Enhancing the Scalability in Geographically distributed Cloud storages

A Swathi #1, Nara Prasanthi #2, Shaik Simrin #3, Divi Sandhya #4, Cheethirala Pavani Jayasri #5, Chandholu Sriranga Abhilash #6
#1 Asst.Professor , #2,3,4,5,6 B.Tech., Scholars
Department of Computer Science & Engineering, Qis College Of Engineering And Technology, Ongole, Prakasam(Dt)India.

Abstract:

Cloud computing is becoming a very popular word in industry and is receiving a large amount of attention from the research community. Replica management is one of the most important issues in the cloud, which can offer fast data access time, high data availability and reliability. By keeping all replicas active, the replicas may enhance system task successful execution rate if the replicas and requests are reasonably distributed. However, appropriate replica placement in a large-scale, dynamically scalable and totally virtualized data centers is much more complicated. To provide cost-effective availability, minimize the response time of applications and make load balancing for cloud storage, a new replica placement is proposed. The replica placement is based on five important parameters: mean service time, failure probability, load variance, latency and storage usage. However, replication should be used wisely because the storage size of each site is limited. Thus, the site must keep only the important replicas.We also present a new replica replacement strategy based on the availability of the file, the last time the replica was requested, number of access, and size of replica. We evaluate our algorithm using the CloudSim simulator and find that it offers better performance in comparison with other algorithms in terms of mean response time, effective network usage, load balancing, replication frequency, and storage usage.

Introduction:

If we store all words ever spoken by human beings, its size will be equal to the size of data generated in just two days1 . Furthermore, the latest report from the International Data Corporation (IDC) expects the generated data to bypass the 175 zettabytes2 by 2025 [13]. These data are produced by various sources such as sensors, social media, or scientific simulations. This deluge of data allows empowering businesses as well as academia and has a direct impact on our lives. For example, large Internet companies such as Google and Facebook analyze their daily collected data to improve users' experiences, while research institutes such as Argonne National Laboratory run complex universe simulations to push further the boundaries of human knowledge [5]. Hence, enabling scalable and efficient data management to transform these gigantic data volumes into useful information is indispensable. To cope with the tremendous volumes of Big Data, large-scale infrastructures and scalable data management techniques are needed. Cloud computing has been evolving as the de-facto platform for running data-intensive

applications. Clouds provide the illusion of infinite resources which can be accessed in a cost-effective manner. Recently, cloud providers such as Amazon, Microsoft, and Google have equipped their infrastructures with millions of servers distributed world-wide to ease the management of Big Data. For example, Amazon Web Services (AWS) has almost 5 million servers in total [144] hosted 1 It is estimated that 2.5 exabytes are produced every day [11] and all words ever spoken by human beings have a size of 5 exabytes [11]. 21 zettabyte equals to 1021 bytes, or 1 million petabytes in hundreds of data centers on 5 continents [6], with millions of services launched every day [5]. Meanwhile, large-scale storage systems have been emerging to store and access data in cloud data centers. They run on top of thousands of machines to offer their aggregated storage capacities, in addition to providing reliable and fast data access. For example, the cloud-based Windows Azure Storage (WAS) system hosts more than an exabyte of data [2] and handles more than two billion transactions per day [4]. The distributed nature of the infrastructures and storage systems introduces an ideal platform to support Big Data analytics frameworks such as Hadoop [19] or Spark [20] to efficiently run data-intensive applications. For instance, over one million Hadoop jobs, processing tens of petabytes of data, are launched every month in Facebook data centers [9]. In general, cloud services, including data analytics services (e.g., Amazon Elastic MapReduce [14], Microsoft Azure HDInsight [12], etc.), are deployed as virtual machines (VMs) or containers. Provisioning a service to run cloud applications requires the service image (an image which encapsulates the operating system and the service software along with its dependencies) to be stored locally on the host server. Otherwise, the corresponding image is transferred through the network to the destination server introducing a delay in the provisioning time. The increasing demand for fast service provisioning in addition to the increasing size and number of service images (e.g., AWS provides more than 20,000 different public images3 , and their sizes could attain dozens of gigabytes [6]) make service image management essential for service provisioning in the cloud. Moreover, the current trend towards multi-site deployment – which is facilitated by geographically distributed clouds [12] – and the wide adoption of Edge computing to enable in-place processing (i.e., by moving computation near data sources) bring new challenges in provisioning VMs and containers. This is due to the limited bandwidth and the heterogeneity of the wide area network (WAN) connections, as well as the limited storage capacities in Edge-servers. Consequently, it is crucial to provide scalable and efficient VM and container images management to ease and enable fast service provisioning in distributed clouds and Edge systems. Traditionally, distributed storage systems operate under replication to ensure data availability. In addition, data analytics frameworks including Spark [20], Flink [18], and Hadoop [19] have extensively leveraged replication to handle machine failures (i.e., tasks of the failed machines can be simply re-executed using other replicas of the data [14]) and improve the

performance of data-intensive applications in clouds (i.e., improve data locality by increasing the probability of scheduling computation tasks on a machine which hosts the input data [12]). However, with the relentless growth of Big Data, and the wide adoption of high-speed yet expensive storage devices (i.e., SSDs and DRAMs) in storage systems, replication has become expensive in terms of storage cost and hardware cost [10]. Alternatively, erasure codes (EC) provide high data availability with low storage overhead. Thus, they are currently deployed in many distributed storage systems [11]. For example, by applying EC, Microsoft reduces the storage overhead in its cloud-based object store by more than 50% compared to replication [13]. While executing data-intensive applications under EC can also result in low storage overhead, this may incur large data transfer because the input data of each computation task (e.g., map tasks in Hadoop) is scattered on several machines. Most large cloud providers, such as Amazon and Microsoft, replicate their Virtual Machine Images (VMIs) on multiple geographically distributed data centers to offer fast service provisioning. Provisioning a service may require to transfer a VMI over the wide area network (WAN) and therefore is dictated by the distribution of VMIs and the network bandwidth in-between sites. Nevertheless, existing methods to facilitate VMI management (i.e., retrieving VMIs) overlook network heterogeneity in geo-distributed clouds. In response, we design, implement and evaluate Nitro, a novel VMI management system that helps to minimize the transfer time of VMIs over a heterogeneous WAN. To achieve this goal, Nitro incorporates two complementary features. First, it makes use of deduplication to reduce the amount of data which is transferred due to the high similarities within an image and in-between images. Second, Nitro is equipped with a network-aware data transfer strategy to effectively exploit links with high bandwidth when acquiring data and thus expedites the provisioning time. Experimental results show that our network-aware data transfer strategy offers the optimal solution when acquiring VMIs while introducing minimal overhead. Moreover, Nitro outperforms state-of-the-art VMI storage systems (i.e., OpenStack Swift) by up to 77%.

Related work

Replication technology has been widely used to enhance the performance of different applications in World Wide Web [9], peer-to-peer networks [10], ad hoc and sensor networking [11,12] and mesh networks [13]. More recently, the appearance of large-scale distributed systems like Grid [14] and Cloud [15] has made data replication become a research hot spot once again. In cloud environments, enormous scientific data and complex scientific applications call for different replication strategies, which have attracted much research recently. Many companies such as Google, Amazon have set up cloud computing infrastructures. Comparing with traditional large scale storage systems, the clouds which are sensitive to workloads and user access patterns focus on presenting and publishing storage service on Internet [16–18]. The main infrastructure and key component of

cloud computing are distributed systems, like GFS, HDFS.

The data replication strategy is adopted by distributed storage system to improve user waiting time, enhance data availability and reduce cloud system bandwidth consumption by sending the user task to different replicas with a coherent state of the same service [19]. With the advancement and development of different technologies, data replication and replica management in distributed systems have been investigated in many studies. The data replication strategies can be classified into two classes: static replication [20] and dynamic replication strategies [2-5]. The replication strategy is preset in static replication algorithms, while the dynamic replication can automatically generate and remove replicas according to changing access patterns. In Ref. [2], a static distributed data replication strategy for GFS is presented. In Ref. [6], a p-median static centralized data replication strategy is presented. The p-median model determines p replica placement nodes that minimize the request weighted total distance between the requesting nodes and the replication nodes holding the replicas assigned.

Hussein et al. [7] presented an adaptive replication strategy (ARS) in the cloud environment. The strategy investigates the availability and efficient access of each file in the data center, and analyses how to enhance the reliability of the data files based on prediction of the user access to the blocks of each file. Also, it redeploys dynamically large-scale different files replicas on various data sites with minimal cost using heuristic search for each replication. The strategy determines the files which are popular files for replication based on checking the recent history of the data access to the files using HLES time series. Once a replication factor based on the popularity of the files is less than a particular threshold, the replication signal will be triggered. Hence, the adaptive strategy finds the suitable replication location based on a heuristic search for the best replication factor of each file. Experimental evaluation shows that the adaptive strategy behaves effectively to improve the availability of the cloud system under study. The limitation of the algorithm is that it considers only the popularity degree in replica placement. Further, they do not take into account the load balancing of the system.

Rajalakshmi et al. [8] proposed a dynamic replica selection and placement (DRSP) to improve availability of data in the cloud. The strategy consists of two main steps: file application and replication operation. The first step contains the replica location and creation by using catalog and index. The index is applied for storing replica file into local or remote location. The indexer also maintains master and slave replicas location. Select site with number of access is greater than the threshold value (Tα) it redirects the algorithm. Equation (1) is to find the threshold (Tα) value the ratio between threshold of replication (RT) and the total number of request. Tα = RT NumberReplica. (1) The second step is used to determine whether there is enough space in the destination to place the requested file or not. The proposed systems are developed under the Eucalyptus cloud environment.

The experimental result demonstrates that a replica selection and placement transparently stores data in geographic locations and improves the data access performance and bandwidth utilization. The shortcoming of them is that they only consider a restricted set of factors affecting the replication decision. Further, they only focus on the improvement of the system performance and they do not address the storage usage issue and consistency of data file.

System model

Clouds are usually composed by large and power-consuming data centers presented to provide the elasticity and scalability required by its customers, and substantial percentage of these data centers process large scale data intensive applications and MapReduce [5] based on such GFS [7], HDFS [21] distributed storage system which has emerged as a key infrastructure and component for building cloud services or applications. In this paper, we assume that the cloud storage cluster is composed of m independent heterogeneous data nodes D1, ..., Dj, ..., Dm storing a total of n different files f1,..., fi,..., fn. Figure 1 presents the system model of our replication management. The replication strategy is applied to distribute n files into m data nodes. We suppose that each access to file fi is a sequential read of the total file, which is a typical assumption in most file systems [3]. We do not consider file partitioning, and thus, each file must be placed entirely onto one data node. This does not limit the generality of our strategy, as each file segment can be treated as a stand-alone file. Thus, the problem we investigated is statically or dynamically assigning no partitioned files in a cloud storage cluster where the file accesses show Poisson arrival rates and fixed service times. In this paper, we only consider that all the data are read only, and thus no data consistency strategy is needed. 4 Adaptive data replication strategy We propose a multi-objective optimized data replication strategy for cloud storage. We take into account several factors such as mean service time, load variance, storage usage, failure probability, and latency to capture the relationship among replica layout and these performances. We try to optimize file availability, mean service time, load variance, and latency in order to find out different trade-offs between these objectives. Since there is more than one objective, we are doing multi-objective optimization to find the optimal replication factor and the replication layout for each file. This enables us to search for solutions that yield close to optimal values for these parameters. Users can define weights according to their own needs such as setting a higher value on their more concerned performance which makes the proposed strategy adaptable. We now explain our new replication scheme, called the adaptive data replication strategy (ADRS), which not only consists of effective replica placement, but also the replica replacement strategy. The ADRS has two main parts: 1) In order to meet the high availability, high fault tolerance and high efficiency requirement, it is necessary to dynamically adjust the sites to place the new replicas according to the current cloud environments. Random replica placement might cause the access skew that

some sites are accessed frequently, but some sites are idle. This may result in load imbalance across cloud storage, as well as poor parallelism and low performance. However, appropriate replica placement in ultra-large-scale, dynamically scalable and totally virtualized data centers is much more complicated. We can improve the response time by generating new replicas and storing them to the light-load sites. When a new replica needs to be generated and prepares to seek the best placement of the replica, it must consider the various factors:

• Mean service time (MST) Mean service time exhibits the ability of the system process rate. Storing popular files in sites with better performance and rarely accessed files in those sites with relatively weak performance is able to improve the average service time of the system. Let S T(i, j) be the expected service time of file fi on the data node Nj (1 j m), It can be calculated by S T(i, j) = d(i, j) × sizei tpj , where the decision variable d(i, j) equals to 1 if the file fi exists on data node Nj, otherwise it sets to 0; sizei is the size of file fi and tpj is the transfer rate of the data node Nj.

• Load variance (LV) Data node load variance is the standard deviation of data node load of all data nodes in the cloud storage cluster which can be used to show the degree of load balancing of the system. The lower value of load variance, the better load balancing is achieved. Since the combination of access rate and service time of the file fi accurately presents the load of it, the load Load(i, j) of fi which is on the data node Nj is defined as follows: Load(i, j) = Acc(i, j) × S T(i, j), (7) where Acc(i, j) is the access rate of read requests coming from data node Nj asking for file fi. If the node is overloaded, ADRS should ignore this site based on the site's load information, because the site has a bad performance in reading and writing operations with high load.

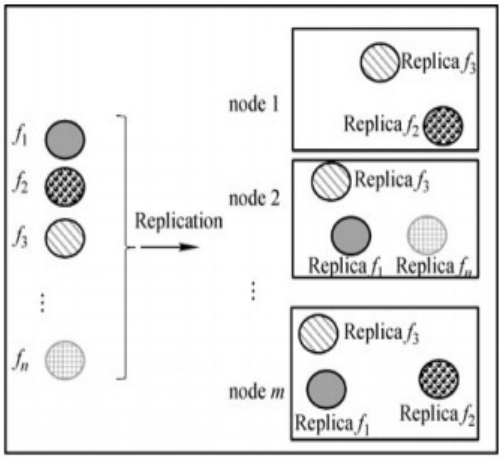


Fig. 1 System model of replication management

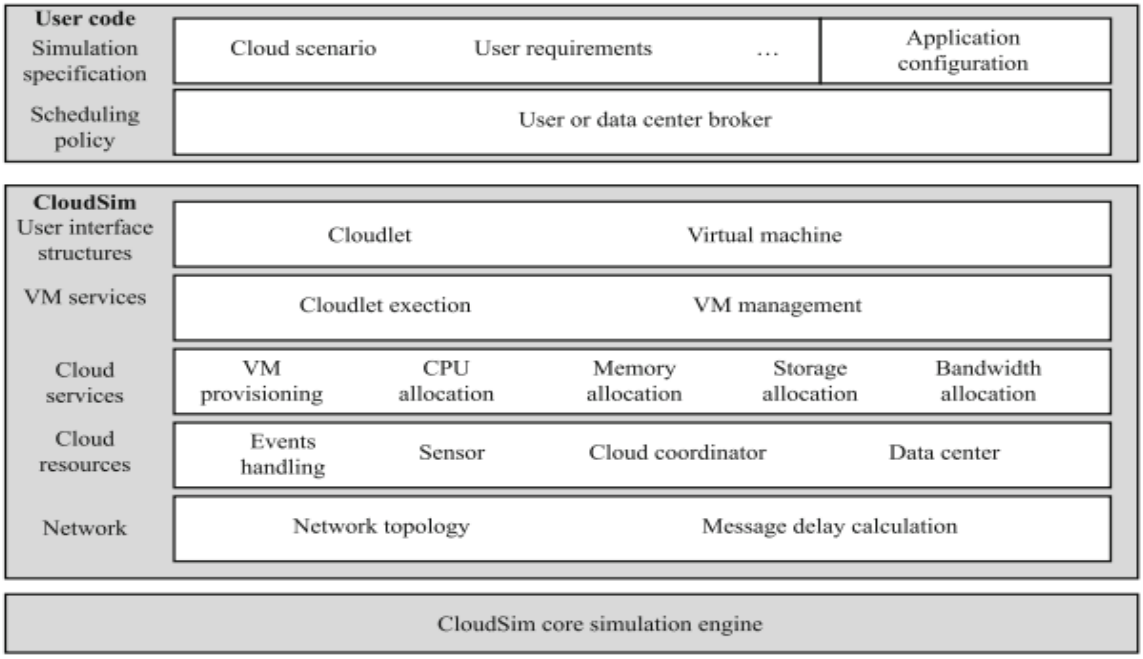
• Storage usage (SU) It is better to place popular files in data nodes with low storage usage to minimize the waiting time.

• Failure probability (FP) If one node fails, a replica of the failed service will be possibly created on a different node in order to process the requests. Thus, it is better to place popular files in data nodes with low failure probability to minimize their latency.

• Latency (L) Minimizing latency is critical for any storage system. Minimizing latency depends on utilizing high bandwidth capacities, as high bandwidth channels yield lesser latency. Thus, it is better to store files in data nodes with high bandwidth to minimize their latency. In our study, we only consider read latency

Experimental results:

The CloudSim simulation layer provides support for modeling of virtualized cloud-based data center environments including dedicated management interfaces for VMs, memory, storage, and bandwidth. The main issues, such as provisioning of hosts to VMs, managing application execution, and controlling dynamic system state, are managed by this layer. A cloud provider, who wants to check the efficiency of different strategies in allocating its hosts to VMs (VM provisioning), would need to implement his policies at this layer. Such operation can be done by programmatically extending the core VM provisioning functionality. There is an obvious distinction at this layer related to provisioning of hosts to VMs. A cloud host can be concurrently allocated to a set of VMs that run applications based on SaaS provider's determined QoS levels. This layer also presents the functionalities that a cloud application developer can extend to do complex workload profiling and application performance evaluation. The top-most layer in the CloudSim stack is the User Code that provides basic entities for hosts (number of machines, their specification, and so on), applications (number of tasks and their requirements), VMs, number of users and their application types, and broker scheduling strategies. By developing the primary entities given at this layer, a cloud application developer can do the following operations: 1) create a mix of workload request distributions, application configurations; 2) model cloud availability scenarios and do robust tests based on the custom setting; and 3) implement custom application provisioning approaches for clouds and their federation.



Conclusion

Cloud computing is becoming increasingly popular; cloud storage services attract more attentions for their high scalability and availability with a low cost. Replication is one of the performance improving techniques for cloud storage system that has been widely used. In this work, we propose adaptive data replication strategy (ADRS) for cloud computing data centers which improves response time. Efficient replica placement can significantly boost the overall performance and load balance of the cloud environment. These motivate us to propose a novel replica placement policy to efficiently distribute workload across cloud nodes. ADRS stores replicas in the best site based on mean service time, failure probability, load variance, latency and storage usage. We also present a new replica replacement strategy based on the availability of the file, the last time the replica was requested, number of access, and size of replica. Through the expansion of CloudSim platform, adaptive data replication is implemented. We compare ADRS to five existing algorithms, ARS, DRSP, CIR CDRM and build-time. Mean response time, effective network usage, storage usage, replication frequency, and load variance are used as the performance evaluation metrics. The experimental results show that ADRS outperforms the other algorithms. Especially, its performance becomes better and better with the incensement of the number of tasks. The effectiveness of the adaptive data replicas management in real cloud environment will be explored in the next step of the research. We also plan to research the trade-off problem for quality attributes such as the availability and costs of various resources.

References

1. Mi H B, Wang H M, Zhou Y F, Rung-Tsong Lyu M, Cai H, Yin G. An online service-oriented performance profiling tool for cloud computing systems. Frontiers of Computer Science, 2013, 7(3): 431–445

2. Fu X, Zhou C. Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. Frontiers of Computer Science, 2015, 9(2): 322–330

3. Chen T, Bahsoon R, Tawil A R. Scalable service-oriented replication with flexible consistency guarantee in the cloud. Information Sciences, 2014, 264: 349–370

4. Wu H, Zhang W B, Zhang J H, Wei J, Huang T. A benefit-aware on-demand provisioning approach for multi-tier applications in cloud computing. Frontiers of Computer Science, 2013, 7(4): 459–474

5. Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. Computer Communication Review, 2008, 38: 63–74

6. Amazon-S3.Amazon simple storage service (Amazon s3). http://www.amazon.com/s, 2009

7. Ghemawat S, Gobioff H, Leung S. The Google file system. In: Proceedings of the 19th ACM Symposium on Operating Systems Principles. 2003

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

9. Qiu L L, Padmanabhan V N, Voelker G M. On the placement of Web server replicas. In: Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies. 2001, 1587–1596

10. Aazami A, Ghandeharizadeh S, Helmi T. Near optimal number of replicas for continuous media in ad-hoc networks of wireless devices. In: Proceedings of the 10th International Workshop on Multimedia Information Systems. 2004

11. Intanagonwiwat C, Govindan R, Estrin D. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking. 2000

12. Tang B, Das S R, Gupta H. Benefit-based data caching in ad hoc networks. IEEE Transactions on Mobile Computing, 2008, 7(3): 289–304

13. Jin S D, Wang L M. Content and service replication strategies in multihop wireless mesh networks. In: Proceedings of ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. 2005

14. Dabrowski C. Reliability in grid computing systems. Concurrency Practice and Experience, 2009, 21(8): 927–959

15. Bonvin N, Papaioannou T G, Aberer K. Dynamic cost-efficient replication in data clouds. In: Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds. 2009, 49–56

16. Milani B A, Navimipour N J. A comprehensive review of the data replication techniques in the cloud environments: major trends and future directions. Journal of Network and Computer Applications, 2016, 64: 229–238

17. Bonvin N, Papaioannou T G, Aberer K. A self-organized, fault tolerant and scalable replication scheme for cloud storage. In: Proceedings of the 1st ACM Symposium on Cloud Computing. 2010, 205–216

18. Nguyen T, Cutway A, Shi W. Differentiated replication strategy in data centers. In: Proceedings of the IFIP International Conference on Network and Parallel Computing. 2010, 277–288

19. Ahmad N, Fauzi A C, Sidek R M, Zin N M, Beg A H. Lowest data replication storage of binary vote assignment data grid. In: Proceedings of the 2nd International Conference on Networked Digital Technologies. 2010, 466–473

20. Bin L, Jiong Y, Hua S, Mei N. A QoS-aware dynamic data replica deletion strategy for distributed storage systems under cloud computing environments. In: Proceedings of the 2nd International Conference on Cloud and Green Computing. 2012, 219–225

21. Shvachko K, Hairong K, Radia S, Chansler R. The Hadoop distributed file system. In: Proceedings of the 26th

Symposium on Mass Storage Systems and Technologies. 2010, 1–10

22. Rahman R M, Barker K, Alhajj R. Replica placement design with static optimality and dynamic maintainability. In: Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid. 2006, 434–437