A NOVEL APPROACH FOR LOAD BALANCING IN THE CLOUD COMPUTING

[1]R.Karthikeyan, [2]A.Prabha

[1]PG Scholar, Department of Computer Science and Engineering, Mohamed Sathak Engineering college, Ramanathapuram, India.
[2]Associate Professor and Head, Department of Computer Science and Engineering,Mohamed Sathak Engineering college, Ramanathapuram, India.
[1]karthikhonda77@gmail.com , [2]Prabha.peeps3@gmail.com

Abstract: Cloud computing introduces some new concepts that entirely change the way applications are built and deployed. Usually, Cloud systems rely on virtualization techniques to allocate computing resources on demand. Thus, scalability is a critical issue to the success of enterprises involved in doing business on the cloud. In this paper, we will describe the novel virtual cluster architecture for dynamic scaling of cloud applications in a virtualized Cloud Computing environment.. Develop an efficient semi-online color set algorithm that achieves good demand satisfaction ratio and saves energy by reducing the number of servers used when the load is low.

Key Words: Cloud Computing, Auto-Scaling, CCBP, Virtualization.

1. INTRODUCTION

Cloud computing is a paradigm that focuses on sharing data and computations over a scalable network of nodes, spanning across end user computers, data centers, and web services. A scalable network of such nodes forms a cloud. An application based on these clouds is taken as a cloud application. In recent years, most of the software, hardware and networking have grown, especially service-based cloud computing has changed the traditional computer and its centralized storage. It has tremendous potential to empowerment, agility, multi-tenancy, reliability, scalability, availability, performance, security and maintenance. The USN National Institute of Standards and Technology (NIST) define cloud computing as follows:

Cloud computing is redefining the way many Internet services are operated and provided, including Video-on-Demand (VoD). Instead of buying racks of servers and building private data centers, it is now feasible for VoD companies to use computing and bandwidth resources of cloud service providers. As an example, Netflix moved its streaming servers, encoding software, data stores and other customer-oriented APIs to Amazon Web Services (AWS) in 2010. One of the most important economic appeals of cloud computing is its elasticity and auto-scaling in resource provisioning.

Traditionally, after careful capacity planning, an enterprise makes long-term investments on its infrastructure to accommodate its peak workload. Over-provisioning is inevitable while utilization remains low during most non-peak times. In contrast, in the cloud, the number of computing instances launched can be

changed adaptively at a fine granularity with a lead time of minutes. This converts the up-front infrastructure investment to operating expenses charged by cloud providers. As the cloud's auto-scaling ability enhances resource utilization by matching supply with demand, overall expenses of the enterprise may be reduced. Unlike web servers or scientific computing, VoD is a network-bound service with stringent bandwidth requirements. As VoD users must download at a rate no smaller than the video playback rate to smoothly watch video streams online, bandwidth, as opposed to storage and computation constitutes the performance bottleneck. Yet, a major obstacle that prevents numerous VoD providers from embracing cloud computing service.

The automatic scaling problem in the cloud environment, and model it as a modified Class Constrained Bin Packing (CCBP) problem where each server is a bin and each class represents an application. We develop an innovative auto scaling algorithm to solve the problem and present a rigorous analysis on the quality of it with provable bounds. Compared to the existing Bin Packing solutions, we creatively support item departure which can effectively avoid the frequent placementchanges1 caused by repacking. We support green computing by adjusting the placement of application instances adaptively and putting idle machines into the standby mode. Experiments and

simulations show that our algorithm is highly efficient and scalable which can achieve high demand satisfaction ratio, low placement change frequency, short request response time, and good energy saving.

The architecture of our system encapsulates each application instance inside a virtual machine (VM). The use of VMs is necessary to provide isolation among entrusted users. Both Amazon EC2 and Microsoft Azure use VMs in their cloud computing offering. Each server in the system runs the Xen hypervisor which supports a privileged domain 0 and one or more domain U. Each domain U encapsulates an application instance, which is connected to share network storage (i.e., the storage tier). The multiplexing of VMs to PMs (Physical Machines) is managed using the Usher framework [9]. (We use the terms "server", "PM", and "node" interchangeably in this paper.) The main logic of our system is implemented as a set of plug-ins to usher. Each node runs a Usher local node manager (LNM) on domain 0 which keeps track of the set of applications running on that node and the resource usage of each application. A L7 switch is in charge of forwarding requests and responses.

2. EXISTING SYSTEM

In the existing system the Class Constrained Bin Packing (CCBP) was modeled two problems which were using the semi-automated color set algorithm. In that concept they were encapsulated the application instance into the virtual machine. The switch collected all the request information of instances and the Local Node Manager (LNM) collect the load information in the existing system. Then this information is passed to the scheduler of the network. The scheduler is invoked when the decision is made. Then it was based on the application placement and the load distribution. This load distribution was based upon future consumption. The information's were forwarded to the Local Node Manager and also to the switch. Then the switch again collects the new information for the new request. In the existing system the traditional bin packing problem is studied. This vector bin packing problem considered the multidimensional constraints when packing item into the bin. The memory consumption is highly increases when the little amount of load is used. Then the existing system does not support the green computing when the system load is low.

3. PROPOSED SYSTEM

In our proposed system the user request is split into the different form. Then this separated user request are send to the instances. From the collection of information the resources are provided to the different users depending upon their requirements. In our proposed system the guard time is measured to calculate the radio time. The guard time is a time interval that is a portion of a burst period where no radio transmission can occur. And also our proposed system can efficiently handle the energy consumption when the user requests are separated.

In our proposed system the user request is split into the different form. Then this separated user request are send to the instances. From the collection of information the resources are provided to the different users depending upon their requirements. In our proposed system the guard time is measured to calculate the radio time. The guard time is a time interval that is a portion of a burst period where no radio transmission can occur. And also our proposed system can efficiently handle the energy consumption when the user requests are separated.

3.1Authentication

Multiple users enter into distributed system and process according to their needs. For providing the security the users first register their personal details into the system. According to the registration they have a unique id for entering the cloud and process their own needs in that. After the registration process the unique id and password can be sending to user's mail. Next, the user can login to the system with the help of their user id and password then they have to send the request into the cloud for our purposes. It provide the security of users and also easy to user can enter the cloud systems. Mainly used to the login purpose of the user

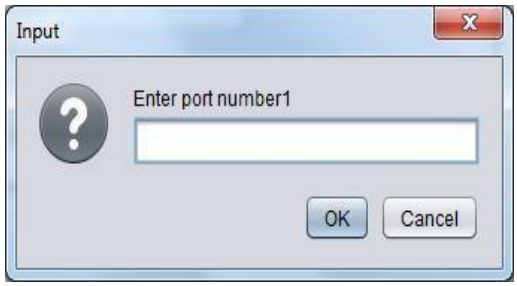

Figure1: User Login

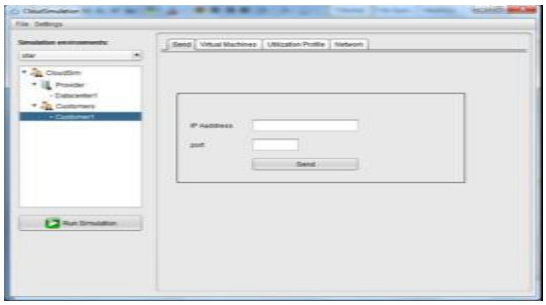

Figure 2: User Enter the Cloud

3.2LOAD BALANCING

Load balancing is to manage the multiple requests coming from the users. Normally, load balancing distributes workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives.

Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. In that purpose it introduces the bottleneck concept to manage the requests in an efficient manner. It can access the multiple requests coming from the cloud and also increase the system throughput and scalability. Load balancing is the main mechanism of manage the multiple requests.

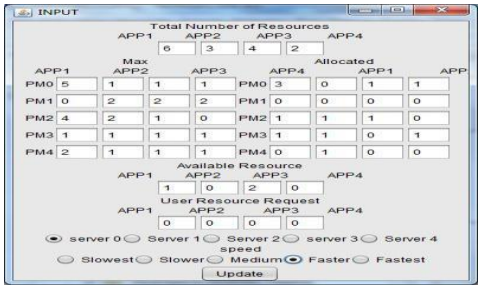

Figure 3: Load Balancing

3.3 SCHEDULING

Scheduling is the main progress in the cloud system. For circulating the resources and also equally manages the users request it have to choose the scheduling process. The round robin based scheduling can be used for maintain the requests in an efficient manner. Here, use the scheduling to maintain the server and also the clients. A bottleneck, in a communications context, is a point in the enterprise where the flow of data is impaired. Effectively, there isn't enough data handling capacity to handle the current volume of traffic.

A bottleneck can occur in the user network or storage fabric or within servers where there is excessive contention for internal server resources, such as CPU processing power, memory, or I/O (input/output). As a result, data flow slows down to the speed of the slowest point in the data path. This slow down affects application performance, especially for databases and other heavy transactional applications, and can even cause some applications to crash. Round-robin (RR) is one of the algorithms.

Round-robin scheduling is simple, easy to implement, and starvation-free. Round-robin scheduling can also be applied to other scheduling problems, such as data packet scheduling in computer networks. It is an Operating System In order to schedule processes fairly, a round-robin scheduler generally employs time-sharing, giving each job a time slot or quantum (its allowance of CPU time), and interrupting the job if it is not completed by then.

The job is resumed next time a time slot is assigned to that process. In the absence of time-sharing, or if the quanta were large relative to the sizes of the jobs, a process that produced large jobs would be favored over other processes. Round Robin algorithm is a pre-emptive algorithm as the scheduler forces the process out of the CPU once the time quota expires.

For example, if the time slot is 100 milliseconds, and job1 takes a total time of 250 ms to complete, the round-robin scheduler will suspend the job after 100 ms and give other jobs their time on the CPU. Once the other jobs have had their equal share (100 ms each), job1 will get another allocation of CPU time and the cycle will repeat.

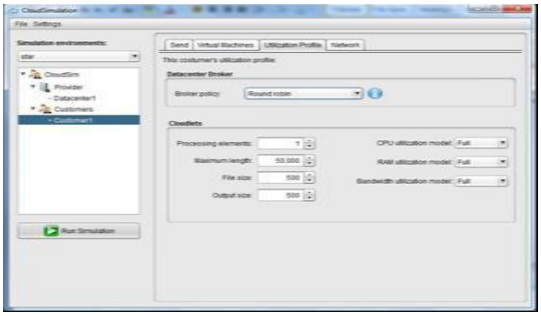

Figure 4: Scheduling Process

3.4RESOURCE ALLOCATION

The final and important process of our concept is resource allocation. Resource allocation is mainly used to provide resources for all users in the cloud system. For providing resources in proper manner it has to done the process of bottleneck and also the scheduling. According to those processes the resources are to be allocated to the users in an efficient manner. The main aim of the resource allocation process is to increase the scalable and throughput of the system.

The Bottleneck Shortest Path Problem is a basic problem in network optimization. The goal is to determine the limiting capacity of any path between two specified vertices of the network. This is equivalent to determining the unsplittable maximum flow between the two vertices. In this analyze the complexity of the problem, its relation to the Shortest Path Problem, and the impact of the underlying machine/computation model Develop a new variable depth search procedure, GLS (Guided Local Search), based on an interchange scheme and using the new concept of neighborhood trees. Structural properties of the neighborhood are used to guide the search in promising directions. While this procedure competes successfully with others even as a stand-alone, a hybrid procedure that embeds GLS into a Shifting Bottleneck framework.

It takes advantage of the differences between the two neighborhood structures proves to be particularly efficient. Then it report extensive computational testing on all the problems available from the literature

Cloud deployments can save money, free businesses from vendor lock-ins that could really sting over time, and offer flexible ways to combine public and private applications. The following are 11 top open-source cloud applications, services, educational resources, support options, general items of interest, and more.

The system's framework for allocating resources to competing co-hosted services (customers). The basic challenge is to determine the resource demand of each customer at its current request load level, and to allocate resources to their most efficient and productive use. Resources are left idle if the marginal cost to use them.

4. SYSTEM ARCHITECTURE

Auto Scaling is the ability to scale up or down the capacity automatically according to conditions of the user define. With Auto Scaling ensure that the number of instances is increasing seamlessly during demand spikes to maintain performance, and decreases automatically during demand reduce to minimize costs.

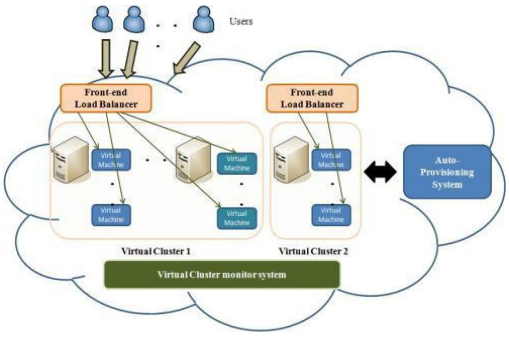

Figure5: Cloud Auto Scaling

4.1 AUTOSCALING ALGORITHM

A web application, Virtual cluster monitor system can detect whether the number of active HTTP sessions are over the threshold in a virtual cluster. For distributed computing task, Virtual cluster monitor system is able to detect whether the number of virtual machine are over the threshold of use of physical resources in a virtual cluster. As shown in Figure 1, the auto-scaling algorithm is implemented in Auto-provisioning system, and Virtual cluster monitor system is used to control and trigger the scale-up and scale-down in Auto-provisioning system on the number of virtual machine instances based on the statistics of the scaling indicator. Virtual appliance image is a set of virtual appliances on a virtual machine. Therefore, to simplify provisioning process, a virtual machine appliance image template that includes the web application and its corresponding web server is stored in the image repository of the Cloud system. The new virtual machines of web servers and web applications can be rapidly created and provisioned to the virtual cluster using the corresponding appliance image template. For web service application, network bandwidth and number of sessions are most important index for the service level agreement (SLA) and quality of service (QoS). Figure 2 illustrates our auto-scaling algorithm for web service application. The algorithm first determines the current VMs with network bandwidth and active sessions above or below given threshold numbers, respectively. If all VMs have network bandwidth and active sessions above the given upper threshold, a new VM will be provisioned, started, and then added to the front-end

load-balancer, respectively. If there are VMs with network bandwidth and active sessions below a given lower threshold and with at least one VM that has no network traffic or active sessions, the idle VM will be removed from the front-end load-balancer and be terminated from the system.

4.2ARCHITECTURE DESIGN

This paper considers two scenarios about web service and parallel processing application in the Cloud Computing environment. A web service should be available at any time and should provide the fastest response time regardless of the number of users served. Therefore, a Cloud service system should scale the service dynamically; extending and shrinking the number of web servers and web service components for large requirement and small requirement. Cloud computing also allows users to access supercomputer-level computing power by distributing tasks into large amount of computing nodes (virtual machines). Users may not know the exact number of computing nodes should be utilized to efficiently perform their jobs. Thus, the under-provisioning and over-provisioning happen often. A Cloud computing system should scale the computing nodes according to the workload.

5. CONCLUSION AND FUTURE WORK

5.1CONCLUSION

The automatic scaling process had to provide an efficient scheduling process in the internet applications. The multiple users can enter into the cloud and accessing some resources in that cloud. Here, it has to perform the actions of scheduling and load balancing for utilizing the resources in an efficient manner. In an internet application, cloud related process is to be performed gradually. Here, it has to propose the round robin based scheduling for schedule the users. The allocation of virtual machine to be done by that process. In that situation the load had to be balanced evenly.

After performing these processes it has to move onto the resource allocation process. In resource allocation to evenly allocate the resources for all users and also save the runtime of the system utilization. For that purpose, use the concept of bottleneck to eventually provide the resources at even timing. Thus proposed concept can provide the efficient resource allocation in the internet applications. The Virtual cluster monitor system is used to collect the use of physical resources of each virtual machine in a virtual cluster. The Auto-provisioning system is utilized to dynamically provision the virtual machines based on the number of the active sessions or the use of the resources in a virtual cluster.

5.2FUTURE WORK

Future work includes plan to extend our system to support differentiated services but also consider fairness when allocating the resources across the applications. It also develops an efficient algorithm to distribute incoming requests among the set of equivalence classes and to balance the load across those server clusters adaptively. CCBP works well when the aggregate load of applications in a color set is high. Another direction for future work is to extend the algorithm to pack application switch complementary bottleneck resources together, e.g., to co-locate a CPU intensive application with a memory intensive one so that different dimensions of server resources can be adequately utilized.

6. ACKNOWLEDGEMENT

The author would like to thank the respective Head of the Institution, Head of the Department and Faculty members for giving valuable advices and providing technical support.

REFERENCES

[1] E. Caron, L. Rodero-Merino, F. Desprez, and A.Muresan, “Autoscaling, load balancing and monitoring in commercial and open source clouds,” INRIA, Rapport de recherche RR-7857, Feb. 2012.2.

[2] J. Famaey, W. D. Cock, T. Wauters, F. D. Turck, B. Dhoedt, and P. Demeester, “A latency-aware algorithm for dynamic service placement in large-scale overlays,” in Proc. IFIP/IEEE Int. Conf. Symp.Integrat. Netw. Manage. (IM’09), 2009, pp. 414–421.

[3] Adam and R. Stadler, “Service middleware for self-managing large-scale systems,” IEEE Trans. Netw. Serv. Manage., vol. 4, no. 3, pp. 50–64, Dec. 2007.

[4] 4.C.Zhang, V. Lesser, and P. Shenoy, “A multi-agent learning approach to online distributed resource allocation,” in Proc. Int.Joint Conf. Artif. Intell. (IJCAI’09), 2009,

[5] S. Osman, D. Subhraveti, G. Su, and J. Nieh, “The design and implementation of zap: A system for migrating computing environments,”SIGOPS Oper.

[6] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, “A scalable application placement controller for enterprise

data centers," in Proc. Int. World Wide Web Conf. (WWW'07), May 2007, pp. 331–340.

[7] Scalr: The Auto Scaling Open Source Amazon EC2 Effort. https://www. scalr.net/. Accessed on May 10, 2012.

[8] D.Magenheimer, "Transcendent memory: A new approach to managing RAM in a virtualized environment," in Proc. Linux Symp., 2009, pp. 191–200.

[9] LinuxDocumentation.http://www.kernel.org/doc /documentation/power/states.txt. Accessed on May 10, 2012.