A Learning Automata based Dynamic Resource Provisioning in Cloud Computing Environments

Hamid Reza Qavami
Department of Computer Engineering and Information Technology, University of Mohaghegh Ardabili
Ardabil, Iran
Cloud Research Center, Amirkabir University of Technology
Tehran, Iran
qavami@gmail.com

Shahram Jamali
Department of Computer Engineering and Information Technology
University of Mohaghegh Ardabili
Ardabil, Iran
jamali@iust.ac.ir

Mohammad Kazem Akbari
Department of Computer Engineering and Information Technology
Amirkabir University of Technology
Tehran, Iran
akbarif@aut.ac.ir

Bahman Javadi
School of Computing, Engineering and Mathematics
Western Sydney University
Sydney, Australia
b.javadi@westernsydney.edu.au

***Abstract*— Cloud computing provides more reliable and flexible access to IT resources, on-demand and self-service service request are some key advantages of it. Managing up-layer cloud services efficiently, while promising those advantages and SLA, motivates the challenge of provisioning and allocating resource on-demand in infrastructure layer, in response to dynamic workloads. Studies mostly have been focused on managing these demands in the physical layer and few in the application layer. This paper focuses on resource allocation method in application level that allocates an appropriate number of virtual machines to an application which requires a dynamic amount of resources. A Learning Automata based approach has been chosen to implement the method. Experimental results demonstrate that the proposed technique offers more cost effective resource provisioning approach while provisions enough resource for applications.**

Keywords- Cloud Computing, Dynamic Environment, Adaptive Resource Provisioning, Approximation, Learning Automata.

I. INTRODUCTION

In this era and because of the need for computations whenever and wherever on the one hand and also the need of individuals and organizations for cost effective heavy duty computation powers, on the other hand, the desire for computation as a utility paradigm have increased more than ever. Cloud computing is a new service offering model that is counted as the latest answer to this desire which offers IT resources as services. Computer resources are offered to users as some kind of infinite resource pool (e.g. processing capacity, Memory, Storage etc.) in cloud computing; That is one of its intrinsic features which severs it from traditional hosting and computing services.

Cloud user can be an individual or an organization that takes services from the Cloud Service Provider (CSP). CSPs provide cloud services via powerful hardware in warehouse scale centers (aka Data Centers). There are numerous data centers in the world and each of them consumes the energy as many as 25,000 households [1]. This clearly shows the necessity of an optimizing resource provisioning policy. In addition, an efficient resource provisioning is able to utilize the resources for reducing user payments.

Generally, the term Resource Provisioning in Cloud Computing is used for the taking in, deploying and managing an application on Cloud infrastructure. One of the main ideas in resource provisioning is to provide resources to applications in a way that reduces power and cost by optimizing and utilizing the available resource. Hence some power management techniques are considered in this field in some investigations. As a whole there is two generic way of resource provisioning:

One is Static Resource Provisioning which usually provides the peak time needed resource all the time for the application. In this kind of provisioning mostly the resources are wasted because the workload is not peaked in reality. The other is Dynamic Resource Provisioning which its basic fundamental idea is to provide the resources based on the application needs. The latter enables cloud providers to use pay-as-you-go billing system which seems fairer in the end users' point of view. The present study uses a learning based application scaling which applies a Learning Automata method for provisioning required resource dynamically, considering application workload changes.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 presents the proposed methodology from the background to the approach. Sections 4 and 5 discuss experimental design and experimental results respectively. Section 6 concludes the paper.

II. RELATED WORKS

One of the head most investigations about power management was carried out by Pinheiro et al. [1] the idea was about addressing power conservation for clusters of workstations or PCs. Elnozahy et al. in [2] combined Dynamic Voltage Frequency Scaling with dynamically turning on/off method called VOVO (vary-on/vary-off) to reduce power consumption. Kusic et al. [3] used Limited Lookahead Control (LLC). The goal was to maximize the resource provider's profit by minimizing both power consumption and SLA violation. Kalman filter was used to predict the number of next coming requests to predict the future state of the system and perform necessary reallocations. Verma et al. [4] solved the problem of power-aware dynamic placement of applications using Bin Packing problem. Van et al. [5] developed an optimization method and by modeling both provisioning and allocating problem they used Constraint Satisfaction Problem (CSP). Lin et al. [6] purposed a new Round Robin algorithm called Dynamic Round Robin (DRR) for allocation and migration of Virtual Machines between hosts. Lin et al. [7] introduced a dynamic Virtual Machine-Varying Based resource allocation using a threshold. Using this threshold their algorithm decides that the current counts of virtual machines which are assigned to an application are sufficient or not, it is the same for over provisioning. The basic differences and advantages of our study as compared to the latter are that first, our work does not need any human admin interferences and is able to approximate next workload instead of a reactive action. Reference [8] presented a thorough review of existing techniques for reliability and energy efficiency and their trade-off in cloud computing. It has also compared pro-active and reactive method in both failure and resource management levels. Calheiros [9] et al. addressed workload prediction and resource adaption using a queuing model and analytical performance, like previous work, there is a human control parameter in this.

A bottle neck detection system for multi-tier web applications using a heuristic approach was aimed by [10], this mechanism is able to detect bottle necks in every tier of the system. Jeyarani et al. in [11] developed a dispatcher using a new PSO (Particle Swarm Optimization) method Called SAPSO (Self-Adaptive PSO) to dispatch virtual machine instances among physical servers efficiently. Zaman et al. [12] showed a new bid based (capital market model) approach for responding to the users' requests. Islam et al. [13] advanced a new machine learning technique by developing a Neural Network system called ECNN (Error Correction Neural Network) and using it side by side with a Linear Regression.

Most of the methods relied on allocating physical resources to virtual resources and load balancing methods. Few of them considered the application layer. And among these rare studies, there is not a fully approximate based study. In [14] the authors tried to cover these leakages using a Heuristic Markovian Approach called SVMP (Smart Virtual Machine Provisioner), which is a novel quasi-DTMC learning based system. Authors of [15] developed novel algorithms based on static and dynamic strategies for both task scheduling and resource provisioning. In [16] a new data-aware provisioning algorithm is proposed to meet user-defined deadline requirements for data-intensive applications. The proposed algorithm takes into account available bandwidth and data transfer time. The [17] also proposed a new scheduling approach consists of a prediction model based on fractal mathematics and a scheduler on the basis of an improved ant colony algorithm.

III. PROPOSED METHODOLOGY

There are several existing studies on resource provisioning techniques and we explored a number of, some of them like [7] and [10] seemed to be good but not feasible in a real cloud environment, because they are reactive approaches and take action when the workload has already arrived, while creating a virtual machine is not instantaneous. The presented system can forecast needs of a cloud application with learning and estimations. Markov chain works well with stochastically changes [18] and is suitable for dynamic workloads. Hence a quasi-DTMC[1] heuristic approach has deployed to overcome the variety of the environment. But with latter studies, the authors found one more suitable method for environments with aggressive changes, which is a Learning Automata. Dependencies on parameters is another problem in investigations like [9], which is not favorable for an autonomous system. Considering the points, we have chosen a simple learning system which is fully autonomous. Also, complexity imposes overhead to the control system and will make an approach difficult to be accepted. Keeping the approach as non-complex as possible help our method to be implemented for each user in Cloud manager, in broker or even on the client side of the cloud system and this is another advantage of SVMP and also the proposed method.

A. *Learning Automata Background*

In classical control theory, the control of a process is based on full knowledge of the target environment. The mathematical model is considered to be known, and the inputs to the environment are deterministic functions of time. Later developments in control theory assumed the uncertainties present in the system. Stochastic control theory assumes that some of the characteristics of the uncertainties are known. However, all those assumptions on uncertainties and/or input functions may be insufficient to successfully

[1] discrete-time Markov chain

control the environment if it changes. It is then mandatory to observe the environment in operation and obtain further knowledge of the system, one approach is to view these as problems in learning [19].

A learning automata [20, 21] is an adaptive decision-making unit that improves its performance by learning how to choose the optimal action from a finite set of allowed actions through repeated interactions with a random environment.

The action is chosen at random based on a probability distribution kept over the action-set and at each instant the given action is served as the input to the random environment. The environment responds to the taken action in turn with a reinforcement signal. The action probability vector is updated based on the reinforcement feedback from the environment.

The objective of learning automata is to find the optimal action from the action-set so that the average penalty received from the environment is minimized. Learning automata have been found to be useful in systems where incomplete information about the environment exists[22].

So a learning automaton contains two main parts (Fig. 1):

1- A random automaton with a limited number of actions and a random environment that the automaton is associated with.

2- The learning algorithm that the automata use to learn the optimal action.

1) *Automata*

Automata can be defined as a quintuple $SA \equiv \{\alpha, \beta, F, G, \varphi\}$ in which $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ is the action set of automata, $\beta \equiv \{\beta_1, \beta_2, ..., \beta_r\}$ is the set of outputs of the automata, $F \equiv \varphi \times \beta \rightarrow \varphi$ is the new status generation function, $G \equiv \varphi \rightarrow \alpha$ the output function which maps the current state to the next output and $\varphi(n) \equiv \{\varphi_1, \varphi_2, ..., \varphi_k\}$ is the internal status set of the automata at the moment of n$^{(th)}$ repeat.

At the beginning of the automata activity, the probabilities of its actions are similar and equal to $\frac{1}{r}$ (Where "*r*" is the total number of automata actions.)

Also, the environment can be represented by a triple $E \equiv \{\alpha, \beta, c\}$, in which $\alpha \equiv \{\alpha_1, \alpha_2, ..., \alpha_r\}$ is the set of inputs of the environment, $\beta \equiv \{\beta_1, \beta_2, ..., \beta_r\}$ is the set of outputs of the environment, and $c \equiv \{c_1, c_2, ..., c_r\}$ is the probability set of punishments. The input of the environment is one of "*r*" actions of the automata.

2) *Learning Algorithm*

The learning algorithm is an algorithm that machine learning method can transform their perceptual perceptions into experiences; then they will take appropriate action in future looking to these experiences. In the learning automata, this algorithm is a function that updates the probability vector, using the feedbacks received from the environment.

Clearly, an action that has a higher probability in this vector would gain a higher chance while choosing we are choosing randomly.

If the learning automaton in nth repeat chooses one of its actions such as α_i and then receives the favorable response from the environment, $p_i(n)$ (the probability of the action α_i) increases and the probability of other actions decreases. Conversely, if the response of the environment is unfavorable, the chance of the action α_i decrease in the chance of other measures of automata will increase. In any case, the changes are made in such a way that the sum of all $p_i(n)$ remains constantly equal to "one".

$$P(n) \equiv \{p_1(n), p_2(n), ..., p_r(n)\} \tag{1}$$

$$\sum_{i=1}^{r} p_i(n) = 1\,, \quad \forall n \quad , \quad p_i(n) = \text{Prob}\,[\alpha(n) = \alpha_i] \tag{2}$$

B. *The Proposed Approach*

For the applied automaton, three modes are considered r=3 so $\alpha \equiv \{\alpha_1, \alpha_2, \alpha_3\}$, and the average utilizations of the virtual machines are considered as the amplification signal (feedback) as well. The followings describe further the designation and setting of the applied automata parameters.

1) *Determination Of Actions*

The total number of actions is equal to three, and the set values are equal to {decrease of resources, unchanged, increase of resources}

a) *"Increase Of Resources" action*

This state of the system is considered to be an appropriate output when automaton recognizes with regard to the feedback signal that resources available for upcoming workload would be inadequate and we would face a shortage of resources.

b) *"Decrease Of Resources" action*

This state of the system is considered to be an appropriate output when, automaton recognizes with regard to the feedback signal that the available resources are more than the real requirement of the workload we are going to have, and in fact, the waste of resources is going to be in progress.

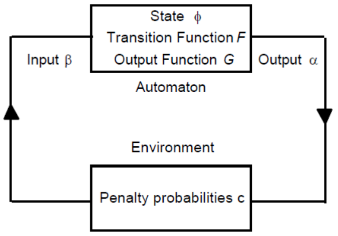

Figure 1. The relation between automata and the environment

c) *“Unchanged” action*

When the automaton determines, according to the feedback signal, that the provisioned resources for the workload we are ahead are suitable and so there is no loss or lack of resources, this state of the system is considered as an appropriate output.

2) *Feedback Signal*

For a variety of reasons, such as restricted access to user information by the broker and CSPs, and also data transfer overheads, the presented method was designed to use the minimum amount of environmental information as possible to make a decision. The finding such parameters itself is a critical and important choice; on one hand, it should meet conditions above, and on the other hand, it should contain sufficient information to make the decision as well.

Considering previous criteria in performance evaluation methods [23, 24] and looking at some experiences of authors in the real environments (e.g. Eucalyptus cloud infrastructures), finally, the (average) virtual machine utilization parameter was selected as the feedback.

Choosing utilization as the feedback signal had two main reasons:

- First, it's easily accessible through hypervisors, or even inside of virtual machines, also its system overhead is negligible.
- And second, because it can indicate the ratio of workload to available resources well.

Since the application considered to be deployed over several virtual machines, the average utilization of all them has been selected as the feedback. So, it will be a good representative of this parameter for all virtual machines.

3) *The learning algorithm*

As it was mentioned before, in the learning automata, this algorithm is a function that updates the probability vector, using the feedbacks received from the environment. If the learning automaton in nth repeat chooses one of its actions such as α_i and then receives the favorable response from the environment, $p_i(n)$ (the probability of the action α_i) increases and the probability of other actions decreases. Conversely, if the response of the environment is unfavorable, the chance of the action α_i decreases while the probabilities of other actions increase.

There are several types of these algorithms but in the following, we defined the one which is used in this paper.

If c_i = 0 (favorable response) and so the recent action is a desirable choice, then its probability will be rewarded and we will have:

$$\begin{aligned} p_i(n+1) &= p_i(n) + a \times (1 - p_i(n)) \\ p_j(n+1) &= (1-a)\, p_j(n) \qquad \forall j, \quad j \neq i \end{aligned} \tag{3}$$

In counter with that, if an unfavorable response was received, so the recent action was an undesirable choice and the penalty probability considered as (c_i = 1), its probability would be punished and then we will have:

$$\begin{aligned} p_i(n+1) &= (1-a)\, p_i(n) \\ p_j(n+1) &= \frac{a}{r-1} + (1-a)\, p_j(n) \qquad \forall j, \quad j \neq i \end{aligned} \tag{4}$$

C. *The Intensity Control*

A point was made during the experiments, although the learning automata seem suitable for environments with large variations and can provide an acceptable response to the needs of these randomized environments, but has a slow rate of convergence. This characteristic could be exacerbated when the number of actions increases. On the other hand, the three steps we have taken for this system alone seems would not be able to meet the requirements of our resource provisioning policy. Consider the condition that the system detects that "Increase of Resources" is now required to be done and based on the workload that it is just enough to add a virtual machine to the load. In the next few moments, with the rise in the loading rate, the same operation is needed again, but this time a virtual machine is not enough for this volume of work (for instance, four virtual machines are needed this time). The question that arises is what should control the number of increases? The first method seems to be through the automata, which can be defined by increasing the number of actions, taking into account a certain number of increases/decreases of virtual machines in them. But as mentioned above, this will increase the convergence rate and in addition, may increase the complexity of the method.

Another answer which is proposed for the issue is a severity of actions control system that works in the direction of learning automata. After the learning automaton detects which state (action) should be selected, it will send the result to the severity control system. This system uses the same feedback parameter that was applied for learning automata, which is the average utilization of virtual machines to determine the intensity of the selected action. The response severity criterion is obtained empirically and by modeling the human decision-making method, it is why the decrease steps are considered conservatively in more stages (Fig. 2).

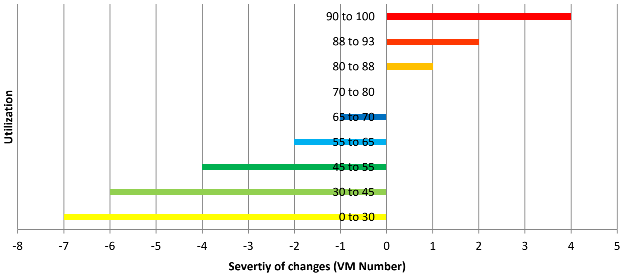

Figure 2. The severity criterion

D. Problem Formulation

The purpose of the proposed automata in this study is to minimize allocated resource to the application while avoiding the saturation of Virtual Machines. In better words for a particular arriving workload (W), we want to minimize the number of virtual machine instances while providing the appropriate amount of resources for the application. So the total processing power in MIPS must be greater than total workload in MI:

Object for:

$$\min\left(\sum_{n=1}^{\mathrm{MaxVM}} \mathrm{VMlist}_{n}^{W}\right) \qquad (5)$$

Subject to:

$$\sum_{1}^{MaxOnlineVMs} MIPS_{VirtualMachines} > \sum_{1}^{CurrentCloudLetNumber} MI_{CloudLets} \qquad (6)$$

Also, the average utilization of virtual machines under workload (W) with allocated virtual machine number (OnlineVMs) is needed to be calculated, so it has been formulated as below:

$$VMs\ Avg.\ Utilization^{W} = \frac{\sum_{i=1}^{\substack{Online\\VMs}} VM_{i}^{W} Utilization}{OnlineVMs} \qquad (7)$$

IV. Experimental Setup

The proposed algorithm (called Learning Automata based VM Provisioner – LAVMP) is implemented by the CloudSim simulator, which is suited to simulate the provision of resources in the cloud [25]. The system contains two basic components. On broker which is considered out of the cloud environment, the broker observes workload of the application (from the cloud user's side) and communicates with CSP to adapt resources. The other is a dispatching system within the cloud that directs workload to virtual servers appropriately.

However this tool typically is not able to simulate Dynamic Virtual Machine provisioning beside the disability to resource provisioning in the application layer, which the authors purpose to simulate, so new components and attributes have been added to the simulator to enable it to handle Dynamic VM provisioning in the application layer.

Besides the LAVMP, we used the same dispatcher which we have implemented in SVMP [14] to direct user workloads (called cloudlets in the Cloudsim) among available VM instances. This dispatcher fills each VM with incoming cloudlets until VM utilization is under 80%, the remaining 20% is reserved for eventual heavy loads. With this method VMs would be utilized in a reasonable manner, moreover over provisioned VMs remain empty of load and can be easily shut down. For some cases like web servers this threshold is considered 85% [10, 26], but LAVMP is designed for more general applications besides it is a learning based system and take some time to learn, so our threshold was set to 80%.

To evaluate the proposed system, four tests are performed using four different methods. The first and second experiments show the behavior of the cloud using two specific static methods. The third and fourth experiments examine the behavior of the cloud system using two dynamic resource provisioning methods.

For SVMP (our previous study), a normal workload was applied which begins with a small amount of processing load, it continuously climbs, and then the amount begins to decrease. But here we are going to use another type of load which seems to be more challenging for both of our smart systems. The second type of work load is a function with stair changes, either ascending or descending [10].

This pattern used in the evaluation has constant static changes over each run time which means, the values of the workload are changing frequently but with a sudden and certain amount. This load has two peak points with different values (Fig. 3).

The authors also tried to challenge their previous work (SVMP) by applying the new workload to it, besides the new method (LAVMP).

At the end for more detailed designing, a Service Level Agreement (SLA) request from user side is assumed, these basic parameters are considered for all VMs which are used in the following experiments (TABLE I.).

TABLE I. SLA Parameters for running the scenario

SLA Parameters				
Workload type	***Max VMs***	***VMs CPU Core(s)***	***Core Processing Power***	***VMs RAM***
CPU Intensive	20	1	400 MIPS	512 MB

Cloud user deploys an application in the cloud on several virtual machines (Web server e.g.).

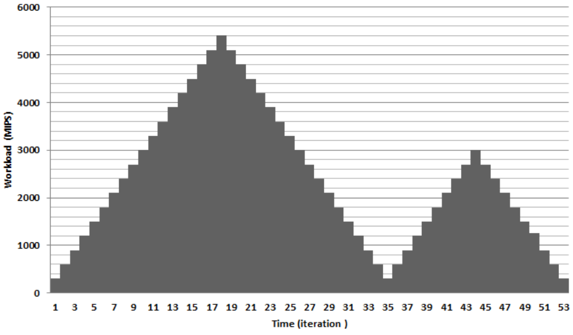

Figure 3. The Workload generation profile for all experiments

Since a CPU intensive workload has been chosen, the CPU utilization is considered as the preferred utilization for the feedback.

V. Experimental Result

As it was mentioned earlier, for simulation four different methods were considered (TABLE II.). It allows us to compare the new approach with other methods in the same condition.

A. *Experiment 1: Over Provisioning-Static*

In the first experiment, 20 virtual machine instances were statically assigned to the application for the assumed workloads. This amount of resources is extracted by looking at experiments and the reaction of our dynamic algorithms under maximum load, considering a conservative behavior for provisioning.

The behavior of the system under the workload described above is profiled (Fig. 4). This type of provisioning is clearly an overabundance. This is a common resource allocation strategy used by users as an application owner in the cloud environment and is actually the same thing we did to get the values mentioned above. Since it is quite difficult for application managers to determine the optimal amount of virtual machines for applications where their workload is very variable, they prefer to spend more on additional resources. Otherwise, they will have to pay huge fines to their users due to a violation of the Service Level Agreement (SLA) and beside that customers' dissatisfaction will also cause irreparable losses to them.

B. *Experiment 2: Mean Provisioning-Static*

This section describes the experimental results using a static provisioning with mean virtual resource provisioning policy. The term mean is used because here the virtual machine number is the mean of minimum allocate able virtual machine (1) and maximum allocate able virtual machine (used in the previous experiment).

The number of virtual machines is constantly equal to 10 and independent of the workload changes. The processing power of this number of virtual machines according to (TABLE I.) is consistently 4,000 MIPS.
The Fig. 4 shows that this method, although is more efficient than the previous one but is saturated by the maximum load (for the first peak of the workload), while the strategy cannot provide a solution. Apart from it, this method faces the same problem just like the previous method, in the second peak of the load; we have had over provisioning and resource waste again.

TABLE II. Designed Experiments for Evaluating LAVMP

Experiment	Provisioning Method
Experiment 1	Max Static Provisioning (Over Provisioning)
Experiment 2	Mean Static Provisioning
Experiment 3	Dynamic Provisioning Using (SVMP) [14]
Experiment 4	Dynamic Provisioning Using Proposed System(LAVMP)

C. *Experiment 3: SVMP-Dynamic*

This section describes the results of the experiment performed using the SVMP algorithm. Unlike previous methods, due to the dynamic nature of this method, the number of virtual machines and therefore the amount of resource provided by the algorithm is not constant but is always changing according to the demands of the application.

The (Fig. 4) shows that this method is able to detect saturated and underutilized spots and handle them by resource supplying and depriving respectively.

D. *Experiment 4: LAVMP-Dynamic*

The results of testing the proposed method under the same workload are demonstrated in this section. This method, just like the previous one, is a dynamic approach, with the difference being of a random nature. Since the choice of modes is based on their probability of occurrence (in fact the chance of their selection) and is random of course, so in different repeats of the experiment the results would be in the same direction but will not be exactly the same. Hence, the published results for this method resulted from ten times repetition of the test and averaging of the extracted results.

Results (Fig. 5) demonstrate that this method, just like the latter dynamic method, is also able to detect the saturation and underutilization. It also shows that the proposed method in this study (LAVMP) has been able to perform better in comparison with the earlier dynamic approach.

E. *Results Discussion*

In the last part of this section we will compare the results of the experiments and finally, these results will be examined and concluded. First, we compare the results of the previous section in the provision of resources.

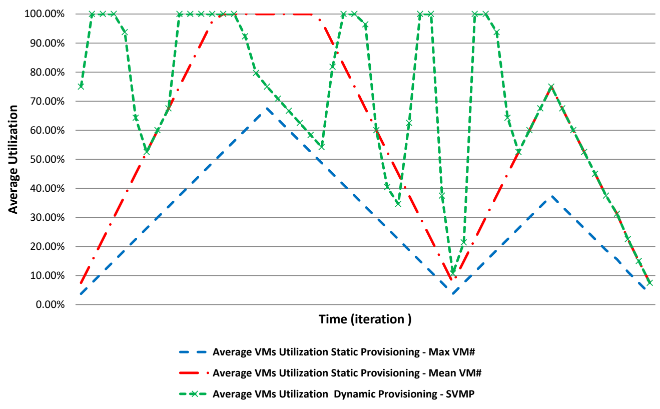

Figure 4. Average utilization of virtual machines using previous methods

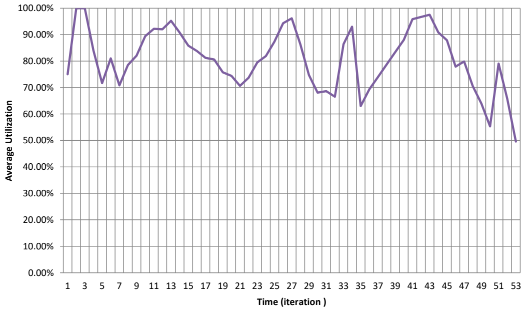

Figure 5. Average utilization of virtual machines with dynamic provisioning policy (LAVMP method).

It seems that the proposed method (LAVMP) has been able to control the situation more accurately and more smoothly compared to the SVMP. It follows from this that the utilization diagram shows a higher average for it (Fig. 6) while the saturation is degraded as well (Fig. 7).

In fact quality plays a very important role in the competitive market, but the amount money that customers spend is also important. This will ultimately be an acceptable way for a service provider to minimize the cost to them while meeting their requirement. Therefore dynamic methods have emphasized on consumer price reductions as well. The importance of this parameter has led us to consider it as one of the parameters for comparing the approaches in this study.

This criterion is calculated in terms of aggregating the unit cost of all virtual machines per unit time during iterations of the experiments[2] (Fig. 8).

The cost in the static mean method is half the maximum provisioning method, which clearly is because the way it was calculated by the mean between Maximum and Minimum allocate able resources. In the SVMP dynamic method, this amount is reduced, but for the proposed method in this study (LAVMP), this value is the lowest among all examined methods.

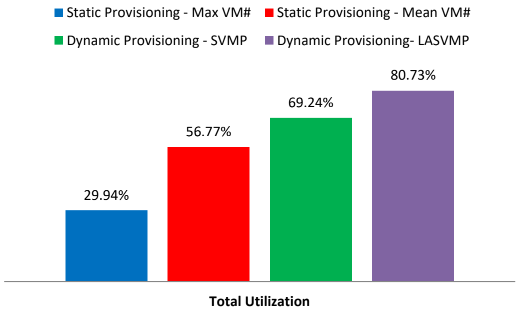

Figure 6. Comparing total virtual machines utilization averages in experimented approaches.

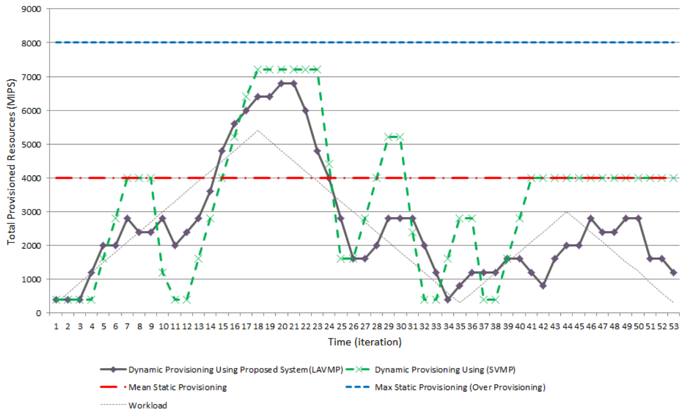

Figure 7. Comparing total resources provisioned by different approaches

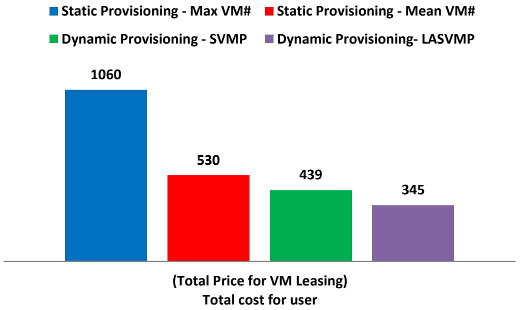

Figure 8. Comparing total virtual machines cost for the cloud user in experimented approaches.

VI. Conclusion

By rapid growth of Cloud Services, users become more interested in using application services (SaaS). Although application providers try to use IaaS for its benefits, resource capacity planning for such an environment is going to be more complex. It takes a few minutes from the time that a CSP receives a VM request to the time that the VM is up and ready to use; that is why for an effective provisioning system, it is necessary to predict the application workload behavior and provide the resources before the workload arrival. The Learning Automata based smart application scaling system which is presented in this paper (LAVMP) addresses a provisioning system which is able to adapt the amount of resource to the application requirements while keeping cost and QoS parameters simultaneously. We are going to evaluate our provisioning systems with more examinations criteria and of course extending our system to employ other learning algorithms to improve performance and accuracy.

[2] The pricing policy might be different in CSPs , so the price is considered as VM Unit Cost Per Unit Time

REFERENCES

[1] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Load balancing and unbalancing for power and performance in cluster-based systems," 2001, doi: 10.1.1.16.2690.

[2] E. N. Elnozahy, M. Kistler, and R. Rajamony, "Energy-efficient server clusters," presented at the Proceedings of the 2nd international conference on Power-aware computer systems, Cambridge, MA, USA, 2003, doi: 10.1007/3-540-36612-1_12.

[3] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and Performance Management of Virtualized Computing Environments Via Lookahead Control," presented at the Proceedings of the 2008 International Conference on Autonomic Computing, 2008, doi: 10.1109/icac.2008.31.

[4] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems," presented at the Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, Leuven, Belgium, 2008, doi: 10.1007/978-3-540-89856-6_13.

[5] H. N. Van, F. D. Tran, and J. M. Menaud, "Performance and Power Management for Cloud Infrastructures," in Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, 2010, pp.329-336, doi:10.1109/cloud.2010.25.

[6] C.-C. Lin, P. Liu, and J.-J. Wu, "Energy-Aware Virtual Machine Dynamic Provision and Scheduling for Cloud Computing," in Cloud Computing (CLOUD), 2011 IEEE International Conference on, 2011, pp. 736-737, doi:10.1109/cloud.2010.25.

[7] W. Lin, J. Z. Wang, C. Liang, and D. Qi, "A Threshold-based Dynamic Resource Allocation Scheme for Cloud Computing," Procedia Engineering, vol. 23, 2011, pp. 695-703, doi:10.1016/j.proeng.2011.11.2568.

[8] Y. Sharma, B. Javadi, W. Si, and D. Sun, "Reliability and energy efficiency in cloud computing systems: Survey and taxonomy," Journal of Network and Computer Applications, vol. 74, 2016, pp. 66-85, doi:10.1016/j.jnca.2016.08.010 .,In Press.

[9] R. N. Calheiros, R. Ranjan, and R. Buyya, "Virtual Machine Provisioning Based on Analytical Performance and QoS in Cloud Computing Environments," in Parallel Processing (ICPP), 2011 International Conference on, 2011, pp. 295-304, doi:10.1109/icpp.2011.17.

[10] W. Iqbal, M. N. Dailey, D. Carrera, and P. Janecek, "Adaptive resource provisioning for read intensive multi-tier applications in the cloud," Future Generation Computer Systems, vol. 27, 2011, pp. 871-879, doi: 10.1016/j.future.2010.10.016.

[11] R. Jeyarani, N. Nagaveni, and R. Vasanth Ram, "Design and implementation of adaptive power-aware virtual machine provisioner (APA-VMP) using swarm intelligence," Future Generation Computer Systems, vol. 28, pp. 811-821, 2012, doi: 10.1016/j.future.2011.06.002.

[12] S. Zaman and D. Grosu, "An Online Mechanism for Dynamic VM Provisioning and Allocation in Clouds," in Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, 2012, pp. 253-260, doi: 10.1109/cloud.2012.26.

[13] S. Islam, J. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," Future Generation Computer Systems, vol. 28, pp. 155-162, 2012,doi:10.1016/j.future.2011.05.027.

[14] H. R. Qavami, S. Jamali, M. K. Akbari, and B. Javadi, "Dynamic resource provisioning in Cloud Computing: A Heuristic Markovian approach," in Cloud Computing: 4th International Conference, CloudComp 2013, Wuhan, China, October 17-19, 2013, V. C. M. Leung and M. Chen, Eds., ed Cham: Springer International Publishing, 2014, pp. 102-111, doi: 10.1007/978-3-319-05506-0_10.

[15] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Algorithms for cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds," Future Generation Computer Systems, vol. 48, 2015, pp. 1-18 , doi: 10.1016/j.future.2015.01.004 .,In Press.

[16] A. Nadjaran Toosi, Richard O. Sinnott, and R. Buyya, "Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using Aneka," Future Generation Computer Systems, 2017, doi:10.1016/j.future.2017.05.042 .,In Press.

[17] H. Duan, C. Chen, G. Min, and Y. Wu, "Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems," Future Generation Computer Systems, vol. 74, 2017, pp. 142-150, doi: 10.1016/j.future.2016.02.016 .,In Press.

[18] S. U. P. Athanasios Papoulis, Probability, Random Variables and Stochastic Processes, 4 ed. vol. 1: McGraw-Hill Europe, 2002.

[19] C. Unsal, "Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles Approach," PhD Dissertation, Electrical and Computer Engineering, 1998.

[20] M. A. L. T. K.S. Narendra, Learning Automata: An Introduction. New York: Prentice Hall, 1989.

[21] M. A. L. T. S. Lakshmivarahan, "Bounds on the Convergence Probabilities of Learning Automata," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-6, 1976, pp. 756-763, doi: 10.1109/TSMC.1976.4309449.

[22] J. Akbari Torkestani, "An adaptive learning to rank algorithm: Learning automata approach," Decision Support Systems, vol. 54, 2012, pp. 574-583, doi: 10.1016/j.dss.2012.08.005.

[23] D. A. Menasce, L. W. Dowdy, and V. A. F. Almeida, Performance by Design: Computer Capacity Planning By Example, 1st ed.: Prentice Hall, 2004.

[24] D. A. Menasce and V. Almeida, Capacity Planning for Web Services: metrics, models, and methods: Prentice Hall PTR, 2001.

[25] 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 & Experience, vol. 41, 2011, pp. 23-50, doi: 10.1002/spe.995.

[26] J. Allspaw, The Art of Capacity Planning: Scaling Web Resources: O'Reilly Media, 2008.