Replica Placement in Cloud Storage based on Minimal Blocking Probability

Meng Xue[1]
The PLA Information Engineering University
No.62, Science Road, Gaoxin District, Zhengzhou, 450000, China
E-mail: xue9122@126.com

Jianjing Shen
The PLA Information Engineering University
No.62, Science Road, Gaoxin District, Zhengzhou,450000, China
E-mail: shenjj@vip.sina.com

Xiaofeng Guo
The PLA Information Engineering University
No.62, Science Road, Gaoxin District, Zhengzhou,450000, China
E-mail: guoxiaofeng@msn.com

Data replica placement is an important issue in the cloud storage system, which includes two key points: how to determine the replica factor and how to select optimal *Datanode* to store replica. With the provision of file store service and file access service for large scale users, the *Datanode* in cloud storage system is regarded as services site. According to the basic principle of queuing theory, the replica placement algorithm based on minimal blocking probability (BPRA) is proposed from the view of resource competition as a dynamic replica placement algorithm. The BPRA algorithm calculates the minimal number of replica according to the file available requirement and dynamically adjusting replica factor according to the file access frequency. In addition, the blocking probability is computed by each *Datanode* respectively and reported to the control node (namely *Namenode*). The simulation results conclusively demonstrate that the replica placement algorithm based on minimal blocking probability can effectively achieve the load balance of cloud storage system, reduce access skew and resource competition, decrease 20%~60% file access latency versus Hadoop Distributed File System (HDFS) default replica placement algorithm.

CENet2015
12-13 September 2015
Shanghai, China

[1]Speaker
Corresponding Author

1. Introduction

The cloud storage combines large scale distributed storage technology and virtualization technology [1], which coordinates large number of various storage devices on the network through software and provides file store service and file access service for cloud users. Those devices include control node and data node, respectively *Namenode* and *Datanode*. The node failure or the network breakdown will cause data unavailable, hence the multi-replica technology is one of the key technologies that guarantee data reliability, data availability and service quality, which have been used in Google file System, Ceph file system and Hadoop distribute file system etc. Multiple replicas are deployed in cloud storage generally. Data and its copies are stored in different data nodes to make sure the file service can be accessed normally by using the replica in the other data node in case of malfunction of one data node.

Replica placement is a key issue in multi-replica technology and the replica placement algorithm has been a hot topic for research in cloud storage. There are two problems in the replica placement: how to determine the replica factor and how to select optimal *Datanode* to store replica. To address these issues, we propose a dynamic replica placement algorithm BPRA based on minimal blocking probability which computes the blocking probability of *Datanode* according to its bandwidth and load.

The proposed BPRA algorithm places replicas on the *Datanode* with minimal blocking probability, which is a dynamic replica placement algorithm that selects optimal *Datanode* to store replica dynamically according to the current access patterns, available bandwidth and data node busy degree, and then the files can be placed evenly within the cloud storage system avoiding the excessive competition of resource. What's more, the blocking probability is calculated by each *Datanode* itself without information from other *Datanode* and the distributed computation makes BPRA quite efficient. It can be proved that BPRA significantly reduce access skew and blocking probability of data node, shorten 20%~60% file access latency when compared with HDFS default replica placement algorithm, and improve the service efficiency of cloud storage system and the user's experience.

2. Related Work

In recent years, the study of replica placement algorithm can be mainly divided into two types, static replica placement algorithm and dynamic replica placement algorithm.

Static replica placement algorithm create replica and select data node when cloud storage system is initialized, whose advantage is easy to be realized and deployed. A multi-objective optimized replication management strategy considered five optimization objects: mean file unavailability (MFU), mean service time (MST), load variance (LV), energy consumption (EC) and mean latency (ML) and optimized the total objective function through an improved artificial immune algorithm [2]. What's more, an intelligent multiple replica placement scheme based on genetic algorithm (GA) was proposed, which assigned the user access request to the specific data replica optimally by biogeography-based optimization (BBO) algorithm [3]. In addition, the particle swarm algorithm and ant the colony algorithm are also introduced to solve the problem of replica placement, which has been widely used in classic multi-objective optimization; however, those methods cannot overcome the weakness of static replica placement algorithm that static replica placement algorithm is unchangeable according to the current access patterns, available bandwidth and data node busy degree.

While dynamic replica algorithm can select optimal *Datanode* to store replica dynamically with those information. The dynamic cost effective replication scheme (CDRM) was proposed for large scale cloud system which built a model to describe the relationship between availability and replication factor, but it cannot echo the changes in network timely[4].The pseudo-random replica placement algorithm (CRUSH) distributed data according to storage capacity and device bandwidth resources; however, the pseudo-random may cause high load of some devices[5]. Latest Access Largest Weight (LALW) was proposed for dynamic data replication, which associated a different weight to each historical data access record and found hot file according to access frequencies. Then the hot file was replicated to suitable sites to realize system load balance[6].

In addition, some new methods have been also introduced to solve the problem of replica placement. The Vickrey-Clarke-Groves(VCG) mechanism achieved replica placement dominant strategy equilibrium thought game theoretic methods, which was discussed in P2P storage system rather in the cloud storage system[7]. A minimum-cost based data replication strategy was proposed for balancing the storage cost and the transfer cost from the perspective of cost-control frequently seen in the areas of marketing management; nevertheless, it failed to concern how files were stored inside of the data centre[8].

Based on the literature survey, we propose a dynamic replica placement algorithm according to the basic principle of queuing theory from the view of resource competition.

3. System Model

First of all, three hypotheses are proposed. Hypothesis I: all the nodes in cloud storage are secure and the security issues are not considered. Hypothesis II: all the data are “write-once, read many”, no data consistency mechanism is discussed. Hypothesis III: the file partition is not considered and each piece of the file is treated as a stand-alone file in this paper.

The cloud system is composed of *m* independent homogeneous or heterogeneous data nodes and *n* different files are stored in those data nodes. Fig. 1 shows the sample model of replica placement in cloud storage. In detail, the square means different files and the same color shows they are the replicas of one file. What’s more, the shape of data nodes represents that they are homogeneous or heterogeneous.

Then, the definition of file and data node is introduced.

Definition I: *n* different files f_i i=1, 2,…,n, which is represented by four tuples $f_i=(p_i,s_i,r_i,\tau_i)$. p_i,s_i,r_i and τ_i refer to the access popularity, size, replica number, and access latency requirement of the file f_i respectively.

Definition II: *m* independent data node D_j j=1, 2, …,m, which is represented by five tuples $D_j=(\lambda_j,\tau_j,fa_j,bw_j,ca_j)$. $\lambda_j,\tau_j,fa_j,bw_j$ and ca_j refer to access time, average service time, failure probability, bandwidth, and node capacity. In addition, fa_j is initialized [0.2 1] when the cloud system is started, then the failure probability can be updated according to the actual operation of each data node.

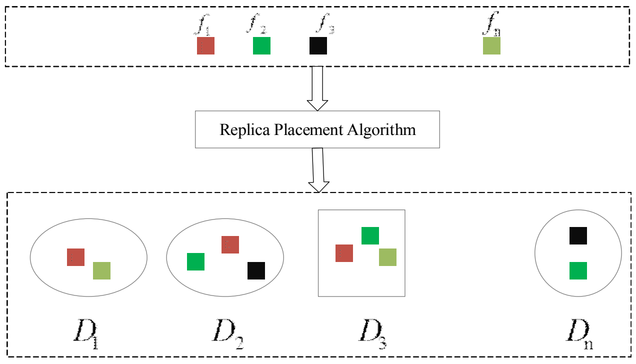


Figure 1: Sample model of replica placement in cloud storage

4 Replica Placement Algorithm

4.1 Determine the Replica Factor

In order to enable the user’s file available, a certain number of file copies are maintained in the cloud storage system, namely replica factor; however, the availability approaches 1 closely and do not increase any more when too many replicas exist. What’s worse that too many replicas will cause storage redundancy and increase the cost of data management. As a result, how to determine the replica factor is one of the challenges in terms of replica placement algorithm.

Symbol function $\Phi(i,j)$ is defined that the value is evaluated 4.1 when the file f_i exists on data node D_j; otherwise the value is evaluated 0. Consequently, the number of replica of file f_i in the cloud storage is $\sum_{j=1}^{m} \Phi(i,j)$.

$$\Phi(i,j)=\begin{cases}1 & if\ f_i \in D_j \\ 0 & else\end{cases} \tag{4.1}$$

$$1-\prod_{j=1}^{m} \Phi(i,j)\times fa_j > A^i_{expect} \tag{4.2}$$

Datanode in the cloud storage are independent and then Equation (4.2) is proposed to satisfy the user's availability requirement A^i_{expect}. *Namenode* calculates the minimal replica factor *min(r$_i$)* with Equation (4.2).

What's more, *Namenode* collects file f_i access frequency $Visit_i$ at recent time T and the proper threshold TH=[$Visit_a$ $Visit_b$] is defined. If $Visit_i > Visit_b$, the file f_i is identified as Hot data and one more replica of the file is created to balance the load of the storage system; if $Visit_a < Visit_i < Visit_b$, the *NameNode* keep the replica factor invariant; if $Visit_i < Visit_a$, the *NameNode* will delete one replica and hold the minimal replica factor *min(r$_i$)* of the file f_i at the same time.

4.2 BPRA Algorithm

In order to reduce the file access skew, blocking probability is used as the criterion to place replicas.

Datanode has no ability of responding new file request when it is processing c_j file requests, where c_j is the largest number of request and D_j can process at the same time. The *Datanode* will be blocked when more requests arrive. According to the queuing theory [9], the blocking probability of *Datanode* is defined in Equation (4.3), where λ_j and τ_j refer to the requested arrival rate and the mean service time of D_j respectively.

$$BP_j=\frac{(\lambda_j\tau_j)^{c_j}}{c_j!}\left[\sum_{k=0}^{c_j}\frac{(\lambda_j\tau_j)^k}{k!}\right]^{-1} \tag{4.3}$$

Requesting the file stored on the *Datanode*, it will be restricted to the bandwidth bw_j. The bandwidth consumption cannot exceed the available bandwidth when there are multiple file request arrival data nodes. This constraint can be expressed as Equation (4.4). Bandwidth bw_j and the file list of D_j are known, c_j will be calculated based on Equation (4.4).

$$BP_j \geq \sum_{i=1}^{c_j}\frac{s_i}{\tau_i} \tag{4.4}$$

Assume the access pattern submits be the poisson distribution and the expectation of the poisson distribution is λ. Consequently, the request arrival rate λ_j and the mean service time τ_j can be calculated by using Equation (4.5) and Equation (4.6) respectively.

$$BP_j=\Sigma_{i=1}^{m_j}\frac{p_i}{r_i}\lambda \tag{4.5}$$

$$\tau_j=\frac{1}{m_j}\Sigma_{i=1}^{m_j}\tau_i \tag{4.6}$$

The blocking probability can be computed with Equation (4.4) when the data node is not blocked. In contrast, the blocking probability will be set to 1. What's more, the constraint in Equation (4.7) should be satisfied when the file f_i is placed on data node D_j : f_i does not exist on the data node D_j; the sum of files' size assigned to the data node D_j must less than its capacity.

$$\begin{cases} \Phi(i,j)=0 \\ \sum_{i=1}^{n} \Phi(i,j)\times s_i + s_i \le ca_j \end{cases} \tag{4.7}$$

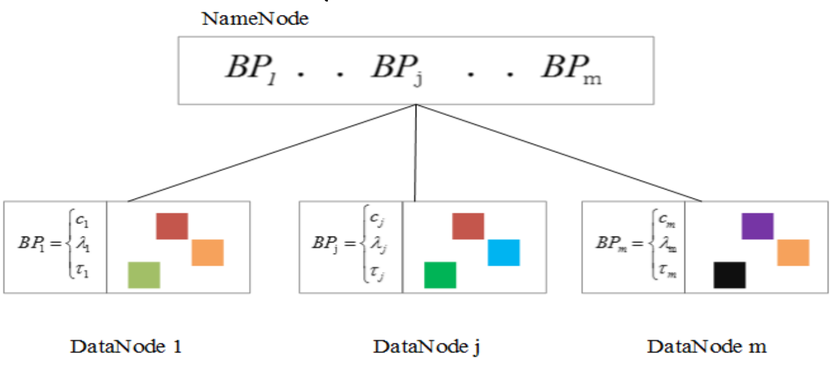

NameNode
BP_1 . . BP_j . . BP_m
$BP_1 = \{c_1, \lambda_1, \tau_1\}$
$BP_j = \{c_j, \lambda_j, \tau_j\}$
$BP_m = \{c_m, \lambda_m, \tau_m\}$
DataNode 1
DataNode j
DataNode m

Figure 2: Replica placement with minimal blocking probability

The model of Replica Placement with minimal blocking probability is shown in figure 2. The blocking probability is computed by *Datanode* itself according to its parameters (c_j, λ_j and τ_j). Then the result will be report to the *Namenode*. The metadata of file in the cloud storage system are maintained in the *Namenode*, which includes basic information and location information. The basic information is consisted of file *ID*, file name and file size, etc. The location information is consisted of file replica factor and the list of *Datanodes* stores the file replication. Algorithm 1 is the pseudo-code description of replica placement algorithm based on minimal blocking probability (short for BPRA).

```
    Algorithm 1: the pseudo-code description of BPRA
    Input  :FileList, DatanodeList,
    Output : LayoutList
    for file 1:FileNum
    Calculate min (r), according to availability
    if factor < min(r)
      num = min(r)-fator
    for replica 1:num
    Select Datanode according to the minimal BP from Namenode
    Put replica on the selected Datanode, Change parameters of Datanode
    Refresh BP, Report to the Namenode
    end for
    end if
    Get $visit_i$ from Namenode
    if $visit_i > Visit_b$
       factor = factor + 1
    then Add one more replica with minimal BP
    else if $visit_i < Visit_a$
       if factor > min(r)
    factor = factor – 1
    then  Delete one replica with maximal BP
       end if
    end if
end for
```

5. Experimental Validation

In order to evaluate the effectiveness of the proposed BPRA algorithm, we've carried out simulation experiments based on the CloudSim and used matlab for data processing[10]. CloudSim is an open-source cloud computing simulation tool developed by Rajkumar from the University of Melbourne in 2009, which has been the most popular simulation tool in cloud computing. In order to prove the validity of the BPRA algorithm proposed in this paper, we compare BPRA algorithm with HDFS default replica placement algorithm.

There are *m* independent data nodes and *n* different files in the cloud storage system. All the *Datanode* have the capacity of 500G and the bandwidth of 100Mbit/s and more details can be found in table 1, the default configure of simulation.

Parameters	Value
RAM	2GB
MIPS	10000mips
Storage Size	500GB
Inter-cluster connectivity bandwidth	1-10Mbit/s
Intra-cluster connectivity bandwidth	100Mbit/s
file size	64M

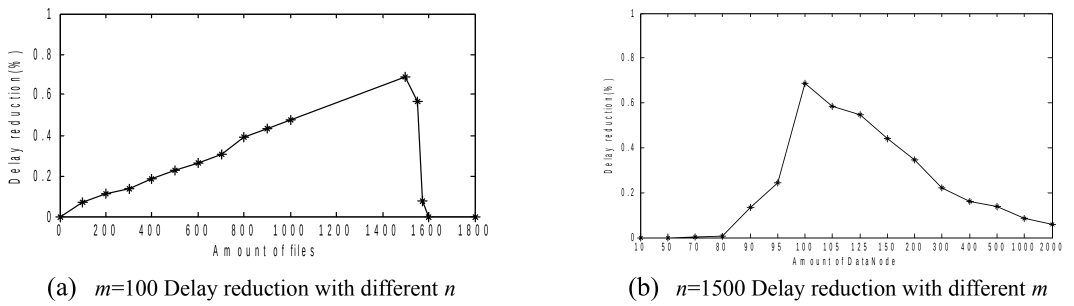
Table 1: Default configuration of simulation

5.1 Average Latency

The access latency is an important evaluation criterion for replica placement algorithm. Fig. (3) compares the delay reduction of BPRA with HDFS default replica placement algorithm with different *m* and *n*. The file access latency will be averaged by access 100 times for each file to guarantee the reliability of the experiment data.

When *m* equals to 100, the delay reduction with various *n* is shown in figure 3(a). The advantage of BPMA over HDFS default placement algorithm is more and more obvious when n takes 600, 800, 1000 and 1500. The BPRA can decrease about 68% file access latency when the amount of files equals to 1500; however, when *n* is less than 200, files in the storage system are relatively sparse and all the *Datanode* have enough ability to process the file request for little competition. On the other hand, files in the storage system are relatively dense when *n* is more than 1600 and all the data node will be blocked under the poisson access pattern.

figure 3(b) shows the delay reduction with various *m* when *n* equals 1500 and the result is similar to figure 3(a). As discussed in figure 3(a), it can't give a full play to the advantages of BPRA algorithm when the files in the storage are excessively dense or sparse.



(a) *m*=100 Delay reduction with different *n* (b) *n*=1500 Delay reduction with different *m*

Figure 3: Delay reduction of BPRA with different m and n

5.2 Load Balance

We performed two experiments to verify the load balance ability of the BPRA algorithm where the *m* and *n* was set as 100 and 300 respectively. The load curve and block probability curve of *Datanode* were drawn in figure 4 and expressly described the load balance ability.

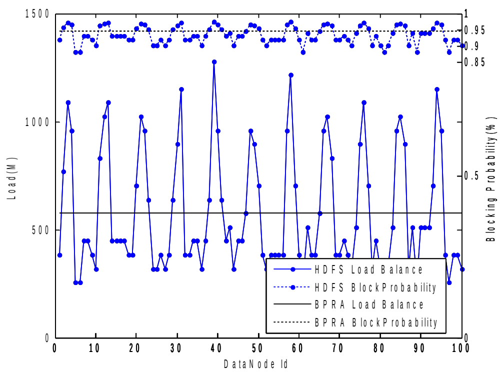
The first experiment used the default configure in Table 1 with the result shown in figure 4(a). As to the result of HDFS default replica placement algorithm, we can see that the load of *Datanode* display large difference and the blocking probability fluctuate in a definitive range. At the same time, the result of BPRA algorithm is quite well for both load balance curve and block probability curve are flattering.

The second experiment divide the *Datanode* into three groups by bandwidth which is shown in Table 2 and the other parameters are same as the first experiment. The result is shown in figure 4(b), we can see that the BPRA algorithm can achieve load balance in each group while the HDFS default replica placement algorithm cannot achieve the load balance.

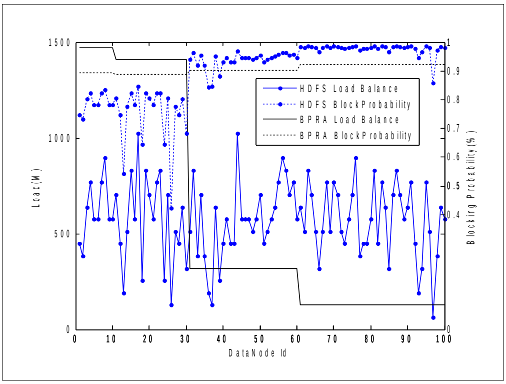
In summary, figure 4 shows that the load balance ability of the BPRA algorithm is much better at selecting the optimal *Datanode* with the minimal blocking probability.

Datanode Id	Bandwidth(Mbit/s)
1-30	500
31-60	100
61-100	50

Table 2: Group Result



(a) Load and Block Probability of Experiment 1



(b) Load and Block Probability of Experiment 2

Figure 4 : Load andblock probability of *Datanode*

6. Conclusion

The main contributions of this paper are summarized as follows: 1) calculate the minimal number of replica according to the requirement of file availability and adjusting replica factor according to the file access frequency dynamically; 2) propose an effective dynamic replica placement algorithm BPRA based on minimal blocking probability from the view of resource competition which is minutely described in Figure2 and Algorithm 1.

The further improvement of the BPRA algorithm will be discussed from two aspects, firstly, the partition of data nodes according to performance; secondly, classification of the users according to their QoS configuration. Additionally, the improvement can enable the cloud storage system to be capable of providing diverse data services to meet different types of the user's service requirements.

References

[1] J. J. Wu, L. D. Ping, X. P. Ge, Y. Wang, J. Q. Fu. *Cloud storage as the infrastructure of cloud computing*[C]. International Conference on Intelligent Computing and Cognitive Informatics (ICICCI). Piscataway, NJ: IEEE: 380-383(2010)

[2]L. S. Qin, Y. L. Zhao, W. Chen. *MORM: A Multi-objective Optimized Replication Management strategy for cloud storage cluster*[J]. Journal of Systems Architecture. 60(2): 234-244(2014)

[3]B. Zhang, X. W. Wang, Huang Min. *A data replica placement scheme for cloud storage under healthcare IoT environment*[C]. 11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD). Piscataway, NJ:IEEE: 542-547(2014)

[4]Q. S. Wei, B. Veeravalli, B. Z. Gong, L. F. Zeng, D. Feng. *CDRM: A Cost-Effective Dynamic Replication Management Scheme for Cloud Storage Cluster*[C]. Proceedings of the 2010 IEEE International Conference on Cluster Computing. Piscataway, NJ:IEEE Computer Society: 188-196(2010)

[5]S. A. Weil, S. A. Brandt, E. L. Miller. *CRUSH: Controlled, scalable, decentralized placement of replicated data*[C]. Proceedings of the 2006 ACM/IEEE conference on Supercomputing. Tampa: IEEE: 122-138(2006)

[6]R. S. Chang, H. P. Chang. *A dynamic data replication strategy using access-weights in data grids*[J]. The Journal of Supercomputing. 45(3): 277-295(2008)

[7]W. Song, Y. L. Zhao. *Application of VCG mechanism in replica placement of P2P storage system*[J]. Journal of Computer Applications. pp. 860-864(2010)(In Chinese)

[8]X. G. Wu. *Minimum-cost Based Data Replication Strategy in Cloud Computing Environment*[J]. Journal of Computer Science. 41(10): 154-159(2014)(In Chinese)

[9]D. Gross. *Fundamentals of Queueing Theory*[M]. 4th ed. Hoboken:Wiley Interscience. pp. 254-257(2008)

[10] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, R. Buyya. *CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms*[J]. Software: Practice and Experience. 41(1): 23-50(2011)