racks.
  $Rack(R_i) \neq Rack(Q_j)$ (1)
  Where Rack() is the function to determine in which rack a node is located.
- The total data replica access time from qualified node $Q_j$ to request node $R_i$ ($T_{access}(R_i,Q_j)$)should be smaller than the QoS requirement of running application in $R_i$ which is $T_{qos}$. $T_{access}(R_i,Q_j) \leq T_{qos}$ (2)
  After finding the qualified nodes by using these two conditions the data block can store its one data replica in each qualified nodes and the qualified nodes update their replication space respectively.

Now we will calculate the total replication cost. In HQFR algorithm the total replication cost is represented by the total storage cost taken by all the requested nodes to store their appropriate replicas. The replication cost is nothing but the total summation of storage costs of all data block replicas.

But we are mainly interested in minimizing replication cost and also the number of QoS violated data replicas. For achieving second objective we are going to propose another algorithm for data replication.

*B.* ***An efficient replica placement algorithm:*** As its name indicates, this algorithm gives an efficient solution to the QoS aware replication problem. In this algorithm we are transforming QoS aware problem to the MCMF [Minimum Cost Maximum Flow] problem. As same to the previous algorithm in this algorithm also we are going to find out $S_qR_i$ the set of qualified nodes for each requested node $R_i$. Then after we will make union of the set of qualified nodes $S_q$ with the newly derived set $S_qR_i$ which is set of qualified nodes corresponding to each requested node $R_i$. Then by using set Sr and $S_q$ form a network flow graph. The vertices in the graph are from both the sets Sr and $S_q$ and each edge represents the pair of appropriate capacity and cost of the data replication. Then by applying a suitable MCMF algorithm find out an efficient solution for that network flow graph. Then after we will perform the same operation for the unqualified nodes corresponding to each requested node $R_i$. Form the new graph from both the sets described above. Solve the graph by using same MCMF algorithm. Consider both the solutions obtained previously and perform an efficient optimal placement of all QoS violated data replicas.

**Input:** A set of requested nodes $S_r$.
**Output:** Optimal placement for the QoS-satisfied and QoS-violated data replicas.
$Sq \leftarrow \emptyset$.
**for** each requested node $r_i$ in $S_r$ **do**
$S_q{}^{r_i} \leftarrow$ Find the correspondingly qualified nodes of $r_i$.
$Sq \leftarrow Sq \cup S_q{}^{r_i}$.
**end for**
Use $S_r$ and $Sq$ to model a network flow graph.
Set appropriate (capacity, cost) values on the edges of the network flow graph.
Apply an existing polynomial-time MCMF algorithm to obtain the MCMF solution of the network flow graph.
Obtain the optimal placement for the QoS-satisfied data replicas from the MCMF solution.
$S_{ur} \leftarrow \emptyset$ and $S_{uq} \leftarrow \emptyset$.
**for** each requested node $r_i$ in $S_r$ **do**
$f_{leaving} \leftarrow$ the amount of flow leaving from $r_i$.
**if** $f_{leaving} < r_f$ **then**
$S_{ur} \leftarrow S_{ur} \cup r_i$.
$\overline{S_q{}^{r_i}} \leftarrow S - S_q{}^{r_i}$. /* The un-qualified nodes for $r_i$*/
$S_{uq} \leftarrow S_{uq} \cup \overline{S_q{}^{r_i}}$.
**end if**
**end for**
Use $S_{ur}$ and $S_{uq}$ to model a new network flow graph.
Set appropriate (capacity, cost) values on the edges of the new network flow graph.
Apply the MCMF algorithm on the new network flow graph to obtain the MCMF solution.
Follow the MCMF solution to make the optimal placement of the QoS-violated data replicas.

Fig. 3. An efficient replica placement algorithm

Because of optimal placement of Qos-violated data replicas the number of these replicas are minimized, which is our main goal. As we have used MCMF algorithm in this scheme, we get our solution in polynomial time. In this scheme the second part is having one flow graph. The amount of this flow graph is the amount of flow leaving from requested node $R_i$.. Here we are considering the amount of flow leaving, which is not added to the total replication cost, which automatically helps in minimizing the total replication cost. Hence we achieved our both objectives. Fig.3 shows our second algorithm.

But we are interested in minimizing energy consumption in data replication. We are going to investigate another algorithm for energy optimization.

*C.* ***Efficient Energy optimization algorithm*:** In this algorithm similar to above algorithm we are finding a set of qualified nodes corresponding to each requested node $R_i$. Create set of such nodes. After that we have check status of each qualified node. So we will collect energy status of each node and make another set for this $E_r$ which is shown in fig.4.

**Input:** A set of requested nodes Sr.
**Output:** Energy optimization in data replicas.
1. Sq ← ∅.
2. **for** each requested node ri in Sr do
3. Sq ri ← Find the correspondingly qualified nodes of ri.
4. Sq ← Sq ∪ Sqri
5. **end for**
6. **ER** <- ∅
7. For each elected node
8. Ei <- Collect energy status of ei
9. End for
10. Decisionpolicy ()
11. end

Fig. 4. Efficient Energy optimization algorithm

Then according to the energy status of nodes, with higher energy should come first. The replication request of that node should be considered first from the set of requested node. So the replication request is performed in minimum time with efficient energy. This algorithm is not derived in real time yet, but we are trying to develop this algorithm soon.

## V. RESULT ANALYSIS

We have created our own cloud computing environment up to our possible levels. We are successful in passing messages between different nodes in the cloud computing system. Now we are applying our applications to the datanodes through the namenodes. We are performing data replication by using our proposed algorithms and trying to sort out our problem.

## VI. CONCLUSION

We have investigated the QoS requirement problem in data replication in cloud computing system. We have been proposed an efficient scheme to solve the QoS aware problem in data replication. First algorithm is inspired from HQFR algorithm. This algorithm cannot give optimal solution to the QoS aware problem. So we proposed another algorithm which gives optimal solution in polynomial time. This algorithm also helps in achieving both the objectives of our paper which is minimization of replication cost and minimization of QoS violated data replicas.

In future, we are going to find out an efficient energy optimization algorithm to energy consumption problem of nodes while performing data replication in cloud computing system. We want to develop a new algorithm which gives proper solution to this problem. We will try our best levels to implement that algorithm in real time.

## REFERENCES

*Journal Papers:*

[1] Jenn-Wei Lin, Chien-Hung Chen, and J. Morris Chang " QoS – Aware Data Replication for Data Intensive Applications in Cloud Computing System", Digital Object Identifier 10.1109 / TCC 2013 IEEE.

[2] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop Distributed File System," in Proc. IEEE 26th Symp. Mass Storage Systems and Technologies (MSST), Jun. 2010, pp. 1–10.

[3] X. Tang and J. Xu, "QoS-Aware Replica Placement for Content Distribution," IEEE Trans. Parallel and Distrib. Syst., vol. 16, no. 10, pp.921–932, Oct. 2005.

[4] Won J. Jeon, I. Guptaand K. Nahrstedt "QoS Aware Object Replication in Overlay Networks", IEEE GLOBECOM 2006, IEEE.

[5] X. Jia, Deying Li, Hongwei Du and Jinli Cao "On Optimal Replication of Data Object at Hierarchical and Transparent Web Proxies", IEEE Transactions on Parallel and Distributed Systems, VOL 16, No. 8. August 2005.

[6] H. Wang, Pangfeng Liu, Jan-Jan Wu "A QoS aware Heuristic Algorithm for Replica Placement" Grid Computing Conference 2006 IEEE.

[7] K.Shashi and T. Santhanam "Replica Replacement Algorithm for Data Grid Environment", ARPN Journal, Vol. 8,No. 2, Feb 2013.

[8] Nihat Altiparmak and Ali S, aman Tosun "Integrated Maximum Flow Algorithm for Optimal Response Time etrieval of Replicated Data" 2012 41st International Conference on Parallel Processing

[9] Da-Wei Sun ,Xing-Wei Wang "Modeling a Dynamic Data Replication Strategy to Increase System Availability in Cloud Computing Environments" JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(2): 256272 Mar. 2012.

[10] Mohamed-K HUSSEIN, Mohamed-H MOUSA"A Light-weight Data Replication for Cloud Data Centers Environment" International Journal of Engineering and Innovative Technology (IJEIT) Volume 1, Issue 6, June 2012

[11] Dejene Boru, Dzmitry Kliazovich, Fabrizio Granelli, Pascal Bouvry, and Albert ,Y. Zomaya"Energy-Efficient Data Replication in Cloud Computing Datacenters".

[12] M. Creeger, "Cloud Computing: An Overview," Queue, vol. 7, no. 5,pp. 2:3–2:4, Jun. 2009.

[13] K. V. Vishwanath and N. Nagappan, "Characterizing Cloud Computing Hardware Reliability," in Proc. ACM Symp. Cloud Computing, Jun.2010, pp. 193–204.

[14] E. Pinheiro, W.-D. Weber, and L. A. Barroso, "Failure Trends in a Large Disk Drive Population," in Proc. 5th USENIX Conf. File and Storage Technologies, Feb. 2007, pp. 17–28.