A Comparative Study of Active Queue Management Algorithms for Network Performance Evaluation

Ammar Hameed Shnain [#1], Sarah Hadi shaheed[*2]

[#]*Computer Technical Engineering department , The Islamic University, Najaf , Iraq*
ammar.hameed.it@gmail.com

[*]*Computer Technical Engineering department , The Islamic University, Najaf , Iraq*
sarah_hadi_eng@yahoo.com

***Abstract*— With the current advances in network activities, there appears to be wide needs for Quality of Services (QoS) and fairness among flows. As such, the links are also forced as a result in order to effectively perform in congestion control and avoidance. For this reason, active queue management algorithms were proposed in order to handle QoS issues. This study will examine the unique the performance effeteness of three AQM algorithms namely REM, AQV, and CoDel due to the little evidence found in literature about their performance. The performance above mentioned algorithms will be evaluate via simulation tool (NS2) in terms of response time, throughput and packet loss in single link network . The throughput result showed that REM performed best, closely followed by CoDel and AVQ. REM has the faster response time, followed by CoDel , with AVQ performing worst overall by a significant margin. the packet loss showed AVQ, and CoDel algorithms to do equally, and seem REM performs very poorly. The performance result of REM, AVQ, and CoDel showed different performance results in which REM and CoDel mechanisms helped control the amount of bandwidth shared within a network.**

Keywords*—*AQM, REM, CoDel, AVQ, Bufferbloat

1. INTRODUCTION

In order to ensure effective packet transfer in distributed networks, it is essential to provide instant control for networks to synchronize the traffic load during stable/unstable network flow. This is mostly initiated for customization of packet transfer over a network. In such scenario, Active Queue Management (AQM) schemes are used for the aim of managing packets with certain Quality of Service (QoS) prerequisites [1]. AQM is identified as a mechanism for controlling queue length in order for it not to run full, adding its maximum delay under load [2]. AQM algorithms stimulate a sufficient drop of network packets inside a buffer based on the properties of Network Interface Controller (NIC) [3]. In addition, reducing network congestion is another aspect AQM algorithms deal with, especially when the buffer becomes full The potential from using AQM in a network is to add to the work of end-system protocols such as TCP, with regards to its congestion control. Bisoy and Pattnaik [4] described its option for increasing network utilization, and thus limit packet loss and delay. Lately, TCP networks consist of bulk files transfers which are found to affect traffic management. With web traffic volume increased, the first AQM algorithm was proposed named Random Early Detection (RED) which was designed to manage certain traffic volumes [5]. After developing RED, much work was devoted to enhance its managing features based on network related settings. Moreover, some previous studies considered RED to contain many drawbacks revealed by extensive analysis [6]. Thus, more work for developing new AQM schemes appeared with emphasis on solving parameter-tuning limitations. In addition, the main characteristic of parameter tuning is that it provides flexible adjustment of the parameter values while having constant duration, as such, certain single value per parameter is needed [7]. However, the problem of tuning a queuing management algorithm in certain environmental settings is considered difficult, which is usually due to the large number of options, while having little knowledge of key parameters that affect the algorithm performance [8].

2. PROBLEM STATEMENT

It has been evident that management related issues affect TCP performance in REM of sharing received packets accurately [9]. According to Sheikhan, et al. [10], the consequences of having the bursts of packets running fully can result in slowing down TCP to function properly in the face of congestion. In addition, limited QoS evidence on AQM performance makes it difficult to prevent TCP failure or slowing down, which can be a result of different performance parameters in certain network conditions [10]. Hence, this study compared three algorithms, namely Random Early Marking (REM) [11], Adaptive Virtual Queue (AVQ) [12], and Controlled Delay (CoDel) Nichols and Jacobson (2012), which were selected due to the evidence found by scholars in fair traffic settings [13].

3. LITERATURE REVIEW

3.1 Active Queue Management (AQM)

Schemes for managing queues consists of three key parts, named 1) the congestion/blockage indicator, 2) the congestion/blockage control function, and 3) the marking algorithm, as shown in Figure 1 [1, 14]. The blockage indicator is employed by the actual queue management to determine any possible congestion whilst the blockage control examines and identifies the necessary steps to be taken when congestion is detected. The response mechanism for congestion signal can be helpful to alert the data transmitter to modify its transmitting rates. For instance, the congestion indicator for drop tail queues can represent the total value of instantaneous queue lengths to which the congestion control function is usually dropped using the probability of a single queue.

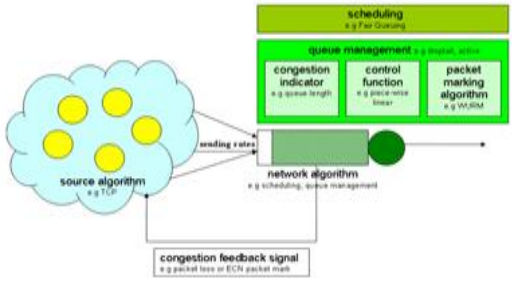

Figure1.AQM in Internet congestion control

3.2 previous Study

The review of the literature shown in Table 1 had led this investigation to conclude that there is no study that previously compared the performance and network stability of AQMs in REM, AVQ, and CoDel. This could pose a strong limitation in current work initiatives to promote the provision of better network stability.

Table 1. A comparison of AQM schemes from the literature

Study	Title	AQM
Bitorika, et al. [18]	A comparative study of active queue management schemes	ARED, REM, CHOKe, PI, AVQ, DRED, GREEN, and LDC
Jiang, et al. [19]	Nonlinear analysis of RED—a comparative study	RED and Adaptive RED
Zhu, et al. [20]	A comparison of active queue management algorithms using the OPNET Modeler	RED, SRED, BLUE, and DRED
Chandra, et al. [21]	Analysis of active queue management algorithms and their implementation for TCP/IP networks using OPNET simulation tool	RED, SRED, DRED, and SDRED
Bagal, et al. [22]	Comparative study of RED, ECN, and TCP rate control	RED, ECN, and TCP rate control
Que, et al. [23]	An improvement algorithm based on RED and its performance analysis	ERED and RED
Joo, et al. [24]	A hybrid active queue management for stability and fast adaptation	HRED and RED
Ji and Dong [25]	Design and analysis of a multi-scale active queue management scheme	RQ-AQM, RED, ARED, PI controller, AVQ, and REM
Fengyuan, et al. [26]	A robust active queue management algorithm based on sliding mode variable structure control	SMVS and RED
Sun, et al. [27]	IAPI: An intelligent adaptive PI active queue management scheme	IAPI and PI

4. RESEARCH METHODOLOGY

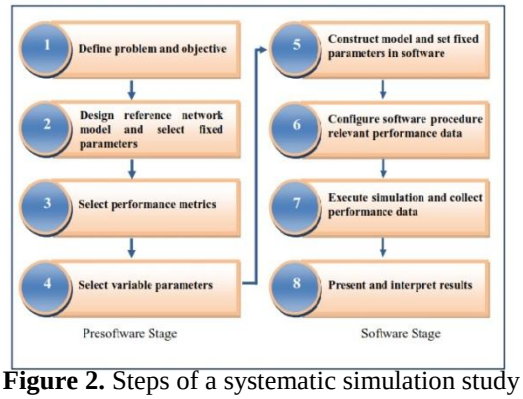

Figure 2. Steps of a systematic simulation study (Mahbub & Raj, 2004)Appendix

In this section, we present the following steps which will be followed in conducting this research:

4.1 Defining Problem and Objectives

The most vital part of this study is perhaps which is necessary for carrying out any research project. We have already presented the problem statement and objectives of this research in the first chapter, which define the purpose of the study implementation depending on the problem statements and objectives.

4.2 Design Reference Network Model and Select Fixed Parameters

All simulations in this study conducted for the network as shown in Figure 3. The topology which we have used in this study consists of eight nodes, in which three nodes are considered the source nodes that are connected with router 1 by three links; the bandwidth used per link is 5 MB; and three nodes are considered as a destination which are connected with router 2 by other three links. Also a single bottleneck link is used between nodes 4 and 5 as shown in Figure 3. All the remaining other links have excess capacity, which ensure that they do not create an additional bottleneck between a source and destination pair. The queuing discipline at node 4 to bottleneck link 4 to 5 use one of the following AQM algorithms to pass the packages, i.e., 1) REM, 2) AVQ and 3) CoDel.

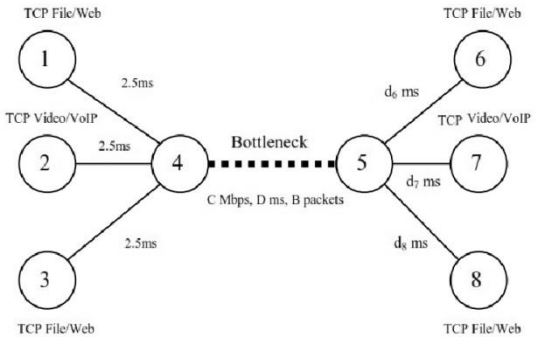

Figure 4. Network topology for Single link network

4.3 Selecting Performance Metrics

This section determine the metrics to be used for the performance evaluation of this study. These performance metrics are referred to as output or response variables that can be observed at the end of the simulation. Performance metrics to be used in this work are discussed below in detail:

4.3.1 Response time

Response time refers to the average amount of time it takes to respond to an ongoing service. The response time of transmission will be calculated based on the time of service and wait time.

4.3.2 Throughput

Throughput is identified as the amount of data transferred successfully from one place to another in a given time period, and typically measured in bits per second (bps), as in megabits per second (Mbps) or gigabits per second (Gbps).

The throughput in this study will be measured using the following equation:

Transmission Time = File Size / Bandwidth (sec)
Throughput = File Size / Transmission Time (bps)

4.3.3 Packet loss

Packet loss refers to the probability of a packet to be lost while it transits from a source node to the final destination.

4.4 Selecting Variable Parameters

In order to obtain performance of an AQM mechanism that will be evaluated, the simulation will run according to different scenarios in which the parameter that influences the performance of AQM mechanism is traffic load. Moreover, in this project the researcher selected three different traffic load (FTP,VOICE,VIDEO) on different network scenarios to see AQM performance and behaviour of algorithms in each scenario through performance metrics results.

4.5 Construct Model and Set Fixed Parameters in Software

In this section the researcher create the reference network scenario (as mentioned in Step 2) in the simulation software. We will also set the values of the fixed parameters in the software model. The actual programming language to be used for the coding depends on the software that used for simulating this work. Furthermore, we will choose NS-2 (Network Simulator) for simulating our work, which is one of the popular simulation software for TCP/IP networks.

4.6 Configure Software to Produce Relevant Performance Data

The software must be configured or programmed to record the values of the performance metrics selected in Step 3.

4.7 Execute Simulation and Collect Performance Data

Finally, the researcher executes the simulation program. These executions are known as "simulation runs." When the simulation runs complete, it will generate a trace file (shown in figure 4) that contains all the information of the simulation to collect performance metrics data , which can be extracted later using an AWK script.

event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

```
r : receive (at to_node)                    src_addr : node.port (3.0)
+ : enqueue (at queue)                      dst_addr : node.port (0.0)
- : dequeue (at queue)
d : drop    (at queue)

r 1.3556 3 2 ack 40 ------- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
```

Figure 4. Example of NS-2 Trace File

4.8 Present and Interpret Results

In this section the researcher plot results by using a common tool called Microsoft Excel, where the performance

metrics will be shown on the y axis and the variables on the x axis of each figure. Furthermore, some useful tables will also be drawn from these raw data to show certain performance results. In addition, we will also present the results in graphs and tables to express and interpret them for useful conclusions

.

5 EXPERIMENTAL SETUP

In order to develop a concrete simulation, a specific topology must be chosen, show in the second step of the methodology. the topology that has been used for the simulation Nodes 1-3 and also 6-8 are producers/consumers of either FTP traffic (designated as "file" in the diagram) or web traffic (TCP ports 80 or 8080, designated as "web" in the diagram). Nodes 2 and 7 are producers/consumers of either streaming video traffic, or VOIP traffic. Each of the links on the left has a fixed transmission time of 2.5 ms, which each of the links on the right has parameterized transmission times of d6, d7 and d8. Finally, the two clusters are connected by a completely parameterized link ("bottleneck" in the diagram) with throughput of C Mbps, delay of D ms, and packet loss of B (a floating point number between 0.0 and 1.0). Thus, in this simulation there are six adjustable parameters. Table 1 shows the overall simulation parameters.

Table 2. Simulation Parameters

Description	Value	Unit
No. of node	8	node
Network area size	100 * 100	m²
Simulation time	50	s
Traffic loads	FTP,voice,video	Bits/s
Daly link	2.5	ms

6 RESULTS

Results for FTP can be seen in Figures 5 through 7. Each figure shows a different performance measure plotted for each of the three variants of Active Queue Management that we have been studying. For throughput it can be seen that REM perform best, closely followed by CoDel. AVQ performs worst in throughput for FTP. This is what one would expect intuitively due to the highly uniform nature of FTP data traffic.

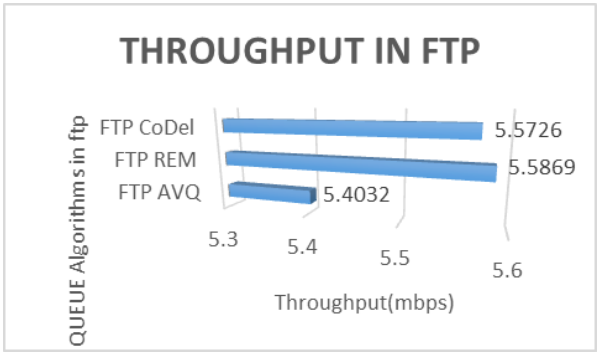

Figure 5. Throughput in FTP for each of the three algorithms

Alternately, one may observe that the three algorithms perform very differently when response time is measured. CoDel has the faster response time, followed by ERM, with ACQ performing worst overall by a significant margin. Since CoDel does best with structured traffic, this result is not surprising, but the magnitude of the variation between the best algorithm and the worst algorithm was not initially an expected outcome.

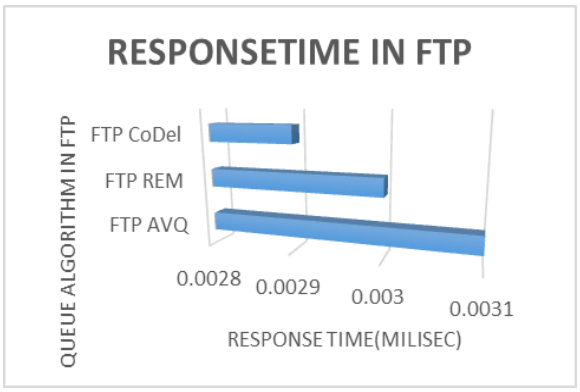

Figure 6. Response Time in FTP for each of the three algorithms

For packet loss, REM performs very poorly (almost 7% loss), while the other two algorithms seem to do equally well. Because user expectations for FTP are that it is not a "real time" protocol, even a 7% packet loss may be acceptable by most.

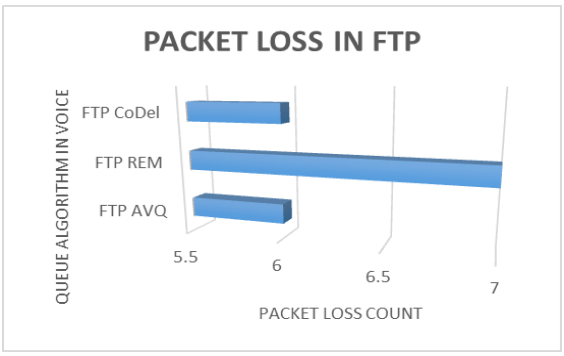

Figure 7. Packet Loss in FTP for each of the three algorithms

Figures 8 through 10 show the four performance measures for each of the three algorithms. It is immediately noticeable that AVQ has poor performance in throughput with respect to the other two algorithms, From an aggregate standpoint REM does best in this metric, as well as in the response time metric, where it is at least twice as responsive as the other two algorithms. VOICE has a fundamentally different packet structure than FTP and it may be acceptable for some packets to be dropped. However, the requirement for in order delivery cannot be relaxed. This can lead to a significant backlog at the sender node.

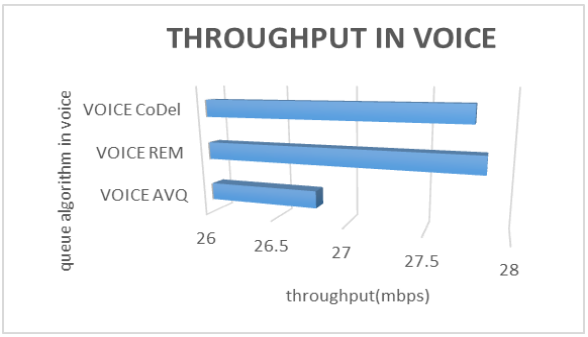

Figure 8. Throughput in VoIP for each of the three algorithms

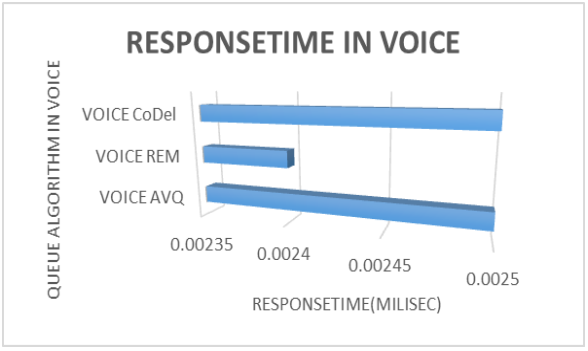

Figure 9. Response Time in VoIP for each of the three algorithms

When measured by packet loss, however, CoDel achieves roughly 8% while the other two algorithms reach more than 9%. This aligns with the expectation that CoDel is often the best algorithm in terms of packet delivery, while it is perhaps not the best in terms of other metrics.

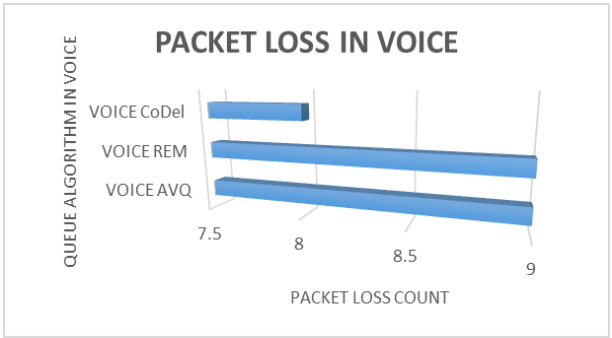

Figure 10. Packet Loss in VoIP for each of the three algorithms

Figures 11 – 13 show the three performance measurements for streaming video, for each of the three algorithms. CoDel ,AVQ and REM do well in terms of throughput. REM also does very well in terms of response time, while both other algorithms have somewhat longer values for that metric. Packet loss is above 3% for all three algorithms, but it is best for REM and worst for CoDel. As with the other results, this is not surprising considering the nature of the traffic. From the standpoint of the user experience, this degree of loss may be considered acceptable, due to frame buffering.

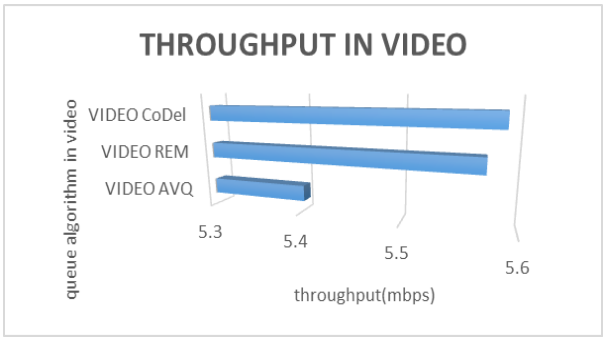

Figure 11. Throughput in video for each of the three algorithms

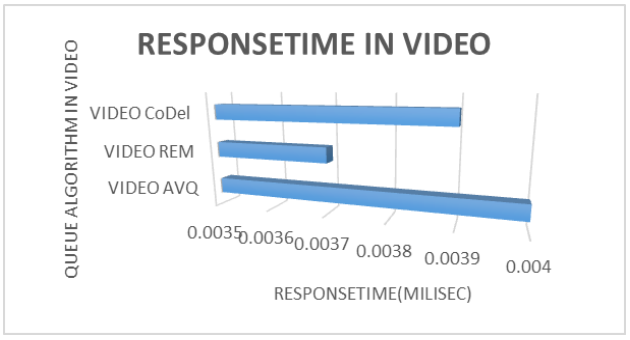

Figure 12. Response Time in video for each of the three algorithms

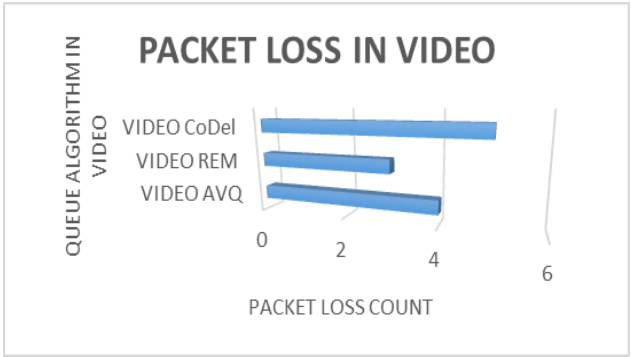

Figure 13. Packet Loss in video for each of the three algorithms

7 CONCLUSION

This study has investigated three different algorithms for active queue management of Random Early Marking or REM, Adaptive Virtual Queue (AVQ), and Controlled Delay (CoDel), and has also performed simulations to judge these three algorithms based on three commonly used performance measures. This study determined the network performance of File Transfer Protocol (FTP), video streaming, and Voice over IP (VOIP) based on network response time (delay), network throughput and network packet loss

The result showed that REM to offer a stable performance in FTP and Video. Meanwhile, ConDel provided the best performance in a voice to which it also regulate the queue performance to respond in less time as compared with other algorithms. However, AVQ performance result showed weak performance in all the three scenarios. This can be reasoned to that AVQ has nothing to do with the managing of bandwidth demand and the number of active users through which the performance measure is commonly regulated around its target value in order to sustain less packet loss and delay.

On the other hand, AVQ was originally designed to alleviate performance during congestion in which it may not necessary foster the performance of a network when it manage a link load situations. From these, the researcher concluded that ConDel and REM provide sufficient performance in a single metric. Moreover, each algorithms has strengths and weaknesses that can be observed by the various behaviours with respect to these three metrics.

ACKNOWLEDGMENT

Our utmost gratitude goes to The Islamic University in Iraq for supporting us by funding, support and facilities provided in order to complete this research

REFERENCES

[1] R. Adams, "Active queue management: A survey," Communications Surveys & Tutorials, IEEE, vol. 15, pp. 1425-1476, 2013.
[2] C. Kasnakoglu, "Active Queue Management of TCP Flows with Self-scheduled Linear Parameter Varying Controllers," International Journal of Computers Communications & Control, vol. 8, pp. 838-844, 2013.
[3] J.-h. Kim, H. Yoon, and I. Yeom, "Active queue management for flow fairness and stable queue length," Parallel and Distributed Systems, IEEE Transactions on, vol. 22, pp. 571-579, 2011.
[4] S. K. Bisoy and P. K. Pattnaik, "Analysis of a Network of AQM Router Supporting TCP Flows and Comparison with XCP," in Fourth International Conference of Emerging Applications of Information Technology (EAIT), 2014 2014, pp. 101-106.
[5] A. H. Ismail, A. El-Sayed, Z. Elsaghir, and I. Z. Morsi, "Enhanced Random Early Detection (ENRED)," International Journal of Computer Applications, vol. 92, pp. 25-28, 2014.
[6] F. Li, J. Sun, M. Zukerman, Z. Liu, Q. Xu, S. Chan, et al., "A comparative simulation study of TCP/AQM systems for evaluating the potential of neuron-based AQM schemes," Journal of Network and Computer Applications, vol. 41, pp. 274-299, 2014.
[7] A. Arcuri and G. Fraser, "Parameter tuning or default values? An empirical investigation in search-based software engineering," Empirical Software Engineering, vol. 18, pp. 594-623, 2013.
[8] A. E. Eiben and S. K. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," Swarm and Evolutionary Computation, vol. 1, pp. 19-31, 2011.
[9] N. Farzaneh and M. H. Yaghmaee, "Joint active queue management and congestion control protocol for healthcare applications in wireless body sensor networks," in Toward Useful Services for Elderly and People with Disabilities, ed: Springer, 2011, pp. 88-95.
[10] M. Sheikhan, R. Shahnazi, and E. Hemmati, "Adaptive active queue management controller for TCP communication networks using PSO-RBF models," Neural Computing and Applications, vol. 22, pp. 933-945, 2013.
[11] S. H. Low and D. E. Lapsley, "Optimization flow control—I: basic algorithm and convergence," IEEE/ACM Transactions on Networking (TON), vol. 7, pp. 861-874, 1999.
[12] S. Kunniyur and R. Srikant, "Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management," in ACM SIGCOMM Computer Communication Review, 2001, pp. 123-134.
[13] S. Zhang, J. Xu, and K.-w. Chung, "On the stability and multi-stability of a TCP/RED congestion control model with state-dependent delay and discontinuous marking function," Communications in Nonlinear Science and Numerical Simulation, vol. 22, pp. 269-284, 2015.
[14] V. Firoiu and M. Borden, "A study of active queue management for congestion control," in INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2000, pp. 1435-1444.
[15] M. H. Suzer, K.-D. Kang, and C. Basaran, "Active queue management via event-driven feedback control," Computer Communications, vol. 35, pp. 517-529, 2012.
[16] K. Chavan, R. G. Kumar, M. N. Belur, and A. Karandikar, "Robust active queue management for wireless networks," Control Systems Technology, IEEE Transactions on, vol. 19, pp. 1630-1638, 2011.
[17] G. Thiruchelvi and J. Raja, "A survey on active queue management mechanisms," International Journal of Computer Science and Network Security, vol. 8, pp. 130-145, 2008.
[18] A. Bitorika, M. Robin, M. Huggard, and C. Mc Goldrick, "A comparative study of active queue management schemes," in Proc. of ICC, 2004.
[19] K. Jiang, X. Wang, and Y. Xi, "Nonlinear analysis of RED—a comparative study," Chaos, Solitons & Fractals, vol. 21, pp. 1153-1162, 2004.
[20] C. Zhu, O. W. Yang, J. Aweya, M. Ouellette, and D. Y. Montuno, "A comparison of active queue management algorithms using the OPNET Modeler," Communications Magazine, IEEE, vol. 40, pp. 158-167, 2002.
[21] H. Chandra, A. Agarwal, and T. Velmurugan, "Analysis of active queue management algorithms & their implementation for TCP/IP networks using OPNET simulation tool," International Journal of Computer Applications, vol. 6, 2010.
[22] P. Bagal, S. Kalyanaraman, and B. Packer, "Comparative study of RED, ECN and TCP Rate Control," ed, 1999.
[23] D. Que, Z. Chen, and B. Chen, "An improvement algorithm based on RED and its performance analysis," in Signal Processing, 2008. ICSP 2008. 9th International Conference on, 2008, pp. 2005-2008.
[24] C. Joo, S. Bahk, and S. S. Lumetta, "A hybrid active queue management for stability and fast adaptation," Communications and Networks, Journal of, vol. 8, pp. 93-105, 2006.
[25] Q.-J. Ji and Y.-Q. Dong, "Design and analysis of a multiscale active queue management scheme," Journal of Computer Science and Technology, vol. 21, pp. 1022-1030, 2006.
[26] R. Fengyuan, L. Chuang, Y. Xunhe, S. Xiuming, and W. Fubao, "A robust active queue management algorithm based on sliding mode variable structure control," in INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2002, pp. 13-20.
[27] J. Sun, S. Chan, and M. Zukerman, "IAPI: An intelligent adaptive PI active queue management scheme," Computer Communications, vol. 35, pp. 2281-2293, 2012.