Future Generation Computer Systems 51 (2015) 120–131

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Workload balancing and adaptive resource management for the swift storage system on cloud

Zhenhua Wang[a], Haopeng Chen[a,*], Ying Fu[a], Delin Liu[a], Yunmeng Ban[b]

[a] Shanghai Jiao Tong University, Shanghai, China
[b] University of Massachusetts at Amherst, Amherst, USA

HIGHLIGHTS

- We propose a workload balancing and adaptive resource management framework for Swift.
- We implement optimization algorithms of dynamic workload balancing for Swift.
- We conduct an experiment to demonstrate the effectiveness of this framework.

ARTICLE INFO

Article history:
Received 30 June 2014
Received in revised form
12 September 2014
Accepted 10 November 2014
Available online 20 November 2014

Keywords:
Distributed storage system
Swift storage system
Workload balancing
Resource management

ABSTRACT

The demand for big data storage and processing has become a challenge in today's industry. To meet the challenge, there is an increasing number of enterprises adopting distributed storage systems. Frequently, in these systems, storage nodes intensively holding hotspot data could become system bottlenecks while storage nodes without hotspot data might result in low utilization of computing resource. This stems from the fact that almost all the typical distributed storage systems only provide data-amount-oriented balancing mechanisms without considering the different access load of data. To eliminate the system bottlenecks and optimize the resource utilization, there is a demand for such distributed storage systems to employ a workload balancing and adaptive resource management framework. In this paper, we propose a framework of workload balancing and resource management for Swift, a widely used and typical distributed storage system on cloud. In this framework, we design workload monitoring and analysis algorithms for discovering overloaded and underloaded nodes in the cluster. To balance the workload among those nodes, Split, Merge and Pair Algorithms are implemented to regulate physical machines while Resource Reallocate Algorithm is designed to regulate virtual machines on cloud. In addition, by leveraging the mature architecture of distributed storage systems, the framework resides in the hosts and operates through API interception. To demonstrate its effectiveness, we conduct experiments to evaluate it. And the experimental results show the framework can achieve its goals.

1. Introduce

With the increasing demand for big data processing, traditional centralized storage systems cannot meet the requirement. Therefore, distributed storage systems become more and more popular because of their horizontal scalability and high robustness. Generally, a distributed storage system contains numbers of storage nodes. Data can be equally distributed to those nodes by some techniques, such as hashing. By distributing data to different storage nodes, the load of accessing data is also distributed to those nodes, which improve the ability of big data processing in distributed storage systems.

However, the load balancing mechanism based on data amount balancing is often unsatisfied in practice. According to the traditional balancing mechanisms, if access load of different data is similar, the workload of all the storage nodes is well balanced. However, the access load of different data is almost different. Hotspot data are accessed more frequently than non-hotspot data. Consequently, storage nodes storing large amount of hotspot data could become system bottlenecks while the "free" storage nodes could cause low computation resource utilization. And the traditional load balancing mechanisms are difficult to achieve the goal of load balancing based on access load. Since the hotspot data are application-related, it is almost impossible for the original

* Corresponding author.
E-mail address: chen-hp@sjtu.edu.cn (H. Chen).

http://dx.doi.org/10.1016/j.future.2014.11.006
0167-739X/© 2014 Elsevier B.V. All rights reserved.

distributed storage systems to predict which data will become hotspot data and which will not. Therefore, how to achieve the goal of workload balancing and adaptive resource management automatically has not been well studied.

To achieve this optimization goal, we propose a workload balancing and adaptive resource management framework in this paper. To demonstrate its effectiveness, we employ it to the Swift Storage System. Swift Storage System is a very representative distributed storage system. As the most important storage component of OpenStack, it has been widely used in the cloud platform. In addition, it is one of the most popular object storage systems.

The contributions of this paper are as follows.

- We propose a workload balancing and adaptive resource management framework based on the virtualization technology, which can be applied to Swift. The frame work is lightweight and requires no source code modification in the guest OS and storage application.
- We implement several optimization algorithms of dynamic workload balancing and adaptive resource management for Swift Storage System on cloud.
- We conduct an experiment to demonstrate the effectiveness of this framework in workload balancing and resource management for Swift Storage System when hotspot data exist.

The rest of the paper is organized as follows. Section 2 introduces some important related work. Section 3 describes motivating experiments to show the poor performance of the default load balancing mechanisms of Swift Storage System when hotspot data exist. Section 4 introduces the framework as well as the design and implementation of the algorithms we integrated. Section 5 presents the experimental results and analysis. Section 6 summarizes the paper and draws conclusions of the result.

2. Related work

As a well-known and typical distributed storage system, Swift is playing an important role in cloud storage. It is the storage component of OpenStack. Therefore, it plays a significant role in the storage of cloud platform. It stores the data of thousands of virtual machines. And it can be used independently as a kind of object storage tool. It can store big data and handle massive request. It is reliable and extensible. Because of its advantages, it is popular and typical. In Swift, there are mainly two kinds of nodes, including proxies and storage nodes. Data requests are sent to proxies and proxies fetch data stored in storage nodes to respond to users. Swift has several features in load balancing, including scalable proxy mechanism, replica load balancing mechanism and data amount balancing mechanism. Through these mechanisms, big data can be well stored, managed and accessed.

Concurrently, virtualization technology is making a significant impact on how resources are used and managed in a cloud computing platform. Several virtualization solutions (Xen [1], XenServer [2] and VirtualBox) are getting more and more mature in resource management. In this paper, the virtualization technology plays a significant role in the design of the proposed framework. It is the basis of workload balancing and resource management.

Load balancing mechanisms for distributed system are also very important. In Swift, there are mainly three default load balancing mechanisms. Brief introductions of them are as follows.

Scalable proxy mechanism allows users to set up more than one proxy. And requests from users can be distributed to these proxies, which can avoid the bottleneck caused by single proxy.

Replica load balancing mechanism balances the workload caused by fetching data through responding with data replicas stored in different storage nodes.

Data amount balancing mechanism tries to distribute data to all the storage nodes evenly. If the access chance of each data is nearly the same, the workload of each storage node is nearly the same.

In addition to these balancing mechanisms of Swift, there are also plenty of relative researches on workload balancing strategy of distributed systems. Haroshi [3] proposes two replication methods for balancing the load on the storages distributed over P2P networks while limiting the degradation of the search performance within an acceptable level. Madathil, D.K. [4] proposes a novel load balancing and performance oriented static data placement strategy which can be applied to distributed storage subsystems in clusters to noticeably improve system responsiveness. In [5], for the problems of workload imbalanced file replicas in distributed storage systems, the authors propose a method based on index name server. Its main idea is to select a suitable file replica to respond to user based on workload analysis of nodes. This method optimizes the strategy of selecting suitable file replicas to respond to users, however, all the nodes and files remain unmodified. In [6], for object storage systems which are different from the traditional distributed storage systems, the authors propose an optimization method for participation of files in object storage systems. In addition to improve the data amount balancing, it also improves the utilization of network resources. In [7], for object storage systems, the authors propose DifferStore: a differentiated storage service in object-based storage systems. DifferStore utilizes a two-layer architecture to efficiently decouple upper-layer application specific storage policies and lower-layer application independent storage functions. Its optimization strategy is based on applications. In [8], the authors proposed a static scheduling method based on workload balancing in the continuous case. This method works well when the case is continued, however, it is limited in other cases.

Above all, most of these works of researches on workload balancing focus on data management, access load balancing for proxies and balancing mechanisms based on replicas in distributed storage system. Their balancing targets are similar to the mechanisms' of Swift. However, almost all the previous researches cannot eliminate the bottleneck caused by hotspot data fundamentally, since they remain the static data storage mechanism unmodified. And how to achieve the goal of load balancing according to actual workload or access-load in distributed storage system has not been well studied. Performance of distributed storage system with hotspot data is often poor. For example, a storage node in Swift storing several hotspot data may become the bottleneck of the system and its default load balancing mechanisms do not work. This paper proposes a framework which can eliminate the bottleneck and achieve the goal of workload aware balancing and resource management for Swift on cloud.

Discovery of workload exception is the basis of workload aware balancing. In [9], the nodes in a cluster are grouped into several small sets. In each set, there is a central node, which is responsible for monitoring other nodes in its set. By this way, the monitoring performance is improved, but the hotspot data cannot be located precisely. Comparing with this architecture, in this paper, each node monitors itself. Consequently, it is much easier to locate the hotspot data. In [10], for each chunk in MongoDB [11], its access-load is evaluated by the number of various operations on it. However, the weights of different operations are assigned by experience, which makes them too subjective. In this paper, the access-load is evaluated by computation resource utilization, which can more objectively reflect its actual access-load. In [12,13], the exception of workload is detected based on the predicted access load. Predicted workload is much better than real-time access load for workload aware balancing since it can avoid workload exception before its happening, but large amount of historical records is needed to guarantee its precision. In [14], the authors propose a reliable state monitoring in cloud datacenters. It focuses on solving the problem of message delay and loss in monitoring. In [15], the authors propose a method of workload-aware system monitoring using performance predictions applied to a large-scale e-mail system. Unlike the traditional method, the workload-aware