B. Kiruthiga *et al*, International Journal of Computer Science and Mobile Applications,
Vol.2 Issue. 2, February- 2014, pg. 38-44 **ISSN: 2321-8363**

Survey on AQM Congestion Control Algorithms

B. Kiruthiga[1], Dr. E. George Dharma Prakash Raj[2]

[1]School of Computer Science and Engineering, Bharathidasan University, Trichy, India

[2]School of Computer Science and Engineering, Bharathidasan University, Trichy, India

[1] kiruthigabaskaran13@gmail.com

[2] georgeprakashraj@yahoo.com

Abstract

Congestion is an important issue which researchers focus on in the Transmission Control Protocol (TCP) network environment. Active queue management (AQM) has been proposed as a router-based mechanism for early detection of congestion inside the network. This paper analyzed several active queue management algorithms are discussed and they are compare with respect to their.

Keywords: Network; Queue; Congestion; AQM

1. Introduction

Congestion in Internet occurs when a large volume of data is being routed on low bandwidth lines or across networks that have high latency and cannot handle large volumes. The result is slowing down of packet movement, packet loss and drop in service quality.

Queue Management in routers plays an important role in taking care of congestion. Two approaches are adopted to solve this problem. First one is Congestion Avoidance preventive technique, which comes into play before network is congested by overloading. Second is Congestion Control, which comes into play after congestion at a network has occurred and the network is overloaded.

2. Active Queue Management

The essence of Internet congestion control is that a sender adjusts its transmission rate according to the congestion measure of the networks. In internet routers, active queue management is the controller. The task is performed by the network or packet scheduler.

3. AQM algorithms

3.1 RED (Random Early Detection)

RED algorithm for RED Gateways was first of all proposed by Sally Floyd and Van Jacobson. RED is an active queue management scheme that provides a mechanism for congestion avoidance. Unlike traditional congestion control schemes that drop packets at the end of full queues, RED uses statistical methods to drop packets in a "probabilistic" way before queues overflow. Dropping packets in this way slows a source down enough to keep the queue steady and reduces the number of packets that would be lost when a queue overflows and a host is transmitting at a high rate.

RED makes two important decisions. It decides when to drop packets and what packets to drop. RED keeps track of an average queue size and drops packets when the average queue size grows beyond a defined threshold. The average size is recalculated every time a new packet arrives at the queue.

RED uses time-averaging meaning that if the queue has recently been mostly empty, RED will not react to a sudden burst as if it were a major congestion event. However, if the queues remain near full, RED will assume congestion and start dropping packets at a higher rate.

Pseudo code for RED algorithm is as follows,

```
For every packet arrival {
Calculate Qave
if (Qave ≥ maxth) {
Drop the packet
}
else if (Qave > minth) {
Calculate the dropping probability pa
Drop the packet with probability pa,
otherwise forward it
}
else {
Forward the packet
}
}
```

Variables

Qave : average queue size
pa : current packet-marking probability
q : current queue size
pb : temporary marking or dropping probability

Fixed parameters

wq : queue weight - 0.1 ~ 0.0001
maxth : maximum threshold for queue
minth : minimum threshold for queue
maxp : maximum dropping probability

3.2 FRED (Flow based Random Early Detection)

FRED algorithm was developed by Lin and Morris. FRED is a modified version of RED, which uses per-active-flow accounting to make different dropping decisions for connections with different bandwidth usages. FRED only keeps track of flows that have packets in the buffer, thus the cost of FRED is proportional to the buffer size and independent of the total flow numbers.

FRED has some interesting features than the RED. They are (i) penalizing non-adaptive flows by imposing a maximum number of buffered packets and surpassing their share to average per-flow buffer usage. (ii) Protecting fragile flows by deterministically accepting flows from low bandwidth connections. (iii) Providing fair sharing for large numbers of flows by using "two packet-buffers" when buffer is used up. (iv) Fixing several imperfections of RED by calculate average queue length at both packet arrival and departure (which also causes more overhead).

Two parameters introduced into FRED: minq and maxq, which are minimum and maximum number of packets that each flow is allow to buffer. FRED uses a global variable avgcq to estimate it. The average per active-flow buffer usage. It maintains a count of buffer packets qlen.

Pseudo code for FRED algorithm is as follows,

```
For each arriving packet P:
Calculate average queue length
Obtain connection ID of the arriving packet: flowi connectionID(P)
If flowi has no state table then
Qleni = 0
Strikei = 0
End if
Compute the drop probability like RED: p maxp
Maxth-avg
Maxh-minth
Maxq minth
If(avg_maxth) then
Maxq 2
End if
If (qlen_maxqll(avg_maxth && qleni>2_avgcq)ll(qleni_avgcq && strikei>1)) Then
ikei = strikei + 1
Drop arriving packet and return
End if
If (minth_avg<maxth) then
If (qleni_max(minq,avgcq))then
Drop packet P with a probability p like RED
End if
Else if (avg<minth) then
Return
Else
Drop packet P
Return
End if
If (qleni==0) then
Nactive = Nactive + 1
End if
Enqueue packet P
```


For each departing packet P:
Calculate average queue length
If (qleni==0) then
Nactive = Nactive – 1
Delete state table for flow i
End if
If (Nactive) then
Avgcq =avg/Nactive
Else
Avgcq=avg
End if

3.3 BLUE

Blue is another extension of RED developed by Wu-Chang and Feng which uses packet loss and link utilization (rather than queue size) as a control variables to measure the network congestion. It maintains a marking probability. If the queue is continually dropping the packets, increments the marking probability. If the queue become empty or idle, decreases the marking probability.

Pseudo code for BLUE algorithm is as follows,
Upon Packet loss (or Qlen > L) event:
if ((now – last_update) > freeze_time)
pm := pm + δ1
last_update := now
Upon link idle event:
if ((now – last_update) > freeze_time)
pm := pm - δ2
last_update := now
where
pm: Marking/Dropping probability
δ1: Amount of increase by pm
δ2: Amount of decrease by pm
now: Current time
last_update: last time pm was changed
freeze_time : minimum time period between two consecutive updates of pm

3.4 AdaptiveCHOKe

AdaptiveCHOKe enforces the concept of queue-based and flow information. It is desirable for AQM schemes to act without storing a lot of information otherwise it becomes a overhead and non-scalable. This algorithm modifies the CHOKe algorithm to remove its drawback. This algorithm also calculates the average queue size of the buffer for every packet arrival. It also indicates two thresholds on the buffer, a minimum threshold minth and a maximum threshold maxth. It reduces both the packet loss rate and the variance in queuing delay.

The pseudo code for AdaptiveCHOKe is as follows,

```
For every packet arrival {
Calculate Qavg
If (Qavg < minth) {
Forward the new packet
}
Else
{
Select randomly a packet from the queue for their flowid
Compare arriving packet with a randomly selected packet.
If they have same flowid {
Drop both the packet
}

Else {
If (Qavg>=maxth) {
Calculate the dropping probability Pa
Drop the packet with probability Pa
}
Else
{
Drop the new packet
}
}
}
}
```

Variables

Qavg : average queue size
Pa : current packet-marking probability
Q : current queue size
Pb : temporary marking or dropping probability
Wq : queue weight
Maxp : maximum dropping probability

Fixed Parameters

Minth : minimum threshold for queue
Maxth : maximum threshold for queue

The average queue size is compared with these thresholds for every arriving packet. If average queue size is less than minth, every arriving packet is queued. If average queue size is greater than maxth, every arriving packet is dropped. This results in queue size below maxth. When the average queue size is greater than minth, every arriving packet is compared with a randomly selected packet from the queue for their flowid. If they have the same flowid, both are dropped. Otherwise the randomly selected packet is placed in the same position in the buffer. In this algorithm, the arriving packet is dropped with a probability depending on the average queue size.

AdaptiveCHOKen which aims to protect well-behaved flows from misbehaving flow and adaptive flows from non-adaptive flows. It also obtains high utilization, low queuing delay and low packet loss with well adaptively tuned parameters.

3.5 P-CHOKe (PiggybackingCHOKe)

P-CHOKe (PiggybackingCHOKe) enforces the concept of queue-based and flow information. The following steps are involved in P-CHOKe algorithm,

Step1: Start Procedure
Step2: For every packet arrival
Step3: Forward the new packet to the gateway
Step4: Select randomly a packet from the queue for their flowid
Step5: The gateway compare arriving packet with a randomly selected packet
Step5: If there is equal flowid
Step6: Drop the packet
Step7: Otherwise forward the packets to the receiver
Step8: The receiver sends the ack to the gateway
Step9: Gateway collects all those acknowledgments
Step10: Gateway sends the ack to the receiver based on the flowid
Step11: End Procedure

P-CHOKe (PiggybackingCHOKe) enforces the concept of queue-based and flow information. In this P-CHOKe algorithm the n senders sends the packets to the gateway or router. The router collects all the packets and then it sends it to the receiver. Then the receiver sends acknowledgement to the router, it collects all the acknowledgements and then sends to the receiver based on their flowid. P-CHOKE obtains high Packet delivery Ratio, low queuing delay, high throughput, and less process time.

4. Advantages and Disadvantages of AQM Algorithms

s.no	Algorithms	Advantages	Disadvantages
1	RED	Early congestion detection. No bias against bursty traffic. No global synchronization.	Difficulty in parameter setting. Insensitivity to traffic load and drain rates.
2	FRED	Good protection from misbehaving flows.	Per – flow state. Difficulty in parameter setting. Insensitivity to traffic load and drain rates.
3	BLUE	Easy to understand. High throughput.	No early congestion detection. Slow response.
4	A-CHOKe	To protect well-behaved flows from misbehaving flow and adaptive flows from non-adaptive flows. It also obtains high utilization, low queuing delay and low packet loss with well adaptively tuned parameters.	Heavy load and unresponsive flow.
5	P-CHOKe	It provides better packet delivery ratio. It provides the low queue delay.	Fair bandwidth allocation.

5. CONCLUSION

This paper briefly surveys comparative analysis of different congestion control algorithms. The AQM algorithms are classified based on congestion metrics and the flow information. Most of the AQMs only require congestion indicators while some of them require both congestion indicator and flow information. Very few require only flow information for detecting congestion. These AQMs are compared based on the various performance metrics. This paper tries to project the desirable quality and shortcoming that exists in each of the algorithm in terms of their performance.

References

[1] S.Dijkstra, "Modeling Active Queue Management Algorithms Using Stohastic PetriNets", Master Thesis, 2004.
[2] G.Thiruchelvi and J.Raja, "A Survey on Active Queue Management Mechanisms", IJSCNS, Vol.8, No.12, December 2008.
[3] Shakeel Ahmad, Adli Mustafa, Bashir Ahmad, Arjamand Bano and Al-Sammarraie Hosam, " comparative study of congestion control techiniques in high speed networks", IJCSIS, Vol.6, No.2, 2009.
[4] K.Chitra and Dr.G.Padmavathi, "Adaptive CHOKe: An Algorithm to Increase the Fairness in Internet Routers", International Journal on Advanced Networking and Applications, Vol: 01, Issue: 06, 2010.
[5] G.F.Ali Ahammed and Reshma 3Banu, "Analyzing the Performance of Active Queue Management Algorithms", IJCNC, Vol:2, No:1, 2010.
[6] Dr.G.Padmavathi and K.Chitra, "Classification and Performance of AQM – Based Schemes for Congestion Avoidance", IJCSIS, Vol: 8, No: 1, 2010.
[7] Alireza Gharegozi, "Greedi Flow Control by FRED Active Queue Management Mechanism", IPCSIT, Vol:7, 2011.
[8] Aos Anas Mulahuwaish, Kamalrulnizam Abu Bakar, Kayhan Zar Ghafoor, "A Congestion Avoidance Approach in Jumbo Frame – Enabled IP Network", IJACSA, Vol: 3, No: 1, 2012.
[9] Alshimaa H.Ismail, Zeiad Elsagheer, and I.Z.Morsi,"Survey on Random Early Detection Mechanism and Its Variants", IOSRJCE, vol.2, Issue.6, Aug-2012.
[10] G. Sasikala, E. George Dharma Prakash Raj, "P-CHOKe: A Piggybacking-CHOKe AQM Congestion Control Method", IJCSMC, Vol. 2, Issue. 8, August 2013.

