www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 7 July, 2014 Page No. 7380-7383

Distribution of cloud resources dynamically by using virtualization

D.Sireesha, M.Chiranjeevi, P.Nirupama
M.Tech: Department of CSE
SIETK, Puttur, INDIA
Associate Professor Department of CSE
SIETK, Puttur, INDIA
Head of the department Department
Department of CSE SIETK, Puttur, INDIA
dasini.sireesha@gmail.com
mallarapu.chiranjeevi@gmail.com

Abstract—In Cloud, data are stored on servers at a remote location. Cloud computing provides the resources to the bussiness customers based on their needs. In cloud computing, Resource Allocation (RA) is the process of assigning available resources to the needed cloud applications over the internet.The Cloud providers deliver application via the Internet, which are accessed from web browser. Resource allocation starves services if the allocation is not managed precisely. 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. By using virtualization technology the resource multiplexing is done in the cloud environment.To allocate the data center resources dynamically,we present a system that uses virtualization technology.And minimizing the number of servers used,we can support the green computing.And to measure the unevenness in the multidimensional resource utilization of a server we introduce the skewness concept. By minimizing skewness, we can improve the overall utilization of server resources. To prevent the overload in the system effectively while saving energy used, We develop a set of heuristics.

IndexTerms—virtualmachine; Resource Allocation; virtualization; green computing

I.INTRODUCTION

Cloud computing is a newly emergent computing environment offers dynamic flexible infrastructures and QoS guaranteed services in a pay-as-you-go manner to the public. System virtualization technology renders to the flexible and scalable system services. The virtualization becomes an important challenge for cloud computing environment to provide a self-managing and autonomic infrastructure. The lack of upfront capital investment is appealing to many businesses in the cloud computing environment.

In cloud computing, computational resources are provided to remote users in the form of leases. For a cloud user,the user can request multiple cloud services simultaneously. In this case, parallel processing in the cloud system can improve the performance. When applying parallel processing in cloud computing, it is necessary to implement a mechanism to allocate resources and schedule the tasks execution order. a resource allocation mechanism with preemptable task execution can increase the utilization of clouds.

In this paper, we present the design and implementation of an automated resource management system.That can make the following contributions:

1.First,we develop an automated resource allocation system that can avoid overload in the system effectively and reducing the number of servers used.

2.By introducing the concept of "skewness" we can measure the uneven utilization of a server. By minimizing skewness, the overall utilization of servers can be improved in the face of multidimensional resource constraints.

3.Finally.we design a load prediction algorithm that can capture the future resource usages of applications accurately without looking inside the VMs.

.

II.RELATED WORK

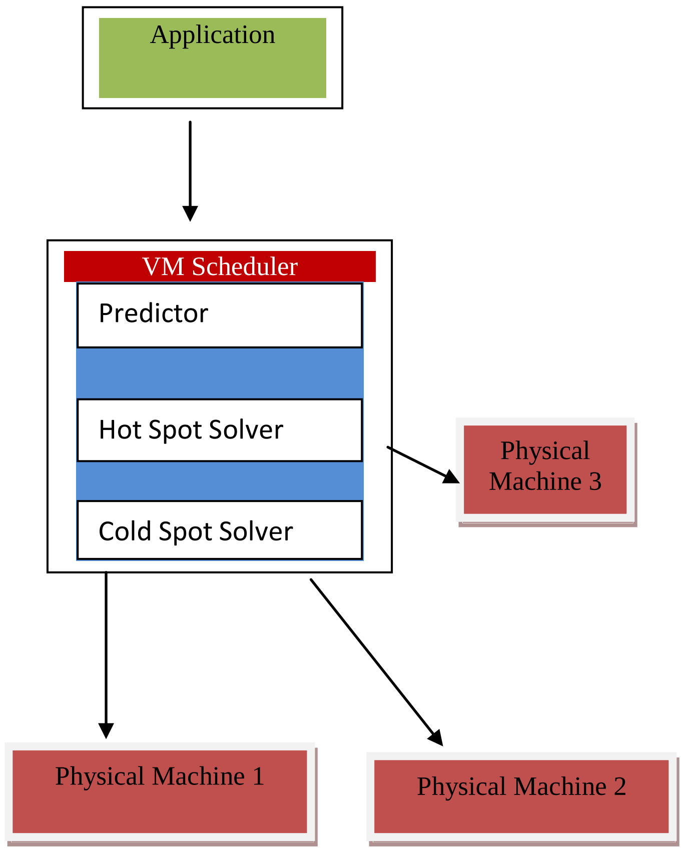
The cloud service provider can provide the services to the cloud user,The datacenter resources can be distributed to the cloud users over the network in the cloud environmnet.

Automatic scaling of Web applications was previously Used technique for data center environments. In,Which each server has replicas of all web applications running in the system. C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici[1] design a network flow algorithms to allocate the load of an application among its running instances. For connection oriented Internet services like Windows Live Messenger,
G. Chen, H. Wenbo, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao[2], presents an integrated approach for loaddispatching and server provisioning. These are not use virtual machines and require the applications be structured in a multitier architecture with load balancing provided through an front-end dispatcher. In contrast, our work targets Amazon EC2-style environment where it places no restriction on what and how applications are constructed inside the VMs. A VM is treated like a blackbox. Resource management is done only at the granularity of whole VMs.
MapReduce [3] is another type of popular Cloud service where data locality is the key to its performance. Quincy[4] adopts a min-cost flow model in task scheduling to maximize data locality while keeping fairness among different jobs. M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy[5] design the “Delay Scheduling” algorithm trades execution time for data locality.

VM live migration is a widely used technique for dynamic resource allocation in a virtualized environment The proposed work also belongs to this category, It combines multidimensional load information into a single Volume metric. It sorts the list of PMs based on their volumes and the VMs in each PM in their volume-to-size ratio (VSR).It considers the PMs and the VMs in the presorted order.

Virtual machine monitors like Xen provide a mechanism for mapping virtual machines (VMs) to physical resources.This mapping is largely hidden from the cloud users. Users with the Amazon EC2 service, 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.

III.SYSTEM ARCHITECTURE



Each PM runs on the VMScheduler. VMScheduler is having the Predictor,Hotspotsolver,ColdSpotSolver . The multiplexing of VMs to PMs is Scheduler managed using the Usher framework. The main logic of our system is implemented as a set of plug-ins to Usher node.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 . 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 on each hypervisor to estimate the working set sizes of VMs running on it. We use the random page sampling technique as in the VMware ESX Server.
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.

The LNM at each node first attempts to satisfy the new demands locally by adjusting the resource allocation of VMs sharing the same VMM. Xen can change the CPU allocation among the VMs by adjusting their weights in its CPU scheduler. 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). If so, some VMs running on them will be migrated away to reduce their load. The cold spot solver checks if the average utilization of actively used PMs (APMs) is below the green computing threshold. If so, some of those PMs could potentially be turned off to save energy. It identifies the set of PMs whose utilization is below the cold threshold (i.e., cold spots) and then attempts to migrate away all their VMs. It then compiles a migration list of VMs and passes it to the Usher CTRL for execution.

1.VM Scheduler

VM Scheduler run and invoked periodically receives from the user LNM (Local Node Manager)the resource demand history of VMs, the capacity and the load history of PMs, and the current layout of VMs on PMs. Then it can forward the request to predictor

2.Predictor

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. The LNM at each node first attempts to satisfy the new demands locally by adjusting the resource allocation of VMs sharing the same VMM. Xen can change the CPU allocation among the VMs by adjusting their weights in its CPU scheduler. The MM Alloter on domain 0 of each node is responsible for adjusting the local memory allocation.

3.Hotspot solver

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). If so, some VMs running on them will be migrated away to reduce their load. Then it can give the request to coldspot solver.

4.Coldspot solver

The cold spot solver checks if the average utilization of actively used PMs (APMs) is below the green computing threshold. If so, some of those PMs could potentially be turned off to save energy. It identifies the set of PMs whose utilization is below the cold threshold (i.e., cold spots) and then attempts to migrate away all their VMs then it forward request to migration list

SKEWNESS ALGORITHM

The Skewness algorithm 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 ri be the utilization of the ith resource.The resource skewness of a server p can be defined as

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

the 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 .ri 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.

IV.RESOURCE BALANCE

skewness algorithm is used to mix workloads with different resource requirements together so that the overall utilization of server capacity is improved. In this paper, we see how our algorithm handles a mix of CPU, memory, and network intensive workloads. We vary the CPU load as before. We inject the network load by sending the VMs a series of network packets. The memory intensive applications are created by allocating memory on demand. Again we start with a small scale experiment consisting of two PMs and four VMs so that we can present the results for all servers.

IV.CONCLUSION

The design, implementation, and evaluation of a resource management system for cloud computing services are developed. Based on the changing of the application demand the system multiplexes virtual to physical resources adaptively. The skewness concept is used to combine VMs with different resource characteristics appropriately so the capacities of servers are well utilized.And the algorithm achieves both overload avoidance and green computing for systems with multiresource constraints.

REFERENCES

[1]. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, “A Scalable Application Placement Controller for Enterprise Data Centers,” Proc. Int’l World Wide Web Conf. (WWW ’07), May 2007.

[2]. G. Chen, H. Wenbo, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao,“Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services,” Proc. USENIX Symp. Networked Systems Design and Implementation (NSDI ’08), Apr. 2008.

[3]. M. Zaharia, A. Konwinski, A.D. Joseph, R.H. Katz, and I. Stoica, “Improving MapReduce Performance in Heterogeneous Environments,” Proc. Symp. Operating Systems Design and Implementation (OSDI ’08), 2008.

[4]. M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, “Quincy: Fair Scheduling for Distributed Computing Clusters,” Proc. ACM Symp. Operating System Principles (SOSP ’09), Oct. 2009.

[5].M. Zaharia, A. Konwinski, A.D. Joseph, R.H. Katz, and I. Stoica, “Improving MapReduce Performance in Heterogeneous Environments,” Proc. Symp. Operating Systems Design and Implementation (OSDI ’08), 2008.