A Performance Evaluation Method of Load Balancing Capability in SaaS Layer of Cloud Platform

Liu Hui1,*, Niu Zhaocheng1, Wu Tiancheng1, Du Junzhao1, Zhang Feng1 and Bi Haoyu1.

1 School Of Computer Science and Technology, Xidian University, Xi’an, Shaanxi, 710071, China
*Corresponding author’s e-mail: liuhui@xidian.edu.cn

Abstract. Cloud platform load balancing has become one of the essential elements in modern cloud platform operations. However, there are still some problems in the evaluation methods of cloud platform load balance, such as unreasonable evaluation indexes and lack of comprehensive evaluation methods. This paper first proposes a performance evaluation method of cloud platform load balancing capabilities. This method analyzes the characteristics of the cloud platform and the actual needs of users, designs and proposes a cloud platform load balancing capability evaluation system, quantitatively analyzes and quantifies specific indexes, and uses the Analytic Hierarchy Process to determine the weight of each index. Finally, we deploy and test our method on the OpenStack cloud platform. Experiments show that the evaluation results of the method proposed in this paper are consistent with the actual results, which verifies the feasibility and rationality of the method proposed in this paper.

1. Introduction

According to the definition of cloud platform by the National Institute of Standards and Technology[1], the basic characteristics of cloud platform include on-demand self-service, ubiquitous network access, rapid flexibility, and measurable services, etc. Among these basic characteristics, the load balancing capabilities of the SaaS layer are mainly related to on-demand self-service and rapid scalability. Load balancing capabilities can enable user applications to rapidly expand while maintaining stable performance, which can be derived from quality of service[2]. Therefore, the load balancing ability can be evaluated from two perspectives: quality of service and stability. Besides, there are multiple load balancing strategies and different roles like service provider and application tenant, all of which could be taken into account to make the evaluation more comprehensive.

2. Related work

Li et al.[3] divided the indexes into server physical indexes and service capability indexes through literature analysis, and proposed several secondary indexes. However, the indexes selected are relatively simple. Bardsiri et al.[4] further expanded the indexes, selecting performance, cost performance, security and other indexes from the perspective of quality of service, but the paper did not give specific formulas for these indexes. Sambit et al.[5] selected two indexes of energy consumption and load execution time, and used different load balancing strategies in CloudSim to conduct a comparative analysis. Literature[6-7] also compared the performance of cloud platform load balancing strategies in several dimensions.

In summary, existing researches lack consistent and quantifiable evaluation indexes and systems. Therefore, how to evaluate the load balancing capabilities comprehensively and quantitatively is still a
problem. From the perspective of the relationship between load balancing and the basic characteristics of cloud platform, the realization principle of load balancing and traditional performance testing methods, we propose a method for evaluating the load balancing ability of the cloud platform SaaS layer based on the Analytic Hierarchy Process (AHP) [8]. And different load balancing strategies were tested on OpenStack, which verified the correctness and effectiveness of the method.

3. Cloud platform load balancing capability evaluation system design
In this section, we propose an evaluation system for the load balancing capability of the cloud platform.

3.1 Quality of service
From the perspective of cloud platform load balancing service providers, quality of service can be defined as the difference between actual application performance and optimal performance, that is, performance loss, which is recorded as:

\[ ql = \frac{\sum_{i=1}^{n} q_{i \text{best}} - q_{i}}{q_{i \text{best}}} \]  

(1)

Among them, \( q_{l} \) represents the performance loss during the load balancing process, \( q_{i} \) represents the actual performance of the \( i \)th request, \( q_{i \text{best}} \) represents the best performance of the \( i \)th request, and \( n \) represents the number of request types.

From the perspective of cloud platform application tenants, the quality of service can be defined as the responsiveness of the cloud platform:

\[ res = 1 - e\left(-\frac{aD^{2}}{Q}\right) \]  

(2)

Among them, \( D \) represents the number of requests for packet loss, \( Q \) represents the number of requests, and \( a \) represents the user-defined weight value.

3.2 Stability
In cloud computing, stability reflects the platform’s ability to provide users with continuous services. We select resource distribution balance degree of CPU, memory, disk I/O, and network bandwidth to measure the stability, which is recorded as:

\[ sd_{u} = \frac{\sum_{i=1}^{n} (u_{i} - \bar{u})^{2}}{n} \]  

(3)

Among them, \( \bar{u} \) represents the average value in this dimension, and \( u_{i} \) represents the value of a specific node in the resource dimension \( u \). The calculation formula of the stability of cluster load balancing \( S \) is:

\[ S = \frac{\sum_{i=1}^{n} \sqrt{sd_{\text{cpu}} + sd_{\text{mem}} + sd_{\text{inetin}} + sd_{\text{inetout}}}}{n} \]  

(4)

3.3 Availability
Availability is related to the successful execution of service and can be reflected by the rate of invalid requests. A period of time can be defined as the minimum time block, and the availability in this time block can be evaluated from the side by the invalid request rate. The calculation formula of \( ur \) is:

\[ ur = \frac{\sum_{i=1}^{n} m_{i}}{m_{i \text{total}}} \]  

(5)

Among them, \( m_{i} \) is the number of invalid requests for the minimum time block \( i \), \( m_{i \text{total}} \) is the number of requests for the minimum time block \( i \), and \( n \) is the number of the minimum time block.
3.4 Overall evaluation

As shown in Figure 1, the AHP is selected to assign weights to the four lower-level indexes. We construct the judgment matrix \( R \) to reflect the relationship between the indexes, which should be modified to meet the consistency test. Then the weight \( w_i \) can be obtained as:

\[
w_i = \frac{(\prod_{j=1}^{n} a_{ij})^\frac{1}{n}}{\sum_{i=1}^{n} (\prod_{j=1}^{n} a_{ij})^\frac{1}{n}}
\]

(6)

Among them, \( a_{ij} \) represents the importance of index \( i \) relative to index \( j \), and \( n \) is the total number of indexes. Finally, the overall performance score \( lb_{bc} \) is obtained:

\[
lb_{bc} = \sum_{i=1}^{n} s_i \cdot w_i
\]

(7)

Among them, \( s_i \) is the quantitative value of index \( i \), and \( w_i \) is the corresponding weight value.

In general, the performance evaluation steps of cloud platform load balancing ability is:
1) Create the load receiving image required by the test on the cloud platform;
2) Run a single instance to receive a small amount of load and record the result;
3) Create multiple back-end receiving instances and Configure the load balancing strategy;
4) Start the load and record the relevant performance indexes;
5) Carry out comprehensive evaluation through the evaluation model based on AHP.

4. Experimental results and analysis

4.1 Experimental software and hardware platform and test data
We choose 5 heterogeneous high-performance servers as the experimental hardware platform. The servers are interconnected through a Gigabit switch. For the experimental software platform we choose the Pike version of the OpenStack cloud platform. The application server is deployed on 3 virtual machines to simulate the load receiving end. The self-developed load generator software is deployed on other nodes to simulate and generate three typical workloads: CPU-intensive load, memory access-intensive load, and network consumption-intensive load.
4.2 Experimental results and analysis
Table 1 shows the number of different types of requests and the average response time of requests in the benchmark test phase and the service test phase. In the service test phase, the load balancer is configured to be a polling strategy, a minimum connection strategy, and a source address hash strategy for testing, and the load sending time is set to 15 minutes.

| Test type          | Number of CPU-intensive Requests | CPU-intensive Response Time | Number of Memory-intensive Requests | Memory-intensive request response time | Number of Bandwidth-intensive Requests | Bandwidth-intensive request response time | Number of timed out requests |
|--------------------|----------------------------------|-----------------------------|------------------------------------|--------------------------------------|----------------------------------------|------------------------------------------|-----------------------------|
| Benchmark          | 2280                             | 2448                        | 2318                               | 509                                  | 6840                                   | 288                                      | 262                         |
| Polling            | 133112                           | 3010                        | 144990                             | 839                                  | 368096                                 | 470                                      | 7820                        |
| Minimum Connection | 158504                           | 2694                        | 143692                             | 772                                  | 369444                                 | 429                                      | 4713                        |
| Source Address Hashing | 159296                           | 2898                        | 145140                             | 829                                  | 380644                                 | 458                                      | 5657                        |

After the load balancing service is configured, the average response time and the number of timeout requests of the three load balancing strategies have increased to varying degrees. The increase ratio of
the minimum connection strategy is less than that of the other two strategies, while the response time and the number of timeout requests of the polling strategy have increased significantly, which shows that in terms of performance loss indexes, the minimum connection strategy is better than the source address hashing strategy, and the source address hashing strategy is better than the polling strategy. When the total amount of requests for the three load strategies is roughly the same, the fewer timeout requests, the better the responsiveness. Therefore, the responsiveness of the least connection strategy is better than that of the source address hashing strategy, and the responsiveness of the source address hashing strategy is better than that of the polling strategy.

As for stability, the greater the change in the resource utilization of a single server, the lower its stability. Observing Figure 2(a)(b)(c) from the above point of view, it can be found that the polling strategy has the lowest stability, and the source address hash strategy has better stability.

Figure 2(e) shows the average request delays in minutes, and sets the maximum tolerable time to 1500ms. It can be seen that the delays of the minimum connection strategy and the source address hash strategy are mostly below the maximum tolerable delay and better than the polling strategy.

| Table 2. Cloud Platform SaaS Layer Workload Balancing Performance Experiment |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|
| Strategy                        | Performance Loss | Response Performance | Resource Distribution Balance | Time Block Availability |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|
| Polling                         | 0.505687        | 0.991188        | 0.065726        | 0.171244        |
| Minimum Connection              | 0.369199        | 0.808638        | 0.036767        | 0.072329        |
| Source Address Hashing          | 0.98            | 0.43            | 0.598           | 0.635           |

The indexes of the different load balancing strategies are shown in Table 2. It can be concluded that the minimum connection strategy is superior to the other two strategy in performance loss and responsiveness. The source address hash strategy is superior to the other two strategy in Resource Distribution Balance and total time block availability. All indexes of the polling strategy are the worst.

Then, we obtain the judgment matrix R, and the consistency test is performed on it. The CR value of the matrix is 0.044157, which meets the test conditions of CR<0.1. According to formula (6), the weight vector of each index can be calculated:

\[ w = (0.096786, 0.312005, 0.162771, 0.428438)^T \] (8)

Finally, the weights are substituted into equation (7), and the comprehensive scores under the polling strategy, minimum connection strategy, and source address hash strategy are 0.303490, 0.205016, 0.219669. The smaller the index, the better the load balancing performance. We can conclude that the source address hashing strategy is better among the three load balancing strategies.

5. Conclusion

Aiming at the problems of unreasonable existing evaluation indexes and lack of comprehensive evaluation methods for cloud platform load balancing ability, this paper provides a performance evaluation method based on AHP. By analyzing the potential influencing factors of load balancing, we puts forward a more comprehensive and reasonable performance evaluation system. The experimental results show that the method is feasible, which provides a certain reference value for the further research on the performance evaluation of cloud platform load balancing.

Acknowledgments

This work is partially supported by a grant from the National Key R&D Program of China #2018YFB1003605, the National Natural Science Foundation of China #62032017, The Youth Innovation Team of Shaanxi Universities, and the Department of science and technology of Shaanxi Province.

References

[1] Mell P M, Grance T. SP 800-145. The NIST Definition of Cloud Computing[M]. National Institute
of Standards & Technology, 2011.

[2] Zhang Tao, Yan Jing, Xu Zhaomiao, Yang Yanli, Zhu Chuanxi, Cheng Jing. Research on Scalability Evaluation Method of SaaS Service in Cloud Computing Environment [J]. Journal of Northwestern Polytechnical University, 2014, 32(06): 998-1000.

[3] Li Z, O'Brien L, Cai R, et al. Towards a Taxonomy of Performance Evaluation of Commercial Cloud Services [C] CLOUD 2012. IEEE, 2013.

[4] Bardsiri A K, Hashemi S M. QoS Metrics for Cloud Computing Services Evaluation [J]. International Journal of Intelligent Systems Technologies and Applications, 2014, 6(12): 27-33.

[5] Sambit K M, Bibhudatta S, Priti P P. Load balancing in cloud computing: A big picture [J]. Journal of King Saud University - Computer and Information Sciences, 2020, 32(2).

[6] Mesbahi M R, Hashemi M, Rahmani A M. Performance evaluation and analysis of load balancing algorithms in cloud computing environments [C] Second International Conference on Web Research. IEEE, 2016.

[7] Roy S. Performance Study of Some Dynamic Load Balancing Algorithms in Cloud Computing Environment [C] SPIN 2015. IEEE, 2015.

[8] Deng Xue, Li Jiaming, Zeng Haojian, Chen Junyang, Zhao Junfeng. Analysis and Application Research on the Weight Calculation Method of AHP [J]. Mathematical Practice and Understanding, 2012, 42(07): 93-100.