An Effiecient Approach for Resource Auto-Scaling in Cloud Environments

Bahar Asgari[1], Mostafa Ghobaei Arani[1], Sam Jabbehdari[2]
[1] Department of Computer Engineering, Mahallat Branch, Islamic Azad University, Mahallat, Iran
[2] Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran

Article Info

Article history:

Received Mar 27, 2016
Revised Jul 6, 2016
Accepted Jul 28, 2016

Keyword:

Auto-scaling
Cloud computing
Markov decision process
Reinforcement learning
Scalability

ABSTRACT

Cloud services have become more popular among users these days. Automatic resource provisioning for cloud services is one of the important challenges in cloud environments. In the cloud computing environment, resource providers shall offer required resources to users automatically without any limitations. It means whenever a user needs more resources, the required resources should be dedicated to the users without any problems. On the other hand, if resources are more than user's needs extra resources should be turn off temporarily and turn back on whenever they needed. In this paper, we propose an automatic resource provisioning approach based on reinforcement learning for auto-scaling resources according to Markov Decision Process (MDP). Simulation Results show that the rate of Service Level Agreement (SLA) violation and stability that the proposed approach better performance compared to the similar approaches.

Corresponding Author:

Bahar Asgari,
Department of Computer Engineering,
Mahallat Branch, Islamic Azad University,
Mahallat, Iran.
Email: Bahar_asgari88@yahoo.com

1. INTRODUCTION

Cloud computing is the number of virtualized connected computers which offers single computational resource dynamically and to compute complex computation [1]-[3]. In other word, cloud computing states to both applicable programs offered as services on the Internet, hardware and software systems in data centers. By definition, in data center hardware and software are called "cloud". Scalability is one of the basic concepts in cloud computing which is important in using higher efficiency of cloud computing [4]. Scalability is referred to increase system functional power to have suitable response against increased work load of course by adding software and hardware resources [5]. Whereas applications, especially application program on web, do not have regular work load patterns so that scalability functions (increase or decrease of scale) should have be done immediately with minimum human intervention to provide resources for applications as soon as possible. Resource scaling with minimum human intervention is called auto-scaling [6]-[8]. Various workloads are of the biggest challenges in different times, so whenever provider wants to meet all the requirements in all times, it should reserve maximum needed resources previously for peak work load to support them. In this situation provider sometimes will be over-provisioning and it is going to be very costly for them (to buy maximum resources at peak times) which leads to lower profit. Therefore functional expenses will be reduced by turning off idle nodes on idle times, but it cannot decrease financial expenses related purchasing and hosting IT equipment's and their depreciation. If provider possesses only enough resources (average capacity) to support average number of requests, the providers may be utilized, but the provider might not have enough local resources to meet clients' request which leads to under- provisioning in some situations so provider has to reject new customers or cancel previous services operating on system. We should design a system which will be able to manage uncertainty and remove any problems in cloud environment. Also it should be able to impact parameters like expense, efficiency, SLA violation etc.

We offer auto-scaling according to reinforcement learning. Reinforcement learning (RL) is a kind of decision making that determines a goal performing functional model, applies policy without previous information. RL has been performed successfully in extensive fields to support auto-control and dedicate resources [9]-[12] which works on the basic assumption of penalty and reward so the factors move toward operations which lead to highest profit. Major part of RL is on the basis of determination of optimal policies in Markov [13],[14].

In this paper we want to propose auto-scaling approach using MDP to manage SLA violation and scaling expense and to preserve system stability. RL has the capacity to response suitably using environment experiences. RL leads to better management of compromise SLA violation and number of scales but it causes higher expenses. The rest of this paper is organized as follows: we review related works about RL in second part; the proposed approach comes in third part in detail. The performance evaluation of proposed approach will be explained in fourth part. Finally conclusion and suggestion will be presented in fifth section.

2. RELATED WORKS

Various studies have been carried out about auto-scaling and its implementation. Current approaches have advantages and disadvantages. As the proposed approach in this paper is based upon RL, we review researches related to this technique in this section.

- Enda Barrett et al. [14] have been considered the parallel Q learning to reduce time of determination about optimal policies and online learning. Their proposed approach uses MDP along with RL.
- Fouad Bahrpeyma et al. [15] suggests RL-DRP approach. They use neural networks in their proposed mechanism. The approach enable cloud service providers to meet high volume of requests without wasting any time, valuable work and at the same time control resources optimally.
- Xavier Dutreilh et al. [16] have proposed using proper initialization in primary stages also increasing the rate of convergence in process of learning to solve problem. They have offered experiments results. Also they have introduced an efficient model to detect changes then completed learning process management based on that.
- Bauer et al. [17] proposed using RL to manage threshold orders. First controller applies these orders to the goal program to reinforce its quality features. Second controller supervises the orders, adapts thresholds and changes conditions, also it deactivates unrelated orders.
- Jia Rao et al. [18] represent a RL aware virtualized machine configuration (VCONF). Central design of VCONF is prepared based on RL aware model to scale and adapt.
- Amoui et al. used RL successfully in management quality of web programs to optimize program's output [19]. Using simulation in initialization of learning functions is one of the interesting aspects of it.

Finally Table 1 shows the comparison of above techniques.

Table 1. Comparison of Techniques

Reference	Auto-scaling technique	Advantages and disadvantages	Contribution
Enda Barrett [14]	Parallel Q learning	Decreasing time of optimal policy determination and online learning Disadvantages: challenges in determination of initial policies	It uses inherent parallelism in distributed computing platforms like cloud
Fouad Bahrpeyma [15]	RL	Fast convergence process Higher utilization	It introduces a new decision making process to use predictably analysis of demand which considers parameters of offer and demand
Xavier Dutreilh [16]	RL	Horizontal scaling Increase in convergence rate in learning stages	Integration in a real cloud controller and auto programming
Bahati [17]	RL	It limits situation as pair of operation-condition and provides the possibility of re-use of learned models in an order set for next stage Reinforcement of load based on effective limit	They proposed using RL to manage threshold orders. First controller applies order to the goal program to improve the features of quality
JiaRao [18]	VCONF	VCONF is good adaptation with online auto configuration policies with heterogeneous VMs VCONF is enable to guide initial setting without decreasing in function of VMs	Central design of VCONF using RL aware model works to scale and adapt
Amoui [19]	RL	Quality management in application of new web design to optimize program output	Using simulation for initialization of learning functions

3. PROPOSED APPROACH

Final goal is to make auto scaling system to have the ability of decreasing costs and increasing system stability; at the same time with SLA requirements and system efficiency It means to use an online policy to dedicate resources with scaling automatically. Proposed approach will be introduced according to RL and MDP. The offered MDP constitutes from 4 categories included conditions, operations, transmitted possibilities and rewards so that decision making about scale up/ scale down will be accomplished based on it.

3.1. Reinforcement Learning (RL)

RL [7],[14],[15] is a computational approach to understand automatic base learning to make the best decisions. It insists on learning via direct involvement of agent and environment. Decision maker refers to the agent who learns from experience and its best action is to perform at its maximum in any environment. An auto scaler is responsible for decisions about scaling without human involvement and its objective is to adapt resources dynamically to the applications according to input workload. It decides to allocate or deallocate resources to the applications based on workload. In any t time which t= 0,1,2,… time sequences are separated, agent shows condition of environment $t_s \in S$ where S is all possible conditions and it selects $a_t \in A(s_t)$ where $A(s_t)$ is all variables in the condition of s_t but in a determined time, sequence of these functions and agent will be the next reward r_{t+1} which finds itself in new condition of s_{t+1}. Agent will select from condition possibilities then operates the possible action. This will be agent's policy that shows πt in which πt (s, a) as $a = a_t$ at the condition $s = s_t$.

So MDP can be shown in four categories included conditions, operations, transmitted possibilities and rewards:

- **S:** Environmental state space
- **A**: total action space
- **P(.|s, a)**defines distribution of governed possibilities on transmitted conditions
  $s_t + 1 \sim p(.\mid s_t, a_t)$
- **Q (.|s, a)** defines distribution of governed possibilities on received reward.
  $R(s_t, a_t) \sim q(.\mid s_t, a_t)$

The objective of learning process inside learning Q is to achieve the optimal policy which reflects by Q amount in general reward and continues by operating in current situation. The amount of Q will be calculated by equation 1 which includes discounted reward (decreased reward) and shows RL process policy.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left(r_{t+1} + \gamma \max_a Q(s_t, a) - Q(s_t, a_t)\right) \quad (1)$$

Where r_{t+1} is medium received reward of selecting a_t in s_t condition a is learning rate and γ is discount coefficient (Reduction). The overall process of RL has been shown in Algorithm 1:

Algorithm1: Reinforcement Learning Algorithm (Q- learning)

```
1. Initialize Q(s,a) arbitrarily
2. Repeat ( for each episode)
3. Initialize s
4. Repeat
5. Choose a from s using policy derived from Q (€- greedy)
6. Take action a and observe r,s'
7. $Q(s_t, a_t) \rightarrow Q(s_t, a_t) + a\left(r_{t+1} + \gamma \max_a Q(s_t, a) - Q(s_t, a_t)\right)$
8. s←s';
9. Until s is terminal
```

3.2. Proposed Algorithm

The proposed algorithm has been offered based on RL, that is defined according to Markov process for auto scaling a MDP. Upper and lower threshold have been defined too and cloud service operation will be monitored after introduction proposed MDP.

Configuration of proposed MDP has been considered as follows:

S: the space of condition: Full utilization, Under utillization, Normal utilization.
A: the space of operation: Scale up, Scale down, No- op.
P (.|s, a) defines distributionpossibility governed on transmitted conditions.
Q (.|s, a) defines distribution possibility governed on received reward.

As we know MIPS means the number of instructions per second. There has been introduced two variables for proposed approach included available MIPS and Requested MIPS, both are variables of service inputs. Q Updating will be done using local regerssion according to history of instructions. Division of two amounts shows the amount of utillization (equation 2) and comparison of upper and lower thresholds determine space of condition. The full utilization and the under utilization condition show in Equation 3 and 4, respectively.

$$Utilization = \frac{Available\ MIPS}{Requested\ MIPS} \tag{2}$$

$$\begin{gathered}\left(\text{Requested}\,MIPS\,/\,\text{Available}\,MIPS\right) > \left(\text{High-Threshold}\right) \Rightarrow \\ \left(\text{Full-Utilization}\right) \Rightarrow \left(\text{Under-Provisioning}\right)\end{gathered} \tag{3}$$

$$\begin{gathered}\left(\text{Requested}\,MIPS\,/\,\text{Available}\,MIPS\right) < \left(\text{Low-Threshold}\right) \Rightarrow \\ \left(\text{Under-utilization}\right) \Rightarrow \left(\text{Over-Provisioning}\right)\end{gathered} \tag{4}$$

After defining full utilization, under utilization and normal conditions, and operating equation 1 (Q(s, a)), SLA violation amount will be acquired by Requested MIPS and available MIPS then decision will be made according to above functions to do current action, it means to increase or decrease virtual machine or no operation. Table 2 represents process of decision making and Figure 1 shows a diagram included provider condition changes regarded to utilization parameter.

Table 2. Decision Making By MDP

	Utilization>High-Threshold	Low-Threshold< Utilization < High -Threshold	Utilization <Low-Threshold
State(t)	Full-Utilization (Under-Provisioning)	Normal-Utilization (Normal- Provisioning)	Under-Utilization (Over-Provisioning)
Next-Action(t+1)	Scale_up	No-op	Scale_down

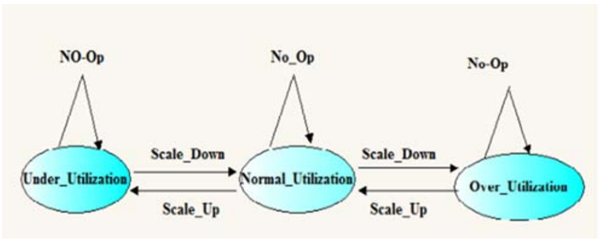

Figure 1. Condition of provider changes regarded to utilization parameter

Proposed algorithm introduced in this paper is represented as semi code offered in algorithm 2 according to Markov and decision making in Table 2.

Algorithm2: Reinforcement Learning (Q-Learning**)**

1. Initialize Q (s, a)=0 ,s=0, a=0, high Range Q=0.8, low Range Q=0.2.
2. Observe the Available MIPS and Requested MIPS.
3. Observe the current state S.

```
4.  If (requested MIPS/available MIPS) > high Range Q) , state [0]= 0; /*Full-Utilization state*/
5.  Else if (requested MIPS/available MIPS) <Low Range Q), state[1]= 1; /* Under-Utilization state */
6.  Else state [2]= 2; /* Normal-Utilization state*/
7.  Loop
8.  Select action, choose for state ,based one of the action selection policy Utilization
9.  Take action, observe r, as well as the new state, s'.
10. Update Q value for the state using the Regression and observed r and the maximum reward possible for the next state.
11. $Q(s_t, a_t) \rightarrow Q(s_t, a_t) + a(r_{t+1} + \gamma \max_a Q(s_t, a) - Q(s_t, a_t))$
12. Set the state s to the new state s' , s←s'
13. Until s is terminal
```

4. PERFORMANCE EVALUATION

There has been used of Cloudsim [20] simulator for simulation. Four kinds of virtual machine corresponded to Amazon EC2 [21] have been performed which their specifications offered in Table III. There have been used four kinds of services regarded to the variety of available services in cloud and we have not focused on type of service or special program so that used services are independent to programs. These services are combination of all heterogeneous programs like HPC, Web and so on. Also work load has been modeled according to normal distribution to be closer to real world. Scaling will be done in 24 hour period and in 5 minutes intervals (288 five minutes), Low-Threshold is considered 0.2 and High-Threshold is considered 0.8 Standard deviation is 3000 MIPS and Diff Range is 0.4 There has been considered a function for initialization cost. As cost function is computed by hour and we have 5 minutes intervals so that we have to multiple overall costs by 300/3600**.**

Table 3. Specification of Virtual Machine

Type Of Virtual Machine	*MIPS (CPU)*	*Core*	*RAM (MB)*	*Price (Cent)*
Micro	500	1	633	0.026
Small	1,000	1	1,700	0.070
Extra Large	2,000	1	3,750	0.280
High-CPU Medium	2,500	1	850	0.560

Algorithm works by updating Q. we have done Q updating and obtaining utilization by local regression. Updating Q will be accomplished according to instruction history; it means next amount will be determined according to prediction of previous amount. Predicted amount should be multiplied by 0.7, because error possibility has been considered as 30 percent. Regression function helps us to scale VMs in the way that decrease failed case along with minimum cost.

The amount of Available MIPS and Requested MIPS is calculated in the main function of proposed approach. Also the amount of SLA violation is calculated using their difference. Requested MIPS is divided to Available MIPS to introduce two dimensional array r [rstate] [0], r [rstate] [1] and r [rsatate] [2] which determines underutilization, full utilization and normal functions. Then the amount of r will be updated and the amount of equation Q (s, a) will be calculated.

Overall MIPS amount is obtained by division of violation rate to the total MIPS then optimal current action will be selected using **Current Action()** and **Select action()** according to the amount of utilization. Then decision for scale down and scale up or null action will be made.Proposed approach which is learning automata aware [22] will be compared to cost aware auto scaling approach which is a simple, automata approach by parameters like cost, SLA violation, initialization cost and number of scaling. There has been defined three scenario for evaluation proposed approach in Table 4.

Table 4. Evaluation Scenarios

Scenario	*Goal*
First scenario	Minimization SLA violation
Second scenario	Minimization Total Cost
Third scenario	Minimization number of scaling

4.1. First Scenario

Evaluation of SLA violation has been considered in first scenario compared to two other approaches. SLA violation will be happened when provider cannot provide predefined measures in SLA for users. Some examples of SLA violation is the number of lost deadlines, lack of warranty on agreed MIPS, lack of warranty on agreed bandwidth, number of rejected requests because of not having enough resources at the peak times. Increasing rate of SLA violation causes lower quality in providing services for user. If Requested MIPS is not match with available MIPS, SLA violation will happen. Figure 2 represents results of comparison of SLA violation in three compared approaches for 4 services. As you can see SLA violation rate in proposed approach is less than the others.

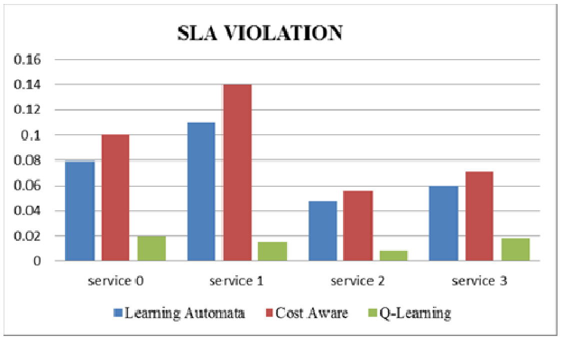

Figure 2. Comparison of SLA violation in services

Figure 3 shows the comparison of overall SLA violation for services included cost aware, learning automata and proposed approach. As you can see results of proposed approach simulation compared to learning automata and cost aware approach has lower rate of SLA violation at the time of simulation so that using Q learning technique in auto scaling leads to reduce SLA violation. So whenever SLA is important for auto scaling, we can use proposed approach.

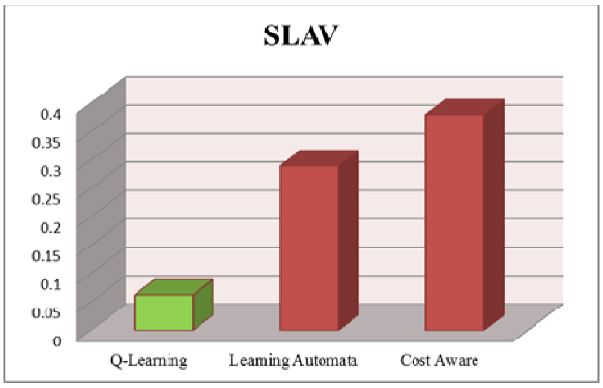

Figure 3. Comparison of overall SLA violation in three approaches

4.2. Second Scenario

We address evaluation of cost measure and comparing it with other approaches. Service cost will be calculated according to hours of utility. It means user pays the cost according to speed, power and capacity of requested resource (CPU, Memory and disk and …) also time of using resource. Naturally cost will be low when we use resource with lower speed and capacity in lower intervals. It can decrease cost but affects other quality factors. So to have a high quality service we have to increase cost. Cost is one of the most important factors for users. It means that user always struggles to accomplish the request with minimum cost. Parameter of cost introduces in three categories: Initialization cost, Runtime cost and Total cost. Initialization cost is initial cost for setting up VMs. Runtime cost equals to cost according to utility per hour which will be paid for VMs operation. Total cost calculates by equation 5:

$$Total\ Cost = Initialization\ Cost + Runtime\ Cost \quad (5)$$

Runtime cost of VM increases nearly 20 percent in simulation of vertical scale up. Cost measure in simulation is calculated according to addition of VM initialization cost and VM runtime cost. Figure 4 shows VM initialization cost. It specifies that a Q aware approach has high initialization cost while an automata aware approach will save initialization cost substantially.

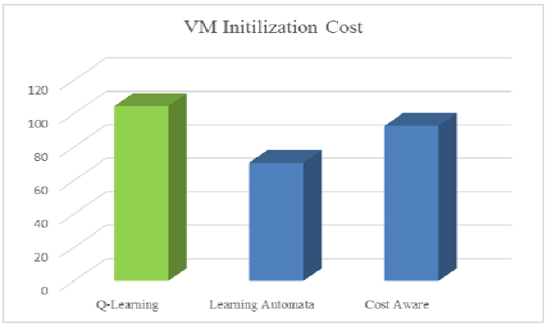

Figure 4. Comparison of initialization cost in three approaches

Figure 5 represents VM runtime cost in 3 services in 24 hours. Results of simulation shows proposed approach has lower runtime cost.

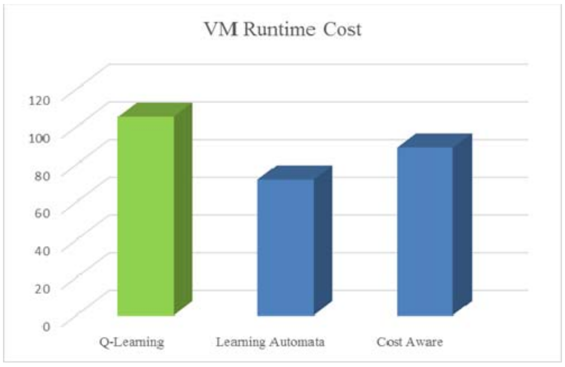

Figure 5. Comparison of VM runtime cost in three approaches

Figure 6 represents results of simulation according to total cost of scaling for 3 compared approaches in a 24 hour period.

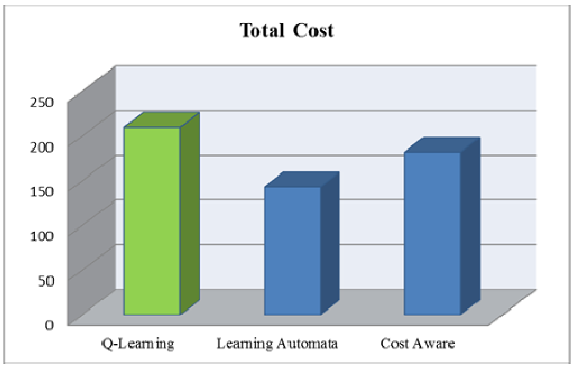

Figure 6. Comparison of total cost in three approaches

Results of simulation according total cost in 24 hours for four services have been represented in Figure 7.

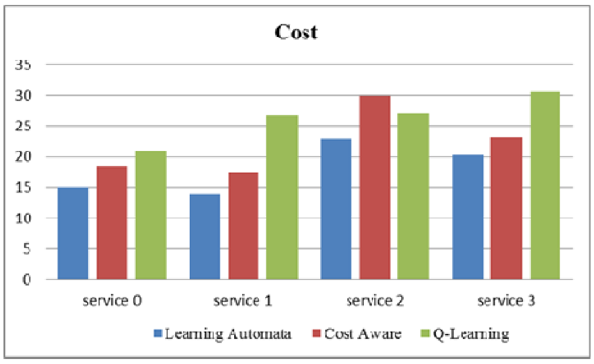

Figure 7. Comparison of overall thetotal cost in three approaches

As it is obvious in Figure 6 and 7, scaling aware of learning automata has lower cost compared to proposed approach and cost aware approach. Proposed approach in this paper has the highest total cost in comparison. Q aware approach has high initialization and runtime costs. Finally total cost which is addition of two mentioned costs shows that the approach will not be proper approach compared to learning automata and cost aware approaches whenever the cost measure is considered.

4.3. Third Scenario

In third scenario we address comparing number of scales with the other two approaches. The number of elimination or adding VMs is one of the important factors in dynamic scaling. It affects speed of response in computing environment. Also it can cause to operational overload and imposes cost to the system. Proper management of the measure helps us to achieve minimum cost, increase rate of response, consequently reduction the rate of SLA violation. Overall scaling function calculates total number of scales. The number of scaling functions for four services has been shown in Figure 8. As you can see the number of scaling functions for proposed approach will not change dramatically so that system will have a proper stability.

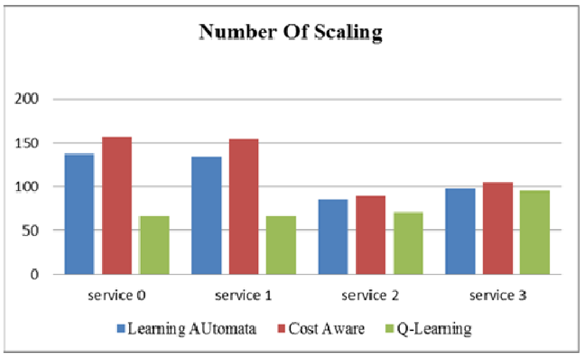

Figure 8. Comparison the number of scaling in three approaches

As it is represented in Figure 9, the number of scaling functions has been decreased in proposed approach compared to two other approaches according to results of simulation. Reduction helps to optimize SLA violation rate, lower cost and higher system stability.

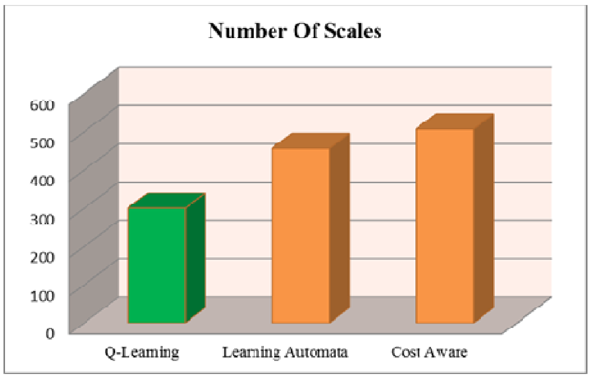

Figure 9. Comparison of overall the number of scaling in three approaches

5. CONCLUSION

Cloud services are distributed infrastructures which extend space of communication and service. The resource providing has been very important because of daily grow of cloud services and scaling issue has been welcomed as one of the most important features of cloud computation. In this paper we have represented an approach based upon reinforcement learning also have addressed Markov model. There are 3 important factors in proposed approaches including SLA violation rate, scaling cost and number of scales. Regarding cost measure, Q aware approach is not proper approach compared to the automata and cost aware approaches. But proposed approach reduces number of scales which leads to optimize rate of SLA violation and system stability. Also proposed approach decreases SLA violation and optimizing SLA leads to increase cost. As a result it makes difficult having minimum cost. On the other hand focusing on the minimum cost leads to SLA violation. So, we can observe substantial reduction in SLA violation and higher system stability by using Q-learning technique in auto scaling.

Therefore it is possible to continue studies about auto-scaling regarded other effective factors and other approaches for example the condition space will be changed according to utilization or we can represent a novice approach in auto-scaling using parallel Q learning and combination of parallel factor and new condition . Also we can apply RL to predicate load in web aware software's. Also it is possible to merge RL and machine learning. Overload in proposed approach should be consider carefully too.

REFERENCES

[1] R. Buyya, *et al.*, "Cloud computing: principles and paradigms," John Wiley & Sons, vol. 87, 2010.
[2] A. Vijaya and V. Neelanarayanan, "A Model Driven Framework for Portable Cloud Services," *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 6(2), pp. 708-716, 2016.
[3] M. G. Arani and M. Shamsi, "An Extended Approach for Efficient Data Storage in Cloud Computing Environment," *International Journal of Computer Network and Information Security*, vol/issue: 7(8), pp. 30, 2015.
[4] M. G. Arani, *et al.*, "An autonomic approach for resource provisioning of cloud services," *Cluster Computing*, pp. 1-20, 2016.
[5] N. Roy, *et al.*, "Efficient autoscaling in the cloud using predictive models for workload forecasting," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on,* pp. 500-507, 2011.
[6] K. Mogouie, *et al.*, "A Novel Approach for Optimization Auto-Scaling in Cloud Computing Environment," *International Journal of Modern Education and Computer Science*, vol/issue: 7(8), pp. 9, 2015.
[7] A. Liu, "Theoretical Analysis for Scale-down-Aware Service Allocation in Cloud Storage Systems," *International Journal of Electrical and Computer Engineering*, vol/issue: 3(1), pp. 21, 2013.
[8] H. Ghiasi and M. G. Arani, "Smart Virtual Machine Placement Using Learning Automata to Reduce Power Consumption in Cloud Data Centers."
[9] R. Hu, *et al.*, "Efficient Resources Provisioning Based on Load Forecasting in Cloud," *The Scientific World Journal*, 2014.
[10] Y. Chevaleyre, *et al.*, "Issues in multiagent resource allocation," *Informatica*, vol/issue: 30(1), 2006.
[11] M. Jacyno, *et al.*, "Understanding decentralised control of resource allocation in a minimal multi-agent system," in *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems,* pp. 208, 2007.
[12] E. Scalas, *et al.*, "Growth and allocation of resources in economics: The agent-based approach," *Physica A: Statistical Mechanics and its Applications*, vol/issue: 370(1), pp. 86-90, 2006.
[13] E. Barrett, *et al.*, "Applying reinforcement learning towards automating resource allocation and application scalability in the cloud," *Concurrency and Computation: Practice and Experience*, vol/issue: 25(12), pp. 1656-1674, 2013.

[14] B. B. G. Abadi and M. G. Arani, "Resource Management of IaaS Providers in Cloud Federation," *International Journal of Grid and Distributed Computing*, vol/issue: 8(5), pp. 327-336, 2015.
[15] F. Bahrpeyma, *et al.*, "An adaptive RL based approach for dynamic resource provisioning in Cloud virtualized data centers," *Computing*, pp. 1-26, 2015.
[16] X. Dutreilh, *et al.*, "Using reinforcement learning for autonomic resource allocation in clouds: Towards a fully automated workflow," in *ICAS 2011, The Seventh International Conference on Autonomic and Autonomous Systems,* pp. 67-74, 2011.
[17] R. M. Bahati and M. Bauer, "Towards adaptive policy-based management," in *Network Operations and Management Symposium (NOMS), 2010 IEEE,* pp. 511-518, 2010.
[18] J. Rao, *et al.*, "VCONF: a reinforcement learning approach to virtual machines auto-configuration," in *Proceedings of the 6th international conference on Autonomic computing,* pp. 137-146, 2009.
[19] M. Amoui, *et al.*, "Adaptive action selection in autonomic software using reinforcement learning," in *Autonomic and Autonomous Systems, 2008. ICAS 2008. Fourth International Conference on,* pp. 175-181, 2008.
[20] R. N. Calheiros, *et al.*, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol/issue: 41(1), pp. 23-50, 2011.
[21] Amazon EC2 instance types, http://aws.Amazon.com/ec2/.
[22] K. Mogouie, *et al.*, "A Novel Approach for Optimization Auto-Scaling in Cloud Computing Environment," *International Journal of Modern Education & Computer Science*, vol/issue: 7(8), pp. 9-16, 2015.

BIOGRAPHIES OF AUTHORS

Bahar Asgari received the B.S.C degree in Information Technology from PNU University, Iran in 2012, and M.S.C degree from Azad University of mahallat, Iran in 2015, respectively. Her research interests include Cloud Computing, Distributed Systems, Big Data and Software Engineering.

Mostafa Ghobaei Arani received the B.S.C degree in Software Engineering from University of Kashan, Iran in 2009, and M.S.C degree from Azad University of Tehran, Iran in 2011, respectively. He is a PhD Candidate in Islamic Azad University, Science and Research Branch, Tehran, Iran. His research interests include Grid Computing, Cloud Computing, Pervasive Computing, Distributed Systems and Software Development.

Sam Jabbehdari currently working as an assistant professor at the department of Computer Engineering in IAU (Islamic Azad University), North Tehran Branch, in Tehran, since 1993. He received his both B.Sc. and M.S. degrees in Electrical Engineering Telecommunication from Khajeh Nasir Toosi University of Technology, and IAU, South Tehran branch in Tehran, Iran, respectively. He was honored Ph.D. degree in Computer Engineering from IAU, Science and Research Branch, Tehran, Iran in 2005. His current research interests are Scheduling, QoS, MANETs, Wireless Sensor Networks and Cloud Computing.