ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Dynamic Resource Allocation using Virtual Machines for Cloud Computing

Mr.N Srinivas
(Associate Professor)
M.Sc(Osmania University), M.Tech(IETE Hyderabad),
Department of Computer Science Engineering.
Vignana Bharathi Institute Of Technology(VBIT)
Aushapur, Hyderabad

S.Sathwik
M.Tech
Department of Computer Science and Engineering
Vignana Bharathi Institute Of Technology(VBIT)
Aushapur, Hyderabad

Abstract

Cloud computing environment provisions the supp ly of computing resources on the basis of dem and, as and when needed. It builds upon advan ces of virtualization and distributed computing to support cost efficient usage of compu ting resources, emphasizing on resource scalability and on-demand services. It allows business outcomes to scale up and down their resources based on needs. Managing the customer demand creates the challenges of on demand resource allocation. Virtual Machine (VM) techno logy has been employed for resource provisioning. It is expected that using virtualized environment will reduce the average job response time as well as executes the task according to the availability of resources. Hence VMs are allocated to the user based on characteristics of the job. Effective and dynamic utilization of the resources in cloud can help to balance the load and avoid situations like slow run of systems. This paper mainly focuses on allocation of VM to the user, based on analyzing the characteristics of the job. Main principle of this work is that low priority jobs (deadline of the job is high) should not delay the execution of high priority jobs (deadline of the job is low) and to dynamically allocate VM resources for a user job within deadline

Keywords: Cloud computing; Resource allocation & Management; Virtualization; Green Computing.

Introduction

Cloud computing provides a "computing-as a-service" model in which compute resources are made available as a utility service — an illusion of availability of as much resources (e.g., CPU, memory, and I/O) as demanded by the user. Moreover, users of cloud services pay only for the amount of resources (a "pay-as-use" model) used by them. This model is quite different from earlier infrastructure models, where enterprises would invest huge amounts of money in building their own computing infrastructure. Typically, traditional data centers are provisioned to meet the peak demand, which results in wastage of resources during non-peak periods. To alleviate the above problem, modern-day data centers are shifting to the cloud. The important characteristics of cloud-based data centers are:

- Making resources available *on demand*. The operation and maintenance of the data center lies with the cloud provider. Thus, the cloud model enables the users to have a computing environment without investing a huge amount of money to build a computing infrastructure.
- *Flexible resource provisioning*. This provides ability to dynamically scale or shrink the provisioned resources as per the dynamic requirements.
- *Fine-grained metering*. This enables the "pay-as-use" model, that is, users pay only for the services used and hence do not need to be locked into long-term commitments. As a result, a cloud-based solution is an attractive

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

provisioning alternative to exploit the "computing-as-service" model.

However, implementing cloud-based data centers requires a great deal of flexibility and agility. For example, the dynamic scaling and shrinking requirement needs *compute resources* to be made available at very short notice. When computing hardware is overloaded, it may be required to dynamically transfer some of its load to another machine with minimal interruption to the users.

Virtualization technology can provide these kinds of flexibilities.
In a cloud environment, the service provider would like to operate the computing resources at
Optimum utilization levels to meet the service level agreements (SLA) of users. Recommitment of resources can result in SLA violations, whereas underutilization of resources would mean loss of revenue for the provider. Thus, efficient *resource management* is a very critical component in cloud-based solutions. *Virtualization* is a popular solution that acts as a backbone for provisioning requirements of a cloud-based solution. Virtualization provides a "virtualized" view of resources used to instantiate virtual machines (VMs).

A VM monitor (VMM) or hypervisor provides a mechanism for mapping Virtual Machines (VMs) to Physical Resources manages and multiplexes access to the physical resources, maintaining isolation between VMs at all times. As the physical resources are virtualized, several VMs, each of which is self-contained with its own operating system, can execute on a physical machine (PM).

This mapping is 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 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, 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. In this paper, we presented the design and implementation of dynamic resource allocation in the Virtualized Cloud Environment which maintains the balance between the following two goals.

Goals to Achieve:

Overload Avoidance. The capacity of a PM must satisfy the resource needs from all VMs running on it. Or else, the PM is overloaded and leads to provide less performance of its VMs.

Green computing. The number of PMs used should be optimized as long as they could satisfy the needs of all VMs. And Idle PMs can be turned off to save energy. There is an in depth tradeoff between the two goals in the face of changing resource needs from all VMs. To avoid the overload, should keep the utilization of PMs low to reduce the possibility of overload in case the resource needs of VMs increase later. For green computing, should keep the utilization of PMs reasonably high to make efficiency in energy.A VM Monitor manages and multiplexes access to the physical resources, maintaining isolation between VMs at all times. As the physical resources are virtualized, several VMs, each of which is self-contained with its own operating system, can execute on a physical machine (PM).

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

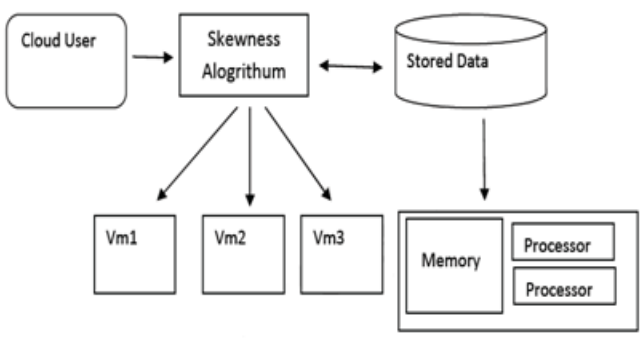
EXISTING SYSTEM LIMITATIONS:

- 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.
- No control over the business assets (data!). The main assets in every company are its data files with valuable customer information.
- Risk of data loss due to improper backups or system failure in the virtualized environment.
- High cost and loss of control.

1.3 PROPOSED SYSTEM

Cloud computing has taken the degree of efficiency and agility realized from virtualization. Virtualization helps efficient use of hardware resources. Hence Virtual Machines are allocated to the user based on their job in order to reduce the number of physical servers in the cloud environment. But most VM resources are not efficiently allocated based on the characteristics of the job to meet out Service Level Agreements (SLA). Hence, we propose a dynamic VM allocation model based on the characteristics of the job, which can dynamically reconfigure virtual resources and thereby increasing the resource utilization.

This paper presents the design and implementation of an automated resource management system that achieves a good balance between the two goals. We make the following contributions. 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 degraded performance of its VMs. Green computing: 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. Hence, develop a resource allocation system that can avoid overload in the system effectively while minimizing the number of servers used. Here, the concept of "skewness" is used to measure the uneven utilization of a server.

By minimizing skewness, the overall utilization of servers in the face of multidimensional resource constraints is increased. We are using cloud sim for implementations.

Cloud computing has taken the degree of efficiency and agility realized from virtualization. Virtualization helps efficient use of hardware resources. Hence Virtual Machines are allocated to the user based on their job in order to reduce the number of physical servers in the cloud environment. But most VM resources are not efficiently allocated based on the characteristics of the job to meet out Service Level Agreements (SLA). Hence, we propose a dynamic VM allocation model based on the characteristics of the job, which can dynamically reconfigure virtual resources and thereby increasing the resource utilization.

When all existing resources (VMs) are allocated to low priority jobs and a high priority job comes in, the low priority job (deadline is high) has to be preempted its resources allowing a high priority job (deadline is low) to run in its resource. When a job arrives, availability of the VM is checked. If the VM is available then job is allowed to run on the VM. If the VM is not available then the algorithm find a low priority job taking into account the job's lease type. The low priority job is paused its execution by preempting its resource. The high priority job is allowed to run on the resources preempted from the low priority.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

When any other job running on VMs are completed, the job which was paused early can be resumed if the lease type of the job is suspend able. If not the suspended job has to wait for the completion of high priority job running in its resources, so that it can be resumed.

The lease types associated with the jobs are

Cancellable: These requests can be scheduled at any time after their arrival time. It need not be resumed later. Cancellable leases do not guarantee the deadline.

Suspendable: Leases of this type can be suspended at any time but should be resumed later. This type of lease guarantees the execution but not in a specific deadline. Suspendable leases are flexible in start time and can be scheduled at any time after their ready time. In the case of preemption, these leases should be rescheduled to find another free time-slot for the remainder of their execution.

Non-Preemptable: The leases associated with such requests cannot be preempted at all.

ADVANTAGES

- ✓ A flexible, scalable infrastructure management platform has been architected and a prototype implemented
- ✓ Measurement of resource usage and end user activities lies hands of the cloud service provider.
- ✓ Opaque cost structure due to highly flexible usage of cloud services.
- ✓ Stable of cost structure.

The proposed system consists of number of servers, predictor, hotspot and cold spot solvers and migration 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.

System Overview

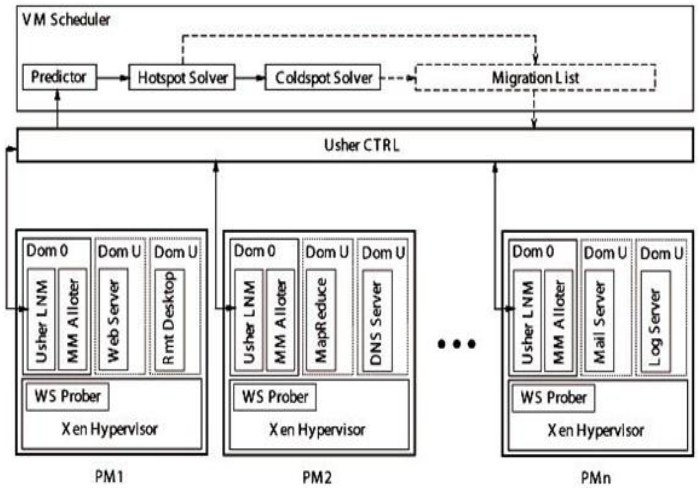


Fig. 1. System Architecture

The architecture of the system is presented in Figure 1. Each physical machine (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. We assume all PMs Share backend storage. The multiplexing of VMs to PMs is managed using the Usher framework. 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 User 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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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 Skewess

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

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.

Hot and Cold Spots

Proposed 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. 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:

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

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. 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. However, we do so only when the average resource utilization of all actively used servers (i.e., APMs) in the system is below a green computing threshold. A server is actively used if it has at least one VM running. Otherwise, it is inactive. 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. Different types of resources can have different thresholds. For example, we can define the hot thresholds for CPU and memory resources to be 90 and 80 percent, respectively. Thus a server is a hot spot if either its CPU usage is above 90 percent or its memory usage is above 80 percent.

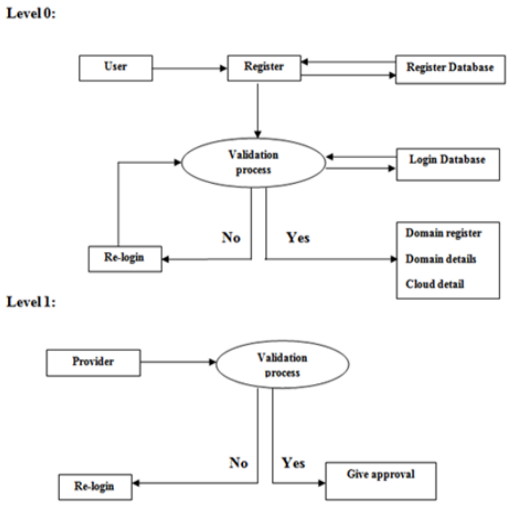


Fig. 2Hotspot and cold spot

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. The challenge here is to reduce the number of active

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

servers during low load without sacrificing performance either now or in the future. We need to avoid oscillation in the system. 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 underutilized 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 share back-end storage.

Hence, the cost of a VM live migration is determined mostly by its memory footprint. Section 7 in the supplementary file explains why the memory is a good measure in depth. We 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 underutilized 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.

The above consolidation adds extra load onto the related servers. This is not as serious a problem as in the hot spot mitigation case because green computing is initiated only when the load in the system is low. Nevertheless, we want to bind the extra load due to server consolidation. We restrict the number of cold spots that can be eliminated in each run of the algorithm to be no more than a certain percentage of active servers in the system. This is called the consolidation limit. Note that we eliminate cold spots in the system only when the average load of all active servers (APMs) is below the green computing threshold. Otherwise, we leave those cold spots there as potential destination machines for future offloading. This is consistent with our philosophy that green computing should be conducted conservatively.

Results and Discussion:

The goal of the skewness algorithm is to mix workloads with different resource requirements together so that the overall utilization of server capacity is improved. In this experiment, we see how our algorithm handles a mix of CPU, memory, and network intensive workloads. Resource allocation status of three servers A, B, C has total memory allocated 500KB each and resource used memory for serverA 0KB,serverB 10KB and serverC 0K. In Fig. 4 each cloud users provide cloud service Resource allocation in green computing. In Fig.5 display Server usage and resource allocation status for user1 using Bar Chart. The cloud computing is a model which enables on demand network access to a shared pool computing resources. Cloud computing environment consists of multiple customers requesting for resources in a dynamic environment with their many possible constraints. The virtualization can be the solution for it. It can be used to reduce power consumption by data centers. The main purpose of the virtualization is that to make the most efficient use of available system resources, including energy. A data center, installing virtual infrastructure allows several operating systems and applications to run on a lesser number of servers, it can help to reduce the overall energy used for the data center and the energy consumed for its cooling. Once the number of servers is reduced, it also means that data center can reduce the building size as well. Some of the advantages of Virtualization which directly

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

impacts efficiency and contributes to the environment include: Workload balancing across servers, Resource allocation and sharing are better monitored and managed and the Server utilization rates can be increased up to *80%* as compared to initial *10-15%.*

Resource Allocation Status

The results are clear and having good contribution:
1) Allocation of resource is done dynamically.
2) Saves the energy using the green computing concept
3) Proper utilization of servers and memory utilization is taken care using skewness.
4) Minimize the total cost of both the cloud computing infrastructure and running application

CONCLUSION

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

REFERENCES

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

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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," in *Proc. of the USENIX Symposium on Networked Systems Design and Implementation (NSDI'08)*, Apr. 2008.

[8] 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.

[9] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing sla violations," in *Proc. of the IFIP/IEEE International Symposium on Integrated Network Management (IM'07)*, 2007.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

[10] "TPC-W: Transaction processing performance council, http://www.tpc.org/tpcw/."

[11] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *Proc. Of the International World Wide Web Conference (WWW'07)*, May 2007.

[12] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, "Quincy: Fair scheduling for distributed computing clusters," in *Proc. of the ACM Symposium on Operating System Principles (SOSP'09)*, Oct. 2009.

[13] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling," in *Proc. of the European conference on Computer systems (EuroSys'10)*, 2010.

[14] T. Sandholm and K. Lai, "Mapreduce optimization using regulated dynamic prioritization," in *Proc. of the international joint conference on Measurement and modeling of computer systems (SIGMETRICS'09)*, 2009.

[15] 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.

[16]Y. Toyoda, "A simplified algorithm for obtaining approximate solutions to zero-one programming problems," Management Science, vol. 21, pp. 1417–1427, august 1975.

[17] 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.

[18] 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.

[19] 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.

[20] 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 (EuroSys'12), 2012.

[21] T. Wood *et al.*, "CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines," *Int'l. Conf. Virtual Execution Environments*, 2011.

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

[23] 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.

[24] 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.