Journal of Computer Science 4 (12): 1020-1023, 2008
ISSN 1549-3636
© 2008 Science Publications

Performance Comparison of Active Queue Management Techniques

T. Bhaskar Reddy and Ali Ahammed
Department of Computer Science and Technology,
S.K. University, Anantapur. India

Abstract: Congestion is an important issue which researchers focus on in the Transmission Control Protocol (TCP) network environment. To keep the stability of the whole network, congestion control algorithms have been extensively studied. Queue management method employed by the routers is one of the important issues in the congestion control study. Active Queue Management (AQM) has been proposed as a router-based mechanism for early detection of congestion inside the network. In this study, we are comparing AQM two popular queue management methods, Random Early Detection (RED) and droptail, in different aspects, such as throughput and fairness Index. The comparison results indicate RED performed slightly better with higher throughput and higher fairness Index than droptail. Simulation is done by using Network Simulator (NS_2) and the graphs are drawn using X- graph.

Key words: RED, droptail, fairness index, throughput, Active Queue Management (AQM), NS_2

INTRODUCTION

When there are too many coming packets contending for the limited shared resources, such as the queue buffer in the router and the outgoing bandwidth, congestion may happen in the data communication. During congestion, large amounts of packet experience delay or even be dropped due to the queue overflow. Severe congestion problems result in degradation of the throughput and large packet loss rate. Congestion will also decrease efficiency and reliability of the whole network, furthermore, if at very high traffic, performance collapses completely and almost no packets are delivered.

As a result, many congestion control methods[2] are proposed to solve this problem and avoid the damage. Most of the congestion control algorithms are based on evaluating the network feedbacks[3] to detect when and where congestion occurs and take actions to adjust the output source, such as reduce the congestion window (cwnd). Various feedbacks are used in the congestion detection and analysis. However, there are mainly two categories: Explicit feedback and implicit feedback.

In explicit feedback algorithms, some signal packets are sent back from the congestion point to warn the source to slow down[4], while in the implicit feedback algorithms, the source deduces the congestion existence by observing the change of some network factors, such as delay, throughput difference and packet loss[4]. Researchers and the IETF proposed Active Queue Management (AQM) as a mechanism for detecting congestion inside the network. Further, they have strongly recommended the deployment of AQM in routers as a measure to preserve and improve WAN performance. AQM algorithms run on routers and detect incipient congestion by typically monitoring the instantaneous or average queue size. When the average queue size exceeds a certain threshold but is still less than the capacity of the queue, AQM algorithms infer congestion on the link and notify the end systems to back off by proactively dropping some of the packets arriving at a router. Alternately, instead of dropping a packet, AQM algorithms can also set a specific bit in the header of that packet and forward that packet toward the receiver after congestion has been inferred. Upon receiving that packet, the receiver in turns sets another bit in its next ACK.

When the sender receives this ACK, it reduces it transmission rate as if its packet were lost. The process of setting a specific bit in the packet header by AQM algorithms and forwarding the packet is also called marking. A packet that has this specific bit turned on is called a marked packet. End systems that experience the marked or dropped packets reduce their transmission rates to relieve congestion and prevent the queue from overflowing.

MATERIALS AND METHODS

In this study, we will compare two popular AQM queue management methods, Random Early Detection (RED)[2] and droptail, in different aspects, such as throughput and fairness Index. which we will give the definition first.

Corresponding Author: T. Bhaskar Reddy, Department of Computer Science and Technology, S.K. University, Anantapur

Throughput: The measure of how soon the receiver is able to get a certain amount of data. It is determined as the ratio of the total data received by the end to the connection time. Throughput is an important factor which directly impacts the network performance.

Fairness index: The measure of whether each TCP connection gets a fair share. Fairness Index (FI) is computed as follows: Let T_1 ... T_i... and T_n be the throughput achieved by each of the N TCP connections. Fairness Index can be expressed as:

$$FI = \frac{(\sum_{Ti})^2}{N\sum_{Ti}^{2}}$$

Fairness Index varies from 0 to 1.

Droptail and red: Droptail queuing method is by far the simplest approach to router queue management. The router accepts and forwards all the packets that arrive as long as its buffer space is available for the incoming packets. If a packet arrives and the queue is currently full, the incoming packet will be dropped. The sender eventually. detects the packet lost and shrinks its sending window. Droptail is the most widely used queue manage algorithm due to its simple implementation and relatively high efficiency. However, droptail has some weakness, such as the bad fairness sharing among TCP connections and the throughput and link efficiency suffer severe degradation if congestion is getting worse. Random Early Detection (RED)[2] seeks to prevent the router's queue from becoming fully used by randomly dropping packets and send signals to the sender to slow the sender down before the queue is entirely full. Two parameters govern RED's behavior, REDmin (the lower threshold) and REDmax (the higher threshold). A RED router maintains a notion of the length of the queue. RED routers maintain a running average of their queue length. When the queue length of some line exceeds a threshold, the line is said to be congested and action is taken. A temporary increase in the queue length notifies the transient congestion, while an increase in the computed average queue size reflects longer-lived congestion and RED router will send randomized feedbacks to some of the connections to decrease their congestion windows. The probability that a connection is notified of congestion is proportional to that connections share of the throughput through the RED router[1]. RED has good fairness among connections because of the feedback randomized mechanism and RED is a good congestion avoidance algorithm to ensure the network reliability.

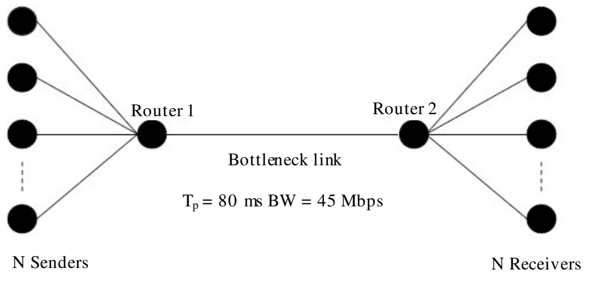

Fig. 1: Simulation topology

Simulation: We use the Network Simulator (NS_2)[5]. TheNS_2 is a discrete event simulator developed by the University of California at Berkeley and the Virtual Intenetwork Tested (VINT) project. The NS_2 support two languages ,system programming languages C++ for detail implementation and scripting languages TCL for configuring and experimenting with different parameters quickly . The NS_2 has all the essential features like abstraction ,visualization ,emulation, traffic and scenario generation .The X-graph draws a graph on a display with data given either from data files or standard input .It can display up to 64 independent data sets using different colors and line styles for each set.

Simulation model: A Simple network topology is chosen to make it easier to understand the congestion network environment. As shown in Fig. 1, there are n connections in the network, n is variable parameter that means how many connections share the bottleneck link.

We choose N with in 2, 4, 8, 12, 16, 20, 24, 32…. The larger is the number of connections, the worse is the congestion in the bottleneck.

RESULTS AND DISCUSSION

Throughput analysis:

Case 1: The average throughput versus the number N with BW = 45 Mbps, T_p = 80 ms, D = 0 and N varying from 2 to 32 in steps: 2, 4, 8, 12, 16, 20, 24, 32.

In Fig. 2 we conclude that Both RED and DROPTAIL performed almost equally. There was almost a steady increase in the performance as the number of nodes increased.

Case 2: The average throughput versus the number N with BW = 45 Mbps, T_p = 250 ms, D = 0 and N varying from 2 to 32 in steps: 2, 4, 8, 12, 16, 20, 24, 32.

In Fig. 3 we conclude that till node 8 they performed almost equally. From node 8 till node 32 RED performed better than droptail.

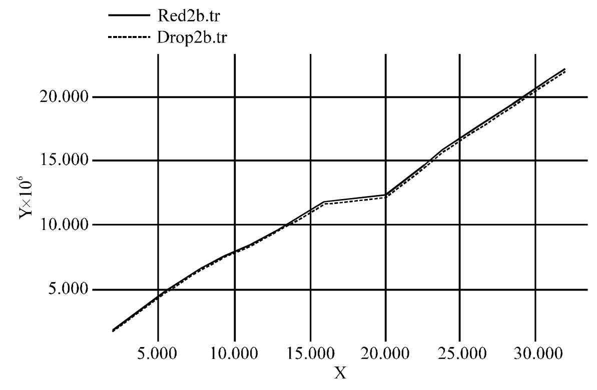

Fig. 2: Node N vs throughput

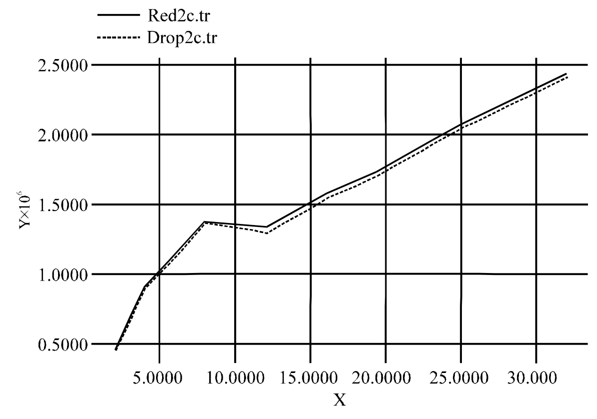

Fig. 3: Node N vs throughput

Table 1: Throughput analysis Case 1

No. of nodes	Red throughput (Mbps)	DROPTAIL throughput (Mbps)
2	2.5	2.5
4	3.5	3.5
8	7.5	7.5
12	8.0	8.0
16	12.0	11.9
20	12.5	12.5
24	17.0	17.0
32	23.0	23.0

Table 2: Throughput analysis Case 2

No. of nodes	Red throughput (Mbps)	DROPTAIL throughput (Mbps)
2	0.40	0.40
4	0.80	0.80
8	1.43	1.43
12	1.40	1.35
16	1.60	1.50
20	1.75	1.70
24	2.20	2.15
32	2.35	2.30

Fairness index analysis:

Case 3: The average fairness index versus D with N = 16, BW = 45 Mbps, T_p = 75 ms and D varying from 10 to 20 in steps: 10, 15 and 20.

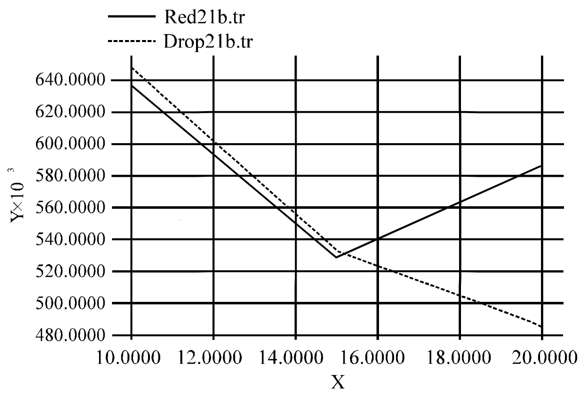

Fig. 4: Delay vs fairness index

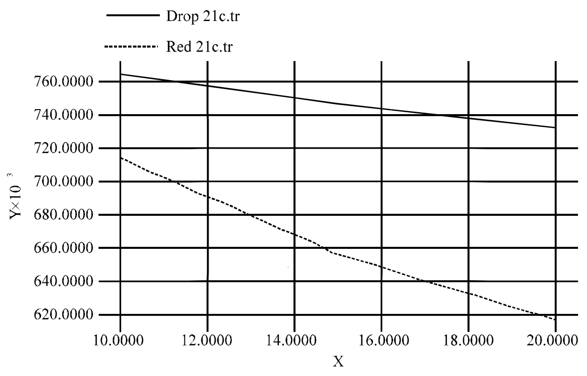

Fig. 5: Delay vs fairness index

Table 3: Fairness index analysis Case 3

Delay	Red	DROPTAIL
10	0.638	0.643
15	0.530	0.532
20	0.585	0.485

Table 4: Fairness index analysis Case 4

Delay	Red	DROPTAIL
10	0.762	0.718
15	0.750	0.655
20	0.738	0.618

In Fig. 4 we can conclude that RED performs better than DROPTAIL. Fairness index of both is less than 0.7 here.

Case 4: The average fairness index versus D with N = 16, BW = 45 Mbps, T_p = 250 ms and D varying from 10 to 20 in steps: 10, 15 and 20.

In Fig. 5 we can conclude that Fairness index of RED is better all the three cases here. Fairness index is never greater than 0.8 here.

CONCLUSION

This study presents a Comparison of two widely used queue management mechanism RED and droptail in several aspects. We design experiments to simulate the queue management techniques and analyze the throughput and fairness. By the comparison we shown that RED performed slightly better with higher throughput and higher fairness Index than droptail.

REFERENCES

1. Floyd, S., R. Gummdi and S. Shenker, 2001. Adaptive RED: Analgorithm for increasing the robustness of RED. https://eprints.kfupm.edu.sa/22933/.
2. Bonald, T. *et al*., 2000. Analytic evaluation of RED performance. Proceedings of the 19th Annual Joint Conference on the IEEE Computer and Communications Societies, Mar. 26-30, IEEE Xplore Press, USA., pp: 1415-1424. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=832539.
3. Usha, M. and R.S.D. WahidaBanu, 2005. Towards a simple service class for distributed multimedia applications in next generation IP networks. Proceeding of the ISCA 18th International Conference on Parallel and Distributed Computing Systems, Sep. 12-14, IEEE Xplore Press, Las Vegas, USA., pp: 161-166. http://dblp.uni-trier.de/rec/bibtex/conf/ISCApdcs/2005.
4. Usha, M. and R.S.D. Wahida Banu, 2006. Enhanced quality of service through jitter control for next generation service computing. Proceeding of the National Conference NCAC, Chennai, India, February 2006, pp: 225-232.
5. ns-2 Simulator, http:Hwww.isi.edu/nsnam/ns.