An Active Queue Management for High Bandwidth-Delay Product Networks

Shahram Jamali and Seyed Reza Zahedi

***Abstract*—As the per-flow product of bandwidth and latency increases, TCP/RED becomes inefficient and prone to instability. To address this problem we are going to design a novel active queue management (AQM) that supports TCP to achieve high performance in such networks. This paper proposes a novel algorithm, in which, particle swarm optimization (PSO) technique is used to dynamic tuning of RED's parameters. This algorithm formulates the active queue management issue as an optimization problem and employs PSO technique to direct it to its optimum point. Dynamic tuning of parameters helps the proposed algorithm to be adapted to special situations such as high bandwidth and large delays. Simulation results show that the proposed algorithm behaves remarkably better than RED in terms of queue size, number of dropped packets, and bottleneck utilization for these networks.**

***Index Terms*—AQM, RED, high bandwidth delay product.**

I. Introduction

For the Internet to continue to expand, its congestion control mechanism must remain succeed as the network evolves. Technology trends indicate that the future Internet will have a large number of very high-bandwidth links. Less ubiquitous but still commonplace will be satellite and wireless links with high latency. These trends are problematic because TCP/RED reacts adversely to increases in bandwidth or delay.

Mathematical analysis of current congestion control algorithms shows that, regardless of the queuing scheme, as the delay-bandwidth product increases, TCP becomes oscillatory and prone to instability. By casting the problem into a control theory framework, Low et al. [1] show that as capacity or delay increases, RED [2], Random Early Marking (REM) [3], Proportional Integral Controller [4], and Virtual Queue [5] all eventually become oscillatory and prone to instability. Furthermore, Katabi and Blake show that Adaptive Virtual Queue (AVQ) [6] also becomes prone to instability when the link capacity is large enough (e.g., gigabit links).

To address this problem we are going to design a novel active queue management (AQM) that suppurts TCP to achieve high perfrmance in such networks. This algorithm formulates the active queue management issue as an optimization problem and employs PSO technique [7]-[11] to direct the system to its optimum point. Dynamic tunning of parameters helps the proposed algorithm to be adapted to high-bandwidth and large-delay (HBD) environments. Structure of this paper is as follows. Section II, briefly presents preliminaries for RED active queue management scheme. In Section III we present an introduction to PSO method. Section IV describes a procedure through which the PSO-based active queue management scheme is developed for HBDP networks. Section V brings simulation results for the proposed algorithm and finally Section VI presents concluding remarks.

Manuscript received November 11, 2012; revised March 5, 2013.

Shahram Jamali is with the Electrical and Computer Engineering Department University of Mohaghegh Ardabili, Ardabil, Iran (e-mail: Jamali@iust.ac.ir).

Seyed Reza Zahedi is with the Computer Engineering Department, Islamic Azad University-Khalkhal Branch Khalkhal, Iran (e-mail: Seyed.reza.zahedi@gmail.com).

II. RED Preliminaries

Nowadays, despite the significant developments of recent years in AQM theory and technology, RED [2] controller is used in almost all routers of the Internet. This is due to its good and robust performance for a wide class of scenarios and shapes of incoming loads and wide range of operating conditions. Furthermore, it is easy to implement with low computational load on routers. A router implementing RED accepts all packets until the queue reaches its minimum threshold min_{th}, after which it drops a packet with a linear probability distribution function. When the queue length reaches its maximum threshold max_{th}, all packets are dropped with a probability of one. The basic idea behind RED is that a router detects congestion early by computing the average queue length *avg* and sets two buffer thresholds max_{th} and min_{th} for packet drop as shown in Fig. 1. The RED algorithm, therefore, includes two computational parts: computation of the average queue length and calculation of the drop probability. The average queue length at time *t*, is defined according to equation (1).

$$avg(t) = (1-w)avg(t-1) + wg(t) \quad (1)$$

The $avg(t)$ is the new value of the average queue length at time *t*, $q(t)$ is instantaneous queue length at time *t*, and w is a weight parameter for calculating $avg(t)$. The average queue length tracks the instantaneous queue length. However, because w is much less than one, *avg* changes much slower than *q*. Therefore, *avg* follows the long-term changes of *q*, reflecting persistent congestion in communication networks.

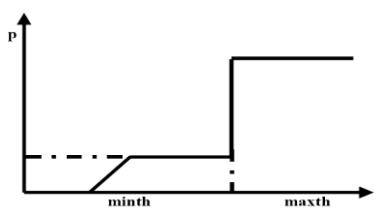


Fig. 1. RED drop function

In Fig. 1 max_p is the maximum packet drop probability. By making the packet drop probability a function of the level of congestion, RED gateway has a low packet-drop probability during low congestion, while the drop probability increases as the congestion level increases.

Many researches are carried out for improving RED algorithm [12]-[21]. For example in [12] the proposed method tries to keep the average queue length close to predetermined queue length by network administrator dynamically. But this research has many drawbacks. In this research it is assumed that traffic does not change suddenly. This assumption regarding burst traffic of networks can not be correct. On the other hand, determination of average queue length by administrator is problematic and this value is variable regarding traffic type and maximum capacity of queue. Because of these reasons the proposed method must be parameterized in each network individually. In another research [13], authors use fuzzy method for decreasing the number of lost packets. Generally, fuzzy-based methods which follow from a set of rules have two major drawbacks. First, it may be occurred a new situation in real environment which is not predicated by designer and so leads to fail of the method. Second, if in the real environment a very little modification is occurred, designer must modify whole of the rules and so it is against the scalability. In [15], RED thresholds are adjusted manually, assuming that the ratio of minimum threshold to maximum threshold stands at 1:3. In other researches [16], [17], values of maximum and minimum thresholds are adjusted manually. Some other researches on this field can be studied in [18]-[20]. In this work we will go in this path by proposing a PSO-based procedure to dynamically adjustment of RED parameters, namely max_{th} and min_{th}.

III. Particle Swarm Optimization Algorithm

Particle swarm optimization algorithm, which is tailored for optimizing difficult numerical functions and based on metaphor of human social interaction, is capable of mimicking the ability of human societies to process knowledge [7]. It keeps track of its coordinates in hyperspace which are associated with its previous best fitness solution, and also of its counterpart corresponding to the overall best value acquired thus far by any other particle in the population.

In PSO algorithm, a swarm consists of m-particle, in which each particle is treated as a point in a N-dimensional space which adjusts its "flying" according to its own flying experience as well as the flying experience of other particles. Each particle uses velocity to determine the direction and value of its "flying", which follows the current optimum in a N-dimension space. The position and velocity of particle i at iteration k can be respectively expressed as:

$$X_i(k)=[X_{i1}(k), X_{i2}(k), \ldots, X_{iN}(k)] \quad (2)$$

$$V_i(k)=[V_{i1}(k), V_{i2}(k), \ldots, V_{iN}(k)] \quad (3)$$

Each particle keeps track of its coordinates in the solution space which are associated with the best solution that has achieved so far by that particle. This value is called personal best $P_{i,best}$, which can be expressed as:

$$P_{i,best}=[P_{i1}(k), P_{i2}(k), \ldots P_{iN}(k)] \quad (4)$$

Another best value that is tracked by the PSO is the best value obtained so far by any particle in the neighborhood of that particle. This value is called global best Pg, best, which can be expressed as:

$$P_{g,best}=[P_{g1}(k), P_{g2}(k), \ldots P_{gN}(k)] \quad (5)$$

The basic concept of PSO lies in accelerating each particle toward its personal best and the global best locations. The velocity and position of particle i at iteration k+1 can be calculated according the following equations:

$$V_i(k+1)= wV_i(k)+c_1r_1(P_{i,best}(k)-x_i(k))+ c_2r_2(P_{g,best}(k)-x_i(k)) \quad (6)$$

$$X_i(k+1)= X_i(k) + V_i(k+1) \quad (7)$$

where $V_i(k)$ is the velocity, w is the inertia weight, c_1 and c_2 are constants which determine the influence of the particle's best previous position $P_i(k)$ and the population's best previous position $P_g(k)$. Parameters r_1 and r_2 are random numbers uniformly distributed within [0, 1] and Xi (k) is the position of the particle i.

As it is reported in [7], this optimization technique can be used to solve many of the same kinds of problems as GA, and does not suffer from some of GA's difficulties. It has also been found to be robust in solving problem featuring non-linearity, non-differentiability and high-dimensionality.

IV. Proposed Method

Performance of RED algorithm, respect to the control of the queuing delay and link utilization, obviously depends on the optimal tuning of the RED's parameters such as max_{th} and min_{th}. Link utilization will be low, if the thresholds are small, whereas congestion might occur, if the thresholds are set to high value. As we said, fixed values for parameters of RED are not able to locate or identify the global optimum for dynamic conditions and special environments such as high bandwidth-delay networks. In order to consider this issue and provide improved performance, we use PSO technique for tuning of RED's parameters and present a novel AQM algorithm. This algorithm monitors network performance and tries to keep it near its optimum point. It aims to keep the queue length in its lowest level and the bottleneck utilization in an acceptable level. For this purpose, as can be found in Fig. 2, the measured performance is defined as an objective function for the PSO algorithm and PSO algorithm search those values for max_{th} and min_{th} that offer the optimal performance for the network.

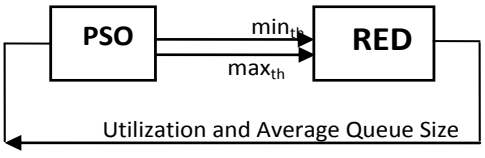


Fig. 2. Structure of PSO-RED algorithm

As the first step of our design process, we outline an objective function to link the design requirements with the optimization algorithm. Considering all the mentioned factors, the objective function is defined as follows:

$$\text{Minimize avg}(t) \quad \text{Subject to Utilization} > 0.9 \tag{8}$$

Now, by using this objective function, we can employ PSO algorithm to design a dynamic mechanism to adjust thresholds. For this purpose, the minimum and maximum thresholds can be calculated according the following equations:

$$V_{mini}(k+1)= w\, V_{minth}(k)+c_1r_1(Gbest_{minth}-min_{th}(k))+ c_2r_2(Lbest_{minth}-min_{th}(k)) \tag{9}$$

$$Min_{th}(k+1)=min_{th}(k)+V\,min_{th}(k+1) \tag{10}$$

$$V_{maxi}(k+1)= w\, V_{maxth}(k)+c_1r_1(Gbest_{maxth}-max_{th}(k))+ c_2r_2(Lbest_{maxth}-max_{th}(k)) \tag{11}$$

$$Max_{th}(k+1)=max_{th}(k)+V\,max_{th}(k+1) \tag{12}$$

where ($Gbest_{minth}$, $Gbest_{maxth}$) is the particle's best previous position and ($Lbest_{minth}$, $Lbest_{maxth}$), is the population's best previous position. We use w=0.6, c_1=0.5, c_2=0.5 and r_1 and r_2 are set stochastically

V. Implementation and Simulation Results

To study PSO-RED's behavior in HBDP networks, we use ns-2 simulation and present a set of simulation results that show its behavior for such environments. For this purpose we consider network of Fig. 3 and define two scenarios over it. In each scenario, we simulate the network once under RED algorithm and then under PSO-RED algorithm to compare their performance in terms of bottleneck utilization, queue length, number of dropped packets and smoothness.

A. Scenario 1: A high Bandwidth-Delay Product networks

Here, we consider a twenty-connection network with single bottleneck link of capacity 1 *Gbps* shared by 20 FTP flows. The bottleneck link delay is 100 ms and packet *size*=500 bytes. All other links have bandwidth of 2 Gbps and their delay is 5 ms. Hence, for all flows we have *RTT*=220 ms. We simulate this network for 300 seconds. Figs. 4-6 shows the simulation results. In order to reference to the results of these Figs, we note that:

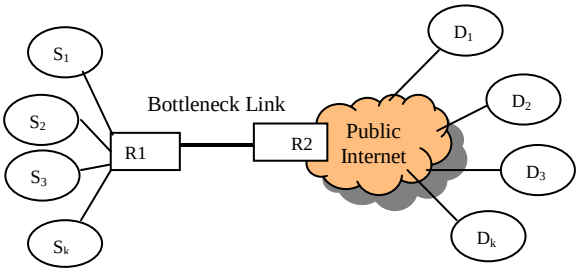
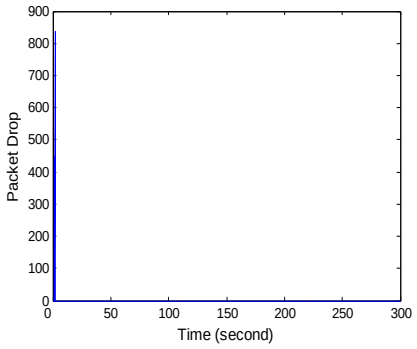


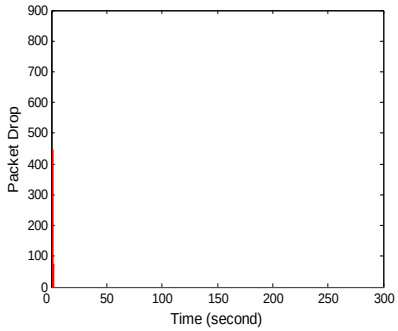
Fig. 3. General network model

1) Packet Drop: Fig. 4 and Table I show that PSO-RED behaves better than RED in term of dropped packets count. This comes from fine tuning of max_{th} and min_{th} that is performed by PSO-RED.
2) Queue Evolution: As can be found in Fig. 5, in this HBDP environment PSO-RED's queue size is much smaller than that of RED. This means that queuing delay of PSO-RED is negligible.
3) Utilization: According to Figs. 6 and 7, PSO-RED's utilization is quite similar to RED's utilization. But, table 2 shows that average utilization of PSO-RED is more than RED about 8%.

You can find another view of comparison results between RED and PSO-RED in table 1. This table shows that average behavior of PSO-RED is better than PSO in terms of utilization, drop rate and queue size.



a. RED's drop counts



b. PSO-RED's drop counts

Fig. 4. PSO-RED drops fewer packets than RED in BDP environments

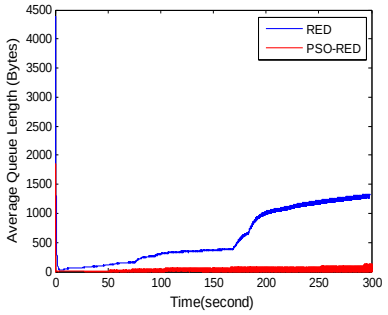


Fig. 5. PSO-RED's queue length is lower than RED's queue length

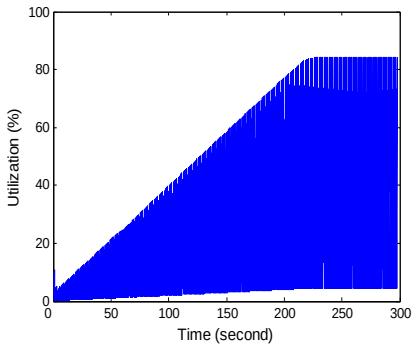


Fig. 6. Bottleneck utilization of RED for a bandwidth-delay network

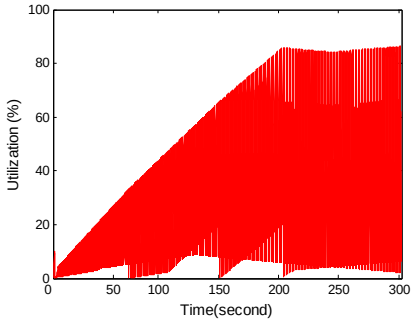


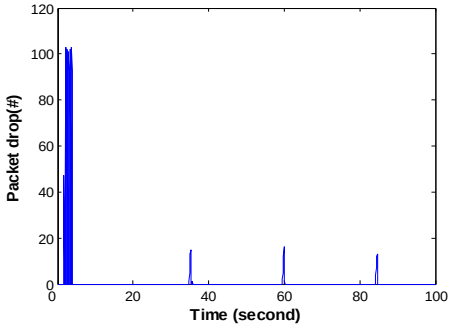
Fig. 7. Bottleneck utilization of PSO-RED for a bandwidth-delay network

TABLE I: COMPARISON OF PSO-RED AND RED ALGORITHMSIN A HIGH BANDWIDTH-DELAY NETWORKS

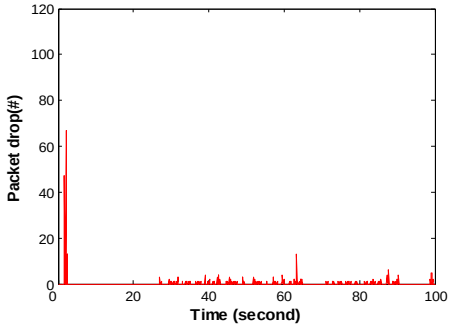
	Dropped Packets Count	Average queue Size (bytes)	Average Utilization
RED	2421	7504	25%
PSO-RED	524	215	33.52%

B. *Scenario 2: A Long-Delay Network*

In order to study PSO-RED behavior for long-delay networks, we consider another scenario in which, there are 30 sources. The bottleneck link delay is 100 ms and its bandwidth is 50 Mbps. Other links have the same propagation delay of 50 ms and bandwidths of 1Gbps. Same RTTs of all flows are 400ms. Fig. 8 shows that PSO-RED behaves better than RED in term of dropped packets count. Fig. 9 shows the average queue length of PSO-RED and RED algorithms during 300 seconds of simulation time, in which, PSO-RED acts remarkably better than RED. Fig. 10 and Fig. 11 show the bottleneck utilization. You can find in Table II that PSO-RED's utilization is higher than RED's about 8%. Overall comparison of RED and PSO-RED in table 2 confirm that PSO-RED behaves better than RED in this scenario as in the previous one.



a. RED's drop counts



b. PSO-RED's drop counts

Fig. 8. PSO-RED drops fewer packets than RED in for long-delay network

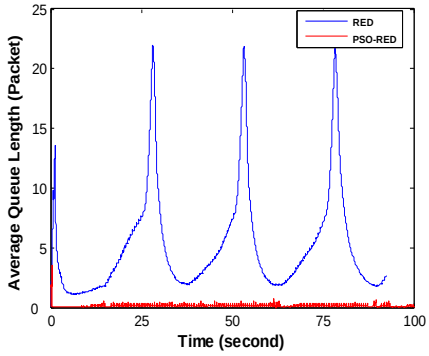


Fig. 9. Queue size of PSO-RED and RED algorithms

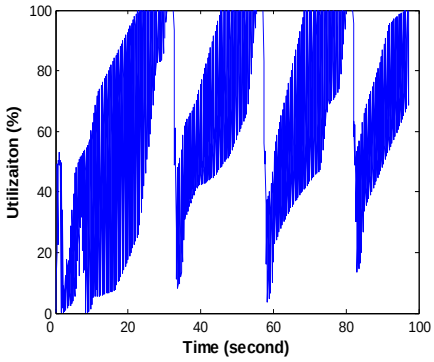


Fig. 10. Bottleneck utilization of RED for long-delay network

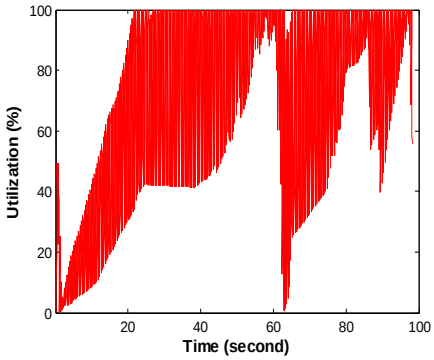


Fig. 11. Bottleneck utilization of PSO-RED for long-delay network

TABLE II: COMPARISON OF PSO-RED AND RED ALGORITHMS FOR LONG-DELAYED NETWORK

	Dropped Packets Count	Average queue Size (bytes)	Average Utilization
RED	1270	5071	63.07%
PSO-RED	366	87	71.30%

VI. CONCLUSION

In this paper we proposed PSO-RED algorithm as an improvement over RED active queue management scheme to support TCP in HBDP environments. The main feature of this algorithm is that it sets max_{th} and min_{th} parameters based on the network dynamic conditions. PSO-RED aims to direct network to an optimum point in which queue length is low while high bottleneck utilization. For this purpose it uses PSO algorithm to solve the optimization problem. The

simulating results indicate that the packet drop count, queue size and link utilization of PSO-red are better than RED algorithm in HBDP environments.

REFERENCES

[1] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, "Dynamics of TCP/AQM and a scalable control," *IEEE INFOCOM*, 2002.

[2] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, 1993.

[3] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "Rem: Active queue management," *IEEE Network*, 2001.

[4] C. Hollot, V. Misra, D. Towsley, and W. Gong, "On designing improved controllers for aqm routers supporting TCP flows," *IEEE INFOCOM*, 2001.

[5] R. Gibbens and F. Kelly, "Distributed connection acceptance control for a connectionless network," *Intl. Tele-traffic Congress*, 1999.

[6] D. Katabi and C. Blake, "A note on the stability requirements of adaptive virtual queue," *MIT Technical Memo*, 2002.

[7] S. H. Zahiria and S. A. Seyedin, "Swarm intelligence based classifiers," *Journal of Franklin Institute*, 2007.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," *IEEE Int. Conf. Neural Networks*, 1995

[9] R. C. Eberhart and Y. Shi, "Particle swarm optimization: Development, applications and resources," *IEEE Congress on Evolutionary Computation*, 2001.

[10] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann Publishers, San Francisco, 2001.

[11] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. On Evolutionary Computation*, 2002.

[12] C. Chien and W. Liao, "A self-configuring red gateway for quality of service (qos) networks," *IEEE ICME*, 2003.

[13] C. Hollot, V. Misra, D. Towlsey, and W. Gong, "A control theoretic analysis of RED," *IEEE INFOCOM*, 2001.

[14] W. Chen and Sh.Yang, "The mechanism of adapting RED parameters to TCP traffic," *Journal of Computer Communications*, 2009.

[15] T. Ye and S. Kalyanaraman, "Adaptive tuning of red using on-line simulation," *GLOBECOM*, 2002.

[16] L. Rossides, A. Sekercioglu, A. Pitsillides, A. Vasilakos, S. Kohler, and P.Tran-Gia, "Fuzzy red: congestion control for tcp/ip diff-serv," *MELECON*, 2000.

[17] V. Vukadinoviæ and L. Trajkoviæ, "RED with dynamic thresholds for improved fairness," School of Engineering Science Simon Fraser University Vancouver, BC, Canada, 2000.

[18] M. Christiansen, K. Jeffay, D. Ott, and F. Donelson. Tuning red for web traffic. USA. [Online]. Available: http://WWW.Cs.Unc.Edu/Research/Dirt.

[19] J. Orozco and D. Ros, "An adaptive rio (a-rio) queue management algorithm," *Lecture Notes in Computer Science*, 2003.

[20] M. Maowidzki, "Simulation-based study of ECN performance in red networks," Military Communication Institute, Poland.

[21] M. Jahanshahi and M. R. Meybodi, "An Adaptive Congestion Control Method for Guaranteeing Queuing Delay in RED-Based Queue Using Learning Automata," *Mechatronics and Automation, ICMA*, 2007.

[22] Ns-2.Network Simulator. [Online]. Available: http://www.isi.edu/nsnam/ns.

Shahram Jamali is an assistant professor leading the Autonomic Networking Group at the Department of Engineering, University of Mohaghegh Ardabili. He teaches on computer networks, network security, computer architecture and computer systems performance evaluation. Dr. Jamali received his M.Sc. and Ph.D. degree from the Dept. of Computer Engineering, Iran University of Science and Technology in 2001 and 2007, respectively. Since 2008, he is with Department of Computer Engineering, University of Mohaghegh Ardabil and has published more than 50 conference and journal papers.

Seyed Reza Zahedi received his B.S. in computer engineering from Tafresh Azad University, Markazi, in 2006, the M.S. in computer engineering from Arak Azad University, Markazi, in 2010.. Currently, he is working as a faculty in Department of Computer Science and Engineering at Islamic Azad University Khalkhal branch. His research interests include swarm intelligence, AQM.