Towards Energy Efficient Change Management in A Cloud Computing Environment

Hady AbdelSalam[1], Kurt Maly[1], Ravi Mukkamala[1], Mohammad Zubair[1], and David Kaminsky[2]

[1] Computer Science Department, Old Dominion University, Norfolk, VA 23529, USA
{asalam, maly, mukka, zubair}@cs.odu.edu
[2] Strategy and Technology, IBM, Research Triangle, Raleigh, NC 27709, USA
dlk@us.ibm.com

Abstract. The continuously increasing cost of managing IT systems has led many companies to outsource their commercial services to external hosting centers. Cloud computing has emerged as one of the enabling technologies that allow such external hosting efficiently. Like any IT environment, a Cloud Computing environment requires high level of maintenance to be able to provide services to its customers. Replacing defective items (hardware/software), applying security patches, or upgrading firmware are just a few examples of the typical maintenance procedures needed in such environments. While taking resources down for maintenance, applying efficient change management techniques is a key factor to the success of the cloud. As energy has become a precious resource, research has been conducted towards devising protocols that minimize energy consumption in IT systems. In this paper, we propose a pro-active energy efficient technique for change management in cloud computing environments. We formulate the management problem into an optimization problem that aims at minimizing the total energy consumption of the cloud. Our proposed approach is pro-active in the sense that it takes prior SLA (Service Level Agreement) requests into account while determining time slots in which changes should take place.

Key words: Cloud Computing, Autonomic Manager, , Change Management, Energy Efficient.

1 Introduction

IT companies have been looking for solutions that enable users to access computing power of supercomputers with thin client devices. The cloud computing concept has emerged as an appropriate solution that attracted the attention of large IT companies such as IBM and Google. A cloud [3, 5] can be defined as a pool of computer resources that can host a variety of different workloads, including batch-style back-end jobs and interactive user applications. A cloud computing platform dynamically provisions, configures, reconfigures, and de-provisions servers as needed. Several commercial realizations of computing clouds are already available today (e.g., Amazon, Google, IBM, and Yahoo, . . . etc).

Due to the tremendous increase in energy costs in the past few years, it is expected that efficient power management will play an essential role in the success of large IT environments such as computing clouds. Power management in such environments can be very challenging when hundreds or even thousands of servers are located within a relatively small area. The impact of high power consumption is not only limited to the energy cost but it also extends to the cost of initial investment of cooling systems to get rid of the generated heat and the continuous cost needed to power these systems. In order to reduce operational cost at these centers while meeting any performance based SLAs (Service Level Agreement), several techniques have been proposed in the literature. Among these techniques, we can find hardware techniques such as processor throttling and dynamic frequency scaling and also software techniques such as virtualization and Advanced Configuration and Power Interface (ACPI) standards.

Servers in computing clouds typically go through several software and hardware upgrades. Change management is the process that determines the appropriate time in which these upgrades should take place without violating any of the underlying SLAs. Not only maintaining and upgrading the cloud should occur with minimal disruption and administrative support but it also should go hand in hand with power management to guarantee efficient utilization of the cloud resources.

In previous work published in [1], we proposed and implemented a basic autonomic manager for change management that can determine whether applying a specific change would violate any of the system SLAs. We extended our autonomic manager in [2] by integrating it with a scheduler that can simplify change management by proposing time slots in which changes can be applied without violating any of SLAs requirements. Motivated by the importance of developing energy efficient techniques for computing cloud environments, in this paper we extend our previous work by proposing a pro-active, energy-aware technique for change management in IT environments governed by SLAs requirements such as computing clouds. The remainder of the paper is organized as follows. In section 2, we describe the architecture of our cloud environment. The details of our proactive approach is presented in section 3. Section 4 offers conclusions and describes future work.

2 Cloud Architecture

Figure 1 shows one possible architecture for a cloud. In this architecture, requests from cloud clients get into the system through the cloud gateway. After necessary authentication and based on the current load on the servers, the load balancing module forwards client requests to one of the clouds servers dedicated to support this type of requests. This implies that the load balancing module at the cloud gateway should have up-to-date information about which client applications are running on which servers and the current load on these servers. One possible way to make such information available is to integrate the gateway and load balancing controller with the job scheduler in order to be able to distribute the load on cloud servers correctly.

From statistical and historical information of client applications, SLA requirements can be mapped into MIPS. The MIPS requirements are then used to determine the number of servers and their operating frequencies in order to minimize total power consumption. Once the jobs scheduler module determines the number of servers and the optimal running frequencies, client applications are being assigned to servers based on the SLA requirements of each client. Typically, this process may involve running several instance of the same application on different servers and distributing requests of users of the same client over these servers based on their current load.

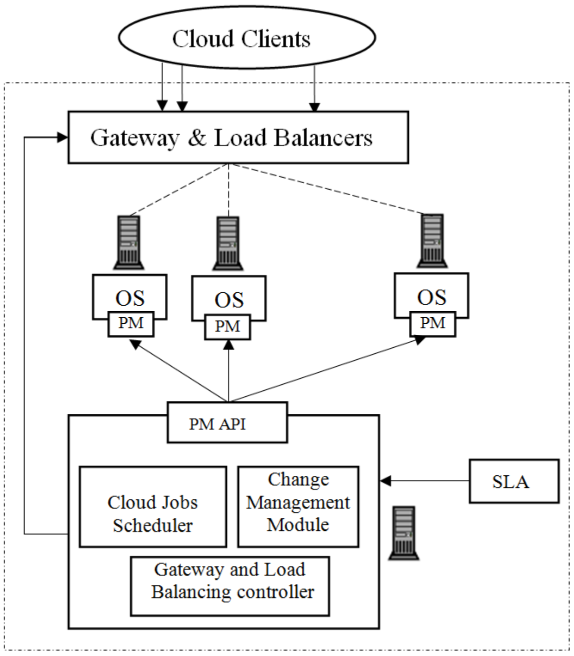

Fig. 1. Cloud Architecture

Mapping cloud SLA requirements into MIPS and then into a number of servers and their running frequencies implies that the power management (PM) module at the cloud central controller should interact with the power management module on each server. Through this interaction, the central controller can change the running frequencies of these servers when necessary. In section 3, we show how the number of servers and the running frequencies are actually determined under different cloud workloads.

3 Pro-Active Change Management

A cloud consists of one or more server groups; each group has a number of servers that are identical in hardware and software configuration. Thus, all the servers in the same group are equally capable of running any client application. Our proactive approach will be presented in section 3 assuming a homogeneous cloud where all the servers are identical (e.g. one server group). It can be extended straightforwardly to heterogeneous clouds by restricting the users of each client to one server group and looking at the heterogeneous cloud as a group of homogeneous sub-clouds.

Cloud clients sign a service level agreement with the company running the cloud. In this agreement, each client determines its computing needs by aggregating the processing needs of its user applications, the expected number of users, and the average response time per user request. To be able to estimate the computing power (MIPS) needed to achieve the required response time, the client should provide the cloud administrators with any necessary information about the type of the queries expected from its users. One way of doing this is through providing a histogram that shows the frequency of each expected query. Cloud administrators run these queries on testing servers and estimate their computing needs based on response time requirements. Once a client has been running its services on the cloud for some time, the computing needs requirements can be updated from actual statistics from the cloud.

Average response time for a user query depends on many factors, (i.e. the nature of the application, the configuration of the server running the application, and the current load on the server when running the application). The focus of this paper is on clouds supporting interactive applications like web services and web applications. In the future we will address other types of clouds[3]. Under this assumption, we can approximate the minimum average response time constraint as determined by the SLA by the minimum number of instructions that the application is allowed to execute every second. This kind of approximation can be easily done for interactive applications. For example, assuming that the response time for a user query was measured to be t seconds, when the query runs solely on x MIPS server. Now, if we run the same query along with other queries on the same server, to guarantee a minimum average response time of r seconds, the computing power dedicated for the query must be at least $M = \frac{t*x}{r}$ MIPS. Hence, when a new user joins the cloud, the load balancer module should forward requests of the new user to one of the cloud servers such that it has at least M MIPS of its computing power available. Based on client SLAs and the expected number of users for each client, the cloud should pro-actively provide enough number of servers to satisfy required response time.

Advanced hardware-based power management techniques such as dynamic voltage/frequency scaling (DVS) [4] allow servers to run on a wide range of

[3] For computing intensive applications, the best strategy may be to assign these applications to one or more powerful servers running on their top speed so they can finish as soon as possible. This strategy would give servers a chance to remain idle for longer periods saving their total energy consumption

operating frequencies. The relationship between the power consumption and the running frequency takes the form $P = A + B * f^3$, where A and B are constants that depend on the hardware configuration of the server. The most important feature of the cubic relationship shown above is that the first term in the RHS $[A]$ accounts for power consumption of components that do not scale with the operating frequency, while the second term $[B * f^3]$ accounts for changes in CPU power consumption when the operating frequency changes.

Assuming that the total computing workload of all cloud clients is L_T, and that the cloud has a large number of servers, each of them can run on a range of operating frequencies, we are interested to determine the optimal number of servers to use and the corresponding operating frequency of each server such that power consumption is a minimum. We have proven that the optimal number of servers is given by $k = \sqrt[3]{\frac{2B}{A}} L_T$, and the optimal operating frequency is $f_i = \frac{L_T}{k}$. Unfortunately, the optimal values for k and f_i depend on L_T which in most scenarios changes periodically over time, based on the type and number of user requests determined by SLAs (see figure 2). The length of the period depends on the constituting SLAs (e.g. one day, one week, or even one month). To include the dynamic nature of the total cloud load $L_T(t)$ into our model, we divide each period into what we call running segments or running slots. During the same segment, the cloud total load is fixed, however it can change from one segment to the next. Under this assumption, we can model the total cloud load as shown in the example in figure 2. In that illustration, we assumed a cloud with three different SLAs, and a periodicity of one day. Figure 2 also shows the total cloud load on the system as the sum of the computing load of individual SLAs. Furthermore, we eliminate minor changes in the total cloud load by approximating the load on the cloud by a tight upper bound envelope.

Based on the cloud total load shown in the curve in figure 2 and using the formulae shown above, we can pro-actively evaluate optimal values for k and f_i for all upcoming segments and prepare required resources ahead of time. Hence, when the running segments starts, there will be enough resources to serve users requests satisfying their response time requirements. The number of idle servers in each time segment equals the difference between the total number of cloud servers and $k(t)$. These idle servers are optimal candidates for pending change management requests.

4 Conclusions

In this paper we have described a pro-active energy-aware change management scheme for cloud computing environments that serve primarily clients with interactive applications such as web applications and web services. We used the cubic relationship between power consumption and the frequency at which a server is running to develop an analytical description of the optimal number of servers and the corresponding optimal running frequencies needed to satisfy clients' SLAs. We further describe how these formulae can be used to determine time intervals when servers become available for change management and how

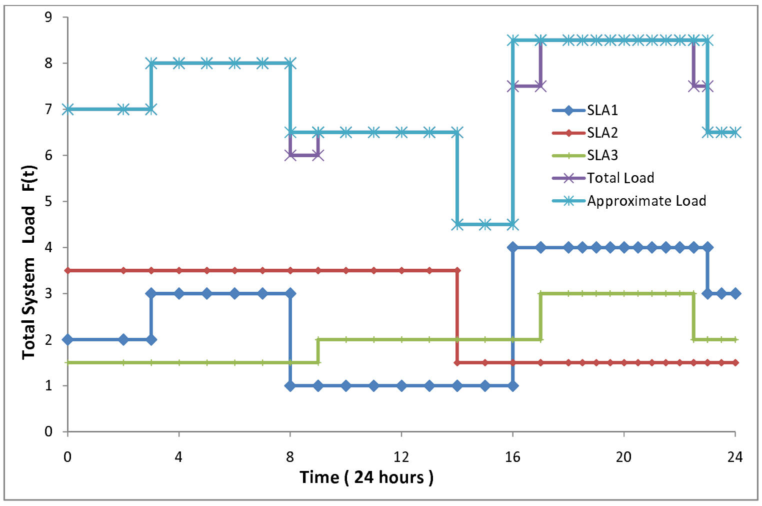

Fig. 2. Actual and Approximated Total Load Due to Several SLAs.

many such servers will be available. Thus we have provided mechanisms on how to avoid over-provisioning of clouds and how to minimize power consumption for a given set of SLAs.

References

1. H. AbdelSalam, K. Maly, R. Mukkamala, and M. Zubair. Infrastructure-aware autonomic manager for change management. In *POLICY '07: Proceedings of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 66–69, Washington, DC, USA, 2007. IEEE Computer Society.
2. H. AbdelSalam, K. Maly, R. Mukkamala, M. Zubair, and D. Kaminsky. Scheduling-capable autonomic manager for policy based it change management system. In *EDOC'08: Proceedings of the 12th IEEE International Enterprise Computing Conference*, Mnchen, Germany., September 2008.
3. G. Boss, P. Malladi, D. Quan, L. Legregni, and H. Hall. Cloud computing. *High Performance On Demand Solutions (HiPODS)*, October 2007.
4. Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 303–314, New York, NY, USA, 2005. ACM.
5. H. Erdogmus. Cloud computing. *IEEE Software*, 26(2):46, 2009.