Allocation of Resources Dynamically using Virtual Machines for Cloud Computing

[1]Pooja.Kulkarni, [2]Patil.Yogita
M.Tech. Scholar, Assistant Professor
AIET, GLB, Karnataka
[1]Kulkarni.pooja.26@gmail.com, [2]agyogita@gmail.com

Abstract **- In this paper, we present the design and implementation of an automated resource management system that achieves a good load balance between the two system. Here we present a system that uses virtualization technology to allocate data centre resources dynamically based on application demands. We introduce the concept of "skewness" to measure the unevenness in the multidimensional resource utilization of a server. By minimizing skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources. This algorithm also helps to distribute the load evenly among servers thus requiring less number of servers to execute more number of tasks.**

Key Words **- Cloud Computing, Resource management, Virtualization**

I. INTRODUCTION

Cloud computing is a popular trend in current computing which attempts to provide cheap and easy access to computational resources. Compared to previous paradigm, cloud computing focuses on treating computational resources as measurable and billable utilities. From the clients point of view, cloud computing provides an abstraction of the underlying hardware architecture. This abstraction saves them the costs of design, setup and maintenance of a data centre to host their Application Environments (AE).Whereas for cloud providers, the arrangement yields an opportunity to profit by hosting many AEs[1]. This provides benefits to both parties, but leaves the providers in a position where they must have an efficient and cost effective data centre.

The elasticity and the lack of upfront capital investment offered by cloud computing is appealing to many businesses. There is a lot of discussion on the benefits and costs of the cloud model and on how to move legacy applications onto the cloud platform. Here we study a different problem such as cloud service provider best multiplex its virtual resources onto the physical hardware is considered. This is important because much of the touted gains in the cloud model come from such multiplexing. Studies have found that servers in many existing data centers are often severely under-utilized due to over-provisioning for the peak demand [2][3]. The cloud model is expected to make such practice unnecessary by offering automatic scale up and down in response to load variation. Besides reducing the hardware cost, it also saves on electricity which contributes to a significant portion of the operational expenses in large data centers.

Virtual machine monitors (VMMs) like Xen provide a mechanism for mapping virtual machines (VMs) to physical resources [4]. This mapping is largely hidden from the cloud users. Users with the Amazon EC2 service [5], for example, do not know where their VM instances run. It is up to the cloud provider to make sure the underlying physical machines (PMs) have sufficient resources to meet their needs. VM live migration technology makes it possible to change the mapping between VMs and PMs while applications are running [6], [7]. However, a policy issue remains as how to decide the mapping adaptively so that the resource demands of VMs are met while the number of PMs used is minimized. This is challenging when the resource needs of VMs are heterogeneous due to the diverse set of applications they run and vary with time as the workloads grow and shrink[8]. The capacity of PMs can also be heterogenous because multiple generations of hardware co-exist in a data center.

We aim to achieve the following goal in our algorithm:

Overload avoidance: the capacity of a PM should be sufficient to satisfy the resource needs of all VMs running on it. Otherwise, the PM is overloaded and can lead to degrade the performance of its VMs.

This paper makes the following contributions:

- We develop a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used.
- We introduce the concept of "skewness" to measure the uneven utilization of a server. By minimizing skewness, we can improve the overall utilization of servers in the face of multi-dimensional resource constraints.

The rest of this report is organized as follows. In Section II, we provide a general description of the problem. Section III describes system architecture. Section IV provides the details of the skewness algorithm. Finally, Section V summarizes our conclusions and outlines our future directions.

II. PROBLEM DESCRIPTION

Develop a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used. To avoid the overloading of PM and degrading of VM performance there is a need to increase the capacity of PM in order to satisfy the resource need. By introducing the concept of Skewness it is possible to measure the uneven utilization of server. Also it helps to improve the overall utilization of servers in the face of multidimensional resource constraints.

III. SYSTEM ARCHITECTURE

Existing system has their limitations such as Virtual Machine Monitor that allows multiple different operating systems to coexist simultaneously in the same hardware machine in a protected way and offer high performance. It also presents Xen, virtual machine monitor which allows multiple commodity operating systems to share conventional hardware in a safe but without sacrificing either performance or functionality. This is challenging when the resource needs of VMs are heterogeneous due to the diverse set of applications they run and very with time as the workload grow and shrink. To overcome this problem proposed system is implemented. The architecture of the system is presented in Figure 1.Each PM runs the Xen hypervisor (VMM) which supports a privileged domain 0 and one or more domain U. Each VM in domain U encapsulates one or more applications such as Web server, remote desktop, DNS, Mail, Map/Reduce, etc. Here it is assumed that all PMs Share backend storage.

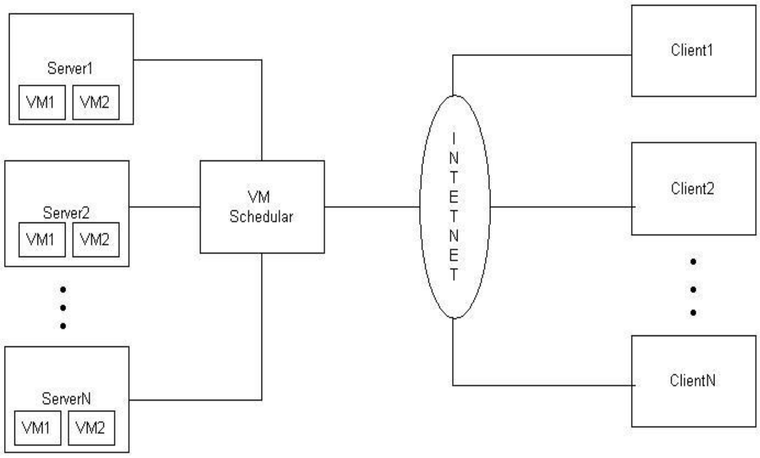

Fig 1. System architecture

The multiplexing of VMs to PMs is managed using the Usher framework [9]. The main logic of our system is implemented as a set of plug-ins to Usher. Each node runs an Usher local node manager (LNM) on domain 0 which collects the usage statistics of resources for each VM on that node.

The CPU and network usage can be calculated by monitoring the scheduling events in Xen. The memory usage within a VM, however, is not visible to the hypervisor. One approach is to infer memory shortage of a VM by observing its swap activities [10]. Unfortunately, the guest OS is required to install a separate swap partition. Furthermore, it may be too late to adjust the memory allocation by the time swapping occurs. Instead we implemented a working set prober (WS Prober) on each hypervisor to estimate the working set sizes of VMs running on it[11].

IV. THE SKEWNESS ALGORITHM

We introduce the concept of *skewness* to quantify the unevenness in the utilization of multiple resources on a server. Let *n* be the number of resources we consider and *ri* be the utilization of the *i*-th resource. We define the resource skewness of a server *p* as

$$\text{skewness}(p) = \sqrt{\sum_{N}^{i=1} (r_i/r' - 1)^2}$$

where *r'* is the average utilization of all resources for server *p*. In practice, not all types of resources are performance critical and hence we only need to consider bottleneck resources in the above calculation. By minimizing the skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources. To perform skewness algorithm we should need server status spot. Types of server status spot:

- **Hot spot:** We define a server as a *hot spot* if the utilization of any of its resources is above a hot threshold. This indicates that the server is overloaded and hence some VMs running on it should be migrated away.

We define the temperature of a hot spot p as the square sum of its resource utilization beyond the hot threshold:

$$\text{temperature}(p) = \sum_{r \in R} (r - r_t)^2$$

where R is the set of overloaded resources in server p and *rt* is the hot threshold for resource r. (Note that only overloaded resources are considered in the calculation.) The temperature of a hot spot reflects its degree of overload. If a server is not a hot spot, its temperature is zero.

- **Cold spot:** We define a server as a cold spot if the utilizations of all its resources are below a cold threshold. This indicates that the server is mostly idle and a potential candidate to turn off to save energy.
- Warm spot**:** We define a server as a warm spot if the utilizations of all its resources are below a hot threshold and above the cold threshold. This indicates that the server is ready to run VMs.

Algorithm: Skewness algorithm

Input: No. Server, No.Rresources
Output: Server status
1. Let NP=No. of server
2. Let N=No. of resources, B= server status
3. Let SK(NP) skewness value array;
4. for i=1 to NP
5. for j=1 to N
6. Let R(j)=jth resource utilization
7. SUM_sk=0
8. // Calculating skewness
    SUM_sk=SUM_sk+(R(J)/aver_ut-1)2
9. // increment j by 1
    j = j+1
10. SK(i)=√SUM_sk
11. increment i <- i+1
12. Find biggest value in SK and assign to B
13. B server status selected by skewness algorithm

So based on the status of the server B, system transfer the overload to other system.

V. CONCLUSION AND FUTURE WORK

The design, implementation of a resource management system for cloud computing services are presented successfully. Our system multiplexes virtual to physical resources adaptively based on the changing demand. The skewness metric used to combine VMs with different resource characteristics appropriately so that the capacities of servers are well utilized. Our algorithm achieves overload avoidance for systems with multi-resource constraints.

The future work of this project is to predict the future resource needs of VMs. Prediction algorithm plays an important role in improving the stability and performance of our resource allocation decisions.

REFERENCES

[1] G. J. Popek and R. P. Goldberg, “Formal Requirements for Virtualizable Third Generation Architectures,” ACM Communications, vol. 17, no. 7 pp. 412–421, 1974.
[2] M. Armbrust et al., “Above the clouds: A berkeley view of cloud computing,” University of California, Berkeley, Tech. Rep., Feb 2009.
[3] L. Siegele, “Let it rise: A special report on corporate IT,” in The Economist, Oct. 2008.
[4] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” in Proc. of the ACM Symposium on Operating Systems Principles (SOSP’03), Oct. 2003.
[5] “Amazon elastic compute cloud (Amazon EC2), http://aws.amazon.com/ec2/.”
[6] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live migration of virtual machines,” in Proc. of the Symposium on Networked Systems Design and Implementation(NSDI’05), May 2005.
[7] M. Nelson, B.-H. Lim, and G. Hutchins, “Fast transparent migration for virtual machines,” in Proc. of the USENIX Annual Technical Conference,2005
[8] Zhen Xiao,”Dynamic resource allocation using virtual machine in cloud computing environment” IEEE, Weijia Song, and Qi Chen,June 2013.
[9] M. McNett, D. Gupta, A. Vahdat, and G. M. Voelker, “Usher: An extensible framework for managing clusters of virtual machines,” in Proc. of the Large Installation System Administration Conference (LISA’07), Nov. 2007.
[10] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, “Black-box and gray-box strategies for virtual machine migration,” in Proc. Of the Symposium on Networked Systems Design and Implementation (NSDI’07), Apr. 2007.
[11] C. A. Waldspurger, “Memory resource management in VMware ESX server,” in Proc. of the symposium on Operating systems design and implementation (OSDI’02), Aug. 2002.
[12] P. Padala, K.-Y. Hou, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, and A. Merchant, “Automated control of multiple virtualized resources,” in Proc. of the ACM European conference on Computer systems (EuroSys’09), 2009.