International Journal of Engineering Research and General Science Volume 3, Issue 4, July-August, 2015
ISSN 2091-2730

Virtual Machine Techniques for Dynamic Resource Allocation in Cloud System

Rathod Dnyaneshwar N.[1] , Prof. Kulkarni Prasad[2]

[1] Student of ME Software Engineering, Aditya Engineering College, Beed

[2] Department of Computer Engineering, Aditya Engineering College, Beed

ABSTRACT- Now a days, the importance of cloud computing grows rapidly. Cloud Computing Environment allows business/company users to scale up and down their resource usage based on their business needs. Many of the touted gains in the cloud model come from resource multiplexing through virtualization mechanism. In this paper, I display a system that uses virtualization technology to allocate data center resources dynamically based on application demands and support green computing by optimizing the number of servers in use. For this purpose I used concept of Virtual Machine (VM) & Physical Machine (PM). I present the concept of "skewness" to measure the unevenness in the multi-dimensional resource utilization of a server. By reducing skewness, we can combine various types of workloads nicely and improve the overall utilization of server resources. I develop a set of heuristics that prevent overload in the system effectively while saving energy used. I present a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used.

Keywords: Cloud Computing, Resource Management, Virtualization, skewness, Green Computing, Physical Machine

I. INTRODUCTION

Cloud computing is the delivery of computing and storage capacity as a service to a community of end recipients. The name comes from the use of a cloud shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts services with a user's data, software and computation over a network. The remote accessibility enables us to access the cloud services from anywhere at any time. To gain the maximum degree of the above mentioned benefits, the services offered in terms of resources should be allocated optimally to the applications running in the cloud. The elasticity and the lack of upfront capital investment offered by cloud computing is appealing to any businesses. In this paper, we discuss how the cloud service provider can best multiplex the available virtual resources onto the physical hardware. This is important because much of the touted gains in the cloud model come from such multiplexing. Virtual Machine Monitors (VMMs) like Xen provide a mechanism for mapping Virtual Machines (VMs) to Physical Resources [3]. This mapping is hidden from the cloud users. Users with the Amazon EC2 service [4], for example, do not know where their VM instances run. It is up to the Cloud Service Provider to make sure the underlying Physical Machines (PMs) has 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 [5], but, 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. The capacity of PMs can also be heterogeneous because multiple generations of hardware co-exist in a data center. To achieve the overload avoidance that is 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 degraded performance of its VMs. And also the number of PMs used should be minimized as long as they can still satisfy the needs of all VMs. Idle PMs can be turned off to save energy.

II. EXISTING SYSTEM

In [2] author proposed architecture, using feedback control theory, for adaptive management of virtualized resources, which is based on VM. In this VM-based architecture all hardware resources are pooled into common shared space in cloud computing infrastructure so that hosted application can access the required resources as per there need to meet Service Level Objective (SLOs) of application. The adaptive manager use in this architecture is multi-input multi-output (MIMO) resource manager, which includes 3

controllers: CPU controller, memory controller and I/O controller, its goal is regulate multiple virtualized resources utilization to achieve SLOs of application by using control inputsper-VM CPU, memory and I/O allocation. The seminal work of Walsh et al. [3], proposed a general two-layer architecture that uses utility functions, adopted in the context of dynamic and autonomous resource allocation, which consists of local agents and global arbiter. The responsibility of local agents is to calculate utilities, for given current or forecasted workload and range of resources, for each AE and results are transfer to global arbiter. Where, global arbiter computes near-optimal configuration of resources based on the results provided by the local agents. In [4], authors propose an adaptive resource allocation algorithm for the cloud system with preempt able tasks in which algorithms adjust the resource allocation adaptively based on the updated of the actual task executions. Adaptive list scheduling (ALS) and adaptive min-min scheduling (AMMS) algorithms are use for task scheduling which includes static task scheduling, for static resource allocation, is generated offline. The online adaptive procedure is use for re-evaluating the remaining static resource allocation repeatedly with predefined frequency.

The dynamic resource allocation based on distributed multiple criteria decisions in computing cloud explain in [6]. In it author contribution is tow-fold, first distributed architecture is adopted, in which resource management is divided into independent tasks, each of which is performed by Autonomous Node Agents (NA) in ac cycle of three activities:(1) VM Placement, in it suitable physical machine (PM) is found which is capable of running given VM and then assigned VM to that PM, (2) Monitoring, in it total resources use by hosted VM are monitored by NA, (3) In VM selection, if local accommodation is not possible, a VM need to migrate at another PM and process loops back to into placement and second, using PROMETHEE method, NA carry out configuration in parallel through multiple criteriadecision analysis.

III. PROPOSED SYSTEM

This proposed system consists of number of servers, predictor, hotspot and cold spot solvers and migrat ion list. Set of servers used for running different applications. Predictor is used to execute periodically to evaluate the resource allocation status based on the predicted future demands of virtual machines.

In this paper, I present the design and implementation of an Digitized resource management system that achieves a good balance between the two goal*:*

- Overload avoidance: the capacity of a Physical Machine should be sufficient to satisfy the resource needs of all Virtual Machines running on it. Otherwise, the PHYSICAL MACHINE is overloaded and can lead to degraded performance of its VIRTUAL MACHINEs.
- Green computing: the number of PHYSICAL MACHINEs used should be minimized as long as they can still satisfy the needs of all VIRTUAL MACHINEs. Idle PHYSICAL MACHINEs can be turned off to save energy.

IV. MODULE DESCRIPTION

After careful analysis,in this system has been identified to have the following modules:

1. **Cloud Computing Module.**
2. **Resource Management Module.**
3. **Virtualization Module.**
4. **Green Computing Module.**

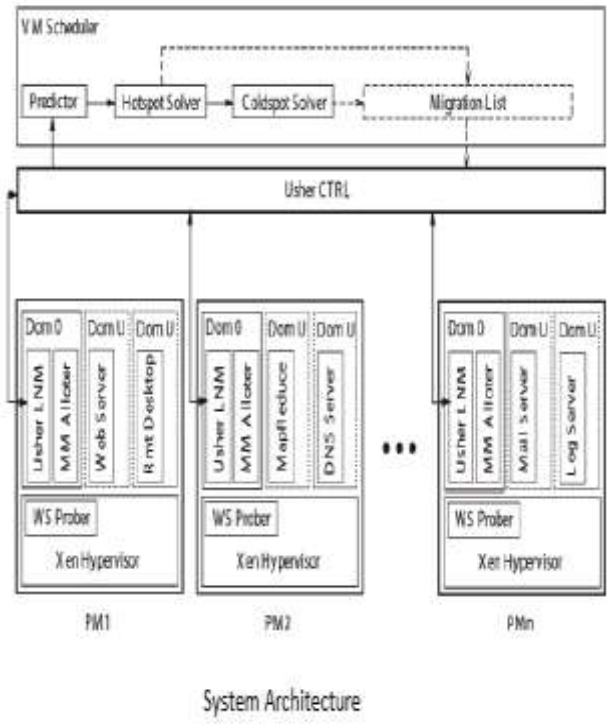

Figure 1. Proposed System Architecture

1.Cloud Computing Module:

Cloud computing refers to applications and services offered over the Internet. These services are offered from data centers all over the world, which collectively are referred to as the "cloud." Cloud computing is a movement away from applications needing to be installed on an individual's computer towards the applications being hosted online. Cloud resources are usually not only shared by multiple users but as well as dynamically re-allocated as per demand. This can work for allocating resources to users in different time zones.

2. Resource Management Module:

Dynamic resource management has become an active area of research in the Cloud Computing Environment. Cost of resources changes significantly depending on configuration for using them. Hence efficient management of resources is of prime interest to both Cloud Providers and Cloud Users. The success of any cloud management software critically de-pends on the flexibility; scale and efficiency with which it can utilize the underlying hardware resources while pro-viding necessary performance isolation. Successful resource management solution for cloud environments, needs to provide a rich set of resource controls for better isolation, while doing initial placement and load balancing for efficient utilization of underlying resources.

3. Virtualization Module:

Virtualization, in cloud Environment, is the creation of a virtual means rather than actual Version of something, such as a hardware platform, operating system, and a storage device or network resources.VIRTUAL MACHINE live migration is a widely used technique for dynamic resource allocation in a virtualized environment. The process of running two or more logical computer system so on one set of physical hardware. Dynamic placement of virtual servers to minimize SLA violations.

4. Green Computing Module:

Many efforts have been made to curtail energy consumption. Hardware based approaches include novel thermal design for lower cooling power, or adopting power-proportional and low-power hardware. Dynamic Voltage and Frequency Scaling (DVFS) to adjust CPU power according to its load in data centers. My work belongs to the category of pure-software low-cost Solutions. It requires that the desktop is virtualized with shared storage. Green computing ensures end user satisfaction, regulatory compliance, telecommuting, virtualization of server resources.

We sort the list of cold spots in the system based on the ascending order of their memory size.. We sort the list of cold spots in the system based on the ascending order of their memory size. Since we need to migrate away all its VMs before we can shut down an under-utilized server, we define the memory size of a cold spot as the aggregate memory size of all VMs running on it. Recall that our model assumes all VMs connect to a shared back-end storage. Hence, the cost of a VM live migration is determined mostly by its memory footprint. I try to eliminate the cold spot with the lowest cost first. For a cold spot *p*, we check if we can migrate all its VMs somewhere else. For each VM on *p*, we try to find a destination server to accommodate it. The resource utilizations of the server after accepting the VM must be below the *warm threshold*. While we can save energy by consolidating under-utilized servers, overdoing it may create hot spots in the future. The warm threshold is designed to prevent that. If multiple servers satisfy the above criterion, we prefer one that is not a current cold spot. This is because increasing load on a cold spot reduces the likelihood that it can be eliminated. However, we will accept a cold spot as the destination server if necessary. All things being equal, we select a destination server whose skewness can be reduced the most by accepting this VM. If we can find destination servers for all VMs on a cold spot, we record the sequence of migrations and update the predicted load of related servers. Otherwise, we do not migrate any of its VMs. The list of cold spots is also updated because some of them may no longer be cold due to the proposed VM migrations in the above process.

V. THE SKEWNESS ALGORITHM

In this paper the concept of *skewness is used* to quantify the unevenness in the utilization of multiple resources on a server. Let *n* be the number of resources we consider and n_i be the utilization of the *i*-th resource. We can define the resource skewness of a server *p* as

$$skewness(p) = \sqrt{\sum_{i=1}^{n} (\frac{r_i}{\overline{r}} - 1)^2}$$

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.

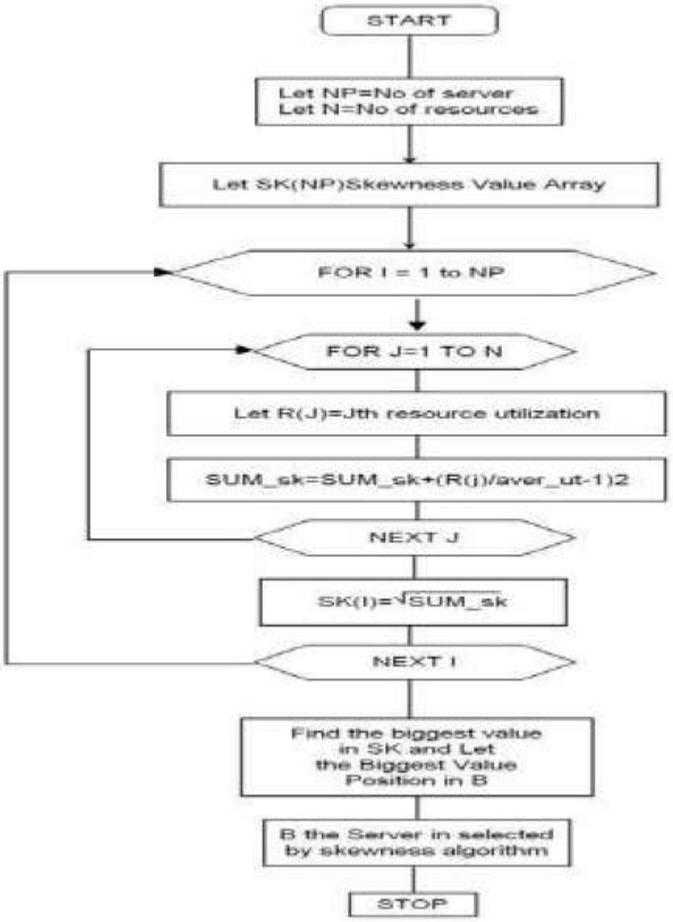

Figure 2 Flow Chart for Skewness Algorithm

VI. PERFORMANCE & RESULT DISCUSSION

We can calculate the performance of our algorithm using trace driven simulation. Note that this simulation uses the same code base for the algorithm as the real implementation in the experiments. This ensures the fidelity of simulation results.Traces are per-minute server resource utilization, such as CPU rate, memory usage, and network traffic statistics, collected using tools like "perfmon" (Windows), the "/proc" file system (Linux), "pmstat/vmstat/netstat"commands (Solaris), etc.. The raw traces are pre-processed into "Usher" format so that the

symbol	meaning	value
h	hot threshold	0.9
c	cold threshold	0.25
w	warm threshold	0.65
g	green computing threshold	0.4
l	consolidation limit	0.05

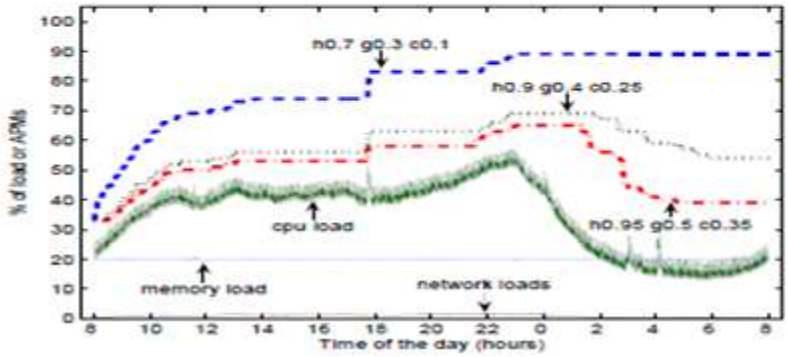

Figure 3 Impact of threshold on number of APM's

Effect of thresholds on APMs

To observe the performance of our algorithm in more extreme situations, I also create a synthetic workload which mimics the shape of a sine function (only the positive part) and ranges from 15% to 95% with a 20% random fluctuation I first evaluate the effect of the various thresholds used in our algorithm. I use random VM to PM mapping in the initial layout. The scheduler is invoked once per minute. The bottom part of Figure 2 shows the daily load variation in the system. The x-axis is the time of the day starting at 8am. The y-axis is overloaded with two meanings: the percentage of the load or the percentage of APMs (i.e., Active PMs) in the system. Recall that a PM is *active* (i.e., an APM) if it has at least one VM running. As can be seen from the figure, the CPU load demonstrates diurnal patterns which decrease substantially after midnight. The memory consumption is fairly stable over the time. The network utilization stays very low.

VII. CONCLUSION

Here I have present the technique that will progress through the design, implementation, and evaluation phases of a resource management system for cloud computing services using virtual machine technology. My system multiplexes virtual to physical resources adaptively based on the changing demand. I use the skewness metric to combine VMs with different resource characteristics appropriately so that the capacities of servers are well utilized. This algorithm achieves both overload avoidance and green computing for systems with multi-resource constraints.

REFERENCES:

[1] Zhen Xiao, *Senior Member, IEEE*, Weijia Song, and Qi Chen "Dynamic Resource Allocation using Virtual Machines for Cloud Computing Environment" in IEEE TRANSACTION ON PARALLEL AND DISTRIBUTED SYSTEMS YEAR 2013

[4] "Amazon elastic compute cloud (Amazon EC2)," http://aws.amazon.com/ec2/, 2012.

[2] T. Das, P. Padala, V. N. Padmanabhan, R. Ramjee, and K. G. Shin, "Litegreen: saving energy in networked desktops using virtualization," in *Proc. of the USENIX Annual Technical Conference*, 2010.

[3] Y. Agarwal, S. Savage, and R. Gupta, "Sleepserver: a software-only approach for reducing the energy consumption of pcs within enterprise environments," in *Proc. of the USENIX Annual Technical Conference*, 2010.

[4] N. Bila, E. d. Lara, K. Joshi, H. A. Lagar-Cavilla, M. Hiltunen, and M.Satyanarayanan, "Jettison: Efficient idle desktop consolidation with partial vm migration," in *Proc. of the ACM European conference on Computer systems.*

[5] M. Armbrust *et al.*, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep., Feb 2009.

[6] D. Meisner, B. T. Gold, and T. F. Wenisch, "Powernap: eliminating server idle power," in *Proc. of the international conference on Architectural support for programming languages and operating systems (ASPLOS'09)*, 2009.

[7] Y. Agarwal, S. Hodges, R. Chandra, J. Scott, P. Bahl, and R. Gupta, "Somniloquy: augmenting network interfaces to reduce pc energy usage," in *Proc. of the USENIX symposium on Networked systems design and implementation (NSDI'09)*, 2009.

[8] A. Singh, M. Korupolu, and D. Mohapatra, "Server-storage virtualization: integration and load balancing in data centers," in *Proc. of the ACM/IEEE conference on Supercomputing*, 2008.

[9] L. Siegele, "Let it rise: A special report on corporate IT," in *The Economist*, Oct. 2008.

[10] L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, Oct. 2008.

[11] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," Proc. IFIP/IEEE Int'l Symp. Integrated Network Management (IM '07), 2007.

[12] R. Nathuji and K. Schwan, "Virtualpower: coordinated power management in virtualized enterprise systems," in *Proc. of the ACM SIGOPS symposium on Operating systems principles (SOSP'07)*, 2007.

[13] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," Proc. Symp. Networked Systems Design and Implementation (NSDI '05), May 2005.

[14] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast transparent migration for virtual machines," in *Proc. of the USENIX Annual Technical Conference*, 2005.

[15] 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