DESIGN AND PERFORMANCE OF AQM MODEL BASED NETWORK

Sushil Kumar Sharma*

1. 0 INTRODUCTION

Congestion control is a very active research area in network community. In order to supply the well known transmission control protocol (TCP), active queue management mechanisms have been developed. AQM regulates the queue length of a router by actively dropping packets. Various mechanisms have been proposed in the literature such as Random Early Detection (RED) , Random Early Marking (REM) , Adaptive Virtual Queue (AVQ) and many others .

Increasing access to data communications is creating sweeping changes around the globe. More people are using a wider range of services, requiring more data to be transported. As a result, there has been a surge of interest in designing low-loss and low delay networks by encouraging users to adapt to changing networks conditions using minimal information from the network. Ultimately the performance of a communications network will be judged by the Quality of Service (QoS) perceived by users. This end user QoS can be affected by many factors outside of the control of the network operators. For example, the quality of an MPEG-1 video stream, which has very limited error suppression capabilities, will be perceptibly lower than that of a video service with error suppression and correction capabilities built-in after both streams have been transported across an inherently lossy network like the Internet. The differences in the perceived quality will obviously be affected by the performance of the transport network, but the main differences will come from the nature of the service being transported . Aggregate queuing delay or latency is the amount of time it takes the senders packet, once it enters the network, to be delivered to its destination. A packet is a form of data in a sequence of binary digits in a packet-switched network. Transmission Control Protocol (TCP) divides a file into efficient sized packets that are separately numbered. A simple packet contains an IP header, packet number protocol, source address (TCP sender), destination Internet address, and data. The TCP sends the packet into the network where it is then routed through various links and hubs. Each link is connected to a hub server that routes packets through routers and switches that have a queue size and possibly a different algorithm for handling queue congestion.

The hub server's router and switches take input links and route the packets to the appropriate output link. When a packet enters a hub server, it will be placed into a routers queue when there is congestion. If there is no congestion the packet will be sent immediately with virtually zero delay. Depending on the length of the queue and where the packet was placed in the queue determines the amount of the time it will reside in queue until it is sent. Once all the packets have arrived to its destination, the TCP receiver will reassemble the file in the receiver TCP by putting the packets in-order and assembling the data back into a file.

*Assistant Professor, Government Post Graduate College, Ambala Cantt

1.1 Network based paradigm

Web servers, have now become an integral part of our information services infrastructure. Controlling the timing performance of each individual connection to a network server is a challenging theoretical problem with important practical implications. A network server's response to allocated resources is highly non-linear. In addition, the workload is stochastic and its parameters could change abruptly over a wide range of values. The differential equation model used by control theory does not model Web servers well, except in the limited case of heavy workload that allows for fluid approximations. In spite of the tradition of using differential equation models for network servers, we believe that the key to success is not to force-fit a queuing system into a differential equation model. The controller corrects errors due to the inaccuracies in the queuing model for the real networked server system. To integrate queuing theory with control, we need to develop a general procedure that will allow us, based on experimental data alone; to get an accurate linear model for residue errors from any queuing model based feed forward control. However, some of the best practices for identifying models for physical systems are actually counter-productive.

2.0 AQM ALGORITHMS

Few commonly used AQM Algorithms which are commonly used in congestion environment are :

2.1 DROP TAIL

The traditional technique for managing router queue lengths is to set a maximum length (in terms of packets) for each queue, accept packets for the queue until the maximum length is reached, then reject (drop) subsequent incoming packets until the queue decreases because a packet from the queue has been transmitted. This technique is known as 'drop tail', since the packet that arrived most recently (i.e., the one on the tail of the queue) is dropped when the queue is full. This method has served the Internet well for years, but it has two important drawbacks:

- Lock-Out: In some situations drop tail allows a single connection or a few flows to monopolize queue space, preventing other connections from getting room in the queue. This "lock-out" phenomenon is often the result of synchronization or other timing effects.
- Full Queues: The drop tail discipline allows queues to maintain a full (or, almost full) status for long periods of time, since tail drop signals congestion (via a packet drop) only when the queue has become full. It is important to reduce the steady state queue size, and this is perhaps queue management's most important goal.

2.2 THE RED (RANDOM EARLY DETECTION) ALGORITHM

One of the biggest problems with TCP's congestion control algorithm over drop tail queues is that sources reduce their transmission rates only after detecting packet loss due to queue overflow. Since a considerable amount of time may elapse between the packet drop at the router and its detection at the source, a large number of packets may be dropped as the senders continue transmission at a rate that the network cannot support

RED alleviates this problem by detecting incipient congestion *early* and delivering congestion notification to the end-hosts, allowing them to reduce their transmission rates before queue over- flow occurs. Van Jacobson and Sally Floyed first introduced the RED algorithm in August of 1993. RED has been designed with the objective to minimize packet loss and queuing delay, avoid global synchronization of sources, maintain high link utilization and remove biases against bursty sources. One way to solve this problem is to use a large amount of buffer space at the RED gateways. For example, it has been suggested that in order for RED to work well, an intermediate router requires buffer space that amounts to twice the bandwidth-delay product. This approach, in fact, has been taken by an increasingly large number of router vendors. Unfortunately, in networks with large bandwidth-delay products, the use of large amounts of buffer adds considerable end-to-end delay and delay jitter. This severely impairs the ability to run interactive applications. In addition, the abundance of deployed routers which have limited memory resources makes this solution undesirable.

2.3 THE BLUE ALGORITHM

The BLUE algorithm resolves some of the problems of RED by employing the use of hybrid flow control scheme along with a queue size congestion measuring scheme. It uses flow and queue events to modify the congestion notification rate. This rate is adjusted by two factors: packet loss from queue congestion and link utilization or underutilization. A key difference the BLUE algorithm has from RED, is that uses packet loss rather than the average queue length , If the queue is continually dropping packets due to buffer overflow, BLUE thus increasing the rate at which it sends back congestion notification or dropping packets. Conversely, if the queue becomes empty or if the link is idle, BLUE decreases its marking probability. This effectively allows BLUE to “learn” the correct rate it needs to send back congestion notification or dropping packets.

2.4 REM (RANDOM EXPONENTIAL MARKING) ALGORITHM

REM aims to achieve both high utilization and negligible loss and delay in a simple and scalable manner. The key idea in achieving this is to decouple congestion measure from performance measure such as loss, queue length or delay. While congestion measure indicates excess demand for bandwidth and must track the number of users, performance measure should be stabilized around their targets independently of the number of users. REM that has the following key features :

- It attempts to match user rates to network capacity while clearing buffers (or stabilize queues around a small target), regardless of the number of users.
- The *end-to-end* marking (or dropping) probability observed by a user depends in a simple and precise manner on the *sum* of link prices (congestion measures), summed over all the routers in the path of the user.

CONCLUSION

Finally the major advantages of an active queue management mechanism can be summarized as follows:

- Reducing number of packets dropped in routers: Keep average queue size small, hence leaving enough space for bursts.
- Providing lower-delay interactive service: By keeping average queue size small, end-to-end delays will be shorter.
- Avoiding lock-out behavior: Avoid bias against low bandwidth and bursty flows. Guarantee that a newly arriving packet 'almost always' finds a place in the buffer.

We need to rebuild the infrastructure that will enable us to deal with congestion control and preserve the interest of sender in a better way by implementing Active Queue Management Algorithms.

REFERENCES

1. **G.F.Ali Ahammed, Reshma Banu,** Analyzing Performance of AQM

2. **Saad Biaz and Nitin Vaidya,** De-randomizing"Congestion Losses to Improve TCP Performance over Wired-Wireless Networks" Proc. of IEEE Global Telecommun. Conf.

3. **S. Floyd and V. Jacobson,** "Random early detection gateways for congestion avoidance", IEEE/ACMTransactions on Networking, vol. 1, pp 397-413, Aug, 1993.

4. **Serhat ÖZEKES,** EVALUATION OF ACTIVE QUEUE MANAGEMENT ALGORITHMS

5. **Athuraliya S., Lapsley D. E., Low S. H., (2001)**, "Random early marking for Internet congestion control", IEEE/ACM Transactions on Networking, Vol. 15, No:3, 48-53

6. **Lin D., Morris R., (1997)**, "Dynamics of Random Early Detection", In Proc. of ACM SIGCOMM, 127-137

7. **Feng W., Kandlur D., Saha D., Shin K. , (1999/a),** "A Self-Configuring RED Gateway", In Proc. IEEE INFOCOM, 1320–1328

8. **Yassine Ariba, Yann Labit and Frederic Gouaisbaut** , Design and performance evaluation of State Space based AQM