Hindawi
Journal of Control Science and Engineering
Volume 2017, Article ID 2428982, 8 pages
https://doi.org/10.1155/2017/2428982

Research Article

An Optimized Replica Distribution Method in Cloud Storage System

Yan Wang and Jinkuan Wang

College of Information Science and Engineering, Northeastern University, Shenyang 110819, China

Correspondence should be addressed to Yan Wang; wangyan@mail.neuq.edu.cn

Received 25 January 2017; Revised 16 July 2017; Accepted 6 August 2017; Published 12 September 2017

Academic Editor: Carlos-Andrés García

Aiming at establishing a shared storage environment, cloud storage systems are typical applications of cloud computing. Therefore, data replication technology has become a key research issue in storage systems. Considering the performance of data access and balancing the relationship between replica consistency maintenance costs and the performance of multiple replicas access, the methods of replica catalog design and the information acquisition method are proposed. Moreover, the deputy catalog acquisition method to design and copy the information is given. Then, the nodes with the global replica of the information replicate data resources, which have the high access frequency and the long response time. Afterwards, the Markov chain model is constructed. And a matrix geometric solution is used to export the steady-state solution of the model. The performance parameters in terms of the average response time, finish time, and the replica frequency are given to optimize the number of replicas in the storage system. Finally, numerical results with analysis are proposed to demonstrate the influence of the above parameters on the system performance.

1. Introduction

Now, in order to further improve data availability in cloud storage systems, data replication technology has become a hot popular topic. Especially in the current commercial cloud system, typical examples include Amazon S3 (Amazon Simple Storage Service) [1], Google File System (GFS) [2], and Hadoop Distribution File System (HDFS) [3]. HDFS and GFS are based on centralized storage directory. Amazon S3 uses consistent hashing distribution strategy based on replicating.

In terms of dynamic replication policy, Sahi and Dhaka [4] presented a replica algorithm on predicting the workload of cloud service about the Neural Network; Bao [5] developed a replication mechanism to solve replica dynamic adjustment problems. Ooi et al. [6] proposed a dynamic data replication strategy according to access records. Therefore, researchers discussed replication distribution optimization problem. In literature [7], Ghilavizadeh et al. proposed a dynamic replication optimal distribution strategy. It reduced the total task execution time and improved the accessibility.

At the base of the dynamic replication mechanisms mentioned above, literatures [8–10] optimized load balance in the storage system. An optimized strategy was proposed in literature [11] for load balance. In literatures [12, 13], there were optimized or elastic methods that enabled the capabilities of VMs (Virtual Machines) to be scaled to encompass the dynamically changing resource demand of the aggregated virtual services. As we all know, data reliability is one of the indexes for evaluating the merits of a storage system performance. Mansouri [14] introduced a prediction algorithm for load prediction based on time sequence. In literature [15], researchers computed the performance in terms of the resource priority and storage space. They constructed a reliability analysis model and applied it to the current cloud storage system. In order to increase availability, fault tolerance, and load balancing, the impact of data grid architecture on dynamic replication performance is investigated in literature [16]. Ajitabh and Ritu [17] introduced a new approach as a fault tolerance and flexible policy.

Researches above focused on the distribution of a replica and data reliability. But they did not take the impact of the number of replicas on the cloud storage system performance into account. In cloud storage system, the more replicas there are, the more nodes in response to users' request data will

be. However, if there are more replicas, network bandwidth and resource consumption will be greater in cloud computing center.

In this paper, considering both the data availability and the average access latency, a replica catalog and replica information obtained method is designed as the basis for placing the replica. Moreover, we construct Markov chain [18–20] to evaluate the number of replicas. By introducing the concept of replication factor, we give the formulas for computing the appropriate location to place the replica. We propose a policy on how to update and delete replica based on the above analysis. Finally, we investigate the system performance.

2. Replica Directory Structure Based on Domain

The data center is described as a hierarchical structure, shown in Figure 1. Nearby data nodes in the system are connected by a domain node forming a domain. Each domain has a domain head-node. The domain head-node is responsible for managing information of other nodes in the domain and all information of replica in the domain. Meanwhile, it periodically performs replica distribution algorithm. Non-head-nodes manage their own data, such as catalog of access file, and regular exchange of information between head-node and non-head-node.

Definition 1. All shared resources, including file, database, and data tables, can be expressed as logical resources. Each logical resource with a globally unique logical resource identifier is denoted as LR.

Definition 2. Each logical resource can have one or more replicas. Each replica has a globally unique physical replica of identifier, denoted as PR. The first registration resource is called the original replica.

The set of all the replicas of logical resource is denoted as $\mathrm{LR} = \{lr_1, lr_2, \ldots, lr_i\}$. A node with physical replica is denoted as p; the set of these nodes is $\{p_1, p_2, \ldots, p_i\}$, where i is the number of replicas.

The same replica of the same logical resource has the same identifier but different physical identifiers. Each node in the network maintains a local replica catalog.

When a node registers a shared resource, the data management manages the LR and assigns PR.

To dynamical replica the management service assigns PR and maps this PR with LR. Replica management records the original replica information of this logical resource, while servicing the replica information into the replica database of the original replica.

We design a replica catalog structure based on two-direction structure. One of the catalogs is the structure between the replica catalog of logical resource and the node with its replicas. Another structure is the replica and its replica catalog of logical resource. From the first structure, we can get the specific access situation. From the second structure, the replica can connect to the replica catalog of logical resource. Moreover, we can get all the information of the logic resource.

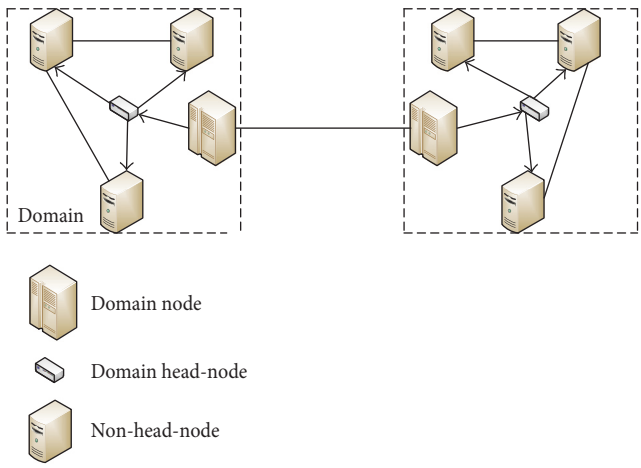

Figure 1: Structure for data center.

In order to obtain the access information within the global scope, we should be sure of the relationship between the various types of nodes and management responsibilities.

Domain node is the node that has strong function, high bandwidth, long time online charging for data replication, and replica management.

Each ordinary node can simultaneously connect to multiple domain nodes but can only choose one domain node as its parent node.

Each domain node can be simultaneously connected to multiple domain nodes, but only one or no domain node can be its parent node. Domain node without parent node is called domain head-node.

All parent-child relationships between all domain nodes cannot form a loop.

For the domain node, in addition to the local replica catalog information with the shared resources, it also includes replica catalog information of its child nodes. When registering a shared resource node, if the node is a domain node, management service will copy data into a local replica catalog; otherwise, the copy will be inserted into the parent node replica catalog.

All the available logical resources will be recorded to the corresponding side in this catalog according to this information collection and access method. The catalog includes the initial access request for the node, a replica of the node with physical access to the real-time network, the actual response time, and the waiting queue length on the node.

3. Replica Distribution Method

Domain node is responsible for the data replica of the original resources and original node of the child nodes. Moreover, it counts the scope of the visits in the global domain. Each node has only one common parent. There is no intersection in the sharing of resources belonging to each domain node. Domain head-node is responsible for information access frequency, cost of access, and the information of the node when the node

obtains resource in the domain. Each domain node can have no more than one parent, so there is no intersection between domains.

3.1. Replica Lacking Nodes

Definition 3. In unit time, F denotes the relation of the frequency of access and the number of replicas as follows:

$$F = \frac{\sum_{i=0}^{I} m'_{cj,i}}{I_j}, \tag{1}$$

where $m'_{cj,i}$ is the number of times for accessing the physical replica per unit time for LR_i and I_i is the number of replicas each time the replicas are built.

If the access frequency is greater than a given threshold ω, we denote F as hot resource.

If the number of times for access is high to the logic resource but the logical resource has a lot of replicas, the resource is not hot.

Definition 4. Per unit time, the ratio is made by the sum of the response time that accesses all the replicas and the number of times for accessing all the replicas. It is denoted as follows:

$$\overline{T}_{rj} = \frac{\sum_{k=0}^{m'_{cj}} T_{rk}}{m'_{cj}}, \tag{2}$$

where m'_{cj} represents the number of times that all replicas have being accessed per unit time. T_{rk} indicates the kth time that users access the replica of logical LR_i per unit time.

For each hot resource, we need to further determine the average response time in order to decide whether the resource needs to create a replica or not.

Response time of resources depends on many factors, including bandwidth, node performance, and the size of the data resource. The larger physical size of logical resources is, the longer average response time may be. But it does not explain the fact that there are not enough replicas of the resource. It is also necessary to introduce unit average response time to determine whether the logic resource requires a replica or not.

We create a replica for resources with no replica using two evaluation indexes, that is, the access frequency and the average response time. We export high frequency of access and long average response time as the judgment for creating a replica rather than using one single evaluation index to judge the creation.

For each logical resource needed to create a replica, domain node according to a replica of the node p calculates the average response time as follows:

$$T_{rj,D_k} = \frac{\sum_{p \in D_k} T_{rj,i}}{m'_{cj,D_k}}, \tag{3}$$

where $T_{rj,i}$ is the response time for p_i accessing the logical resource LR_i. m'_{cj,D_k} is the number of times that all nodes in the domain D_k access the resource LR_i.

If $T_{rj,D_k} > \overline{T}_{rj}$, the domain D_k is called a replica lacking domain. Average response time that the node requests the resource within the domain can reflect lack of resource and access degree. If a domain average response time is greater than the average response time of logic resource LR_i, it indicates that the number of replicas in domain D_k is smaller than nodes in other domains. We need to add more replicas for resource LR_i. In the domain with long average response time, while we create replicas, it will minimize the average response time of the logic resource LR_i.

3.2. The Number of Replicas. We give a two-dimensional discrete-state process $\{m_c(t), m_s(t)\}$, where $m_c(t) = i$, $i \in \{0, 1, 2, \ldots\}$, denotes the number of requests for resource at time t and $m_s(t) = j$, $j \in \{0, 1, 2, \ldots, I\}$, denotes the number of available replicas for resource at time t. The state space for the process is as follows: $\Xi = \{(i, j) \mid i = 1, 2, \ldots, j = 1, 2, \ldots, I\}$, when $m_c(t) = i$ and the system is at level i.

We define the time for an available replica as the loop of the replica. Assume that the loops of the replicas are independent and each of them follows an exponential distribution with parameter λ_s and $\lambda_s > 0$. The generation period of the replicas is independent and identically distributes random variables. Each of them follows an exponential distribution with parameter μ_s and $\mu_s > 0$. Data access request arrival process is a Poisson process with the arrival rate $\mu_c > 0$ and the service time for access request is with the parameter $\lambda_c > 0$ as the exponential distribution.

When the number of requests for the resource is i in the system, this denotes that the system is at level i. At the same time, the number of the replicas is n. When the number of available replicas is n and $0 \le n \le I$, the number of the available replicas in the next state may increase by one due to the generation or reducing the node leaving the field, but it is also likely to be unchanged. Use $\mathbf{B}$ to represent the system matrix at the state transition matrix under the same level as follows:

$$\mathbf{B} = \begin{bmatrix} -I\mu_s & I\mu_s & & & & \\ \lambda_s & -(I-1)\mu_s - \lambda_s(I-1)\mu_s & & & & \\ & \ddots & \ddots & \ddots & & \\ & & (I-1)\lambda_s & (I-1)\lambda_s - \mu_s & \mu_s & \\ & & & I\lambda_s. & -I\lambda_s & \end{bmatrix}. \tag{4}$$

When the system is at level i, there is no new access request in the system and no access request reaching the end.

Let $\mathbf{X}_i$ be the transition rate matrix for the system being at level i and $i > 0$. $\mathbf{X}_i$ is given as

$$\mathbf{X}_i = \begin{cases} \operatorname{diag}(-\mu_c) + \mathbf{B} & i = 0 \\ \operatorname{diag}(-\mu_c - \min(i, j)\lambda_c) + \mathbf{B} & 1 \le i \le I-1 \\ \operatorname{diag}(-\mu_c - j\lambda_c) + \mathbf{B} & i \ge I. \end{cases} \quad (5)$$

We denote $\mathbf{Y}_i$ as the transition rate matrix for the system transferring from level i to level $i+1$. $\mathbf{Y}_i$ is obtained as follows:

$$\mathbf{Y}_i = \begin{cases} \operatorname{diag}(0, \min(i, j)\lambda_c, \ldots, \min(i, j)\lambda_c), & 1 \le i \le I-1 \\ \operatorname{diag}(0, \lambda_c, 2\lambda_c, \ldots, I\lambda_c), & i \ge I. \end{cases} \quad (6)$$

Let $\mathbf{Z}$ be the transition rate matrix for the system transferring from level i to level $i+1$. $\mathbf{Z}$ is then given as follows:

$$\mathbf{Z} = \operatorname{diag}(\mu_c, \ldots, \mu_c). \quad (7)$$

Let $\mathbf{A}$ be the transition rate matrix of the Markov chain $\{m_c(t), m_s(t)\}$ as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{X}_0 & \mathbf{Z} & & & & & \\ \mathbf{Y}_1 & \mathbf{X}_1 & \mathbf{Z} & & & & \\ & \mathbf{O} & \mathbf{O} & \mathbf{O} & & & \\ & \mathbf{Y}_{I-1} & \mathbf{X}_{I-1} & \mathbf{Z} & & & \\ & & \mathbf{Y} & \mathbf{X} & \mathbf{Z} & & \\ & & & \mathbf{O} & \mathbf{O} & \mathbf{O} & \end{bmatrix}. \quad (8)$$

Let $\pi_{i,j} = \lim_{t\to\infty} P\{m_r(t) = i, m_d(t) = j\}$ be the stationary probability of the Markov chain. We denote the stationary probability vector for the system being at level i as follows:

$$\boldsymbol{\pi}_i = (p_{i0}, p_{i1}, p_{i2}, \ldots, p_{iI}). \quad (9)$$

The system steady-state vector is expressed as $\boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots)$.

$\boldsymbol{\pi}$ is given by the following equation:

$$(\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_I)\mathbf{A}[\mathbf{Q}] = 0$$

$$\sum_{i=0}^{I-1} \boldsymbol{\pi}_i e + \boldsymbol{\pi}_I (\mathbf{I} - \mathbf{Q})^{-1} e = 1 \quad (10)$$

$$\boldsymbol{\pi}_i = \boldsymbol{\pi}_I \mathbf{Q}^{i-I}, \ i > I,$$

where $\mathbf{A}$ is given as

$$\mathbf{A}[\mathbf{Q}] = \begin{bmatrix} \mathbf{X}_0 & \mathbf{Z} & & & \\ \mathbf{Y}_1 & \mathbf{X}_1 & \mathbf{Z} & & \\ & \mathbf{O} & \mathbf{O} & \mathbf{O} & \\ & & \mathbf{Y}_{I-1} & \mathbf{X}_{I-1} & \mathbf{Z} \\ & & & \mathbf{Y} & \mathbf{QY} + \mathbf{X} \end{bmatrix}, \quad (11)$$

where $\mathbf{Q}$ is the ratio vector.

Let 0 be the initial value of $\mathbf{Q}$ and $\mathbf{Q} = -(\mathbf{Z} + \mathbf{Q}_0^2\mathbf{Y})\mathbf{X}^{-1}$. Iterative method can find an approximate solution for $\mathbf{Q}$ shown as follows.

Step 1. Initialize I and $\mathbf{Q}_0$

$$I \Longleftarrow 1 \ \mathbf{Q}_0 \Longleftarrow 0 \quad (12)$$

Step 2. $\mathbf{Q}_1 \Leftarrow -(\mathbf{Z} + \mathbf{Q}_0^2\mathbf{Y})\mathbf{X}^{-1}$

Step 3. If $\mathbf{Q}_n - \mathbf{Q}_{n-1} > \sigma$

$$I = I + 1$$
$$\mathbf{Q}_n = -\left(\mathbf{Z} + \mathbf{Q}_{n-1}^2\mathbf{Y}\right)\mathbf{X}^{-1} \quad (13)$$

End if

Step 4. $\mathbf{Q} = \mathbf{Q}_n$.

$\mathbf{Q}_0$ is an initial value and $n \ge 1$ is the n-order approximate solution of $\mathbf{Q}$.

3.3. Replica Placement

Definition 5. During per unit time, the rate, which contains the number of response requests with all the requests, is called a node service rate defined as O.

Service rate can measure the service capacity of a node, including node performance, dynamics, and reliability.

Definition 6. Replication factor R is used to measure whether a node is suitable to have a replica as follows:

$$R = \frac{S}{L} \times W \times O, \quad (14)$$

where O is the service rate; S is the storage space of node P; L is the size of physical file of resource LR_j; and W is the average available bandwidth. Available bandwidth can be calculated from the ratio of the length of the waiting queue on the current node and network bandwidth. The greater the length of the waiting queue is, the less available bandwidth will be.

In the cloud storage system, due to the dynamic characteristics of the nodes, network transmission delay becomes an important indicator having an impact on data access. Meanwhile, the processing time of CPU and I/O can be negligible. Therefore, the replication factor, node service rate, available storage, the average available bandwidth, and the length of the average waiting queue are proportional. The size of replication factor indicates whether the node is suitable to place the replica or not.

4. ORRA Algorithm

ORRA algorithm is based on the situation that whether the data resources are hot resources or not, replicas are enough

and the place to locate the replica is suitable. With ORRA algorithm, we can prevent that if there are too much replicas to cause the waste of resources. At the same time, the effective use of appropriate nodes storing replicas could avoid the difficulty of finding the position.

We give the ORRA algorithm as follows.

Input. The information of replica.

Output. The number of replicas and the placement to place replicas.

Step 1. Calculate the access frequency of node resource. If there are more than one pregiven threshold, the data resources are hot. We need to create more replicas.

Step 2. For hot resources, compute their average response time. If the average response time of hot resource is greater than a pregiven threshold value, then it needs to create a replica of the logical resource.

Step 3. A node needs to create replicas, if average response time of all the nodes in the domain is greater than that of the logical resource; it needs to create replicas in the domain.

Step 4. In each domain, be sure of the number of replicas and the appropriate nodes to place replica.

Step 5. Compute the replication factor, select the node with the smallest replication factor as the place to create a replica of the location, and return node address to the domain head-node.

Step 6. Create replicas of data on the specified node.

Step 7. Create a replica of the data on the node calculated.

5. Numerical and Simulation Results

There are 1000 nodes and the bandwidth between nodes is randomly disposed within the range of 100 MB~1000 MB. The number of initial average files is 2000 and each file distributes evenly across the network with 1000 MB.

Two thresholds of algorithm are as follows: one is δ_a, as average visit frequency for resource, and the other is δ_r to take all the resources of the average unit response time.

5.1. Optimization Policy of Number of Replicas

5.1.1. Performance Indicators of Storage System. Reliability of data is the probability of data existing at any given time in the system. The higher the reliability of the data is, the better the storage system is. Data reliability β is defined as the probability that the data has at least one replica for it. In a steady state, the probability for the number of available replicas in the system is given as follows:

$$\sum_{i=0}^{\infty} \pi_{in}. \tag{15}$$

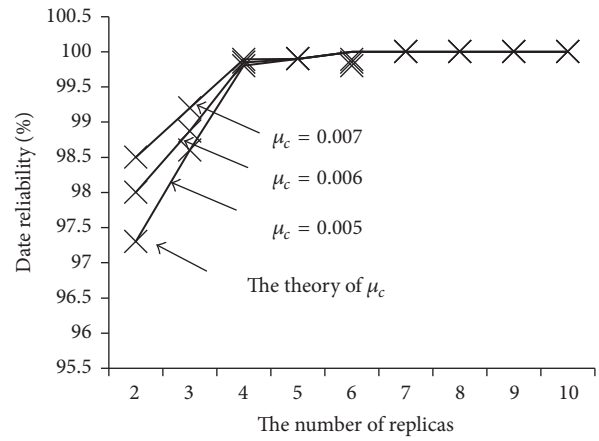

Figure 2: Data availability with number I of replicas.

And β is given as follows:

$$\beta = \sum_{n=1}^{I} \sum_{i=0}^{\infty} \pi_{in}. \tag{16}$$

Data access speed as part of the QoS directly represents the quality of service in the storage system. In order to meet user requirements for data accessibility, the storage system should be able to guarantee its stored data that can be accessed within a reasonable time. Data access latency is defined as the period from the beginning of an access request submitted for the requested data until the user gets back that data. The average latency of data access is given as follows:

$$D = \frac{1}{\mu_c} \sum_{i=0}^{\infty} i \sum_{n=0}^{I} \pi_{i,n}. \tag{17}$$

5.1.2. Optimization Strategy. According to the experimental results, the number of replicas of a single data is optimized by constructing a profit function.

In the experimental system, we set the parameters as follows: the number of replicas is I for one data object in the system. The life of all replicas is subjected to parameter λ_s = 0.001 of the exponential distribution. Data access request transmission time follows an exponential distribution with parameter λ_c = 0.4. Accuracy of Q is $\varepsilon = 10^{-10}$. When μ_c = 0.6, the threshold of creating the replica is 3. When μ_s is 0.005, 0.006, and 0.007, respectively, the trends of single data reliability with the number of replicas are shown in Figure 2.

As can be seen from Figure 2, for the same time for creating replica, reliability increases as the number of replicas increases and trends to 100%. When the number of replicas is at a certain value, the reliability of the data object parameters increases with μ_c. This is because when time for creating replica is in a certain circumstance, the more the number of replicas there is, the smaller the probability of complete data loss is. When there are fixed numbers of replicas, the time for creating replica is larger, the speed for creating replica is faster, and the probability of losing created data is smaller before the creation.

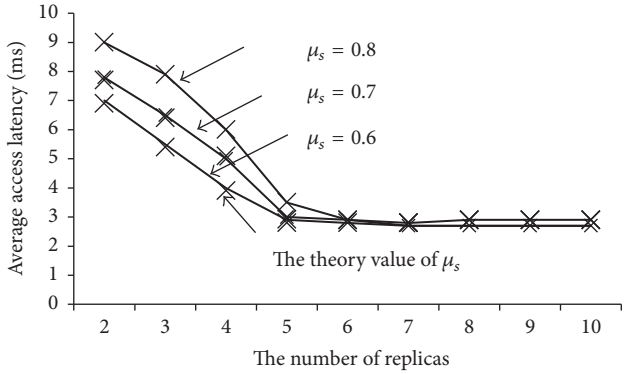

Figure 3: Average access latency with number I of replicas.

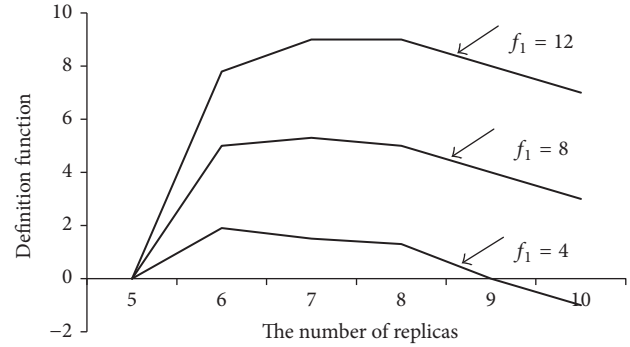

Figure 4: Benefit function with number I of replicas.

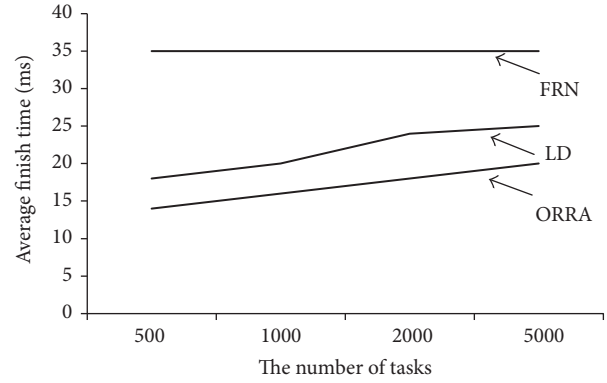

Figure 5: The comparison of average finish times for tasks.

When the threshold of creating the replica is 3, the access time parameter is 0.4. μ_s is 0.6, 0.7, and 0.8, respectively. The access delay means and the number of replicas are shown in Figure 3.

As shown in Figure 3, when access rate is in a certain circumstance, with the increase of the number of replicas, the average data access latency shows a downward trend and stabilized. When there is fixed number of replicas, the average access latency with increasing data access rate increases. This is because when the data access rate is in a certain circumstance, the number of replicas is larger than that of the rate in a normal circumstance. Moreover, the single replica node accessing load to bear is smaller, and the access speed is faster. But after the number of replicas increases to a certain value, the access speed is stabilized. When there are fixed numbers of replicas, the access rate is higher. The access load for a single replica of the data becomes more. Meanwhile, data access speed is slower.

The numerical results above indicate that, for a single node, if it is assigned more replicas, the data has higher reliability. While the average access latency of single data is smaller, the access speed is faster. But when the number of replicas is sufficient, the impact given by increasing the number of replicas is very small. In the meantime, the larger the number of replicas is, the more storage space is occupied. Replicas of administrative expenses also increase.

When the number of replicas is I, reliability and average latency data access to the data are $\beta(I)$ and $D(I)$, respectively.

Supposing that the system meets the reliability requirements, the data of profits when time reduces by one unit is f_1. Each additional fee that there is one more replica is defined as f_2. Profit function is constructed as follows:

$$F(I) = f_1\left(D\left(I_0\right) - D\left(I\right)\right) - f_2\left(I - I_0\right). \tag{18}$$

Set time to create parameter $\mu_s = 0.005$; the reliability of the user requirement is 99% and $I_0 = 5$. $\lambda_s = 0.001$, $\mu_c = 0.8$, and $f_2 = 1$. Profit function with the number of replicas is as the trends shown in Figure 3.

As it is shown from Figure 4, for the number of replicas being 5, there is some difference. In the case where influences of various factors are known, we can find the optimal number of replicas I' which makes system profit maximum. In this case, $f_1 = 4$, $I' = 6$, and the profit value is 1.9. When $f_1 = 8$ and $f_1 = 12$, the optimal number of replicas is $I' = 7$; then the profits value is 5.2 and 8.8, respectively.

5.2. Performance Comparison. ORRA algorithm will be experimented with a fixed number of replicas called FRN (Fixed Replicas Number) algorithm and Local Dynamic (LD) replication algorithm [21, 22]. In FRN, a replica restriction has a fixed number for each data. Each data center has an identification number. In LD, if there is a data read operation by updating the requirements, the operation of the data reading is dynamical.

We use an average finish time and average response time in experiments as the file replica query performance evaluation index.

The average response time and average finish time under different data replication policy are shown in Figures 5 and 6.

FRN algorithm creates a replica through analyzing the economic model based on the replica process, but the process is constructed without considering the dynamic replica process. The average time to complete the task and response time are the longest, respectively, in FRN. LD algorithm has the shortest average finish time. LD algorithm creates a large number of replicas to local file access as the main means of reducing the response time. The algorithm could avoid off-site access and the average time to complete the task. From this policy, the response time is relatively small.

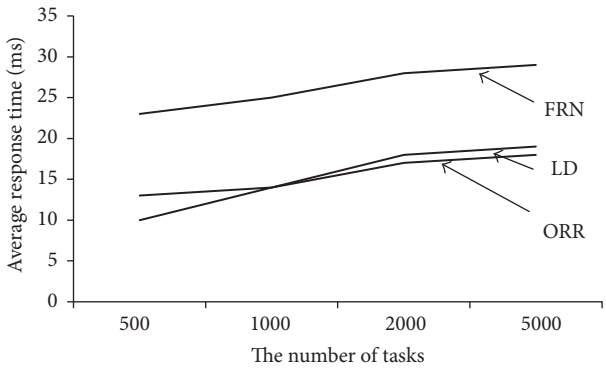

Figure 6: The comparison of average response time.

ORRA algorithm calculates the number of replicas. The replica location in the ORRA algorithm from a global point of view reduces the response time. It is not the final goal to reduce the access to local resources. Thus the average response time of ORRA is greater, but the task's finish time and average response time are less compared to those of LD algorithm. Meanwhile, from the performance analysis by Markov chain method, the ORRA algorithm has the appropriate replica numbers for making the profit maximum.

6. Conclusion

Replication is an important factor in storage system. In this paper, we presented a dynamic data replication strategy to optimize the finish time and the response time. We firstly proposed the replica catalog approach to obtain a replica of all the information about the logical data resource. In order to determine the absence of a replica of the domain based on globally available logic resources, we gave the number of replicas of the calculation algorithm with constructing the Markov chain to capture the replica creation process. Moreover, we provided performance and numerical results analysis and surveyed the influence of the kind of several parameters on the system's performance. The results showed that the appropriate number of replicas and the dynamic creation of replica improve the system's performance and satisfy the users' requirement, while maximizing the benefit.

Conflicts of Interest

Yan Wang declares that there are no conflicts of interest regarding the publication of this paper.

References

[1] Amazon Team. Amazon simple storage service (Amazon S3), Amazon, 2013. http://aws.amazon.com/s3/.

[2] S. Ghemawat, H. Gobioff, and S. Leung, "The google file system," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pp. 29–43, October 2003.

[3] D. Borthakur, "The Hadoop distributed file system: architecture and design," *Hadoop Project Website*, vol. 11, no. 11, p. 10, 2007.

[4] S. K. Sahi and V. S. Dhaka, "Study on predicting for workload of cloud service using," in *Proceedings of the Artificail Neural Network , 2015, International Conference on Computing for Sustainable Global Development*, pp. 331–335, New Delhi, India, 2015.

[5] G. Bao, "Researching on the Placement of Data Replicas in the System of HDFS Cloud Storage Cluster," in *Proceeding of the 2013 Chinese Intelligent Automation Conf*, pp. 259–269, Yangzhou, China, June 2013.

[6] B.-Y. Ooi, H.-Y. Chan, and Y.-N. Cheah, "Dynamic service placement and replication framework to enhance service availability using team formation algorithm," *The Journal of Systems and Software*, vol. 85, no. 9, pp. 2048–2062, 2012.

[7] Z. Ghilavizadeh, S. J. Mirabedini, and A. Harounabadi, "A new fuzzy optimal data replication method for data grid," *Management Science Letters*, vol. 3, no. 3, pp. 927–936, 2013.

[8] F. Xiong and Y. B. Wang, "Qos-aware Replica Placement in Data Grids," *Journal of Systems Engineering and Electronics*, vol. 36, no. 4, pp. 784–788, 2014.

[9] R. Kingsy Grace and R. Manimegalai, "Dynamic replica placement and selection strategies in data grids - A comprehensive survey," *Journal of Parallel and Distributed Computing*, vol. 74, no. 2, pp. 2099–2108, 2014.

[10] G. Aupy, A. Benoit, M. Journault, and Y. Robert, "Power-aware replica placement in tree networks with multiple servers per client," *Sustainable Computing: Informatics and Systems*, vol. 5, article no. 112, pp. 41–53, 2015.

[11] X. G. Wang, J. Cao, and Y. Xiang, "Dynamic cloud service selection using an adaptive learning mechanism in multi-cloud computing," *Journal of Systems and Software*, vol. 100, pp. 195–210, 2015.

[12] W. L. Ding and Y. B. Han, "A Replica Placement Method during Data Stream Processing," *J. of Electronics Information Technology*, vol. 36, no. 7, pp. 1755–1761, July 2014.

[13] T.-W. Um, H. W. Lee, W. Ryu, and J. K. Choi, "Dynamic resource allocation and scheduling for cloud-based virtual content delivery networks," *ETRI Journal*, vol. 36, no. 2, pp. 197–205, 2014.

[14] N. Mansouri, "Adaptive data replication strategy in cloud computing for performance improvement," *Frontiers of Computer Science*, vol. 10, no. 5, pp. 925–935, 2016.

[15] N. Mansouri, "QDR: a QoS-aware data replication algorithm for Data Grids considering security factors," *Cluster Computing*, vol. 19, no. 3, pp. 1071–1087, 2016.

[16] U. Tos, R. Mokadem, A. Hameurlain, T. Ayav, and S. Bora, "Dynamic replication strategies in data grid systems: a survey," *Journal of Supercomputing*, vol. 71, no. 11, pp. 4116–4140, 2015.

[17] M. Ajitabh and T. Ritu, "A replica distribution based fault tolerance management for cloud computing," *J. of Computer Science Information Technology*, vol. 5, no. 5, pp. 6880–6888, 2014.

[18] G. Zhang, L. Cong, L. Zhao, K. Yang, and H. Zhang, "Competitive resource sharing based on game theory in cooperative relay networks," *ETRI Journal*, vol. 31, no. 1, pp. 89–91, 2009.

[19] E. Akbari, F. Cung, H. Patel, A. Razaque, and H. N. Dalal, "Incorporation of weighted linear prediction technique and M/M/1 Queuing Theory for improving energy efficiency of Cloud computing datacenters," in *Proceedings of the IEEE Long Island Systems, Applications and Technology Conference, LISAT 2016*, New York, NY, USA.

[20] K. I. Kim, S. Y. Bae, D. C. Lee, C. S. Cho, H. J. Lee, and K. C. Lee, "Cloud-Based gaming service platform supporting multiple devices," *ETRI Journal*, vol. 35, no. 6, pp. 960–968, 2013.

[21] H. Luo, J. Liu, and X. Liu, "A two-stage service replica strategy for business process efficiency optimization in community cloud," *Chinese Journal of Electronics*, vol. 26, no. 1, pp. 80–87, 2017.

[22] H. Zhang, Z. H. Liu, and B. Lin, "QoS-aware replica placement algorithm in cloud storage environment," *J. of Chinese Computer Systems*, vol. 37, no. 9, pp. 1915–1919, 2016.