

## *Applying Workload Shaping Toward Green Cloud Computing*

Woongsup Kim<sup>†</sup>

Dept. of Computer & Information Communications Engineering  
Dongguk University, Seoul, Korea

### **Abstract**

Energy costs for operating and cooling computing resources in Cloud infrastructure have increased significantly up to the point where they would surpass the hardware purchasing costs. Thus, reducing the energy consumption can save a significant amount of management cost. One of major approach is removing hardware over-provisioning. In this paper, we propose a technique that facilitates power saving through reducing resource over provisioning based on virtualization technology. To this end, we use dynamic workload shaping to reschedule and redistribute job requests considering overall power consumption. In this paper, we present our approach to shape workloads dynamically and distribute them on virtual machines and physical machines through virtualization technology. We generated synthetic workload data and evaluated it in simulating and real implementation. Our simulated results demonstrate our approach outperforms to when not using no workload shaping methodology.

**Key words :** Cloud Computing, Traffic Shaping, Green Computing

### **1. INTRODUCTION**

Cloud Computing is associated with a new paradigm for the virtually unlimited provision of computing infrastructure [1]. In cloud, resources are transparently provisioned by the cloud provider and hence resources are unlimitedly provisioned virtually. This new paradigm facilitates to reduce costs associated with the acquisition and management of hardware and software resources. However, energy costs for operating and cooling such cloud resources have increased significantly up to the point where they would surpass the hardware resource purchasing costs. Thus, reducing the energy consumption can save a significant amount of management cost. Moreover, heat generated from operating multiple computing devices would increase the probability of system failure. Therefore, reducing power consumption can also save the cost of equipping heat cooling devices and increases system reliability.

Several approaches exist in order to save energy [2,3]. One of popular approach is to remove hardware over-provisioning. Typical cloud computing infrastructure is over-provisioned in

order to maintain service availability, as the infrastructure should be able to handle expected peak demands and the peak demand events does not occur frequently and a large number of computing resources are idle much of time. Saving in power consumption is achieved if we can shut down or hibernate idle or not much used physical machines while maintaining available for the required amount of services.

In this paper, we propose a technique that facilitates power saving based on virtualization technology [4]. Through virtualization technology, the number of active physical machines can be controlled dynamically using virtualization, and dynamic resizing of the number of active physical machines allows power saving in managing cloud infrastructure. Our approach is to use workload shaping technology which reorder tasks with various priorities and distributes them to appropriate virtual machines. Our approach controls virtual machines activation to minimize the amount of power usage, by moving or turning-off virtual machines. Our workload shaping technology intends for effective virtual machine configuration in terms of power saving.

This paper is organized as follows. The mathematical model power model is reviewed in Section 2. The mathematical power model is presented in order to provide a evaluation of the effects in the power consumption using our

---

Manuscript received : Sept. 29, 2012 / revised : Nov. 1, 2012

<sup>†</sup>Corresponding Author: woongsup@dongguk.edu

Tel: +82-2-2260-3833, Fax: +82-2-2260-3833,

Dept. of Computer & Information Communications Engineering  
Dongguk University, Seoul, Korea

shaping approach. Our approach, using workload shaping methodology to reduce power consumption, is presented in Section 3. Section 4 describes our simulated result based on CloudSim simulator [5], and our experimental result for real cloud system configuration. We compared our result to the cloud system configuration without any power consumption consideration. Finally, we gave some conclusions in Section 5.

## 2. POWER CONSUMPTION EVALUATION MODEL

Power consumption in a single physical computing machines in a cloud infrastructure is composed of two components: base power consumption independent of CPU clock frequency and power consumption dependent on CPU clock frequency. Equation (1) describes the overall power consumption in a cloud infrastructure.

$$P_i = \int_{t_0}^{t_1} x_i(t)(c_i + \beta_i f_i(t)) \quad (1)$$

where,  $P_i$  is the power consumption of physical machine  $i$  during the time spanning  $t_0 \leq t \leq t_1$ ,  $x_i(t)$  denotes machine on-off states at time  $t$  (when a machine  $i$  is in on state  $x_i(t) = 1$  and when a machine is off state  $x_i(t) = 0$ ),  $c_i$  is the constant power consumption of the machine  $i$ .  $c_i$  includes CPU's base power consumption and the power consumption of all the other components in the machine  $i$ . CPU also consumes power  $\beta_i f_i(t)$  which is varied with CPU operating frequency  $f_i$ . Hence the power consumption of all the physical machines in a cloud infrastructure can be expressed as follows:

$$P_{all} = \sum_{i=1}^N \int_{t_0}^{t_1} x_i(t)(c_i + \beta_i f_i(t)) \quad (2)$$

Here  $P_{all}$  represents the sum of all physical machines' power consumption and is expressed with the sum of  $P_i$  in (1).  $N$  is the number of physical machines in a cloud infrastructure.

The objective of power optimization problem is to minimize  $P_{all}$  without breaking conditions for the service availability. The service availability conditions are represented as follows

$$\sum_{i=1}^N x_i(t) \lambda_i(t) = \lambda_{cloud}(t) \quad (3)$$

$$x_i(t)(1 - x_i(t)) = 0, i = 1, 2, 3, \dots, N \quad (4)$$

$$g(\alpha_i f_i) > h(\lambda_i(t)), i = 1, 2, 3, \dots, N, \quad (5)$$

where  $\lambda_i(t)$  is service requests for the machine  $i$  at time  $t$ ,  $\lambda_{cloud}$  is service requests from all the cloud clients at time  $t$ ,  $g(\alpha_i f_i)$  is machine  $i$ 's processing capacity at CPU's frequency  $f_i$ , and  $h(\lambda_i)$  is the required amount of CPU usage to process

requests  $\lambda_i$  without delaying service delivery. Equation (3) describes all service requests at any time should be processed in any active machines. Equation (4) describes all physical machines have either on or off states. Final condition is that the processing amount of requests to machine  $i$  must be less than machine  $i$ 's processing capacity (Equation 5).

## 3. APPLYING WORKLOAD SHAPING TOWARD POWER CONSUMPTION SAVING

Our approach is to use workload shaping technique to save power consumption in a cloud infrastructure which is composed of multiple resources. The workload shaping is to shape workloads dynamically in different concurrently running physical machines in order to meet specific purpose, while the workload in every physical machine fits each machine's processing capacity.

To this end, we used two methodologies: task buffering and virtual machine scheduling. Task buffering is a technique to smooth out the variability associated with continuous task requests. The purpose of task buffering in our approach is each machine holds running tasks close to its capacity, which make easy virtual machine scheduling. Virtual machine scheduling is a methodology that organize virtual machines associated with physical machines. From our assumption in cloud environment, a physical machine can run one or more virtual machines and any virtual machine can move from machine to machine without interruption due to current migration technology.

Fig. 1 describes our approach using workload shaping technology. Our approach maintain several *task buffers* which are associated with one or more virtual machines. Tasks are classified into several types in advance and *task buffer* is assigned to hold predetermined type of task only. When new request arrives to the system, *Job Dispatcher* first identify task types and assigned the new request to the appropriate *task buffer*. *Load Controller* smooths out variability in requests arrival to virtual machines using *task buffer*. As we do not know the number of requests in advance, initially one task buffer is associated with one virtual machine and the association is expanded to multiple virtual machines based on the amount of task requests. Virtual machines run on physical machines in the cloud infrastructure. Based on processing capacity of physical machines and current required processing amount in virtual machine, a physical machine can have multiple active virtual machines. Initially, the smallest number of physical machines are active holding  $m$  virtual machines and the number of active physical machines can be expanded to complete all the task requests without violating system availability.

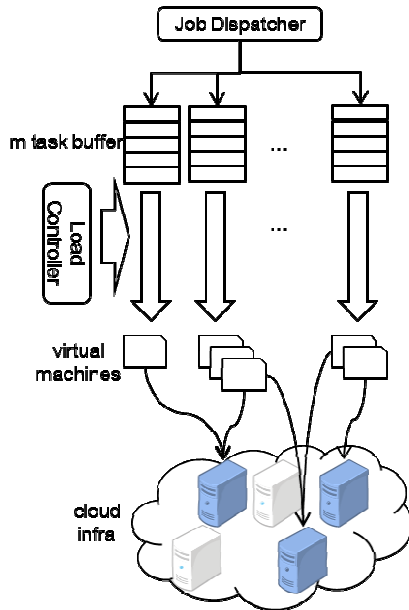


Fig. 1. system architecture for applying workload shaping to cloud infrastructure.

In our approach, tasks are identified based on expected execution time. Tasks with similar execution time are assigned to the same task buffer. That means, a virtual machine always runs tasks with similar expected execution time. We use execution history to expect task's execution time. *Load Controller* monitors virtual machines' status and decides whether to invoke new virtual machine or to withdraw the existing virtual machine. To avoid virtual machine over-provisioning, *Load Controller* utilize task buffer to hold immediate task execution. In our approach, we set the deadline of a task as  $\rho \times \{\text{expected execution time}\}$  and *Load Controller* can hold a job request in the task buffer at most  $(\rho - 1) \times \{\text{expected execution time}\}$ . The condition to invoke new virtual machine is  $\forall i, h_i(\lambda_i(t)) > \psi \times g_i(\alpha_i f_i)$ ,  $i=1,2,3,\dots,k$ , where  $k$  is the number of current active virtual machines for  $l$  type task,  $h_i(\lambda_i(t))$  is the amount of usage in virtual machine  $i$  at time  $t$ ,  $g_i(\alpha_i f_i)$  is the virtual machine capacity at the CPU clock frequency  $f_i$  for  $l$  type task, and  $\psi$  is the threshold to avoid service delay due to task saturation in the virtual machine. Virtual machines are withdrawn when there are no task running. Through migration, ideally virtual machines can move or merge from physical machine to machine to reduce the number of active physical machines. Due to implementation problem, we could not finish automatic virtual machine migration. So current our approach assumes virtual machine turn-on and turn-off. The condition to turn on new physical machine is similar to that of virtual machine. The condition to turn off an existing physical machine is when there is no virtual machine running the physical

machine.

#### 4. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we presents our evaluation result both in simulated environment and in the real cloud implementation.

For the experiment, we used a log file created from tasks executing indexing service for 10 GB data. To simulate up-down pattern in the number of job requests, we run several indexing services simultaneously. Fig 2. shows job request pattern we used for the experiment.

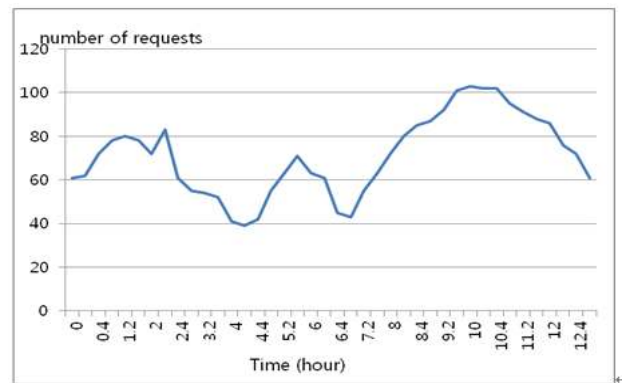


Fig. 2. The number of job requests used in the experiment

##### A. Power Consumption Comparison in Simulated Environment

We have measured power consumption in simulated environment using data in Fig. 2. For experiment, we used various parameters in terms of task buffer size and smoothing parameter  $\rho$ . We also set CPU clock frequency level into 2 (when the machine is idle and when the machine runs any task) and CPU power consumption doubled based on the level. The ratio between  $c_i$  and  $\beta_i f_i(t)$  is 2:1. From our experimental result shown in Table I, our approach is effective than non-workload shaping approach, and task buffer size has more impact on power consumption saving.

Table I. Relative Power Consumption in Simulated Environment  
(m: task buffer size,  $\psi = 0.8$ )

Condition	Power Consumption (%)
No workload shaping	100
m = 1, $\rho = 2$	98.2
m = 1, $\rho = 3$	97.7
m = 3, $\rho = 2$	94.5
m = 3, $\rho = 3$	93.8

$m = 5, \rho = 2$	92.2
$m = 5, \rho = 3$	91.2

#### B. Power Consumption Comparison in Cloud Implementation

We implement workload shaping technology in cloud infrastructure. Our implementation is based on fedora 6.0 and two dual core 2.5 GHz Pentium CPU. We also add additional server for load generator with job dispatcher. The power consumption is measured using direct measuring device every 10 minute. As can be seen from Table II, the power saving rate is around 5% which is meaningful saving effect. The power consumption in Table II is average value displayed at power measuring device.

Table II. Power Consumption Comparison in Real Implementation  
(m: task buffer size,  $\psi = 0.8$ )

Condition	Power Consumption (avg. W)	Power Consumption (%)
No workload shaping	411	100
$m = 3, \rho = 3$	387.3	94.2
$m = 5, \rho = 3$	382.3	93.0

## 5. CONCLUSION

We have worked for power saving optimization in cloud environment through workload shaping. We have designed a new methodology using workload shaping technology to optimize power consumption. Our workload shaping technology is used to smooth out job requests and hence to be helpful for optimal scheduling of virtual machines and physical machines. Finally, we demonstrates our approach is useful and effective in reducing power consumption in cloud infrastructure.

In the future, we need to use more meaningful and practical data used for cloud environment, in order to justify our work. In addition, we need to compare the existing workload shaping algorithms and refine our algorithm to provide more optimal solution. Finally, we should add more configurable option such as virtual machine migration and merge to save power consumption.

## Acknowledgment

This work was supported by Dongguk Research Fund of 2012.

## REFERENCES

- [1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing, v15," 2009. Available at <http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc>
- [2] San Murugesan, "Harnessing Green IT: Principles and Practices," IEEE IT Professional, January–February 2008, pp 24-33
- [3] W. H. Kemp, "The Renewable Energy Handbook: A Guide to Rural Energy Independence, Off-Grid and Sustainable Living," Aztext Press, 2006.
- [4] H. Meinhard, "Virtualization, clouds and IaaS at CERN," VTDC 12, pp 27-28, New York USA, 2012..
- [5] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, pp. 25-50, 2011..



**Woongsup Kim** was born in Seoul, Korea in 1970. He received the B.S. degree in Computer Engineering from Seoul National University, Korea in 1998, M.S. degree in Computer and Information Science from University of Pennsylvania, USA, in 2001, Ph.D. degree in Computer Science and Engineering from Michigan State University, USA, in 2006 respectively. Since 2007, he has been with Department

of Computer and Information Communications Engineering, Dongguk University, Seoul, Korea, where he is currently a Assistant Professor. His research interests are in the areas of Software Engineering, and Cloud Computing. Dr. Kim is a Member of IWIT of Korea.