International Journal of Energy, Information and Communications
Vol.5, Issue 4 (2014), pp.17-26
http://dx.doi.org/10.14257/ijeic.2014.5.4.02

Design of Cost Function for VM Allocation in Cloud Computing

Yeongho Choi and Yujin Lim*
University of Suwon
{ceewoo, yujin}@suwon.ac.kr

Abstract

Cloud service, as one of major technologies in modern IT business, has attracted a lot of attentions from academia and industry. Due to the dynamics of user demands, service providers need VM(Virtual Machine) provisioning mechanism to estimate the amount of resources demanded by cloud users in the next time interval and to prepare the resources elastically. In this paper, we describe issues of VM provisioning and introduce the state-of-the-art technologies for each issue. Besides, for the efficient VM provisioning, we propose a cost function to calculate the total expense of a service provider under workload fluctuation. To evaluate the effectiveness of our cost function, we show the performance evaluation with real workload data.

Keywords: *Cloud computing, Resource allocation, VM provisioning*

1. Introduction

Cloud computing gradually becomes an issue for modern IT business because of its flexibility, convenience, and low cost. There are three cloud computing models, which are IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service). IaaS provides computing infrastructure and physical or virtual resources like network bandwidth, storage, and CPU. PaaS provides computing platforms which typically include operating system, programming language execution environment, database, and web server. SaaS provides access to application software as on-demand software. All three types are delivered in four ways, which are publicly, privately, via a community, or in a hybrid cloud. Based on the models, service providers offer different types of services to cloud users with different demands. Cloud computing provides a pay-per-use payment. In other words, cloud users pay cost as much as using services. The major service providers, namely Amazon, Microsoft, and Google, offer many types of services and applications to cloud users through monitoring, managing, and provisioning resources. Amazon EC2 (Amazon Elastic Compute Cloud) and Microsoft Azure are examples of IaaS [1]. The IaaS services manage resources as a shared pool of configurable computing resources (VMs: Virtual Resources) to dynamically provision and release with minimal management effort, as shown in Figure 1.

The resources demanded by users are more various than these in traditional IT environments. In cloud environments, service providers are difficult to elastically prepare different types of resources and reply changes in demands of users [2]. Thus, VM provisioning that estimates and prepares the resource to meet dynamic demands of users is one of challenging issues in cloud computing paradigm.

VM provisioning is a strategy for managing resources by allocating resources on an "as needed" basis. It automatically adapts to workload changes related to applications

* Corresponding author

ISSN: 2093-9655 IJEIC

for facilitating the adaptive management of system and offering end-users guaranteed QoS(Quality of Services). Typically, the provisioning is achieved by two operations - static and dynamic resource provisioning [3, 4]. In static resource provisioning, VMs are created with specified size and then consolidated onto a set of physical servers. The VM capacity does not change. In dynamic resource provisioning, VM capacity is dynamically adjusted to match workload fluctuations. Static provisioning often applies to the initial stage of capacity planning. In both static and dynamic provisioning, the estimation of the resource amount is one of the most important steps. The objective of the estimation is to ensure that VM capacity is commensurate with the workload. While over-provisioning wastes costly resources, under-provisioning degrades application performance and may violate SLA (Service Level Agreement).

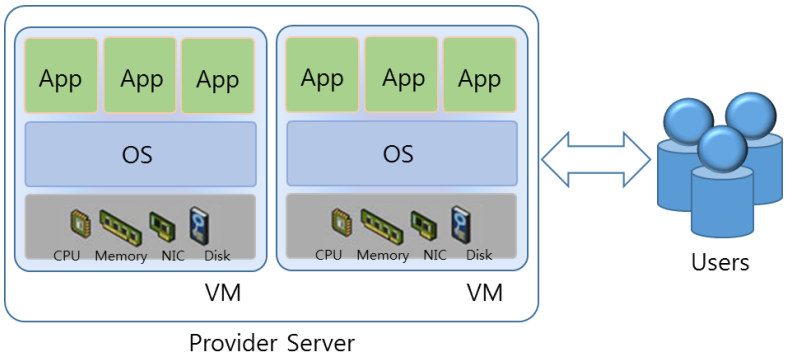

Figure 1. VMs as a shared pool of configurable resources

In this paper, we present issues and related studies of VM provisioning, and propose a new approach for the VM provisioning to minimize the total expense of a service provider.

2. VM Provisioning

In cloud computing, service providers need VM provisioning to maximize their revenues and to guarantee QoS to cloud users. To maximize the revenues, adjust the amount of VMs to match workload fluctuations. Over-provisioning wastes resources and under-provisioning may make the service providers pay penalty due to the SLA violations. Issues of VM provisioning are as follows [5].

A. VM provisioning delay: In practice, it takes a few minutes to provision a VM from an IaaS provider. This time delay is affordable for normal services but is unacceptable for tasks that need to scale out during computation. To enable the on-the-fly scaling, new VM needs to be ready in seconds upon requests. Thus, handling technologies of a sudden spike in the incoming demands of cloud users need to reduce the request fulfillment time [6, 7].

B. Admission control: Admission control approaches aim to prevent server overloading under high load situations [8]. Besides, it prevents over-admission on existing VMs due to the VM provisioning delay. Therefore, VM provisioning needs to be augmented with an admission control mechanism. One common characteristic of traditional approaches is that they make decisions only on acceptance or rejection of incoming user demand. However, it may be possible to defer the incoming demand until some new VMs are

provisioned or some existing VMs become less loaded. Thus, a new admission control mechanism needs to choose between using an existing VM and provisioning a new VM for accommodating new incoming demand.

C. Load prediction: Many traditional VM provisioning approaches use reactive provisioning. However, in the reactive provisioning, it starts a provisioning operation only after a significant increase in the load is detected. Due to the VM provisioning delay, the reactive approach may fail to handle the increased load. Alternatively, proactive approaches use prediction of future load to provision VMs preemptively [9]. Thus, the service providers need load prediction techniques to promptly and accurately estimate the amount of resources for the next time interval.

D. Reduced number of VMs: Creation and maintenance of VMs are resource-intensive operations. For saving the resources, it needs to consolidate under-utilized VMs from time to time in order to reduce under-utilization of VMs and total number of provisioned VMs. In this case, the main challenge is to augment VM provisioning with a server consolidation mechanism, which uses a reduced number of VMs along with a reduced number of VM migrations.

E. Sharing of VM resources: Instead of traditional provisioning at least one VM per application, shared hosting effectively supports provisioning a fraction of a VM per application. It results in a reduced number of total VMs. The sharing of VM resources among multiple concurrent applications improves VM utilization, which helps in reducing the total number of required VMs. Thus, cloud providers get more revenues from computing resources that would be idle or under-utilized, and accommodate sudden spikes in demands of the users.

F. Automatic adjustment: To ensure revenues of service providers and QoS for cloud users under diverse load conditions, it is necessary that the VM provisioning, admission control, and server consolidation approaches automatically adjust and adapt themselves according to the load conditions. For the prediction-based VM provisioning, we can use a weighting coefficient, which is automatically adjusted and tuned based on load conditions parameters.

3. Related Studies

We introduce the related studies of the admission control, the sharing of VM resources, and the load prediction. Admission control is a technique that constrains service requests to prevent overload of a server. Therefore, overload prevention depends on acceptance or rejection of new requests. The overloaded server may lead to deteriorate the performance, such as response time and throughput. There are representative studies for the admission control. First, ACVAS (Adaptive Admission Control for Virtualized Application Servers) is an admission control based on session-based admission control techniques [10, 11]. Instead of using the traditional on-off control, the session-based technique controls resources per session, which reduces the risk of over-admission. ACVAS uses the measured and the predicted resource utilizations to offer resources to the users. Second, QoS requirements are dynamic and are ruled by the SLA contracts which specify the unit price of a service invocation and the corresponding QoS level. QoS guarantee has a direct impact on providers' revenues. In [12], the probability is

estimated that the response time of a service invocation violates the SLA contract. Using the probability, the acceptance or rejection is determined to guarantee QoS.

In cloud computing, sharing of VMs resources improves resource utilization. As shown in Figure 2, *Provider 1* has more storage resources than the other resources like CPU and network bandwidth. *Provider 2* and *Provider 3* have more CPU and network resources than other resources, respectively. By sharing the remaining resources with each other, the resource utilization is improved. It enables that the providers accommodate more cloud users and a sudden spikes in demand. We present two studies for the sharing of VMs resources. First, for cost-efficiency, CRAMP (Cost Efficient Resource Allocation for Multiple Web Applications with Proactive Scaling) provides a finer deployment granularity than the smallest VM provided by the contemporary IaaS providers [13]. This is especially important when running a large number of web applications, most of which may have very few users at a given time, while a few of them may have many users. Thus, CRAMP shares VM resources by supporting shared hosting. Fewer VMs are used to run several web applications and unnecessary costs are avoided without compromising QoS. Second, the service providers make a pool to share the remaining resources with each other [14]. They cooperate to establish a resource pool to support internal users and to offer services to public cloud users. Through developing the stochastic linear programming game model and analyzing the stability of the coalition formation among cloud providers, the hierarchical cooperative game model is proposed for the sharing of VM resources.

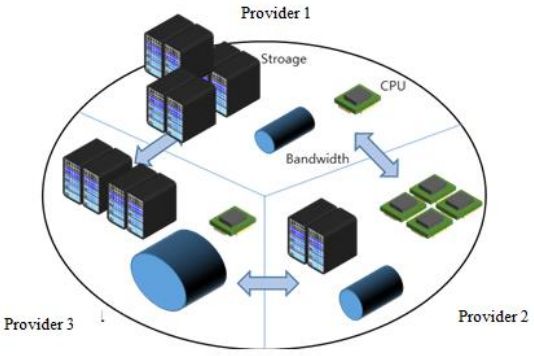

Figure 2. Sharing of VMs resources

The service providers predict the amount of resources to provide in the next time interval and prepare the resources for QoS guarantee of cloud users. If the providers prepare instantly VMs when the cloud users request service, it generates the VM provisioning delay for creating new VMs. The delay may lead to deteriorate the performance and violate the SLA. In [15], the cost function is presented to calculate the expense of a service provider. If the amount of resources prepared by a service provider is more than the actual demand of the users, the remaining resources are wasted. On the other hand, if the amount of resources prepared is less than the actual demand, the performance may be deteriorated and the SLA contract may be violated. In this case, the provider should pay the penalty to cloud users. Using the cost function, different load prediction techniques, such as Moving Average, Auto Regression, Artificial Neural Network, Support Vector Machine, and Gene Expression Programming, are analyzed.

In this paper, we propose a model to predict the amount of resources in the next time interval to maximize the revenue of a service provider through minimizing the expense of the provider by the SLA violation and the waste of the remaining resources.

4. Proposed model

This section presents a new approach for the VM provisioning to minimize the total expense of a service provider. Our goal is to predict the service rate in the next time interval in order to maximize the revenue while minimizing the total expense of a service provider. For the goal, in this paper, we propose a cost function to calculate the total expense of a provider by using estimation of resources which is demanded by users in the next interval. We consider two different kinds of costs: the cost of wasted resources for the over-provisioning and the penalty of SLA violations for the under-provisioning. In the over-provisioning that resources prepared by the service provider are more than actual demands of the users, the users would not be affected but the service provider will suffer the waste of unused resources. In this case, the provider should pay the cost of wasted resources. In the under-provisioning that the prepared resources of the service provider are less than actual demands of the users, the VM provisioning delay for additionally creating new VMs is generated. The delay of the VM provisioning may lead to deteriorate the performance and violate the SLA. In this case, the service provider should pay the penalty for SLA violations. As a result, the total expense can be represented as [16]:

$$C_{total} = \omega C_{expense}^{over} + (1-\omega) C_{expense}^{under} \quad (1)$$

Here ω is a smoothing factor ($0 \leq \omega \leq 1$) to tune the importance between two costs. For example, if the workload of the cloud is high, the system operator can decrease ω to focus more on the SLA penalty.

Our model has a server with M/G/1 queue. We handle transaction-based service rate and arrival rate for each time interval. We also do not assume any specific scheduling discipline. Either FCFS or processor sharing could be used, as both disciplines have been frequently considered reasonable abstractions for transactional service centers [17, 18].The main parameters to be used in the system model are shown in Table 1.

To calculate the total expense, we define the cost of wasted resources and the penalty for SLA violations. In the over-provisioning, the expense includes the cost of wasted resources (C_{waste}) and the cost for providing resources (C). The cost for providing resources includes the server aging, electricity, and labor cost. The expense of service provider by the over-provisioning is defined as follow:

$$C_{expense}^{over} = \min(x, \lambda) C + \max(0, x-\lambda) C_{waste}. \quad (2)$$

In the under-provisioning, the expense of service provider includes the penalty for SLA violations ($C_{penalty}$) and the cost for providing resources (C). In order to calculate the penalty for SLA violations, we use the probability of response time SLA violation due to the additional VM provisioning as $P(T_{res} \geq T_{res}^{SLA})$. Models for the service response time distribution are only for some types of queues. Moreover, some of the available models are quite complex. Markov's Inequality [19, 20] can provide an upper-bound on the probability that the service response time (T_{res}) exceeds the SLA threshold (T_{res}^{SLA}). The upper-bound depends on the average service response time $E[T_{res}]$. Although it might provide somewhat loose upper-bounds [21], we choose to use the simple and computational efficient Markov's Inequality as an approximation of $P(T_{res} \geq T_{res}^{SLA})$.

Table 1. Parameters of the model

Symbol	Description
C_{total}	Total expense of a service provider
$C_{expense}^{over}$	Expense of service provider for over-provisioning
$C_{expense}^{under}$	Expense of service provider for under-provisioning
C	Cost of providing resources
C_{waste}	Cost of wasted resources in over-provisioning
$C_{penalty}$	Penalty for SLA violations in under-provisioning
ω	Weighting factor
x	Service rate in the next time interval
λ	Predicted arrival rate in the next time interval
λ'	Average arrival rate
S	Average service time
Avg	Average service rate
U	Utilization of server
T_{res}	Service response time
T_{res}^{SLA}	Threshold of service response time to be guaranteed by SLA

In order to use Markov's Inequality, we first compute the average service response time in the next time interval as follows [17]:

$$E[T_{res}] = \frac{S}{1-\lambda' S}. \quad (3)$$

We assume that λ' is x to calculate the average service response time of our system model. Thus, (3) is rewritten as follows:

$$E[T_{res}] = \frac{S}{1-xS}. \quad (4)$$

Using $S = \frac{U}{x}$ and $U = \frac{\lambda}{Avg}$, we compute the average service time as follows [22]:

$$S = \frac{\lambda}{xAvg}. \quad (5)$$

Using (4) and (5), we obtain as follows:

$$E[T_{res}] = \frac{\lambda}{|Avg-\lambda|x}. \quad (6)$$

Then, we use (6) to compute the probability of the response time SLA violation by applying Markov's Inequality as follows:

$$P(T_{res} \geq T_{res}^{SLA}) \leq \frac{E[T_{res}]}{T_{res}^{SLA}}. \quad (7)$$

We can improve (7) as follows:

$$P(T_{res} \geq T_{res}^{SLA}) \fallingdotseq min\left(\frac{E[T_{res}]}{T_{res}^{SLA}}, 1\right). \quad (8)$$

As a result, the expense of the service provider for the under-provisioning is defined as follow:

$$C_{expense}^{under} = min(x, \lambda)C + max(0, \lambda - x)P(T_{res} \geq T_{res}^{SLA})C_{penalty}. \quad (9)$$

Using (8), we rewrite (9) as follows:

$$C_{expense}^{under} = min(x, \lambda)C + max(0, \lambda - x)\, min\left(\frac{E[T_{res}]}{T_{res}^{SLA}}, 1\right) C_{penalty}. \quad (10)$$

Using (6), (10) is rewritten as follows:

$$C_{expense}^{under} = min(x, \lambda)C + max(0, \lambda - x)\, min\left(\frac{\lambda}{|Avg-\lambda|xT_{res}^{SLA}}, 1\right) C_{penalty}. \quad (11)$$

As a result, the total cost is defined as follows:

$$C_{total} = \omega\{min(x, \lambda)C + max(0, x - \lambda)C_{waste}\} \quad (12)$$
$$+(1-\omega)\left\{min(x, \lambda)C + max(0, \lambda - x)\, min\left(\frac{\lambda}{|Avg-\lambda|xT_{res}^{SLA}}, 1\right) C_{penalty}\right\}.$$

Because the under-provisioning is not coincided with the over-provisioning, we rewrite (12) as follows:

$$C_{total} = \begin{cases} \omega\{\lambda C + (x-\lambda)C_{waste}\} + (1-\omega)\lambda C & \text{if } x > \lambda \\ \omega xC + (1-\omega)\left\{xC + (\lambda - x)\, min\left(\frac{\lambda}{|Avg-\lambda|xT_{res}^{SLA}}, 1\right) C_{penalty}\right\} & \text{if } x < \lambda \end{cases} \quad (13)$$

5. Experimental Result

To verify our cost function, this section shows the performance of our model and compare with the total expense by using traditional prediction techniques. In order to predict the amount of resources (λ) which is demanded by users in the next time interval, we use CLT(Central Limit Theorem), EWMA(Exponentially Weighted Moving Average) and AR(Auto Regression) as traditional prediction techniques. We perform simulation experiments with real workload data from the Intel Netbatch workload archive [23]. The data records the VM requests for one month (from October 2012 to November 2012). Each request record contains various features, such as *job ID, group ID, user, submit time, start time, finish time, exist status, wall time, max VM, memory*, and *core*, etc. We obtain the average arrival rate (λ'), average service rate (Avg), and actual service rate (x) using *submit time, start time, finish time*, and *exits status*. For other parameter, we set $\omega = \{0.3, 0.7\}$, $T_{res}^{SLA} = 0.7$, $C = 6$, $C_{waste} = 8$, and $C_{penalty} = 10$. When $\omega = 0.3$, it puts the importance of the under-provisioning. When $\omega = 0.7$, it puts the importance of the over-provisioning.

Figure 3 shows the prediction error of the traditional prediction techniques. The prediction error indicates the difference between the predicted arrival rate (λ) and the actual service rate (x) in order to show the performance of the prediction techniques. The results show that the techniques have similar performance in terms of the prediction error. Figure 4 and 5 show the total expense using (13) with the predictors. In Figure 4, the under-provisioning case has weight more than the over-provisioning. In Figure 5, the over-provisioning has more weight than the under-provisioning. By tuning ω, we consider the various system scenario in terms of the workload. Table 2 summarizes the evaluation results of various predictors. The results show that on average, the CLT technique achieves the best performance.

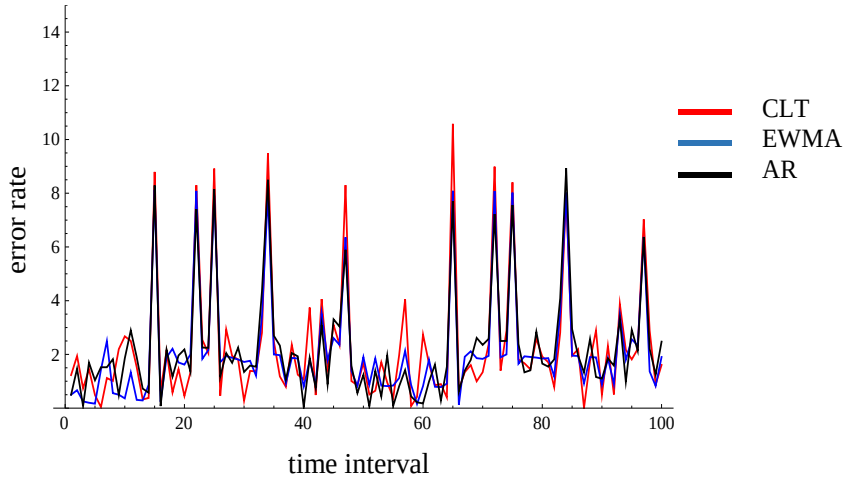

Figure 3. Prediction error of traditional prediction techniques

Table 2. Total expense using various predictors

predictor	average total expense (ω =0.3)	average total expense (ω =0.7)
Central Limit Theorem	20.28	10.81
Exponentially Weighted Moving Average	20.51	11.70
Auto Regression	21.49	13.41

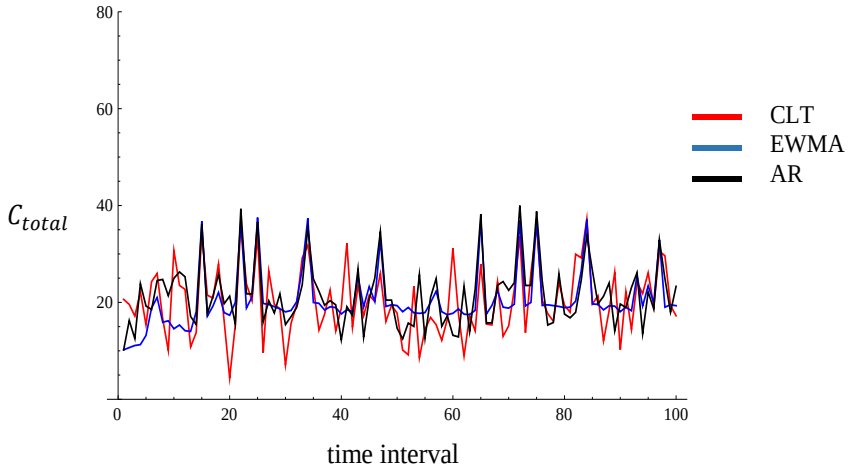

Figure 4. Total expense calculated by using prediction techniques ($\omega = 0.3$)

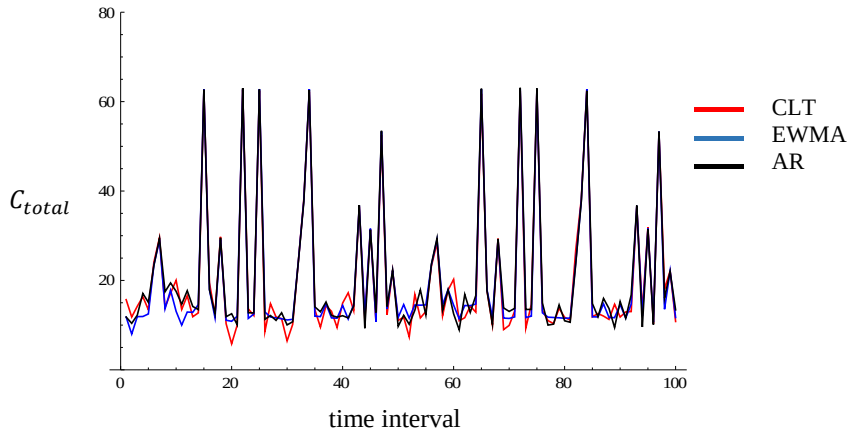

Figure 5. Total expense calculated by using prediction techniques ($\omega = 0.7$)

6. Conclusion

We addressed the state-of-the-art technologies of the resource management in cloud computing. In the computing, service providers allocate resources as needed by users while maximizing their revenues from SLA. To meet their goal, service providers need elastic and dynamic VM provisioning techniques. We introduced the related studies of the VM provisioning, such as admission control, VM resource sharing, and prediction of the resource amount. Then, we propose a cost function to calculate the total expense of a service provider by using estimation of resources which is demanded by users in the next interval. By using the cost function, the experimental evaluation compares the performance of the traditional predictors in terms of prediction error and total expense of the service provider.

For the future work, by using the proposed cost function, we present a new estimation model of the resource in the next time interval to minimize the total cost of a service provider. According to the model, the service provider prepares the resources in the next interval to maximize its revenue.

Acknowledgements

This work was supported by the GRRC program of Gyeonggi province. [(GRRCSUWON2014-B3), Development of Cloud Computing-based Intelligent Video Security Surveillance System with Active Tracking Technology]

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2013053435)

References

[1] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2/.
[2] M. N. Bennani and D. A. Menasce, "Resource Allocation for Autonomic Data Centers using Analytic Performance Models", Proceedings of the IEEE International Conference on Automatic Computing (ICAC), **(2005)** June 13-16; Seattle, USA.
[3] M. N. Bennani and D. A. Menasce, "Resource Allocation for Autonomic Data Centers using Analytic Performance Models", Proceedings of the IEEE International Conference on Automatic Computing, **(2005)** June, pp. 229-240; Seattle, US.

[4] D. Kusic and N. Kandasamy, "Risk-aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems", Proceedings of the IEEE International Conference on Automatic Computing, **(2006)** June, pp. 74-83; Dublin, Ireland.
[5] D. Kusic and N. Kandasamy, "Risk-Aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems", Proceedings of the IEEE International Conference on Automatic Computing (ICAC), **(2006)** June 13-16; Dublin, Ireland.
[6] P. Komal Singh and A. K. Sarje, "VM Provisioning Method to Improve the Profit and SLA Violation of Cloud Service Providers", Proceedings of the IEEE International Conference on Cloud Computing in Emerging Markets, **(2012)** October 11-12; Bangalore, India.
[7] A. Ashraf, "Cost-Efficient Virtual Machine Provisioning for Multi-tier Web Applications and Video Transcoding", Proceedings of the IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, **(2013)** May 13-16; Delft, Netherlands.
[8] A. Ashraf, B. Byholm, J. Lehtinen and I. Porres, "Feedback Control Algorithms to Deploy and Scale Multiple Web Applications per Virtual Machine", Proceedings of the 38th Euromicro Conference on Software Engineering and Advanced Applications, **(2012)** September 5-8; Izmir, Turkey.
[9] J. Almeida, D. Ardagna and C. Francalanci, "Resource Management in the Autonomic Service- Oriented Architecture", Proceedings of the IEEE International Conference on Autonomic Computing, **(2006)** June 13-16; Dublin, Ireland.
[10] S. Muppala and X. Zhou, "Coordinated session-based admission control with statistical learning for multi-tier internet applications", Journal of Network and computer Applications, vol. 1, **(2011)**, pp. 20-29.
[11] A. Ashraf, B. Byholm and I. Porres, "A Session-Based Adaptive Admission Control Approach for Virtualized Application Servers", Proceedings of the IEEE International Conference on Utility and Cloud Computing, **(2012)** November 5-8; Chicago, USA.
[12] L. Cherkasova and P. Phaal, "Session-based admission control: mechanism for peak load management of commercial web sites", IEEE Transactions on Computers, vol. 6, **(2002)**, pp. 669-685.
[13] A. Ashraf, B. Byholm and I. Porres, "CRAMP: Cost-efficient Resource Allocation for Multiple web applications with Proactive scaling", Proceedings of the IEEE International Conference on Cloud Computing Technology and Science, **(2012)** December 3-6; Taipei, Taiwan.
[14] D. Niyato, A. V. Vasilakos and Z. Kun, "Resource and Revenue Sharing with Coalition Formation of Cloud Providers: Game Theoretic Approach", Proceedings of the IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, **(2011)** May 23-26; Newport beach, USA.
[15] Y. Jiang, C. Perng, T. Li and R. N. Chang, "Cloud analytics for capacity planning and instant VM provisioning", IEEE Transactions on Network and Service Management, vol. 3, **(2013)**, pp. 312- 325.
[16] Y. Choi and Y. Lim, "A Study on Dynamic VM Provisioning Approach in Cloud Computing", Proceedings of the International Conference Smart Technologies for Energy, Information and Communication, **(2014)** August 5-6; Narashino, Japan.
[17] A. M. Daniel, W. D. Lawrence and A. F. A. Virgilio, "Performance by Design", Prentice Hall, **(2003)**.
[18] D. Villela, P. Pradhan and D. Rubenstein, "Provisioning Servers in the Application Tier for E-commerce System", Proceedings of the 12th IEEE International Workshop on Quality of Service, **(2004)** June 07-09; Hague, Netherlands.
[19] K. Leonard, "Queueing System", Wiley Interscience, **(1975)**.
[20] P. Athanasions and P. S. Unnikrishna, (Eds.), "Probability, Random Variables, and Stochastic Processes", Prentice Hall, **(2002).**
[21] B. Abrahao, V. Almeida, J. Almeida, A. Zhang, D. Beyer and F. Safai, (Eds.), "Self-adaptive SLA-driven capacity management for internet services", Proceedings of the 10th IEEE/IFIP on Network Operations and Management, **(2006)** April 03-07; New York, U.S.
[22] W. Greg, CSC407 Software Architecture Winter, **(2007).**
[23] D. G. Feitelson, "Parallel workloads archive: Logs", **(2014)**, http://www.cs.huji.ac.il/labs/parallel/workload/l_intel_netbatch/index.html.

