A new robust adaptive controller under dynamic network environment

Qing Li, Zhu Qing-xin, and Wang Ming-wen

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China
qingli_new@163.com

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China
qxzhu@uestc.edu.cn

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China
sohuwmw@sohu.com

Abstract: To suppress the disturbance of unresponsive flows including HTTP and UDP connections, a novel AQM scheme, called RA-AQM, is proposed in this paper. RA-AQM is an adaptive controller for congestion control, and it can adjust its control parameters according to the network state. Simulation results show that the RA-AQM is robust and stable against traffic load fluctuations and unresponsive flow disturbances. RA-AQM can quickly regulate queue length to the target value, give fast response and keep the oscillation small.

Keywords: Congestion Control, Active Queue Management, Robust Adaptive Controller

1. INTRODUCTION

Currently, many active queue managements (AQM) have been proposed to deal with the performance degradation of network caused by congestion. Because it is a more effective feedback strategy than Drop-Tail, AQM is deployed on routers in addition to the TCP at end hosts. AQM enhances routers to detect and notify end-systems of impending congestion earlier by dropping or marking (ECN mechanism [1]) packets before buffer overflow. Together with TCP, AQM can improve network performance in terms of delay, link utilization, packet loss rate, and system fairness.

In [2], the Internet Engineering Task Force (IETF) recommends the deployment of AQM on routers, especially random early detection (RED) [3]. However, it is difficulty to tune RED parameters for different network environments. Many schemes are proposed to improve RED. In [4-5], control theory is introduced to study the behavior of RED and design new AQM schemes. It is believed that AQM should be able to stabilize the queue length on a target value. In [6], PI controller is proposed to realize AQM better. PI controller introduces an integral factor to reduce the steady state error, but it has sluggish behavior. Later, PD [7] and PID [8] are designed to improve the response to congestion, while increasing the oscillation of queue
length. All of these controllers are statically deployed. They are degraded by the variation of network parameters, such as the number of TCP flows and Round Trip Time (RTT). The disturbances by unresponsive flows such as UDP and HTTP connections also seriously affect the performance of AQM controller [9]. Thus there always exists tradeoff between responsiveness and stability in the AQM with static parameters. That is, faster response will lead to lower stability, and vice versa.

Recently many adaptive AQM schemes are proposed to enhance the performance. Among them we mention API [10] R-PI [11], S-PI [12] and STPI [13]. STPI is based on the estimation of network bandwidth and number of TCP flows. API, R-PI and S-PI adjust the gain of PI controller in order to quicken the response to the bursty flows. The improvement of response for AQM achieved only for the variation of long-lived responsive flows. When the unresponsive flows exist, the queue length also shows heavy oscillation.

In this paper, we consider the designing of an adaptive AQM that can work well when exists the disturbance of unresponsive flows. A new controller is proposed based on optimal control theory in frequency field. The parameters of new controller can be set directly. As shown in [13], the link capacities experienced by long-lived TCP flows are quite variable. The long-lived TCP workload is also time varying, and companying with the variety of RTT. Following the approach in [13], we add a self-tuning structure to AQM and obtain a new AQM, called Robust Adaptive AQM or RA-AQM in short, to deal with the variation in link-capacity and TCP workload. The self-tuning structure is used to estimate link-capacity and dominant RTT of long-lived TCP flows online. Simulations show that RA-AQM is robust and its performance is better than other controllers.

The rest of the paper is organized as follows. In Section 2, we give some motivations of RA-AQM. Section 3 discusses the design of RA-AQM. In Section 4, we give some simulations of RA-AQM using ns-2 [14], and compare its performance with other controllers including PI, ARED, and STPI. Finally, we give conclusions in Section 5.

2. MOTIVATIONS FOR A ROBUST ADAPTIVE AQM

Network environment is very complex in practice, and many network parameters such as RTT, traffic load and link capacity are time varying. The designing of AQM controllers such as PI, RED, etc., all based on the hypothesis that traffic flows are responsive to the signal provided by AQM in routers. However, unresponsive flows such as HTTP and UDP connections are bound to occur in real network. Such AQM controllers perform poorly in the presence of network parameters variations and unresponsive flows. To show this we give some simulation results below for some representative AQM controllers using ns-2.

We will consider the situations where TCP flows vary and keep static all the time. The

![Figure 1. Simulation network topology.](image-url)
network topology shown in Figure 1 has a single bottleneck link. The queue of node A is deployed with AQM including RED, and PI, with a capacity of 350 packets, and target queue length of 150 packets. For RED, the minimum threshold is set to be 80 packets, and the maximum is 250 packets. We use the default value in ns-2 for other parameters. In situation of variant traffic load, the number of TCP flows is 200, 50, 350, and 100 at time 0, 50, 100, and 150 respectively. From Figure 2, we can see that the queue length oscillates badly under dynamic network environment. Heavy oscillation will introduce jitter of queuing delay and high drop rate of packets. In face of unresponsive flows such as HTTP and UDP connections, the performance of static AQMs is also degraded, as shown in Section 4.

Control theory is believed to be the best systematic approach to understand the behavior of TCP/AQM. Naturally, the problem to reject disturbance should be resolved by using the control theory. In [15], a smith predictor for processes with time delay is designed based on optimal control theory in frequency field. Following the thoughts of [15], in Section 3, a new controller for AQM is designed to reject disturbance and compensate delay. To make the controller adaptive to the dynamic network environment, we add a self-tune structure to the controller. The new controller is called robust adaptive AQM or RA-AQM.

3. DESIGN OF THE RA-AQM CONTROLLER

3.1. Control Theoretic Modeling Of TCP/AQM

In [4], a TCP flow dynamic model is developed. By small-signal linearization about an operating point, a simplified version of the model to approximate the dynamics is given in [5]. Using the linearized TCP model, the TCP/AQM system is regarded as a feedback control system shown in Figure 3. TCP mechanism of end systems and queue of routers compose the plant, and the AQM control law is the controller.

The open-loop transfer function of the plant with time delay is given by

\[ P(s) = P_{TCP}(s)P_{queue}(s)e^{-sR} = \frac{K_p e^{-sR}}{(T_1 s + 1)(T_2 s + 1)} \]

where \( K_p = (RC)^3/4N^2 \), \( T_1 = R \), \( T_2 = R^2C/2N \), \( C \) is the capacity of link, \( N \) is the

Figure 3. Block diagram of AQM control system.
number of TCP connections, and \( R \) is the RTT. \( P(s) \) is a second-order model with time delay. In [16], a first order model \( \tilde{P}(s) \) is used to approximate \( P(s) \) as follow

\[
P(s) \approx \tilde{P}(s) = \frac{K_p e^{-\theta s}}{\tau s + 1}
\]

where \( \theta = T_1 + T_2 + R - \sqrt{T_1^2 + T_2^2} \), and \( \tau = \sqrt{T_1^2 + T_2^2} \).

### 3.2. Design Of A Disturbance-Rejected Controller

A method of designing a smith predictor based on optimal control theory is given in [15]. Optimal control approach is used to minimize the output caused by disturbance. Similarly we can design a controller to suppress the disturbance of unresponsive flows such as UDP and HTTP connections. Our design is based on the first order lag process of TCP/AQM system.

To simplify the control structure of (2), we approximate the time delay factor \( e^{-\theta s} \) with a first order Pade approximation below

\[
\hat{P}(s) = \frac{K_p (1 - \frac{\theta}{2} s)}{(\tau s + 1)(1 + \frac{\theta}{2} s)}
\]

Since \( \hat{P}(s) \) is stable, we get its Youla parameterization [18]

\[
C(s) = \frac{Q(s)}{1 + P(s)C(s)}
\]

From Figure 3, the transfer function from disturbance \( d \) to system output \( q \) can be written as

\[
G(s) = 1 - \hat{P}(s)Q(s)
\]

To deal with the disturbance, define the performance objective as follow

\[
\min \| W(s)G(s) \|_2
\]

where \( W(s) \) is the weighting function, and \( \| \cdot \|_2 \) is the two-norm. We can treat unresponsive flows as constant bit flows [8-9] and can be further considered as unit step input. Thus we define the disturbance input as \( \frac{1}{W(s)} = \frac{1}{s} \). The function \( W(s)G(s) \) has a pole on the imaginary axis. \( \| W(s)G(s) \|_2 \) must be finite, hence we have \( \lim_{s \to 0} G(s) = 0 \) to cancel the pole. From (5), we get \( Q(0) = \hat{P}(0) = 1/K_p \), then any stable \( Q(s) \) can be written as

\[
Q(s) = \frac{1}{K_p} + sQ_1(s)
\]

where \( Q_1(s) \) is a stable function. Substituting (5) and (7) into \( \| W(s)G(s) \|_2 \), we have (referring to [15] for details)

\[
\| W(s)G(s) \|_2^2 = \left\| \frac{\theta}{1 - s\theta/2} \right\|_2^2 + \left\| \frac{\tau}{\tau s + 1} - \frac{K_p}{\tau s + 1}Q_1(s) \right\|_2^2
\]

From (8), we get the unique optimal when \( Q_1(s) = \tau/K_p \). By (7), we have \( Q(s) = (\tau s + 1)/K_p \). To roll \( Q(s) \) at the high frequency, we introduce a low pass filter
\(1/(\gamma s + 1)\) and get a proper suboptimal solution \(\hat{Q}(s)\) of (8) as follow

\[
\hat{Q}(s) = \frac{\tau s + 1}{K_p(\gamma s + 1)}
\]

From (9) and (4), we have the controller as follow

\[
C(s) = \frac{K_p(\tau s + 1)(1 + \theta/2)}{\theta^2 s^2 + (\gamma + \theta)s}
\]

In [16], the effect of parameter \(\gamma\) is discussed. We take \(\gamma = 0.8\theta\) as the recommended value. We can convert (10) into a difference equation. Let \(f_s\) be the sample frequency. At every \(t = nT = n/f_s\) sampling instant, we have

\[
p(n) = k_1p(n-1) - k_2p(n-2) + k_3\delta q(n) - k_4\delta q(n-1) + k_5\delta q(n-2)
\]

where

\[
k_1 = a\left(\frac{K_p\theta^2}{T} + K_p(\theta + \gamma)\right), \quad k_2 = a\left(\frac{K_p\theta^2}{2T}\right), \quad k_3 = a\left(\frac{\tau\theta}{2T} + (\tau + \theta) + T\right),
\]

\[
k_4 = a\left(\frac{\tau\theta}{T} + (\tau + \theta)\right), \quad k_5 = a\left(\frac{1}{2T} + \frac{K_p\theta^2}{2T} + K_p(\theta + \gamma)\right)
\]

and \(\delta q(n) = q(n) - q_{ref}\). \(q_{ref}\) is the desired queue length to which we want regulate. \(q(n)\) is the sampling queue length, and the output \(p(n)\) is the drop probability.

Example 1: Consider the network setup as \(C = 3750\) packets/s, \(N = 60\), \(R = 0.40\) ms, and \(f_s = 160\). From (11), we have following controller

\[
p(n) = 1.9654p(n-1) - 0.9654p(n-2) + 3.3506 \times 10^5 \delta q(n) - 6.6445 \times 10^5 \delta q(n-1) + 3.2940 \times 10^5 \delta q(n-2)
\]

3.3. Self-Tuning Structure For Controller

The controller given by (11) is decided by network parameters including link-capacity \(C\), TCP workload \(N\), and round trip time \(R\). For responsive TCP flows, both of them are time varying under dynamic network environment. Thus there should be a mechanism to estimate the approximate value of \(C\), \(N\) and \(R\).

As in [13], we measure the link-capacity by keeping track of departed packets periodically. Let \(\hat{C}\) denote the ratio of departed TCP packets to the router’s busy time. Then we estimate \(C\) by an exponentially weighted moving average as follow

\[
C = (1-w_c)C + w_c \hat{C}
\]

where \(w_c\) is the weight.

For TCP workload, there are many methods to estimate the TCP connection number, for example by recording the field of flow ID in packet header or sampling the packets in queue. Here we assume the TCP load \(N\) is known. In fact, to estimate \(N\) is easier than to get the “effective” RTT for routers. From the TCP fluid model in [4], we have \(\sqrt{p_0/2} = N/RC\) in steady-state, where \(p_0\) is the equilibrium drop probability. Thus the effective \(R\) can be
estimate as an exponentially weighted moving average

\[ R = (1 - w_R) R + w_R \sqrt{\frac{N^2}{2} \frac{p_0}{2C^2}} \]  

(14)

where \( w_R \) is the weight.

Hence (11), (13) and (14) represent the new AQM, we’ll call it RA-AQM hereafter.

4. SIMULATION RESULTS

In this section, we evaluate RA-AQM controller by simulations with ns-2 simulator and compare it with PI, STPI and ARED [17]. The network topology is shown in Figure 1. We use the default setup for the PI and STPI [13]. For ARED, the minimum threshold is set to be 80packets, and the maximum is 250packets. We consider the following cases.

4.1. Variational TCP Traffic Load

It is shown in Section 2 that the AQMs with static parameters perform badly. Here we change the AQM to STPI and RA-AQM, and repeat the simulations again under the same scenario. Figure 4 shows the evolvement of queue length. Under the control of RA-AQM, the oscillations disappear. Though STPI performs better than PI and RED, the response is sluggish, which lead to the oscillation of queue length. From Figure 5, we can see that the drop probability of RA-AQM reaches the equilibrium more quickly than PI and STPI. Thus the response of RA-AQM is faster than others.

Figure 4. Compare queue length vs. time for variational network load: (a) STPI; (b) RA-AQM.

Figure 5. Compare drop probability vs. time of PI, STPI and RA-AQM for variational network load.

Figure 6. Compare queue length vs. time for mixed network load: (a) PI; (b) ARED; (c) STPI; (d) RA-AQM.
4.2. Long-lived TCP Flows Mixed With HTTP And UDP Connections

To simulate the real network environment, we mix long-lived TCP flows with unresponsive flows such as HTTP and UDP connections. We run 250 long-lived TCP connections for all the simulation time. The bursty HTTP traffic involves 800 sessions, and the number of pages per session is 150. We start 30 UDP flows at 5s, and stop at 30. Afterwards, 50 UDP flows are started again at 60s, and decrease to 10 at 80s. Figure 7 shows the simulation results. We can see that RA-AQM control the queue length well at the desired value, while other AQM oscillate heavily.

4.3. Heavy Traffic Load

When the traffic load of network is heavy, AQM should stabilize the queue length at a target value to avoid congestion. In this simulation, we start 600 TCP/Reno connections randomly at the beginning. We start 800 HTTP sessions, and the number of pages per session is 200. The simulation lasts 50s. Figure 8 shows the results for PI, STPI, ARED and RA-AQM. The queue under PI keeps full for all the simulation time. It only takes RA-AQM about 6s to control the queue length to the target value. And the stability of RA-AQM is better than that of others. At the same time, RA-AQM decrease the drop rate by about 10% comparing with STPI.

5. CONCLUSIONS

In this paper, we have designed a robust adaptive Active Queue Management scheme called RA-AQM, which can deal with the disturbance of unresponsive flows well. RA-AQM can tune its parameters online and is adaptive to the variation in network state. The simulation
results show that RA-AQM can quickly regulate queue length to the desired value, respond much faster than other AQMs and keep the oscillation small. It is robust to UDP unresponsive flows and bursty HTTP connections.

REFERENCES