More Adaptive Explicit Congestion Notification (MAECN) -
AECN with Dynamic Drop Probability

M.S. Thesis Proposal

Choong-Soo Lee

Advisors
Mark Claypool and Bob Kinicki

Reader

Computer Science Department
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609

September 25, 2001

Abstract

Increase in Internet traffic has increased the importance of congestion control and fairness. Active queue management, such as Random Early Detection (RED) seeks to improve congestion and fairness. However, RED serves flows unfairly when there are many TCP flows with various roundtrip times (RTT). Adaptive Explicit Congestion Notification (AECN) is a direct extension of RED to address this issue by classifying flows into three classes: robust, average and fragile, based on their RTTs. We propose an extension to AECN called MAECN (More Adaptive ECN) to improve AECN to handle more diverse traffic mixes. We believe that MAECN will serve various types of traffic more fairly than AECN. We will measure, analyze and compare MAECN with other active queue management schemes through simulations.
TCP, the most widely used transport protocol on the Internet, detects dropped packets as an indication of congestion. TCP uses the AIMD (Additive Increase, Multiplicative Decrease) scheme, where a dropped packet causes the flow to roughly halve its data rate. A source hint is an idea for an end host to add information to packets so that routers can use it to provide better fairness and congestion control. ECN (Explicit Congestion Notification) [Fl94] is an extension to TCP and uses a source hint to allow congestion notification by a router with both marking and dropping of packets.

Active queue management takes advantage of cooperative end-to-end protocols. RED (Random Early Detection) [FJ93], a popular active queue management mechanism, monitors the outgoing link queue and keeps track of a weighted average of the queue size. RED calculates a drop probability for an incoming packet based on the average queue size, a minimum threshold, a maximum threshold and a drop probability line. However, RED is not an all-around solution for active queue management. It does not work well for mixtures of different types of flows [Zh01]. For example, if incoming traffic contains non-responsive flows like UDP, RED does not have a mechanism to protect responsive flows. Similarly, if responsive flows have different round-trip times, RED cannot protect flows that have higher latency. Figure 1 shows goodput (the total throughput without the retransmission) for 10 robust flows with 50ms latency, 10 average flows with 100ms latency and 10 fragile flows with 200ms latency in a RED router. The y-axis shows the total goodput of each type of flow, which is total throughput without the retransmissions. The x-axis shows the time at which the goodput is measured. Clearly, the robust flows dominate the average and fragile flows.

![RED](image)

**Figure 1**: Unfairness in Goodput for Heterogeneous Flows in RED
In order to correct some of these problems, AECN (Adaptive Explicit Congestion Notification) [Zh01] was developed as an extension to RED. AECN makes three major modifications to RED. First, AECN marks packets, as does ECN [Fl94], instead of dropping them. Second, AECN keeps three virtual queues for three types of traffic: robust, average and fragile, based on the roundtrip times of each flow. AECN assumes that the roundtrip time source hint is provided by the end hosts, where each flow tags the calculated roundtrip time on all outgoing IP packets. Third, AECN implements mark front [LJ01] which marks the packet at the beginning of the queue instead of the one at the end.

Through these modifications, AECN improves RED in the following ways. First, marking packets instead of dropping them allows cooperative flows to use marked packets as congestion notification. This improves goodput because there are fewer dropped packets, leading to fewer retransmissions. Second, classification of traffic assists AECN to adjust the dropping probability for each class to assist fragile flows and restrain the robust flows. Third, once AECN decides it needs to mark a packet, AECN picks the first packet of the same class at the front of the virtual queue instead of choosing the packet that just came in, speeding up congestion notification. Figure 2 depicts the same run as in Figure 1, except the router is AECN instead of RED. Note each type of flows gets about the same bandwidth.

Figure 2 : AECN treats heterogeneous flows more fairly than RED.

AECN’s improvements solve some problems of RED but not all of them. Over the Internet, traffic going through a router is bound to change. It might be composed of only robust traffic at one time and might have only fragile traffic at another. AECN is less effective when the traffic
mixture changes, AECN also fixes the parameter on how much to assist fragile flows and how much to restrain robust flows, independent of the degree of robustness or fragileness. Figure 3 depicts the goodput of each type of flow in another AECN run where the number of each type of flow varies over time and the latency varies. The robust flows have a latency of 20ms, the average ones 60ms and the fragile ones 200ms. Here, the robust and average flows are more robust due to lower latencies. For the first 90 seconds, all three types of flows are on, then for the next 90 seconds, the robust ones turn off. Then for the next interval, the robust flows turn back on while the average flows turn off. In the last interval, the average flows turn back on and the fragile flows turn off. Due to the fixed nature of AECN, the parameters that worked for the previous situation no longer work well to treat flows fairly.

Figure 3: AECN cannot adjust to treat different mixtures of flows fairly.

**Approach**

To further improve AECN, we propose MAECN (More Adaptive ECN). AECN used a heuristic instead of a mathematical basis to assist fragile flows and restrain robust ones. For MAECN, we will use the TCP response function as our basis for computing drop probabilities in order to get a better mechanism for dealing with different flows. For example, one way of simplifying the equation is to work out the relationship between drop probability and roundtrip time. The following equation is the result of simplifying the TCP response function from [FHPW00]:

\[ T = \frac{c}{R p^s} \]
The simplification leaves the sending rate of the flow \( T \) as a function of the roundtrip time \( R \), the loss-rate \( p \), and constants \( a \) and \( c \) to compensate for what is eliminated. Because of the simplification, we do not know the exact values of \( a \) or \( c \). So, we take a step further to derive the relative relationship between two pairs of drop probabilities and roundtrip times:

\[
\begin{align*}
p &= \begin{cases} 
  p_{\text{base}} \left( \frac{R_{\text{base}}}{R} \right)^\alpha, & R < R_{\text{base}}, \alpha \geq 0 \\
  p_{\text{base}} \left( \frac{R_{\text{base}}}{R} \right)^\beta, & R \geq R_{\text{base}}, \beta \geq 0 
\end{cases}
\end{align*}
\]

The reason for separating equations into two regions is that we have reason to believe that the treatment for robust and fragile flows should not be symmetrical. The above derivation is one method we would like to try to improve performance, but there may be other ways to use the TCP response function for a better treatment.

Since the function will be relative as shown above, we need to determine \( p_{\text{base}} \) and \( R_{\text{base}} \). We will set \( p_{\text{base}} \) to be equivalent to \( \max_p \), a parameter in both RED and AECN. In an effort to make MAECN as dynamic as possible, \( R_{\text{base}} \) will be a weighted average of roundtrip times of the incoming flows under the assumption that the flows will include their roundtrip times in the outgoing packets.

We hypothesize MAECN improve AECN by eliminating the need to keep the virtual queues that AECN keeps. Furthermore, there are not just three classes of traffic but infinitely many, in the sense that each packet will be treated differently according to its roundtrip time. Keeping a weighted roundtrip time will help MAECN to better determine the nature of incoming traffic.

**Evaluations**

We will test the performance of MAECN through simulations and compare it to RED and AECN. Network Simulator 2 (NS), developed by the University of California at Berkeley, will be used to run the simulations.

First, we will implement MAECN into NS. NS already supports RED and ECN, and AECN was implemented in NS by Zici Zheng at WPI [Zh01]. Then, we will design network topologies and scenarios with many different types of flows. In particular, we will measure and compare the performances of MAECN, RED and AECN, analyze the results, and attempt to further improve the approach. Specifically, we are interested in fairness, throughput and drop statistics.

If this thesis is successful, our contribution will include the design, implementation and evaluation of an active queue management scheme that dynamically adjusts its parameters to improve fairness while preserving throughput under many network conditions and traffic mixes.
Extensions

Time permitting, we will also investigate how more source hints can assist active queue management to treat the incoming traffic better. MAECN takes advantage of one source hint, roundtrip time. One other source hint that is of interest to us is the rate of the flow. Since fairness is how equally the bandwidth is distributed to flows, the flow rate might be a good tool to improve active queue management.

[FS01] provides an algorithm to adapt $max_p$ based on the average queue size. Applying this to MAECN’s $p_{base}$ will make MAECN more adaptive, not only to assist and restrain flows but also to keep the average queue size at around the center between $minthresh$ and $maxthresh$ instead of at the bounds.

Timeline

September ’01 ~ October ’01 : Implementation of MAECN for NS, design of simulation topologies and scenarios
November ’01 ~ January ’02 : Simulations, data collection and analysis
February ’02 ~ April ’02 : Thesis write-up and presentation

Reference