Exploring Data Placement Policies in Cloud Computing: Algorithms and Approaches

Dr. Anna Müller, Dr. Tomasz Nowak, Dr. Helena Kovács
Department of Physics, University of Heidelberg, Heidelberg, Germany
Institute of Applied Mathematics, University of Warsaw, Warsaw, Poland
Faculty of Engineering, Eötvös Loránd University, Budapest, Hungary

ABSTRACT

Cloud computing platform can provide infinite storage capacity, computing ability as well as information services; it now has become the popular new application platform for both individuals and enterprises. The storage capacity of a data center is limited. Therefore, how to place data slices in appropriate data center proves to be an important factor influencing the platform ability. The data placement strategy we design in this paper takes the cooperation costs among data slices into account. It lowers the distributed transaction costs as much as possible, especially the cost differences among different distributed transactions. At the same time, this strategy also cares about the global load balance problem in data center. It is developed on the basis of genetic algorithm and ensures that the strategy can quickly converge to efficient data placement solutions. In this paper, survey on data placement strategy named hierarchical structure data placement strategy, it combines the semi-definite programming algorithm with the dynamic interval mapping algorithm and also about several other algorithms and data placement approaches.

Keywords: *Cloud, Data placement Algorithm, Data Storage, Cloud services.*

I. INTRODUCTION

Cloud computing is a typical network computing mode, which emphasizes the scalability and availability of running large-scale application in virtual computing environment [1]. There are various types of data, including common files, large binary files such as virtual machine image file, formatted data like XML, and relational data in database. Thus, a distributed storage service of cloud computing has to take large-scale storage mechanism for various data types into account, as well as the performance, reliability, security, and simplicity of data operation.[2-4].With the development of Computerized Business Application, the amount of data is increasing exponentially. Cloud computing provides high performance computing resources and mass storage resources for massive data processing. In distributed cloud computing systems, data intensive computing can lead to data scheduling between data centers. Reasonable data placement can reduce data scheduling between the data centers effectively, and improve the data acquisition efficiency of users. The network storage system under cloud computing environment consists of thousands, and even ten thousands of storage devices. Different systems have different underlying devices, for example, the storage device set could be chunk device disk for SAN and GFPS, or OSD (object storage device) for object storage systems Lustre and ActiveScale, or PC for PVFS and P2P [5].

II. LITERATURE REVIEW

A data placement strategy mainly solves the problem of selecting storage device for data storage. An effective mechanism shall be adopted to establish the mapping relationship between data sets and storage device sets. Then, data sets generated by applications in the storage system are placed into different storage device sets. Meanwhile, certain particular goals need to be met, and different data placement strategies are designed for different purposes. The strategy of placing several replications of the data into different devices is mainly for the purpose of fault tolerance and data reliability improvement. Distributing data equally could realize a more balanced I/O load. Fig.1 shows the data set placement strategy, based on this cloud storage algorithm are built to meet an efficient data placement in cloud environment.

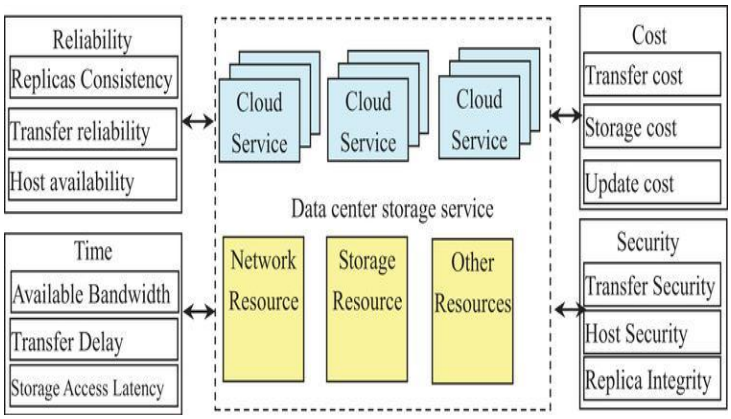

Figure 1: Data set placement strategy.

The data placement strategy under cloud computing environment is designed to meet the following goals[17].

2.1. Fairness

The size of the data stored in each device is proportional to the storage volume of that device [6].

2.2. Self-Adaptability

With the elapse of time, the volume of storage devices is dynamic and varied. Take the case of adding a new device and the case of deleting an existing device for example. When the scale of the storage system changes, a data placement strategy is applied to reorganize the data, making the data distributed to device sets satisfy the fairness criteria over again. Furthermore, it needs to be ensured that the migrated data volume is close to the optimal migration data volume. This would reduce the overheads of data migration. The optimal data volume to migrate is equal to the data volume that is acquired by the added device, or equal to the data volume on the deleted device. The self-adaptability of the data placement strategy is measured by the ratio of its actual migrated data volume to the optimal migration data volume. Therefore, the ratio value of 1.0 represents the optimal condition.[17]

2.3. Redundancy

Getting several replications copied for the data, or enabling the data remain accessible through the use of erasure code when one of the replications is lost. So that fairness can balance the IO loads, self-adaptability can reensure fairness in accordance with storage scale change, and the data size migrated and the IO bandwidth occupied can also be decreased. Finally, the data reliability can be improved.[17]

2.4. Availability

It is crucial that a system could be normally accessed in all cases. Once the system is unavailable, all functions would fail to perform normally. To improve system availability, it is necessary to regularly have the data location adjusted according the availability of storage devices, thus maximizing the system's availability [7].

2.5. Reliability

It indicates whether the system could be normally accessed during a certain period of time. As the large-scale storage system contains thousands of storage devices, the probability of disk failure is rather high. When applying a data placement strategy, indicators of reliability such as data size need to be used in designing parameters of the placement strategy. Thus, a storage system with higher reliability is obtained. [17]

2.6. Space-Time Effectiveness

It means that few time and space is used in calculating data location along with the data placement strategy. When designing the data placement strategy for large-scale network storage system, certain particular goals need to be met depending on different application demands. However, it is impossible to meet all goals at the same time. [17]

Related Work

Some data management systems under the cloud computing environment have already been emerged currently, for example, Google File System [8] and Hadoop [9, 10], both of which have hidden the infrastructure used to store application data from the user. Google File System is mainly used for Web search application, but not for process application under the cloud computing environment. Hadoop is a more commonly distributed file system, which is

used by many companies including Amazon [11] and Facebook. Cumulus project [12] has proposed a cloud architecture of single data center environment.

D. Yuan, Y. Yang, X. Liu and J. Chen, et. al, proposed a data placement strategy which takes two types of data sets into consideration one which is the initial data set before the workflow initiation and second is the intermediate or the generated datasets in between the workflow execution. For a Scientific workflow it is necessary to place the datasets on the same execution site required by the same tasks. But in some cases it may happen that a dataset must be placed in a particular fixed location, then it becomes a challenge to move such large datasets. Yuan proposed a matrix based K-means clustering strategy to place the data on cloud based scientific workflows. K-means binary clustering is applied to pre cluster datasets and the tasks are assigned greedily to execution site such that it contains most of the input datasets. Most interdependent datasets are placed together which reduces data movement. Yuan's clustering technique is sensitive to the selection point in any iteration[16].

U. V. Catalyurek, K. Kaya and B. Ucar, , et. al, proposed a hyper graph partitioning based method for data placement in the cloud. This method tries to solve both the task assignment and the data placement problems using multi constraint hyper graph partitioning. The total file size to be reduced is equal to the cut size and minimizing the cut size is the objective of the problem. It is a two phase scheduling approach, in the first phase the data placement and the task assignment are done and in the second phase the execution takes place with respect to the dependencies. This method tries to reduce the communication cost and tries to place the input, output and intermediate data on execution sites. [16]

Wei Guo, Xinjun Wang, et. al, proposed a data placement strategy based on genetic algorithm on a cloud computing platform. This work tries to lower the distributed transaction costs by reducing the communication costs between two data centers. The data placement strategy is based on the genetic algorithm and also takes care of the load balance problem. It reduces the distributed transaction cost and also balances the load among different data centers. [16]

Qiang Li, Kun Wang, Member, Suwei Wei, Xuefeng Han, Lili Xu, et. al, proposed a data placement strategy based on consistent hashing and clustering algorithm. The hashing algorithm places the data on a specific storage server to achieve better data placement. This provides with the better scalability and better fault tolerance. The clustering in this method is improved by using two methods: Case based reasoning and coordination filtering. This clustering technique has improved clustering accuracy when compared to K-means clustering technique. [16]

III. STUDY OF DATA PLACEMENT ALGORITHM

With the expansion of network scale, the number of data storage devices keeps increasing. The existing data placement algorithm is insufficient to address the system's self-adaptability. Adding new or removing existing devices could lead to a new data placement over again, which will result in an increase of data migration overheads so that the occupation of IO bandwidth is inevitable [13, 14]. Therefore, the data reliability cannot be guaranteed, and the overheads are too large to use a duplicate copy for data reliability assurance [15]. In the case of data placement for file with several replications, different replications of the same file should be placed onto different device sets as many as possible. So that when a certain storage device within a storage device set cannot function properly, the client could get the target file data located on other device sets as usual. Thus, it could improve the availability and reliability of files. Algorithm design should focus to improve the availability and reliability of data. Some of the algorithms used in data placement in cloud storages are as follows. The infrastructure of cloud computing environment is usually composed of hundreds or even thousands of server nodes. Every server node consists of processor, memory. We assume that the data are already prepared before requests arrive and can be accessed any time in the beginning. The initial deployment of data in cloud computing platform is needed.

3.1 An dynamic data placement algorithm

Definition 1：Each node s∈S, in node s q is the number of data blocks , p is the number of data blocks which meet the job requests , the definition of node cover rate $\gamma(s) = p/q$. 60 [16]

Definition 2: If node s and node st have the same data block replica, then node s and node st are data-exchangeable.

The formula (1) is used to compute resource utilization of nodes:

$U=e.\ U_{cpu}+(1-e)\ .\ U_{disk}$ (1)

U_{cpu} stands for utilization of CPU, U_{disk} stands for utilization of the disk, e is a scale factor. Dynamic data placement algorithm is shown in Algorithm . [16]
Input : Map

Output: Data placement strategy
1. Find S_{util} in which resource utilization > U_{up} (<U_{down})with Map
2. while $S_{util} \neq 0$ dϕ
3. ∀ s∈S_{util}, compute U_x
4. If s∈S_{util} and $U_x > U_{up}$ do
5. Find s' data-exchangeable node s_t, where s_t meet s $U_{st} < U_{up}$
6. Turn on node s_t ,transfer p which overloads in node s to s_t
7. Scan Map, find s' which meets $U_{x'} > U_{up}$ and add s' into S_{util}
8. end if
9. If s∈S_{util} and $U_x < U_{down}$ do
10. Find s' data-exchangeable node s_t, where s_t meet $U_{st} > U_{down}$
11. Transfer p which belongs to node s into s_t, and turn off node s_t
12. Scan Map, find s' which meets $U_{x'} < U_{down}$ and add S'into S_{util}
13. end if
14 . end While.

Dynamic data placement algorithm which effectively solve the problem of data placement in cloud computing platform, and this algorithm makes use of large optimization space for node scheduling strategies. The batch scheduling optimization strategy not only achieves energy-saving effect but also successfully solves the constrained problems which are power-restraint problem and time-constrained problem in cloud computing platforms. [16]

3.2 Optimal chunks placement (ocp) algorithm:
Definition: (Chunks Placement): Assume that Φj = Sm k=1 {ϕj,k} is a placement set for chunks of object j, where ϕj,k represents a subset of DCs (i.e, ϕj,k ⊂ D) that containing r replicas of kth chunk. Therefore, for all ϕj,k and Φj , we have |ϕj,k | = r and |Φj | = m × r, respectively.

An object has m chunks and each of them is replicated in r separate DCs. Without any special policy to select DCs for storing the chunks of an object, it might be some replicas of a chunk placed in more reliable DCs (that is DCs with less failure probability) whilst other replicas of another chunks are stored in less reliable ones.. In order to maximize that, we should maximize availability of each chunk, that is $(1 - \prod dl \in \phi j,k\ f(dl))$, which is between 0 and 1. Therefore, the availability of all chunks should be close to each other as much as possible. Ideally, the availability of all chunks should be equal to each other. If it is feasible, ∀k 6= k 0 , fck = fc k 0 , where fck is the failure of kth chunk with r replicas. Since n is a small constant [19] and the number of replicas, r, is 2 or 3 at most [20], it is possible to search all the problem space in order to find the optimal placement of chunks.

Algorithm 3.3: Optimal Chunks Placement
Input : δ,r, m
Output: CA[k][S]
1. S = C(δ,r)
2. Procedure OCP(S,r, m)
3. forall (ϕ ∈ S) do
4 .PCA[0][ϕ] ← 1
5. end
6. for k ← 1 to m do
7. CA[k][S] ← 0
8. forall (ϕ ∈ S) do
9. P ← ∀ϕ 0 ∈ S|∀di , di ∈ ϕ 0 ∧ di ∈/ ϕ
10.CA[k][ϕ] ← A(ϕ) × PCA[k − 1][P]
11. if (k == 1) then

12. PCA[k][P] ← max ϕ 0∈P (CA(ϕ 0))
13 end
14 if (k > 1 and kr < mr) then
15 PCA[k][P] ← OCP(P,r, k − 1)
16 end
17 CA[k][S] ← max(CA[k][ϕ], CA[k][S])
18 end
19 end

We have performed experiments to evaluate this algorithm in order to measure the minimum cost of replication whilst the expected availability in the form of number of nines is satisfied. It maximizes the expected availability of objects under a given budget with the assumption that the objects are split to chunks.[18]

IV. CONCLUSION

The data placement across cloud storage is a big deal in cloud computing. To obtain availability and reliability done through data placement strategy and their algorithms. In this strategy many techniques are followed and still some issues are there. In this paper various algorithms techniques have been surveyed. The data placement strategy techniques have explained. Finally the algorithms are compared by their features. Various parameters needed for the data placement in cloud storage can be included in the future for better result on these techniques.

REFERENCE

1. *Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," Journal of Internet Services and Applications, vol. 1, no. 1, pp. 7–18, 2010.*
2. *J. Ma, "Managing metadata for digital projects," Library Collections, Acquisition and Technical Services, vol. 30, no. 1-2, pp. 3–17, 2006.*
3. *W. Wang, W. Zhang, H. Guo, H. Bubb, and K. Ikeuchi, "A safety-based approaching behavioural model with various driving characteristics," Transportation Research C, vol. 19, no. 6, pp. 1202–1214, 2011*
4. *W. Wang, Vehicle's Man-Machine Interaction Safety and Driver ASSiStance, China Communications Press, Beijing, China, 2012.*
5. *V. T. Tran, G. Antoniu, B. Nicolae, L. Bougé, and O. Tatebe, "Towards a grid file system based on a large-scale BLOB management service," in Grids, P2P and Services Computing, pp. 7–19, 2010*
6. *L. Amsaleg, M. J. Franklin, A. Tomasic, and T. Urhan, "Improving responsiveness for wide-area data access," Data Engineering, vol. 20, pp. 3–11, 1997*
7. *A. Deshpande and Z. Ives, "Adaptive query processing," Foundations and Trends in Databases, vol. 1, no. 1, pp. 1–140, 2007*
8. *S. Ghemawat, H. Gobioff, and S. T. Leung, "The google file system," in Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03), pp. 29–43, ACM, New York, NY, USA, October 2003*
9. *Apache Hadoop, http://hadoop.apache.org/.*
10. *"Hadoop distributed file system," http://hadoop.apache.org/docs/r0.18.0/hdfs_design.pdf.*
11. *Amazon Elastic MapReduce, http://aws.amazon.com/elasticmapreduce/.*
12. *J. Tao, M. Kunze, A. C. Castellanos, L. Wang, D. Kramer, and W. Karl, "Scientific cloud computing: early definition and experience," in Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC'08), pp. 825–830, Dalian, China, September 2008.*
13. *J. Dhok, N. Maheshwari, and V. Varma, "Learning based opportunistic admission control algorithm for MapReduce as a service," in Proceedings of the 3rd India Software Engineering Conference (ISEC'10), pp. 153–160, ACM, Mysore, India, February 2010*
14. *R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, pp. 599–616, 2009.*
    K. H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters," in Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid'07), pp. 541–548, Rio De Janeiro, Brazil, May 2007
15. *Yanwen Xiao , Yaping Li , Jinbao Wang HongGao "An Energy-Efficient Data Placement Algorithm and Node Scheduling Strategies in Cloud Computing Systems"- 2nd International Conference on Advances in Computer Science and Engineering (CSE 2013).*

16. *Qiang Xu*, Zhengquan Xu, Tao Wang "A Data-Placement Strategy Based on Genetic Algorithm in Cloud Computing"*
17. *http://www.cloudbus.org/students/YaserPhDThesis2017.pdf*
18. *M. J. Fischer, X. Su, and Y. Yin, "Assigning tasks for efficiency in hadoop: extended abstract," in Proceedings of the 22nd ACM symposium on Parallelism in algorithms and architectures, ser. SPAA '10. New York, NY, USA: ACM, 2010, pp. 30–39*
19. *A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "Above the clouds: A berkeley view of cloud computing," Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, vol. 28, 2009.*