Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 1, January 2014, pg.48 – 52

RESEARCH ARTICLE

Virtual Machine-Based Resource Management System for Cloud Computing Services

Sri hari Reddy Medapati
M.Tech student, CSE,
Sri Sai Madhavi Institute of Science & Technology,
Mallampudi, Rajahmundry, Andhra Pradesh,
srihari1203@gmail.com

K. Sathi Reddy
Assistant Professor, CSE,
Sri Sai Madhavi Institute of Science & Technology,
Mallampudi, Rajahmundry, Andhra Pradesh,
sathireddy@ymail.com

Abstract— ***Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software and information are provided to users over the network. Cloud computing providers deliver application via the Internet, which are accessed from web browser, while the business software and data are stored on servers at a remote location. In cloud computing, Resource Allocation (RA) is the process of assigning available resources to the needed cloud applications over the internet. Resource allocation starves services if the allocation is not managed precisely. Resource provisioning solves that problem by allowing the service providers to manage the resources for each individual module. Resource Allocation Strategy (RAS) is all about integrating cloud provider activities for utilizing and allocating scarce resources within the limit of cloud environment so as to meet the needs of the cloud application. This paper presents dynamically allocating resources for cloud computing services using virtual machine.***

Keywords— Cloud computing; Green computing; Resource; Skewness; Virtual machine

I. INTRODUCTION

Cloud computing [1] is the next generation in computation. Possibly people can have everything they need on the cloud. Cloud computing is the next natural step in the evolution of on-demand information technology services and products. Cloud Computing is an emerging computing technology that is rapidly consolidating itself as the next big step in the development and deployment of an increasing number of distributed applications. Cloud computing nowadays becomes quite popular among a community of cloud users by offering a variety of resources. Cloud computing platforms, such as those provided by Microsoft, Amazon, Google, IBM, and Hewlett-Packard, let developers deploy applications across computers hosted by a central organization. These applications can access a large network of computing resources that are deployed and managed by a cloud computing provider. Developers obtain the advantages of a managed computing platform, without having to commit resources to design, build and maintain the network. Yet, an important problem that must be addressed effectively in the cloud is how to manage QoS and maintain SLA for cloud users that share cloud resources.

The cloud computing technology makes the resource as a single point of access to the client and is implemented as pay per usage. Though there are various advantages in cloud computing such as prescribed and abstracted infrastructure, completely virtualized environment, equipped with dynamic infrastructure, pay per consumption, free of software and hardware installations, the major concern is the order in which the requests are satisfied. This evolves the scheduling of the resources. This allocation of resources must be made efficiently that maximizes the system utilization and overall performance. Cloud computing is sold on demand on the basis of time constrains basically specified in minutes or hours. Thus scheduling should be made in such a way that the resource should be utilized efficiently.

There are three primary classes of cloud computing service models (Figure 1):

In infrastructure as a service (IaaS), a cloud based virtual server providing networking and mass storage services and other infrastructure services. The user does not manage or control the data centre but may have control over the data or operating systems placed into the infrastructure. For example, Amazon web service (AWS). In platform as a service (PaaS), the service level where a computable platform upon which the user can host and develop applications and services by using programming language and API's is provided. The user can control the deployed applications and sometimes the application-hosting environment as well. However, the infrastructure (servers, OS, storage) is still in the control of the cloud provider. Examples include Windows Azure and Google App engine.

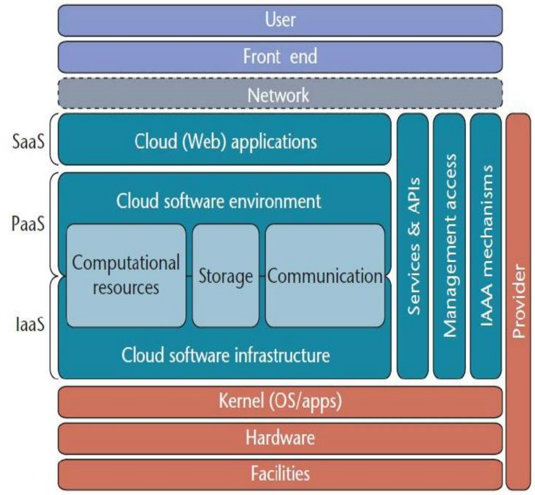

Figure 1: Cloud Computing Architecture

In software as a service (SaaS), applications are running on a cloud infrastructure or platform which is accessible via a thin client interface (browser) or program interface. The user only has the possibility to manage some user specific settings, because the provider does not accommodate cloud features; they only provide applications running 'in the cloud'. SaaS is an alternative to having the software running on local machines and good examples are online office applications (Google Docs), online CRM systems (SalesForce CRM), webmail (Google Mail) and Social Network Sites (Twitter, Facebook) [5].

In cloud computing, Resource Allocation (RA) is the process of assigning available resources to the needed cloud applications over the internet. Resource allocation starves services if the allocation is not managed precisely. Resource provisioning solves that problem by allowing the service providers to manage the resources for each individual module. Resource Allocation Strategy (RAS) is all about integrating cloud provider activities for utilizing and allocating scarce resources within the limit of cloud environment so as to meet the needs of the cloud application. It requires the type and amount of resources needed by each application in order to complete a user job.

II. RELATED WORK

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 byusing control inputs per-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) VMPlacement, 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 criteria decision analysis. This approach is potentially more feasible in large data centers than centralized approaches.

III. PROPOSED WORK

In the proposed work, we present 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.

A. System Overview

The architecture of the system is presented in Figure 2. Each physical machine (PM) runs the Xen hypervisor (VMM) which supports a privileged domain 0 and one or more domain U [7]. Each VM in domain U encapsulates one or more applications such as Web server, remote desktop, DNS, Mail, Map/Reduce, etc. We assume all PMs share a backend storage. The multiplexing of VMs to PMs is managed using the Usher framework [8]. 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 statistics collected at each PM are forwarded to the Usher central controller (Usher CTRL) where our VM scheduler runs. The VM Scheduler is invoked periodically and receives from the LNM the resource demand history of VMs, the capacity and the load history of PMs, and the current layout of VMs on PMs. The scheduler has several components. The predictor predicts the future resource demands of VMs and the future load of PMs based on past statistics. We compute the load of a PM by aggregating the resource usage of its VMs.

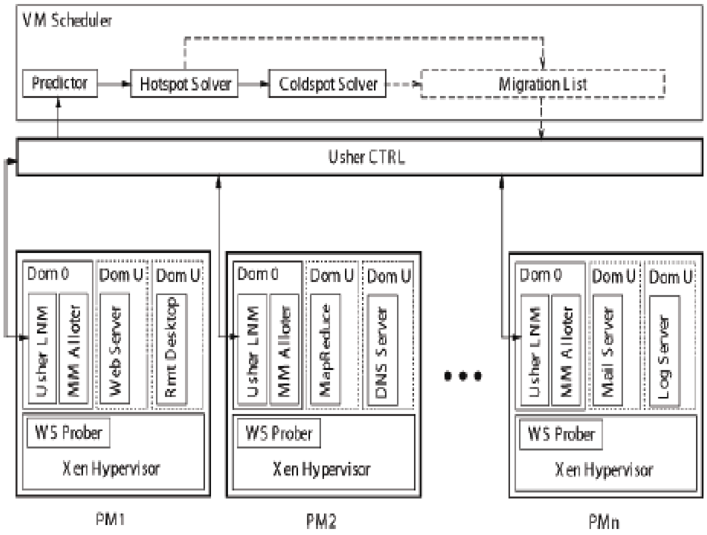

Figure 2: System Architecture

The LNM at each node first attempts to satisfy the new demands locally by adjusting the resource allocation of VMs sharing the same VMM. The MM Alloter on domain 0 of each node is responsible for adjusting the local memory allocation. The hot spot solver in our VM Scheduler detects if the resource utilization of any PM is above the hot threshold (i.e., a hot spot). The cold spot solver checks if the average utilization of actively used PMs (APMs) is below the green computing threshold.

B. Skewness Algorithm

We introduce the concept of “skewness” to measure the unevenness in the multi-dimensional resource utilization of a server. By minimizing skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources. 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

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

Our algorithm executes periodically to evaluate the resource allocation status based on the predicted future resource demands of VMs. We define a server as a hot spot if the utilization of any of its resources is above a hot threshold. We define the temperature of a hot spot p as the square sum of its resource utilization beyond the hot threshold:

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

where R is the set of overloaded resources in server p and rt is the hot threshold for resource r.

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. Finally, we define the warm threshold to be a level of resource utilization that is sufficiently high to justify having the server running but not so high as to risk becoming a hot spot in the face of temporary fluctuation of application resource demands.

C. *Hotspot Mitigation*

We sort the list of hot spots in the system in descending temperature (i.e., we handle the hottest one first). Our goal is to eliminate all hot spots if possible. Otherwise, keep their temperature as low as possible. For each server *p*, we first decide which of its VMs should be migrated away. We sort its list of VMs based on the resulting temperature of the server if that VM is migrated away. We aim to migrate away the VM that can reduce the server’s temperature the most. In case of ties, we select the VM whose removal can reduce the skewness of the server the most. For each VM in the list, we see if we can find a destination server to accommodate it. The server must not become a hot spot after accepting this VM. Note that this reduction can be negative which means we select the server whose skewness increases the least. If a destination server is found, we record the migration of the VM to that server and update the predicted load of related servers. Otherwise, we move on to the next VM in the list and try to find a destination server for it. As long as we can find a destination server for any of its VMs, we consider this run of the algorithm a success and then move on to the next hot spot. Note that each run of the algorithm migrates away at most one VM from the overloaded server. This does not necessarily eliminate the hot spot, but at least reduces its temperature. If it remains a hot spot in the next decision run, the algorithm will repeat this process.

D. Green Computing

When the resource utilization of active servers is too low, some of them can be turned off to save energy. This is handled in our green computing algorithm. Our green computing algorithm is invoked when the average utilizations of all resources on active servers are below the green computing threshold. 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.

IV. **CONCLUSIONS**

Cloud Computing is proving to be a phenomenal technology where computing services are provided over the computer networks, with on-demand elastic resources like computing energy, storage capacity, memory and network Virtualization provides an efficient solution to the objectives of the cloud computing paradigm by facilitating creation of Virtual Machines (VMs) over the underlying physical servers, leading to improved resource utilization and abstraction. In this paper, we present 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. We introduce the concept of “skewness” to measure the unevenness in the multi-dimensional resource utilization of a server. By minimizing skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources.

REFERENCES

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

[2] “Adaptive Management of Virtualized Resources in Cloud Computing Using Feedback Control,” in First International Conference on Information Science and Engineering, April 2010, pp. 99-102.

[3] W. E. Walsh, G. Tesauro, J. O. Kephart, and R. Das, “Utility Functions in Autonomic Systems,” in ICAC ’04: Proceedings of the First International Conference on Autonomic Computing. IEEE Computer Society, pp. 70–77, 2004.

[4] Jiayin Li, Meikang Qiu, Jian-Wei Niu, Yu Chen, Zhong Ming, “Adaptive Resource Allocation for Preempt able Jobs in Cloud Systems,” in 10th International Conference on Intelligent System Design and Application, Jan. 2011, pp. 31-36.

[5] P.T.Jaeger, J.Lin, and M. grimes, Cloud computing and information policy: Computing in a policy cloud? Journal of Information Technology and politics, 2009.

[6] Yazir Y.O., Matthews C., Farahbod R., Neville S., Guitouni A., Ganti S., Coady Y., “Dynamic resource allocation based on distributed multiple criteria decisions in computing cloud,” in 3rd International Conference on Cloud Computing, Aug. 2010, pp. 91-98.

[7] 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.

[8] 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.