

The Research on Load Balancing of Middleware based on Cloud Computing

Wenlong Feng^{1, a}, Mengxin Huang^{1, b}, Yu Zhang^{1, c}

¹Institute of Information, HaiNan University, Haikou 570206, China

^aemail:fwlfwl@163.com, ^bemail:huangmx09@163.com, ^cemail:yuzhang_nwpu@163.com

Keywords: Cloud Computing; Service; Middleware; Load Balancing; Node

Abstract. According to the characteristics of multi level and large amount of access of cloud computing services, a load balancing mechanism based on middleware is proposed. The mechanism analyses distributed structure of cloud computing services, and determines the topology and control method of load balancing. It also designs management function model and load threshold of middleware node, sets a communication mechanism between the management node and working node, and implements the load balance of the middleware. Simulation results show that the mechanism can do the load balancing of middleware in real time according to the service access amount, and improves the reliability of the cloud computing services.

Introduction

With the development of computer technology and the popularity of the network, more and more electronic commerce services, smart tourism services and other complex services are based on the cloud platform. Complex services can make use of the resources of the cloud computing platform, and has a good scalability and cross platform. It can reduce development costs, reduce technical barriers and help service maintenance. The application of complex services based on cloud computing environment is becoming more and more extensive.

Compared with traditional services, cloud computing services have significant characteristics. Traditional services tend to focus on a particular area of business, and its functions are often have single and limited. Traditional services are often designed, developed, debugged and deployed by the same organization, and the load balancing can use the same strategy in the display, logic and data layer. As the developer of traditional service is fully understood middleware design of the entire service, so it easy to achieve load balancing [1]. Cloud computing services have complex functions, multiple levels and the large amount of access, and often include a number of service middleware which developed by a number of services provided. These middleware of cloud computing service are belonging to different organizational units, deployed in different geographic locations. For example, the smart travel services provided by the cloud platform consists of traffic, hotels, attractions and other sub services, and these sub services are implemented by the middleware. In addition to the traditional load balancing technology, the load balancing of cloud should give more consideration to the heterogeneity of middleware and cross geographical features. The load balancing implementation of loud computing service middleware is the basis for the normal and reliable operation of cloud computing services.

The Strategy of Middleware Load Balancing

The middleware load balancing strategy includes two aspects: control topology structure and load balancing level granularity. The traditional control topology is centralized, and the load balancing is done by a server. This strategy can not adapt to the large number of nodes and the application services across the region. Load balancing level granularity is mainly considered to achieve load balancing in which level, the smaller the size, the better the load balancing control.

The control topology structure.

A user gives service application to cloud platform and cloud platform find the appropriate nodes according to user needs. Due to the number of nodes is lager and these nodes are deployed in

different geographic locations, so the traditional centralized structure can not be applied to cloud computing services. In this paper, we propose a design scheme of node topology based on geographic location: firstly, all nodes are divided into the different region. Second: each region has a management node, which is responsible for the management of all nodes in the region. Third: the management node should communicate with management nodes of the adjacent region and timely update the load information of nodes [2]. Topology structure is shown in figure1.

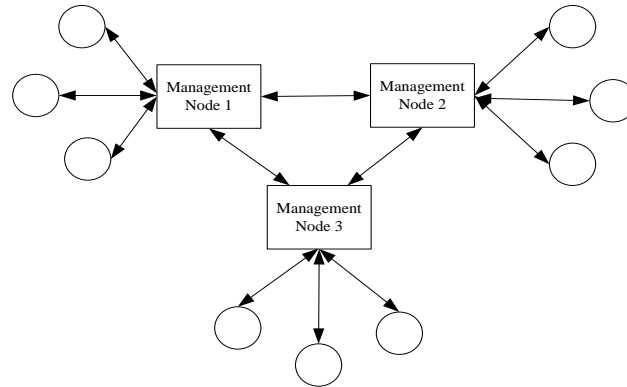


Fig 1 The control topology structure

This load balancing control topology is a distributed and centralized integrated application, which has the advantages of both centralized and distributed structure. It is suitable for the complex service based on cloud computing [3].

The load balancing level.

Load balancing adopts two layers of balanced granularity: load balancing of different middleware on the same node and load balancing of middleware on different nodes [4].

(1) The load balancing of different middleware in the same node

Working nodes of cloud platform is constructed and maintained by cloud computing organization and middleware service providers only rents part of resources of nodes, and many middleware which provide different functions are often deployed in the same node. These middleware can be independent or be with constraint relations. Middleware is usually in form of instances for invocation of the upper service. Due to the limited capacity of the nodes, the number of middleware instances for a node exist a threshold. When exceeding the threshold, nodes performance significantly decrease, so configuration strategy for the number of various middleware instances is a key to load balancing. Configuration strategy includes two aspects: node threshold and middleware threshold; static configuration and dynamic configuration [5].

The node threshold: when the number of instances of all middleware exceeds a certain value, the node sends an alert to the management node, and no longer accepts the new service request. This value is called the node threshold, and its size is generally 80% of the total number of all middleware instances which is configured by node configuration strategy.

The middleware threshold: when the number of instances of a middleware exceeds a value, it indicates that the middleware is in a busy state, and no longer accepts the new call request, so the value is called the middleware threshold.

The static configuration: it is adapt to the new deployed middleware, and the number of the new middleware instance is determined by the node capacity and middleware services QoS. The configuration strategy is as follows: there is the same range of value for all new middleware, and the range is determined by the node capacity. The QOS detail describes parameters of middleware such as functions, frequency, throughput, delay, etc, so node can determine the number of instances according to the QoS. For example, the range of value of a node for all new middleware is 30-50, and middleware A and B are new middleware. The number of A and B can be set as 45 and 35 respectively due to their different QoS. The static configuration defines the number of instances mainly based on QoS of middleware, so this strategy exists some defects: first, because the QoS is provided by the service provider, and service provider's reputation are different; second, the cloud

service composed by multiple middleware accomplishes the function, and a middleware can participate in multiple cloud services, then relationship between cloud services and middleware is a many to many. Middleware called frequency change frequently. The static configuration can not reflect the dynamic changes of middleware [6].

The dynamic configuration: according to the node's running situation, the number of instances of middleware in the node is adjusted to improve the service quality and the utilization of resources. This adjustment includes two kinds: one is the cyclical adjustment, and the other is the temporary adjustment. The cyclical adjustment is to adjust the number of instances based on the nodes in a period of time. It is based on historical data, and the more historical data, the more accurate the adjustment. The cyclical adjustment is shown as the following process:

- 1) The node statistics the number of call times of each middleware and the total number of calls times of middleware, and calculates the proportion of calls times of each middleware in all calls times.

- 2) The node gets the maximum value of the middleware instances that can be configured by the node, and then determine the number of instances according to the ratio of the various middleware calls.

For example, the maximum number of all middleware instances for a node is 1200, and there are three middleware as A, B and C deployed in the node. If calls times of A, B and C are 100, 200, and 300 respectively in a period, then call ratio of A, B and C are $1/6$, $2/6$ and $3/6$, so the number of instances of the A is 200, the number of instances of the B is 400, the number of instances of the C is 600. The sampling period is chosen according to the node operation and service utilization. Nodes and service running is busier, the sampling period is shorter, so as to better reflect the real-time changes of nodes. Nodes and service running is idle, which shows that the middleware is not changed, and the sampling period can be longer.

The temporary adjustment is done in a specific time or in the emergency case. For example, access to a middleware A suddenly increases in a time and A already is in busy state, so the node does treatment in accordance with the following process: first checks to see working state of other middleware, selects the middleware which is the most idle, reduce the number of instances of the middleware, then increases the number of instances of A. If other middleware nodes are busy, the nodes will not be adjusted. Temporary adjustment is not a long term, when the special moment is over, the node will restore the number of instances of all middleware based on the cyclical adjustment.

- (2) The load balancing of middleware in the different node

The significant characteristics of cloud services are the large amount of access and the distribution of nodes s. Such as Taobao, daily user visits is up to millions. The user is distributed over the country, and service nodes also are distributed in all parts of the country, so the load balancing between service nodes is the basis of the reliable operation of the service.

Node is divided into service nodes and management nodes. The management node storages two aspects of information: one is the load information of all nodes in the region, including the working state of nodes and the calls times of middleware; the other is the load information of all nodes in the adjacent region through communicating with adjacent management nodes. Due to the adoption of the centralized and distributed control mode, the regional division is key to load balancing between service nodes. If division of region is reasonable, it can reduce the response time of service and improve service reliability. The division of region follows the principle: first, nodes which is adjacent geographical location is as far as possible in the same region, so that it can reduce the network transmission time, reduce the service response time. Second, if there is a large amount of access in some region, it should set up the more management nodes, overcome the bottleneck problem of the management node, and reduce the load of the single node. The working procedure is as follows:

- 1) Users submit service applications, and cloud platform determines the user's location, knows the middleware category of services required. Then the user's request is sent to the management node.

2) The management node sees load information of all nodes in this area, considers the busy degree of the middleware and all nodes, selects the most appropriate working node and then feedback it to the user.

3) If all nodes are full load or no middleware is suited to users' demand in the region, the management node will communicate with adjacent management node, forwards user's demand, then the adjacent management node begins to search until it find the right node.

The Load Balancing Design of Middleware

The design of management node.

The design of management node is the key problem to realize the load balancing of the middleware. The management node should achieve two main functions: first, it accepts the service application, analyzes the middleware used for the composition of services; second, it selects the appropriate working node, and achieves the load balancing of middleware. The function of management node includes the following modules: service demand analysis module, data acquisition module, demand forwarding module and node selection module [7].

The service demand analysis module: receiving user's service application, judging the user's area, analyzing the number and type of middleware required by services.

The Data acquisition module: communicating with all working nodes in the region periodically, getting the load status of all nodes in the region; communicating with all adjacent management nodes periodically, getting the load status of all working nodes in all adjacent areas.

The demand forwarding module: when all nodes in the area do not meet the service needs of the working node, the management node will forward the service needs to the appropriate adjacent management node in accordance with the load state of nodes in all neighboring regions

The node selection module: the management node or adjacent management node selects the appropriate working node according to the selection strategy.

The function of management node module is shown in figure2:

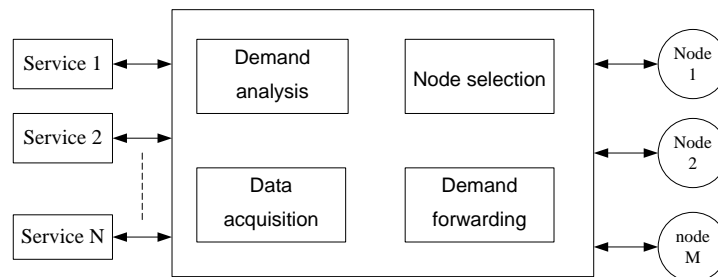


Fig 2 The management node function structure

The node selection process is as follows:

(1) If there are some middleware that meet the needs of the service in this area, and load of these middleware has no more than the load threshold value, the working node selects the middleware that its load is minimal.

(2) If there is no suitable working node in this area, the management node forwards service needs to the adjacent management node which continues to find the most suitable nodes in the adjacent area.

(3) If there are some appropriate working nodes in multiple adjacent areas, the system selects the appropriate working node which has the fastest response time.

The management node achieves also the management of the node in addition to complete the load balancing function. Because the service was established in the cloud computing platform, working nodes will do some operations often such as add, delete and modify, so the management node must grasp working node state in the region. It is completed mainly through regular communication mechanism: within the specified time all nodes must communicate with the management node, report their working state. The management node does not receive the

information of a node over a specified period of time, then the node is labeled as not available, and the management node does not select the node [8].

The design of working node.

The working node mainly has two functions: the first is the service function that is completed through the deployed middleware; the second is the monitoring function. First of all, it monitors the middleware, timely reflects the load changes, and masters the real-time performance of the middleware. Secondly, when the middleware load is too heavy, causes its performance reduced, it is able to timely alarm, prompts the server to optimize the resources, and adjusts the relevant configuration.

The monitoring model consists of four modules: data acquisition module, load algorithm module, performance evaluation module and alarm module.

The data acquisition module: it collects information of all nodes deployed middleware, including middleware name, caller name, the number of middleware instance, middleware and calling method. These data are periodically collected, reflecting the status of the load change of middleware in time.

The load algorithm module: according to information of middleware supplied by the data acquisition module, it calls the load algorithm, and then can get the specific middleware load. The load algorithm uses two layers of granularity: middleware busy degree algorithm and middleware method busy degree algorithm, so it can more accurately reflect the real load status of all kinds of middleware.

Performance evaluation module: according to information of the specific middleware and load conditions, calculates middleware performance indicators such as response time and throughput. Due to functions of the middleware are different, and the performance index is different, so it should do the specific analysis of SLA of middleware.

The alarm module: When the performance of the middleware is significantly decreased and can not meet the SLA requirements, the alarm module can make alarm and no longer accepts new service applications.

The monitoring function of working node is shown in figure3:

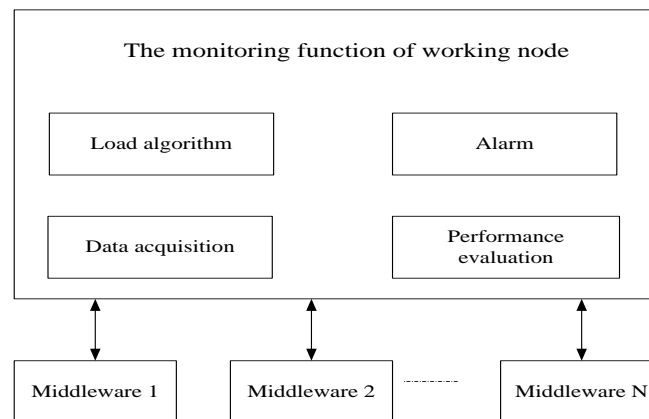


Fig 3 The monitoring model structure

Simulation Results

The load balancing of middleware proposed by this paper has been applied in the smart tourism platform based on cloud computing environment developed by Hainan University. The platform based on cloud computing technology; data components are from domestic tourism companies, and business components such as route customization, personalized recommendation are developed by the Hainan University. The load balancing of middleware is simulated on the platform Simulation results are as follows:

1) when access to the single service is relatively small, the service response time is basically consistent with the traditional load balancing mechanism. With the increase of the load, the load

balancing mechanism in this paper is better than the traditional mechanism, the greater the load, the more obvious the advantage of the mechanism.

2) when access to a variety of services, due to the load balancing mechanism of this paper adopts a combination of static and dynamic configuration, service response time is better than the traditional mechanism, the greater the load, the advantage is more obvious.

Conclusion

The load balancing of middleware based on cloud computing environment overcomes the disadvantage of the traditional load balancing that load grain is rough. It considers characteristics of multi-level services, large amount of access and foreign distribution of nodes, and discusses load balancing of middleware in a node and more nodes. The mechanism adopts the combination of centralized and distributed control structure, carries region management of nodes, and introduces the concept of static allocation and dynamic configuration. The mechanism can reflect the state of load of nodes and a variety of middleware in time, achieve load balancing of middleware services, and ensure the normal operation of the service.

Acknowledgements

This work was financially supported by the Foundation projects: the natural science foundation of Hainan Province (project number: 614232) and National Science-technology Support Plan (project number: 2015BAH55F01).

Reference

- [1] LI XUE MING, CHEN LI LI. The research on security of Web service composition[J]. Application and Research of computer, 2009,26(4):1523-1524.
- [2] YAO JING, HE JU HOU. The cloud computing load balancing mechanism based on adaptive bee colony algorithm[J]. Computer application, 2012,32(9):2448—2450
- [3] MONDAL B, DASGUPTA K, DUTTA P. Load balancing in cloud computing using stochastic hill climbing—a soft computing[C] // C3IT-2012: Proceedings of the 2nd International Conference on Computer, Communication, Control and Information Technology. New York: Elsevier, 2012:783 —789.
- [4] YU GUO FU, LI HONG JIAN, TANG HONG. The research on dynamic load balancing algorithm based on feedback mechanism [J]. Computer application research, 2012,29(2):527—529.
- [5] LIU ZHI JIA. Research on load balancing technology based on Cloud Computing [J]. Journal of Guangxi Teachers Education University(Natural Science Edition), 2011;28(2):94-96
- [6] ZHU JIA YU, XIAO DAN, WANG FEI. Multi—dimensional QoS constrained scheduling mechanism based on load balancing for cloud computing [J]. Computer Engineering and Applications, 2013,49(9):85-86
- [7] Sadhasivam S, Jayarani R. Design and implementation of an efficient two-level scheduler for cloud computing environment[C] // International Conference on Advances in Recent Technologies in Communication and Computing. Kottayam , Kerala:[s.n.], 2009:884-886.
- [8] HUA XIA YU, ZHENG JUN, HU WEN XIN. An ant colony optimization algorithm based on cloud computing environment[J]. Journal of East China Normal University(Natural Science Edition), 2010(1):127-134