The 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'07)

REINFORCEMENT LEARNING FOR ACTIVE QUEUE MANAGEMENT IN MOBILE ALL-IP NETWORKS

Nemanja Vučević, Jordi Pérez-Romero, Oriol Sallent, Ramon Agustí
Dept. TSC, Universitat Politècnica de Catalunya (UPC)
Barcelona, Spain

ABSTRACT

In future all-IP based wireless networks, like the envisaged in the Long Term Evolution (LTE) architectures for future systems, network providers will have to deal with large traffic volumes with different QoS requirements. In order to increase exploitation of network resources wisely, intelligent adaptive solutions for class based traffic regulation are needed. In particular, Active Queue Management (AQM) is regarded as one of these solutions to provide low queuing delay and high throughput to flows by smart packet discarding.

In this paper, we propose a novel AQM solution for future all-IP networks based on a reinforcement learning scheme that allows controlling both the queuing delay and the packet loss of the different service classes. The proposed approach is evaluated through simulations and compared against other algorithms used in the literature, like the Random Early Detection (RED) and the Drop From Tail (DFT), confirming the benefits of the proposed algorithm.

I. INTRODUCTION

With the aim to provide seamless mobility and service access, mobile communications are nowadays directed towards all-IP network solutions. In that sense, 3GPP is currently standardising the requirements for Evolution of current 3G systems (up to and including Release 6) to all-IP Network system from Release 7 [1]. The expectations are the integration of existing 3GPP access systems with future enhancements and other wireless (WLAN, WiMAX, etc.) and wired access technologies (xDSL, Cable, etc.). Independence of the layer 2 in all-IP enables session connectivity across multiple access systems, with faster and cheaper incorporation of new technologies and services.

Merging of communications technologies is expected to introduce huge amount of IP traffic that should be handled by operators in the different segments of the all-IP network. Therefore, new challenges are to be faced in order to provide end-to-end quality of service (QoS) to users with different types of services, while using network resources efficiently. This will avoid the over provision in the different segments with the consequent capital expenditure saving for network operators.

Differentiated services (DiffServ) [2], appear as a possible solution to QoS provision in next generation all-IP networks. Traffic of the same class is classified under one flow and treated as one. Priority in network assignment depends on the flow's class denoted as per-hop-behaviour (PHB). Two PHB groups are recognized for DiffServ: expedited forwarding (EF) [3] and assured forwarding (AF) [4]. EF is characterised by low loss, latency and jitter. For the AF class bandwidth and jitter guarantees are expected to have lower latency demands than for the EF class. In turn, DiffServ can have different types of AF classes with different levels of drop precedence of IP packets. On the other hand, traffic with neither bandwidth nor high delay restriction is identified as best effort (BE) traffic. Usual mapping sorts VoIP users in EF, video streaming users in AF and www browsing, mms, email, ftp users in BE class.

The queue management is a well known issue in IP networks with best effort traffic [5]. It already plays a role in commercial core networks in order to increase network efficiency and support QoS. The current solutions are, however, oriented to network over-dimensioning and absolute precedence of prioritized classes with respect to best effort service to avoid buffer congestion. In presence of congestion or overload situations simple tail drop from limited buffers is applied, usually only for BE class [6].

Nevertheless, the voice service, as legacy service of the mobile communications systems, is still dominant in vast aspects of mobile networks. Migration of conversational users to PS (packet switched) networks (VoIP) will lead to IP domains with a significant fraction of EF users. In these environments, active queue management becomes even more sophisticated and improvements for class based queues might contribute to increase network efficiency.

In this paper, a novel intelligent active queue management algorithm is proposed, the Reinforcement Learning-Queuing Delay Limitation (RL-QDL). It enables adaptive queue management and copes with the QoS requirements of the different DiffServ classes in terms of both packet dropping ratio and end-to-end delay. The algorithm will be evaluated by means of simulations and compared against other proposals existing in the literature, like the Random Early Detection (RED) and the Drop From Tail (DFT), revealing a better behaviour to improve resource utilisation and to adapt to the varying traffic conditions.

The rest of this paper is organized as follows. In section II, end-to-end QoS in all-IP is described. A review of the different baseline solutions to the queue management problem and its place in all-IP networks is given in section III. This is followed by our algorithm described in section IV. In section V modelling scenario is described, and results are compared. In the end, conclusions are summarised in section VI.

II. QOS IN ALL-IP NETWORKS

While providing advanced services and seamless mobility independently of the access technology, main objectives of all-IP networks include end-to-end QoS and security guarantees. However, it is not easy to predict the IP traffic model for this network, while at the same time QoS should not be degraded. As examples for possible methods to achieve this, intelligent routing and dynamical load control mechanisms are usually mentioned [1].

1-4244-1144-0/07/$25.00 ©2007 IEEE.

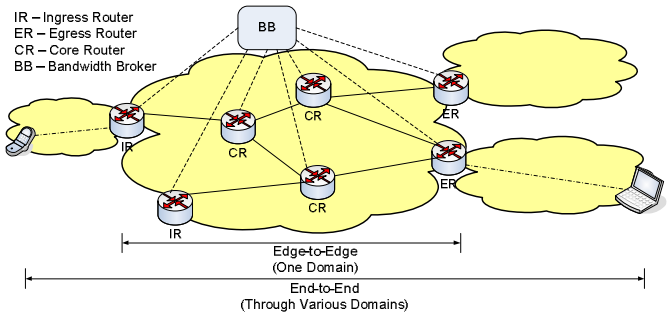


Figure 1: Edge-to-edge QoS support.

On its way, in the all-IP network, traffic is passing through various domains, as illustrated in Figure 1. One domain might correspond to the network provisioned by one service operator, including the radio access part with one or more technologies and the core network. The Ingress Router (IR) is the first router to receive the packet in one domain and the Egress Router (ER) is the one to send it out of it. Intermediate routers are denoted as Core Routers (CR). The called party or end server might be in another domain that can either be a direct neighbour of the first one or be located several domain hops away. Each domain needs to meet specific edge-to-edge QoS guarantees predefined for it, between ingress and egress router. This should be carried out following previous negotiation, in order to support the desired end-to-end QoS. Inside one domain the bandwidth broker (BB) is managing the resource handling and QoS provisioning, depending on the accomplishment of QoS expectations in egress routers.

III. Active Queue Management in All-IP Networks

The router model considered here is shown in Figure 2. It assumes class based queuing (CBQ), with support for three different classes, namely EF, one AF and BE. The entering traffic is directed in the corresponding buffer by the aggregate classifier. Queue management is responsible for packet behaviour inside the buffer. In turn, the scheduler is in charge of link resource assignment by determining the class whose packet will be transmitted next. For a given class the packets of the buffer are served following a first input first output (FIFO) policy.

The need for a proper queue management strategy emerges from the queuing behaviour itself. Packet buffering restrains data loss; however, queuing directly introduces delay into the system. Therefore, buffers should be related to the maximum delay the system is permitted to introduce. At the same time, queues should be able to absorb the data bursts arising from the IP traffic variability. Consequently, the queue limits should also be related with the bursts size.

Under high traffic conditions, just by limiting the buffer size, the accumulation of the queues will continue and, while accumulating delay in flows, at the same time the ability of absorbing incoming data bursts will be reduced. Therefore AQM strategies are introduced to control which packets should be dropped from the queue.

In all-IP networks AQM becomes a tool to support QoS under extreme conditions and, together with dynamic routing and scheduling, it takes care of traffic handling. Typically, AQM has its place as a mechanism for congestion control in best effort networks [7]. However, AQM methods for CBQ have been rarely studied, and no dominant solution has been established. Therefore, the algorithm proposed in this paper will be compared with two basic queue management solutions that are detailed in the following.

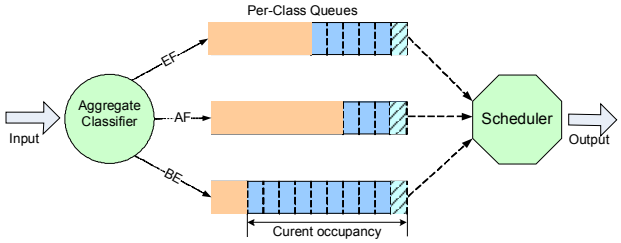


Figure 2: Router model.

A. Drop From Tail (DFT) algorithm

This baseline algorithm consists in a simple limitation of the queue lengths [8]. In that case, every packet that encounters full queue on arrival will be dropped. Though simple, this method is still a usual solution in queue management. In our case, the queue size will be specific for each class.

B. Random Early Detection (RED) algorithm

The IETF RFC recommendation on active queue management [5] is to use RED [9] as the default queue management algorithm unless otherwise needed by the operator. In order to face overload situations RED algorithm drops packets by assigning to each incoming packet a certain dropping probability depending on the queue occupancy N (i.e. the number of packets in the buffer) at the time of the packet arrival. For this purpose two thresholds for queue occupancy N_{MIN} and N_{MAX}, and a maximum drop probability p_{MAX} are defined. Then, the packet dropping probability in RED is calculated as:

$$p_N = \begin{cases} 0 & N \le N_{MIN} \\ p_{MAX} \cdot \dfrac{avr - N_{MIN}}{N_{MAX} - N_{MIN}} & N_{MIN} < N < N_{MAX} \\ 1 & N_{MAX} \le N \end{cases} \quad (1)$$

Where avr is the estimate of the average queue size calculated with weight factor $\alpha (0<\alpha<1)$:

$$avr = avr \cdot (1-\alpha) + N \cdot \alpha \quad (2)$$

Although this algorithm was originally developed for single buffer case, in our study RED will be applied independently to the buffer of each of the classes, like in [10].

IV. Proposed AQM Solution, RL-QDL

We propose an AQM mechanism whose objective is the preservation of QoS when the network is highly loaded. Let's suppose that QoS specifications are expressed in terms of delay, loss and throughput requirements inside the domain. QoS violation occurs whenever the traffic does not meet the specific QoS demands (e.g. when a packet exceeds a maximum delay, when a packet is lost or when the throughput is below a specific limit).

The AQM algorithm will define specific delay thresholds, denoted as QDL (queuing delay limits) for each class, so the packets whose delay is longer than the threshold will be

dropped. The packet check for discarding is carried out from the front of the queue before next packet is to be scheduled, as shown in Figure 3(a). Determination of appropriate values for the QDL thresholds can depend on traffic characteristics, system state, and QoS demands. With aim to have adaptable mechanism that may deal with all the previous issues, a reinforcement learning (RL) [11] algorithm is introduced to determine the QDL, forming the RL-QDL AQM mechanism.

In this study we consider RL agent that explores continuous state and action space using Gaussian unit search behaviour. In previous work, Hui and Tham have applied RL approach like this on dynamic adjustment of weights in packet schedulers [12]. Actually, in our work, to demonstrate RL-QDL algorithm's benefits, we base simulations on their network topology and QoS indicators. However, our approach to the network's adaptability, dealing with revenue increasing problem, is completely different. To improve system performances, we apply RL to dynamic adjustment of AQM parameters while leaving packet scheduler static.

As illustrated in Figure 3(b), each RL agent in the router computes a reward function that reflects the QoS fulfilment by the different packets in the network. It is computed using information from the same router (throughput, loss) and from egress routers (delay and rate violation for traffic passed through that router) that is received from the BB. Reward for router *i* is therefore calculated as:

$$r_i = \sum_j (c_j \cdot t_{i,j} - p_{loss,j} \cdot l_{i,j} - p_{dly,j} \cdot d_{i,j} - p_{thr,j} \cdot th_{i,j}) \quad (3)$$

Index *j* stands for traffic class, parameter c_j is the charge per bit for the traffic successfully carried in that class, and $t_{i,j}$ is the amount of traffic (bits) passed through router *i* in the same class. Parameters $p_{loss,j}$, $p_{dly,j}$ are penalizations per packet, for the number of packets lost in transport ($l_{i,j}$) and for those not meeting delay requirements ($d_{i,j}$), respectively. Parameter $p_{thr,j}$ is penalization per each activity interval of a traffic source in which throughput requirements are not accomplished ($th_{i,j}$).

Reward is accumulated in time by exponential averaging, giving higher influence to most recent reward samples:

$$\hat{r}_n = \gamma \cdot r_{n-1} + (1-\gamma) \cdot \hat{r}_{n-1} \quad (4)$$

where *n* is the number of current time step, r_{n-1} is the past reward value, $\hat{r}_n$ is the baseline rewards accumulate, and the weight of exponential averaging constant is γ, $0<\gamma<1$.

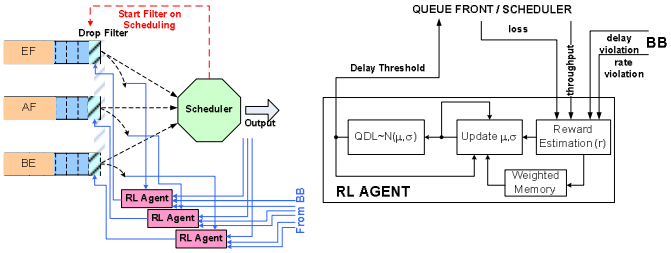
During exploration/evaluation cycle, for each class and in each of the routers (see Figure 3(b)), the RL-QDL AQM algorithm computes, in each time step *n*, the QDL thresholds. For the *j*-th class in time step *n* the value of the delay threshold is denoted as $QDL_{n,j}$ and is obtained as a Gaussian random variable with average $\mu_{n,j}$ and standard deviation $\sigma_{n,j}$.

$$QDL_{n,j} = N(\mu_{n,j}, \sigma_{n,j}) \quad (5)$$

The average and standard deviation are updated in each time step, according to the measured reward function as:

$$\mu_{n+1,j} = \mu_{n,j} + \alpha_{\mu,j} \cdot (r_n - \hat{r}_n) \cdot (QDL_{n,j} - \mu_{n,j}) \quad (6)$$

$$\sigma_{n+1,j} = \sigma_{n,j} + \alpha_{\sigma,j} \cdot (r_n - \hat{r}_n) \cdot \frac{(QDL_{n,j} - \mu_{n,j})^2 - \sigma_{n,j}^2}{\sigma_{n,j}} \quad (7)$$



(a) (b)

Figure 3: (a) Router with RL-QDL AQM solution. (b) RL agent model.

Parameters $\alpha_{\mu,j}$ and $\alpha_{\sigma,j}$ are RL adaptive rate constants for μ and σ, respectively. As said before, evaluation relies on the search behaviour of the Gaussian unit, based on reaction to previous evaluation. After the change of *QDL*, if the average reward has increased, the mean for next evaluation (μ_{n+1}) will follow the previous change direction and in the case of reward degradation it will follow the opposite direction. The update of σ aims at narrowing or broadening the search around μ depending on the suitability of the different obtained parameters. For details the reader is referred to [13].

V. Simulation Model

The simulation model considers the network structure shown in Figure 4, which is based on [12]. There are 8 traffic sources (S0-S7), 6 routers (R1-R6), 8 receivers (D0-D7) and a bandwidth broker (BB). All the receivers are receiving traffic from one (corresponding) source (Sx-Dx,x=0,..,7). BB is passing the information on QoS violation from egress routers (R5, R6) to all the routers in the network. Each source is a traffic generator with a specific class and behaviour. The capacities of each link are indicated in the figure.

Sources from S4-S7 have the same characteristics as sources from S0-S3, respectively. Each source Sx generates sessions according to an exponential session interarrival time and a constant session duration. Characteristics of each source are given in Table 2. The specific session interarrival time is varied depending on the total load to be simulated. The average ON and OFF interval duration for the ON/OFF sessions is 0.5s with an exponential distribution for EF, AF and BE(S1) traffic and a Pareto distribution for BE(S3) traffic. During the ON period, packets with constant length (125 bytes) are generated with exponential interarrival time.

All the routers implement weighted fair queuing (WFQ) [14] scheduling with static weights. The simulations were repeated for two different weight proportions in schedulers W_{EF}:W_{AF}:W_{BE}=1:1:1 and W_{EF}:W_{AF}:W_{BE}=3:2:1.

To measure the degree of QoS fulfilment, the same function (3) is used to compute the reward of the overall network. This time throughput and all the penalties are calculated on an edge-to-edge basis, for the entire domain (i.e. throughput, delay and rate penalties are measured in egress routers, while the packet loss is obtained from all the routers).

Table 1: Parameters of the reward function.

Class (j)	c_j	$p_{loss,j}$	$p_{dly,j}$	$p_{thr,j}$
EF	0.0001	0.2	0.2	-
AF	0.00004	0.08	0.04	10
BE	0.00001	0.02	-	-

Table 2: Source characteristics.

Source	Class	Source type	Rate per session in the ON period (kb/s)
S0, S4	EF	ON/OFF	64
S1, S5	BE	Always ON	64
S2, S6	AF	ON/OFF	300
S3, S7	BE	ON/OFF	128

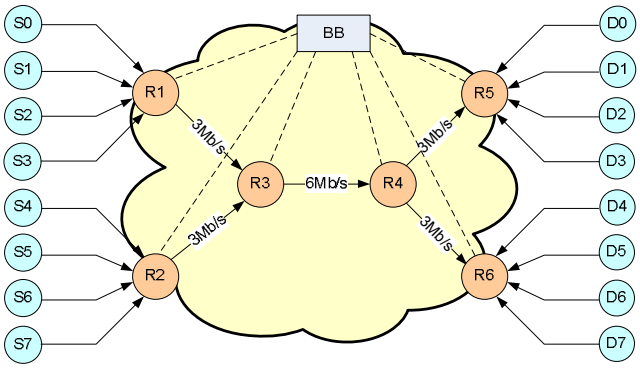


Figure 4: Network model.

A. *Simulation Parameters*

The expected quality of service requirements and parameters of the reward function are based on those from [12]. In EF class all the packets delayed for more than 10ms are penalized. In AF class packets are penalized when delayed for more than 35ms or when equivalent throughput in ON time interval was lower than 200 kb/s. Lost packets are penalized in all the classes. The parameters of the reward function are given in Table 1.

For the DFT algorithm the queue length equals 10 for EF class and 100 for AF and BE classes. In turn, for RED parameter setup follows instructions from [15], and in all the classes we have: N_{MIN}=5 packets, N_{MAX}=3*N_{MIN}, α=0.002, p_{MAX}=0.1.

Regarding RL-QDL algorithm, based on experience and various testing, the RL parameters were set as follows: α_μ=0.001, α_σ=0.0001, γ=0.2 for all the classes. The start values for the variance (all the classes) was set to $\sigma_0^2=10^{-6}$, and the starting average queuing time limits were: μ_0^{EF}=7.5ms, μ_0^{AF}=15ms and μ_0^{BE}=30ms. Update time interval for RL is set to 10s. For simplicity reasons we suppose that the queues themselves are infinite. However, notice that, since the algorithm dynamically drops packets, the number of packets in the queue will eventually be limited.

B. *Results*

All the simulations were done using OPNET network simulator. The length of the simulations was 5000s, where statistics were collected after the first 2000s. In case of RL - QDL, the RL mechanism was started after 500s.

1) *Case study 1: Load variation*

In the first case study we consider the same traffic load (bit/s) for all the classes, i.e. one third of the offered traffic belongs to EF, one third to AF and one third to BE class independently of the overall offered load. Then the system behaviour is analysed when varying the total average load from 60% to 100% of the link capacity. The load variation is simulated by scaling the average interarrival session time in each source. In particular, Table 3 gives session duration times and session interarrival times for an average load of 100%; as well as overall load for that case (i.e. the total offered load from one source).

Figure 5 shows the results for reward per second obtained in simulations. The specific WFQ weights are marked in parentheses. It can be seen that for load below 70%, the difference among applied algorithms is negligible. When increasing the load, obviously higher revenue is obtained when higher classes are treated with more priority (W_{EF}:W_{AF}:W_{BE}=3:2:1). Depending on the schedulers weight proportion DFT or RED give better results one than the other. However, in all the cases RL-QDL achieves a higher reward than the other alternatives. This shows that our adaptive AQM algorithm can adjust and contribute to increase in QoS independently of the scheduler's weights. In case of 100% loaded network, average system reward gains at least 5,8% in W_{EF}:W_{AF}:W_{BE}=3:2:1 case, and when W_{EF}:W_{AF}:W_{BE}=1:1:1 gain is at least 28,9%.

The interaction of results learned from classes can be deducted from Figure 6 and Figure 7, which compare loss and delay for RED and RL-QDL algorithm, for a 100% loaded network. It can be noticed that, while having less packets dropped in all the classes (higher overall throughput), the average delay in both EF and AF class is still lower for RL-QDL algorithm. This implies that the proposed method can drop packets more efficiently when dealing with congestion. The effect is even more significant when W_{EF}:W_{AF}:W_{BE}=1:1:1.

In Figure 8 cumulative distribution function of edge-to-edge delay in EF class is presented. As for RED the delay depends on serving rate and queue length, it will increase with traffic load increase and in case of W_{EF}:W_{AF}:W_{BE}=1:1:1 will become unsuitable. Meanwhile, delay is directly controlled by our algorithm and is better adjusted to system demands again.

Table 3: Sessions' distribution (100% loaded network).

Source	Session Interarrival (s)	Session duration (s)	Load (kb/s)
S0, S4	0.96	30	1000
S1, S5	7.68	60	500
S2, S6	4.5	30	1000
S3, S7	3.84	30	500

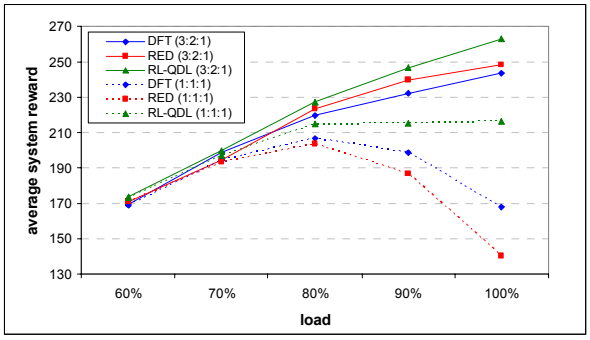


Figure 5: System's reward per second.

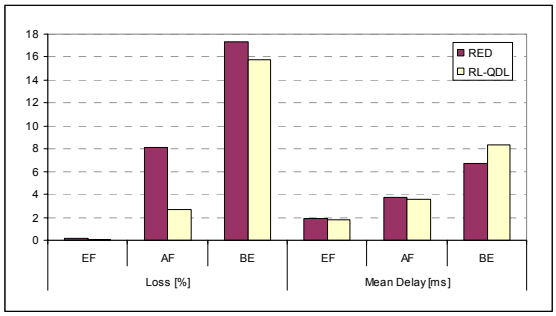


Figure 6: Mean delay and loss when W_{EF}:W_{AF}:W_{BE}=3:2:1.

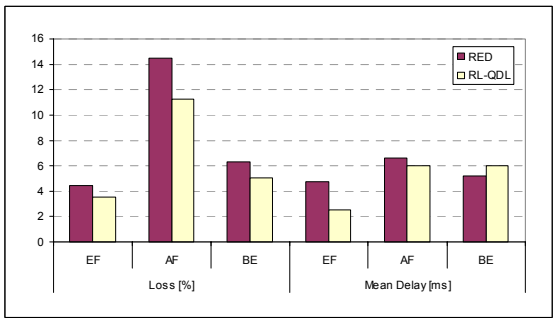


Figure 7: Mean delay and loss when W_{EF}:W_{AF}:W_{BE}=1:1:1.

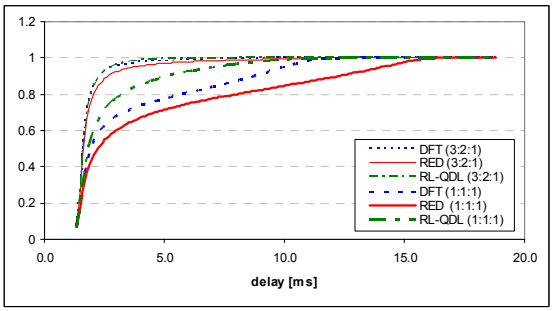


Figure 8: Cumulative distribution function of delay in EF class for 100% loaded network.

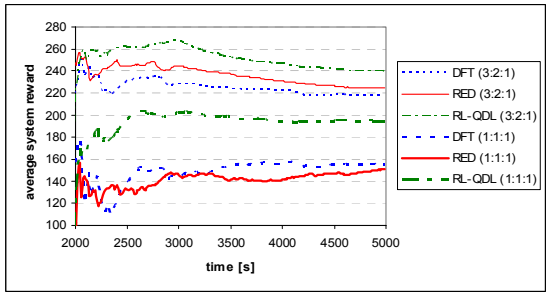


Figure 9: System reward for dynamic class proportion.

2) Case study 2: Dynamic class proportion

Second test was to observe the system's behaviour when proportion in contribution to overall traffic is dynamically changed among classes. In particular, the average number of users generated in source S4 was halved after 3000s. At the same moment and for the rest of the simulation the rest of the bandwidth is occupied by adding more users in source S7. Starting from the values from Table 3, system is all the time 100% loaded in average. In this way the number of users of the premium class (EF) has been significantly reduced in some of the routers, and the bandwidth is occupied by more BE traffic. The results obtained for this case are shown in Figure 9, revealing the trend of having higher revenue with RL-QDL. The difference is especially significant in case of equal prioritization of classes (W_{EF}:W_{AF}:W_{BE}=1:1:1). Note that the fluctuations at the beginning of the figure are due to the stabilization of the statistics, as only after 2000s the statistics are starting being collected.

VI. Conclusion

In this work we have presented a new active queue management algorithm that uses reinforcement learning to increase system performances (RL-QDL). In tests made, our algorithm gave better results than the DFT and RED classical approaches, of up to 29%. The advantage of the proposed RL-QDL algorithm is its ability to learn from the environment in order to adapt to networks with different load levels, independently of the class prioritization and changes in traffic proportion, thanks to the reinforcement learning capabilities. Consequently, it becomes an eligible candidate to take place as a solution to support QoS provisioning in all-IP networks.

Acknowledgement

This work has been performed in the framework of the IST AROMA project, which is partly funded by the European Community and by the Spanish Research Council (CICYT) under grant TEC2006-26873-E, of the COSMOS project TEC2004-00518, and FPU scholarship (MEC, Spain).

References

[1] 3GPP TR 22.978: "All-IP Network (AIPN) feasibility study", Available at: http://www.3gpp.org/

[2] S. Balke et al, An Architecture for Differentiated Services, Dec. '98, RFC 2475

[3] V. Jacobsen et al, An Expedited Forwarding PHB, June '99, RFC 2598

[4] J. Heinanen et al,Assured Forwarding PHB Group, June '99,RFC 2597

[5] B. Braden et al, "Recommendations on Queue Management and Congestion Avoidance in the Internet", April '98, RFC 2309

[6] A Solution Brief from Ericsson and Juniper Networks, "Beyond Best Effort: Telecom-Quality IP Infrastructure for Mobile Softswitching", Ericsson AB 2006 and Juniper Networks 2006, 297 01-FGB 101 113

[7] P. Gevros et al, "Congestion Control Mechanisms and the Best Effort Service Model", *IEEE Network*, vol. 15, pp. 16-26, May/June 2001.

[8] V. Sharma, J. T. Virtamo, "A finite buffer queue", *GLOBECOM '99*, vol. 1B, pp. 1053-1057, 1999

[9] S. Floid, V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397-413, Aug. 1993

[10] F. Agharebparast, V.C.M. Leung, "Improving the Performance of RED Deployment on a Class Based Queue with Shared Buffers", *IEEE GLOBECOM '01*, vol. 4, pp. 2363–2367, Nov. 2001

[11] R. S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction*, A Bradford Book, MIT Press, Cambridge, MA 1998

[12] T. Chee-Kin Hui, C.-H. Tham, "Adaptive Provisioning of Differentiated Services Networks Based on Reinforcement Learning", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 33, pp. 492-501, Nov. 2003

[13] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Learning*, vol. 8, pp 229-256, 1992

[14] A. Demers, S. Keshavt and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm", *Proc. SIGCOMM '88*, vol. 19, Sept. 1989

[15] S. Floyd, "Disscusion of Setting Parameters", Available at: www-nrg.ee.lbl.gov/floyd/red.html