ISBN 978-981-11-7861-0
Proceedings of 2018 the 8th International Workshop on Computer Science and Engineering (WCSE 2018)
Bangkok, 28-30 June, 2018, pp. 666-671
doi: 10.18178/wcse.2018.06.110

A Load Balancing Algorithm with Dynamic Adjustment of Weight

Rong He [1+] and Xinming Tan [2]

[1] School of Computer Science &Technology, Wuhan University of Technology, Wuhan, China
[2] School of Computer Science &Technology, Wuhan University of Technology, Wuhan, China

Abstract. In the distributed service cluster, the load balancing algorithm is an important means to ensure the efficient use of service calls and the rational use of resources. Through analysis of characteristics of distributed service call and defects of existing load balancing algorithms, this paper proposed a new load balancing algorithm which can dynamically adjust weight. The new algorithm combines the server performance and the connection number to express the server real-time load more accurately, uses the dynamic feedback mechanism to adjust the weights, and simplifies the request distribution algorithm, which reduces the response time of service and improves the comprehensive utilization rate of the server.

Keywords: distributed service cluster, dynamic feedback, load balancing.

1. Introduction

With the rapid development of mobile Internet and the expansion of the scale of application, connections between applications have become more and more complex. In recent years, many enterprises have used distributed service framework to decompose an application into different smaller services and do service governance.

In a distributed service cluster which based on the distributed service framework, the consumer choose an appropriate one from the service provider list to invoke, according to the load balancing algorithm in a service invocation. Load balancing algorithm is an important means to realize the balanced load in a cluster.

The commonly used load balancing algorithms in cluster are divided into two main categories: static load balancing algorithm and dynamic load balancing algorithm. In the static load balancing algorithm, such as the Random algorithm and the Weighted Round-Robin algorithm, server providers can be invoked in random or sequential way according to the server's weight. The server's load is never considered. However, in actual situations, the performance of servers is quite different. The setting of weights is greatly influenced by human factors, and the static weight values can't reflect the dynamic load and real-time processing capabilities of servers. In the dynamic load balancing algorithm, for example, the Weighted Least Connection algorithm, which based on the current ratio as the number of connections and the weights of the server to select server provider of the smallest ratio. Although it takes into account the load of the server, it is not reasonable to simply use the number of server connections to represent the server's load state.

In recent years, the research on load balancing algorithms has never stopped. Reference [1] presented a load balancing algorithm of adaptive weights, by the algorithm, the weight of the server node could update automatically when the load balancer received the task request, and the probability of distributing new request was calculated by a load difference. The algorithm realized the dynamic update of the weight, but there was a problem of too much computing overhead.

The consistent hashing algorithm was improved in [2]. The new algorithm quantified the server real-time load and compares it with the threshold of the server load, and calculated the new weight through the

[+] Corresponding author.
E-mail address: hrvae@sina.com.

formula. The algorithm realized dynamic feedback, but because it was difficult to calculate the threshold, which could cause a large error.

A transparent dynamic feedback load balancing algorithm was proposed in [3]. This algorithm introduced a load margin to dynamically adjust the load distribution of nodes, but the algorithm relied too much on the performance of load balancer itself, which was not universal.

Aiming at the merits and demerits of static and dynamic load balancing algorithms, and combining with the characteristics of distributed service invocation, in this paper, a load balancing algorithm based on dynamically adjusting weights is put forward based on the distributed service cluster constructed by Dubbo [4]. This algorithm can guarantee the weight of provider server is adjusted dynamically according to the real-time load condition, thus more reasonable to carry out the service call request distribution to ensure efficient use of resources and processing rate of the server. At the same time the algorithm simplifies the allocation request algorithm, and reduces the average response time.

2. Problem Statement

2.1. Static Load Balancing Algorithm

Taking the Weighted Round-Robin algorithm as an example, it uses the weight of the provider server to reflect the performance difference of the server. If the default weights were the same, consumers will simply poll the providers with the same weight in order of arrangement. If the weights are different, the bigger the server weight is, the better the performance will be, and then accept more new requests [5].

The characteristic of the Weighted Round-Robin algorithm is that the allocation algorithm is simple and easy to use. But in the actual situation, the server performance is different, and the weight of static configuration is greatly influenced by human factors. By default, the same weight can’t make full use of the servers with good performance, resulting in a certain waste of resources, but when the weight difference is relatively large, there is a problem of slow cumulative server requests. And the algorithm does not take into account the real-time status of the server, and the weight does not fully represent the server performance.

2.2. Dynamic Load Balancing Algorithm

Taking the weighted least connection number algorithm as an example, it selects the appropriate server to respond to a new service request according to the ratio of the current connection number to the weight of the server [6].

The characteristics of weighted least connection algorithm is that it considers the performance of the server and the connection number on the effects of the load, but the weight setting is not flexible enough. It's not enough to reflect the server's real-time load only according to the current number of servers. Once the request is received, the ratio between connection number and weight will be recalculated, resulting in a larger time cost [7].

According to the above analysis, the algorithm we design needs to meet the following goals:

To fully consider the processing capacity and real-time load status of each server.

To ensure that the system load does not tilt when running for a long time.

To minimize the complexity of the algorithm on the premise of guaranteeing the first two targets.

3. Load Balancing Algorithm with Dynamic Adjustment of Weight

According to the target, there are three key problems to be solved in the algorithm:

- How to fully consider the processing capability and real-time load status of each server and quantify them as the basis for adjusting weight?
- When should we adjust the weight value and how to adjust it?
- How to simplify the allocation algorithm?

The following solutions are proposed for these three problems.

3.1. The Comprehensive Load of the Server

The load status of the server has a great relationship with its own utilization of hardware resources, including CPU, memory, IO, network bandwidth and so on. When the hardware resource utilization rate increased to a certain extent, the request server processing speed will slow down, if the utilization rate continues to rise, the performance of the server will be greatly reduced. So the hardware utilization of the server needs to be taken into account in the algorithm.

Considering the time cost and the accuracy of the results, four representative parameters, CPU utilization ratio, memory utilization ratio, IO occupancy rate and network bandwidth utilization rate, which respectively express by C, M, I and N, are selected to express the comprehensive load of the server. The comprehensive load is the sum of the influence factors and the results of the coefficient as in Equation 1.

$$R = k1C + k2M + k3I + k4N \quad (1)$$

The sum of k1, k2, k3 and k4 is 1, and they are the weight coefficients of the corresponding parameters respectively. For different cluster systems, the importance of each weight coefficient is also different. These parameters can be set dynamically, providing some flexibility for administrators.

Because the hardware utilization rate is real-time monitoring data, so it can be used to measure the real-time load status of the server. The larger the value of the R is, the larger the current server's load is, and then the server's weight can be combined to determine the server's current ability to handle new requests.

3.2. Dynamic Adjustment of Weight

For the server's hardware configuration and performance are also important factors affecting the server's load capacity. It sets an initial weight for servers to represent them, expressed in Wi. If the configuration and performance of the server are better, the server's processing power is stronger, which means that it can respond more calls requests, so the value of Wi is bigger. In the process of cluster service invocation, the load balancing algorithm adjusts the weight dynamically according to the real-time load of the server with the change of the current connection number of the server.

Suppose the number of the provider servers is n, and at some point, the weight of the No.i server is expressed in Wi, and the number of service connections is expressed in Li. Calculate the sum of the weights of all servers, as in Equation 2.

$$W(\text{sum}) = \sum_{1}^{n} Wi \quad (2)$$

Calculate the number of connections, as in Equation 3.

$$L(\text{sum}) = \sum_{1}^{n} Li \quad (3)$$

If the judgment condition of the update weight is satisfied, the new weight is expressed by W'i. Calculate the ratio of the number of connections to the weight of the No.i server, as in Equation 4.

$$LW\text{i} = \frac{Li \times W(sum)}{L(sum) \times Wi} \quad (4)$$

And the average of the average ratio of all servers is expressed in A_LW, as in Equation 5.

$$A\_LW = \frac{\sum LWi}{n} \quad (5)$$

- LWi>A_LW indicates that the load of the No.i server is heavier. It needs to reduce the weight of the No.i server appropriately, which will reduce the subsequent requests and avoid the service delay caused by the slow response time of the server.
- LWi<A_LW indicates that the load of the No.i server is lighte. It can increase the weight of the No.i server appropriately, to share more calls requests, which will improve the resource utilization of the current server.

- LWi=A_LW indicates that the current load of the No.i server is more balanced, so the weight does not need to be changed.

The dynamic weight of the server is updated according to the current weight and current connection number of the server, as in Equation 6.

$$W'i = Wi - \left(1 - \frac{LWi}{A\_LW}\right) \tag{6}$$

.

3.3. Load Balancing Algorithm with Dynamic Adjustment of Weight

Suppose the number of the provider servers is n, and the weight of the No.i server is expressed in Wi, the new weight is expressed by W'i. At some point, the number of service connections is expressed in Li, the maximum value of Li is expressed by Li_{max}, the comprehensive load of the No.i server is expressed by Ri, the sum of all server's comprehensive load is expressed by R(sum), as in Equation 7.

$$R(\text{sum}) = \sum_{1}^{n} Ri \tag{7}$$

.

The sum of all server's weight is expressed by W(sum), as in Equation 2.

(1) Considering the hardware configuration of the server, an initial weight is set for the server. The better the server hardware configuration is, the better the server performance is, and the higher the initial weight is.

(2) When the new connection request arrives, the provider server is polled according to the initial weight. If the ratio of the No.i server is smaller than the ratio of the total, as in

$$\frac{Ri}{Wi} \le \alpha \frac{R(sum)}{W(sum)} \tag{8}$$

.

It indicates that the No.i server is small in load and has great weight and good performance, so the new request is sent to the server. The alpha is the adjustable coefficient, and its value is not less than 0 and less than 1, and it can be constantly modified to find the optimal solution according to the demand.

(3) If all the servers in the cluster can't meet the above situation, then we need to consider changing the weight. Firstly, we compare the ratio of the connection number and the weight of the No.i server with the average of all servers. The former is expressed in LWi, and the latter is expressed in A_LW, as in Equation 5.

If LWi is more than A_LW, it needs to reduce the weight of the No.i server appropriately. On the other hand, the weight of the No.i server needs to be properly added. The new weight is updated, as in Equation 6.

(4) After the weight updates, polling access to the No.j server continues. If the comprehensive load of the server is the smallest, as in Equation 9.

$$\frac{Rj}{Wj} = \min\left\{\frac{Ri}{Wi}\right\} \tag{9}$$

.

It means that the ratio of the load and the weight of the No.j server is the smallest, then the new request is sent to the No.j server.

(5) If the current number of connections is less than its threshold, which indicates the No.i server is in the overloading state, so the number of the provider servers should be increased.

3.4. Analysis of Algorithm

In the load balancing algorithm with dynamic adjustment of weight, the comprehensive load of the server takes into account a number of server performance indicators. Based on the real-time hardware usage and real-time load of the server, the algorithm constantly adjusts the proportion of server weight to the number of service requests. It avoids the unfair distribution of requests caused by improper weight settings, and makes effective use of server resources, so as to improve the quality of system service governance. The flow chart of the load balancing algorithm with dynamic adjustment of weight is shown in Figure 1.

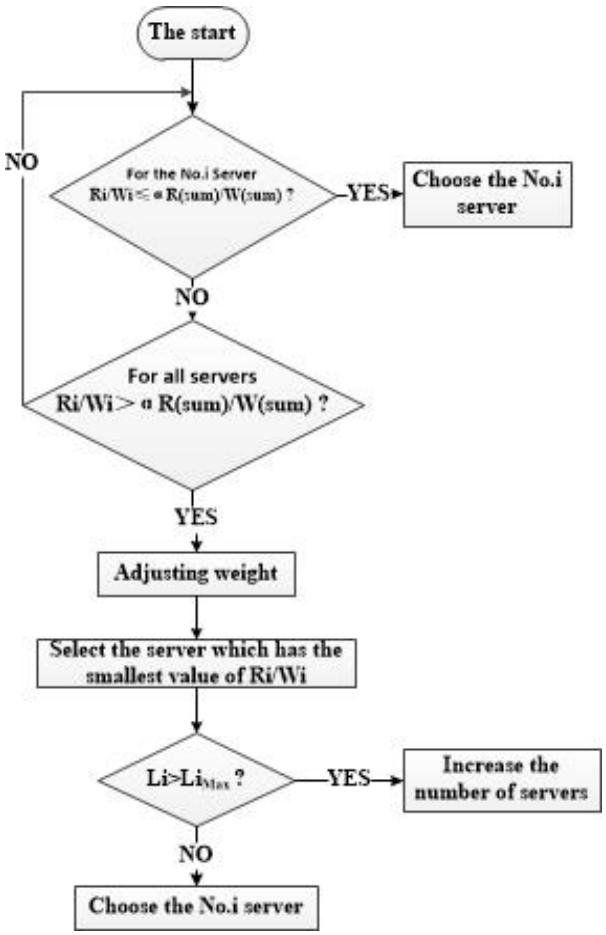

Fig. 1: The process of algorithm

4. Experiment and Result Analysis

In order to verify the effectiveness of the algorithm, we use the provider server A and B to form a small server cluster, where the memory of server B is 8GB, and the memory of server A is 4GB, which has great difference in performance. Set the same initial weight for two providers, provide the same service name, and use the single zookeeper deployed on server A to serve as the service registry center. The test environment configuration is shown in Table1.

Table 1: Configuration information for testing

Role	Server A	Server B	Service consumers
CPU	Intel Xeon E5-2680 CPU @2.50GHz	Intel Xeon E5-2680 CPU @2.50GHz	Intel(R) Core(TM) i5-2450M CPU @2.50GHz
Memory	4.00GB	8.00GB	4.00GB
System	Windows Server 2008 R2 64bit	Windows Server 2008 R2 64bit	Windows 10 Pro 64bit
Version of Dubbo	Dubbo-2.5.8, DubboMonitor-2.5.8, Zookeeper-3.4.9		
Java Environment	jdk1.8.0_121 64bit	jdk1.8.0_121 64bit	jdk1.8.0_121 64bit

When the experiment is carried out, the parameters used for calculating the comprehensive load of the server are all 0.25, indicating that the importance of CPU utilization, IO utilization ratio and other factors is equal, so the value is equally divided. The values of these parameters can also be adjusted according to the actual needs. In the experiment, we perform the 40 thousand simulation concurrent request test on the Random algorithm, the Weighted Round-Robin algorithm, the Weighted Least Connection algorithm and the load balancing algorithm with dynamic adjustment of weight. The number of requests and response time processed by each provider is recorded, as in Table 2

Table 2: Experimental result

Algorithm	The number of requests processed		Total response time (SEC)	Average response time (millisecond)
	Server A	Server B		
Random	19943	20057	182.81	4.5703
Round-Robin	20000	20000	175.21	4.3803
Least Connection	19768	20232	177.65	4.4413
DAW	15024	24976	146.93	3.6733

According to the statistical results, we can see that:

In the Random algorithm, the Weighted Round-Robin algorithm, and the Weighted Least Connection algorithm, the weight is a fixed value, and also the most important factor that affects the distribution of service requests. However, in the case of distinct performance differences between server A and server B in a cluster, the number of responses to requests is roughly the same for each server.

In the DAW algorithm, the number of responses to requests of server B is more than server A obviously, this is because the DAW algorithm adjusts weight according to the real-time status of the provider server constantly, the weight can always reflect real-time load status, so server B of better performance can handle more requests within a certain range, and better reflect its performance advantages, the efficiency of the whole system is also improved. Compared with the first three algorithms, the DAW algorithm also significantly reduces the average response time.

5. Conclusion

Compared with the Random algorithm, the Weighted Round-Robin algorithm, and the Weighted Least Connection algorithm, the DAW algorithm proposed in this paper is more suitable for a distributed cluster with distinct differences in the processing capabilities of the provider servers. The DAW algorithm measures the load status of the provider server based on many factors, and adjusts the server's weight dynamically according to the real-time number of connections and hardware utilization rate. In this way, the server is not limited by static weight, and can be fully utilized regardless of its performance. Ultimately, it reduces the average response time of the server and improves the comprehensive utilization of the server.

6. References

[1] Wu, Ying, S. Luo, and Q. Li. "An Adaptive Weighted Least-Load Balancing Algorithm Based on Server Cluster." International Conference on Intelligent Human-Machine Systems and Cybernetics IEEE, 2013:224-227.

[2] Kun, L. I. "Server Load Balancing Algorithm Based on Dynamic Feedback." Electronic Science & Technology (2015).

[3] GENG Qiang, and HUANG Xue-qin. "An Algorithm of Load Balancing Based on Adaptive Weight." Science Technology and Engineering (2013).

[4] Information on http://dubbo.io/

[5] Hsiao, Hung Chang, et al. "Load Balance with Imperfect Information in Structured Peer-to-Peer Systems." IEEE Transactions on Parallel & Distributed Systems 22.4(2011):634-649.

[6] ZHOU Song-quan. "An Improved Dynamic Load-balancing Algorithm for Cluster." Computer and Modernization (2012).

[7] Mangipudi, Krishna, and V. Basani. "Method and apparatus for policy based class service and adaptive service level management within the context of an internet and intranet." US, US 7124188 B2. 2006.