A NEW PCA BASED HIGH DIMENSIONAL REINFORCEMENT LEARNING MULTI AGENT APPROACH FOR TRAFFIC SHAPING IN ROUTERS

[1]M. TAHERI TEHRANI, [2]H. AJORLOO

[1,2]Faculty Member of Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
E-mail: ajorloo@sharif.edu

Abstract- In this paper, the concept of Principle Component Analysis (PCA) is invoked besides reinforcement learning and multi-agent systems to develop a novel intelligent high dimensional reinforcement learning traffic shaper for dynamic and real time allocation of the rate of generation of tokens in a Token Bucket algorithm instead of static allocation of this parameter. This implementation when is compared to our previous work where a simple reinforcement learning traffic shaper was developed, the better and more reasonable utilization of bandwidth and less traffic overload in other parts of the network is more appeared. Indeed, the imposed PCA on the inputs of the reinforcement algorithm gives this ability to the traffic shaping agents to use more vital parameters of the network in their decision process without any concern about exceeding the volume of the calculations and the time. This novel work is also valuable in this aspect that it offers a high dimensional functionality to the reinforcement learning algorithm in the context of multi-agent systems where incrementing dimension is a practical limitation. These methods are implemented in our previous proposed intelligent simulation environment to be able to compare better the performance metrics. The results obtained from this simulation environment show satisfactory behaviors from the aspects of keeping whole dropping probability low while injecting as many packets as possible into the network in order to utilize the available bandwidth as much as possible.

Keywords- High Dimensional, Principle Component Analysis(PCA), Traffic Shaping In Routers.

I. INTRODUCTION

Successful telecommunication requires efficient resource allocation that can be achieved by developing adaptive control policies. Reinforcement learning presents a natural framework for the development of such policies by trial and error in the process of interaction with the environment (Peshkin et al., 2002). In the telecommunication network environments there is a collection of resources and techniques that require such attention in the form of optimal allocation and utilization while there is no precise model for them and if we could to develop such precise models, the complexity of them will lead to impractical control policies.

Therefore, in network environments, reinforcement learning algorithm is the best candidates to derive control policies and efficient utilization due to this realty that it doesn't need a precise model of the environment and act based on direct interactions with the network.

Traffic shaping is one of these areas in the communication networks that can be more considered for utilization of bandwidth and optimal resource allocation. This technique conditions the input stream so that the characteristics are amenable to the scheduling mechanisms to provide the required QoS guarantees.

This article investigates multi-agent reinforcement learning in the context of a concrete approach called PCA analysis to develop intelligent high dimensional traffic shaping agents in the routers and gives this functionality to them to use more vital parameters of the network in their decisions. In practice, Reinforcement learning algorithms have limitations when they are assumed to consider more than one parameter in their framework due to the complexity and exceeding the time of calculations. Even in the case of using one dimensional input parameter, due to the continuous nature of the parameters, there are also some concerns about the volume of calculations through the learning process for real time applications.

In our previous work, a multi-agent reinforcement learning framework were proposed for traffic shaping in the routers in which only in maximum case two parameters, NR-drop in the front of router and Used Buffer Size ratio (UBS) in the router, are used as the decision factors. There we imported these two factors in a two dimensional state plane and then a novel discretization approach was developed to map two dimensional state plane to a finite number of states i.e. state detector. With these designs, not only we imposed a type of discretization nature to the parameters, we reduced the two dimensional factors to a unit discretized parameter i.e. the number of state in which network is working, and then was inserted as input parameter to the reinforcement learning. At that point there was also this open problem that how reinforcement learning will encounter practically with high dimensional inputs when it wants to consider more parameters in its decision process. The idea behind this paper is to invoke PCA analysis on the inputs of the reinforcement learning algorithm to

give high dimensional capacity to this algorithm to consider more parameters without any effect on the performance and time of calculations. Indeed, PCA analysis reduces the dimensions of the input parameters with extraction of the important dynamic parts of them. More precisely, the implementation of this approach is as follows. At first, all the effective and desired parameters that the designer wants to enter in the decision process of reinforcement learning and consequently traffic shaping agent are selected without any limitation and concern. This part shows the power of the proposed approach. The only concern raised may be the checking practical issues in the network that if it is feasible to carry desired parameters to the target router or not. Then PCA is imposed on these parameters and some of the most principal components of them are extracted and are inserted as input to the agent.

The structure of this paper is as follows. In the second and third sections, theoretical framework of the Principal Component Analysis (PCA) and the Reinforcement Learning (RL) issues are briefly discussed respectively. The proposed system model and the way the proposed approaches are implemented in the multi-agent framework is given in the fourth Section. Next, the proposed simulation framework and simulation results are discussed in the fifth section. In Section 6, concluding remarks and some of future developments are presented.

II. PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (PCA) is a mainstay of modern data analysis, a black box that is widely used but sometimes poorly understood. It is a standard tool in modern data analysis, in diverse fields from neuroscience to computer graphics, because it is a simple, non-parametric method for extracting relevant information from confusing data sets. With minimal effort, PCA provides a roadmap for how to reduce a complex data set to a lower dimension to reveal the sometimes hidden, simplified structures that often underlie it. The goal of principal component analysis is to identify the most meaningful basis to re-express a data set (Shlens et al., 2014).

PCA can totally impose three ordered effects on the input vectors set. Here input vector is meant the data produced and observed for each parameter. These input vectors compose the input matrix together. In the first effect, PCA extract principal components of these input vectors by multiplying input matrix with its eigenvectors matrix. As it can be seen in this way of extraction and based on algebraic properties, the extracted principal components are orthogonal and therefore are a potential new basis set for re-expressing the original data. In the second stage, the main extracted components are arranged in order of importance, so that the components of the bigger changes come first and gives a view to designer overall the input data. Finally, parameters that have little role in changing and dynamic data is deleted.

When PCA is used, the standard approach is that the input vectors are normalized, Meaning that they have zero mean and unit variance to them. One of the parameters PCA method considers is the minimum fraction variance method. This parameter identifies those components that had less than this parameter role in the dynamic of input vectors and they should be deleted. For example, if this parameter is 1%, PCA eliminates those components that have less than 1% of the total variance contribution on the data set.

The output of PCA is expressed by two lower dimensional matrixes named scores and coef and an importance vector named latent. These matrixes show template and format of the original data. Matrix scores, shows the principal components derived from the original data on the original space and matrix coef, shows the coefficients of principal components.

In fact this matrix shows the eigenvectors of the covariance matrix of the original data that are put in order of importance in the columns of the matrix. Finally, the importance percentage of these eigenvectors or in other words the importance percentages of principal components are put in order in latent vector. This vector has an essential role in the designing.

III. REINFORCEMENT LEARNING (RL)

Reinforcement learning has gained attention and extensive study in recent years as a learning method that does not need a model of its environment and can be used online. Reinforcement learning is well suited for multi-agent systems, where agents know little about other agents, and the environment changes during the learning (Junling et al., 1998).

Modern reinforcement learning research uses the formal framework of Markov Decision Processes (MDPs). In this framework, the agent and environment interact in a sequence of discrete time steps, t = 0, 1, 2, 3… and on each step, the agent perceives the environment to be in a state s, and selects an action a. In response, the environment makes a stochastic transition to a new state and emits a numerical reward r which is in [0, 1]. The agent seeks to maximize the reward it receives in the long run. . For example, the most common objective is to choose each action a so as to maximize the expected discounted return,

$$E(\sum_{t=0}^{\infty} \gamma^t r_t) \qquad (1)$$

Where γ is a discount-rate parameter, $0 \le \gamma \le 1$

The simplest reinforcement learning algorithms apply directly to the agent's experience interacting with the environment and change the policy in real time. For example, Watkins' Q-Learning algorithm (Watkins et al., 1992), one of the simplest reinforcement learning algorithms, uses the experience of each state transition to update each element of a table. This table, denoted Q, has an entry Q(s, a) for each pair of state s and action a. Upon the transition from on state to the next state, having taken action a and received reward r, this algorithm performs the update as below where indexes as t+1 show transition to the next step:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_t) - Q(s_t, a_t)] \quad [2]$$

IV. THE SYSTEM MODEL

In this section, the system model is presented in two parts. In the first part, the structure of traffic shaping agent and the implementation of RL is presented and in the second part, the developed PCA based approach is presented specifically how it can be implemented besides RL algorithm. Before developing the models, it is emphasized that these approaches are implemented at Source Routers (SRs) and therefore, any parameter used in the framework should have this feasibility to be carried to these routers. In the proposed multi-agent framework, routers are categorized into two main groups of Network Routers (NRs) and Source Routers (SRs). The SRs are those routers that are connected to end nodes and NRs are those that are indirectly connected to any end node and act as connections between subnets (Fig. 1).

In the traffic shaping framework, each port of SR is characterized with a token bucket with buffer length of b (bits) and token generation rate of g (bits/s). In this case the source can only inject a complete data packet into the network if there are enough tokens for a complete packet transmission, and tokens are discarded if the token bucket overflows (Radhakrishnan et al., 1996). In case of burst generation of packets, if the total number of bits is larger than of tokens, then only the first few packets whose lengths are smaller or equal to the total number of tokens can pass, and the others should wait for the time that enough tokens are generated for them. The NRs are responsible for generating information for the SRs that they can adjust their parameters in line with the goal of minimization of the packet loss probability in the network. To achieve this goal, the parameters of the token bucket should be chosen in a way to make the loss as small as possible. Therefore a flexible mechanism for choosing the token bucket parameters g is desirable. To do so, an intelligent system is designed which learns the best g for the token bucket in each state of the network. In this scheme, the action a, determines g of each SR, and can take a value between 0 and 1. The value of g is related to a as,

$$g = a \times g_{\max} \quad (3)$$

where $g_{\max}$ is determined by

$$g_{\max} = w_j \quad (4)$$

with w defined as the bandwidth of the medium connected to the jth port of the router in which the traffic to be shaped. The input parameters to the algorithm that will be the basis of decisions of the agent for determining its action a, are those factors that have effect on the traffic shaping mechanism and are our target factors to be controlled. In fact these parameters are those target factors that agent should consider for utilizing the bandwidth. For example, in our previous work only two factors were considered, the packet drop in the front NR (NR-drop) and UBS ratio at SR where the multi-agent framework is placed.

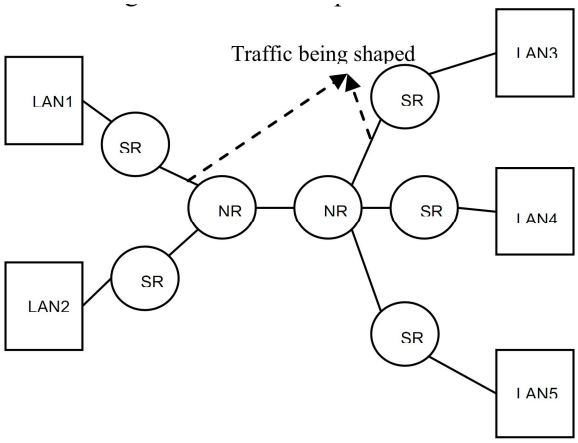


Fig. 1: The network model under consideration

The core of the developed reinforcement learning agent is a state plane which implements a novel discretization approach and detects the state of the environment under control. This state detector is a two dimensional plane and takes only two input parameters and based on the designed discretization map identifies the state of the environment. What is feed to the algorithm is the number of this state which shows the status of the network if it is in god conditions or not. This novel approach not only gives this opportunity to use two input parameters directly but also gives finally a discretized number to the RL algorithm. For example in our previous work the NR-drop and UBS are used (Fig.3).

To show how the state plane is configured and how the discretization process works to produce a finite number of the states which will be used as input parameter to the agent; the calculations will be presented here. For this purpose, the mentioned input parameters from the previous work will be

considered. It should be noted that in general, any parameter can be replaced. The two parameters used were packet dropping percentage sensed by port i of SR at time t, namely

$p_{t,i}$ or NR-drop and UBS to maximum buffer size ratio at the i_{th} sending port of SR connecting to m_{th} NR, namely $b_{t,im}$. The reward r_{t+1} is determined by how effective was the action *a*, at time *t* in changing the state from a worse one to a better one. The reward takes a value in [0, 1]. This procedure is discussed in the following sections. Fig. 2 depicts a block diagram of the proposed system.

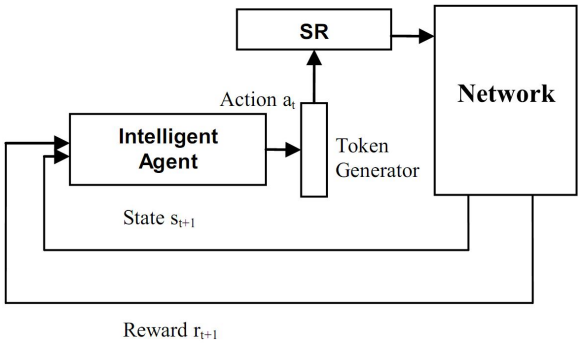


Fig. 2: Block diagram of the proposed system for one traffic shaping agent.

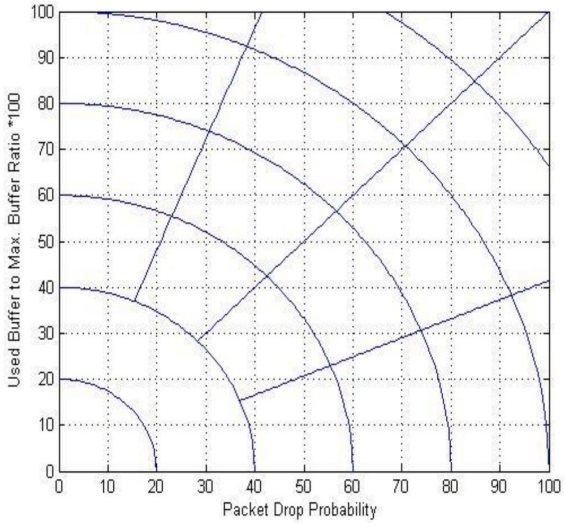


Fig. 3: State plane of the network.

To calculate the desired input parameters here, NR-drop and UBS, it should be noted calculation of the latter is rather straightforward but for the former some problems arise. First how one can measure the parameter value, and second, after measurement how one can transfer this feedback information to the place that it can be used. To answer the first problem it is assumed that the network support the Explicit Congestion Notification (ECN) mechanism which has been proposed as a solution for conveying the congestion signalling rapidly and explicitly to TCP senders. This mechanism is utilized for estimating NR-drop at SR, because the ECN mechanism (Ramakrishnan et al., 1999) marks packets instead of dropping them as a means of signalling congestion. This mechanism marks packets as dropped with a probability proportional to network congestion. Because the feedback information consists of only two one-bit flags, this does not create a great deal of overhead. The port i of the SR router checks the ACK packets and counts how many of them are marked and calculates p_{ti} at the end of each duty cycle of the network,

$$p_{t,i} = \frac{Number \quad of \quad marked \quad packets}{Total \quad number \quad of \quad packets} \tag{5}$$

Having these two parameters and feeding them into the state detector, the current state of the network can be determined. In Fig. 3 can be seen that the state plane of the network is discretized into 20 states.

To calculate reward one may follow the simple rule

$$r_{t+1} = \begin{cases} \dfrac{d_t - d_{t+1}}{d_t} & d_{t+1} < d_t \\ 0 & d_{t+1} \geq d_t \end{cases} \tag{6}$$

Where *d* is calculated as below:

$$d_t = \sqrt{p_{t,i}^2 + b_{t,im}^2} \tag{7}$$

It can be seen in this formulation that the aim of learned agents is to keep used buffer size at SR and the rate of packet drop at NR low. It can be clearer if you notice to Fig.3 where d is the radius of the state plane and the goal of agent is to move near origin of the state plane where the UBS and NR-drop become zero. These calculations all will complete the design of state plane and discretization goals. In this part, that's the time to develop the domain of the considerations of traffic shaping agent and the PCA approach be invoked. Here, PCA framework is placed before the state plane which accepts only two parameters as input. In other words by placing PCA framework before state detector, this capacity is created that any number of relevant parameters that can have any effect on the dynamic of traffic shaping be considered. PCA framework takes these input parameters and after extracting principal components of the data set, delivers the most effective and the essence of our data to the state detector. As the state detector only accepts two parameters, PCA framework is set to deliver only two of the most important components to the state detector. Since the most of dynamics of the inputs are summarized in the first principal component, this assumption doesn't effect on the precision. The state detector after importing these two parameters onto its state plane performs the discretization process like before and identifies the state in which the network is placed and finally inserts this unit discretized number to the traffic shaping agent. To enable PCA implementation approach to do its job well along with the

reinforcement learning algorithm and in the early stages of learning, all the parameters are not applied to the learning process and only two of them, for example NR-drop and UBS, are applied and after lasting some periods of learning phase and until the volume of measured data for all the parameters is enough, PCA algorithm is applied on all the parameters and coefficients matrix are stored. After this stage and in each next step, all parameters are applied to the learning process and PCA will be performed and only two principal components are delivered to the agent as input parameters.

Also for PCA coefficient matrix to be updated, various policies dependent on the context of the system and the desired efficiency can be considered. It can be updated after the first time it is made and in each next step with the arrival of new data for each parameter or that it frequently occurs after every few steps.

V. SIMULATION

5.1 SIMULATION ENVIRONMENT

To achieve the goal of designing a precise model of the system capable of simulating real dynamic behaviour of the real network systems, It is assumed that mediums, LANs, routers and packets as entities that are existed in a large high speed networks. Each unit contains its own essential sub-units and has its own responsibilities, behaviours, parameters, limitations, etc. For example, routers contains different number of ports where each port has its own buffer size, and token generation unit with responsibilities such as routing, sending packets, receiving packets, etc. Taking into consideration the resource restrictions, it is tried to split each unit's behaviour into separate modules and define them in functions which are as much independent as it is possible This independency of modules with essential interactions among them provides the user with the chance to change the specific behaviours of each unit only by redesigning the desired module and without the need to redesign the whole model. This feature helps simulate desired network under different circumstances such as different routing algorithms, different traffic shaping policies and other aspects in cycles the simulations are executed. Cycle times are depicted in the results. Besides, it gives the chance to have great access to desired details of each unit and modules. For example we can develop our new multi-agent framework in this simulation environment and compare it with many other approaches.

5.2 SIMULATION RESULTS

In this section, the evaluation of proposed approach is provided. This evaluation is done in the simulation environment that was briefly explained in the previous section. To perform the simulations, it is supposed that a SR with four ports is shaping traffic towards NRs. Therefore, a multi-agent framework is placed on the SR, in which for each port, one agent will be considered. Also there is a variation of input flow rate of packets arriving to the router to expose it to the diverse specifications of real traffics. The simulation circumstances and scenarios are set in a way to be able to compare the results yielded here with our previous work where we had a simple traffic shaping agent.

At SR, for traffic shaping agents, it is important to shape the traffic at a maximum possible rate till no drop occurs at NRs. For this purpose, agents learn through RL algorithm to shape traffic based on the measurements of this parameter and any other effective parameter desired to enter the decisions. In this section although we have this power to select any parameter for entering to traffic shaping process due to the power of PCA, we consider only three parameters, NR- drop in the front of SR port, Used Buffer Size ration in SR port (UBS) and SR-drop we experience in the port of SR which also can have an effective role in our algorithm. As seen here in comparison to our previous work, we incremented the dimensions of the input parameters to our traffic shaping agent without any concern about the volume of calculations in reinforcement learning process. Also it can be noted that the added parameter, SR-drop is a local parameter in SR and therefore has practical feasibility to be used. In the following, firstly, the results of the simple traffic shaping multi-agent framework from our previous work will be presented and then our high dimensional PCA approach will be simulated and evaluated.

In our previous work, traffic shaping was done with RL algorithm and agents had a fixed and predefined buffer size. These agents didn't regard to the SR-drop and they only tried to shape the traffic towards NR with respect to the rate of drop at NR and the ratio of Used Buffer Size at SR. As seen in Fig. (4), for a variation of the input flow rate at one of the ports of SR i.e. percentage of input bandwidth which is in this simulation 10MB, the variations in the other parameters are depicted. In this figure, input flow rate has a decrease at initial cycles and two increases at final cycles. This type of variations is applied to SR to reveal dynamics of the approach. It is seen in the figure that the agent for this port has initially utilized 100% of its buffer memory and the amount of its used buffer size at the whole cycles is high also, the amounts of packets shaped from SR to NR is in a level that keeps NR-drops to zero in except for some initial cycles. This means that what is important for this framework is minimum drop at NR.

In the evaluation of the PCA based multi-agent framework, besides NR-DP and UBS, a third parameter, SR-Drop in the SR port, will be

considered. As previously mentioned, for PCA technique to do its job well, and the data obtained from the process be enough, in the early stages of learning, each agent uses of the two-parameters, UBS and NR-DP, and the third parameter, SR-Drop does not apply to it. In fact, at these early stages of the learning, the process acts like our previous work. At 1000th cycle of learning that the data obtained from the measurement of three parameters, UBS, NR-DP and SR-DP is sufficient, PCA technique is applied using MATLAB software on the formed input matrix of orders 1000×3 and its output matrixes yielded are stored.

After this point, PCA will work in its normal status and will take all of three parameters and will deliver two of its principal components to the agent to use as its states in the state plane.

Here, the state plane is completely like what was used in the previous section and only change is on the name of states, here instead of using the original two dimensional parameters directly, we used two principal components of a bigger number of parameters. It should be noted that because of the long duration of the learning process which is in average 2000000 cycles, the first 1000 cycles that was used without PCA technique is really nothing against it. The latent vector that indicates the amount of information available on the main components or in other words the importance percentage of components is given in Table.1. As seen in this table, most of the information is summarized in the first and second components which are respectively 84.95% and 14.70%. During cycles of 2000 and 3000 of learning process, the PCA coefficients matrix is again updated and latent vectors are given in the tables 2 and 3 respectively.

These results show in practice, for this model and with advancement of learning cycles, no significant change is seen in the percentage of data in the two first principal components i.e. the SR-DP, NR-DP and UBS have not much change in dynamic. This important issue shows that PCA components and coefficients don't have significant change from the first obtained coefficients in the 1000th cycle and therefore it's not needed to update them again. Another important point is that in comparison to the previous work the number of learning cycles have been increased.

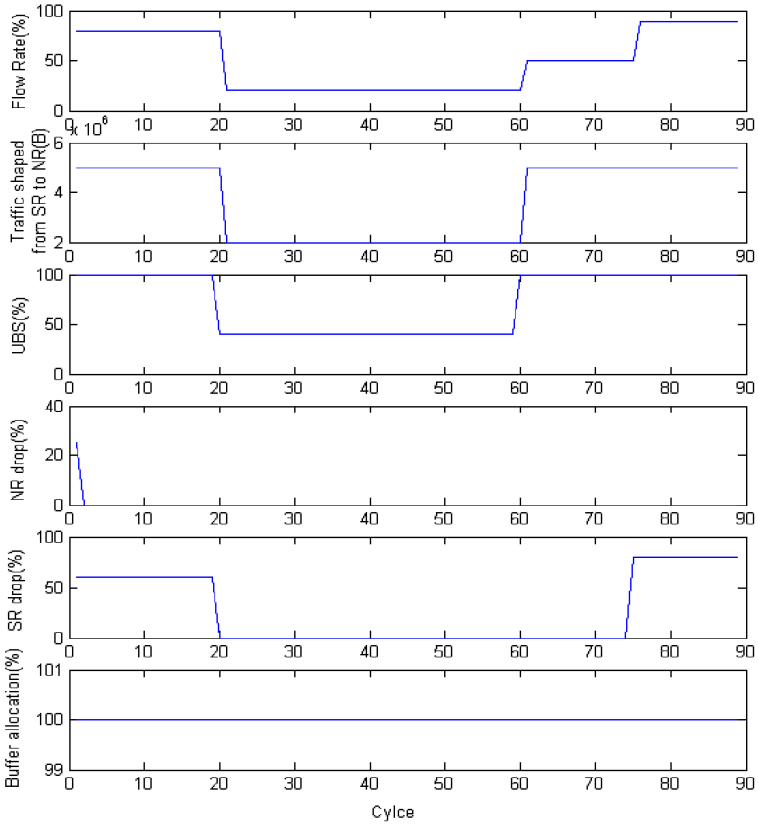


Fig. 4. Simple traffic shaping framework without PCA approach: it works based only on two input parameters.

The results of this approach and for the first port of the router are given in the Fig. 5. In comparison with the results of the corresponding port in the previous section that the traffic shaping agent used directly only two parameters and didn't use PCA approach, it can be seen that in this case the transmission rate from SR towards NR is increased and the drop in SR i.e. SR-drop is decreased. In other words, by adding the SR-drop, the traffic shaping agent has tried to send at a higher rate to decrease the drop rate at its buffer while it still considers drop at NR i.e. NR-drop as a vital decision factor and doesn't permit to

increase and at most of times keep it in zero levels. At this point we can see the power of PCA approach especially in comparison to the previous section; the bandwidth is more utilized while the overall drop in the network is decreased. With this developed framework, another one can consider more relevant and effective parameters without any concern for complexity and absolutely the performance may be more improved.

Principle Components	Eigenvalue	Variance %
1st Component	1746.50	84.95%
2nd Component	302.40	14.70%
3rd Component	6.90	0.33%

Table 1: the results of latent vector in 1000th cycle of learning.

Principle Components	Eigenvalue	Variance %
1st Component	1730.70	85.33%
2nd Component	291.60	14.37%
3rd Component	5.90	0.29%

Table2: the results of latent vector in 2000th cycle of learning.

Principle Components	Eigenvalue	Variance %
1st Component	1729.1	85.37%
2nd Component	290.40	14.33%
3rd Component	5.70	0.28%

Table3: the results of latent vector in 3000th cycle of learning.

CONCLUSION

In this paper some of the most important realities about the nature of the computer networks were addressed and in accordance with them concrete approaches were developed. First of all, the distributed nature of the networks was addresses and in this regard the multi-agent frameworks were proposed which can satisfy the needed circumstances and provide a good scalability factor for the networks.

Secondly, complexity of the networks were addressed and with having in mind this issue that precise models for the networks hardly can be achieved and if achieved have practically limitations to be implemented, the intelligent RL approach was proposed as the main mechanism for reaching to the desired control targets. RL algorithms have no need to a precise model of the network and by direct interaction with the environment and by trial and error try to learn the dynamic of the network and achieve the best policy for control of the network. Finally another important reality was addressed, the diverse nature of the network.

When it comes to control the networks, a diverse and high dimensional parameters will be raised to consider which practically due to the volume of calculations and saving the time is a limitation. This issue is a harder problem in the case of RL algorithm as previously said. In this regard PCA were proposed which can be performed on the high dimensional input parameters and extract the principal normalized components for our undergoing algorithm.

As seen in this paper, these approaches along with each other gave this opportunity to us to utilize efficiently the bandwidth while keeping the whole drop in the network low.

As future works, the PCA reinforcement learning approach developed here can be used in any other part of the networks without any limitation existed in previous works and with importing any number of input parameters to the agent, the target environment is controlled. Absolutely these frameworks can be put in practice in other complex environments to control needed mechanisms.

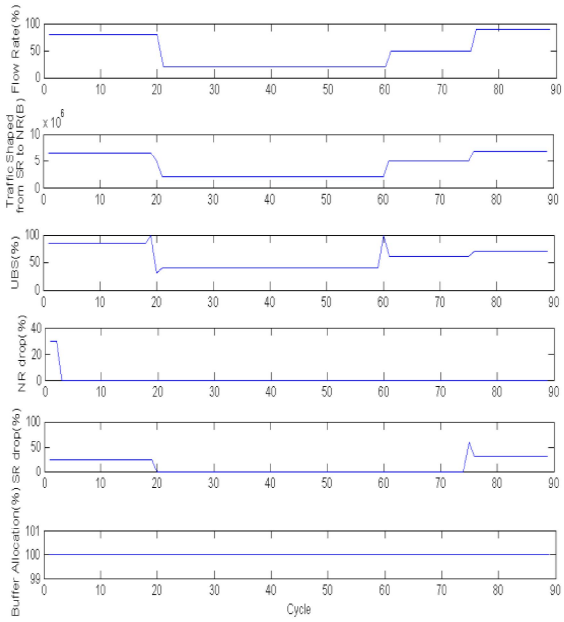


Fig. 5. Applying PCA to the traffic shaping agent: The effect of adding more parameters to the mechanism of agent.

REFERENCES

[1]. Peshkin, Leonid, Virginia Savova, "Reinforcement Learning for Adaptive Routing", International Joint Conference on Neural Networks (IJCNN),2002.

[2]. Taheri Tehrani M., A. A Safavi , M.R. Kharazmi ,Javad Arefi, " A New Multi Agent Approach for Traffic Shaping and Buffer Allocation in Routers", UKACC Control Conference, Manchester, UK, 2008

[3]. Shlens J., "A Tutorial on Principle Component Analysis", Version 3.02, USA, 2014.

[4]. Najmaei N., M. Zamani, I. Shames, S. A. A. Safavi, "An Object Oriented Approach to Simulation of High Speed Networks", Technical Report, Dept. of Electrical Engineering, Shiraz University, Fall 2007.

[5]. Junling Hu, Michael P. Wellman, "Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm", Proc. 15th International Conf. on Machine Learning, 1998.

[6]. Watkins, C. J. C. H., and Dayan, P. Q-learning. Machine Learning, 8:279-292, 1992.

[7]. Ramakrishnan K. and S. Floyd, "A proposal to add explicit congestion notification (ECN) to IP." RFC 2481, 1999.

[8]. Vila, p , J. L. Marzo, R. Fabregat, D. Harle, "A Multi-agent Approach to Dynamic Virtual Path Management in ATM Networks", 2002.

[9]. Radhakrishnan. S, S.V. Raghavan, Ashok K. Agrawala, "Flexible Traffic Shaper for High Speed Networks: Design and Comparative Study with Leaky Bucket", Computer Networks and ISDN Systems 28, 453-469, 1996.

[10]. Park, Eun-Chan Park and Chong-Ho Choi, "Adaptive Token Bucket Algorithm for Fair Bandwidth Allocation in DiffServ Networks", School of Electrical Engineering and Computer Science, Seoul National University, Seoul, KOREA, csl.snu.ac.kr/publication, 2004.

★★★