ASIAN JOURNAL OF PHARMACEUTICAL AND CLINICAL RESEARCH

INNOVARE ACADEMIC SCIENCES
Knowledge to Innovation

Special Issue (April)

Online - 2455-3891
Print - 0974-2441

Full Proceeding Paper

ENERGY EFFICIENT LOAD BALANCING FOR CLOUD DATA CENTER

AMEY RIVANKAR, ANUSOOYA G

School of Computing Science and Engineering, VIT University, Chennai, Tamil Nadu, India. Email:asha.s@vit.ac.in

asha.s@vit.ac.in Received: 23 January 2017, Revised and Accepted: 03 March 2017

ABSTRACT

Cloud computing is the latest trend in large-scale distributed computing. It provides diverse services on demand to distributive resources such as servers, software, and databases. One of the challenging problems in cloud data centers is to manage the load of different reconfigurable virtual machines over one another. Thus, in the near future of cloud computing field, providing a mechanism for efficient resource management will be very significant. Many load balancing algorithms have been already implemented and executed to manage the resources efficiently and adequately. The objective of this paper is to analyze shortcomings of existing algorithms and implement a new algorithm which will give optimized load balancing result.

Keywords: Cloud computing, Load balancing, OpenStack.

© 2017 The Authors. Published by Innovare Academic Sciences Pvt Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4. 0/) DOI: http://dx.doi.org/10.22159/ajpcr.2017.v10s1.19604

INTRODUCTION

Cloud computing has emerged as one of the premier technologies that has revolutionized the way services are made available to users using cloud data centers. Cloud computing has driven the creation of data centers made up of more than thousand servers that are capable of supporting a large number of services. The cost of maintaining this cloud data centers is extremely high. A recent study has shown that power and cooling costs of data centers have increased immensely by 800% since 1992 [1]. Hence, we need to introduce load balancing techniques that efficiently utilize the servers and in turn help in energy efficiency.

Load balancing satisfies two important needs, first to utilize servers efficiently and second to promote performance. In order for load balancing to work it should also satisfy below major goals:

a. Cost effective: System performance is improved at an economical cost.
b. Scalability: The distributed framework in which the algorithm is actualized may change in size or topology. Hence, the algorithm must be sufficiently versatile to permit such changes to be taken care at ease.
c. Priority: Jobs need to be handled based on priority.

Data centers are one of the major backbones of cloud computing. Clouds span over the data centers and work with the notion of location independence. Cloud computing achieves its two property of reliability and redundancy using multiple data centers. Another important supporting feature for cloud computing is its network infrastructure. With proper networking strategies, a way of green computing can be achieved. Even though cloud does not care about a particular locality, it gives importance to relative network locality and topology. Cloud load balancing is used in the cloud computing environment to distribute loads among different computing resources. Load balancing is having greater importance in cloud data centers as it dynamically allocates the workloads among the data centers and thereby meets the increased service demands.

Benefits of load balancing

1. Redundancy: It depicts the way of running at least two servers thus giving an ensured event that one of the servers becomes occupied.
2. Scalability: Despite the fact that modest resources are given, correct host solution should be found for request handling.
3. Resource optimization: Using load balancing, one can enhance how network traffic is circulated to the server group which in turn ensures the best performance.

CLOUD SERVICE MODELS

Cloud service delivery is isolated into three models. These models have been designed and developed based on varied requirements of the user. The three cloud service models are defined as follows:

1. Cloud software as a service: This service allows the consumer to use the provider's applications running on a cloud infrastructure. The applications are available for different client devices through a thin client interface such as a web browser.
2. Cloud platform as a service: This service allows the consumer to deploy onto the cloud infrastructure his own applications created using programing languages and tools which are supported by the provider. The provider controls and manages all the underlying cloud infrastructure, while the consumer controls the deployed applications.
3. Cloud infrastructure as a service: This service allows the consumer to obtain all the underlying computer resources such as processor, storage, networks, and memory where he can deploy operating system and run arbitrary software. Here, the consumer has control over the infrastructure.

LOAD BALANCING ALGORITHM FOR CLOUD COMPUTING

1. Ant colony optimization: Is an algorithm which makes use of ant's movements to determine the shortest path while selecting the server. Marco Dorigo first introduced the Ant Systems (AS) in 1992. IT is one of the best optimization techniques, which finds the shortest path. Ants use a chemical called as pheromone which is used for communication. Similar idea is used in the load balancing concept.
2. Round robin: This is one of the most used algorithms for load balancing. In this algorithm, a quantum of time is allocated to a system, so when a new request is coming to a server, the master node selects the system from a pool of systems and allocates the incoming request to that system. Once, it has used its speculated quantum of time the next system is allocated the request. This goes on till all the incoming requests have been handled. The algorithm makes efficient use of system resources.
3. Particle swarm optimization: Is a self-versatile worldwide search based optimization technique. The algorithm is like other population-based algorithms like genetic algorithms, but there is no immediate recombination of individuals of the population. Rather, it depends on

the social conduct of the particles. In each era, each particle adjusts its trajectory based on its best position (local best) and the position of the best particle (worldwide best) of the entire population. This idea increases the stochastic way of the particle and converge rapidly to global minima with a sensible good solution.

SOLUTION METHODOLOGY

To implement a new algorithm, a comparative study needs to be carried out between existing algorithms. This can be done by making use of cloud analyst tool or by creating server instances in OpenStack. Once this is carried out, we can then intend to create a new algorithm for load balancing which would inherit features of existing algorithms.

LOUD ANALYST

Cloud analyst [9] is a graphical user interface-based software that is developed on CloudSim architecture. CloudSim allows us to experiment with reenactment of distinct load balancing algorithms. It can likewise be utilized for showing, recreation, and other experimentation. The essential issue with CloudSim is that all the work should be done automatically. It permits the client to do rehashed recreations with slight change in parameters effortlessly and rapidly. The cloud analyst allows the user to setup user bases that are generating the application request and also the location of the data centers that handle this request. While configuring many parameters such as number of clients, number of virtual machines (VMs), number of processors, amount of storage, system data transmission, and other essential parameters can be set. This tool then provides the reenactment result and demonstrates them in graphical structure. The outcome is provided in graphical form containing reaction time, handling time, cost, etc.

Cloud analyst has following preconfigured load balancing algorithms, which user can use (Fig. 1):

1. Round robin
2. Equally, spread current execution load
3. Throttled.

OPENSTACK

OpenStack is an open source platform for building and overseeing distributed computing stages for open and private clouds. OpenStack is managed by the OpenStack foundation, a non-benefit which directs both improvement and community-building around the task. Keeping in mind the end goal to execute our situation, we make utilization of DevStack. DevStack is a tool for deploying OpenStack in Ubuntu. In the wake of introducing DevStack on the VM, we need to make cases which will mean the servers of the constant environment. Furthermore, we have to make use of "HAProxy" which is a reliable, high-performance TCP/HTTP Load balancer. This will enable us to try different load balancing algorithms and hence check their performance.

OpenStack has many distributors like:

1. Platform 9,
2. Ubuntu OpenStack,
3. Mirantis OpenStack,
4. VMware integrated OpenStack,
5. HPE Helion OpenStack,
6. IBM Cloud Manager with OpenStack,
7. Red Hat OpenStack.

OpenStack has following preconfigured load balancing algorithms, which user can use and play with (Fig. 2):

1. Round robin
2. Least connections
3. Source IP.

PROPOSED IDEA

In this paper, we would like to study existing load balancing algorithms provided by OpenStack. We need to setup up a pool of instances which will handle the requests. The algorithm will be tested based on the response time and the power utilized to implement a given task. This will be tested using a software named powertop. Next, we will carry out the above experiment with other existing algorithms but which are not in OpenStack. On the basis of these results, we will try to implement a new algorithm which will try to provide better and energy efficiency.

SYSTEM SETUP

The OpenStack can be installed on a single- or multi-node setups. The multi-node setup requires proper network configuration and usually requires two network interface cards in a single system to allow inter-node communication. Also in this setup, all the components are installed on different physical systems. In a single-node setup, all the OpenStack components are installed on a single system. Above OpenStack installation can also be done in the virtualized environment making use of software such as Oracle VirtualBox and VMware workstation.

Below mentioned are the minute system configuration required for deployment of OpenStack:

6 GB RAM, +2.5 GHz processor, 50 GB storage.

We have made use of Red Hat Enterprise Linux distribution of OpenStack (RDO) for installation of OpenStack in an Oracle VirtualBox.

Fig. 1: Cloud analyst graphical user interface

The physical system configuration is as follows:

32 GB RAM, octacore processor, 1 TB storage.

Oracle VirtualBox provides the user with numerous networking settings that you can add to your VM. Providing information on this networking settings is very important to set up your OpenStack. Some of this are mentioned below:

1. Network address translation (NAT): Is the simplest option from the perspective of the guest system for accessing external networks. External access to the guest system is not possible. If we use this setup for OpenStack than we won't be able to access the OpenStack from the host as well as other VM's.
2. Host only networks: Is used to create a network which allows for the communication between the host and the other VM's. However, the VM's cannot connect to the external network, hence not feasible for OpenStack install as it requires packages to be downloaded from the internet.
3. Bridged: Your guests will get an IP address on the same subnet as your host. We use this if you are running servers on the guest and wish to connect from other computers on the local area network.
4. Internal networking: Is used to setup a common network between the guests only. Communication with the host system or another network outside of VirtualBox is not possible while using this.
5. NAT networking: Is the combination of NAT and internal networking. This network allows for the internet access obtained to the host as well as the access within the VM's connected to this network. Direct communication between the host and VM is not allowed, so we have to make use of port forwarding. This also allows you to make use of static IPs.

We have made use of NAT networking for the OpenStack installation setup as shown below (Fig. 3) with the port forwarding (Fig. 4).

Once this configuration setting is done, we can go forward with OS install and finally with the OpenStack install. We have made use of CentOS 7 minimal version for this install.

We made use of RDO packstack, which is a utility to install OpenStack in Red Hat Linux based OS. There are three ways to make use of packstack for OpenStack installation:

1. All-in-one: This will install all OpenStack services on a single host without prompting for any configuration information.
2. Answer file: This will make use of an answer file which will contain all the configuration settings required for OpenStack install. The answer file contains set of yes/no options.
3. Prompts: In this, you simply use packstack command which initiates the install, and asks for user inputs when trying to install certain components.

If everything was set up correctly you will be provided with successful completed message as shown (Fig. 5).

Next you can browse to the OpenStack dashboard URL, to use the OpenStack features such as instance creation, network pool and load balancer pool. OpenStack is developed using python and HTML, so one can easily go about the code and make modifications as required.

To make a power comparison test between existing load balancing algorithms, first, we need to create instances and install on it a software named powertop. This will analyze the current power consumption of that instance. This powertop data has to be fetched back to the OpenStack running machine and a comparison is to be made.

Apart from this, our next aim is to add a new load balancing algorithm into the OpenStack environment and compare it existing algorithms.

Fig. 2: OpenStack login interface

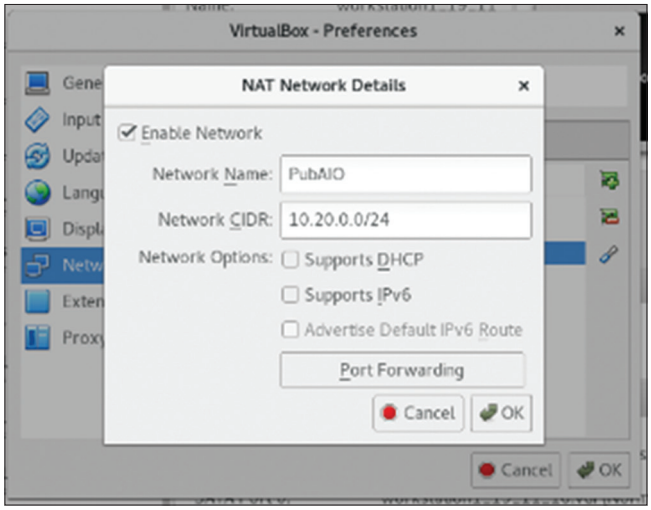

Fig. 3: Network address translation network setup

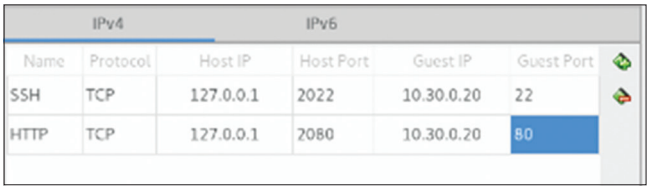

Fig. 4: Port forwarding in VirtualBox

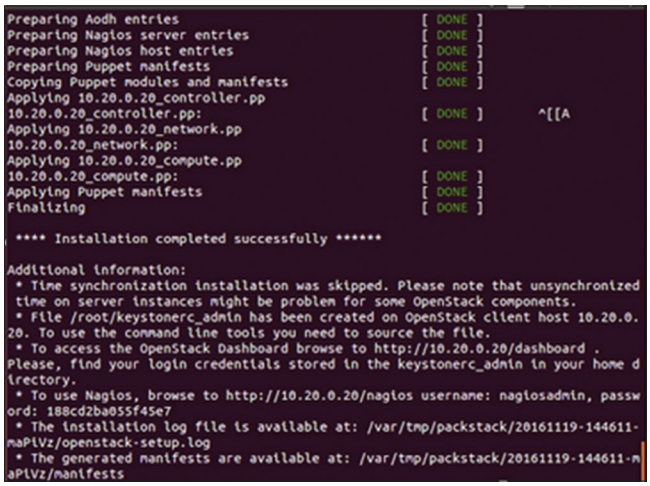

Fig. 5: OpenStack installation completed

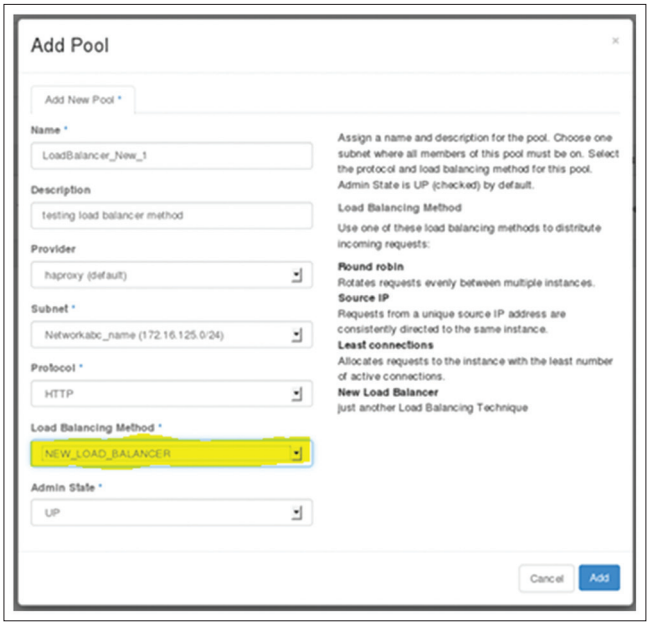

Fig. 6: User interface changes to add new option in dropdown

To make UI changes, we had to make modifications to the following files in the OpenStack directory, "/usr/share/OpenStack-dashboard/OpenStack_dashboard/dashboards/project/loadbalancers/"

1. Forms.py
2. Workflow.py
3. Tables.py
4. Create_pool_help.py.

This above changes allowed us to add the UI changes in the OpenStack dashboard as shown (Fig. 6). We need to make further changes which will allow us to add the actual algorithm file in the setup, which can be in turn used and tested with the other existing algorithms.

CONCLUSION

From the study done on these papers, it can be concluded that developing an efficient load balancing algorithm is of much importance and using OpenStack is the modest idea. It has been used by many companies to implement their cloud and thus provides future scope. It is also very easy to implement and make modifications as required. The introduction of proposed idea will certainly reduce company cost in terms of power utilization.

FUTURE SCOPE

In this paper, we have proposed a new load balancing algorithm to be tested in OpenStack for energy efficiency. By analyzing the cited parameters, we can further enhance the load balancing study and find the algorithm for better resource optimization. The future work includes overcoming the problem of deadlocks and server overflow and to further optimize the devised technique. We can also implement a new service broker policy in the simulator.

REFERENCES

1. Cao J, Li K, Stojmenovic I. Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers. IEEE Trans Comput 2014;63(1):45-58.
2. Mohapatra S, Rekha KS, Mohanty S. A comparison of four popular heuristics for load balancing of virtual machines in cloud computing. Int J Comput Appl 2013;68(6):33-8.
3. Lamba S, Kumar D. A comparative study on load balancing algorithms with different service broker policies in cloud computing. Int J Comput Sci Inf Technol 2014;5(4):5671-7.
4. Abdulhussein AN, Joshi JH, Twinamatsiko AM, Lashkari AH, Sadeghi M. An Efficient Load Balancing Algorithm for virtualized Cloud Data Centers; 2014.
5. Kaur P, Kaur PD. Efficient and enhanced load balancing algorithms in cloud computing. Int J Grid Distrib Comput 2015;8:9-14.
6. Mashaly M, Kühn PJ. Load balancing in cloud-based content delivery networks using adaptive server activation/deactivation. Engineering and Technology (ICET), 2012 International Conference on IEEE; 2012.
7. Mahapatra S, Yuan X. Load balancing mechanisms in data center networks. The 7th International Conference Expo on Emerging Technologies for a Smarter World (CEWIT); 2010.
8. Singh A, Goyal P, Batra S. Anoptimized round robin scheduling algorithm for CPU scheduling. Int J Comput Electric Eng IJCEE 2010;2(7):2383-5.
9. Ahmed T, Singh Y. Analytic study of load balancing techniques using tool cloud analyst. Int J Eng Res Appl 2012;2:1027-30.
10. Rahman A, Liu X, Kong F. A survey on geographic load balancing based data center power management in the smart grid environment. IEEE Commun Surv Tutor 2014;16(1):214-33.
11. Shao H, Rao L, Wang Z, Liu X, Wang Z, Ren K. Optimal load balancing and energy cost management for internet data centers in deregulated electricity markets. IEEE Trans Parallel Distrib Syst 2014;25(10):2659-69.