Future Generation Computer Systems 65 (2016) 10–32

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers

Navneet Kaur Gill, Sarbjeet Singh *

Computer Science and Engineering, UIET, Panjab University, Chandigarh, India

HIGHLIGHTS

- The paper presents a dynamic, cost-aware, optimized data replication strategy.
- The concept of knapsack has been used to optimize the cost of replication.
- The strategy involves re-replication also, without compromising data availability.
- Mathematical descriptions and illustrations have been provided for different phases.
- The strategy has been simulated and evaluated to demonstrate its effectiveness.

ARTICLE INFO

Article history:
Received 28 June 2015
Received in revised form
3 April 2016
Accepted 14 May 2016
Available online 27 May 2016

Keywords:
Data availability
Data replication
Cost of replication
Knapsack
Re-replication
Re-balancing

ABSTRACT

In cloud computing, it is important to maintain high data availability and the performance of the system. In order to meet these requirements, the concept of replication is used. As the number of replicas of a data file increases, the data availability and the performance also increases, but at the same time, the cost of creating and maintaining new replicas also increases. In order to enjoy the maximum benefits of replication, it is essential to optimize the cost of replication. The cloud systems are heterogeneous in nature as the different data centers have different policies, hardware and software configurations. As a result of this, the replicas of a data file placed at different data centers have different availabilities and replication costs associated with them. In this paper, a dynamic, cost-aware, optimized data replication strategy is proposed that identifies the minimum number of replicas required to ensure the desired availability. The concept of knapsack has been used to optimize the cost of replication and to re-replicate the replicas from higher-cost data centers to lower-cost data centers, without compromising the data availability. Mathematical descriptions and illustrations have been provided for the different phases of the proposed strategy, keeping in mind the heterogeneous nature of the system. The proposed strategy has been simulated using the CloudSim toolkit. The experimental results indicate that the strategy is effective in optimizing the cost of replication and increasing the data availability.

1. Introduction

At present, cloud computing is the most popular distributed computing paradigm which provides IT resources to its users in the form of utilities [1–5]. The virtualized resources are leased to users over the Internet and are provisioned "on-demand". Moreover, the resources are made available for use through "pay-per-use" model, wherein users pay for the demanded resources as per their usage. Thus, the main reason behind the increasing popularity of the cloud is: "why one should buy, when one can get it on rent".

Providing reliable services along with high data availability and the performance are the important requirements that need to be essentially met. The concept of replication is used to ensure these requirements. Data replication is the process of keeping multiple copies of a data file at multiple sites in order to achieve better performance, availability and reliability [6–9]. Replication is commonly used in data intensive applications where data is shared and need to be accessed from different sites, situated at different geographical locations [6,7]. Replication increases availability of the data. If some of the sites holding the requested file fail, the request can still be served from other sites holding the replica of the desired file. In addition to this, replication also improves the

* Corresponding author. Tel.: +91 9815951674.
E-mail addresses: navneetgill31@gmail.com (N.K. Gill), sarbjeet@pu.ac.in (S. Singh).

http://dx.doi.org/10.1016/j.future.2016.05.016
0167-739X/

overall performance of the system as the user's requests can be satisfied from the nearest sites holding the replica of the requested data file. This improves the access time and the response time of the system. Last but not the least; replication also makes the system more reliable. If any replica of a data file gets corrupted, or fails due to some reason, the user's requests can still be served from other non-corrupted sites holding the desired replica. Hence, replication provides high data availability, high performance and also increases the reliability of the system.

As "every coin has two sides", replication too has some drawbacks associated with it. As the number of replicas of a data file increases, the cost of creating and maintaining these replicas also increases. In fact, there are three basic issues associated with the replication process [10]. The first issue is to determine the data file that needs to be replicated and the time when the replication process should be triggered for it. Creating replicas of all the data files is not a sensible approach because it increases the overhead of maintaining large number of replicas and also increases the cost of replication beyond the acceptable level. So it is sensible to create replicas of only those data files that have higher access rate and are more popular compared to other data files. Replicating unpopular data files is not a beneficial approach. Thus, those data files that are popular, and are high in demand, should be replicated in order to enjoy the maximum benefits of the replication. The second issue is to determine how many new replicas should be created for a data file that is selected for replication. Creating extra replicas of a data file leads to wastage of storage space and it is also difficult to maintain the large number of replicas. Moreover, the cost of replication also increases with the increase in the number of replicas. Thus, only appropriate number of replicas should be created. The appropriate number of replicas is decided based on the demand of the data file in the system at a particular moment of time. As the number of replicas of a data file increases, the availability also increases side by side. However, once the certain numbers of replicas are created, the availability of the data file reaches its maximum and creating further replicas do not affect the availability of the system. Instead, it increases the overhead charges. Hence, deciding about how many new replicas of a particular data file should be created at any particular moment of time is also an important issue. The third issue is to decide about the locations where the new replicas can be placed. Again, if the replicas are placed randomly, then they will not result in increase in the performance of the system effectively. Thus, replicas should be placed in such a way that the performance of the system is maximized along with the increase in the availability. These three issues need to be addressed properly in order to enjoy the maximum benefits of replication. Moreover, the cloud systems are heterogeneous in nature as they are composed of different data centers of different configurations, hardware and policies. So, the data centers that provide high processing power, high fault tolerance and high data availability have higher cost compared to those that have lower performance and capabilities. Furthermore, the cost of replication of a data file is also different on different data centers.

In this paper, we have proposed a dynamic replication strategy for heterogeneous cloud systems that optimizes the cost of replication in such a way that the availability and the performance of the system remain maintained at the desired levels. All the three issues of replication are addressed considering the heterogeneous nature of the system. An algorithm has been proposed comprising of different phases, addressing these issues. Mathematical descriptions have been provided along with suitable illustrations. The concept of Knapsack is used to optimize the cost of replication. The proposed strategy has been simulated using the CloudSim toolkit. The experiments elucidate the effectiveness of the proposed replication strategy.

The rest of the paper is organized as follows: Related work is discussed in Section 2. The system design and architecture is presented in Section 3, followed by Section 4 that describes the proposed algorithm in detail. Section 5 presents details regarding the simulation and discusses the experimental results. Finally, Section 6 concludes the paper with future scope.

2. Related work

Replication plays an important role in all distributed systems. Two different replication strategies are in widespread use: static replication [11–13] and dynamic replication [14–27]. Based on these two strategies, different replication algorithms and schemes have been proposed by different researchers. In static replication, the number of replicas for each data file is determined in advance, manually. In this strategy, the placement of replicas is also pre-decided and these replicas are managed manually. It does not adapt according to the changes in the environment and users' access patterns. However, in dynamic replication strategies, the replicas of each data file are created, placed and maintained dynamically according to the changes in the environment and users' access patterns. In cloud computing systems, users' access patterns may keep on changing time to time, hence, to main high availability along with better performance, the replication strategies should adapt dynamically to changes in the environment. In cloud, dynamic replication strategies are considered to be more desirable than static replication strategies.

Based on users' access patterns, six different replica placement strategies are proposed in [14]. These are: No replication, Best Client, Cascading, Plain caching, Caching plus Cascading and Fast Spread. The "No Replication" strategy is to place all data files only at the root node of the hierarchical data structure with no replication. The "Best Client" strategy is to place the replicas of a data file at the client generating maximum number of requests to this data file. However, it is not efficient for all the access patterns. On the other hand, the "Cascading" strategy is to place the replicas in the direction and the path of the best client node using top-down approach. But it is efficient only when the locality of access patterns is known. Further, this strategy is to be used if the main focus is to reduce access latency. The "Plain Caching" strategy is to place the copy of the data file locally at the requesting client node. The "Caching plus Cascading" strategy is combination of the "Cascading" and the "Plain Caching" strategies. To reduce bandwidth consumption, the "Fast Spread" strategy is used, wherein the replicas are placed at all the nodes throughout the path of the client node whenever it requests to access the data file. This one is the best in case of random access patterns. Since the prediction of user access patterns and the workload is very difficult to predict in advance, few more replica placement algorithms are proposed in [15], based on two factors: utility and risk index. In these algorithms, the distance between the requesting site and the site holding the replica, and the information about the number of requests originating from the requesting site, are utilized to calculate the utility and the risk index. The main focus is to place the replicas at the site having the lowest utility and the highest risk index. In [16], an advanced version of the "Fast Spread" strategy, namely the Enhanced Fast Spread (EFS) is proposed. The main drawback of the "Fast Spread" strategy is that the replicas that are replaced may be more important than those replicas that are replacing them. Therefore, in EFS, the replacement is done keeping in mind the importance of the replicas. The importance of replicas is calculated using the information like the number of accesses and the frequency of these accesses along with the size of the replicas.

Since replication has additional overheads like storage cost, maintenance cost, etc., therefore only those data files should be replicated that are popular and need to be replicated to meet the