International Journal of Scientific Engineering and Technology Research

ISSN 2319-8885
Vol.03,Issue.33
October-2014,
Pages:6519-6524

www.ijsetr.com

Memory Resource Management in Cloud Environment

FOUZIA BEGUM[1], MR T.K. SHAIK SHAHVALI[2]

[1]PG Scholar, Dept of CSE, Lords Institute of Engineering & Technology, Hyderabad, India, Email: fouz504@gmail.com.
[2]Professor, Dept of CSE, Lords Institute of Engineering & Technology, Hyderabad, India, Email: sshavali.Liet@gmail.com.

Abstract: Cloud computing allows business customers to scale up and down their resource usage based on needs. Many of the touted gains in the cloud model come from resource multiplexing through virtualization technology. 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 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. We develop a set of heuristics that prevent overload in the system effectively while saving energy used. Trace driven simulation and experiment results demonstrate that our algorithm achieves good performance.

Keywords: Cloud Computing, Resource Management, Virtualization, Green Computing.

I. INTRODUCTION

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: how can a cloud service provider best multiplex its virtual resources onto the physical hardware? 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 underutilized 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. The capacity of PMs can also be heterogenous because multiple generations of hardware coexist in a data center.

We aim to achieve two goals 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 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.

There is an inherent tradeoff between the two goals in the face of changing resource needs of VMs. For overload avoidance, we 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, we should keep the utilization of PMs reasonably high to make efficient use of their energy.

In this paper, we present the design and implementation of an automated resource management system that achieves a good balance between the two goals. We make 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 multidimensional resource constraints.

- We design a load prediction algorithm that can capture the future resource usages of applications accurately without looking inside the VMs. The algorithm can capture the rising trend of resource usage patterns and help reduce the placement churn significantly.

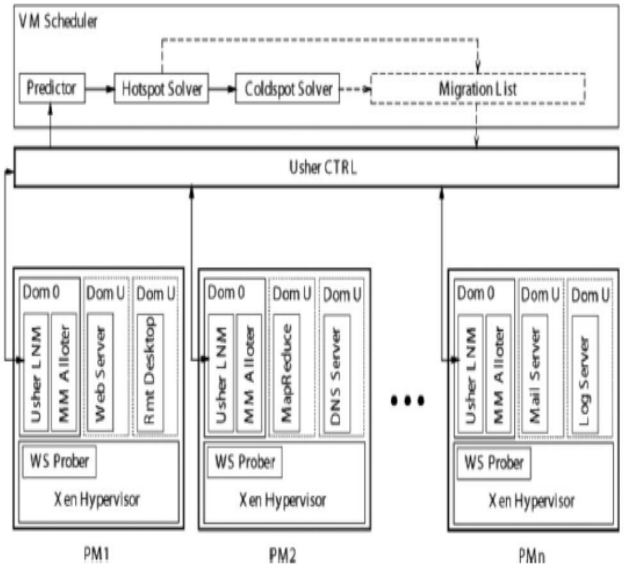


Fig.1. System architecture.

The rest of the paper is organized as follows. SectionII provides Related Works and SectionIII describes Skewness Algorithm. SectionsIV present simulation and experiment results, respectively. SectionV concludes.

II. RELATED WORKS

Cloud Resources Provisioning scheme, which is flexible enough to adopt to the various general MCC reference use cases being described. The main feature of the employed MCC Services Admission Control algorithm lies in the fact that it jointly handles radio and computing resources rather than confronting the problem as two independent resource management sub-problems The queuing model to optimize the resource allocation for multimedia cloud in priority services scheme**.** Which specifically formulate and solve the resource cost minimization problem and the service response time minimization problems. An optimal cloud resource provision (OCRP) algorithm is proposed by formulating a stochastic programming model, The (OCRP) algorithm can provisioning computing resources for being used in multiple provisioning stages as well as a long term plan, for example four stages in a quarter plan and twelve stages in a annual plan. The demand and price uncertainty is considered in OCRP. In this paper we purposed that different approaches to obtain the solution of the OCRP algorithm are considered including deterministic equivalent formulation, average approximation, and Benders decomposition.

Numerical studies are extensively performing in which the results clearly show that with the OCRP algorithm cloud consumer can successfully minimized the average cost of resource provisioning in cloud computing environment. The OCRP algorithm shows that we can find an optimal solution for resource provisioning and VM placement. It uses only two un-certainties only the need and price. In this paper RCRP algorithm is used which is an extension of OCRP where four uncertainty factors r considered. Grid providing services which are not of desired quality one of the major uncertain factors of grid is single point of failure where one unit on the grid de-grades which will cause the entire system to de-grade. Hence suggests cloud which is used for adaption of many services. The benefit of cloud is that it will prevent single point of failure and also will decrease hardware costing. Resource allocation strategies (RAS) at a glance the input parameters to RAS and the way of resource allocation vary based on the services and infrastructure and the nature of applications which will demand resources. The schematic diagram depicts the classification of Resource Allocation Strategies (RAS) proposed in cloud paradigm. The following section discusses the RAS employed in cloud.

A. Execution Time

Different kinds of resource allocation mechanisms are proposed in cloud. The actual task execution time and pre-emptible scheduling is considered for various resource allocations. It overviews the problem of resource contention and increases resource utilization by using different modes of renting computing capacities but estimating the execution time for a job is a hard task for a user and errors are made very often. But the VM model considered in is heterogeneous and proposed for IaaS.

B. Policy

Since centralized user and resource management lacks in scalable management of users, resources and organization-level security policy, we proposed a decentralized user and virtualized resource management for IaaS by adding a new layer called domain in between the user and the virtualized resources. Based on role based access control (RBAC), virtualized resources are allocated to users through domain layer.

C. Virtual Machine (VM)

A system which can automatically scale it's infrastructure resources is designed the system composed of a virtual network of virtual machines capable of live migration across multi- domain physical infrastructure Cloud computing services providers deliver their resources based on virtualization to satisfy the need of users. In cloud computing, the amount of resources required can vary preserve request. Therefore the providers have to offer Different amounts of virtualized resources per request. To provide worldwide service, a provider may have data centers that are geographically distributed through-out the world. The user locations vary in geographical locations. Since cloud computing services are delivered over the internet there may be undesirable response latency between the users and the database. Hence, for the best recent service, the provider needs to find a data center and physical machine that has a light workload and is geographically close to the users. The proposed model finds the best match for the user requests based on two evaluations: The geo-graphical distances

between the user and database center and the workload of data center this model allows the users to find a data center that is guaranteed to be the closest distance and have the light workload and it finds a light workload physical machine within the data center for a provider.

III. SKEWNESS ALGORITHM

We introduce a concept skewness which would be useful to measure the variable utilization of the server. By minimizing skewness we can find the various utilization of the servers.

- **Hot spot** is a small area in which there is relatively higher temperature than the surroundings.
- **Cold spot** is the area in which there is a decrease in ambient temperature.

Here we use the hot spot and cold spot to just explain the way in which the green computing algorithm has been used. The threshold technology is thus maintained here to make it clearer. The overload avoidance and the green computing concept are being used to make the resource management precise.

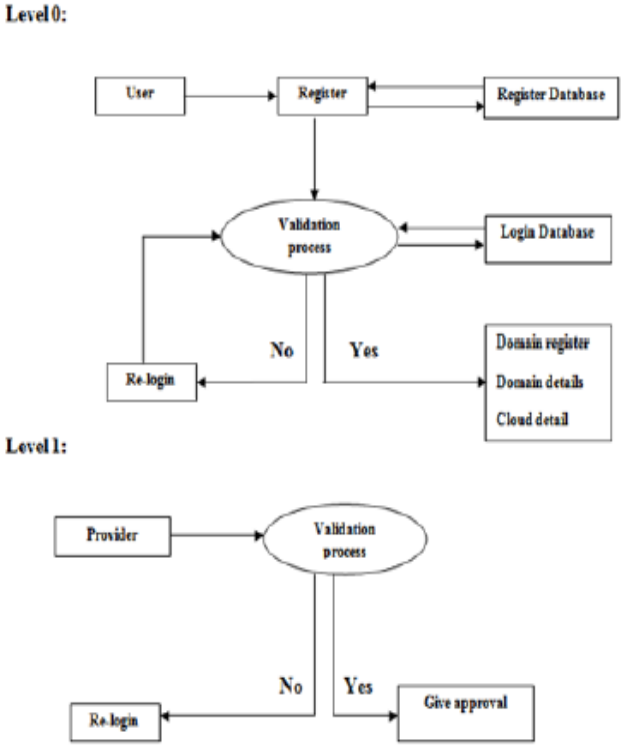


Fig.2. Hotspot and cold spot.

Our algorithm evaluates the allocation of resources based on the demands of VM. Here we define the server a hotspot and if the utilization exceeds the above the hot threshold then it symbolizes that the server is overloaded and Vm's are moved away. The temperature is zero when the server is not a hot spot. We define a cold spot when the utilization of the resources is below the clod threshold which indicates that the server is idle and it has to be turned off in order to save energy. This is done when mostly all servers are actively used below the green computing threshold else it is made inactive.

1. Hot spot mitigation

The sorted lists of hot spots are arranged in a order so that we can eliminate them else keep the temperature low. Our goal is to move away the VM's that can reduce the server's temperature. Among all we select the one which can reduce skewness.

2. Green computing

Green computing aims to attain economic viability and improve the way computing devices are used. It is the environmentally responsible and eco-friendly use of computers and their resources. When the resources utilization of servers is low in such cases they are turned off wherein we use this green computing algorithm. The very important challenge here is to reduce the number of actively participating servers. Thus we have to avoid oscillation in the system. Our algorithm is used when utilization of all active servers are below the green computing threshold. Dynamic resource management has become an active area of research in the Cloud Computing paradigm. Cost of resource varies significantly depending on configuration for using them. Hence efficient management of resource is of prime interest to both Cloud Provider and Cloud Users. The success of any cloud management software critically depends on the flexibility; scale and efficiency with which it can utilize the underlying hardware resource while providing 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 resource. VM live migration is 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.

IV. SIMULATIONS

We evaluate the performance of our algorithm using trace driven simulation. Note that our simulation uses the same code base for the algorithm as the real implementation in the experiments. This ensures the fidelity of our 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 simulator can read them. We collected the traces from a variety of sources:

- Web Info Mall The largest online Web archive in China (i.e., the counterpart of Internet Archive in the US) with more than three billion archived Web pages.
- Real Course the largest online distance learning system in China with servers distributed across 13 major cities.
- Amazing Store The largest P2P storage system in China.

We also collected traces from servers and desktop computers in our university including one of our mail servers, the central DNS server, and desktops in our department. We post processed the traces based on days collected and use random sampling and linear combination of the data sets to generate the workloads needed. All simulation in this section uses the real trace workload unless otherwise specified.

TABLE I: Parameters in our Simulation

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

The default parameters we use in the simulation are shown in Table 1. We used the FUSD load prediction algorithm with "↑ α = -0.2, ↓ α = 0.7, and W = 8. In a dynamic system, those parameters represent good knobs to tune the performance of the system adaptively. We choose the default parameter values based on empirical experience working with many Internet applications. In the future, we plan to explore using AI or control theoretic approach to find near optimal values automatically.

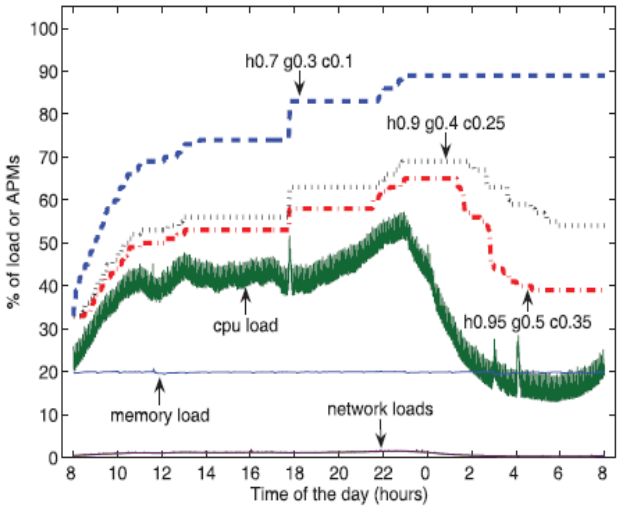


Fig.3. Impact of thresholds on the number of APMs

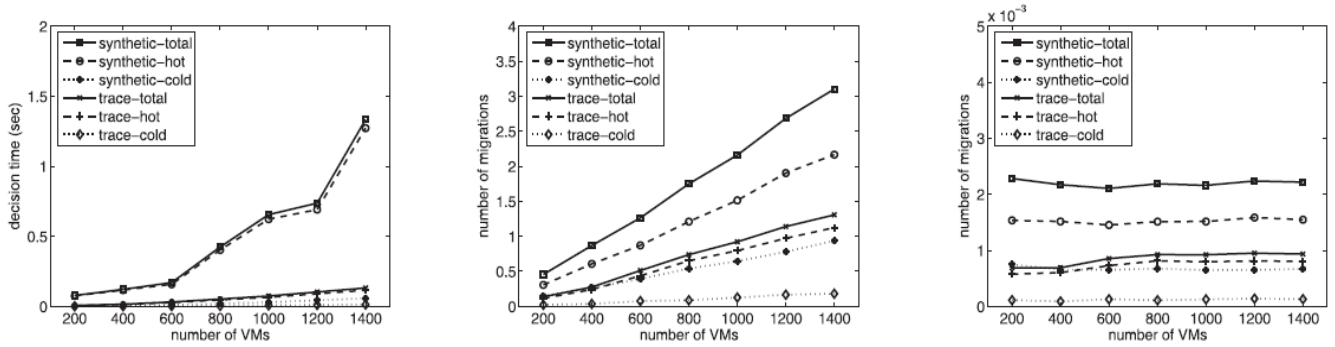
A. Effect of Thresholds on APMs

We first evaluate the effect of the various thresholds used in our algorithm. We simulate a system with 100 PMs and 1,000 VMs (selected randomly from the trace). We use random VM to PM mapping in the initial layout. The scheduler is invoked once per minute. The bottom part of Fig.3 shows the daily load variation in the system. The x-axis is the time of the day starting at 8 am. 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.

The top part of Fig.3 shows how the percentages of APMs vary with the load for different thresholds in our algorithm. For example, "h0.7 g0.3 c0.1" means that the hot, the green computing, and the cold thresholds are 70, 30, and 10 percent, respectively. Parameters not shown in the figure take the default values in Table 1. Our algorithm can be made more or less aggressive in its migration decision by tuning the thresholds. The figure shows that lower hot thresholds cause more aggressive migrations to mitigate hot spots in the system and increases the number of APMs, and higher cold and green computing thresholds cause more aggressive consolidation which leads to a smaller number of APMs. With the default thresholds in Table 1, the percentage of APMs in our algorithm follows the load pattern closely. To examine the performance of our algorithm in more extreme situations, we also create a synthetic workload which mimics the shape of a sine function (only the positive part) and ranges from 15 to 95 percent with a 20 percent random fluctuation. It has a much larger peak-to-mean ratio than the real trace of the supplementary file, which can be found on the Computer Society Digital Library.

B. Scalability of the Algorithm

We evaluate the scalability of our algorithm by varying the number of VMs in the simulation between 200 and 1,400. The ratio of VM to PM is 10:1. The results are shown in



(a) average decision time (b) average number of migrations (c) number of migrations per VM

Fig.4. Scalability of the algorithm with system size.

Fig.4. Fig.4a shows that the average decision time of our algorithm increases with the system size. The speed of increase is between linear and quadratic. We break down the decision time into two parts: hot spot mitigation (marked as "hot") and green computing (marked as "cold"). We find that hot spot mitigation contributes more to the decision time. We also find that the decision time for the synthetic workload is higher than that for the real trace due to the large variation in the synthetic workload. With 140 PMs and 1,400 VMs, the decision time is about 1.3 seconds for the synthetic workload and 0.2 second for the real trace. Fig. 4b shows the average number of migrations in the whole system during each decision. The number of migrations is small and increases roughly linearly with the system size.

We find that hot spot contributes more to the number of migrations. We also find that the number of migrations in the synthetic workload is higher than that in the real trace. With 140 PMs and 1,400 VMs, on average each run of our algorithm incurs about three migrations in the whole system for the synthetic workload and only 1.3 migrations for the real trace. This is also verified by Fig. 4c which computes the average number of migrations per VM in each decision. The figure indicates that each VM experiences a tiny, roughly constant number of migrations during a decision run, independent of the system size. This number is about 0.0022 for the synthetic workload and 0.0009 for the real trace. This translates into roughly one migration per 456 or 1,174 decision intervals, respectively. The stability of our algorithm is very good. We also conduct simulations by varying the VM to PM ratio. With a higher VM to PM ratio, the load is distributed more evenly among the PMs. The results are presented in the supplementary file, which is available online.

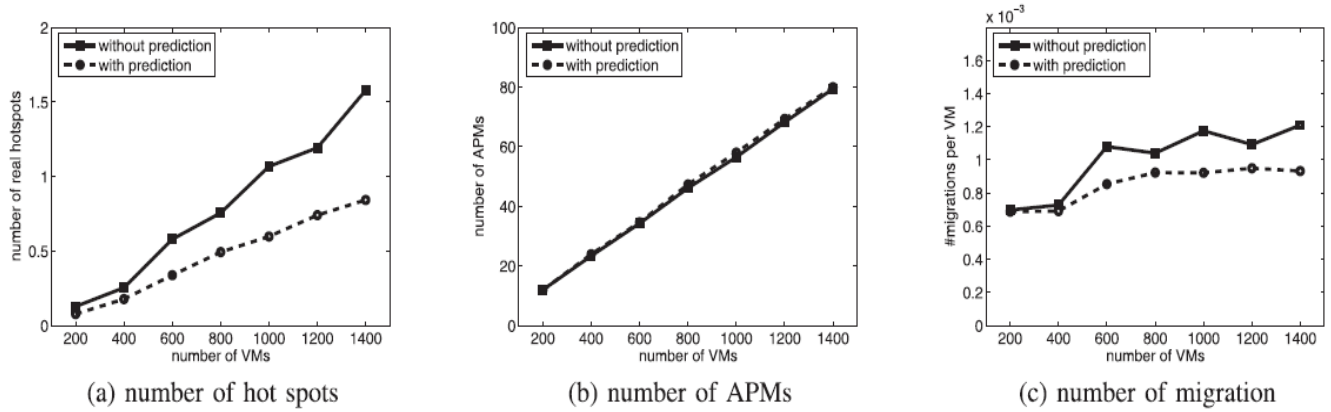


Fig.5. Effect of load prediction.

C. Effect of Load Prediction

We compare the execution of our algorithm with and without load prediction in Fig. 5. When load prediction is disabled, the algorithm simply uses the last observed load in its decision making. Fig.5a shows that load prediction significantly reduces the average number of hot spots in the system during a decision run. Notably, prediction prevents over 46 percent hot spots in the simulation with 1,400 VMs. This demonstrates its high effectiveness in preventing server overload proactively. Without prediction, the algorithm tries to consolidate a PM as soon as its load drops below the threshold. With prediction, the algorithm correctly foresees that the load of the PM will increase above the threshold shortly and hence takes no action. This leaves the PM in the "cold spot" state for a while. However, it also reduces placement churns by avoiding unnecessary migrations due to temporary load fluctuation. Consequently, the number of migrations in the system with load prediction is smaller than that without prediction as shown in Fig. 5c. We can adjust the conservativeness of load prediction by tuning its parameters, but the current configuration largely serves our purpose (i.e., error on the side of caution). The only downside of having more cold spots in the system is that it may increase the number of APMs. This is investigated in Fig. 5b which shows that the average numbers of APMs remain essentially the same with

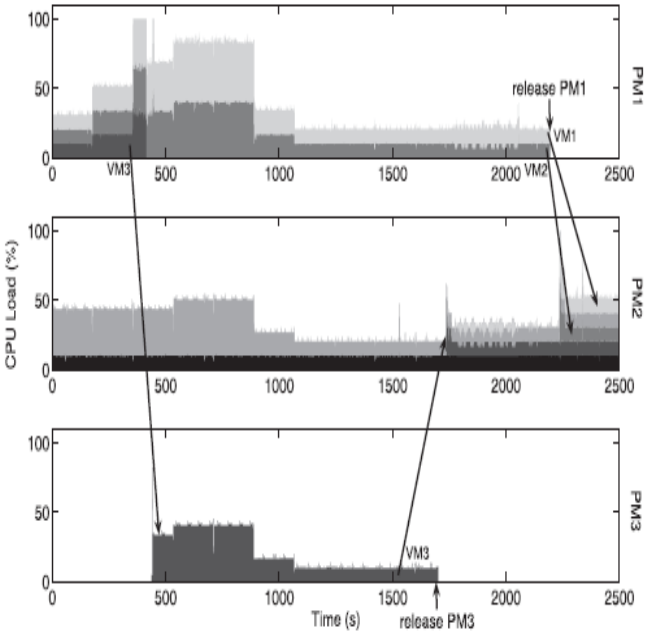


Fig.6. Algorithm effectiveness.

or without load prediction (the difference is less than 1 percent). This is appealing because significant overload protection can be achieved without sacrificing resources efficiency. Fig. 5c compares the average number of migrations per VM in each decision with and without load prediction. It shows that each VM experiences 17 percent fewer migrations with load prediction we start with a small scale experiment consisting of three PMs and five VMs so that we can present the results for all servers in Fig. 6.

V. 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.

VI. REFERENCES

[1] Zhen Xiao, Senior Member, IEEE, Weijia Song, and Qi Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment", IEEE Transactions on Parallel and Distributed Systems, Vol. 24, No. 6, June 2013.

[2] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.

[3] L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, 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," Proc. ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.

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

[6] 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.

[7] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," Proc. USENIX Ann. Technical Conf., 2005.

[8] M. McNett, D. Gupta, A. Vahdat, and G.M. Voelker, "Usher: An Extensible Framework for Managing Clusters of Virtual Machines," Proc. Large Installation System Administration Conf. (LISA '07), Nov. 2007.

[9] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-Box and Gray-Box Strategies for Virtual Machine Migration," Proc. Symp. Networked Systems Design and Implementation (NSDI '07), Apr. 2007.

[10] C.A. Waldspurger, "Memory Resource Management in VMware ESX Server," Proc. Symp. Operating Systems Design and Implementation (OSDI '02), Aug. 2002.

[11] 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.

[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," Proc. ACM European conf. Computer Systems (EuroSys '09), 2009.