NASLA: Novel Auto Scaling Approach based on Learning Automata for Web Application in Cloud Computing Environment

Monireh Fallah
Department of Computer Engineering Islamic Azad University of Mahallat, Iran

Mostafa Ghobaei Arani
Department of Computer Engineering Islamic Azad University of Parand Tehran, Iran

Mehrdad Maeen
Department of Computer Engineering Islamic Azad University of Yadegar, Iran

ABSTRACT

Considering the growing interest in using cloud services, the accessibility and the effective management of the required resources, irrespective of the time and place, seems to be of great importance both to the service providers and users. One of the best ways for increasing utilization and improving the performance of the cloud systems is the auto-scaling of the applications; this is because of the fact that, due to the scalability of cloud computing, on the one hand, cloud providers believe that sufficient resources have to be prepared for the users, and on the other, the users also have a tendency towards the “pay as you go” system of payment for the resources. This paper seeks to offer an approach, based on the learning automata, for the scalability of the web applications, which combines virtual machine clusters and the learning automata in order to provide the best possible way for the scaling up and scaling down of the virtual machines. The results of this study indicate that the proposed approach has decreased the number of SLA violations (in percentage), while it has a smaller load of scalability compared to the other approaches in this regard.

Keywords

Cloud computing, Auto scaling, Learning automata, SLA violation

1. INTRODUCTION

Cloud computing technology is currently one of the popular and developing technologies and a successful example of distributed computing. Cloud computing is a model for an easy provision of network access, based on demand, for a shared storage of configurable computing resources (i.e. networks, servers, applications, services, etc.), which is capable of being provided and released very quickly with minimal management efforts, and minimal interaction with the service provider [1,2].

Cloud computing technology is an attempt to propose a new mechanism for the provision of the necessary infrastructures for the users and for the creation of the illusion of access to unlimited resources in the minds of the users. Those who work in the field of cloud computing technology have considered various advantages for it including flexibility, reliability, scalability, security, a decrease in costs, an unlimited capacity of the resources, etc. However, among all these capabilities, scalability seems to enjoy more popularity compared to the others and, one can find very few documents about cloud computing in which the issue of scalability has not been discussed. Since all applications, and in particular web applications, do not follow regular workload patterns, the scaling operations (i.e. scale up or scale down) must be carried out in real time and with minimal human intervention so that the recourses would be provided for the applications as soon as possible. Such a scaling of the resource, which is done automatically and with minimal human intervention, is called Auto-scaling [3].

This paper seeks to propose a novel approach for the auto-scaling of the resources of the web applications. The proposed approach is derived from the threshold scalable algorithm and is based on the learning automata. The learning automata is defined using the online mode of the virtual machines and the threshold, and the input workload selects the best way of scaling (i.e. scale up, scale down). The proposed approach decreases the overhead of the scaling process, and increases the efficiency of the available virtual machines as much as possible; moreover, with regard to the rate of SLA violation, this algorithm has a lower percentage of SLA violation compared to the basic threshold algorithm. The rest of this paper is organized into the following sections: the second section has been dedicated to the review of the related works. The scaling framework, the learning automata, and the proposed approach are presented in the third section of the paper. In the fourth section, the evaluation and the results of the simulation of the proposed approach will be discussed and in the fifth section the conclusion and suggestions for further research will be presented.

2. RELATED WORKS

Various studies have been conducted so far with regard to the scaling of the web applications. One of the traditional approaches in this regard is the one which is based on the threshold, thus, most of the studies in this regard have focused on the method of identifying a threshold for different workloads and their purpose has been to increase scalability in different patterns of workload. In this section, the studies related to the approach that is based on the threshold are reviewed.

Dutreilh et al. [4] have put forth a theory using the horizontal auto-scaling technique based on the threshold, in which they emphasized that, in order to prevent the fluctuations in the system, workload the resources (such as the number of virtual machines or the number of specified CUPs) must be carefully planned. They used a mixed workload in their experiment which comprised a mixture of five sinusoidal oscillations and they used response time as their metric.

Hasan et al. [5] studied a set of four thresholds in two time intervals. Further, they employed a multiple domain, such as the input workload of the CPU, the response time, and the workload resulting from communication between networks, as their metric. They concluded that the addition of a VM happens when both the level of CPU workload and the response time are higher than the specified thresholds.

Han et al. [6] used response time as their metric for specifying the threshold and they calculated the amount of their input workload, which was a combined one, through navigation and order-taking based on the clients' behavior, and they applied it on a special test bed that they called IC CLOUD. The type of scaling that they used was a combination of the vertical and horizontal scaling.

Chieu et al. [7], however, proposed a novel architecture for the dynamic scaling of web applications based on the threshold. This theory was proposed based on a series of active sessions; however, this was an extension of the RightScale [10] method in which, if the number of active sessions in all instances was higher than the threshold, one instance would be prepared, and if there were instances with the number of active sessions less than the hypothetical lower threshold and with at least one instance with no active sessions, then the idle instance would be shut down.

Kupferman et al. [8] used a horizontal scaling method in their study, in which they used the CPU workload as their metric and derived their input workload through the weekly, short-term transient and random traffic patterns. Their method had the same problems as the RightScale [10] rules-based voting system and their algorithm was heavily dependent on the threshold level defined by the user. In order to save financial resources and solve the problem of the voting system, they used the interesting idea of smart skill. While explaining their method, they stated that there is no reason to stop the function of a virtual machine before its clock time expires, even if there is little workload.

Simmons et al. [9], however, utilized the method of threshold laws with real workload in their study. They used the results obtained from the 1998 World Cup applications as their workload and they applied it on the Amazon EC2 provider test bed, the RightScale [10] and a simple web application. Further, in order to solve the problem of the voting system and the threshold, they created a tree structure. This structure is a means with which a set of established policies are evaluated.

Vaquero et al [11], proposed the dynamic, scalable cloud applications. Their work was much more advanced and rigorous than the other works done on the scalability of the applications in the domain of cloud computing. They proposed a new architecture for the dynamic scaling of the web applications based on the threshold and in the domain of virtual cloud computing. They presented their approach using a front-end load-balancer for routing and balancing the user requests on web applications which were stationed on web servers; these web servers were considered to be examples of virtual machines.

Hung et al. [12], proposed the auto-scaling for the cloud computing system. They presented an auto-scaling algorithm for the necessary preparations and the automatic balancing of the resources of the virtual machine, which were based on the active sessions of the application. Moreover, the energy costs have also been included in this proposed algorithm and the evaluation of the energy consumption has been presented based on the proposed model.

3. THE PROPOSED APPROACHE

One of the challenges that face the scaling approach based on the threshold is the proper time for carrying out the operation of scale up/scale down on the resources in the storage. Thus, in order to specify the proper time to carry out the operation of scale up/scale down, the learning automata was used in this study. In this section, first, the required scalability framework for the application of the proposed approach is explained, and then the learning automata and the proposed algorithm are discussed.

3.1 Scalability Framework

In order to put the application that is proposed in this study into practice, it is essential to have a scalability framework. The framework that is adopted in this study includes three major components, i.e. auto provision system, virtual cluster monitoring and dynamic scalability.

Auto provision system: In this component there is a broker which balances the workload of the web application. The broker works as an interface which receives requests from the users and, while preserving the workload balance, sends them to the cluster servers. Since this broker can be configured dynamically, it allows the cloud to add the web server to the virtual cluster automatically and dynamically. In this component, the broker manages the creation and distribution of the virtual machines in the clusters using NASLA.

Virtual cluster monitoring: The job of this component is to calculate the percentage of the occupied capacity of the virtual machines for each of the virtual clusters and send it to the broker. The auto provision system can, then, horizontally decrease or increase the number of the virtual machines based on the workload of the cluster using the broker command. If an application in the virtual cluster uses up most of its resources, the auto-scaling creates a new virtual machine that runes the same application. The virtual cluster monitoring system has the capability to detect the number of virtual machines that have reached the cluster threshold. The auto-scaling algorithm has been applied to the auto provision system and, in order to control and start the scaling operation (i.e. scale up/scale down) in the auto provision system and on the number of virtual machine instances, the monitoring system begins to work based on the scaling index statistics.

Dynamic scalability: in order to provide service level and quality of service agreements in the web server applications, network bandwidth and the number of sessions are of great significance. Initially, the algorithm specifies the number of online virtual machines, the bandwidth, and the number of upper and lower threshold active sessions respectively. If the network bandwidth and the number of active sessions are more than the upper threshold, a virtual machine is prepared and begins to work and then the broker is notified. However, if the number of virtual machines and the bandwidth are less than the lower threshold and, if there is at least one virtual machine which has no traffic or active sessions, then the idle virtual machine will be eliminated from the broker list and it will be terminated.

The computing resources such as the CPU and memory usage are important indexes for the evaluation of the amount of workload in the virtual clusters. If all of the virtual machines consume resources at a rate higher than the upper threshold, a new virtual machine is created, presented and starts to work. If the amount of resources of some of the virtual machines is less than the lower threshold and there is at least one virtual machine that has nothing to do, the idle virtual machine will be removed from the cluster.

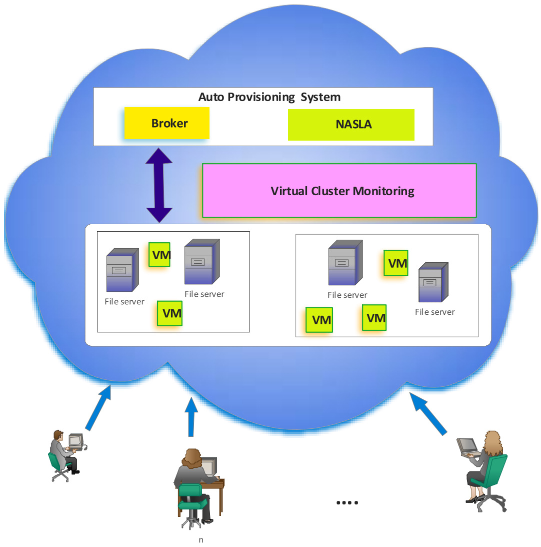

Figure 1: The proposed scalability framework

In the framework presented here (see figure 1), the clusters are host to several servers and they are connected to a broker as well. The broker is aware of the number of virtual machines assigned to each cluster as well as the thresholds and one can say that it corresponds with the virtual machines in the clusters. The broker manages the conditions of each cluster using our proposed algorithm, hence, it requests the creation of a new virtual machine if needed, and in case the clusters do not need some of the virtual machines it prevents the idle virtual machines from working and removes them. In section 3.3 the proposed algorithm (i.e. the NASLA) is described.

3.2 Learning Automata

Learning automata can be defined as an abstract object with a limited number of functions. The function of this object includes the selection of a function from among a set of functions at a time, and then evaluating it in a random environment following which a response is sent to the learning automata. Using this response, the learning automata choose their action for the next phase and this is how the learning automata gradually identify the optimal action. The manner in which the learning automata use the response of the environment for the selection of the next action is specified by the learning algorithm. What is meant by the environment here is all the external influences and conditions that affect the automata. The automata and the environment form a cycle in which the output of the automata α is the input of the environment and the output of the environment α is the input of the automata. The function of the automata can be described as a chain of repetitive cycles in which the automata and the environment interact with one another. The relationship between the learning automata and the environment (see Figure 2).

In this type of automata if the action α_i in the nth stage is selected and receives a reward from the environment, the probability of p_i (n) related to the action will increase and the probability of other actions will decrease. In case of a penalty, however, the probability of p_i (n) related to the action will decrease and the probability of other actions will increase. In either case, the changes will be made in such a way that the sum of all p_i (n)s would always stay constant and equal to one. Therefore, if in repeating an n number of actions α_i is selected, then for the repetition of n+1 we will have the following [13,14,15]:

- For optimal response(β=0):

$$P_i(n+1) = p_i(n) + a\left[1 - p_i(n)\right] \qquad (1)$$

$$p_j(n+1) = (1-a)\,p_j(n) \quad \forall\, j, j \neq i \qquad (2)$$

- For respond undesirable(β=1):

$$p_j(n+1) = (1-a)\,p_j(n) \quad \forall\, j, j \neq i \qquad (3)$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)\,p_j(n) \forall j,\, j \neq i \qquad (4)$$

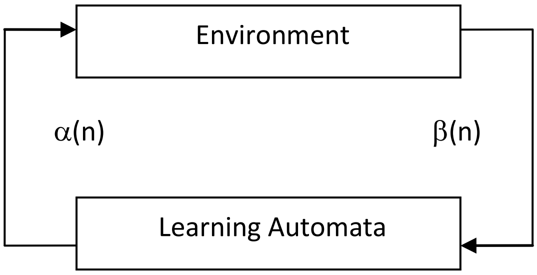

Figure 2: The Relationship between a learning automata and it's enviroment

3.3 The Proposed Algorithm (NASLA)

In this section, A Novel Auto Scaling Approach Based on Learning Automata (NASLA) for Web Application in Cloud Computing Environment will be presented and then the proposed approach will be evaluated based on the two criteria of scaling overhead (the number of deletions and additions done by the virtual machines) and the SLA violation percentage. We have assumed a monitoring component in the proposed algorithm in order to calculate the occupied space of the virtual machine which is represented by UC. The probability of scale up and scale down in the beginning is 0.5 and, in order to compare the occupied space in the VMs the lower threshold is taken to be 0.2 and the upper threshold as 0.8.

The way the whole system works is that the algorithm runs on n value of input workload which is made up of n number of system clusters. Each time the system runs, the occupied space of the virtual machine will be compared to the lower threshold and if it is more than the lower threshold reward will be given and, for each reward, the probability of p*i* (i.e. scale up) will increase; however, if it is less than the lower threshold, penalty will be given and p*i* will decrease and the probability of p_j (i.e. scale down) will increase. The values of the probabilities of p*i* and p_j will be updated and changed each time the system runs anew. In the end and after n times repeating and updating the last status of the UC, one of the actions of “scale up” or “scale down” with the highest probability will be selected by the virtual machine. If the selected action is scale-up, this algorithm compares the occupied space of the running virtual machine with the upper threshold and if it is more than the threshold, it will send the command for the creation of a new virtual machine; otherwise, if the selected action is scale-down, it will send the command for the removal of the running virtual machine. This part of the algorithm is the reason why there is a proper harmony between the selection of the type of action and the current status of the virtual machine. The general process of the proposed algorithm is presented in figure 3.

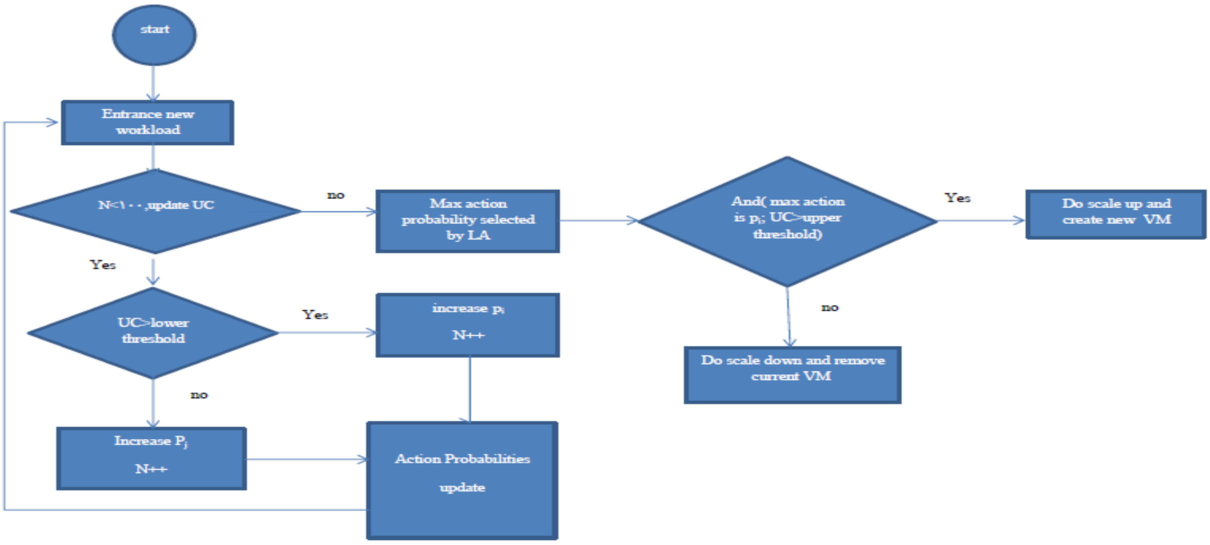
start
Entrance new workload
N<\ · · ,update UC
no
Max action probability selected by LA
And(max action is p_i; UC>upper threshold)
Yes
Do scale up and create new VM
no
Do scale down and remove current VM
Yes
UC>lower threshold
Yes
increase p_i
N++
no
Increase P_j
N++
Action Probabilities
update

Figure 3: The proposed algorithm based on learning automata

4. PERFORMANCE EVALUATION

In order to simulate the proposed approach, the Cloudsim [16] simulator has been utilized with a data center of 20 hosts, 4 clusters and a broker. Each cluster has a workload of its own which enters the system in 5 minute intervals. This workload has been created by a normal load distribution. What is meant here by workload is the amount of the CPU MIPS requested by the cluster. In the beginning, each cluster starts to work with one VM. The specifications of the VMs in the clusters, which are based on the Amazon EC2 [17] sample virtual machines, are presented in Table 1.

Table 2. Clusters in use

Cluster Type	VM Type	VM CPU (MIPS)
Cluster 1	High-CPU Medium	2500
Cluster 2	Extra Large	2000
Cluster 3	Small	1000
Cluster 4	Micro	500

A broker was used as an interface for receiving the requests and sending them to the virtual machines. A broker is, in effect, some sort of an agent which balances the workload in the framework of scaling. In order to evaluate the proposed approach (NASLA), it was compared with the basic auto-scaling approach (AS) [12] based on the two criteria of overhead scaling, and SLA violation.

Overhead scaling: one of the factors that have an immense effect on dynamic scaling is the number of the additions and deletions of the virtual machines. On the one hand, this criterion has a role in the acceleration of responding to the requests in the computing environment, and on the other, the number of times that these processes have been run has a role in the calculation of the costs of the providers; thus, the less the amount of this criterion, the less the costs, and the faster the responding will be which will, ultimately, present us with an optimized process with minimal costs.

SLA violation: SLA violation occurs when a provider is unable to provide the pre-defined criteria (i.e. the Service Level Objectives (SLO)) in the SLA for the users. The number of missing deadlines, failure to guarantee the agreed MIPS, failure to guarantee the agreed bandwidth, the number of rejected requests due to the shortage of resources in the peak times, etc., are but a few instances of SLA violation.

The results of the experiments have been yielded from an average of 20 simulations. For each cluster, the number of additions and deletions of the virtual machines are specified both based on the basic algorithm and the NASLA separately (see figure 4). The results indicate that in all criteria, NASLA has yielded better results compared to the AS.

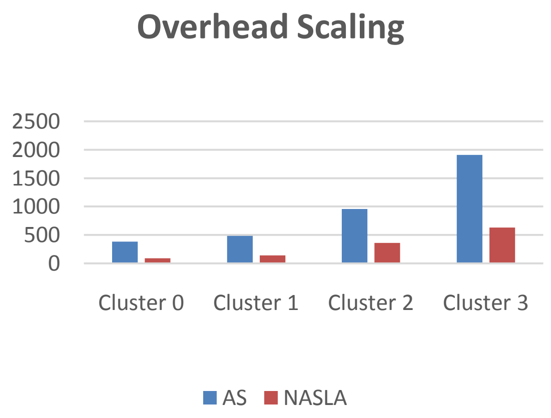

Figure 4: The overhead scaling comparison between AS and NASLA

The rate of SLA violation in each cluster calculated by the application of both algorithms (see Figure 5). The results indicate that by minimizing the scaling operation though optimizing it, the SLA violation percentage will also be optimized, as SLA drops to a minimum only when qualitative characteristics such as the accessibility of the resources, high throughput, and short response time are present. These factors have been provided by the application of this algorithm as far as it was possible.

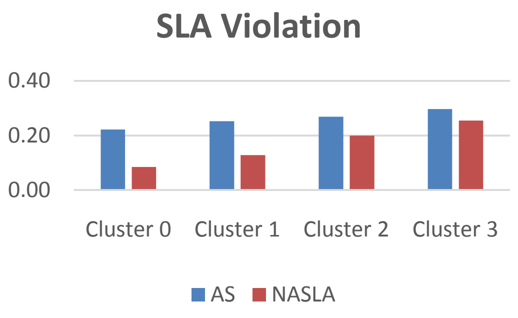

Figure 5: The SLA comparison between AS and NASLA

The comparison made between the total number of additions and deletions of virtual machines in both algorithms, based on an increase in clusters (see Figure 6); it indicates that as the clusters increase in number, around 4000 additions and deletions have occurred in actions with specific amount of workload; however, when NASLA was used, this number decreased to less than half of the number above.

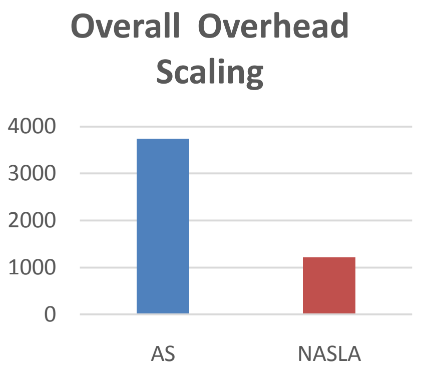

Figure 6: The comparison of overall overhead between AS and NASLA

A comparison between SLA violation percentages of the whole system in both algorithms, which indicates, yet again, that in the total number of clusters and based on the proposed method, the rate of this violation had been less than half of the basic method (See Figure 7). Therefore, the NASLA has yielded better results compared to the SA in all of the criteria.

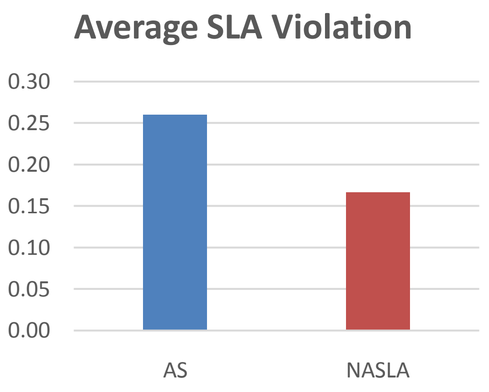

Figure 7: The average SLA comparison between AS and NASLA

5. CONCLUTION AND FUTURE WORKS

Considering the growing interest in cloud services, the accessibility and the effective management of the required resources, regardless of the time and place, are of great important both to the providers and to the users. The auto-scaling of the applications is an appropriate approach to increase efficiency and improve the performance of the cloud systems. This paper is an attempt to propose a new method for improving the quality of the services provided by clouds and for dealing with the SLA violation factor more effectively. The method that is proposed here is an auto-scaling method based on threshold that seeks to provide the best environmental responses for every mode of the virtual machines in cloud clusters, using the learning automata. Such a response would be the best response both in terms of the time and costs and it is a suitable method for minimizing SLA violation. As a new research, the variety of thresholds could be examined for choosing the best threshold of this method. The timing of the workload entry for service provision, the method of producing the input workload, and the provision of the security level of the virtual machines assigned to each request are among the factors that have not been addressed in detail in this paper and so, they are open ended topics that need to be addressed in future research studies.

6. REFERENCES

[1] Foster, Ian, Yong Zhao, Ioan Raicu, and Shiyong Lu. "Cloud computing and grid computing 360-degree compared." In Grid Computing Environments Workshop, 2008. GCE'08, pp. 1-10. Ieee, 2008.

[2] Behnaz Seyed Taheri, Mostafa Ghobaei Arani and Mehrdad Maeen. " ACCFLA: Access Control in Cloud Federation using Learning Automata. "International Journal of Computer Applications 107(6):30-40, December 2014.

[3] Lorido-Botran, Tania, Jose Miguel-Alonso, and Jose A. Lozano. "A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments." Journal of Grid Computing (2014): 1-34.

[4] Dutreilh, Xavier, Nicolas Rivierre, Aurélien Moreau, Jacques Malenfant, and Isis Truck. "From data center resource allocation to control theory and back." In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pp. 410-417. IEEE, 2010.

[5] Hasan, Masum Z., Edgar Magana, Alexander Clemm, Lew Tucker, and Sree Lakshmi D. Gudreddi. "Integrated and autonomic cloud resource scaling." In Network Operations and Management Symposium (NOMS), 2012 IEEE, pp. 1327-1334. IEEE, 2012.

[6] Han, Rui, Li Guo, Moustafa M. Ghanem, and Yike Guo. "Lightweight resource scaling for cloud applications." In Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on, pp. 644-651. IEEE, 2012.

[7] Chieu, Trieu C., Ajay Mohindra, Alexei A. Karve, and Alla Segal. "Dynamic scaling of web applications in a virtualized cloud computing environment." In e-Business Engineering, 2009. ICEBE'09. IEEE International Conference on, pp. 281-286. IEEE, 2009.

[8] Kupferman, Jonathan, Jeff Silverman, Patricio Jara, and Jeff Browne. "Scaling into the cloud." CS270-Advanced Operating Systems (2009).

[9] Simmons, Bradley, Hamoun Ghanbari, Marin Litoiu, and Gabriel Iszlai. "Managing a SaaS application in the cloud using PaaS policy sets and a strategy-tree." In Proceedings of the 7th International Conference on Network and Services Management, pp. 343-347. International Federation for Information Processing, 2011.

[10] RightScale. Set up Autoscaling using Voting Tags. http://support.rightscale.com/03-Tutorials/02-AWS/02-Website_Edition/Set_up_Autoscaling_using_Voting_Tags, 2012. [Online; accessed 13-September-2012].

[11] Vaquero, Luis M., Luis Rodero-Merino, and Rajkumar Buyya. "Dynamically scaling applications in the cloud." ACM SIGCOMM Computer Communication Review 41, no. 1 (2011): 45-52.

[12] Hung, Che-Lun, Yu-Chen Hu, and Kuan-Ching Li. "Auto-Scaling Model for Cloud Computing System." International Journal of Hybrid Information Technology 5, no. 2 (2012).

[13] Thathachar, M., and P. Shanti Sastry. "Varieties of learning automata: an overview." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 32, no. 6 (2002): 711-722.

[14] Najim, Kaddour, and Alexander S. Poznyak. Learning automata: theory and applications. Pergamon Press, Inc., 1994.

[15] Anari, Babak, Mohammad Reza Ahmadi, Mostafa Ghobaei Arani and Zohreh Anari. "Optimizing Risk Management Using Learning Automata." International Journal of Computer Science Issues (IJCSI) 10, no. 3 (2013).

[16] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. 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, no. 1, pp. 23–50, 2011.

[17] Amazon EC2 instance types, http:// aws . amazon.com/ EC2 / instance-types