Analysis of Active Queue Management Algorithms & Their Implementation for TCP/IP Networks Using OPNET Simulation Tool

Himanshu Chandra
B.Tech Electronics & Communication
VIT University, INDIA

Ajay Agarwal
B.Tech Mechanical Engineering
VIT University, INDIA

T. Velmurugan
Assistant Professor (Senior)
VIT University, INDIA

ABSTRACT
A number of active queue management algorithms for TCP/IP networks such as RED, SRED, DRED and SDRED have been proposed in the past few years. This article presents a comparative study of these algorithms using simulations. The evaluation is done using the OPNET Modeler, which provides a convenient and easy-to-use platform for simulating largescale networks. The performance metrics used in the study are queue size, packet drop probability, and packet loss rate. The study shows that, among the four algorithms, SRED and DRED are more effective at stabilizing the queue size and controlling the packet loss rate while maintaining high link utilization. The benefits of stabilized queues in a network are high resource utilization, bounded delays, more certain buffer provisioning, and traffic-load-independent network performance in terms of traffic intensity and number of TCP connections.

Keywords
AQM, SRED, DRED, RED, OPNET.

1. INTRODUCTION
Active queue management has been recommended by the Internet Engineering Task Force (IETF) as a way of mitigating the above stated performance limitations of TCP over drop-tail networks. Random Early Detection (RED) is the first active queue management algorithm proposed for deployment in TCP/IP networks. The basic idea behind an active queue management algorithm is to convey congestion notification early to the TCP endpoints so that they can reduce their transmission rates before queue overflow and sustained packet loss occur. It is now widely accepted that a RED controlled queue performs better than a drop-tail queue. However, RED has some parameter tuning issues that need to be carefully addressed for it to give good performance under different network scenarios. As a result several algorithms, like Stabilized RED (SRED) & Dynamic RED (DRED), have been developed as alternatives to RED. Although these algorithms also control congestion by discarding packets with a load dependent probability whenever a queue in the network appears to be congested, they are designed with the objective of maintaining stabilized network queues, thereby minimizing occurrences of queue overflows and underflows, and providing high system utilization. The Stabilized Dynamic RED (SDRED) active queue management algorithm proposed also has similar objectives and is discussed in this article. However, unlike SRED and DRED, this new active queue management algorithm uses a combination of two protocols and control approach to compute the drop probabilities used to discard packets during times of queue congestion. The main objective of this article is to present a comparative analysis of the performance of the RED, SRED, DRED and SDRED algorithms using the OPNET Modeler.

This simulation tool enables us to compare the performance of the algorithms by examining queue sizes, drop probabilities, and packet loss rates. We believe a comparative study of this kind can provide a better understanding of the active queue management algorithms proposed for TCP/IP congestion control.

2. Random Early Detection (RED)
Random early detection (RED), also known as random early discard or random early drop is an active queue management algorithm. It is also a congestion avoidance algorithm.

In the traditional tail drop algorithm, a router or other network component buffers as many packets as it can, and simply drops the ones it cannot buffer. If buffers are constantly full, the network is congested. Tail drop distributes buffer space unfairly among traffic flows. Tail drop can also lead to TCP global synchronization as all TCP connections "hold back" simultaneously, and then step forward simultaneously. Networks become under-utilized and flooded by turns. RED addresses these issues.

It monitors the average queue size and drops (or marks when used in conjunction with ECN) packets based on statistical probabilities. If the buffer is almost empty, all incoming packets are accepted. As the queue grows, the probability for dropping an incoming packet grows too. When the buffer is full, the probability has reached 1 and all incoming packets are dropped.

RED is more fair than tail drop, in the sense that it does not possess a bias against bursty traffic that uses only a small portion of the bandwidth. The more a host transmits, the more likely it is that its packets are dropped. Early detection helps avoid global synchronization.

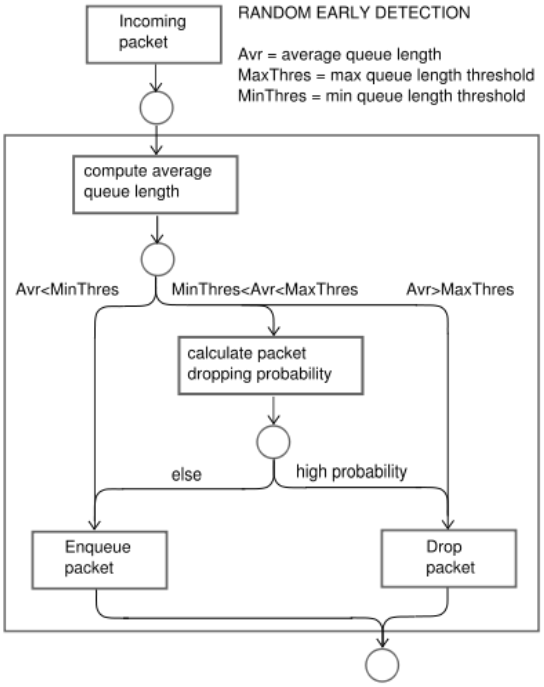

Figure 1: Block Diagram of RED

2.1 RED – Simulation using OPNET

Queue Size in Packets is shown in this simulation Figure 2. Shows a very high degree of variation in the size of the Queue. Amounting to large amount of packet and data losses.

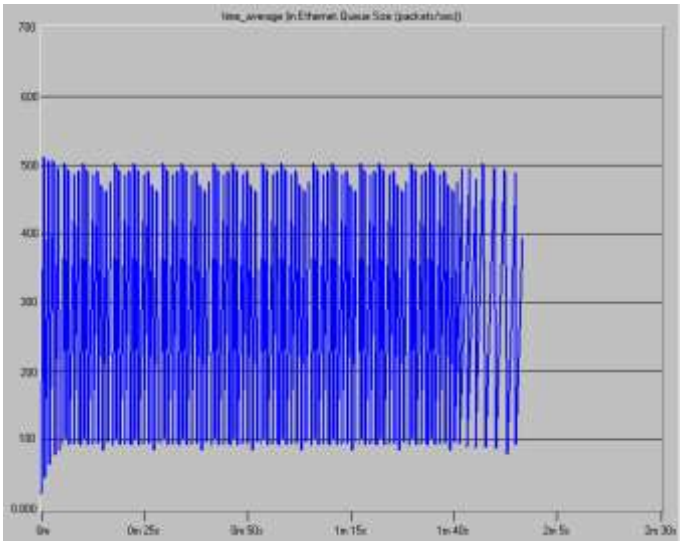

Figure 2: RED Simulation Result

RED makes quality of service (QoS) differentiation impossible. Weighted RED (WRED) and RED In/Out (RIO) provide early detection with some QoS considerations.

3. Stabilized RED (SRED)

The SRED algorithm is designed to stabilize the queue size at a level independent of the number of active connections. In SRED, the packet drop probabilities are computed from estimates of the number of active flows and the instantaneous queue size. The number of active flows in the queue is estimated using a simple form of flow cache (called the zombie list). The zombie list is equivalent to a list of M recently seen flows augmented with a count and timestamp field. Starting with an empty list, for every arriving packet, the packet flow identifier (source address, destination address, source port number, destination port number, etc.) is added to the list, the count of the zombie is set to zero, and its timestamp is set to the arrival time of the packet.

The estimation process works as follows once the list is full. Whenever a packet arrives, it is compared with a randomly selected zombie in the zombie list. A "hit" is declared if the arriving packet's flow matches the zombie. In this case the count of the zombie is increased by one, and the timestamp is reset to the arrival time of the packet in the buffer. A "no hit" is declared if there is no match, and in this case, with probability p the flow identifier of the packet is overwritten over the zombie chosen for comparison.

3.1 SRED – Simulation using OPNET

Queue Size in Packets is shown in this simulation Figure 3. Queue Size mostly varies around 300 Packets.

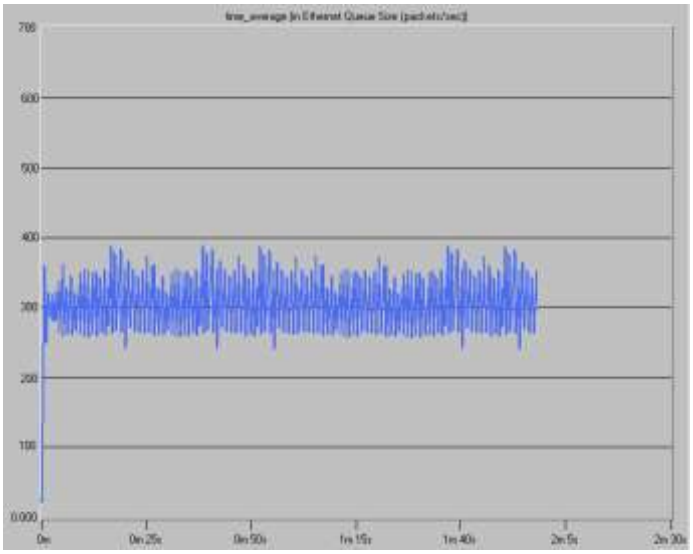

Figure 3: SRED Simulation Result

4. Dynamic RED (DRED)

The DRED algorithm uses a simple feedback control approach to randomly discard packets in the queue. The objective of the DRED algorithm is to stabilize the actual queue size q(n) at a predetermined target queue size T,

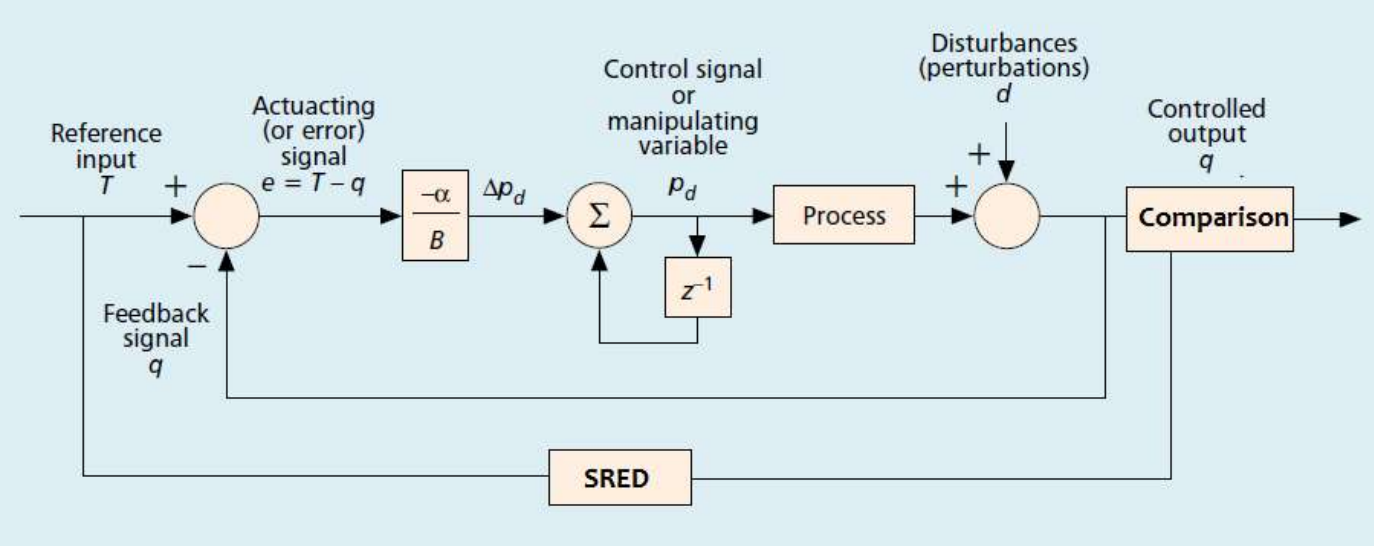

Figure4: Block Diagram of SDRED Algorithm

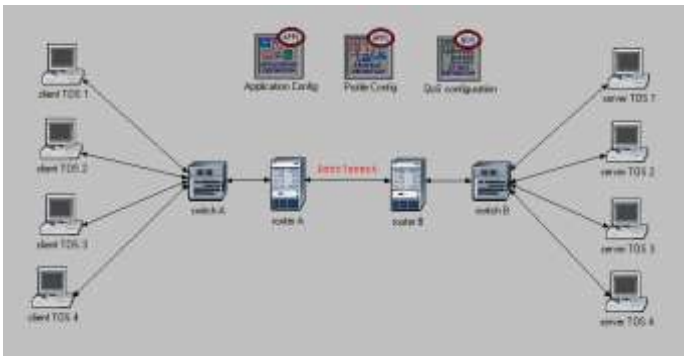

Figure 5: The Network Topology

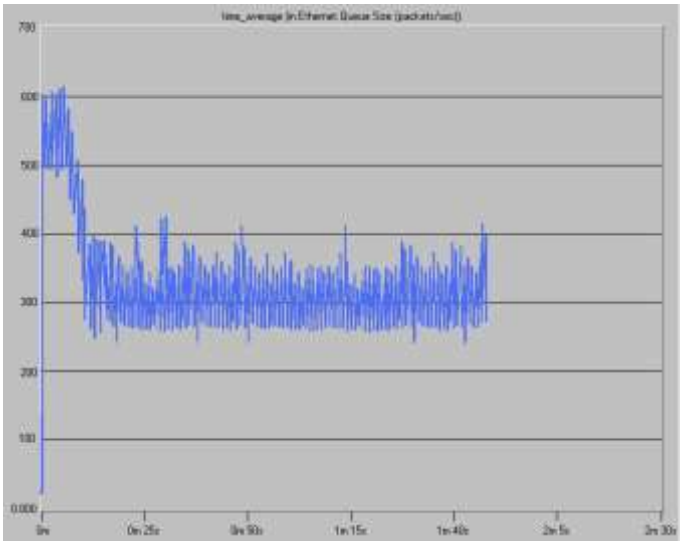

Figure 6: DRED Simulation Result

independent of the network traffic load. The actual queue size is sampled every Dt units of time and used to produce an error signal e(n) = q(n) – T. This error signal is then used to adapt the drop probability Pd, so that e(n) can be kept as small as possible.

The basic strategy in the DRED algorithm controller is to use a first-order low-pass filter with a filter gain b to generate a filtered error signal.

$$\hat{e}(n) = (1 - b)\hat{e}(n - 1) + be(n)$$

The drop probability pd(n) of DRED is adjusted according to the max-min operation of the filtered error signal, the control gain a, and the buffer size B. In order to keep the resource utilization high, DRED does not drop packets when q(n) is less than a minimum threshold L (i.e., the no-drop threshold), which is set to L = 0.9B in this study.

4.1 DRED – Simulation using OPNET

Queue Size in Packets is shown in this simulation Figure 6. Queue Size is around 550 at start up and then mostly varies around 300 Packets.

5. Stabilized Dynamic RED (SDRED)

The SDRED algorithm is a newly proposed algorithm. This algorithm mainy combines the DRED and SRED to form a Stabilized Dynamic RED. The combination of SRED & DRED protocols makes the new protocol take up all the advantages of both the systems. The output of the new system is more efficient. At every instinct the minimum drop probability is calculated based on the feedback system and the threshold system.

$$psdred(q) = \min \{ Pd(n), Psred(q) \}$$

Figure 4 shows the block diagram describing SDRED in detail.

5.1 SDRED – Simulation using OPNET

Queue Size in Packets is shown in this simulation Figure 7. Queue Size is around 500 at start up and then mostly varies around 450 Packets.The Packet drop size is about 50 Packets.

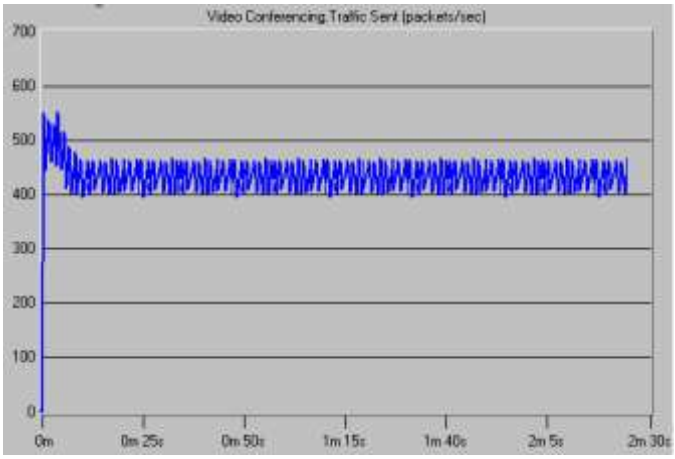

Figure 7: SDRED – Simulation Result

6. Conclusions

This paper shows that the proposed algorithm of SDRED is better than the existing algorithms and the earlier proposed DRED algorithm. This Algorithm is of vital use in Video Conferencing systems as well as for other real time applications. The usage of this algorithm can be widely implemented even in heavy load scenarios. To explain the benefits of this new system, the following comparison table can be taken into light.

SDRED Algorithm	DRED Algorithm
• Has the double benefit of SRED & DRED	• Has only DRED Protocol Implementation.
• The Packet loss is as low as 50Packets	• The packet loss is around 100Packets
• The buffer size used is 475 Packets	• The Buffer Size used is 350Packets.
• Information Loss is Minimized.	• Considerable loss of Information.
• Best suited for Video & Audio Applications where dropped packets can disturb the entire system.	• Best suited for text communication where data losses can be recovered.

7. REFERENCES

[1] R. Braden, D. Clark and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," 2002.

[2] C. Semeria, "Supporting Differentiated Service Classes: Queue Scheduling Disciplines," Vol – 1 , 2000.

[3] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," Vol-12, 1999.

[4] G. Mazzini, R. Rotate, and G. Seta, "A Closed Form Solution of Bernoullian Two- Classes Priority Queue," 200.

[5] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, IEEE Std. 802.11, 1997.

[6] Jacobson, "Congestion Avoidance and Control," Proc. ACM SIGCOMM '88, Aug. 1988, pp. 314–29.

[7] W. Richard Stevens. "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," IETF RFC 2001, Jan. 1997.

[8] B. Braden et al., "Recommendation on Queue Management and Congestion Avoidance in the Internet," IETF RFC 2309, Apr. 1998.

[9] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Trans. Net., vol. 1, no. 4, Aug. 1993, pp. 397–413.

[10] W. Feng et al., "BLUE: A New Class of Active Queue Management Algorithms," Technical Report CSE-TR- 387-99, Dept. EECS, Univ. MI, Apr. 1999.

[11] T. J. Ott, T. V. Lakshman, and L. H. Wong, "SRED: Stabilized RED," Proc. IEEE INFOCOM '99, Mar. 1999, pp. 1346–55.

[12] J. Aweya, M. Ouellette, and D. Y. Montuno, "A Control Theoretic Approach to Active Queue Management," Comp. Net., vol. 36, issue 2–3, July 2001, pp. 203–35.

[13] R. Morris, "Scalable TCP Congestion Control," Proc. IEEE INFOCOM 2000, Tel Aviv, Israel, Mar. 26–30, 2000, pp. 1176–83.

[14] S. Floyd, "TCP and Explicit Congestion Notification," ACM Comp. Commun. Rev., vol. 24, no. 5, Oct. 1994, pp. 10–23.

8. BIOGRAPHY

Mr. Himanshu Chandra, has a B.Tech. in Electronics & Communication Engineering from VIT University, Vellore, Tamil Nadu, INDIA. He has published an IEEE Paper on queuing disciplines in 2009, published a paper on Rural Governance at IIM Calcutta. He has completed a course in Strategic Management at The London School of Economics, U.K. He has won the global social entrepreneurship challenge by University of Minnesota under the Acara Institute and has developed a drinking water solution through the process of Solar Distillation and his design has been filed for Patent. Also he is the Co-Founder of a non government organization "iKENNEX" which works for fostering technological development and scientific thinking in colleges.

Mr. Ajay Agarwal is pursuing B.Tech in Mechanical Engineering at VIT University, Vellore, Tamil Nadu, INDIA. He has undergone extensive training in industries of various sectors specializing in Industrial and Technical Solutions, Trouble Shooting and Consulting. He has designed an Improved Contact Type Coal and Ore Crusher which is a major improvement over the existing exorbitant technology being used in the industry. He has also designed an Ornithopter (a mechanical bird) which is R.F. controlled and has a flight time of over 10 minutes. He is also the Vice President (HR & Technical) of a non government organization "iKENNEX" which works for fostering technological development and scientific thinking in colleges.