Data Replica Placement in Cloud Storage System

ZHANG Tao
Nanjing Branch in China Electric Power Research Institute, Nanjing, 211106, China
zhangtao@epri.sgcc.com.cn

Abstract

As a main applicational way of cloud computing, cloud storage system can provide highly available data services. And data replicas replacement strategy has played a significant role in a cloud storage system.

Data replicas placement strategy has been discussed in this paper, and a heuristic method of data replica placement is designed. Related simulations demonstrate that: The algorithm we proposed has a better performance whether in a storage sensitive environment or not.

Keywords: Cloud Computing; Data Replication; Replica Placement; Communication Cost

1. Introduction

As a brand new model of network computing, cloud computing, the crucial technology that will lead the industry revolution in the coming twenty years, has become a popular information technology in recent years, and has been widely accepted as the third technology revolution following personal computers and the Internet[1,2].

Cloud computing aims to provide service to users on demand by unitedly managing and dispatching the massive, highly virtualized computing resources that in the network. People can access the service conveniently like using public resources such as water and electricity. And it has achieved the goal of distributing resources according to one's needs in a real sense.

Cloud storage, as a typical application way of cloud computing, aims to establish an information-sharing storage environment that across the Internet so as to provide users with data storage service at any given time and place, and is gaining popularity nowadays.

To increase the data availability of cloud storage system, data replication has been widely used[3]. Data replication refers to duplicating multiple copies for the data, and these copies are stored in the cloud storage system on different data nodes. When users access the data from one certain node, it will access the replica of the data that is on the nearest adjacent node, and therefore it can improve the data access speed.

In addition, data replication can also reduce and balance cloud storage server load, improve access latency, enhance fault tolerance of cloud storage system, and ensure the overall reliability of cloud storage system.

Although a lot of research were conducted in distributed file systems and network storage in the past, it is difficult to directly apply these findings to the cloud storage system:

1) In a cloud storage system, to further improve load-balancing of the cloud storage server and speed of accessing the database, Data Striping is often used;

2) Compared to traditional network storage, the main purpose of data replication in cloud computing is increasing the data availability, and on that basis we consider the communication cost and data server load when accessing data.

In a cloud storage system, due to the bandwidth and storage space constrains and the spatial distribution of data centers, it, therefore, is an urgent problem how to store data effectively, improve the data availability and ensure rapid access of data source.

So far there have been many achievements in the research of the data replication placement strategy in the cloud storage system. For example, Alexandre etc. [4] proposed the virtual partition method which divided data based on the range of attribute values and store data replications using the technology of chain reunion, and Yan Bingheng etc.[5] proposed the LBS algorithm which is a dispatching algorithm based on scheme multiplex. Converting applications into restricts, it can choose the suitable storage node or dispatching scheme that already exists by analysing the relationships between the restricts. However, all of these algorithms do not consider the storage environment and the problem of data transfer overhead, and therefore are not suitable for using in a cloud storage system. Also, there are some researches that focus on the computer application and transfer time overhead after the deployment of data, such as the strategy of three period data layout proposed by ZHENG Pai etc[6], which can choose the scheme that has the minimum time overhead based on genetic algorithm, and then select integrated optimal solution by assessing the scheme above depending on the two functions of dependent damage degree and load balancing degree, and the data layout strategy based on the

Clustering Matrix proposed by DongYuan etc.[7]. By constructing the dependency matrix and using BEA algorithm to convert, it can partition the Clustering Matrix. Both of the two algorithms are calculated for a specific workflow, while most of the applications in a cloud environment use the data set that is undetermined when deploying data.

There are two key problems need to be solved when we use data replication in a cloud storage system: First, according to the probability of each data center availability, how to maintain minimum replicas to ensure a given availability requirement; Second, how to distribute a given number of data replications to suitable data centers.

The second problem of data replications in a cloud storage system is also discussed in this paper, and we design an algorithm of heuristic data replica placement. In this algorithm, to lower the total communication cost of accessing data, multiple data replicas are stored in suitable data centers in advance. Related simulations demonstrate that: Our algorithm can achieve the goal of better data replica placement on account of the storage capacity of storage nodes. Without considering the node storage capacity, our algorithm can have the effect that even closed to the optimal placement.

2. System Model and Problem

The storage-sensitive data replica placement model of cloud storage system we discussed in this paper is composed of n independent cloud data center refered as $DC = \{dc_1, dc_2, \cdots, dc_n\}$ (storage node). Every storage node is directly connected with some of the others, and any two storage nodes in this sytem is connected directly or indirectly.

As is show in figure 1, a round circle represents a specific storage, and the number in the circle represents the identity of the storage node. What in the bracket represents the space capacity of storage node that is available for storing file replica and is denoted as sci.

The system model of our algorithm is shown in figure1. The undirected connected graph G=(V,E) represents the cloud storage system. We denote the collection of the storage nodes in the system as *V*, and for each storage node $v \in V$.

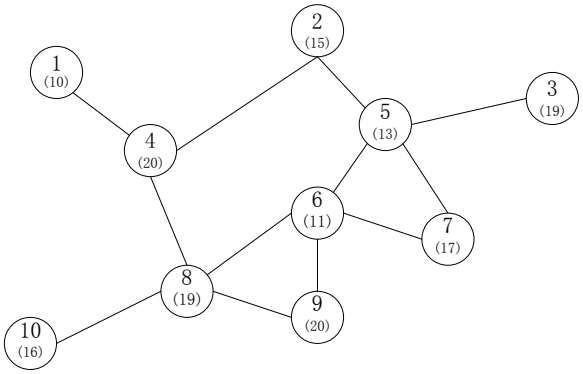

Figure 1. System Model

Among the cloud storage system model G=(V,E) that we show above, set of edges $E = V \times V$ refers to the collection of the communication links that exist between the nodes, and each edge belongs to $E = \{e \mid e = (u, v)\}$. And $c(u, v)$ refers to the communication cost of a certain path between the node *u* and *v*. It is the total communication cost of all the edges in the path. When the node *u* access the data, it will always find the replica of the data along the path that starts with the node *u* in the figure *G*, and the communication cost is the sum of the communication cost of all the edges that in the path between the node that stores the data and node *u*.

Hence, the replica placement problem in this paper can be formally defined as follows:

Definition 1 (**Replica Placement Problem**) Given a undirected connected graph G, data request number f_{ij}, data set *DF*, and storage capacity of each node *v*, and communication cost $c(u, v)$ of each edge (u, v). The replica placement problem is how to find an optimal node set *S* for data replicas ($S \subseteq V$), such that replica set *S* satisfies the storage capacity constraint of every node and obtains the minimal communication cost.

3. Placement Algorithm

In the data replica placement model of cloud storage system, the *m* data files that need to place are denoted as $DF = \{df_1, df_2, \cdots, df_m\}$, and the size of the data file is denoted as df_i. To lower the communication cost, the system need to replicate the data. And in this algorithm, an extremely important data structure will be shown to help choose the data files at a certain sequence and create replications for them. The accessed frequency matrix is denoted as $F = \begin{pmatrix} f_{11} & \cdots & f_{1m} \\ \cdots & \cdots & \cdots \\ f_{n1} & \cdots & f_{nm} \end{pmatrix}$, and f_{ij} refers to the frequency of all the users that belong to data center dc_i access the data file df_j. In our algorithm, we use matrix

$C = \begin{pmatrix} c_{11} & \cdots & c_{1m} \\ \cdots & \cdots & \cdots \\ c_{n1} & \cdots & c_{nm} \end{pmatrix}$ to denote the communication cost between two data centers, and c_{ij} refers to the communication cost of transfering files from dc_i to dc_j.

The elements in set A and B refer respectively to the total frequency and variance of accessing each data file. Let say the set of files that in descending order by priority is *DF'* and the set of data centers that don't have any other space available is *DC'*, which is the set of data centers that don't have the enough storage space for the file which has the minimum size in the cloud storage system.

Our algorithm makes use of the definition in general data replica placement algorithm. The system first get the data file which has the highest priority, and then create replication for it at local optimum storage node, and refresh the set *A*, *B*, *DF'*, *DC'* simultaneously. Repeat this procedure above until the storage space of all the storage nodes is run out.

The pseudo-code of the algorithm is as follows:

Input: *DC*: set of data centers $dc_1, dc_2, \cdots, dc_n$

DF: set of data files $df_1, df_2, \cdots, df_m$

Begin

01. $DF' = \varnothing;\ DC' = \varnothing;\ S_1 = S_2 = \cdots = S_m = \varnothing;$

02.**for** (**each** $df_j \in DF$)

03. calculate A and B;

04. **end for**

05. DF' = sort (DF, A, B);

06.**while** ($DC - DC' = \varnothing$)

07. **for** (**each** df_j with highest frequency in *DF'*)

08. $c_m = \infty;$

09. **for** (**each** $dc_i \notin s_j$ && $dc_i \notin DC'$)

10. *temp*=0;

11. $c = \sum_{j=1}^{n} f_{ij} \square \min_{k \in \{dc_i\} \cup S_j} c_{ik}$;

12. *if* ($c < c_m$)

13. { $c_m = c$; *temp*=*i*;}

14. **end if**

15. **end for**

16. $S_j = \{dc_{temp}\} \bigcup S_j;\ sc_{temp} = sc_{temp} - ds_j;$

17. **if** ($sc_{temp} < \min(\bigcup_{j=1}^{m} ds_j)$)

18. $DC' = DC' \bigcup \{dc_{temp}\};$

19. **end if**

20. update *DF'*;

21. **end for**;

22. **end while**;

At the end of the algorithm, we calculate the total communication cost.

4. Performance Evaluation

To evaluate the performance of data replica placement algorithm of cloud storage system, we implement a simulation tool and conduct some experiments based on a randomly generated cloud storage model.

In the cloud storage model, there are 100 nodes in the graph, and the amount of other parameters is randomly generated. To evaluate the performance of the replica placement algorithm, we also implement the random placement algorithm and optimal placement algorithm by the method of exhaustion.

Fig.2 shows the experiment results with storage constraint. When the number of data files increases, the total communication costs of the both two algorithms arise respectively. This is because that there was more storage space available for storing data replicas in storage nodes with a less number of data files, and the communication costs of the entire system fell accordingly. While the increase of the number of data files would result in a decreased number of data files replicas, so the total communication costs of the entire system rised. But the total communication costs in the system that using storage sensitive heuristic data replica placement algorithm was always lower than that using storage sensitive random placement algorithm, and therefore had a better performance.

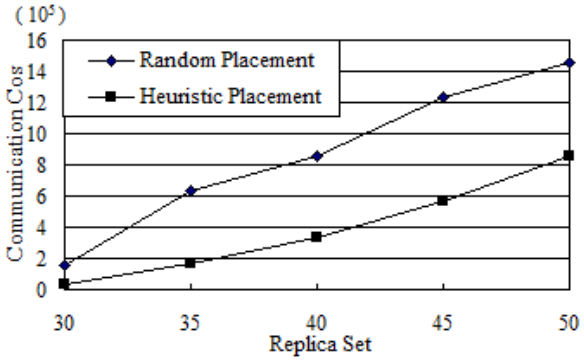

Fig.2. heuristic data replica placement experiment with storage constraint

Fig.3 shows the experiment results without storage constraint. With the increasing number of data replicas, all of the three total communication costs fell. This is because that when the number of data file replicas rised, there would be fewer long-distance access of data, thus reduced the total communication cost. When the number of data set increase, the total communication costs of all the three algorithms rised accordingly. And the communication costs of general heuristic placement algorithm and optimal placement

algorithm are pretty close when there are three types of data set and much fewer data replicas. Obviously the two algorithms are superior to regular random algorithm. The total communication cost of general heuristic placement algorithm is equal to that of optimal placement algorithm especially when data replica number is 1.

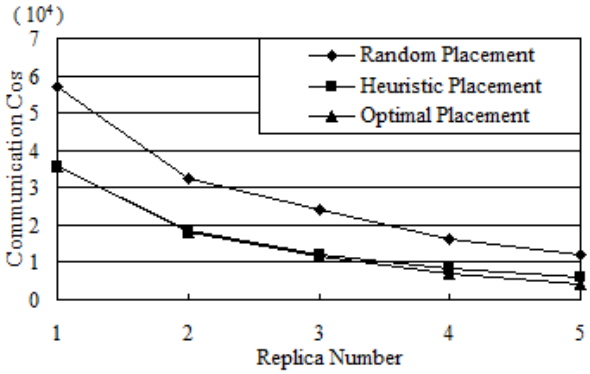

Fig.3 heuristic data replica placement experiment without storage constraint

.

5. Conclusion

In this paper, we have discussed the problem of data replica placement in cloud storage system and designed a replica placement algorithm, and it is based on the existing data of communication costs between data centers and frequency of accessing data, and can calculate the communication cost of the entire system when the number of data replicas reaches its minimum level. The experiments show that the heuristic algorithms we proposed could effectively lower the total communication cost, and are obviously superior to random placement algorithm in the same experimental environment.

References

[1] M. V. Luis, R.M. Luis, etc. “A Break in the Clouds: Towards a Cloud Definition”. *ACM SIGCOMM Computer Communication Review*, 2009,39(1):50-55.

[2] Ian Foster, Yong Zhao, et al. “Cloud Computing and Grid Computing 360-Degree Compared”. *Proceedings of the Grid Computing Environments Workshop*, pages 1-10, 2008.

[3] The Apache Software Foundation. Hadoop[EB/OL]. http://hadoop. apache.org/core/, 2009.

[4] ALEXANDRE, CAMILLE, PATRICK VET et al. “Parallel OLAP query processing in database clusters with data replication”. *Distributed and Parallel Databases*, 2009, 25:97-123.

[5] YAN BH, QIAN DP. “A scheduling Algorithm for Load Balance Sensitive Storage”. *Journal of Xi'an Jiaotong University*, 2009,43(10):61-65.

[6] Zheng Pai Cui Li-Zhen et al. “A Data Placement Strategy for Data-Intensive Applications in Cloud”. *Chinese Journal of Computers*, 2010, 33(8):1472-1480.

[7] DONG Y, YUN Y et al, “A data placement strategy in scientific cloud workflows”. *Future Generation Computer Systems*.2010.26:1200-1214.